Autonomic cloud placement of mixed workload: An adaptive bin packing algorithm

Asser N. Tantawi and Malgorzata Steinder

IBM T.J. Watson Research Center
Yorktown Heights, NY, USA
tantawi, steinder@us.ibm.com

ICAC 2019
June 17-19, 2019
Umeå, Sweden

Outline

The Cloud Placement Problem (CPP)
A cloud view

- Cloud infrastructure
  - N physical entities (PE): Physical Machine (PM) in a virtualized environment, data storage device, bare metal machine
  - Communication network, topology hierarchy, etc
- A user request
  - An application (job) to be deployed in the cloud
  - M logical entities (LE) of the application: Virtual Machine (VM), data volume, Container in an OS-level virtualized environment
  - Constraints related to the deployment of the application
- Placement of LEs on PEs
  - Satisfy types of LE and PE
  - Match LE need to available PE resources
  - Meet constraints related to networking and location among LEs
  - Optimize an objective function

Consider a Kubernetes system
Down to Earth

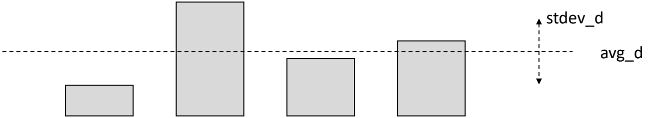
- A cluster of nodes (PEs), with available resource capacities
- A stream of user deployment requests, each comprising a set of pods (LEs)
- A pod requests a specified amount of resources
- The Scheduler assigns a node to a pod (places a pod on a node), given some objectives

The problem

- Pods have multi-dimensional resource demands (CPU, memory, storage, GPU)
  - Set of standard and/or custom sizes
  - Mix changes unpredictably over time
- Cluster of nodes with (heterogeneous) multi-dimensional resource capacities
- Online placement using simplistic, extreme, standard policies (pack, spread) may lead to
  - Resource fragmentation
  - Rejection of large-sized and/or disproportional pods
- Static (optimized) choice of a placement policy for a given workload mix may prove inefficient as the mix changes
- Need an adaptive placement policy

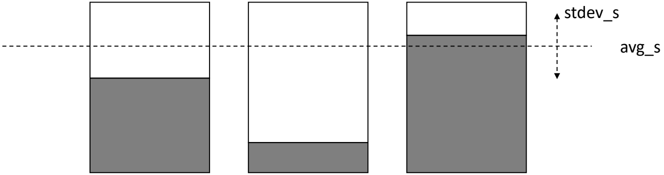
The main idea
in one dimension

Workload:
Resource demand



cov_d = stdev_d / avg_d

System:
Resource availability



Scheduler target:
cov_d = cov_s

cov_s = stdev_s / avg_s

Multidimensional variability

Use a scalar measure, known as the Albert and Zhang multivariate coefficient of variation,

$$\gamma = \sqrt{\frac{\boldsymbol{\mu}^T \, \boldsymbol{\Sigma} \, \boldsymbol{\mu}}{(\boldsymbol{\mu}^T \, \boldsymbol{\mu})^2}}$$

where $\boldsymbol{\mu}$ is the average vector and $\boldsymbol{\Sigma}$ is the covariance matrix.

- Simple in computation complexity (no need to do matrix inversion)
- Captures the correlation structure among the resources
- Preserves the scaling among the resource components based on the average usage

The optimization problem

- Demand variability
  - ν_r and $\upsilon_{r1,r2}$: the average and covariance measures of relative demand over a given time period for resource types $r, r1, r2 \in \{1, 2, \cdots, R\}$.
  - $\boldsymbol{\mu}^{dem}(t) = [\nu_r]$ and $\boldsymbol{\Sigma}^{dem}(t) = [[\upsilon_{r1,r2}]]$: the relative demand average and covariance at time t.
- System variability
  - $\mathbf{u}$(n): the resource utilization matrix after pod is placed on node n.
  - $\boldsymbol{\mu}^{sys}(n)$ and $\boldsymbol{\Sigma}^{sys}(n)$: the (marginal) average and covariance of $\mathbf{u}$(n) across cluster $\mathcal{N}$.
- Problem
  - Given a pod with demand $\mathbf{d}$ at time t and an observed relative demand measure of variability $\gamma^{dem}(t)$.
  - Find an assignment of a node p_n which solves

$$\min_{n} \left(\gamma^{sys}(n) - \gamma^{dem}(t)\right)^2,$$

such that demand constrains are satisfied, $n = 1, 2, \cdots, N$.

