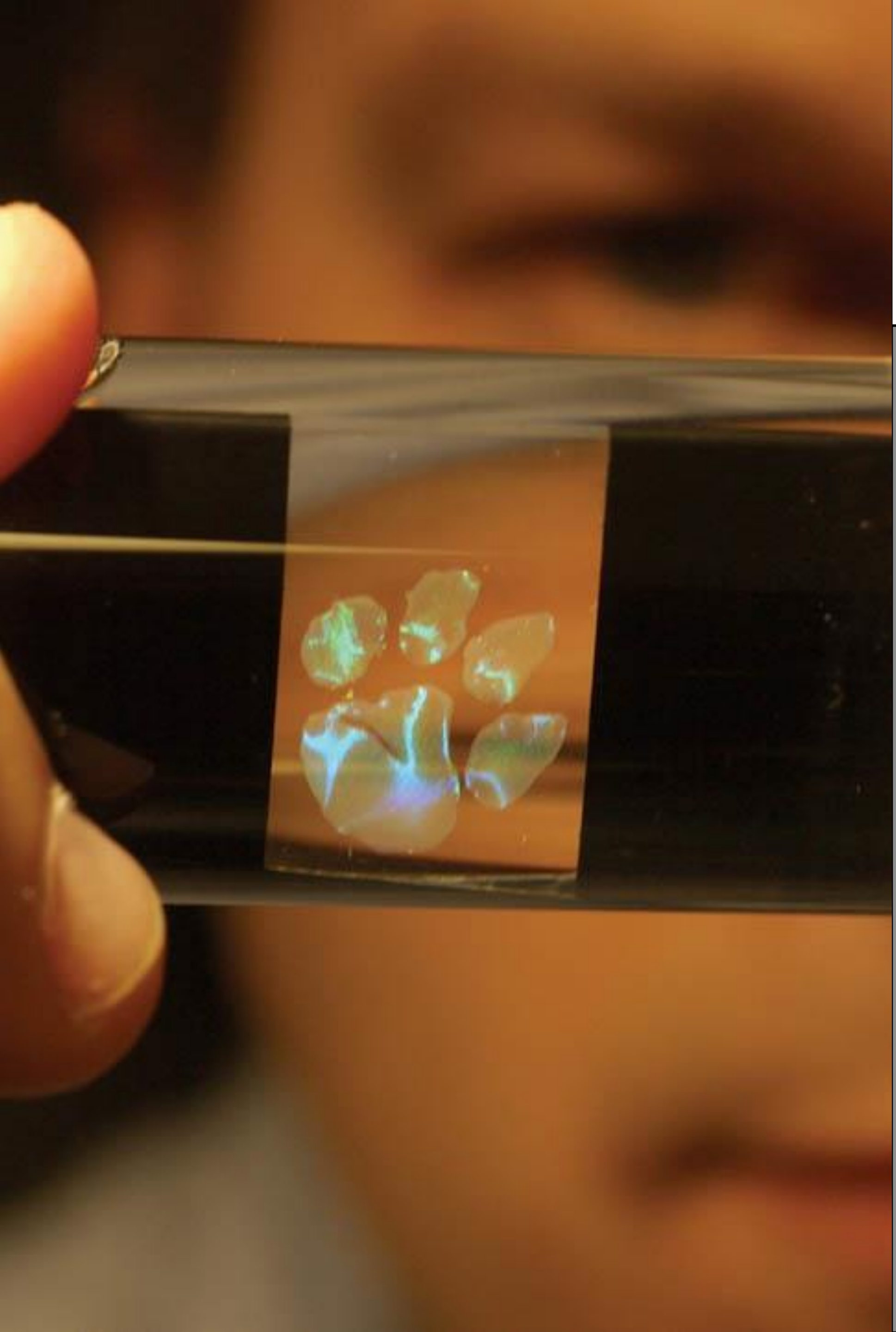


CLEMSON UNIVERSITY

CORP: Cooperative Opportunistic Resource Provisioning for Short-Lived Jobs in Cloud Systems

