Spock: Exploiting Serverless Functions for SLO and Cost Aware Resource Procurement in Public Cloud

Jashwant Gunasekaran, Prashanth Thinakaran,
Mahmut Kandemir, Bhuvan Urgaonkar, George Kesidis,
Chita Das

Computer Science and Engineering
The Pennsylvania State University

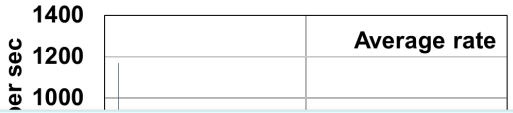
PennState

Outline

- Elastic Web Services.
- VM-based Resource Procurement.
- Serverless Functions.
- Cost of VMs vs Cloud Functions.
- Spock Hybrid Elastic Scaling.
- Implementation and Evaluation.
- Results.

Elastic Web Services

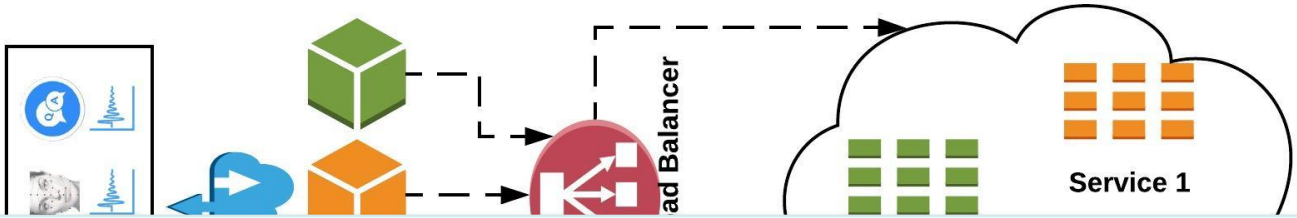
- Short lived queries
  - Strict SLO.



Typical example.?
ML based web services

- Resources Required
  - acquired/released on demand.
  - Average to Peak ratio is high.

ML Inference Engine

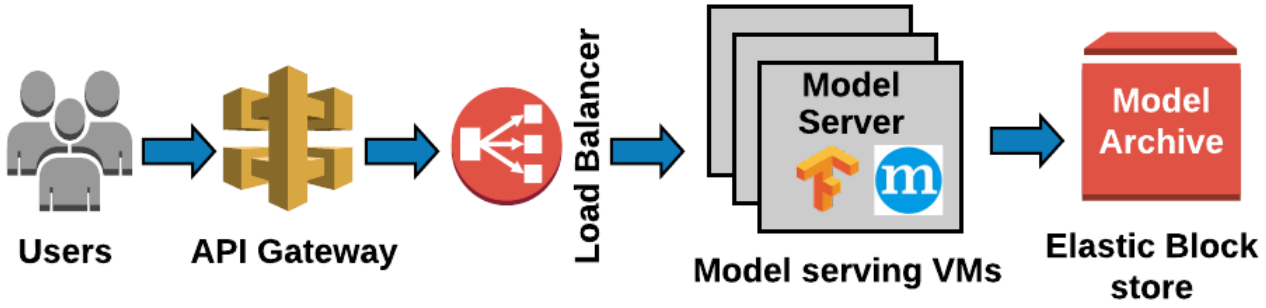


How to provision resources.?

Outline

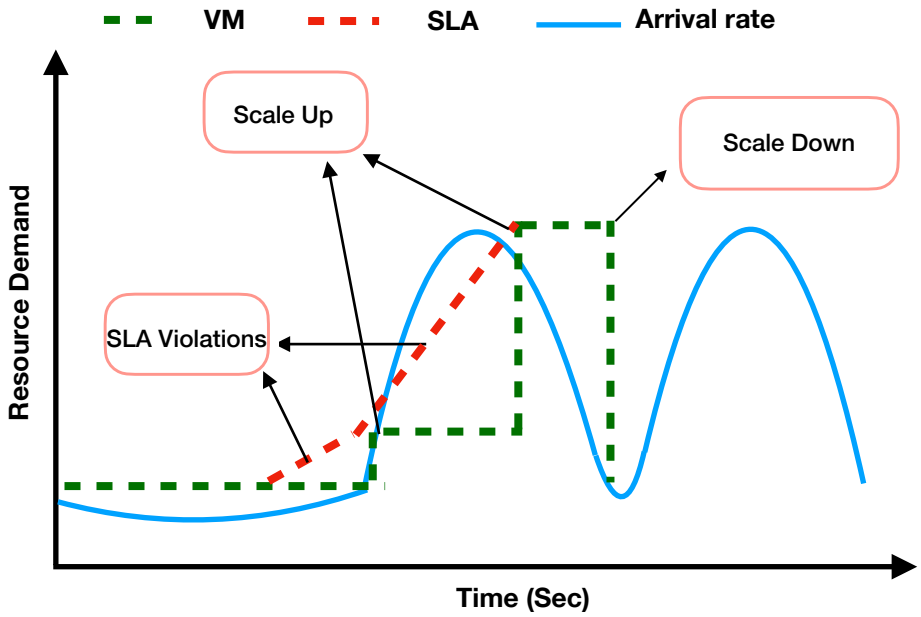
- Elastic Web Services.
- VM-based Resource Procurement.
- Serverless Functions.
- Spock Hybrid Elastic Scaling.
- Implementation and Evaluation.
- Results.

