Predicting Performance and Cost of Serverless Computing Functions with SAAF

Robert Cordingly, Wen Shu, Wes Lloyd

August 17-24, 2020

School of Engineering and Technology
University of Washington Tacoma
CBDCom 2020: IEEE International Conference on Cloud and Big Data

Outline

- Background and Motivation
- Research Questions
- Serverless Application Analytics Framework (SAAF)
- Experiments and Modeling
- Experiment Results
- Conclusions

Serverless: Function-as-a-Service

- Developers deploy small applications called micro-services
- Cloud providers automatically scale and manage cloud infrastructure instead of developers
- Can scale from zero users to thousands instantly
- Guarantee high availability and fault tolerance

AWS Lambda

IBM Cloud Functions

The cost of FaaS vs VMs

The cost of VMs:

- (Number of VMs) x (Uptime) x (Price)
- Billed for entire VM uptime even when idle.

λ

The cost of FaaS:

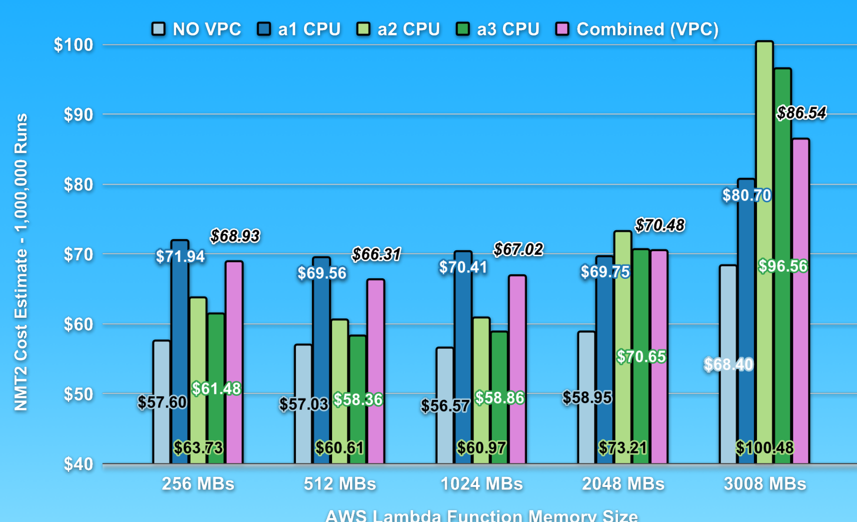
- (Function Runtime) x (Memory Setting) x (Price)
- Billed only for runtime used.

Performance Variation in FaaS

λ

- **(Application Runtime)** x (Memory Setting) x (Price)

- Many factors can contribute to variation in FaaS performance.



Outline

- Background and Motivation
- ➡ Research Questions
- Serverless Application Analytics Framework (SAAF)
- Experiments and Modeling
- Experiment Results
- Conclusions

Research Questions

RQ-1: (Performance Variance) What factors are responsible for performance variation on FaaS platforms?

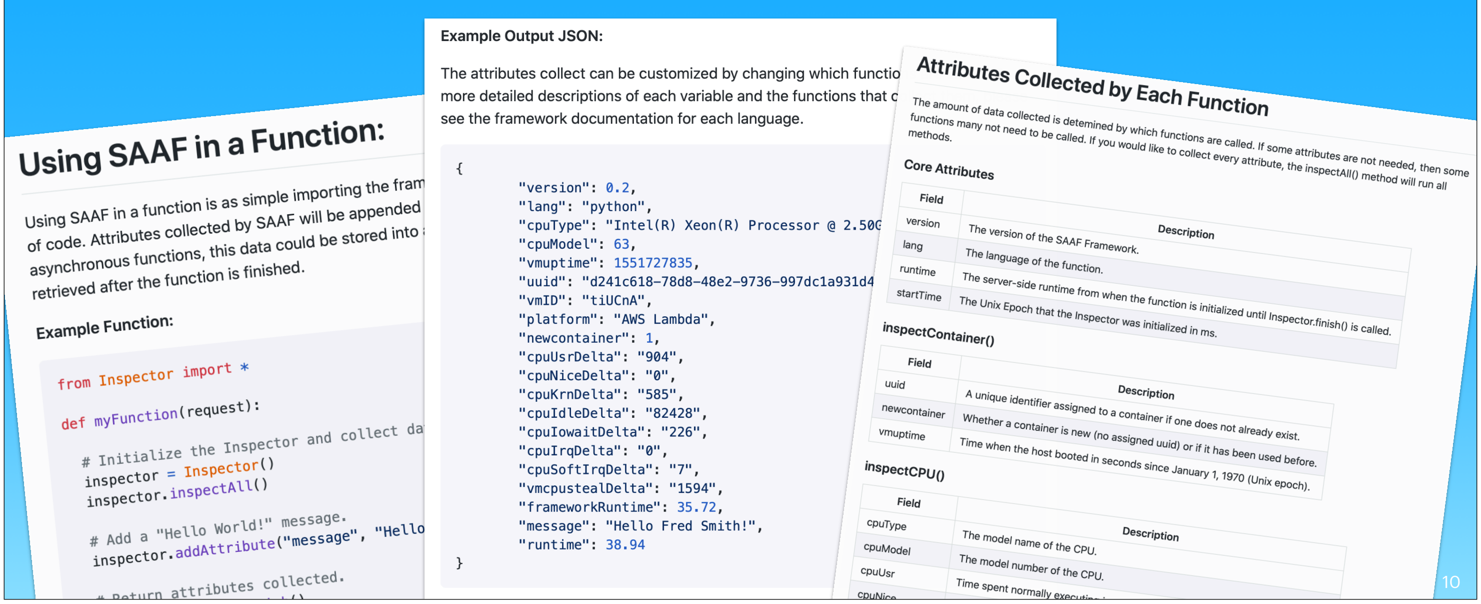
RQ-2: (FaaS Runtime Prediction) When leveraging Linux CPU time accounting principles and regression modeling, what is the accuracy of FaaS function runtime predictions for deployments to different memory settings or CPUs?