Jinwei Liu, Haiying Shen and Liuhua Chen

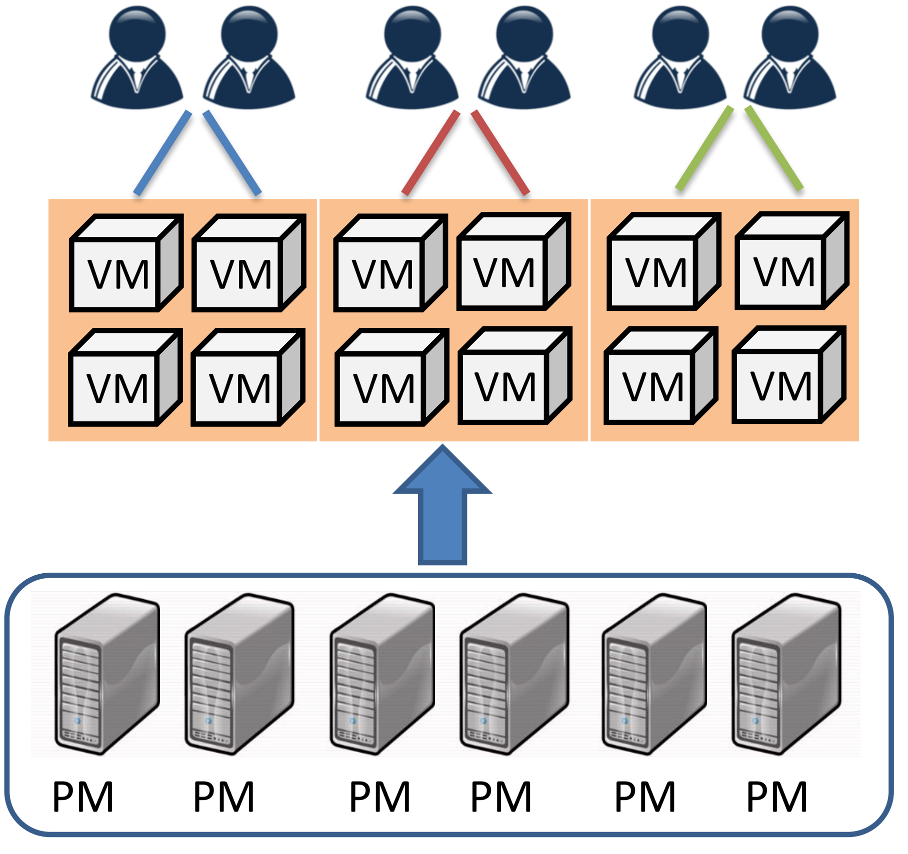
Dept. of Electrical and Computer Engineering
Clemson University, SC, USA

Outline

- Introduction
- Cooperative Opportunistic Resource Provisioning (CORP) Problem
- Design of CORP
- Performance Evaluation
- Conclusions

Introduction

- Resource allocation in cloud computing



Introduction (cont.)

- Resource allocation options
  - Reservation-based resource allocation
    - Amazon Reserved Instance

amazon web services

Amazon EC2 Reserved Instances

  - Demand-based resource allocation
    - Amazon On-demand Instances

Amazon Web Services EC2
On-Demand Instances

  - Opportunistic-based resource allocation
    - Amazon Spot Instances, [Carvalho, SoCC'14]

AMAZON EC2 SPOT INSTANCES

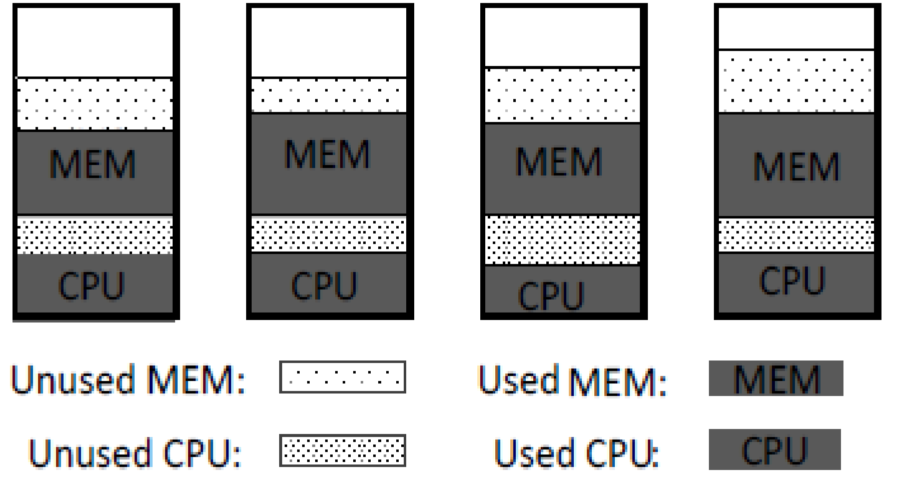
AWS.AMAZON.COM/EC2/SPOT

Motivation

- Limitation of the three resource allocation options
  - Reservation-based resource allocation
    - Much more resources are wasted
    - Higher cost
  - Demand-based resource allocation
    - Resources cannot be fully utilized all the time
    - High cost
  - Opportunistic-based resource allocation
    - Either have no SLOs or cannot improve resource utilization while ensuring SLO availability for short-lived jobs

