Autonomic Allocation of Communicating Virtual Machines in Hierarchical Cloud Data Centers

Summarized by Matthew Jablonski

George Mason University

mjablons@gmu.edu

September 14, 2015

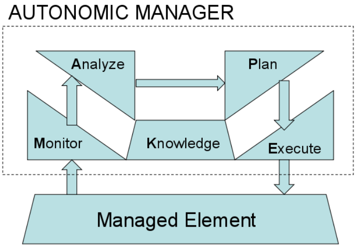
Aldhalaan, A. and Menasce, D.A. (2014)
Autonomic Allocation of Communicating Virtual Machines in Hierarchical Cloud Data Centers
2014 International Conference on Cloud and Autonomic Computing (ICCAC) pp. 161-171.

Overview

Review of Autonomic Computing Goals

Human beings establish high level goals used by autonomic controllers that follow the MAPE-K loop:

1. Analyze data collected by monitors
2. Plan steps for optimization, configuration, failure recovery, and for protection against attacks
3. Execute on these plans



Paper Goals

Contributions of this paper are:

1. Pricing model for cloud resource usage based on how close communication-wise, VMs are allocated by cloud provider (CP)*
2. Formalization of problem of finding an optimal allocation for requested virtual machines (VMs) that maximizes revenue, based on how close the requested VMs are allocated
3. An efficient heuristic algorithm to solve this NP-hard problem

** - Pricing model provides incentives to CP to reduce performance* uncertainties

High Level Assumptions and Knowledge

- CP typically has hierarchically organized networking infrastructure
- Communication cost between two servers varies significantly depending on their relative location in the infrastructure
- Communication latency increases and bandwidth decreases as we move:
  1. Within same server;
  2. Between different servers in same rack;
  3. Between different racks in same cluster;
  4. Between different cluster in same data center;
  5. Between different data centers

Visualization of Hierarchical Infrastructure

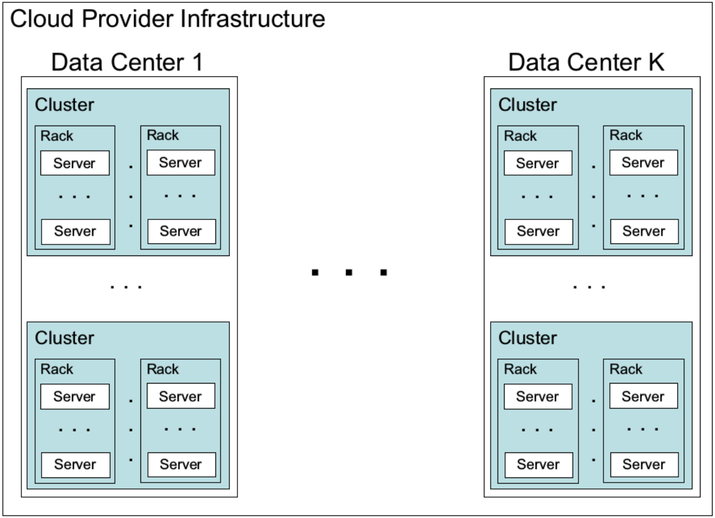


Figure: Infrastructure of a Cloud Provider

Formalization of Problem and Knowledge

- **Wanted:** algorithm to maximize the revenue of a cloud provider (CP) subject to availability and capacity constraints
- **Know:** CP typically has hierarchically organized networking infrastructure (based on previous slide)

Hierarchical Infrastructure Notation

1. CP - cloud provider (assume $CP = 1$)
2. D: # data centers in CP; $(1 \leq d \leq D,\ d \in \mathbb{N})$
3. $C(d)$: # of clusters of racks in d; $(1 \leq c(d) \leq C(d),\ c(d) \in \mathbb{N})$
4. $R(c,d)$: # of racks in cluster $c(d)$; $(1 \leq r(c,d) \leq R(c,d),\ r(c,d) \in \mathbb{N})$
5. $S(r,c,d)$: # of servers in rack $r(c,d)$; $(1 \leq s(r,c,d) \leq S(r,c,d),\ s(r,c,d) \in \mathbb{N})$
6. N: # number of VM types offered by CP; $(1 \leq t \leq N,\ t \in \mathbb{N})$

Revenue Model Goals

Objective: The closer a consumer's VMs are, the better the performance of its application and the higher the price that a consumer will have to pay.

