



# 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

1

## Outline

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

2

The AWS logo is displayed inside a light blue cloud shape. It consists of the lowercase letters "aws" in a bold, black, sans-serif font, with the orange Amazon smile arrow underneath.

Google Cloud

The Azure logo, featuring a stylized blue 'A' icon followed by the word "Azure" in a blue, sans-serif font.

IBM Cloud

3

## 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

The AWS Lambda logo, featuring a stylized orange and brown icon of a lambda symbol above the text "AWS Lambda".

IBM Cloud  
Functions

4

# 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.



## The cost of FaaS:

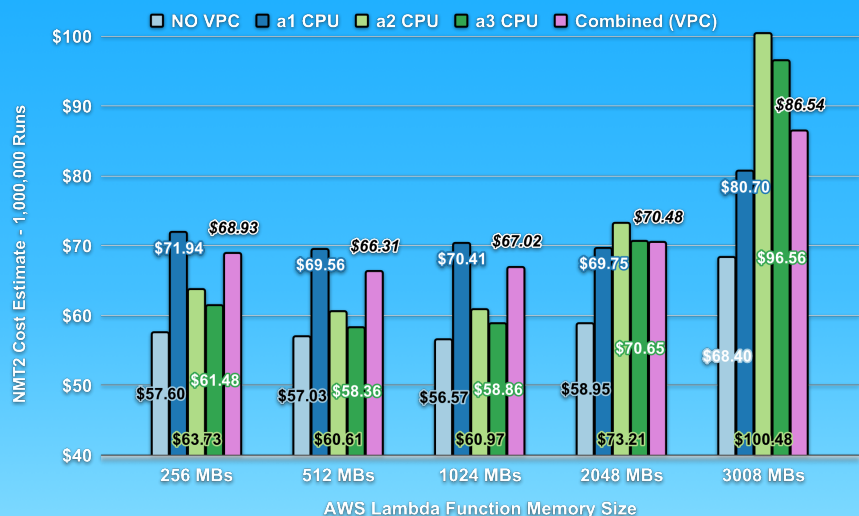
- (Function Runtime) x (Memory Setting) x (Price)
- Billed only for runtime used.

# Performance Variation in FaaS



- **(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

7

## 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?

8



# Outline

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

# Serverless Application Analytics Framework (SAAF)

## Using SAAF in a Function:

Using SAAF in a function is as simple as importing the framework into your code. Attributes collected by SAAF will be appended to the function's return value. For asynchronous functions, this data could be stored into a database and retrieved after the function is finished.

### Example Function:

```
from Inspector import *  
  
def myFunction(request):  
    # Initialize the Inspector and collect data  
    inspector = Inspector()  
    inspector.inspectAll()  
  
    # Add a "Hello World!" message.  
    inspector.addAttribute("message", "Hello World!")  
  
    # Return attributes collected.  
    return inspector.attributes
```

### Example Output JSON:

The attributes collected can be customized by changing which functions are called. For more detailed descriptions of each variable and the functions that collect them, see the framework documentation for each language.

```
{  
  "version": 0.2,  
  "lang": "python",  
  "cpuType": "Intel(R) Xeon(R) Processor @ 2.50GHz",  
  "cpuModel": 63,  
  "vmuptime": 1551727835,  
  "uuid": "d241c618-78d8-48e2-9736-997dc1a931d4",  
  "vmID": "tiUCnA",  
  "platform": "AWS Lambda",  
  "newcontainer": 1,  
  "cpuUsrDelta": "904",  
  "cpuNiceDelta": "0",  
  "cpuKrnDelta": "585",  
  "cpuIdleDelta": "82428",  
  "cpuIwaitDelta": "226",  
  "cpuIrqDelta": "0",  
  "cpuSoftIrqDelta": "7",  
  "vmcpusteatDelta": "1594",  
  "frameworkRuntime": 35.72,  
  "message": "Hello Fred Smith!",  
  "runtime": 38.94  
}
```

## Attributes Collected by Each Function

The amount of data collected is determined by which functions are called. If some attributes are not needed, then some functions may not need to be called. If you would like to collect every attribute, the inspectAll() method will run all methods.