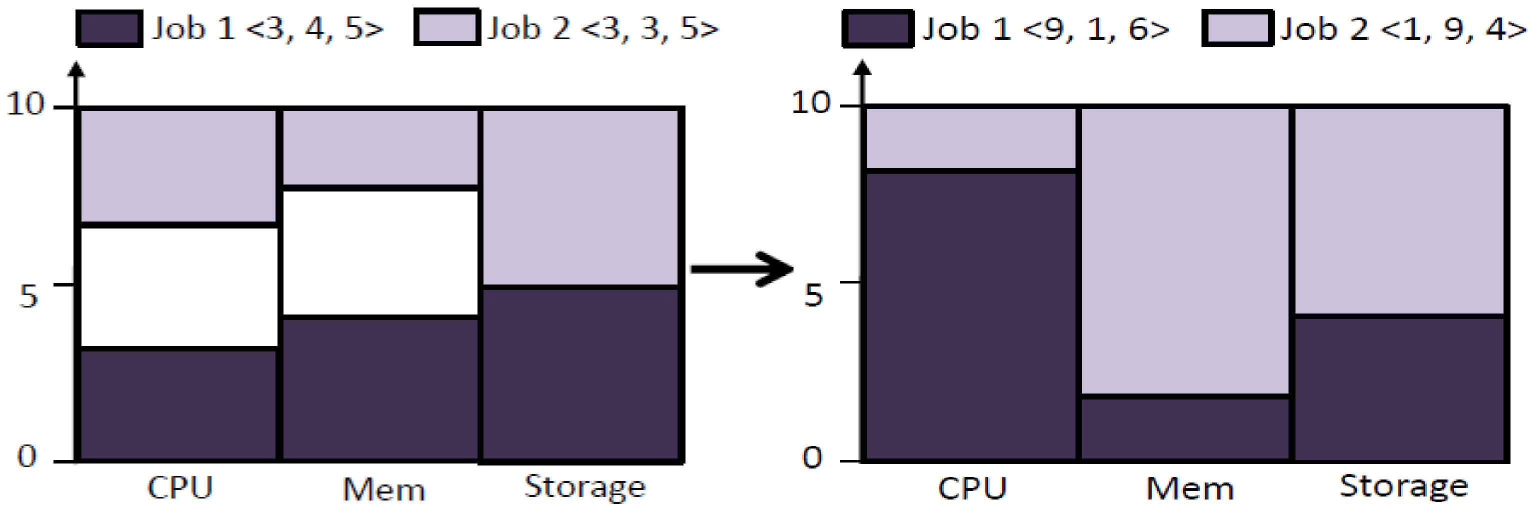
Motivation (cont.)

- Resource wastage
  - Resource wastage in terms of unused resource



Motivation (cont.)

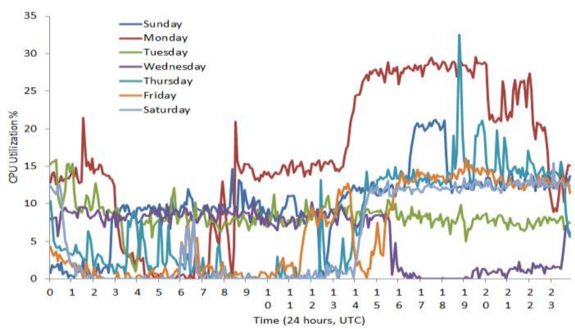
- Resource wastage caused by resource fragmentation



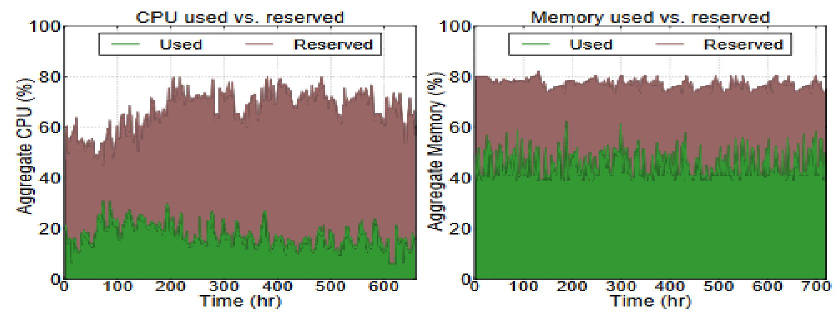
Without avoiding fragmentation With avoiding fragmentation

Existing Cloud Systems

- Resource wastage & low resource utilization
  - Amazon EC2 servers have low resource utilization
  - Production cluster at Twitter has low resource utilization



Amazon EC2 servers



Production cluster at Twitter [46]

- Goal: design an efficient resource allocation scheme that can increase resource utilization while ensuring SLO availability

[46] C. Delimitrou and C. Kozyrakis. Quasar: Resource-Efficient and QoS-Aware Cluster Management. In Proc. of ASPLOS, 2014.

