IRWIN AND JOAN JACOBS
CENTER FOR COMMUNICATION AND INFORMATION TECHNOLOGIES

To Cloud or not to Cloud: Optimizing Cloudbursting Costs

Mark Shifrin, Rami Atar and Israel Cidon

CCIT Report #797
November 2011

DEPARTMENT OF ELECTRICAL ENGINEERING
TECHNION - ISRAEL INSTITUTE OF TECHNOLOGY, HAIFA 32000, ISRAEL

CCIT Report #797 November 2011

To Cloud or not to Cloud: Optimizing Cloudbursting Costs

Technical Report

ABSTRACT

The emerging hybrid cloud architectures allow organizations (users) to augment the private infrastructure with practically unlimited public cloud resources in order to cost effectively meet their intermittent peak demands. In such scenarios, users first utilize their already paid private computation infrastructure and offload selected tasks to the public cloud when the private resources become overloaded. Consequently, there is a need to devise efficient on-line task offload algorithms that optimize the overall user cost while maintaining adequate quality of service. Such algorithms should take into account the difference in communication and computation requirements between the different task types. For example, it is clear that between two tasks with the same computational requirements, the task with the lower migration cost is a better candidate to be offloaded to the cloud. In this work, we devise optimal on-line decision algorithms by modeling and solving several associated multi-dimensional Markov Decision Process problems. We address the case in which arriving tasks have multiple communication costs and prove the structural properties of the optimal threshold policy. In addition, we also apply the MDP framework for the complement problem facing cloud providers. If certain cloud resources are not pre-allocated to users, it makes sense for the cloud provider to offer them for opportunistic on-demand usage. We model and provide optimal policies for the buildup of a task backlog by accepting or rejecting tasks that carry user offered price or by dynamically changing advertised prices for tasks. The analytical results are supported by numerical evaluations. We demonstrate the practical advantage of threshold type policies and provide an insight of their dependence on system parameters.

Keywords

Cloud Computing, Markov Decision Processes, Offloading algorithms.

1. INTRODUCTION

The emergence of cloud computing is changing the ways in which organizations address their information technology (IT) needs. The concept of cloud computing brings out a new way of increasing computational and storage capacity or adding capabilities on the fly without investing in a new infrastructure, training new personnel, or licensing new software. In theory, the cloud agility and elasticity make the cloud the best IT solution model. However, many practical issues such as limited network speeds, lack of strict SLA guarantees, lack of cloud standards, information regulatory compliance and the wish to preserve the full control over their core IT resources and know-how limit the full adoption of the cloud model. Consequently, there is a topical trend to leverage the best of both worlds by keeping the minimal essential legacy IT infrastructure while adopting the public cloud where it is more cost effective. One of the terms which is frequently used to describe this paradigm is a "hybrid cloud". Essentially, the hybrid cloud refers to a business that keeps some of its server operations on-premise, while also utilizing the services of a cloud provider to augment or supplement the internal infrastructure.

Another topical term associated with the hybrid cloud is "cloudbursting". According to this concept, in case the internal data center runs out of computing resources, the organization "bursts" or "offloads" the additional workload to an external cloud on an on-demand basis. The internal computing resource is the "Private Cloud", while the external cloud is typically a "Public Cloud", for which the organization gets charged on a pay-per-use basis. Effective cloudbursting offers the organization a solution to a critical and very basic performance-related and economical dilemma. The pre-cloud old-fashioned solution utilizes over-provisioning, e.g., stacking a computational gear inventory that meets the peak demands. However, if the influx of tasks is characterized by high variability with rare peaks, such a solution is extremely resource wasteful. At the same time, the organization revenues and its business conduct may be considerably harmed if the peak demands are not met. ([1]). Therefore, the hybrid cloud architecture combined with effective "cloudbursting" offers an effective solution to offload the task load peaks from the resource-limited and cost reduced private cloud to a seemingly infinite cloud.

The hybrid cloud environment poses new research challenges associated with effective task offloading from the private data center to the cloud. If all the computational tasks were identical, a simple cloudbursting mechanism would track the backlog of tasks waiting for local execution; if the backlog

crossed a certain value associated with the maximal allowed latency, arriving tasks would be directed to the cloud. As the IT tasks are heterogeneous in terms of computation, communication and storage requirements, the cost-effective design of such task offloading mechanisms becomes more challenging. Therefore, the goal of this paper is to explore the modeling and algorithmic solution to these new problems. To that end, we first propose a model for comparing different task offloading algorithms at the cloud user level. We assume that the goal of the cloud user is to minimize the cost of serving the arriving tasks by using the combination of a resource limited Local Data Center (LDC) and cloudbursting while meeting a predefined QoS level. The model definition involves finding an offloading algorithms to handle the flow of several task types with different resource consumption requirements. Such algorithms should take into account the difference in communication and computation requirements between different task types.
Next, we devise optimal on-line decision algorithms by modeling and solving several associated Markov Decision Process problems. We address the case where arriving tasks have multiple network communication costs, associated with their offload to the cloud, and prove the structural properties of the optimal threshold policy. Although this problem is inherently a multi-dimensional, we show that the optimal solution structure is threshold policy. The dependency between multiple task types is expressed only by the values of the optimal threshold above which particular tasks are offloaded. We quantify and study the behavior of the optimal policy through numerical examples.
To the best of our knowledge, this is the first work that proposes optimal offloading policies for the hybrid cloud environment in which the tasks can be served in the existing LDC or dispatched to the cloud for a given fee. The need for optimal algorithms aiming at facilitating the interaction between the cloud and the users is not limited to the task offloading problem. To illustrate this, we also apply the MDP framework for the complementary problem by facing the cloud provider in optimizing its revenue. If a certain amount of cloud resources is not pre-leased by users, it makes sense for the cloud provider to offer these resources for opportunistic on-demand usage. We model and derive optimal policies for the management of the cloud tasks backlog in two cases. The first one is by accepting or rejecting tasks that carry different offered prices (in a bid process). The second one is by dynamically changing the advertised prices for tasks.

The optimization of cloudbursting, independent of the underlying cloud/LDC infrastructure, naturally lends itself to an optimization of queuing system which can be tackled as a Markov Decision Problem (MDP). The mere understanding that the optimal policy is of the threshold type, allows us to avoid the complete solution to the MDP problem. Instead, approximation techniques can be applied such as Q-learning and approximate policy estimation. Consequently, establishing the optimality of a threshold-type policy, facilitates the solution to MDP problems with large state space.

To summarize, the contribution of this paper is four-fold:

- A user-level MDP based model for allocating heterogeneous tasks to the Local Data Center or to the public cloud.
- Deriving optimal offloading policy for the model presented above. We further present and prove several structural properties of the optimal policy which helps in developing efficient numerical solution for systems of large dimension state.
- Presenting a cloud revenue optimization problem and the corresponding MDP model for the cloud provider task pricing, task admission policy, and the structural properties of the optimal policies.
- We bring a demonstration of efficient utilization of the threshold-type policy. By exploring the dependence of the optimal policy on the system parameters we both suggest a practical usage and learn about the properties of the policy

Many works have addressed the optimization of the cloud infrastructure. For example, [16] suggests that the cloud should be organized in a federated structure and presented as a pool of all cloud resources. Task load balancing techniques inside the cloud were also extensively discussed, for example in [5], [18]. Cloud pricing methods are addressed and proposed in [10] and [19]. [11] [14], [17] suggest software solution for the interaction between users and cloud, but do not present efficient algorithmic solution to the offloading model.[15] develops a rate-limiting architecture for the cloud. [8] presents a solution to the permanent migration of tasks to the cloud, while our paper addresses dynamic "cloudbursting".

Other works are related to the analytical methods used in our paper. Especially, tools from fluid and diffusion approximations have been applied to address models that are closely related to those we introduce here. In particular, a significant amount of work has been done in recent years on models with a large number of servers (see [2],[3],[4],[9] and references therein). An example of deriving threshold-type policy for networking system can be found in [7].

The rest of the paper is organized as follows. In section 2, we describe the model of the user level hybrid cloud system, demonstrate the solution to the related MDP, and prove its structural property. Section 3 is dedicated to the cloud revenue optimization. We distinguish between two options. First, the cloud which differentiates the prices is discussed. We start with a simple case of single task type with two optional price levels and further extend it to several task types with several price levels. The cloud model is concluded with setting with constant prices. The final section is devoted to numerical results and practical models.

