A Multi-Objective Scheduling Policy for Edge Serverless Platforms

New Challenges in Scheduling Theory - Aussois 2022

Anderson Andrei Da Silva,
Advisors: Y. Georgiou, M. Mercier, G. Mounié, D. Trystram.
Université Grenoble Alpes, Ryax Technologies

Contact: anderson-andrei.da-silva@inria.fr, silva.andersonandrei@gmail.com
https://andersonandrei.github.io/

16 of May of 2022

Introduction

Serverless Computing

- It has emerged as a new paradigm of abstraction, platform and implementation of cloud functions [Barcelona-Pons et. al, 2019; Baldini et. al, 2017].

Serverless Computing

- It has emerged as a new paradigm of abstraction, platform and implementation of cloud functions [Barcelona-Pons et. al, 2019; Baldini et. al, 2017].It has been characterized by mainly two points:
  - a simplified programming model, abstracting the operational concerns.
  - a billing model based on the functions execution time instead of the resource allocation.

Serverless Computing

- It has emerged as a new paradigm of abstraction, platform and implementation of cloud functions [Barcelona-Pons et. al, 2019; Baldini et. al, 2017].It has been characterized by mainly two points:
  - a simplified programming model, abstracting the operational concerns.
  - a billing model based on the functions execution time instead of the resource allocation.
- By default, it is possible to deploy rapidly stateless functions which respond to events. And, they can be coordinated to behave as micro-services and to be statefull.

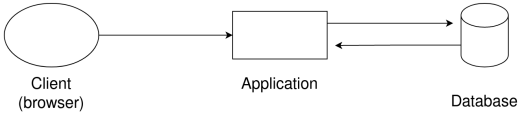
Serverless Computing

- It has emerged as a new paradigm of abstraction, platform and implementation of cloud functions [Barcelona-Pons et. al, 2019; Baldini et. al, 2017].It has been characterized by mainly two points:
  - a simplified programming model, abstracting the operational concerns.
  - a billing model based on the functions execution time instead of the resource allocation.
- By default, it is possible to deploy rapidly stateless functions which respond to events. And, they can be coordinated to behave as micro-services and to be statefull.
- It presents an evolution of the Cloud Computing model in the sense of use of micro-services and containers.

Serverless Computing

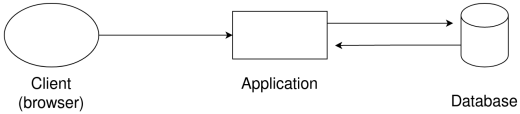
- It has emerged as a new paradigm of abstraction, platform and implementation of cloud functions [Barcelona-Pons et. al, 2019; Baldini et. al, 2017].It has been characterized by mainly two points:
  - a simplified programming model, abstracting the operational concerns.
  - a billing model based on the functions execution time instead of the resource allocation.
- By default, it is possible to deploy rapidly stateless functions which respond to events. And, they can be coordinated to behave as micro-services and to be statefull.
- It presents an evolution of the Cloud Computing model in the sense of use of micro-services and containers.
  - A container is a package of code and dependencies that allow applications to be executed easily. They are composed by layers, which contains separeted parts of the dependencies or the code.
    - Through our experiments we identified that the creation of containers can take considerable time, depending on the functions depenendecies. But, due to the containers composition - layers - they can share common data.

Micro-services and Containers

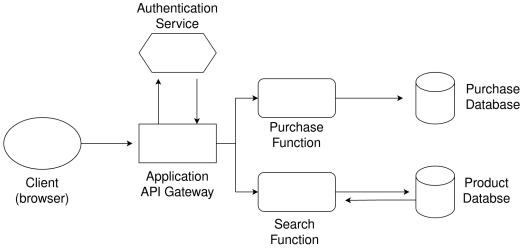


(a) Example of a monolithic architectural designed application.

