Network aware Scheduling for Cloud Data Center

a chapter of MS thesis work

Dharmesh Kakadia,

Advised by

Prof. Vasudeva Varma

SIEL, IIIT-Hyderabad, India.

joint work with Nandish Kopri, Unisys.

dharmesh.kakadia@research.iiit.ac.in

Scheduling : History

The word scheduling is believed to be originated from a latin word *schedula* around 14th Century, which then meant *papyrus strip*, slip of paper with writing on it. In 15th century, it started to be used as mean *timetable* and from there was adopted to mean scheduler that we currently use in computer science.
Scheduling in computing, is the process of deciding how to allocate resources to a set processes. [1]

[1] Source : WIkipedia

Scheduling : Motivation

- The resource arbitration is at the heart of the modern computers.
- It is *old* problem and likely to keep busy intelligent minds for few more decades.
- Save the world !!

Scheduling : Definition

In mathematical notation, all of my work can be summarized as,

$$Map < VM, PM >= f(Set < VM >, Set < PM >, context)$$

context can be

1. Process and Machine Model
2. Heterogeneity of Resources
3. Network Information

Coming up with function f

Thesis Problem

How to come up with function f ? That,

- Saves Energy in Data Center while, maintaing SLAs
- Saves battery of Mobile devices
- Saves Cost in MultiCloud environment
- Improves network scalability and performance

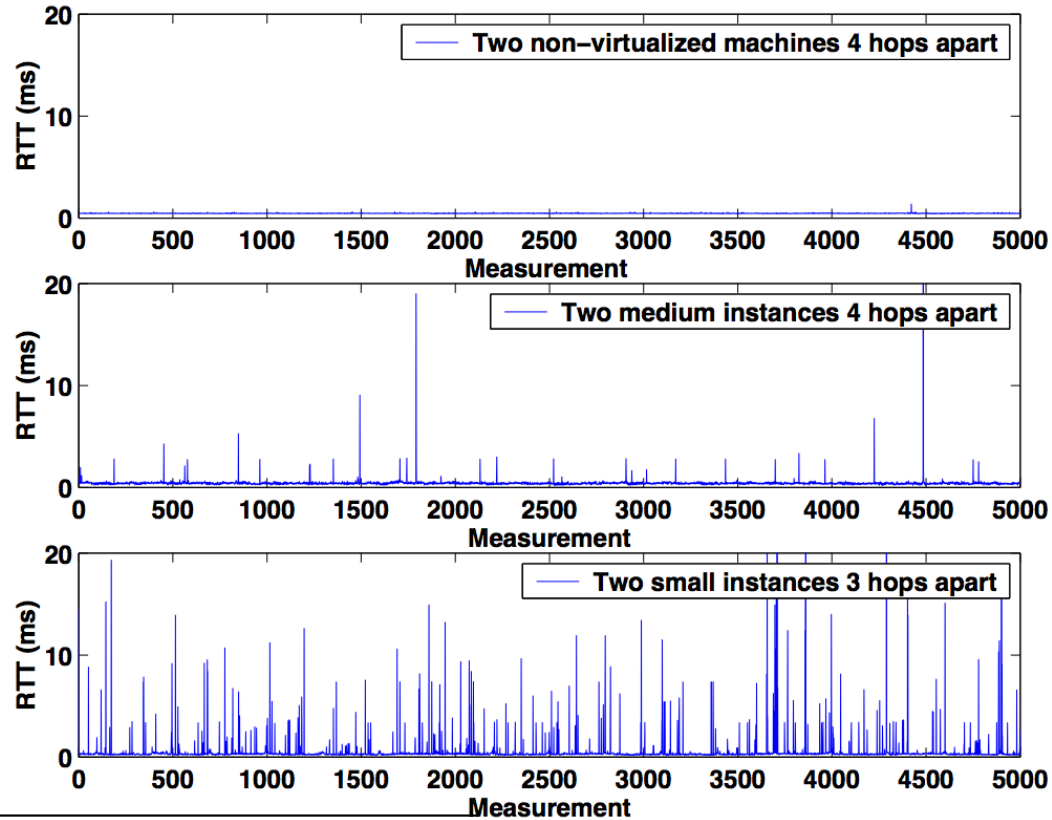
Today's Presentation

Come up with function f ? That,

- Saves Energy in Data Center while, maintaing SLAs
- Saves battery of Mobile devices
- Saves Cost in MultiCloud environment
- **Improves network scalability and performance**

Network Performance in Cloud

- In Amazon EC2, TCP/UDP throughput experienced by applications can fluctuate rapidly between 1 Gb/s and zero.
- Abnormally large packet delay variations among Amazon EC2 instances. [2]



[2] G. Wang et al. *The impact of virtualization on network performance of amazon ec2 data center.* (INFOCOM'2010)

Scalability

- Scheduling algorithm has to scale to millions of requests
- Network traffic at higher layers pose signifiant challenge for data center network scaling
- New applications in data center are pushing need for traffic localization in data center network

Problem

VM placement algorithm to consolidate VMs using network traffic patterns

Subproblems

- *How to identify?* - cluster VMs based on their traffic exchange patterns
- *How to place?* -placement algorithm to place VMs to localize internal datacenter traffic and improve application performance

How to identify?

