A Procurement Auction Market to Trade Residual Cloud Computing Capacity

Paolo Bonacquisto, Giuseppe Di Modica, *Member, IEEE*, Giuseppe Petralia, and Orazio Tomarchio, *Member, IEEE*

Abstract—Currently in the cloud market resources are mainly allocated according to the fixed-price, direct selling model. Market principles such as the supply-demand rate are not taken into consideration by cloud providers. In the literature several market-based resource allocation models and algorithms have been proposed, showing that dynamic pricing models might result more convenient for both the providers and the consumers of cloud resources. In this paper, we discuss about the use of alternative auction-based mechanisms to sell the "residual" computing capacity, i.e., the capacity which the provider is not able to allocate through direct-selling. The design of a procurement market for computing resources is proposed: in such a market we devised an adaptive bidding strategy that suggests to the provider the right actions useful to attain its business objective. The resource overbooking mechanism is also proposed to overcome the problem of resource underutilization. A software simulator was implemented with the aim of testing and validating the proposed mechanisms. Results show that, by fine-tuning their own strategy, providers manage to pursue specific objectives.

Index Terms—Cloud market, procurement auction, overbooking, adaptive strategy, CloudSim

1 Introduction

The pricing strategy mostly adopted by commercial IaaS providers is known as the "On-demand": customers are charged for the time frame during which the resource is actually utilized. Providers ask customers to pay a *fixed price* for accessing computing capacity by the hour. The only chance for the customer to get discounts on the price-per-hour is to opt for the *reservation* or the *flat rate* charging models. According to these options, customers get a discount on the price but are committed to longer periods of lease (from a month to a year). The pay-as-you-go model is intrinsic to the cloud computing paradigm. The cloud technology natively allows customers to consume resources for the strictly required time. If the on-demand pricing is clearly convenient to customers, for a provider it is also easy and cheap to enforce. But, imposing fixed prices for the hourly usage of a virtual machine (VMs) may not necessarily guarantee the highest revenues for the providers. Many argues that alternative schemes based on the price bargaining might turn to be more profitable for providers.

Providers will unlikely abandon the direct-sell mechanism. Regular customers issue requests which most of the times have a well known timing. For this kind of requests the most appropriate model is the *direct-sell/fixed-price*, in that it provides guarantees for both the provider and the customer. It is a matter of fact, however, that providers are not able to fully allocate their computing capacity. Taking a look at the one-hour time window, there is a variable portion of computing resources (which we are going to name *spare* resources) which remain unsold and therefore do not produce income. We argue that if direct-sell fails to allocate the overall providers' nominal computing capacity, alternative (possibly supply-demand based) pricing schemes should be adopted to allocate the spare capacity. Provided that costs for running the spare machines are covered, providers may be willing to sell that capacity at lower prices.

In this paper, we propose to employ a **procurement auction** mechanism to allocate providers' spare computing resources. We analyze the factors that mainly impact the strategic choices of providers in the allocation of computing capacity through auctions. The purpose of this work is to define a bidding strategy which, in the aim of maximizing the business objective, guides the providers in the choice of the right actions to take in the context of a procurement process. A software simulator of a procurement market was also implemented, on which tests were run in order to prove the effectiveness of the proposed strategy.

The remainder of the paper is structured as follows. Section 2 proposes a review of the literature and discusses the motivation of the work. Section 3 introduces the proposed idea and delves into technical details of the procurement auction mechanism. Section 4 discusses the perspective of providers in a procurement market and introduces an adaptive strategy to be used by the provider when participating in procurement auctions. In Section 5 simulation results are presented and discussed. Finally, the work is concluded in Section 6.

2 Background and Rationale

Amazon currently leases its unused capacity by adopting an auction-inspired price strategy that lets the customers acquire resources for a price which is lower than the

• *The authors are with the Department of Electrical, Electronic and Computer Engineering, University of Catania, Catania 95125, Italy. E-mail: {paolo.bonacquisto, orazio.tomarchio}@dieei.unict.it, dimodica@unict.it, peppepetra@yahoo.it.*

Manuscript received 12 June 2014; revised 7 Oct. 2014; accepted 25 Oct. 2014. Date of publication 10 Nov. 2014; date of current version 4 Sept. 2015. Recommended for acceptance by B. He and B. Veeravalli. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TCC.2014.2369435

standard one.[1] As far as we know, the Amazon's is the only commercial example of dynamic price strategy that is alternative to the fixed-price. If on the one hand it is true that customers benefit from low prices, on the other one the proprietary mechanism by which the virtual machines' price fluctuates has never been disclosed. The pricing model is still unclear and is not proved to be resistant to potential malicious behaviors of customers (dishonest customers can abuse the system and obtain short-term advantages by bidding large maximum price bid while being charged only at the lower spot price [1]). Furthermore in [2] and [3] authors prove that the Amazon's Spot Price is not market driven, rather it is typically generated as a random value near to the hidden reserve price within a tight price interval. The obvious consideration is that a provider itself, being an interested party, should not be the guarantor of the "fairness" of the price determination.

The literature is full of works addressing market-based mechanisms of resource allocation. Many IT researchers work on the application of auction mechanisms to the problem of computing resources' optimal allocation. Most of them devise efficient strategies to allocate computing resources in both grid and cloud contexts [4], [5], [6]. In [7] authors propose GridEcon, a marketplace of computing resources where prices are determined using an exchange market. In this exchange, the providers and customers both express their ask and bid prices and matchings are granted by the proposed allocation mechanism. In [8] authors discuss several strategies that cloud providers should adopt in order to reach high performance and overcome most of drawbacks of auctions, such as high overheads and high latency using techniques like overbooking and Flexible Advanced Reservations. They propose several bidding functions but each one takes into account just one parameter among those monitored by a cloud provider.

Stemming from the consideration that in cloud contexts customers always need bundles of resources rather than a single resource, many believe that combinatorial auctions are the most appropriate mechanism for allocating resources in the cloud [9]. In [1] authors propose a suite of computationally efficient and truthful auction-style pricing mechanisms, which enable customers to fairly compete for resources and cloud providers to increase their overall revenue. In [10] the scenario of multiple resource procurement in the realm of cloud computing is addressed: the proposed idea is to pre-process the customer requests, analyze the auction and declare a set of vendors bidding for the auction as winners based on the Combinatorial Auction Branch on Bids (CABOB) model. In [11] a combinatorial, double-auction, resource allocation model is instead proposed. The efficiency of the proposed economic model is proved in the paper, but to our advice that idea is not technically viable since a bundle allocated to a customer is composed of computing resources offered by different providers, thus forcing the customer to deploy their application on a geographically distributed cluster of machines. Many other works focus on solving the problem of optimal sale of resources in combinatorial auctions, which is known to be NP-hard.

1. https://aws.amazon.com/ec2/purchasing-options/spot-instances/

The auction mechanisms proposed so far in the literature put the provider in a privileged position in the market: computing resources are seen as scarce and precious goods, whose allocation is carried out through competitions run among customers. Customers have to call higher prices in order to have chances to get the resources. We argue that this viewpoint must be overturned. Spare resources are resources which providers do not manage to allocate through direct-sell. From the provider's perspective spare resources must be regarded as perishable goods that need to be sold within a certain time frame otherwise they get wasted. Not selling a virtual machine in a given slot time means a profit loss to the provider, who anyway is spending money to keep the physical machines up and running. We then look at the trade of computing resources from a new perspective, in which providers, in the aim of maximizing their data center's occupancy rate, may be willing to attract customers by lowering the offer price. On their turn, customers may get what they need, at the time they need it, at a price which is lower than the standard price at which they usually buy.

Let us then assume a provider's overall computing capacity is split into two different pools: the *direct-sell pool*, containing resources which the provider is confident to sell through the direct-sell mechanism, and the *spare pool*, which represents the portion of computing capacity that on average remains unsold. From now on we will put the focus on the spare pool, denoted by S. On that pool, we will refer to C_o as the overall computing capacity. This value does not change in time as long as the provider does not decide to modify the spare pool's size. We then define the *committed capacity* $C_c(t)$ as the computing capacity committed to serve customers' requests at time t. Of course, this variable can assume values in the range $[0..C_o]$. Alternatively, we will refer to $R_c(t) = 1 - C_c(t)$ as the *residual capacity* (which is the free or unassigned capacity of the pool). Let $U(t)$ be the utilization rate, defined as

$$U(t) = \frac{C_c(t)}{C_o}. \tag{1}$$

The lower U, the higher the profit loss for the provider. In the aim of maximizing the utilization rate (minimizing the residual capacity) providers need to adopt new selling strategies. We argue that providers, in order to avoid "wasting" computing capacity, are willing to give up a portion of profit per computing unit (same as it happens for sale of perishable goods).

