supercomputing manuscript No.
(will be inserted by the editor)

User Subscription based Resource Management for Desktop-as-a-Service Platforms

Bert Vankeirsbilck · Lien Deboosere · Pieter Simoens · Piet Demeester · Filip De Turck · Bart Dhoedt

the date of receipt and acceptance should be inserted later

Abstract The Desktop-as-a-Service (DaaS) idiom consists of utilizing a cloud or other server infrastructure to host the user's desktop environment as a virtual desktop. Typical for cloud and DaaS services is the pay-as-you-go pricing model in combination with the availability of multiple subscription types to accommodate the needs of the users. However, optimal cost-efficient allocation of the virtual desktops to the infrastructure proves to be a combinatorial NP-hard problem, for which a heuristic is presented in the current article. We present a cost model for the DaaS service, from which a revenue of different configurations of virtual desktops to the servers can be derived. In this cost model, both subscription fee and penalties for degraded service are recorded, that are described in Service Level Agreements (SLAs) between the service provider and the users, and make realistic assumptions that different subscription types result in particular SLA contracts. The heuristic proposed states that for a given user base for which the virtual desktops (VDs) must be hosted, the VDs should be spread evenly over the infrastructure. Experiments through discrete event simulation show that this heuristic yields an approximation within 1% of the theoretically achievable revenue.

Keywords User profile · subscription type · cloud computing · resource overbooking · resource allocation · DaaS

1 Introduction

As cloud computing emerged, the possibilities of remote computing (i.e., the paradigm where heavy computations are not executed on the terminal but

Ghent University - iMinds
Gaston Crommenlaan 8, bus 201, B-9050 Ghent, Belgium.
E-mail: Bert.Vankeirsbilck@intec.ugent.be
Lien Deboosere: Melexis NV Belgium

rather on a remote server), regained public attention. The advantages of ubiquitous access to personal data and applications without the need for complex configuration and maintenance of hardware and upgrades has proven to appeal to the general community and has even been put forward as a fifth utility [1]. From a sustainable environment point-of-view, sharing computing resources between users provides opportunities to optimize the resource distribution. Instead of overprovisioning personal computers that are largely underutilized during their lifecycle, resources can be leased adequately. For example, volunteer computing initiatives such as BOINC [2] and XtremWeb [3] use the spare resources of personal computers of volunteers for scientific or other computational work. With over 250,000 volunteers and nearly 600,000 computers contributing to BOINC as of March 10, 2013, an average (measured over 24 hours) of 9.716 PFLOPS is available this way. Although being aware that no homogeneous computer network is used, this would comply to an average of over 16 GFLOPS spare computing resources per computer.

In this light, the Desktop-as-a-Service (DaaS) idiom is particularly interesting. This technology consists of providing remote desktop services using cloud computing, typically by executing virtual machines that host an entire desktop environment for the users, hence the term Virtual Desktop (VD). The user connects to his VD using a thin client protocol (e.g., Microsoft Remote Desktop Protocol (RDP) [4] or Virtual Network Computing (VNC) [5]), using a viewer device that merely requires to be capable of transceiving user input and graphical responses over the network and handling user interaction. At the cloud site, multiple VDs can be co-hosted on a physical server. The incentives for the service provider to optimally employ computing resources to ensure a high Return on Investment (RoI) naturally drive towards placing the spare resources of one user's VD on the server at the disposal of an other's. However, as customer base size and customer satisfaction are also an important factor in the revenue model of the service provider, the resource sharing must be managed adequately. Service Level Agreements (SLAs) are the contracts between the customer and the service provider, that aim to define rights and duties of both parties in order to describe the mutual expectations, and hence serve as a model for the customer satisfaction. If fair SLAs are negotiated, the service provider can expect the users to be satisfied if the provider keeps true to the duties defined in the contract. This way, the negotiated SLAs serve as a means to reconcile the interests of the customers and the service provider, where there is a low probability for the user to experience resource shortages while the server infrastructure utilization is highly profitable.

In a practical interpretation, the number of SLA violations i.e., occurrences that the demanded resources comprised in the subscription could not be allocated to the users VD, can be seen as a measure of the reliability of the resource provisioning. The service provider sets those guarantees differently depending on the subscription type (and hence the paid service fee), allowing for different margins on service quality. In this work, we have focused on compute resources. In practice, a shortage of compute resources will be perceived by the user as extended application execution times. In the case of scientific applications that are executed on High Performance Computing (HPC) clusters and are non-interactive by nature, a slightly longer execution time in the order of seconds or minutes for bulk tasks could typically be tolerated. However, this paper focuses on virtual desktops with a more interactive nature, i.e., a remote graphical interface that is to be operated over a network. In such environments, in total about 150 ms of end-to-end latency from user mouse or keyboard action to presentation of the result to the user, or in more delay sensitive cases e.g., gaming, only 80 ms is tolerable [6]. In this case, the decreased compute resources on the server will have a considerable impact on the interactivity and the overall Quality of Experience (QoE). A gold user will have more guarantees on the resources allocated and hence on superior quality, while a silver user will need to accept a relatively inferior quality.