VMCluster is a group of VMs that has large communication cost (c_{ij}) over time period T.

How to identify?

VMCluster is a group of VMs that has large communication cost (c_{ij}) over time period T.

$$c_{ij} = AccessRate_{ij} \times Delay_{ij}$$

$AccessRate_{ij}$ is rate of data exchange between VM_i and VM_j and $Delay_{ij}$ is the communication delay between them.

VMCluster Formation Algorithm

$$AccessMatrix_{n \times n} = \begin{bmatrix} 0 & c_{12} & \cdots & c_{1n} \\ c_{21} & 0 & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots \\ c_{n1} & c_{n2} & \cdots & 0 \end{bmatrix}$$

c_{ij} is maintained over time period T in moving window fashion and mean is taken as the value.

for each row $A_i \in$ AccessMatrix **do**
  if maxElement$(A_i) > (1 + opt_threshold) * avg_comm_cost$ **then**
    form a new VMCluster from non-zero elements of A_i
  end if
end for

How to place ?

How to place ?

- Which VM to migrate?

How to place ?

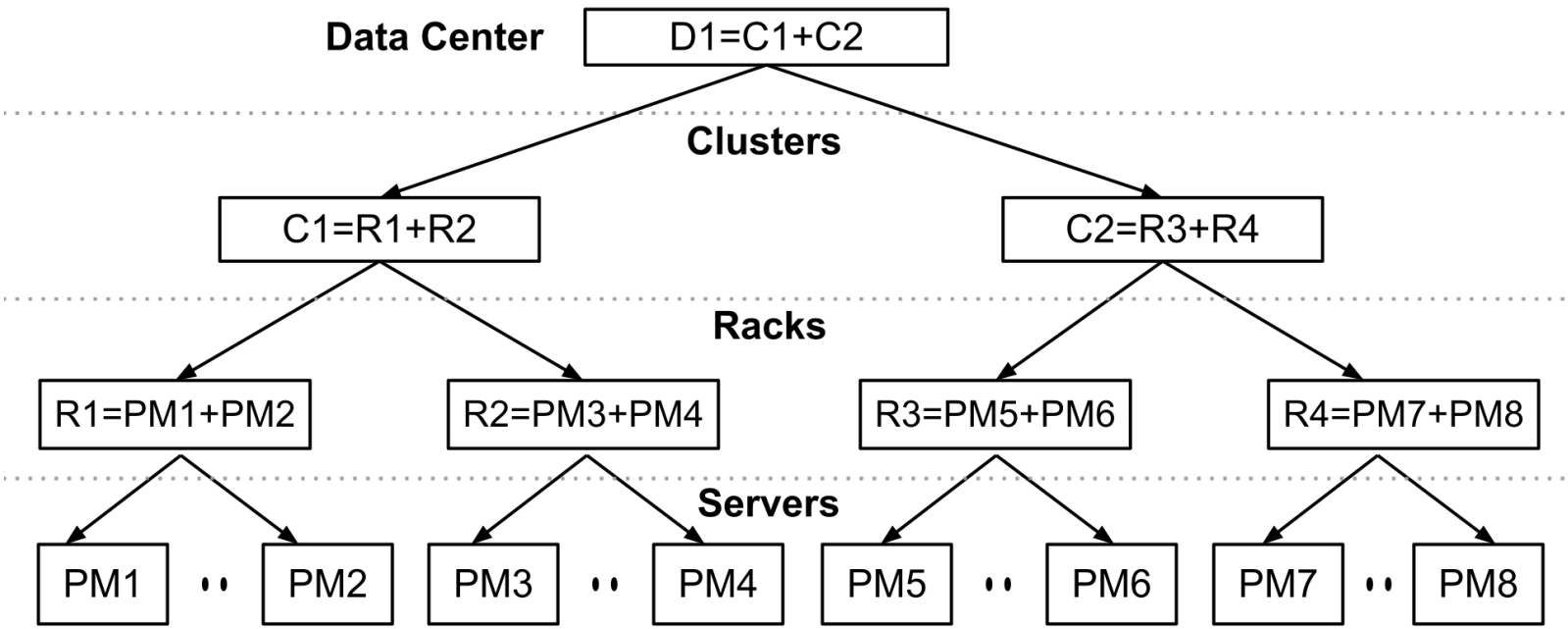
- Which VM to migrate?
- Where *can* we migrate?

How to place ?

