I.J. Intelligent Systems and Applications, **2016, 10, 11-20**
Published Online October 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijisa.2016.10.02

A Risk-Aware Application Scheduling Model in Cloud Computing Scenarios

Ala Arman
DTI - Universit à degli Studi di Milano, Crema, 26013, Italia
E-mail: ala.arman@unimi.it

Abstract—Cloud users usually have different preferences over their applications that outsource to the cloud, based on the financial profit of each application's execution. Moreover, various types of virtual machines are offered by a cloud service provider with distinct characteristics, such as rental prices, availability levels, each with a different probability of occurrence and a penalty, which is paid to the user in case the virtual machine is not available. Therefore, the problem of application scheduling in cloud computing environments, considering the risk of financial loss of application-to-VM assignment becomes a challenging issue. In this paper, we propose a risk-aware scheduling model, using risk analysis to allocate the applications to the virtual machines, so that, the expected total pay-off of an application is maximized, by taking into account of the priority of applications. A running example is used through the paper to better illustrate the model and its application to improve the efficiency of resource assignment in cloud computing scenarios.

Index Terms—Application scheduling, cloud computing, expected monetary value, risk analysis.

I. Introduction

Cloud computing [1] is a potent Internet-based service, which offers a new technique to provide huge amount of shared resources offered by several cloud service providers (CSPs). There are two main advantages for adopting a cloud solution [2], which have been discussed in the literature. First, the end user does not need to be involved in the configuration and maintenance of cloud services provided by the CSP. Second, thanks to "pay-as-you-go" pricing model, the end-user only pays for the resources that requests and uses. Therefore, she can evade unnecessary costs [3], specially the high initial cost of setting up the application deployment environment. Additionally, there are other advantages of cloud computing discussed in the literature [4]–[6], like high elasticity and availability, reliability, multi-tenancy, on demand self-service, broad network access, etc.

Therefore, an increasing number of users tend to outsource their existing applications to the cloud environments in order to benefit from their significant features. However, application scheduling, that is, assign the available resources on a cloud environment to an application, has become a prominent problem in cloud scenarios because it directly affects the application's performance. In worse cases, poor application scheduling will result in the violation of quality of service (QoS) or even application failure.

Also, the CSPs provide several types of VMs to the users, in order to run their applications on the cloud, considering their heterogeneous requirements. For example, Amazon Elastic Compute Cloud (Amazon EC2) [7], is a web service that provides different types of VMs to the users. Each type of VM, which is offered by Amazon EC2, has different characteristics in terms of hourly rental price, available resource dimensions, such as CPU, RAM, disk space, network capacity, etc., to offer more flexibility for choosing appropriate type of VM(s) to be assigned to the user's applications, considering various requirements of each application. Furthermore, nowadays, the modern CSPs provide different levels of availability for the VMs, each with a service credit, in order to improve the management of service level agreement (SLA). Service credit is a percentage of (monthly in case of Amzaon EC2) rental charge of a VM, which is paid to the user as a penalty, if the VM is not available. For example, currently, Amazon EC2 considers monthly uptime percentage *MUP* of a VM to define its availability levels. Table presents different *MUP* levels, as well as their related service credits [8] for the VMs, provided by Amazon EC2. For instance, if *MUP*<99.0, thirty percent of total monthly charge for renting the VM is paid to the user as a penalty and so on.

The cloud users usually have sensitive applications, so that, assigning them to the VMs inappropriately without considering the expected pay-off of each application's execution, could lead to huge financial loss of the user and decreasing her satisfaction by the offered cloud service significantly. As a result, the problem of assigning a set of applications with different financial execution profit, to a set of VMs, each with a rental price and several availability levels where each availability level is associated with a distinct probability of occurrence and a service credit, so that, the expected pay-off for each application is maximized, considering the priority of applications, becomes a challenging issue.

A naive approach to choose the most suitable VM for each application would consist in assigning an application with higher priority from the user's view to a cheaper VM.

However, such an approach would risk to ignore the several availability levels of VMs, each with a probability of occurrence and different service credit. To address this

issue, in this paper, we propose a risk-aware application scheduling model aimed at decreasing the financial risk of application-to-VM assignment, considering the priority of applications defined by the user. To this purpose, our solution first produces, for each VM, a penalty which is paid to the user in case of SLA violation; Then it calculates the expected benefit of each application if it is assigned to each VM in V , considering the priority of applications; finally, it calculates the expected pay-off of each application, adopting a *risk analysis technique* (expected monetary value [9]), if it is assigned to each VM. Finally, we assign each application to a VM that the application has the highest expected pay-off with.

Table 1. An example for deferent levels for MUP and their service credits

MUP level	Service Credit
Less than 99.95% but equal to or greater than 99.0%	10%
Less than 99.0%	30%

The rest of this paper is as follows. Section II provides some related works that have been reported in this field. Section III briefly discusses about the research background of this paper, including cloud computing, virtualization technology, resource allocation in cloud environments, and risk analysis. Section IV presents our problem. Section V illustrates our solution. Finally, Section IV concludes the paper and discusses about future work.

