Cost-Efficient VM Configuration Algorithm in the Cloud using Mix Scaling Strategy

Presenter: Li Lu

Li Lu, Jiadi Yu, Yanmin Zhu, Guangtao Xue, Shiyou Qian, Minglu Li
Department of Computer Science and Engineering,
Shanghai Jiao Tong University

Popularity of Cloud Computing

vs.

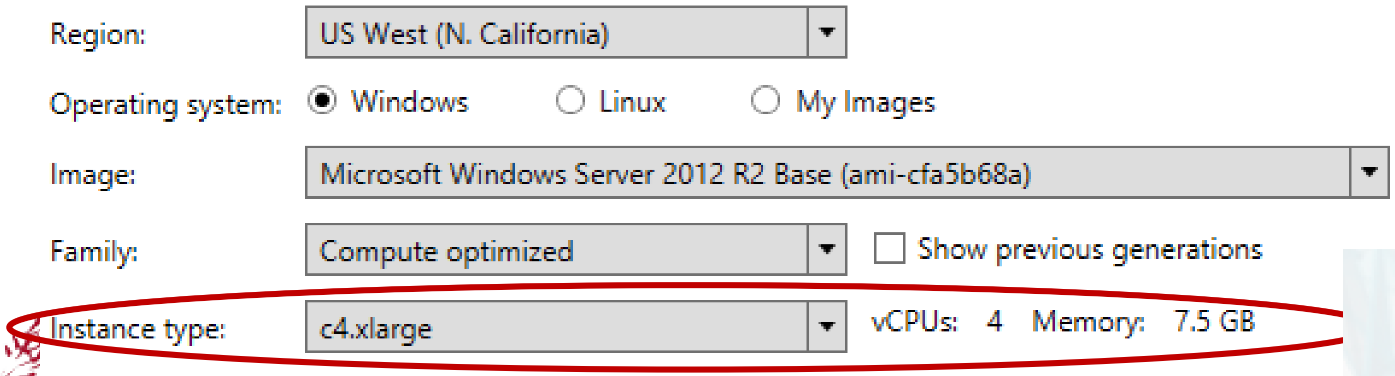
- Cloud Computing vs. Typical Infrastructure
  - Thanks to pay-per-use pricing, more elastic in management
  - Cloud computing can satisfy the peak workload without over-provision computing resources
  - e.g., Brickfish migrates its services to cloud leading to a decrease of cost from $700,000 to $200,000

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

IEEE ICC PARIS 21-25 MAY 2017
INTERNATIONAL CONFERENCE ON COMMUNICATIONS

Difficulties in Managing Cloud Resources

- VM instance type selection
  - Different VM instance type configurations → different performance & cost
- Precise VM instance type selection
  - need accurate prediction of future workload (difficult!)
  - even experienced administrators cannot precisely select VM instance type
- Key point: the tradeoff between cost and performance during the runtime



Region: US West (N. California)

Operating system: ● Windows ○ Linux ○ My Images

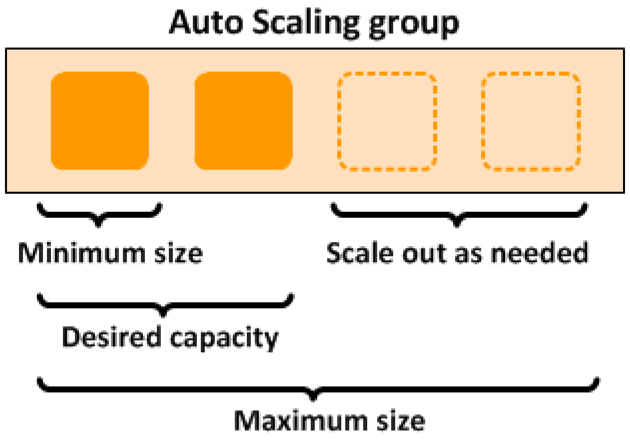
Image: Microsoft Windows Server 2012 R2 Base (ami-cfa5b68a)

