UNIVERSITETI I EVROPËS JUGLINDORE
УНИВЕРЗИТЕТ НА ЈУГОИСТОЧНА ЕВРОПА
SOUTH EAST EUROPEAN UNIVERSITY

Function-as-a-Service Performance Evaluation with Application-level Workflows

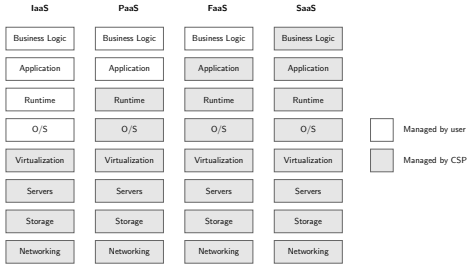
Blend Hamiti
Master Thesis
Mentor: Prof. Dr. Besnik Selimi

Faculty of Contemporary Sciences and Technologies

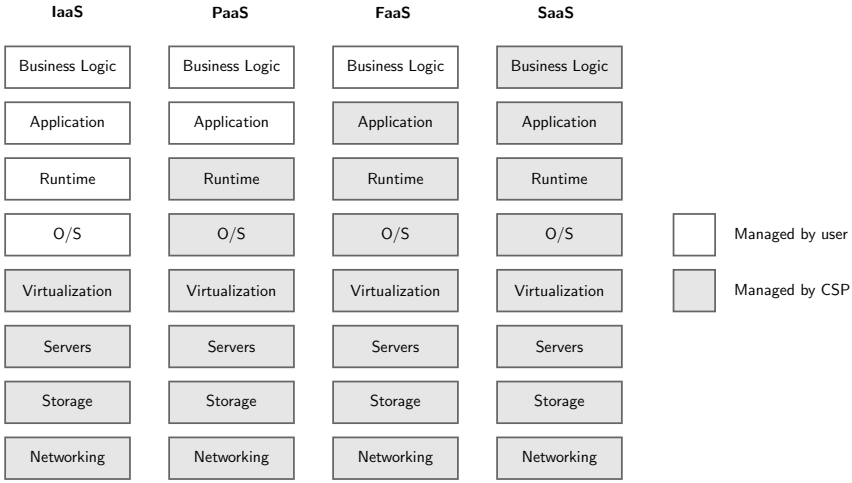
Cloud Computing

Computation services in the cloud

- Applications
- Data storage
- Networking capabilities

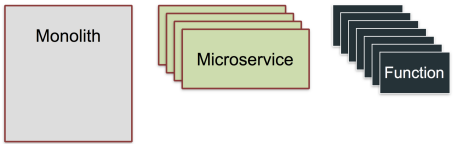


Cloud Computing Models



Function-as-a-Service (FaaS)

- Serverless execution model
- Stateless, event-driven
- Cold start problem



https://dzone.com/articles/introducing-functions-as-a-service-faas

Motivation

- "FaaS Performance Evaluation" systematic literature review [Scheuner+ 2020]
- Shortcomings
  - Micro-benchmarks
  - Performed in AWS
  - Not reproducible

Motivation

- "FaaS Performance Evaluation" systematic literature review [Scheuner+ 2020]
- Shortcomings
  - Micro-benchmarks $\longrightarrow$ **Application-level benchmark**
  - Performed in AWS
  - Not reproducible

Motivation

- "FaaS Performance Evaluation" systematic literature review [Scheuner+ 2020]
- Shortcomings
  - Micro-benchmarks $\longrightarrow$ **Application-level benchmark**
  - Performed in AWS $\longrightarrow$ **Performed in AWS, Azure, GCP**
  - Not reproducible

Motivation

- "FaaS Performance Evaluation" systematic literature review [Scheuner+ 2020]
- Shortcomings
  - Micro-benchmarks $\longrightarrow$ **Application-level benchmark**
  - Performed in AWS $\longrightarrow$ **Performed in AWS, Azure, GCP**
  - Not reproducible $\longrightarrow$ **Reproducibility principles** [Papadopoulos+ 2021]

Motivation

- "FaaS Performance Evaluation" systematic literature review [Scheuner+ 2020]
- Shortcomings
  - Micro-benchmarks $\longrightarrow$ **Application-level benchmark**
  - Performed in AWS $\longrightarrow$ **Performed in AWS, Azure, GCP**
  - Not reproducible $\longrightarrow$ **Reproducibility principles** [Papadopoulos+ 2021]

Goal

Build a reproducable application-level benchmark for AWS, Azure, and GCP.