We advocate that the market model best fitting the just described perspective is the one which provides for the sale of computing resources through **procurement auctions**. Procurement auctions [12] (also called *reverse auctions*) reverse the roles of sellers and buyers, in the sense that the bidders are those who have interest in selling a good (the providers), and therefore the competition for acquiring the right-to-sell the good is run among providers.

Smeltzer and Carr [13] outline potential advantages and drawbacks of adopting reverse auctions for goods allocation. Further, they point out the appropriate conditions which must apply for the reverse auction to be effective and convenient to both goods' suppliers and buyers. The most

important are a clear specification of the commodity to be allocated and the fragmentation of the market. As for the first point, in the cloud community there is a common understanding of computing capacity's technical specification: information such as core numbers, CPU speed, RAM size, etc. are the only data needed to clearly and unequivocally state the product specification. In respect to the second point, we are proposing an open market of computing capacity where customers may look for spare resources to buy at lower prices, and that will naturally attract many providers interested in allocating spare resources and increasing their market share. Clear advantages for providers are that they may find customers in one single big market with no extra effort and that the market gives them the chance to maximize the occupancy rate of their data centers; on the other side, customers do not have to search for providers' offers (they just post their computing demand to the market) and will get the requested computing capacity at a lower, supply-demand regulated price.

3 A Procurement Market of Virtual Machines

Our proposal addresses the need of a *market exchange* that helps consumers and providers to trade cloud resources. Basically, a market exchange [5], [7] provides a shared infrastructure designed to support the secure trading of resources. On the provider's end, it provides services to post offers to the market and to bargain for their price; on the customer side, services are provided to discover offers and to negotiate for acquiring them. In the market a crucial role is played by the brokering service, which is in charge of enforcing the negotiation protocols (one-to-one, auction) that best suite the supply-demand rate. Besides that, in order for a market exchange to be perceived secure and trustable by participants, other non-functional services must be offered like the authentication of participants, the safe routing of messages among participants, the logging and auditing of any operation involved in the trading of resources.

Among the numerous issues pertaining the design and implementation of a market exchange of cloud resources, we specifically address the one related to the choice of a protocol to negotiate the price of provider's residual computing capacity. For the reasons discussed in Section 2, we believe the *procurement auction* mechanism is the best suiting. In this section we discuss about the trading of computing resources through a procurement-based negotiation scheme.

The basic procurement mechanism is depicted in Fig. 1. In a procurement auction customers communicate their computing demand to the market. A *broker* will take care of demands. For each specific demand, the broker (also called auctioneer) will run a public auction to which any provider (bidder) can participate and compete for acquiring the right to serve the demand. The winning provider (who offered the lowest price) will eventually have to serve the customer's demand. Being the auctions open to the participation of multiple providers, the price competition is granted. For a given demand, the determination of the final price is driven only by the evaluation that each

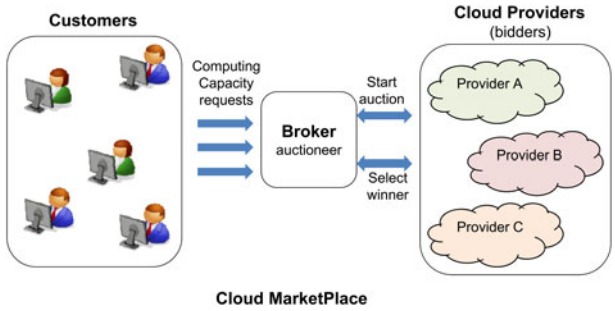


Fig. 1. The procurement auction scenario.

provider has on the demand to be served. Advantages for customers are clear: they will get their demand served at the lowest price. Further, they will no longer have the burden to search for providers, as providers gather autonomously in the market.

Focus in this paper is on two different types of procurement auctions: the **English Reverse** (ER) and the **Second Price Sealed Bid** (SPSB) [14]. The ER is a multi-round auction. According to the CFP, an initial price is called from which discounting bids (offers) are expected. The competition among providers is played in many rounds. While the duration of each round is fixed, the number of rounds that are needed to solve the competition depends on how tough the competition is. In each round the participating bidders post their offer. At the end of a round all bidders are notified with the winning offer's price for which further discounts are expected to be called in the next round. If in a given round no discounting offer is proposed with respect to the winning offer of the previous round, the latter will be appointed the auction's winner. This type of auction allows bidders to gather information of each other's evaluation of the good. The SPSB is, instead, a single round auction. All bidders have the chance to bid just once before the auction is cleared. When bidders receive the CFP, they check the initial price and decide to either bid or not to bid. After either a time-out expires or all participants have posted their bid, the broker clears the auction and allocates the "demand" to the second best bidder. The peculiarity of this auction is that bidders are not aware of each other's offer and that the winner will be acknowledged a price which is higher than their own bid, thus increasing their overall utility (see further on in the paper for a precise definition of utility).

As a basic market rule, a provider is admitted to participate in any CFP they like, with the obligation that if they win the auction, they are committed to serve the customer's demand, otherwise they will incur a penalty. The commitment rule obliges the provider to reserve resources for the CFPs they take part in. Those resources will remain "locked" until the auction is cleared. In a few words, the participation of a provider to an auction is subjected to the provider's availability of the amount of resources for which the auction has been called. If we consider that, on a statistical base, a provider will unlikely win the 100 percent of the auctions they take part in, there will always be an amount of resources which will remain not utilized because they are locked (i.e., awaiting for the respective auction to be cleared). In this work we will address that problem and propose to overcome it by applying the

mechanism of **resource overbooking** [15] to the procurement market. In particular, our objective is to investigate on the impact of such a mechanism on the utilization level of data centers. Section 5 provides interesting feedbacks on that study. The overbooking lets providers compete for more customers' demands than they are able to eventually serve. Providers may participate in a given auction even though, at the time of joining the auction, they have no resource available to serve the demand which is object of that auction. But, what happens if a provider is out of resources at the time they are appointed the auction's winner? If we want this mechanism to be fair and transparent to customers, some market rules must be enforced that grant the customer's right to receive the service on the one hand, and penalize providers that overuse the overbooking on the other one. One of the objective of this work is to define the "reassignment" policy, i.e., the actions to be taken to grant the delivery of the service to the customer in the case that the provider(s) appointed as winner(s) can not provide it. This policy must state a) who among the remaining participants is assigned the right-to-serve, b) who is in charge of paying the penalty and c) how much is due. In this respect some proposals can be found in the literature. In [8] authors discuss some penalty functions which may be used in what they call "computational economies". Those functions may be classified into constant and dynamic ones. The former suggests the application of a constant penalty for those who win the auction but are not able to serve the demand (*defaulting providers*), the latter apply different penalties according to the impact of violation made by each party.

We designed a reassignment scheme which tries to reassign the right-to-serve among those who participate in the auction and applies dynamic penalties. The general rule is that if a winning provider is defaulting, the right-to-serve is passed to the next best-offering provider. This process may iterate until a provider is found which is able to serve the demand. In the case that all participants are defaulting, the CFP is re-issued. The general penalty rule is that every defaulting provider in the chain will have to pay a fee that is proportional to the difference between the final price (that called by the provider who will eventually serve the demand) and the price they had called out. The proportionality is implemented through the concept of price "distance" from the final price. Let x_{final} be that price, and x_{winner} the price called out by the provider who won the auction. Also, let x_i be the price called by the generic provider P_i in the chain of defaulting providers. Of course, $x_{winner} <= x_i < x_{final}$. We define the price distance of a given price x_i as $d_i = x_{final} - x_i$. According to this definition, d_{winner} is the highest among price distances, and will represent the *overall penalty* to be proportionally shared among defaulting providers. Also, let us define the penalty coefficient as

$$c_i = \frac{d_i}{\sum_{i=1}^{n} d_i}. \tag{2}$$

The reader may notice that the summation of coefficients is equal to 1. Finally, the portion of penalty each defaulting provider will have to pay is calculated as

$$p_i = c_i \times d_{winner}. \tag{3}$$

In the end, the provider who will serve the demand is also awarded the amount deriving from the overall penalty (d_{winner}), but the price for serving the demand (which is due by the customer) will be that called by the bidder who was appointed the auction's winner (x_{winner}).

