Online Supplement for "Service Agreement Trifecta: Backup Resources, Price and Penalty in the Availability-Aware Cloud"

Shuai Yuan, Sanjukta Das, R. Ramesh

Department of Management Science & Systems, School of Management, University at Buffalo, The State University of New York, Buffalo, New York 14260, {shuaiyua, sdsmith4, rramesh}@buffalo.edu

Chunming Qiao

Department of Computer Science and Engineering, School of Engineering and Applied Sciences, University at Buffalo, The State University of New York, Buffalo, New York 14260, qiao@computer.org

The layout of this online supplement is as follows: Section I provides the details of the proposed availability-aware SLA framework, to highlight our research contributions. Section II derives the closed form expressions for three general cases that cover all possibilities of the lower limit location of the expected penalizable downtime discussed in Section 4.1 of the paper. Section III consists of proofs of the lemmas which could not be included in the paper due to page restrictions. Section IV provides a divide-and-conquer algorithm as our assumption might not hold in some cases in the determination of optimal number of backups. Section V highlights the impact of accuracy level of piecewise linear approximation approach. First we experimentally show the changes of the number of segments derived from the approximation approach as the level of accuracy varies. Then we demonstrate the optimal number of backup VMs is more sensitive to the approximation accuracy as the downtime distribution has high variance. Visualized comparisons between original downtime distribution and piecewise linear approximations under different accuracy levels are also provided. Section VI extends the analysis in Section 5.1 by evaluating how penalty rate and provisioning costs affects VM pricing; Section VII supplements the benchmarking comparisons in Section 6.5 by varying the ratio between provisioning cost and penalty rate. Finally, Section VIII computationally evaluates the penalty deferral strategy in comparison with a status quo strategy where the buyer does not have the option of deferring the penalty.

I. Proposed SLA Framework

Figure 1 of the main paper summarizes the proposed availability-aware SLA framework with five modules. The proposed framework is distinctively different from the traditional contract design and negotiation frameworks discussed in the literature. We develop the details of the proposed modules in the following discussion.

1. *Transient Downtime Distribution Derivation*. This module analyzes both resource and availability requirements of VMs and in addition, models the failure and repair process. (Du, Das et al. 2015) developed an algorithm to empirically derive the transient downtime probability density function (pdf) of (n, k)based on the limiting behavior of the underlying birth-death process of VM failures and repairs, using a sample path randomization approach. Note that although steady-state probabilities for the birth-death process can be obtained relatively easily, we may not be able to establish in practice that the system has reached steady state within the contract duration. Consequently, the transient pdf that represents exact downtime behaviors under all conditions is needed. Using this transient downtime pdf as inputs, we model the expected penalty cost for a given client in Section 4 and Section 5 of the paper.

2. *VM Availability Prediction*. This module contains analytic and numerical models for the estimation of VM availability under transient conditions. We perform a piecewise linear approximation using a two-stage top-down segmentation approach (TSTD) following (Wang and Chaovalitwongse Forthcoming) to obtain a functional form of the pdf. This approximation allows us to model any general distribution at a level of accuracy specified by the datacenter provider. Further, it enables the provider to control the accuracy with which the downtime distribution is fitted. We demonstrate in Section 6.1 of the paper why none of the well-behaved distributions can be relied upon to accurately represent the transient downtime distribution in our server log dataset.

3. *Availability-aware Resource Provisioning*. In this module, we design algorithms for availability-aware resource provisioning, and in particular, the allocation of VM backups. While the provider can reduce the

likelihood of SLA violation by providing more backup VMs, this reduction trades-off against a higher provisioning cost, with the client paying only for the VMs demanded. It is therefore necessary for the provider to determine the optimal number of backup VMs such that his expected total cost is minimized. The expected total cost is the summation of the provisioning cost and the expected penalty cost. The risk of under-provisioning is modeled via the concept of expected penalizable downtime, which is the amount that exceeds allowable downtime under the SLA-specified availability guarantee. Using the piecewise linear segment approximation of the downtime pdf from the module of VM Availability Prediction, we derive distinct closed form solutions of the expected penalizable downtime. We prove that the expected penalizable downtime is decreasing and convex in k.

4. *Availability-aware Schedule Design*. This module develops the price-penalty schedule that would be offered to a client as the basis for further contract negotiations. In practice, this would include mechanisms for cloud services when heterogeneous clients are considered, especially for those clients with customized requirements (with respect to service availability and other QoS attributes) that cannot be satisfied by the predefined SLA framework. The price-penalty schedule presents the break-even prices at varying penalty levels such that the provider would be indifferent between the choices. It serves as an important tool for the provider to both determine the lower bound of the price that can be set for a given penalty and service level, and explore the preference structure of the client if a negotiation ensues after quoting a price above the minimum level. An error in the accurate determination of the lower bound of the price may result in: failed negotiations; supporting an unprofitable contract to make sure that the SLA is met because risks were miscalculated or ignored when prices were negotiated and because the provider fears loss of business; a decision to willfully violate the unprofitable contract thus incurring penalties and possibly damaged client relationships. One of our goals is to help the provider determine his basis for price negotiation with the client. This lower bound on the price reduces the likelihood of the provider running at a loss at the time of contract execution as it specifically accounts for the various risks affiliated with the contractual liabilities. The focus of our research is in line with that of (Kauffman and Sougstad 2008) who model a variety of

scenarios where profit-maximizing pricing decisions may not be optimal when considered in relation to the contract risks. Our study contributes to their significant work on IT services contract negotiations and likewise, can help inform post-contract interactions. Further, the choice of the penalty level itself is likely to be dictated by the mission-criticality of the end use of the VMs. Driven by the extent of direct impact that the cloud services have on the client's business, the client is likely to gravitate towards a commensurate level of penalty for service unavailability. This signal from the client may be used in future negotiation steps to generate more suitable offers.

5. *Penalty Deferral Strategy*. This module focuses on specific instances where the contract fails to execute as expected, i.e., the uptime guarantee is not satisfied within a given service window. In most pay-as-you-go cloud platforms such as AWS, Microsoft Azure, and VMware, penalty payments take the form of "service credits" which are applied towards the next billing cycle. Service credits are typically only determined by fixed percentage of service payment. This strategy reflects the provider's desire to defer penalties, while making the redemption of the service credits contingent on the client staying for one more service window. Note that these are typically tiered credits that are percentages of the monthly service charge. These credits do not consider the fact that the impact on business (loss of customer confidence, revenue, etc) due to service unavailability could typically exceed the service credit amount. This issue, combined with the way service credits are designed and the lack of choice therein, are often cited as "hidden problems" from the client side regarding cloud SLAs (Griggs 2013).These problems serve as barriers to the wider adoption of cloud services. In our proposed strategy, we address the provider's need to defer penalties while designing an alternative that is more reflective of the realities of the client's business and hence, addresses some of the challenges just discussed. Therefore, we focus specifically on the service deficit case and introduce a deferred pricing approach as a viable alternative to service credits. This strategy models a scenario where the client is given a choice to either collect the penalty payment at the end of the service window or defer it to the next period. The provider incentivizes the deferral by charging a lower price for the next period. This serves two purposes: 1) it locks the client in for another period, and 2) it offers the

provider an opportunity to make up for the service shortfall as a risk-hedging method and possibly avoid having to pay the high penalty at all.

Next, Figure 1 below summarizes the interactions among the constructs in the proposed SLA Framework. In this research, we consider an SLA between a cloud service provider and its client as follows. The client requires n Virtual Machines (VM) to be available over a specified contract period T, at an uptime guarantee of α of the duration T. Such requirements can either be client-specified or part of a schedule of service levels formally offered by a provider. An SLA usually entails a penalty π for the provider if the promised service level is not met. Driven by the need to comply with an SLA, a provider typically allocates a set of k additional VMs as backup to the client as best-effort sustenance of the uptime guarantee. Thus, downtime will result only if at least $k + 1$ VMs concurrently fail.

