Association for Information Systems
AIS Electronic Library (AISeL)

ICIS 2010 Proceedings

International Conference on Information Systems (ICIS)

2010

TOWARDS AN EFFICIENT DECISION POLICY FOR CLOUD SERVICE PROVIDERS

Arun Anandasivam
Karlsruhe Institute of Technology, anandasivam@kit.edu

Christof Weinhardt
Karlsruhe Institute of Technology, christof.weinhardt@kit.edu

Towards an efficient decision policy for Cloud service providers

Completed Research Paper

Arun Anandasivam
Karlsruhe Institute of Technology
Englerstr. 14, Karlsruhe, Germany
anandasivam@kit.edu

Christof Weinhardt
Karlsruhe Institute of Technology
Englerstr. 14, Karlsruhe, Germany
weinhardt@kit.edu

Abstract

Cloud service providers may face the problem of how to price infrastructure services and how this pricing may impact the resource utilization. One aspect of this problem is how Cloud service providers would decide to accept or reject requests for services when the resources for offering these services become scarce. A decision support policy called Customized Bid-Price Policy (CBPP) is proposed in this paper to decide efficiently, when a large number of services or complex services can be offered over a finite time horizon. This heuristic outperforms well-known policies, if bid prices cannot be updated frequently during incoming requests and an automated update of bid prices is required to achieve more accurate decisions. Since CBPP approximates the revenue offline before the requests occur, it has a low runtime compared to other approaches during the online phase. The performance is examined via simulation and the pre-eminence of CBPP is statistically proven.

Keywords: Cloud Computing, Bid-price control, Revenue Management

Introduction

In the Cloud Computing market, Cloud service[1] providers face dynamic and unpredictable consumer behavior. The methodology, through which prices are set in a dynamic environment by providers, can influence the demand behavior of price sensitive consumers (Bitran and Caldentey, 2003). Consequently, consumers with a low valuation for a service would use it during cheaper periods. Business consumers are willing to pay a higher amount for its usage. By identifying the right price for a customer and a requested service at a certain point in time, providers can achieve higher revenues (Kimes, 1989).

An important aspect in Revenue Management is the comprehensive view of prices in conjunction with advance reservation. Advance reservation is already applied to Cloud services. For example, RenderRocket[2] offers on-demand access to rendering software to create animated movies or to render architectural and industrial designs. They charge users “on-demand hourly” with rates starting from $1.50 per server hour. However, they also offer advance reservation of servers on a daily, weekly or monthly basis. This offer allows a guaranteed priority on reserved instances and an on-demand scalable system. Although advance reservation and the analysis of consumer behavior can reduce uncertainty in demand, there are still unpredictable occurrences. An example of unpredictable service requests is the problem Animoto[3] faced in spring 2008. Animoto offers to create customized web-based videos automatically by uploading images and music. It needs substantial computing power for video processing. In spring 2008, Animoto had an unpredictable demand for their service, when 750,000 people signed up for it within three days. However, the existing infrastructure was not able to manage it. The instant availability of Amazon’s EC2 allowed them to add up to an additional 3,500 virtual instances to satisfy the spike in demand[4].

Furthermore, Cloud services are characterized by various properties defined in Service Level Agreements (SLAs), which enable the implementation of price discrimination strategies for distinguishing similar services based on their SLA attributes (e.g. higher throughput or lower availability rate). From a provider’s point of view, offering various kinds of services based on advance reservation and on-demand instances can lead to uncertain demand requests. When demand outweighs supply, a provider has to decide, whether to accept an incoming request or reject it in favor of a request arriving later for a service with higher revenue. In the case of Amazon’s Spot instances the price can be calculated by analyzing several scenarios with different price settings and virtual resource limitation in order to determine the appropriate pricing strategy.

In this paper, a decision concept for a provider is presented to accept or reject incoming requests for services in order to increase revenue in a scarce resource market. A provider offers several Cloud services, which use the same resources from the provider’s resource pool. The goal of the provider is to sell the most expensive services to the paying customer (Phillips, 2005). When a consumer requests a service with low revenue, the provider has the possibility to accept this request or to wait for a prospective customer asking for the high valued services. Different decision rules well known from Revenue Management for the Airline Industry are analyzed to understand how to apply Revenue Management concepts to Cloud Computing. This contribution comprises of a more efficient decision rule called Customized Bid-Price Policy (CBPP) in Section ‘Optimization Model’. The settings for the simulation as a research methodology and the relevant hypotheses are described in Section ‘Simulation Environment and Research Questions’. Its efficiency is analyzed via simulation-based optimization in Section ‘Results’.

---

[1] The optimization model in this paper focuses on infrastructure as a services (IaaS). Platform (PaaS) and Software as a service (SaaS) are not considered. In this paper the term cloud service is used synonymously with infrastructure as a service.

[2] http://www.renderrocket.com, last accessed on April 04, 2010.

[3] http://www.animoto.com, last accessed on April 04, 2010.

[4] NY Times article (2008): http://www.nytimes.com/2008/05/25/technology/25proto.html?_r=3&adxnnl=1&oref=slogin&ref=business&pagewanted=print&adxnnlx=1212768774-L4fMNfgaHc/0lDK5wKcevQ, last accessed on March 03, 2010.

Related Work

Each offered service represents a booking class, which has a fixed price. The provider has to decide, if a service request should be accepted or rejected. Thus, a limit defining how many requests are operable for each booking class has to be identified, which is known as capacity control. Nested booking limits allow the prevention of bookings for services with higher revenue being rejected in favor of bookings with lower revenue. They define how much capacity is reserved for a certain booking class. Every service has limited access to resources like CPU, memory, storage, or bandwidth. Due to multiple resources a nested booking limit control must be defined for each resource. This is called virtual nesting control (Williamson, 1992; Smith and Penn, 1988). It is difficult to forecast demand appropriately for virtual classes. The requirement of mapping services to virtual classes also increases complexity (Talluri and van Ryzin, 2004).

Bid prices are interpreted as an approximation of the opportunity cost of reducing the resource capacities, which are needed to satisfy incoming service requests (Bertsimas and Popescu, 2003). Möller et al. (2008) describe bid prices as monetary values of a single capacity unit for a resource. The sum of the resource demands of a request weighted with the corresponding resource bid prices defines the bid price of a service. If this sum exceeds the revenue yielded by the sale of one unit of the respective service, the request is rejected, otherwise it is accepted (Williamson, 1992). Regular updates of bid price values are necessary to guarantee a continuous precision of the bid prices. Less accurate bid prices can lead to accept/reject decisions of minor value. Continuously updated bid prices are based on the current booking situation at a certain point in time t. That is, if a large amount of capacity has already been sold, the bid prices turn out to be higher.

Bichler and Setzer (2007) propose an admission control for media on-demand services, e.g. a media streaming service. The authors compare an adaptive admission control based on a Deterministic Linear Programming (DLP) model (Williamson, 1992) with static admission rules, and point out the benefits of the adaptive method. Their results show that the adaptive DLP control rejects early service requests, and thereby is able to accept high-revenue service requests arriving later.

The first paper analyzing Revenue Management concepts for on-demand Information Technology (IT) scenarios was published by Dube et al. (2005). In the suggested model one resource is offered at different prices. By assuming that the customer behavior follows a logit model, the authors analyzed an optimization model for a small number of price classes and provided numerical results. Although the authors state that ”in an on-demand operating environment, customers and jobs, or service requests arrive at random”, the behavior of price sensitive customers can be influenced by offers different prices for the same product, which in turn reduces the randomness (Bitran and Caldentey, 2003; Wilson, 1995).

Nonetheless, a certain degree of uncertainty still exists due to the unpredictable customer behavior and unpredictable events. Customers may cancel their resource reservations or may not show up. Overbooking strategies can be used to accept more reservations than the actual available capacity, which can effectively minimize losses of revenue caused by customer no-shows. The benefits of overbooking for shared hosting platforms were emphasized by Urgaonkar et al. (2002). They did not optimize the revenue by classifying different services, but only the throughput rate. Cancelations and no-shows reduce the efficiency of resource usage. Sulistio et al. (2008) analyzed how overbooking strategies can be applied to maximize revenue. Different prices were charged for one resource and three overbooking policies were implemented and compared via simulation.