In this paper, we focus on the resource management in DaaS, proposing a service cost model taking user subscription fees and SLA contract violation penalties into account. Using this model, the DaaS service provider can optimize the resource utilization as described above while keeping the user satisfied, thereby optimizing service revenue. In this model, we specifically account for multiple user subscription types, where users pay a different fee for the service, expecting adequate service quality in return. We investigate how these subscription types influence the strategies for distributing virtual desktops in the server park.

The remainder of this paper is structured as follows. Related work is discussed in Section 2. The parameters of the model of the envisioned DaaS service are described in Section 3. Our simulation environment used and the obtained experimental results are presented in Section 4. Finally, conclusions are drawn in Section 5.

2 Related work

In prior work [7], we proposed allocation, reallocation and consolidation strategies based on overbooking without accounting for different user subscriptions. The current paper leverages this prior work to derive models that take user subscription types into account that incorporate different subscription fees, SLA policies and SLA violation penalties.

Resource algorithms for static workloads of web applications in a virtualized service hosting platform are presented in [8]. The metric used in these algorithms is the *yield* which is defined as the ratio between the resource fraction allocated and the maximum resource fraction potentially used. Furthermore, the algorithms assume static workloads, i.e., workloads with a constant resource need. However, this assumption is not valid for the DaaS cuse studied in this paper. In our work, the resource requests are modeled as a non-constant probability distribution. Also, in our cost model, penalties for SLA violations are taken into account to base our equivalent for the *yield* function and allocation decisions on.

Resource overbooking techniques have been studied in various contexts, e.g., in the field of video-on-demand servers [9] and Asynchronous Transfer Mode (ATM) networks [10]. Such overbooking methods stem from revenue management [11], and are related to pricing models. In the current work, overbooking is also associated with different pricing models to represent a realistic cost model. Furthermore, since our work considers stochastic resource request models, the models derived here resemble approaches typically encountered in classic logistic overbooking situations with *stochastic no-shows or cancelations* existing in airline [12] or hotel services [13]. The importance of resource overbooking and application profiling in a shared internet hosting platform is demonstrated in [14]. The target is to maximize the revenue of the resource provider. It is shown that by co-hosting different kinds of services (IO bound vs. CPU bound) on the servers instead of dedicating servers to one specific service type, the service hosting efficiency can be further increased. In particular, the paper shows that combining a network intensive Apache web service with a compute intensive PostgreSQL backend service allows supporting 2 to 4 times more applications with equal infrastructure than partitioning the servers into specialized Apache and Postgres instances. This optimized service placement policy allows a lower price for the DaaS service as the cost per VD decreases. In contrast to our work, the case of different subscription types contending for the same class of resources is not studied. Our work assumes contention for a single resource type, i.e., compute resources. However, the model and approach could be extended to optimize placement of services with requirements for multiple resource types.

3 Infrastructure and user model

We derive a model for allocation of VDs in DaaS based on the trade-off between costs and revenue for the service provider. VDs are defined in terms of a request distribution, which is essentially modeled as a probability density function of the amount of resources requested by the VD. Typically, customers pay a monthly fee to subscribe to the service and pay for the used resources on a pay-as-you-go basis for DaaS. The subscription type the customer opts for defines the fee, the amount of resources that will be put available to the customer's VD as well as the service quality guarantee. This service quality guarantee is defined in an SLA, that states in which fraction of the requests, the requested resources will be allocated to the VD. In this view, an SLA violation is a term used to indicate the event that the service provider fails to provide the resources as described in the SLA. To further formalize the quality guarantees, the SLA also specifies the modalities form compensation for SLA violations. In our model, the guarantee to avoid SLA violations is implemented as a reservation of resources according to the VD request distribution. For example, an SLA contract guaranteeing that in 90% of the requests, the requested resources will be allocated to the VD, can be enforced by reserving the 90^{th} percentile of the request distribution of the VD. However, the amount of reserved resources limits the number of VDs hosted on the server. Furthermore, there is a probability that fewer resources will be requested than reserved. Both observations have fed the popularity of applying overbooking as a method to increase the utilization of the server infrastructure. This technique involves reserving less resources for the VDs than needed to ensure meeting the negotiated allocation guarantee.

For the derivation of the revenue models in this section, a staged approach is taken. First, the case of hosting VDs without taking overbooking into account is investigated. Then, the effect of overbooking is evaluated, in the assumption that a VD can not acquire additional resources except for those reserved for the VD. Finally, overbooking is considered where a VD can be allocated free resources available on the server due to underreservation of the capacity or non-utilization of reserved resources.

3.1 No overbooking

We first examine the case when no overbooking is applied, i.e., where all users are guaranteed to receive the requested resources at all times. We are interested in finding the optimal selection of users on a server. In other words, we look for the number of VDs n_i from N subscription types that maximizes the total revenue for the service provider, within the constraint that no more resources can be reserved than available on the server, as in Equation (1).

$$\begin{aligned} \mathit{Revenue}_h &= \sum_{i=1}^{N} n_{i,h}\alpha_i \\ \mathit{subject\ to} \ \sum_{i=1}^{N} n_{i,h}R_i &\le C_h \end{aligned} \tag{1}$$

with:

N	= the number of user subscription types
$n_{i,h}$	= the number of VDs of subscription type i on hosting server h
α_i	= the revenue from VDs of subscription type i
R_i	= the resources reserved for VDs of subscription type i
C_h	= the capacity of the hosting server h

Solving this optimization problem using the method of Lagrange multipliers [15] yields two distinct cases. The first represents the case in which $\frac{\alpha_i}{R_i}$, i.e., the revenue per reserved resource, is equal for each subscription type. In this case, there is no preference of hosting one type of VD over another. Otherwise, a given subscription type would yield more revenue per reserved resource, implying that this subscription type is preferred over others to allocate resources to.