- Revenue functions depend on values of communication strength ($C[i,j]$) between VMs (i and j)
- $C[i,j]$ can be estimated as bytes sent between i and j, and can be obtained by consumer or provided as service by CP

More Notation

- C: $K \times K$ communication strength matrix such that $0 \leq C[i,j] \leq 1$, $C[i,j] = C[j,i]$, $C[i,i] = 1$, $\forall i,j \in \{1,\ldots,K\}$
- $A_i = (s,r,c,d)$: allocation of VM i on server s of rack r of cluster c of data center d
- $r^p_{i,j}$: revenue obtained by the CP when allocating VM i according to allocation A_i and VM j according to allocation A_j to a category p customer - **This is the output of the revenue function**

Types of Co-Location for VMs i and j

Co-location Type (α)	Description
1	Same server $A_i = A_j = (s, r, c, d)$
2	Different servers of the same rack $A_i = (s_i, r, c, d), A_j = (s_j, r, c, d) \mid s_i \neq s_j$
3	Different racks, same cluster $A_i = (s_i, r_i, c, d), A_j = (s_j, r_j, c, d) \mid$ $s_i \neq s_j \wedge r_i \neq r_j$
4	Different clusters, same data center $A_i = (s_i, r_i, c_i, d), A_j = (s_j, r_j, c_j, d) \mid$ $s_i \neq s_j \wedge r_i \neq r_j \wedge c_i \neq c_j$
5	Different data centers $A_i = (s_i, r_i, c_i, d_i), A_j = (s_j, r_j, c_j, d_j) \mid$ $s_i \neq s_j \wedge r_i \neq r_j \wedge c_i \neq c_j \wedge d_i \neq d_j$

Revenue Model Definitions

1. Linear revenue function decreases for α where $1 \leq \alpha \leq 5; \alpha \in \mathbb{N}$

Linear Revenue Function

$$r_{i,j}^{p}(\alpha) = \left[\frac{r_{min}^{p} - r_{max}^{p}}{4} \cdot \alpha + \frac{5r_{max}^{p} - r_{min}^{p}}{4}\right] \times C[i,j]$$

2. Exponential revenue function for α is another option

Exponential Revenue Function

$$r_{i,j}^{p}(\alpha) = \left(\frac{(r_{max}^{p})^{5}}{r_{min}^{p}}\right)^{1/4} \times e^{ln(r_{min}^{p}/r_{max}^{p})\cdot\alpha/4} \times C[i,j]$$

In both cases, note:

- $r_{i,j}^{p}(1) = r_{max}^{p}$ and $r_{i,j}^{p}(5) = r_{min}^{p}$
- $r_{i,j}^{p}(\alpha) = r_{i,j}^{p}(A_i, A_j)$

Optimization Problem for Revenue is NP-Hard

Expressing Optimization

$$\textbf{max } \mathcal{R} = \sum_{p=1}^{P} \sum_{A_i, A_j, i<j; i,j \in [1,\ldots,K]} r_{i,j}^{p}(A_i, A_j)$$

s.t. (server capacity constraint)

$$c_h = C_h - \sum_{t=1}^{N} n_{t,h} \cdot d_t \ \forall h$$

Notes and Even More Notation:

- c_h: current available capacity of server $h = s(r, c, d)$
- C_h: nominal capacity of server h in compute units
- $n_{t,h}$ depends on VM allocations; number of VMs of type t allocated to server h
- d_t: capacity needed to instantiate and operate a VM of type t on a server measured in compute units
- Number of possible allocations is the order of H^K; H: total number of servers in cloud infrastructure; K: number of VMs requested by consumer

Heuristic Algorithms

Overview of Efficient Heuristic Approach

- **Basic VM Allocation Heuristic (BVAH):** Does not deallocate any already allocated VM to find near-optimal placement for requested VMs
- **Advanced VM Allocation Heuristic (AVAH):** Uses BVAH and considers the possibility of deallocating some recently allocated VMs, allocating VMs of new request, and reallocating the deallocated VMs
- **NoComm:** Allocates VMs equivalent to BVAH and AVAH methods, but does not take into account value of communication strength index

