European Journal of Operational Research 244 (2015) 637–647

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Innovative Applications of O.R.

Revenue management for Cloud computing providers: Decision models for service admission control under non-probabilistic uncertainty

Tim Püschel [a], Guido Schryen [b,*], Diana Hristova [b], Dirk Neumann [a]

[a] Chair for Information Systems Research, Albert-Ludwigs-Universität Freiburg, Platz der Alten Synagoge, 79108 Freiburg, Germany
[b] Management Information Systems, Universität Regensburg, Universitätsstr. 31, 93053 Regensburg, Germany

ARTICLE INFO

Article history:
Received 4 March 2014
Accepted 15 January 2015
Available online 30 January 2015

Keywords:
Admission control
Informational uncertainty
Revenue management
Cloud computing

ABSTRACT

Cloud computing promises the flexible delivery of computing services in a pay-as-you-go manner. It allows customers to easily scale their infrastructure and save on the overall cost of operation. However Cloud service offerings can only thrive if customers are satisfied with service performance. Allowing instantaneous access and flexible scaling while maintaining the service levels and offering competitive prices poses a significant challenge to Cloud computing providers. Furthermore services will remain available in the long run only if this business generates a stable revenue stream. To address these challenges we introduce novel policy-based service admission control models that aim at maximizing the revenue of Cloud providers while taking informational uncertainty regarding resource requirements into account. Our evaluation shows that policy-based approaches statistically significantly outperform first come first serve approaches, which are still state of the art. Furthermore the results give insights in how and to what extent uncertainty has a negative impact on revenue.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Cloud computing denotes a computing model that enables ubiquitous and on-demand network access to a shared pool of configurable resources, which can be rapidly provisioned and released with minimal management effort (Mell & Grance, 2009). Resources typically refer to IT infrastructures, platforms or software, which are provided as services on a per-usage basis. While Cloud computing enjoys wide popularity for users, Cloud providers face fierce competition that has led to an erosion of the margins from $1 per CPU/hour to just a few cents (Amazon, 2015).

As the revenue side is reduced by the competition, Cloud providers need to minimize their operation costs to remain competitive. Reducing the operation costs is, however, quite difficult as the workload is highly uncertain, as the exact distribution of job arrivals is unknown. To the Cloud providers' dismay, most of modern applications are online services that require immediate processing (e.g. Outlook.com, Dropbox, GDrive). As a consequence of the immediacy traditional batch processing is not applicable; nor is rescaling of the Cloud possible due to the set up time of adding additional resources to the Cloud. Thus, Cloud providers need to manage the trade-off between maintaining excess resources as a buffer and operating the Cloud with minimal resources. While the former strategy assures to meet all quality of service assertions even if uncharacteristically many jobs arrive, the latter strategy clearly optimizes on the operating costs potentially leaving customers unsatisfied with unmet quality of service assertions.

An effective Admission control that determines which job requests are processed, may alleviate this trade-off: If the workload exceeds a critical threshold, the Cloud is susceptible to fail the quality of service assertion. In extreme cases this can even result in a system overload compromising the stability of the entire system. Admission control can act as an instrument for Cloud providers to control the exact number of jobs that are confronting the Cloud in the short run. The admission control decision is, however, hampered by the uncertain jobs arrivals and, in addition, by resource uncertainty. Resource uncertainty accounts for the fact that it is literally impossible to predict the exact resource requirements necessary to meet the quality of service assertions, due to the involved complexity in the underlying IT infrastructure (cf. Kounev, Nou, & Torres, 2007).

The field of revenue management has developed many solutions to related problems in other as well as related industries. Those solutions, however, are not applicable as they do not account for the peculiarities of online applications run in Clouds, such as resource uncertainty. Cloud systems are too complex, to obtain good predictions on required resources for a certain job. Thus, resource uncertainty

* Corresponding author. Tel.: +499419435634; fax: +499415435635.
E-mail addresses: tjpueschel@gmail.com (T. Püschel), guido.schryen@wiwi.uni-regensburg.de (G. Schryen), diana.hristova@wiwi.uni-regensburg.de (D. Hristova), dirk.neumann@is.uni-freiburg (D. Neumann).

http://dx.doi.org/10.1016/j.ejor.2015.01.027
0377-2217/© 2015 Elsevier B.V. All rights reserved.

results in a tremendous resource overestimation (Caglar & Gokhale, 2014). We extend the revenue management literature by introducing an admission control scheme that is tailored toward the needs of services requiring instantaneous access. We cope with the complicated issue of resource uncertainty by applying fuzzy set theory to revenue management. Our work incorporates feedback and significantly extends models presented in Püschel and Neumann (2009) and Püschel, Schryen, Hristova, and Neumann (2012).

Our work contributes to the literature by suggesting and testing various service admission control policies using extensive simulations. We show that the admission problem can be solved in polynomial time, which is prerequisite for instantaneous decisions. Furthermore, we show that our policies succeed in satisfying the technical requirements stemming from Cloud computing while contemporaneously securing additional revenue for the Cloud provider. In our analysis, we demonstrate how uncertainty can affect the tradeoff between the revenue base and the service request acceptance rate.

The remainder of this paper is structured as follows: In the second section, we discuss the determinants of the "Cloud Admission Control Problem". Subsequently, we review related work in Section 3 based on these determinants. In Section 4, we propose decision models that account for various real-time admission control policies of Cloud providers that focus uncertainty regarding resource demands. The fifth section comprises an evaluation, where we test and compare our models with respect to their attractiveness in terms of revenue and service request acceptance rate. In Section 6, we provide managerial implications. The paper concludes with a summary and an outlook on new research avenues.

2. Determinants of the Cloud Admission Control Problem

Before we formulate the Cloud Admission Control Problem (henceforth CACP), it is useful to state the characteristics of Cloud applications representing the requirements on the CACP. In total we account for seven different characteristics:

Production inflexibility: Providers usually maintain a Cloud infrastructure, which consists of a fixed amount of resources (e.g. servers). While resources can easily be added to the Cloud infrastructure, it takes some time until the Cloud is reconfigured. This reconfiguration delay dictates the Cloud infrastructure to be fixed in the short run. In case of online job processing the Cloud infrastructure cannot be adapted to the job admission decision.

Perishability: Resources managed as Cloud offer computation and storage capacities. If the Cloud is not (fully) used it is not possible to store the excess capacity for later consumption.

Real-time decision-making: Due to the trend toward online processing, Cloud providers face the challenge to control job admission in real time. Effective mechanisms need to work in online and real-time scenarios as well. With respect to the CACP this translates into the requirement that the admission control mechanism needs to be of very low computational cost to be computationally tractable.

Limited/no information on future demand/jobs: Due to the "pay-as-you-go environment", Cloud service providers have to serve customers with lack of information (i.e. non-clairvoyant). In contrast to machine scheduling problem where relevant data are available (e.g. distributions of job arrivals and job characteristics), Cloud providers have only vague information on (i) the job arrival rate, (ii) the exact resource need of jobs, and on (iii) the customer's willingness to pay.

Non-probabilistic uncertainty of required resources: Cloud customers typically provide estimates on their jobs' resource requirements. As customers usually do not utilize the estimated resources for the entire job lifecycle, Cloud customers can exploit the flexibility of Cloud infrastructures by devoting excess resources to other customers. The uncertainty in accurate resource prediction stems from two sources: (i) the customer's ability to make accurate predictions and (ii) the type of the job. The Cloud provider may gather information on how well the customers calculate their predictions. Customers tend to overestimate their job resource requirements to have a buffer in case the job needs more resources. In practice, customers just use a fraction of their requested and consequently allocated resources. This claim can be verified by observing the actual usage of Google's data centers. Apparently, the used resources are way lower than the allocated resources (Reiss, Tumanov, Ganger, Katz, & Kozuch, 2012). The Cloud provider can exploit this buffering behavior, as the unused resources can be used for other jobs, increasing the overall utilization of the Cloud. The information on the customers is, however, limited, if Cloud providers offer their services to the public, as there will be a frequently varying customer base. As such, it is impossible to have probabilistic information on the accuracy of the resource predictions. The second uncertainty in resource prediction stems from the type of the job that is directed to the Cloud. For example, for routine jobs (e.g. weekly accounting or controlling tasks of customers), the required resources may be exactly predicted drawing on prior observations. For jobs that are rarely conducted (e.g. data mining jobs) the required resources largely depend on the analyzed data. Predicting the job resources for those jobs is naturally very difficult.