Nair and Bapna (2001) introduced Revenue Management concepts for a similar application domain, namely of an Internet Service Provider. The provider has to decide whether to accept an incoming customer request or to reject it. The application domain is different from Cloud Computing as it does not take advance reservation and resource-service dependency into account. Customers can get an internet access only instantly.

Optimization Model

The decision of accepting or denying a request depends on the applied policy or heuristics. Capacity control comprises heuristic approaches for the original dynamic programming problem. A Bellman equation defines the optimal policy for accepting or rejecting requests. Since the speed of computation matters (especially for large resource/product settings), bid-price control is an approximation method to quickly update the policies after the arrival of new requests. It provides a good estimate, but not always an optimal solution. The calculation of the optimum increases exponentially with the number of resources m and services n (Talluri and van Ryzin, 2004).

Cloud services in the Revenue Management context depend on various parameters. A service provider offers different kinds of services. Each service is based on physical resources to generate the services. These services are requested at different points in time. The decision when to accept or reject a request may change during the time horizon for the same service, since the calculated bid price for every resource depends on the current utilization of the resource. A customer has the possibility to book a service within a booking period T. T corresponds to the total amount of time remaining to book services and is finite and countable. Time is discrete with a point in time $t \in \{T, T-1, \dots, 1\}$ (Adelman, 2007). In practice, the booking period in the Cloud service domain is considerably smaller than in airline Revenue Management. This contributes to a more spontaneous setting with customers booking services according to the more agile business environment and changing requirements for Cloud services. The provider has to manage and control m different resources $h \in \{1, \dots, m\}$. In the present context, the term "resources" describes the computing capacities. Examples for resources are CPU power, memory, storage or bandwidth. Resources are necessary to provide Cloud services consuming these resources. Resources are allocated to n different classes with each service $i \in \{1, \dots, n\}$. Each service class represents a (virtual) computing environment that can be used by the customer for computational purposes. The definition of the services depends on the defined setting $m \times n$ (resources×services), e.g. setting 3×3 considers three resources and three services. Formally, the matrix $\boldsymbol{A}$ describes the mapping between the resources and services. An element a_{hi} represents the usage of resource h by one unit of service i. $\boldsymbol{A}_i$ shows all resources consumed by service i and $\boldsymbol{A}^h$ the services using resource h. The resource consumptions of the different services are expressed by matrix $\boldsymbol{A}_i$. The price of service i is denoted with r_i, and depends on the amount of resources required by the service. r_i is the revenue yielded by the sale of one unit of service i. Prices are fixed over a predefined booking period.

D_{iT} represents the amount of requests for service i arriving in the complete booking period T. D_{iT} can be subdivided into the expected demand arriving between the current point in time t and the end of the booking period, denoted as D_{it} (demand-to-come), and the demand prior to the beginning of the booking period until t as $\check{D}_{it}$ (see Figure 1).

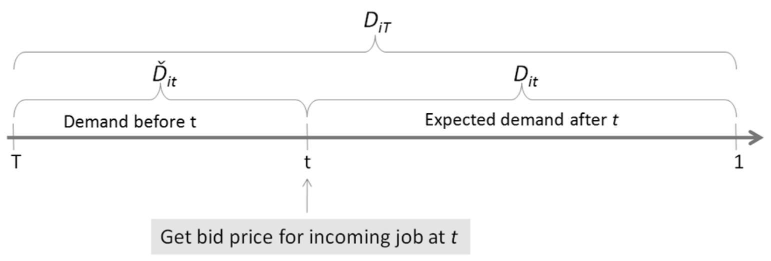

D_{iT}
$\check{D}_{it}$
D_{it}
Demand before t
Expected demand after *t*
T
t
1
Get bid price for incoming job at *t*

Figure 1: Demand definition in a finite time

A request for service i at time t arrives with probability p_{it}, and thus, the arrival of a request for service i at time t is a random variable X_t with $X_t = (\{0,i\} \mid i \in \{1, \dots, n\})$, i.e. $X_t = 0$ if no request comes in at t. T is finite and countable, and thus, the arrival process of requests by the customers is a time-discrete stochastic process X, which is a sequence of random variables X_t . The demand for service class i arrived from T until t is described by $\check{D}_{it}$. If a request for service i occurs in time slot t, the demand arrived until t changes from its previous value $\check{D}_{it}$ at time $t+1$ to its new value $\check{D}_{it}+1$ at time t (note that the time runs backwards).

Bid Price Control

The dynamic programming problem was heuristically approached by several authors. Two different models are presented in this section: *Deterministic Linear Programming (DLP)* and *Certainty Equivalent Control (CEC)*.

Deterministic Linear Programming Model: The network model for bid-prices assumes expected demand information and excludes the stochastic nature of the demand (Glover et al., 1982; Williamson, 1992). Based on demand forecasts the expected aggregate demand-to-come D_{it} for the remaining booking periods is calculated, and it is assumed that the demand is equal to its mean values. An approximation for the objective-value function V is obtained by:

$$\max V(x) = \sum_{i=1}^{n} r_i \cdot x_i \quad (1)$$

$$s.t. \sum_{i \in A^h} a_{hi} \cdot x_i \leq c_h - \bar{c}_{ht} \; \forall h \in \{1, \dots, m\} \quad (2)$$

$$0 \leq x_i \leq D_{it} \; \forall i \in \{1, \dots, n\} \quad (3)$$

The condition for accepting a request is: The revenue r_i yielded through the sale of service i must be greater than or equal to the sum of the resource consumptions of service i weighted with the corresponding bid-prices (Talluri and van Ryzin, 1998):

$$r_i \geq \sum_{h \in A_i} a_{hi} \cdot \pi_{ht}. \quad (4)$$

Additionally, it must be ensured that there is still sufficient capacity of every resource available to satisfy the request. This is expressed by the condition:

$$a_{hi} \leq c_h - \bar{c}_{ht} \;\; \forall h \in A_i \quad (5)$$

(1) is the objective function, which maximizes the total revenue. The total revenue results from the sum of the prices r_i charged for each service multiplied by the number of units of each service sold in the booking period x_i. Constraint (2) ensures that enough capacity of each resource is available to satisfy the need for capacity by the number of allocated units of the services. Constraint (3) guarantees that the number of services sold are not below zero and do not exceed the expected demand-to-come.

The main benefit of the DLP model is that it can be solved efficiently, which makes it popular for practical applications. Its performance strongly depends on the size of the network as well as on the reliability of the demand forecasts. However, this model does not imply any uncertainty in demand. Another drawback of the DLP model is that the dual variables of resources can be zero. Hence, capacity for a resource is higher than the mean demand (Talluri and van Ryzin, 1998). Consequently, too low bid-prices can be the result. The accuracy of bid-price values in the DLP model depends on how frequent these recalculations are performed. The most frequent calculation of bid-prices is carried out by recalculating bid-prices each time a request occurs.

Certainty Equivalent Control: Another approach called certainty equivalent control extends the concept of bid-prices and directly calculates an approximation of the opportunity cost for every service and not for every resource (Bertsimas and Popescu, 2003). For this purpose it solves two instances of the DLP problem described above: The first instance solves the initial DLP problem (1) and the second instance subtracts the amount of resources demanded by the request from the remaining capacity of the resource

$$c_h - \bar{c}_{ht} - a_{hi} \;\; \forall h \in \{1, \dots, m\}. \quad (6)$$

The approximation of the opportunity cost of service i is then obtained by subtracting the objective function value of instance 2 (*V'(x)*) from the objective function value of instance 1 (*V(x)*). This approximation does not depend on the optimal dual variables. Thus, the drawback of multiple optimal dual variables of the linear programming model is eliminated. The CEC policy requires forecasts for the total demand for each service, as well as forecasts for the expected demand-to-come (D_{it}). The main advantage of the CEC policy arises from the numerous and periodic updates of the approximation of the opportunity costs, thereby guaranteeing a certain accuracy. Since CEC is based on the DLP problem, it shares the same disadvantage of only incorporating expected demand and not considering uncertainty of the demand process. However, Bertsimas and Popescu (2003) have proven that CEC outperforms DLP.