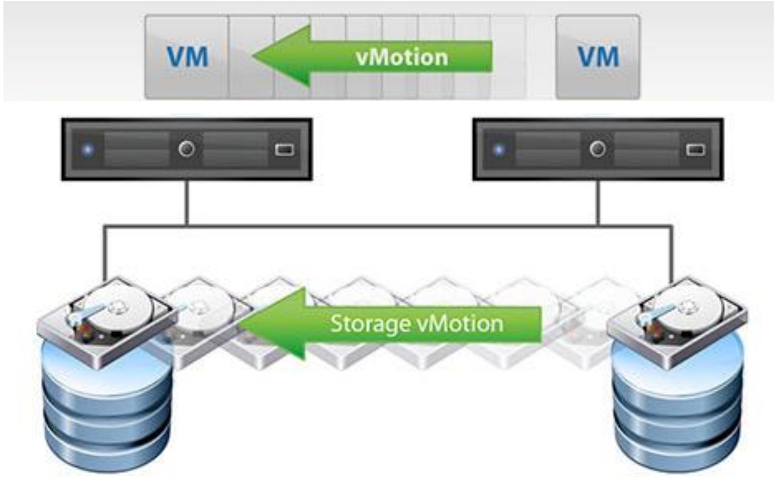
Family: Compute optimized ☐ Show previous generations

Instance type: c4.xlarge vCPUs: 4 Memory: 7.5 GB

Existing Solutions

- Cost-aware homogeneous VM configurations
  - Same VM instance type
- Multi-mechanisms in VM configurations
  - Local-resize, replication, migration
- However, during the runtime in cloud,
  - Utilizing heterogeneous VM instance types is more cost-efficient
  - Migration of VM leads to high performance degradation





Outline

- Problem Definition
- Cost-efficient Mix Scaling Algorithm
- Evaluation
- Conclusion

VM Configuration Model

- objective: minimize the renting cost of cloud resources
- constraints: the service rate of the configuration should be larger than the arrival rate of requests

$$\min \sum_{i=1}^{K} x_i c_i$$

$$s.t. \sum_{i=1}^{K} x_i \mu_i \geq \lambda$$

$$x_i \in N, \qquad i = 1, 2, \ldots, K$$

- the number of VM instance types: K
- the cost of the ith VM instance type: c_i
- the maximum service rate of ith VM instance type: μ_i
- the arrival rate of requests: λ
- the number of ith VM instance type in the configuration: x_i

Differences between Two Constitute Configurations

- Due to the workload fluctuation, the two constitute VM configurations x_{old} and x_{new} are almost always different in all time slots.
  - Note that x_{old} and x_{new} are K-dimension vectors
- 3 situations may occur:
  - $x_{new} \geq x_{old}$: more VMs of all types are needed to meet performance requirement
  - $x_{new} \leq x_{old}$: less VMs of all types are needed to be cost-efficient
  - $x_{new} \neq x_{old}$: need to add or delete several VMs of different instance types
- For the first 2 situations, renting more or deleting several VMs would be OK
- For the 3rd situation, migrations would occur, which should be control to improve the performance

Cost-Migration Delay Tradeoff

- Tradeoff: Cost vs. Migration delay
  - For Cost: the objective minimizes the cost

$$\min \sum_{i=1}^{K} x_i c_i$$

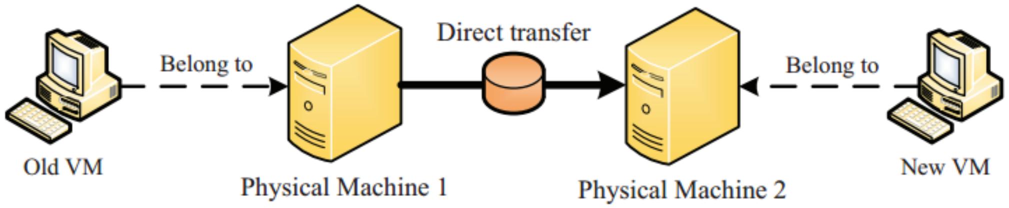
  - For Migration delay: need to modeled

