# Proactive VNF Provisioning with Multi-timescale Cloud Resources: Fusing Online Learning and Online Optimization

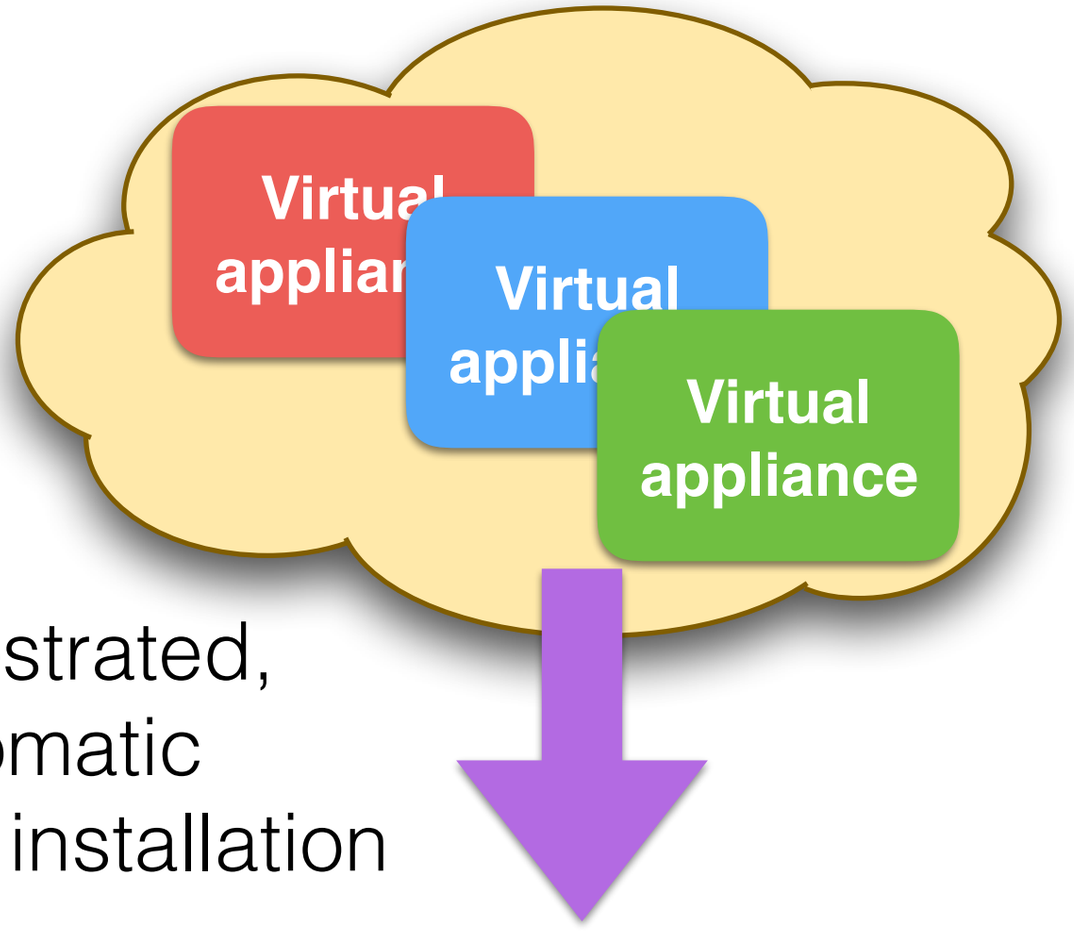
Xiaoxi Zhang[†], Chuan Wu[†], Zongpeng Li[§], Francis C.M. Lau[†]