2. USER PERSPECTIVE HYBRID CLOUD

We present a setting, in which the LDC can be modeled as a server (e.g. a server cluster) that processes an inflow of tasks of several types. The arriving tasks are either backlogged for a local processing or sent to the public cloud. We assume that there is a maximal waiting delay constraint associated with the local waiting, so tasks must be cloudburst if their expected local waiting time is too long. In the following, we present the details of the above model. Arriving tasks that

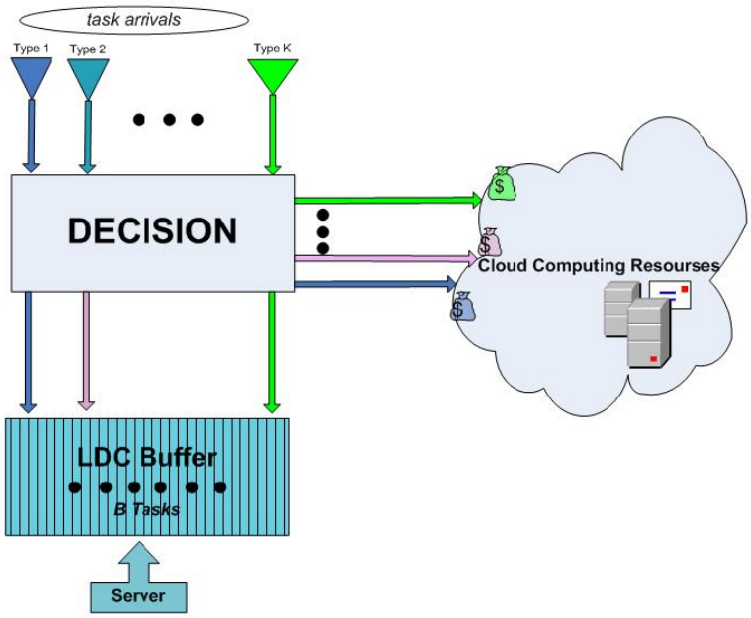

Figure 1: User model system chart

are backlogged for local execution are queued. Such tasks cannot be outburst any longer. Service is FIFO and non-interruptible. In order to embody the delay constraint of the LDC, we assume that the number of tasks awaiting service never exceeds a certain limit denoted as B. Therefore, new arrival occurring while there are already B tasks in the queue is sent to the cloud. As mentioned above, the tasks influx is heterogenous, meaning that different communicating costs are associated with each task type if handed to the cloud. The chart, corresponding to the described model is depicted in Figure 1. Denote by k the number of task types. A cost C_i (includes processing and communication) is incurred whenever a type-i task is sent to the cloud. The types are labeled in such a way that

$$C_1 \geq C_2 \geq \cdots \geq C_k > 0. \tag{1}$$

For the sake of simplicity, we assume that the task processing time distribution at the LDC is independent of the task type. Upon each arrival, a decision is to be made (assuming the buffer is not full) whether to accept the task to the LDC or to offload it to the cloud and pay the corresponding cost.

Denote by A_i and R_i, $i = 1, \ldots, k$, the counting processes for arrival and, respectively, remote offloading, for type i. Namely,

$A_i(t)$ represents the number of tasks of type i that have arrived up to time t,

$R_i(t)$ represents the number of tasks of type i sent to the cloud up to time t.

All counting processes mentioned in this paper are assumed to have right-continuous sample paths. Further, A_i are modeled as independent Poisson processes of intensities λ_i, respectively, and the service time distribution is assumed to be exponential of rate $\mu > 0$, independent of the task type. Next, for each i, let $U_i(t)$ be a process taking values in $\{0, 1\}$, describing the control decisions determining R_i from A_i. Namely, $U_i(t) = 1$ if and only if a type-i task arriving at time t is sent to the cloud. As a result, we can write

$$R_i(t) = \int_0^t U_i(s) dA_i(s) \tag{2}$$

The total number of tasks being enqueued in the LDC is denoted by $X(t)$. The initial condition of X is denoted by $x \in \{0, 1, ..., B\}$. The process X is given by

$$X(t) = x + \sum_{i=1}^{k} A_i(t) - \sum_{i=1}^{k} R_i(t) - D(t), \tag{3}$$

where D denotes the departure process, counting the number of completed tasks (of all types). The total discounted cost associated with a control process $U(t) = (U_1(t), \ldots, U_k(t))$ is given by

$$J(x, U) = E\Big[\int_0^\infty e^{-\gamma s} \sum_{i=1}^{k} C_i dR_i(s)\Big], \tag{4}$$

where $\gamma > 0$ is a discount factor.

A control process U is said to be *admissible* if (i) it is adapted to the filtration generated by $(\{A_i\}, X)$; and (ii) under U, the process $X(t)$ satisfies the constraint

$$X(t) \leq B \text{ for all } t. \tag{5}$$

The first condition expresses the requirement that control decisions are made based on events from the past and present, so that the decision maker has no access to information from the future. The second condition addresses the buffer limit: If for some t one has $X(t) = B$ and a task of type i arrives

then $U_i(t)$ must be set to 1. The class of all admissible control processes is denoted by $\mathcal{U}$. The value function for the optimal control problem is defined as

$$V(x) = \inf_{U \in \mathcal{U}} J(x, U), \qquad x \in \{0, 1, ..., B\}. \tag{6}$$

This is a problem of continuous time Markov decision processes. For such a problem a principal tool is the characterization of the function V as the solution to a Bellman equation. Using this tool we can show that a policy of threshold type is optimal.

REMARK 2.1. *It is natural to work with an average cost rather than a discounted one. However, it is well known that, provided $\gamma > 0$ is sufficiently small, the optimal policies with and without discount, are the same. We later detail about what is known as Blackwell optimality.*

Denoting $\delta = (\mu + \gamma + \sum_i \lambda_i)^{-1}$, the value function uniquely solves the Bellman equation (see e.g., [20] Chapter 8)

$$V(x) = \delta\mu V(x-1) + \sum_{i=1}^{k} \delta\lambda_i \min[V(x) + C_i, V(x+1)], \qquad x \in \{1, 2, \ldots, B-1\}, \tag{7}$$

with boundary conditions

$$V(0) = \delta\mu V(0) + \sum_{i=1}^{k} \delta\lambda_i \min[V(0) + C_i, V(1)],$$
$$V(B) = \delta\mu V(B-1) + \sum_{i=1}^{k} \delta\lambda_i [V(B) + C_i]. \tag{8}$$

Denoting $\delta' = (\gamma + \sum_i \lambda_i)^{-1}$, the boundary condition at zero could be written in a more standard form as $V(0) = \sum_{i=1}^{k} \delta'\lambda_i \min[V(0) + C_i, V(1)]$. However, the form (8) will be useful in the analysis.

The threshold structure is provided by the following.

THEOREM 2.1. *There exist constants $B - 1 = b_1 \geq b_2 \geq b_3 \geq \cdots \geq b_k$ such that the following policy is optimal:*

- *$U_1(t) = 0$ if and only if $X(t) \leq b_1$; that is, always accept type-1 tasks unless the buffer is full;*
- *For $i = 2, \ldots, k$, $U_i(t) = 0$ if and only if $X(t) \leq b_i$; that is, accept type-i tasks if and only if the buffer contains b_i or fewer tasks awaiting service.*

The rest of this section is devoted to the proof of this result. The first step will be to prove that V is nondecreasing and convex. To this end, consider the operator T, acting in the space of functions from $\{0, 1, \ldots, B\}$ to $\mathbb{R}$, defined as

$$TU(x) = \delta\mu U(x-1) + \sum_{i=1}^{k} \delta\lambda_i \min[U(x) + C_i, U(x+1)], \qquad x \in \{1, 2, \ldots, B-1\}$$
$$TU(0) = \delta\mu U(0) + \sum_{i=1}^{k} \delta\lambda_i \min[U(0) + C_i, U(1)],$$
$$TU(B) = \delta\mu U(B-1) + \sum_{i=1}^{k} \delta\lambda_i (U(B) + C_i), \tag{9}$$

for $U : \{0, 1, \ldots, B\} \to \mathbb{R}$. Then the Bellman equation reads $TV = V$. Denote

$$\|U\| = \max_x |U(x)| \tag{10}$$

and let S be the set of functions from $\{0, 1, \ldots, B\}$ to $\mathbb{R}$ that are nondecreasing, convex, and having slope bounded by C_1, that is

$$U(x+1) - U(x) \leq C_1, \qquad x \in \{0, 1, \ldots, B-1\}.$$

The following lemma asserts that T preserves S, and moreover, acts on it as a strict contraction.