Transient downtime distribution used to predict service availability is a function of n, k, and T. For a given combination of n and T, clearly, as k increases, failure resiliency increases. Hence, the likelihood of SLA violation decreases. However, the cost to the provider also increases as k increases, in addition to potential wastage of resources. Therefore, the determination of the optimal number of backup VMs that minimizes the expected total cost is the key problem to solve in availability–aware resource provisioning. The expected total cost is the summation of the provisioning cost and the expected penalty, where the provisioning cost, penalty and the backup allocation strategy also interact. If h is sufficiently high, the service provider may choose to allocate fewer k to in turn bear the risk of a higher penalty. When π is quite high, the provider will allocate many more backup resources in order to make the SLA violation probability as small as possible. We also derive a price-penalty schedule where breakeven prices are derived for various penalty levels. When the penalty is relatively low, the provider will allocate fewer backup VMs, thereby enabling him to lower the breakeven price for the VMs. On the other hand, a higher penalty rate will drive the provider to offset the risk of higher expected penalty costs through additional backup provisioning; this in turn leads the provider to set a higher breakeven price for such a contract.

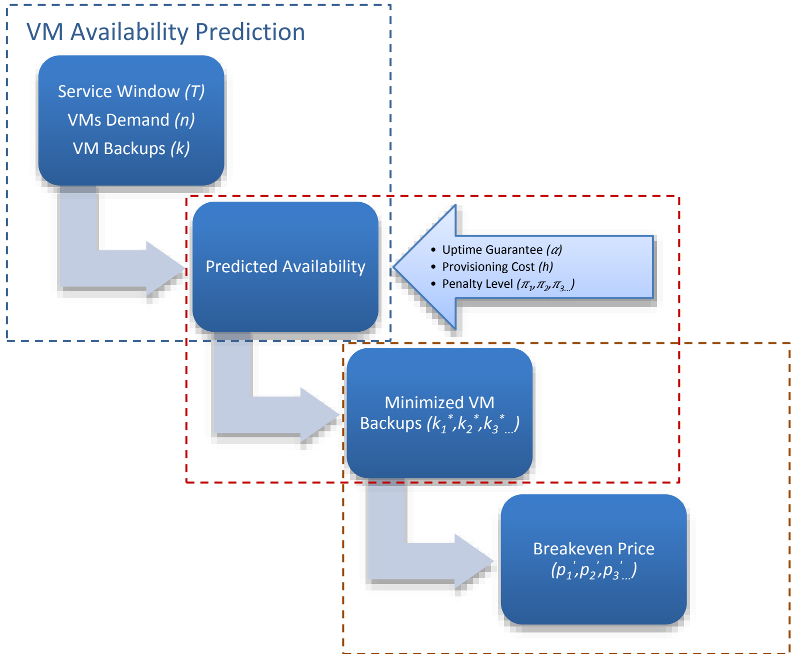


Figure 1. Interactions between Constructs in Availability-aware SLA Framework

II. Closed Form Solutions of Expected Penalizable Downtime

Case 1. Lower limit in the first segment: $\mathbf{0 \leq (1-\alpha)T \leq \gamma_0}$

$$\int_{(1-\alpha)T}^{T} (\tau - (1-\alpha)T) v(\tau) d\tau$$

$$= \int_{(1-\alpha)T}^{\gamma_0} (\tau - (1-\alpha)T)(a_0\tau + b_0) d\tau + \int_{\gamma_0}^{\gamma_1} (\tau - (1-\alpha)T)(a_1\tau + b_1) d\tau$$

$$+ \int_{\gamma_1}^{\gamma_2} (\tau - (1-\alpha)T)(a_2\tau + b_2) d\tau + \cdots + \int_{\gamma_{i-1}}^{\gamma_i} (\tau - (1-\alpha)T)(a_i\tau + b_i) d\tau + \cdots$$

$$+ \int_{\gamma_{m-2}}^{T} (\tau - (1-\alpha)T)(a_{m-1}\tau + b_{m-1}) d\tau$$

$$= \left(\frac{a_0}{3}{\gamma_0}^3 + \frac{b_0 + \alpha a_0 T - a_0 T}{2}{\gamma_0}^2 + (\alpha T b_0 - T b_0)\gamma_0 + \left(-\frac{1}{6}\alpha^3 + \frac{a_0}{2}\alpha^2 - \frac{a_0}{2}\alpha + \frac{a_0}{6}\right)T^3\right.$$

$$\left. + \left(\frac{b_0}{2}\alpha^2 - b_0\alpha + \frac{b_0}{2}\right)T^2\right)$$

$$+ \left(\frac{a_1}{3}{\gamma_1}^3 + \frac{b_1 + \alpha T a_1 - T a_1}{2}{\gamma_1}^2 + (\alpha T b_1 - T b_1)\gamma_1 - \frac{a_1}{3}{\gamma_0}^3\right.$$

$$\left. + \frac{T a_1 - b_1 - \alpha T a_1}{2}{\gamma_0}^2 + (T b_1 - \alpha T b_1)\gamma_0\right)$$

$$+ \left(\frac{a_2}{3}{\gamma_2}^3 + \frac{b_2 + \alpha T a_2 - T a_2}{2}{\gamma_2}^2 + (\alpha T b_2 - T b_2)\gamma_2 - \frac{a_2}{3}{\gamma_1}^3\right.$$

$$\left. + \frac{T a_2 - b_2 - \alpha T a_2}{2}{\gamma_1}^2 + (T b_2 - \alpha T b_2)\gamma_1\right) + \cdots$$

$$+ \left(\frac{a_i}{3}{\gamma_i}^3 + \frac{b_i + \alpha T a_i - T a_i}{2}{\gamma_i}^2 + (\alpha T b_i - T b_i)\gamma_i - \frac{a_i}{3}{\gamma_{i-1}}^3\right.$$

$$\left. + \frac{T a_i - b_i - \alpha T a_i}{2}{\gamma_{i-1}}^2 + (T b_i - \alpha T b_i)\gamma_{i-1}\right) + \cdots$$

$$+ \left(-\frac{a_{m-1}}{3}{\gamma_{m-2}}^3 + \frac{T a_{m-1} - b_{m-1} - \alpha T a_{m-1}}{2}{\gamma_{m-2}}^2 + (T b_{m-1} - \alpha T b_{m-1})\gamma_{m-2}\right.$$

$$\left. + \left(\frac{a_{m-1}}{2}\alpha - \frac{a_{m-1}}{6}\right)T^3 + \left(\alpha b_{m-1} - \frac{b_{m-1}}{2}\right)T^2\right)$$