RQ-3: (Assessing Workload Predictability) How effective are system metrics at evaluating reliability of performance predictions?

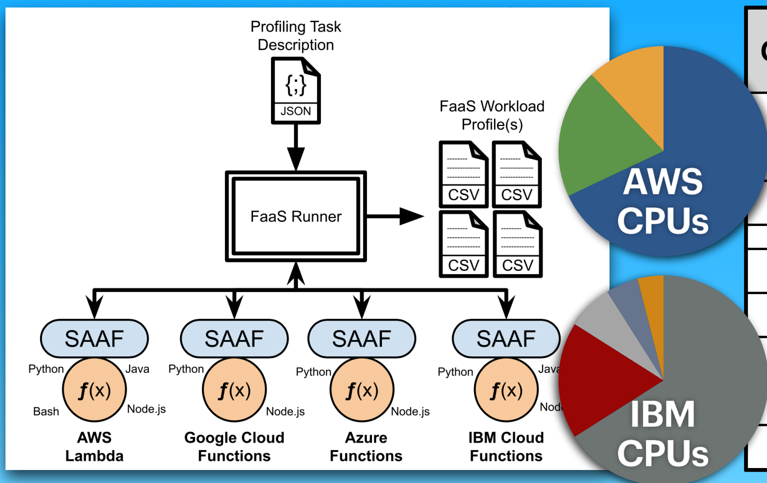
Outline

- Background and Motivation
- Research Questions
- ➡ Serverless Application Analytics Framework (SAAF)
- Experiments and Modeling
- Experiment Results
- Conclusions

Serverless Application Analytics Framework (SAAF)



FaaS Runner and Hardware Distribution



Cloud	Intel Xeon CPU	VM	%
AWS	E5-2680v2 @ 2.8 GHz, 10 core	c3	67.5
AWS	E5-2676v3 @ 2.4 GHz, 12 core	m4	19.9
AWS	E5-2686v4 @ 2.3 GHz, 18 core	r4	12.5
IBM	E5-2683v3 @ 2.0 GHz, 14 core	n/a	18.4
IBM	E5-2683v4 @ 2.1 GHz, 16 core	b1	66.1
IBM	E5-2650v4 @ 2.2 GHz, 12 core	u1	3.8
IBM	E5-2690v4 @ 2.6 GHz, 14 core	c1	7.2
IBM	Gold 6140 @ 2.3 GHz, 18 core	n/a	4.5

Outline

- Background and Motivation
- Research Questions
- Serverless Application Analytics Framework (SAAF)
- ➡ Experiments and Modeling
- Experiment Results
- Conclusions

