Cost-Effective Resource Configuration for Cloud Video Streaming Services

Yunyun Jiang*, Xiaosong Ma† and Wenguang Chen*
*Department of Computer Science and Technology, Tsinghua University, Beijing, China
†Qatar Computing Research Institute, Hamad Bin Khalifa University, Doha, Qatar
Emails: jiangyy09@mails.tsinghua.edu.cn, xma@qf.org.qa, cwg@tsinghua.edu.cn

***Abstract*—Video streaming services are migrating to cloud environments for the economic expense with good scalability. However, cloud providers offer flexible resource configurations, e.g., on-demand, reserved and spot instances, with significant different pricing policies, of which one single configuration is suboptimal for cloud video streaming services. In this paper, we propose hybrid configuration schemes of cloud video streaming services to reduce the cost. To achieve this goal, we first introduce a lightweight prediction algorithm to predict the future video traffic. With the predicted video traffic, we then give the *Hybrid-R* hybrid configuration scheme by configuring both on-demand and reserved instances, and the *Hybrid-RS* hybrid configuration scheme by further configuring spot instances. Our evaluations using traces from real video service providers show that our configuration schemes can reduce cost by at least 20% compared to the unoptimized ones with negligible overhead.**

I. Introduction

In the past few years, the scale of video streaming services is dramatically increased [1, 2], due to the popularity of both internet services, e.g. Youtube, and online education services, e.g. MOOCs. While the scale is increased in a long term (month by month) and the user requests varies in a short time (hour by hour), it is an economical way to migrate video streaming services to the cloud, leveraging the benefit of the flexible on-demand (pay-as-you-go) cost. A number of video service providers have gained benefits from this trend, e.g., Netflix has started to transfer its storage services to Elastic Compute Cloud (EC2) of Amazon [3].

While video services can have a lower cost with the on-demand configurations in the cloud, there are still margins to better use the flexible configurations offered by cloud platforms. For instance, Amazon EC2 provides three kinds of virtual machine (VM) instances, i.e., on-demand, reserved and spot instances, with different pricing policies [4]. The *on-demand* instance provides traditional pay-as-you-go pricing policy. The *reserve* instance offers a discount with proper mount of upfront fee. And the *spot* instance offers the lowest price, but no guarantee on available resource. These flexible pricing policies bring opportunities for reducing the instance renting cost of video streaming services when properly configured.

To investigate the benefits from the flexible pricing policies, we study the access pattern of video traffic, and thereby evaluate the cost of the instance configuration offered by EC2. By studying both results from research paper [1] and statistical data published on mainstream VoD (Video on Demand) websites [5, 6], we observe that the video traffic varies in each hour of a day but repeats a similar pattern on each day. Figure 1 shows the averaged video traffic (in terms of the number of video tasks) in each hour of one day, which is extracted from the log traces of XuetangX [5], a Coursera-like online education platform, and VoD website of China Telecom [1], a Youtube-like video website. The traffic basically increases from 6 a.m. to 8 p.m. and thereafter decreases to the next morning, which conforms with the user habit of work and after-work hours. In the figure, the bottom and top of every vertical virgule represent the minimize and maximum value of the average task number of the time region and the rectangle in the middle represents 90% medium value of them. Since the video traffic changes over time, reserved instances can not be simply employed to bring down the cost.

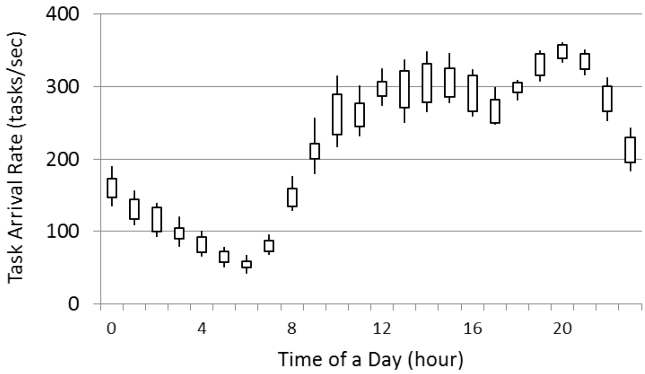

Fig. 1: Average video traffic in each hour of one day

To illustrate the renting cost waste of the improper configurations of VM instances, we calculate the cost of one day using each pricing policy offered by EC2. Table I lists the averaged one-day cost (as the last column) of each pricing policy, when serving the video traffic shown in Figure 1. To ensure QoS (Quality of Service), each VM instance is limited to serve at most 20 video tasks. Expense of *on-demand* instance configurations is the accumulated value of the dynamic each-hour fees. Expenses of *reserved* instance configurations are cut down by nearly half. The cost comes from two parts: the upfront fee and the unit price, which respectively have the averaged value shown in the middle two columns of Table I.

TABLE I: Average one-day cost using each pricing policy of EC2

Instance Type		Upfront($/d)	Unit Price($/h)	Cost($)
On-Demand		0	0.853	298.283
Reserved	Light-R	1.764	0.369	148.275
	Medium-R	4.090	0.227	148.307
	High-R	4.986	0.155	165.414
Spot		0	0.081	29.832
Hybrid-R		/	/	**125.637**

The cost benefit is because of the discount of the unit price, though the upfront fee has been pre-charged. Expense of *spot* instance configurations is significantly reduced, but resource is not steady and thus the service is not guaranteed. Sole spot instance configuration is not practical.

Hybrid configuration is the core idea of our work. It employs different types of instances to coordinate with the video fluctuation, to get the optimal renting strategy and lower rental fees. By utilizing higher-reserved instances to serve the basic streaming tasks in idle hours and complementing lighter-reserved instances to manage the massive requirements in peak hours, the former ones provide a lower unit price for the high utilization ratio, and the later ones have less upfront fee that can save money when there are no so much tasks to serve. In this case, we choose 4 high-reserved instances, 8 medium-reserved instances and 7 light-reserved instances to serve the tasks and get more than 20$ cost saving instantly, which could be enlarged to tons of millions dollars in commercial systems. As video traffic has a regular variation cycle, it is possible to give a long-term prediction and apply it into a dynamic on-line resource configuration scheme for the video streaming services in cloud.

Our contributions are summarized as follows:

1) We observe that hybrid configuration of instance types leads to lower renting cost of video streaming services in cloud platform, due to fluctuation of video traffic. By studying the correlation of video and user access information, we introduce a lightweight prediction algorithm.
2) We propose the *Hyrbid-R* hybrid configuration to optimize the instance usages among on-demand and high-, medium- and light-reserved instances, based on the predicted video traffic. We further explore the incorporation of spot instances to further lower the cost, and propose the *Hybrid-RS* hybrid configuration.
3) We evaluate our algorithms using datasets from real-word on-line education and entertainment video services. Results show that our algorithms can save renting cost by at least 20%.