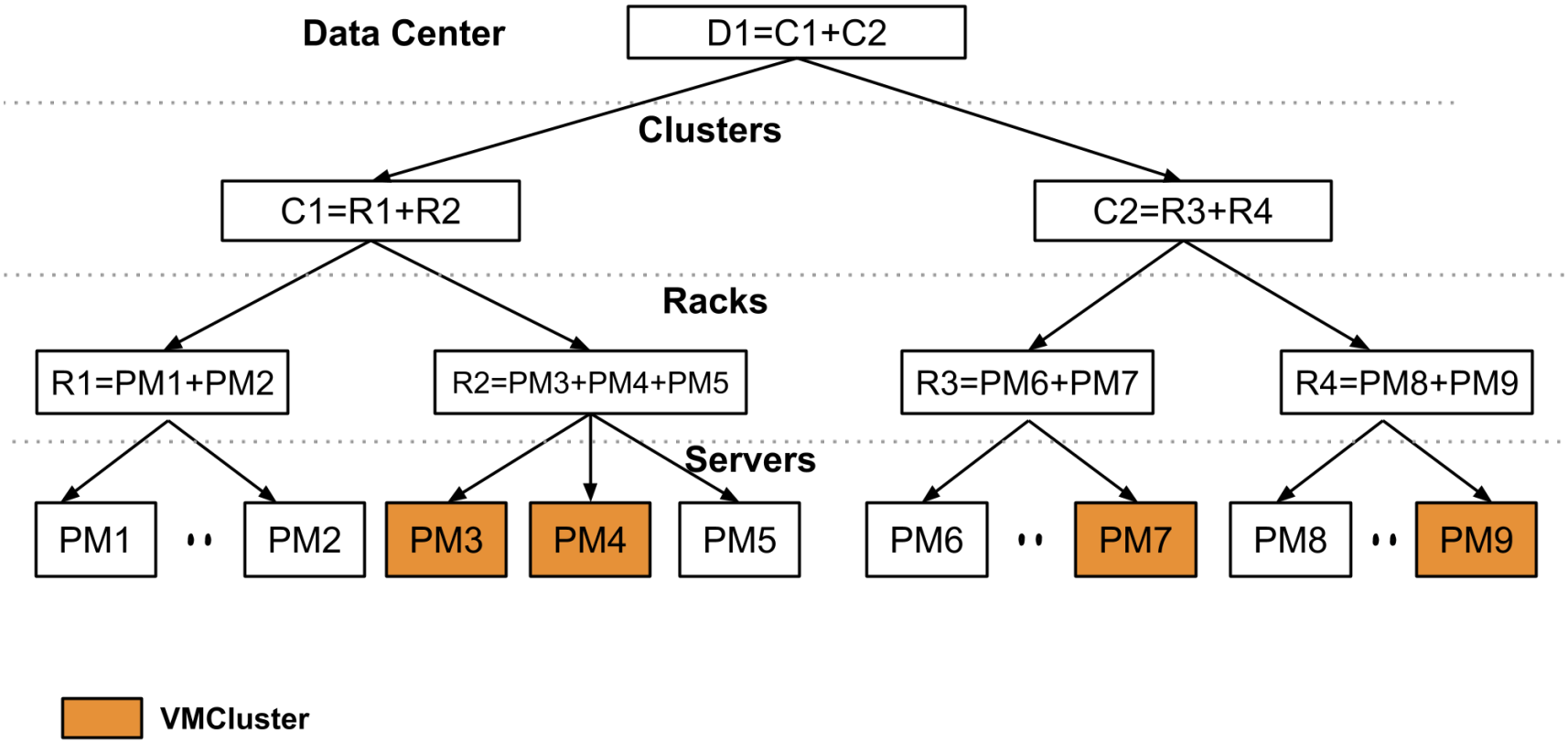
- Which VM to migrate?
- Where *can* we migrate?
- Will the the effort be worth?