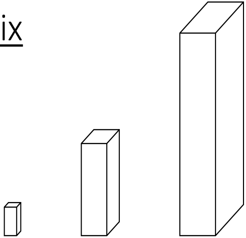
Experimental setup

Cluster

- Nodes = 32
- Capacity
  - CPU = 32
  - MEM = 256
  - GPU = 4

Single pod placement

Pod Workload Mix

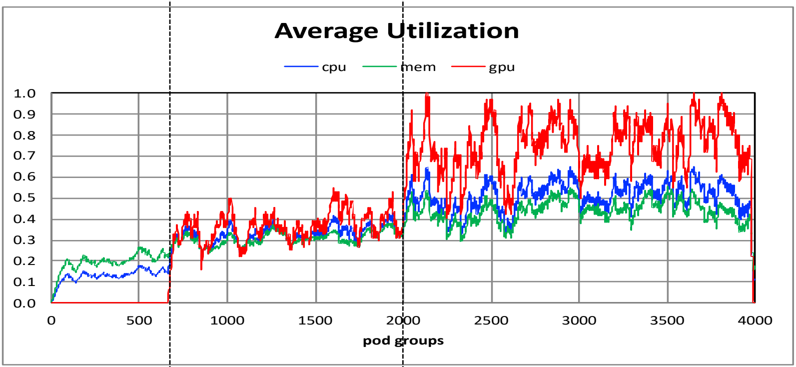


Type	A	B	C
CPU demand	2	8	16
MEM demand	24	32	96
GPU demand	0	2	4
CPU average load	0.15	0.20	0.20

Simulation experiment

Phase: I II III

Pods: A | A + B | A + B + C

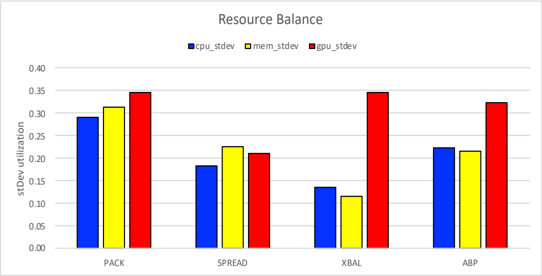


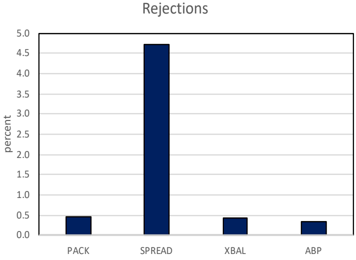
666 2000 4000

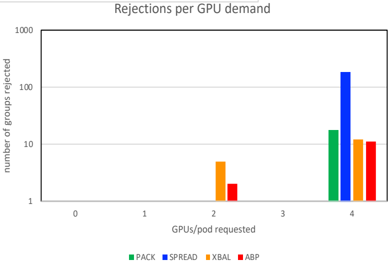
Placement policies

- Extreme, simplistic
  - Pack
  - Spread
- Optimized
  - XBAL
    - selective balancing and un-balancing based on a weight vector
    - $\mathbf{w} = [1, 0, -2]$, i.e. balance on CPU and pack on GPU with twice as much weight
- Adaptive Bin Packing
  - ABP