Adding to the obtained optimal solution, the DaaS service provider can optimize the servers utilization factor by grouping VDs such that the unreserved resources on the server are minimized.

3.2 Overbooking

Applying overbooking implies more resources are contracted than physically available on the server infrastructure. The degree of overbooking defines how many resources are underreserved, and has an impact on the probability of allocating less resources to the user than reserved. Due to this overbooking, there is no absolute guarantee that customers will be allocated the requested resources as described in the SLA, introducing costs in the form of penalties. For the DaaS provider, overbooking creates the opportunity to host additional VDs on a server.

$$\begin{aligned} Revenue_i &= \alpha_i + \beta_i P[R_i - OD_i < req \leq R_i] \\ &= \alpha_i + \beta_i \left(F_i(R_i) - F_i(R_i - OD_i)\right) \end{aligned} \tag{2}$$

with:

OD_i	= the amount of underreserved resources for VDs of subscription type i
req	= the amount of resources requested by the VD
β_i	= the cost (penalty) involved when a VD of subscription type i is not allocated the requested resources (with $\beta_i < 0$)
$F_i(r)$	= the cumulative probability distribution of resource request sizes from VDs with subscription type i

From the revenue model for individual subscription types in Equation (2), we construct a model including the varying subscription types to evaluate how VDs can be optimally selected to maximize total revenue on the host server in Equation (3).

$$\begin{aligned} Revenue_h &= \sum_{i=1}^{N} [n_{i,h}(\alpha_i + \beta_i(F_i(R_i) - F_i(R_i - OD_i)))] \\ &\qquad \textit{subject to } \sum_{i=1}^{N} [n_{i,h}(R_i - OD_i)] = C_h \end{aligned} \tag{3}$$

Again, applying the method of Lagrange multipliers yields two cases. In the first case, $\frac{\alpha_i+\beta_i(F_i(R_i)-F_i(R_i-OD_i))}{R_i-OD_i}$ is identical for all subscription types. This means that the trade-off between earned subscription fees and incurred penalties for SLA violations per actually reserved resource should be constant over the different subscription types. Note that based on the overbooking degree, the parameters α and β will vary. More specifically, from the overbooking degree fixed over the entire infrastructure, the conditions are computed in function of the values for α_i and β_i which vary per subscription type. Hence, all VDs of an equal subscription type are subject to the same α and β. In the second case, the revenue per subscription type differs, implying that one subscription type is preferred to be hosted over others.

3.3 Resource pool

The previous cases ignore that spare resources on the server can also be used to accommodate VDs that request more resources than reserved. Especially when overbooking is applied, distributing spare capacity at runtime will decrease the penalties incurred by SLA violations. To this end, the unreserved resources and the reserved but unused resources from the VDs are gathered into a resource pool.

As users from different subscription types exhibit particular resource request patterns they also differ in the average amount of resources contributed and extracted from the resource pool. The probability distribution of the amount of resources contributed to the resource pool per VD, is presented in Equation (4) and depends on the request distribution of the VD. We consider a positive amount of added resources to the resource pool as this allows to characterize the distribution of the initial resource pool size. Therefore, the probability of adding a negative amount of resources is 0. In case a VD requests more resources than there have been reserved for, no resources can be added to the resource pool, hence the probability of adding 0 resources to the resource pool equals $P[req \geq R_i - OD_i] = 1 - F_i(R_i - OD_i)$. A VD adding a positive amount of resources x results from requesting x resources less than reserved. Hence, the probability of adding this x resources is equal to $f_i(R_i - OD_i - x)$. An analog deduction can be made to obtain the distribution of resources extracted from the resource pool.

$$a_i(x) = \begin{cases} 0 & x < 0 \\ 1 - F_i(R_i - OD_i) & x = 0 \\ f_i(R_i - OD_i - x) & x > 0 \end{cases} \tag{4}$$

with:

$f_i(r) =$ the probability distribution function of resource request sizes from VDs with subscription type i

The average amount of resources added to the resource pool per VD can be derived from this distribution as the weighted average of the possible values $E[Add_i] = \int_{-\infty}^{\infty} x a_i(x) dx$. The average size of the resource pool can then be written as in Equation (5).

$$\begin{aligned} E[ResourcePoolSize_h] = C - \sum_{i=1}^{N}(n_{i,h}(R_i - OD_i)) \\ + \sum_{i=1}^{N}(n_{i,h}E[Add_i]) \end{aligned} \tag{5}$$

Although it is possible to compute the total resource request size distribution, the resource pool size distribution and the distribution of total resource demanded from the resource pool that results from the simultaneous requests of the VDs on a server, they do not allow to draw any conclusions on the probability of experiencing SLA violations per individual VD or per VD subscription category. More specifically, the influence of the availability of a resource pool that can accommodate additional resources needed by the VDs on the host will decrease the probability of SLA violations, but the exact value is hard to model theoretically in a closed form.