VM-based Procurement



VM-based Procurement

- Initial warm pool of active VMs.
- Procure more VMs on demand.
- Autoscaling during request surge.



Disadvantages

- Very long VM setup times.

Possible alternative.?

- Under-provisioned during sudden surge.

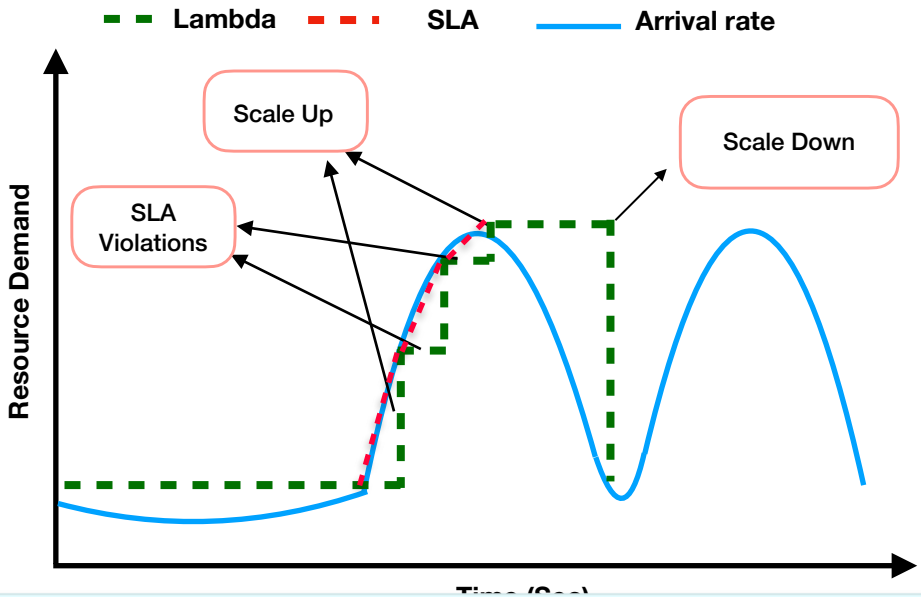
Outline

- Elastic Web Services.
- VM-based Resource Procurement.
- Serverless Functions.
- Spock Hybrid Elastic Scaling.
- Implementation and Evaluation.
- Results.

Serverless Functions

Serverless Functions

- Pay per second.
- Cost efficient.
- Scale instantaneously.

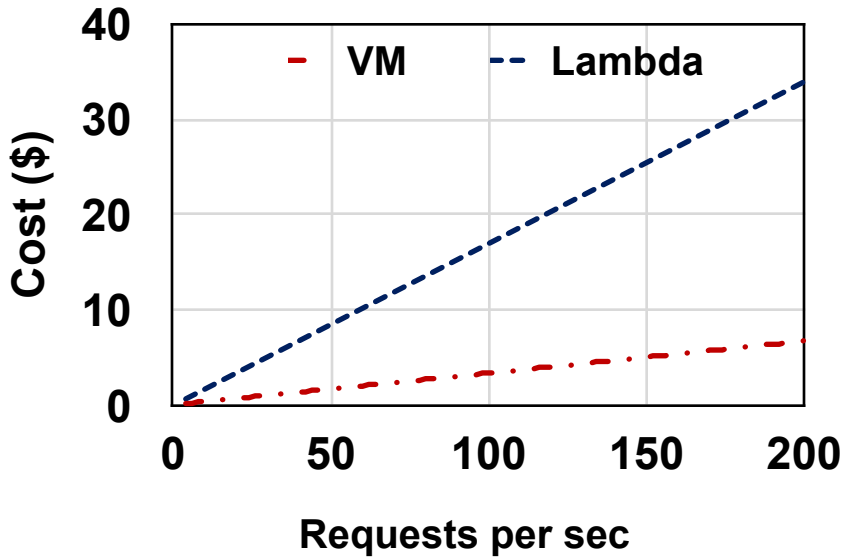


But, is serverless a replacement for VMs.?
Lets see an example.