II. Background

A. Pricing Policies in EC2

The total renting cost of the VoD system in cloud platform is mainly influenced by two pricing factors: Storage facilities and Instance types. Elastic Block Store (EBS) is one of the most frequently used storage facilities on the EC2 platform. EBS provides persistent block level storage volumes for cooperating with EC2 instances in the cloud. It can be mounted on to the instances and accessed as the block devices expediently with extra rental fees by time, space and I/O volume. To guarantee the user accessing of global video database, we assume using EBS as the storage service in our configuration scheme. There are three types of VM instances in EC2:

On-Demand Instances allow users to pay for compute capacity by the hour with no long-term commitments. It may take 2-5 minutes delay if the users apply the on-demand instances temporarily, but the configuration scheme can calculate the instances requirements and apply them in advance, so the on-demand instances are suitable for video streaming tasks.

Reserved Instances provide the users with a significant discount and capacity reservation compared to on-demand ones. There are three payment options, light, medium and high-reserved instances, they have the incrementally upfront fees and decreasingly hourly charging discount, the light and medium-reserved instances are only charged by using hours and the high-reserved ones are charged through the whole reserved term. The reserved instances are most suitable for the video streaming services and are mainly used in our configuration scheme.

Spot Instances enable the users to bid for unused EC2 capacity. Instances are charged by the spot price, if the users' maximum bid prices exceed the current spot price, their requests are fulfilled and the instances will run until either the users choose to terminate them or the spot price increases above their bid prices. The unit price of spot instances is much lower than other instances, but when the spot price rises higher than the bid price, the resource will be taken back quite soon. So we introduce task migration strategy in our scheme to make use of spot instances for further cost conserving.

B. VoD in Cloud

To better understand the cloud-based VoD infrastructure, we give a presumed architecture picture to illustrate the relationships between each part of the system. As shown in Figure 2, the main parts of the system are the *user* and *Web Server*, the *Task Scheduler*, *Instances* and *EBS*.

When a *user* launches a video playing requirement, a streaming task is generated by the *Web Server* and sent to the *Task Scheduler*, who will decide which *Instance* to distribute the task in according to the video contents and other features like the user IP address region and the server clusters bandwidth conditions. The *Instance* will read the required video program from the global video database storage *EBS* and transmit to the *user*. To serve the users from different regions, there are multiple replica of such systems across different server locations.

The core idea of our work is to give a optimal configuration scheme of the different types of *Instances*, according to the video traffic variations, with the lowest renting cost of the VoD system on the cloud platform. The first step of the work is utilizing the basic on-demand and reserved instances with the predicted video traffic to propose an initial instance renting plan, like C_1 in the figure; The second step is C_2, which introduces the low-cost spot instances into the configuration to provide an advanced cost-efficient scheme.

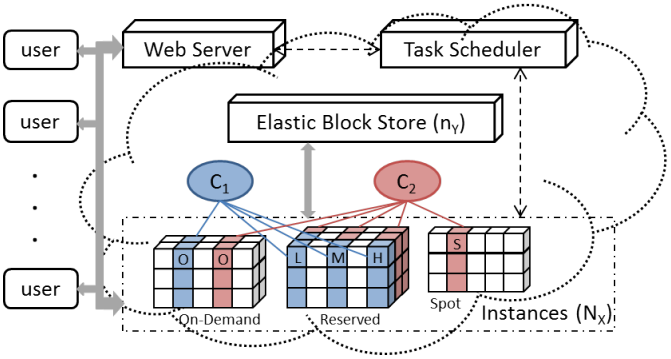

Fig. 2: VoD system architecture in cloud platform

C. Problem Formalization

The cost-effective resource configuration problem for cloud video streaming services can be defined as follows. Notations used in this paper are summarized in Table II.

Input:

- A series of streaming tasks whose hourly volume V = $\{v_k \mid k$ = 1, 2, 3, ...$\}$ is predicted for a long term (a year in our case) by their historical log traces;
- The upfront fees $\tilde{p}_L$, $\tilde{p}_M$ and $\tilde{p}_H$, hourly unit prices p_O p_L, p_M, p_H and p_S of the On-demand, Light, Medium and High-reserved and Spot instance of cloud platform;
- The maximum task number w that a virtual machine can process simultaneously with the Quality of Service (QoS, 95% streaming tasks can be served with no more than 5% delay of their playing time in our case) guarantee;
- The upfront and unit price $\tilde{p}_Y$ and p_Y of the EBS space;

Output:

- The total renting cost C of the streaming tasks V;
- The numbers of the applied On-demand, Light, Medium, High-reserved and Spot VM instances, n_O, n_L, n_M, n_H and n_S, that give the minimum C;

C can be calculated as follows. C_X and T_X are the rental fee and running time which is also the charging time of the X-type instances, C_Y is the cost on EBS. Instance types set I = $\{X \mid X = O, L, M, H, S\}$.

$$\begin{aligned} C &= C_Y + \sum_{X \in I} C_X \\ &= (p_Y * T_Y + \tilde{p}_Y) * n_Y + \sum_{X \in I} (p_X * T_X + \tilde{p}_X) * n_X \end{aligned} \tag{1}$$

The optimal configuration scheme is: $N = \{n_X \mid X \in I\}$. In the equation, T_X can be calculated by n_X and the video traffic volume V, and it is accurate to hours, the same as V, because the VM instances are charged by the time unit of hours. Every other week, we will collect the newly generated task information and bring back into the historical data to get a more accurate prediction for the next week.

TABLE II: Notations used in this paper

Notation	Description
v_k	Video traffic of the kth hour of a cycle.
V	$V = \{v_k \mid k$ = 1, 2, 3, ...$\}$.
w	VM Instance capacity of video streaming tasks.
$\tilde{p}_X$	Upfront fee of the X instance.
p_X	Unit price of the X instance.
$\tilde{p}_Y$	Upfront fee of the EBS space.
p_Y	Unit price of the EBS space.
n_X	Number of the applied X instance.
n_Y	Size of the applied EBS space.
N	$N = \{n_X \mid X$ = O, L, M, H, S, ...$\}$.
T_X	Running time of the X instance.
T_Y	Renting time of the EBS space.
C	Total renting cost of the video streaming tasks V.

III. Video traffic prediction

A. Definitions

1) *Video Traffic*: According to a series of real world VoD system log traces we have got from [5] and [1], there are some common informations the system will log about each video streaming task, like the begin time, end time, user ID, IP address and the accessed video program ID. From the begin and end time of the tasks, we can calculate the video traffic in a certain time region, so we define the *Video Traffic* from time *a* to time *b* (refer as *VT*[*a, b*]) as the number of all the tasks that end after *a* and begin before *b*.