[†]The University of Hong Kong

[§] University of Calgary

INFOCOM'17 // 1-4 May 2017 // Atlanta, GA, USA

# Network Function Virtualization (NFV)

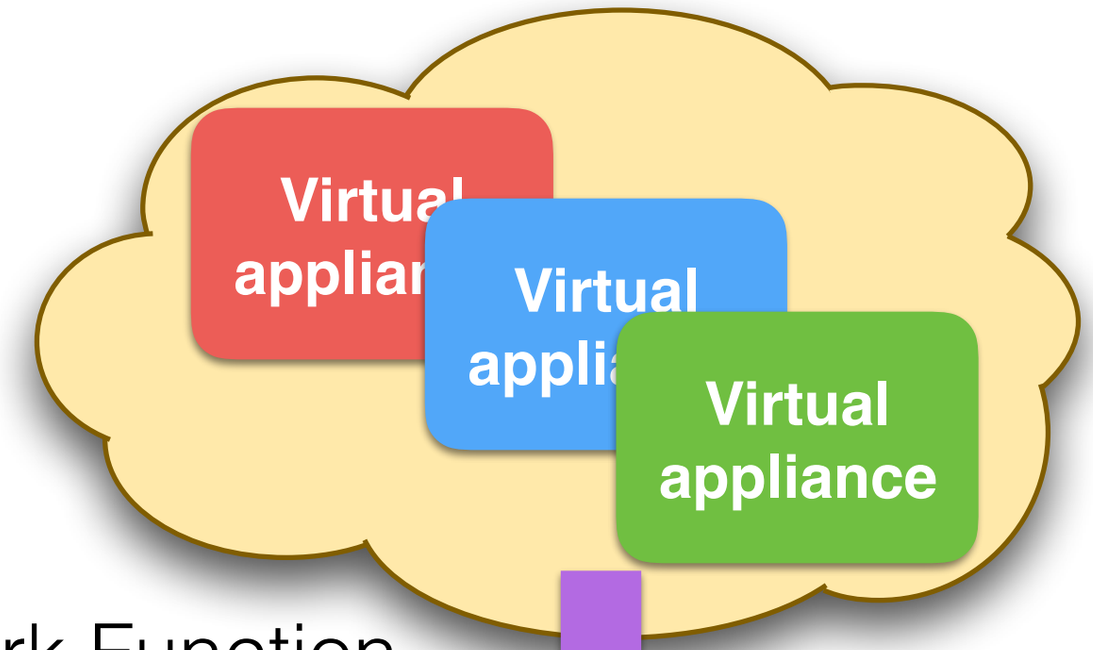



Orchestrated,
automatic
& remote installation

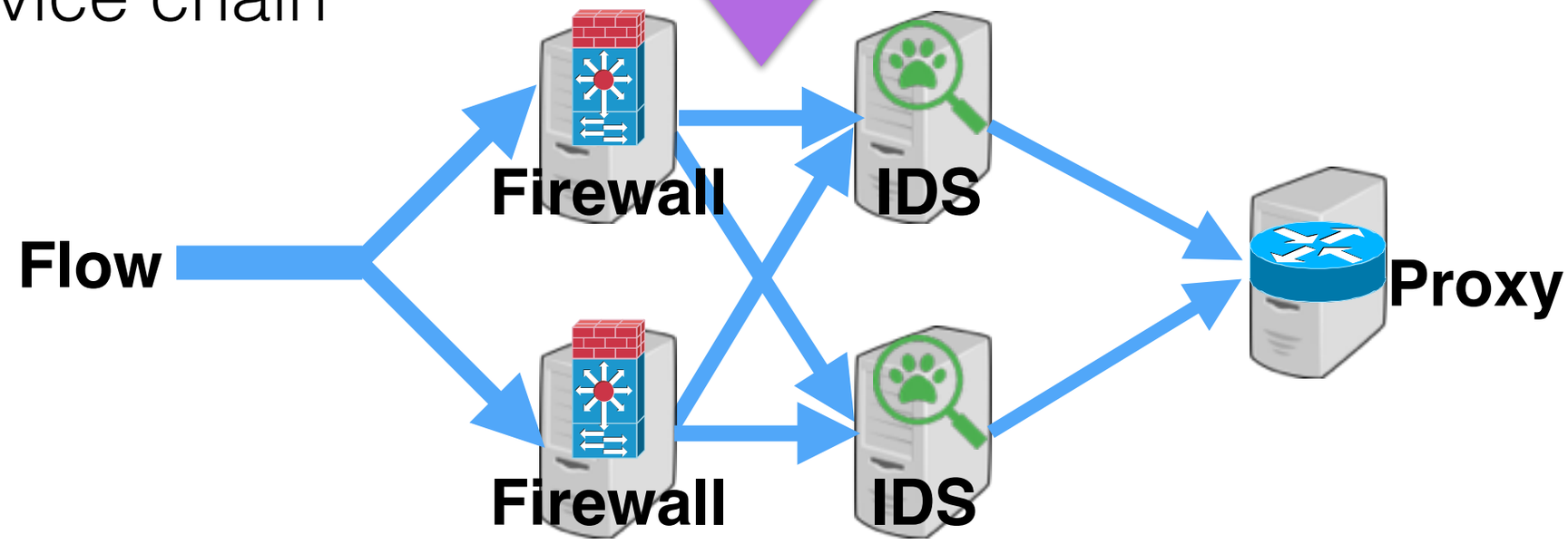
standard VMs

standard servers

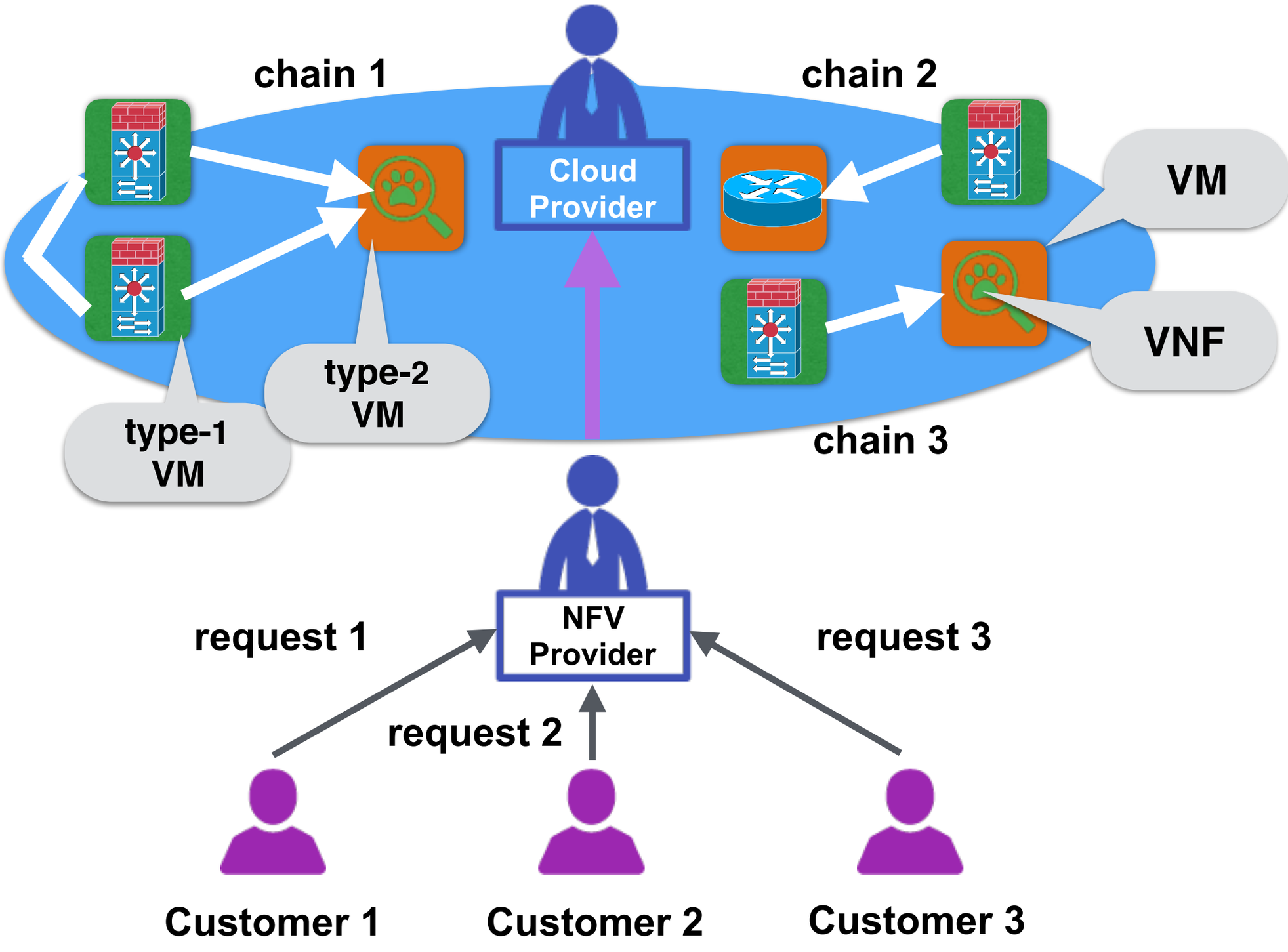
# Network Function Virtualization (NFV)




Virtualized Network Function (VNF) service chain



# NFV brokerage service



# Related work

- Online reactive VNF provisioning [Ghaznavi et al. 2015]

  This work targets a proactive approach

- Approximate algorithms for offline VNF provisioning [Cohen et al. 2015]

  This work targets an online VNF provisioning

- Heuristics considering VNFs deployed on homogeneous servers [Moens et al. 2014]