LEMMA 2.1. *One has $TS \subset S$. Moreover, there exists a constant $a \in (0, 1)$ such that*

$$\|TU - TW\| \leq a\|U - W\| \text{ for every } U, W \in S.$$

PROOF. To prove the first assertion, let $U \in S$ be given. Then for $2 \leq x \leq B-1$,

$$TU(x) - TU(x-1) = \delta\mu(U(x-1) - U(x-2)) + \sum_{i=1}^{k} \delta\lambda_i \{\min[U(x) + C_i, U(x+1)] - \min[U(x-1) + C_i, U(x)]\}. \tag{11}$$

Hence, using the nondecreasing property of U, $TU(x) - TU(x-1) \geq 0$. A similar calculation for $x = 1$ and $x = B$ gives $TU(x) - TU(x-1) \geq 0$ as well, and the nondecreasing property of TU follows.

To show that the slope of TU is bounded by C_1, we use again (11). Since U satisfies such a condition, it follows that $U(x-1) - U(x-2) \leq C_1$ and that each of the expressions in curly brackets is bounded by C_1. Since $\delta\mu + \sum_i \delta\lambda_i \leq 1$, it follows that $TU(x) - TU(x-1) \leq C_1$. A similar calculation for $x = 1$ and $x = B$ gives an analogous result, and it follows that the slope of TU is bounded by C_1.

To prove that TU is convex, we will use the fact that if W is any convex function mapping $\{0, 1, \ldots, B\}$ to $\mathbb{R}$ and C a constant, then the function $Z : \{0, 1, \ldots, B\} \to \mathbb{R}$ defined by

$$Z(x) = \begin{cases} \min[W(x) + C, W(x+1)] & \text{if } x \leq B-1, \\ W(B) + C & \text{if } x = B, \end{cases}$$

is also convex. The elementary proof of this fact is omitted. Denote the transformation mapping W to Z by T_C. That

is, $Z = T_C W$. Then TU can be written as

$$\delta\mu\tilde{U} + \sum_{i=1}^{k} \delta\lambda_i Z_i,$$

where $Z_i = T_{C_i} U$, and

$$\tilde{U}(x) = \begin{cases} U(x-1) & \text{if } x > 0, \\ U(0) & \text{if } x = 0. \end{cases}$$

Owing to the fact that U is convex and nondecreasing, $\tilde{U}$ is seen to be convex. It follows that TU is convex, as the sum of $k+1$ convex functions.

We have thus shown $TU \in S$. Since $U \in S$ is arbitrary, this proves $TS \subset S$.

To prove the second assertion, let $U, W \in S$. Consider first $x \in \{1, 2, \ldots, B-1\}$. By (9), denoting $a \vee b = \max(a, b)$, $a \wedge b = \min(a, b)$ and using the inequality

$$|(a \wedge b) - (c \wedge d)| \leq |a - c| \vee |c - d|,$$

we have

$$\begin{aligned} &|TU(x) - TW(x)| \\ &\leq \delta\mu |U(x-1) - W(x-1)| \\ &\quad + \sum_{i=1}^{k} \delta\lambda_i [|U(x) - W(x)| \vee |U(x+1) - W(x+1)|] \\ &\leq a \, \|U - W\|, \end{aligned} \tag{12}$$

where $a = \delta\mu + \sum_{i=1}^{k} \delta\lambda_i$. By the definition of δ, $a < 1$. For $x = 0$ and $x = B$, the calculation is similar, and gives the same result, namely $|TU(x) - TW(x)| \leq a\|U - W\|$. We conclude that $\|TU - TW\| \leq a\|U - W\|$. □

Proof of Theorem 2.1.

We use the contraction mapping principle (see e.g. [12, Theorem V.18]). The set S, equipped with the metric $\rho(U, W) = \|U - W\|$ is a complete metric space. The map $T : S \to S$ is a strict contraction, as shown in the above lemma. As a result, T has a unique fixed point. That is, there exists a unique $U \in S$ for which $TU = U$. Recall that V is the unique solution to the same equation in the space of *all* functions from $\{0, 1, \ldots, B\}$ to $\mathbb{R}$. As a result, $V = U$. This shows $V \in S$, namely, that V is nondecreasing and convex.

In order to show the threshold property of the policy, we employ the method from [20], which builds on convexity. One can read off an optimal feedback control from the Bellman equation (7), as follows. Given $0 \leq x \leq B-1$, if a class-i arrival occurs when $X(t) = x$, send it to the cloud (and pay C_i) if and only if

$$V(x) + C_i < V(x+1). \tag{13}$$

Since V is convex, $V(x+1) - V(x)$ is nondecreasing in x, and so, if (13) holds for some (i, x), it also holds for (i, x') for all $x < x' \leq B-1$. In other words, class-i task acceptance occurs if and only if $X(t) \leq b_i$ for suitable constants b_i. The ordering of b_i, as alluded to in the statement of the theorem, is also clear by this argument. It remains to show that $b_1 = B-1$. By the above discussion, it suffices to show that $V(x) + C_1 \geq V(x+1)$ for all $0 \leq x \leq B-1$. This, however, follows from the fact that the slope of V is bounded by C_1, as $V \in S$. ■

Note, that the threshold-type structure of the optimal policy is independent of the capabilities of user infrastructure. This offers a planning option which considers a trade-off between the one-time hardware spending, and the resulting optimal cost.

3. CLOUD MODEL

The MDP approach is also very useful in studying the cloud operation optimization in the hybrid cloud setup. The objective of the decision process associated with operating the cloud is to maximize the public cloud provider profit over time. Although from the user's perspective the cloud is often regarded as an infinite resource, in operating the cloud infrastructure, one must take into account the finite resource availability as well as their effective utilization. Consequently, the cloud provider may take advantage of periods with very high cloudbursting demands to select the most profitable tasks and may offer itself underutilized resources at reduced prices for opportunistic on-demand usage.

We propose two models of the cloud. In the first one, the cloud manager attempts to maximize revenue by manipulating prices so as to influence the demand flows. In other words, as the demand for outbursting increases or as the cloud resources are better utilized the cloud provider can consider a price increase despite a possible reduction in the offered task load.

In this model we assume that there are no additional decision variables besides price control. In particular, the cloud must accept all arrivals at the advertised price as long as the number of backlogged tasks does not exceed a predefined queue capacity (which stands for the maximal waiting time, which is part of the cloud SLA). Note, that when a task is rejected, the associated revenue is lost. In the second model, the cloud manager makes an acceptance/rejection decision upon each arrival, according to fixed prices associated with each task. In both cases, the main result is that the optimal policy is of the threshold type structure.

3.1 Optimizing cloud revenue with dynamic advertised prices

We first present several notations. The types of tasks are indexed by the set $\{1, ..., I\}$.

For each task type i there are K levels of prices denoted by C_{ij}, $i \in \{1, \ldots, I\}$, $k \in \{1, \ldots, K\}$. Prices are ordered so that, for each i,

$$C_{i1} \leq C_{i2} \leq \cdots \leq C_{iK}.$$

The cloud influences the arrival rates by dynamically varying the prices for handling a task. At any given time, for each $i \in \{1, ..., I\}$, one and only one of the prices $C_{i1}, \cdots, C_{iK}$ is advertised. It is assumed that, in response to an advertised price C_{ik}, the type-i arrival rate is a given constant λ_{ik}. Naturally, it is assumed that the rates are ordered, so that

for each i,

$$\lambda_{i1} \geq \lambda_{i2} \geq \ldots \geq \lambda_{iK}.$$

It is further assumed that when, for each i, there are x_i tasks of type i in the cloud, one has

$$\sum_{i=1}^{I} w_i x_i \leq B \tag{14}$$

where $\{w_i\}$ denote task sizes, and B is the buffer limit. In this paper, for simplicity, we assume $w_i = 1$ for all i.

Let E_{ik}, $i \in \{1, \ldots, I\}$, $k \in \{1, \ldots, K\}$ be independent Poisson processes of intensities λ_{ik}, respectively. Denote by A_i the counting processes for arrival of type-i tasks. Denote by $A_{ik}(t)$ the number of type-i tasks priced at level k, that have arrived up to time t. Then

$$A_i(t) = \sum_{k=1}^{K} A_k(t),$$

and it is assumed that, for each i and k,

$$A_{ik} = \int_0^t U_{ik}(s) dE_k(s).$$

Here, for each i, $\{U_{ik}, 1 \leq k \leq K\}$ is an $\mathtt{S}$-valued process, where

$$\mathtt{S} = \Big\{ u \in \{0,1\}^K : \quad \sum u_k \leq 1 \Big\}.$$

Given i, an advertisement of price C_{ik} at time t corresponds to selecting $U_{ij}(t) = 1$ for $j = k$ and $U_{ij}(t) = 0$ for all other j. The option $U_{ik}(t) = 0$ for all k is also possible, and corresponds to rejection, which in the present model is used only when the buffer is full. We thus regard U_{ik} as the control processes.