2) *Minimum Instance Requirement*: The VM instances have a capacity upper bound to guarantee the QoS of the video streaming tasks [2], so for any period of time like *a* to *b*, there will be a least number of the VM instances to properly serve the corresponding *Video Traffic*, which is defined as *Minimum Instance Requirement* of time region *a* to *b* (*MIR*[*a, b*]). As the instance capacity is defined as *w* in the former section, we can give the relation between VT and MIR as follows:

$$MIR[a, b] = \lceil \frac{VT[a, b]}{w} \rceil$$

Due to the characteristic of periodical variation of the video streaming services, the Exponential Smoothing (ES) method is quite a suitable way for the video traffic prediction. It is a kind of time series analysis method developing from Whole Period Average method and Moving Average method [7], it is also one of the most frequently used prediction algorithms.

B. Prediction Strategy

The finest charging unit granularity of the reserved VM instances' upfront on EC2 is by year. So we consider giving a bunch of video traffic data in the quantity of a week and using the predicted data for another further week information, repeating the iteration to get the video traffic of a month till a year. The traffic is accurate to an hour, which is the EC2 charging unit.

Among the above three time series prediction algorithms, the Whole Period average method uses all the historical data of the time series with the equipotent precedence; the Moving Average method takes no account of long-dated data and gives the near-term data a higher weight; the Exponential Smoothing method combines both of their advantages, it does not abandon the old data but rather gives them a gradually subdued influencing potency, which means the weight will converge to 0 along with the distance increasing between the historical data to current moment on the time series.

As the video traffic variation appears to be a trigonometric function curve, we choose the Cubic Exponential Smoothing method as our prediction algorithm. The smooth curve of the video traffic variation can be fitted with a quadratic polynomial:

$$VT[k, k+t] = \sum_{i=k}^{k+t} (d_k + r_k * i + acc_k * i^2/2)$$

We can see that $VT[k, k+t]$ is the video traffic from time k to $k+t$, d_k is the video traffic variation trend of time k, r_k

is the rate of video traffic variation trend of time k and acc_k is the variation acceleration. These time-varying parameters can be estimated by the following cube ES formula:

$$d_k = \alpha * VT_k + (1-\alpha)(d_{k-1} + r_{k-1} + acc_{k-1}/2)$$
$$r_k = \beta * (d_k - d_{k-1}) + (1-\beta)(r_{k-1} + acc_{k-1})$$
$$acc_k = \gamma * (r_k - r_{k-1}) + (1-\gamma) * acc_{k-1}$$

α, β and γ are smoothing factors, with the value range of [0, 1], which determine the influencing potency between the near-term variation and the long-dated historical data [8]. We adjust the values of smoothing factors according to the real-time observed prediction errors based on the dynamic estimation method [9].

C. Prediction Validation

To evaluate the performance of the prediction algorithm, we divide the log trace data into two sets: one for off-line training and the other for on-line configuration simulation. We calculate the predicted *Video Traffic* of the tasks in the second sets by the rules extracted from the first one, then compare with their real value recorded in the original trace.

Accuracy: We choose the *Goodness of Fit* and the *Linear Correlation Coefficient* as the measuring standards of the prediction algorithm. The *Goodness of Fit* here is considered as the fitting degree between the predicted value and the real value. It is simply defined as follows:

$$Goodness\ of\ Fit = VT_p\ /\ VT_r$$

The VT_p and VT_r stand for the predicted and real *Video Traffic*. From the definition we can see that the more the value of Goodness of Fit is close to 1, the better the prediction algorithm is. So we calculate the Relative Standard Deviation of all the Goodness of Fit of each task and give their mean value 0.081. It means that the prediction algorithm can give an acceptable *Video Traffic* with the accuracy probability of 92%. As for the *Linear Correlation Coefficient* between the historical and the predicted data in our case is 0.907, which means the two datasets are positive correlation and have a strong correlation. The following experiments show that this result is befitting enough for the configuration algorithms.

Overhead: As the prediction informations are obtained by just a few query operations from the existing off-line training database, the overhead is almost negligible compared with the following configuration algorithms. Besides, most of the commercial VoD service websites will play advertisements for dozens of seconds before the video program, which provides sufficient time window for the prediction and even the configuration procedures.

The subsequent experiments of the resource configuration algorithms will show that our prediction algorithm is precise enough to provide the proper predicted video traffic to the configuration algorithms. But we will also explore more accurate techniques to satisfy more complex systems like serving multiple types of tasks in heterogeneous data center environments in the future.

IV. On-line Configuration Scheme

A. Hybrid-R: On-Demand and Reserved Instances

As the example in Table I shows, neither the on-demand instances nor the three types of the reserved instances solely can not achieve the rental fee as low as the Hybrid-reserved instances scheme, this is because the video traffic does not maintain in a stable level but varies by the cycle of 24 hours with wave peak and trough. If we all use the light-reserved instances, the basic video traffic lower than the wave trough that exists all the time will cost a higher unit price; On the contrary, if we all use the high-reserved instances, the rush-hour video traffic will be settled in a lower unit price but we also must suffer the high upfront during the idle periods. To balance between the reserve level and the video traffic variation, we introduce the *Instance Utilization Ratio* as the criterion of choosing which type of instances with a certain video traffic in their variation cycle.

1) Instance Utilization Ratio: We define the *Instance Utilization Ratio* (IUR) as the percentage of the actual running time (t_{run}) divided by the renting time (t_{rent}) of the instances:

$$Instance\ Utilization\ Ratio = t_{run}\ /\ t_{rent}$$

If a instance is high-reserved, it will be charged for both upfront fee and unit price during its whole renting time no matter it is running or not, while if the instance is light or medium-reserved, it will only be charged for upfront fee during its idle time.

Figure 3 shows the rising trend of the 3-year-term reserved rental fee of the basic type of storage optimized I2 instances with Linux operating systems during the whole renting period. We can see that in the over a thousand days, if the running time reaches to about 200 days, using the light-reserved instances will be cheaper than the on-demand instances, we call this threshold as IUR_{O-L}, which is 18.3% in this case; Similarly, if the running time is higher than about 520 days, the medium-reserved instances will be superior to the light-reserved ones, when the running time exceeds 910 days, the high-reserved instances become the most economical one, we call the two thresholds as IUR_{L-M} and IUR_{M-H}, which are 47.5% and 83.1%. The three thresholds will be the main criteria of allocating the proportion of the different types of the instances,

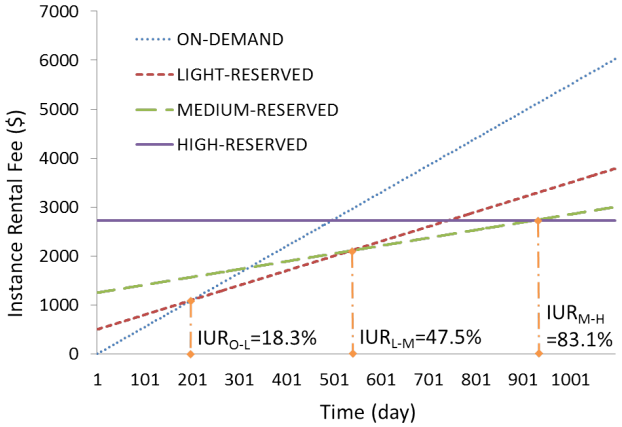

Fig. 3: The relationship between the three types of reserved instances and on-demand instance and the *Instance Utilization Ratio*

to form the resource configuration scheme. As all the video programs resource are stored on EBS and accessed by the instances through network, we do not need to consider the data locality problem in the following configuration algorithms.