Research Questions

Hypothesis 1

Performance of each CSP varies between consecutive runs.

Hypothesis 2

Performance of each CSP is different from the others.

Research Questions

Hypothesis 1

Performance of each CSP varies between consecutive runs.

Hypothesis 2

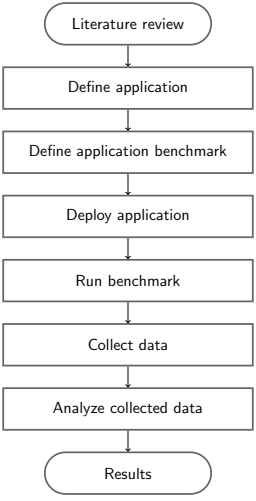
Performance of each CSP is different from the others.

RQ 1 How do we define the benchmark?

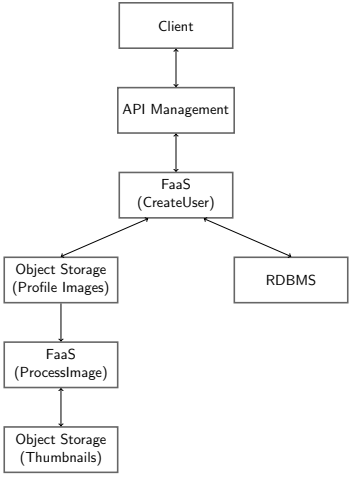
RQ 2 How can we quantify the performance variation in each CSP?

RQ 3 What aspects of the application should be considered to compare performance between CSPs?

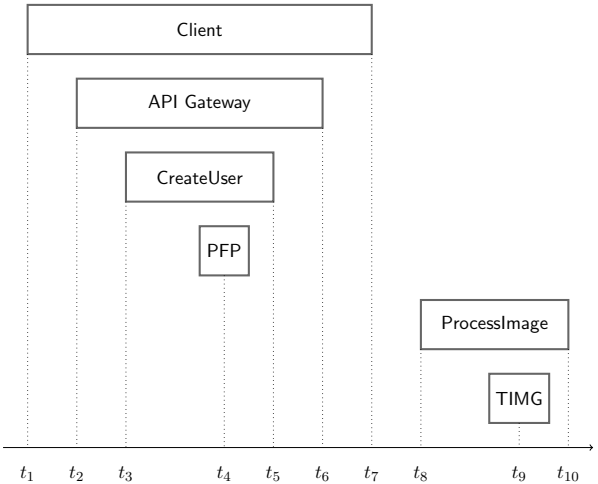
Experiement phases



Phase 1/6: Defining the application

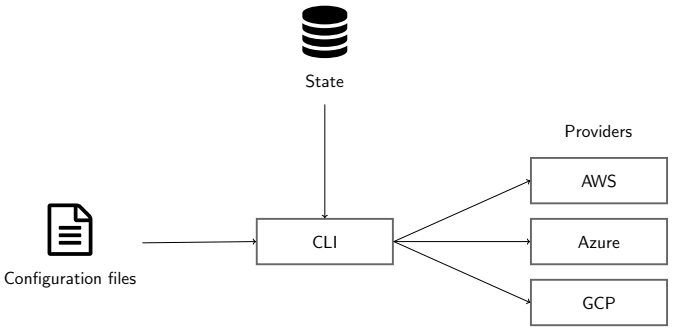


Phase 2/6: Defining the Application Benchmark



Phase 3/6: Deploying the Application

- Using Terraform
- Infrastructure defined in declarative HCL configuration files[1]