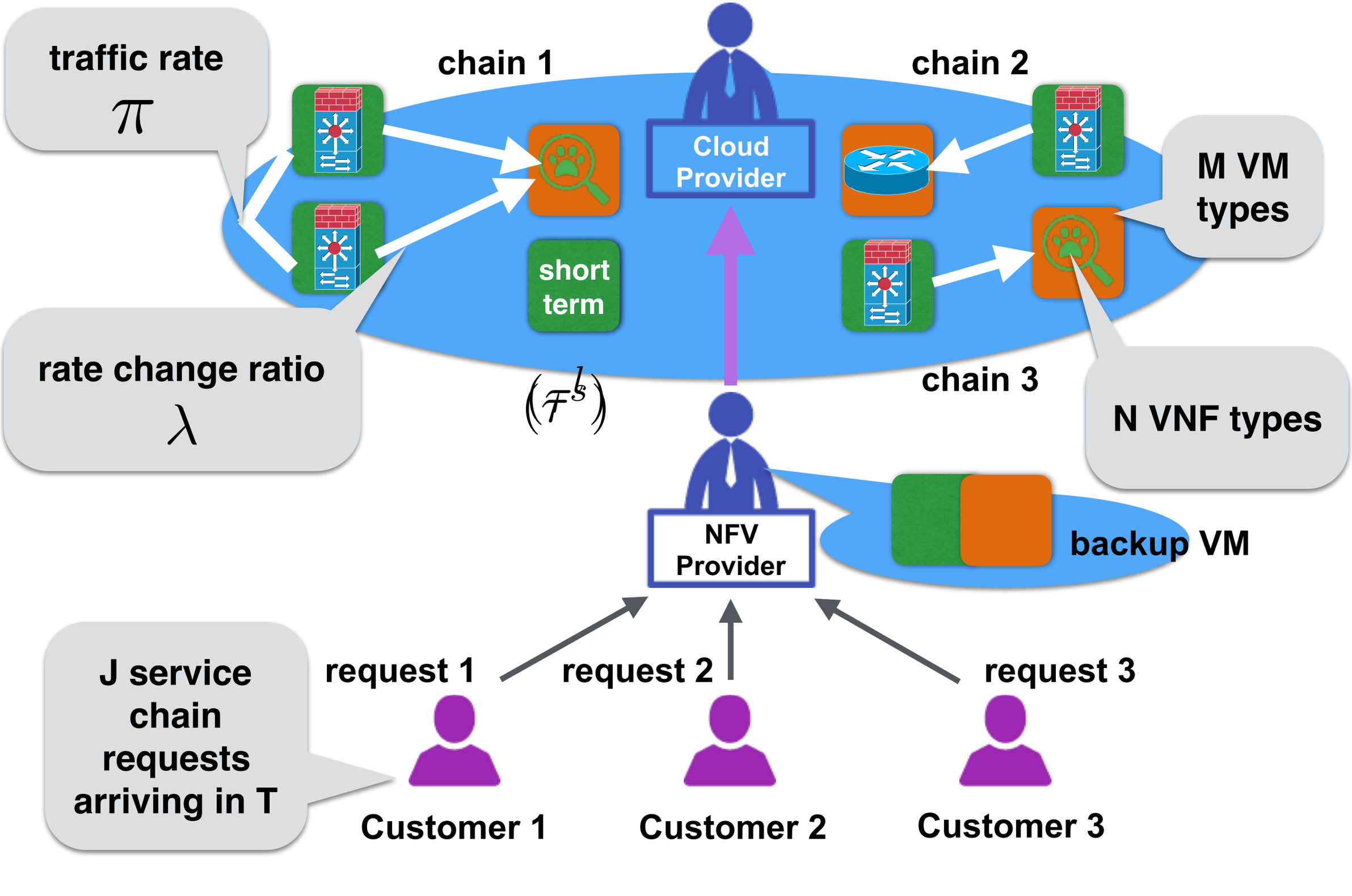
  This work considers two timescales of VM instances

M. Ghaznavi et al., "Elastic Virtual Network Function Placement," in *Proc. of* IEEE CloudNet, 2015.

H. Moens et al., "VNF-P: A Model for Efficient Placement of Virtualized Network Functions," in *Proc. of CNSM*, 2014

. R. Cohen et al., "Near Optimal Placement of Virtual Network Functions," in *Proc. of IEEE INFOCOM*, 2015

# Problem model



# An online proactive approach

Minimizing total cost for the NFV provider to meet customers' traffic demands, which needs to address:

- How to predict the # of VMs needed in the next time slot?
- How many VNFs of each type should be created?
- When and how many VMs of each type should be purchased?
- VMs should be purchased in long term or short term?

# Problem formulation

$$\text{minimize} \sum_{t\in[T]} \sum_{m\in[M]} \{\alpha_m u_m(t) + \beta_m y_m(t) + d_m(y_m(t) + [u_m(t) - u_m(t-\tau^s)]^+) + s_m(t) w_m(t)\}$$

cost for purchasing long-term VMs

cost of deploying VNFs on type-m VMs

cost for using backup VMs

cost for purchasing short-term VMs

**Online decision variables:**

- $x_n(t)$: # of type-n **VNFs** active at t
- $u_m(t)$: # of type-m VMs in **short** term purchased at t
- $y_m(t)$: # of type-m VMs in **long** term purchased at t
- $w_m(t)$: # of type-m **backup** VMs used at t

# Problem formulation

$$\text{minimize} \sum_{t\in[T]} \sum_{m\in[M]} \{\alpha_m u_m(t) + \beta_m y_m(t) + d_m(y_m(t) + [u_m(t) - u_m(t-\tau^s)]^+) + s_m(t) w_m(t)\}$$

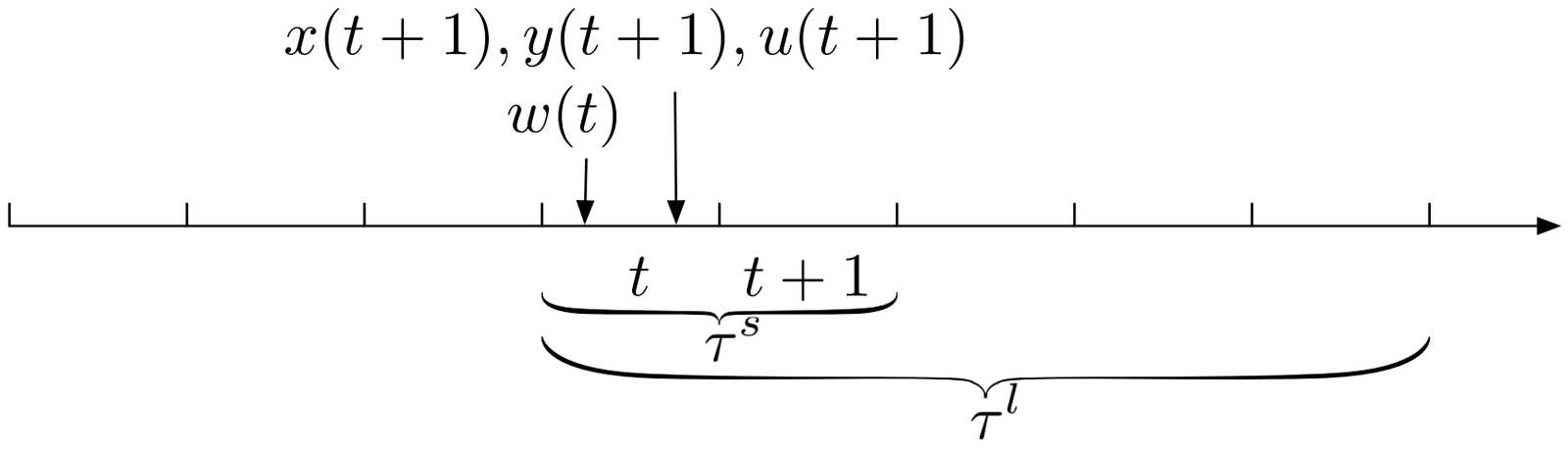
cost for purchasing long-term VMs

cost of deploying VNFs on type-m VMs

cost for using backup VMs

cost for purchasing short-term VMs

**Online decision variables:**



# Problem formulation

$$\text{minimize} \sum_{t\in[T]} \sum_{m\in[M]} \{\alpha_m u_m(t) + \beta_m y_m(t)$$

$$+d_m(y_m(t) + [u_m(t) - u_m(t-\tau^s)]^+) + s_m(t)w_m(t)\}$$

**# of VMs needed <= total # of VMs (purchased + backup)**

$$s.t. \quad \sum_{n\in[N]} x_n(t)\mathbf{1}_n^m \leq \sum_{t-\tau^l+1}^{t} y_m(k) + \sum_{t-\tau^s+1}^{t} u_m(k) + w_m(t), \ \forall m \in [M], t$$