Case 2. Lower limit in any segment $\boldsymbol{i \in (1, m-3)}$ ***(between the second and the penultimate segments):*** $\boldsymbol{\gamma_{i-1} \leq (1-\alpha)T \leq \gamma_i}$

$$\int_{(1-\alpha)T}^{T} (\tau - (1-\alpha)T)v(\tau)d\tau$$

$$= \int_{(1-\alpha)T}^{\gamma_i} (\tau - (1-\alpha)T)(a_i\tau + b_i)d\tau + \int_{\gamma_i}^{\gamma_{i+1}} (\tau - (1-\alpha)T)(a_{i+1}\tau + b_{i+1})d\tau$$

$$+ \cdots + \int_{\gamma_{m-2}}^{T} (\tau - (1-\alpha)T)(a_{m-1}\tau + b_{m-1})d\tau$$

$$= \left(\frac{a_i}{3}\gamma_i{}^3 + \frac{b_i + \alpha a_i T - a_i T}{2}\gamma_i{}^2 + (\alpha T b_i - T b_i)\gamma_i + \left(-\frac{1}{6}\alpha^3 + \frac{a_i}{2}\alpha^2 - \frac{a_i}{2}\alpha + \frac{a_i}{6}\right)T^3\right.$$

$$\left. + \left(\frac{b_i}{2}\alpha^2 - b_i\alpha + \frac{b_i}{2}\right)T^2\right)$$

$$+ \left(\frac{a_{i+1}}{3}\gamma_{i+1}{}^3 + \frac{b_{i+1} + \alpha T a_{i+1} - T a_{i+1}}{2}\gamma_{i+1}{}^2 + (\alpha T b_{i+1} - T b_{i+1})\gamma_{i+1} - \frac{a_{i+1}}{3}\gamma_i{}^3\right.$$

$$\left. + \frac{T a_{i+1} - b_{i+1} - \alpha T a_{i+1}}{2}\gamma_i{}^2 + (T b_{i+1} - \alpha T b_{i+1})\gamma_i\right) + \cdots$$

$$+ \left(-\frac{a_{m-1}}{3}\gamma_{m-2}{}^3 + \frac{T a_{m-1} - b_{m-1} - \alpha T a_{m-1}}{2}\gamma_{m-2}{}^2 + (T b_{m-1} - \alpha T b_{m-1})\gamma_{m-2}\right.$$

$$\left. + \left(\frac{a_{m-1}}{2}\alpha - \frac{a_{m-1}}{6}\right)T^3 + \left(\alpha b_{m-1} - \frac{b_{m-1}}{2}\right)T^2\right)$$

For instance, if the lower limit lies in the second segment, i.e. $\gamma_0 \leq (1-\alpha)T \leq \gamma_1$, then the expected penalizable downtime is derived as follows:

$$\int_{(1-\alpha)T}^{T} (\tau - (1-\alpha)T)v(\tau)d\tau$$

$$= \int_{(1-\alpha)T}^{\gamma_1} (\tau - (1-\alpha)T)(a_1\tau + b_1)d\tau + \int_{\gamma_1}^{\gamma_2} (\tau - (1-\alpha)T)(a_2\tau + b_2)d\tau + \cdots$$

$$+ \int_{\gamma_{i-1}}^{\gamma_i} (\tau - (1-\alpha)T)(a_i\tau + b_i)d\tau + \cdots + \int_{\gamma_{m-2}}^{T} (\tau - (1-\alpha)T)(a_{m-1}\tau + b_{m-1})d\tau$$

$$= \left(\frac{a_1}{3}{\gamma_1}^3 + \frac{b_1 + \alpha a_1 T - a_1 T}{2}{\gamma_1}^2 + (\alpha T b_1 - T b_1)\gamma_1 + \left(-\frac{1}{6}\alpha^3 + \frac{a_1}{2}\alpha^2 - \frac{a_1}{2}\alpha + \frac{a_1}{6}\right)T^3\right.$$

$$\left. + \left(\frac{b_1}{2}\alpha^2 - b_1\alpha + \frac{b_1}{2}\right)T^2\right)$$

$$+ \left(\frac{a_2}{3}{\gamma_2}^3 + \frac{b_2 + \alpha T a_2 - T a_2}{2}{\gamma_2}^2 + (\alpha T b_2 - T b_2)\gamma_2 - \frac{a_2}{3}{\gamma_1}^3\right.$$

$$\left. + \frac{T a_2 - b_2 - \alpha T a_2}{2}{\gamma_1}^2 + (T b_2 - \alpha T b_2)\gamma_1\right) + \cdots$$

$$+ \left(\frac{a_i}{3}{\gamma_i}^3 + \frac{b_i + \alpha T a_i - T a_i}{2}{\gamma_i}^2 + (\alpha T b_i - T b_i)\gamma_i - \frac{a_i}{3}{\gamma_{i-1}}^3\right.$$

$$\left. + \frac{T a_i - b_i - \alpha T a_i}{2}{\gamma_{i-1}}^2 + (T b_i - \alpha T b_i)\gamma_{i-1}\right) + \cdots$$

$$+ \left(-\frac{a_{m-1}}{3}{\gamma_{m-2}}^3 + \frac{T a_{m-1} - b_{m-1} - \alpha T a_{m-1}}{2}{\gamma_{m-2}}^2 + (T b_{m-1} - \alpha T b_{m-1})\gamma_{m-2}\right.$$

$$\left. + \left(\frac{a_{m-1}}{2}\alpha - \frac{a_{m-1}}{6}\right)T^3 + \left(\alpha b_{m-1} - \frac{b_{m-1}}{2}\right)T^2\right)$$

Case 3. Lower limit in the last segment: $\boldsymbol{\gamma_{m-2} \leq (1-\alpha)T \leq T}$

$$\int_{(1-\alpha)T}^{T} (\tau - (1-\alpha)T)v(\tau)d\tau = \int_{(1-\alpha)T}^{T} (\tau - (1-\alpha)T)(a_{m-1}\tau + b_{m-1})d\tau$$

$$= \left(-\frac{a_{m-1}}{6}\alpha^3 + \frac{a_{m-1}}{2}\alpha^2\right)T^3 + \frac{b_{m-1}}{2}\alpha^2 T^2$$

III. Proof of Lemmas

Lemma 2: The expected total cost is a convex function of k.

Proof: From Equation (1), for any $k_1, k_2 \in k, k_1 \neq k_2$,

$$Q_{\frac{k_1+k_2}{2}} - \frac{Q_{k_1}+Q_{k_2}}{2} = h\frac{(k_1+k_2)}{2}T + \pi\int_{\tau_0}^{T} v\left(\tau, \frac{(k_1+k_2)}{2}\right)(\tau-\tau_0)d\tau - \frac{hk_1T+\pi\int_{\tau_0}^{T} v(\tau,k_1)(\tau-\tau_0)d\tau + hk_2T+\pi\int_{\tau_0}^{T} v(\tau,k_2)(\tau-\tau_0)d\tau}{2}$$

$$= \pi\int_{\tau_0}^{T} v\left(\tau, \frac{(k_1+k_2)}{2}\right)(\tau-\tau_0)d\tau - \frac{\pi\int_{\tau_0}^{T} v(\tau,k_1)(\tau-\tau_0)d\tau + \pi\int_{\tau_0}^{T} v(\tau,k_2)(\tau-\tau_0)d\tau}{2}$$

$$= \pi\left(\int_{\tau_0}^{T} v\left(\tau, \frac{(k_1+k_2)}{2}\right)(\tau-\tau_0)d\tau - \frac{\int_{\tau_0}^{T} v(\tau,k_1)(\tau-\tau_0)d\tau + \int_{\tau_0}^{T} v(\tau,k_2)(\tau-\tau_0)d\tau}{2}\right).$$

From Lemma 1, we get,

$\int_{\tau_0}^{T} v\left(\tau, \frac{(k_1+k_2)}{2}\right)(\tau-\tau_0)d\tau < \frac{\int_{\tau_0}^{T} v(\tau,k_1)(\tau-\tau_0)d\tau + \int_{\tau_0}^{T} v(\tau,k_2)(\tau-\tau_0)d\tau}{2}$. Therefore, $\pi\left(\int_{\tau_0}^{T} v\left(\tau, \frac{(k_1+k_2)}{2}\right)\tau d\tau - \frac{\int_{\tau_0}^{T} v(\tau,k_1)\tau d\tau + \int_{\tau_0}^{T} v(\tau,k_2)\tau d\tau}{2}\right) < 0$. That is,

$Q_{\frac{k_1+k_2}{2}} - \frac{Q_{k_1}+Q_{k_2}}{2} < 0$, or $Q_{\frac{k_1+k_2}{2}} < \frac{Q_{k_1}+Q_{k_2}}{2}$. Therefore, the expected total cost is a convex function in k.