The total number of tasks present in the buffer is given by

$$X(t) = x + \sum_{i=1}^{I} A_i(t) - D(t), \tag{15}$$

where $x \in \{0, \ldots, B\}$ denotes an initial condition, and D is the departure process, counting the number of completed tasks of all types. The service time distribution is assumed to be exponential of rate $\mu > 0$, independent of the task type. Similarly to the previous section, a control process U is regarded admissible if it is adapted to the filtration generated by $(\{E_{ik}\}, X)$, and is such that the buffer limit $X(t) \leq B$ is kept at all times. The class of admissible control processes is denoted by $\mathcal{U}$.

The total discounted reward associated with a control process U is given by

$$\begin{aligned} J(x,U) &= E\Big[\int_0^\infty e^{-\gamma s} \sum_{i=1}^{I} \sum_{k=1}^{K} C_{ik} dA_{ik}(s) \Big] \\ &= E\Big[\int_0^\infty e^{-\gamma s} \sum_{i=1}^{I} \sum_{k=1}^{K} C_{ik} U_{ik}(s) dE_{ik}(s) \Big], \end{aligned} \tag{16}$$

where $\gamma > 0$ is a discount factor. The value function is defined as

$$V(x) = \sup_{U \in \mathcal{U}} J(x,U), \qquad x \in \{0, 1, ..., B\}. \tag{17}$$

3.1.1 Single task type

We analyze first the case with tasks of a single type. Since there is only one task type we omit the index i from C_{ik}, λ_{ik}, etc.

THEOREM 3.2. *There exist constants*

$$0 = b_0 \leq b_1 \leq \cdots \leq b_K = B + 1,$$

such that the following policy is optimal: Announce price C_i at time t if and only if $b_{i-1} \leq X(t) < b_i$.

PROOF. We provide the proof for the case with two price levels. This proof can be extended to several price levels in a straightforward way. Denote $\delta_1 = (\mu + \gamma + \lambda_1)^{-1}$, $\delta_2 = (\mu + \gamma + \lambda_2)^{-1}$ and $\bar{\delta} = (\gamma + \mu)^{-1}$. The value function uniquely solves the Bellman equation ([20] Chapter 8)

$$\begin{aligned} V(x) = \max\{&[\delta_1 \mu V(x-1) + \delta_1 \lambda_1 (V(x+1) + C_1)], \\ &[\delta_2 \mu V(x-1) + \delta_2 \lambda_2 (V(x+1) + C_2)]\}, \\ &x \in \{1, 2, \ldots, B-1\}, \end{aligned} \tag{18}$$

with the boundary conditions

$$\begin{aligned} V(0) = \max\{&[\delta_1 \mu V(0) + \delta_1 \lambda_1 (V(1) + C_1)], \\ &[\delta_2 \mu V(0) + \delta_2 \lambda_2 (V(1) + C_2)]\}, \end{aligned} \tag{19}$$

$$V(B) = \bar{\delta} \mu V(B-1). \tag{20}$$

For a function $U : \{0, 1, \ldots, B\} \to \mathbb{R}$, consider the property

$$\tilde{U}(x) := \beta U(x+1) - \alpha U(x-1) \text{ is nonincreasing in } x, \tag{21}$$

for $1 \leq x \leq B-1$, where $\beta = \lambda_1 \delta_1 - \lambda_2 \delta_2$, $\alpha = \mu \delta_2 - \mu \delta_1$. We will argue that V has this property. To this end, consider the operator T, acting in the space of functions from $\{0, 1, \ldots, B\}$ to $\mathbb{R}$, defined as

$$\begin{aligned} TU(x) = \max\{&[\delta_1 \mu U(x-1) + \delta_1 \lambda_1 (U(x+1) + C_1)], \\ &[\delta_2 \mu U(x-1) + \delta_2 \lambda_2 (U(x+1) + C_2)]\} \\ &\qquad\qquad x \in \{1, \ldots, B-1\}, \\ TU(B) = \bar{\delta} \mu U(B-1), &\qquad\qquad x = B, \\ TU(0) = \max\{&[\delta_1 \mu U(0) + \delta_1 \lambda_1 (U(1) + C_1)], \\ &[\delta_2 \mu U(0) + \delta_2 \lambda_2 (U(1) + C_2)]\}, \qquad x = 0, \end{aligned} \tag{22}$$

for $U : \{0, 1, \ldots, B\} \to \mathbb{R}$. Then the Bellman equation reads $TV = V$. Let S be the set of functions $U : \{0, 1, \ldots, B\} \to \mathbb{R}$ that are non-increasing and possess the property (21).

LEMMA 3.2. *One has $TS \subset S$. Moreover, there exists a constant $a \in (0,1)$ such that*

$$\|TU - TW\| \leq a \|U - W\| \text{ for every } U, W \in S.$$

PROOF. To prove the first assertion, let $U \in S$ be given. For $x \in \{2, \ldots, B-1\}$, $TU(x) - TU(x-1) \leq 0$ by (22), using the fact that the maximum of two nonincreasing functions is nonincreasing. A verification for $x = 1$ and $x = B$ gives $TU(x) - TU(x-1) \leq 0$ as well, and the nonincreasing property of TU follows.

Proving the property (21) of TU amounts to showing that

$$\beta TU_{x+1} - \alpha TU_{x-1} \text{ is nonincreasing.} \tag{23}$$

Denoting $W_x^i = \delta_i \mu U_{x-1} + \delta_i \lambda_i (U_{x+1} + C_i)$, $i \in \{1,2\}$, the expression in (23) can be written as

$$\beta \max_i W_{x+1}^i - \alpha \max_i W_{x-1}^i. \tag{24}$$

We will use the following fact, omitting its elementary proof. Let numbers a_k^i and b_k^i, be given, where k ranges over a finite set and $i \in \{1,2\}$. Then

$$a_k^1 - b_l^1 \geq a_k^2 - b_l^2 \text{ for all } k \text{ and } l$$

implies

$$\max_k a_k^1 - \max_k b_k^1 \geq \max_k a_k^2 - \max_k b_k^2.$$

To show that (24) is nonincreasing in x it suffices to show that for any $i \in \{1,2\}$ and $j \in \{1,2\}$, $\beta W_{x+1}^i - \alpha W_{x-1}^j$ is nonincreasing in x.

To prove that $\beta W_{x+1}^1 - \alpha W_{x-1}^1$ is nonincreasing, expand as follows:

$$\begin{aligned}
&\beta\delta_1\lambda_1(U_{x+2} + C_1) + \beta\delta_1\mu U_x \\
&\quad - \alpha\delta_1\lambda_1(U_x + C_1) - \alpha\delta_1\mu U_{x-2} \\
&= \delta_1\lambda_1(\beta U_{x+2} - \alpha U_x) \\
&\quad + \delta_1\mu(\beta U_x - \alpha U_{x-2}) + C \\
&= \delta_1\lambda_1\tilde{U}(x+1) + \delta_1\mu\tilde{U}(x-1) + C.
\end{aligned} \tag{25}$$

Since both $\delta_1\lambda_1$ and $\delta_1\mu$ are positive, (25) is nonincreasing as the sum of two nonincreasing functions. The case $\beta W_{x+1}^2 - \alpha W_{x-1}^2$ is treated similarly.

