ESSCA2018 – European Symposium On Serverless Computing
December 21, 2018. Zurich, Switzerland

An Effective Resource Management Approach in a FaaS Environment

Andreas Christoforou, Andreas Andreou

Department of Electrical Engineering, Computer Engineering and Informatics,
Cyprus University of Technology, Cyprus

Cyprus University of Technology

SEIIS
Software Engineering and Intelligent Information Systems Research Lab

DOSSIER-Cloud

Horizon 2020
Project no. 692251,
DOSSIER-Cloud

Outline

- Serverless Computing
- Motivation
- Proposed Approach
- Experimental Process
- Results and Discussion
- Conclusions and Future Work

Serverless Computing

Definition

- Emerged through the continuous and vast development of the **Cloud**
- Provides a service in which developers can **write** and **deploy** code **without** provisioning or managing **servers** or containers

Impact

- On several software engineering aspects such as **development process**, **pricing model** and Quality of Service (**QoS**) assurance

Weaknesses

- It is not suitable for **long term tasks** because of the limited time a service can run
- There is increasing **complexity** of the underlying architecture

Serverless Computing

Function as a Service (FaaS)

- The main representative of this new service
- Can be triggered through an API call or by an event

Major Serverless Providers

- AWS Lambda
- IBM Cloud Functions (Apache OpenWhisk)
- Google Cloud Functions
- Microsoft Azure Functions

Motivation

Problem

- The identification of the **optimum scenario** for resource allocation to serve adequately a specific workload is a **tedious**, **computationally complex** and **time-consuming** process since multiple objectives need to be satisfied

Why it is important

- It is essential for the **software development process** itself, which is directed towards **satisfying the SLA** and providing **QoS assurance**

- **RQ1:** Is it possible to implement **easy to use** and **efficient** resource management algorithms in a FaaS platform?
- **RQ2:** How intelligent techniques can deliver **efficient** resource management to developers in a FaaS environment with the **minimum** possible **cost** and **time**?

Approach

Aim

- Allocate a sufficient amount of resources in a FaaS environment to serve a specific workload adequately

Step 1: Identify the optimal solution for both objectives **cost** and **performance** by utilizing an **exhaustive algorithm** on a low demand environment and a small-scale workload

- Brute-force search

Step 2: Apply **intelligent algorithms** over the results obtained from step1 aiming to reach to solutions **faster** and **cheaper**

- Multi-Objective Genetic Algorithms (MOGAs)

Target : Provide the **decision makers** with the set of **optimal solutions** and support them to **take decisions** as to which values of the decision variables are **most suited** based on the **targets** and the **requirements** of their application

Approach

Genetic Algorithms

- Type of evolutionary algorithms, which are widely used to solve search-based optimization problems

Multi-objective Genetic Algorithms

- Applied in case of problems that require simultaneous optimization of **multiple criteria**
- In case of conflicting or competing objectives, deliver a set of **optimal solutions** (**Pareto front**) instead of a single one
- Each optimal solution constitutes a specific balance between the objectives under optimization

Approach

Our Application

Platform

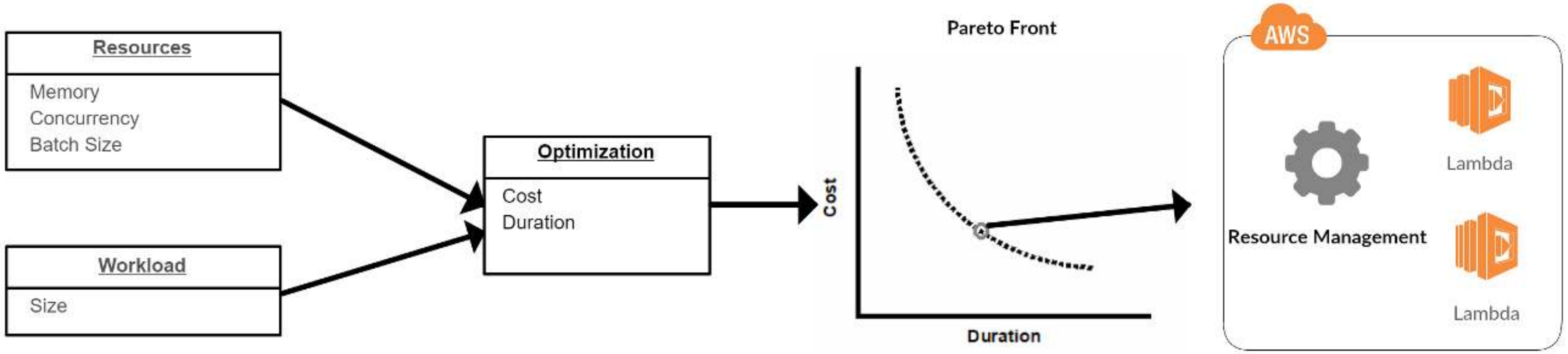
- AWS Lambda

Decision variables (candidate solutions)