violations

Constant arrival rate

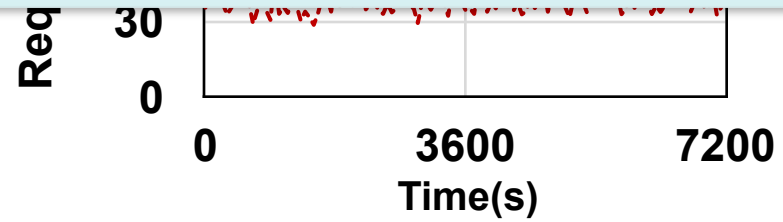
- Constant arrival rate.
- Cost compared under iso-performance.
- All requests have similar SLA compliance.
- VMs are 100% utilized.

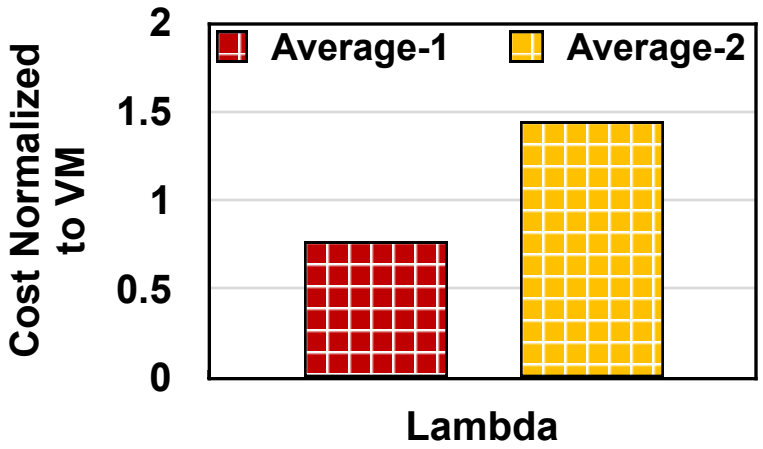


Varying arrival rate

Cost-effective Solution.?

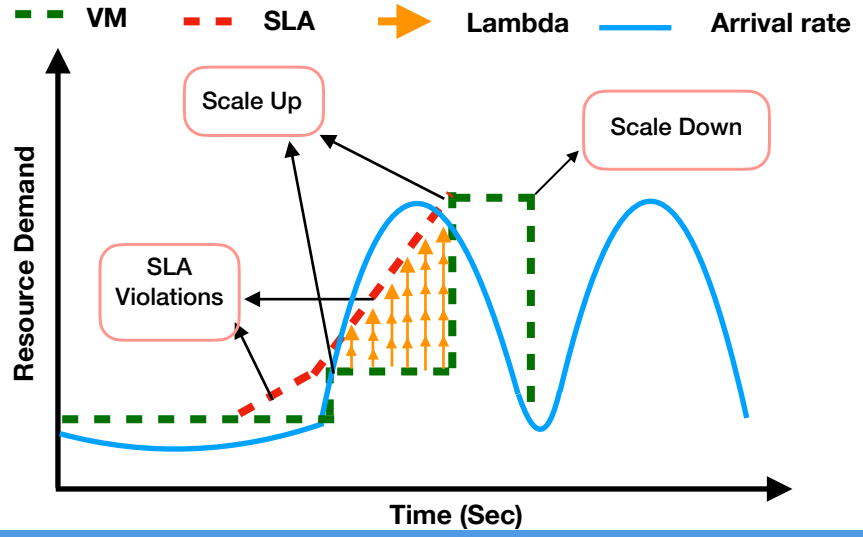
- Each request is an ML inference for caffenet-model
- Cost compared under iso-performance.
- All requests have similar SLA compliance.
- VMs are provisioned for the peak request rate.





Why SPOCK.?