QED.

IV. Divide-and-conquer Algorithm When the Convexity Assumption Does not Hold

Inputs: $i = 1, j = n$, Q_k as calculated from (1) given n, T, α, $v(\tau)$, h, π, and $k = [i, j]$ & integer

```
Minimize Total Cost(i, j, Q_min)//     Q_min is the output of the function
{       if i = j − 1 then Q_min = Min(Q_i, Q_j);
        else if i = j then Q_min = Q_i;
        else
        {
                mid = (i+j)/2;
                Q = Min (Q_i, Q_mid, Q_j)

                If Q = Q_i, then solve  Minimize Total Cost(i, mid, Q_min);
                If Q = Q_j, then solve Minimize Total Cost(mid, j, Q_min);
                If Q = Q_{(i+j)/2}, then solve
                        Minimize Total Cost(i, mid, Q_min');
                        Minimize Total Cost(mid, j, Q_min'');
                        Q_min = Min(Q_min', Q_min'');
        }
}
k* = arg min(Q_min);
Return  k*
```

Figure 2. Divide-and-conquer Algorithm to Derive Optimal Number of Backup VMs

The computational complexity of the algorithm is $\boldsymbol{O(n)}$.

Proof: In the worst case, for each iteration, the lowest expected value is always corresponding to the midpoint, which satisfies the condition, $Q = Q_{\frac{i+j}{2}}$ .Thus, the original problem is divided into two subproblems which iteratively continue until it reaches the terminal case where there are at most two elements ($i = j - 1 \; or \; i = j$). Let $T(n)$ be the time in the worst-case scenario, therefore

$$T(n) = \begin{cases} O(1), & n = 1 \; or \; n = 2 \\ 2T\left(\frac{n}{2}\right) + O(1), & n > 2 \end{cases}$$

Applying the Master Theorem(Cormen and Leiserson 2001),

$$T(n) = O\left(n^{\log_2 2}\right) = O(n) \quad \textbf{QED.}$$

V. Impact of Accuracy Level of Piecewise Linear Approximation Approach

We are interested in checking how many more segments are utilized at higher parameter values as opposed to low values, such as 0.5. In Figure 3, below, we plot the number of line segments for two cases, $n = 50$ and $n = 100$ and $T = 30\ days$, with accuracy, r, set to five different levels ranging from 0.1 to 0.99. The greater the number of line segments, the resulting piecewise linear function will be more finely tuned to the real downtime distribution obtained from the historical data. When the accuracy level goes from 0.5 to 0.8, the number of segments increases significantly, from 169 to 333 ($n = 50, k = 13, T = 30\ days$) and 130 to 272 ($n = 100, k = 21, T = 30\ days$). This result implies the linear approximation algorithm responds to high levels of accuracy (≥ 0.8) by significantly increasing the number of segments in approach. We ran this experiment to demonstrate the increased level of precision achieved through the hundreds of line segments that represent the approximated function. This establishes that, as far as approximation is concerned, from the standpoint of increased precision, there is indeed something to be gained by choosing a higher accuracy parameter value.

We next examine how the accuracy parameter value affects the contract parameters. In an experiment using $T = 30\ days$ with $n = 50$, when the accuracy level is 0.1 and 0.5, $k^* = 13$ corresponds to the minimum expected total cost. This changes when the accuracy level is increased to an amount ≥ 0.8, where k^* becomes 14 as shown in Figure 4. Hence, a provider would be misled if he chose a lower accuracy parameter. Note that the downtime distribution is a key ingredient not just in determining the optimal k, it also plays a role in determining the price alongside k^*. Therefore, the imprecision due to a low accuracy level gets even more amplified during price-setting. This result is also consistent with our previous finding where the effect on the penalizable downtime is also drastic when we vary the approximation level from 0.5 to 0.8. This points to a greater concern of being misled in terms of decisions that are driven by the downtime distribution when using a low accuracy level. Hence, while extreme precision of 0.99 may be unnecessary (and perhaps reassuringly so), the accuracy level still needs to be reasonably high (at least 0.8), given its impact on the various contract constructs.