BVAH High Level Steps

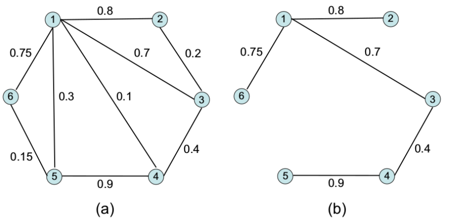
BVAH Steps:

1. Build graph where vertices are the VMs and the edges are non-zero communicate strength indices *
2. Build max spanning tree for the graph (e.g., using Kruskal w/ negative values for edge labels) *
3. Sort edges of max spanning tree in descending order of communication strength *
4. Allocation loop; $k \leftarrow 1$
5. If $k = K$ then Stop else let $\mathcal{L}(k) = (v, w)$
6. If VMs v and w have been allocated, go to step 10
7. If VM v has been allocated but VM w has not, then AllocateCloseTo(w, v), go to step 10 **
8. If VM w has been allocated but VM v has not, then AllocateCloseTo(v, w), go to step 10 **
9. If VMs v and w have not been allocated then CoAllocate(w, v) **
10. $k \leftarrow k + 1$, go to step 5

** - Low level details for each step are in the paper*

*** - Definitions for* CoAllocate *and* AllocateCloseTo *in backup slides*

BVAH Algorithm Example Run



Example of the Operation of the BVAH Algorithm

	Data Center							
	Cluster 1				Cluster 2			
	Rack 1		Rack 2		Rack 1		Rack 2	
	S1	S2	S1	S2	S1	S2	S1	S2
-	1	0	2	1	0	1	1	1
(a)			4,5					
-	1	0	0	1	0	1	1	1
(b)			4,5				1	2
-	1	0	0	1	0	1	0	0
(c)			4,5			6	1	2
-	1	0	0	1	0	0	0	0
(d)	3		4,5			6	1	2
-	0	0	0	1	0	0	0	0

Corresponding Allocation Table

AVAH Steps

Definitions for AVAH

- β: new allocation request that arrived
- $\mathcal{D}$: set consisting of M most recent allocation requests and corresponding revenus obotained by these allocations

1. Allocate VMs in β using BVAH and compute total revenue $\mathcal{R}$
2. For each $\beta_d \in \mathcal{D}$ do
   - 2.1 Let $\mathcal{R}_d$ be revenue from β_d. Deallocate VMs in β_d
   - 2.2 Allocate VMs in β using BVAH and obtain $\mathcal{R}^{new}$
   - 2.3 Allocate VMs in β_d using BVAH and obtain $\mathcal{R}_d^{new}$
   - 2.4 If $(\mathcal{R}_d^{new} + \mathcal{R}^{new}) > (\mathcal{R}_d + \mathcal{R})$ then to go step 3
   - 2.5 Deallocate β_d; Deallocate β; Allocate β_d and β in this order using BVAH
3. Update $\mathcal{D}$ by removing the least recent request and adding the request β to $\mathcal{D}$

Benefit to this algorithm for consumer and/or CP?

NoComm Definition

Create two variants to BVAH and AVAH where $C[i,j] = 1 \; \forall i,j$

- BVAH $\rightarrow$ B-NoComm
- AVAH $\rightarrow$ A-NoComm

Both are first fit strategies based on ignoring the values of the communication strength index for allocation purposes, only.

$r_{i,j}^{p}$ still calculated using original values of $C[i,j]$

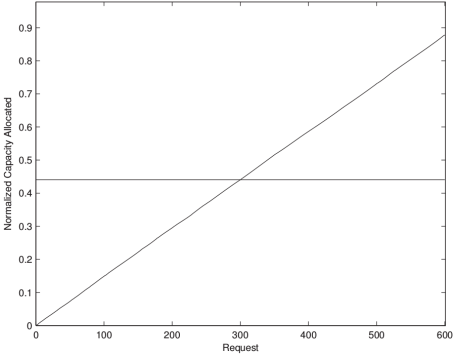
Benefits: Simply considers CP is organized hierarchically and has different communication costs depending on location of VMs in infrastructure and considers capacity constraints