3.4 Allocating Virtual Desktops in the cloud infrastructure

As the server park is composed of individual physical machines, a strategy needs to be divised to allocate VDs to their machines. Theoretically, this translates into an NP-hard bin-packing problem as discussed in detail in [16]. As discussed in [7], the allocation of VDs to the available servers needs to be regularly re-assessed due to the dynamic arrival and departure process of the customers and consolidation of VDs for the sake of putting servers into sleep mode for energy-saving purposes. More specifically, exhaustively computing the optimal configuration of number of VDs n_i of all subscription types involves assessing a full factorial over all servers, from which the combination with the highest total revenue must be chosen. As the number of servers in cloud computing infrastructure typically are large and the number of subscription types can be considerable as well, and this exhaustive search method does not scale well, it is not appropriate for run-time execution. Therefore, we propose a heuristic method to arrive at a suitable distribution of the VDs over the available servers. This heuristic is based on solving a relaxed problem that serves as an estimate to the original problem. In a second step in the heuristic method, the solution of the estimated problem is approached for the original problem statement.

The original allocation optimization problem is described in Equation (6), stating that the service revenue should be maximized from choosing the combined optimal configuration of users over all servers.

$$\begin{gathered} Max\left(\sum_{h=1}^{M} Revenue_h\right) \\ subject\ to\ \sum_{i=1}^{N}(n_{i,h}R_i) \leq C_h \end{gathered} \tag{6}$$

with:

M = the number of servers available in the infrastructure

The constraints in this problem have been relaxed such that the server park is approached as one instance with a capacity of the sum of all servers, in which all VDs should be allocated, as formulated in Equation (7).

$$
\begin{aligned}
&Max\left(\sum_{h=1}^{M} Revenue_h\right)\\
&\textit{subject to}\ \sum_{h=1}^{M}\sum_{i=1}^{N}(n_{i,h}R_i) \leq \sum_{h=1}^{M} C_h
\end{aligned}
\tag{7}
$$

Given the optimization constraints derived in Section 3.1 and Section 3.2 and assuming that the current set of VDs to allocate to the infrastructure is known, the overbooking degree and pricing model can be set. Hence, solving this relaxed problem yields an upper bound on the revenue achievable with the hosting infrastructure. Specifically, as a result of relaxing the problem, the resource pool can be shared over all VDs, in contrast to the original problem definition where a resource pool per hosting server is available for the VDs allocated to that particular server only. For this relaxed problem statement, the average resource pool size can be easily computed as the sum of the averages of the VD contributions to the resource pool, as described in Equation (8).

$$
\begin{aligned}
E[ResourcePoolSize_{relaxed}] = &\sum_{h=1}^{M}(C_h - \sum_{i=1}^{N}(n_{i,h}(R_i - OD_i))\\
&+ \sum_{i=1}^{N}(n_{i,h}E[Add_i]))
\end{aligned}
\tag{8}
$$

The second step in the heuristic algorithm involves tracing back the solution from the relaxed problem to the original problem. This step is approached by defining how to divide the total resource pool as available in the relaxed problem over the M servers in the infrastructure. To balance SLA-violations between servers, we pursue an equal resource pool for each server, i.e., $E[ResourcePoolSize_{approx}] = E[ResourcePoolSize_{relaxed}]/M$, that serves as an approximation of the resource pool size that would be obtained by the optimal configuration of VDs. Computation of the number of VDs of each subscription type, as those found in the relaxed solution can simply be divided by the number of servers. For a heterogeneous server park, a proportional division provide the optimal continuous number of VDs of the various subscription types. The resulting value of VDs per server requires to be rounded to the lower discrete value. To allocate the remaining VDs that are not yet allocated due to fractioning in the distinct servers, a best-fit strategy can be adopted to host these as well.

Finally, the heuristic allocation strategy results in a configuration as described below:

– *Step 1: Even distribution of VDs over all servers*
The base number of VDs of subscription type i on each host is given by:

$$
n_i^{base} = n_i/M \tag{9}
$$

– *Step 2: Allocation of VD fractions* The number of fractioned VDs to be allocated and spread in a best-fit way over the servers is given by:

$$n_i^{fractioned} = n_i \% M \tag{10}$$

4 Experimental results

4.1 Experiment setup

To assess the impact of applying a resource pool with overbooking for multiple subscription types, we consider two subscription types, *golden* and *silver* to be distributed optimally over the servers in a DaaS environment. For the golden subscription type, the SLA defines that in 99.865% of the resource requests, the requested resources will be allocated to the VD of the user. Assuming the request sizes of these subscribers to be normally distributed around 25000 resources with a standard deviation of 5000, a reservation of $R_g = F_g^{-1}(99.865\%) = \mu_g + 3\sigma_g = 40000$ resource units ensures this guarantee to be fullfilled. The silver subscription type has less guarantee: 84.135% of the resource requests will be integrally allocated. We assume the VDs of these subscribers to be normally distributed around 10000 resources with a standard deviation of 1500. Hence, a reservation of $R_s = F_s^{-1}(84.135\%) = \mu_s + \sigma_s = 11500$ resource units ensures meeting the promised allocation guarantee. These distributions were obtained by the observation that the resource requirement of a virtual desktop depends on many factors such as the amount of active applications and events in the operating system. Assuming that the central limit theorem [17] can be applied drives to model the resource request distribution for a virtual desktop of type i as a normal distribution $N(\mu_i, \sigma_i^2)$. The choice of means μ_i are derived from the planning guide described in [18], that states that a server is able to host on average 10 normal VDs or 4 heavy VDs. For our experiments, we chose the golden subscription type to use heavy VDs and the silver subscribers to use normal VDs.

Assuming that the service provider uses a cost model such that a fair customer service is obeyed, the conditions discussed in Section 3.1 state that the revenue per reserved resource unit should be equal for each subscription type. Assuming that for the golden subscription, a revenue α_g of 400 credits per reserved resource is set, the subscription fee for the silver subscription is determined proportionaly to yield $\alpha_s = \frac{\alpha_g R_s}{R_g} = 115$ credits per reserved resource.

When considering resource overbooking within a customer service founded on fairness between subscription types, the penalties involved with breaking the negotiated guarantee to allocate the requested resources to the customer's VD are determined as described in Section 3.2. Assuming the penalty per SLA violation for the golden subscriptions is set to a reasonable $1/8^{th}$ of α_g, the penalty for the silver user is derived depending on the overbooking degree. The derived configurations for the experiment are summarized in Table 1.

Table 1 Overview of the configurations applied for the experiments.

Subscription type & request distribution	R_i	SLA guarantee (%)	α_i	Overbooking 1000 β_i	Overbooking 2000 β_i	Overbooking 3000 β_i
Golden (N(25000, 5000^2))	40000	99.865	400	-50	-50	-50
Silver (N(10000, 1500^2))	11500	84.135	115	-34.7	-32.4	-49.2

Using these assumptions, we have simulated VD resource requests, resource overbooking and resource allocations in an extension of the CloudSim 2.0 environment [19]. CloudSim is a discrete event simulator for modeling and simulating cloud computing infrastructures and services. The extensions to this simulation environment include the provisioning for reserving resources for VDs and the facilities to gather resources in a resource pool as well as to distribute them to extracting VDs. While simulating 1600 VD requests per VD per simulation, 30 iterations per experiment were executed. We assume a user base of 10 golden subscription VDs and the remainder to be silver subscription VDs, and assume a homogeneous server park with a capacity of each individual server of 150000 resources.

4.2 Results

4.2.1 No overbooking

When no overbooking is applied and given the capacity of the servers in the server park, the amount of VDs of each subscription type that can be co-hosted on each server is easily determined. Knowing the number of VDs of each type, the revenue yielded by the server can be computed according to the model in Equation (1). Table 2 presents the outcome of these computations for the possible co-hosting configurations of subscription types in our experiments. From these options, the optimum allocation of VDs to servers can be determined. As obeying fair customer service implies that the pricing model is set such that the revenue per reserved resource unit is equal for the different subscription types, the revenue created in this case without overbooking is linear with respect to the total amount of resources reserved on the server. Hence, as presented in the table, although the subscription fees were computed for equal revenue, the different configurations persented in the table do not create identical total revenue for the server due to the varying amount of total reserved resources.

4.2.2 Overbooking without resource pool

When overbooking is applied, the probability of encountering SLA violations (i.e., $P_i[SLA]$ for subscription type i) increases. As a result, the penalties for SLA violations could change the revenue of co-hosting configurations which influences the preferences for co-hosting configurations of subscription types. As

Table 2 Allocation alternatives and the resulting revenue per server in the case no overbooking is applied.

n_g	0	1	2	3
n_s	13	9	6	2
Reserved (resource units)	149500	143500	149000	143000
Unreserved (resource units)	500	6500	1000	7000
Revenue (credits)	1495	1435	1490	1430

Table 3 Allocation alternatives and the resulting revenue per server in the case overbooking is applied without maintaining a resource pool. Note that for the different overbooking values specific α_i and β_i are used as computed in Table 1.

Overbooking (resource units)	n_g	$P_g[SLA]$ (%)	n_s	$P_s[SLA]$ (%)	Unreserved (resource units)	Revenue (credits)
1000	0	/	14	21.09	3000	1507.53
	1	0.11	10	21.07	6000	1476.81
	2	0.12	6	20.99	9000	1446.18
	3	0.12	3	20.98	1500	1522.98
2000	0	/	15	47.18	7500	1495.70
	1	0.35	11	47.26	7500	1496.37
	2	0.35	7	47.27	7500	1497.47
	3	0.33	3	47.30	7500	1498.54
3000	0	/	17	68.27	5500	1383.98
	1	0.70	13	68.31	2500	1457.74
	2	0.69	8	68.23	8000	1450.77
	3	0.69	4	68.28	5000	1524.58
	4	0.70	0	/	2000	1598.60