II. Related Works

Previous works related to our proposal devoted to the assignment of a set of applications or tasks to a set of available VMs in a cloud domain, e.g., [10]–[16], that discuss how to assign applications or tasks, considering their different requirements as well as the characteristics of VMs. Most of these approaches fall short of considering the priority of applications, defined by the user. For example, the authors in [14] provide a scheduling strategy, considering multiple SLA parameters, for deploying applications in cloud environments, in order to optimize the performance of applications as well as reducing the possibilities of SLA violations. However, the authors did not consider the priority of applications, defined by the user in their scheduling approach. In addition, [16] offers a SLA-based resource provisioning solution in order to maximize the resource utilization and the profit of provider. The authors define multiple penalty types including a fixed penalty, a proportional penalty, and a delay-dependent penalty in SLA. A penalty will incur if the number of SLA violations exceeds a predefined threshold. We see that their solution shares with us the view of considering penalty in SLA for assigning the applications to the VMs. However, contrary to our approach presented in this paper, an application with lower penalty rates has more priority from the CSP's view to be assigned to a VM. Therefore, again here, the priority of applications is not based on the user's preferences. Moreover, few researches, e.g., [17], [18], have addressed the issues of application or task scheduling in cloud computing scenarios, while a user has different preferences over her applications or tasks. For instance, the authors in [17] propose an algorithm for dynamic allocation of VMs to a set of prioritized tasks based on multiple SLA parameters, such as memory, network bandwidth, and requested CPU time by applying a preemption mechanism for executing tasks with higher priority. While our proposal and [17] share the idea of applying the priority of applications or tasks to schedule them, [17] does not take into account rental price and different availability levels of VMs, each with a probability of occurrence, and a service credit to schedule the tasks, which is one of the main contributions of this paper. Our approach strives to optimize the application scheduling in cloud computing scenarios by maximizing the expected total pay-off of each application-to-VM assignment, considering the priority of applications, which is according to the financial execution profit of each application.

III. Research Background

This section briefly discusses some fundamental concepts of cloud computing and virtualization. Also, it describes the issues of resource allocation in cloud environments and the concept of expected monetary value (EMV) in quantitative risk analysis.

A. Cloud Computing

Cloud computing has become one of the most hottest and controversial topics in academic and industrial environments. It relies [19] on the practice of moving computing to the Internet. Intuitively, cloud users [20] outsource their data and applications to the cloud and access them remotely in a simple and pervasive way. In cloud computing environments, everything is offered as a service, that is, XaaS, e.g., SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Infrastructure as a Service), which are three main service delivery models defined in cloud scenarios. They form a layered system structure for cloud computing, as presented in Fig.1. The most bottom layer (IaaS), which mostly is used by IT and network architects, is composed of physical and virtualized computing, storage, network resources, etc. The examples of IaaS providers are Amazon EC2, VMware [21], etc. The middle layer (PaaS), which is mostly used by application developers and testers, refers to offering platform layer resources, such as operating system support, software development frameworks, etc. The examples of PaaS providers include Google App Engine [22], Amazon SimpleDB [23], etc. The top layer (SaaS), which is used mostly by end-users, provides several types of on-demand cloud-based applications over the Internet. The examples of SaaS providers are Salesforce.com [24], Rackspace [25], etc.

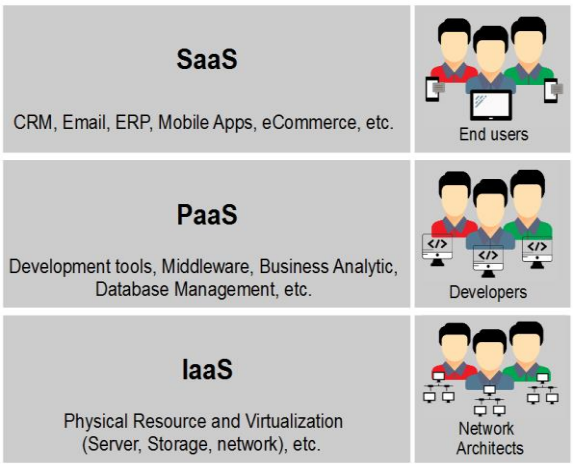

Fig.1. Cloud service delivery models

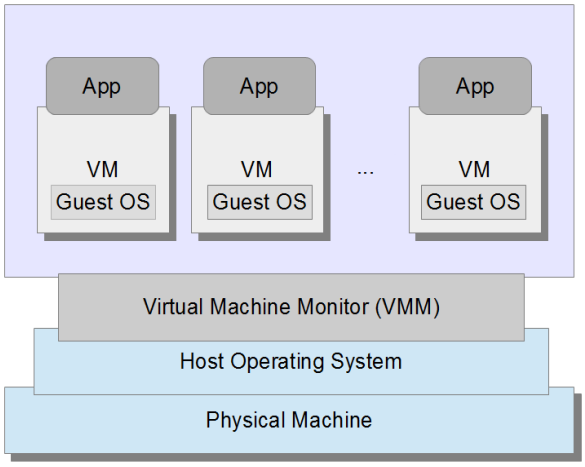

Fig.2. A general schema of virtualization

In cloud computing scenarios, since the demands of cloud user vary significantly during time, it is not possible to meet all of her requirements by the service(s) [26], which are provided by the CSPs. Also, it is essential that cloud users have guarantees from the CSPs [27], on service delivery. Therefore, in order to consider these challenges and requirements, a contract which is called service level agreement (SLA) is signed between a cloud user and a CSP, through a negotiation process. Two main aspects must be considered in a SLA: *1) quality of service (QoS)* [28] requirements, that is, the measurable ability of a CSP to offer network and computation services, such that, the user's expectations from the offered service(s) are fulfilled, such as bandwidth availability, response time, CPU utilization, etc., *2) penalties*, if QoS requirements are not met [29] by a provider. For example, as we discussed already in Section 0, Amazon EC2, provides QoS guaranties in multiple availability levels, with respect to the monthly uptime percentage (*MUP*) of a VM. Also, it describes penalties in terms of service credits, which is a percentage of monthly rental charge of a VM.