- Memory allocation
- Maximum concurrency functions
- Batch size

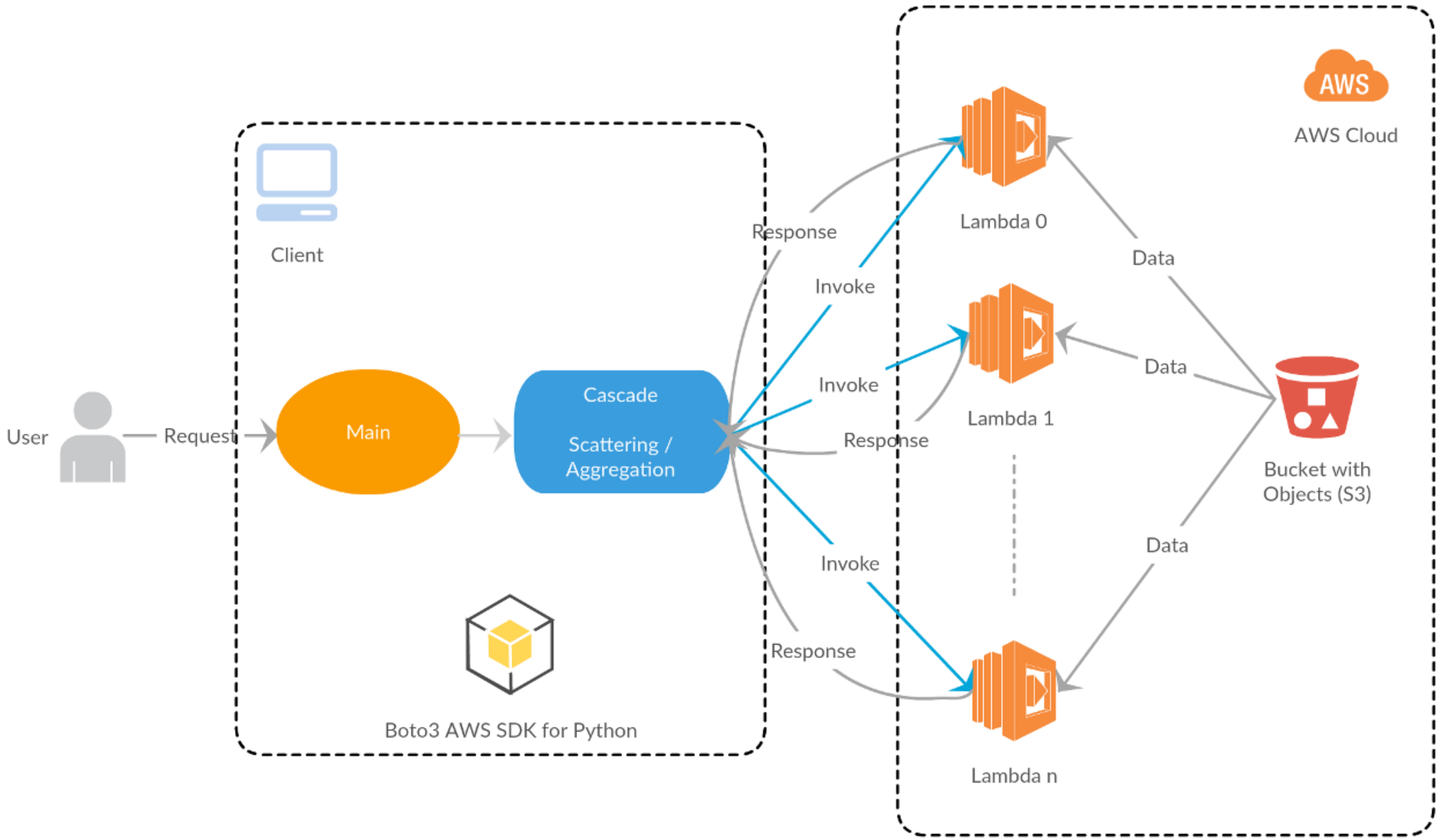
Optimization Objectives

- Minimize Cost ($)
- Maximize Performance (minimize duration) (*ms*)