### Core Attributes

Field	Description
version	The version of the SAAF Framework.
lang	The language of the function.
runtime	The server-side runtime from when the function is initialized until Inspector.finish() is called.
startTime	The Unix Epoch that the inspector was initialized in ms.

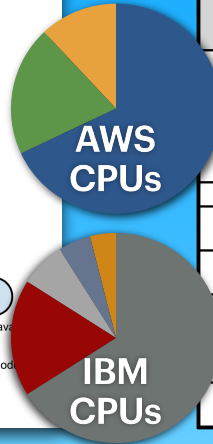
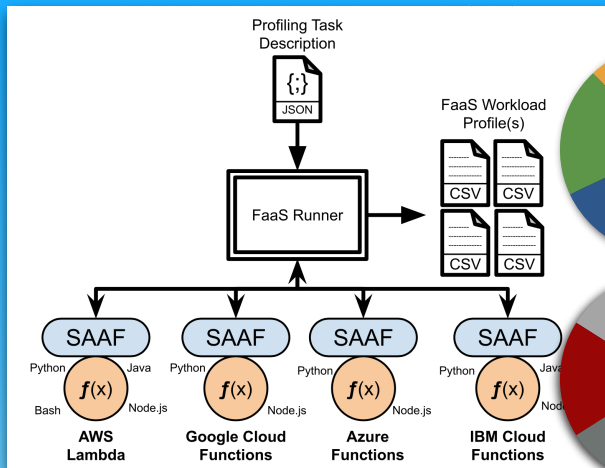
### inspectContainer()

Field	Description
uuid	A unique identifier assigned to a container if one does not already exist.
newcontainer	Whether a container is new (no assigned uuid) or if it has been used before.
vmuptime	Time when the host booted in seconds since January 1, 1970 (Unix epoch).

### inspectCPU()

Field	Description
cpuType	The model name of the CPU.
cpuModel	The model number of the CPU.
cpuUsr	Time spent normally executing user code.
cpuNice	Time spent normally executing nice code.

# 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

$$a \times b \div c$$

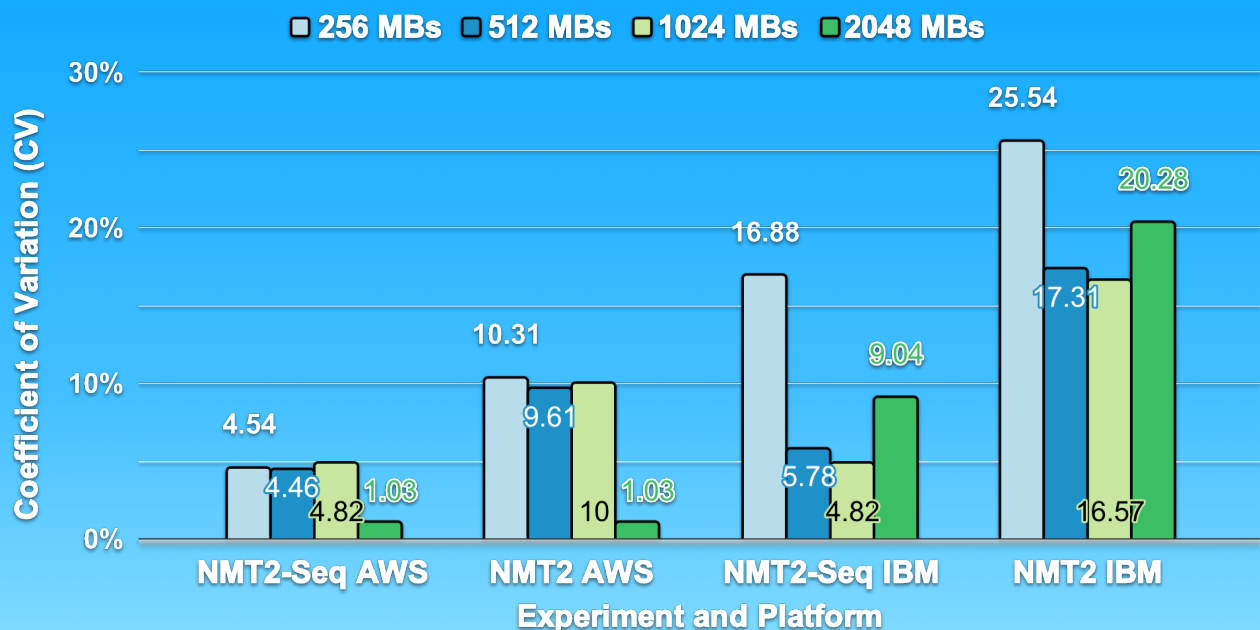
Name	Definition
NMT1	Fixed # of Calcs, <b>No</b> Memory Stress, <b>1</b> Thread, concurrent calls
MT1	Fixed # of Calcs, <b>Memory</b> Stress, <b>1</b> Thread, concurrent calls
NMT2-seq	Fixed # of Calcs, <b>No</b> Memory stress, <b>2</b> Threads, <b>Sequential</b> calls
NMT2	Fixed # of Calcs, <b>No</b> Memory Stress, <b>2</b> Threads, concurrent calls
MT2	Fixed # of Calcs, <b>Memory</b> Stress, <b>2</b> Threads, concurrent calls
SCNMT1	<b>Scaling</b> Calcs, <b>No</b> Memory Stress, <b>1</b> Thread, concurrent calls
SCMT1	<b>Scaling</b> Calcs, <b>Memory</b> Stress, <b>1</b> Thread, concurrent calls
SCNMT2	<b>Scaling</b> Calcs, <b>No</b> Memory Stress, <b>2</b> Threads, concurrent calls
SCMT2	<b>Scaling</b> Calcs, <b>Memory</b> Stress, <b>2</b> Threads, concurrent calls
SCSMT2	<b>Scaling</b> Calcs, <b>Scaling</b> Memory Stress, <b>2</b> 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/wlloydw/CalcsService](https://github.com/wlloydw/CalcsService)