2) Configuration Algorithm: With the assistance of IUR and the predicted video traffic, we propose the on-line resource configuration algorithm for the video streaming services in cloud platform, as shown in Algorithm 1. The inputs of the algorithm include the predicted video traffic V, the upfront fees and unit prices of the four types of instances, the instance capacity w and the configuring time cycle Z. And the outputs are the applied numbers of the four types of instances and the approximate lowest renting cost they present.

The algorithm is mainly divided into two steps:

Step 1: Calculate the *Instance Utilization Ratio (IUR)* of the four types of instances; To do so, we introduce a group of binary linear equations:

$$y_i = p_i * x_i + \tilde{p}_i \ (i = o, l, m, h)$$

Algorithm 1 Hybrid-R: Hybrid Resource Configuration Algorithm using On-Demand and Reserved Instances

Require:
The predicted hourly streaming video traffic of the next service cycle time Z: V = $\{v_k \mid k$ = 1, 2, 3, ...$\}$;
VM Instance capacity of video streaming services w;
The upfront fees and unit prices of the three types reserved instances: $\tilde{p}_L$ and p_L, $\tilde{p}_M$ and p_M, $\tilde{p}_H$ and p_H;
The unit price of the on-demand instance: p_O;

Ensure:
The number of the applied four types of instances that give the lowest cost: N = $\{n_i \mid i$ = o, l, m, h$\}$;
The total renting cost C of the video streaming tasks V;

/*Step 1. Calculate the *IURs*/
for instance type i:
    y_i = p_i * x_i + $\tilde{p}_i$ (y_i: renting cost, x_i: running time)
for $x_i \geq 0$ && $x_i \leq Z$:
    if y_o == y_l:
        then IUR_{O-L} = x_i / Z;
    else if y_l == y_m:
        then IUR_{L-M} = x_i / Z;
    else if y_m == y_h:
        then IUR_{M-H} = x_i / Z;
/*Step 2. Calculate the instances renting proportion*/
for $v_k \in V$: v_{max} is the maximum value of v_k;
for $n_i \geq 0$ && $n_i \leq \lceil v_{max}/w \rceil$:
    if IUR_{M-H} == $\sum_k n_i$ / v_k:
        then n_h = n_i;
    else if IUR_{L-M} == $\sum_k n_i$ / v_k:
        then n_m = n_i;
    else if IUR_{O-L} == $\sum_k n_i$ / v_k:
        then n_l = n_i;
    else n_o = $\lceil v_{max}/w \rceil$ - n_h - n_m - n_l;

$$C = (p_Y * Z + \tilde{p}_Y) * n_Y + \sum_{X=O,L,M,H} (p_X * Z + \tilde{p}_X) * n_X$$

return N = $\{n_i \mid i$ = o, l, m, h$\}$, C.

In the equations, the value range of instance type i is consist of on-demand (o), light-reserved (l), medium-reserved (m) and high-reserved (h), y_i stands for the total renting cost of type i instances, x_i stands for the service time of the type i instances. If the instance is high-reserved, x_i would be the whole renting time, otherwise, x_i would only be its actual running time. p_i and $\tilde{p}_i$ are the unit price and upfront fee of the instance i. To calculate the IUR_{O-L}, IUR_{L-M}, IUR_{M-H}, we let y_o and y_l, y_l and y_m, y_m and y_h be respectively equivalent, just like the three intersections in Figure 3, and the IUR will be the corresponding x_i divided by the scheduling duration time Z.

Step 2: Calculate the applied amount of each type of instances and the total rental fee for the most cost-effective scheme; We first scan V for the maximum value v_{max} of hourly video traffic v_k, to get the maximum hourly instance demand quantity $\lceil v_{max}/w \rceil$, in which w is the instance capacity of the video streaming tasks. Then according to the $IURs$ and the predicted video traffic V, the applied amount of high-reserved instances should be the number that achieves the utilization ratio of IUR_{M-H} among the whole video traffic, in the same way, the amount of medium and light-reserved instances should be the numbers that achieve the utilization ratio of IUR_{L-M} and IUR_{O-L}, and the number of on-demand instances should be the rest of all the required resources. As the on-demand and reserved instances all use the EBS spaces, n_Y is the total size of the global video database. The total cost of the scheme can be calculated by Equation 1.

The service cycle time Z in our algorithm is set to one year, same as the shortest reservation duration in EC2. But the re-execution period can not be extended to such a long time. As the video traffic of the VoD systems variates continually and most of them have a rising tendency in the aspects of both user scale and accessing amount. So we provide a dynamic modulation strategy that in every other week, adding the newest video traffic information into the input of the Video Traffic Prediction algorithm to update the V of Algorithm 1, and regulating the applied number of the VM instances.

B. Hybrid-RS: Hybrid-R with Spot Instances

As the on-demand instance has a much higher price than the reserved instance while the spot instance has a much lower one, we hereby utilize the task migration mechanism to dynamically schedule the streaming services to further reduce the total renting cost of the VoD system.

1) Spot Instance Specialties: Spot Instances are spare EC2 instances for which the users can name their own price. The spot price is set by EC2, which fluctuates in real-time according to spot instances supply and demand. When user's bid exceeds the spot price, the spot instance is launched and will run until the spot price is higher than user's bid or the user choose to terminate them. There are three main specialties of the spot instances:

- Spot instances perform exactly like other EC2 instances but have possibility that might be interrupted.
- The users only need to pay no more than their maximum bid price per hour.
- If the spot instance is interrupted by EC2, the user will not be charged for the interrupted hour.

We can see that the features are all beneficial to apply video streaming services on spot instances, except for the sudden interruption may become the main restriction. So we have to select the most applicative tasks for the trial. EC2 now provides a new Spot Instance Termination Notices [10] policy that reminds the users of resource reclaiming two minutes beforehand, so that the users can save their status, upload final log files, or move the tasks to other instances in this time. This new policy allows more types of applications to benefit from the scale and low price of spot instances, and also helps us to introduce the spot instances to the configuration scheme.

2) Applicative Tasks: There are four types of tasks that work well with Spot Instances.

