# *Handling Flash Deals with Soft Guarantee in Hybrid Cloud*

Yipei Niu[1], Fangming Liu[1], Xincai Fei[1], Bo Li[2]

Email: fmliu@hust.edu.cn

[1]*Huazhong University of Science & Technology*
[2]*The Hong Kong University of Science & Technology*

# What are flash deals?


Pre-Order


- **Amazon Prime Day**
  - Prime Day is a one-day-only global shopping event
  - New deals are released as often as every five minutes
- **New iPhones pre-order**
  - iPhone 6 preorders were slated to start at midnight
- **WeChat red envelope**
  - WeChat has offered virtual red envelope containing virtual money that can be cashed out
  - envelope and hence the money

**Flash deals offer benefits to subscribers within short time!**

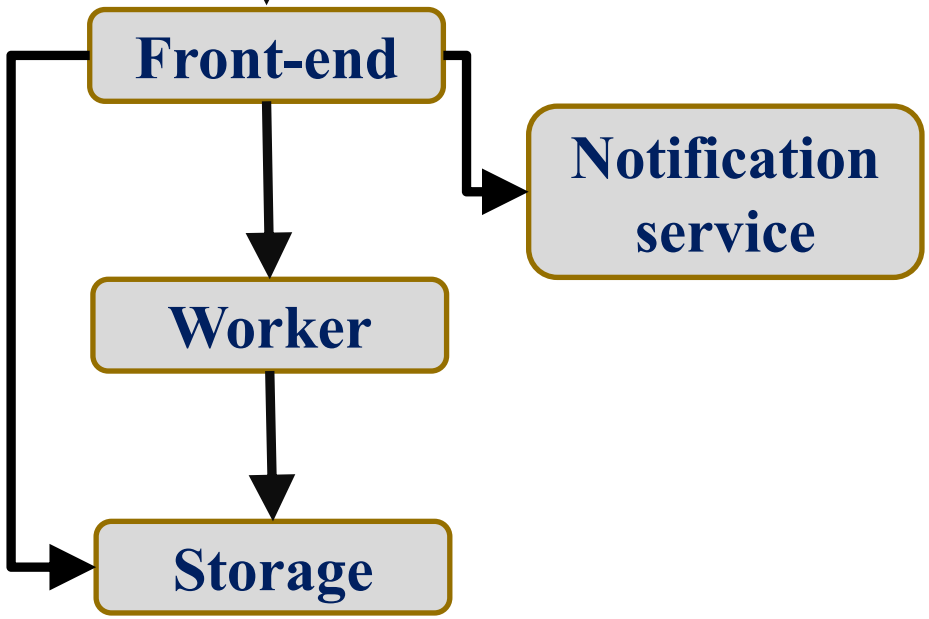
# Fast & Simple

**Simple**

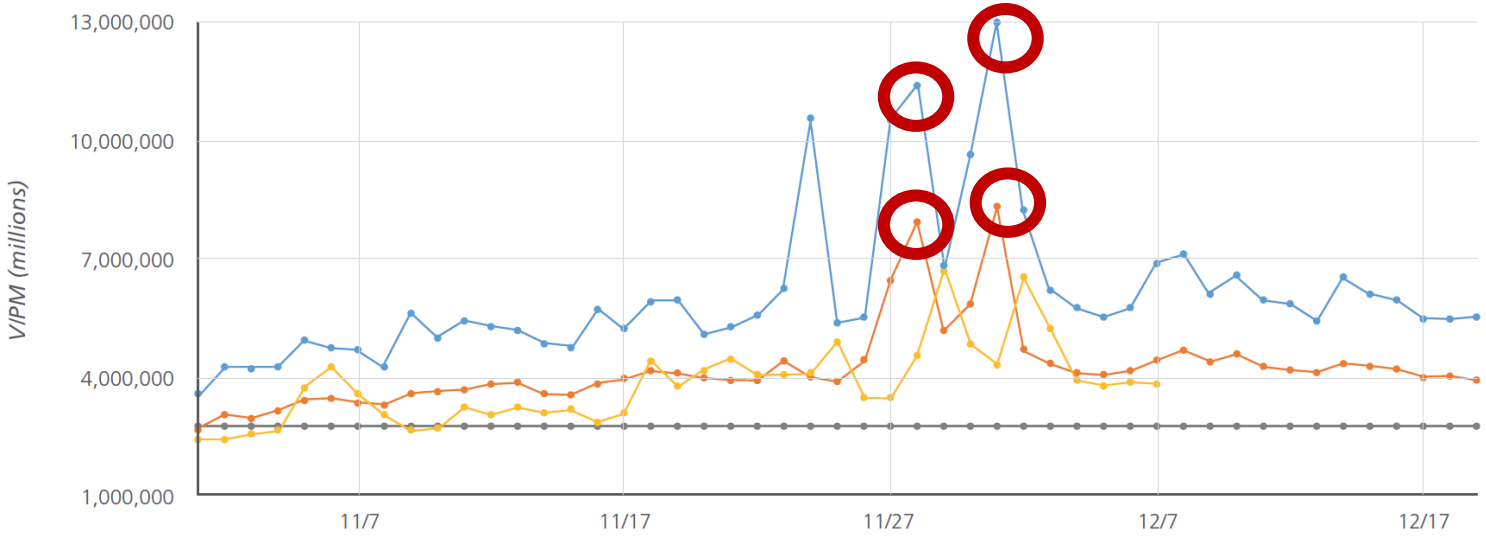
- Easy to get
  - One click on mouse
  - Shake smartphone
- Straightforward business logic
  - First some persons win