the VDs can only be allocated at most the reserved resources, the probability of not acquiring the requested resources is fixed to $1 - F_i(R_i - OD_i)$, independent of other VDs on the same host. Hence, the probability of encountering an SLA violation is defined by $F_i(R_i - F_i(R_i - OD_i))$ as seen in the results from the simulation presented in Table 3. These results were obtained through simulation in CloudSim, resulting in SLA violation probabilities that comply to the theoretical values ($P_g[SLA]$ of 0.12%, 0.33% and 0.68%, and $P_s[SLA]$ of 21.08%, 47.19% and 68.27% for overbooking degrees of 1000, 2000 and 3000 resource units respectively), indicating that sufficient samples are taken to ensure statistical relevance. Again, as different configurations of subscription types on a server result in a varying amount of total reserved resources, the configuration with the least unreserved resources will create the highest revenue.

4.2.3 Overbooking with resource pool

Maintaining a resource pool and reusing the spare resources collected in it guarantees a decrease in the probability of SLA violations in comparison to the overbooking case without resource pool. Indeed, the resource reservations still apply but additional resources could be acquired from the resource pool, determining the overbooking case without resource pool as the lower bound

Table 4 Allocation alternatives and the resulting revenue per server in the case overbooking is applied with a resource pool from which resources can be extracted. Note that for the different overbooking values specific α_i and β_i are used as computed in Table 1.

Overbooking (resource units)	n_g	$P_g[SLA]$ (%)	n_s	$P_s[SLA]$ (%)	Unreserved (resource units)	Revenue (credits)
1000	0	/	14	0.03	3000	1609.83
	1	< 0.01	10	< 0.01	6000	1550
	2	< 0.01	6	< 0.01	9000	1490
	3	< 0.01	3	< 0.01	1500	1545
2000	0	/	15	2.45	7500	1713.09
	1	0.01	11	0.04	7500	1664.84
	2	< 0.01	7	< 0.01	7500	1605
	3	< 0.01	3	< 0.01	7500	1545
3000	0	/	17	13.15	5500	1844.97
	1	0.15	13	6.52	2500	1853.25
	2	0.01	8	0.05	8000	1719.80
	3	< 0.01	4	< 0.01	5000	1660
	4	< 0.01	0	/	2000	1600

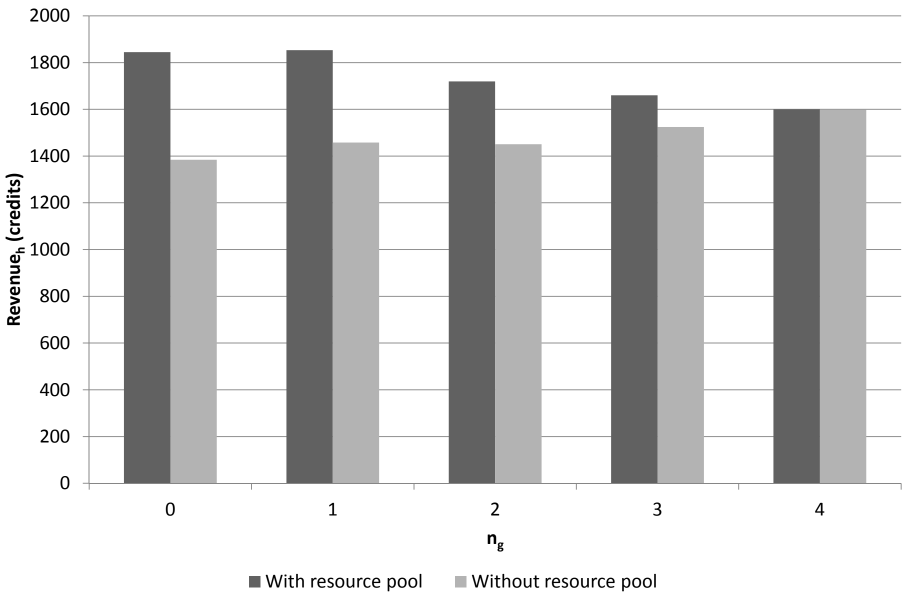

Fig. 1 Comparison of the revenue created in a identical configurations of VDs, for an overbooking degree of 3000 resource units.

on the revenue. Table 4 shows how the revenue increases due to the decrease in SLA violations.

Figure 1 compares the revenue resulting from equal configurations for either overbooking with or without resource pool expressed as a function of the average resource pool size per user. We can conclude that optimal configurations in the case without resource pool do not guarantee optimality for the case with resource pool. In fact, as shown in the figure, the conclusions could as

Table 5 Outcome of the heuristic approach for hosting 13 golden VDs and 27 silver VDs on a homogeneous server infrastructure consisting of 5 servers with capacity 150000 resource units each.

n_g	$P_g[SLA]$ (%)	n_s	$P_s[SLA]$ (%)	Revenue (credits)
13	< 0.01	27	< 0.01	8305
3	< 0.01	5	0.05	1774.92
2	< 0.01	6	< 0.01	1490

well be adverse. Specifically, for the case without resource pool, the optimum is dependent of the unreserved resources on the host only, while in the case with a resource pool the size distribution of the resource pool as well as the number of VDs that require resources from it influence the probability of SLA violations. The figure correlates to hosting VDs as in Table 3 and Table 4, for an overbooking degree of 3000 resource units.