Calcs Service Workloads

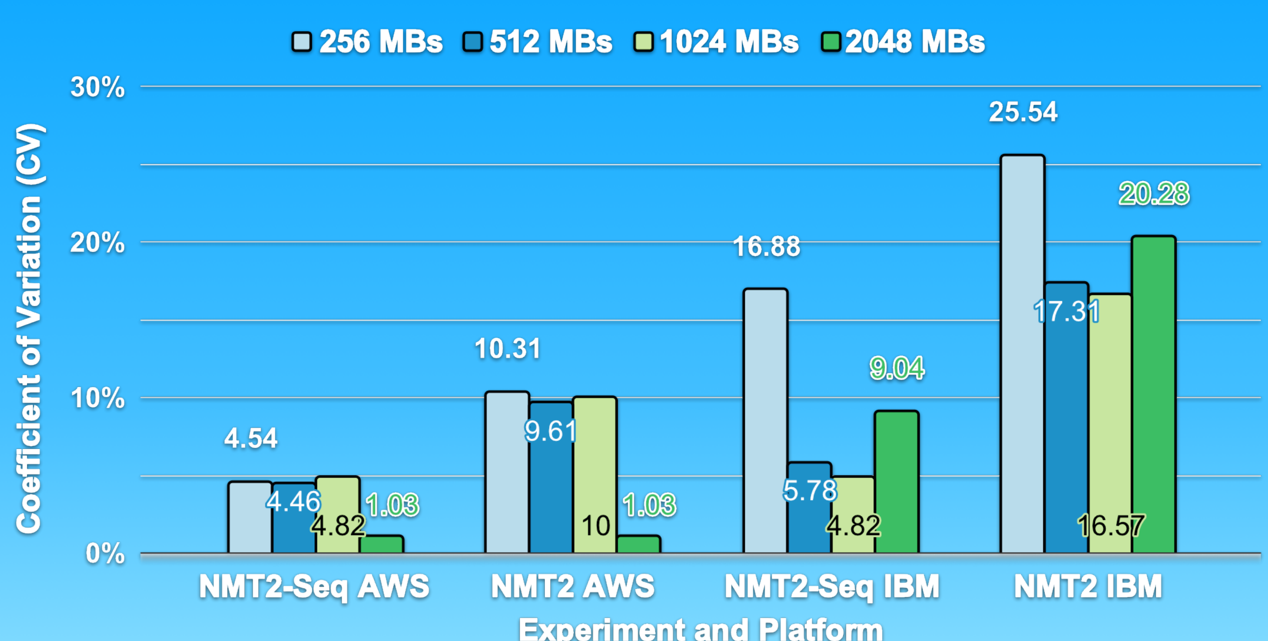
a × b ÷ c

Name	Definition
NMT1	Fixed # of Calcs, No Memory Stress, 1 Thread, concurrent calls
MT1	Fixed # of Calcs, Memory Stress, 1 Thread, concurrent calls
NMT2-seq	Fixed # of Calcs, No Memory stress, 2 Threads, Sequential calls
NMT2	Fixed # of Calcs, No Memory Stress, 2 Threads, concurrent calls
MT2	Fixed # of Calcs, Memory Stress, 2 Threads, concurrent calls
SCNMT1	Scaling Calcs, No Memory Stress, 1 Thread, concurrent calls
SCMT1	Scaling Calcs, Memory Stress, 1 Thread, concurrent calls
SCNMT2	Scaling Calcs, No Memory Stress, 2 Threads, concurrent calls
SCMT2	Scaling Calcs, Memory Stress, 2 Threads, concurrent calls
SCSMT2	Scaling Calcs, Scaling Memory Stress, 2 Threads, concurr. calls

Name	Calculations	Memory Stress	Threads	Tenancy
NMT1	40 million	No	1	n
MT1	40 million	array=1 million	1	n
NMT2-seq	40 million	No	2	1
NMT2	40 million	No	2	n
MT2	40 million	array=1 million	2	n
SCNMT1	30-60m step 3m	No	1	n
SCMT1	30-60m step 3m	array=1 million	1	n
SCNMT2	30-60m step 3m	No	2	n
SCMT2	30-60m step 3m	array=1 million	2	n
SCSMT2	30-60m step 3m	1-1m, step 100k	2	n

Calc Service on GitHub: github.com/wlloyduw/CalcsService

Variation Caused By Multitenancy



FaaS Pricing and Performance Obfuscation

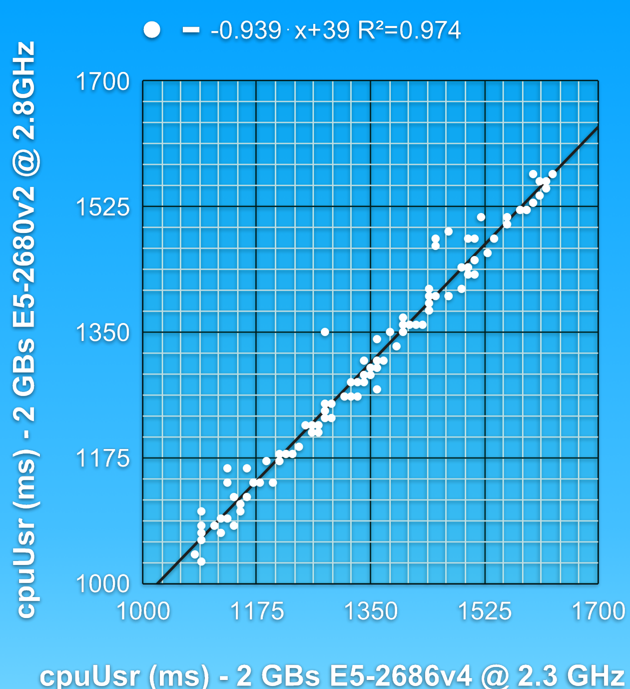
AWS vs IBM

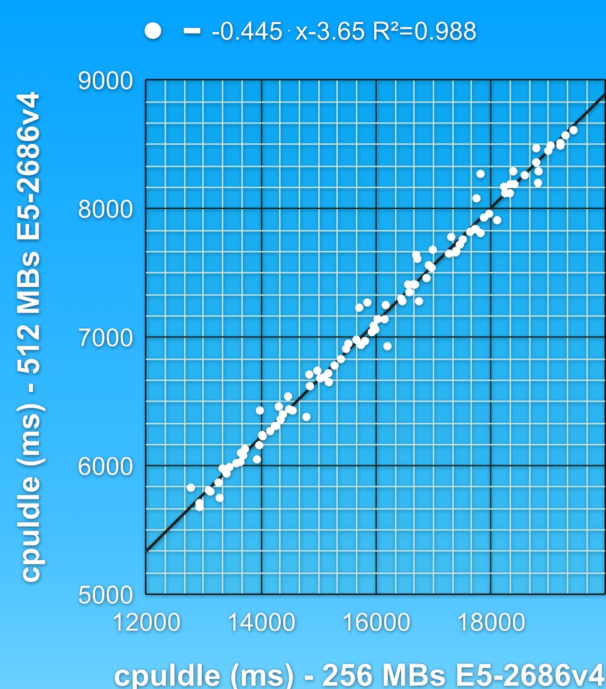
AWS Lambda

- Price (USD):
  Runtime (s) x Memory (GBs) x $0.0000166667
- Scales performance with memory setting.
- Tenancy has a small impact on performance.
- For our workloads, a mid range memory setting 512-1024 MBs provides the best price to performance ratio.