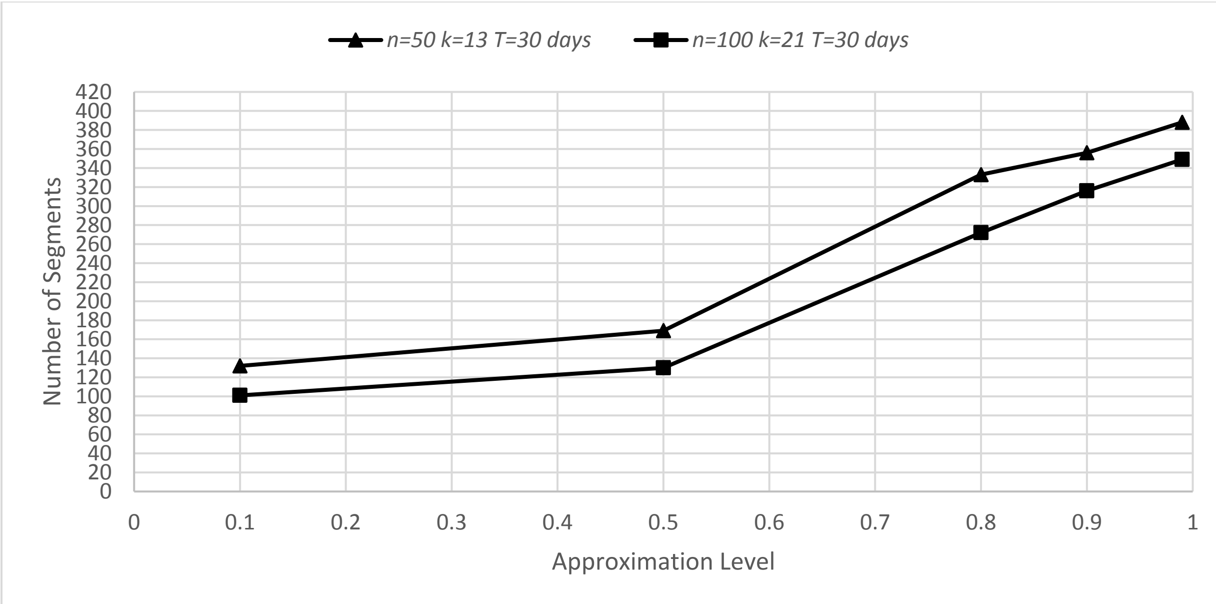


Figure 3. Impact of Approximation Level on Number of Segments

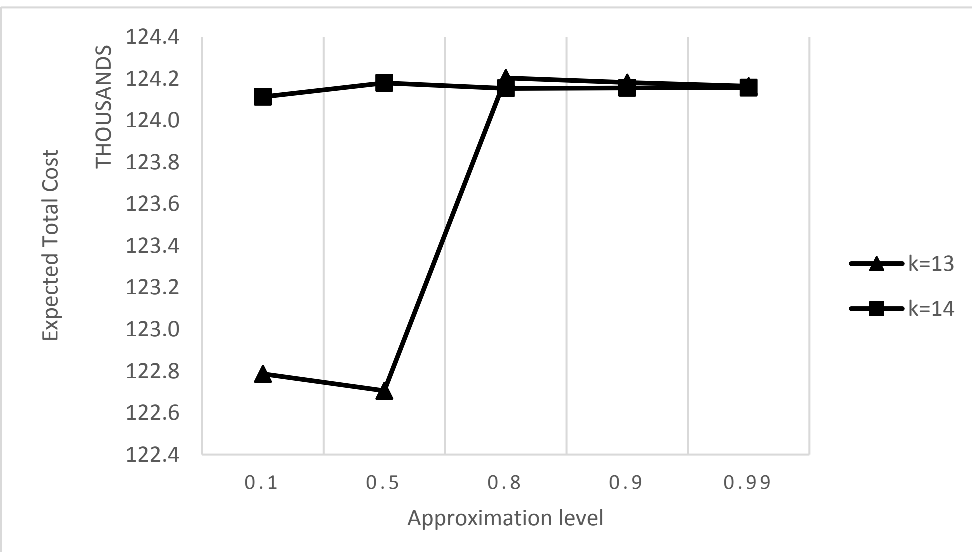
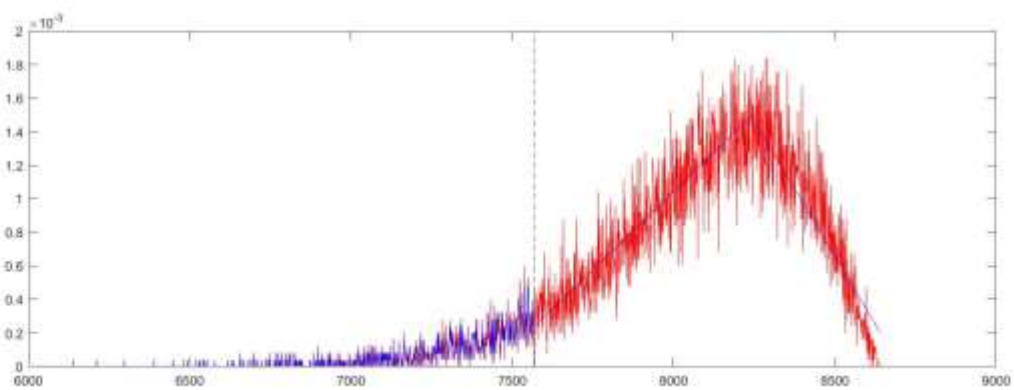
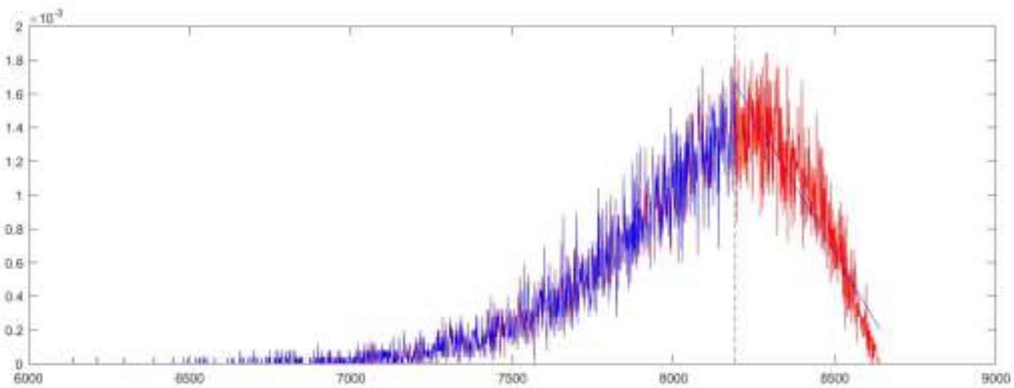


Figure 4. Impact of Approximation Level on Total Cost

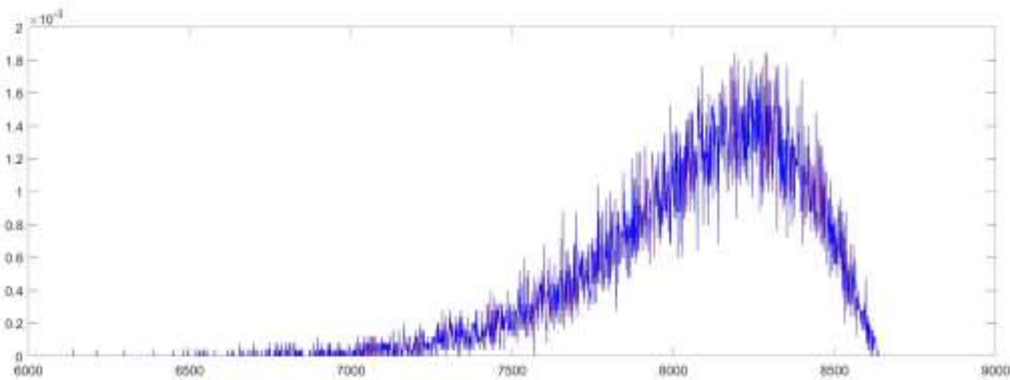
We also provide a visualization of the effects of piecewise linear approximation under various accuracy levels in Figure 5. Given a downtime distribution of $n = 50, k = 1, T = 30\ days$, we vary the accuracy level at 0.1, 0.5, 0.99, respectively. In each graph of Figure 4, the true downtime distribution derived from the server log data is shown in red, while the piecewise linear function which approximates this distribution is shown in blue. Inspecting the graph for accuracy level = 0.99, we see that the two distributions is identical enough that the red curve is almost entirely hidden under the blue. Looking at the graph for 0.5, we see this same phenomenon unfold up to a certain point, denoted by the dotted vertical line. Beyond that point, we notice just one blue linear segment, representing a very large portion of that downtime pdf. Note that this gross linearization happens in a region that is especially critical – high values of downtime. The result is even poorer for 0.1, understandably. However, going back to the accuracy level of 0.99, the result implies that the piecewise linear approximation approach is likely to result in a very precise estimate on the downtime distribution. The practitioner could use this approach on the historical data of any datacenter to derive the key results with highly accurate approximations.



$a)\ r = 0.1$



$b)\ r = 0.5$



$c)\ r = 0.99$

Figure 5(a,b,c). Impact of Approximation Level on Accuracy of Fitting at $n = 50, k = 1, T = 30\ days$

VI. Effect of Penalty and Provisioning Cost on Resource Price

Here we computationally extend our analysis in Section 5.1 by studying the impact of penalty and provisioning cost on VM pricing. The experiments are run for $n = 50$ and 100, with $T = 30$ days. We derive the breakeven price corresponding to k^* for a given (n, T) combination according to our model in Section 5.1. Similar to the experiments above, we study the sensitivity to increasing penalty rates and then study the effects of increasing provisioning costs. As expected, the breakeven price increases with higher penalty rates, with a notably sharp increase followed by a gradual increase. As penalty rates increase, the provider allocates more backup VMs to hedge the risk of SLA violation, which in turn necessitates a higher price. However, with increasing k, the risk of SLA violation progressively shrinks; as a consequence, the expected penalty cost grows at an ever-slowing pace, thereby reducing the rate of price increase.

