

1

Outline

Paper Overview

Related Work

Summary of new technologies

Key contributions Author Evaluation and Conclusion

Strengths and Weaknesses

Future Work

2

Paper Overview

Serverless or FaaS - buzzword in field of cloud computing.

Serverless applications are composed of multiple independent functions, each of which can be implemented in a range of programming languages.

Goal is to compare performance cost analysis between different programming languages used in these functions.

Paper talks about the design and implementation of serverless performance testing framework using AWS Lambda and Microsoft Azure.

Paper evaluation: Python is the best choice, C# is the best economical option and NodeJS should be avoided.

3

Related Work

Papers discussing the emergence of serverless computing, its deployment for cloud applications.

Talks about the paradigm shift from Infrastructure as a service, its advantages and risks.

Talks about characteristics, requirements and the possible challenges to FaaS.

Article about microservices, talks about how microservices are deployed in FaaS.

4

Related Work (Cont.)

Thesis about SaaS using AWS Lambda, majorly using NodeJS to measure the non functional requirements such as security and performance measures.

Papers about cloud computing, talking about the changing cloud infrastructure, benefits of serverless computing and roadmap of challenges that need to be tackled.

Paper talking about deploying machine learning and artificial intelligence applications to FaaS.

5

Summary of New Technology

Performance considerations: Cold-start vs Warm-start.

Investigates real-time latencies between these container considerations.

Cost Considerations: Microservices cost framework called 'CostHat'.

Provides cost information in real-time.

Language Considerations: Different programming languages used to test the performance and cost analysis of FaaS.

Languages used were NodeJS, Go, Python, Java and .NET Core , used on AWS Lambda.