IBM Cloud Functions

- Price (USD):
  Runtime (s) x Memory (GBs) x $0.000017
- Does not scale performance with memory setting.
- Tenancy has a massive impact on performance.
- For infrequently invoked functions, the minimum memory setting is always the best choice.





$$\text{Runtime} = \frac{(\text{cpuUsr} + \text{cpuKrn} + \text{cpuIdle} + \text{cpuIOWait} + \text{cpuIntSrvc} + \text{cpuSftIntSrvc})}{(\#\text{ of cores})}$$

Making Predictions Across More Configurations

CPU:

- 256 MBs **a1** → **a2**
- 256 MBs **a1** → **a3**
- 256 MBs **a2** → **a3**
- 512 MBs **a1** → **a2**
- 512 MBs **a1** → **a3**
- 512 MBs **a2** → **a3**
- 1024 MBs **a1** → **a2**
- 1024 MBs **a1** → **a3**
- 1024 MBs **a2** → **a3**
- 2048 MBs **a1** → **a2**
- 2048 MBs **a1** → **a3**
- 2048 MBs **a2** → **a3**

Memory:

- a1 **256MBs → 512MBs**
- a1 **256MBs → 1024MBs**
- a1 **256MBs → 2048MBs**
- a2 **256MBs → 512MBs**
- a2 **256MBs → 1024MBs**
- a2 **256MBs → 2048MBs**
- a3 **256MBs → 512MBs**
- a3 **256MBs → 1024MBs**
- a3 **256MBs → 2048MBs**

Platform:

- 256MBs **a1 → i1**
- 256MBs **a1 → i2**
- 256MBs **a1 → i3**
- 256MBs **a1 → i4**
- 512MBs **a1 → i1**
- 512MBs **a1 → i2**
- 512MBs **a1 → i3**
- 512MBs **a1 → i4**
- 1024MBs **a1 → i1**
- 1024MBs **a1 → i2**
- 1024MBs **a1 → i3**
- 1024MBs **a1 → i4**
- 2048MBs **a1 → i1**
- 2048MBs **a1 → i2**
- 2048MBs **a1 → i3**
- 2048MBs **a1 → i4**

Repeat for SCNMT2, SCMT2, and SCSMT2 Workloads

CPU Aliases:

- **a1:** Amazon Intel Xeon E5-2680v2
- **a2:** Amazon Intel Xeon E5-2676v3
- **a3:** Amazon Intel Xeon E5-2686v4
- **i1:** IBM Intel Xeon E5-2683v3
- **i2:** IBM Intel Xeon E5-2683v4
- **i3:** IBM Intel Xeon E5-2650v4
- **i4:** IBM Intel Xeon E5-2690v4

Outline

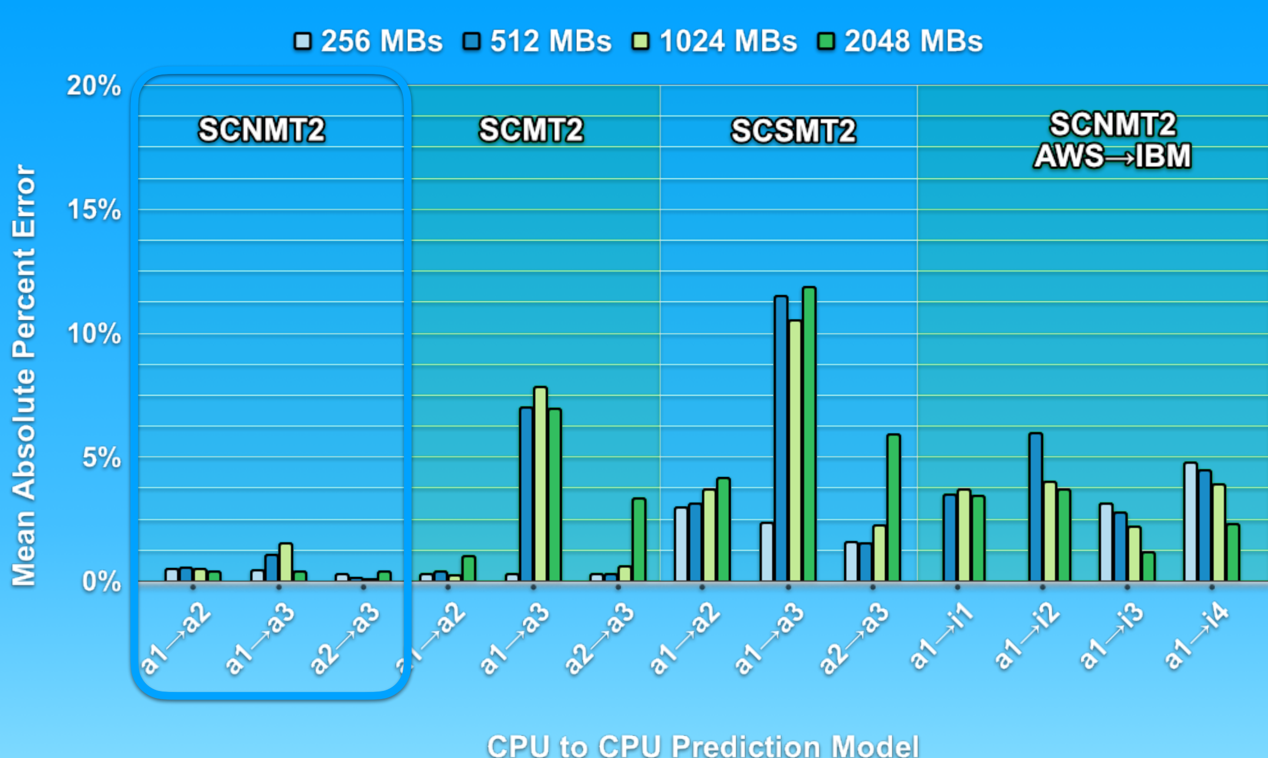
- Background and Motivation
- Research Questions
- Serverless Application Analytics Framework (SAAF)
- Experiments and Modeling
- ➡ Experiment Results
- Conclusions