**Fast**

- Limited profit
  - Discounted merchandise
  - Newly released iPhone
  - WeChat Red Envelope
- Short duration
  - Refresh every 5 minutes
  - Midnight on release day
  - Spring Festival Gala



# Yet crowded

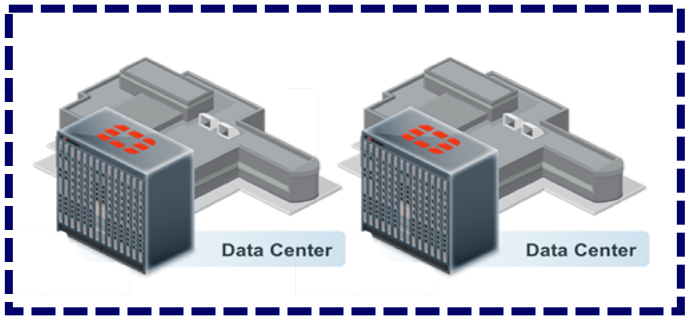



- Sales on Amazon's Prime Day ***exceeded*** Black Friday in 2014
- The times of shaking phones reached a total of ***11 billion*** and a peak of ***810 million per minute***
- The pre-orders exceeded ***two million*** in the first 24 hours, making Apple Store unresponsive

**How to handle such fast, simple, and crowded flash deals?**

# Is private cloud OK?

- **Private cloud**
  - Dedicated datacenter or server cluster
  - Virtual resources provided by cloud providers




**Private cloud**

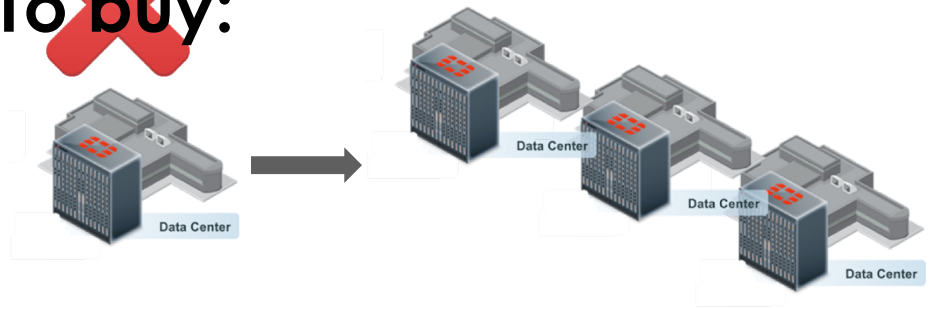
- **Private cloud solution**
  - **Advantages**
    - Enhanced security
    - Ultimate control
  - **Disadvantages**
    - Limited capacity
    - Low scalability
    - Complex to operate

- **Requirement of security**
  - Protect confidential data
- **Requirement of performance**
  - Maximum uptime
  - Fast page load time

**How to increase capacity and improve scalability?**

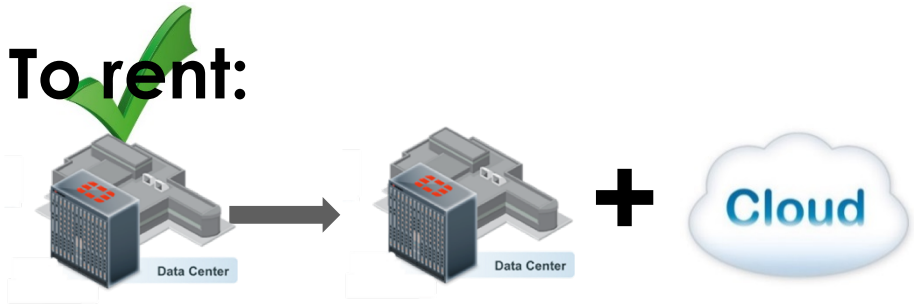
# To buy or To rent?