Customized Bid Price Policy

A different Network Capacity Control approach for calculating bid prices was proposed by Klein (2007). The idea of this approach is to use simple linear additive functions to calculate bid prices every time a request occurs. The functions are based on parameters, which can easily be kept on track during the booking period, such as the current amount of reserved capacity $\bar{c}_{ht}$ as well as the expected demand-to-come D_{it}. It uses a continuous time model with a booking period represented by the time interval $[T;1]$. The concept further involves the determination of coefficients (control variables) via simulation-based optimization. The control variables are used for calibrating the bid-price functions adequately, which are evaluated offline before the time horizon. The Customized Bid-Price Policy (CBPP) approach is based on the resource-oriented bid-price function of Self-Adjusting Bid-Price (SABP) from (Klein, 2007). The bid price of resource h at time t is calculated by the formula:

$$\pi_{ht} = \bar{\pi}_h + \alpha_h \cdot \frac{\bar{c}_{ht}}{c_h} - \beta_h \cdot \frac{u_{ht}}{U_{hT}} \tag{7}$$

The control variables $\bar{\pi}_h$, α_h and β_h are determined via a genetic algorithm. The parameters α_h and β_h are control variables required for calibration purposes in the simulation-based optimization. The customized bid-price function is based on two parts. The first part ($+\alpha_h \cdot \frac{\bar{c}_{ht}}{c_h}$) increases the bid price $\bar{\pi}_{\mathrm{ht}}$, if requests are accepted. The amount of reserved capacity of resource h at time t ($\bar{c}_{ht}$) is divided by the total capacity of resource h (c_h), and hence can only take values in $[0;1]$. An acceptance of a request for service i leads to an increase of the bid price of resource h to the amount of the delta of the value $\frac{\bar{c}_{ht}}{c_h}$. The increase of the bid price also depends on the value of α_h.

The decreasing part ($-\beta_h \cdot \frac{u_{ht}}{U_{hT}}$) lowers the bid price every time a request is made. As explained above, u_{ht} corresponds to the amount of capacity needed to satisfy the demand until t for the services $i \in A_h$. U_{hT} is the capacity of resources that is required to satisfy the total expected demand of the complete booking period calculated by $\sum_{i \epsilon A^h} a_{hi} \cdot D_{iT}$. U_{hT} can also be denoted as the total resource demand. It requires a forecast of the total demand for service class i in the booking period. The quotient $\frac{u_{ht}}{U_{hT}}$ is always in $[0;1]$, and increases over time as more demand is realized. Hence, the bid price decreases with incoming requests. Reasonable values for the control variables $\bar{\pi}_h$, α_h, and β_h are obtained by a genetic algorithm in order to minimize inefficient outcomes.

The main advantage of SABP and CBPP is the very frequent recalculation of bid prices with minimum computational effort. Thereby, information about the current booking situation is always considered, and the bid prices exhibit a certain precision. Klein (2007) states, that this approach is robust to errors in the forecast. If the realized demand is less than the forecasted one, less capacity units may be reserved, and more capacity is available to satisfy incoming requests. Thus, the increasing part of the bid-price function is lower, and more requests with lower revenues can be accepted. On the other hand, if the realized demand is higher, the bid price increases strongly as more capacity is reserved due to the acceptance of requests. Furthermore, a strict fragmentation of the services into only two subsets (S1: Accepted, i.e. $r_i \geq \sum_{h \epsilon A_i} a_{hi} \cdot \pi_{ht}$; S2: Rejected, i.e. $r_i < \sum_{h \epsilon A_i} a_{hi} \cdot \pi_{ht}$) is avoided. This fragmentation arises in models, which do not perform a frequent recalculation of bid prices. Given static bid prices, it is fixed, which service classes can be accepted and which not. Through permanently updating the bid prices a certain accuracy is guaranteed, and a strict fragmentation cannot occur. Moreover, the offline optimization of the control variables allows saving time during the period, when requests arrive. A sum of the current bid prices for every resource has to be calculated, while the bid prices are updated automatically over time.

Note that the SABP concept of Klein (2007) has been developed for managing resources in the context of airlines. The resource consumption of airline services, which are seats in certain booking classes on certain flights at certain dates, is very different to the resource usage by services in Clouds. To demonstrate this, consider the following example. An airline offers different types of booking classes (e.g. economy class and business class) on each of its flight. Depending on the booking classes and on the flights offered, the airline can offer a certain number of services. However, in this case the resource consumptions by the different services usually are all one unit of a single resource. Table 1 represents a service-resource mapping for Cloud services with $m = 3$ resources and $n = 3$ services.

It can be easily observed that the resource demands (a_{hi}) in the two areas of application show a major difference. Matrix $\boldsymbol{A}_i$ in case of services in Clouds exhibits significantly higher natural numbers than in case of airline services, which usually consume one seat on a single-leg flight. Even if an airline offers multi-leg services, that is, if someone buys a sequence of flights over multiple destinations, the resource usage per flight is always one seat. Note that group bookings are not considered in these models. Group bookings require a different demand modeling, since the amount of services in group bookings vary over time. The service resource mapping is fixed over the entire time horizon. Hence, accepting one booking request in the context of services in Clouds means a considerably larger change in capacity compared to the acceptance of a booking request for an airline ticket, but is not comparable to the group booking problem. Furthermore, accepting service with different kinds of resource demand can significantly affect the yield. It is possible to accept services which have a lower revenue, but need abounded resources, while high-fare services have to be rejected due to resource scarcity.

Table 1: Resource usage by services in Clouds (left) and airline domain (right)

Service *i* Resource *h*	*i*=1	*i*=2	*i*=3
h=1	2	4	8
h=2	2	8	4
h=3	8	2	4

Service *i* Resource *h*	*i*=1	*i*=2	*i*=3
h=1	1	0	0
h=2	0	1	1
h=3	0	0	1

Depending on the type of service requested by the consumer, the capacity of multiple resources may be decreased in case of an acceptance, which applies to all three services in this example (Table 1). Therefore, a stronger competition for multiple resources takes place between services in Clouds. Hence, the usage of the original bid-price function by Klein (2007) would lead to very high values of the increasing and decreasing part of the function in the setting of Cloud services, and thus, some bid price values can turn out to be inappropriate. This is avoided by using relative values in (7).

Simulation Environment and Research Questions

The relevant research stream for this paper is the decision support and design science stream. Related disciplines in this stream are operations research, computer science, economics, marketing and strategic management (Banker and Kauffman, 2004). Revenue Management is an interdisciplinary approach taking economics, statistics, software tools and strategic decisions into account to maximize the revenue (Secomandi et al., 2002). Talluri and van Ryzin (2004) stress the relevance of marketing methods for Revenue Management to identify consumer behavior in order to apply the mathematical models to practice in a better way. Hence, there is a great concurrence between these two research methodologies.

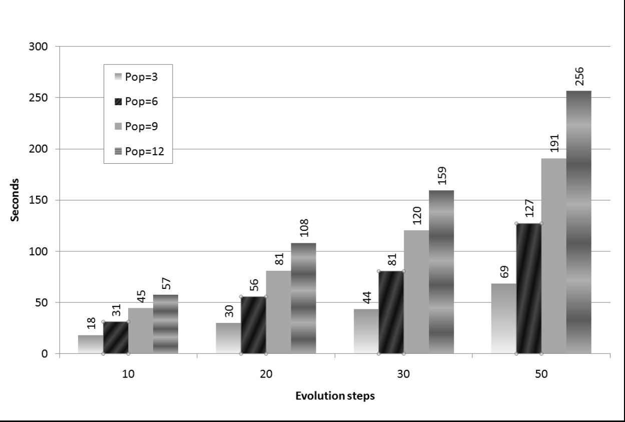
Banker and Kauffman (2004) indicate that simulation is a common research method to analyze and solve the identified problem in the Information Systems discipline. Genetic algorithms have been proven to perform well for simulation based optimization (Holland, 1975). Simulation as a research method is applied here for two purposes. First, the parameters for CBPP are determined via simulation-based optimization to identify the optimal control variables. Second, simulations help to understand complex interaction between various parameters in different kind of settings. Hence, data for statistical analysis is created via simulation in order to reveal the dependencies.