Optional tasks: The users can run their optional tasks when spot prices are low and stop them when the prices rise too high.

Delayable tasks: These tasks have deadlines that allow the users to be flexible about when to run their computations.

Acceleratable tasks: The users can run spot instances to accelerate their computing when the spot price is low while maintaining a baseline layer by other instances.

Large scale tasks: These tasks require computing scale that the users can not access any other way. With spot, they can cost-effectively run thousands or more instances.

Compared with the above categories, video streaming tasks have the highest possibility that they can be delayable as the web player usually buffer minutes of videos in the client side. However, not all the streaming tasks can endure the sudden break in the middle of playing, we hereby set the following rules that restrict the types of video streaming tasks that can be applied onto spot instances, and how to reasonably utilize the spot instances to balance the cost control and user demand:

a. Short tasks are more suitable for trial operation on spot instances. As mentioned before, the spot instance can be retrieved by the supplier at any time, so the longer the streaming tasks are, the riskier they might be interrupted. As for the criterion of short tasks, we hereby formulate it as from no more than 5 minutes video programs in our algorithms.

b. There are also some higher-priority critical tasks that are not appropriate for spot instances. For example, the live show of important events, the paid video program from membership users, which once be suspended, will all bring more severe negative effects to user experience.

c. If a qualified task failed to run on the spot instance due to resource absence, instead of waiting or trying again for the new available spot instances, we would migrate the task to guaranteed services like on-demand instances, to prevent another breaking down during the users' watching period.

3) Task Migration: The current state of art techniques can manage the task migration procedure [11–14], and keep the overhead under a reasonable level. Besides, some commercial VoD system service providers like BokeCC [15] are also applying the heartbeat mechanism to collect user behavior informations and inserting checkpoints of task monitoring. By the above works and our real log traces, we conclude that the overhead of a streaming task migration can be controlled in dozens of seconds, it is the time requirement of relocating procedure of the interrupted tasks. As the streaming tasks always have a buffered video segment in the length of a couple minutes on the client side, and the minimum charging unit of EC2 is hour, the migration time consuming is basically unaware for the users and negligible for the cloud platform.

4) On-line Configuration Algorithm: Based on Algorithm 1 and the historical data of the real-world log trace, we can calculate the proportion of the applicative tasks for spot instances, extract the non-spot video traffic to get N_{lmh} and C_{lmh} of the basic configuration of the reserved instances, then deploy Algorithm 2 to split the total video traffic V into V_o, V_{lmh} and V_s, use the real time task information to dynamically schedule them on spot and on-demand instances. The algorithm is mainly carried out by two parallel parts:

Part 1: Try to assign a spot instance for an applicative task, if the resource is retrieved during the serving period, count back the task to that using on-demand instances and record it in ir_k that composes the set of interrupted task numbers IR. The cost of spot instances C_s and the Task Interruption Rate (TIR) can be calculate by V_s, P_s and IR;

Part 2: Schedule the tasks from on-demand instances to the reserved ones if there are spare volume, the priority is high-reserved higher than medium-reserved higher than light-reserved, as their upfront fees have been paid but the unit prices are incremental. Unsubscribe an on-demand instance ahead of time if there are no more tasks on it for cost conservation, calculate the updated cost of the on-demand instances C_o;

The total cost of the using-spot instance scheme $C_{Hybrid-RS}$ is the summation of C_{lmh}, C_s and C_o. After the end of the algorithm, we can give the Cost Saving Proportion (CSP) of the new scheme compared with the non-spot scheme $C_{Hybrid-R}$:

$$CSP = (C_{Hybrid-R} - C_{Hybrid-RS})/C_{Hybrid-R}$$

Algorithm 2 Hybrid-RS: On-line Resource Scheduling Algorithm for video streaming services in cloud

Require:
Real-time video streaming tasks: $T = \{t_k \mid k = 1, 2, ...\}$;
Video programs length $L = \{l_j \mid j = 1, 2, ...\}$;
VM Instance capacity of video streaming services w;
The unit price of the on-demand instance: p_o;
The bid price of spot instances: p_b;
The real-time hourly spot price $P_s = \{p_{si} \mid i = 1, 2, ...\}$;
Ensure:
The cost of spot and on-demand instances: C_s, C_o;
The Task Interruption Rate: TIR;
for t_k in hour i, whose video length is l_j, $v_{si} \in V_s$:
  if $l_j \leq$ 300 sec, :
    then $t_k \in T_s$, ++ v_{si};
/*Part 1. Spot Instances: C_s*/
for $v_{si} \in V_s$, $v_{oi} \in V_o$, $IR = \{ir_i \mid i = 1, 2, ...\}$:
  if $p_B < p_{Si}$:
    then - - v_{si}, ++ v_{oi}, ++ ir_i;
$C_s = \sum_i \lceil v_{si}/w \rceil * p_{si}$, $TIR = \sum_i ir_i \ / \sum_i v_{si}$;
/*Part 2. On-demand Instances: C_o*/
for $v_{lmhi} \in V_{lmh}$, $v_{oi} \in V_o$, $n_{lmh} = n_l + n_m + n_h (\in N_{lmh})$:
  if $v_{lmhi} < w * n_{lmh}$:
    then ++ v_{lmhi}, - - v_{oi};
$C_o = \sum_i \lceil v_{oi}/w \rceil * p_o$;
return C_s, C_o, TIR.

V. Experimental Results

A. Evaluation Datasets and Environments

Datasets: In the evaluation we use two datasets: real-world log traces from an on-line education platform and a multimedia entertainment website. The first one is from a popular on-line education platform XuetangX [5], and the second one is generated with characteristics from mainstream video services [6] based on traces from China Telecom VoD system [1]. Details of the two datasets are as follows:

1) On-line Education (EDU) dataset: The on-line education dataset consists of a log trace covering 121 days from XuetangX [5]. We choose a part of the log trace as the off-line training set and the rest part as the experimental test case. Different from the commercial entertaining VoD system, the data from XuetangX has some specialties like the single category of video type and a more specific user group.

2) Multimedia Entertainment (ENT) dataset: The multimedia entertainment dataset is generated based on a log trace covering 7 months in a VoD system with about 150,000 users deployed by China Telecom. This log trace is authentic and practical, however, its scale is not representative enough. It can be counted that there are about twenty millions tasks totally in half year, that is about one task per second in average. While in a mainstream VoD service website, there are always hundreds of tasks per second accessing even one popular video.

To overcome the disadvantage of this dataset, we reference both the summarized characteristics of real VoD system user behavior [6] and the statistical rules published in [1] to obtain a large-scaled series of streaming tasks as our experimental dataset. This log trace has the same quantitative variation and length distribution as practical situation so it can better reflect the algorithms' functionality in real world.