Next we treat the case of $\beta W_{x+1}^2 - \alpha W_{x-1}^1$. Expanding, we have

$$\begin{aligned}
&\beta\delta_2\lambda_2 U_{x+2} + \beta\delta_2\mu U_x \\
&- \alpha\delta_1\lambda_1 U_x - \alpha\delta_1\mu U_{x-2} + C
\end{aligned}$$

where C is a constant. Rewrite this as

$$\begin{aligned}
&\beta\delta_1\lambda_1 U_{x+2} + \beta\delta_1\mu U_x \\
&+ \beta(\delta_2\lambda_2 - \delta_1\lambda_1)U_{x+2} + \beta(\delta_2\mu - \delta_1\mu)U_x \\
&- \alpha\delta_1\lambda_1 U_x - \alpha\delta_1\mu U_{x-2} + C \\
&= \lambda_1\delta_1\tilde{U}_{x+1} + \delta_1\mu\tilde{U}_{x-1} - \beta^2 V_{x+2} + \alpha\beta U_x \\
&= \lambda_2\delta_2\tilde{U}_{x+1} + \delta_1\mu\tilde{U}_{x-1}
\end{aligned}$$

This is the sum of two nonincreasing functions and therefore nonincreasing. In the case $\beta W_{x+1}^1 - \alpha W_{x-1}^2$, a similar calculation gives $\lambda_1\delta_1\tilde{U}_{x+1} + \delta_2\mu\tilde{U}_{x-1}$, leading to the same conclusion. This concludes the proof of (23) for x between 2 and $B-2$. The boundary cases are dealt with analogously, and we omit the details.

To prove the contraction property, let $U, W \in S$. Consider first $x \in \{1, 2, \ldots, B-1\}$. We have

$$\begin{aligned}
&TU(x) - TW(x) = \\
&= \max\{[\delta_1\mu U(x-1) + \delta_1\lambda_1(U(x+1) + C_1)], \\
&\qquad [\delta_2\mu U(x-1) + \delta_2\lambda_2(U(x+1) + C_2)]\} \\
&\quad - \max\{[\delta_1\mu W(x-1) + \delta_1\lambda_1(W(x+1) + C_1)], \\
&\qquad [\delta_2\mu W(x-1) + \delta_2\lambda_2(W(x+1) + C_2)]\}
\end{aligned} \tag{26}$$

Using $|\max(a,b) - \max(c,d)| \leq \max(|a-c|, |b-d|)$, and denoting $\Delta = U - W$,

$$\begin{aligned}
&|TU(x) - TW(x)| \\
&\quad \leq \max\{|\delta_1\mu\Delta(x-1) + \delta_1\lambda_1\Delta(x+1)|, \\
&\qquad |\delta_2\mu\Delta(x-1) + \delta_2\lambda_2\Delta(x+1)|\}
\end{aligned}$$

Since $\delta_k\mu + \delta_k\lambda_k < 1$, this gives

$$|TU(x) - TW(x)| \leq a\|\Delta\| = a\|U - W\|,$$

where $a < 1$. A similar calculation for $x = 0$ and $x = B$ gives an analogous inequality, and we conclude that $\|TU - TW\| \leq a\|U - W\|$. □

Proof of Theorem 3.2. Based on the above lemma, the technique from the proof of Theorem 2.1 gives $V \in S$. Finally, the optimal action at state x can be read from the Bellman equation (18). The action depends on whether the inequality

$$\begin{aligned}
&\delta_1\mu V(x-1) + \delta_1\lambda_1(V(x+1) + C_1) \\
&> \delta_2\mu V(x-1) + \delta_2\lambda_2(V(x+1) + C_2)
\end{aligned}$$

holds. This can be written as

$$\beta V(x+1) - \alpha V(x-1) > C := -C_1\delta_1\lambda_1 + C_2\delta_2\lambda_2,$$

namely $\tilde{V}(x) > C$. The monotonicity property of $\tilde{V}$ thus gives the threshold property.

3.1.2 Multiple task types

We describe the optimal decision of I types of tasks, each one with K different prices and K corresponding arrival rates. Denote the arrival rate k of task of type i as λ_{ik} and the corresponding price(reward) as c_{ik}. We assume no dependence between costs and rates of different types. Hence, there are $L = I^K$ different possibilities of choosing the price sets within all types of tasks. Clearly, the optimal policy dictates a set of the arrival rates and the corresponding prices in each state x. Consider the enumeration $m = 1...L$ of all such sets. Denote each set as $\{\lambda_{ik}\}_m$, and assign the corresponding δ_m to each set. For example $\delta_m = (\mu + \gamma + \lambda_{1i} + \lambda_{2j} ... + \lambda_{iK})^{-1}$ for some given set $\{\lambda_{ij}\}_m$. Denote the arrival rate of task type i of price level j pertaining to the set m as λ_{ij}^m.

We state the structure of the optimal policy in the following theorem:

THEOREM 3.3. *There exists a sequence of sets $\{\lambda_{ik}\}_{m_x}$, such that for every cloud state $x = 0, \ldots, B-1$, there is an optimal set of arrival intensities $\{\lambda_{ik}\}_{m_x}$, $m_x \in \{1...L\}$, such that if it holds $m_i = m_j$, then i and j belong to the monotonous interval of states*

$$[x - s, x - s + 1, \ldots, x - 1 + t, x + t],$$

$x - s \leq i \leq j \leq x + t$ for which the optimal set is

$$m_{x-s} = \ldots, m_i, \ldots = m_j, \ldots = m_{x+t}$$

Omitting the boundary cases, the optimal solution is defined

by the following operator:

$$\begin{aligned}
&TV(x) = \max\{ \\
&\big[\delta_1\mu V(x-1) + \delta_1\big(\lambda_{11}(V(x+1)+c_{11}) + \\
&\quad \lambda_{21}(V(x+1)+c_{21}) + ... \\
&\quad \lambda_{I1}(V(x+1)+c_{I1})\big)\big], \\
&\big[\delta_2\mu V(x-1) + \delta_2\big(\lambda_{12}(V(x+1)+c_{12}) + \\
&\quad \lambda_{21}(V(x+1)+c_{21}) + ... \\
&\quad \lambda_{I1}(V(x+1)+c_{I1})\big)\big], ... \\
&\big[\delta_L\mu V(x-1) + \delta_L\big(\lambda_{1K}(V(x+1)+c_{1K}) + \\
&\quad \lambda_{2K}(V(x+1)+c_{2K}) + ... \\
&\quad \lambda_{IK}(V(x+1)+c_{IK})\big)\big]\}
\end{aligned} \tag{27}$$

That is, in each state x the optimal $\{\lambda_{ik}\}_{m_x}$ with corresponding $\{c_{ik}\}_{m_x}$ are chosen. Next, consider the sum of the rate intensities and the corresponding sum of the prices as follows:

$$\Lambda_m = \sum_{i=1}^{I}\sum_{k=1}^{K}\lambda_{ik}^m \quad , \quad C_m = \sum_{i=1}^{I}\sum_{k=1}^{K}c_{ik}^m$$

Then, equation (27) takes the following form:

$$\begin{aligned}
&V(x) = \max\{ && (28)\\
&[\delta_1\mu V(x-1) + \delta_1\Lambda_1(V(x+1)+C_1)], \\
&[\delta_2\mu V(x-1) + \delta_2\Lambda_2(V(x+1)+C_2)]... \\
&[\delta_L\mu V(x-1) + \delta_L\Lambda_L(V(x+1)+C_L)]\}, \\
&\quad x \in \{1,2,\ldots,B-1\} && (29)
\end{aligned}$$

Note, that equation (29) corresponds to the model of a single task type with L possible price levels. Since the arrival rates Λ can be ordered, the solution, and the the structural property of the policy are identical to that of the single task type case. The only difference is that each price and corresponding arrival rate represent a unique set of I different tasks prices of I corresponding task types. We conclude that the policy structure has a threshold-type, as defined in the theorem and omit the straightforward proof.

It is noteworthy, that task arrival or departure event of any task type can alter the optimal prices of all the tasks types in the following state. Generally, we expect to have sets with higher prices for the busy states of the cloud, and lower prices once the cloud is comparatively idle. The actual order of the thresholds would be defined by the differences between the constants c_{ik} and λ_{ik}.

3.2 Optimizing cloud revenue with fixed prices

In this subsection we discuss a cloud model with fixed prices and limited cloud resource. Unlike the previous case with variable prices, the cloud can reject the tasks, according to their fixed offered prices, even when the cloud backlog is not full. This scenario corresponds to an opportunistic cloud setting. The user may select to work with a certain task type and the cloud doesn't guarantee accepting the tasks. Alternatively, the cloud decision can correspond to the internal cost associated with different task tasks.

