Università di Pisa

Starflow: An Hybrid HPC-Cloud convergent tool

Luca Ferrucci

Dipartimento di Informatica

Università di Pisa

Hybrid HPC architecture: why?

PROBLEMATICS

Traditional HPC infrastructures fails to scale while computational demands continue to grow, in particular for Data Stream Processing applications

Explosive growth of data generated from various sources, e.g. IoT devices!

BENEFITS

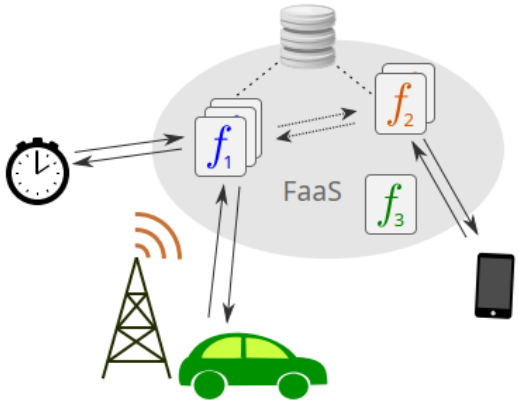
1. Scalability
2. Cost Efficiency
3. Flexibility
4. Improved Resource Utilization
5. Geographic Distribution
6. Fault tolerance
7. Data Management
8. Energy Efficiency
9. Rapid Deployment

CHALLENGES

1. Applications sensitive to latency
2. Privacy and security concerns
3. Heterogenous environment
4. Resource management
5. No performance predictability

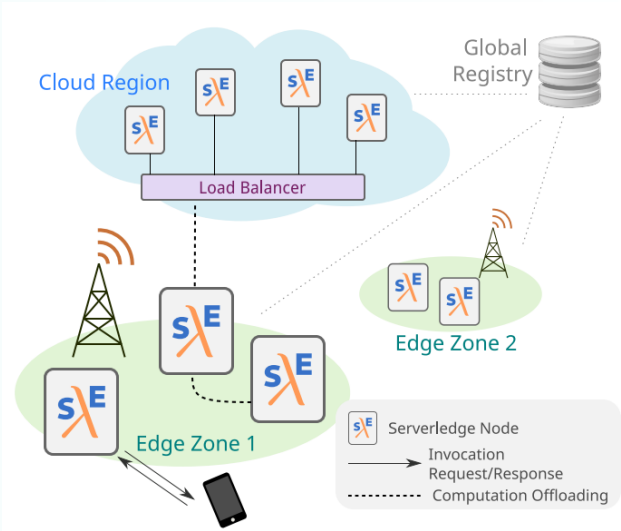
Function-as-a-service

- Application logic decomposed into a set of functions
- Serverless execution of such functions in isolated environments
  - Lightweight virtualization (containers)
- On-demand execution
  - Abstracting away the complexity of server management
  - Optimization or resource utilization
  - Focus on deploying and writing individual functions
- Event-driven execution
- Fine-grained execution model
- Seamless scalability
- Fine-grained pricing
- Now offered by major Cloud providers
  - AWS Lambdas
  - Google Cloud Functions



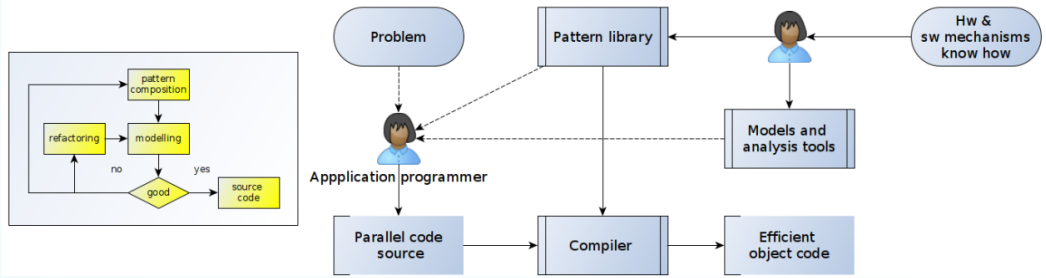
ServerLedge: a Faas framework enabling Edge computing

- Faas framework designed to support Edge environments
- Implemented in Go, it supports multiple programming language
  - Python, Js, Go
  - Native code through custom images
- Nodes organized into Cloud regions and Edge zones
  - No centralized entry point! Request sent to the closest Edge zones
- Global Registry
  - Replicated
  - Local registry acting as a cache and a DB for local only infos
- Decentralized scheduling and resource allocation
- Request offloading (horizontal and vertical)
- Live function migration