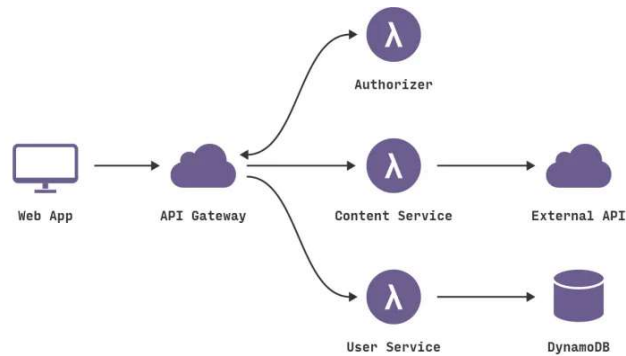
6

Serverless Review

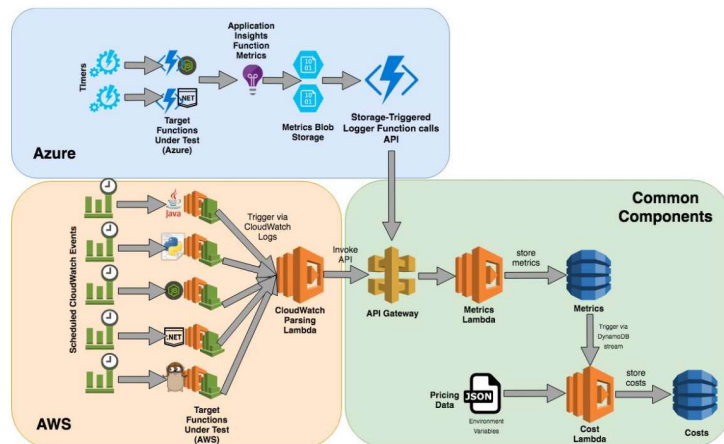
Serverless Performance Considerations

Serverless Cost Considerations

Serverless Cost Modeling

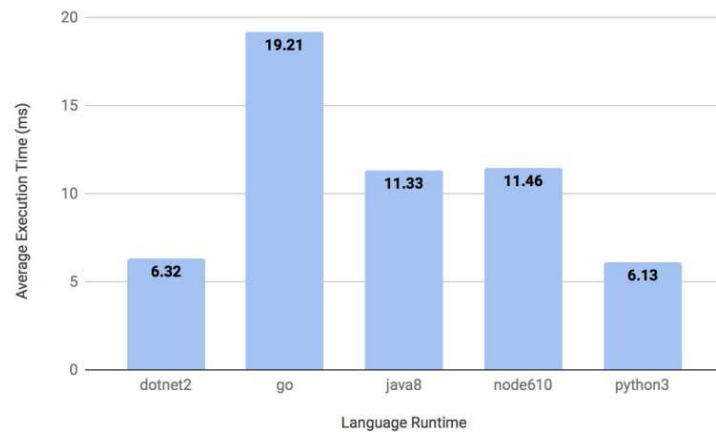


7



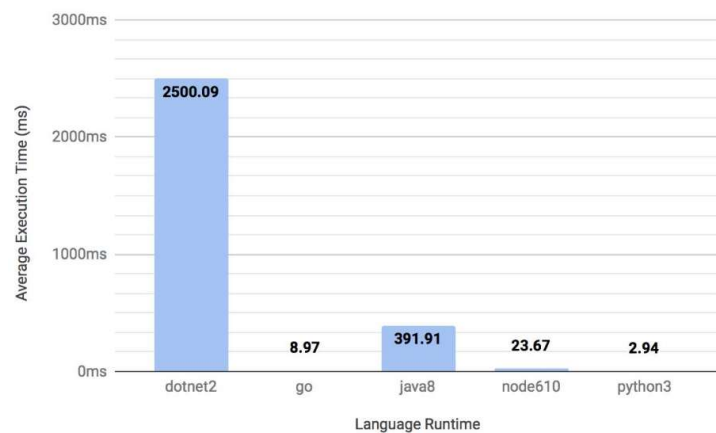
Serverless Performance Framework

8



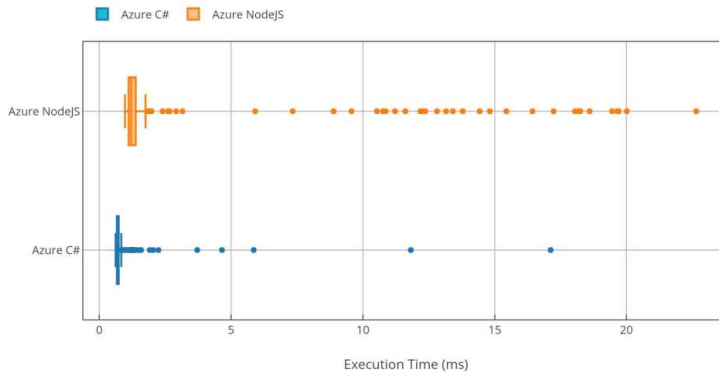
Warm Start Tests on AWS

9



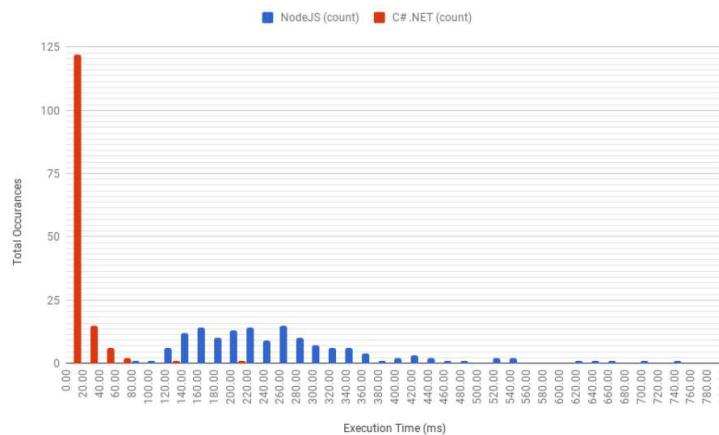
Cold Start Tests on AWS

10



Warm Start Tests on Azure

11



Cold Start Tests on Azure

12

AWS Lambda & Azure functions

Two serverless platforms: AWS Lambda, Azure Functions

Two programming languages: .NET C#, NodeJS

13

Average Performance between Azure and AWS

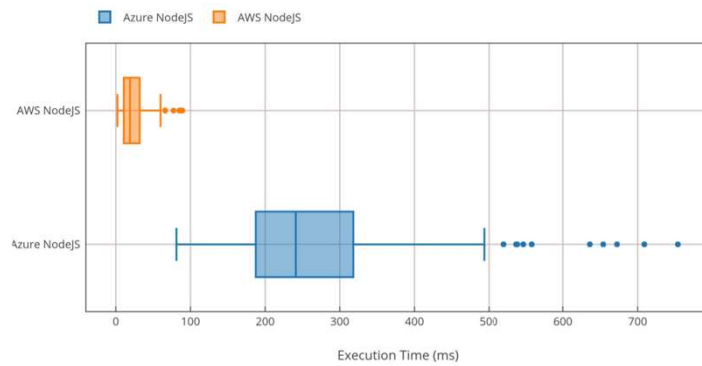
Serverless Platform	Language Runtime	Warm Start Average (ms)	Cold Start Average (ms)
AWS	.NET C#	6.32	2500.09
AWS	NodeJS	11.46	23.67
Azure	.NET C#	0.93	16.84
Azure	NodeJS	4.91	276.42