- Use serverless functions along with VMS
  - Reduce SLO violations during request surge
  - Reduce intermittent over-provisioning VMs



Key Motivation

- It is non-trivial to predict the peak request rate at any given time period.
- Provisioning VMs for the peak demands would always lead to higher cost of deployment. While, under provisioning VMs leads to severe SLO violations for queries.
- Using serverless functions would overcome the SLO violation problem. However, it is not cost effective.

Outline

- Elastic Web Services.
- VM-based Resource Procurement.
- Serverless Functions.
- Spock Hybrid Elastic Scaling.
- Implementation and Evaluation.
- Results.

Spock Scheme

- Schedule queries on VM's if available.
- If VM's are fully utilized, redirect queries to lambda functions.
- Spawn a new VM in the meantime.
- After spin-up incoming requests are sent to new VMs.
- Scale down VMs after three minutes of inactivity.

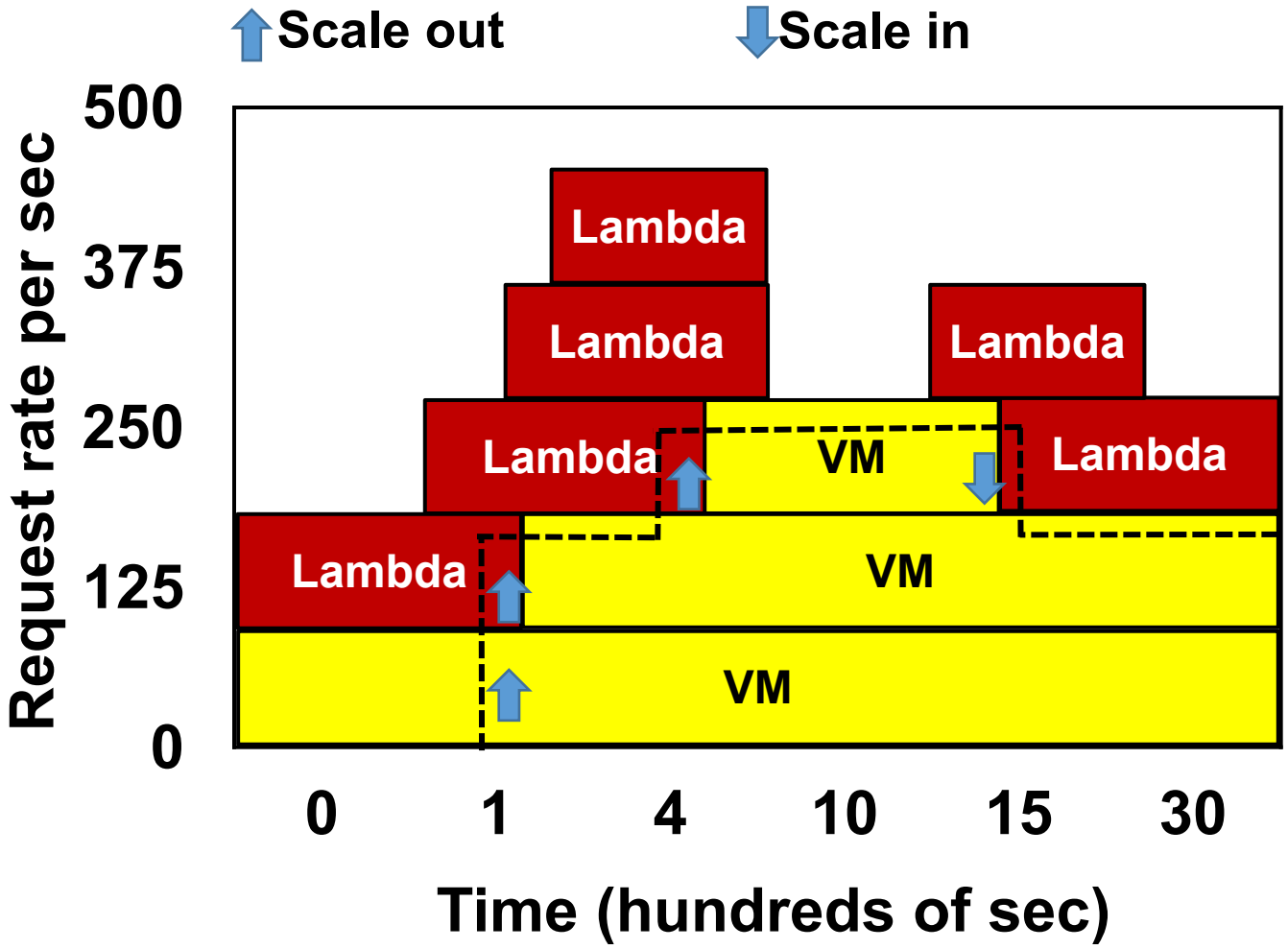
Two Scaling Policies

- **Reactive**
  - Spin-up new VMs as when request surge

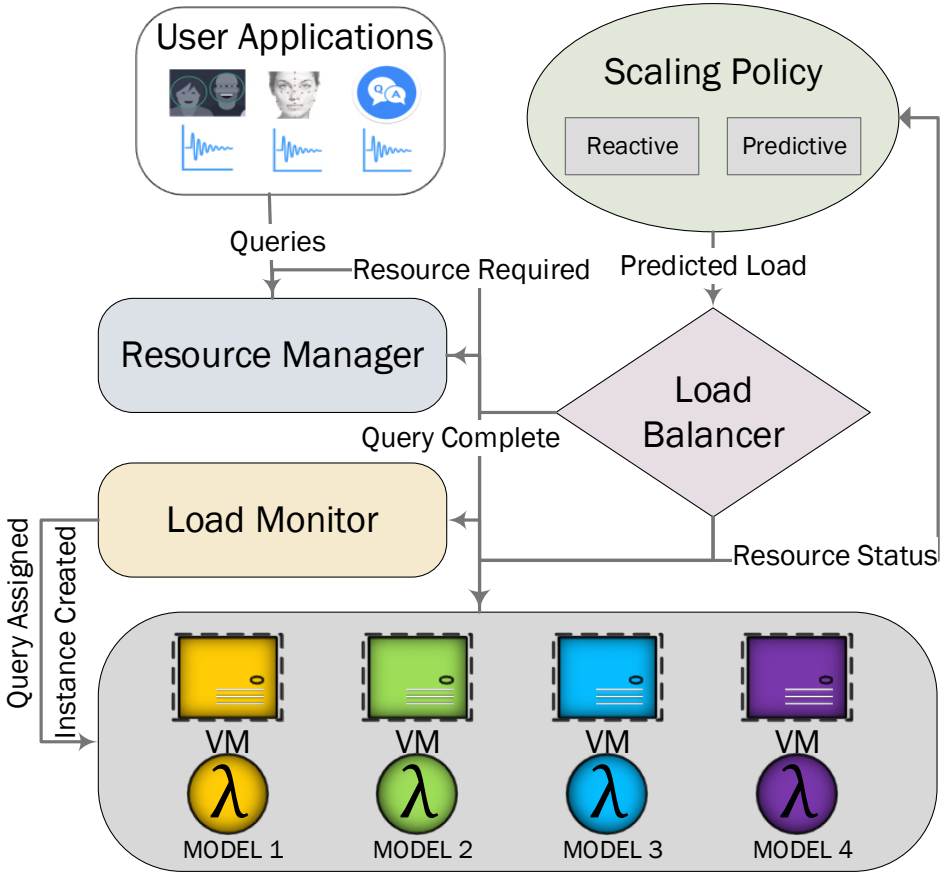
Lets see an example.

- **Predictive**
  - Using moving window linear regression predict request every 1 minute.
  - Spin up new VMs based on prediction.