B. Virtualization

Virtualization technology [30][31] is one of the most important key features of cloud computing, which refers to presenting the illusion of running many smaller VMs on a physical machine, each hosting a separate operating system instance. Simply put, each physical machine can run multiple VMs and each VM can be used by a different user, which is considered as one of the most crucial benefits of virtualization. Moreover, it is possible to utilize and assign different partitions of resources on the same physical machine to a VM. Also, since the resource requirements of a user change quickly due to the mobility of users, reallocation of resources is easier using virtualization [32] because virtual devices are software-based and offer a uniform interface through standard abstractions. In addition, leveraging virtualization technology, it is possible to encapsulate workloads and transfer them to idle or unused systems, which causes avoiding or delaying purchasing additional servers by consolidating existing systems. Fig.2, depicts a high level presentation of virtualization concept.

As can be seen, each VM uses an operating system, that is, the guest OS, which could be different from another VM on the same physical machine. Also, the virtual machine monitor (VMM) is a software layer, which mediates the interactions between a VM and the monitored host that the VM is running on. In other words, VMM masks [33] the complexities of physical machine from the guest execution environment.

C. Resource Allocation In Cloud Enviroments

Thanks to significant features of virtualization technology, which was partially discussed in Section 0, the cloud users are able to run several applications, each with different characteristics and requirements on a set of VMs. However, application or task scheduling in a virtualized dynamic environment like cloud is not an easy task. In fact, due to the presence of heterogeneous types of resources in a cloud infrastructure as well as different tasks or applications, with various characteristics and requirements, the allocation of resources to tasks or applications in cloud computing environments is considered as a NP-hard [34][35] (non-deterministic polynomial-time hard) problem. Therefore, it is so difficult to find an optimal solution in a polynomial time. The issue of resource allocation in cloud environments have been discussed in the literature, considering its several aspects such as security [36], resource consumption efficiency [15][37], etc. but the majority of previous studies have the limitation of not taking into account of the financial profit of a cloud user by running her tasks or applications on a cloud environment. A cloud user expects to maximize her financial profit, while migrating her applications to a cloud environment. Also, she usually has different preferences over her applications, according to the financial execution profit of each application, that is, an application with higher financial execution profit has higher priority from her view. The scenario gets more complex when in a virtualized data center, different VMs with various rental prices are provided by a CSP, with different availability levels, each with a probability of occurrence and a service credit. As a result, in such context, with high level of uncertainty, the application-to-VM assignment becomes a challenging issue.

In this work, we resolve this issue by using risk analysis techniques. In the next section, in order to be clearer, we will present some basic concepts in quantitative risk analysis, which is one of the main techniques to deal with risk and uncertainty. Also, we will discuss briefly about the expected monetary value (EMV) concept, which we will use later in this paper.

D. Risk Analysis

Risk [38] is an indicator of what could happen to assets of an organization if they are not properly protected. Risk analysis refers to a systematic review for estimating the magnitude of risks, which an organization is exposed. The main reason to perform risk analysis is [39] to support decision making in order to find the right balance between different concerns, such as cost, safety, etc. One of the most important applications of risk analysis is decision making under risk [15], which occurs when a decision maker is uncertain about the occurrence of a state of nature (event), but the probability of each state of nature is known. One of the most recommended quantitative tools and techniques for decision making under risk is called expected monetary value (EMV), which calculates the average outcome when the future includes scenarios that may or may not happen [40]. EMV for a course of action j, is the pay-off X_{kj} for each combination of event k multiplied by, the probability $\Pr_k$ of occurrence of event k, summed over all events. It is formally defined as follows:

$$EMV_j = \sum_{k=1}^{d} \Pr_k X_{kj}. \quad (1)$$

The advantage of using EMV is considering uncertainty, by taking into account of a probability for each event k. Also, the problem becomes simpler by reducing the information about a course of action j to a scalar value EMV_j.

IV. Problem Definition

We consider a reference scenario like the one presented Fig.3., which is characterized by a user wishing to migrate to the cloud a set of *n* applications $A = \{a_1,...,a_n\}$. The user defines for each application a_i a priority according to the hourly execution profit π_i of application a_i, so that, each application a_i with higher hourly execution profit π_i has higher priority from the user's view. Table 2. shows the three applications that we consider in our running example and their related parameters. As can be seen, applications *streamApp*, *enterpriseApp*, have the priority of first (high), second (medium), and third (low) priority from the user's view, respectively.