We shortly list the parameters of this model. The service of the cloud resource is FIFO and non-interruptible. The tasks influx is heterogeneous, where different costs are associated with each task type. Denote by K the number of task types. A reward C_i is earned whenever a type-i task is accepted at the cloud. Again, we use the labeling:

$$C_1 \geq C_2 \geq \cdots \geq C_K > 0. \tag{30}$$

Upon each arrival, a decision is to be made (assuming the cloud resource is not full) weather to accept the task by the cloud or to reject it. In this setting the process X is given by

$$X(t) = x + \sum_{i=1}^{K}A_i(t) - \sum_{i=1}^{K}R_i(t) - D(t). \tag{31}$$

Since the arrival rates are constant, A_i in (31) are represented by Poisson processes. Denote $U_i(t)$, the decision process $i \in \{1,\ldots I\}$, $U_i \in \{0,1\}$, where $U_i = 1$ means accepting the task. The rejection process counting the number of rejected tasks is given by

$$R_i(t) = \int_0^t (1 - U_i(s))dA_i(s).$$

Denoting the counting process for accepted tasks by $L_i = A_i - R_i$, the discounted reward is given by

$$J(x,U) = E\Big[\int_0^\infty e^{-\gamma s}\sum_{i=1}^{I}C_i dL_i(s)\Big]. \tag{32}$$

The value $V(x)$ is defined as the supremum of $J(x,U)$ over admissible controls U.

Denoting $\delta = (\mu+\gamma+\sum_i\lambda_i)^{-1}$, the value function uniquely solves the Bellman equation as follows

$$\begin{aligned}
V(x) = \delta\mu V(x-1) + \sum_{i=1}^{K}\delta\lambda_i\max[V(x), V(x+1)+C_i], \\
x \in \{1,2,\ldots,B-1\},
\end{aligned} \tag{33}$$

with boundary conditions

$$\begin{aligned}
V(0) &= \delta\mu V(0) + \sum_{i=1}^{K}\delta\lambda_i\max[V(0), V(1)+C_i], \\
V(B) &= \delta\mu V(B-1) + \sum_{i=1}^{K}\delta\lambda_i V(B).
\end{aligned} \tag{34}$$

The proof of the following result is similar to that of the previous results, hence omitted.

THEOREM 3.4. *There exist constants*

$$B = b_1 \geq b_2 \geq \cdots \geq b_K \geq 0$$

such that the following policy is optimal.

- *For $i \in \{1,2,\ldots,K\}$, $U_i(t) = 0$ if and only if $X(t) < b_i$; that is, accept type-i tasks if and only if the buffer contains fewer than b_i tasks awaiting service.*

The cloud scenario which considers an opportunistic framework with differentiated price levels, offers a potential to a

Table 1: *User Hybrid Cloud - Threshold Examples*

LDC Buffer size	Prices	Thresholds
35	3, 4, 5	1, 20, 35
35	3.1, 3.11, 3.12	4, 22, 35
18	0, 2, 3	-1, 10, 18
18	1, 2, 3, 4, 5	-1, 0, 1, 10, 18

wide variety of optimization issues both at the user level which aims to maximize the performance at the minimum cost, and the cloud level which seeks the optimal solutions to the long-time revenue maximization.

Connection to average cost objectives

We have demonstrated properties of solutions to both user and cloud optimization problems using MDPs as a with *discounted* cost/reward criteria. For the long-run goals it makes sense to optimize average cost criteria. The relation between the two types of cost is well-understood (see e.g. [6, Chapter 4.1]). In particular, when the discount factor γ is sufficiently close to zero, the optimal policy for discounted cost is also optimal for the long-run average cost (ibid.). Consequently, our result are valid for analogous MDP formulations with long-run average cotsts/rewards.

4. PRACTICAL IMPLEMENTATION AND NUMERICAL RESULTS

In this section we give the results of the exploration of dependence of the optimal solution on different system parameters. In sequel, we demonstrate the insight, how the conclusions facilitate the practical usage of the threshold-type policy.
Generally, the practical importance of the proofs of the structure of the optimal policy is two-fold. First, since the state space of the problem was shown to be one-dimensional the numerical solution is commutatively effective. Second, the fact that the optimal policy has a threshold-type structure makes the policy *predictable*. That is, once solution of some system, denote it as system A is known, the solution of another system which differs from A in some single parameter, denote it as A', can be intuitively guessed, and then improved during the run-time.
We start with demonstrating examples of threshold policies for sample systems. Next, we explore the dependance of the thresholds on system size and formulate a computationally-effective algorithm based on the observations.

4.1 Numerical Results for the User Model

The implementation of the numerical MDP solution allows to sense the actual values of the thresholds thus validating the theorems proved in section 2 and 3. Numerical examples for the user hybrid cloud model are presented in table 1. Each threshold in the column of the thresholds corresponds to the price on the same position in the column of prices. Note, that -1 means that the corresponding task is always offloaded to the cloud. As expected, the task with the highest price is always scheduled to the LDC. The first two cases demonstrate the dependence on the price differences. As we decrease the difference between the prices, the thresholds of the cheaper task types move towards the LDC buffer limit.

4.1.1 Impact of the load.

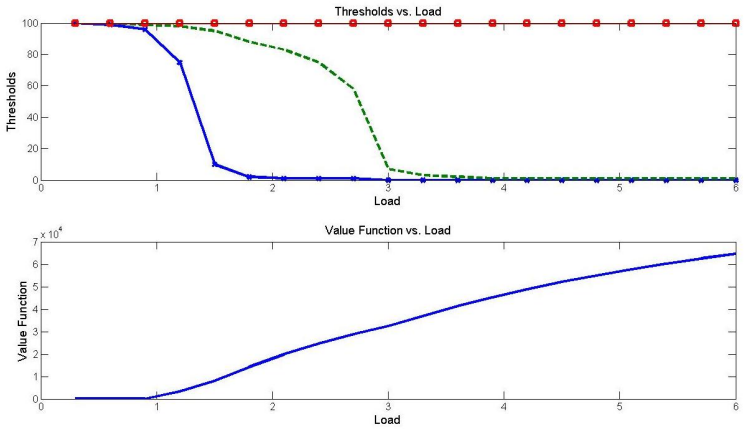

Figure 2: Value function and the threshold values as a function of the load

Next, we capture the impact of the load $\rho = \frac{\mu}{\sum_{i=1}^{K} \lambda_i}$ on the threshold and the value function. The setting of 3 task types symbolically priced by prices $5, 10, 20$ and LDC buffer of size $B = 100$ was tested under variable load, starting at 0.3 and ascending to 6.0. The results are demonstrated in Figure 2. Note, that the thresholds of the tasks with the lowest price drop rapidly as the load grows. This stems from the fact that the system "prefers" to reserve the space for the most expensive task type. Correspondingly, the value function grows. Note, that the ordering of the thresholds stays constant. This observation suggests, that the optimal policy for the higher/lower load can be intuitively estimated once the policy for the lower/higher load is known.

4.1.2 Impact of the LDC buffer size.

Next, we focus on the system with a constant load of $\rho = 1$, while the LDC buffer was gradually increased. Since the size of the LDC buffer dictates the ability to accommodate the arriving tasks, we expect that the increase of the buffer would decrease the value function. Indeed, simulation results shown in Figure 3 support this. In addition, it can be observed that the threshold results grow monotonously in the buffer size. We conclude, that in this case both the threshold and the value function can be intuitively predicted based on the previously experienced solutions.

4.1.3 Variations in task prices.

The influence of the task price was studied in the following sense. The setting of 3 prices $(5, C_2, 20)$, where C_2 was ranging from 5 to 100 was simulated while the load $\rho = 1$ was set. The results are shown in Figure 4. The threshold of the second task type gradually increases until it surpasses the threshold of the third task type. The fact that the second price is growing while the other two prices staying constant explains the increase in the value function. Likewise, it explains the fact that the threshold of the cheapest task is changing.
It turns out, that the "intensity" of the changes in the threshold highly depends on the load. Figure 5 demonstrates the same simulation setting, but with the load $\rho = 5$, while $\lambda_i > \mu, \forall i$. One can observe that the thresholds drastically switch right after they become equal. It is intuitively ex-

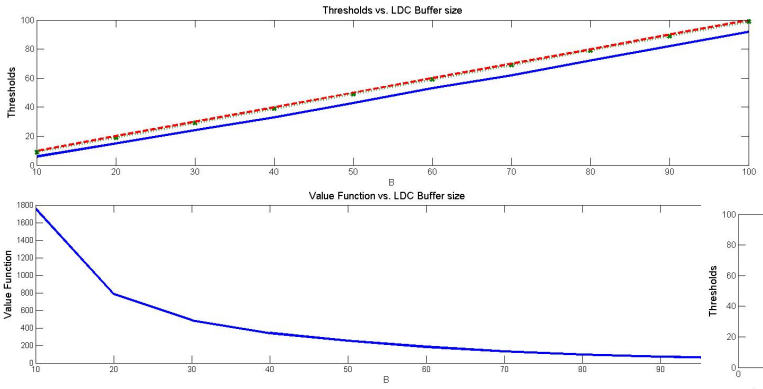

Figure 3: Value function and the threshold values as a function of the LDC buffer size