Migration Delay Modeling

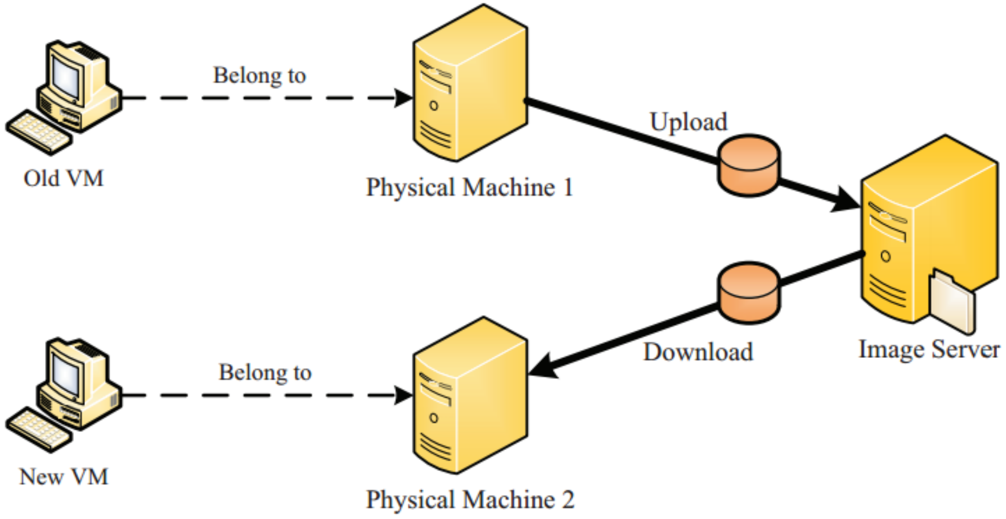
- Migration Mechanism in Cloud
  - Instead of directly migration, migration in cloud should utilize the image server as a bridge
- Migration Delay can be modelled as:

$$\alpha = 2\frac{D}{b} + s$$

- where *D* is the image size, *b* is the bandwidth, *s* is the start time of a new VM



Normal Migration



Migration in Cloud

Cost-Migration Delay Tradeoff (COMDT) Problem

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{K} x_i c_i \\
s.t. \quad & \sum_{i=1}^{K} x_i \mu_i \geq \lambda \\
& x_i \in N, \qquad i = 1, 2, \ldots, K
\end{aligned}
$$

Original Problem

→

$$
\begin{aligned}
\min \quad & \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i=1}^{K} x_i(t) c_i \\
s.t. \quad & \sum_{i=1}^{K} x_i(t) \mu_i \geq \lambda(t), \forall t \\
& \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \alpha(t) \leq MT \\
& x_i \in N, \qquad \forall i, t
\end{aligned}
$$

Migration Delay Constraint

Cost-Migration Delay Tradeoff Problem

Outline

- Problem Definition
- Cost-efficient Mix Scaling Algorithm
- Evaluation
- Conclusion

Difficulty in Solving the COMDT Problem

- The COMDT problem aims to
  - minimize the long-term cost
  - constrain the long-term migration delay
- Notice that there are two limits in the objective and the migration delay constraint
  - Hard to solve with typical optimization techniques
  - Adopt Lyapunov optimization techniques

$$\min \enclose{circle}{\lim_{T\to\infty}} \frac{1}{T}\sum_{t=0}^{T-1}\sum_{i=1}^{K} x_i(t)c_i$$

$$s.t. \quad \sum_{i=1}^{K} x_i(t)\mu_i \ge \lambda(t), \forall t$$

$$\enclose{circle}{\lim_{T\to\infty}} \frac{1}{T}\sum_{t=0}^{T-1}\alpha(t) \le MT$$

$$x_i \in N, \qquad \forall i,t$$

Cost-Efficient Mix Scaling Algorithm

- Virtual Queue Construction $Q(t)$
- Lyapunov Drift Construction $\Delta L(t)$
- One-slot Optimization Problem Construction
- Optimization Problem Solving

