The Constrained Ski-Rental Problem and its Application to Online Cloud Cost Optimization

Ali Khanafer*, Murali Kodialam†, and Krishna P. N. Puttaswamy†
*Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, USA,
Email: khanafe2@illinois.edu
†Bell Laboratories, Alcatel-Lucent, Murray Hill, NJ, USA,
Email: {murali.kodialam,krishna.puttaswamy_naga}@alcatel-lucent.com

***Abstract*—Cloud service providers (CSPs) enable tenants to elastically scale their resources to meet their demands. In fact, there are various types of resources offered at various price points. While running applications on the cloud, a tenant aiming to minimize cost is often faced with crucial trade-off considerations. For instance, upon each arrival of a query, a web application can either choose to pay for CPU to compute the response fresh, or pay for cache storage to store the response so as to reduce the compute costs of future requests. The Ski-Rental problem abstracts such scenarios where a tenant is faced with a to-rent-or-to-buy trade-off; in its basic form, a skier should choose between renting or buying a set of skis without knowing the number of days she will be skiing.**

In this paper, we introduce a variant of the classical Ski-Rental problem in which we assume that the skier knows the first (or second) moment of the distribution of the number of ski days in a season. We demonstrate that utilizing this information leads to achieving the best worst-case expected competitive ratio (CR) performance. Our method yields a new class of randomized algorithms that provide arrivals-distribution-free performance guarantees. Further, we apply our solution to a cloud file system and demonstrate the cost savings obtained in comparison to other competing schemes. Simulations illustrate that our scheme exhibits robust average-cost performance that combines the best of the well-known deterministic and randomized schemes previously proposed to tackle the Ski-Rental problem.

I. Introduction

Cloud service providers (CSPs) such as Amazon and Microsoft rent out resources, such as CPU, memory, storage, etc., at various price points and offer their tenants the ability to elastically scale the resources up (or down) depending on the demand. Taking advantage of these services, cloud-based applications have been widely deployed in the recent years at a rapid pace. Since many services have been virtualized, it is easy for an enterprise to scale the amount of resources needed to satisfy the current demand for a service by scaling the number of virtual machines (VMs) supporting that service. Interestingly, scaling the number of VMs is not the only way to reduce costs in a cloud-based service. Consider, for instance, a web application running on the cloud. Each time this service receives a query, the application has the following two options:

- Recompute the query from scratch. This involves the CPU and I/O costs, if any, for using the disk.
- Compute the result and store it in the cache. This will incur the storage cost of the cache; however, it would save the CPU and I/O costs the next time the query is executed with the same parameters.

Choosing the more economic option for a given application will depend on the relative costs of CPU, I/O, and cache storage. In addition, this will depend on the frequency at which this application is accessed. A similar scenario arises when running a file system in the cloud. When a request for a block of data arrives, the application has the following options:

- Read the data from the disk and return it to the user incurring an I/O cost.
- Store the block in the cache and return it from the cache subsequently, which incurs the I/O and storage costs of the cache. However, it does not incur the disk I/O cost which is typically more expensive than the cache I/O cost.

Evidently, there are many cost-based decisions that have to be made in the cloud even when the traffic is not varying considerably. This problem becomes quite important when the costs of different options vary widely. For example, consider the pricing in Table I. Within Amazon, a user could choose to store an object in ElastiCache (acts as a cache), which has (over 100×) a high storage cost but free I/Os, or Amazon S3 (acts as a disk), which has a low storage cost but very high per I/O costs. If many requests to the same file arrive in a short interval, then it is cheaper to store and serve the file from the cache instead of serving it from the disk. But if the queries are far apart, then it is cheaper to serve the file from the disk directly.

Service Name	Storage	Read	Write
(A) ElastiCache	38	0	0
(A) S3	0.110	1×10^{-6}	10×10^{-6}
(M) Azure	0.125	1×10^{-6}	1×10^{-6}

TABLE I
Cost optimization opportunities across providers (as of July 27th 2012). Read and Write costs are per operation, while storage is per GB per month. A is for Amazon and M is for Microsoft.

Another dimension to this problem is the fact that the costs of different options vary across different CSPs. For instance, in Table I, Azure has 10 times lower write costs while 10% higher storage costs compared to S3. Hence, there is scope for

*This work was done while the first author was a summer intern at Bell Laboratories, Alcatel-Lucent.

978-1-4673-5946-7/13/$31.00 ©2013 IEEE

splitting a service across multiple CSPs to further optimize the locations of the disk, the read operation, and the write operation. There are performance implications for some of these decisions which we do not deal with in this paper. Instead, we focus on cost optimization. In our ongoing work, we are considering cost optimization along with performance constraints.

Generally, due to the per unit time cost of various resources, there are many situations where costs can be optimized in the cloud by trading off compute vs. storage, disk vs. cache, bandwidth vs. cache, etc. Some of these problems have been explored in the recent past [1], [2]. In fact, these problems can be abstracted using the classical Ski-Rental problem, which encapsulates the fundamental trade-off between renting or buying a certain service when the period of usage is not known *a priori* to the person interested in the service. Ski-Rental problems were first described in [3] in the context of snoopy caching. In its basic form, a designer is faced with the option of either buying or renting a set of skis. The designer does not know the number of days she will be skiing and is interested in minimizing the overall cost of her trip. Many variants of the Ski-Rental problem have been studied in the literature; see [4], [5] and the references therein. In the compute versus storage example, the act of buying the skis can be mapped to recomputing the query, and the unknown number of ski days is equivalent to the fact that we do not know how many times and how frequently a query will be executed.

There are two classes of competitive algorithms used to tackle the Ski-Rental problem: deterministic and probabilistic. The performance measure of these algorithms is the competitive ratio (CR): the ratio between the cost incurred when an *online* algorithm is used and that incurred when an *offline* algorithm (that knows the future) is utilized. The deterministic algorithm has a worst-case CR of 2, whereas the probabilistic algorithm has a worst-case CR of $\frac{e}{e-1}$. It has been shown that these ratios are the best possible using a standard argument called Yao's Minimax Principle; see [6], p. 35. From a worst-case CR standpoint, one would prefer the randomized approach to the deterministic one. Nonetheless, one can ask the following interesting question: *can we further improve the worst-case performance of the randomized algorithm given extra information about the distribution of the arrivals?*

In this paper, we are interested in performing *worst-case expected CR* analysis in search for an improvement upon the deterministic and randomized approaches previously proposed. In other words, we aim to devise randomized algorithms that provide worst-case performance guarantees *independent* of the distribution of the arrivals. To this end, we formulate a *constrained version* of the Ski-Rental problem and show that our solution to this problem gives rise to distributions that outperform both the deterministic and randomized approaches. The main contributions of this paper are as follows:

- We formulate the problem as a continuous-kernel zero-sum game between the algorithm designer who seeks to minimize the expected CR and an adversary (or nature) attempting to maximize it. Also, we derive the optimal mixed-strategies for both players in closed form.
- We propose a new variant of the Ski-Rental problem where the algorithm designer can exploit the knowledge of the first and second moments of the adversary's strategy. We call this problem the Constrained Ski-Rental Problem. Our formulation leads to a new class of randomized algorithms that provide *arrivals-distribution-free* performance guarantees; we show that our algorithms outperform existing approaches in the worst-case expected CR sense.
- Finally, we apply our theoretical findings to cloud file systems and assess the performance of our proposed approach using numerical studies.