Communication Cost Tree

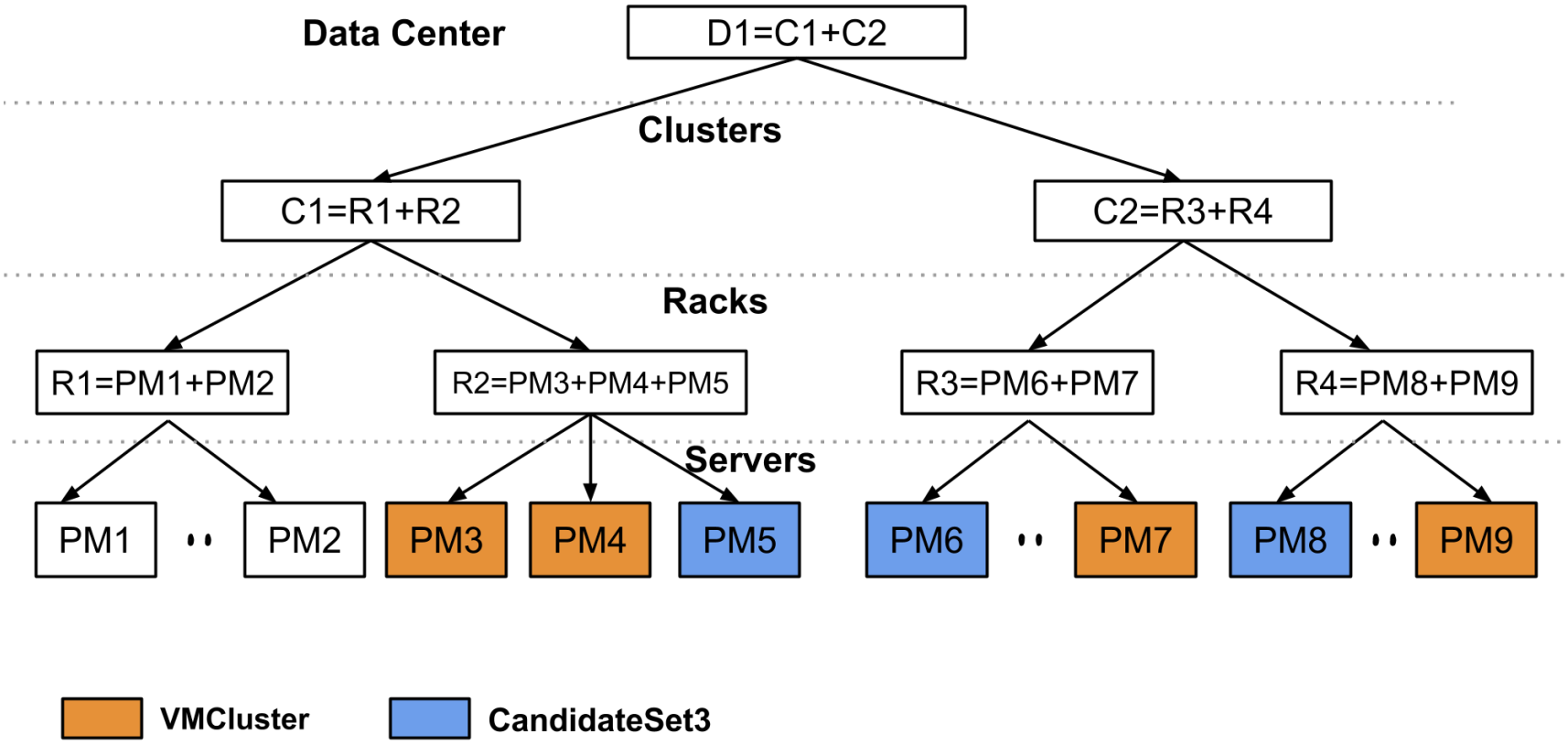
- Each node represents cost of communication of devices connected to it.