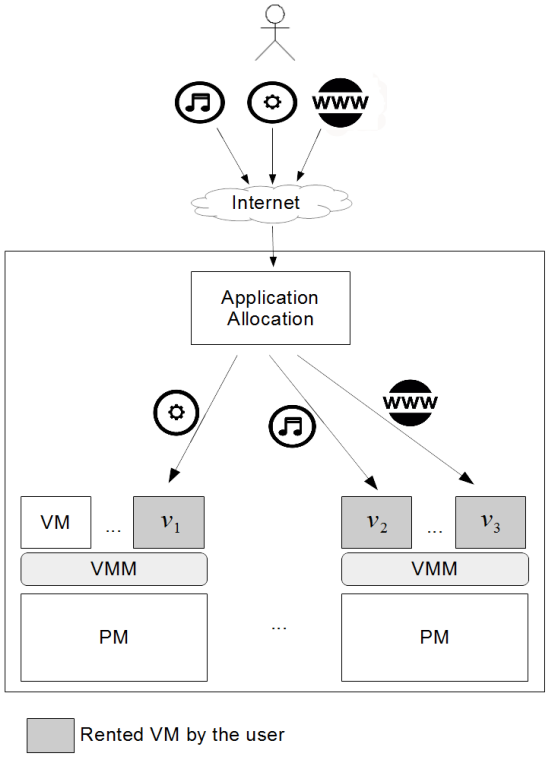

Fig.3. Reference Scenario

The user needs to assign each application in A to a VM in a set $V = \{v_1,...,v_n\}$ of rented VMs by her, which are running on multiple physical machines (PM). Each $v_j \in V$ is then associated with a *characteristic double* $VC_j = (c_j, L_j)$, where c_j is the rental hourly charge of v_j in US dollar terms. $L_j[1,...,d]$ is the *HUP level vector* for v_j, where $L_j[k]$ represents the $k-th$ *HUP* level of VM v_j. We assume that, in each hour, *HUP* of each VM $v_j \in V$ falls in one of *HUP* levels defined in L_j. Also, $Lev^{(j)} = \{Lev_1^{(j)},...,Lev_d^{(j)}\}$ denotes a set of *characteristic quadruples*, where quadruple $Lev_k^{(j)} = (\min_k, \max_k, \Pr_k^{(j)}, \Gamma_k)$ is associated with the *HUP* level $L_j[k]$. $\min_k$ and $\max_k$ are the minimum and maximum values (in percent) for *HUP* level $L_j[k]$, respectively.

Table 2. The user's applications

Application Name	Service Type	Hourly Profit ($)	Priority
streamApp	Music Streaming	4.9	1
enterpriseApp	Complex Business Logic	4.5	2
webApp	Low Traffic Website	2.1	3

Table 3. Rented VMs by the user and their related parameters

VM Name	Rental Hourly Charge (c)	*HUP* Level	min *HUP* (%)	max *HUP* (%)	Probability of Occurrence (Pr)	Service Credit (Γ)
v_1	0.015	1	99.96	100	0.997	0
		2	99	99.95	0.002	30
		3	95	98.99	0.0009	50
		4	90	94.99	0.0001	60
v_2	0.004	1	99.96	100	0.96	0
		2	99	99.95	0.0395	30
		3	95	98.99	0.0004	50
		4	90	94.99	0.0001	60
v_3	0.001	1	99.96	100	0.86	0
		2	99	99.95	0.03	30
		3	95	98.99	0.095	50
		4	90	94.99	0.015	60

$\Pr_k^{(j)}$ and Γ_k are the probability that *HUP* of VM v_j falls in the *HUP* level $L_j[k]$ and the service credit of *HUP* level $L_j[k]$, respectively. Table 3. shows the set of VMs in our running example. As can be seen there are three VMs, each with a rental hourly charge (c_j). Also, each VM has four *HUP* levels $L_j[k]$, each with a probability of occurrence $\Pr_k^{(j)}$ and a service credit Γ_k. Referring to Table 3., for example, $VC_1 = (0.015, L_1[1,...,4])$ and $Lev_3^{(1)} = (95, 98.99, 0.0009, 50)$.

Given a set of applications A and a set of VMs V, a *characteristic double* $VC_j = (c_j, L_j)$ and a set of *characteristic quadruples* $Lev^{(j)} = \{Lev_1^{(j)}, ..., Lev_d^{(j)}\}$, for each $v_j \in V$, a simple solution would select a cheaper VM for an application with higher priority. However, such a trivial approach may ignore the different availability levels of VMs, each with a different service credit and a probability of occurrence. Therefore, this approach might then be considered not desirable. To prevent such a situation, we propose to adopt a risk-aware approach aimed at choosing a VM $v_j \in V$ for an application $a_i \in A$, which maximizes the expected hourly pay-off π_i of application a_i, considering the priority of applications. We consider our solution, as we will see in the next section, as a one-to-one allocation function $alloc: A \to V$, that takes the set A of application instances as input and assigns each application $a_i \in A$ to a VM $v_j \in V$ as output, so that, application a_i has the highest expected hourly pay-off, if it is assigned to v_j, i.e., $alloc(a_i) = v_j$. Therefore, v_j is an optimal VM for application a_i. Table 4. shows the notations that we will use in this paper together with their related descriptions.

Table 4. Notations

Notation	Description
π_i	average hourly financial execution profit of a_i
c_j	rental hourly charge of v_j
$\min_k$	minimum value of *HUP* of a VM in level k
$\max_k$	maximum value of *HUP* of a VM in level k
$\Pr_k^{(j)}$	probability that *HUP* of v_j falls in level l_k
Γ_k	service credit of *HUP* level l_k for each VM
$\xi_j[k]$	hourly penalty of *HUP* level l_k for v_j
$b_j[k]$	expected hourly benefit of current application a_i in *HUP* level l_k, if $alloc(a_i) = v_j$
$\theta_i[j]$	expected hourly pay-off for a_i, if $alloc(a_i) = v_j$

V. Proposed Approach

Our approach to choose a VM in V for each application in A, so that, the expected hourly pay-off of each application is maximized, considering the priority of applications, operates in five steps (See Fig.5.): *1) Sorting* the set A; *2) hourly penalty* calculation of for each VM in V; *3) expected hourly benefit* of each application if it is assigned to each VM in V; *4) expected hourly pay-off* calculation of each application if it is assigned to each VM in V; *5) update* the set *V*. In the following, we present out approach in detail.