Best-effort vs. priority-based processing: Cloud providers typically serve two different customer groups, which can be distinguished with respect to their quality of service (QoS) requirements (Buyya, Yeo, Venugopal, Broberg, & Brandic, 2009). While some customers accept service delivery on a best-effort base, i.e. without guarantee that the job is executed, other customers may require a guarantee that the job is executed according to the contracted service levels. The former customer group pays less for their service execution, while the latter group will only pay in the case the service levels are met. Clearly, Cloud providers favor customers with contracted service levels (henceforth gold clients) over best-effort customers, who are processed only when resources are left.

Resistance to strategic behavior: Since customers also strive to maximize their utility, it is appealing for them to act strategically in order to gain advantages. Basically customers have three ways in which they can adapt their actions. Firstly they might be able to shift their demand in time. They might also be able to split jobs into several smaller ones or merge several jobs into one big one. Their last option is to vary their price bid. Certain strategic behavior can lead to a significant reduction in revenue of the Cloud provider. Furthermore, it can reduce customer satisfaction. Customers might be unwilling to shoulder the additional effort necessary for strategic behavior.

3. Related work

Quite recently, the CACP has gained in attention by the service literature. These approaches are, however, driven by the technology and not founded by management literature. Obviously, the determinants of the Cloud Admission Control Problem suggest the use of revenue management mechanisms. However, revenue management has been developed primarily for the airline industry—as such, not all requirements of Cloud services are met. Thus, we review both literature streams and evaluate them with regard to the CACP.

3.1. CACP in IT-service environments

The first stream stems from computer science and focuses on technical aspects. Ferguson, Nikolaou, Sairamesh, and Yemini (1996) discuss the general applicability of economic theories to resource management. Being a primer, their results are general, but not specifically associated with actual implementations. An interesting approach to realize high service levels and end-to-end QoS is the Globus Architecture for Reservation and Allocation (Foster et al., 1999). This approach uses advance reservations to guarantee QoS. A related approach to

achieve autonomic QoS aware resource management is based on online performance models (Kounev et al., 2007). They introduce models that are able to predict the impact of the acceptance of an incoming job on the performance of the system and the ability to fulfill the service-level agreements of all jobs. Elements of client classification, such as price discrimination based on customer characteristics, have been addressed in Newhouse, MacLaren, and Keahey (2004). The authors—albeit technically motivated—move in the direction of applying methods from revenue management applied in other industries. However, they do not consider other discrimination factors, such as priority on job acceptance or higher quality of service. Aiber et al. (2004) present an architecture for autonomic self-optimization based on business objectives. Their focus is on the technical implementation of economic principles, but not on the employed principles. Boughton, Martin, Powley, and Horman (2006) present research on how workload class importance can be considered for low-level resource allocation. They focus on competing workloads in databases and investigate what business policies can be used to efficiently allocate resources.

3.2. Revenue management and the CACP

Another stream aims at adapting more sophisticated concepts from revenue management and admission problems in service computing environments. The use of revenue management concepts by Internet Service Providers was researched by Nair and Bapna (2001). They consider the decision to accept or reject customers but did not take different service types or advanced reservation into account. Yeo and Buyya (2004) show an approach for a pricing function depending on a base pricing rate and a utilization pricing rate. The idea behind utilization based pricing is that when utilization is high, demand for the resources is high as well so that customers are willing to pay higher prices for resources. If utilization is low, lower prices are charged to attract more customers (Bitran & Caldentey, 2003). One of the first papers that analyze more sophisticated revenue management concepts for cluster systems was published by Dube, Hayel, and Wynter (2005). The model offers one resource for different prices. Assuming that customer behavior follows a logit model, the authors analyze an optimization model for a small number of price classes and provide numerical results. Maglaras and Meissner (2006) discuss dynamic pricing strategies for revenue management problems where a company owns a fixed capacity of a resource which can be used to deliver different products. They show that this problem can be simplified to a form where the firm controls the aggregate rate at which resources are used. Eren and Maglaras (2009) study a monopoly pricing scenario for a seller with limited market information. They consider different demand learning capabilities for sellers. Their findings suggest that policies where the pricing policy is not continuously updated and re-optimized perform well. This makes them a good alternative for "active" models, which may require more rigid assumptions. Anandasivam and Weinhardt (2010) present a policy-based decision model for Cloud providers. However, the bid price mechanism on which this model is based requires assumptions on workload and demand patterns. For example, the provider has to be able to perform a sufficiently accurate demand forecast: each service request for the same service requires the same exact amount of capacity (group bookings are not possible). Another stream of literature incorporates the uncertainty in admission control into the resource management. Ramalho (1998) examines the application of fuzzy logic to the CAC (Connection Admission Control) traffic control function in ATM (Asynchronous Transfer Mode) broadband communication networks. Based on observations of the traffic in the ATM link, the paper derives a Fuzzy Logic Based CAC (FCAC) for the maximum cell loss ratio from adding a new connection to a given traffic scenario under uncertainty. An alternative approach to measurement-based admission control for multiclass networks with link sharing for "applications with ill specified traffic characteristics" using adaptively measured maximal rate envelopes is proposed in Qiu and Knightly (2001). By assuming that future packet arrivals will not exceed the past maximal rate envelopes they develop a method that among other indicators reflects the uncertainty of the prediction of future workloads and allows controlling important QoS parameters, such as loss probability.

3.3. Discussion

The related work covers different aspects of these research problems. Table A.4 (see online Appendix A) shows which of the main objectives, characteristics, and basic approaches are discussed in related work. However, the mechanisms are only applicable to the Cloud market to some extent and an overall decision model that accounts for the aforementioned characteristics for Cloud service providers is still missing. Furthermore, many of the mechanisms require rigid assumptions. For example, the revenue management mechanisms dealing with non-clairvoyance (see Table A.4) do not need the exact information about future jobs, but they do need forecasting models with sufficiently exact prediction. To address these issues we present novel service request admission control models in the following section.

4. Service request admission control models

In this paper, we consider service request admission control models that account for the characteristics discussed in the above section. They extend previous work (Püschel & Neumann, 2009; Püschel et al., 2012). We embed our models into a research framework that is structured along two dimensions: The first dimension distinguishes situations in which the Cloud service provider can expect to get reliable and crisp resource requirements from the customers (situation under certainty), from those situations where the Cloud service provider only has available vague estimates of the resources required (situation under uncertainty). In practice, this vagueness reduces the prediction quality of required resources. While the (un)certainty of required resources is exogenously given, in a second dimension Cloud service providers can decide on the applied job admission policy, which provides guidance for the decision on whether to accept or to reject an incoming job request, by taking into account profit- and service-orientation. The policy to apply is a decision variable and thus represents an endogenous component. We suggest three policies in this paper: first-come first-served (FCFS) policy (P1), dynamic pricing policy (P2), and a client classification policy (P3).

As our models suggested in this section are extensions and modifications of the well-known "knapsack problem", we briefly present and discuss the "knapsack problem". The need for modifications and extensions of the "knapsack model" lies in its unrealistic assumption of complete information: all incoming jobs including all their requirements need to be known by the service provider. This assumption is rarely met in a "pay-as-you-go" environment, as we face it in a Cloud service setting. Thus, approaches are necessary that address the need for making real-time decisions regarding the acceptance/rejection of incoming job requests under the goal of revenue maximization and in the absence of knowledge of later jobs and their resource requirements. We refer to these approaches as "online policies". All six models previously mentioned implement such online policies. In the second subsection, we present the (basic) model "first-come first-served under certainty", on which the other five models are based. It should be noticed that the concept of "certainty" does not target the knowledge of future jobs, but certainty regarding the real resource requirements of the job currently submitted. In the third subsection, we account for uncertainty (regarding required resources). We present the theoretical underpinning, fuzzy set theory, and propose the "first-come first-served under uncertainty" model. In the fourth subsection we present the four remaining models.

4.1. Model under perfect information

If we assume that the provider has perfect information and certainty about future events, it is possible to calculate the revenue maximizing solution. This means to ignore *non-clairvoyance* and *non-probabilistic uncertainty regarding the quality of customers' resource predictions* (see Section 2). The necessary information includes incoming service requests, prices, the exact resource requirements of each job, and capacity available in the future.