**To buy:**




- Cost Increases linearly
  - Infrastructure
- Unable to scale up or down based on workloads
  - Temporary use

**To rent:**

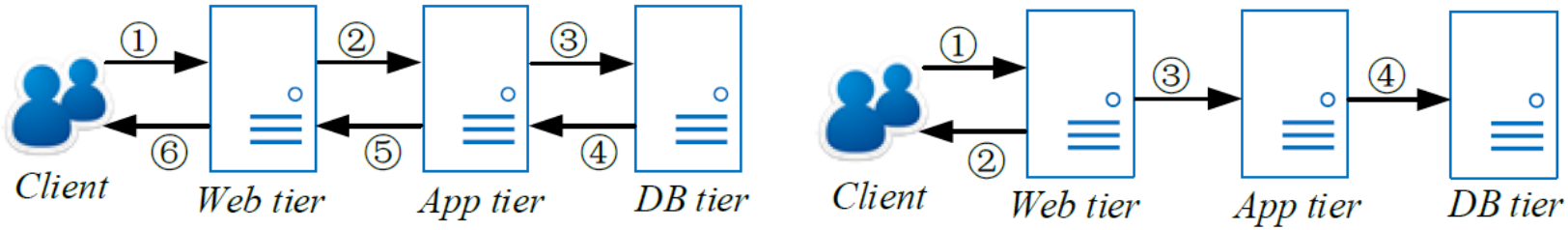



- Low price
- Auto scaling
  - Scalable capacity
  - Easy to operate
- Potentially unlimited resources

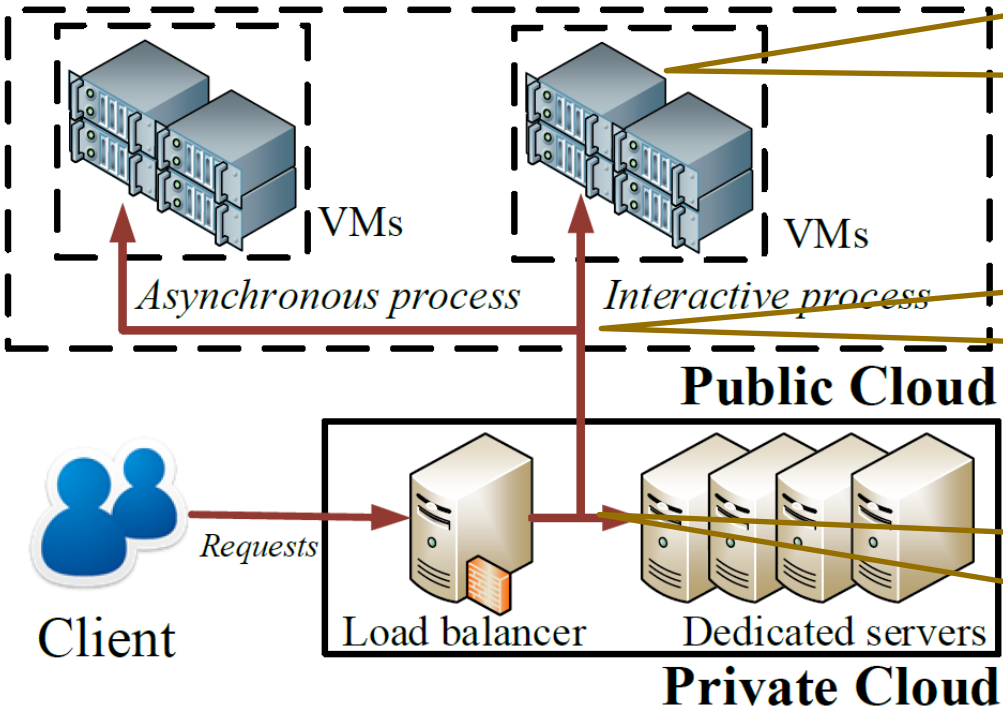
**Hybrid cloud solution is a promising choice!**

# Is hybrid cloud enough?

- Revisit flash deals
  - Flash deals always bring benefits
  - Flash deals involve simple operations
- Postpone serving requests
  - Incentive to wait longer to get benefits
  - Serve partial requests instantly
  - Postpone serving others
- One example
  - Instead of waiting for the results returned from the application tier (1, 2, 3, 4, 5 in left)
  - Web servers send responses back to users (2 in right)
  - Guarantee the requests served asynchronously within deadline (3, 4 in right)



# Hybrid cloud with soft guarantee




- ✓ Obtaining the best performance
- ✓ Preventing cost from exceeding budget
- ✓ Assigning partial requests to the asynchronous process
- ✓ Distributing workloads between the private and public clouds

- Problems
  - Without prior knowledge of requests
  - How to schedule requests
  - How to adjust the scale of public cloud