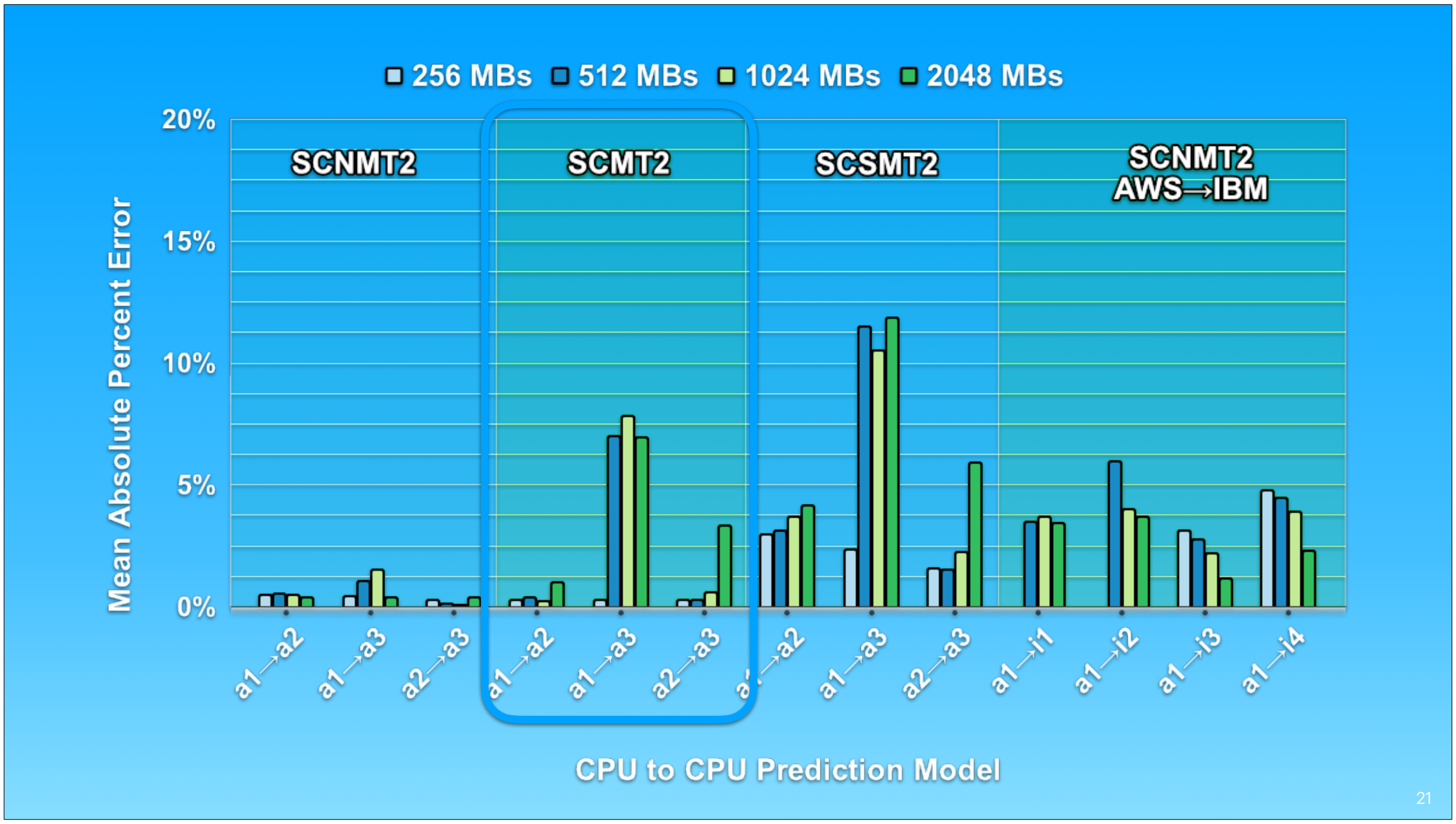
Workloads for Predictions

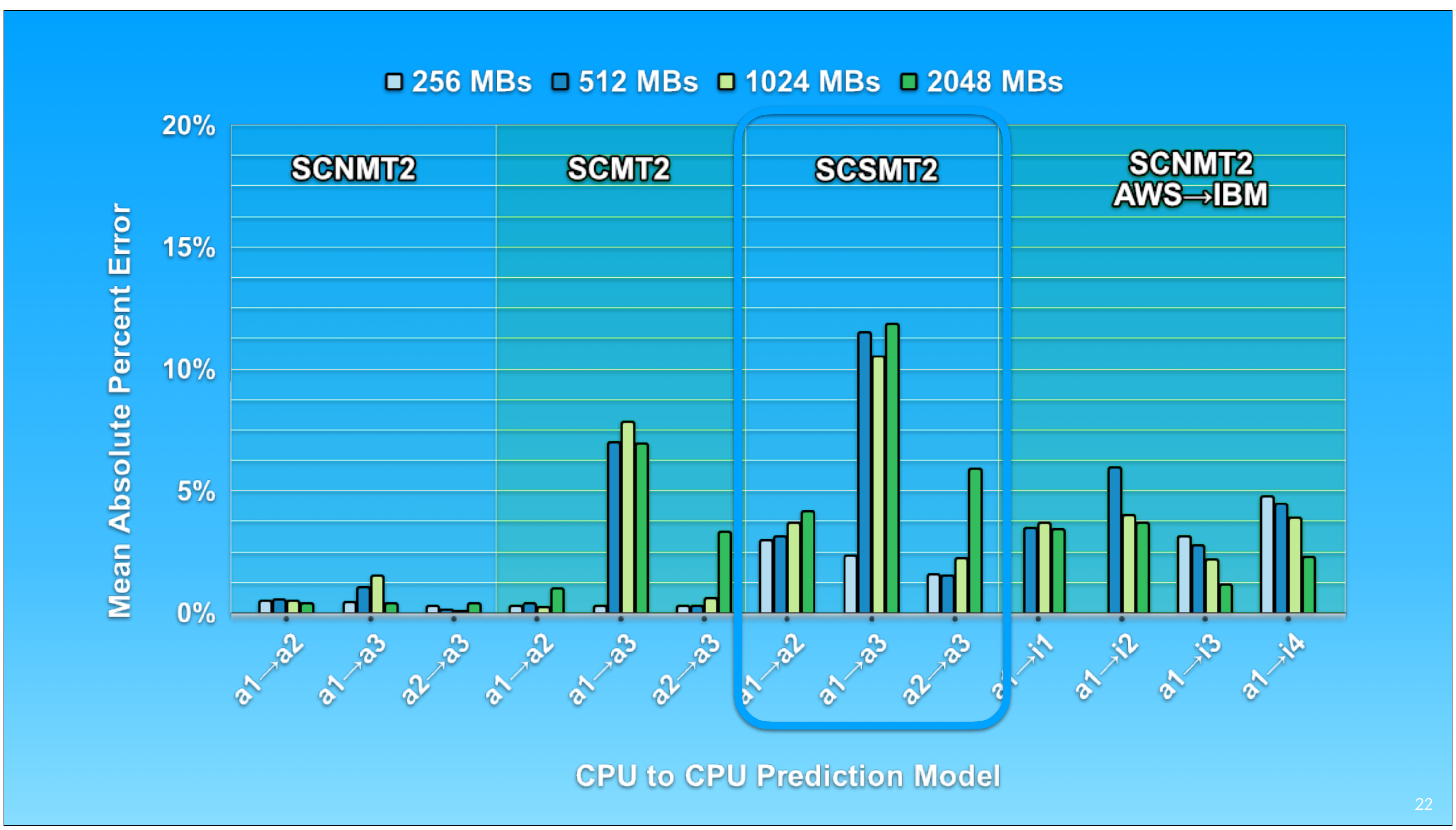
a × b ÷ c

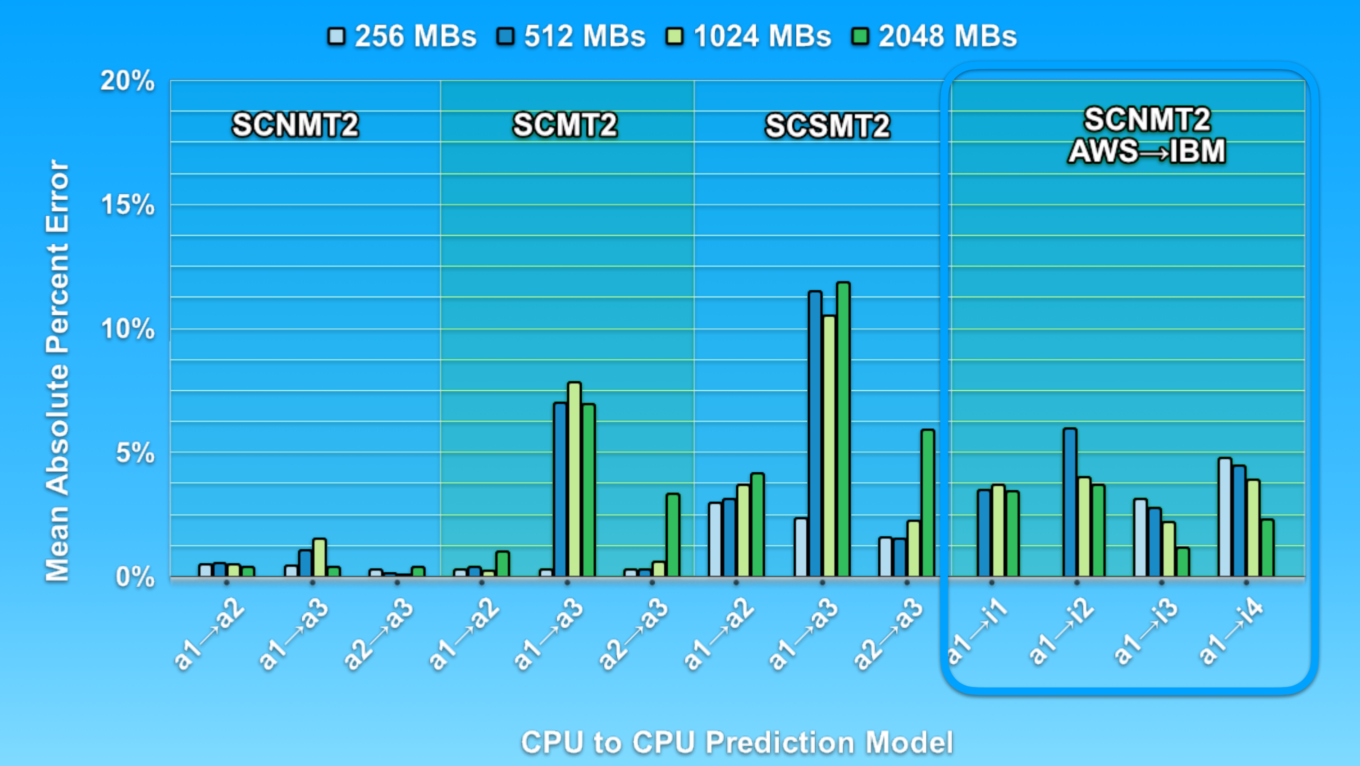
Name	Definition
NMT1	Fixed # of Calcs, No Memory Stress, 1 Thread, concurrent calls
MT1	Fixed # of Calcs, Memory Stress, 1 Thread, concurrent calls
NMT2-seq	Fixed # of Calcs, No Memory stress, 2 Threads, Sequential calls
NMT2	Fixed # of Calcs, No Memory Stress, 2 Threads, concurrent calls