Let us make an explanatory example. Suppose provider P_1 wins the auction calling a price $x_1 = 2$, but is defaulting. Then, the next best bidder P_2 will be selected, who called the price $x_2 = 5$. Again, suppose P_2 is not able to honor. P_3 who called price $x_3 = 7$ is then selected, and is finally able to serve the request. The overall due penalty is $d_{winner} = x_{final} - x_{winner} = 5$, to be shared among defaulting providers P_1 and P_2 in the following way:

$$p_1 = c_1 \times d_{winner} = \frac{d_1}{d_1+d_2} \times d_1 = 3.125$$
$$p_2 = c_2 \times d_{winner} = \frac{d_2}{d_1+d_2} \times d_1 = 1.875.$$

So, the serving provider P_3 is granted a price of 7, obtained by summing up the penalty $p_1 = 3.125$, the penalty $p_2 = 1.875$ and the first winning bid $x_1 = 2$ (which is owed by the customer). In conclusion, the described mechanism a) penalizes more the bidders that behaved more aggressively during the auction, b) preserves the customer by granting them the auction's official winning price and c) awards the bidder who eventually serves the customer's demand.

4 The Provider's Perspective in Procurement Auctions

If we better analyze the context of cloud auctions, a computing resource may be seen as a good whose actual value (price) is common to all providers. In fact, though for computing resources we can not yet speak of conventional "market prices", all providers in their regular sales adopt well known, leveled prices. We can then conclude the actual value of such kind resources is somewhat common to providers. In the context of a procurement auction of computing resources, the estimate E_{pi} of the *i*th provider on a given good may differ from the the estimate E_{pj} of the *j*th provider on the base of the diverse needs each provider may have while pursuing their own business objective.

Generally speaking, the primary objective of an auction's participant is to maximize what is commonly known as *utility*. Given a resource to be allocated through an auction sale, the participant's utility for that resource is defined as the difference between the winning bid price and the participant's evaluation on the resource [1]. In the procurement market depicted in Section 3 providers participate to many auctions, even simultaneously. In such a context, the maximization of the utility should not be the strategy to pursue, rather, should be considered as part of a balanced strategy which takes also into account other factors contributing to the profit maximization.

Recalling the concept of datacenter's utilization ($U(t)$) introduced in Section 2, there is a monetary cost associated to idle computing resources not allocated to any customer's demand. The overall investment made by providers of computing resources in terms of both CAPEX and OPEX has been investigated and outlined in an interesting report

[16]. In that work, costs for planning, developing and maintaining a data center are discussed. Beyond the useful hints provided to designers of data centers, that work introduces a *cost model* which the owners of data centers are encouraged to use and take into consideration in their medium and long term business objectives. Drawing inspiration from that work, in Appendix A we make a tentative to spread the total cost of ownership (TCO) of a sample data center to all the potentially runnable computing units,[2] so as to obtain an indication on the average cost per computing unit sustained by the provider. This datum is useful to the provider to better understand what is the cost they incur by not allocating a given virtual machine in a given slot time.

Taking into account this cost in the context of procurement auctions, a provider's strategy which tries to maximize just the *utility* may turn to be unsuccessful. The more a provider tends to increment utility in a given auction, the higher the likelihood to lose the auction. In its turn, the higher the number of lost auctions the larger the size of "wasted" computing power, which again, generates costs but does not generate revenue. On the contrary, a strategy which tries to maximize only the datacenter's utilization (no matter the sale price) might not produce the revenue required to compensate the cost. Also, in the aim of maximizing the utilization, a good strategy might seem to point on allocating virtual machines to customers requesting them for long periods (let us call them long-termed virtual machines). Unfortunately, committing virtual machines for longer periods on the one end increments the datacenter's utilization in the long run, but on the other one prevents them from being traded in more profitable auctions. Finally, given an amount of computing power, it may be used to allocate either many short-sized virtual machines or a few large-sized virtual machines: depending on the market dynamics and on the specific needs of the provider, one strategy may turn to be more profitable than another one.

4.1 An Adaptive Strategy

The desirable situation for a provider is to have the datacenter 100 percent occupied with long-termed and large-sized virtual machines sold at the highest possible price. In practice, the unpredictability of the market dynamics (demand's arrival rate, strategies of other providers participating to the auctions) makes this target unachievable. What we are going to present here, thus, is not the definition of an optimum strategy for the provider. Rather, we propose a tool, in the form of a tunable strategy, which the provider may use to pursue their business objective in a procurement auction market. On the basis of some parameters preliminarily tuned by the provider, such a strategy will suggest the right actions to undertake when they compete for the acquisition of a good in a procurement auction (e.g., whether to participate in a given auction, to bid in a given round, not to bid, which price to offer).

2. By computing unit we mean the minimum unit of computing power which the data center can supply. In the rest of the paper, it will correspond to the minimum size of virtual machine that a provider decides to offer.

We have identified a non-exhaustive list of factors which may strongly influence the strategy of a provider in a procurement auction.

- The duration of the customer task (demand) to be served (L). The longer the task, the higher the profitability for the provider, since the required capacity will be committed for a longer time. This parameter directly affects the datacenter's utilization rate. A provider might prefer to participate in auctions where long tasks are traded. But on the other hand, they are aware that serving a longer task means committing capacity for longer periods. If the provider happens to win the competition for that task by offering a very low price, for the whole duration of the task it has missed the chance to use that capacity to compete for more profitable (even shorter) tasks.
- The type of virtual machine instance required to serve the customer task (T). Of course, the profitability of a task is directly proportional to the task's requirements in terms of amount of computing capacity per hour, so providers may be motivated in pointing on auctions calling for a higher capacity/hour. But depending on the actual utilization level of both each single host and the whole datacenter, it might not be possible to serve more tasks requiring high capacity virtual machines.
- The cost sustained to run a virtual machine. As showed in Appendix A this cost is specific to each provider and changes in time, as it depends on the current datacenter's utilization level. When participating in an auction, providers may act more or less aggressive in bidding a price. Needless to say that a provider should never call for a task a price which is lower than the cost to run it. This is why we are going to refer to that cost as the *lower bound price* (L_b).
- The reference price of the virtual machine in the fixed-price market (P_f). From the provider's perspective, serving a task is profitable as long as its current called price (P_a) does not get too far from the average price at which the virtual machine is traded in standard fixed-price markets. The larger the gap, the less aggressive the strategy.

Basically, a strategy is responsible for suggesting the provider how to behave in a procurement auction market. So let us suppose that a CFP has been issued in the procurement market. The steps taken to decide the participation in the auction are shown in Fig. 2.

First, the availability of the resources required to serve the demand is checked. Availability of resources must be continuously checked as it dynamically changes in time. In case real resources are not available and the overbooking is enabled, the overbooking pool is checked. In the end, if resources are found either in the regular or in the overbooking pool, the strategy will advice the participation.

Suppose now that the provider has joined the auction. At the beginning of each auction's round the broker calls out the price for which descending offers are expected. The strategy is again called into play to suggest the price of the bid. Fig. 3 shows the sequence of the enforced actions.

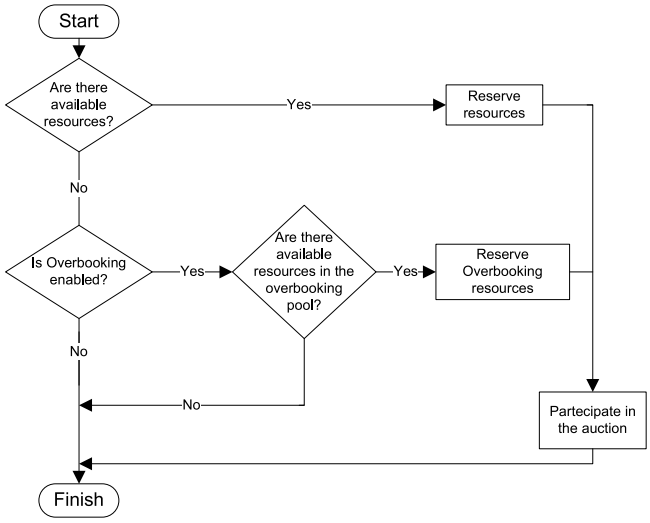


Fig. 2. Strategy actions to join an auction.