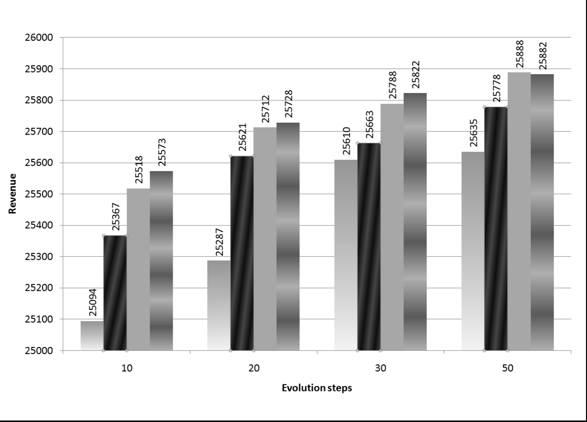
Furthermore, several assumptions, which are common in the Revenue Management context, also apply in this simulation. It is assumed that the provider has certain demand information from past booking periods, and is able to perform a more or less accurate demand forecast. Fluctuations in demand will be handled by generating different demand scenarios during the simulation. Customers, who do not use the booked services, do not get a refund. Hence, if consumers book a service they will have to pay for it. Cancelations and no-shows are not considered in the model. Demand for each service is independent and can be modeled by a probability distribution (Weatherford and Belobaba, 2002).

Several parameters describe the setting of the simulation. This paper focuses especially on the interdependency between the bound values for the genetic algorithm parameters, the price variation, the service-resource mapping and their impact on Demand-to-Capacity ratio (DCR), the bid price for each resource and the achieved revenue. Changes in demand and capacity will obviously influence the DCR and thus the outcome of the algorithm.

Genetic Algorithm

Genetic algorithms, introduced in 1975 by John Holland (Holland, 1975), are used as a tool for search and optimization. They belong to the class of evolutionary algorithms defined in the 1960s by Rechenberg (1973). Genetic algorithms are optimization concepts, which search a solution space of a given problem for reasonable solution values. The interested reader is referred to (Goldberg, 1989). The objective of the genetic algorithm is to find adequate values for the control variables of the customized bid price function (7). The bid price function contains three control variables: The base bid price variables $\bar{\pi}_h$ as well as α_h and β_h. All three parameters are calculated for each single resource h. Hence, the genetic algorithm has to find three adequate values for each resource, which leads to a total number of genes in a chromosome of $3 \cdot m$.





(a) Runtime comparison

(b) Revenue comparison

Figure 2: Runtime and revenue analysis with different population size and evolution steps for a 4x5 setting

The genetic algorithm needs more computational power than DLP and CEC to identify the best gene values for an optimal decision. Naturally, the runtime is dependent on the population size (POP), the number of variables (genes) to search for and the number of evolution steps (ES). The number of genes are static and represented by the variables α_h, β_h and $\bar{\pi}_h$. The impact of the values of these variables is analyzed below. A higher population size leads to significantly more computation time. For example, the runtime for a setting with a population size = 9 and 30 evolution steps is almost 4 times longer than for a population size of 12 with the same amount of evolution steps on average (Figure 2a). An increase of the evolution steps will result in a higher runtime as well. A doubling of the number of evolution steps leads almost to a doubling of the runtime of the genetic algorithm. It is important to state that a longer evolution time does not necessarily lead to a better solution. For example, in Figure 2b the setting with POP = 9 and ES = 30 gains higher revenue than the settings with POP = 6 and ES = 50. Furthermore, the runtime is slightly better (Figure 2a). Moreover, a rise in the population size leads to a longer runtime of the genetic algorithm, but it also adds diversity to the population, which in turn increases the probability of finding a better solution more quickly in terms of number of evolution steps. However, even in this case, the performance will not monotonically increase with higher population size. For ES = 50 the setting with POP = 9 performs slightly better than with POP = 12 on average despite the latter setting having a 34% longer runtime. The difference between the setting with the lowest revenue (POP = 3; ES = 10) and the setting with the highest revenue (POP = 9; ES = 50) is about 3% increase in revenue, which is significant in the Revenue Management context (van Ryzin and Vulcano, 2008). Since runtime and revenue are based on average values analyzed over 50 different runs, the outcome can strongly vary in these runs. Thus, an analysis on the standard deviation for every setting reveals that the deviation decreases with the increase of evolution steps (Table 2). Hence, it is more likely to receive a good revenue with more evolution steps and with a higher population size.

Table 2: Standard deviation for all settings

	Pop=3	Pop=6	Pop=9	Pop=12
ES=10	323,1	179,8	143,1	140,6
ES=20	232,4	146,0	135,2	129,4
ES=30	151,7	142,8	117,5	114,0
ES=40	128,9	121,6	107,1	94,3

Simulation Process and Hypotheses

The simulation process can be subdivided into three phases: Demand modeling, offline calculation and online calculation (see Figure 3). Compared to online algorithms like DLP and CEC, CBPP requires an offline calculation phase to determine the appropriate values for the control variables.

Demand modeling defines the incoming request according to a certain probability function. A standard approach is to use a non-homogeneous Poisson process (Talluri and van Ryzin, 1999). Although a discrete time model is assumed for the demand modeling and the Poisson process is based on continuous time model, this demand data can be applied for discrete time simulation as well (Bertsimas and Popescu, 2003; Subramanian et al., 1999). Several studies have chosen a Beta distribution in combination with a Gamma distribution to model the demand data.

The Gamma distribution is used for the expected demand of a service and the Beta density function for the distribution of the incoming services over the time horizon (Bertsimas and de Boer, 2005; Weatherford et al., 1993). In this paper this approach was adapted. The flexible structure of the Beta density function enables a flexible modeling of the demand over time for each service. Hence, the probability of each service class can have its peaks at different points in time during the entire period, which is required to have a certain mix of different services over time and not to follow a strict low-before-high order. Kimms and Mueller-Bungart (2007) provide a thorough literature review about on-demand data assumptions for Revenue Management and describe the demand modeling in detail.

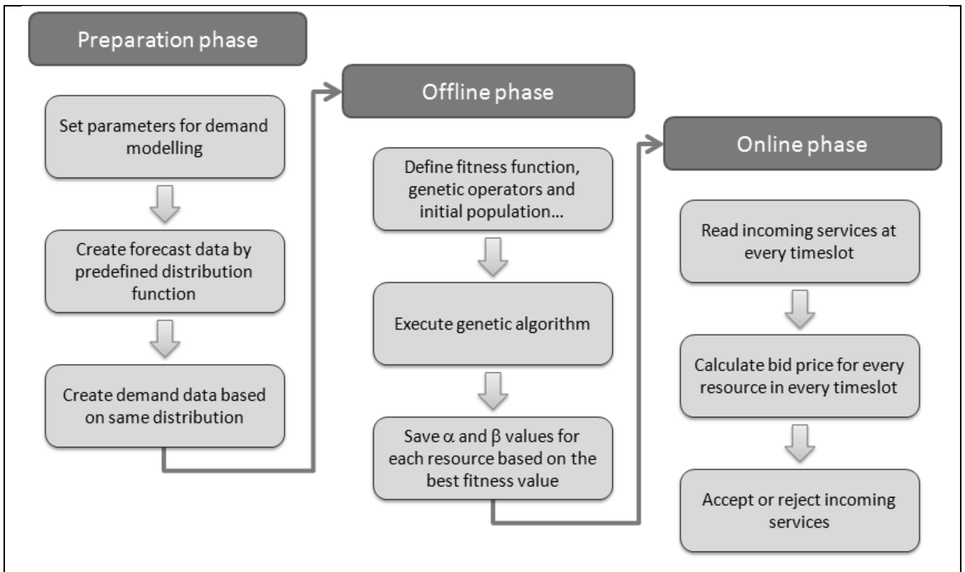

Preparation phase
Set parameters for demand modelling
Create forecast data by predefined distribution function
Create demand data based on same distribution
Offline phase
Define fitness function, genetic operators and initial population...
Execute genetic algorithm
Save α and β values for each resource based on the best fitness value
Online phase
Read incoming services at every timeslot
Calculate bid price for every resource in every timeslot
Accept or reject incoming services