4.3 Allocating Virtual Desktops in the cloud infrastructure

Applying the proposed heuristic described in Section 3.4 to distribute the VDs for optimizing the total service revenue obtained can be computed when considering the server park as a unified computing instance. This represents the theoretical maximum revenue that can be obtained for the given set of VDs and the total capacity of the infrastructure. This situation is simulated in the CloudSim environment, where 13 golden VDs and 27 silver VDs were hosted on a homogeneous server infrastructure composed of five servers. These servers have a resource capacity of 150000 resource units each. The VDs to be distributed over the infrastructure are characterized as in Table 1. The relaxed problem, i.e., pretending the total server infrastructure is one large server, yields a revenue of 8305 credits. When coupling back to the original problem, the heuristic imposes that at least $13/5 = 2$ golden VDs and $27/5 = 5$ silver VDs are to be hosted per server. The fractioned VDs are distributed over these servers in a best-fit way. This results in $13\%5 = 3$ golden VDs and $27\%5 = 2$ silver VDs must be distributed over the 5 servers. Hence finally, the heuristic states that 3 servers should host 3 golden VDs in combination with 5 silver VDs, and the other 2 servers should host 2 golden VDs in combination with 6 silver VDs to create optimal revenue. Note that the resulting configurations might require increasing the overbooking degree to enable accomodating all VDs from the user base.

The results from this experiment are presented in Table 5, and show that combined, the configurations in the server park yield 8304.7 credits, which is less than 1% under the theoratically archievable revenue.

The performance of the proposed heuristic has been compared to results obtained by alternative approaches to configure a predefined number of VDs on the server infrastructure. To this end, we assume a server park of 5 servers with capacity of 150000 resource units each. Three use cases were evaluated, i.e., for user bases with 10 golden and 44 silver VDs, 16 golden and 33 silver VDs and 19 golden and 13 silver VDs. For these use cases, adequate overbooking degrees and cost parameters were configured. For example, an overbooking degree of 3000 resource units is applied for the first use case where 10 golden VDs and 44 silver VDs are to be hosted on the infrastructure. The theoretical maximum revenue is obtained through simulation of this configuration, with an infrastructure consisting of a single server with the combined capacity of all servers actually deployed case, harnessing a capacity of 750000 resource units. The resulting revenue amounted to 9060 credits as shown in Table 6. Figure 2 presents the results achieved with the three use cases, where the proposed heuristic, indicated as *spread* is compared to the theoretical maximum revenue as well as alternative approaches. For the first alternative approach considered, mixture of the VDs of different types on one host is avoided, hence the naming *dedicated servers*. The second alternative encompasses a First Come, First Served (*FCFS*) approach, where VDs are allocated to hosts as they arrive into the system. As the arrival process is simulated by randomization, the average value is considered as the revenue for this approach. The figure shows that the proposed heuristic yields the highest revenue. For the cases studied, at least 99.6% of the theoretically obtainable revenue is achieved.

5 Conclusion

In this paper, the pay-as-you-go pricing model in combination with the availability of multiple subscription types to accommodate the needs of the users is modeled for cloud and DaaS services. Gradually increasing the complexity of the service management strategies, we derive models that indicate how optimal cost-efficient allocation of the virtual desktops to the infrastructure is obtained. A common used technique to optimize revenue in the field of operations and logistics is resource overbooking, that is based on contracting more resources than physically available on the server infrastructure. Applying this technique in combination with allowing to share free resources between VDs on the hosting server creates a situation in which the optimization problem is NP-hard. Therefore, a heuristic is proposed that aims to spread the VDs of the different subscription types over the server park. Using the CloudSim environment, simulations of the theoretically validatable cases are shown to be accurate, ensuring that the sampling size is adequately chosen. With this simulation environment, the proposed heuristic is evaluated in terms of performance in relation to the optimum revenue that is obtained when assuming

Table 6 The resulting revenue in the case overbooking is applied with a resource pool from which resources can be extracted, considering the server park as one large server. Note that for the different overbooking values specific α_i and β_i are used as computed in Table 1

Overbooking (resource units)	n_g	$P_g[SLA]$ (%)	n_s	$P_s[SLA]$ (%)	Unreserved (resource units)	Revenue (credits)
3000	10	< 0.01	44	< 0.01	6000	9060