MT2	Fixed # of Calcs, Memory Stress, 2 Threads, concurrent calls
SCNMT1	Scaling Calcs, No Memory Stress, 1 Thread, concurrent calls
SCMT1	Scaling Calcs, Memory Stress, 1 Thread, concurrent calls
SCNMT2	Scaling Calcs, No Memory Stress, 2 Threads, concurrent calls
SCMT2	Scaling Calcs, Memory Stress, 2 Threads, concurrent calls
SCSMT2	Scaling Calcs, Scaling Memory Stress, 2 Threads, concurr. calls

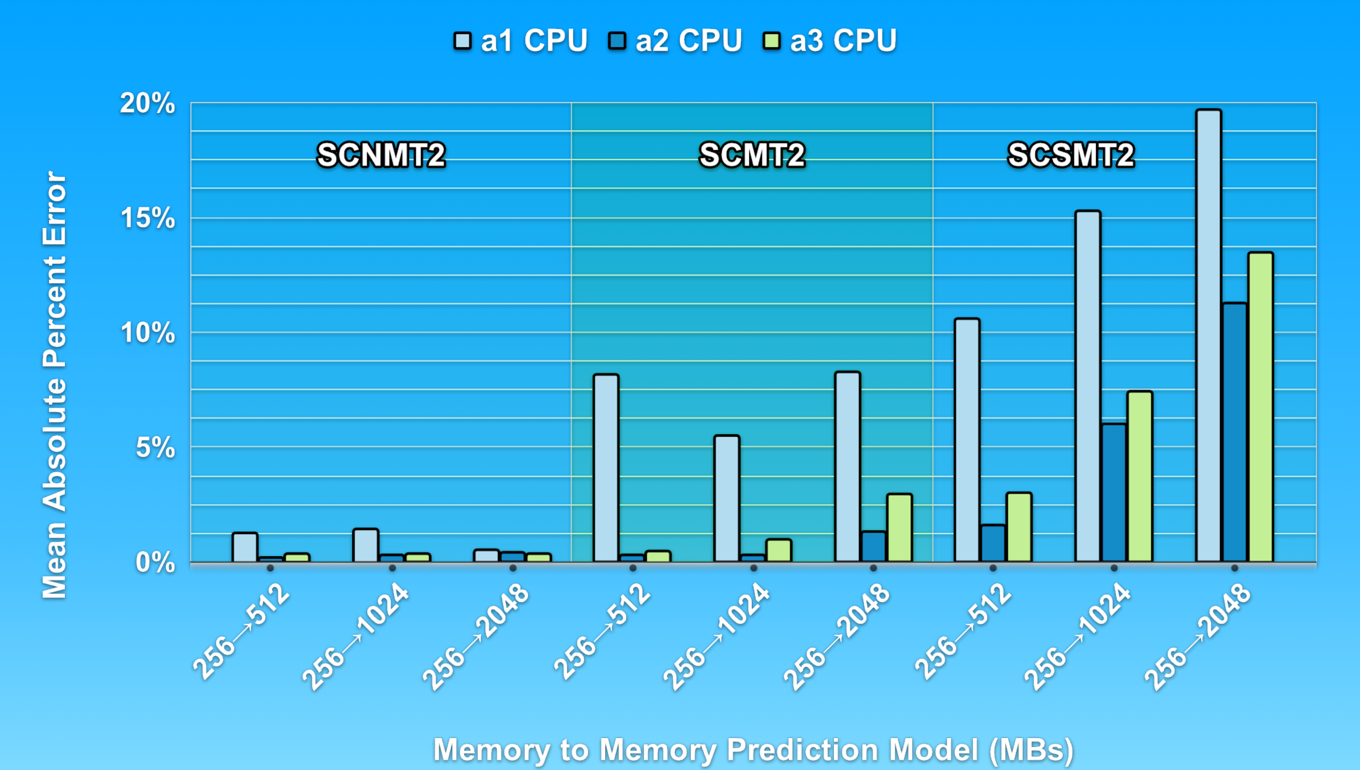
Name	Calculations	Memory Stress	Threads	Tenancy
NMT1	40 million	No	1	n
MT1	40 million	array=1 million	1	n
NMT2-seq	40 million	No	2	1
NMT2	40 million	No	2	n
MT2	40 million	array=1 million	2	n
SCNMT1	30-60m step 3m	No	1	n
SCMT1	30-60m step 3m	array=1 million	1	n
SCNMT2	30-60m step 3m	No	2	n
SCMT2	30-60m step 3m	array=1 million	2	n
SCSMT2	30-60m step 3m	1-1m, step 100k	2	n