In this case, a simple instance of this problem is:

$$\max_{x} \sum_{j=1}^{|J|} x_j * fp_j \tag{O1}$$

$$\text{subject to: } \sum_{j=1}^{|J|} c_{jr}(t) * x_j \leq c_r(t) \quad \forall t \in T,\ \forall r \in R \tag{C1.1}$$

where T is the set of all regarded time slots; J is the set of available jobs; R is the set of all resource types; fp_j is the price paid for job j; x_j is a binary allocation variable indicating whether job j is accepted or rejected; $c_{jr}(t)$ is the capacity of resource type r required by job j in time slot t; and $c_r(t)$ is the total capacity available for resource type r during time slot t. (O1) is the objective function and represents the achieved revenue. The constraints (C1.1) are resource/capacity constraints over resources R and time slots T. They assure that not more capacity can be allocated than is available. The maximization problem can be formulated as a linear program. It is a generalization of the knapsack problem and thus NP-hard, and therefore it is computationally intractable for large problem instances (the proof can be found in online Appendix E). Beyond the computational challenge to solve problem instances optimally, we also face the problem that perfect information on incoming jobs is not available in practice. Thus, we suggest different "online policies", which can be executed in realtime.

4.2. First-come first-served policy under certainty

In the FCFS policy under capacity constraints an incoming job is accepted if and only if there is enough capacity available for all resources. A job that is rejected will not be served unless it will be resubmitted by a client with adapted time slots or/and adapted resource requirements. In this case, the resubmitted job is treated as a new one. This procedure applies to all of our suggested policies. The following mathematical formulation represents the **"first-come first-served policy under certainty"** model:

$$\max_{x} \sum_{j=1}^{|J|} \left(\frac{1}{2} * x_j\right)^j \tag{O2}$$

$$\text{subject to: } \sum_{j=1}^{|J|} c_{jr}(t) * x_j \leq c_r(t) \quad \forall t \in T,\ \forall r \in R \tag{C2.1}$$

The objective function (O2) represents the sequential nature of the policy by ensuring that an incoming job j is served if all required resources are available (C2.1). The reason for this is that not accepting job j cannot be compensated in terms of contribution to the objective value by accepting all future incoming jobs $(j+1), \ldots, |J|$. It should be noticed that in contrast to the objective function (O1) of the basic deterministic model (see p. 10) the FCFS nature of (O2) accounts for the fact that in practice information on future jobs is not known; the set J of considered jobs contains only already submitted jobs. The objective function (O2) is shared by all six service request admission control models in order to implement our main assumption that incoming jobs require real-time acceptance decisions. (C2.1) represents the capacity constraints which assure that not more capacity is allocated than is available for each resource type, such as CPU, storage and bandwidth, thus fulfilling the requirement of not degrading QoS due to acceptance of too many jobs. The same objective function and capacity constraint is used for all policies.

4.3. First-come first-served policy under uncertainty

While the idea of the FCFS policy is now introduced, it remains to explain how we model uncertainty in the FCFS policy (and in all other policies introduced later). When customers submit their jobs to Cloud providers together with their predictions on required resources, the impreciseness of these predictions is due to *lack of information*, *belief*, and *linguistic characterizations* (of the customers), which all are deemed some of the most important roots of uncertainty (Zimmermann, 2000). It should be noticed that these roots of uncertainty are particularly inappropriate for being addressed with probabilistic uncertainty theories, which are often based on historic data. Accounting for these roots of uncertainty, we select fuzzy set theory. A brief introduction into the main concepts of fuzzy set theory and fuzzy optimization is provided in online Appendix D.

The sound theoretical concepts of fuzzy sets and fuzzy arithmetic allow for flexibly extending the model under certainty. To this end we utilize the aforementioned concepts of fuzzy set theory by fuzzifying resource requirements of the customers' service requests and modelling these requirements with triangular fuzzy numbers. Thereby, we yield linear optimization models with crisp, binary decision variables, with a crisp objective function, and with fuzzified coefficients in the constraints. Such models are recognized as a specific type of fuzzy linear optimization model in the fuzzy mathematical programming literature. Baykasoğlu and Göçken (2008, Table 2) refer to this model type as "type 4" with non-fuzzy, binary (B) variables. Unfortunately, the operations research literature is silent on how to solve instances of this model type (Baykasoğlu & Göçken, 2008).

In the **"first-come first-served policy under uncertainty"** model, the objective function (O2) is the same as in the "first-come first-served policy under certainty" model. However, in the constraints the resource requirements are now fuzzy numbers denoted by $\widetilde{c}_{jr}$.

$$\sum_{j=1}^{|J|} \widetilde{c}_{jr}(t) * x_j \leq c_r(t) \quad \forall t \in T,\ \forall r \in R \tag{$\widetilde{C2.1}$}$$

4.4. Other models

Having introduced the models for the FCFS policies under certainty and under uncertainty, we now present the remaining four models structured along the policies.

4.4.1. Dynamic pricing

The dynamic pricing policy follows the key idea that when resources become scarce their prices increase. More specifically, it extends the FCFS policy by relating the price fp_j paid for a job j to resource-specific utilization levels: accepting a job j requires that, in each time slot $t \in T$ where job j runs, the average price paid per used time slot (fp_j/s_j) (s_j is the runtime of job j in terms of the number of timeslots) is not below a weighted sum of reservation prices, with the weights being the required capacities $c_{jr}(t)$ and the reservation prices (per unit) rp_k being the (utilization based) minimum prices at which the provider sells his resources. We assume that providers use a set of utilization levels $K = \{l_0, \ldots, l_{n-1}\}$ of their resources and assign a specific reservation price rp_k to level l_k. Both the utilization levels and the reservation prices are set by the Cloud provider and remain constant during short-term revenue management. They can be derived based on the provider's cost of resources and the degree of depreciation depending on the utilization level. If in a time slot t the utilization level $U_r(t)$ of a resource r exceeds level l_k, then the provider requires to get at least the reservation price rp_k for (one unit of) the

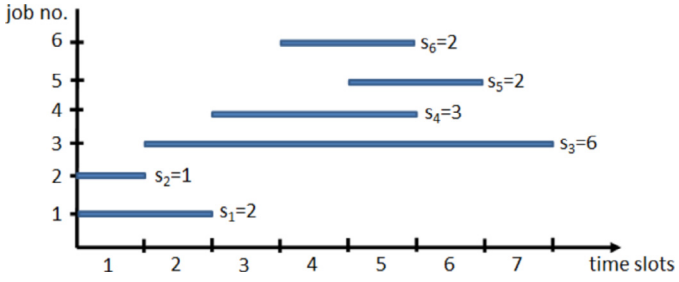


Fig. 1. Time slots of jobs.

respective resource and time slot. It should be noticed that while we distinguish different utilization levels in a time slot t for different resources, we do not provide for resource-specific reservation prices.

Dynamic pricing requires constraint (C2.2), which results to the following constraints of the **"dynamic pricing policy under certainty"** model:

$$\sum_{j=1}^{|J|} c_{jr}(t) * x_j \leq c_r(t) \quad \forall t \in T,\ \forall r \in R \tag{C2.1}$$

$$\sum_{r=1}^{|R|}\sum_{k=1}^{|K|}([H(U_r(t) - l_{k-1}) - H(U_r(t) - l_k)] * c_{jr}(t) * rp_{k-1} * x_j)$$

$$\leq \frac{fp_j}{s_j} \forall t \in T, \quad \forall j \in J \tag{C2.2}$$

The utilization level $U_r(t)$ for resource r and time slot t is defined by:

$$U_r(t) := \frac{1}{c_r(t)} * \sum_{j=1}^{|J|} c_{jr}(t) * x_j, \quad \forall r \in R, \forall t \in T \tag{4.1}$$

H is the discrete heaviside step function defined by $H(x) = 1$ if $x \geq 0$, $H(x) = 0$ otherwise. Its use ensures that for each resource r and for each time slot t only that summand is non-zero (or "activated") that corresponds to the respective utilization level.