Outline

- Introduction
- Cooperative Opportunistic Resource Provisioning (CORP) Problem
- Design of CORP
- Performance Evaluation
- Conclusions

Cooperative Opportunistic Resource Provisioning (CORP) Problem

- System model
  - N_v: # of VMs in the system
  - l: # of resource types in the system
  - n_t: # jobs submitted at time t
  - $r_{ij,t}$: amount of type j resource allocated to J_i
  - $d_{ij,t}$: Job J_i demand on type j resource
  - C_{ij}: capacity for type j resource of VM v_i

- Problem statement
  - Problem: Given a certain amount of resources, resource demands of each job, resource capacity constrains of VMs, how to allocate the VM resources to jobs to achieve high resource utilization while avoiding SLO violations as much as possible?

Challenges of Resource Provisioning

- Challenges of resource provisioning for short-lived jobs
  - How to accurately predict the amount of unused resource of short-lived jobs with resource fluctuations
  - How to reduce resource fragmentation in multi-resource allocation
  - How to fully utilize the resource without compromising SLO availability

Outline

- Introduction
- Cooperative Opportunistic Resource Provisioning (CORP) Problem
- Design of CORP
- Performance Evaluation
- Conclusions

CLEMSON UNIVERSITY

Design of CORP

- Key idea: Reallocate the unused resource to the new arriving jobs with a certain probability
  - Predict the amount of unused resource using deep learning
  - Use Hidden Markov Model (HMM) to predict the fluctuations of the amount of unused resource for error correction
  - Use job packing to reduce resource fragmentation
  - Use probabilistic-based resource preemption

Predict Unused Resource

- Prediction process
  - Predicting the amount of unused resource using deep learning with HMM
    - Feed-forward evaluation:

$$g_i(d) = F((\Sigma_{j=1}^{c} w_{ij}(d-1,d) \cdot g_j(d-1)) + e_i)$$

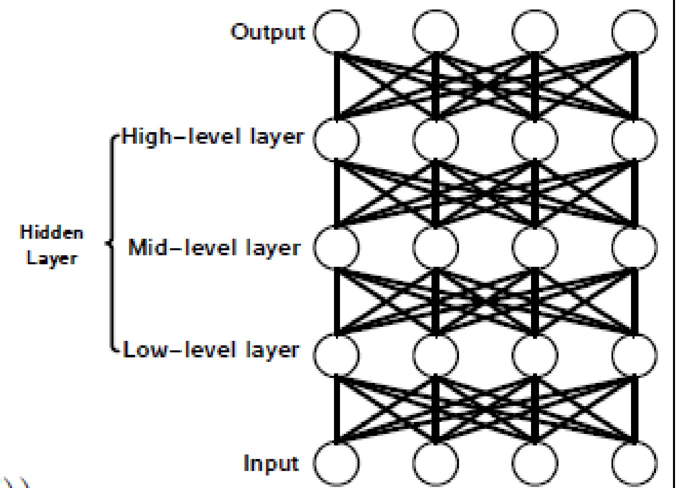
    - Back-propagation

$$E_i(d_h) = (t_i(d_h) - g_i(d_h)) \cdot F'(g_i(d_h))$$

$$E_i(d) = (\sum_{j=1}^{m} E_j(d+1) \cdot w_{ji}(d,d+1)) \cdot F'(g_i(d))$$

    - Weight updates

$$\Delta w_{ij}(d-1,d) = \mu \cdot E_i(d) \cdot g_j(d-1),\ \forall j = 1,\ldots,c$$



