Prediction-Based Admission Control for IaaS Clouds with Multiple Service Classes

Marcus Carvalho, Daniel Menascé, Francisco Brasileiro

Summarized by
Uma Tadakamalla
Outline

I. Introduction

II. Problem Statement

III. Prediction Based Admission Control

IV. Evaluation Methodology

V. Results

VI. Conclusions
I. Introduction

- **Problem domain**: cloud computing

- **Problem**: How to make efficient resource management decisions
  
  - to maximize cloud provider’s profit
  
  - while meeting Service Level Objectives (SLO) for different classes of requests.
I. Introduction

- **Capacity Planning**
  - Decides capacity needed to execute an expected workload

- **Admission control**
  - Decides whether to admit or reject a request in order to meet SLOs of allocated requests

- **Scheduling**
  - Chooses the VMs that will be running and the physical machines

![Diagram of resource management process]

*Figure 2. Cloud resource management process.*
I. Introduction

VM request state diagram

Figure 1. VM request state diagram in our IaaS cloud model.
I. Introduction

Admission Control

- decides on the minimum set of requests to reject so that SLOs of admitted requests are met.
- helps cloud providers efficiently manage resources
II. Problem Statement

- $M$ number of machines owned by provider
- $K$ number of classes offered by provider
- $C_m$ capacity of machine $m$
- $W_k$ workload stream for class $k$
- $S_{jk}$ resource capacity requested for the $j$-th VM request of class $k$
- $D_{jk}$ service demand in time units for the $j$-th VM request of class $k$
- $P_{jk}$ penalty to the provider in violating SLOs for the $j$-th VM request of class $k$
- Epoch $E_i$ - time slots of finite duration. $N$ number of epochs
- $b(E_i)$ begin of epoch $E_i$, $e(E_i)$ end off epoch $E_i$, $\Delta_i$ Duration of epoch
II. Problem Statement, cont.

- **Optimization problem**

  Given input \( \Phi = \langle N, M, K, W_k, S_{jk}, D_{jk}, C_m, A_{im} \rangle \)

  Objective function is to maximize profit: revenue - penalty

  \[
  \max \quad \mathcal{P} = \sum_{k=1}^{K} |W_k| \left( \sum_{m=1}^{M} \sum_{i=1}^{N} x_{jk} \cdot y_{ijkm} \cdot g_{jk} \cdot \Delta_i - p_{jk} \right) \quad (1)
  \]

- \( x_{jk} \{0, 1\} \): admission of j-th VM request of class k in machine m at epoch i.
- \( y_{ijkm} \{0, 1\} \): allocation of j-th VM request of class k in machine m at epoch i.
- \( g_{jk} \): revenue per time unit for running the j-th VM request of class k
- \( P_{jk} \): penalty for violating SLO for the j-th VM request of class k
II. Problem Statement, cont.

- **Capacity constraint**

  Capacity allocated for all VMs in a machine must be less than machine capacity and unavailable machines do not have any allocations

  \[ \sum_{k=1}^{K} \sum_{j=1}^{|W_k|} y_{ijkm} \cdot S_{jk} \leq C_m \cdot A_{im}, \quad \forall i, m \quad (2) \]

  - \( y_{ijkm} \{0, 1\} \): allocation of \( j^{th} \) VM request of class \( k \) in machine \( m \) at epoch \( i \).
  - \( S_{jk} \): resource capacity request for the \( j^{th} \) VM request of class \( k \).
  - \( C_m \): Capacity of machine \( m \)
  - \( A_{im} \{0, 1\} \): availability of machine \( m \) at epoch \( i \)
II. Problem Statement, cont.

- **Admission constraint**
  VM request can only be allocated iff it has been admitted.
  \[ y_{ijk} \leq x_{jk}, \quad \forall i, j, k, m \]  
  (3)

- **SLA constraint**
  SLO fulfillment rates must be greater than or equal to the minimum accepted rate defined for each class.
  \[ \theta_k \geq \theta_k^{\text{min}}, \quad \forall k \]  
  (4)
II. Problem Statement, cont.