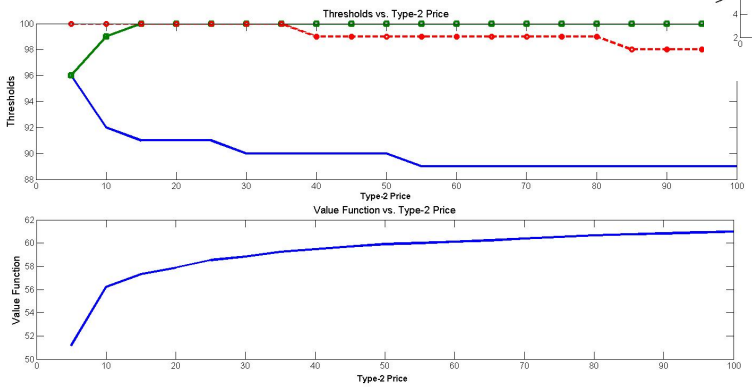

Figure 4: Value function and the threshold values as a function of the LDC buffer size for load $\rho = 1$

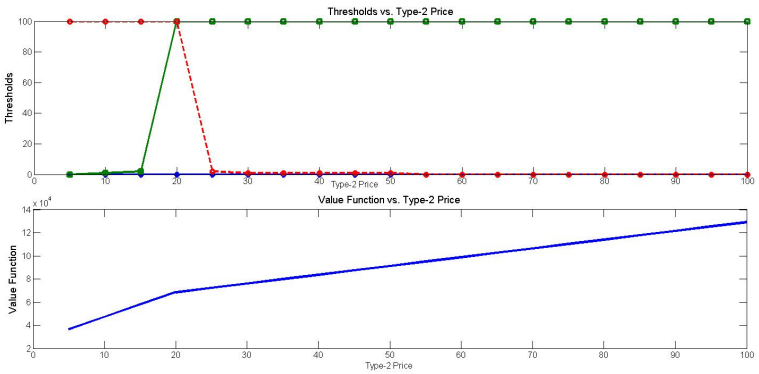

Figure 5: Value function and the threshold values as a function of the LDC buffer size for load $\rho = 5$

plained by that for the high load $\rho \gg 1$ most of the tasks are offloaded to the cloud. Next, we studied the case where the total load is high, but the arrival rate of the type-2 (with arrival rate equal to λ_2) task is significantly lower than μ. This simulation is shown on Figure 6. In this case the threshold for the C_3, which is priced higher than C_1, doesn't drop drastically after the point where type-2 task threshold surpasses it. This can be deduced from that the arrival rate of type-2 tasks is very low anyway. It is noteworthy, that the value function changes relatively fast as long as $C_2 < C_3$ and stays nearly constant after $C_2 > C_3$. This is also due to the low arrival rate of the type-2 tasks. Their probability to be offloaded to the cloud is relatively low, so starting the point where they are always scheduled to the LDC the change in the value function is expected to be insignificant.
The conclusion is that in the case where the prices are being dynamically modified by the cloud once an intuitively approximated policy is about to be applied on the basis of the previous solution one should consider the load factors.

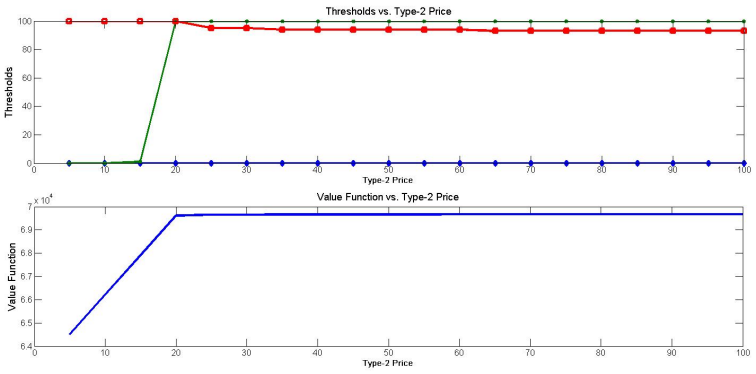

Figure 6: Value function and the threshold values as a function of the LDC buffer size for load $\rho \gg 1$, $\lambda_2/\mu \ll 1$

4.2 Practical Utilization of Threshold Policy

Based on the observations above, a derivation of a computation-effective method of finding the optimal policy makes sense. We propose a version of such an algorithm based on the example as follows. Consider system A with task prices $\{C_i\}$, rates $\{\lambda_i\}$ and μ and the LDC buffer size B. Next, con-

sider system A' which equals to A in all parameters but th LDC buffer size, $B' > B$. This scenario may take place the the Hybrid cloud structure is able to dynamically increas the LDC capacity, for example then the LDC is shared b another enterprize. The algorithm for finding the optima policy for the system A' is based on the solution of A an can be summarized as follows:

- Observe the optimal solution for the system A and in tuitively estimate the influence of the altered parame ter on the system A', e.g. B.
- Guess the optimal solution (the value function and th thresholds) based on the estimation.
- Numerically improve the guessed policy by continuing the numerical calculation starting from the guessed point.

As long as the guessed solution is closed to the optimal solution for the system A', the convergence will be significantly faster than it is performed from the scratch. Therfore, systems which experience changes in the parameters from time to time will react faster once one of the parameters is altered.

4.3 Numerical Results for the Cloud Model

Similarly to the user model, we explore the cloud model. Our main point of interest is the setting where the cloud dynamically advertises the prices. (The formulation of the case of cloud with constant pricing is similar to the user case and we expect similar observations and conclusions to apply.) Selected numerical results of the cloud with dynamically advertised prices are presented in table 2. Next, we study the dependence on the different parameters of the cloud setting.

4.3.1 *Impact of the cloud capacity.*

Consider a cloud structure with a cloud capacity which can vary from time to time. This scenario can be in effect, for example, then the cloud manager allocates part of the resources for the usage with a constant pricing, while the other part is used for the dynamically advertised pricing. The key observation is that the cloud capacity has a critical influence on the threshold values. Figure 7 shows, that once the capacity is increased, the highest arrival rate becomes to be active (i.e. optimal) for the lower cloud states, while for the low capacity it wasn't active at all. The same tendency is observed in the second threshold; however, the growth is seen to start "earlier". The threshold of the highest price (i.e the lowest arrival rate) increase approximately linearly with the capacity.

This leads us to the notable conclusion that the high number of price levels might be unnecessary for the small cloud resources. In other simulations, we saw that for the small capacities it was relatively rare to spot several price levels with active thresholds, rather then only two levels were chosen for the entire states scale or a single price was preferred. This can lead not only to the pricing rule but also to the projected number of the needed price levels to be advertised.

4.3.2 *Impact of the cloud service rate.*

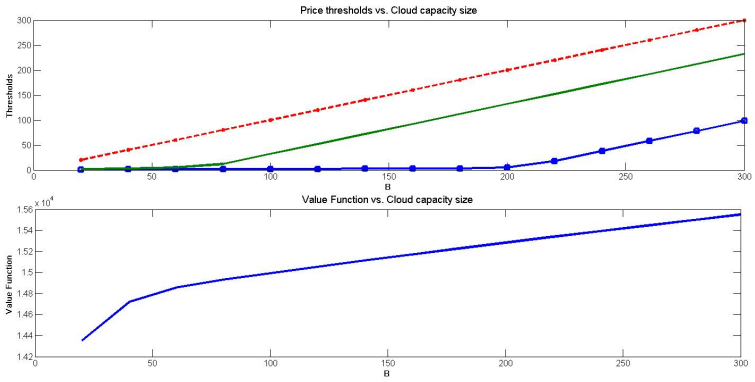

Figure 7: Value function and the threshold values as a function of the cloud capacity