Virtual Queue

- Migration delay → Virtual queue
  - $Q(0) = 0$
  - $Q(t+1) = \max\{Q(t) + \alpha(t) - MT, 0\}$
- The equivalence of migration delay constraint and the stability of virtual queue
  - $\lim_{T\to\infty} \sum_{t=0}^{T-1} \alpha(t) \leq MT \Leftrightarrow \lim_{T\to\infty} \frac{Q(t)}{T} = 0$
- Thus, we first construct the virtual queue and utilize it to replace the migration delay constraint

Lyapunov Drift

- To represent the stability of the virtual queue, we define two notations based on Lyapunov optimization framework
  - Lyapunov function: $L(t) = \frac{1}{2}Q(t)^2$
  - Lyapunov drift: $\Delta L(t) = E\{L(t+1) - L(t) | Q(t)\}$
- There always exists an upper bound of the Lyapunov drift:
  - $\Delta L(t) \leq M + Q(t)E\{2\frac{D(t)}{b} + B | Q(t)\}$
  - where $M = \frac{1}{2}(2\frac{D_{max}}{b} + s - MT)^2$, $B = s - MT$

One-slot Optimization Problem

- Utilizing the upper bound, we formulate the objective of the one-slot optimization problem
  - $VC(t) + \Delta L(t) \le M + VC(t) + Q(t)E\{2\frac{D(t)}{b} + B|Q(t)\}$
  - where *C(t)* is the objective of COMDT problem
- To minimize this objective, the one-slot optimization problem is

$$\begin{aligned} &\min \; VC(t) + Q(t)(2\frac{D(t)}{b} + B) \\ &s.t. \; \sum_{i=1}^{K} x_i(t)\mu_i \ge \lambda(t), \forall t \\ &\qquad x_i \in N, \qquad \forall i,t \end{aligned}$$

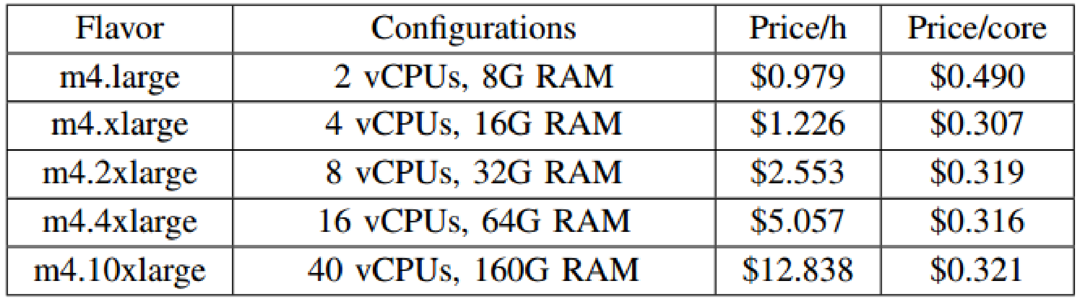
- Finally, we adopt typical optimization techniques to solve it

Outline

- Problem Definition
- Cost-efficient Mix Scaling Algorithm
- Evaluation
- Conclusion

Simulation Setup

- Workload λ:
  - Generated by TPC-W
  - 2 types of workload: low-fluctuation & high fluctuation
- VM types: 5 types as follows
  - capacity μ: preliminary runtime test on our OpenStack platform
  - price c: the same as AWS