Micro-services and Containers

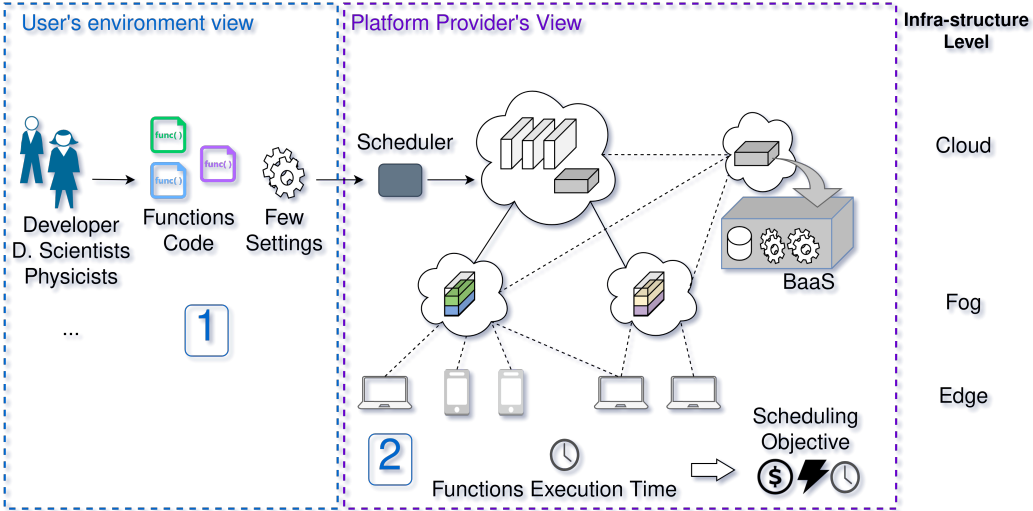


(a) Example of a monolithic architectural designed application.



(b) A micro-services approach.

Scheduling of Functions on Heterogeneous Edge Serverless Platforms

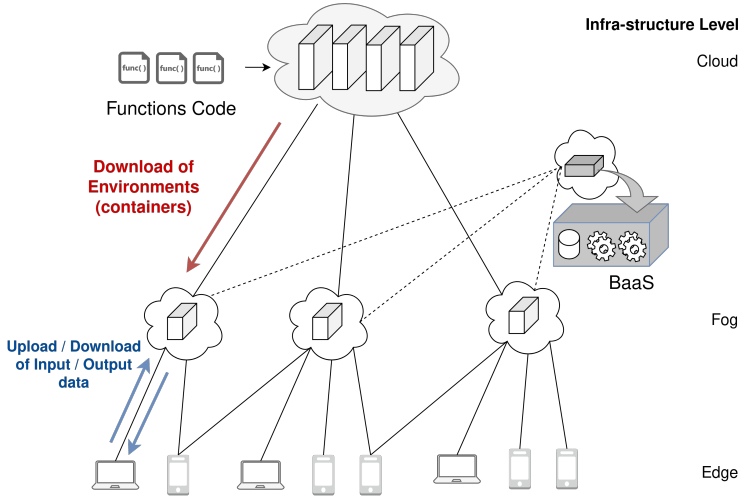


FOA: Function Orchestration Algorithm

Inpired by the algorithm of Shmoys and Tardos [1].

[1] David B. Shmoys and Éva Tardos. 1993. An Approximation Algorithm for the Generalized Assignment Problem. Math. Program. 62, 1–3 (Feb. 1993), 461–474.

Motivation: A Multi-Objective Approach



FOA's Linear Program - Assumptions and Variables

Assumptions:

1. Functions and environments are independent and known in advance;
2. Their cost and processing time on each machine are known;
3. The dependency between functions and environments is known.

FOA's Linear Program - Assumptions and Variables

Assumptions:

1. Functions and environments are independent and known in advance;
2. Their cost and processing time on each machine are known;
3. The dependency between functions and environments is known.

List of variables:

- number of machines M; number of functions N; number of environments K;
- cost of a function on a machine $c[0..M][0..N]$;
- time of a function on a machine $p[0..M][0..N]$;
- cost of a environment on a machine $d[0..M][0..K]$;
- time of a environment on a machine $b[0..M][0..K]$.
- function j needs environment $env[j]$, with $env[0..N]$;
- function placement on machines $x[0..M][0..N]$, with $x_{ij} >= 0$;
- environment placement on machines $e[0..M][0..K]$, with $e_{ik} >= 0$.

FOA's Linear Program

The Linear Program minimizes the cost under a makespan constraint of value T

$$Min \sum_{i=1}^{M} \sum_{j=1}^{N} c_{ij} \times x_{ij} + \sum_{i=1}^{M} \sum_{k=1}^{K} d_{ik} \times e_{ik} \quad (1)$$

$$\forall j \leq N, \quad \sum_{i=1}^{M} x_{ij} \geq 1 \quad (2)$$

$$\forall i \leq M, \quad \sum_{j=1}^{N} p_{ij} \times x_{ij} + \sum_{k=1}^{K} b_{ik} \times e_{ik} \leq T \quad (3)$$

$$\forall i \leq M, \forall j \leq N, \quad x_{ij} \leq e_{i,env[j]} \quad (4)$$

$$\forall i \leq M, \forall k \leq K, \quad \sum_{j=1}^{N} x[i][j] \leq N_{ik} \times e[i][k] \quad (5)$$

$$\forall i \leq M, \forall k \leq K, \quad e_{ik} \leq 1 \quad (6)$$

Methodology

Methodology

First phase: Understanding Serverless Platforms' behaviors.