The rest of the paper is organized as follows. In Section II, we outline the standard Ski-Rental problem and present the Constrained Ski-Rental problem in Section III. In Sections IV and V, we solve the problems of the designer and the adversary. Finally, we evaluate our solution using simulations as well as synthetic and real-world file system datasets in Section VI. We conclude the paper in Section VII.

II. The Ski-Rental Problem

The Ski-Rental problem captures the trade-off between buying and renting a product (or service) when the time period for which the product is going to be used is not known in advance. In the standard Ski-Rental problem, a user (designer) is interested in determining whether to buy skis at a cost of $\$B$ or to rent it at a cost of $\$1$ per day. Clearly, if the user skis for less than B days, it is better to rent the skis. On the other hand, if she skis for more than B days, then it is better to buy the skis at the outset. The challenge stems from the fact that the user does not know ahead of time how many days she is going to ski.

Consider the equivalent problem in a cloud cost optimization setting. In particular, consider the problem that was outlined in the introduction as to whether to recompute the result of a web query each time it is requested or whether to store the result and serve it out. Let the cost of storing the query be $\$1$ per unit time and the cost of recomputation be $\$B$. If the next query arrives before B time units, it is better to store the result and serve it out. If the next query arrives after B time units, then it is better to recompute when the query arrives. The system does not know ahead of time the time unit of the next query. A similar analogy can be made for the disk versus cache storage problem. To address this uncertainty, it is of interest to devise online algorithms capable of determining the optimal choice for the user at every time instant.

In [3], the authors propose a deterministic approach (referred to as **DET** in the rest of the paper) which dictates that the designer should rent the skis up until the B-th day at which point she should buy the skis. The CR achieved by this scheme is 1 for arrivals before B and 2 for arrivals occurring after time B. Hence, this scheme is 2-competitive, i.e., it yields a worst-case CR of 2. In [7], the authors propose an optimal randomized scheme (referred to as **PROB** in the rest of the paper) which switches from renting to buying

the skis at a carefully chosen random time. This algorithm achieves an expected CR of $\frac{e}{e-1}$ regardless of the arrival time. It has also been shown that the best achievable performance of deterministic and probabilistic algorithms are 2 and $\frac{e}{e-1}$, respectively. Here, we propose a new approach which makes use of extra information about the adversary's strategy. We construct our scheme in two steps:

- We start with the assumption that the designer possesses information about the adversary's strategy. In particular, we consider two types of information: the mean μ and the second moment σ. Intuitively, knowing the average number of ski days can help the designer decide on the whether she should rent or buy the skis.
- We first derive the optimal policy for this problem under a constraint on the first moment; we refer to this policy by μ-**PROB**. Then, we derive the optimal policy for the Ski-Rental problem with a constraint on the second moment; we refer to this policy by σ-**PROB**. We demonstrate that the optimal policies are *almost independent of the first and second moments* (in a sense made precise later). These policies can be used even when the number of ski days is not known.

We show both theoretically as well as experimentally that the performance of μ-**PROB** and σ-**PROB** is more robust than the performance of **DET** and **PROB**. We now formulate the Constrained Ski-Rental Problem.

III. The Constrained Ski-Rental Problem

Consider a Ski-Rental problem where the rental price is \$1[1] and the buying price is $\$B$ ($B > 1$) . Let x be the time at which the designer decides to buy the skis, and let $p(x)$ be the probability distribution over x. Also, let y be the arrival time (or the number of snow days) chosen by the adversary with $q(y)$ being the probability distribution over y. The designer (adversary) is interested in selecting $p(x)$ ($q(y)$) in such a way that would minimize (maximize) the expected CR. The cost incurred by the designer is a function of the number of snow days, which is controlled by the adversary and is not known by the designer, and the randomized strategy applied by the designer. The designer's strategy is an online algorithm as the designer acts without the knowledge of the number of snow days. We will denote the expected cost incurred by the designer by $C(p(x), y)$. Let OPT(y) denote the cost incurred by an optimal offline algorithm. With these definitions, we can now write the CR as:

$$c = \frac{C(p(x), y)}{\text{OPT}(y)}.$$

Let us first determine the possible values for OPT(y). If $y \leq B$, then the strategy that minimizes the overall cost is renting for the period $[0, y]$. On the other hand, if $y > B$, then it is optimal to buy the skis. Formally, we have

$$\text{OPT}(y) = \begin{cases} y, & \text{if } y \leq B, \\ B, & \text{otherwise.} \end{cases} \quad (1)$$

[1]The results we present here apply to problems with any renting price and setting it to unity is merely a scaling adopted for simplicity.

The value of $C(p(x), y)$ depends on when the designer decides to buy the skis. If $x \leq y$, then the designer will have to pay $\$x$ for the rental period in addition to the buying price of $\$B$. However, if $y < x$, then the designer will not have to buy the skis and will make a payment of $\$y$ as renting fees. Hence, for $y \leq B$, the expected cost can be written as

$$C(p(x), y) = \int_0^y (x + B)p(x)dx + \int_y^B yp(x)dx. \quad (2)$$

The case when $y > B$ will be discussed in the next section.

Because the objectives of the designer and the adversary are conflicting, it is natural to use game theory to derive the optimal strategies for both players. In Section III-B, we will formulate the problem as a continuous-kernel zero-sum game. However, before we indulge in formulating and analyzing the continuous-kernel zero-sum game, let us first consider a discretized version of the game, i.e., we will first formulate a matrix zero-sum game. This will aid us in understanding the decision process of the designer and the adversary.