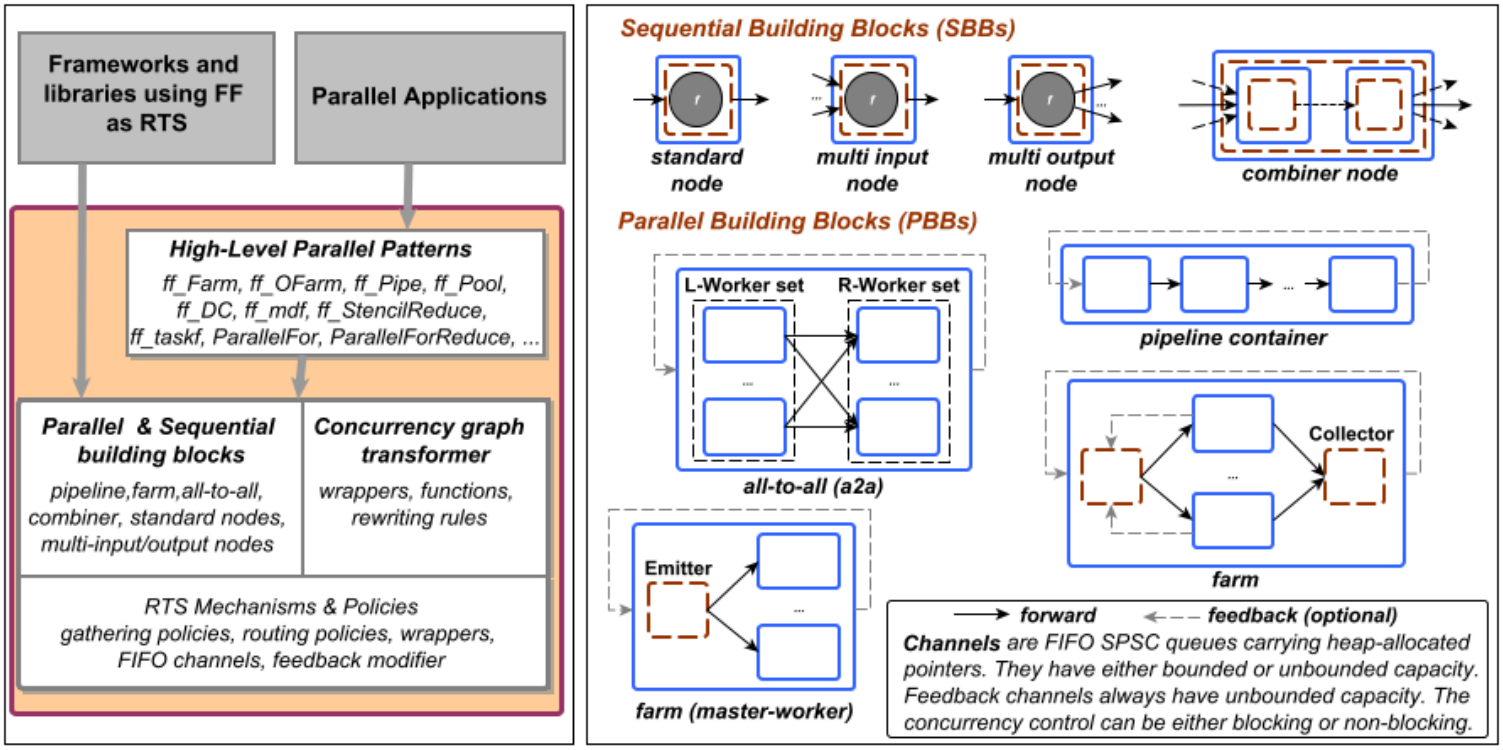
FastFlow

- Structured parallel programming framework targeting Shared-memory multi-cores, in particular for Data stream applications
  - Support also accelerators and FPGAs
  - Last edition support also offloading to Cluster of workstations (distributed environment)
- Written in C/C++, header only library, super efficient communications
- Parallel desing patterns:
  - Application programmer provide a composition of basic patterns called Building Blocks
    - Node code block
    - NodeCombiner code block
    - Pipeline parallel block
    - Farm parallel block
    - All-to-all parallel block



- Only DAG derived from the composition of basic patterns could be defined

FastFlow: high level schema and BBs



The solution: our vision

Extend the Runtime Support of FF to integrate the outsourcing of the internal business logic of a FFNode as a function on one or more Serveledge clusters