Analogously to the FCFS policy, we derive the constraints of the corresponding model **"dynamic pricing policy under uncertainty"**.

$$\sum_{j=1}^{|J|} \widetilde{c}_{jr}(t) * x_j \leq c_r(t) \quad \forall t \in T,\ \forall r \in R \tag{$\widetilde{C2.1}$}$$

$$\sum_{r=1}^{|R|}\sum_{k=1}^{|K|}([\widetilde{H}(\widetilde{U}_r(t) - l_{k-1}) - \widetilde{H}(\widetilde{U}_r(t) - l_k)] * \widetilde{c}_{jr}(t) * rp_{k-1} * x_j)$$

$$\leq \frac{fp_j}{s_j} \forall t \in T, \quad \forall j \in J \tag{$\widetilde{C2.2}$}$$

Here the current utilization $\widetilde{U}_l(t)$ is now a fuzzy number, defined by

$$\widetilde{U}_r(t) := \frac{1}{c_r(t)} * \sum_{j=1}^{|J|} \widetilde{c}_{jr}(t) * x_j, \ \forall r \in R, \forall t \in T. \tag{4.2}$$

The discrete heaviside step function needs to be adapted so that it is defined over fuzzy numbers: $\widetilde{H}(\widetilde{x}) = 1$ if $\widetilde{x} \geq \widetilde{0}$, $H(\widetilde{x}) = 0$ otherwise, with $\widetilde{0}$ being the fuzzy number "zero".

4.4.2. Client classification

The third policy extends the FCFS policy by implementing client classification (strict priority policy) which helps improve customer satisfaction. The key idea of the policy is that a job is accepted only if it either submitted by an important customer, referred to as "gold customer", or if the current utilization level $U_r(t)$ does not exceed a fixed value l_c for all resources in all time slots. A Cloud provider classifies (known) customers as "gold customers" before jobs are submitted; the classification remains constant and is short-term but may be changed in the long run, e.g., based on a service-level agreement with the respective customer. In order to distinguish gold customers from others, we introduce parameter cc_j, which equals 1 if job j is submitted by a gold customer and 0 else. Constraint (C2.3) implements the described requirements. The constraints of the **"client classification policy under certainty"** model are given below:

$$\sum_{j=1}^{|J|} c_{jr}(t) * x_j \leq c_r(t) \quad \forall t \in T,\ \forall r \in R \tag{C2.1}$$

$$(1 - cc_j) * U_r(t) \leq l_c \quad \forall t \in T, \forall r \in R, \forall j \in J \tag{C2.3}$$

The constraints of the fuzzified model **"client classification policy under uncertainty"** look as follows:

$$\sum_{j=1}^{|J|} \widetilde{c}_{jr}(t) * x_j \leq c_r(t) \quad \forall t \in T,\ \forall r \in R \tag{$\widetilde{C2.1}$}$$

$$(1 - cc_j) * \widetilde{U}_r(t) \leq l_c \quad \forall t \in T, \forall r \in R, \forall j \in J \tag{$\widetilde{C2.3}$}$$

4.5. Numerical example

We provide a numerical example that shows how the suggested policies FCFS, dynamic pricing and client classification are applied (under certainty). The example includes six jobs (job numbers correspond to the sequence of submission) and seven time periods. For the sake of simplicity, each of the jobs requires one instance, i.e., a job of service type 1 requires 4 units of CPU, 4 units of storage and 16 units of bandwidth (cf. Table B.5 in online Appendix B). Fig. 1 shows the time slots of the jobs, Table 1 shows all remaining job data. The available resources remain constant over all seven time slots ($j = 1, \ldots, 7$): CPU: $c_1(t) = c_1 = 50$, storage: $c_2(t) = c_2 = 24$, bandwidth: $c_3(t) = c_3 = 32$.

The application of the FCFS policy leads to the acceptance of jobs 1, 2, 3 and 4, with an overall revenue of 7.8. Fig. 2a shows the allocated resources for each time slot based on the jobs that run in the respective time slots. Jobs 5 and 6 are not accepted as, for each of the jobs, in time slot 5 the capacity of resource 3 would be exceeded.

For the application of the dynamic pricing policy, we use three utilization levels and corresponding reservation prices: $l_0 = 0$ percent ($rp_0 = 0$), $l_1 = 50$ percent ($rp_1 = 0.01$), $l_2 = 70$ percent ($rp_2 = 0.03$). Fig. 2b shows the results for our numerical example. Jobs 1, 2, 4 and

Table 1
Data of jobs.

Number	Price paid	Price paid per	Gold	Type	Resources		
	(fp_j)	slot (fp_j/s_j)	customer		CPU	Storage	Bandwidth
6	1.6	0.8	✓	2	8	16	4
5	4	2		2	8	16	4
4	3	1		1	4	4	16
3	1.8	0.3	✓	1	4	4	16
2	1	1		2	8	16	4
1	2	1		3	16	8	8

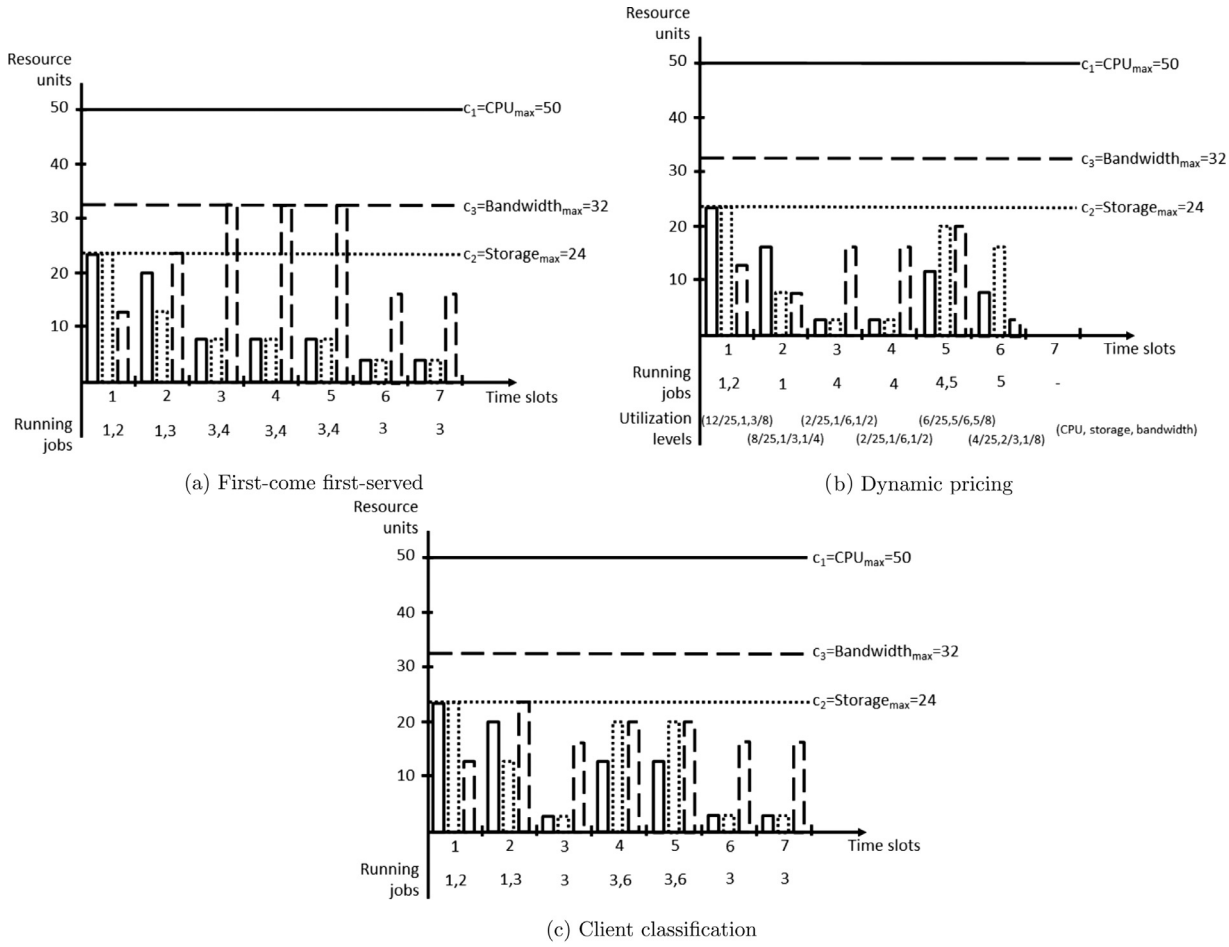


Fig. 2. Results of the application of policies.