A. Matrix Zero-Sum Game

Assume that the (pure) strategy space for both the designer and the adversary is the countably infinite set $\{1, 2, 3, \cdots\}$. Let $A = [A_{ij}]$ be the matrix of the zero-sum game with the designer being the row player and the adversary being the column player.

$$\underbrace{\begin{pmatrix} B & \frac{B}{2} & \frac{B}{3} & \cdots & \frac{B}{B-1} & 1 & 1 & 1 & \cdots \\ 1 & \frac{B+1}{2} & \frac{B+1}{3} & \cdots & \frac{B+1}{B-1} & \frac{B+1}{B} & \frac{B+1}{B} & \frac{B+1}{B} & \cdots \\ 1 & 1 & \frac{B+2}{3} & \cdots & \frac{B+2}{B-1} & \frac{B+2}{B} & \frac{B+2}{B} & \frac{B+2}{B} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & \cdots & 1 & \frac{2B-1}{B} & \frac{2B-1}{B} & \frac{2B-1}{B} & \cdots \\ 1 & 1 & 1 & \cdots & 1 & 1 & 2 & 2 & \cdots \\ 1 & 1 & 1 & \cdots & 1 & 1 & \frac{B+1}{B} & \frac{2B+1}{B} & \cdots \\ 1 & 1 & 1 & \cdots & 1 & 1 & \frac{B+1}{B} & \frac{B+2}{B} & \cdots \\ 1 & 1 & 1 & \cdots & 1 & 1 & \frac{B+1}{B} & \frac{B+2}{B} & \cdots \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \end{pmatrix}}_{:=A}$$

The i-th row corresponds to the case where the designer chooses to rent the skis for $i-1$ days and buy the skis on the i-th day. The j-th column corresponds to the adversary choosing the number of snow days to be j. Hence, the (i, j)-th element of A is the CR corresponding to the designer choosing i and the adversary choosing j.

By studying the matrix game A, we notice that the $(B+1)$-st column dominates the B-th column, i.e., $A_{i(B+1)} \geq A_{iB}$ with $A_{i(B+1)} > A_{iB}$ for at least one i. In fact, the $(B+j)$-th column dominates the $(B+j-1)$-th column for $j \geq 1$. Hence, we can remove the dominated columns from the matrix, and the resulting matrix will be strategically equivalent to A [8]. After removing the dominated columns, we readily see that the B-th row dominates the $(B+i)$-th rows, $i \geq 1$, i.e., $A_{Bj} \leq A_{(B+i)j}$ with $A_{Bj} < A_{(B+i)j}$ for at least one j, $i \geq 1$. After

removing the dominated rows, the resulting matrix game $\tilde{A}$ can be written as

$$\tilde{A} = \begin{pmatrix} B & \frac{B}{2} & \frac{B}{3} & \cdots & \frac{B}{B-1} & 1 \\ 1 & \frac{B+1}{2} & \frac{B+1}{3} & \cdots & \frac{B+1}{B-1} & \frac{B+1}{B} \\ 1 & 1 & \frac{B+2}{3} & \cdots & \frac{B+2}{B-1} & \frac{B+2}{B} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & \cdots & 1 & \frac{2B-1}{B} \end{pmatrix}$$

Note that the first B rows in the $(B+j)$-th, $j \geq 1$, columns have the same values. Hence, the adversary will exhibit the same performance regardless of which column it chooses, as long as $j > B$. It is important to note that, by strict dominance, we were able to convert an infinite game into a finite one. More importantly, one can obtain an exact equilibrium for the game and does not need to construct an ϵ-equilibrium as the performance the adversary achieves as $j \to \infty$ is identical to what it achieves when $j = B + k$, $k \geq 1$, $k \in \mathbb{N}$.

B. Continuous-Kernel Zero-Sum Game

We can obtain insights from $\tilde{A}$ when deriving the strategies of the designer and the adversary in the continuous-time case. In the discrete-time case, the strategy space of the designer reduces to $\{1, 2, \cdots, B\}$. Hence, in the continuous-time case, the designer needs only to assign probabilities over the interval $[0, B]$. We can then write the designer's strategy space as

$$\mathcal{P} = \left\{ p(x) \geq 0, x \in [0, B] : \int_0^B p(y)dy = 1 \right\}.$$

However, the situation is different for the adversary as its strategy space in the discrete-time case becomes $\{1, 2, \cdots, B-1, K\}$, where $K > B$. Hence, the adversary must construct a two-part randomized strategy: a probability density over the interval $[0, B)$ and a probability mass at $K \geq B$. We will denote the probability mass at K by $q_K = q(y = K)$. Also, we will consider two different constraints on the adversary's strategy. In Section IV-A, we will assume that the adversary has a constraint on the first moment. Formally, we can write

$$\int_0^B yq(y)dy + q_K \cdot K = \mu. \tag{3}$$

The strategy space of the adversary in this case is

$$\mathcal{Q} = \{q(y) \geq 0, y \in [0, B) \cup K, K \geq B : \int_0^B q(y)dy + q_K = 1 \text{ and (3) is satisfied} \}.$$

In Section IV-B, we will replace the constraint on the first moment with one on the second moment as follows:

$$\int_0^B y^2 q(y)dy + K^2 q_K = \sigma. \tag{4}$$

We now formulate the zero-sum game played by the designer and the adversary. The objective function of the designer is the *expected* CR denoted $J(p, q)$ – it follows that the objective function of the adversary is $-J(p, q)$. Hence, it is the average, with respect to $q(y)$, of the CR for $y \in [0, B)$ and $y = K$. Because $K \geq B$, and $x \in [0, B]$, we conclude that the cost incurred by the player will be $x + B$ at K. Thus, using (1) and (2), we can write the expected CR as

$$\begin{aligned} J(p, q) &= \mathbb{E}_q[c] \\ &= \int_0^B \frac{C(p(x), y)}{y} q(y)dy + q_K \cdot \int_0^B \frac{x + B}{B} p(x)dx. \end{aligned}$$

We will denote the zero-sum game by $\mathcal{G} = \{\mathcal{P}, \mathcal{Q}, J\}$. The solution concept we adopt in studying $\mathcal{G}$ is the mixed-strategy saddle-point equilibrium defined below.

Definition 1: The pair $(p^\star, q^\star)$ constitutes a saddle-point equilibrium in mixed-strategies for $\mathcal{G}$ if

$$J(p^\star, q) \leq J(p^\star, q^\star) \leq J(p, q^\star),$$

for any $p \in \mathcal{P}$ and $q \in \mathcal{Q}$. ■

By Von Neumann's minimax theorem [8], we know that

$$\min_{p(x) \in \mathcal{P}} \max_{q(y) \in \mathcal{Q}} J(p, q) = \max_{q(y) \in \mathcal{Q}} \min_{p(x) \in \mathcal{P}} J(p, q). \tag{5}$$

In the following, we will make use of this fact to derive the optimal mixed-strategies for both players.