**flow rate coming into VNF n <= total capacity of deployed VNFs**

$$\sum_{j\in[J]:\ t_j^- \leq t \leq t_j^+, n\in\mathbb{L}^j} z_{n^-}^n(j,t) \leq C_n x_n(t), \forall n \in [N], t$$

**flow rate coming into 1st VNF of each chain**

$$z_0^{n_1}(j,t) = \pi_j(t), \forall j \in [J], t$$

**flow rate from type-n VNFs of to next-hop type-n⁺ VNFs**

$$z_n^{n^+}(j,t) = \lambda_n^j(t) z_{n^-}^n(j,t), \ \forall j \in [J], n \in \mathbb{L}^j, t$$

$$z_{n^-}^n(j,t) \geq 0, \quad \forall j \in [J], n \in \mathbb{L}^j, t$$

# Problem formulation

$$\text{minimize} \sum_{t\in[T]} \sum_{m\in[M]} \{\alpha_m u_m(t) + \beta_m y_m(t) + d_m(y_m(t) + [u_m(t) - u_m(t-\tau^s)]^+) + s_m(t) w_m(t)\}$$

$$s.t. \quad \sum_{n\in[N]} x_n(t)\mathbf{1}_n^m \le \sum_{t-\tau^l+1}^{t} y_m(k) + \sum_{t-\tau^s+1}^{t} u_m(k) + w_m(t), \ \forall m \in [M], t$$

**# of type-n VNFs needed at t+1**

**cumulated flow rate change ratio of VNF n of chain j at t**

$$x_n(t+1) = \left\lceil \frac{\sum_{j\in[J]:t_j^- \le t+1 \le t_j^+, n\in\mathbb{L}^j} \hat{\lambda}_n^j(t+1)\pi_j(t+1)}{C_n} \right\rceil, \quad \forall n \in [N], t \in [T-1]$$

**processing capacity of a type-n VNF**

$$z_{n^-}^n(j, t+1) = \hat{\lambda}_n^j(t+1)\pi_j(t+1), \quad \forall n \in [N], j \in [J], t \in [T-1]$$

**flow rate coming from previous-hop VNFs of n to type-n VNFs**

$$w_m(t), y_m(t), u_m(t), x_n(t) \in \mathbb{Z}_+, \quad \forall n \in [N], m \in [M], t$$

# Problem formulation

$$\text{minimize} \sum_{t\in[T]} \sum_{m\in[M]} \{\alpha_m u_m(t) + \beta_m y_m(t) + d_m(y_m(t) + [u_m(t) - u_m(t-\tau^s)]^+) + s_m(t) w_m(t)\}$$

$$s.t. \quad \sum_{n\in[N]} x_n(t)\mathbf{1}_n^m \leq \sum_{t-\tau^l+1}^{t} y_m(k) + \sum_{t-\tau^s+1}^{t} u_m(k) + w_m(t), \ \forall m \in [M], t$$

**# of type-n VNFs needed at t+1**

**Unavailable at t**

$$x_n(t+1) = \left\lceil \frac{\sum_{j\in[J]:t_j^- \leq t+1 \leq t_j^+, n\in\mathbb{L}^j} \hat{\lambda}_n^j(t+1)\pi_j(t+1)}{C_n} \right\rceil, \quad \forall n \in [N], t \in [T-1]$$

**processing capacity of a type-n VNF**

$$z_{n^-}^n(j, t+1) = \hat{\lambda}_n^j(t+1)\pi_j(t+1), \quad \forall n \in [N], j \in [J], t \in [T-1]$$

**flow rate coming from previous-hop VNFs of n to type-n VNFs**

$$w_m(t), y_m(t), u_m(t), x_n(t) \in \mathbb{Z}_+, \quad \forall n \in [N], m \in [M], t$$

# Proactive Online VNF Provisioning Algorithm (POLAR)

- An effective approach to predict the needed # of VNFs ( $x_n(t+1)$ ) to handle upcoming traffic

- An efficient online algorithm to decide VM purchasing, given the prediction of needed VNFs

# Proactive Online VNF Provisioning Algorithm (POLAR)

- An effective approach to predict the needed # of VNFs ( $x_n(t+1)$ ) to handle upcoming traffic

**Randomization + online gradient decent method**

- An efficient online algorithm to decide VM purchasing, given the prediction of needed VNFs

**A two scale online ski-rental friendly algorithm**

# Prediction of needed VNFs

$$\text{minimize}_{x_n(t)\in\mathbb{Z}_+,\forall t\in[T]} \sum_{t\in[T]} |x_n(t) - x_n^*(t)|$$

Non-convex !

prediction

real VM demand

$$f_{nt}(\theta_n(t)) = |\theta_n(t) - x_n^*(t)|, \quad \forall n \in [N], t \in [T],$$

**loss function**

**OGD:** $\theta_n(t+1) = \theta_n(t) - \eta \nabla f_{nt}(\theta_n(t))$

$$\mathcal{D}_{x_n(t)}(\theta_n(t)) = \begin{cases} Pr[x_n(t) = \lfloor \theta_n(t) \rfloor + 1] = \theta_n(t) - \lfloor \theta_n(t) \rfloor \\ Pr[x_n(t) = \lfloor \theta_n(t) \rfloor] = 1 - \theta_n(t) + \lfloor \theta_n(t) \rfloor \end{cases}$$

Randomly deciding $x_n(t)$ by rounding

# Regret of prediction module

$$E[\sum_{t\in[T]}\sum_{n\in[N]}|x_n(t)-x_n^*(t)|]-\min_{\bar{\theta}_n\geq 0}\sum_{t\in[T]}\sum_{n\in[N]}E_{x_n(t)\sim\mathcal{D}_{x_n(t)}(\bar{\theta}_n)}[|x_n(t)-x_n^*(t)|]$$

$$\leq \frac{2\sqrt{T}}{\sqrt{2}-1}\sum_{n\in[N]}x_n^{*max}$$

sub-linear with T

Maximal number of type-n VNFs needed at any time

# Online algorithm for VM purchasing

Given # of VMs needed (including estimations for t+1)