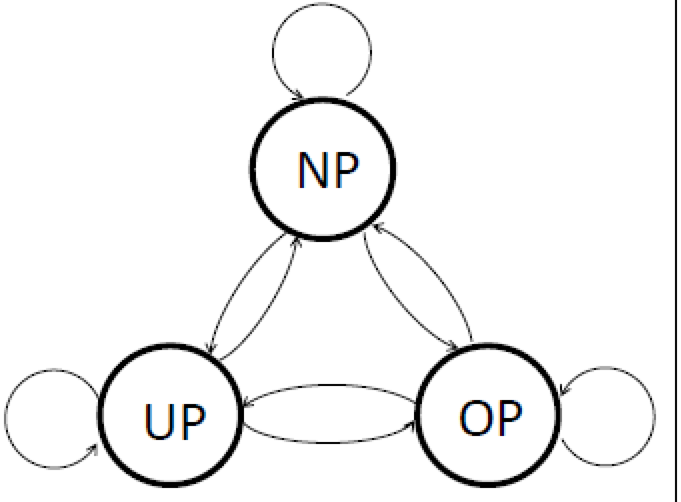
Deep neural networks

Predict Unused Resource (cont.)

- Prediction process
  - Predicting fluctuations of the amount of unused resource using HMM
    - Why predict fluctuations of unused resource?
      - Short-lived jobs usually do not exhibit certain resource utilization, which results in the fluctuations of the amount of unused resource
    - Solution: Use HMM model to predict the fluctuations of the amount of unused resource

Predict Unused Resource (cont.)

- Predicting fluctuations of the amount of unused resource using HMM
  - HMM model
    - Three hidden states:
      over-provisioning (OP),
      normal-provisioning (NP),
      under-provisioning (UP)
    - Three observation symbols:
      peak (P), center (C), valley (V)



Predict Unused Resource (cont.)

- Error correction based on observation symbols
  - Observation symbols:
    - Peak: $\hat{u}_{t+L} = \hat{r}_{j1} + min(h_{cpu} - m_{cpu}, m_{cpu} - l_{cpu})$
    - Valley: $\hat{u}_{t+L} = \hat{r}_{j1} - min(h_{cpu} - m_{cpu}, m_{cpu} - l_{cpu})$
      - h_{cpu}: the highest amount of unused resource
      - l_{cpu}: the lowest amount of unused resource
      - m_{cpu}: the average amount of unused resource
  - Rationale: 1) $h_{cpu} - m_{cpu}$ (or $m_{cpu} - l_{cpu}$) indicates the deviation between the amount of unused resource in peak (or valley) and the Ave. of the amount of unused resource; 2) min is more conservative for ensuring sufficient resource being able to allocate to jobs

Probabilistic-based Resource Preemption

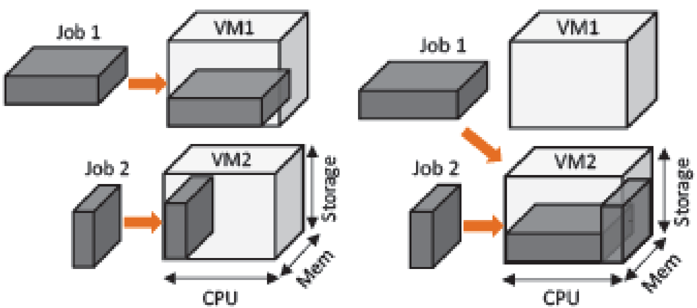
- Two states
  - Locked: unused resource cannot be preempted (i.e., reallocated)
  - Unlocked: unused resource can be preempted

$$\Pr(0 \leq \delta_{t+L} \leq \varepsilon) \geq P_{th} \quad (1)$$

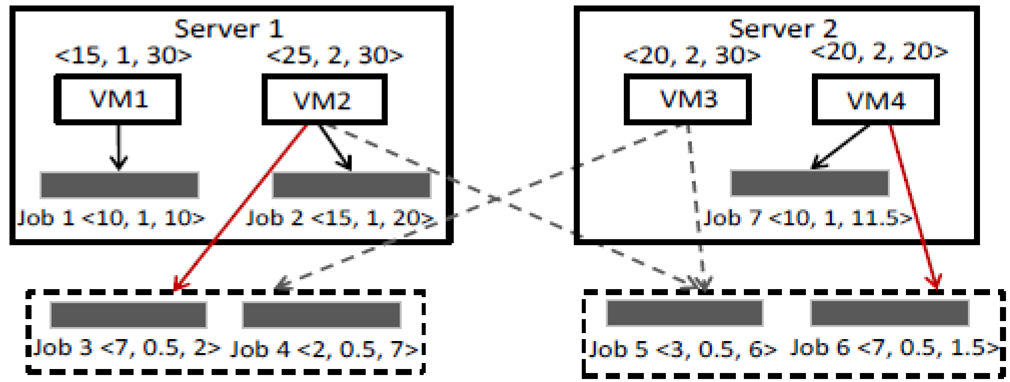
  - ε: pre-specified prediction error tolerance
  - P_{th}: pre-defined probability threshold