IV. The Designer's Problem

The designer's optimal strategy $p^\star(x)$ can be obtained by solving the following problem:

$$\min_{p(x) \in \mathcal{P}} \max_{q(y) \in \mathcal{Q}} J(p, q).$$

To solve this problem, we will take the following steps:

1. We first construct the dual to the maximization problem. The dual turns out to be a linear program (LP) with two equality constraints.
2. By differentiating one of the equality constraints twice, we obtain a first-order ODE which can be used, along with the fact that $p(x)$ is a probability distribution function (PDF), to obtain $p(x)$ as a function of the Lagrange multiplier associated with constraint on $q(y)$.
3. By substituting the obtained PDF into the original equality constraints, we obtain an LP in the Lagrange multipliers which can then be readily solved.

Note that in Step 3, we manage to convert an infinite dimensional optimization problem (as we were originally solving for $p(x)$) to a finite scalar optimization problem.

We will first derive $p^\star(x)$ for the case when μ is constrained. Then, we will proceed to the case when σ is constrained.

A. First-Moment-Constrained Ski-Rental Problem

We will start by constructing the dual problem. The Lagrangian associated with the maximization problem is

$$\mathcal{L}(q(y), \lambda_1, \lambda_2) = \int_0^B \underbrace{\left(\frac{C(p(x), y))}{y} - \lambda_1 - \lambda_2 y \right)}_{:= h_1(y)} q(y)dy$$
$$+ q_K \cdot \underbrace{\left(\int_0^B \frac{x + B}{B} p(x)dx - \lambda_1 - \lambda_2 K \right)}_{:= h_2} + (\lambda_1 + \lambda_2 \mu).$$

The dual function $g(\lambda_1,\lambda_2) = \sup_{q(y)\in\mathcal{Q}} \mathcal{L}(q(y),\lambda_1,\lambda_2)$ is therefore given by

$$g(\lambda_1,\lambda_2) = \begin{cases} \lambda_1+\lambda_2\mu, & \text{if } h_1(y)=0, h_2=0, \quad \forall y,\\ \infty, & \text{otherwise.}\end{cases}$$

Hence, after adding the constraints on $p(x)$, the dual becomes

$$\min_{p(x)\in\mathcal{P},\lambda_1,\lambda_2} \lambda_1+\lambda_2\mu \tag{6}$$

$$\text{s.t. } \frac{C(p(x),y)}{y} = \lambda_1+\lambda_2 y, \tag{7}$$

$$\int_0^B \frac{x+B}{B}p(x)dx = \lambda_1+\lambda_2 K, \tag{8}$$

$$\forall y\in[0,B], \quad \lambda_1,\lambda_2\geq 0.$$

Because (7) holds for all y, we can differentiate both sides twice with respect to y and replace y with x to obtain

$$\frac{d}{dx}p(x) = \frac{1}{B}(p(x)+2\lambda_2).$$

This is a first-order ODE whose solution is

$$p(x) = \alpha e^{\frac{x}{B}} - 2\lambda_2. \tag{9}$$

To solve for α, we use the fact that $p(x)$ is a PDF to obtain

$$\alpha = \frac{1+2\lambda_2 B}{B(e-1)}.$$

By substituting (9) into (7) and (8), we obtain the following equivalent conditions:

$$\left(2\frac{2-e}{e-1}B\right)\lambda_2 + \frac{e}{e-1} = \lambda_1, \tag{10}$$

$$\left(\frac{3-e}{e-1}B - K\right)\lambda_2 + \frac{e}{e-1} = \lambda_1. \tag{11}$$

Further, we must have $p(x)\geq 0$, for $x\in[0,B]$, or equivalently

$$\left(1-\frac{e^{\frac{x}{B}}}{e-1}\right)\lambda_2 \quad\leq\quad \frac{e^{\frac{x}{B}}}{2B(e-1)}.$$

Hence, requiring the PDF to be positive imposes the following constraints on λ_2:

$$\lambda_2 \quad\leq\quad \frac{e^{\frac{x}{B}}}{2B(e-1-e^{\frac{x}{B}})}, \quad \text{if } 0\leq x\leq B\log(e-1), \tag{12}$$

$$\lambda_2 \quad>\quad \frac{e^{\frac{x}{B}}}{2B(e-1-e^{\frac{x}{B}})}, \quad \text{if } B\log(e-1)< x\leq B. \tag{13}$$

Note that the right hand side (RHS) of (12) is positive and strictly increasing for $x\in[0,B\log(e-1))$, whereas the RHS of (13) is negative for $x\in(B\log(e-1),B]$. Therefore, we must have

$$0\leq\lambda_2\leq\frac{1}{2B(e-2)}.$$

The designer's problem is thus equivalent to the following LP:

$$\max_{\lambda_1,\lambda_2} \quad \lambda_1+\lambda_2\mu \tag{14}$$

$$\text{s.t.} \quad \left(2\frac{2-e}{e-1}B\right)\lambda_2+\frac{e}{e-1}=\lambda_1,$$

$$\left(\frac{3-e}{e-1}B-K\right)\lambda_2+\frac{e}{e-1}=\lambda_1,$$

$$\lambda_1\geq 0, \quad 0\leq\lambda_2\leq\frac{1}{2B(e-2)}.$$

By the fundamental theorem of LPs, we know that the solutions to this LP form a convex polytope, and that each basic feasible solution (λ_1,λ_2) is a corner point of the polytope (and vice versa). Note that if $\lambda_2>0$, we must have $B=K$ in order to satisfy (10) and (11) simultaneously. Hence, we have two corner points: $b_1 = \left(\frac{e}{e-1},0\right)$ and $b_2 = \left(1,\frac{1}{2B(e-2)}\right)$. The corresponding values to these points are:

$$(\lambda_1+\lambda_2\mu)\Big|_{b_1} = \frac{e}{e-1}, \ (\lambda_1+\lambda_2\mu)\Big|_{b_2} = 1+\frac{\mu}{2B(e-2)}. \tag{15}$$