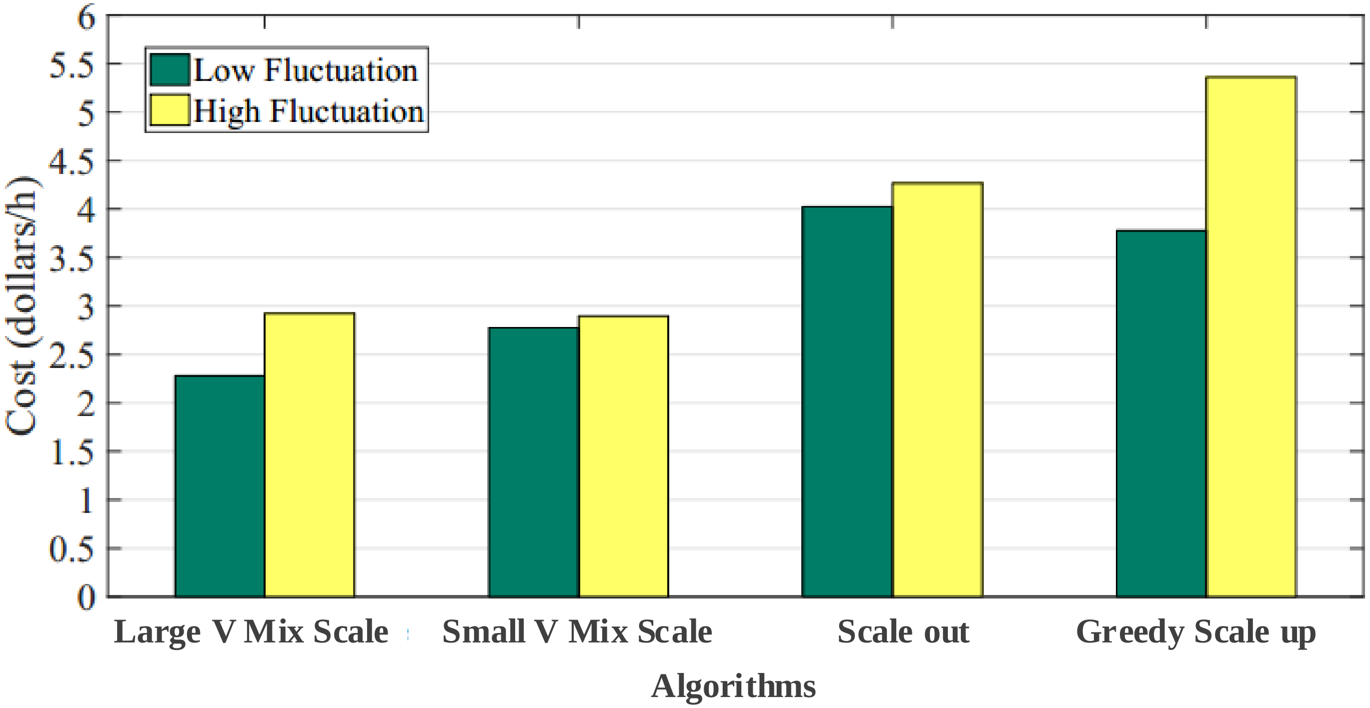
Flavor	Configurations	Price/h	Price/core
m4.large	2 vCPUs, 8G RAM	\$0.979	\$0.490
m4.xlarge	4 vCPUs, 16G RAM	\$1.226	\$0.307
m4.2xlarge	8 vCPUs, 32G RAM	\$2.553	\$0.319
m4.4xlarge	16 vCPUs, 64G RAM	\$5.057	\$0.316
m4.10xlarge	40 vCPUs, 160G RAM	\$12.838	\$0.321

Comparison methods

- 4 algorithms:
  - scale out: only use one type VM, and scale the number of the VM
  - greedy scale up: first scale the VM type, then the number
  - mix scale: our algorithm. 2 variations
    - small V mix scale: focus more on migration delay
    - large V mix scale: focus more on cost