Figure 3: The three simulation phase

Not only the demand has a certain probability function, but the prices and the service-resource mapping as well. Prices are selected from a uniform distribution between [10;50] and every value a_{hi} for the service-resource mapping between [0;10]. These values were necessary to appropriately distinguish between each service setting and to reduce homogeneity in the price set. The simulation-based optimization is commonly based on forecast (van Ryzin and Vulcano, 2008; Gosavi et al., 2007). Hence, forecast values for every timeslot have to be determined by using the same probability function as for the demand. Then, the actual demand is created and it can be compared to the forecast values in every timeslot. However, an analysis on forecast errors is not considered in this paper.

Frank et al. (2008) suggest to separate the demand generation from the actual simulation. Therefore, the determination of the control variables is done by the genetic algorithm in the *offline calculation phase*. The fitness function is equal to the revenue achieved by the algorithm. A higher revenue means a higher fitness value of the chosen chromosome. The genetic operators are crucial to identify good parameter values during the evolution steps. The initial population size and the evolution steps have to be defined as well. For the simulation the population size was set to 9 and the evolution steps to 25. This setting does not always provide the best results (see Section 'Genetic Algorithm'), but it is an average benchmark for a trade-off with an acceptable runtime and acceptable volatility of the revenue results. Hence, results can be improved by selecting a bigger population size and a higher number of evolution steps. After the parameters are set, the genetic algorithm has to be executed. The outcome is the achieved revenue and values for the parameters α_h, β_h and $\bar{\pi}_h$. These values were determined based on the forecasted demand, the current price setting, the predefined service-resource mapping and the predefined bounds for the control variables. They will be used in the online phase.

In the *online phase* at most one request arrives per timeslot. CBPP analyzes, if the request can be accepted or rejected based on the control variables determined in the offline phase. If the bid price is equal or below the fixed price for the service, the service will be accepted; otherwise, it will be rejected. After each timeslot (whether a request has arrived or not) the bid price for every resource will be updated according to α_h, β_h and $\bar{\pi}_h$.

Klein (2007) has outlined the difficulty of updating bid prices during incoming requests for online algorithms. Thus, DLP or CEC algorithms are not executed every time a request occurs. One reason is the runtime of these algorithms are too long to execute between two requests. Although software like CPLEX[5] can handle very complex scenarios (Bixby, 2002), the execution of a sum for CBPP is always faster than a linear programming problem. The complex calculation of CBPP is done offline, while the update of the bid prices is performed online during the incoming requests. Moreover, requests can arrive within milliseconds, which require a simple automatically updating algorithm. This paper analyzes the behavior of CBPP statistically and provides an evaluation of its performance in the Cloud context to answer the research question:

Research Question: How accurately can a simple linear function approximate well known algorithms for bid price calculation without reoptimization between two or more timeslots, taking the assumptions and requirements from Revenue Management in general and from Cloud Computing into account?

The first part of CBPP is an offline algorithm and the relevant parameters are optimized before the time horizon of incoming requests. Then, the bid prices are automatically updated online, whenever a request occurs. If online algorithms like DLP or CEC cannot be updated at every incoming request, CBPP can be applied to automatically adapt the current bid price. This paper analyzes different variations of the online algorithms. At first, both algorithms are updated in every timeslot (DLP-D and CEC-D). Then, a quasi-static version (DLP-S and CEC-S) is analyzed, which only permits to be updated DLP and CEC in certain timeslots (e.g. every second, third or tenth timeslot). The performance of CBPP is measured by the revenue achieved in the simulation. Several aspects have to be analyzed when comparing the online and offline algorithms. Obviously, the revenue of CBPP has to be higher than the online algorithms for different kinds of fare class settings:

Hypothesis 1: The revenue yielded with CBPP is higher than the revenue obtained with CEC-S.

Hypothesis 1 will give a general idea of the performance of CBPP. A more detailed analysis is necessary to understand, if the revenue is differing between the fare class settings. This should not be the case to derive a general statement for Hypothesis 1. Otherwise, a sensitivity analysis is necessary to understand, which variables affect the outcome. For example, the choice of the price for the services can influence the accept/reject decision significantly. Changes in price may lead to other combination of service acceptance and thus to different outcome:

[5] http://www.cplex.com/

Hypothesis 2: The revenue yielded with CBPP does not vary among different prices for the same service-resource mapping.

Furthermore, the control variables for the genetic algorithm, namely α_h, β_h and $\bar{\pi}_h$, have an impact on the accuracy of the genetic algorithm and thus on the outcome. Every control variable requires an upper and a lower bound. These bounds narrow down the possible values of the control variables for the genetic algorithm. If the spread is too small, the optimization function is less flexible, which results in lower revenue. A large spread provides too many options from which the genetic algorithm can select. It reduces the probability of finding the near-optimal solution unless the population size and evolution steps are high enough. Therefore, the impact of the control variables has to be analyzed:

Hypothesis 3: The fine-granular selection of upper bounds does not influence the revenue.

In Revenue Management the parameter Demand-to-Capacity ratio (DCR) plays an important role in the understanding of how bid price decisions may change, when demand changes (Weatherford and Belobaba, 2002). DCR defines the ratio between the total incoming requests and the total capacity of each resource. A ratio of one means the total demand can be satisfied. DCR > 1 represent an excess in demand and in case of DCR $\leq$ 1, resources are not used at the end of the time period. The modeling of DCR in the simulation will influence the revenue. A very high demand (e.g. DCR > 2) for certain or all services may result in a minor impact of the bid price control. If there are too many high-fare service requests (vs. available capacity), all of them can be accepted and the highest revenue is yielded. If capacity is still available, other services can be accepted, if it is possible to backfill the available capacity. The bid-price control will reject most of the demand for low-fare services unless some of them are necessary for backfilling. Furthermore, a service provider, who faces such a high demand, tends to shift resources in order to satisfy this demand. Since this paper analyzes a scarce market, the case of DCR $\leq$ 1 is disregarded. The optimal strategy in this case would be to accept all services.

The assumptions of Revenue Management mentioned in the research question refer to the demand modeling applied from the Revenue Management context, since no data profile of current demand data in Clouds is available yet. In the simulation the demand is modeled according to Kimms and Mueller-Bungart (2007).

Results

The CBPP algorithm provides a heuristic approach to automatically and efficiently update the bid prices between two or more timeslots in order to maximize the revenue. Hence, three hypotheses were defined to understand the dependency between the parameters and the simulation outcome.

Statistical Results

Hypothesis 1 states that CBPP outperforms CEC-S. According to van Ryzin and Vulcano (2008) an increase of 1% to 3% is already significant in Revenue Management. To corroborate the hypothesis, a one-tailed one sample t-test was executed. CBPP has to show at least 2% higher revenue than CEC-S to gain significant results (H_0: $m \leq 2\%$ and H_1 : $m > 2\%$). Different service-resource mappings were chosen to show the significance over several settings. The service-resource mapping was created by a uniform distribution (see Section 'Simulation Process and Hypotheses'). Consider that in certain settings, CBPP can also underperform compared to the static algorithms. It depends on the prices between high and low valued services as well as on the resource usage of each service. Thus, 50 different service-resource mapping with different prices were created randomly. Each mapping consists of 20 different forecast dataset and 100 demand dataset. One forecast dataset was used for five different demand scenarios. The data analysis is based on 5000 revenue outcomes for one comparison (e.g. 3×3 with DCR=1.2), which is sufficient for a sound statistical analysis. Moreover, various DCR settings can influence the revenue outcome. Thus, the different settings for DCR vary between 1.1 and 1.8. (see Section 'Simulation Process and Hypotheses'). The DCR is usually set for every resource individually.