These values are the resulting expected CRs. Note that when $\lambda_2=0$, the problem becomes a classical Ski-Rental problem without a constraint on the mean whose optimal expected CR under a randomized algorithm is known to be $\frac{e}{e-1}$, which is what we obtain under b_1. By comparing the obtained values at b_1 and b_2, we conclude that if $\frac{\mu}{B}\leq 2\frac{e-2}{e-1}$, the optimal PDF is

$$p^\star(x) = \begin{cases} \frac{1}{B(e-2)}(e^{\frac{x}{B}}-1), & 0\leq x\leq B,\\ 0, & \text{otherwise.}\end{cases} \tag{16}$$

Otherwise, the optimal solution is

$$p^\star(x) = \begin{cases} \frac{1}{B(e-1)}e^{\frac{x}{B}}, & 0\leq x\leq B,\\ 0, & \text{otherwise.}\end{cases} \tag{17}$$

This is it to be expected; when the first moment is high (more precisely, when $\mu>2\frac{e-2}{e-1}B$), knowing its value does not provide the designer with extra information. Hence, the optimal randomized strategy becomes **PROB**. Aside from the fact that the value of μ decides which PDF is optimal, it is interesting to note that (16) is independent of μ. Thus, one can use this optimal PDF *without* needing to compute μ.

B. Second-Moment-Constrained Ski-Rental Problem

We can perform similar analysis by replacing the constraint on the first moment μ with one on the second moment σ of the adversary's strategy. Accordingly, we replace (3) with (4) in the definition of $\mathcal{Q}$. By following the same steps leading to (14), we can obtain the following LP to be solved by the designer:

$$\max_{\lambda_1,\lambda_2} \quad \lambda_1+\lambda_2\sigma$$

$$\text{s.t.} \quad \left(3\frac{5-2e}{e-1}B^2\right)\lambda_2+\frac{e}{e-1}=\lambda_1,$$

$$\left(\frac{14-5e}{e-1}B^2-K^2\right)\lambda_2+\frac{e}{e-1}=\lambda_1,$$

$$\lambda_1\geq 0, \quad 0\leq\lambda_2\leq\frac{1}{3B^2(2e-5)}.$$

It readily follows that if $\lambda_2 > 0$, we must have $B = K$ in order to satisfy the equality constraints simultaneously. Also, we have two corner points: $\tilde{b}_1 = \left(\frac{e}{e-1}, 0\right)$ and $\tilde{b}_2 = \left(1, \frac{1}{3B^2(2e-5)}\right)$. The corresponding expected CRs are:

$$(\lambda_1 + \lambda_2\sigma)\Big|_{\tilde{b}_1} = \frac{e}{e-1}, \ (\lambda_1 + \lambda_2\sigma)\Big|_{\tilde{b}_2} = 1 + \frac{\sigma}{3B^2(2e-5)}. \tag{18}$$

Hence, if $\frac{\sigma}{B^2} \leq 3\frac{2e-5}{e-1}$, the optimal strategy becomes

$$p^\star(x) = \begin{cases} \frac{2}{B(2e-5)}\left(e^{\frac{x}{B}} - \frac{x+B}{B}\right), & 0 \leq x \leq B, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Otherwise, the optimal solution is the PDF corresponding to **PROB** given in (17). We again notice that the optimal strategy is independent of the extra information the designer possesses, namely the second moment σ. Fig. 1 shows the distributions of **PROB**, μ-**PROB**, and σ-**PROB**.

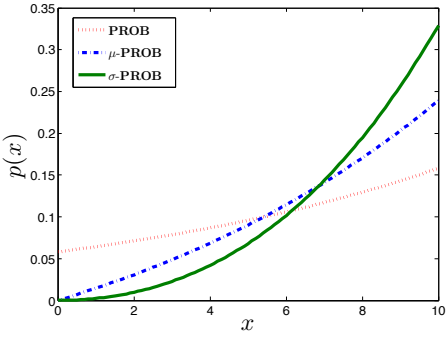

Fig. 1. Depiction of the PDFs of **PROB**, μ-**PROB**, and σ-**PROB**.

C. Performance Comparison

We will first compare the four schemes (**DET**, **PROB**, μ-**PROB**, and σ-**PROB**) based on their CR performance for all arrival values. Fig. 2 compares the CR performance of the four schemes for $B = 10$. For the proposed methods, the curves were computed using the LHS of (7) for $y \leq B$ and the LHS of (8) for $y > B$. We conclude that our approach strikes a balance between the CR performance of **DET** and **PROB**. Also, σ-**PROB** outperforms μ-**PROB** for $y \leq B$. As we impose constraints on higher moments, we expect that the performance will be further improved over this interval. This improvement is accompanied by a slight deterioration in performance over the interval $y > B$.

Let us compare the CR achieved by μ-**PROB**, σ-**PROB**, and that achieved by **PROB** for $y > B$. We first compute

$$(\lambda_1 + \lambda_2 B)\Big|_{b_2} = \frac{2e-3}{2(e-2)}, \ (\lambda_1 + \lambda_2 B^2)\Big|_{\tilde{b}_2} = \frac{6e-14}{3(2e-5)}.$$

Hence, we find that the difference between the performance of μ-**PROB**, σ-**PROB**, and **PROB** to be

$$\frac{2e-3}{2(e-2)} - \frac{e}{e-1} = 0.114, \ \frac{6e-14}{3(2e-5)} - \frac{e}{e-1} = 0.182.$$

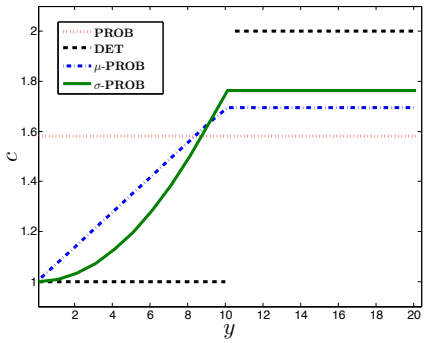

Fig. 2. The CR of the four algorithms. The buying price B is \$10.

We readily see that our scheme exhibits comparable performance to **PROB** for $y > B$.

Now, we average over the arrival values and compare the schemes based on their worst-case expected CR. For **PROB**, we already know that the expected CR is $\frac{e}{e-1}$. In order to compare our methods with **DET**, we need the following lemmas.

Lemma 1: In the First-Moment-Constrained Ski-Rental problem, the worst-case expected CR for **DET** is at least $1+\frac{\mu}{B}$ when $0 \leq \mu \leq B$, and it is at least 2 when $\mu > B$.