Experimental Process

Experimental Environment



Experimental Process

Experimental Environment

- Our multi-objective optimization approach was adjusted and configured based on the **AWS Lambda** platform considering the available options offered
- **Cost objective** is the minimization of the total cost required for the completion of the process of the input workload
  - Calculated using the formulas and rules as these are given by Amazon
  - Depends on:
    - the number of lambda functions executions
    - the total duration of all executed functions
    - the allocated memory
- **Performance objective** is the minimization of the total duration needed for the workload process completion
  - Calculated as the time from the moment the user sends the start request until the application delivers back to the user the total count of words

Experimental Process

Decision Variables

1. Memory allocation
   - Denotes the amount of memory you want to allocate for your lambda function
   - Can get values ranging from **128MB** to **3008MB** with **64MB** increment step
2. Concurrent execution limit
   - Can be set from **1** to **1000**
3. Batch size
   - Represents the number of files that each function will process
   - Is relative to the percentage to the workload size and fall into the following set: [1, 2, 5, 10, 20, 25, 50, 100]

Workload

- **100** text files with each file containing **638** words stored in a **S3 bucket**

***To reduce execution time and cost we discarded solutions for concurrency limit over 100**

Experimental Process

Exhaustive Algorithm

- Executed and delivered all possible candidate solutions
- The number of ***Possible Solutions*** (**PS**) is calculated to be equal to **36800**

$$|PS| = \sum_{1}^{N} \sum_{1}^{M} \sum_{1}^{K}$$

$$where,\ N{=}1..100,\ M{=}1..46,\ K{=}1..8$$

Experimental Process

Multi-objective Genetic Algorithms

- **Three**(3) well-known and widespread **MOGAs** were selected to asses their ability to solve the problem
  - The Non-dominated Sorting Genetic Algorithm II (**NSGA-II**)
  - The Non-dominated Sorting Genetic Algorithm III (**NSGA-III**)
  - The Strength Pareto Evolutionary Algorithm 2 (**SPEA2**)
- Implementation was performed using **Platypus**[1], a Python based multi-objective optimization algorithms library
- **Basic configurations**
  - Best practices for setting the MOGAs configuration have been used
  - Crossover operator: Simulated Binary Crossover (SBX)
  - Mutation operator: Polynomial Mutation (PM)
- **Objective**

$$minimize f(x) = (f_{duration}(x), f_{cost}(x)), x \in PS$$

[1] https://platypus.readthedocs.io/en/latest/index.html

Results

Exhaustive algorithm execution

- Delivered a complete list of ***Possible Solutions*** (**PS**) which constitute the aggregation of five different executions

MOGAs execution

- Used the results from the exhaustive algorithm
- Each MOGA was run **100** times for different values of ***Fitness Evaluations (FE)*** ranging from **500** to **4500** with increment step **500**