- NP-complete problem

- Solution requires knowledge of future demands and machine availability values – unrealistic
III. Prediction based admission control

- **Quota-based approach**
  - limits the amount of resources that can be allocated to a class.
  - assign dynamically quotas to each class
  - based on available capacity for a class and its VM availability SLO

- **Preemptive priority scheduling**
  - Allocates VMs of higher priority classes first
  - Capacity available for a class depends not only on the available cloud capacity but also on the demand from higher priority classes
  - \( k=1 \rightarrow \) highest priority; \( k=K \rightarrow \) lowest priority class
III. Prediction based admission control

Available capacity for class k at epoch I

- total capacity of all machines – capacity used up so far by requests of classes one to k-1

\[ c_{ik} = \sum_{m=1}^{M} A_{im} \cdot C_m - \sum_{k'=1}^{k-1} \sum_{j=1}^{||W_{k'}||} x_{jk'} \cdot y_{ijk'm} \cdot S_{jk'}. \quad (5) \]

- \( A_{im} \in \{0,1\} \) availability of machine m at epoch i

- \( C_m \) capacity of machine m

- \( x_{jk} \in \{0,1\} \) admission of the j-th VM request of class k

- \( y_{ijk'm} \in \{0,1\} \) allocation of j-th VM request of class k in machine m and epoch i

- \( S_{jk} \) resource capacity requested for the j-th VM request of class k
III. Prediction based admission control

VM availability for the j-th VM of class k

- Percentage of time a request is allocated.

\[
\alpha_{jk} = \frac{\sum_{i=1}^{N} \sum_{m=1}^{M} y_{ijkm} \cdot \Delta_i}{r(V_{jk}) - s(V_{jk})} \cdot 100\%.
\]

- \( y_{ijkm} \in \{0, 1\} \) allocation of j-th VM request of class k in machine m and epoch i
- \( S_{jk} \) resource capacity requested for the j-th VM request of class k
- \( \Delta_i \) Duration of epoch i
- \( R(V_{jk}) \) release time of j-th VM request of class k
- \( S(V_{jk}) \) submission time of the j-th VM request of class k
III. Prediction based admission control

Little’s Law

Mean number of requests in a system

- mean VM sojourn time * mean throughput

\[ L = R \cdot \lambda = \frac{S}{\alpha} \cdot \frac{c}{S} = \frac{c}{\alpha} \]  \hspace{1cm} (7)

- \( L \) mean number of requests in a system
- \( R \) mean VM sojourn time
- \( \lambda \) mean throughput
- \( S \) mean service demand
- \( \alpha \) mean VM availability
- \( c \) mean available class capacity

- \( L \) – good estimator of the maximum number of requests that can be admitted aiming at a VM availability \( \alpha \)
III. Prediction based admission control

- **Quota** at epoch $i$ for class $k$ achieving a VM availability of $\alpha_k$

$$l_{ik} = \frac{\mathcal{F}(b(E_i), k, h)}{\alpha_k}$$  \hspace{1cm} (8)

- $\mathcal{F}(.)$ predictive function estimates available class capacity in the following $h$ time units
- $b(E_i)$ begin of epoch $E_i$
- $k$ class
- $h$ time units
IV. Evaluation Methodology

Admission Control Heuristics

- each one uses different forecasting method for quota
  
  ➢ Pred-cmean
    - min of averages of last hour, last day – 24 hours and seasonal day
  
  ➢ Pred-ets
    - weighted average of input values – Error correction, Trend and Seasonal day
  
  ➢ Greedy-quota
    - uses current available capacity
  
  ➢ No-adm-control
IV. Evaluation Methodology

- **Revenue per unit time**

  - obtained by running j-th VM request of class k
  - proportional to the VM requested capacity and availability SLO

  \[ g_{jk} = S_{jk} \cdot \alpha_k^{min}. \]
IV. Evaluation Methodology

- **Penalty**
  - For violating the VM availability SLO
  - proportional to VM requested capacity and availability SLO and the total time VM was running or pending

\[
\begin{array}{ll}
p_{jk} = \begin{cases} 
0, & \text{if } \alpha_{jk} \geq \alpha_{k}^{min} \\
S_{jk} \cdot \alpha_{k}^{min} \cdot (r(V_{jk}) - s(V_{jk})), & \text{otherwise.} 
\end{cases}
\end{array}
\] (11)
IV. Evaluation Methodology