- Adaptation of the functions available on the benchmark FunctionBench[2].,
- Deployment of the Serverless Platform OpenWhisk[3] on top of the cluster GRID5000[4],
- Measurement and calibration of results from the execution of the adapted benchmark on the deployed Serverless Platform.

[2] https://github.com/kmu-bigdata/serverless-faas-workbench

[3] https://openwhisk.apache.org/

[4] https://www.grid5000.fr

[5] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, scalable, and accurate simulation of distributed applications and platforms," Journal of Parallel and Distributed Computing, vol. 74, no. 10, pp. 2899–2917, Jun. 2014. [Online]. Available: http://hal.inria.fr/hal-01017319

Methodology

First phase: Understanding Serverless Platforms' behaviors.

- Adaptation of the functions available on the benchmark FunctionBench[2].,
- Deployment of the Serverless Platform OpenWhisk[3] on top of the cluster GRID5000[4],
- Measurement and calibration of results from the execution of the adapted benchmark on the deployed Serverless Platform.

Second Phase: Evaluation of scheduling policies (FOA, baseline) in a simulated environment on top of Batsim/ Simgrid[5], using the calibrated results.

[2] https://github.com/kmu-bigdata/serverless-faas-workbench

[3] https://openwhisk.apache.org/

[4] https://www.grid5000.fr

[5] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, scalable, and accurate simulation of distributed applications and platforms," Journal of Parallel and Distributed Computing, vol. 74, no. 10, pp. 2899–2917, Jun. 2014. [Online]. Available: http://hal.inria.fr/hal-01017319

Baseline: Kubernetes' Cache Locality Priority

Kubernetes[6] is an open-source system for automating deployment, scaling, and management of containerized applications.

[6] https://kubernetes.io/

Baseline: Kubernetes' Cache Locality Priority

Kubernetes[6] is an open-source system for automating deployment, scaling, and management of containerized applications.

We extracted one component of its scheduling policy, and adapted as follow:

Require: functions_queue, machines_available
  while functions_queue is not empty **do**
    f ← functions_queue[0]
    container_required ← f.container
    machines_candidates ← sort(machines_available, container_required)
    m ← machines_candidates[0]
    allocate(f, m)
  end while

[6] https://kubernetes.io/

DoE and Evaluation Metric

Design of Experiments:

Parameters	Values
Workload Size	200, 600, 1000
Platform Size	100, 300, 500
Heterogeneity Level	3, 5, 7
Scheduling Policies	FOA, *CacheLocality*
Random Seeds	30

Total of 1620 experiments.

DoE and Evaluation Metric

Design of Experiments:

Parameters	Values
Workload Size	200, 600, 1000
Platform Size	100, 300, 500
Heterogeneity Level	3, 5, 7
Scheduling Policies	FOA, *CacheLocality*
Random Seeds	30

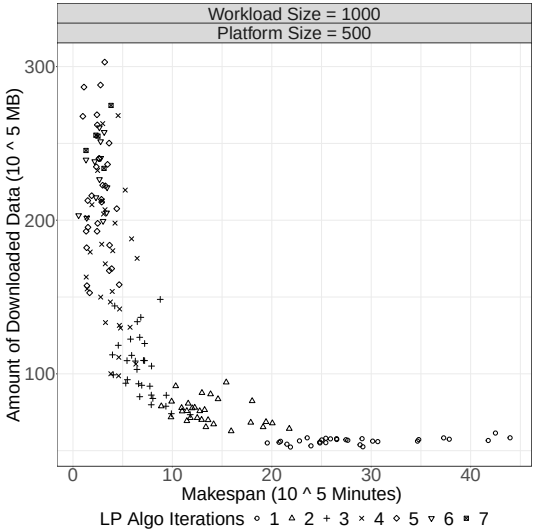
Total of 1620 experiments.