Evaluation Environment: With the supporting of the real-world datasets, we launch a series of simulated experiments based on the EC2 platform. As the video streaming services are I/O-intensive kind of tasks, we choose the 8-times-large scale of storage optimized I2 instances with Linux operating system of on-demand and reserved instance candidates, and the 8-times-large scale of storage optimized HS1 instances with Linux operating system of spot instance candidates [4]. As all the instances are using the EBS spaces, it needs the total size of the global video database, which are charged according to its pricing standard [16]. We use the one-year reserved instances charging standards and the spot instance pricing history on EC2 console by the period of one year. The bid price is set to the average value of the medium reserved instance price. The video traffic capacities of the two types of instances are more or less the same, which are verified to be serving 350 video streaming tasks synchronously.

B. Algorithm Performance

We first evaluate the performance in terms of renting cost of our proposed configuration algorithms, and compare them to each type of instances separately. The algorithms are respectively evaluated using both of the two datasets.

Figure 4 and Figure 5 show the normalized renting cost of a series of configuration schemes: On-Demand, Light-Reserved, Medium-Reserved, High-Reserved, Hybrid the above four types without or with Spot instances (Hybrid-R, Hybrid-RS), from the on-line education dataset and the multimedia entertainment dataset. The single spot instances configuration scheme are not included because they can not afford serving the whole dataset log trace due to their instability. To evaluate the precision of the video traffic prediction algorithm, we import both the real and predicted video traffic data informations to the configuration schemes, corresponding to the off-line and on-line version of the algorithms. For intuitively comparison, we normalize the result of off-line configuration scheme of On-Demand instance to 1, and other results to the relevant times of it. The experimental results characteristics of the two datasets are summarized as follows:

i The video traffic prediction is effective, as the differences between on-line and off-line algorithms are small. From the calculating results we can see that the accuracies of the prediction are all higher than 90%, which is quite acceptable under normal conditions. The prediction differences of the multimedia entertainment dataset are even more smaller than the on-line education dataset, it is because the video traffic fluctuation of the former one is more substantial and regular due to the entertaining time of the users usually concentrate to off-work time while the studying time can be arranged more flexibly. Besides, the larger video traffic of the multimedia entertainment dataset also provides greater advantage for the prediction algorithm.

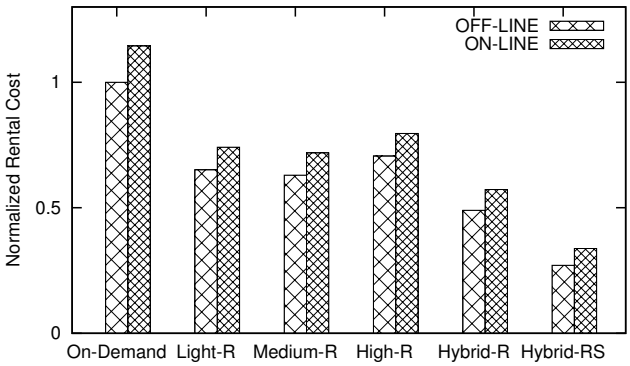

Fig. 4: Normalized Rental Cost of the resource configuration schemes and their off-line version (On-line Education Dataset)

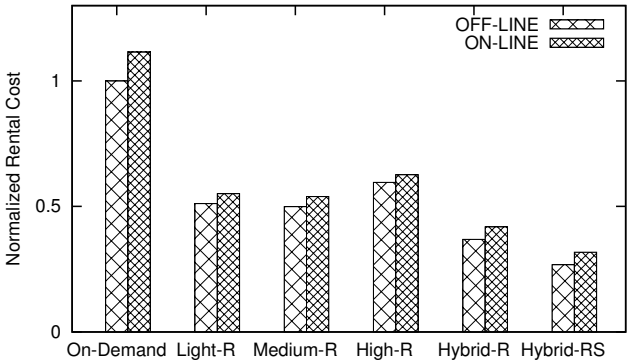

Fig. 5: Normalized Rental Cost of the resource configuration schemes and their off-line version (Multimedia Entertainment Dataset)

ii The hybrid resource configuration scheme of the four basic instances types can effectively save the renting cost compared with using each instances separately. The figures show that the Hybrid-R scheme has more than 60% cost saving compared with the off-line On-Demand instance scheme with the multimedia entertainment dataset, and also has about 20% superiority to other configuration schemes. The results of the on-line education dataset is a bit inferior because of the less video traffic and its narrow fluctuation range.

iii The import of the spot instances can significantly improve the results of the Hybrid-R configuration scheme, which is more obvious in the on-line education dataset. As the criterion of applicative tasks that applied on spot instances is more suitable for the high-proportion short video programs of the on-line education website. The trial of utilizing spot instances in resource configuration scheme can contribute the cost saving up to nearly 70% in both datasets. But we still need to consider the negative effects of using spot instances, the interruption during video program playing, which is the main problem in user experience. We will discuss the trading off between them in the subsequent sections.

C. Algorithm Complexity

We also evaluate the algorithm complexity in terms of algorithm execution time, in order to evaluate its impact on the latency of on-line configuration. We use both theoretic and measured execution time in this evaluation. The two datasets are analyzed and evaluated separately and compared together.

1) Theoretic Time Complexity: The first algorithm has nearly the constant level complexity, because it uses the predicted video traffic information as input, the primary calculations are the $IURs$ and the instance number of each type, which are hardly affected by task number scale increase. The only scale-relevant part is the total renting cost calculation, but it just has a slight impact on the global time complexity.

In the second algorithm, besides the resource configuration part, all the tasks need to be checked once for estimating adaptability on spot instance, so it has the $O(nlogn)$ time complexity. It means that the time cost will grow linearly by the increase of video traffic, which is acceptable for the scalability of the problem. All the theoretic analysis can be verified in the following statistics results.

2) Measured Time Complexity: We use both the On-line Education (EDU) dataset and the Multimedia Entertainment (ENT) dataset as basis, expand the task number per second till 10 times and record the total resource configuring time and task scheduling time to measure the relationship between time cost and the problem scale. Figure 6 gives the charts of which X-axis is the magnification of the dataset task number and the Y-axis is the calculation time of the algorithms. We have three observations:

i The first algorithm (Hybrid-R) has a nearly constant time complexity in both datasets, the calculation time costs are all in the level of dozens of milliseconds and barely increase by the expanding of the video traffic;

ii The second algorithm (Hybrid-RS) has a linear growth trend in both datasets, the calculation time costs increase within a dozen times along with the video traffic expansion, which conforms to the former theoretic analysis about the $O(nlogn)$ time complexity;

iii The calculation time of the EDU dataset is shorter than the ENT dataset because of smaller video traffic. While the increment rate of the EDU dataset in Hybrid-RS algorithm is higher than the ENT dataset due to the larger short task proportion.