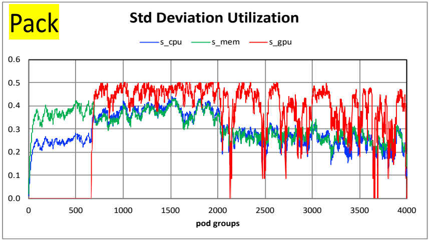
Overall results

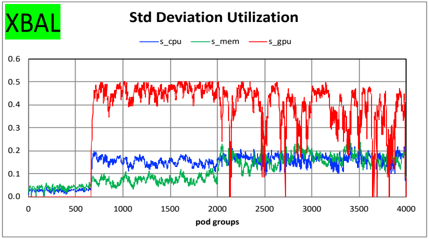


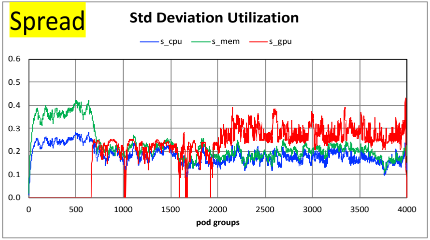


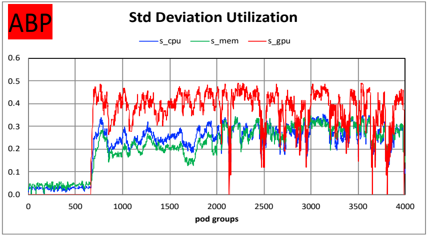


Balance deviation



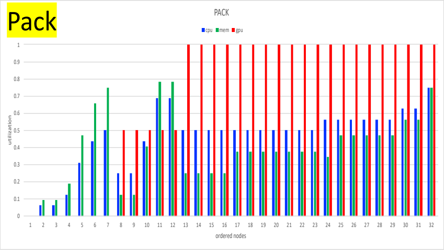


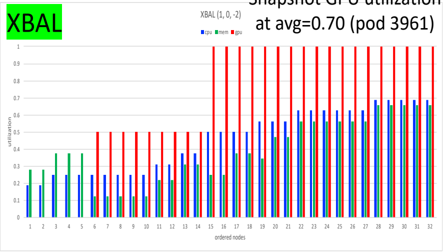


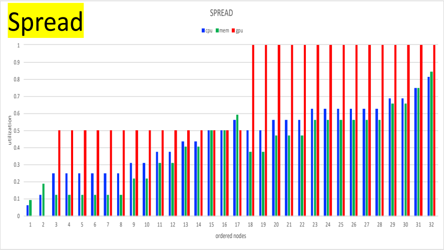


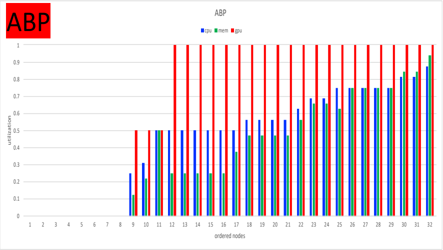
Making room

Snapshot GPU utilization at avg=0.70 (pod 3961)

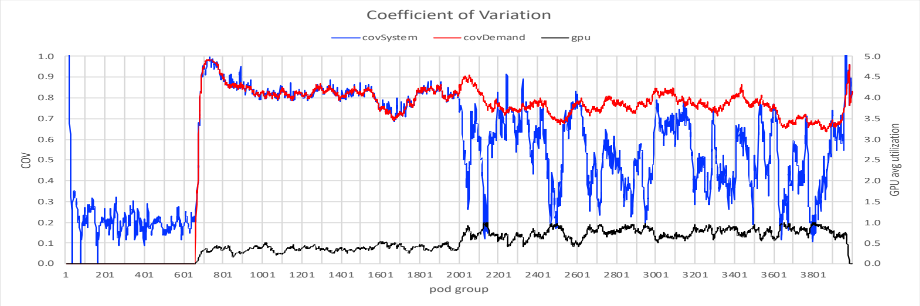


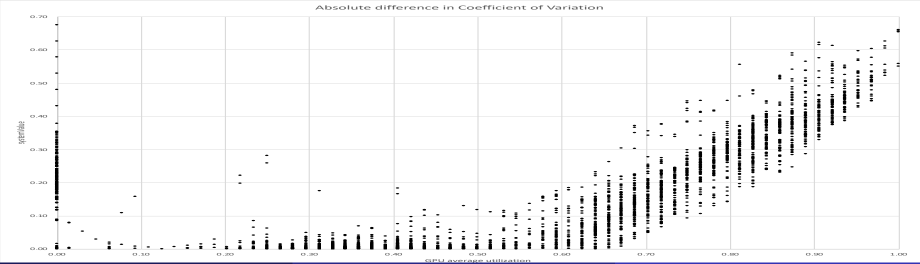






Equalizing variation





Summary

- A novel, autonomic, Adaptive Bin Packing (ABP) algorithm which attempts to equalize measures of variability in the demand and the allocated resources in the cloud, without the need to set any configuration, is introduced.
- ABP learns the nature of the mix by collecting the average vector and covariance matrix of resource demand.
- ABP is compared to simplistic, extreme packing policies (spread and pack) as well an optimized packing policy.
- The behavior of ABP, and its adaptability to the demand mix, is demonstrated through experimental results based on simulations.
- ABP performs close to the optimized policy, yet evolves to an extreme policy as the mix becomes homogeneous.

Backup

Demand variability

	CPU	Memory	GPU
Analytic	0.271	0.198	0.500
Observed	0.300	0.218	0.566

Table: Average observed demand $\boldsymbol{\mu}^{dem}$.

		CPU	Memory	GPU
Analytic	CPU	0.032	0.021	0.073
	Memory	0.021	0.016	0.047
	GPU	0.073	0.047	0.167
Observed	CPU	0.033	0.022	0.073
	Memory	0.022	0.017	0.049
	GPU	0.073	0.049	0.166

Table: Covariance matrices observed demand $\boldsymbol{\Sigma}^{dem}$.