Basically, the bid price is computed by a formula which is going to be described later on. Next, the current lower bound price is evaluated (remind that the lower bound price is function of the actual level of utilization of the datacenter). If the computed bid price to call is higher than the lower bound price then the bid is called out. Otherwise the provider will abandon the auction and the reserved resources are released.

In the process of computing the next bid to call, a strategy may be more or less "aggressive", i.e., may propose higher or lower discounts. The strategy is adaptive, in the sense that is able to adapt its aggressiveness according to the earlier discussed factors. Providers are also allowed to tune the strategy's aggressiveness by adequately weighting these factors according to their own business needs. Recalling a formula presented in [17], the adaptive strategy will suggest the next bid to call as

$$bid = \frac{n-1}{n-(1-\alpha)} \times lastWinningBid, \tag{4}$$

where n is the number of bidders participating in the auction and $lastWinningBid$ is the price offered by the bid that won the last round. In case of single-round auctions, $lastWinningBid$ will be the auction's starting price. The parameter α is calculated as follows:

$$\alpha = w_1 \times \frac{P_a}{P_f} + w_2 \times \frac{T_{vm}}{T_{max}} + w_3 \times \frac{L}{L_{max}}. \tag{5}$$

Each factor is weighted by coefficients (w_1,w_2,w_3) whose summation is 1. Coefficients are tunable by providers according to their need. A preliminary version of the formula in 5 was first introduced in our previous work [18]. Here we briefly recall the meaning of its terms. $\frac{P_a}{P_f}$ is the ratio between the resource's initial price in the auction and the corresponding price in the standard fixed-price market. $\frac{T_{vm}}{T_{max}}$ is the ratio between the computing power demanded by the request and the computing power of the most powerful available resource. Finally, $\frac{L}{L_{max}}$ represents the ratio between the time period for which the computing resource is requested and the maximum time period for which a resource can be requested.[3]

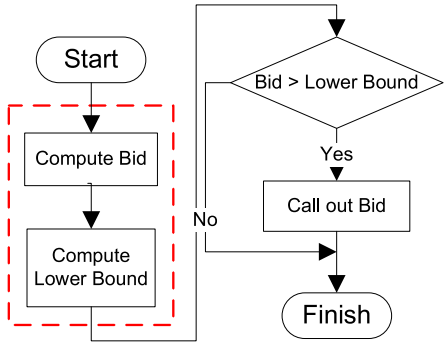


Fig. 3. Strategy actions to compute the next bid to call.

By tuning the coefficients, providers are able to customize their strategy to privilege one aspect over another one. For instance, providers who needs to grow their datacenter's *utilization rate* will increase the w_1 coefficient, while those who target the *utility* will have to unbalance their strategy on the w_2. Of course, combinations of coefficients give rise to heterogeneous strategies.

In order to implement the overbooking mechanism, it is necessary to establish the size of the pool of resources that must be devoted to the overbooking. Of course, that size can not be as large as the provider desires, as a very large size may cause the provider to be frequently defaulting. Our technique consists in monitoring the history of the auctions joined by the provider but in which the provider did not turn to be the winner. In particular, under observation is the rate of MIPS which got reserved in the last 500 auctions and that eventually did not get commit to a demand. Such a value is defined as

$$mips_r = \frac{mips_{reserved}(500) - mips_{committed}(500)}{mips_{reserved}(500)}. \tag{6}$$

This rate determines the percentage of resources that may be considered for the overbooking. Let us make a practical example. For the sake of simplicity let us suppose that the observation window for the auction's participation is 10 (instead of 500). The history says that

1) traded MIPS = 24000; the auction was won,
2) traded MIPS = 24000; the auction was lost,
3) traded MIPS = 24000; the auction was lost,
4) traded MIPS = 40000; the auction was won,
5) traded MIPS = 24000; the auction was won,
6) traded MIPS = 24000; the auction was lost,
7) traded MIPS = 37000; the auction was lost,
8) traded MIPS = 24000; the auction was lost,
9) traded MIPS = 24000; the auction was lost,
10) traded MIPS = 12000; the auction was won.

The total committed MIPS are 100.000 (auctions 1,4,5,10); the total reserved MIPS are 257.000. This brings the value

3. In real situations there is obviously no bound to the time period for which a resource can be requested. As remarked before, our proposal is targeted to accommodate short-termed requests, so for the purpose of our work we are going to consider tasks lasting no longer than 24 hours.

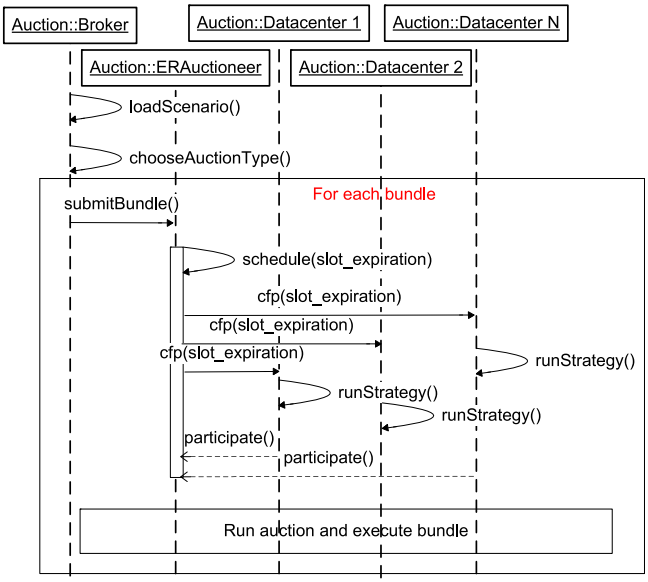


Fig. 4. Auction preparation.

$$mips_r = \frac{257.000 - 100.000}{257.000} = 0.6.$$

So the overbooking pool's size will be 0.6 times the size of the overall resource pool.

5 Implementation and Experiment Results

To assess the viability of the proposed approach a simulator of a procurement market has been implemented. The objective was to define a tool capable of simulating a) the procurement auction processes, b) the behavior of the participating providers and c) the arrival of customers' demand for VMs. Tests conducted on simulator aimed at evaluating the responsiveness of the providers' strategies to the declared business objective.

5.1 Simulator's Architectural Details

The Cloudsim tool [19] has been used to implement the simulation environment where procurement auctions are run. With respect to the existing Cloudsim's components, a new component called *Auctioneer* has been introduced. It cooperates with the Cloudsim's *Broker* to manage auctions for cloud applications. The Cloudsim's *Datacenter* component has been extended to add functions for a) reserving the resources needed to serve a request, b) estimating bids and c) implementing the overbooking mechanism. Also, the *AdaptiveStrategy* class has been implemented which models the strategy providers may adopt. Finally, the Cloudsim's *Cloudlet* component, which represents the task submitted by a customer to Cloudsim, has been extended to include features such as the duration of the requested service, the submission time of the demand, the type of the requested VM, and all the necessary information needed to analyze, for statistic purpose, the data extracted from the simulator.

Fig. 4 depicts the sequence of interactions, occurring among the simulator components, to set up an auction. In the first step the Broker loads the file of the simulation scenario containing the list of bundle requests (see further for

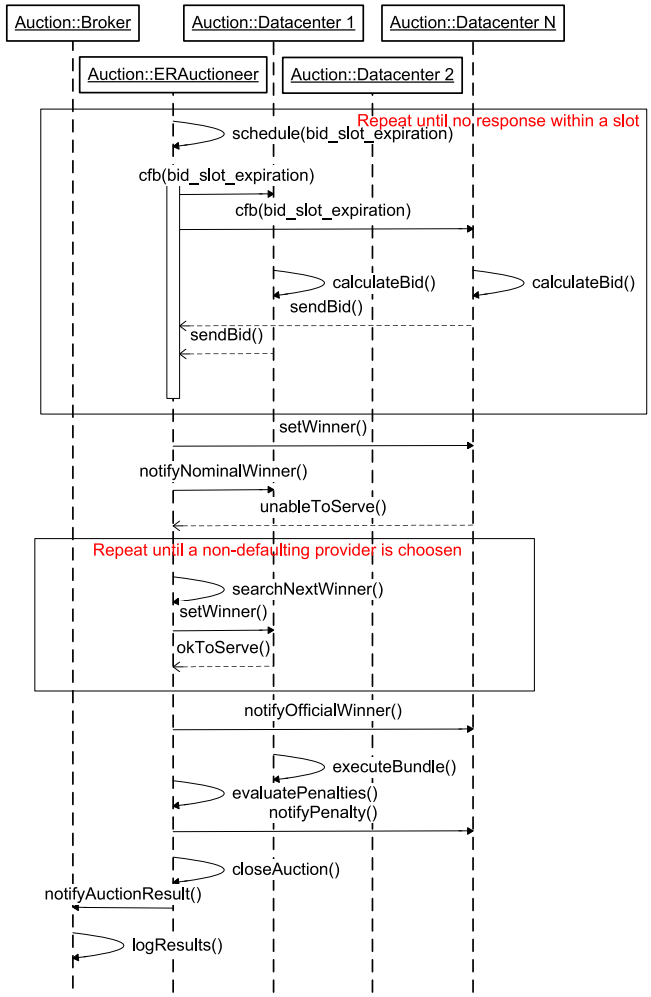


Fig. 5. Auction run.