14

Box Plot of Cold Start Performance

Platforms: Azure, AWS

Language: NodeJS

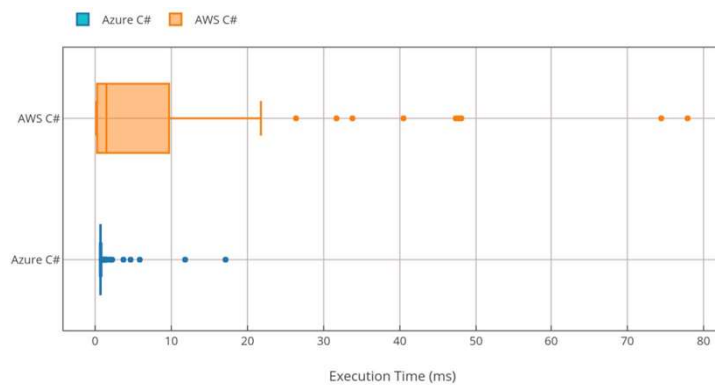


15

Box Plot of Warm-Start Performance

Platforms: Azure, AWS

Language: .NET C#



16

Cost Analysis

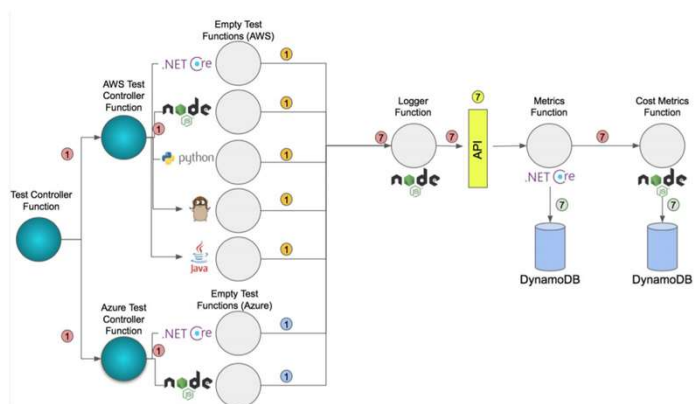
Three main factors:

- Execution time
- Fixed invocation cost per individual function execution
- Memory allocated to the function

Postscript

- Cost per milliseconds of execution: both bill in 100ms blocks.
- "Free-tier" allocations: 1 million executions per month and 400k GB/s of execution time

17



CostHat Model of Modified Serverless Performance Framework

18

AWS: Cold-Start Performance Mapped to Cost

Language Runtime	Average Execution Time (ms)	Average Billed Duration (ms)	Average Cost Per Million (\$)
C# .NET	2500.09	2600.00	5.61775
Golang	8.97	100.00	0.408375
Java 8	391.91	400.00	1.0335
NodeJS	23.67	100.00	0.408375
Python	2.94	100.00	0.408375

19

AWS: Cost of Cold-Start execution Varying Throughput (TPS)

Language Runtime	Cost Per Day @ 100-TPS	Cost Per Day @ 30k-TPS	Cost Per Year @ 100-TPS	Cost Per Year @ 30k-TPS
C# .NET	\$48.54	\$14,561	\$17,716	\$5,314,840
Golang	\$3.53	\$1,059	\$1,288	\$386,355
Java 8	\$8.93	\$2,679	\$3,259	\$977,774
NodeJS	\$3.53	\$1,059	\$1,288	\$386,355
Python	\$3.53	\$1,059	\$1,288	\$386,355

20

Azure: Cost of Cold-Start execution at Varying Throughput (TPS)

Language Runtime	Cost Per Day @ 100-TPS	Cost Per Day @ 30k-TPS	Cost Per Year @ 100-TPS	Cost Per Year @ 30k-TPS
.NET C#	\$3.45	\$1,036.80	\$1,261	\$378,432
NodeJS	\$6.91	\$2,073.60	\$2,523	\$756,864

21

Conclusion

Python is the clear choice on AWS Lambda; C# .NET is the clear best choice for Azure Functions, also across both the two platforms.

Cold-start scenarios expose the cost implications of choosing a poorly performing runtime.

Need to be cautious: NodeJS on Azure Functions, Java and especially C# .NET on AWS Lambda.

Composition of functions in serverless applications is a crucial design decision.

22

Question



23

Thank You!



24