[1] https://github.com/blendhamiti/faas-benchmark

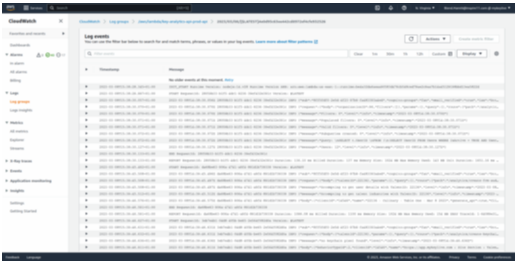
Phase 4/6: Benchmark Execution

- Built mock client program
- 100 measurements for each CSP
- Measurements stored in CSV file [Hamiti 2022]

File header	Timestamp
$client\_request\_timestamp$	t_1
$client\_response\_timestamp$	t_7
$apiGateway\_request\_timestamp$	t_2
$apiGateway\_response\_latency\_ms$	t_6
$createUser\_start\_timestamp$	t_3
$createUser\_end\_timestamp$	t_5
$profileImage\_upload\_timestamp$	t_4
$processImage\_start\_timestamp$	t_8
$processImage\_end\_timestamp$	t_{10}
$thumbnail\_upload\_timestamp$	t_9

Phase 5/6: Data Collection

- Tedious & time consuming
- Multiple logs sources
  - API request logs
  - Function execution logs
- Different methods to collect logs in each CSP
  - Copy manually from UI
  - Intercept network requests
  - Use MQL queries

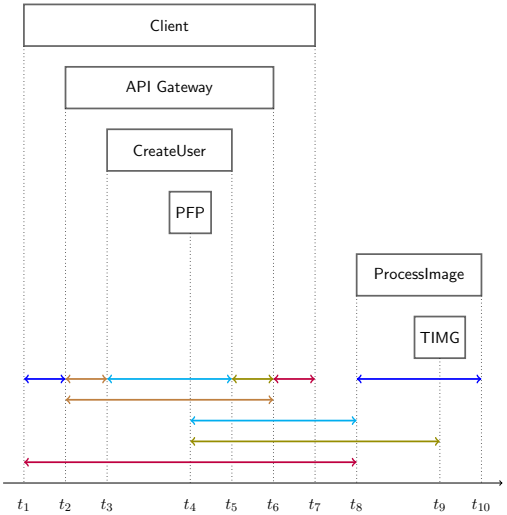


Phase 6/6: Analyzing the Collected Data

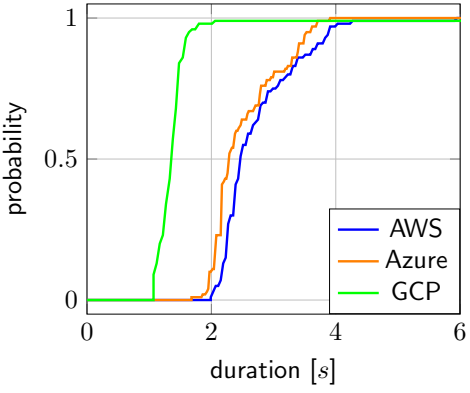
- 10 time durations defined
  - Dur. $t_a t_b$ for each pair (t_a, t_b):