the definition of bundle). After choosing the type of auction to be run, bundles are iteratively submitted to the Auctioneer. For a given bundle the Auctioneer broadcasts a CFP to all interested Datacenters. The CFP has a slot expiration parameter indicating the deadline by which all Datacenters interested in acquiring that bundle will have to apply. Each Datacenter may or may not decide to participate according to their strategy (in the diagram, Datacenter n.2 decides not to participate). The auction is started upon expiry of the CFP. Fig. 5 shows the simulation steps in the specific case of an ER auction. First, the Auctioneer sets up the duration of the bid slot, i.e., the slot within which Datacenters are allowed to communicate their bids. Then, according to the ER protocol, it will iteratively send out call-for-bids (*cfb*). The Auctioneer will stop calling cfb when no bids arrive in a given bid slot (i.e., no Datacenter is calling a lower price). The winning Datacenter is notified and will have to communicate if it is able to serve the request. In the case it is not (*unableToServe()* message), the Auctioneer will iteratively scroll through the list of bidders (in price ascending order) until a non-defaulting Datacenters is found. That Datacenter is appointed as the official winner and thus gets the right to execute the Bundle, while defaulting Datacenters are notified the due penalties. Finally, the auction is closed and the results are reported back to the Broker.

TABLE 1
Amazon's Virtual Machines types

Type	Name	Architecture	N vCPU	N Compute Units	RAM	Storage	Bandwidth	Workload
General purpose	m1.small	32/64-bit	1	1	1.7 GB	160 GB	Low	0.6%
	m1.medium	32/64-bit	1	2	3.75 GB	410 GB	Moderate	0.3%
	m1.large	64-bit	2	4	7.5 GB	820 GB	Moderate	56%
	m1.xlarge	64-bit	4	8	15 GB	1.6TB	High	7%
	m3.xlarge	64-bit	4	13	15 GB	0	Moderate	0.1%
Compute Optimized	c1.medium	32/64-bit	2	5	1.7 GB	350 GB	Moderate	28.9%
Memory Optimized	m2.xlarge	64-bit	2	6.5	17.1 GB	420 GB	Moderate	7.1%

5.2 Characterization of the Customers Demand

In a previous work [20] we presented the result of simulations in which the customer's demand was characterized by single-machine requests. The assumption was that customers were allowed to submit requests for no more than a single virtual machine. In this paper that assumption is relaxed as we introduce the possibility for customers to request for *bundles* of virtual machine. In real cloud contexts, in fact, customers usually need to acquire bundles of virtual machines to satisfy their computing needs.

To characterize the customers' demand for computing capacity, the same pattern of requests reported in the Google's cluster data trace [21] has been reproduced. The trace file stores usage information collected during a 29-day period in the month of May 2011 in one of Google's production cluster cell composed of about 12 K machines. In particular, we have reproduced the same workload of the Google's trace (in terms of jobs and tasks) and used it to simulate the customer's demand in the procurement market. The reason behind this choice is that the Google cluster's workload is characterized by machine requests which range from a few minutes to one-day usage. We believe such workload characterization may be a good candidate to model the customers' demand for short-term VMs, which providers may be willing to serve with their residual capacity. Actually, the customers' demand to submit to the procurement market was obtained by filtering out all the Google workload's micro requests falling behind the hour usage.

The types of requests appearing in the trace have been mapped onto their equivalent Amazon's virtual machine types. In Table 1 the characteristics of those machines are reported along with the workload percentage of each VM type with respect to the overall daily workload. The considered Google workload is made of uncorrelated job requests. We had to work on it in order to simulate a pattern of bundle requests. The idea was to generate bundles by putting together job requests. So, a new bundle request is generated by selecting two or more jobs from the Google workload. For simulation purposes, we considered only the job requests arrived in the first 24 hours (about 9.600 jobs). From that pool we generated bundles made up of an average of three jobs, with a minimum of two and a maximum of five jobs: in the end 31,200 bundles were generated. The bundle's generation function works this way. First a random number is generated in the range [2..5]; then the generator looks up in the trace and selects that number of consecutive jobs to form the bundle. We imposed as a constraint that in the same bundle there may not be more than two jobs of the same type. The most demanding bundle (in terms of requested computing capacity) was then the one requesting two m3.xlarge instances, two m1.xlarge instances and one m2.xlarge instance. In order to be able to compare bundles in a quantitative scale, we introduced the concept of bundle's "weight". The weight is defined by means of the reference prices of VMs at which they are traded in the standard fixed-price markets. So, for instance, the just mentioned most demanding bundle (which currently has a price reference of 2.37 $/h) is assigned the weight 1. All other bundles have a weight which is given by their respective reference price divided by 2.37. On a total of 31 thousand generated bundles, 20 thousand bundles (70 percent) have a weight falling below 0.5, while the remaining 11 thousands (30 percent) go over 0.5. Finally, bundles have a duration request which is uniformly distributed in the range [1h..6h].

5.3 Datacenters' Features

To test the adaptive strategy, we created a set of 24 Datacenters, of which 22 adopt the proposed adaptive strategy and 2 adopt a *Random strategy*. The latter make bids by using as α parameter a random value in the [0..1] range (they have no specific business objective to pursue). Each Datacenter is provided with 400 physical machines (hosts): 100 equipped with 64 cores, 100 equipped with 128 cores, 100 hosts equipped with 256 cores and 100 hosts equipped with 512 cores, for an overall computing power of 93 K cores. Every host is also equipped with 512 GB of RAM and has storage for 100 TB. Features of Datacenters have been chosen in such a way that all the Datacenters participating in the procurement market will be able to serve the earlier discussed workload.

The 22 Datacenters have been split into two sets, of which only one make use of the overbooking. The coefficients characterizing the α parameter are shown in Table 2. As the reader may notice, strategies were expressly split in *unbalanced*, for which Datacenters point on either one or two factors, and *balanced*, for which all the weights are assigned the same value. The objective of the simulation is to show that strategies actually guide Datacenters in the choice of the tasks to compete for.

In the tests, the spare resources which providers will use to compete in auctions will be the 20 percent of their overall resources; the remaining 80 percent is sold in the traditional fixed-price market. In the context of the simulations we will interchangeably use the terms Providers and Datacenters.