Metrics

- **Admission rate**
  percentage of requests admitted

- **SLO fulfilment**
  percentage of admitted VM requests that has VM availability at least VM availability defined by SLO

- **Mean cloud utilization**
  percentage of the available cloud capacity that is allocated for VMs

- **Profit efficiency**
  profit obtained for heuristic h normalized by the highest profit of all heuristics for the same scenario
V. Results

- Dataset
  - Google workload traces – from a cluster of 12k physical machines over 29 days
V. Results

- **Scenario 1: Base Scenario**
  - Capacity size factor = 1
  - 3 classes of requests
    - Prod
    - Batch
    - Free
  - VM availability SLOs
    - prod = 100%
    - batch = 90%
    - free = 50%
V. Results

Base Scenario

- SLO fulfilments are higher for predictive heuristics as they reject requests to meet SLOs of admitted requests and also make better predictions than greedy.
- Admission rates are lower for predictive heuristics as they reject requests.
V. Results

- **Scenario 2: Capacity sensitivity analysis**
  - Capacity size factor = \{0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3\}
  - 3 classes of requests
    - Prod
    - Batch
    - Free
  - VM availability SLOs
    - prod = 100%
    - batch = 90%
    - free = 50%
V. Results

Capacity Planning sensitivity analysis

➢ **SLO fulfilment**
  - Lower for no-adm-ctrl and greedy quota, increases as capacity increases
  - For predictive heuristics, it’s not significantly affected

➢ **Admission rate**
  - 100% for no-adm-ctrl
  - For others it increases as the capacity increases
V. Results

Capacity Planning sensitivity analysis – mean utilization

- Cloud utilization decreases as the cloud capacity increases.
- Predictive heuristics has lower utilizations than others due to the lower admission rates needed to achieve high SLO fulfillments.
V. Results

Capacity Planning sensitivity analysis – profit efficiency

- Profits for no-adm-ctrl are negative for capacity factors < 1.1
- All heuristics had high profits when resources are overprovisioned
- Predictive heuristic had the best profits overall
V. Results

Scenario 3: Availability SLOs Sensitivity Analysis

- Capacity size factor = 1
- 3 classes of requests
  - Prod
  - Batch
  - Free
- VM availability SLOs
  - prod: 100%
  - (batch, free):
    - very-low (50%, 10%)
    - low (70%, 30%)
    - middle (90%, 50%)
    - high (99%, 70%)
    - very-high (99.9%, 90%)
V. Results

Availability SLOs Sensitivity Analysis

- **SLO fulfilment**
  - Lower for no-adm-ctrl
  - Increases as the SLO strength increases for quota-based heuristics because they define more conservative quotas, which results in higher SLO fulfillments.

- **Admission rate**
  - Decreases for predictive heuristics when SLOs increase, because more rejections are required to meet SLO objectives.
V. Results

Availability SLOs Sensitivity Analysis – Cloud utilization

- For no-adm-control cloud utilization is always 100%
- For quota based heuristics, utilization decreases when the SLO strength increases because of lower admission rates.

Figure 7. Mean cloud utilization when varying the SLO strength.
VI. Conclusions

1. Admission control mechanisms are necessary to fulfill availability SLOs for different classes when the cloud capacity is not overprovisioned.

2. Prediction-based heuristics have consistently high SLO fulfillments and the highest profits compared to that using other heuristics for all the scenarios analyzed.

3. Prediction-based heuristics were not significantly affected by different capacity planning and SLO decisions.

4. Instead of offering all classes with high SLO targets, it is better to offer a wide range of SLOs in order to achieve high utilization.
Questions and Discussion