# of VMs needed - active VMs

>0

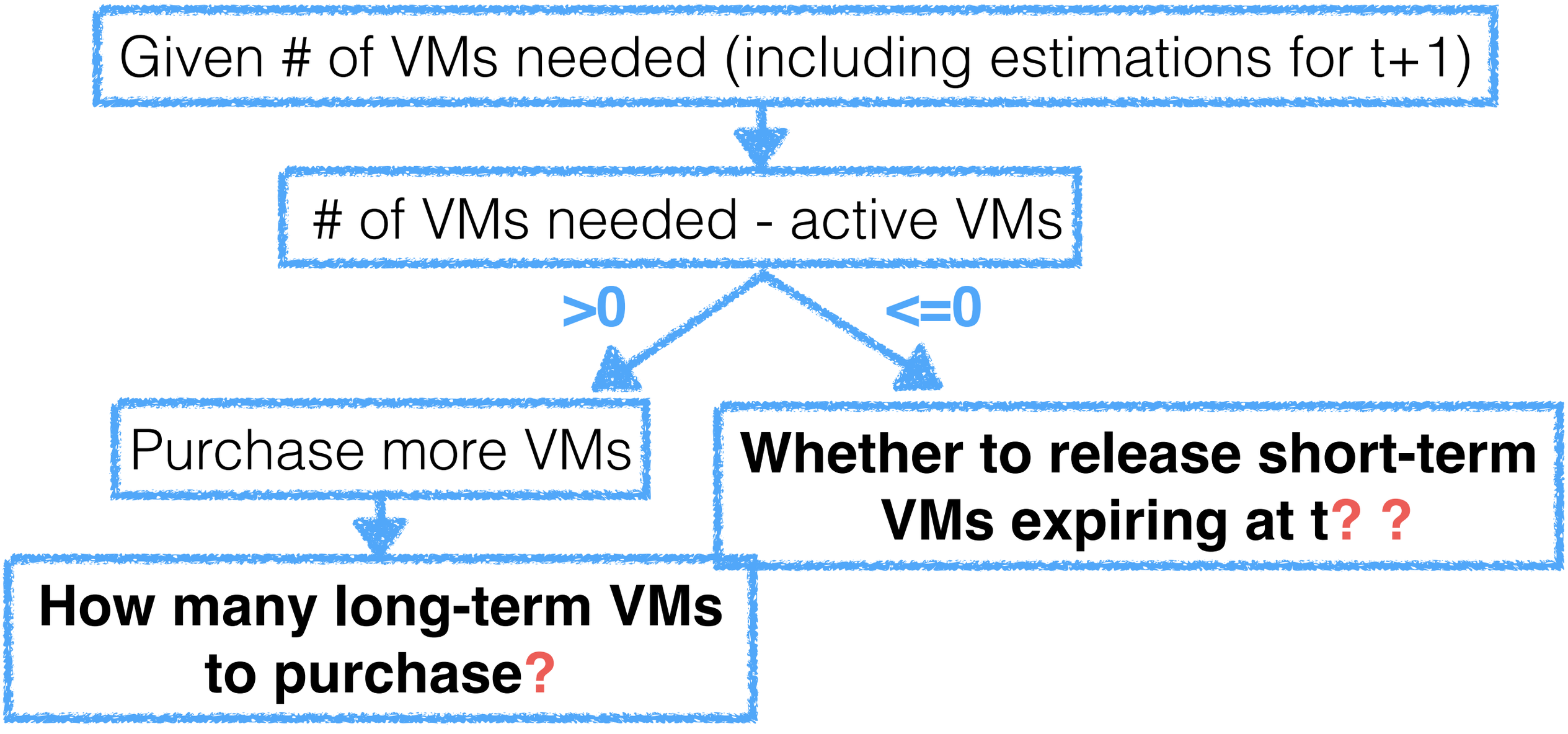
<=0

Purchase more VMs

How many long-term VMs to purchase?

Whether to release expiring short-term VMs ? ?