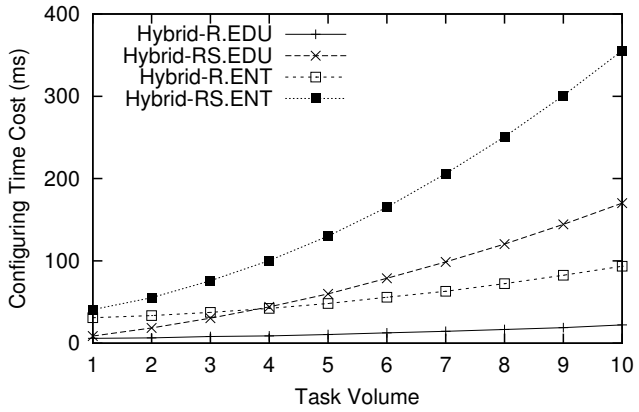

Fig. 6: Comparison of configuration algorithms on time complexity, X-axis is the magnification of the video traffic in On-line Education (EDU) dataset and Multimedia Entertainment (ENT) dataset

From the time complexity evaluation results we can see that both of the two algorithms are working efficiently on our datasets, the dozens of milliseconds time cost bring negligible overhead to the whole VoD system. The configuration schemes have a satisfactory scalability by the video traffic expansion, they are promising choices for cost conservation strategies of video streaming services in cloud.

D. Spot Instance Performance Price Ratio

According to the algorithm performance and complexity evaluation, we can see that introducing the spot instance into the configuration scheme brings both renting cost conservation and scheduling time overhead. It is a trading-off problem to balance the advantage and disadvantage. In this section, we will quantify the relationship between the key influence factors and the performance price ratio of the configuration scheme, to give a direct perspective of the pros and cons of the spot instance.

To simplify the comparison of the algorithm efficiency, we hereby define the Performance Price Ratio (PPR) of the resource configuration scheme as the Cost Saving Proportion (CSP) divided by the Task Interruption Rate (TIR) in Algorithm 2, which is represented as follows:

$$PPR = CSP/TIR$$

The most critical parameter of the resource configuration scheme is the task number proportion that try on using the spot instance, which is decided by the short task estimating threshold. If the threshold is too short, there will be few tasks can use the spot instances, then the cost conservation is non-significant; If the threshold is too long, there will be too much tasks on the spot instances, then the overhead of rescheduling

the interrupted tasks will also impact the system performance, even bring negative effects to the user experience.

Figure 7 shows the normalized performance price ratio of Algorithm 2, by the variation of the short task threshold within 20 minutes, the PPR curves of the On-line Education (EDU) dataset and the Multimedia Entertainment (ENT) dataset manifest the same variation trend but different peak point. They are summarized as follows:

i The PPR curves both increase at the beginning and decrease when the threshold is broad enough. It is because the broader the threshold is, the more tasks will be assigned to try running on spot instances, the higher the Task Interruption Rate will be. If a task fails to use the spot instance, it has to be rescheduled to on-demand instance, which is much more expensive, so the Cost saving Proportion is relatively reduced. Thus it can be seen that the short task threshold can not be set too large in our spot instance trial experiment.

ii The PPR peak point of the EDU dataset appears at 5 minutes, while that of the ENT dataset is on 10 minutes, a bit higher than the former one. The reason is that the tasks of the EDU dataset are generally shorter than that of the ENT dataset, as the on-line educational video programs are usually a few to a dozen minutes, yet the multimedia entertainment website often has plenty of TV series and movies that reach a length of tens of minutes even up to hundreds of minutes. To get a similar short task proportion that try running on spot instance, the ENT dataset will reach a higher threshold than the EDU dataset.

From the experiment result we can see that to get an optimal performance price ratio on utilizing the spot instances in the cost-effective resource configuration scheme, we should set a relatively conservative short task threshold and properly modulate it to a appropriate value according to the dataset characteristics. As Algorithm 2 dynamically schedules applicative tasks onto spot instances along with the VoD system operation, we can shift the threshold in a small window like 5 to 10 minutes and monitor the variation tendency of PPR, to determine an optimal short task criterion.

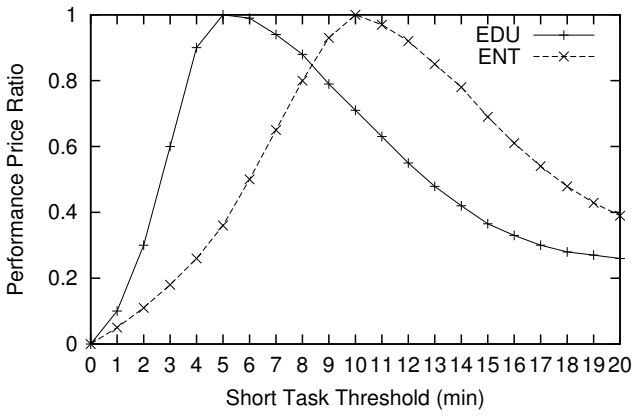

Fig. 7: The Hybrid-RS algorithm Normalized Performance Price Ratio variation with the short task threshold increasing of the On-line Education (EDU) dataset and Multimedia Entertainment (ENT) dataset

VI. Related work

VoD Video Traffic Prediction: H. Yu et al. contributes one of the earliest comprehensive user behavior analysis in large-scale VoD system [1]. It gives a statistical introduction of streaming tasks characteristic, content access patterns and their implications based on empirical data. Many works have been concentrated on video traffic prediction involving many kinds of areas. They can be as simple as the pure exponential smoothing predicting methods [7, 17], or more sophisticated forecasting approaches [18, 19]. Though the strategy we used in this paper is the simple exponential smoothing algorithm, it can properly provide the video traffic informations which is accurate enough to assist the configuration schemes, and also, with a negligible overhead. So it is unnecessary to employ a sophisticated prediction algorithm at present, but it also will be a valuable exploration direction.

Instance Configuring: Many researchers have presented works about cost minimization on public cloud platforms, like the probabilistic model [20], the analytical performance price model [21] and spot instances trail model [22–24]. Paragon [25] is an on-line interference-aware scheduler in heterogeneous data center. It predicts the characteristics of the incoming workload by identifying similarities to previous applications and greedily schedules them in an interference-minimized and server utilization-maximized way. Our proposed configuration scheme is a integrated complement of the above works in the VoD system clusters.

Cost Conservation: Many research works have been conducted for cost conservation from different levels, ranging from architecture level to data center level [26–30]. In VoD clusters, Y. Chai has proposed a enegery-conserving data migaration for streaming storage systems [31]. This work focuses on the energy consumption in storage systems and optimizes data migration algorithms. In comparison, our scheme focuses on the running time consumption of virtual instances and optimizes the rented instance types configuration.