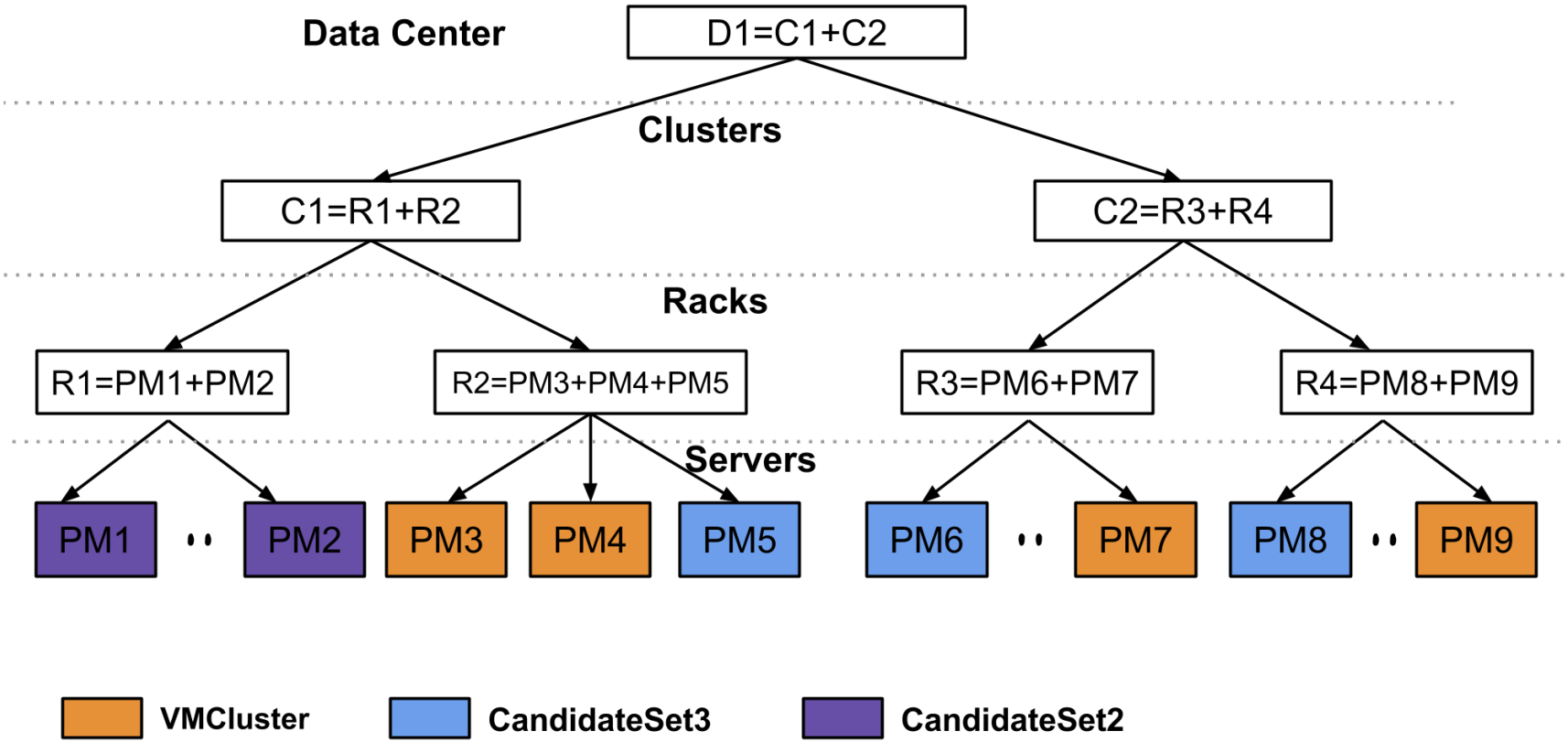
Example : VMCluster



Example : $CandidateSet_3$



Example : $CandidateSet_2$



How to place ?

How to place ?

Which VM to migrate?

$$VMtoMigrate = \arg\max_{VM_i} \sum_{j=1}^{|VMCluster|} c_{ij}$$

How to place ?

Which VM to migrate?

$$VMtoMigrate = \arg\max_{VM_i} \sum_{j=1}^{|VMCluster|} c_{ij}$$

Where *can* we migrate?

$$CandidateSet_i(VMCluster_j) = \{c \mid \text{where c and } VMCluster_j \text{ have a common ancestor at level } i\} - CandidateSet_{i+1}(VMCluster_j)$$

How to place ?

Which VM to migrate?

$$VMtoMigrate = \arg\max_{VM_i} \sum_{j=1}^{|VMCluster|} c_{ij}$$

Where *can* we migrate?

$$CandidateSet_i(VMCluster_j) = \{c \mid \text{where c and } VMCluster_j \text{ have a common ancestor at level } i\} - CandidateSet_{i+1}(VMCluster_j)$$

Will the the effort be worth?

$$PerfGain = \sum_{j=1}^{|VMCluster|} \frac{c_{ij} - c'_{ij}}{c_{ij}}$$

Consolidation Algorithm

- Select the VM to migrate
- Identify CandidateSets
- Select destination PM
  - Overload the destination
  - Gain is significant

Consolidation Algorithm

for $VMCluster_j \in$ VMClusters **do**
  Select *VMtoMigrate*
  for i from leaf to root **do**
    Form $CandidateSet_i(VMCluster_j - VMtoMigrate)$
    for $PM \in candidateSet_i$ **do**
      if UtilAfterMigration(PM, *VMtoMigrate*)
      $<$overload_threshold AND PerfGain(PM, *VMtoMigrate*)
      $>$ significance_threshold **then**
        migrate VM to PM
        continue to next VMCluster
      end if
    end for
  end for
end for

Trace Statistics

Traces from three real world data centers, two from universities (uni1, uni2) and one from private data center (prv1) [4].

Property	Uni1	Uni2	Prv1
Number of Short non-I/O-intensive jobs	513	3637	3152
Number of Short I/O-intensive jobs	223	1834	1798
Number of Medium non-I/O-intensive jobs	135	628	173
Number of Medium I/O-intensive jobs	186	864	231
Number of Long non-I/O-intensive jobs	112	319	59
Number of Long I/O-intensive jobs	160	418	358
Number of Servers	500	1093	1088
Number of Devices	22	36	96
Over Subscription	2:1	47:1	8:3