INPUT
$A = \{a_1,...,a_n\}$ /* set of applications*/ $V = \{v_1,...,v_n\}$ /* set of VMs*/
$VC_1,...,VC_n$ /*sets of characteristic doubles*/
$Lev^{(1)},...,Lev^{(n)}$ /* of characteristic quadruples */
OUTPUT
$alloc : A \rightarrow V$
MAIN
1: Sort A based on the priority p_i of each application in increasing order.
2: let ξ_j be the *hourly penalty* vector of size $|d|$
3: **for** $j = 1,...,|n|$ **do**
4: **for** $k = 1,...,|d|$ **do**
/* hourly penalty */
5: $\xi_j[k] := c_j \Gamma_k$
6: **end for**
7: **end for**
8: **for** $i = 1,...,|n|$ **do**
9: **for each** $v_j \in V$ **do**
10: let b_j be the *expected hourly benefit vector* of size $|d|$
11: **for** $k = 1,...,|d|$ **do**
/*expected hourly benefit*/
12: $b_j[k] := (\frac{\min_k + \max_k}{2}\%)\pi_i + \xi_j[k] - c_j$
13:let θ_i be the *expected hourly pay-off* vector of size $|n|$
/*expected hourly pay-off */
14: $\theta_i[j] = \sum_{k=1}^{d} \Pr_k^{(j)} b_j[k]$
15: **end for**
16: $alloc(a_i) = v_{j'} \in V$ s.t.
$\exists v_j \in V, v_j \neq v_{j'} : \theta_i[j] > \theta_i[j']$
17: **end for**
/*Update the Set V */
18: $V := V \setminus \{v_{j'}\}$
19: **end for**

Fig.5. Algorithm for risk-aware assignment of the applications to the VMs

Sorting. As we discussed earlier, in Section 0, in our proposed solution, the application-to-VM assignment is based on the priority of applications, defined by the user.

Therefore, first, we sort the set A based on the priority of application a_i in increasing order (line 1). Therefore, more important applications will be assigned to VMs that are more suitable for them in order to maximize their expected hourly pay-off. Therefore, referring to our running example, if we sort the set A based on the priority of applications in increasing order, $A = \{streamApp, enterpriseApp, webApp\}$.

Hourly penalty. Next, we calculate, for each VM $v_j \in V$, the hourly penalty, in each *HUP* level $L_j[k]$. To do this, we build a vector of hourly penalty $\xi_j[1,...,d]$, where $\xi_j[k] = c_j \Gamma_k$, is the hourly penalty of VM v_j in *HUP* level $L_j[k]$ (lines 2-7). Back to our running example, Table 5. shows the calculation of hourly penalty ξ of each VM $v_j \in V$, in each *HUP* level l_k. For instance, the hourly penalty of VM v_1 in *HUP* level $L_1[2]$ is $\xi_1[2] = c_1\Gamma_2 = 0.30 * 0.015 = \$0.0045$.

In the remainder of this section, we illustrate in detail, our proposed solution. For simplicity, in the following, we refer our discussion to one application only (*streamApp*), with the note that the process described is executed for all applications in A.

Expected hourly benefit. For each $v_j \in V$, we compute the expected hourly benefit of application a_i in *HUP* level l_k (lines 8-12). To do this, we build a vector of expected hourly benefit $b_j[1,...,d]$, where $b_j[k]$ is the expected hourly benefit of application a_i, in *HUP* level $L_j[k]$ of v_j, if $alloc(a_i) = v_j$. It is given by:

$$b_j[k] := (\frac{\min_k + \max_k}{2}\%)\pi_i + \xi_j[k] - c_j \qquad (2)$$

Table 6. depicts the calculation of expected hourly benefit $b_j[k]$ of application *streamApp* for each $v_j \in V$, in each *HUP* level $L_j[k]$. For example, the expected hourly benefit $b_1[2]$ of application *streamApp* in the second *HUP* level $L_1[2]$ of v_1 is equal to $b_1[2] = (\frac{99 + 99.95}{2}\%)4.9 + 0.0045 - 0.0015 = \$4.863$. Note that, while other approaches such as mean, median, etc. could be used, we consider the midrange of each *HUP* level $L_j[k]$, using $\min_k$ and $\max_k$, to calculate $b_j[k]$.

Expected hourly pay-off. In order to consider the uncertainty in assigning the current application (*streamApp* in our running example) to a VM $v_j \in V$, so that, the expected hourly pay-off of current application, that is, *streamApp* in our running example, is maximized, we then propose to adopt a risk analysis technique, that permits to take into account of the probability of occurrence $\Pr_k^{(j)}$ of each *HUP* level $L_j[k]$ of each VM $v_j \in V$.

Table 5. Calculation of hourly penalty for each VM

VM Name	Rental Hourly Charge (c)	*HUP* Level	Service Credit (Γ)	Hourly Penalty (ξ)
v_1	0.015	1	0	0
		2	30	0.0045
		3	50	0.0075
		4	60	0.009
v_2	0.004	1	0	0
		2	30	0.0012
		3	50	0.002
		4	60	0.0024
v_3	0.001	1	0	0
		2	30	0.0003
		3	50	0.0005
		4	60	0.0006

Table 6. Expected hourly benefit for *streamApp* considering each VM

VM Name	Rental Hourly Charge (c)	*HUP* Level	min *HUP* (%)	max *HUP* (%)	Hourly penalty (ξ)	Expected Hourly Benefit (b)
v_1	0.015	1	99.96	100	0	4.884
		2	99	99.95	0.0045	4.863
		3	95	98.99	0.0075	4.745
		4	90	94.99	0.009	4.526
v_2	0.004	1	99.96	100	0	4.895
		2	99	99.95	0.0012	4.871
		3	95	98.99	0.002	4.750
		4	90	94.99	0.0024	4.530
v_3	0.001	1	99.96	100	0	4.898
		2	99	99.95	0.0003	4.873
		3	95	98.99	0.0005	4.752
		4	90	94.99	0.0006	4.531

While noting that there are different approaches can be applied (e.g., expected opportunity loss [9]), we consider the expected monetary value (EMV) [9]. Taking into account of the discussion in Section III.D, in our cloud scenario, events correspond to *HUP* levels of VMs. Therefore, we build a vector of expected hourly pay-off $\theta_i[1,....,n]$ for application a_i, where $\theta_i[j] = \sum_{k=1}^{d} \Pr_k^{(j)} b_j[k]$, is the expected hourly pay-off of application a_i, if $alloc(a_i) = v_j$ (lines 13-14). Table 7. shows the calculation of expected total hourly pay-off θ_1 for the application *streamApp*, considering each VM in V.