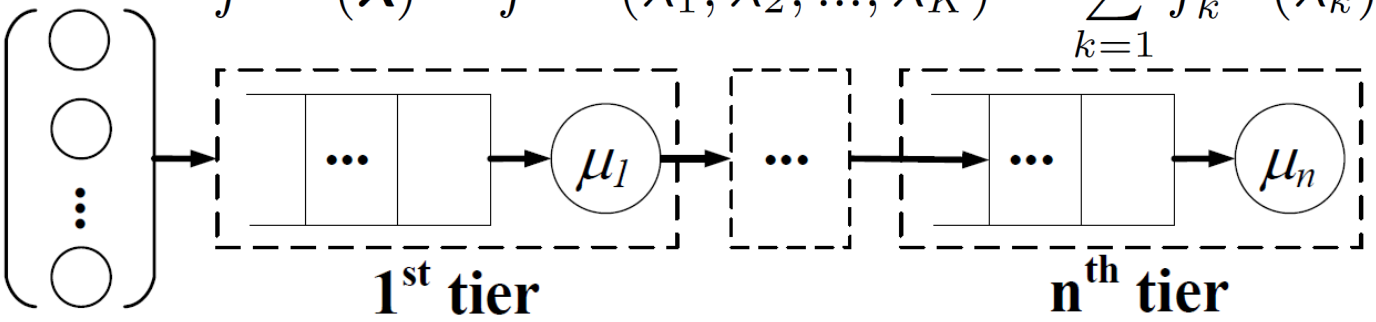
# Modeling flash deal applications

- Single-tier Architecture [1][2][3][4]
  - Request arrival follows Poison process
  - Service time is generally distributed
  - Model the application as an M/G/1/PS queue
  - Response time in queue

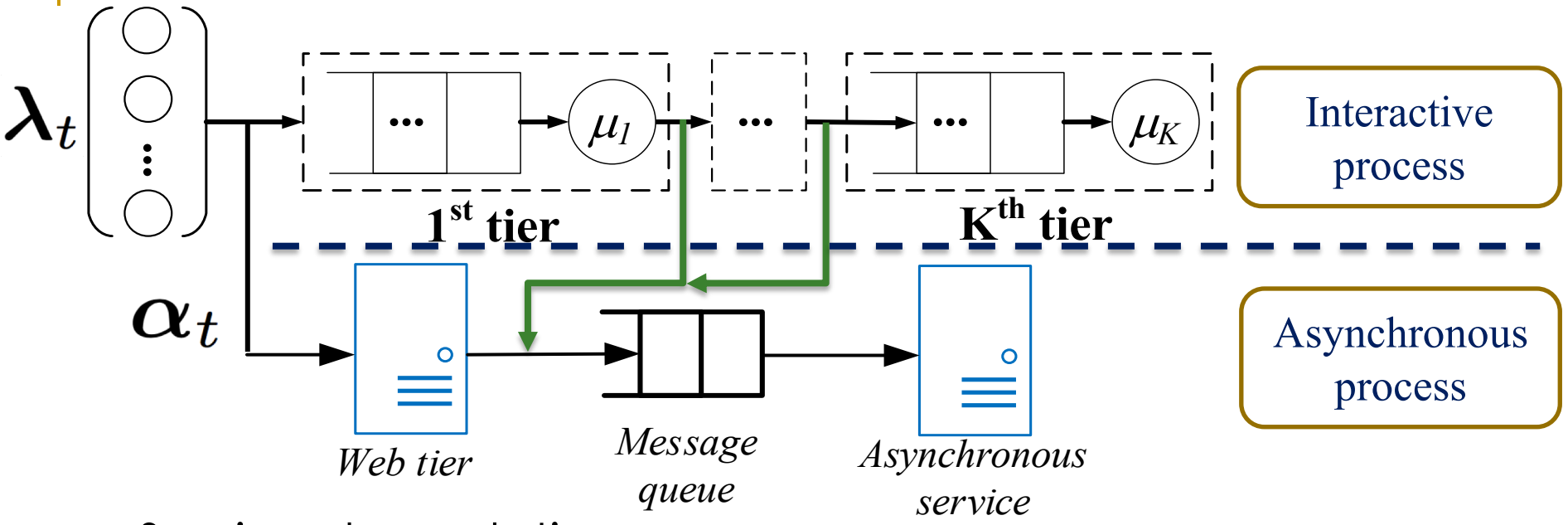
$$f^{ST}(\lambda) = \int_0^{\infty} \frac{x}{1-\rho} dF(x) = \frac{E[X]}{1-\rho}$$

- Multi-tier Architecture [5][6][7]
  - Lemma 1. the arrival rate $\lambda_{k+1} = \lambda_k$ , when the queueing system is stable
  - Response time in queue

$$f^{MT}(\boldsymbol{\lambda}) = f^{MT}(\lambda_1, \lambda_2, ..., \lambda_K) = \sum_{k=1}^{K} f_k^{ST}(\lambda_k)$$



# Extending multi-tier with service degradation




- Service degradation
  - Each message binds to a series of tasks
  - Classify messages into different priority classes $k \in \{1, 2, ..., K\}$
  - Model the asynchronous process as a priority queue

$$f^{AP}(\boldsymbol{\alpha}) = \frac{C \sum_{k=1}^{K} \alpha_k}{(1 - \sum_{k=1}^{K-1} \frac{\alpha_k}{\mu^{AP}})(1 - \sum_{k=1}^{K} \frac{\alpha_k}{\mu^{AP}})}$$

# Evaluating response time

- Private cloud – single tier
  - $\lambda_t^V$ is the number of requests assigned to the private cloud during the $t^{th}$ time slot
  - Model flash deals in private cloud as single-tier architecture
  - Response time can be evaluated as $d_t^V = f^{ST}(\lambda_t^V)$
- Public cloud – multi tier with soft guarantee
  - $\lambda^U$ is the number of requests assigned to the public cloud during the $t^{th}$ time slot
  - Model flash deals in private cloud as multi-tier architecture
  - Interactive process

$$d_t^{IP}(\boldsymbol{\lambda}_t, \boldsymbol{\alpha}_t) = f^{MT}(\boldsymbol{\lambda}_t - \boldsymbol{\alpha}_t)$$