# Online algorithm for VM purchasing



**Whether to release short-term VMs expiring at t? ?**

**How many long-term VMs to purchase?**

# of VMs needed - active VMs

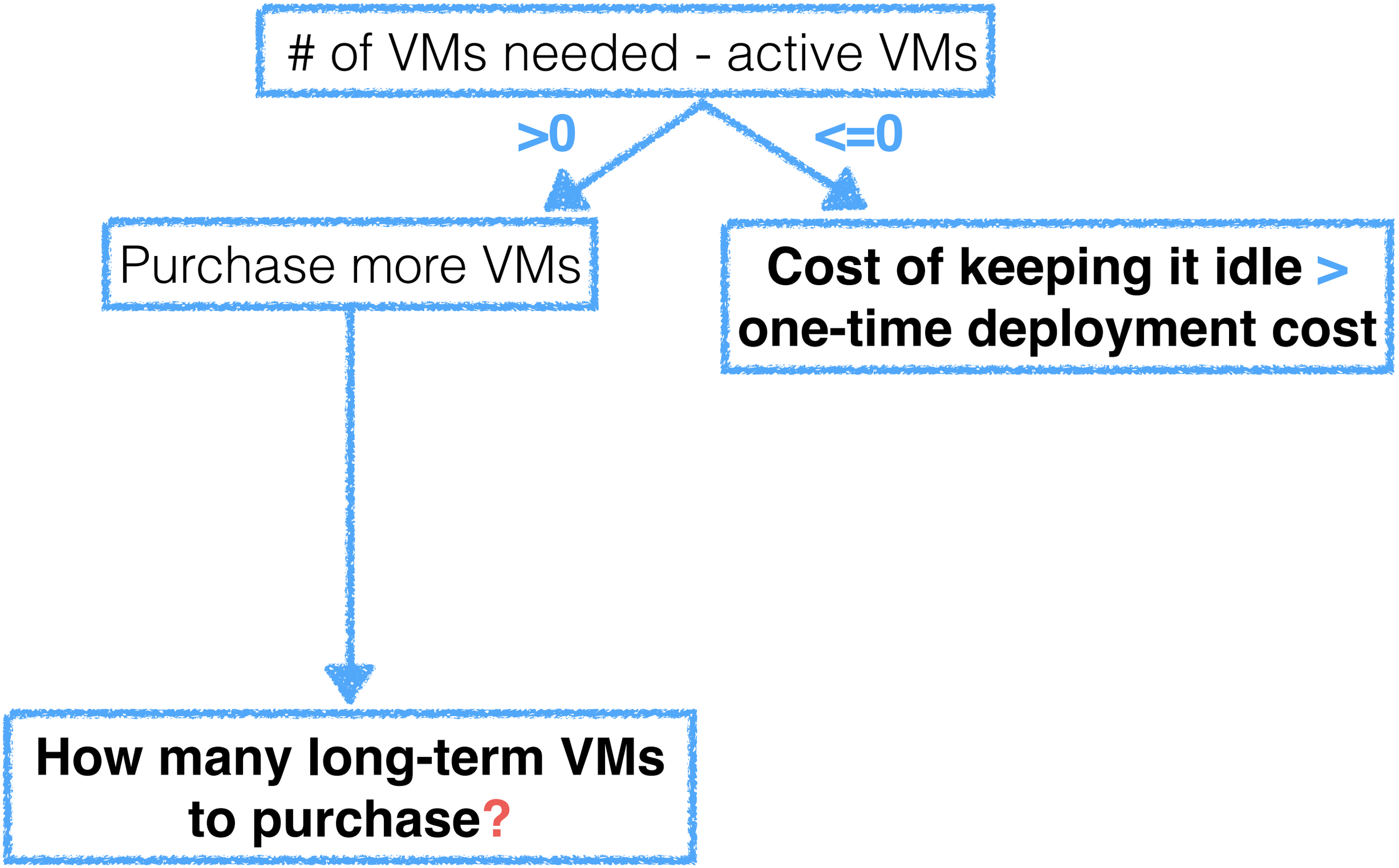
>0

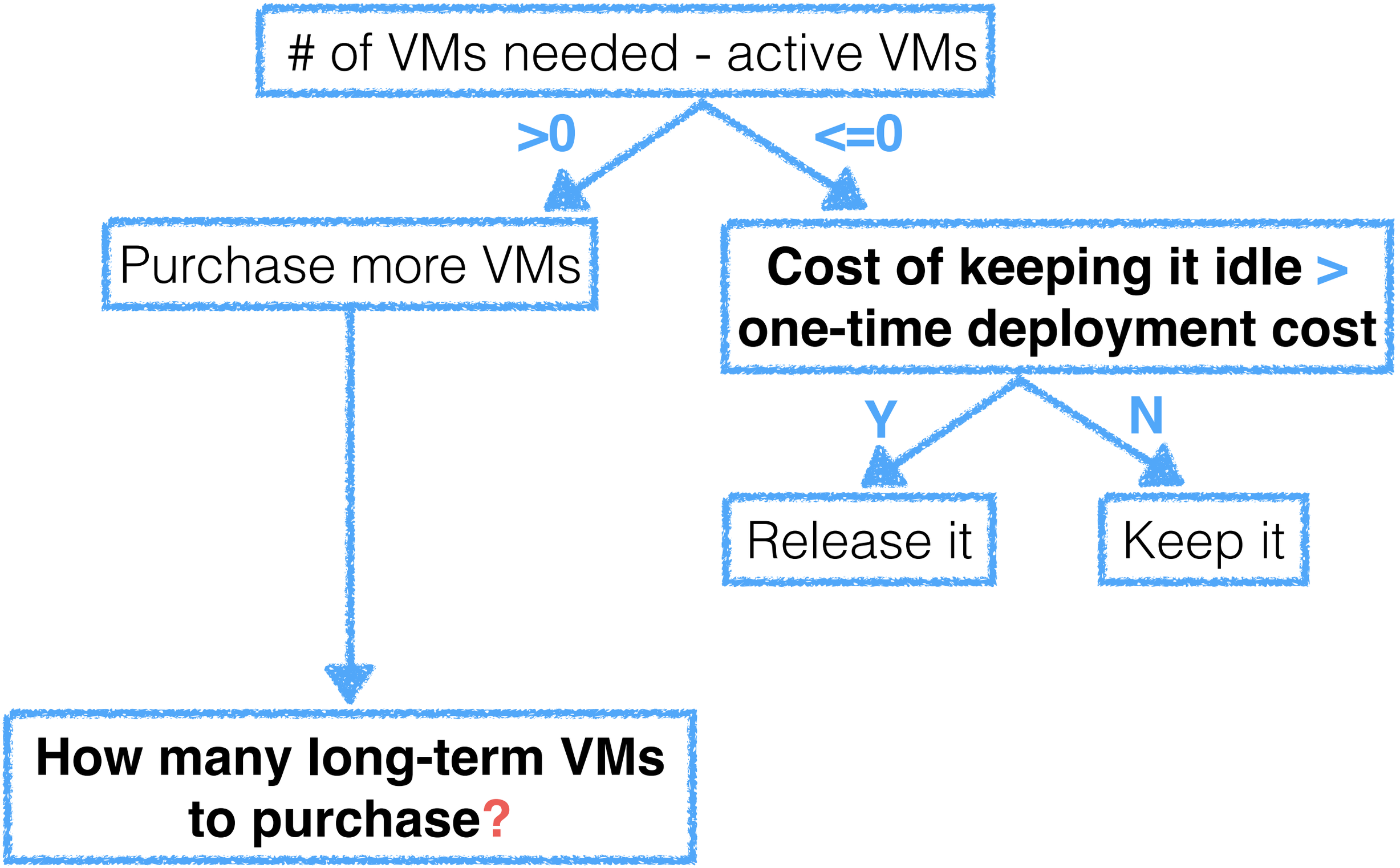
<=0

Purchase more VMs

**Whether to release short-term VMs expiring at t? ?**

**How many long-term VMs to purchase?**


# of VMs needed - active VMs
>0
<=0
Purchase more VMs
Cost of keeping it idle > one-time deployment cost
How many long-term VMs to purchase?


# of VMs needed - active VMs
>0
<=0
Purchase more VMs
Cost of keeping it idle > one-time deployment cost
Y
N
Release it
Keep it
How many long-term VMs to purchase?