5.4 Experiments

We designed three simulation scenarios. The common part of each scenario is the submitted demand, which is

TABLE 2
Coefficient Setting for the Datacenters' Strategies

Datacenter ID	Strategy	w_1	w_2	w_3	Overbooking	Datacenter ID	Strategy	w_1	w_2	w_3	Overbooking
DC1	Adaptive	0.8	0.1	0.1	No	DC13	Adaptive	0.33	0.33	0.33	No
DC2	Adaptive	0.8	0.1	0.1	Yes	DC14	Adaptive	0.33	0.33	0.33	Yes
DC3	Adaptive	0.1	0.8	0.1	No	DC15	Adaptive	0.33	0.33	0.33	No
DC4	Adaptive	0.1	0.8	0.1	Yes	DC16	Adaptive	0.33	0.33	0.33	Yes
DC5	Adaptive	0.1	0.1	0.8	No	DC17	Adaptive	0.33	0.33	0.33	No
DC6	Adaptive	0.1	0.1	0.8	Yes	DC18	Adaptive	0.33	0.33	0.33	Yes
DC7	Adaptive	0.45	0.45	0.1	No	DC19	Adaptive	0.33	0.33	0.33	No
DC8	Adaptive	0.45	0.45	0.1	Yes	DC20	Adaptive	0.33	0.33	0.33	Yes
DC9	Adaptive	0.45	0.1	0.45	No	DC21	Adaptive	0.33	0.33	0.33	No
DC10	Adaptive	0.45	0.1	0.45	Yes	DC22	Adaptive	0.33	0.33	0.33	Yes
DC11	Adaptive	0.1	0.45	0.45	No	DC23	Random				No
DC12	Adaptive	0.1	0.45	0.45	Yes	DC24	Random				Yes

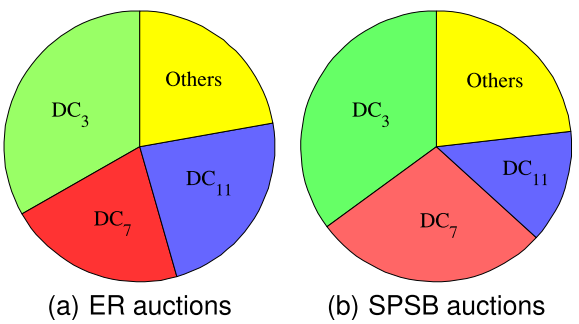
represented by the workload discussed in Section 5.2. In the first two scenarios the overall computing capacity (the market's "offer") is represented by the 24 Datacenters described in Section 5.3; the difference is that in the first one the ER mechanism is used, while the SPSB is opted in the second one. In the third scenario the ER mechanism is tested against three different market's offer capacity: 14, 24 and 34 Datacenters respectively. Let us first analyze the results obtained from testing the first two simulation scenarios.

Data gathered from the simulations demonstrate that each provider, by properly sizing the coefficients of their strategy, is able to achieve the pre-fixed objective. In Fig. 6 we depict the percentage of bundles having a weight ≥ 0.5 won by non-overbooker Datacenters in auctions of type ER and SPSB respectively. Fig. 6a shows that in the case of ER auctions the Datacenter getting the highest number of heavy weighted bundles is DC_3; in Table 2 the reader may notice that its strategy strongly privileges the acquisition of that kind of bundles. Good results are also obtained by Datacenters DC_7 and DC_{11}, whose strategy targets two factors, one of which is the bundle type. The rest of eight Datacenters get the last slice of the pie. Comparable results are obtained in SPSB auctions (see Fig. 6b).

In Fig. 7 we depict the percentage of 6h long bundles won by non-overbooker Datacenters in ER and SPSB auctions respectively. Results for ER auctions are depicted in Fig. 7a: DC_5 has a strategy set on the duration (see Table 2) and gets the largest slice of bundles. DC_9 and DC_{11}, which have a strategy partially set on the duration, manage to win a discrete number of long-termed bundles. The remaining eight Datacenters get the rest. Again, similar results are obtained in SPSB auctions (see Fig. 7b).

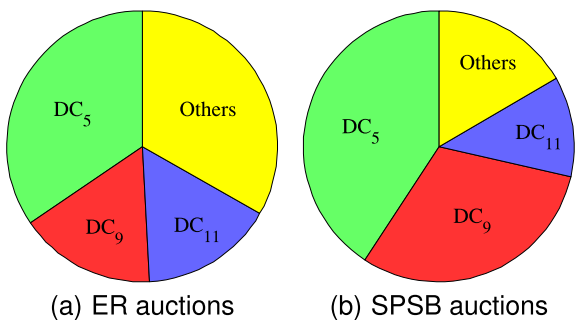
In Fig. 8 we present the result regarding the *utilization* level of datacenters, comparing those which use overbooking and those which do not. In Fig. 8a it is evident how in ER auctions the non-overbookers on average reach at most the 90 percent while overbookers get up to a full utilization; in SPSB auctions (Fig. 8b), instead, there is almost no difference between overbookers and non-overbookers in terms of utilization level. What happens is that in short-running auctions (like the single-round) overbooking is not effective at all. The overbooking allow providers to have lots of resources reserved to participate in many auctions, thus increasing the opportunity to win more bundles and increase their utilization. This mechanism is not fully exploitable in single-round auctions, where resources are reserved for very short periods (the duration of just one round, indeed) and so the provider may soon know whether those resources are to be released or not. In multi-round auctions there is a longer period of uncertainty during which the provider does not know whether to release or not release the resources, so the overbooking may be very helpful.

In Fig. 9 the values of *utility* reached by all Datacenters (overbookers and non-overbookers) are reported. In ER auctions (Fig. 9a) the overbookers on average get better values of utility than non-overbookers. Best utilities have been observed for the Datacenters DC_1 and DC_2, which have strategies set on the price ratio. Datacenters DC_7, DC_8, DC_9 and DC_{10} have good results as well. In SPSB auctions all Datacenters happen to have better utilities than those obtained in ER auctions. In single-round auctions (like the



(a) ER auctions (b) SPSB auctions

Fig. 6. Rate of bundles with weight ≥ 0.5 won by Datacenters.



(a) ER auctions (b) SPSB auctions

Fig. 7. Rate of bundles with duration of 6 hours won by Datacenters.

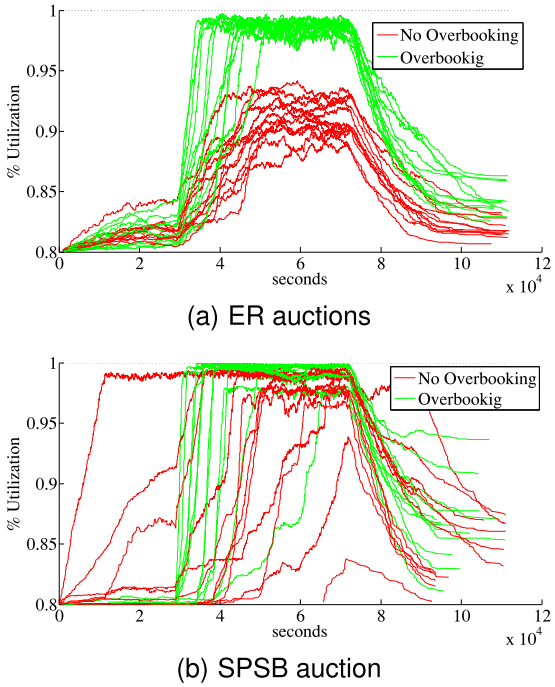


Fig. 8. Utilization level of datacenters.

SPSB) the fight for acquiring a bundle is not as tough as in multi-round auctions (like the ER). Further, the overbooking seem to bring no benefits in term of utility; in some cases, on equal strategies overbookers obtain even worse results than non-overbookers.

In Fig. 10 the datum concerning the penalties is shown. The reduced amount of penalties paid by Datacenters gives the idea of how well the overbooking mechanisms works. The pool size of resources to dedicate to the overbooking is elastic. At a given time t, it is a function of the rate of resources previously reserved to participate in auctions in which the Datacenter did not eventually get the bundle. By automatically sizing the pool to the auction participation's history, the number of times the Datacenter turns to be defaulting is very low.

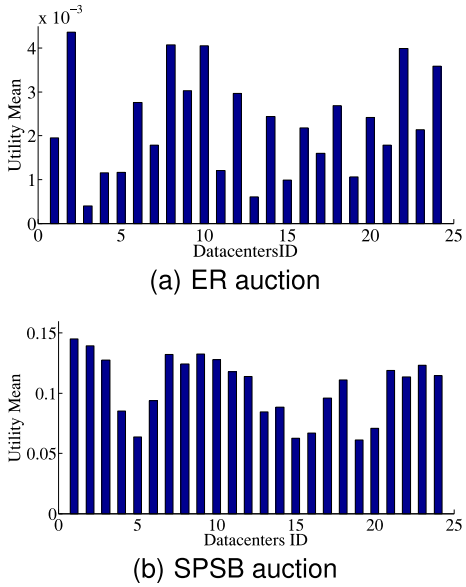


Fig. 9. Average utility of datacenters.