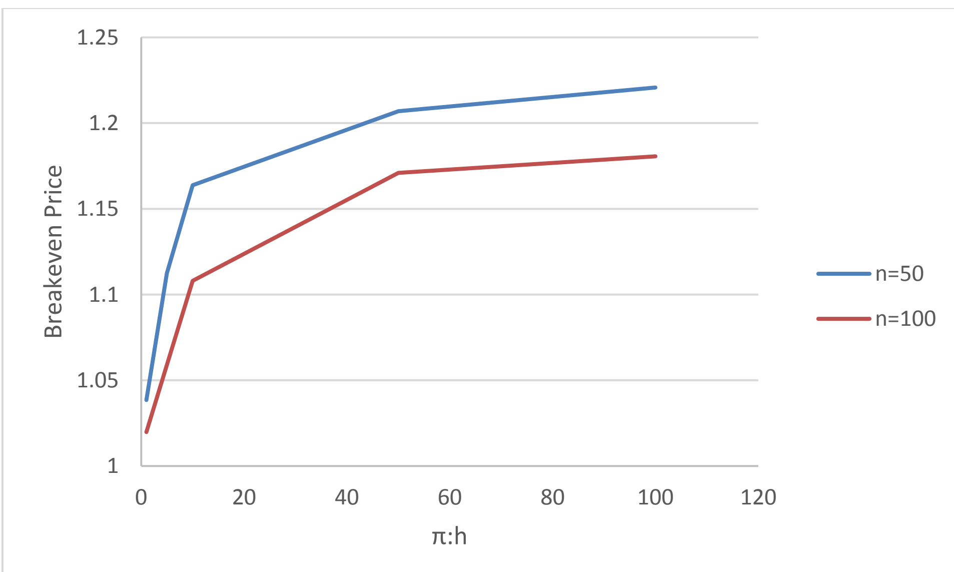
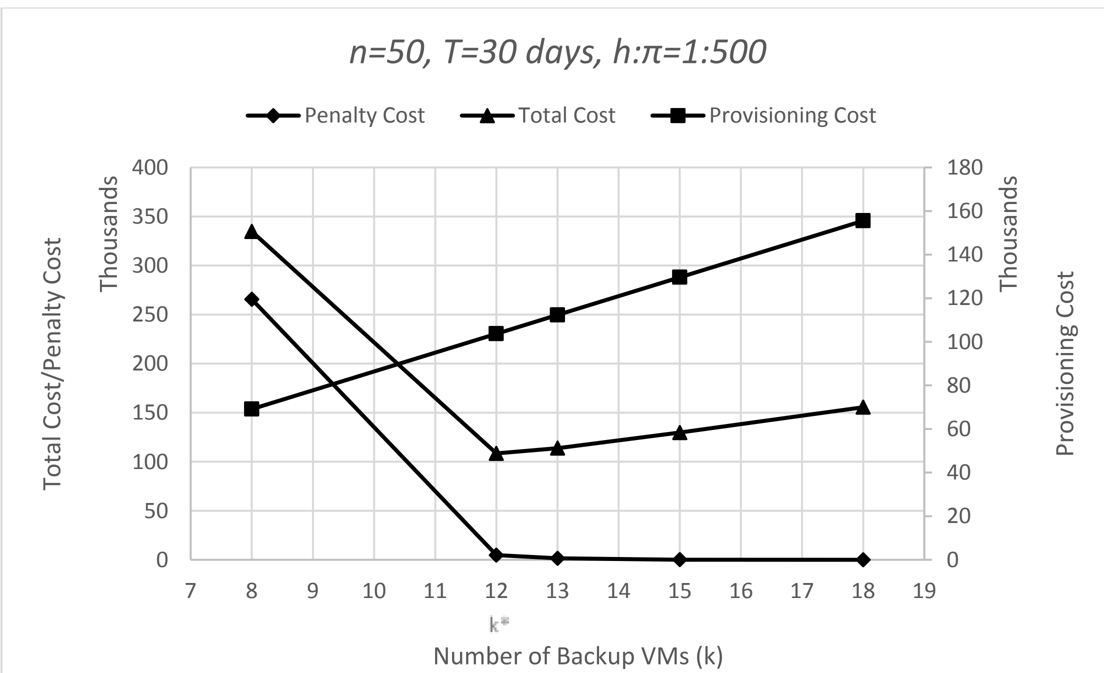


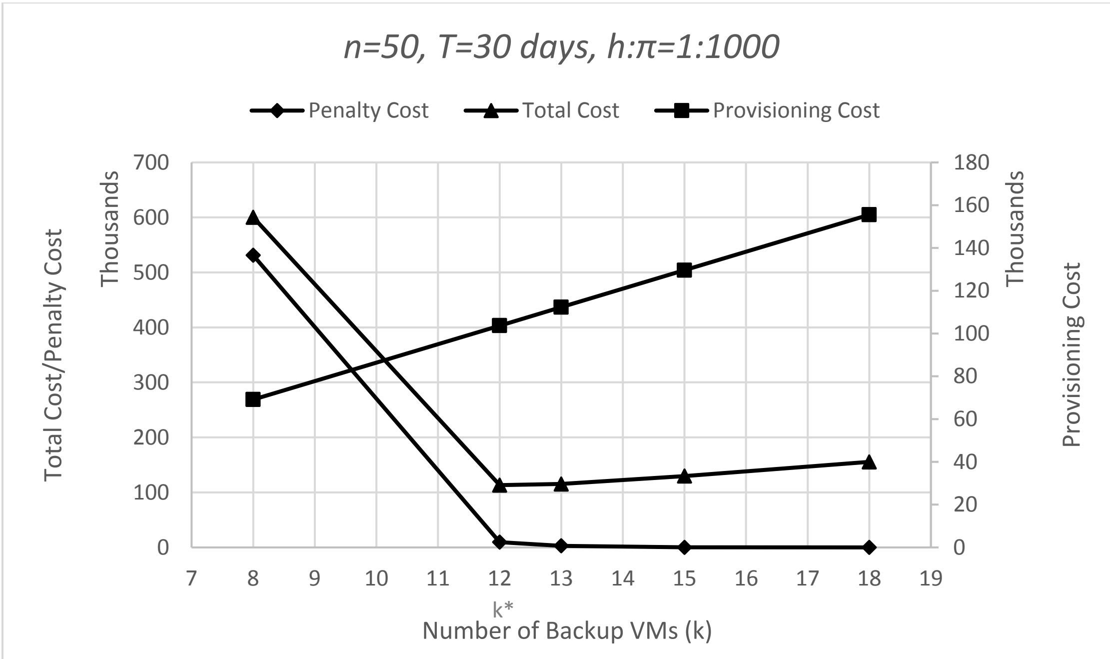
Figure 6. Sensitivity of Impact of Penalty Cost on Breakeven Pricing

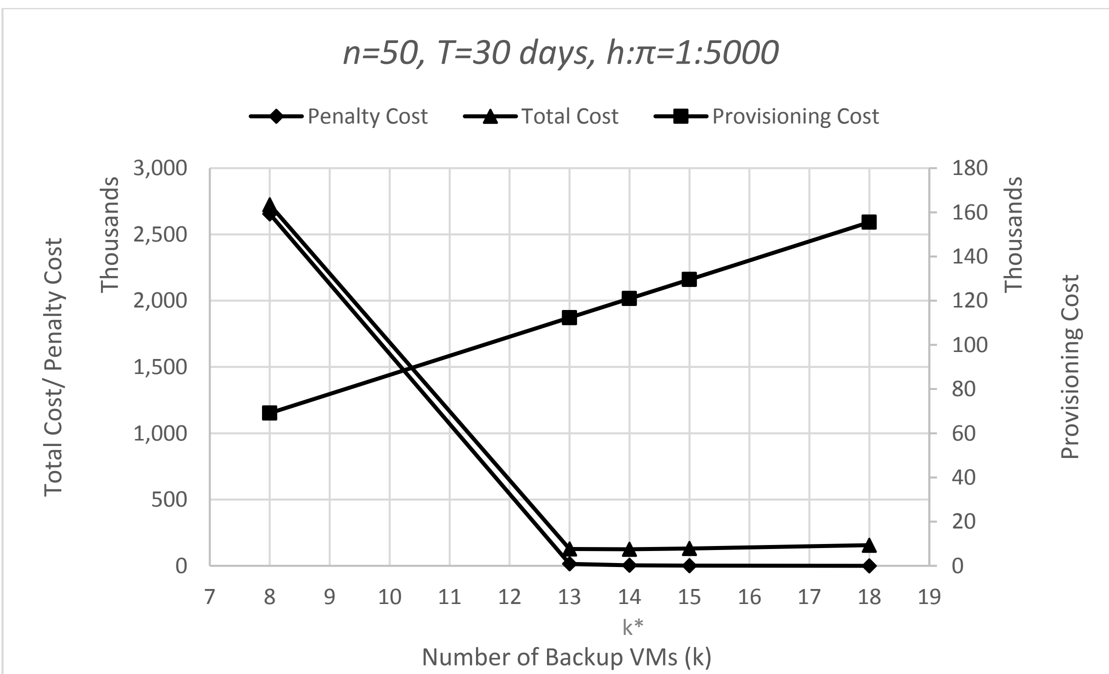
VII. Comparison on the Costs between Divide-and-conquer Algorithm and the Benchmark under Various Provisioing-penalty Ratios