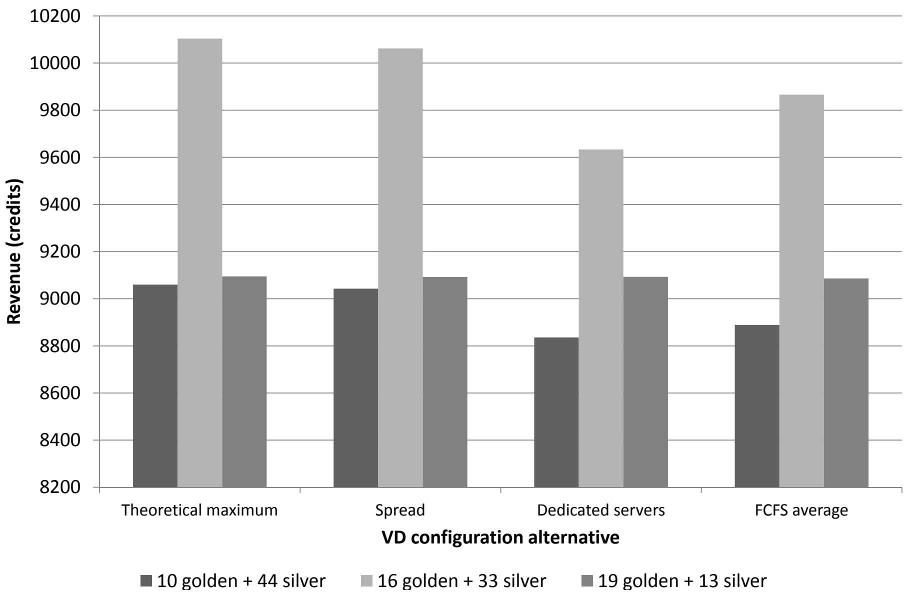

Fig. 2 Comparison of the revenue achieved by the theoretical case of assuming one large server instance, the proposed heuristic (spread), the alternative approache where mixture of VD types on a host is avoided (dedicated servers) and a First Come First Served strategy (FCFS) of which an average of the possible configurations is taken. Three use cases with varying number of VDs of each type are evaluated.

that the server infrastructure is one large server with a capacity the sum of the resources of all individual servers. The heuristic is also compared to allocation strategies. It is shown that the proposed heuristic outperforms the others and provides the best approximation of the theoretic optimum.

Acknowledgements

Bert Vankeirsbilck is funded by a Ph.D grant from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). Part of this work has been funded by the UGent GOA project "Autonomic Networked Multimedia Systems".

References

1. Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6):599 – 616, 2009.
2. D.P. Anderson. Boinc: a system for public-resource computing and storage. In *Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on*, pages 4–10, Nov 2004. http://boinc.berkeley.edu/.

3. Franck Cappello, Samir Djilali, Gilles Fedak, Thomas Herault, Frédéric Magniette, Vincent Néri, and Oleg Lodygensky. Computing on large-scale distributed systems: XtremWeb architecture, programming models, security, tests and convergence with grid. *Future Generation Computer Systems*, 21(3):417 – 437, 2005.
4. Microsoft Corporation. Windows Remote Desktop Protocol (RDP). http://msdn2.microsoft.com/en-us/library/aa383015.aspx.
5. Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, and Andy Hopper. Virtual network computing. *IEEE Internet Computing*, 02(1):33–38, 1998.
6. N. Tolia, D.G. Andersen, and M. Satyanarayanan. Quantifying interactive user experience on thin clients. *Computer*, 39(3):46 – 52, march 2006.
7. Lien Deboosere, Bert Vankeirsbilck, Pieter Simoens, Filip Turck, Bart Dhoedt, and Piet Demeester. Efficient resource management for virtual desktop cloud computing. *The Journal of Supercomputing*, 62(2):741–767, 2012.
8. Mark Stillwell, David Schanzenbach, Frédéric Vivien, and Henri Casanova. Resource allocation algorithms for virtualized service hosting platforms. *J. Parallel Distrib. Comput.*, 70:962–974, September 2010.
9. H. Vin, P. Goyal, and A. Goyal. A statistical admission control algorithm for multimedia servers. In *Proceedings of the second ACM international conference on Multimedia*, pages 33–40, 1994.
10. R.R. Boorstyn, A. Burchard, J. Liebeherr, and C. Oottamakorn. Statistical service assurances for traffic scheduling algorithms. *Selected Areas in Communications, IEEE Journal on*, 18(12):2651 –2664, December 2000.
11. Lawrence R Weatherford and Samuel E Bodily. A taxonomy and research overview of perishable-asset revenue management: yield management, overbooking, and pricing. *Operations Research*, 40(5):831–844, 1992.
12. Marvin Rothstein. An airline overbooking model. *Transportation Science*, 5(2):180–192, 1971.
13. Varda Liberman and Uri Yechiali. On the hotel overbooking probleman inventory system with stochastic cancellations. *Management Science*, 24(11):1117–1126, 1978.
14. Bhuvan Urgaonkar, Prashant Shenoy, and Timothy Roscoe. Resource overbooking and application profiling in a shared internet hosting platform. *ACM Transactions on Internet Technology*, 9:1:1–1:45, February 2009.
15. I.B. Vapnyarskii. *Encyclopedia of Mathematics*, chapter Lagrange Multipliers. Springer, 2001.
16. Michael R. Garey and David S. Johnson. *Computers and Interactability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.
17. M. Rosenblatt. A central limit theorem and a strong mixing condition. *Proceedings of the National Academy of Sciences of the United States of America*, 42(1):43–47, 1956.
18. Citrix. Xendesktop planning guide - hosted vm-based resource allocation, 2010.
19. Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A. F. De Rose, and Rajkumar Buyya. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *SOFTWARE-PRACTICE & EXPERIENCE*, 41(1):23–50, JAN 2011.