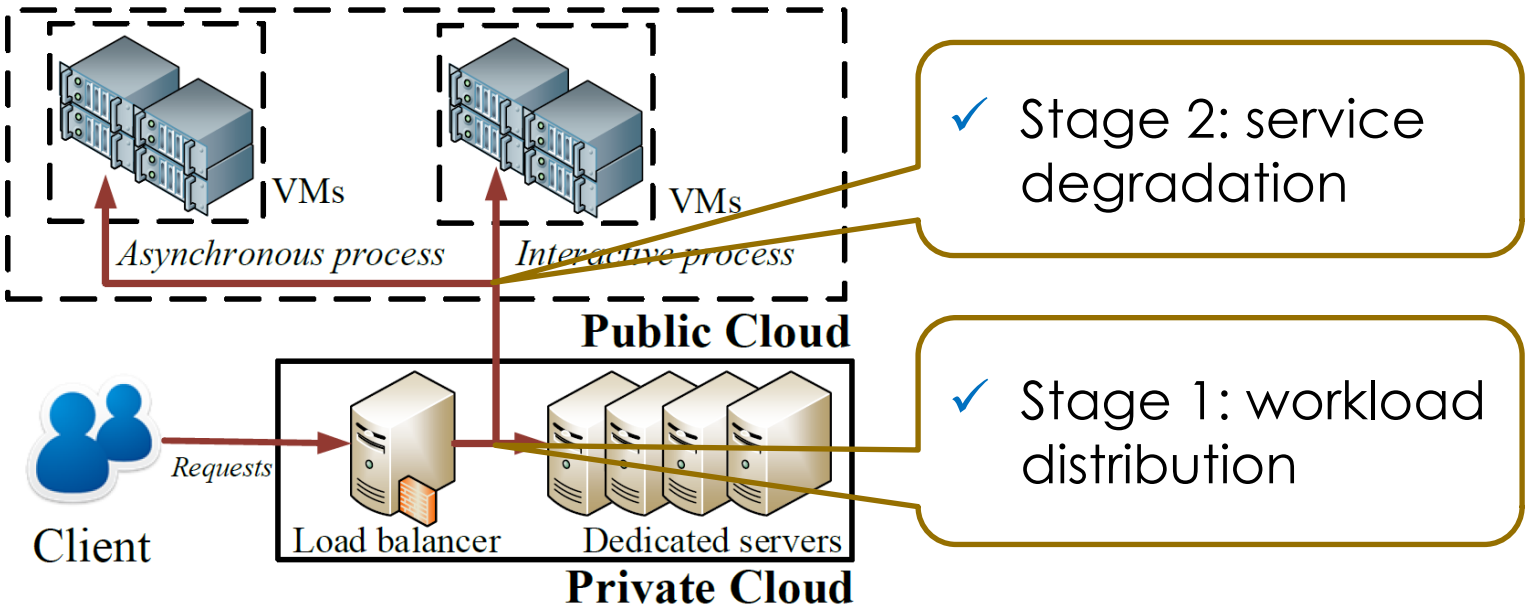
  - Asynchronous process

$$d_t^{SG}(\boldsymbol{\alpha}_t) = f^{AP}(\boldsymbol{\alpha}_t)$$

- Hybrid cloud
  - Response time can be evaluated as

$$d_t^H = \frac{\lambda_t^V}{\lambda_t} d_t^V + \frac{\lambda_t^U}{\lambda_t}(d_t^{IP} + d_t^R) + \frac{\lambda_t - \lambda_t^V - \lambda_t^U}{\lambda_t} \cdot D$$

# Request scheduling problem




- Workload distribution
  - Distributing requests between the private and public clouds

$$\min \quad d_t^H(\lambda_t^V, \lambda_t^U)$$
$$\text{s.t.} \quad \lambda_t^V + \lambda_t^U - \lambda_t \leq 0$$

- Service degradation
  - Assigning partial requests to asynchronous process

$$\min \quad d_t^{IP}(\boldsymbol{\alpha}_t)$$
$$\text{s.t.} \quad d_t^{SG}(\boldsymbol{\alpha}_t) \leq L$$

# Capacity adjusting problem

- We set a budget $b$
- The number of EC2 instances a tenant can boot is $n$
- The decision on leasing n EC2 instances is $\boldsymbol{x}_t^{(n)}$
- Performance-Cost ratio of leasing n EC2 instances

$$\pi_t^{(n)} = \frac{cap_t}{a_t \cdot n_t} = \frac{1}{a_t d_t^H \boldsymbol{J}^{\mathrm{T}} \boldsymbol{x}_t^{(n)}}, \boldsymbol{J} = (1, 2, ..., N)$$