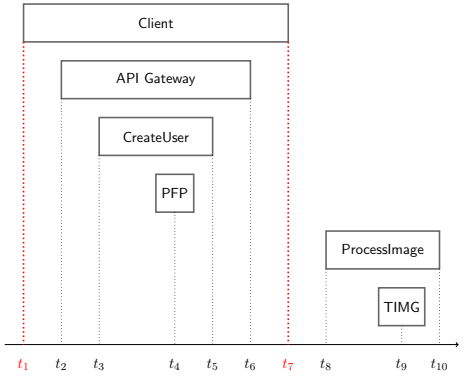
$$t_a t_b = t_b - t_a$$

- Calculations
  - Mean and STD
  - CDF

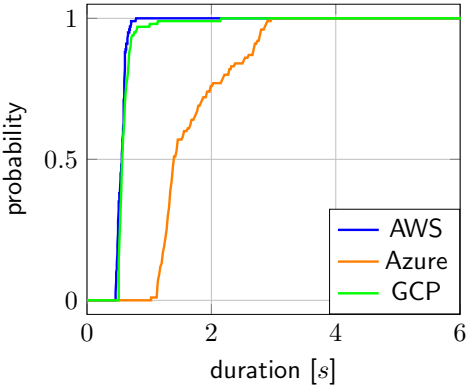


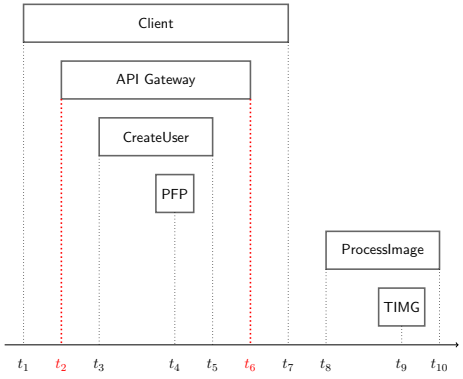
Synchronous Process



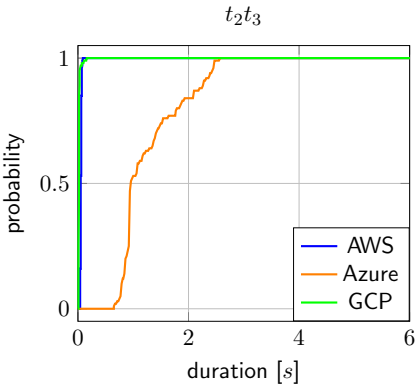


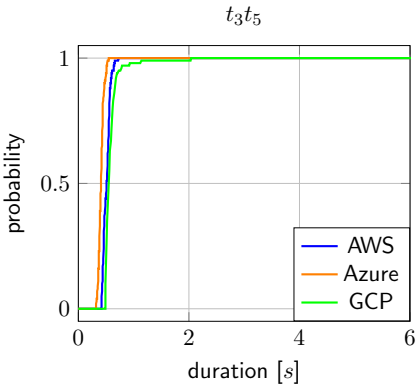
Synchronous Process (w/o network latency)



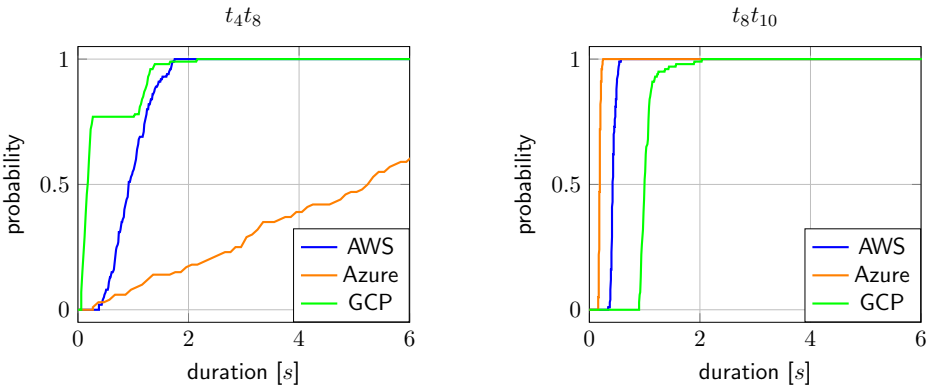


Triggering (left) and Execution (right) of CreateUser Function





Triggering (left) and Execution (right) of ProcessImage Function



Discussion

Hypothesis 1

Performance of each CSP varies between consecutive runs.

RQ 2 How can we quantify the performance variation in each CSP?

A Use COV value of durations to classify into three categories: high, moderate, low.

$< 15\%$ $< 40\%$

$$COV = \frac{STD}{mean} \cdot 100\%$$

Discussion

Hypothesis 1

Performance of each CSP varies between consecutive runs.

RQ 2 How can we quantify the performance variation in each CSP?

A Use COV value of durations to classify into three categories: high, moderate, low.

$< 15\%$ $< 40\%$

	Sync. Function		Async. Function	
	Trigger	Execution	Trigger	Execution
AWS	moderate	high	moderate	high
Azure	low	high	N/A	high
GCP	low	moderate	low	high

Discussion

Hypothesis 1

Performance of each CSP varies between consecutive runs.

RQ 2 How can we quantify the performance variation in each CSP?

A Use COV value of durations to classify into three categories: high, moderate, low.