For instance, the expected hourly pay-off of application *streamApp*, if it is assigned to v_2, is equal to

$$\theta_1[2] = \sum_{1 \le k \le 4} \Pr_k^{(2)} b_2[k] = 0.96*4.895 + 0.0395*4.871 + 0.0004*4.750 + 0.0001*4.530 = \$4.893.$$

Therefore, $\theta_1 = [4.883, 4.893, 4.877]$. We assign the current application a_i to a VM, which it has the highest expected hourly pay-off with (line16).

Referring to our running example, the current application, *streamApp*, which is the most important application from the user's view, is assigned to v_2, i.e., $alloc(streamApp) = v_2$ because application *streamApp* has the highest expected hourly pay-off, if it is assigned to v_2, that is, $\theta_1[2] = \$4.893$. It is interesting to see that the chosen VM is not the cheapest VM in V. However, considering the different probability of occurrence and hourly penalty for each *HUP* level, the application *streamApp* is assigned to v_2, using EMV technique.

Updating. Finally, since we assume that only one application can be executed on a VM ($alloc: A \rightarrow V$ function is a one-to-one allocation function), after assigning application a_i to v_j, we update the set of VMs V by removing v_j from V (line 18). Therefore, considering our running example, $V = \{v_1, v_3\}$.

Table 7. Expected hourly pay-off for *streamApp* considering each VM

VM Name	*HUP* Level	Probability of Occurrence (Pr)	Expected Hourly Benefit (b)	Expected hourly pay-off (θ)
v_1	1	0.997	4.884	4.883
	2	0.002	4.863	
	3	0.0009	4.745	
	4	0.0001	4.526	
v_2	1	0.96	4.895	4.893
	2	0.0395	4.871	
	3	0.0004	4.750	
	4	0.0001	4.530	
v_3	1	0.86	4.898	4.877
	2	0.03	4.873	
	3	0.095	4.752	
	4	0.015	4.531	

Fig.4, shows a decision analysis tree for the selection of a VM for application *streamApp* with maximum expected hourly pay-off. As can be seen, the decision tree is composed of three nodes that represent the VMs in V. For each VM, there are four leaf nodes, which each of them illustrates the expected hourly benefit $b_j[k]$ of *streamApp* for *HUP* level $L_j[k]$. If we continue to assign applications in A to VMs in V, according to our proposed approach, $alloc(enterpriseApp) = v_1$ and $alloc(webApp) = v_3$.

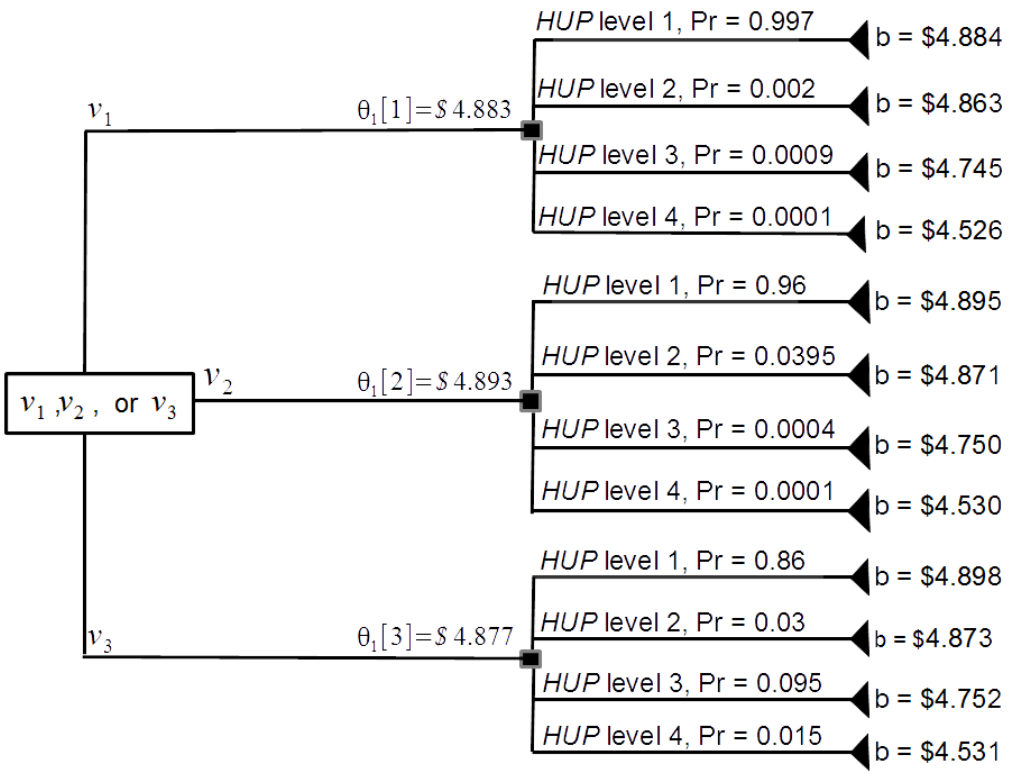

Fig.4. Decision tree to find an optimal VM for *StreamApp*

VI. Conclusions and Future Work

The paper presented a model for financial risk-aware application scheduling in cloud computing scenarios using risk analysis. We were interested in assigning the applications to the VMs, so that, the expected total pay-off for each application is maximized, considering the priority of applications, defined by the user. The proposed solution satisfies this requirement by considering several availability levels for a VM, each with a probability of occurrence and a service credit as a penalty rate.