13

## Variation Caused By Multitenancy



14

# FaaS Pricing and Performance Obfuscation

## AWS vs IBM

### AWS Lambda

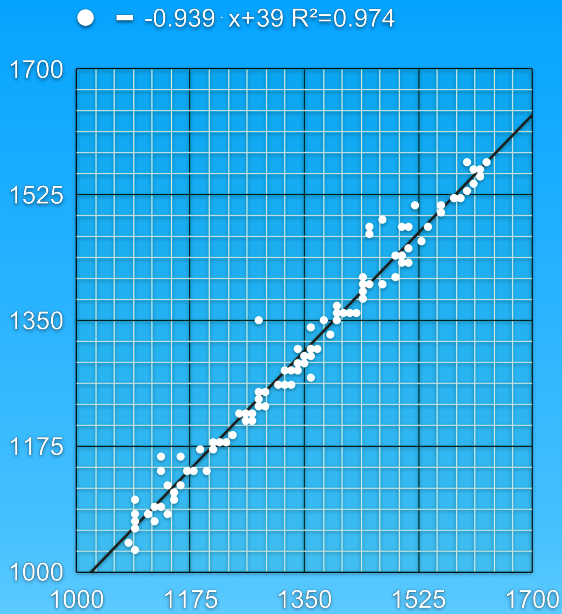
- Price (USD):  
Runtime (s) x Memory (GBs) x \$0.0000166667
- Scales performance with memory setting.
- Tenancy has a small impact on performance.
- For our workloads, a mid range memory setting 512-1024 MBs provides the best price to performance ratio.