$< 15\%$ $< 40\%$

	Sync. Function		Async. Function	
	Trigger	Execution	Trigger	Execution
AWS	moderate	high	moderate	high
Azure	low	high	N/A	high
GCP	low	moderate	low	high

Conclusion

Performance consistency is moderate.

Discussion

Hypothesis 2

Performance of each CSP is different from the others.

RQ 3 What aspects of the application should be considered to compare performance between CSPs?

A Look at function trigger and execution durations.

	Sync. Function		Async. Function	
	Trigger	Execution	Trigger	Execution
AWS	0.06 ± 0.01	0.51 ± 0.06	0.98 ± 0.34	0.43 ± 0.04
Azure	1.29 ± 0.54	0.41 ± 0.05	5.28 ± 2.95	0.19 ± 0.02
GCP	0.02 ± 0.02	0.59 ± 0.17	0.41 ± 0.49	1.05 ± 0.17

Discussion

Hypothesis 2

Performance of each CSP is different from the others.

RQ 3 What aspects of the application should be considered to compare performance between CSPs?

A Look at function trigger and execution durations.

	Sync. Function		Async. Function	
	Trigger	Execution	Trigger	Execution
AWS	0.06 ± 0.01	0.51 ± 0.06	0.98 ± 0.34	0.43 ± 0.04
Azure	1.29 ± 0.54	0.41 ± 0.05	5.28 ± 2.95	0.19 ± 0.02
GCP	0.02 ± 0.02	0.59 ± 0.17	0.41 ± 0.49	1.05 ± 0.17

Conclusion

There are significant performance differences between the platforms.

Conclusion

Summary

- Performance consistency is moderate
- Significant performance differences between the platforms

Further Work

- Re-running benchmark to verify current results
- Including more cloud platforms & services
- Making log data collection more practical

References

Blend Hamiti. **Resources for "Function-as-a-Service Performance Evaluation with Application-level Workflows"**. Version 0.1. Sept. 2022. DOI: 10.5281/zenodo.7112754.

Alessandro Vittorio Papadopoulos, Laurens Versluis, André Bauer, Nikolas Herbst, Jóakim von Kistowski, Ahmed Ali-Eldin, Cristina L. Abad, José Nelson Amaral, Petr Tůma and Alexandru Iosup.
Methodological Principles for Reproducible Performance Evaluation in Cloud Computing. In: *IEEE Transactions on Software Engineering* 47.8 (Aug. 2021). Conference Name: IEEE Transactions on Software Engineering, pp. 1528–1543. ISSN: 1939-3520. DOI: 10.1109/TSE.2019.2927908.

Joel Scheuner and Philipp Leitner. **Function-as-a-Service performance evaluation: A multivocal literature review**. en. In: *Journal of Systems and Software* 170 (Dec. 2020), p. 110708. ISSN: 0164-1212. DOI: 10.1016/j.jss.2020.110708.

Calculated durations

	AWS			Azure			GCP		
	Mean	STD	COV	Mean	STD	COV	Mean	STD	COV
t_1t_7	2.74	0.64	23.2%	2.52	0.54	21.2%	1.41	0.52	36.8%
t_2t_6	0.56	0.06	11.5%	1.69	0.55	32.6%	0.62	0.18	29.6%
t_1t_2	2.00	0.62	31.1%	0.69	0.15	21.6%	0.64	0.36	55.8%
t_6t_7	0.18	0.07	39.7%	0.14	0.08	52.3%	0.15	0.06	38.0%
t_2t_3	0.06	0.01	18.1%	1.29	0.54	42.2%	0.02	0.02	78.6%
t_3t_5	0.51	0.06	12.2%	0.41	0.05	11.6%	0.59	0.17	29.2%
t_5t_6	-0.01	0.01	53.1%	-0.01	0.01	152.3%	0.00	0.00	40.4%
t_4t_8	0.98	0.34	34.5%	5.28	2.95	55.8%	0.41	0.49	118.7%
t_8t_{10}	0.43	0.04	10.0%	0.19	0.02	9.3%	1.05	0.17	16.3%
t_4t_9	1.41	0.34	24.1%	5.47	2.95	54.0%	1.46	0.59	40.6%