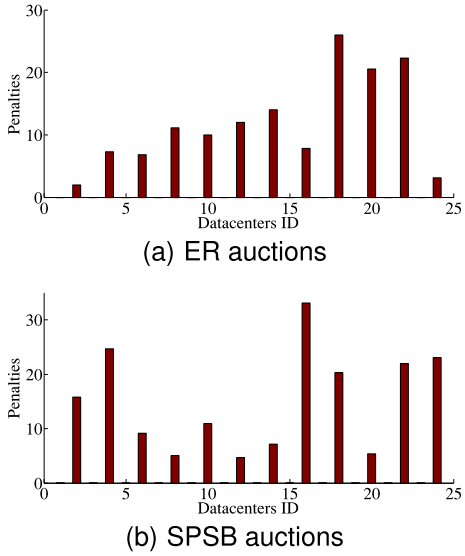


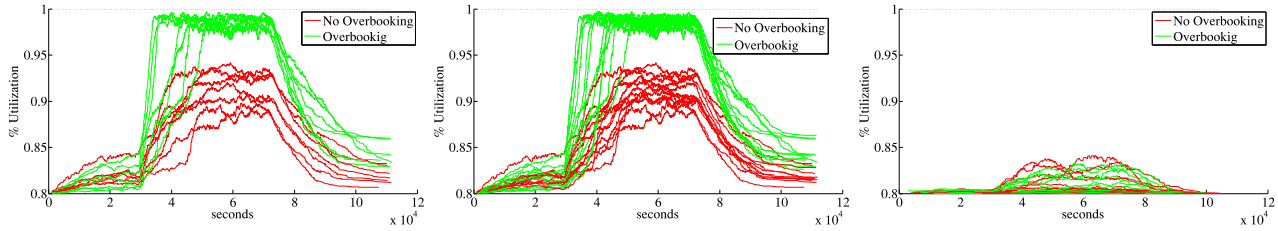
Fig. 10. Penalties incurred by datacenters.

We now report the result obtained in the third simulation scenario, where ER auctions are run on different market's offer capacity. In Fig. 11 the utilization datum is shown. The utilization observed in the case of 34 Datacenters is the worst. In fact, in this scenario the competition is tougher than the other cases, and on average each Datacenter happens to win a lower number of auctions. In the two cases of 14 and 24 Datacenters respectively, we observed almost no difference in respect to the utilization index. In Fig. 12 the datacenters utility is instead reported. The reader may notice that the higher the market's offer capacity, the worse the utility obtained by Datacenters (Fig. 12c). Again, a larger market's capacity triggers a tougher competition among Datacenters, and of course, in the battle for acquiring the demand, prices fall down.

To conclude, the actual objective of this work was not to define the optimum strategy for Datacenters. There may not be an optimum strategy in a dynamic context in which useful information (other providers' strategies, customer demand's pattern) can not be known in advance. The proposed strategy is an attempt to encode the business objective of providers into an automatic procedure which is able to act on behalf of the provider. Neither our aim was to compare the ER to the SPSB mechanism. Each mechanism has its own pros and cons, whose discussion is out of the scope of this work. But both have shown to be compatible with our view of a procurement auction market as a natural approach to the allocation of spare resources in an open market of cloud computing resources.

6 Conclusion

Commercial IaaS providers are used to allocate resources mainly through the direct-selling, fixed-price model which is acknowledged to be economically inefficient. Many researchers have proposed to replace that model with market-oriented models, according to which prices should be determined on the base of the supply-demand rate. We argue that providers will be reluctant to abandon the direct-sale mechanism, as it generates certain revenues and eases



(a) Market capacity: 14 DCs (b) Market capacity: 24 DCs (c) Market capacity: 34 DCs

Fig. 11. Datacenters utilization in different market configurations in the case of ER auctions.

the work of customer retention. Nevertheless, with current pricing schemes providers are not able to sell their overall computing capacity, so alternative selling mechanisms should be sought in order to allocate the residual capacity. In this paper we proposed to trade that capacity in an procurement market open to the participation of any provider. The main contribution of this work is the definition of an adaptive strategy that providers may use to attain their business objective in procurement auctions. In the paper other considerations have also been made on the use of the resource overbooking as a chance for the provider to increase its win rate without incurring huge penalties.

APPENDIX A
A COST MODEL FOR CLOUD DATACENTERS

According to [16], the total cost of ownership of a data center may be defined as

$$Cost_{total} = Cost_{space} + Cost_{p\_hardware} + Cost_{p\_cooling} + Cost_{operation}, \quad (7)$$

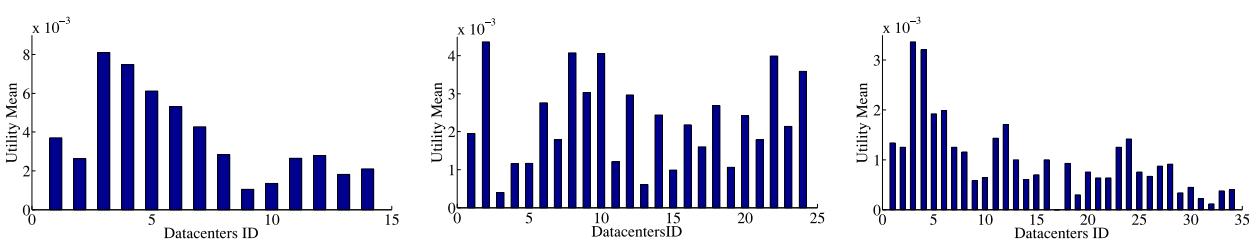
where $Cost_{space}$ is the cost incurred for the space on which the data center is located, $Cost_{p\_hardware}$ is the cost of electricity required to run the computing equipment, $Cost_{p\_cooling}$ represents the cost sustained for cooling the computer rooms, and $Cost_{operation}$ encompasses all complementary costs for operating and running the data center, such as personnel's salary and software licenses. The $Cost_{p\_hardware}$ and $Cost_{p\_cooling}$ are the most significant ones. Obviously, the more computing units are turned on, the higher the cost sustained for supplying electricity and cooling. The remainder of cost items, instead, does not depend on the number of computing units turned on, and may then be regarded as fixed costs which impact anyway the overall cost for maintaining a computing unit (be it running or turned off). In the report the TCO is furtherer detailed as

$$Cost_{total} = \left[\left(\frac{\$}{ft^2}\right)(A_{critical}, ft^2)\right] + \left[(1 + K_1 + L_1 + K_2 L_1) U_{\$,grid} P_{ConsumedHardware}\right] + \left[R\left(M_{Total} S_{avg} + IT_{Dep} + \sigma_1\right)\right]. \quad (8)$$

The first addendum represents the cost of the real estate as a function of Net Operating Income (NOI) and the actual space occupied by the equipment of computation $(A_{critical}, ft^2)$. The second addendum represents the combined costs of the power required for the hardware computing and the cooling equipment; the factors K_1, K_2 and L_1 form the cost increase due to both the depreciation of the equipment and the inefficient use of resources in the data center. The last term represents the total operating cost, including the cost for personnel, depreciation of IT equipment, software and related licenses.

We may notice that the second addendum in formula 8 accounts for the operational cost, and strongly depends on the level of utilization of the hosts populating the data center. We designed a simple scenario of a data center, consisting of 400 hosts, and tried to derive from the Equation (8) the total cost of ownership as a function of the utilization level. In Table A1parameters appearing in the formula along with the respective values used in the scenario are shown. The total monthly cost of the data center reduces to

$$Cost_{total} \cong [50 \times 200] + \left[(1 + J_1 + 1.3 + \frac{J_1}{2} \times 1.3) \times 0.13 \times 100000\right] + 50000 + 30000 = 19500 \times \left(6.14 + \frac{1}{U\%}\right). \quad (9)$$



(a) Market capacity: 14 DCs (b) Market capacity: 24 DCs (c) Market capacity: 34 DCs

Fig. 12. Datacenters utility in different market configurations in the case of ER auctions.