Table 3 shows the difference between CBPP and CEC-S in yielded revenue. Almost all these cases are highly significant ($p < 0.001$) or significant ($p < 0.01$). However, in the settings with few resources and services (e.g. 3×3) no significant increase can be proven except for DCR=1.8. In most cases the revenue increases with higher DCR.

Table 3: Results for Hypothesis 1: Revenue difference between CBPP and CEC-S or DLP-S, respectively (values in italic denote significance at the level of at p = 0.01.)

	CEC-S					DLP-S				
DCR	1.1	1.2	1.4	1.6	1.8	1.1	1.2	1.4	1.6	1.8
3×3	1.7%	1.6%	2.2%	2.2%	*3.5%*	0.8%	1.2%	1.4%	0.9%	2.1%
4×4	*2.8%*	*4.0%*	*5.2%*	*4.7%*	*5.2%*	*2.8%*	*4.4%*	*5.0%*	*6.5%*	*6.3%*
4×5	*2.9%*	*3.5%*	*6.4%*	*5.5%*	*6.4%*	2.2%	*3.8%*	*5.4%*	*5.9%*	*6.9%*
4×7	2.5%	*4.5%*	*6.5%*	*6.6%*	*6.9%*	*3.3%*	*4.7%*	*5.8%*	*6.3%*	*7.3%*
4×10	*3.1%*	*3.8%*	*4.6%*	*4.8%*	*5.2%*	*2.7%*	*3.6%*	*4.2%*	*5.1%*	*6.8%*
6×8	*3.0%*	*4.2%*	*5.6%*	*6.3%*	*6.6%*	*2.9%*	*4.6%*	*5.2%*	*5.3%*	*6.4%*
10×10	*3.2%*	*3.7%*	*4.9%*	*4.6%*	*5.1%*	*3.0%*	*5.1%*	*7.3%*	*8.6%*	*7.9%*

A DCR above 2 decreases the revenue significantly. Comparing the outcome between DLP-S and CEC-S, DLPS seems to outperform CEC-S in the small settings. For 3×3 the revenue difference between CBPP and CEC-S is always higher than between CBPP and DLP-S. DLP-S performs well for small settings. Even for DLP-S, there is no significant revenue increase for the 3×3 setting, if CBPP is applied. In general, the small setting provides a worst case scenario. Since the runtime is short, heuristics are not relevant for these cases. Furthermore, such small settings are very uncommon. For larger settings CBPP can increase the revenue significantly.

The analysis above focuses on the general outcome of CBPP. Different settings for DCR, for genetic parameters and for service-resource mapping were chosen. The dependency between the various parameters of CBPP has to be scrutinized. For the following hypotheses an Analysis Of Variance (ANOVA)[6] was conducted.

Table 4: Results for Hypothesis 2

(a) Small price changes of a specific service affect revenue difference between CBPP and CEC-S

	i=1	*i*=2	*i*=3
i=1	X	X	X
i=2	p < 0.01**	X	X
i=3	p < 0.01**	p=0.43	X
i=4	p=0.13	p=0.02*	p=0.4

(b) Small price changes of a specific service affect revenue difference between CBPP and DLP-S

	i=1	*i*=2	*i*=3
i=1	X	X	X
i=2	p < 0.01**	X	X
i=3	p < 0.01**	p=0.27	X
i=4	p=0.07*	p < 0.01**	p=0.2

Hypothesis 2 is a negated phrase. It is assumed that small changes in the price of a service do not have an impact on the revenue (alternative hypothesis). In other words, applying CBPP to different price setting will lead to similar results in all runs. Table 4 shows the results for a 4×4 setting, when only small increase or decrease of one specific price changed the outcome of CBPP and CEC-S or DLP-S, respectively. There is a significant difference, since certain services are accepted by one service more than by others. If prices change, CBPP seems not always to outperform other algorithms, because in some cases the improvement is less than 1%. For example, price changes for $i = 4$ does not significantly differ from the original setting. However, a marginal difference in the price of $i = 2$ had a big impact on the outcome of CBPP, while the price for CEC-S was quite stable compared to the original

[6] A Levene test proved the homoscedasticity of the statistical population.

setting. Consequently, prices for different Cloud services do not have the same impact on the outcome, but they can have an impact. Thus, a sensitivity analysis is required to understand the dependency.

As outlined in Section 'Genetic Algorithm' the parameters of CBPP influence the revenue. In particular, the upper bounds are in focus of this analysis after presenting the impact of the population size and the evolution steps. Similar to hypothesis 2, hypothesis 3 is formulated in a negated way. The results in Table 5 show that at least for one case there is a significant deviation in the revenue over all runs. Consequently, the choices of upper and lower bounds affect the outcome, thus the alternative cannot be corroborated. Especially, for bounds with low values, there is a high variance in the outcome and therefore the probability of receiving near-optimal solution decreases. The objective function is not flexible enough to identify good revenue results. Hence, in this case a sensitivity analysis is necessary as well to understand why a certain bounds should be preferred. The bounds should be neither too low nor too high. Low bounds will not enable a correct adaptation of a and b. Too high bounds with a low population size result in scattered values for these parameters and consequently prevent the identification of near-optimal solutions. The results in table 5 are created for a 4×4 setting with DCR=1.4 and a population size of 9. Similar results were achieved for other fare classes and settings as well.

The results of Hypotheses 2 and 3 emphasize the necessity for further analysis to determine how these variables affect the revenue. Thus, a sensitivity analysis is presented in the following section to explore the dependency and to elicit the cases, where CBPP performs worse than other algorithms.

Table 5: Results for Hypothesis 3: Upper bounds affect revenue

	$\alpha_U = \beta_U = 2$	$\alpha_U = \beta_U = 3$	$\alpha_U = \beta_U = 5$
$\alpha_U = \beta_U = 3$	$p < 0.001$***	X	X
$\alpha_U = \beta_U = 5$	$p < 0.01$**	$p = 0.06$*	X
$\alpha_U = \beta_U = 8$	$p < 0.01$**	$p = 0.08$*	$p = 0.12$

Sensitivity Analysis

The sensitivity analysis is necessary to understand the impact of different parameters in the simulation setting. A common and reasonable approach is to fix all parameters except the examined one. The main parameter is the bid price. The bid price is calculated by the algorithms and thus influenced by many other parameters. The goal is to identify the best bid prices to approximate the optimum accurately.

The genetic upper bounds define the search space to receive good values for α_h and β_h of every resource h. These parameters have been analyzed by selecting seven representative bound settings for each service-resource mapping. The average revenue of CBPP, the standard deviation over all simulation runs and the revenue difference to CEC-S are used as a metric to understand the dependency. The values for α_U and β_U varied from 1 to 30. For the price range chosen for this simulation, a bound for $\alpha_U = \beta_U = 10$ yields the highest revenue with the lowest deviation. As assumed low bounds do not have a high standard deviation, but the revenue is lower than in the best case. High values for α_h and β_h can decrease the revenue on average compared to $\alpha_U = \beta_U = 10$, since the standard deviation is higher than in the best case. These statements are valid for all the tested settings. In general, there is a difference of about 1% in revenue between the worst case and the best case. According to van Ryzin and Vulcano (2008) this is already a significant delta. The volatility of the revenue (standard deviation) can increase up to 40% between the best case (here usually $\alpha_U = \beta_U = 10$) and the worst case ($\alpha_U = \beta_U = 20$). A countermeasure for avoiding volatility for high bounds is to increase the population size and the evolution steps to achieve better values on average. Then, the probability of finding appropriate values is higher, but the runtime increases as well (see Section 'Genetic Algorithm').