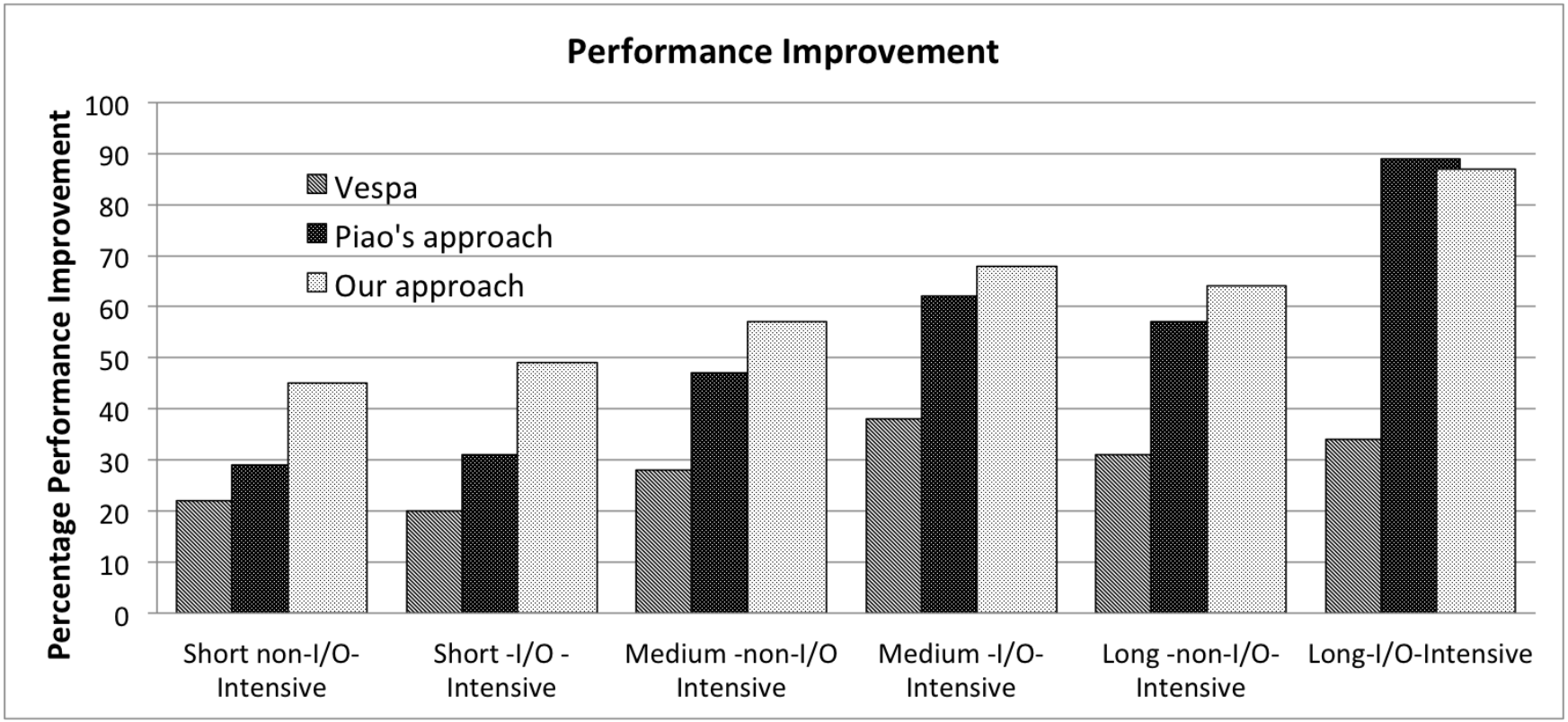
Experimental Evaluation

We compared our approach to traditional placement approaches like Vespa [1] and previous network-aware algorithm like Piao's approach [2].

- Extended NetworkCloudSim [3] to support SDN.
- Floodlight[3] as our SDN controller.
- The server properties are assumed to be HP ProLiant ML110 G5 (1 x [Xeon 3075 2660 MHz, 2 cores]), 4GB) connected through 1G using HP ProCurve switches.