Job packing

- Leverage complementarity of jobs' requirements on different resource types



(a) Without packing **(b) With packing**



An example of allocating resources to jobs using packing strategy

$$DV(i,j) = \sum_{k=1}^{l}\left(\left(d_{jk} - \frac{d_{jk}+d_{ik}}{2}\right)^2 + \left(d_{ik} - \frac{d_{jk}+d_{ik}}{2}\right)^2\right) \quad (2)$$

$$volume_j = \sum_{k=1}^{l} \hat{r}_{jk}/C'_k \quad (3)$$

Outline

- Introduction
- Cooperative Opportunistic Resource Provisioning (CORP) Problem
- Design of CORP
- Performance Evaluation
- Conclusions

Performance Evaluation

- Methods for comparison
  - RCCR [4]: Opportunistic-based resource allocation based on time series forecasting

    [4] M. Carvalho, W. Cirne, F. Brasileiro, and J. Wilkes. Long-term slos for reclaimed cloud computing resources. In *Proc. of SoCC*, Seattle, 2014.

  - CloudScale [26]: Demand-based resource allocation based on FFT (fast Fourier transform) for prediction

    [26] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes. Cloudscale: Elastic resource scaling for multi-tenant cloud systems. In *Proc. of SoCC*, Cascais, Oct. 2011.

  - DRA [36]: Demand-based resource allocation based on monitoring

    [36] G. Shanmuganathan, A. Gulati, and P. Varman. Defragmenting the cloud using demand-based resource allocation. In *Proc. of SIGMETRICS*, pages 67-80, Pittsburgh, June 2013.

Experiment Setup

- Trace-driven experiments on a real cluster Palmetto & Amazon EC2
  - Nodes deployment
    - 50 Nodes in Palmetto [34]
    - 30 Nodes in Amazon EC2 [35]
  - Trace from Google [39]
  - CPU & Mem consumption based on Google trace [39]
  - Bandwidth consumption for each task: 0.02 MB/s [40]

[34] Palmetto cluster. http://citi.clemson.edu/palmetto/.
[35] Amazon EC2. http://aws.amazon.com/ec2.
[39] Google trace. https://code.google.com/p/googleclusterdata/.
[40] A. L. Shimpi. The ssd anthology: Understanding SSDs and new drives from OCZ. Feb. 2014. http://dx.doi.org/10.1007/s11276-005-6612-9.

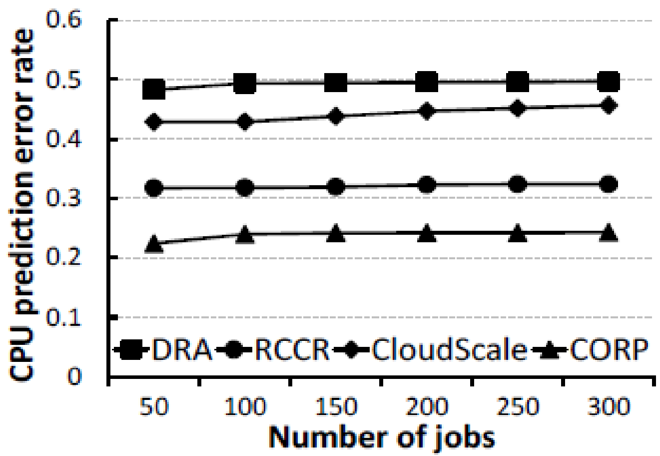
Experiment Setup (cont.)

- Parameter settings

Parameter	Meaning	Setting	Parameter	Meaning	Setting
N_p	# of servers	30-50	h	# of layers in DNN	4
N_v	# of VMs	100-400	N_n	# of units per layer	50
|J|	# of jobs	50-300	H	# of states in HMM	3
l	# of resc. types	3	θ	Significance level	5%-30%
P_{th}	Prob. threshold	0.95	η	Confidence level	50%-90%