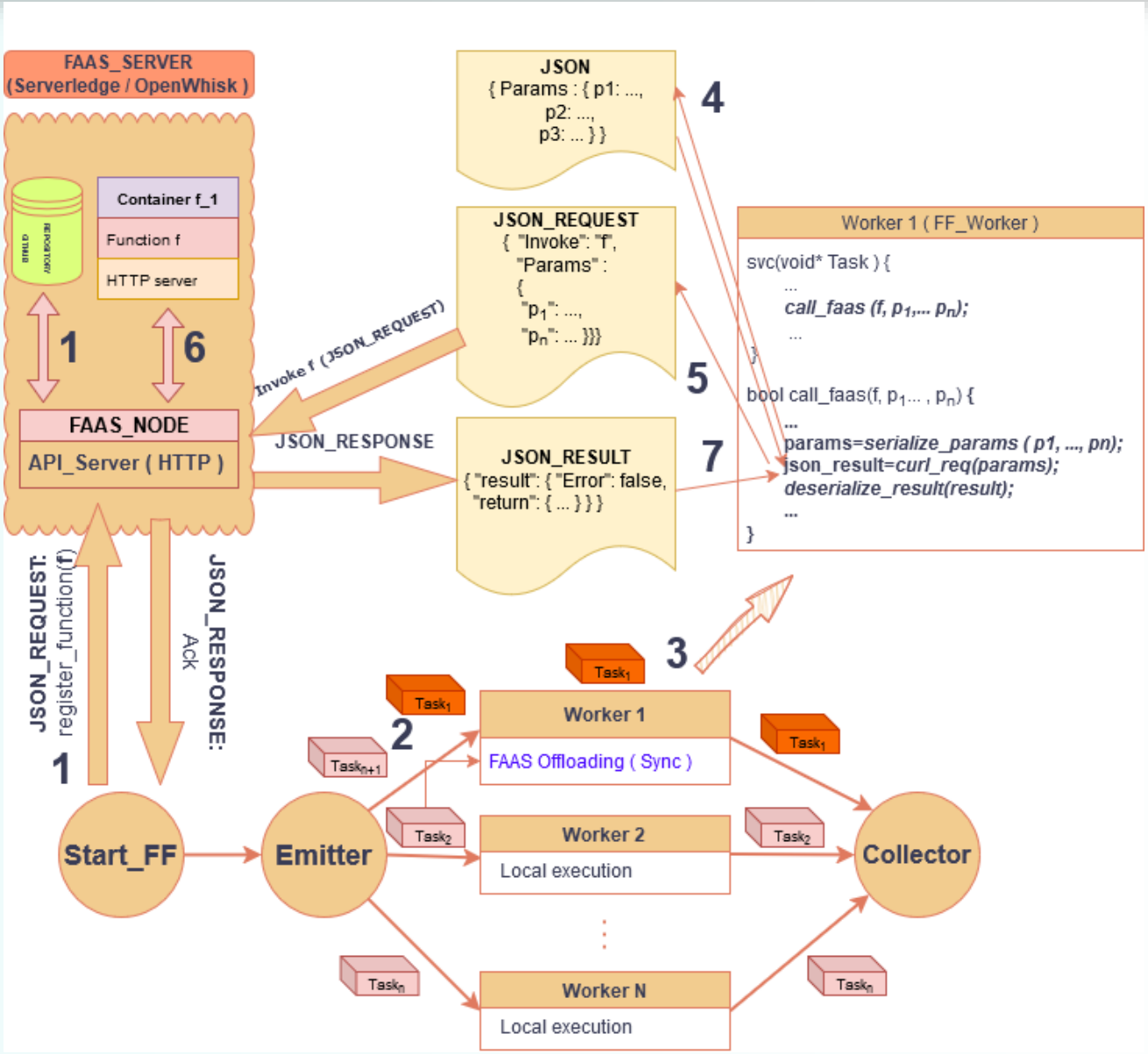
BENEFITS

- Lightweight virtualization
- Transparent model
- Cloud/Edge resource exploitation
- Higher fine-grained unit of computation
- No hand-made changes to code!