5 are accepted, which results in an overall revenue of 10.0. Job 3 is not accepted for the following reason: accepting job 3 would require to allocate $(20, 12, 24)$ units of (CPU, storage, bandwidth) in time slot 2, with $(50, 24, 32)$ units being available. Thus, the utilization levels amount to (40 percent, 50 percent, 75 percent) so that job 3 is not accepted because $0.01 \cdot 4 + 0.03 \cdot 16 = 0.52 > 0.3$. Job 6 is not accepted as the required units of storage (36) exceed the available units of storage (24).

For the application of the client classification policy, we use the utilization level $l_c = 0.7$. Fig. 2c shows the results for our numerical example. Jobs 1, 2, 3 and 6 are accepted, which results in an overall revenue of 6.4. Job 4 is not accepted for the following reason: accepting job 4 would require to allocate $(8, 8, 32)$ units of (CPU, storage, bandwidth) in time slot 4, with $(50, 24, 32)$ units being available. Thus, the utilization level of bandwidth is 1 so that job 4 is not accepted because $1 > 0.7$. Similarly, job 5 is not accepted for the following reason: accepting job 5 would require to allocate $(12, 20, 20)$ units of (CPU, storage, bandwidth) in time slot 5, with $(50, 24, 32)$ units being available. Thus, the utilization level of storage is $5/6$ so that job 5 is not accepted because $5/6 > 0.7$.

4.6. Computational complexity

Low complexity and therefore low computational costs are paramount for real-time decision mechanisms. We show that all policies executed under certainty and under uncertainty run in polynomial time of the number of service requests $|J|$ and the number of resource types $|R|$. We first prove this property for execution under certainty: In the mathematical formulation of the policies, the objective function (O2) serves to sort the jobs by id (which is assigned in order of arrival). Sorting can be done in $O(|J| * \log|J|)$. In a running system this is not necessary since jobs already arrive in order. Thus, it is sufficient to check whether the constraints are satisfied. Each constraint (C2.1)–(C2.3) can be checked in $O(|R| * |J|^2)$. The overall complexity is $O(|R| * |J|^2)$.

As shown above, the overall complexity of the online model under certainty is $O(|R| * |J|^2)$. While each arithmetic operation with fuzzy instead of crisp numbers adds complexity, this complexity is not dependent on R or J, but rather is a constant computational cost per operation. Thus the overall complexity is still $O(|R| * |J|^2)$.

5. Evaluation

To validate the proposed models and policies, and to estimate the effect of different degrees of uncertainty on the revenue, a thorough evaluation is done. We first show certain properties of the model analytically. As some of the proofs require rigid assumptions we further evaluate the model using simulations based on real world workloads. The simulation setting and the workloads used for the evaluation are explained. Subsequently, the results are presented and discussed. The simulator was implemented as an object-oriented program in MATLAB.

5.1. Analytical evaluation

The first objective for the proposed decision model is *revenue maximization*. Without information about future jobs it is not possible to achieve the maximum revenue. To analyze the performance of our

model we compare the revenue of the dynamic pricing policy with a state of the art FCFS approach.

Proposition 1. *Let p^{I}_{avg} and $p^{II}_{avg}(rp)$ be the average price of the accepted jobs for policy I and for policy II, respectively, with rp being the (only) reservation price and l being the utilization level at which rp becomes applicable. Let $ar \geq 1$ be the job arrival rate, and let the price X be a random variable with known cumulative distribution function P(X).*

Then the dynamic pricing policy outperforms the FCFS approach (both under certainty and uncertainty) in terms of revenue if $1 * p^{I}_{avg} < \min[1, P(X \geq rp) * ar] * p^{II}_{avg}(rp)$ under some additional assumptions.

The proof and a numerical example can be found in online Appendix E.

To analyze the second objective, *customer satisfaction*, we evaluate the criteria defined in Section 2. These are an improvement in acceptance probability for important customers and certain fairness aspects.

Proposition 2. *A policy with strict priority outperforms a first come first served approach (both under certainty and uncertainty) in terms of gold customer acceptance probability.*

The proof can be found in online Appendix E.

To address resistance to strategic behavior described in Section 2, we first need to consider which options for strategic behavior customers have. The first option some customers might have is to shift their demand to off peak hours. Another option is to either split their jobs into several smaller jobs or merge several jobs into a bigger one. The last option for strategic behavior is to vary their price bids.

While many customers need instantaneous access to services, others might be able to shift their demand to times where services can be accessed at a lower price. Such behavior is actually in the interest of the provider as it leads to a more balanced system utilization. One of the key benefits of introducing dynamic pricing is to give customers incentives for such behavior.

To discuss the behavior of merging or splitting jobs we analyze merge-proofness and split-proofness (Moulin, 2007). A mechanism is merge-proof if users cannot benefit by merging several jobs to one bigger job. Analogously a mechanism is split-proof if users cannot benefit by splitting one job to several smaller jobs. This makes sure users cannot increase their chance of acceptance or lower their price by exploiting strategic opportunities at the expense of others. As the following propositions show, our model is merge-proof but not split-proof.

Proposition 3. *The different policies of the model (both under certainty and uncertainty) are merge-proof, i.e. users can not benefit by merging several jobs to one bigger job.*

Proposition 4. *The different policies of the model (both under certainty and uncertainty) are not split-proof, i.e. users can benefit by splitting one job to several smaller jobs.*

The proof of these propositions are included in online Appendix E. Moulin (2007) shows that in general it is not possible to achieve both merge- and split-proofness.

As described earlier the third possible strategic behavior for customers is varying their price bids. Whether such strategic behavior can be beneficial to customers depends on the exact pricing conditions. If customers are charged the applicable reservation price for respective utilization level they cannot benefit from varying their bids. If pay-as-you-bid charging is used a certain type of customer (who does not rely on instantaneous access) could benefit but only to the extent of lowering the price to the reservation price or shifting demand to off-peak times.

The analytical evaluation shows that the model delivers improvement both in terms of revenue and customer satisfaction. Since the proof of the improvements in revenue requiresvery specific assumptions it is necessary to further evaluate the revenue improvements delivered by the model using numerical simulations where these assumptions are not necessary.

5.2. Simulation setup

In our simulation, we use real workloads based on data from the Parallel Workload Archive (Feitelson, 2015). The SHARCNET log, which was provided to the Parallel Workload Archive by John Morton (john@sharcnet.ca) and Clayton Chrusch (chrusch@sharcnet.ca), is used as basis for these simulations. It contains 1,195,242 jobs sent to a set of 10 clusters in Ontario, Canada from a period December 2005 until January 2007. The SHARCNET log was chosen as basis because it contains a large variety of jobs with different runtimes, numbers of used CPUs, and varying submit and start times. The workload further shows high variation in demand over time.

Jobs running less than one hour or more than 10 days were filtered. Subsequently job runtimes where rounded down to full hours to allow a timeslot based allocation. After filtering invalid jobs, 566,701 jobs were left and finally used in the simulation. Although this filtering approach reduces the variability in the data, the filtered data set is still quite large and its size gives enough variability to elicit the major managerial implications.

Based on these workloads, nine joblists with different prices and service type assignment were generated as described in the following paragraphs. For the evaluation, we consider three types of services with requirements for processing power, memory, and storage (see Table B.5 in online Appendix B). Service 1 represents a bandwidth heavy service type such as video-streaming or a content delivery network; service 2 represents services requiring mainly storage such as file hosting or online backup drives; the third service type represents a CPU-intensive service such as portfolio optimization or video-transcoding. Table B.6 (see online Appendix B) shows the contents of the job lists and the source of the data used. For submission time, start time, runtime, and customer ID the data were used as present in the workload trace. The service type was drawn from a discrete uniform distribution. The number of instances per service was adapted from the job requirements. The pricing information was generated using a truncated normal distribution in order to get non-negative prices. Full prices were calculated by multiplying unit prices with the number of instances and time slots and rounding the value up to the next integer. It was assumed that gold clients are willing to pay a markup of 20 percent for the priorities.

In our simulation, the capacity is determined by CPU, storage, and bandwidth capacity and limited to 1050 for each of these. This capacity was chosen to accommodate some of the larger jobs from the SHARCNET log but still have time periods where demand exceeds supply. Jobs can either start in the same timeslot in which they are submitted or in a later timeslot. Fig. F.6 (see online Appendix F) gives a general visualization of the simulation process.