In the proposed approach, we did not consider the risk attitude of a user [41] in assigning the applications to the

VMs, including *1) risk aversion*, e.g., if the applications of user are real time applications that cannot tolerate unavailability; *2) risk seeking*, e.g., if the applications of user are time invariant, like offline downloading, which can be performed even with risk loss during the transaction; *3) risk neutral*, e.g., if the applications of user can tolerate the risk of delay for a certain time period. As a future work, we will focus on the estimation of risk in application-to-VM assignment in cloud computing scenarios, using risk analysis by segregating users into different categories based on their risk behavior.

REFERENCES

[1] M. Y. Saeed and M. N. A. Khan, "Data Protection Techniques for Building Trust in Cloud Computing," *Int. J. Mod. Educ. Comput. Sci.*, vol. 7, no. 8, p. 38, 2015, "doi:10.5815/ijmecs.2015.08.05".

[2] A. Arman, A. Al-Shishtawy, and V. Vlassov, "Elasticity Controller for Cloud-Based Key-Value Stores.," *Parallel Distrib. Syst. Int. Conf.*, pp. 268–275, 2012, "doi:10.1109/ICPADS.2012.45".

[3] A. Zia and M. N. A. Khan, "A Scheme to Reduce Response Time in Cloud Computing Environment," *Int. J. Mod. Educ. Comput. Sci.*, vol. 5, no. 6, p. 56, 2013, "doi:10.5815/ijmecs.2013.06".

[4] S. Rajan and A. Jairath, "Cloud computing: The Fifth Generation of Computing," in *Communication Systems and Network Technologies (CSNT), 2011 International Conference on*, 2011, pp. 665–667, "doi: 10.1109/CSNT.2011.143".

[5] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," 2011, "doi: 10.6028/NIST.SP.800-145".

[6] S. Lee and K.-K. Seo, "A Hybrid Multi-criteria Decision-making Model for a Cloud Service Selection Problem Using BSC, Fuzzy Delphi Method and Fuzzy AHP," *Wirel. Pers. Commun.*, vol. 86, no. 1, pp. 57–75, 2016, "doi: 10.1007/s11277-015-2976-z".

[7] "Amazon Elastic Compute Cloud (Amazon EC2)." [Online]. Available: http://aws.amazon.com/ec2/. [Accessed: 11-Feb-2016].

[8] "Amazon EC2 Service Level Agreement." [Online]. Available:https://aws.amazon.com/ec2/sla/. [Accessed: 11-Feb-2016].

[9] D. M. Levine, M. L. Berenson, D. Stephan, and others, *Statistics for managers using Microsoft Excel*, vol. 660. Prentice Hall Upper Saddle River, NJ, 1999.

[10] X. Tang, K. Li, M. Qiu, and E. H.-M. Sha, "A Hierarchical Reliability-Driven Scheduling Algorithm in Grid Systems," *J. Parallel Distrib. Comput.*, vol. 72, no. 4, pp. 525–535, 2012, "doi: 10.1016/j.jpdc.2011.12.004".

[11] H. N. Van, F. D. Tran, and J.-M. Menaud, "Performance and Power Management for Cloud Infrastructures," in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, 2010, pp. 329–336, "doi: 10.1109/CLOUD.2010.25".

[12] S. Zhang, B. Wang, B. Zhao, and J. Tao, "An Energy-Aware Task Scheduling Algorithm for a Heterogeneous Data Center," in *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, 2013, pp. 1471–1477, "doi: 10.1109/TrustCom.2013.178".

[13] M. Sun, T. Zang, X. Xu, and R. Wang, "Consumer-Centered Cloud Services Selection Using AHP," in *2013 International Conference on Service Sciences (ICSS)*, 2013, pp. 1–6, "doi: 10.1109/ICSS.2013.26 ".

[14] V. C. Emeakaroha, I. Brandic, M. Maurer, and I. Breskovic, "SLA-Aware Application Deployment and Resource Allocation in Clouds," in *Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual*, 2011, pp. 298–303, "doi: 10.1109/COMPSACW.2011.97".

[15] K. Black, *Business statistics: for contemporary decision making*. John Wiley & Sons, 2011.

[16] S. K. Garg, S. K. Gopalaiyengar, and R. Buyya, "SLA-Based Resource Provisioning for Heterogeneous Workloads in a Virtualized Cloud Datacenter," in *Algorithms and Architectures for Parallel Processing*, Springer, 2011, pp. 371–384, "doi: 10.1007/978-3-642-24650-0_32".

[17] C. S. Pawar and R. B. Wagh, "Priority based dynamic resource allocation in Cloud computing with modified waiting queue," in *Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on*, 2013, pp. 311–316, "doi: 10.1109/ISSP.2013.6526925 ".

[18] D. C. Devi and V. R. Uthariaraj, "Load Balancing in Cloud Computing Environment Using Improved Weighted Round Robin Algorithm for Nonpreemptive Dependent Tasks," *Sci. World J.*, vol. 2016, 2016, "doi: 10.1155/2016/3896065".