Proof: Because we are looking for a lower bound on the worst-case CR, it suffices to find a distribution $\hat{q}(y)$ that yields the proclaimed values. Formally, we have

$$\max_{q(y)\in\mathcal{Q}} \min_{p(x)\in\mathcal{P}} J(p,q) \geq \min_{p(x)\in\mathcal{P}} J(p,\hat{q}). \tag{20}$$

Consider the following candidate distribution for $0 \leq \mu \leq B$:

$$\hat{q}(y) = \begin{cases} 1 - \frac{\mu}{B}, & y = 0, \\ \frac{\mu}{B}, & y = B^+, \\ 0, & \text{otherwise,} \end{cases}$$

where $B^+ = B + \epsilon$, $\epsilon > 0$ is small. This distribution clearly satisfies the first moment constraint as $\epsilon \to 0$. Assuming **DET** is used, we can now compute $\mathbb{E}_{\hat{q}} = 1\cdot\left(1 - \frac{\mu}{B}\right) + 2\cdot\frac{\mu}{B} = 1 + \frac{\mu}{B}$. Similarly, to obtain $\mathbb{E}_{\hat{q}}[c] = 2$ when $\mu > B$, we can choose:

$$\hat{q}(y) = \begin{cases} 1, & y = \mu, \\ 0, & \text{otherwise.} \end{cases}$$

■

Lemma 2: In the Second-Moment-Constrained Ski-Rental problem, the worst-case expected CR for **DET** is at least $1 + \frac{\sigma}{B^2}$ when $0 \leq \sigma \leq B^2$, and it is at least 2 when $\sigma > B^2$.

Proof: The proof is similar to that of Lemma 1. The PDF guaranteeing a worst-case CR of $1+\frac{\sigma}{B^2}$ when $0 \leq \sigma \leq B^2$ is

$$\hat{q}(y) = \begin{cases} 1 - \frac{\sigma}{B^2}, & y = 0, \\ \frac{\sigma}{B^2}, & y = B^+, \\ 0, & \text{otherwise,} \end{cases}$$

and that yielding an expected CR of 2 when $\sigma > B^2$ is

$$\hat{q}(y) = \begin{cases} 1, & y = \sqrt{\sigma}, \\ 0, & \text{otherwise.} \end{cases}$$

■

From Lemmas 1 and 2, (15) and (18), and by comparing the obtained expected CRs for the schemes at hand, we can draw the following conclusions:

- When $0 \leq \mu \leq 2\frac{e-2}{e-1}B$ $\left(\text{or } 0 \leq \sigma \leq 3\frac{2e-5}{e-1}B^2\right)$, μ-**PROB** (or σ-**PROB**) always outperforms **DET** and **PROB**;
- When $\mu > 2\frac{e-2}{e-1}B$ $\left(\text{or } \sigma > 3\frac{2e-5}{e-1}B^2\right)$, **PROB** outperforms **DET**, because $\frac{e}{e-1} < 1 + \frac{\mu}{B}$ $\left(\text{or } \frac{e}{e-1} < 1 + \frac{\sigma}{B^2}\right)$ for these values of μ, and $\frac{e}{e-1} = 1.582 < 2$. However, note that the optimal solution obtained using our methodology *defaults* to **PROB** over this range as can be seen from the conditions leading to (16) and (17) (or (19)).

Hence, we conclude that for any value of μ (or σ) our approach produces the policy yielding the best worst-case expected CR. Fig. 3 demonstrates this fact for $B = 10$. In order to rigorously

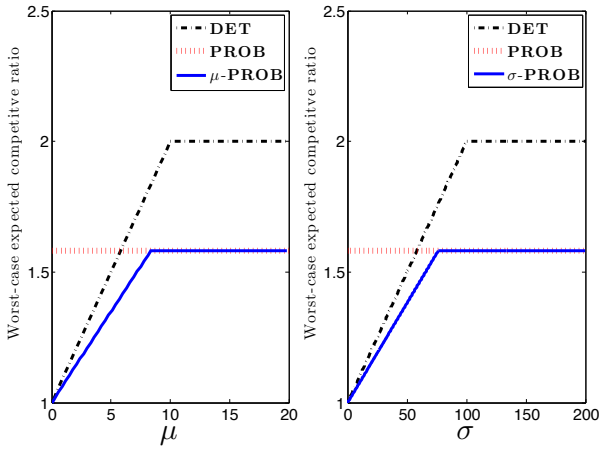

Fig. 3. The worst-case expected CR for the four algorithms. $B = 10$.

show that this is the best possible CR, we generate an arrival time (number of ski days) distribution such that the bound given by the algorithm μ-**PROB** is tight. In order to do this, we solve the adversary's problem.

V. The Adversary's Problem

Let us restrict our attention to the case when only the first moment of $q(y)$ is constrained[2]. The adversary attempts to solve the following problem:

$$\max_{q(y)\in\mathcal{Q}} \min_{p(x)\in\mathcal{P}} J(p,q).$$

We will take the following steps to solve this problem:

1. We first construct the dual to the minimization problem. The dual is shown to be an LP with an equality constraint.
2. By differentiating the equality constraint twice, we obtain a first-order ODE which can be used to solve for $q(y)$.
3. By evaluating the equality constraint at specific values and by using Von Neumann's theorem, we can solve for q_K. To obtain a range of possible values for K, we use the constraint on the first moment.

[2]The adversary's problem under a second moment constraint is similar to the one with a constraint on the mean and is omitted due to space limitation.

We proceed by constructing the dual problem. By following similar steps to the above, we obtain:

$$\max_{q(y)\in\mathcal{Q},\lambda} \quad \lambda \tag{21}$$

$$\text{s.t.} \quad g(x) + q_K \cdot \frac{x+B}{B} = \lambda, \tag{22}$$

$$g(x) = \int_0^x q(y)dy + \int_x^B \frac{x+B}{y} q(y)dy,$$

$$\forall x \in [0,B], \quad \lambda \geq 0.$$

By differentiating (22) twice, we obtain the following ODE:

$$\frac{d}{dy}q(y) = \frac{B-y}{By}q(y),$$

whose solution is given by

$$q(y) = \beta y e^{-\frac{y}{B}}.$$

To fully characterize $q^\star(y)$, we need to solve for β, K, and q_K. Using the fact that $q(y)$ must integrate to $1 - q_K$, we get

$$\beta = \frac{1-q_K}{B^2\left(1-\frac{2}{e}\right)}.$$

To obtain q_K, we invoke Von Neumann's theorem and make use of the fact that the values of (6) and (21) must be equal, because they are dual to each other. When $\frac{\mu}{B} > 2\frac{e-2}{e-1}$, we have $\lambda^\star = \frac{e}{e-1}$. By evaluating (22) at $x = B$, we obtain