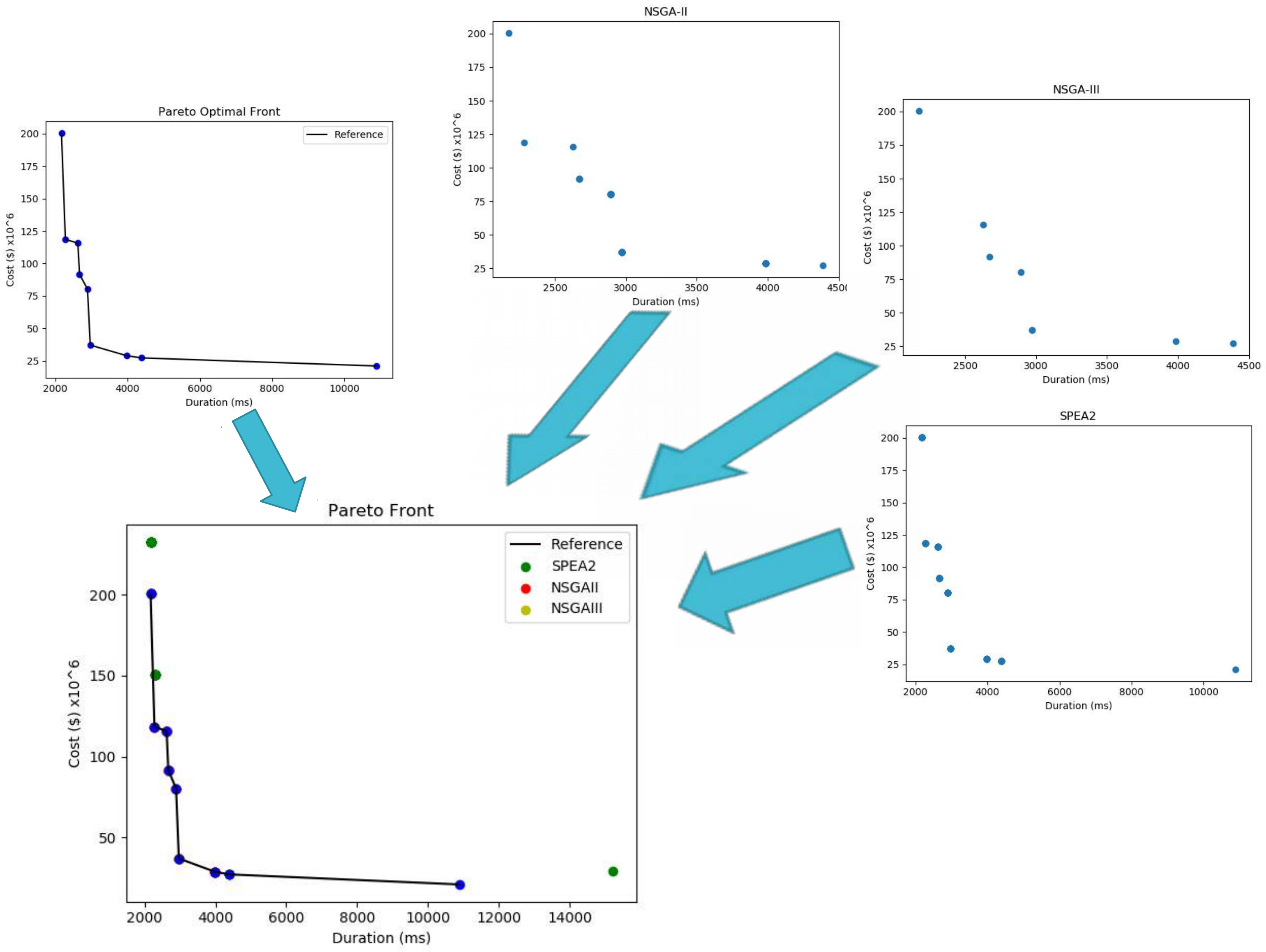
Pareto front

- **Pareto optimal front** calculated based on reference optimal solutions
- 3 **Pareto near-optimal solutions** calculated based on the three MOGAs

MOGAs performance comparison

- The ***Hypervolume (HV)*** and the ***Inverted Generational Distance (IGD)*** quality indicators were utilized to assist the performance comparison

Results


Pareto Optimal Front
Reference
Cost ($) x10^6
Duration (ms)
NSGA-II
Cost ($) x10^6
Duration (ms)
NSGA-III
Cost ($) x10^6
Duration (ms)
SPEA2
Cost ($) x10^6
Duration (ms)
Pareto Front
Reference
SPEA2
NSGAII
NSGAIII
Cost ($) x10^6
Duration (ms)

Results

Discussion

- **Observing the Pareto fronts:**
  - All three MOGAs approached the optimal solutions to a high degree
- **Observing quality indicators:**
  - The differences observed the compared algorithms are too small
  - **HV indicator**:
    - **SPEA2** presents the best performance with second the **NSGA-II** and last the **NSGA-III**
  - **IGD indicator**
    - None of the algorithms seems to prevail

Conclusions

Summary

- This research work performed a **preliminary investigation** to assess whether **heuristic approaches** for **multi-objective optimization** are able to solve the problem of finding a set of near-optimal solutions and support developers in a FaaS environment to select an **efficient resource allocation** scheme with respect to **cost** and **time**
- This assessment has been **verified** through the experimental process followed

Future Work

- Investigate whether these heuristic approaches are able to support **real-time efficient resource allocation** over workloads with **unknown characteristics**

Thank You!

Andreas Christoforou

andreas.christoforou@cut.ac.cy