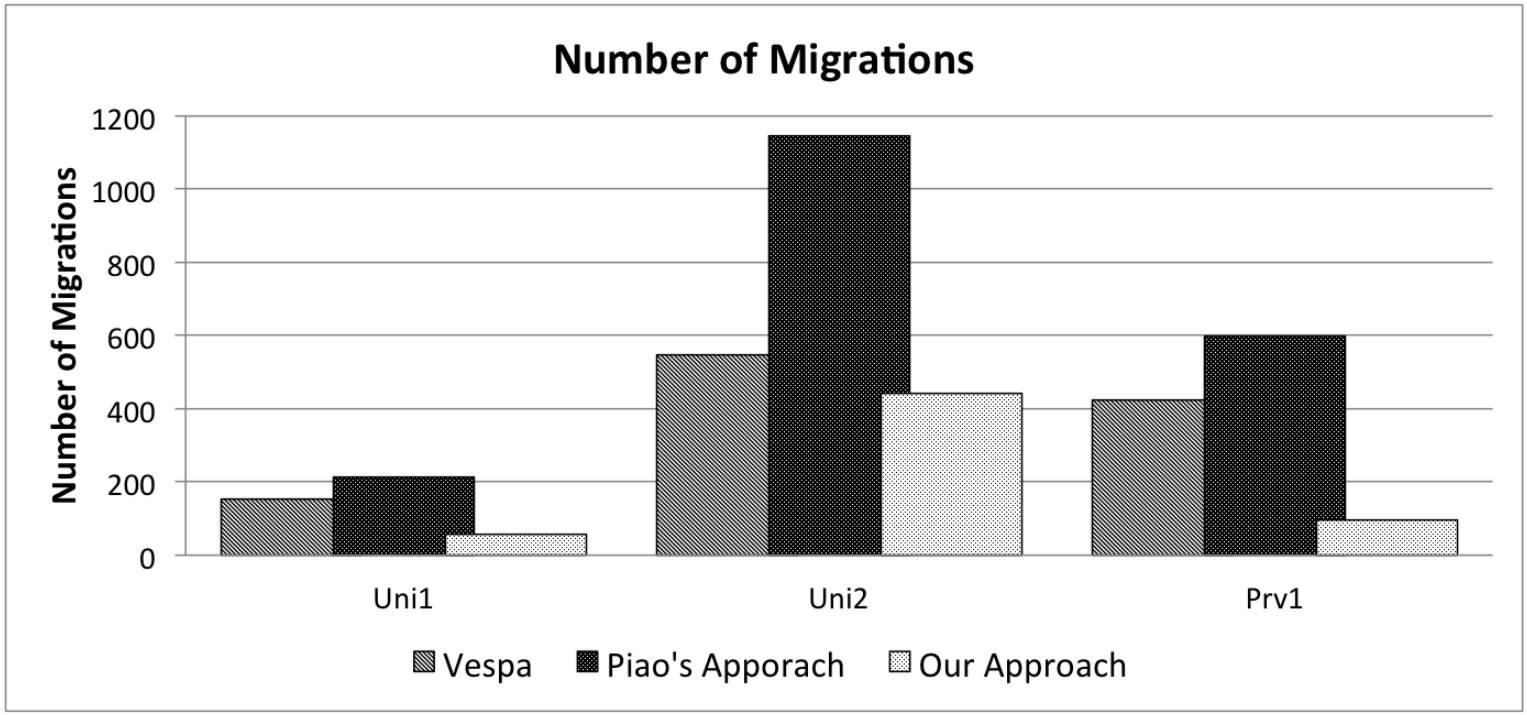
[3] http://www.projectfloodlight.org/

Results : Performance Improvement



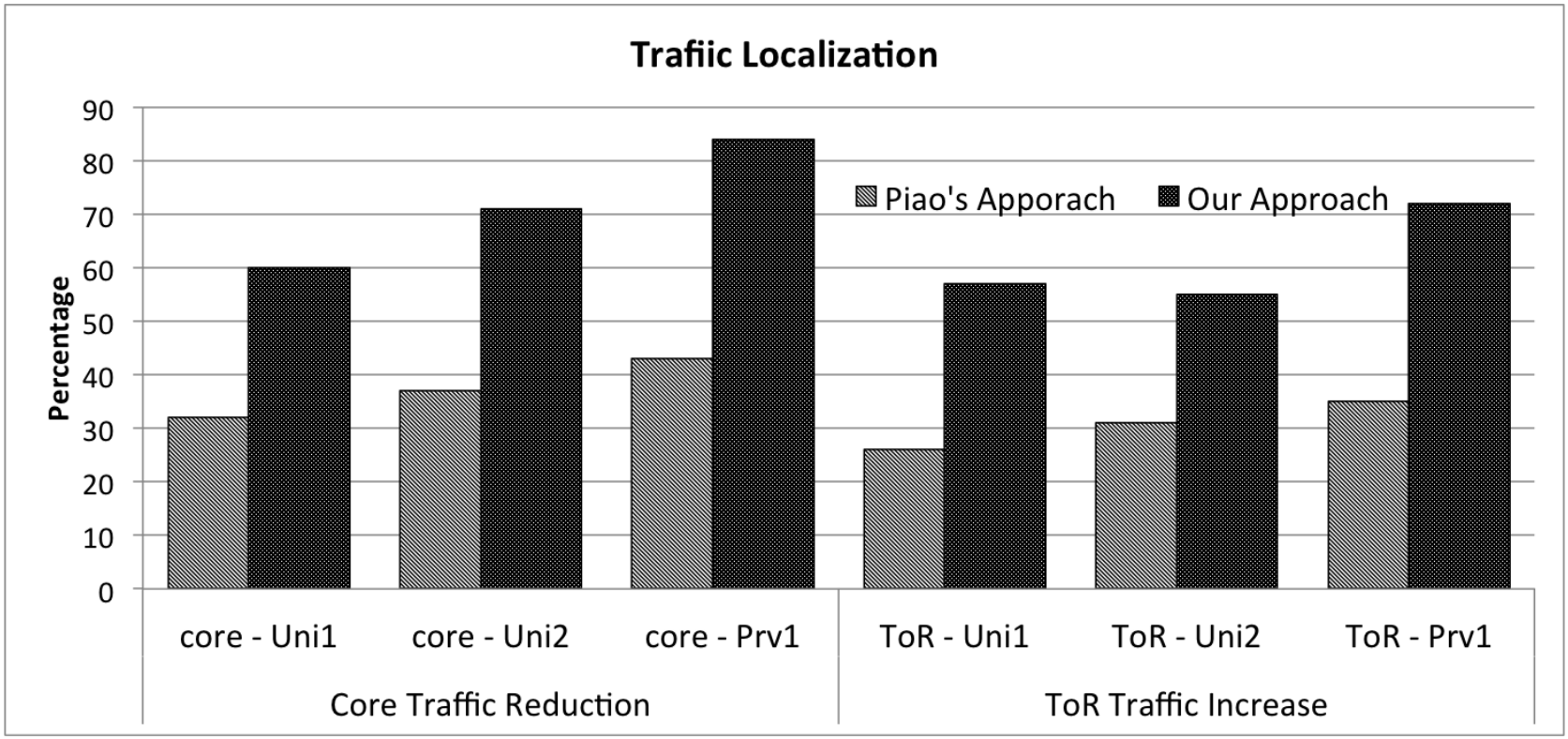
- I/O intensive jobs are benefited most, but others also share the benefit
- Short jobs are important for overall performance improvement