Evaluation metric: To compare FOA's results (makespan, cost and number of machines used), fi, and the baseline results, CacheLocality, CLi:

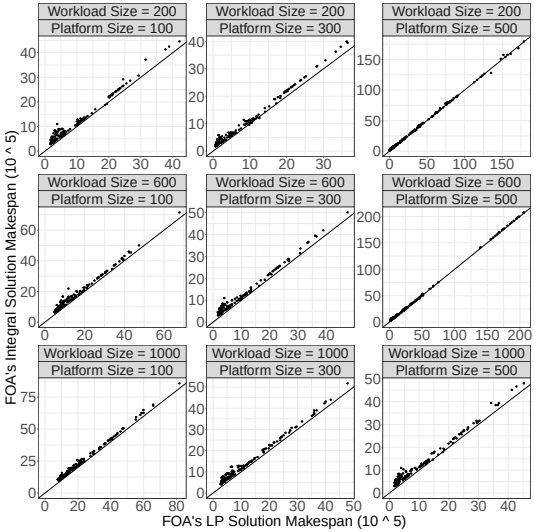
$$average\_percentage\_gain\_of\_ = \frac{\sum_{i=1}^{N} f_i}{\sum_{i=1}^{N} cl_i} \times 100 \quad (7)$$

Experimental Results

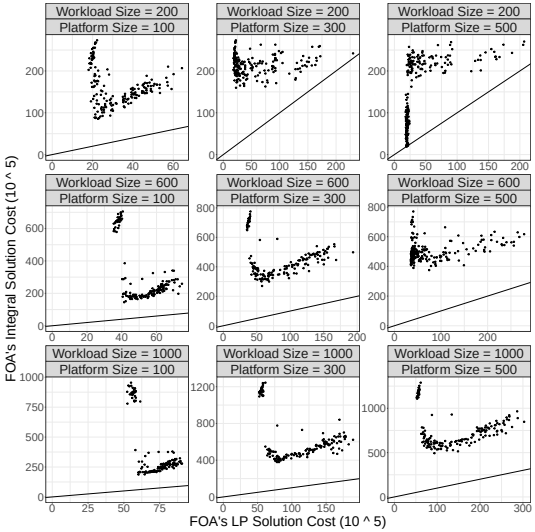
LP-FOA's Trade-off between Makespan and Cost



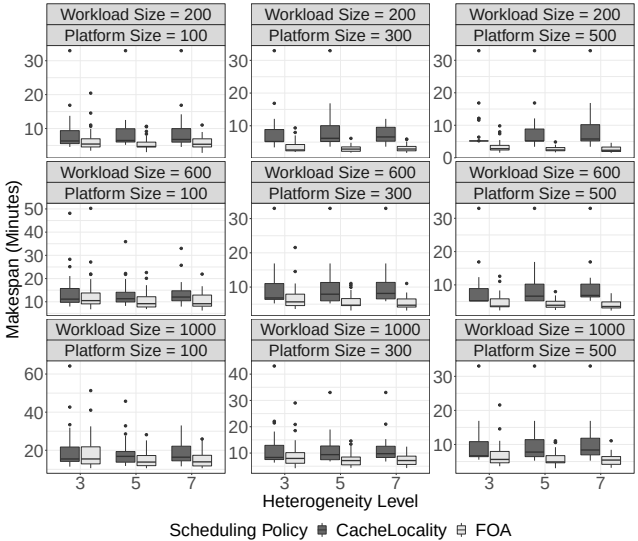
Converting FOA's Fractional Solution to Integral Solution: Makespan



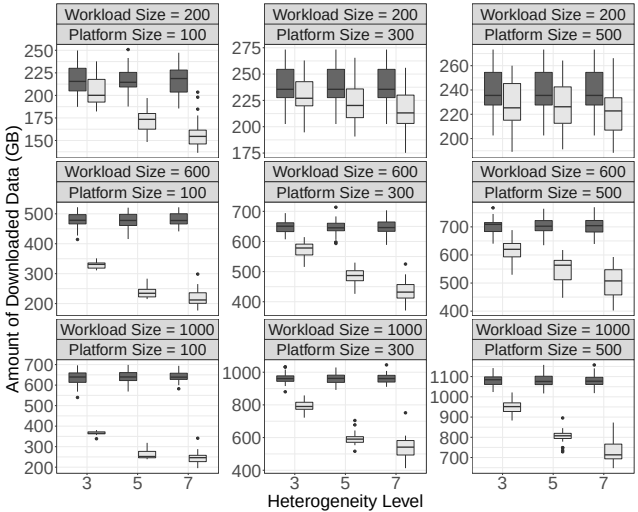
Converting FOA's Fractional Solution to Integral Solution: Cost