$$q_K = \frac{1}{e-1}.$$

Because in this case the mean is large, and the designer does not use her knowledge of the mean, the adversary can relax the constraint on the mean and K can be chosen freely. In the case when $\frac{\mu}{B} \leq 2\frac{e-2}{e-1}$, we know that $\lambda^\star = 1 + \frac{\mu}{2B(e-2)}$, and hence

$$q_K = \frac{\mu}{2B(e-2)}.$$

Here, the adversary needs to satisfy the mean constraint by selecting K properly. Using (3), we get

$$K^\star = \frac{2(e-2)^2 + (2e-5)}{e-2}B - 2(2e-5)\frac{B^2}{\mu}.$$

If we select $K^\star \geq B$, then we must have $\frac{\mu}{B} > 2\frac{e-2}{e-1}$. But this again becomes the case where the information about the mean does not benefit the designer. Hence, the interesting case to consider is when $0 \leq K^\star \leq B$. This translates to requiring:

$$\frac{2(2e-5)(e-2)}{e(e-2)^2 + (2e-5)} \leq \frac{\mu}{B} \leq 2\frac{e-2}{e-1}.$$

Hence, the adversary's optimal strategy when $\frac{\mu}{B} \leq 2\frac{e-2}{e-1}$ is

$$q^\star(y) = \begin{cases} \frac{2B(e-2)-\mu}{2B^3(e-2)\left(1-\frac{2}{e}\right)} y e^{\frac{y}{B}}, & 0 \leq y < B, \\ \frac{\mu}{2B(e-2)}, & y = K^\star, \\ 0, & \text{otherwise.} \end{cases} \tag{23}$$

Otherwise, the optimal solution is

$$q^\star(y) = \begin{cases} \frac{y}{B^2(e-1)} e^{1-\frac{y}{B}}, & 0 \le y < B, \\ \frac{1}{e-1}, & y = K^\star, \\ 0, & \text{otherwise.} \end{cases}$$

The following theorem amalgamates what we have shown in Sections IV and V.

Theorem 1: When imposing a constraint on the first (or second) moment of the adversary's strategy, the best possible worst-case expected CR that can be achieved is $1 + \frac{\mu}{2B(e-2)}$ $\left(\text{or } 1 + \frac{\sigma}{3B^2(2e-5)}\right)$, for $0 \le \mu \le 2\frac{e-2}{e-1}B$ $\left(\text{or } 0 \le \sigma \le 3\frac{2e-5}{e-1}B^2\right)$. In both cases, the achieved CR outperforms that of **DET** and **PROB**.

Proof: The proof follows from Von Neumann's minimax theorem. In Section IV we have derived the optimal strategy of the designer under the two different constraints. Their corresponding performance was shown in (15) and (18). Further, we have derived the optimal adversarial strategy in (23) under the first moment constraint which achieves (15). By (5), we conclude that the derived optimal strategies will yield the best worst-case expected CR. Also, Lemmas 1 and 2 and the subsequent arguments show that the obtained expected CR outperforms that of **DET** and **PROB**. ■

VI. Numerical Results

We now present a detailed evaluation of our scheme using simulations as well as synthetic and real-world file system workloads.

A. Simulation Results

The simulations are based directly on our analysis in the previous sections. Theorem 1 states that our approach guarantees the best possible worst-case expected CR. However, by (20), we conclude that when the arrivals distribution is not selected optimally, we obtain a lower bound on the worst-case expected CR. Here, we simulate this phenomenon using three distributions: uniform, exponential, and log-normal. From Fig. 3, we notice that μ-**PROB** and σ-**PROB** exhibit similar performance. Hence, we will will only show simulations for μ-**PROB** in the rest of this section.

Fig. 4 plots $\mathbb{E}_q[c]$ for different values of μ when the arrivals are uniformly distributed over $[0, 2\mu]$ and $B = 10$. We see that **DET** exhibits optimal performance for $2\mu \le B$; this is because the arrivals distribution in this case falls entirely in the interval where the CR of **DET** is 1. However, as the value of μ increases, the CR of **DET** becomes 2, and our approach outperforms **DET** eventually. Further, our approach outperforms **PROB** for small μ values.

Fig. 5 depicts the same simulation but for an exponential distribution with parameter $\frac{1}{\mu}$ and $B = 25$. We can again see that μ-**PROB** outperforms **PROB** for small values of μ. We find that **DET** outperforms our approach for small values of μ. This is to be expected since the exponential distribution places most of its weight on the interval $[0, B]$ over which **DET** has a CR of 1. Note, however, that as μ increases, the exponential distribution places more weight outside $[0, B]$ and the performance of **DET** worsens.

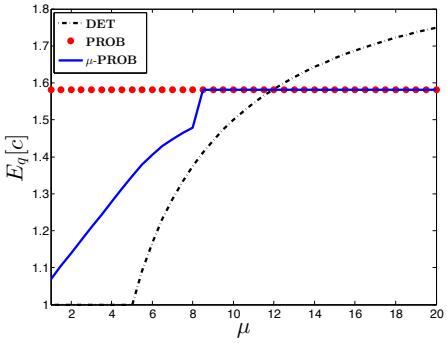

Fig. 4. Expected CR for $\mathcal{U}[0, 2\mu]$-distributed arrivals. $B = 10$.

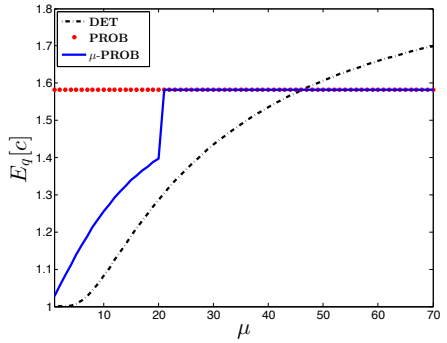

Fig. 5. Expected CR for $\text{Exp}\left(\frac{1}{\mu}\right)$-distributed arrivals. $B = 25$.

Finally, Fig. 6 simulates a log-normal arrival distribution with a standard deviation of 0.05. Similar to the above two cases, μ-**PROB** dominates **DET** as the value of μ increases. From the above three experiments, we conclude that our scheme exhibits an intermediate performance between **DET** and **PROB**: it outperforms **PROB** for small values of μ and outperforms **DET** for large μ values.