System variability

	CPU	Memory	GPU
PACK	0.459	0.396	0.703
SPREAD	0.465	0.404	0.703
XBAL	0.457	0.393	0.703
ABP	0.455	0.390	0.703

Table: Average resource usage $\boldsymbol{\mu}^{sys}$.

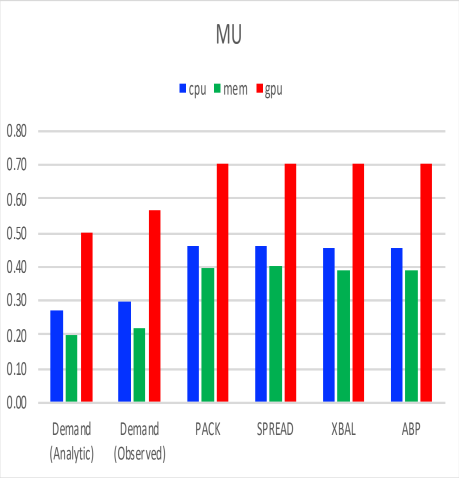
Policy		CPU	Memory	GPU
PACK	CPU	0.035	0.030	0.055
	Memory	0.030	0.043	0.013
	GPU	0.055	0.013	0.175
SPREAD	CPU	0.036	0.038	0.051
	Memory	0.038	0.044	0.047
	GPU	0.051	0.047	0.095
XBAL	CPU	0.033	0.029	0.061
	Memory	0.029	0.033	0.039
	GPU	0.061	0.039	0.143
ABP	CPU	0.090	0.087	0.122
	Memory	0.087	0.094	0.106
	GPU	0.122	0.106	0.191

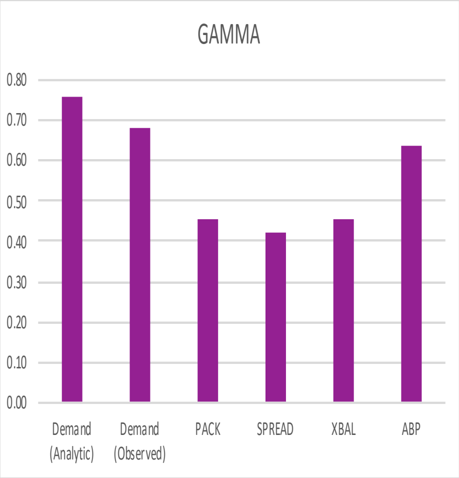
Table: Covariance matrices resource usage $\boldsymbol{\Sigma}^{sys}$.

Analytic	Observed	PACK	SPREAD	XBAL	ABP
0.761	0.679	0.456	0.422	0.457	0.636

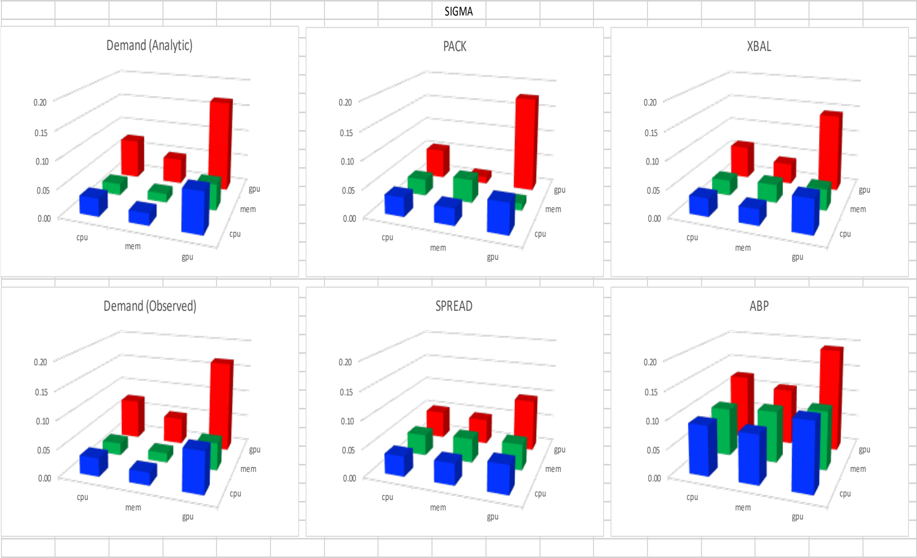
Table: Coefficient of variation γ.

Coefficient of variation





Covariance matrix



Ordered GPU utilization