# of VMs needed - active VMs

>0

<=0

Purchase more VMs

**How many long-term VMs to purchase?**

**Cost of keeping it idle > one-time deployment cost**

Y

N

Release it

Keep it

# of VMs needed - active VMs

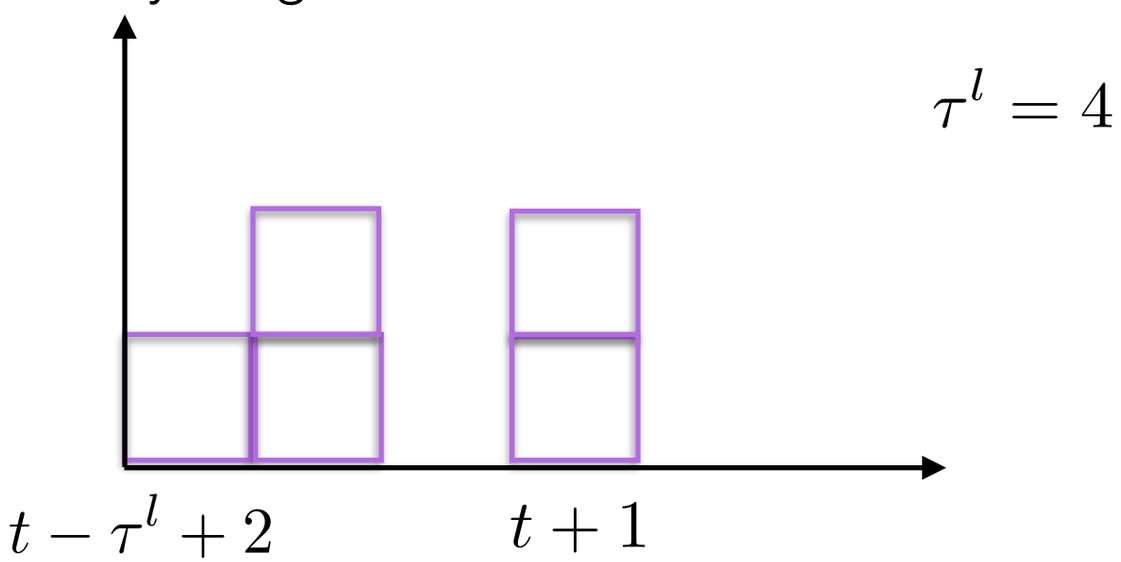
>0

<=0

Purchase more VMs

**How many long-term VMs to purchase?**

# of VMs needed - # of those covered by long term VM instances




**Cost of keeping it idle > one-time deployment cost**

Y

N

Release it

Keep it

# of VMs needed - active VMs

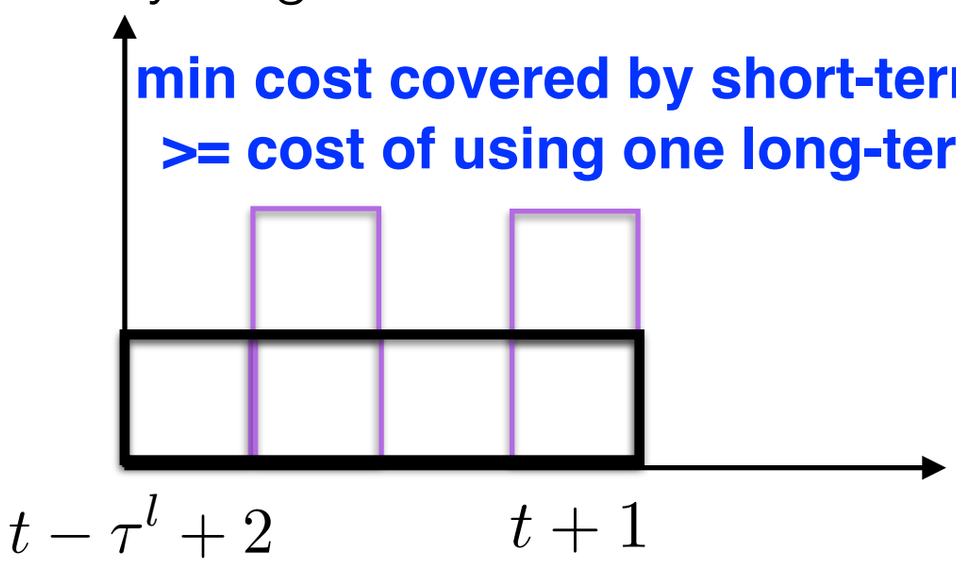
>0

<=0

Purchase more VMs

**How many long-term VMs to purchase?**

# of VMs needed - # of those covered by long term VM instances




**Cost of keeping it idle > one-time deployment cost**

Y

N

Release it

Keep it

# of VMs needed - active VMs

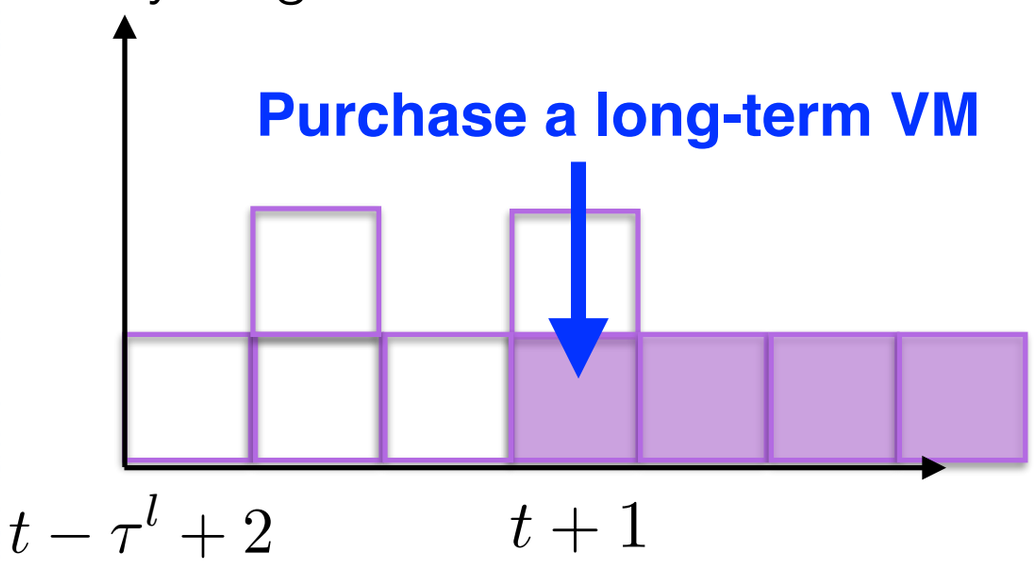
>0

<=0

Purchase more VMs

**How many long-term VMs to purchase?**

# of VMs needed - # of those covered by long term VM instances




**Cost of keeping it idle > one-time deployment cost**

Y

N

Release it

Keep it

# of VMs needed - active VMs

>0

<=0

Purchase more VMs

**Cost of keeping it idle > one-time deployment cost**

Y

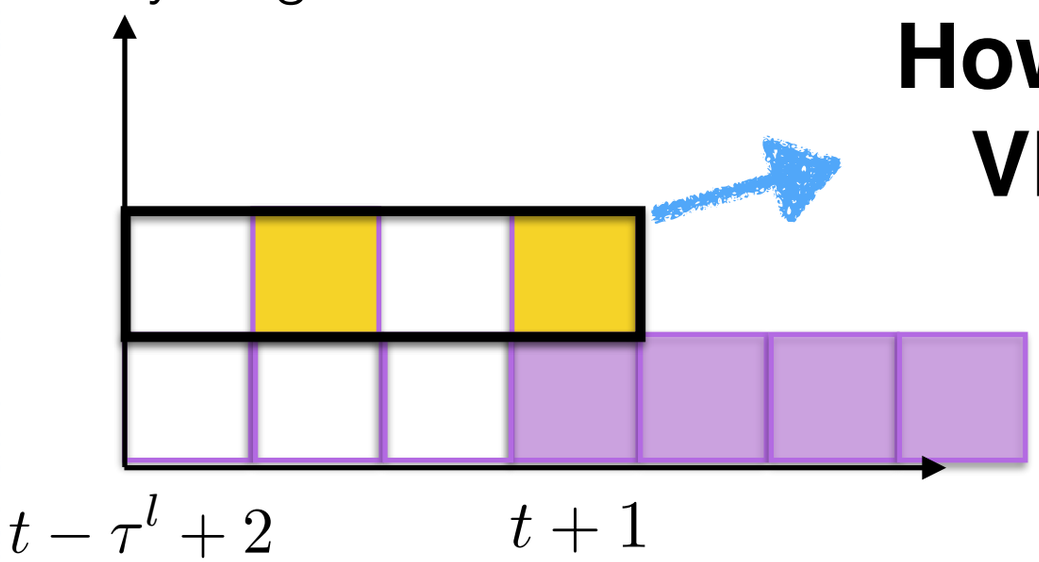
N

Release it

Keep it

**How many long-term VMs to purchase?**

# of VMs needed - # of those covered by long term VM instances




**How many short-term VMs to purchase?**

**How many short-term VMs to purchase ?**

Expiring short term VMs are insufficient

Expiring short term VMs are sufficient

Renew all expiring ones and purchase more short term VMs

Renew iff cost of keeping it idle <= one-time deployment cost

# Competitive ratio of online VM purchasing

- **Under precise estimation** of the # of VMs needed, the online algorithm is 3-competitive

# Main results

- Total error of prediction at most $\frac{2\sqrt{T}}{\sqrt{2}-1}\sum_{n\in[N]} x_n^{*max}$
- 3-competitive online VM purchasing strategy under precise estimation of the # of VMs needed
- A semi-competitive ratio which approaches 3 asymptotically

$$\text{semi-competitive ratio} = \frac{\text{cost (our proactive approach)}}{\text{cost (optimal VM purchasing strategy with best static prediction)}}$$

# Main results

- Total error of prediction at most $\frac{2\sqrt{T}}{\sqrt{2}-1}\sum_{n\in[N]} x_n^{*max}$
- 3-competitive online VM purchasing strategy under precise estimation of the # of VMs needed
- A semi-competitive ratio of POLAR which approaches 3 asymptotically

$$3\left(1+\max_{m\in[M]}\frac{2s_m^{max}\chi\tau^l}{(\sqrt{2}-1)(\beta_m+d_m)\sqrt{T}}\right)$$

$$\chi = \frac{\text{maximal number of VMs needed at any time}}{\text{minimal number of VMs needed at any time}}$$