Spock resource procurement



Overall Design of Spock

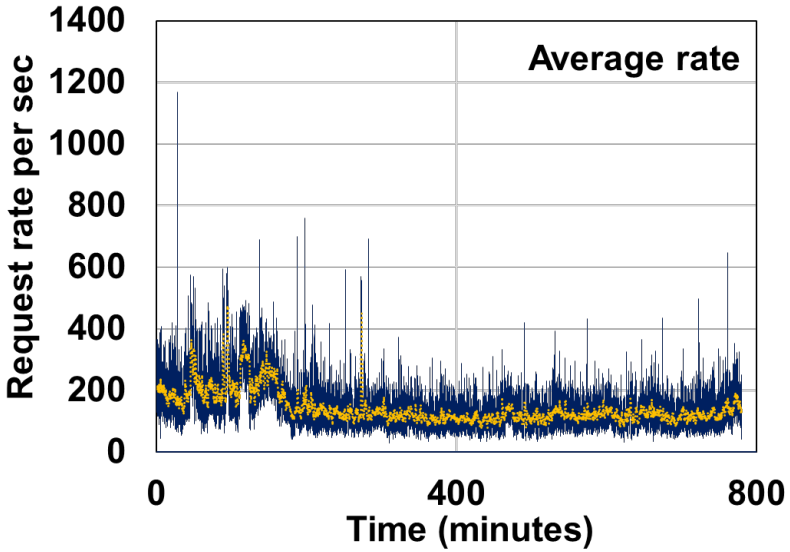


Outline

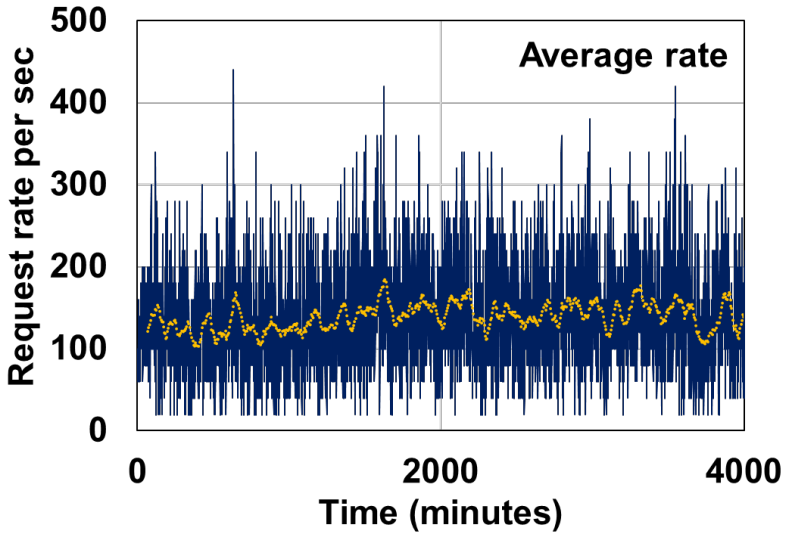
- Elastic Web Services.
- VM-based Resource Procurement.
- Serverless Functions.
- Spock Hybrid Elastic Scaling.
- Implementation and Evaluation.
- Results.

Evaluation

- Two traces used to generate ML inference workload.



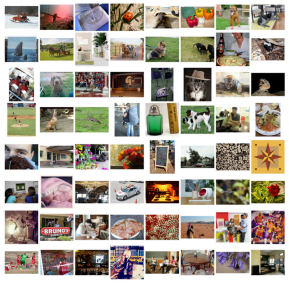
WITS



Berkeley

Evaluation

- Mxnet Framework.
- AWS resources.
- Pretrained ML models on imagenet dataset.



Query Type	Memory Required (GB)	Memory Allocated (GB)	Average Execution (ms)	Requests per vCPU for VMs
Caffenet	1.024	3.072	300	4
Googlenet	0.456	2.048	450	3
Squeezenet	0.154	2.048	130	6
Resnet-18	0.304	3.072	320	3
Resnet-200	1.024	3.072	956	1
Resnext-50	0.645	3.072	560	2