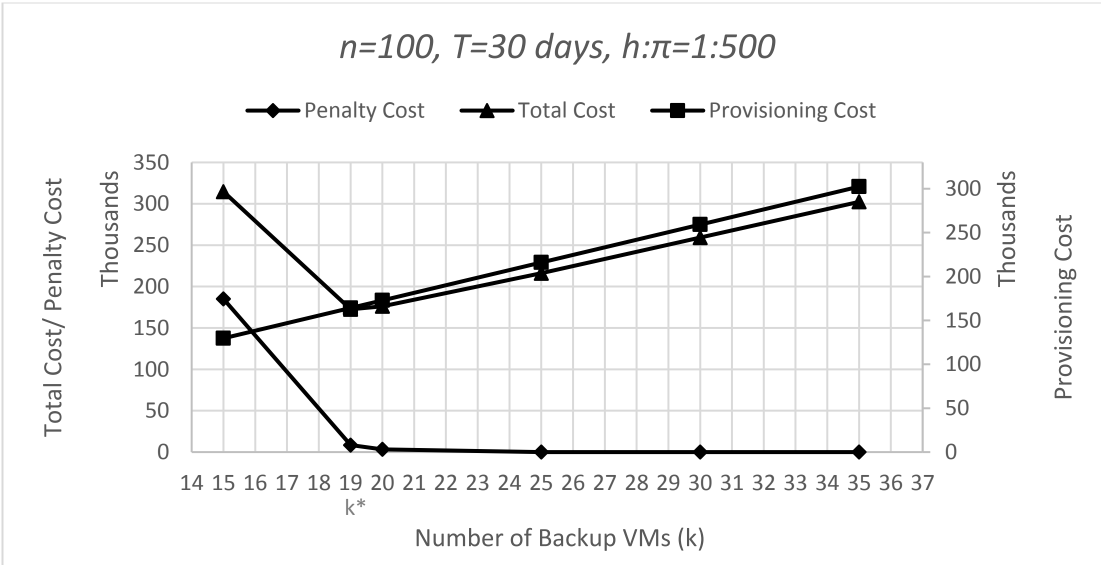
We had set the ratio between provisioning cost and penalty rate to 1:100 in Section 6.5. We now vary the ratio ranging from 1:500, 1:1000, to 1:5000 for both $n = 50$ and $n = 100$ scenarios in Figure 7. As the penalty rate increases in comparison to the provisioning cost, we notice that the expected total cost increases significantly, even if k^*does not increase as seen when comparing the figures for 1:500 and 1:1000. In fact, as the ratio begins to favor the penalty more, the expected total cost curve gets pulled towards the penalty curve. Note that given that the outlier values obscure the reading of the graphs, we have also provided the same data in a tabular form (Table 1).

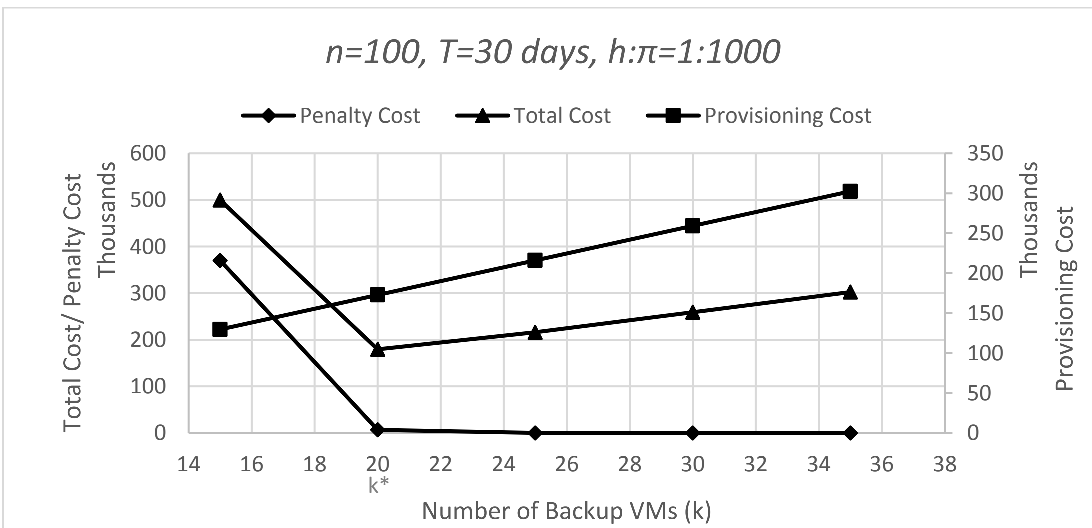
As an aside, these figures also empirically validate the convexity of the expected total cost function as proven in Lemma 2.




n=50, T=30 days, h:π=1:1000
Penalty Cost
Total Cost
Provisioning Cost
Total Cost/Penalty Cost
Thousands
Provisioning Cost
Thousands
700
600
500
400
300
200
100
0
180
160
140
120
100
80
60
40
20
0
7
8
9
10
11
12
13
14
15
16
17
18
19
k*
Number of Backup VMs (k)


n=50, T=30 days, h:π=1:5000
Penalty Cost
Total Cost
Provisioning Cost
Total Cost/ Penalty Cost
Thousands
Provisioning Cost
Thousands
3,000
2,500
2,000
1,500
1,000
500
0
180
160
140
120
100
80
60
40
20
0
7
8
9
10
11
12
13
14
15
16
17
18
19
k*
Number of Backup VMs (k)


n=100, T=30 days, h:π=1:500
Penalty Cost
Total Cost
Provisioning Cost
Total Cost/ Penalty Cost
Thousands
350
300
250
200
150
100
50
0
14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37
k*
Number of Backup VMs (k)
300
250
200
150
100
50
0
Thousands
Provisioning Cost


n=100, T=30 days, h:π=1:1000
Penalty Cost
Total Cost
Provisioning Cost
Total Cost/ Penalty Cost
Thousands
600
500
400
300
200
100
0
14 16 18 20 22 24 26 28 30 32 34 36 38
k*
Number of Backup VMs (k)
350
300
250
200
150
100
50
0
Thousands
Provisioning Cost