Average Cost

- Our algorithm with small V achieves 30.8% and 26.3% higher cost-efficiency than that of scale out and greedy scale up algorithms
- Our algorithm with large V achieves 31.1% and 26.5% higher cost-efficiency than that of scale out and greedy scale up algorithms

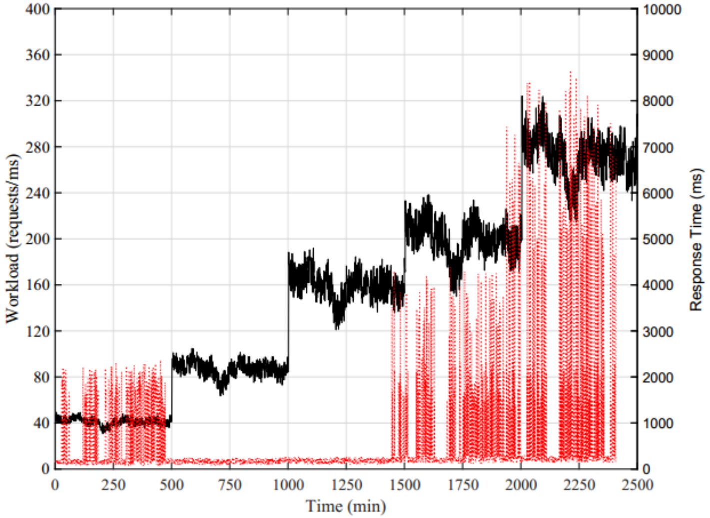


上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

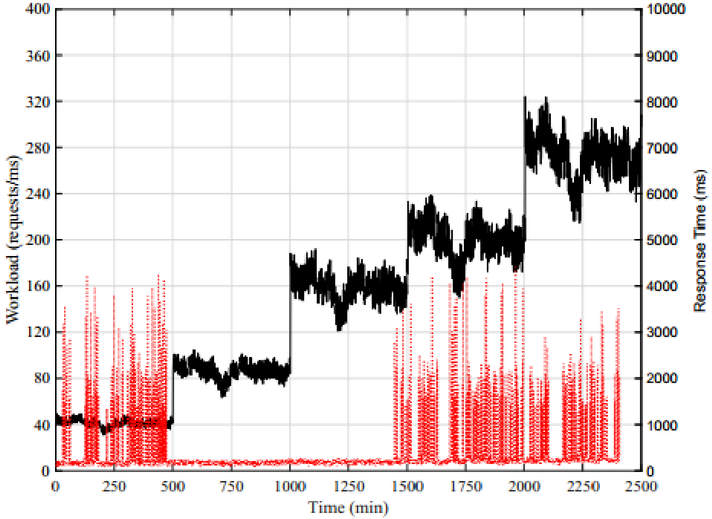
IEEE ICC PARIS 21-25 MAY 2017
INTERNATIONAL CONFERENCE ON COMMUNICATIONS

Response Time

- Under the same workload, small V mix scale algorithm can reduce 38.19% migration delay to further reduce the response time compared with large V mix scale algorithm.



Large V



Small V

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

IEEE ICC PARIS
INTERNATIONAL CONFERENCE ON COMMUNICATIONS
21-25 MAY 2017

Outline

- Problem Definition
- Cost-efficient Mix Scaling Algorithm
- Evaluation
- Conclusion

Conclusion

- Formulate the cost-migration delay tradeoff problem
  - both cost of cloud resources and migration delay are considered
- Propose the cost-efficient mix scaling algorithm
  - solve the COMDT problem utilizing the Lyapunov optimization techniques
- Demonstrate the efficiency and feasibility of the algorithm
  - save 31.1% and 26.5% cost while controlling migration delay compared with scale out and scale up algorithms

Thank you!

Q & A

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

IEEE ICC PARIS
INTERNATIONAL CONFERENCE ON COMMUNICATIONS
21-25 MAY 2017