Evaluation of CORP

- Experimental results on the cluster
  - Prediction error rate

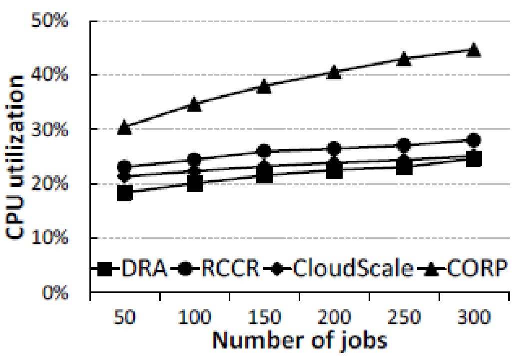


(a) Prediction error rate

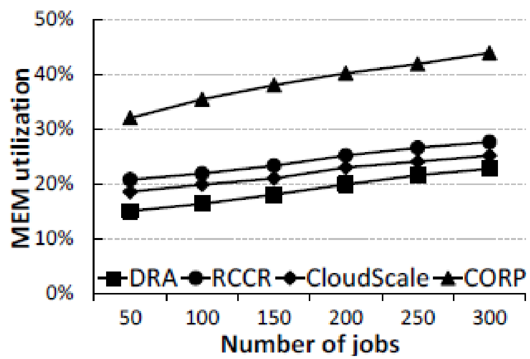
  - Prediction error rate: CORP < RCCR < CloudScale < DRA
  - Reason: advantages of deep learning; CORP considers fluctuations, uses HMM to correct prediction errors

Evaluation of CORP

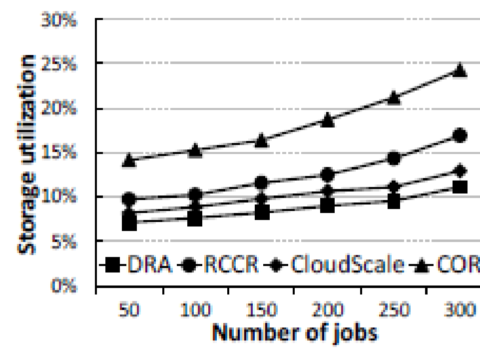
- Experimental results on the cluster
  - Resource utilization



(a) CPU



(b) Memory

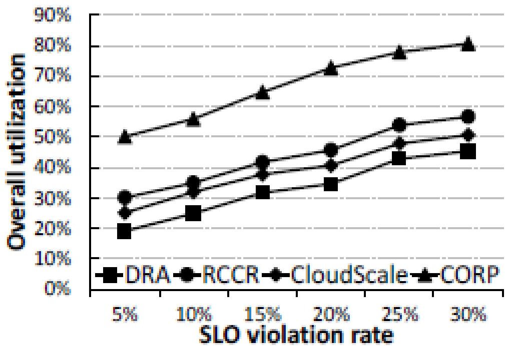


(b) Storage

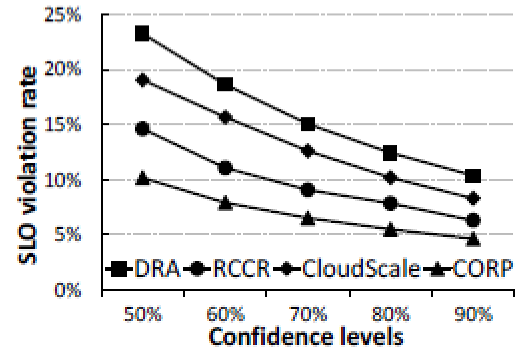
  - Utilization of CPU, MEM and storage: CORP > RCCR > CloudScale > DRA
  - Reason: opportunistic-based resource allocation; use deep learning & HMM to accurately predict the unused resource and avoid over-provisioning; use job packing to reduce resource fragmentation

Evaluation of DSP

- Experimental results on the cluster
  - Overall resource utilization & SLO violation rate



(a) Resource utilization vs. SLO violation rate

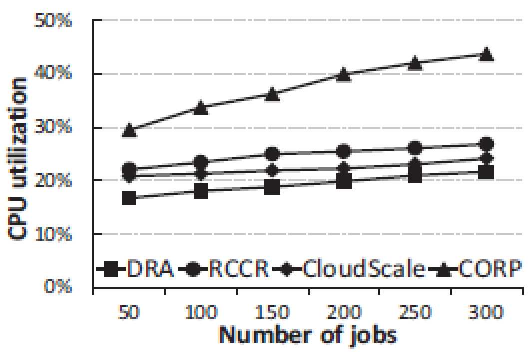


(b) SLO violation rate vs. confidence intervals

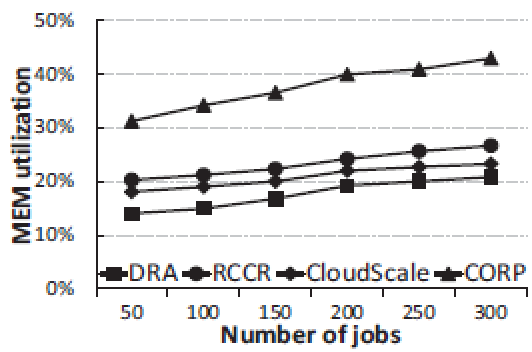
  - Observation: resource utilization increases as SLO violation rate increases; given an SLO violation rate: CORP > RCCR > CloudScale > DRA
  - Reason: the higher the SLO violation rate, the lower the probability of over-provisioning occurring; advantages of deep learning; CORP considers fluctuations, uses HMM to correct prediction errors