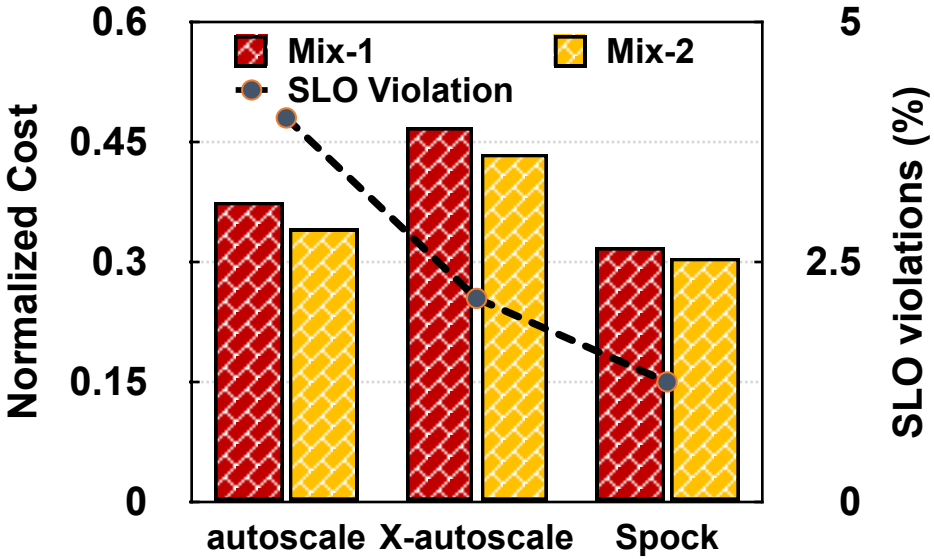
Evaluation

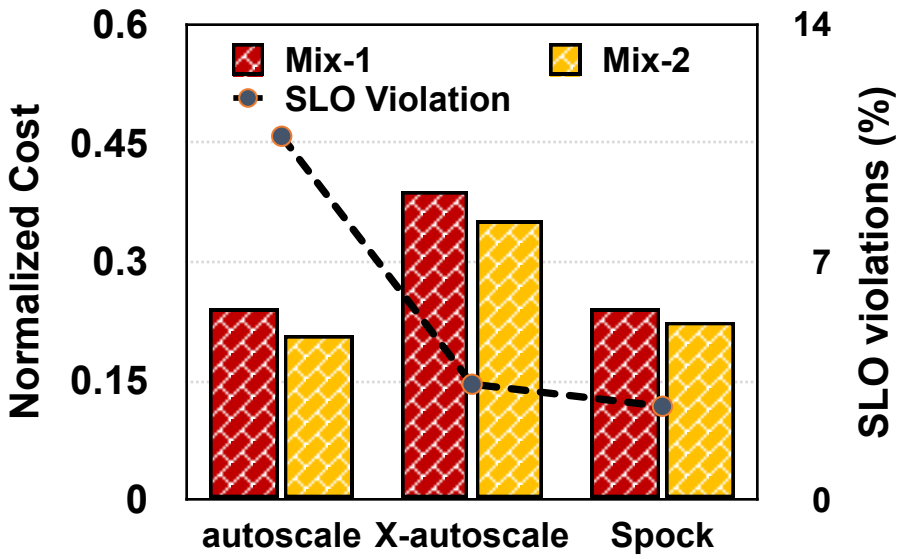
- Two scaling policies
  - **Predictive**
  - **Reactive**
- Three resource procurement schemes
  - **Autoscale**
  - **X-autoscale**
  - **Spock**

Outline

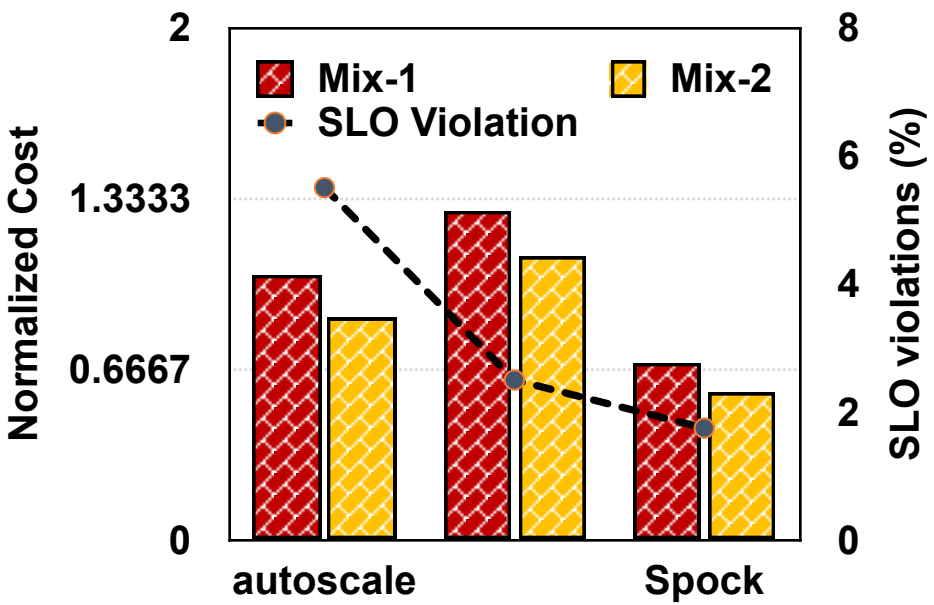
- Elastic Web Services.
- VM-based Resource Procurement.
- Serverless Functions.
- Spock Hybrid Elastic Scaling.
- Implementation and Evaluation.
- Results.