### 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.

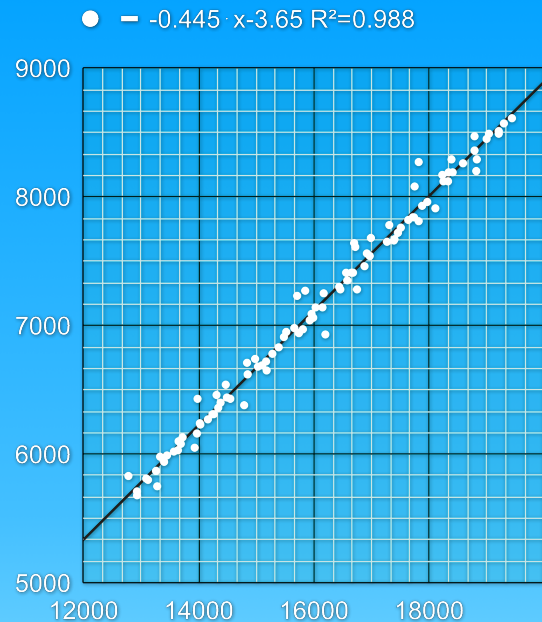
15

cpuUsr (ms) - 2 GBs E5-2680v2 @ 2.8GHz



cpuUsr (ms) - 2 GBs E5-2686v4 @ 2.3 GHz

cpuIdle (ms) - 512 MBs E5-2686v4