Evaluation of CORP

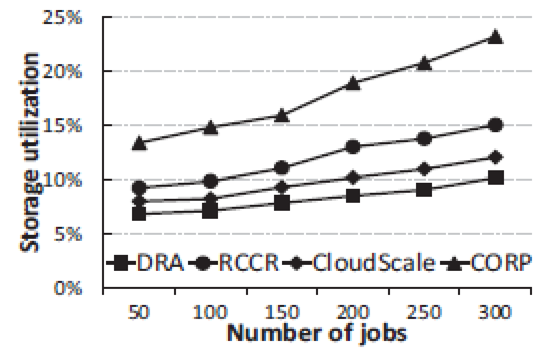
- Experimental results on Amazon EC2
  - Resource utilization



(a) CPU



(b) Memory

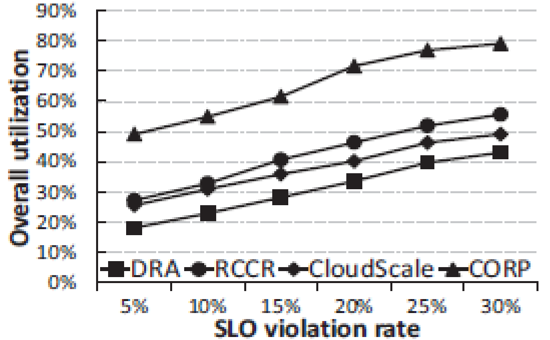


(b) Ave. resource utilization

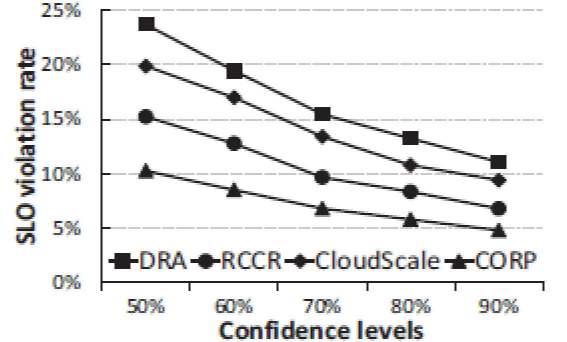
  - Utilization of CPU, MEM and storage: CORP > RCCR > CloudScale > DRA
  - Reason: opportunistic-based resource allocation; use deep learning & HMM to accurately predict the unused resource and avoid over-provisioning; use job packing to reduce resource fragmentation

Evaluation of CORP

- Experimental results on Amazon EC2
  - Overall resource utilization & SLO violation rate



(a) Resource utilization vs. SLO violation rate on Amazon EC2



(b) SLO violation rate vs. confidence intervals on Amazon EC2

– Observation:

a) resource utilization increases as SLO violation rate increases; given an SLO violation rate, overall resource utilization follows CORP > RCCR > CloudScale > DRA

b) SLO violation rate decreases as the confidence levels increases; SLO violation rate follows CORP < RCCR < CloudScale < DRA

Outline

- Introduction
- Cooperative Opportunistic Resource Provisioning (CORP) Problem
- Design of CORP
- Performance Evaluation
- Conclusions

Conclusions

- Our contributions
  - Use deep learning algorithm to predict the amount of unused resource of short-lived jobs
  - Consider the fluctuations of the amount of unused resource and present the HMM model to predict the fluctuations of the amount of unused resource for error correction
  - Present a job packing strategy to reduce the resource fragmentation and fully utilize the resource
  - Extensive experimental results based on a real cluster and Amazon EC2 validate the performance of CORP

- Future work
  - Consider designing a distributed deep learning training system to reduce the computation overhead
  - Consider both short-lived and long-lived jobs and design an efficient resource allocation strategy with high resource utilization
  - The fluctuation of the workloads

Thank you!

Questions & Comments?

Jinwei Liu, PhD

jinweil@clemson.edu

Electrical and Computer Engineering

Clemson University