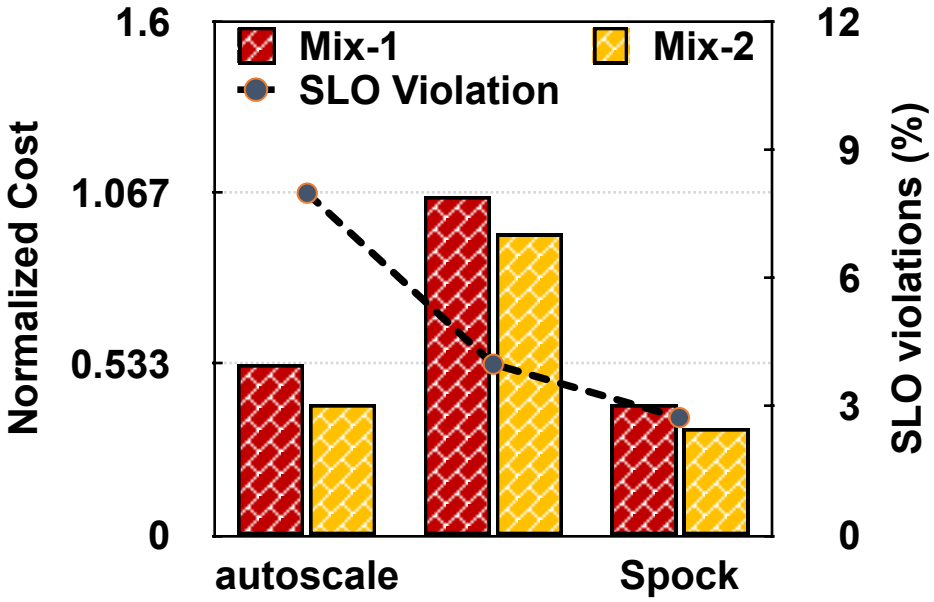
Berkely Trace Results



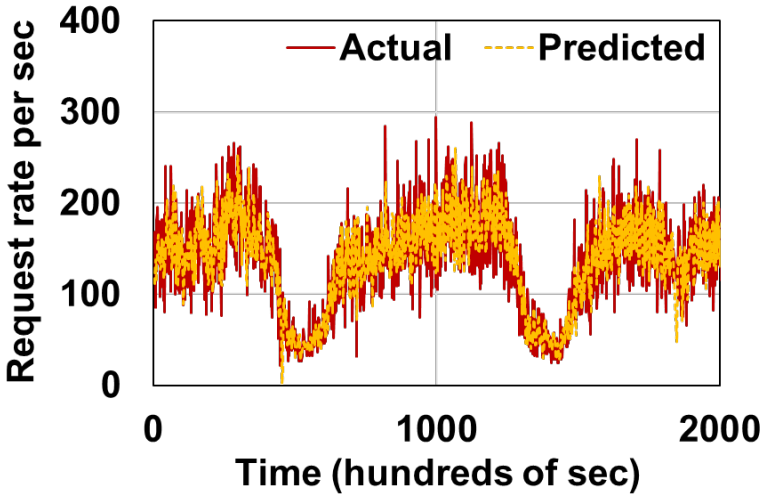


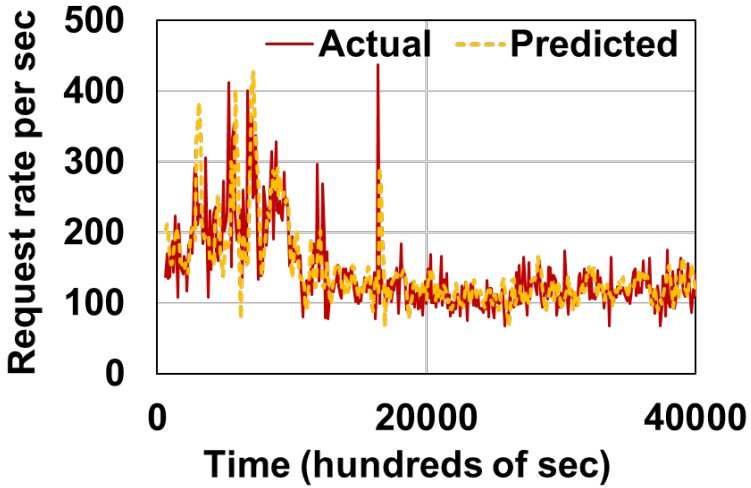
WITS Trace Results





Spock Prediction Accuracy





Spock Resource Procurement