The fuzzified models make use of the concept of *Triangular Fuzzy Numbers* (c, a, d), as described in online Appendix D. More precisely, we draw on symmetric *Triangular Fuzzy Numbers*, where $(a - c) = (d - a)$. Alternatively, we write $(c, a, d) = ((1 - g) * a, a, (1 + g) * a)$, $g \in [0, 1]$. In our implementation, we use symmetric fuzzy numbers with $g \in \{0.01, 0.05, 0.1\}$ depending on the level of uncertainty we face. The provision of different levels of uncertainty is useful in two regards: First, it parameterizes the decision model, which in turn allows for checking the robustness of the model. Second, it thereby accounts for different situations of uncertainty. As mentioned in Section 2, the level of uncertainty can vary depending on both the customer and the service request. In this regard, we define the following four scenarios:

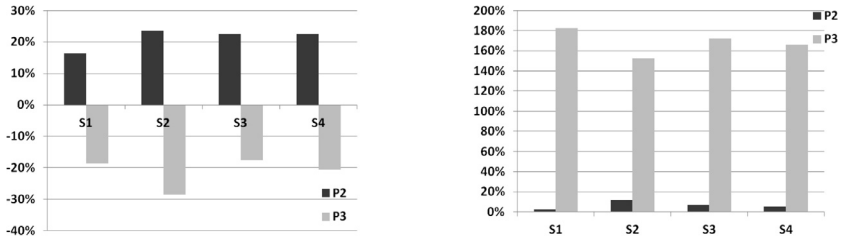
S1: No uncertainty is present.

S2: We assume that the uncertainty of the prediction quality of required resources is mainly based on the type of service request (see

Table 2
Revenue and gold acceptance comparison for all scenarios.

Policy	Scenario	μ rev.	(σ/μ) rev.	Relative cost of uncertainty (in percent)	$\mu\Gamma_g$ (in percent)	$(\sigma/\mu)\Gamma_g$
P1	S1	21249352.78	0.0187	–	4.22	0.0299
	S2	20431190.78	0.0137	−3.85	4.04	0.0614
	S3	20226752.44	0.0119	−4.81	4.36	0.0366
	S4	19956299.11	0.0117	−6.09	4.14	0.0397
P2	S1	24722267.67	0.0169	–	4.33	0.0462
	S2	25231416.11	0.0145	2.06	4.51	0.0499
	S3	24764950.78	0.0093	0.17	4.66	0.0349
	S4	24431015.33	0.0086	−1.18	4.36	0.0432
P3	S1	17297469.78	0.0074	–	11.93	0.0110
	S2	14608163.89	0.0123	−15.55	10.19	0.0120
	S3	16671910.22	0.0104	−3.62	11.86	0.0112
	S4	15858202.11	0.0086	−8.32	11.02	0.0069

P1: FCFS policy P2: Dynamic pricing policy
P3: Client classification policy



Revenue change of policies P2 and P3 compared to benchmark P1 for each scenario

Increase in gold acceptance ratio of policies P2 and P3 compared to benchmark P1 for each scenario

P1: FCFS policy, P2: Dynamic pricing policy, P3: Client classification policy, Si: Scenario i

Fig. 3. Mean revenue and mean gold acceptance ratio over all simulations for each of the three policies under all four scenarios.

Table B.5 in online Appendix B). We assume that for service type 1 resource requirements are relatively easy to estimate so that we assume the lowest level of uncertainty ($g = 0.01$). For service type 2 we use $g = 0.05$. We further assume that due to the high CPU requirements of service type 3 requirements are most difficult to estimate, which results in the highest level of uncertainty ($g = 0.1$).

S3: We assume that the uncertainty of the prediction quality of required resources is mainly based on the type of customer who requests the service. We distinguish between customers with a known good prediction quality and others. We regard those customers as "known" who have userIDs with many jobs in the chosen workload trace. For known customers, we choose $g = 0.01$, for the other customers we choose $g = 0.1$.

S4: We assume that the uncertainty of the prediction quality of required resources is based on both the type of service request and the type of customer. Due to this high uncertainty, for all jobs and customers we use the same level of uncertainty ($g = 0.1$).

5.3. Simulation results

We structure the presentation of our simulation results along the two dimensions of our research framwework. We first present policy-related findings for both situations under certainty and under uncertainty (Findings 1–2), then we present findings that refer to the differentiation between situations under certainty and under uncertainty (Findings 3–5). We refer to the difference of revenues r_1 and r_2 achieved in a situation under uncertainty and in a situation under certainty, respectively, as "absolute cost of uncertainty". We refer to the ratio $(r_1 - r_2)/r_2$ as the "relative cost of uncertainty". Finally, we present a finding on runtimes.

Table 2 contains policies, scenarios, mean revenues, the coefficients of variation of revenue, the relative cost of uncertainty and the mean ratios of accepted gold jobs $\mu\Gamma_g$ as well as the coefficients of variation for the gold acceptance ratio $(\sigma/\mu)\Gamma_g$.

Fig. 3 depicts the mean revenue and mean gold acceptance ratio over all simulations for each of the three policies under all four scenarios.

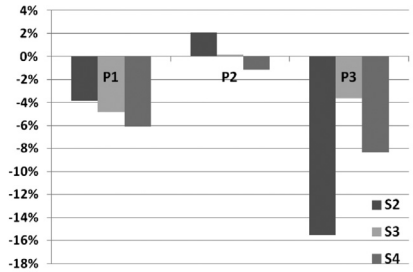
Finding 1. The dynamic pricing policy significantly outperforms the FCFS policy and the client classification policy in terms of revenue under both certainty and uncertainty.

As Fig. 3 indicates, the dynamic pricing policy outperforms the FCFS policy regarding revenue in both situations under certainty and under uncertainty. This result is statistically significant at the 0.01 level (see online Appendix C for a detailed description of the statistical methodology).

As in all instances the revenue values obtained through the client classification policy were lower than those obtained through the FCFS policy, we can apparently conclude (without detailed statistical analysis) that the dynamic pricing policy also outperforms the client classification policy in terms of revenue.

Finding 2. The client classification policy significantly outperforms the FCFS policy and the dynamic pricing policy in terms of gold customer acceptance ratio.

Fig. 3 indicates the superior behavior of the client classification policy in terms of the gold acceptance ratio. Overall, the gold acceptance ratio is relatively low with about 4 percent for the FCFS policy and the dynamic pricing policy, and 10 percent–12 percent for the client classification policy. This is caused by the highly volatile demand of specific users in the SHARCNET workload trace.



P1: FCFS policy, P2: Dynamic pricing policy, P3: Client classification policy, Si: Scenario i

Fig. 4. Relative cost of uncertainty for each policy.

We now turn to the differentiation between situations under certainty and under uncertainty and also discuss the impact of the different uncertainty scenarios on the revenue. The results shown in Fig. 4 and Table 2 indicate the following findings:

Finding 3. The FCFS policy achieves lower revenues with increasing uncertainty.

The FCFS policy achieves the highest revenue in situations under certainty. When uncertainty is present and based on either the type of service request or on the type of customer, then the revenue declines on average by 3.85 percent or 4.81 percent, respectively. When uncertainty is based on both, the lowest revenue is achieved.

Finding 4. In the presence of uncertainty, the client classification policy achieves much lower revenues compared to the FCFS policy and the dynamic pricing policy.

When uncertainty is present, the client classification policy shows much higher losses of revenue (in terms of the relative cost of uncertainty) compared to the FCFS policy and the dynamic pricing policy. The relative cost of uncertainty amounts to more than 15 percent, even in the case of a low level of uncertainty. As a consequence, the incentive for Cloud providers to eliminate uncertainty is particularly high when client classification is applied.

Finding 5. The dynamic pricing policy can achieve higher values when (a low or medium level of) uncertainty is present.

While in the presence of a high level of uncertainty (scenario 4) the revenue decreases (as in the case of the FCFS policy and the client classification policy), it marginally increases in the case of a low or medium level of uncertainty (scenarios 2 and 3, respectively). This finding is counterintuitive and requires further analysis. In principle, the increase of revenue is based on a conservative acceptance approach (required resources are assumed to be slightly higher than the given figures). This approach leads to the phenomenon that (a) first, jobs are rejected that would have been accepted in the case of certainty and (b) as a result, jobs that are submitted later are accepted due to available resources. This can lead to an overall increase of revenue as opposed to the situation under certainty because the accepted jobs are more profitable than the rejected ones. This effect is mitigated with increasing level of uncertainty and even disappears in the presence of a high level of uncertainty when the decision maker is generally more careful in accepting jobs.