VII. Conclusion and future work

Cost conservation is becoming an important design issue in video clusters with increased popularity of video services in cloud platform. Existing users ignore the phenomena that applying single type of instance could lead to sub-optimal rental fee cost. In this paper, we propose a lightweight video traffic prediction algorithm and two heuristics cost-conserving on-line configuration algorithms based on that. The algorithms are evaluated using both log traces from on-line education systems and multimedia entertainment platform that follow the empirical characteristics from mainstream video service websites. Results show that our algorithms save significant cost with negligible overhead.

There are three aspects we would like to improve in the future. The first is to improve the accuracy of video traffic prediction with more sophisticate techniques. The second is to consider more task types in the model, like another common trans-coding tasks in VoD systems, which are often used in video program format and code rate conversion. The third is to introduce our algorithms to a more complex model with heterogeneous servers with task interference and more complicated user behaviors.

ACKNOWLEDGMENT

This work is supported by National High-tech R&D Program (863 Program, Grant No. 2012AA010903), and National Natural Science Foundation of China (Grant No. 61232008).

REFERENCES

[1] H. Yu, D. Zheng, B. Zhao *et al.*, "Understanding user behavior in large-scale video-on-demand systems," in *ACM SIGOPS Operating Systems Review*. ACM, 2006.

[2] S. Feng, H. Zhang, and W. Chen, "Shall I use heterogeneous data centers? a case study on video on demand systems," in *Proceedings of the 15th IEEE International Conference on High Performance Computing and Communications (HPCC)*. IEEE, 2013.

[3] AWS Case Study: Netflix. [Online]. Available: http://aws.amazon.com/solutions/case-studies/netflix/

[4] The Amazon Elastic Compute cloud pricing standards of VM instances. [Online]. Available: http://aws.amazon.com/ec2/pricing/

[5] The massive open online course platform in china. [Online]. Available: http://www.xuetangx.com/

[6] The index center of Sohu VoD system. [Online]. Available: http://index.tv.sohu.com

[7] S. Niu, J. Zhai, X. Ma *et al.*, "Cost-effective cloud hpc resource provisioning by building semi-elastic virtual clusters," in *Proceedings of International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*. ACM, 2013.

[8] E. S. Gardner, "Exponential smoothing: The state of the art," *Journal of Forecasting*, vol. 4, no. 1, pp. 1–28, 1985.

[9] D. W. Trigg and A. G. Leach, "Exponential smoothing with an adaptive response rate," *OR, Operational Research Society*, vol. 18, no. 1, pp. 53–59, 1967.

[10] The new policy about spot instance termination notice of EC2. [Online]. Available: https://aws.amazon.com/blogs/aws/new-ec2-spot-instance-termination-notices/

[11] H. Li, L. Zhong, J. Liu *et al.*, "Cost-effective partial migration of vod services to content clouds," *IEEE International Conference on Cloud Computing*, pp. 203–210, 2011.

[12] A. Khajeh-Hosseini, D. Greenwood, and I. Sommerville, "Cloud migration: A case study of migrating an enterprise it system to iaas," in *IEEE International Conference on Cloud Computing*. IEEE, 2010, pp. 450–457.

[13] R. Santhosh and T. Ravichandran, "Pre-emptive scheduling of on-line real time services with task migration for cloud computing," in *International Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME)*. IEEE, 2013, pp. 271–276.

[14] S. Hosseinimotlagh, F. Khunjush, and S. Hosseinimotlagh, "Migration-less energy-aware task scheduling policies in cloud environments," in *International Conference on Advanced Information Networking and Applications Workshops (WAINA)*. IEEE, 2014, pp. 391–397.

[15] The professional VoD service provider. [Online]. Available: http://www.bokecc.com

[16] The Amazon EC2 pricing standards of EBS volumes. [Online]. Available: https://aws.amazon.com/ebs/pricing/

[17] B. Billah, M. L. King, R. D. Snyder *et al.*, "Exponential smoothing model selection for forecasting," *International Journal of Forecasting*, 2006.

[18] S. Makridakis, A. Andersen, R. Carbone *et al.*, "The accuracy of extrapolation (time series) methods: Results of a forecasting competition," *International Journal of Forecasting*, vol. 1, no. 2, pp. 111–153, 1982.

[19] S. Makridakis and M. Hibon, "The m3-competition: results, conclusions and implications," *International Journal of Forecasting*, vol. 16, no. 4, pp. 451 – 476, 2000.

[20] A. Andrzejak, D. Kondo, and S. Yi, "Decision model for cloud computing under sla constraints," in *IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2010, pp. 257 – 266.

[21] H. Zhao, M. Pan, X. Liu *et al.*, "Optimal resource rental planning for elastic applications in cloud market," in *Proceedings of IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2012, pp. 808–819.

[22] B. Javadi, R. Thulasiramy, and R. Buyya, "Statistical modeling of spot instance prices in public cloud environments," in *IEEE International Conference on Utility and Cloud Computing (UCC)*, Dec 2011, pp. 219–228.

[23] S. Yi, D. Kondo, and A. Andrzejak, "Reducing costs of spot instances via checkpointing in the amazon elastic compute cloud," in *IEEE International Conference on Cloud Computing*, July 2010, pp. 236–243.

[24] A. Marathe, R. Harris, D. Lowenthal *et al.*, "Exploiting redundancy for cost-effective, time-constrained execution of HPC applications on Amazon EC2," in *Proceedings of the 23rd International Symposium on High-performance Parallel and Distributed Computing*, ser. HPDC '14. ACM, 2014, pp. 279–290.

[25] C. Delimitrou and C. Kozyrakis, "Paragon: Qos-aware scheduling for heterogeneous datacenters," in *Proceedings of the 18th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM, 2013.

[26] Í. Goiri, W. Katsak, K. Le *et al.*, "Parasol and GreenSwitch: managing datacenters powered by renewable energy," in *Proceedings of the 18th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM, 2013.

[27] K. Shen, A. Shriraman, S. Dwarkadas *et al.*, "Power containers: an os facility for fine-grained power and energy management on multicore servers," in *Proceedings of the 18th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. ACM, 2013.

[28] S. Govindan, D. Wang, A. Sivasubramaniam *et al.*, "Leveraging stored energy for handling power emergencies in aggressively provisioned datacenters," in *ACM SIGARCH Computer Architecture News*. ACM, 2012.

[29] S. Liu, K. Pattabiraman, T. Moscibroda *et al.*, "Flikker: saving dram refresh-power through critical data partitioning," *ACM SIGPLAN Notices*, 2012.

[30] F. Ahmad and T. Vijaykumar, "Joint optimization of idle and cooling power in data centers while maintaining response time," *ACM SIGPLAN Notices*, 2010.

[31] Y. Chai, Z. Du, D. A. Bader *et al.*, "Efficient data migration to conserve energy in streaming media storage systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 11, pp. 2081–2093, 2012.