Experiment Setup

- Randomly generated 30 workloads of 600 requests each for total of 18000 requests
- C, VM types, and customer class for each request are randomly generated
- Same workloads used against all VM allocation strategies previously discussed
- Computed average per-request revenue and cumulative workload, along with 95% confidence intervals for all 30 workloads and requests

Parameter	Value
D	2
$C(d)$	10 for all data centers
$R(c,d)$	4 for all clusters
$S(r,c,d)$	20 for all racks
N	3
P	3
d_t	1, 2, and 4
K	10
$C_{s(r,c,d)}$	10 for all servers
M	20
$r^p\max = 5.5, 6.7, \text{ and } 7.9$	for $p = 1,2,3$
$r^p\min = 1, 2, \text{ and } 3$	for $p = 1,2,3$

Parameter values for experiment

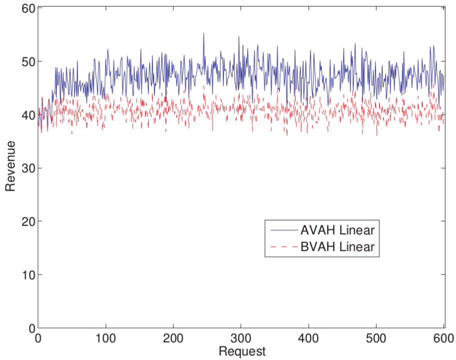


Normalized allocated capacity over time

Workload generated so $p \leq 0.9$

Strategy	Accumulated Revenue	Avg. Request Revenue	UpperOPT
Linear Revenue Model			
BVAH	12,227 ± 103	40.7 ± 0.14	43.6
AVAH	13,491 ± 114	47.1 ± 0.18	51.9
B-NoComm	9,741 ± 82	32.5 ± 0.13	-
A-NoComm	9,079 ± 77	31.1 ± 0.12	-
Exponential Revenue Model			
BVAH	12,195 ± 102	40.6 ± 0.11	43.6
AVAH	12,336 ± 104	41.8 ± 0.11	51.1
B-NoComm	7,867 ± 102	26.2 ± 0.11	-
A-NoComm	9,074 ± 77	30.9 ± 0.12	-

BVAH v. AVAH at linear revenue

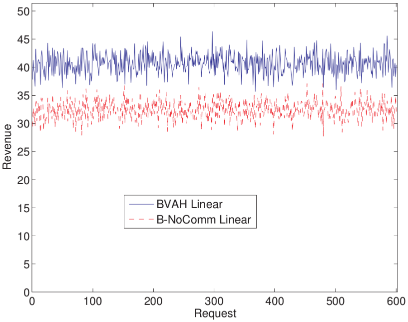


$$\text{BVAH} = 40.7 \pm 0.14$$
$$\text{AVAH} = 47.1 \pm 0.18$$

AVAH has 17% higher revenue at 95% confidence

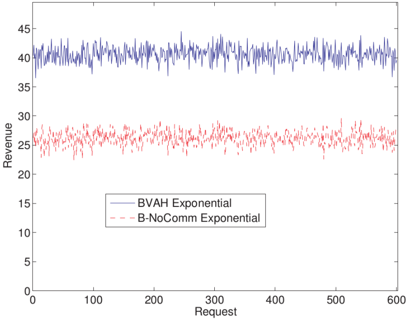
Experimental Results Plotted - BVAH v. B-NoComm

BVAH v. B-NoComm at linear revenue



BVAH $= 40.7 \pm 0.14$
B-NoComm $= 32.5 \pm 0.13$
BVAH has 25% higher revenue at 95% confidence

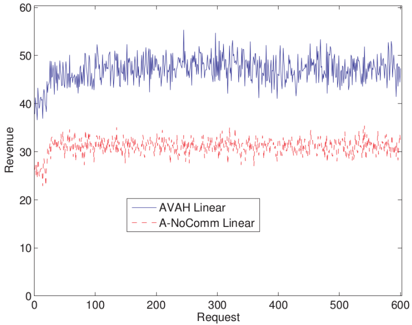
BVAH v. B-NoComm at exponential revenue



BVAH $= 40.6 \pm 0.11$
B-NoComm $= 26.2 \pm 0.11$
BVAH has 55% higher revenue at 95% confidence