- Problem formulation

Online problem

NP hard

$$\max \quad \sum_{t=1}^{m} \boldsymbol{\pi}_t^{\mathrm{T}} \boldsymbol{x}_t$$

$$\text{s.t.} \quad \sum_{i=1}^{t} \boldsymbol{c}_i^{\mathrm{T}} \boldsymbol{x}_i \leq b, \quad \boldsymbol{c}_i^{\mathrm{T}} = a_i \cdot \boldsymbol{J}^{\mathrm{T}}$$

Controlling cost

$$i = 1, 2, ..., m,$$

$$\boldsymbol{\pi}_t = (\pi_t^{(1)}, \pi_t^{(2)}, ..., \pi_t^{(n)}, ..., \pi_t^{(N)})$$

PC ratio

$$\boldsymbol{x}_t \in \{\boldsymbol{x}_t^{(1)}, \boldsymbol{x}_t^{(2)}, ..., \boldsymbol{x}_t^{(n)}, ..., \boldsymbol{x}_t^{(N)}\}$$

Capacity decision

# Capacity adjusting algorithm

- Define a partial linear problem on

$$\{0, s\}, \text{ where } s = \epsilon m \text{ and } 0 < \epsilon < 1$$

- The partial linear problem

$$\begin{aligned} \max \quad & \sum_{t=0}^{s} \boldsymbol{\pi}_t^T \boldsymbol{x}_t \\ \text{s.t.} \quad & \sum_{t=0}^{s} \boldsymbol{c}_t^T \boldsymbol{x}_t \leq (1-\epsilon)\epsilon b, \\ & \boldsymbol{x}_t \in K, t \in [0, s]. \end{aligned}$$

- The corresponding dual problem

$$\begin{aligned} \min \quad & (1-\epsilon)\epsilon b \cdot p + \sum_{t=0}^{s} y_t \\ \text{s.t.} \quad & c_{tj} \cdot p + y_t \geq \pi_{tj}, j \in [1, N] \\ & p, y_t \geq 0, t \in [0, s]. \end{aligned}$$

  - $(\hat{p}, \hat{\boldsymbol{y}})$ represents the optimal solution to problem

- Decision on capacity adjustment

$$x_{tj}(p) = \begin{cases} 1, & \pi_{tj} > \boldsymbol{p} \cdot \boldsymbol{c}_{tj}, \\ 0, & \pi_{tj} < \boldsymbol{p} \cdot \boldsymbol{c}_{tj}. \end{cases} \quad x_{tj}(p) = \begin{cases} 1, & j = \arg\max_{j \in N} \{\pi_{tj} - p \cdot c_{tj}\}, \\ 0, & else. \end{cases}$$

# Capacity adjusting algorithm

- Two special cases
  - If $\pi_{tj} = p \cdot c_{tj}$
  - If $\pi_{tj} - p \cdot c_{tj} = \pi_{tl} - p \cdot c_{tl}$
- The algorithm becomes ineffective
- Inspired by the existing literature [8][9], we make Assumption 1

**Assumption 1.** *For any $p$, there can be at most 1 column of $\boldsymbol{\pi}_t \in \{\boldsymbol{\pi}_i | i \in [1, m]\}$, i.e., $\pi_{tj}$, such that $\pi_{tj} = p \cdot c_{tj}$ or $\pi_{tj} - p \cdot c_{tj} = \pi_{tl} - p \cdot c_{tl}$.*

- Summary of capacity adjusting algorithm

$$\boldsymbol{x}_t(\hat{p}) \longrightarrow \boldsymbol{x}_t(p^*) \longrightarrow \boldsymbol{x}_t^*$$

Dual problem on [0, s] | Dual problem on [0, m] | Origin problem on [0, m]

# Optimality analysis

- Q1: is $\boldsymbol{x}_t(p^*)$ the same to $\boldsymbol{x}_t^*$ ?

  **Lemma 2.** *For all $t \in [0, m]$, under Assumption 1, $\boldsymbol{x}_t(p^*)$ and $\boldsymbol{x}_t^*$ differs no more than 1 value of $t$.*

- Q2: is $\boldsymbol{x}_t(\hat{p})$ accurate enough as a substitute to $\boldsymbol{x}_t(p^*)$?

  **Lemma 3.** *The primal solution derived using sample dual price $\hat{p}$ is a feasible solution to the linear problem with high probability of $1-\epsilon$ that*

  $$\sum_{t=1}^{m} \boldsymbol{c}_t^T \boldsymbol{x}_t(\hat{p}) \leq b, \text{ given that } b \geq \frac{3m\ln(N/\epsilon)}{\epsilon^3}$$

- Q3: how much is the gap between $\boldsymbol{x}_t(\hat{p})$ and $\boldsymbol{x}_t^*$ ?