Results : Number of Migrations



- Every migration is not equally beneficial

Results : Traffic Localization



- 60% increase ToR traffic (vs 30% by Piao's approach)
- 70% decrease Core traffic (vs 37% by Piao's approach)

Results : Complexity – Time, Variance and Migrations

Measure	Trace	Vespa	Piao's approach	**Our approach**
Avg. scheduling Time (ms)	Uni1	504	677	**217**
	Uni2	784	1197	**376**
	Prv1	718	1076	**324**

Results : Complexity – Time, Variance and Migrations

Measure	Trace	Vespa	Piao's approach	**Our approach**
Avg. scheduling Time (ms)	Uni1	504	677	**217**
	Uni2	784	1197	**376**
	Prv1	718	1076	**324**
Worst-case scheduling Time (ms)	Uni1	846	1087	**502**
	Uni2	973	1316	**558**
	Prv1	894	1278	**539**

Results : Complexity – Time, Variance and Migrations

Measure	Trace	Vespa	Piao's approach	**Our approach**
Avg. scheduling Time (ms)	Uni1	504	677	**217**
	Uni2	784	1197	**376**
	Prv1	718	1076	**324**
Worst-case scheduling Time (ms)	Uni1	846	1087	**502**
	Uni2	973	1316	**558**
	Prv1	894	1278	**539**
Variance in scheduling Time	Uni1	179	146	**70**
	Uni2	234	246	**98**
	Prv1	214	216	**89**

Results : Complexity – Time, Variance and Migrations

Measure	Trace	Vespa	Piao's approach	**Our approach**
Avg. scheduling Time (ms)	Uni1	504	677	**217**
	Uni2	784	1197	**376**
	Prv1	718	1076	**324**
Worst-case scheduling Time (ms)	Uni1	846	1087	**502**
	Uni2	973	1316	**558**
	Prv1	894	1278	**539**
Variance in scheduling Time	Uni1	179	146	**70**
	Uni2	234	246	**98**
	Prv1	214	216	**89**
Number of Migrations	Uni1	154	213	**56**
	Uni2	547	1145	**441**
	Prv1	423	597	**96**

Conclusion

- Network aware placement (and traffic localization) helps in Network scaling.
- VM Scheduler should be aware of migrations.
- Think like a scheduler and think *rationally*. You may not want *all* the migrations.

Related Publication

1. **Network-aware Virtual Machine Consolidation for Large Data Centers.** Dharmesh Kakadia, Nandish Kopri and Vasudeva Varma. In *NDM collocated with SC'13*.
2. **Optimizing Partition Placement in Virtualized Environments.** Dharmesh Kakadia and Nandish Kopri. *Patent P13710918*.

References

1. C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici. A scalable application placement controller for enterprise data centers. (WWW'2007)

2. J. Piao and J. Yan. A network-aware virtual machine placement and migration approach in cloud computing. (GCC'2010)

3. S. K. Garg and R. Buyya. Networkcloudsim: Modeling parallel applications in cloud simulations. (UCC'2011)

4. T. Benson, A. Akella, and D. A. Maltz. Network traffic characteristics of data centers in the wild. (IMC'2010)

@ MSR

Working with Dr. Kaushik Rajan, on a performance modeling tool, Perforator to predict the execution time/ Resource requirements of Map Reduce DAGs.

1. Started with Hadoop and Hive jobs, Want to move to all the supported frameworks on YARN.
2. Integrating this work with Reservation based Scheduler (YARN-1051). What reservation to ask for?
3. More details @ http://research.microsoft.com/Perforator. Now have detailed results over more general jobs.

Thank you