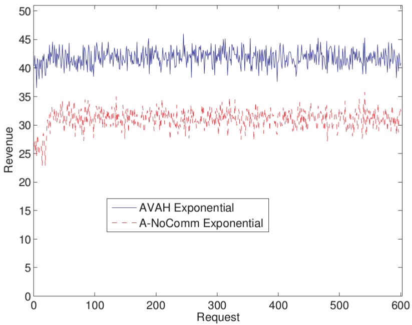
Experimental Results Plotted - AVAH v. A-NoComm

AVAH v. A-NoComm at linear revenue



AVAH $= 47.1 \pm 0.18$
A-NoComm $= 31.1 \pm 0.12$
AVAH has 51% higher revenue at 95% confidence

AVAH v. A-NoComm at exponential revenue



AVAH $= 41.8 \pm 0.12$
A-NoComm $= 30.9 \pm 0.12$
AVAH has 36% higher revenue at 95% confidence

Discussion and Questions (20-25 Minutes)

Possible topics but I'm willing to talk about anything:

- Critiques on paper?
- Thoughts on heuristical approaches?
- What may or may not have been taken into account?
- Could AVAH be further optimized?
- Why does AVAH loop through all $\beta_d \in \mathcal{D}$ instead of just through cases where $\beta \cap \beta_d \neq \emptyset$?
- Thoughts on benefits of these models (hint: Look in "Concluding Remarks")?
- Why is optimization problem NP-hard but not NP-complete? (Yay CS 600!)
- Align with real world revenue models?
- Any experience with paying for real world cloud services?
- What real world services could benefit from this model?
- Avenues of attack with insider knowledge of CP implementation? (My favorite ?)

Backup Slides

BVAH/AVAH CoAllocate Algorithm

Algorithm 1 CoAllocate Algorithm

`CoAllocate` (VM v, VM w);
/* allocate VMs v and w as close as possible of each other */
if $\exists$ server s such that $c_s \geq d_v + d_w$ **then**
allocate v and w on s
else
if $\exists$ a rack with servers s_1 and s_2 s.t. $c_{s_1} \geq d_v$ and $c_{s_2} \geq d_W$ **then**
allocate v on s_1 and w on s_2
else
if $\exists$ a cluster with servers s_1 and s_2 s.t. $c_{s_1} \geq d_v$ and $c_{s_2} \geq d_w$ **then**
allocate v on s_1 and w on s_2
else
if $\exists$ a datacenter with servers s_1 and s_2 s.t. $c_{s_1} \geq d_v$ and $c_{s_2} \geq d_w$ **then**
allocate v on s_1 and w on s_2
else
if $\exists$ datacenters d_1, d_2 with servers s_1 and s_2 s.t. $c_{s_1} \geq d_v$ and $c_{s_2} \geq d_w$ **then**
allocate v on s_1 and w on s_2
else
no allocation possible
end if
end if
end if
end if
end if

BVAH/AVAH AllocateCloseTo Algorithm

Algorithm 2 AllocateCloseTo Algorithm

```
AllocateCloseTo (VM $v$, VM $w$);
/* allocates VM $v$ as close as possible to VM $w$ */
Let $A_w = (s, r, c, d)$
if $c_s \geq d_v$ then
    allocate $v$ on $s$
else
    if $\exists$ a server $s_1$ on rack $r$ s.t. $c_{s_1} \geq d_v$ then
        allocate $v$ on $s_1$
    else
        if $\exists$ a server $s_1$ on cluster $c$ s.t.. $c_{s_1} \geq d_v$ then
            allocate $v$ on $s_1$
        else
            if $\exists$ a server $s_1$ on data center $d$ s.t. $c_{s_1} \geq d_v$
            then
                allocate $v$ on $s_1$
            else
                if $\exists$ a server $s_1$ on data center $d_1$ ($d_1 \neq d$)
                s.t. $c_{s_1} \geq d_v$ then
                    allocate $v$ on $s_1$
                else
                    no allocation possible
                end if
            end if
        end if
    end if
end if
```