TABLE 3
Parameters of the Data Center and Related Values

Parameter	Description	Value
R	Number of racks utilized in data center	6 full units
NOI	Net Operating Income(in $)	$\frac{50\$}{ft^2}$
$(A_{critical}, ft^2)$	Area of the computer room (data center)	$200 ft^2$
J_1	Capacity utilization factor, depending on U%:current Utilization percentage of the system	$\frac{1}{U\%}$
K_1	Burdened power delivery factor	$\cong J_1$
K_2	Burdened cooling cost factor	$\cong \frac{J_1}{2}$
L_1	Cooling load factor	$1.3W$
$U_{\$,grid}$	Cost of grid power (in $/MWh or $/Watt per month)	$0.12 \frac{\$}{Watt}$
$P_{ConsumedHardware}$	Power consumed by the hardware, networking or cooling equipment (Watts)	$100000W$
IT_{Dep}	Straight line monthly depreciation of IT equipment, typically over a 3-year period (per rack)	50000$
$Cost_{PersonnelSoftware}$	Total monthly cost for software, licensing and personnel	30000$

TABLE 4
Lower Bound Prices to Compensate the TCO

	Amazon's VMs Reference Price	Number of runnable VMs	Lower Bound Price (U=50%)	Lower Bound Price (U=100%)
M1 SMALL	0.06	96000	0.002	0.002
M1 MEDIUM	0.12	48000	0.005	0.004
M1 LARGE	0.24	12000	0.018	0.016
M1 XLARGE	0.48	3000	0.074	0.064
M3 XLARGE	0.5	1846	0.120	0.105
C1 MEDIUM	0.145	9600	0.023	0.020
M2 XLARGE	0.410	7384	0.030	0.026

So the total hourly cost is

$$Cost_{total_h} \cong 27.1 \times \left(6.14 + \frac{1}{U\%}\right).$$

Hosts are featured as follow:

- 100 hosts equipped with 64 cores
- 100 hosts equipped with 128 cores
- 100 hosts equipped with 256 cores
- 100 hosts equipped with 512 cores

In the considered scenario, we are assuming that VMs will be offered in sizes which recall those offered by Amazon (see Table 1). We tried to figure out what is the minimum price at which VMs should be leased (let us call it *lower bound price*) in order to compensate the total cost of the data center shown in Equation (9). In particular, since that cost is dependent on the utilization level, the price will have to vary accordingly. Then, for each VM type we estimated what the lower bound price should be in the case that only that type of VM is offered by the data center. In Table A2 we report the indication of lower bound prices for every VM type, in the case that the data center's utilization level is 50 and 100 percent respectively. The reader may notice that, in the majority of cases, the lower bound price of a VM (read, the cost for running a VM) decreases with the utilization's increment. We used these data in the experimental phase of our work, which was discussed in Section 5.

REFERENCES

[1] Q. Wang, K. Ren, and X. Meng, "When cloud meets ebay: Towards effective pricing for cloud computing," in *Proc. IEEE Conf. Comput. Commun.*, 2012, pp. 936–944.

[2] O. Agmon Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing amazon EC2 spot instance pricing," in *Proc. IEEE 3rd Int. Conf. Cloud Comput. Technol. Sci.*, 2011, pp. 304–311.

[3] H. Xu, and B. Li, "Dynamic cloud pricing for revenue maximization," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 158–171, Jul. 2013.

[4] I. Foster, C. Kesselman, C. Lee, B. Lindell, K. Nahrstedt, and A. Roy, "A distributed resource management architecture that supports advance reservations and co-allocation," in *Proc. 7th Int. Workshop Quality Service*, 1999, pp. 27–36.

[5] D. Neumann, J. Stoesser, A. Anandasivam, and N. Borissov, "SORMA-Building an open grid market for grid resource allocation," in *Proc. 4th Int. Workshop Grid Econ. Bus. Models*, 2007, vol. 4685, pp. 194–200.

[6] R. Buyya, R. Ranjan, and R. N. Calheiros, "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services," in *Proc. 10th Int. Conf. Algorithms Archit. Parallel Process.*, 2010, pp. 13–31.

[7] M. Risch, J. Altmann, L. Guo, A. Fleming, and C. Courcoubetis, "The GridEcon platform: A business scenario testbed for commercial cloud services," in *Proc. Grid Econ. Bus. Models*, 2009, vol. 5745, pp. 46–59.

[8] K. Chard and K. Bubendorfer, "High performance resource allocation strategies for computational economies," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 72–84, Jan. 2013.

[9] S. Zaman and D. Grosu, "A combinatorial auction-based mechanism for dynamic VM provisioning and allocation in clouds," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 129–141, Jul.–Dec. 2013.

[10] G. Vinu Prasad, S. Rao, and A. Prasad, "A combinatorial auction mechanism for multiple resource procurement in cloud computing," in *Proc. 12th Int. Conf. Intell. Syst. Design Appl.*, 2012, pp. 337–344.

[11] P. Samimi, Y. Teimouri, and M. Mukhtar, "A combinatorial double auction resource allocation model in cloud computing," *Inf. Sci.*, 2014, doi: 10.1016/j.ins.2014.02.008.
[12] P. Klemperer, "Auction Theory: A Guide to the Literature," *J. Econ. Surv.*, vol. 13, no. 3, pp. 227–286, 1999.
[13] L. R. Smeltzer and A. Carr, "Reverse auctions in industrial marketing and buying," *Bus. Horizons*, vol. 45, no. 2, pp. 47–52, 2002.
[14] S. Parsons, J. A. Rodriguez-Aguilar, and M. Klein, "Auctions and bidding: A guide for computer scientists," *ACM Comput. Surv.*, vol. 43, no. 2, p. 10, Feb. 2011.
[15] A. Sulistio, K. H. Kim, and R. Buyya, "Managing cancellations and no-shows of reservations with overbooking to increase resource revenue," in *Proc. 8th IEEE Int. Symp. Cluster Comput. Grid*, May 2008, pp. 267–276.
[16] P. Chandrakant and S. Amip. (2009). Cost model for planning, development and operation of a data center [Online]. Available: http://www.hpl.hp.com/techreports/2005/HPL-2005-107R1.pdf
[17] J. McAfee, R. P. e McMillan, "Auctions and bidding," *J. Econ. Lit.*, vol. 15, pp. 699–738, 1987.
[18] P. Bonacquisto, G. Di Modica, G. Petralia, and O. Tomarchio, "Procurement auctions to maximize players' utility in cloud markets," in *Proc. 4th Int. Conf. Cloud Comput. Services Sci.*, Apr. 2014, pp. 38–49.
[19] R. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw.: Pract. Exp.*, vol. 41, pp. 23–50, 2011.
[20] P. Bonacquisto, G. Di Modica, G. Petralia, and O. Tomarchio, "A strategy to optimize resource allocation in auction-based cloud markets," in *Proc. IEEE Int. Conf. Services Comput.*, Jun. 2014, pp. 339–346.
[21] Google. (2011). Traces of google workloads [Online]. Available: http://code.google.com/p/googleclusterdata/

Paolo Bonacquisto received the bachelor's degree in computer science engineering and the master's degree in computer science engineering, both from the University of Catania, Italy, in 2007 and 2013, respectively. He is a research collaborator at the Department of Electrical, Electronic and Computer Engineering of University of Catania. His research interests are in cloud computing and specifically auction-based resource allocation.

Giuseppe Di Modica received the degree in computer engineering from the Engineering Faculty of University of Catania, Italy, in 2000. In 2005, he received the PhD degree in computer science and telecommunication engineering from the University of Catania. Since then, he has been engaged in research on distributed systems with the Department of Computer and Telecommunications Engineering of Catania University. His research interests include mobile agents, ad-hoc networks, wireless sensor networks, mobile computing, service oriented architectures, grids and cloud computing. He is currently a research assistant at the Engineering Faculty of University of Catania, Italy. He is a member of the IEEE.

Giuseppe Petralia received the bachelor's degree in compute science engineering and the master's degree in computer science engineering, both from University of Catania, Italy, in 2008 and 2012, respectively. He is a research collaborator at the Department of Electrical, Electronic and Computer Engineering of the University of Catania. His main research interests are cloud computing, market-oriented resource allocation, service level agreements and semantic web.

Orazio Tomarchio received the degree in computer engineering from the Engineering Faculty of the University of Catania, Italy, in 1995. In 1999, he received the PhD degree in computer science from the University of Palermo. He is currently an assistant professor with the Department of Electrical, Electronic and Computer Engineering of the University of Catania, beginning from February 2002. He has been engaged in teaching several degree and master courses mainly focused on software engineering, security and programming in distributed systems. His scientific activity was initially focused on studying distributed systems, particularly with regard to innovative programming and management techniques based on the mobile code paradigm. Recently, his scientific interests have been also focused on middleware for mobile ad-hoc network and for personal and ubiquitous computing, mobile P2P computing, Grid and service oriented architectures, security, SLA and market-oriented resource allocation in Cloud computing, and semantic technologies. He is a member of the IEEE.