cpuIdle (ms) - 256 MBs E5-2686v4

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

16

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

## 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

Repeat for SCNMT2, SCMT2, and SCSMT2 Workloads

17

## Outline

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

18

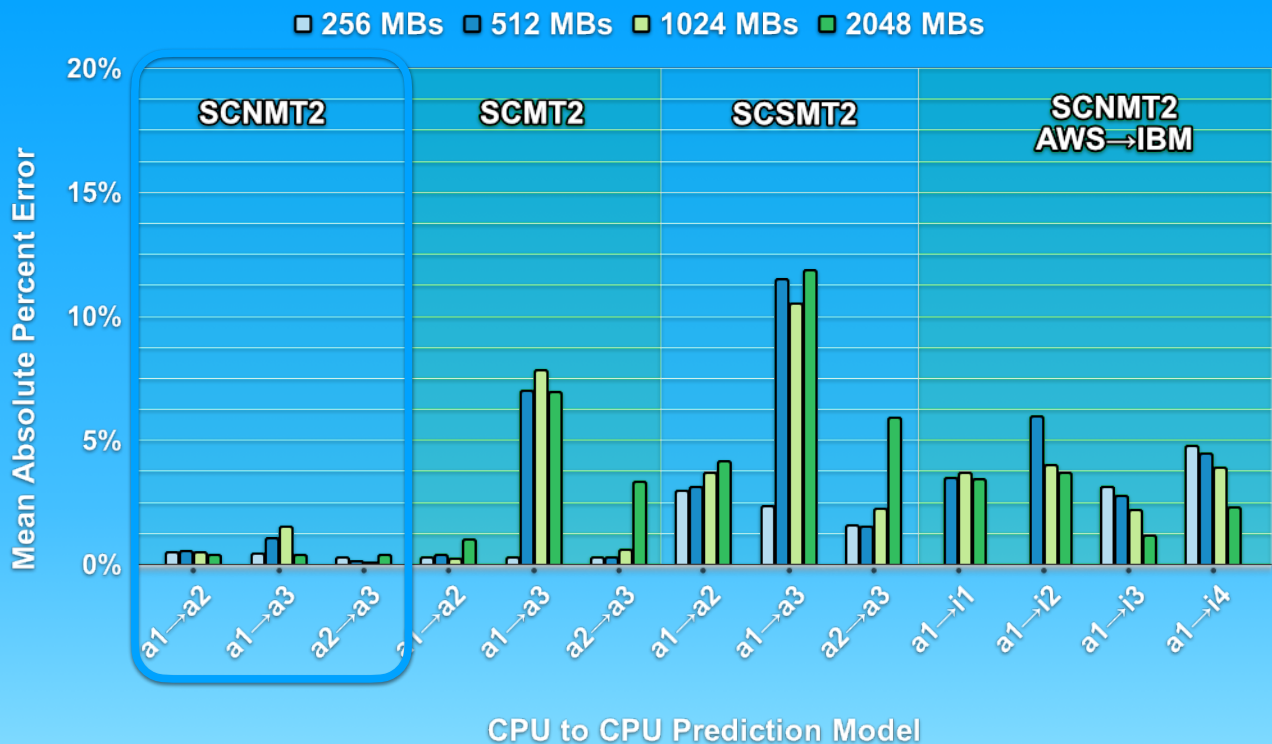


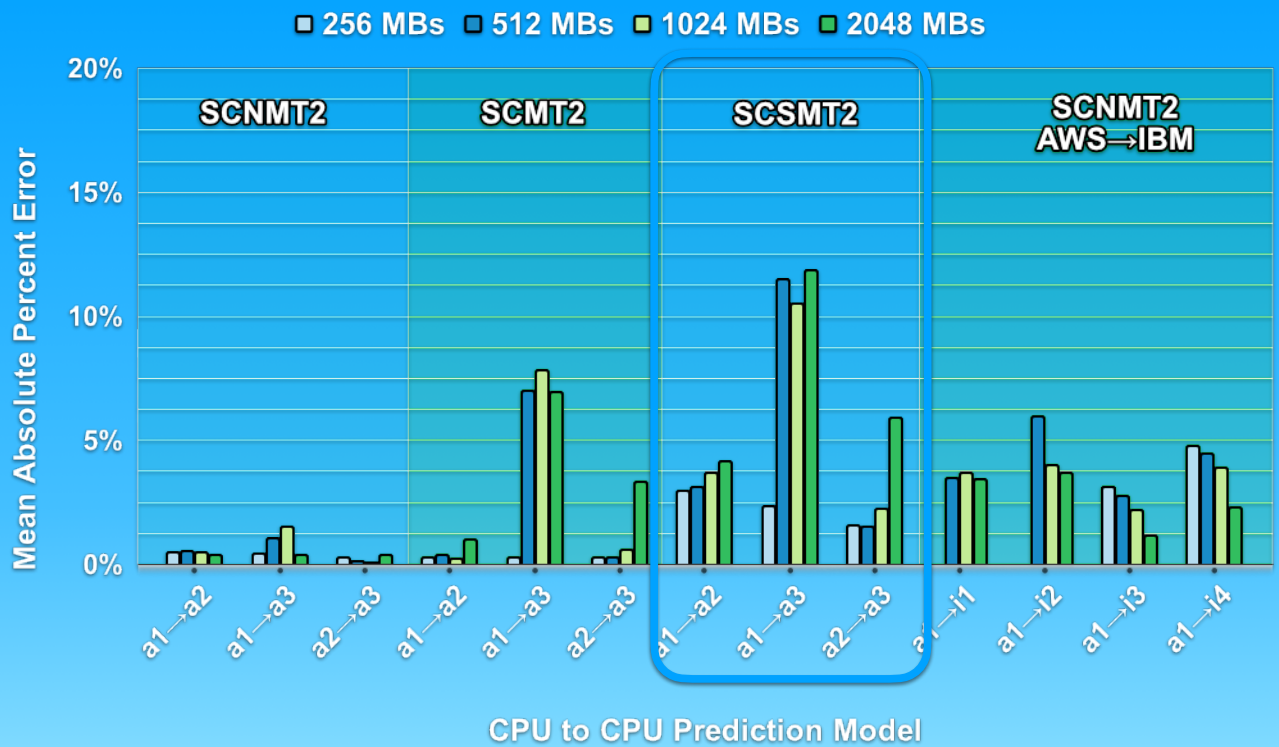
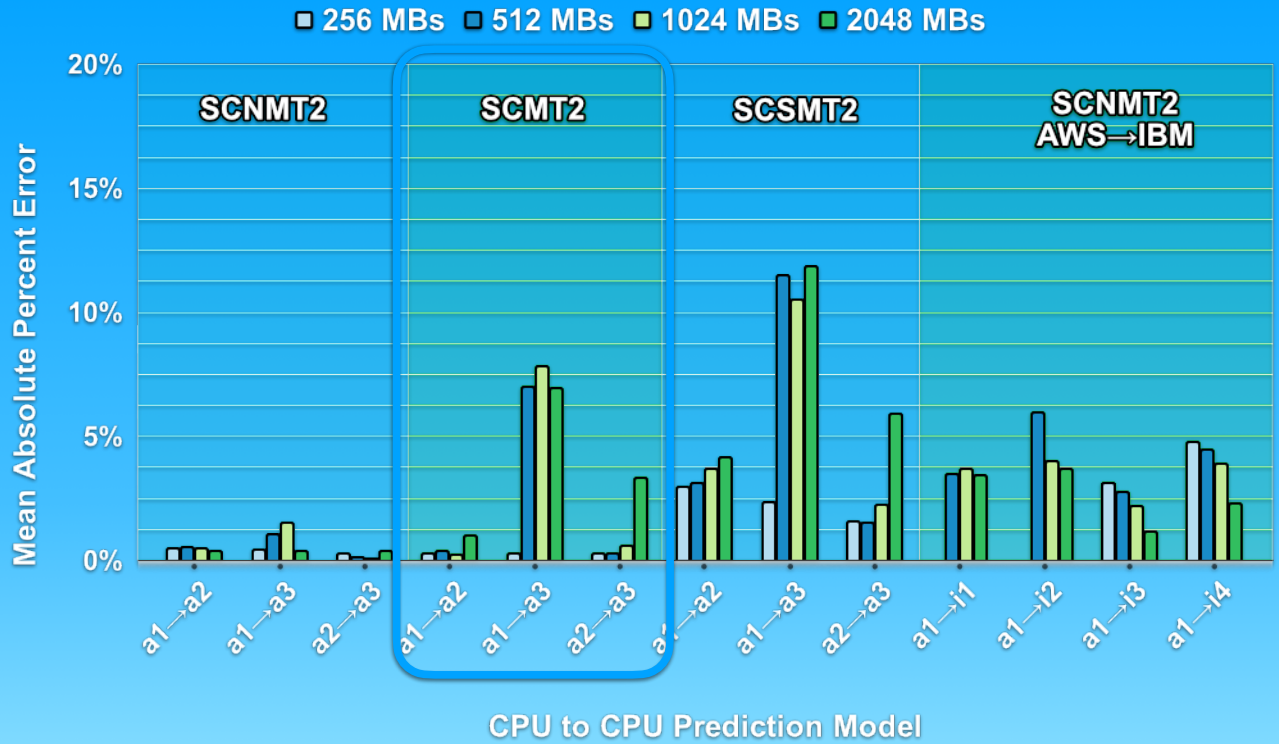
# Workloads for Predictions