Providers in some domains change prices frequently. They are forced to do this due to internal (e.g. increasing cost) or external (e.g. competition) factors. However, price variation has a great impact on the decision policy and the revenue. It is important to understand the dependency between prices and other factors. In the previous section, it was shown that small price changes already lead to significant revenue gain or loss. A first step is to incrementally vary the prices for each service and to monitor when the revenue significantly drops.

Table 6: Price variations for randomly selected scenarios

	i=1		*i*=2		*i*=3		*i*=4	
Basic price	**16.2**	**rev diff**	**18.9**	**rev diff**	**20.5**	**rev diff**	**33**	**rev diff**
Variation 1-4	11	5.4%	16.5	-0.2%	**21**	**4.9%**	23	4.5%
Variation 5-8	12	5.3%	17	0.3%	**22**	**1.5%**	24	4.9%
Variation 9-12	13	4.5%	17.5	0.4%	23	1.%	35	4.7%
Variation 13-16	14	4.5%	**18**	**0.4%**	24	0.6%	36	4.2%
Variation 17-20	15	4.6%	**18.5**	**5.0%**	25	0.9%	37	4.4%
Variation 21-24	**16**	**5.1%**	19.5	4.9%	26	0.6%	38	4.3%
Variation 25-28	**17**	**0.6%**	20	4.8%	27	0.7%	39	4.3%

Table 6 gives an overview on the price changes. Note that this table represents 28 variations, since only one price is changed at a time to determine the effects of changes accurately. The prices are changed in small steps to understand the dependency of the price setting and the prices of each service do not overlap with the other services. The critical cases are shown in bold in the table. In these cases, there is a significant jump in the revenue between two selected variation. For example, the price change from 16 (variation 21) to 17 (variation 25) drops the revenue by 4.5%. With a low price for service $i = 1$ the revenue is not changing significantly. A lower price than the randomly chosen price 16.2 does not have an impact on the revenue difference between CBPP and CEC-S[7]. However, a higher price decreases the value by more than 4%. Here, the price for service $i = 1$ was increased to 17 (variation 25).

Table 7: Definition of basic scenario for sensitivity analysis and capacity occupation

(a) Service-resource mapping and price in the basic scenario

	i=1	*i*=2	*i*=3	*i*=4
h=1	2	4	8	6
h=2	7	5	4	4
h=3	4	9	4	3
h=4	4	3	4	9
Price	16.2	18.9	20.5	33

(b) Capacity occupation for specific scenarios in variation 21 and 25 at the end of the period

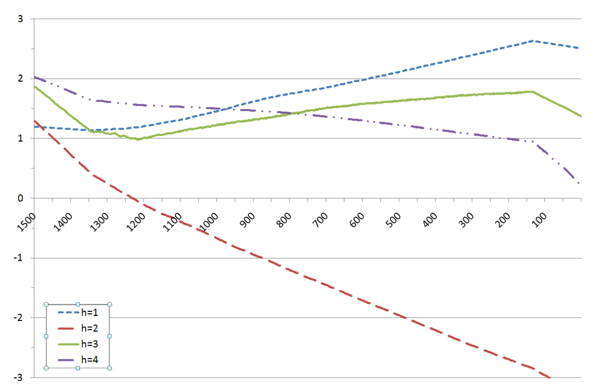
	Variation 21	Variation 25
h=1	99.9%	99.7%
h=2	75.5%	84.6%
h=3	85.4%	81.1%
h=4	91.2%	99.9%
Average	88.8%	91.3%

Comparing the outcome from CBPP and CEC-S the revenue of CBPP is similar in both cases. Variation 25 leads to a slightly higher outcome by 1% due to the higher price of service $i = 1$. CEC-S shows a significant increase in the revenue (in some settings up to 8%). Both algorithms are based on forecasts. While CBPP is relatively robust against forecast errors, CEC-S had problems to compensate this loss. In variation 21, 10% fewer requests for service $i = 4$ arrived than the forecasted value. Since, CEC-S has reserved capacity for the expensive service $i = 4$, other service requests were rejected. Especially, in the beginning of the period, more service $i = 1$ were rejected in

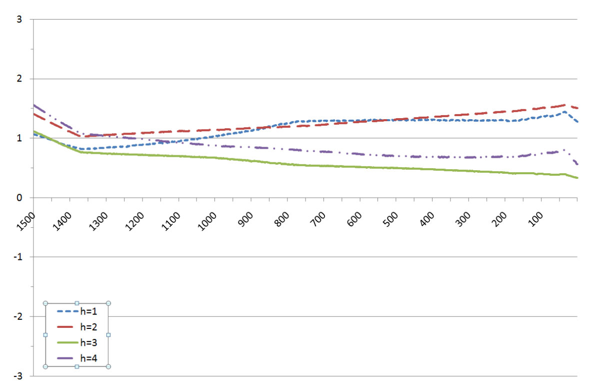
[7] The comparison with DLP-S has been disregarded, since it showed a similar behavior to CEC-S for the first analysis.

variation 21 than in variation 25. A total bid price of 16.4 at the beginning resulted in declining service $i = 1$ requests during the first quarter of the period, when most of the requests for service $i = 1$ arrive. The higher price for service $i = 1$ in variation 25 caused better bid prices and thus a better acceptance rate. While in variation 21 the scarce resource was $h = 1$, in variation 25 resource $h = 4$ was the bottleneck (Table 7b). Resource $h = 4$ mainly used by service $i = 4$ and resource $h = 1$ by service $i = 3$, which reflects the lower acceptance rate of service $i = 3$ in order to accept more service $i = 4$ requests.

Both services compete for resource $h = 1$. Hence, this resource is rarely available in both cases. CBPP generally accepted less service $i = 4$ requests and focused more on service $i = 1$ and $i = 2$. Although the revenue discrepancy was not significant in variation 25, CBPP yielded a higher revenue in both cases than CEC-S[8]. For variation 3 and 7 similar interpretations apply.



(a) Resource bid prices of variation 14 with price for *i* = 2: 18



(b) Resource bid prices of variation 18 with price for *i* = 2: 18.5

Figure 4: Bid price development over time for each resource (4×4 setting)

The case of service $i = 2$ outlines another important aspect of CBPP. Variation 14 and 18 are distinguished by the prices for service $i = 2$ with a discrepancy of 0.5. This small change leads to an average decrease of revenue of more than 4% (Table 6). Analyzing several runs, a typical pattern recurs. Figure 4 depicts the development of the bid prices for each resource in the 4×4 setting for CBPP. The resource $h = 2$ plays a crucial role for the outcome of CBPP. It is the most important resource for service $i = 1$ with a resource consumption of 7 (Table 7a). Although service $i = 2$ needs more capacity from resource $h = 3$, it still requires the second highest amount of resource $h = 2$. The price of service $i = 2$ in variation 14 is closer to the price of service $i = 1$ than in variation 18. To accept service $i = 2$ the bid prices on the resource level have to be adapted appropriately. In variation 18 the bid prices are set almost equally and behave in a similar way over time (Figure 4b). Variation 14 shows a difference in the curve progression. The bid price for resource $h = 2$ decreases continuously, since service $i = 2$ needs to be accepted, while other bid prices increase (Figure 4a). This indicates a low α and a high β for resource $h = 2$, while the others have a slightly higher α than β. Hence, more service $i = 2$ requests can be accepted. However, due to the proximity of the prices for service $i = 1$ and $i = 2$, more service $i = 1$ requests are accepted in variation 14 than in variation 18. Consequently, less capacity is available for service $i = 3$ and $i = 4$. A decrease in the price for service $i = 1$ from 16.2 to 15 and keeping the other prices as in variation 14 increases the revenue by 2%. Thus, prices should not be set too close to each other in order to improve the ability of the genetic algorithm to find appropriate bid prices for the resources. This is a disadvantage of the linear dependency of the bid prices. They cannot be adapted flexibly, since they depend on the incoming request and the decision of the previous time slots.

[8] Although the analyzed scenarios for comparing both cases were selected randomly, it was considered to have similar request scenarios and forecasts, if available, to isolate the scenario from other effects.