  **Lemma 4.** *The primal solution constructed using sample dual price $\hat{p}$ is a near-optimal solutions to the linear problem with high probability of $1-\epsilon$ that*

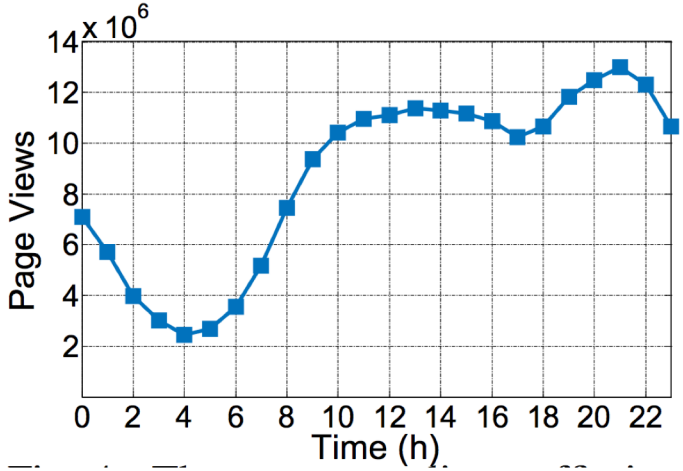
  $$\sum_{t=1}^{m} \boldsymbol{\pi}_t^T \boldsymbol{x}_t(\hat{p}) \geq (1-3\epsilon)OPT, \text{ given that } b \geq \frac{3m\ln(N/\epsilon)}{\epsilon^3}$$

- Q4: how much is the gap between OPT and the algorithm?

  **Proposition 1.** *For any $\epsilon > 0$, the Capacity-Adjusting Algorithm is $1-6\epsilon$ competitive for the online linear problem*

# Evaluation

- **Real world trace**
  - Online traffic in U.S. on Cyber Monday measured by Akamai




- **Testbed**
  - A private cloud on two servers with OpenStack Mitaka
  - A public cloud 20 EC2 large type instances on AWS
- **Implementation**
  - The web tier is deployed by an Apache HTTP server
  - Two Tomcat 9.0 servers as the application tier,
  - Use HttpClient 4.5.2 to generate requests
  - A Servlet querying records of a table from a MySQL database

# Evaluation

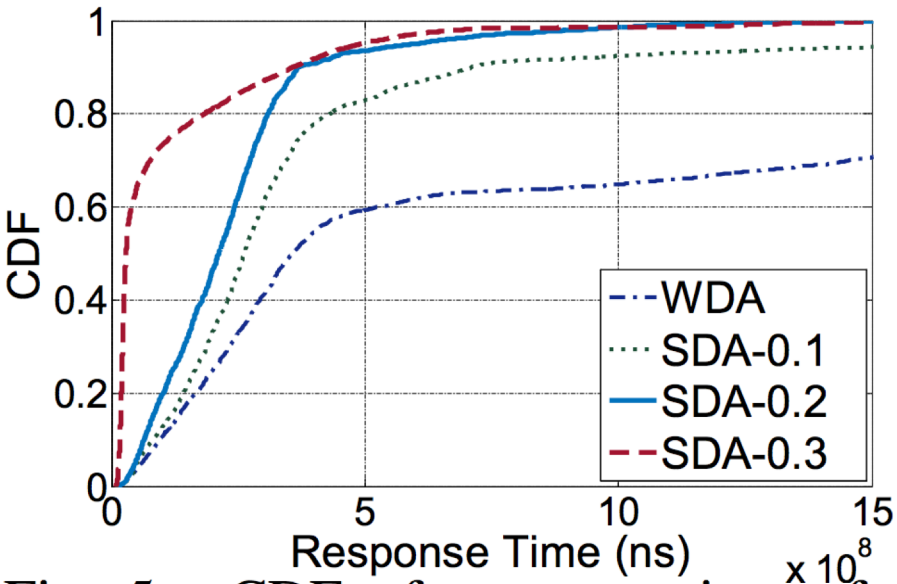



Fig. 5. CDF of response time of service degradation algorithms.

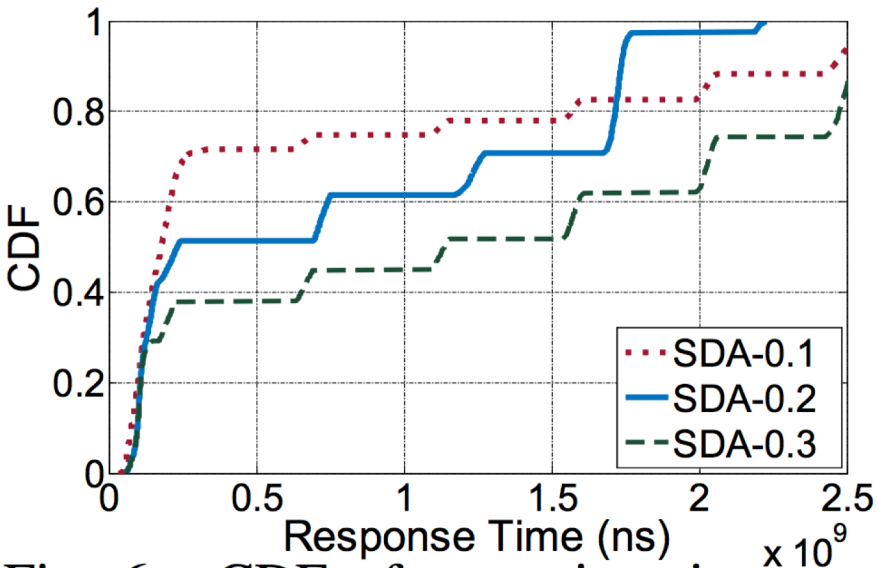



Fig. 6. CDF of execution time among postponed requests.