In order to understand this phenomenon better, it is reasonable to recall that the different levels of uncertainty are modeled with different widths of fuzzy numbers and what the mathematical consequences of using fuzzy numbers and applying fuzzy arithmetic in our models are. The fuzzification of resource demands leads to a fuzzification of the left side of the resource constraints (we yield triangular fuzzy numbers), while the right sides (resource capacities) remain crisp. According to fuzzy arithmetic, a triangular fuzzy number (c, a, d) is smaller than or equals a crisp value z, if and only if $d \leq z$. Consequently, the fuzzification of resource demands can lead to violations of capacity constraints, when the corresponding crisp values do not. The level of this effect depends on the width of the fuzzy numbers. We observe this effect in applying the FCFS policy, where the presence of uncertainty leads to a decrease of the revenue and where the highest level of uncertainty (all fuzzy figures have width $g = 0.1$) leads to the lowest revenue.

Finding 6. The runtimes of executing the suggested policies are low under both certainty and uncertainty.

The runtime of one instance of the simulation with 566,701 jobs was between 10 and 30 seconds for scenario one (we used a PC with Intel Xeon CPU, E5335@2:00 GHz, 3.75 GB RAM). With runtimes between 1500 and 4000 seconds the scenarios with uncertainty took significantly longer, however each decision took only fractions of a second. This shows the low computation complexity of our approach. Due to the capacity constraint no overload situation occurred, assuring QoS.

6. Managerial implications

In this paper, we formulize different rules for job admission to support Cloud service providers effectively managing their computing infrastructures. In fact, the job admission rules embody a device to establish a functioning revenue management in Cloud environments. Cloud providers can use these rules to optimally allocate perishable Cloud resources in an effort to increase revenue. The execution of rules necessarily needs to be automated, as human intervention is too slow once more than thousands of jobs enter the Cloud system at the same time. Typically, rules are implemented as policies, where policies denote declarative rules defined by the user to adapt and control the behavior of the system. Hence, in our case policies are adequate to embed the logic for job admission as part of the business model of the Cloud provider. Due to the decomposition of the rules and the operational management, the use of policies allows a very flexible management of Cloud infrastructures. Changes in the operational management take place by merely changing or adapting the governing policies.

The application of policies has both economic and technological implications for Cloud providers, which we discuss in the succeeding subsections. While the economic implications are derived within the scope of the paper at hand, we also explain the technological applicability with regard to how our models can be applied in real practice. We briefly sketch how the policies were integrated into a running business prototype (Nimis et al., 2008).

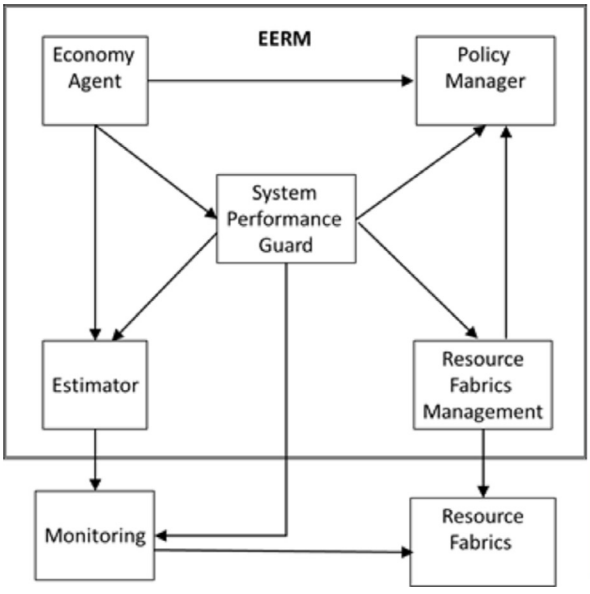
6.1. Economic implications

In this paper, we provide six different policies as a part of a toolbox for Cloud service providers, who need to manage their admission control processes. Essentially, the Cloud providers can freely choose among the different policies that attain their business objective best. We distinguish between two different business objectives, the maximization of short-term revenue and the maximization of gold client satisfaction. The results are qualitatively depicted in Table 3–the table entries refer to the performance of policies with respect to the goal ranging from very good (+ + +) to bad (-).

If the Cloud provider prioritizes (short-term) revenue (cf. upper half of Table 3), the dynamic pricing policies should be used as our results show that the application of dynamic pricing can lead to significant revenue gains. If the Cloud provider faces uncertainty with respect to the required resources, the revenue may increase even further when using the dynamic pricing policy. The revenue attained by the client classification policy is lower than applying FCFS. With increasing uncertainty both policies face a drop in revenue.

Table 3
Policy recommendations for Cloud providers.

Objective: Maximize (short-term) revenue		
Policy/Type of estimates	Reliable estimates of the resources required	Vague estimates of the resources required
FCFS	+	○
Client classification	○	-
Dynamic pricing	++	+++
Objective: Maximize gold customer satisfaction		
Policy / Type of estimates	Reliable estimates of the resources required	Vague estimates of the resources required
FCFS	○	○
Client classification	+++	+++
Dynamic pricing	○	○



Component	Description
Economy Agent	The Economy Agent is responsible for deciding whether incoming jobs are accepted and for calculating their prices.
Estimator	The Estimator component calculates the expected impact on the utilization of the resources.
System Perfomance Guard	The System Performance Guard is responsible for ensuring that the accepted SLAs can be delivered. In case of performance problems with the resources it is notified by Monitoring. After checking the corresponding policies it determines if there is a danger that SLAs cannot be fulfilled. It then takes the decision to suspend or cancel jobs to ensure the fulfilment of the other SLAs and maximize overall revenue.
Policy Manager	To keep the EERM adaptable the Policy manager stores and manages policies concerning client classification, job cancellation or suspension, etc. Policies are formulated in the Semantic Web Rule Language (SWRL)
Resource Fabrics Management	It is an interface to the resources pool, and orders the creation, destruction or modification of the Virtual Machines that handle the sold tasks. However, under certain events like the overload of the system, it can trigger certain rules that can describe some business-driven actions. All policies that have been described in this paper can be executed to the EERM.

Fig. 5. Architecture of the EERM (Nimis et al., 2009, p. 93ff).

In case the provider needs to give internal users or important customers preferred access to the Cloud services (cf. lower half of Table 3), for example, due to service-level agreements, client classification is the appropriate policy. This policy can also be used to offer products at different service levels where availability is significantly higher for gold jobs. Dynamic pricing and FCFS are equally bad in terms of accepted gold jobs. These results are robust against uncertainty regarding the required resources i.e., our results remain valid when Cloud providers account for (non-probabilistic) uncertainty in customers' resource predictions by using fuzzy set models.

In addition, by using our policies providers can determine the cost of giving internal users or important customers preferred access to their services. With this information they can better decide which users should be granted such priority or what price markup is appropriate. The simulation of the suggested policies reveals to what extent different levels of uncertainty reduce revenue. This absolute cost of uncertainty gives information to providers regarding their investments in measures to reduce uncertainty. For example, discounts for customers with high quality predictions of required resources are possible. The absolute cost of uncertainty would then provide an upper bound for such discounts. Our simulations also allow Cloud providers to determine the effect of adding new services with higher uncertainty to their product portfolio.

On the bottom line, our results suggest that Cloud providers should always use the dynamic pricing policy until the gold customer satisfaction drops below a certain threshold. Then, it is best to use the client classification policy until gold customer satisfaction recovers and return above the threshold. The predominant use of dynamic pricing stems from the fact that it is superior in terms of revenue to all other policies. As the client classification policy solely cares for gold customers, it is useful to improve customer satisfaction within a very short period of time.

Lastly it should be noted that all policies–be it under uncertainty or certainty–are executed using a very short runtime, which implies that they are feasible for practical application in the field.