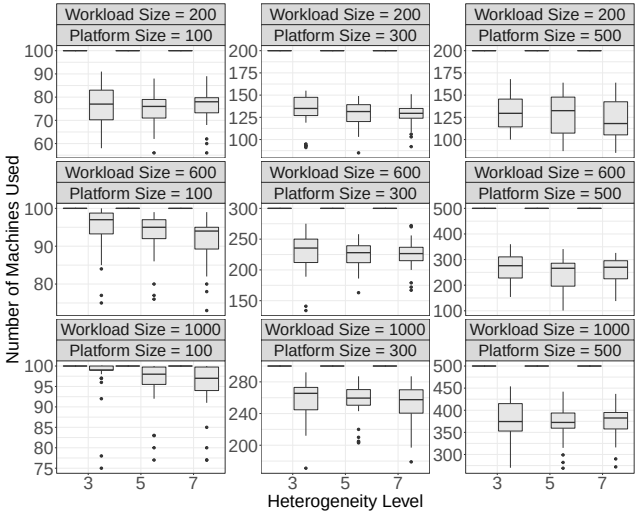
Simulated FOA x Baseline: Makespan



Simulated FOA x Baseline: Cost



Simulated FOA x Baseline: Number of Machines Used



Conclusion and Future Work

Conclusions

- Our proposed scheduling policy, FOA, achieved better makespan, cost and resources usage, outperforming our baseline (in average) as follow:
  - Makespan: from 30% to 90%,
  - Cost: from 40% to 95%.
  - Number of machines used: from 50% to 97%.

Conclusions

- Our proposed scheduling policy, FOA, achieved better makespan, cost and resources usage, outperforming our baseline (in average) as follow:
  - Makespan: from 30% to 90%,
  - Cost: from 40% to 95%.
  - Number of machines used: from 50% to 97%.
- A drawback of FOA is that it takes longer to propose a solution, with the order of minutes, while our baseline takes the order or seconds. The main reasons for that are:
  - The linear program has too many variables and constraints,
  - One main reason is that we run 8 times the linear program, but we found in the results that around 3 times may be enough,
  - We used an open-source library, python-mip, that may not be optimal.

Future Work

- We are investigating a new linear program that better models our heterogeneous platforms,
  - We would like to investigate the usage of our algorithm in more diverse scenarios, with more services and larger platforms,
- A complementary direction could be to implement FOA with other linear program resolutions,
- We believe that this algorithm can also be applied in a Global-Continuum level of Serverless Edge Computing, and we intend to follow this direction as well.

Thank you for your attention!
Any question?

A Multi-Objective Scheduling Policy for Edge Serverless Platforms

A. A. Da Silva, Y. Georgiou, M. Mercier, G. Mounié, D. Trystram

Université Grenoble Alpes, Ryax Technologies

Contact email: anderson-andrei.da-silva@inria.fr

https://andersonandrei.github.io/

https://gitlab.com/andersonandrei/scheduling-functions-on-serverless-computing

References

[1] D. Barcelona-Pons, M. Sánchez-Artigas, G. Parıs, P. Sutra, and P. Garcia López, "On the FaaS Track: Building Stateful Distributed Applications with Serverless Architectures," in Proceedings of the 20th International Middleware Conference. Davis CA USA: ACM, Dec. 2019, pp. 41–54. [Online]. Available: https://dl.acm.org/doi/10.1145/3361525.3361535

[2] I. Baldini, P. Castro, K. Chang, P. Cheng, S. Fink, V. Ishakian, N. Mitchell, V. Muthusamy, R. Rabbah, A. Slominski, and P. Suter, "Serverless Computing: Current Trends and Open Problems," arXiv:1706.03178[cs], Jun. 2017, arXiv: 1706.03178. [Online]. Available:http://arxiv.org/abs/1706.03178

[3] "Aws lambda." [Online]. Available: https://aws.amazon.com/lambda/

[4] "Cloud functions." [Online]. Available: https://cloud.google.com/functions

[5] "Azure functions." [Online]. Available: https://azure.microsoft.com/en-us/services/functions

[6] "Ibm cloud functions." [Online]. Available: https://www.ibm.com/cloud/functions

References

[7] "OpenWhisk."[Online]. Available: https://openwhisk.apache.org/

[8] Jeongchul Kim and Kyungyong Lee, 'Function Bench : A Suite of Workloads for Serverless Cloud Function Service', IEEE International Conference on Cloud Computing 2019, 07/2019

[9] Jeongchul Kim and Kyungyong Lee, 'Practical Cloud Workloads for Serverless FaaS, ACM Symposium on Cloud Computing 2019, 11/2019

[10] "Function Bench". Available: https://github.com/kmu-bigdata/serverless-faas-workbench

[11] "Grid5000". Available: https://www.grid5000.fr

[12] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, scalable, and accurate simulation of distributed applications and platforms," Journal of Parallel and Distributed Computing, vol. 74, no. 10, pp. 2899–2917, Jun. 2014. [Online]. Available: http://hal.inria.fr/hal-01017319

[13] David B. Shmoys and Éva Tardos. An approximation algorithm for the generalized assignment problem. Math. Program., 62(1–3):461–474, February 1993.