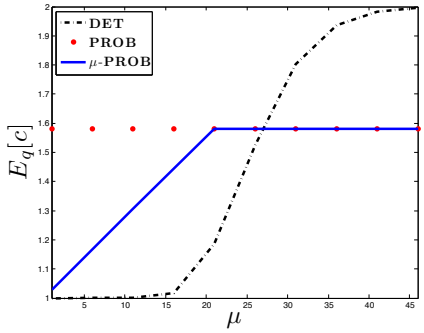

Fig. 6. Expected CR for Log-$\mathcal{N}(\mu, 0.05)$-distributed arrivals. $B = 25$.

B. Cloud File System Based Evaluation

We evaluate a scenario in which a file system is running on the cloud, and it is using two types of storage resources: a disk (akin to Amazon S3) and a cache (Amazon EC2 VM instance memory). The disk I/O is costlier than the cache I/O, while the cache storage cost is more expensive than that of the disk. We set $B = 10$ to indicate that the disk I/O per-block is ten times costlier than storing the block for one unit of time (sec). In this file system, the requests to the file arrive at various points in time, and the server chooses between storing the response in the cache and fetching it fresh from the disk based on **DET**, **PROB**, or μ-**PROB**. Finally, note that while our analysis focused on the CR, our evaluation using file system traces focuses on the total cost.

Synthetic Workloads. We generated two types of synthetic workloads based on the inter-arrival times of the requests. In Fig. 7, we adopt fixed inter-arrival times and measure the cost of running the file system on the cloud for different arrival values. In Fig. 8, we repeat the same experiment with exponentially distributed inter-arrival times and measure the average-cost for different mean values. The figures show that **DET** is the best algorithm for arrivals occurring before time B, while **PROB** is the best for arrivals after time B. They also highlight that μ-**PROB** is quite *robust*; it attempts to approximate the best of the two schemes for different arrival values.

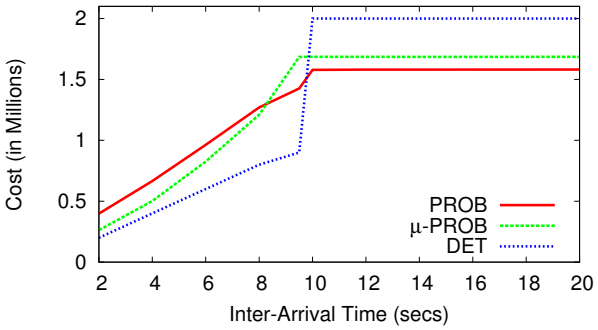

Fig. 7. Cost for arrivals at fixed intervals. $B = 10$.

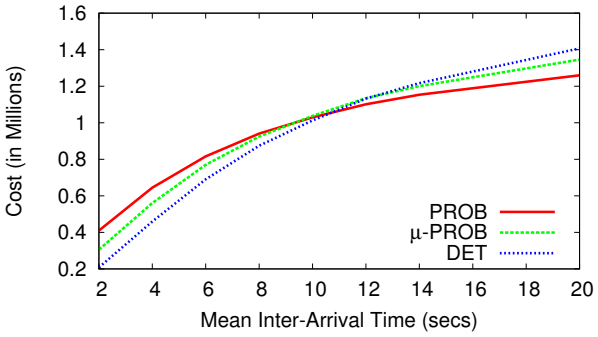

Fig. 8. Average-cost for exponentially distributed inter-arrival times. $B = 10$.

Real Workload. We used the publicly-released workloads of a cloud file system from a prior study by Narayanan *et al.* [9], [10]. There are traces from 36 file systems hosted in an enterprise data center at Microsoft. Each trace contains the arrival requests for disk blocks for a week during February 2008. Using these traces, we ran a cloud file system using ElastiCache and S3 with the pricing values as shown in Table I. For 4 KB as the file system block size, the cost values in Table I will set B to 47 hours. We ran all the file system traces with $B = 47$ and computed the total cost of all the file systems. Due to space limitation, we only present a summary of the results without showing any graphs. The total cost of running these file system traces are: \$1509 for **DET**, \$1345 for **PROB**, and \$1410 for μ-**PROB**. Again, μ-**PROB** provides a robust performance as it reduces the difference between the cheapest and the costliest schemes by nearly 60%.

VII. Conclusion

Many cloud cost optimization problems can be abstracted as a Ski-Rental problem. Existing research has developed deterministic and probabilistic algorithms to tackle the Ski-Rental problem. The Ski-Rental problem is usually studied without exploiting common information about the application workload. In this paper, we introduced a new variant of the problem called the Constrained Ski-Rental problem which assumes that the first (or second) moment of the arrivals distribution is known to the algorithm designer. We demonstrated that using this limited information can lead to a class of randomized algorithms that provide the best arrivals-distribution-free performance guarantees, because they outperform existing approaches in the worst-case expected CR sense. By applying the proposed scheme to cloud file systems, we have shown that it can lead to significant cost savings. Because of the growing importance of optimizing the costs in the cloud, we believe many other applications can benefit from our findings.

References

[1] K. P. Puttaswamy, T. Nandagopal, and M. Kodialam, "Frugal storage for cloud file systems," in *Proc. ACM European Conf. Computer Systems*, 2012, pp. 71–84.

[2] A. Kathpal, M. Kulkarni, and A. Bakre, "Analyzing compute vs. storage tradeoff for video-aware storage efficiency," in *Proc. Workshop on Hot Topics in Storage and File Systems*, 2012.

[3] A. R. Karlin, M. S. Manasse, L. Rudolph, and D. D. Sleator, "Competitive snoopy caching," in *Proc. Symp. Foundations of Computer Science*, October 1986, pp. 244–254.

[4] H. Fujiwara and K. Iwama, "Average-case competitive analyses for Ski-Rental problems," *Algorithmica*, vol. 42, no. 1, pp. 95–107, 2005.

[5] Z. Lotker, B. Patt-Shamir, and D. Rawitz, "Rent, lease or buy: Randomized algorithms for multislope ski rental," *arXiv:1009.3522*, 2008.

[6] R. Motwani and P. Raghavan, *Randomized algorithms*. New York, NY, USA: Cambridge University Press, 1995.

[7] A. R. Karlin, M. S. Manasse, L. A. McGeoch, and S. Owicki, "Competitive randomized algorithms for non-uniform problems," in *Proc. ACM-SIAM Symp. Discrete Algorithms*, 1990, pp. 301–309.

[8] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory*. SIAM Series in Classics in Applied Mathematics, 1999.

[9] D. Narayanan, A. Donnelly, and A. Rowstron, "Write off-loading: practical power management for enterprise storage," in *Proc. USENIX Conf. File and Storage Technologies*, 2008, pp. 256–267.

[10] "MSR Cambridge Traces," http://iotta.snia.org/traces/388.