Towards Demystifying Intra-Function Parallelism in Serverless Computing
7th International Workshop on Serverless Computing (WoSC7)
In conjunction with, ACM/IFIP Middleware 2021

Michael Kiener, Mohak Chadha, Michael Gerndt
michael.kiener@tum.de, mohak.chadha@tum.de, gerndt@in.tum.de
Technische University of Munich (TUM)
Chair of Computer Architecture and Parallel Systems
Garching (near Munich), Germany
7th of November, 2021
https://www.serverlesscomputing.org/wosc7/papers/p7
Outline

1. Motivation
2. Methodology
3. Results
   3.1. CPUs to vCPUs Mapping
   3.2. Performance Results
   3.3. Cost Analysis
4. Conclusion
Motivation

- Serverless services function like black-boxes
- Developer doesn’t have full insights about underlying hardware & infrastructure
- Underutilization of computing resources increases costs significantly
Methodology

• 3 Microbenchmarks
  • Atax
  • Go fast
  • MVT

• 2 Applications
  • Heat
  • Monte Carlo

| Language | Parallelization     | Version  | Compiler & Flags       |
|----------|---------------------|----------|------------------------|
| C++      | OpenMP              | C++11    | g++ -O3                |
| Java     | ExecutorService     | Java 11  | OpenJDK 11             |
| Go       | Goroutines          | 1.16     | gc GOOS=linux           |
Results - Cores to vCPU Mapping

CPU Cores Allocation

![Bar chart showing CPU cores allocation for different memory sizes (512, 2048, 4096, 8192, 10240 MB). The chart indicates the number of CPU cores required for each memory size.](image-url)
Results - Performance

Parallel Speedups for AWS Lambda

Go fast

MVT

Heat

Monte Carlo

Michael Kiener | Towards Demystifying Intra-Function Parallelism in Serverless Computing | WoSC7
Results - Costs

Cost Comparison for AWS Lambda

| Language | Benchmark | Configuration | Maximum Cost Savings |
|----------|-----------|---------------|----------------------|
| C++      | Monte Carlo | 10240MB       | 80,1 %               |
| Java     | Monte Carlo | 10240MB       | 74,1 %               |
| Go       | Monte Carlo | 10240MB       | 81,3 %               |
Cold starts

Impact of billable time difference of cold starts

![Bar chart showing the billable cold start latency in milliseconds for different runtimes and languages. The chart compares C++, Go, and Java on AWS.](image-url)
Conclusion

• Allocated CPUs not always equal to vCPUs

• Parallelization efficiency heavily depends on service and configuration

• Maximum cost savings achieved by parallelizing
  • AWS: 81%
Thank you for your attention

Questions?