- Obviously, scheduling more requests to the asynchronous process can reduce response time remarkably
- The response time of the asynchronous process can be controlled within predefined deadline

# Evaluation

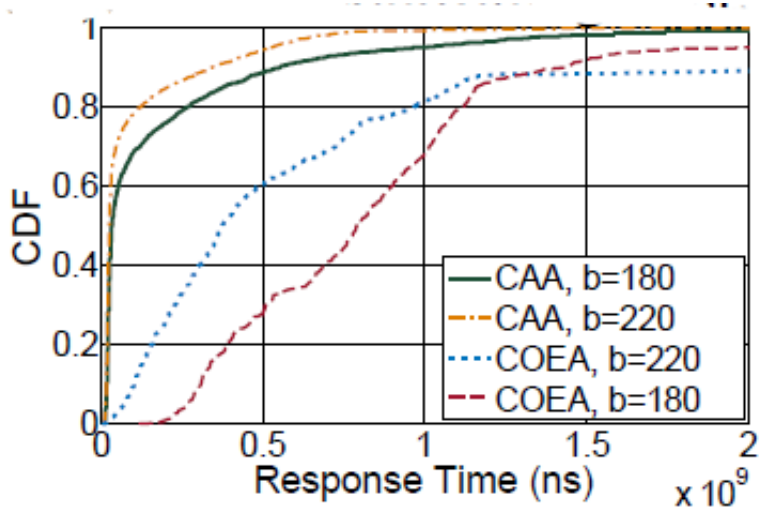



Fig. 12. CDF of response time of CAA and CEOA.

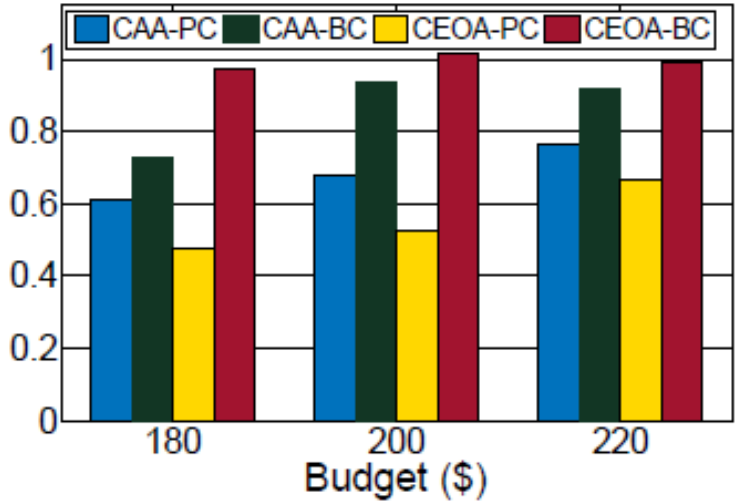



Fig. 13. PC and BC ratios of CAA and CEOA.

- Compared with CEOA, CAA reduces response time by 15% and improves the PC ratio by 19% on average, respectively

# Conclusion

- We  proposed a solution for flash deal applications to withstand flash crowds in a hybrid cloud
- Concerning scheduling requests, we achieved fast response time of the interactive process as well as guaranteed requests served in the asynchronous process within a predefined deadline
- In terms of adjusting capacity, we tuned scale of the public cloud with the objectives of performance-cost ratio maximization as well as outsourcing cost minimization.
- Compared with previous work, our solution reduced response time by 15% on average and effectively maintained cost within the budget.

# Reference


| No. | Paper | Source |
|---|---|---|
| [1] | Modeling differentiated services of multi-tier web applications | MASCOTS06 |
| [2] | Preserving qos of e- commerce sites through self-tuning: A performance model approach | EC ’01 |
| [3] | Autonomous resource provisioning for multi-service web applications | WWW’10 |
| [4] | Provisioning Servers in the Application Tier for E-commerce Systems | IWQoS’04 |
| [5] | Agile dynamic provisioning of multi-tier internet applications | TAAS |
| [6] | Con- trolling quality of service in multi-tier web applications | ICDCS’06 |
| [7] | An analytical model for multi-tier internet services and its applications | SIGMETRICS’05 |
| [8] | The adwords problem: Online keyword matching with budgeted bidders under random permutations | EC’05 |
| [9] | A dynamic near-optimal algorithm for online linear programming | Operations research |

# Q&A

*Thank You!*

***"Cloud Datacenter & Green Computing"*** *Research Group*
*Huazhong University of Science & Technology*

http://grid.hust.edu.cn/fmliu/
fmliu@hust.edu.cn