Conclusion

Providers can implement CBPP to manage revenue and resource utilization more efficiently, when bid prices for resources cannot be updated between two or more incoming requests. Usually, the service-resource mapping is relatively fixed for their settings. They have a portfolio of standard services. Thus, changes in the service-resource mapping are only useful, when providers often renew their portfolio by introducing new innovative services. Answering the research question, the results above have proved that CBPP can outperform standard algorithms by up to 6% on average. However, it highly depends on the applied scenario. First of all, the provider has to be aware of the dependencies between services and resources. They must be able to technically map services to the consumed resources. The service-resource mapping has a great impact on the revenue.

The analysis of runtime in Section 'Genetic Algorithm' provides a worst case scenario, where the algorithm is executed on a single machine. Genetic algorithms are capable of generating parallel threads to improve the runtime significantly. Hence, the runtime of CBPP will be much better, if the algorithm is executed in a cluster or Cloud. The algorithm also enables the improvement of the revenue by selecting a higher population size and evolution steps, since the runtime can be reduced due to parallelization. Furthermore, Cloud service prices are set by providers. They have to consider the current market prices and the willingness to pay of the consumers for their services. Changes in prices will change the bid price for every resource as well and thus influence the accept/reject decision of the provider. In some cases, CEC-S can outperform CBPP, if the prices are set too close. Providers can adjust the prices or the resource consumption of the services to avoid this issue. Service prices can also help to calculate the optimal reservation prices, if services are auctioned. For example, Amazon has three different pricing models. By defining the service price internally as reservation price, Amazon can calculate how many services may be offered as Spot instances, Reservation instances or On-Demand instances.

In this paper, it was assumed that the demand follows a Non Homogeneous Poisson Process (NHPP) and expensive services are likely to be requested later in time than inexpensive services. The provider has to analyze the incoming requests and model the forecasts accurately. Since service-resource mapping often cannot be changed instantly and demand can only be affected indirectly, other parameters play a more important role. Another aspect is the capacity for Cloud services, which is usually relatively fixed. Providers can virtually limit the capacity for certain services to guarantee a promised service level. This approach allows them to calculate the risk of accepting a predefined amount of services and still fulfill the SLA. Obviously, the virtual limit can be extended based on previous experience or if providers are risk takers. Moreover, the DCR depends on the demand and the capacity. By setting different virtual limits, this parameter can limit the resource usage to make it scarce, which will have an impact on the service and thus its price.

References

Adelman, D. 2007. "Dynamic Bid Prices in Revenue Management," *Operations Research* (55:4), pp. 647–661.

Banker, R. and Kauffman, R. 2004. "The Evolution of Research on Information Systems: A Fiftieth-Year Survey of the Literature in *Management Science,*" *Management Science* (50:3), pp. 281–298.

Bertsimas, D. and de Boer, S. 2005. "Simulation-based booking limits for airline revenue management," *Operations Research* (53:1), pp. 90–106.

Bertsimas, D. and Popescu, I. 2003. "Revenue Management in a Dynamic Network Environment," *Transportation Science* (37:3), pp. 257–277.

Bichler, M. and Setzer, T. 2007. "Admission control for media on demand services," *Service Oriented Computing and Applications* (1:1), pp. 65–73.

Bitran, G. and Caldentey, R. 2003. "An Overview of Pricing Models for Revenue Management," *Manufacturing & Service Operations Management*, (5:3), pp. 203–229.

Bixby, R. 2002. "Solving Real-World Linear Programs: A Decade and More of Progress," *Operations Research* (50:1), pp. 3–15.

Dube, P., Hayel, Y., and Wynter, L. 2005. "Yield management for IT resources on demand: analysis and validation of a new paradigm for managing computing centres," *Journal of Revenue and Pricing Management* (4:1), pp. 24–38.

Frank, M., Friedemann, M., and Schroder, A. 2008. „Principles for simulations in revenue management," *Journal of Revenue and Pricing Management* (7:1), pp. 7–16.

Gallego, G. and van Ryzin, G. J. 1994. "Optimal dynamic pricing of inventories with stochastic demand over finite horizons," *Management Science* (40:8), pp. 999–1020.

Glover, F., Glover, R., Lorenzo, J., and McMillan, C. 1982. "The passenger mix problem in the scheduled airlines," *Interfaces* (12:3), pp. 73–79.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, USA.

Gosavi, A., Ozkaya, E., and Kahraman, A. F. 2007. "Simulation optimization for revenue management of airlines with cancellations and overbooking," *OR Spectrum* (29:1), pp. 21–38.

Holland, J. 1975. *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, USA.

Kimes, S. E. 1989. "Yield Management: A Tool for Capacity-Constrained Service Firms," *Journal of Operations Management* (8:4), pp. 348–363.

Kimms, A. and Mueller-Bungart, M. 2007. "Simulation of stochastic demand data streams for network revenue management problems," *OR Spectrum* (29:1), pp. 5–20.

Klein, R., 2007. "Network capacity control using self-adjusting bid-prices," *OR Spectrum* (29:1), pp. 39–60.

Möller, A., Römisch, W., and Weber, K. 2008. "Airline network revenue management by multistage stochastic programming," *Computational Management Science* (5:4), pp. 355–377.

Nair, S. and Bapna, R. 2001. "An application of yield management for Internet Service Providers," *Naval Research Logistics* (48:5), pp. 348–362.

Phillips, R. 2005. *Pricing and Revenue Optimization*, Stanford Business Books, Stanford, USA.

Rechenberg, I. 1973. *Evolutionsstrategie. Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann Holzboog, Berlin, Germany.

Secomandi, N., Abbott, K., Atan, T., and Boyd, E. A. 2002. "From Revenue Management Concepts to Software Systems," *Interfaces* (32:2), pp. 1–11.

Smith, B. and Penn, C. 1988. "Analysis of alternative origin-destination control strategies," in *Proceedings of the 28th Annual AGIFORS Symposium*, pp. 123–144.

Subramanian, J., Jr, and Lautenbacher, C. J. 1999. "Airline Yield Management with Overbooking, Cancellations, and No-Shows," *Transportation Science* (33:2), pp. 147–167.

Sulistio, A., Kim, K. H., and Buyya, R. 2008. "Managing Cancellations and No-Shows of Reservations with Overbooking to Increase Resource Revenue," in *Proceedings of the Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID)*, Washington, DC, USA, pp. 267–276.

Talluri, K. T. and van Ryzin, G. J. 1998. "An Analysis of Bid-Price Controls for Network Revenue Management," *Management Science* (44:11), pp. 1577–1593.

Talluri, K. T. and van Ryzin, G. J. 1999. "A randomized linear programming method for computing network bid prices," *Transportation Science*, (33:2), pp. 207–216.

Talluri, K. T. and van Ryzin, G. J. 2004. *The Theory and Practice of Revenue Management*, Springer, Berlin, Germany.

Urgaonkar, B., Shenoy, P., and Roscoe, T. 2002. "Resource Overbooking and Application Profiling in Shared Hosting Platforms," in *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, pp. 239–254.

van Ryzin, G. and Vulcano, G. 2008. "Computing Virtual Nesting Controls for Network Revenue Management Under Customer Choice Behavior," *Manufacturing and Service Operations Management* (10:3), pp. 448–467.

Weatherford, L., Bodily, S., and Pfeifer, P. 1993. "Modeling the Customer Arrival Process and Comparing Decision Rules in Perishable Asset Revenue Management Situations," *Transportation Science* (27:3), pp. 239–251.

Weatherford, L. and Belobaba, P. 2002. "Revenue Impacts of Fare Input and Demand Forecast Accuracy in Airline Yield Management," *The Journal of the Operational Research Society* (53:8), pp. 811–821.

Williamson, E. 1992. *Airline network seat control*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, USA.

Wilson, R. 1995. "Nonlinear pricing and mechanism design," in *Handbook of Computational Economics (Vol. 1)*, Elsevier, pp. 253–294.