6.2. Technological applicability

In this paper, we mainly analyze the decision policies from a theoretical perspective abstracting from real world applications. Nonetheless, the policies have already been developed, implemented and fully integrated into state of the art resource managers within the scope of the FP6 EU-Project SORMA (http://www.sorma-project.eu). The technical component of an Economically Enhanced Resource Manager (EERM) accounts for the current management gap between the traditional, technical layer of Cloud systems (i.e., classical schedulers and resource managers) and the business layer. The overall goal of the EERM is to isolate economic layers from the complexity of the Cloud Systems and to align both business and performance goals (Wirström et al., 2008). The architecture of the EERM is highlighted in Fig. 5. It shows that EERM provides the infrastructural component, including the policy manager, which manages the execution of our policies. A pilot test where real business users were exposed to the EERM system, demonstrated the applicability of the economic policy

approach to Cloud environments (Windsor, Rosenberg, Villa, & Amar, 2009).

7. Summary and Outlook

In this work, we motivated the need for real-time decision models for the service admission control of Cloud service providers as means of resource-based revenue management. Based on practical requirements, we suggested the use of policies as heuristics, which can deal with both informational certainty and uncertainty regarding actually required resource levels. As the root of uncertainty is not randomness but subjectiveness of human assessments, we drew on fuzzy set theory for modelling uncertainty.

To evaluate the models and policies, we assessed their properties analytically. As this analysis requires certain rigid assumptions we further evaluated them using a simulation based on real world workloads with a simulator implemented in MATLAB. The evaluation showed that the policy based on dynamic pricing can significantly increase revenue. Depending on the type and the level of uncertainty, the increase is between 16.34 percent and 23.49 percent. To validate this observation, appropriate statistical tests were performed. The results further show that the policy which gives gold clients a priority on job acceptance can drastically increase the acceptance ratio for gold customers.

We further discussed the impact of different types of uncertainty on revenue. The uncertainty of resource demands can lead to violations of capacity constraints, when the corresponding crisp values do not, thus reducing the number of accepted jobs. The level of this effect depends on the level of uncertainty, which is modelled with the "width" of triangular fuzzy numbers. However, it can be mitigated or even disappears when the rejection of jobs in early phases leads to the availability of capacities for attractive jobs in later phases, which would otherwise be rejected.

Future work would need to investigate the relation between the degree of uncertainty and revenue. We also plan to research which effects can be observed when both the demand side, i.e. job requirements, and the supply side (available capacity) display degrees of uncertainty.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ejor.2015.01.027.

References

Aiber, S., Gilat, D., Landau, A., Razinkov, N., Sela, A., & Wasserkrug, S. (2004). Autonomic self-optimization according to business objectives. In *Proceedings of the ICAC* (pp. 206–213). Washington, DC, USA.

Amazon (2015). Amazon EC2 pricelist. http://aws.amazon.com/ec2/pricing/.

Anandasivam, A., & Weinhardt, C. (2010). Towards an efficient decision policy for Cloud service providers. In *Proceedings of the ICIS*. Saint Louis, USA.

Baykasoğlu, A., & Göçken, T. (2008). A review and classification of fuzzy mathematical programs. *Journal of Intelligent & Fuzzy Systems*, *19*, 205–229.

Bitran, G., & Caldentey, R. (2003). An overview of pricing models for revenue management. *Manufacturing Service Operations Management*, *5*(3), 203–229.

Boughton, H., Martin, P., Powley, W., & Horman, R. (2006). Workload class importance policy in autonomic database management systems. In *Proceedings of the international workshop on policy* (pp. 13–22). London, Ontario, Canada. 10.1109/POLICY.2006.39

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computing Systems*, *25*(6), 599–616. 10.1016/j.future.2008.12.001

Caglar, F., & Gokhale, A. (2014). iOverbook: Intelligent Resource-Overbooking to Support Soft Real-Time Applications in the Cloud. In *Proceedings of the 7th cloudcomp* (pp. 538–545). Anchorage, Alaska.

Dube, P., Hayel, Y., & 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), 99–102.

Eren, S. S., & Maglaras, C. (2009). Monopoly pricing with limited demand information. *Journal of Revenue and Pricing Management*, *9*(1-2), 23–48.

Feitelson, D. G. (2015). Logs of Real Parallel Workloads from Production Systems, log file #23. http://www.cs.huji.ac.il/labs/parallel/workload/logs.html.

Ferguson, D. F., Nikolaou, C., Sairamesh, J., & Yemini, Y. (1996). Economic models for allocating resources in computer systems. In S. Clearwater (Ed.), *Market-based control: A paradigm for distributed resource allocation* (pp. 156–183). River Edge, NJ, USA: World Scientific Publishing.

Foster, I., Kesselman, C., Lee, C., Lindell, B., Nahrstedt, K., & Roy, A. (1999). A distributed resource management architecture that supports advance reservations and co-allocation. In *Proceedings of the IWQoS'99* (pp. 62–80). London, UK.

Kounev, S., Nou, R., & Torres, J. (2007). Autonomic QoS-aware resource management in grid computing using online performance models. In *Proceedings of the VALUETOOLS'2007* (pp. 1–10). Nantes, France.

Maglaras, C., & Meissner, J. (2006). Dynamic pricing strategies for multiproduct revenue management problems. *Manufacturing & Service Operations Management*, *8*(2), 136–148. 10.1287/msom.1060.0105

Mell, P., & Grance, T. (2009). The NIST definition of cloud computing. Technical Report. http://www.csrc.nist.gov/groups/SNS/cloud-computing/.

Moulin, H. (2007). On scheduling fees to prevent merging, splitting, and transferring of jobs. *Mathematics of Operations Research*, *32*(2), 266–283.

Nair, S., & Bapna, R. (2001). An application of yield management for internet service providers. *Naval Research Logistics*, *48*, 348–362.

Newhouse, S., MacLaren, J., & Keahey, K. (2004). Trading Grid services within the UK e-science Grid. In J. Nabrzyski, J. M. Schopf, & J. Weglarz (Eds.), *Grid resource management: State of the art and future trends* (pp. 479–490). Norwell, MA, USA: Kluwer Academic Publishers.

Nimis, J., Anandasivam, A., Borissov, N., Smith, G., Neumann, D., Wirström, N., et al. (2008). Sorma - business cases for an open grid 2008. In *Proceedings of the 5th Gecon* (pp. 173–184). Las Palmas de Gran Canaria, Spain.

Nimis, J., Macias, M., Rosenberg, E., Wirström, N., Brunner, R., Borissov, N., et al. (2009). SORMA D2.2.a: Final specification and design documentation of the SORMA components. TechnicalReport. http://www.im.uni-karlsruhe.de/sorma/fileadmin/SORMA_Deliverables/D2.2a_final.pdf.

Püschel, T., & Neumann, D. (2009). Management of cloud infrastructures: Policy-based revenue optimization. In *Proceedings of the ICIS*. Phoenix, Arizona, USA.

Püschel, T., Schryen, G., Hristova, D., & Neumann, D. (2012). Cloud service revenue management. In *Proceedings of the ECIS*. Barcelona, Spain.

Qiu, J., & Knightly, E. W. (2001). Measurement-based admission control with aggregate traffic envelopes. *IEEE/ACM Transactions on Networking*, *9*(2), 199–210.

Ramalho, M. F. N. (1998). Uncertainty measures associated with fuzzy rules for connection admission control in ATM Networks. In S. Parsons (Ed.), *Applications of uncertainty formalisms* (pp. 177–197). Springer, Berlin, Germany.

Reiss, C., Tumanov, A., Ganger, G. R., Katz, R. H., & Kozuch, M. A. (2012). Towards understanding heterogeneous clouds at scale: Google trace analysis. Technical Report Intel Science and Technology Center for Cloud Computing.

Windsor, W., Rosenberg, E., Villa, M., & Amar, L. (2009). SORMA D6.2: Pilot study evaluations. Technical Report. http://www.im.uni-karlsruhe.de/sorma/fileadmin/SORMA_Deliverables/D6.2_final.pdf.

Wirström, N., Rasmusson, L., Rana, O., Clair, G. S., Villa, M., Del Grosso, E., et al. (2008). SORMA deliverable D4.4 newly developed components. Technical Report. http://www.im.uni-karlsruhe.de/sorma/fileadmin/SORMA_Deliverables/D4.4_final.pdf.

Yeo, C. S., & Buyya, R. (2004). Pricing for utility-driven resource management and allocation in clusters. In *Proceedings of the ADCOM* (pp. 32–41). Ahmedabad, India.

Zimmermann, H. J. (2000). An application-oriented view of modeling uncertainty. *European Journal of Operational Research*, *122*(2), 190–198.