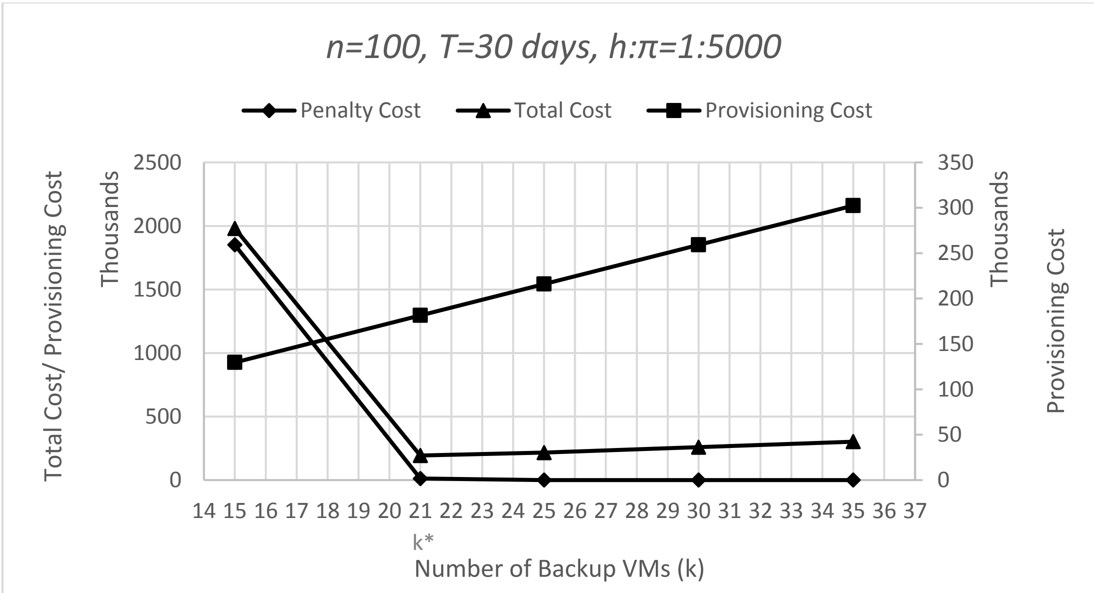


Figure 7. Comparisons on the Costs between Divide-and-conquer Algorithm and the Benchmark

n=50, T=30 days						
1:500		***k=8***	***k=12***	***k=13***	***k=15***	***k=18***
	Provisioning Cost	69120	103680	112320	129600	155520
	Expected Penalty Cost	265578.5	4786.07	1480.459	104.358	5.32
	Total Cost	334698.5	108466.1	113800.5	129704.4	155525.3
1:1000		***k=8***	***k=12***	***k=13***	***k=15***	***k=18***
	Provisioning Cost	69120	103680	112320	129600	155520
	Expected Penalty Cost	531156.9	9572.14	2960.917	208.716	10.64
	Total Cost	600276.9	113252.1	115280.9	129808.7	155530.6
1:5000		***k=8***	***k=13***	***k=14***	***k=15***	***k=18***
	Provisioning Cost	69120	112320	120960	129600	155520
	Expected Penalty Cost	2655000	14804.59	3995.287	1043.58	53.2
	Total Cost	2724120	127124.6	124955.3	130643.6	155573.2

n=100, T=30 days						
1:500		***k=15***	***k=19***	***k=20***	***k=25***	***k=30***
	Provisioning Cost	129600	164160	172800	216000	259200
	Expected Penalty Cost	185087.17	8403.63	3368.58	21.38	0
	Total Cost	314687.17	172563.63	176168.58	216021.38	259200
1:1000		***k=15***	***k=20***	***k=25***	***k=30***	***k=35***
	Provisioning Cost	129600	172800	21600	259200	302400
	Expected Penalty Cost	370174.35	6737.16	42.77	0	0
	Total Cost	499774.35	179537.16	216042.77	259200	302400
1:5000		***k=15***	***k=21***	***k=25***	***k=30***	***k=35***
	Provisioning Cost	129600	181440	216000	259200	302400
	Expected Penalty Cost	1850871.74	12076.14	213.84	0	0
	Total Cost	1980471.74	193516.14	216213.84	259200	302400

Table 1. Comparison on Costs

VIII. Effect of SLA Violation on Penalty-deferred Pricing

In Section 5.2, we analytically derived the Period 2 price at which the provider can incentivize the client to defer the receipt of penalty in the event of SLA violation in Period 1. Here we supplement that analysis by computationally studying the impact of the SLA violation in Period 1 on the discounted price in Period 2. We ran the experiments for $n = 50$, $\alpha = 0.99$, $T = 30\, days$, $p_1 = 10$. We varied the downtime accumulated in the first service window (Period 1) from $90\Delta t$ to $175\Delta t$ since downtime less than $(1-\alpha)t_1$incurs zero penalty. In Figure 8, we plot the target service level in Period 2,α_2^{obj}, and the upper bound on the discounted price $\overline{p_2}$ as derived in Section 5.2. As expected, the target service level in Period 2 has to increase in order to compensate for the service level in Period 1 decreasing. However, after $173\Delta t$, α_2^{obj} becomes equal to 100% , effectively implying that beyond this point, the availability shortfall in Period 1 is so large that there is no possibility to fulfill the uptime commitment across two periods even though the resources committed promise a service level of 100% in Period 2. We also note that as the service shortfall in Period 1 increases, larger discounts need to be provided in Period 2 in order to incentivize the buyer.

Further, we evaluated the provider's profits from the penalty deferral strategy against a benchmark strategy, which is the case where there is no option of deferring the penalty and the buyer collects the penalty at the end of Period 1. Note that under such circumstances, there is a reasonable likelihood that the buyer may switch to another provider. Here we performed two comparisons against: (a) the case where the buyer switches to another provider: the penalty deferral strategy would give significantly higher revenues in Period 2 as the benchmark would yield zero revenue when the buyer switches providers; (b) the much stricter case where the client does not switch to another the service provider and continues to purchase VMs for another period at non-discounted price. When downtime accumulated in the first service window (Period 1) is 90Δt, the profit for benchmark bears a rather small 0.000010213% advantage to the profit for our penalty deferral strategy. However, when we varied the downtime accumulated in Period 1 to $165\Delta t$, this small lead increased to 0.000018344%, perhaps the provider gives up some profits in exchange for securing the SLA for another period in order to generate goodwill. As we mentioned, if we consider the possibility of even a few buyers switching to an alternate provider as, this small difference in profits disappears, making the deferral strategy more attractive to the provider. When downtime accumulated in the first service window is 90Δt and the client chooses to switch to another provider after receives the payment, the profit for our penalty deferral strategy has a significant higher of 104% advantage. As the downtime accumulated to 165Δt, our approach dominates even by 1108%. This is because as the penalizable downtime in Period 1 increases, the penalty that the provider has to pay becomes much higher such that the provider would be eager to seek any opportunity for avoiding it. Thus as a viable and profitable alternative, the penalty deferral strategy provides a chance for locking the client for another period as well as a chance to fulfill the uptime guarantee.

Figure 8. Impact of the Availability Shortfall on the Target Service Level and Incentive Price

Reference

Cormen, T. H. and C. E. Leiserson (2001). Introduction to algorithms.

Du, A. Y., S. Das, Z. Yang, C. Qiao and R. Ramesh (2015). "Predicting Transient Downtime in Virtual Server Systems: An Efficient Sample Path Randomization Approach." IEEE Transactions on Computers **64**(12): 3541-3554.

Griggs, S. (2013). "5 Hidden Problems with Cloud SLAs." from http://www.thewhir.com/blog/5-hidden-problems-cloud-slas.

Kauffman, R. J. and R. Sougstad (2008). "Risk management of contract portfolios in IT services: the profit-at-risk approach." Journal of Management Information Systems **25**(1): 17-48.

Wang, S. and W. Chaovalitwongse (Forthcoming). "Piecewise Linear Segmentation of Time Series Using a Data- Driven Threshold With Guaranteed Accuracy." Pattern Recognition Letters.