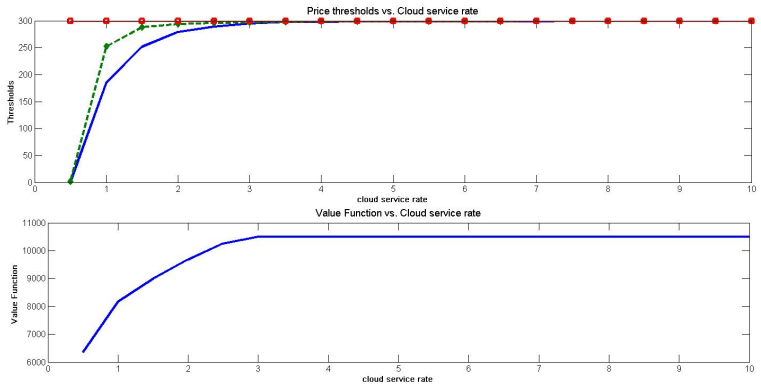

Figure 8: Value function and the threshold values as a function of cloud service rate

The same setting was tested for the constant cloud capacity of $B = 300$, while the service rate of the cloud was gradually increased. It is seen that for the high values of μ the highest arrival rate becomes optimal for all the states. Clearly, any further increasing in μ has only negligible influence on the value function. This can be explained by that the probability of the task rejection is low. One can check from equation (18) that as $\mu \gg \lambda_i, \forall i$ it holds $\delta_i \approx \delta_j, \forall\{i,j\}$. Thus, considering that $V(x)$ is slowly changing with x (see Figure 9) it holds

$$\delta_i \mu V(x-1) \gg \delta_i \lambda_i V(x+1)$$

Therefore the level with highest product $\lambda_i C_i$ will be chosen as the only active level. This conclusion coincides with intuition, that in the system with (almost) all packets being served the best choice will be the average arrival rate multiplied by the price charged for each task. One checks, that since approximately all packets are accepted, equation (16) is reduced to the following:

$$E\Big[\int_0^\infty e^{-\gamma s} C dE\Big] = \int_0^\infty e^{-\gamma s} C\lambda ds = \frac{C\lambda}{\gamma}$$

We substituted the parameters used in the simulation and reproduced the equal result.

We summarize this section by concluding that the optimal

Table 2: *Cloud with Dynamically Advertised Prices - Threshold Examples*

B	μ	λ_i	C_i	b_i
30	0.5	0.3 0.5	14 10	27 30
30	0.45	0.3 0.5 0.7	14 10 8	25 27 30
60	0.5	0.3 0.5 0.6 0.7	14 11 10 9	48 55 57 60
160	0.75	0.3 0.4 0.5 0.6	13 12 10 8.5	114 131 156 160
260	0.45	0.3 0.4 0.5 0.6 0.7 0.75	14 12.5 10.9 9.5 8.3	247 253 256 259 260

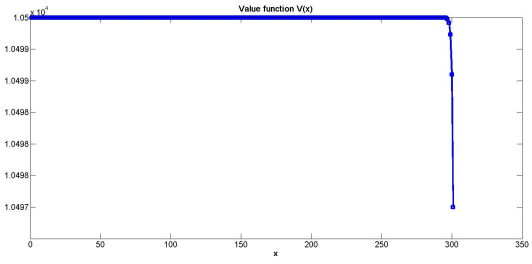

Figure 9: Example of the Value function V(x). Note that a difference between V(B) and V(0) is very small

policy for the cloud model can be intuitively predicted in a same manner as in the user model. In spite of the fact that in some cases the thresholds structure is somewhat less intuitive, still a careful experience by simulation can be effectively exploited for the faster convergence once the system undergoes changes in some of the parameters.

5. CONCLUSIONS

In this paper, the problem of cloudbursting in a hybrid-cloud environment was analyzed. We considered the most salient scenario, referring to a user with limited local resources, such as local data center or a private cloud, presenting the analytical framework based on Markov Decision Processes for the optimal control. The prominent result is the derivation of the threshold policy for the problem of cloudbursting. The precise optimal scheduling rule, once at user's disposal, allows to minimize the cost associated with process of serving the heterogeneous task flow. Unlike previous work that proposed heuristic solutions our work presents optimal policies, which are also very simple to implement.

Furthermore, we demonstrated that the suggested framework is also useful for solving the optimal control problems as far as the cloud optimization is concerned. We exploited this for optimizing the cloud control in two scenarios, which are cloud with task acceptance/rejection and cloud with controlled prices.

The mere existence of the optimal threshold policy can serve as a powerful tool for faster calculations of the optimal policies. It is clear that the user-cloud scheduling problem can be solved by calculating the related MDP. The difficulty arises once the user/cloud capacity grows, thus the state dimension increases and makes the precise calculation impractical. Awareness of the threshold type optimal policy is the key to the solution to this obstacle.

We believe that following our basic examples, a large variety of problems in cloud computing can be successfully tackled using the methods presented in this paper.

6. REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, Above the Clouds: A Berkeley View of Cloud Computing,UC Berkeley Reliable Adaptive Distributed Systems Laboratory, Rep. UCB/EECS-2009-28, EECS Department, U.C. Berkeley, Feb 2009.

[2] M. Armony, Dynamic routing in large-scale service systems with heterogeneous servers, Queueing Systems, 51(3-4), pp. 287-329, 2005

[3] R. Atar, A. Mandelbaum, M. I. Reiman. Scheduling a multi-class queue with many exponential servers: Asymptotic optimality in heavy-traffic, The Annals of Applied Probability, Vol. 14, No. 3, pp. 1084-1134, Aug 2004

[4] R. Atar, A. Mandelbaum, G. Shaikhet, Simplified control problems for multi-class manyserver queueing systems, Mathematics of Operations Research, Vol. 34, no. 4, pp. 795-812, Nov 2009

[5] A.Berl, E. Gelenbe, M. di Girolamo, G. Giuliani, H. de Meer, M. Q. Dang, K. Pentikousis, Energy-Efficient Cloud Computing, The Computer Journal, Vol. 53 Issue 7, Sep 2010

[6] D. P. Bertsekas,Dynamic Programming and Optimal Control, Vol.2, Athena Scientific, 2006

[7] B. Hajek, Optimal Control of Two interacting Service Stations, IEEE Transactions on automatic control, vol. AC-29. no. 6, June 1984

[8] M. Hajjat, X. Sun, Y.-W. E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, M. Tawarmalani, Cloudward Bound: Planning for Beneficial Migration of Enterprise Applications to the Cloud, SIGCOMM 2010

[9] J. M Harrison, A Zeevi, Dynamic scheduling of a multiclass queue in the Halfin-Whitt heavy traffic regime. Operations Research 52, no. 2, pp. 243-257, 2004

[10] D. Kondo, B. Javadi, P. Malecot, F. Cappello, D. P. Anderson, Cost-Benefit Analysis of Cloud Computing versus Desktop Grids, IEEE International Symposium on Parallel and Distributed Processing, 2009

[11] H. Liu, D. Orban, GridBatch: Cloud Computing for Large-Scale Data-Intensive Batch Applications, 8th IEEE International Symposium on Cluster Computing and the Grid, CCGRID 2008

[12] R. Michael, S. Barry. Methods of modern mathematical physics. I. Functional analysis. Academic Press, New York-London, 1972

[13] C. Moretti, J. Bulosan, D. Thain, P. J. Flynn, All-Pairs: An Abstraction for Data-Intensive Cloud Computing, IEEE International Symposium on Parallel and Distributed Processing, 2008

[14] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov, The Eucalyptus Open-source Cloud-computing System, 9th IEEE/ACM International Symposium on Cluster Computing and the Grid

[15] B. Raghavan, K. Vishwanath, S. Ramabhadran, K. Yocum, A. C. Snoeren, Cloud Control with Distributed Rate Limiting, SIGCOMM 2007

[16] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres,M. Ben-Yehuda, W. Emmerich F. Galán, The RESERVOIR Model and Architecture for Open Federated Cloud Computing, IBM Journal of Research and Development

[17] M. A. Salehi, R. Buyya, Adapting Market-Oriented Scheduling Policies for Cloud Computing,Algorithms and Architectures for Parallel Processing, Lecture Notes in Computer Science, Volume 6081/2010, pp. 351-362,2010

[18] B. Sotomayor, R. S. Montero, I. M. Llorente, I. Foster, Capacity Leasing in Cloud Systems using the OpenNebula Engine, In Workshop on Cloud Computing and its Applications (CCA08)

[19] H. Wang ,Q. Jing, R. Chen, B. He, Z. Qian, L. Zhou, Distributed Systems Meet Economics: Pricing in the Cloud, 2nd USENIX Workshop on Hot Topics in Cloud Computing

[20] J. Warland, An Introduction to Queueing Networks, Prentice-Hall, 1988