# Performance evaluation

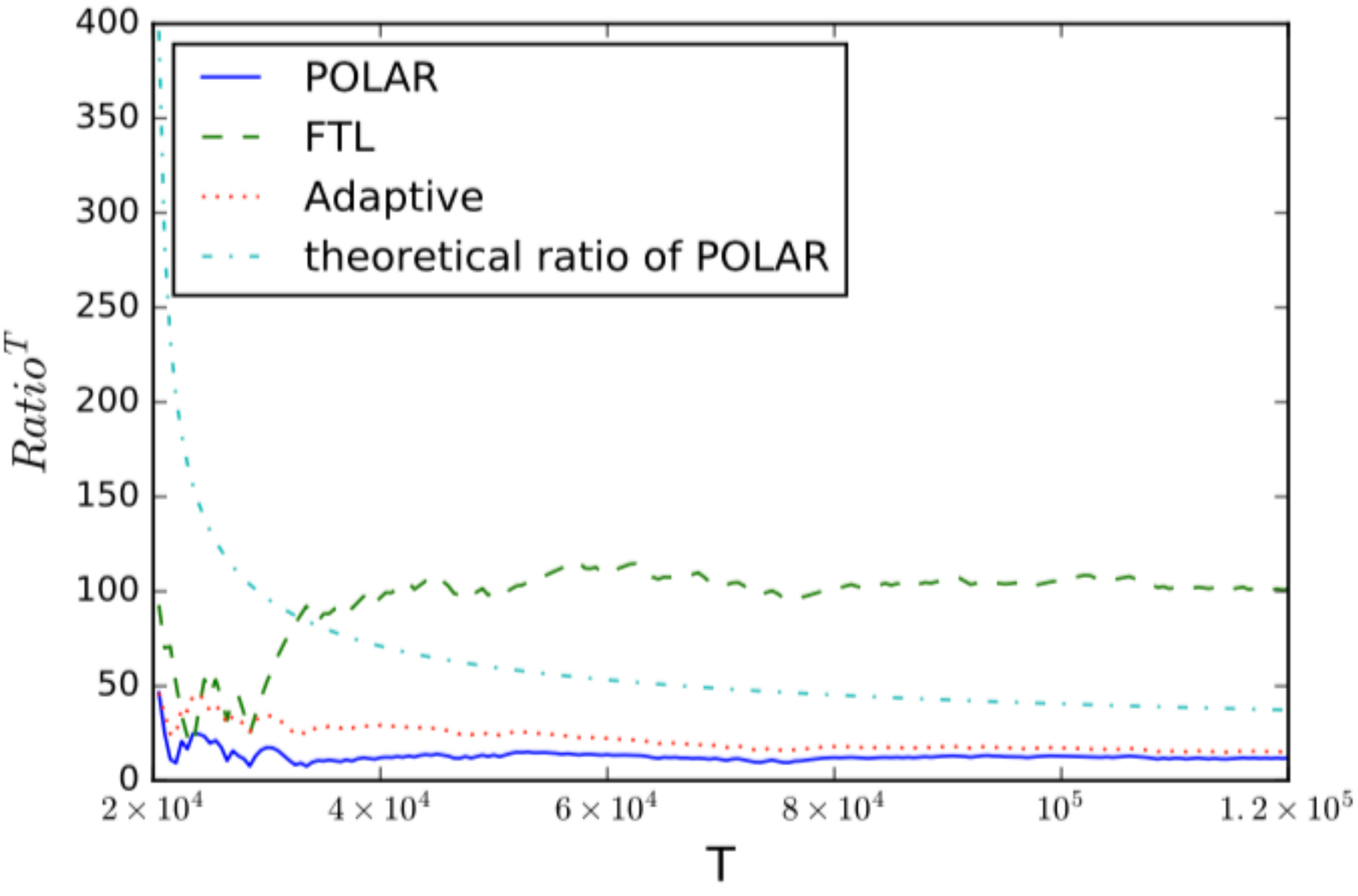
- Simulation setup

  - Flow rates of service chains in time slot t: aggregate size of requested files in time slot t with same IP address in the AWS trace[1]

  - Cost parameters, $\tau^s$, $\tau^l$: Amazon EC2 instances

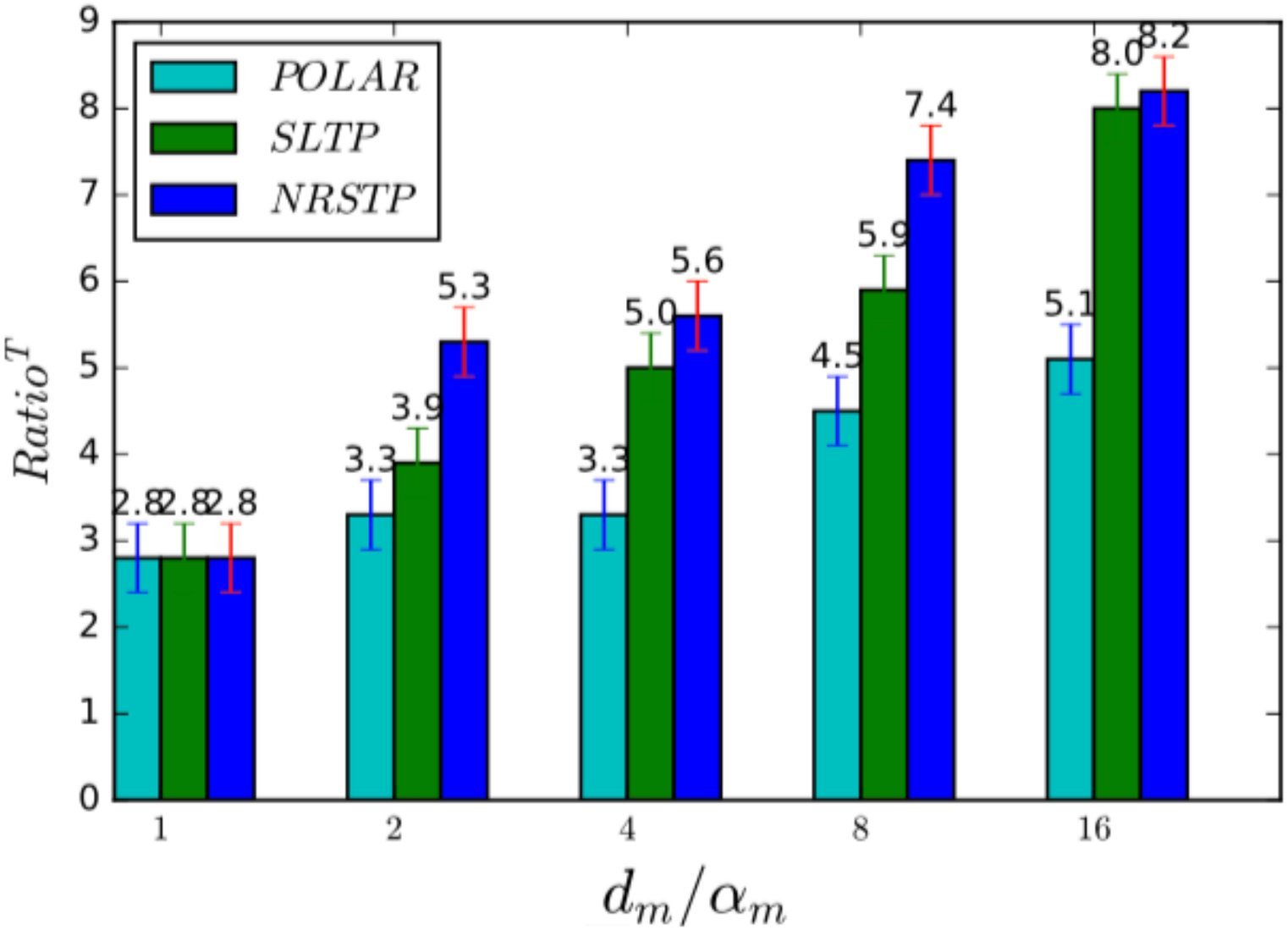
1. *s3://us-east-1.elasticmapreduce.samples*

# Comparison with heuristics




Performance ratio with different prediction module

# Comparison with heuristics




Performance ratio with different online VM purchasing algorithm

# Q&A

Thank you!