[19] R. Buyya, S. Pandey, and C. Vecchiola, "Cloudbus Toolkit for Market-oriented Cloud Computing," in *Cloud Computing*, Springer, 2009, pp. 24–44, "doi: 10.1007/978-3-642-10665-1_4".

[20] L. Wang, G. Von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, "Cloud computing: A Perspective Study," *New Gener. Comput.*, vol. 28, no. 2, pp. 137–146, 2010, "doi: 10.1007/s00354-008-0081-5".

[21] "VMware Infrastructure as a Service." [Online]. Available:https://www.vmware.com/support/services/iaas-production. [Accessed: 25-May-2016].

[22] "Google App Engine Documentation." [Online]. Available:https://www.vmware.com/support/services/iaas-production. [Accessed: 25-May-2016].

[23] "Amazon SimpleDB." [Online]. Available: https://aws.amazon.com/simpledb/. [Accessed: 25-May-2016].

[24] "Salesforce.com." [Online]. Available: http://salesforce.com/. [Accessed: 25-May-2016].

[25] "Rackspace." [Online]. Available: https://www.rackspace.com/cloud. [Accessed: 25-May-2016].

[26] P. Patel, A. H. Ranabahu, and A. P. Sheth, "Service level agreement in cloud computing," 2009.

[27] Y. Wang, S. Chen, and M. Pedram, "Service Level Agreement-Based Joint Application Environment Assignment and Resource Allocation in Cloud Computing Systems," in *Green Technologies Conference, 2013 IEEE*, 2013, pp. 167–174, "doi: 10.1109/GreenTech.2013.33".

[28] K. Bernsmed, M. G. Jaatun, P. H. Meland, and A. Undheim, "Security SLAs for Federated Cloud Services," in *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*, 2011, pp. 202–209, "doi: 10.1109/ARES.2011.34".

[29] A. V. Dastjerdi, S. G. H. Tabatabaei, and R. Buyya, "A dependency-aware Ontology-based Approach for Deploying Service Level Agreement Monitoring Services in Cloud," *Softw. Pract. Exp.*, vol. 42, no. 4, pp. 501–518, 2012, "doi: 10.1002/spe.1104".

[30] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," *ACM SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 164–177, 2003, "doi:

10.1145/945445.945462 ".
[31] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility," *Futur. Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009, 2009, "doi: 10.1016/j.future.2008.12.001".
[32] R. Jain and S. Paul, "Network Virtualization and Software Defined Networking for Cloud Computing: a Survey," *IEEE Commun. Mag.*, vol. 51, no. 11, pp. 24–31, 2013, "doi:10.1109/MCOM.2013.6658648".
[33] M. Shiraz, S. Abolfazli, Z. Sanaei, and A. Gani, "A Study on Virtual Machine Deployment for Application Outsourcing in Mobile Cloud Computing," *J. Supercomput.*, vol. 63, no. 3, pp. 946–964, 2013, "doi: 10.1007/s11227-012-0846-y".
[34] F. Zhang, J. Cao, K. Hwang, and C. Wu, "Ordinal Optimized Scheduling of Scientific Workflows in Elastic Compute Clouds," in *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, 2011, pp. 9–17.
[35] F. Lao, X. Zhang, and Z. Guo, "Parallelizing Video Transcoding Using Map-reduce-Based Cloud Computing," in *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on*, 2012, pp. 2905–2908, "doi: 10.1109/ISCAS.2012.6271923".
[36] T. Xie, X. Qin, and A. Sung, "SAREC: A Security-Aware Scheduling Strategy for Real-time Applications on Clusters," in *Parallel Processing, 2005. ICPP 2005. International Conference on*, 2005, pp. 5–12, "doi: 10.1109/ICPP.2005.68".
[37] M. A. Arfeen, K. Pawlikowski, and A. Willig, "A Framework for Resource Allocation Strategies in Cloud Computing Environment," in *Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual*, 2011, pp. 261–266, "doi: 10.1109/COMPSACW.2011.52".
[38] F. López, M. A. Amutio, J. Candau, and J. A. Mañas, "Methodology for Information Systems Risk Analysis and Management," *Minist. Public Adm.*, 2005.
[39] T. Aven, "Risk analysis. Assessing uncertaintie beyond expected values and probabilities, 2008." Wiley, Chichester, UKT, "doi: 10.1002/9780470694435".
[40] M. J. Thaheem, A. De Marco, and K. Barlish, "A Review of Quantitative Analysis Techniques for Construction Project Risk Management," in *Proceedings of the Creative Construct Conference*, 2012, pp. 656–667.
[41] K. Gokulnath and R. Uthariaraj, "Game Theory Based Trust Model for Cloud Environment," *Sci. World J.*, vol. 2015, 2015, "doi: 10.1155/2015/709827".

Authors' Profiles

Ala Arman was born on September 4, 1982. He received his master's degree in "software engineering of distributed systems" at Royal Institute of Technology (KTH), Stockholm, Sweden, in 2012. His master's dissertation was on "Automated Control of Elasticity for a Cloud-Based Key-Value Store". In 2014, he joined as a PhD researcher at the computer science department of the Universit à degli Studi di Milano. His research interests include security and resource management in cloud computing.

How to cite this paper: Ala Arman, "A Risk-Aware Application Scheduling Model in Cloud Computing Scenarios, International Journal of Intelligent Systems and Applications (IJISA), Vol.8, No.10, pp.11-20, 2016. DOI: 10.5815/ijisa.2016.10.02