256 MBs 512 MBs 1024 MBs 2048 MBs
SCNMT2
SCMT2
SCSMT2
SCNMT2 AWS→IBM
Mean Absolute Percent Error
20%
15%
10%
5%
0%
a1→a2
a1→a3
a2→a3
a1→a2
a1→a3
a2→a3
a1→a3
a2→a3
a1→i2
a1→i3
a1→i4
CPU to CPU Prediction Model
21


256 MBs 512 MBs 1024 MBs 2048 MBs
SCNMT2
SCMT2
SCSMT2
SCNMT2 AWS→IBM
Mean Absolute Percent Error
20%
15%
10%
5%
0%
a1→a2
a1→a3
a2→a3
a1→a2
a1→a3
a2→a3
a1→a2
a1→a3
a2→a3
a1→i2
a1→i3
a1→i4
CPU to CPU Prediction Model
22


256 MBs 512 MBs 1024 MBs 2048 MBs
Mean Absolute Percent Error
20%
15%
10%
5%
0%
SCNMT2
SCMT2
SCSMT2
SCNMT2 AWS→IBM
a1→a2
a1→a3
a2→a3
a1→a2
a1→a3
a2→a3
a1→a2
a1→a3
a2→a3
a1→i1
a1→i2
a1→i3
a1→i4
CPU to CPU Prediction Model


a1 CPU a2 CPU a3 CPU
Mean Absolute Percent Error
20%
15%
10%
5%
0%
SCNMT2
SCMT2
SCSMT2
256→512
256→1024
256→2048
256→512
256→1024
256→2048
256→512
256→1024
256→2048
Memory to Memory Prediction Model (MBs)

Outline

- Background and Motivation
- Research Questions
- Serverless Application Analytics Framework (SAAF)
- Experiments and Modeling
- Experiment Results
- → Conclusions

Conclusions - RQ-1

RQ-1: (Performance Variance) What factors are responsible for performance variance on FaaS platforms?

CPU Heterogeneity

~7.4x

Multi-tenancy

~2.7x

Conclusions - RQ-2

RQ-2: (FaaS Runtime Prediction) When leveraging Linux CPU time accounting principles and regression modeling, what is the accuracy of FaaS function runtime predictions for deployments with different memory settings or CPUs?

Workload Prediction Type	Number of Models	Workload CV	MAPE	Average Cost Error	Average Workload Cost
SCNMT2 – CPU	12	21%	0.51%	$0.36	$70.27
SCMT2 – CPU	12	23%	2.52%	$5.15	$204.23
SCSMT2 – CPU	12	32%	5.10%	$8.86	$173.64
SCNMT2 – Memory	9	20%	0.59%	$0.45	$76.30
SCMT2 – Memory	9	22%	3.83%	$9.07	$236.88
SCSMT2 – Memory	9	36%	10.5%	$19.99	$190.80
SCNMT2 - IBM	14	18%	3.55%	$4.24	$119.38
Overall Average	***77 (sum)***	**--**	***3.49%***	***$6.46***	***$150.45***

Conclusions - RQ-3

RQ-3: (Assessing Workload Predictability) How effective are system metrics at evaluating reliability of performance predictions?

↑ cpuSteal **↓ freeMemory**

Thank You for Watching

Questions or comments?
Please email:
rcording@uw.edu or wlloyd@uw.edu

Download SAAF and FaaS Runner
github.com/wlloyduw/saaf

This research is supported by NSF Advanced Cyberinfrastructure Research Program (OAC-1849970), NIH grant R01GM126019, and the AWS Cloud Credits for Research program.