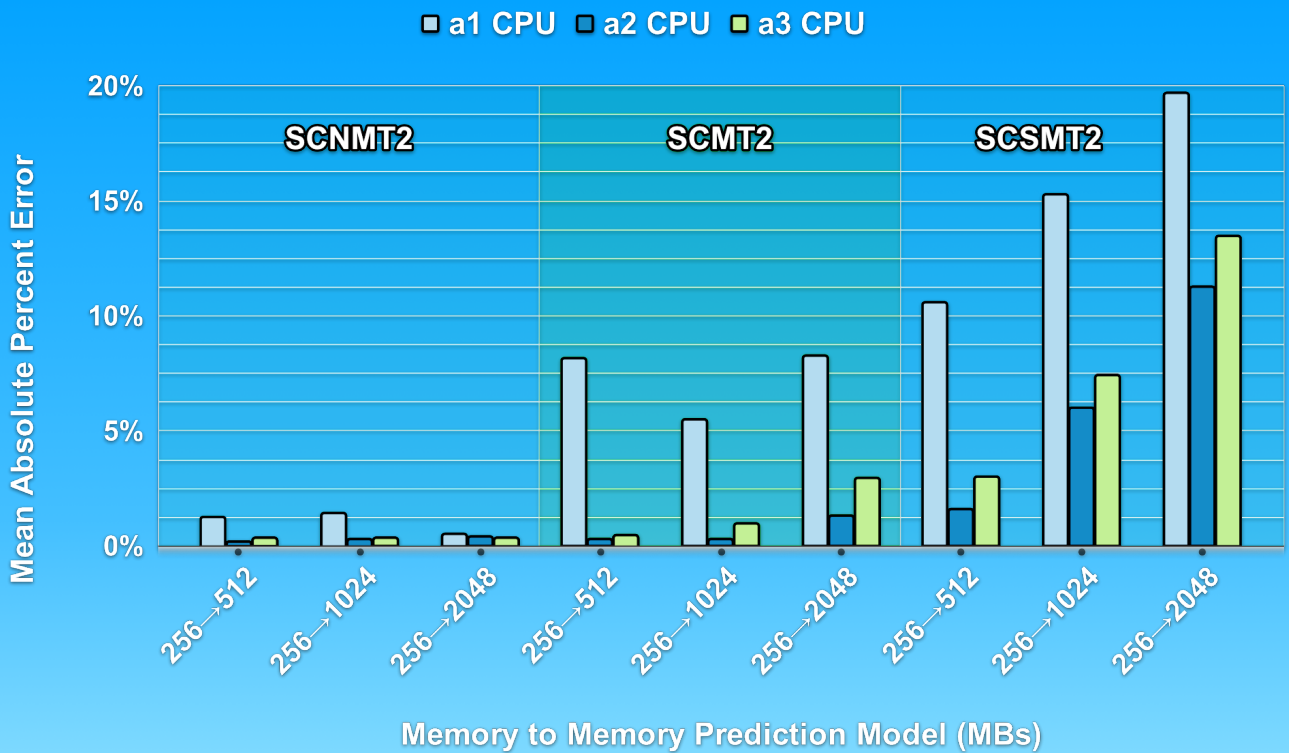
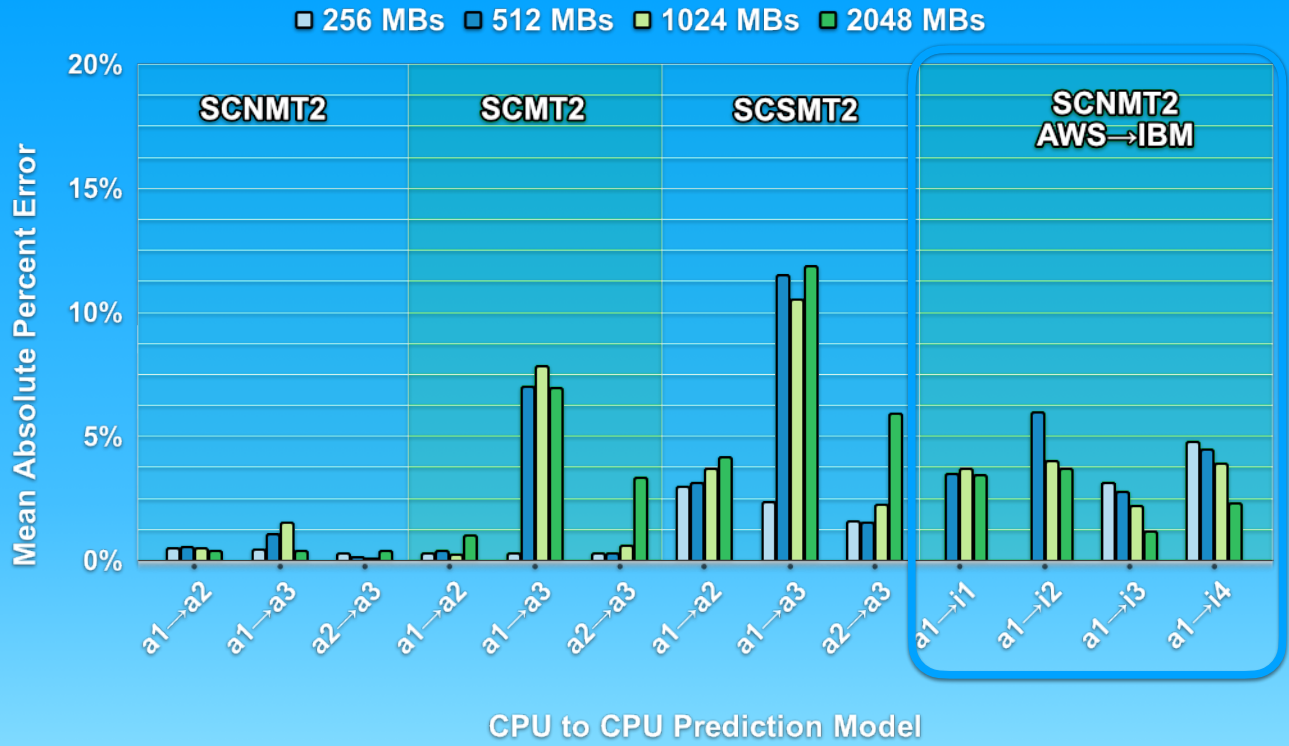
$$a \times b \div c$$

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







# Outline

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

25

## 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

26

## 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
<b>Overall Average</b>	<b>77 (sum)</b>	<b>--</b>	<b>3.49%</b>	<b>\$6.46</b>	<b>\$150.45</b>

27

## Conclusions - RQ-3



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



**cpuSteal**



**freeMemory**

28



# 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](https://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.