Threats to Validity

- Construct validity
  - Too few cloud services
- Internal validity
  - Mismatch between services CSPs
  - Database is public
  - No authorization rules for API (and functions)
  - Errors during data collection
- External validity
  - Not enough measurements

Reproducibility Principles

P1: Repeated experiments

P2: Workload and configuration coverage

P3: Experimental setup description

P4: Open access artifact

P5: Probabilistic result description

P6: Statistical evaluation

P7: Measurement units

P8: Cost

Application Configuration

- FaaS Function Memory Configuration
  - 512MB
- FaaS Function Runtime
  - Node.js 14
  - Linux & Windows
- FaaS Function Triggers
  - Event-based
  - Polling
- DMBS
  - MySQL 5.7
  - Deployed in VPC
- API Gateway Routes
  - Single HTTP POST route

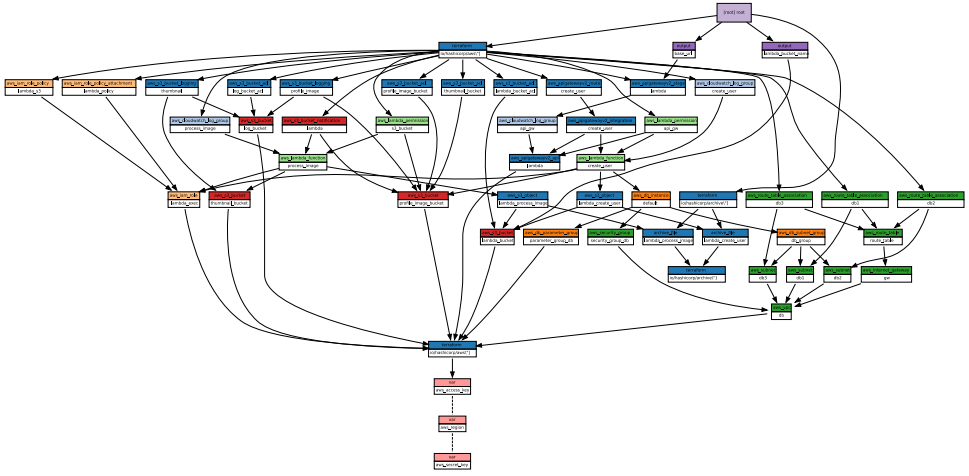
Application Database Schema

```
CREATE TABLE `seeu_db`.`users` (
  `user_id` int(11) NOT NULL AUTO_INCREMENT,
  `email` varchar(90) NOT NULL,
  `password` varchar(255) NOT NULL,
  `name` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`user_id`),
  UNIQUE KEY `email_UNIQUE` (`email`)
);
```

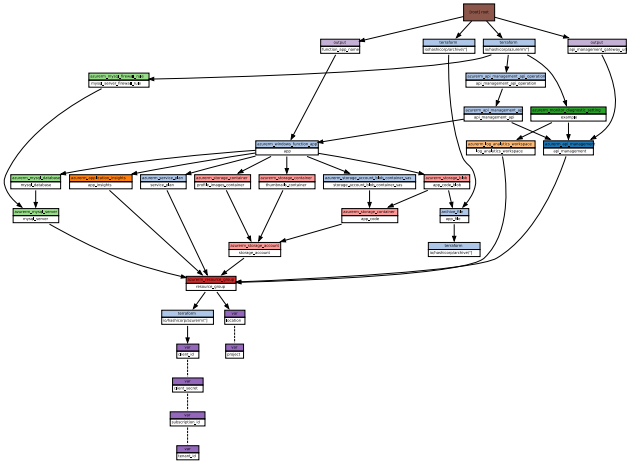
Cloud Services Used

Service type	AWS	Azure	Google Cloud
API Management	API Gateway	API Management	API Gateway
FaaS	Lambda	Functions	Cloud Functions
Logging	CloudWatch	Cloud Logging	Monitor
Object Storage	Simple Storage Service	Blob Storage	Cloud Storage
RDBMS	RDS	Database for MySQL	Cloud SQL

AWS Infrastructure



Azure Infrastructure



GCP Infrastructure