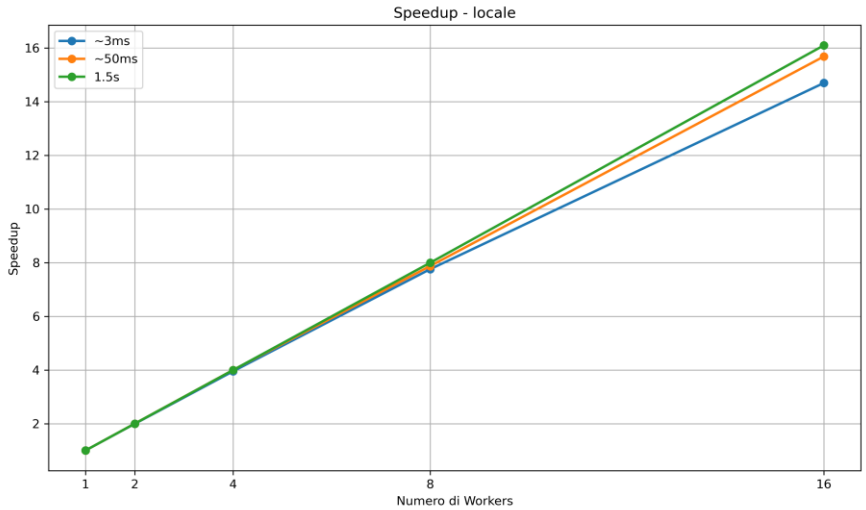
CHALLENGES

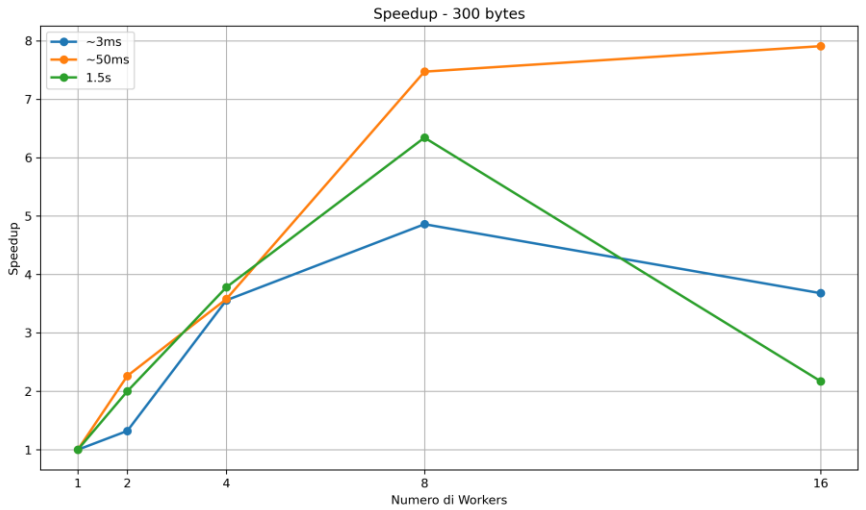
- Cold start
- Container image caching
- Support to stateful functions
- Energy awareness
- Function deployment
- Proactive reaction to peak
- Network and latency optimization

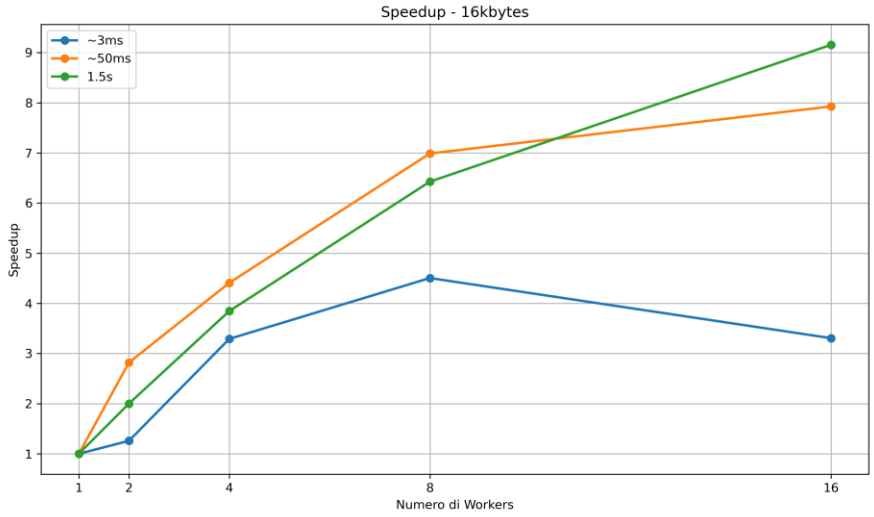
FF function offload on FAAS

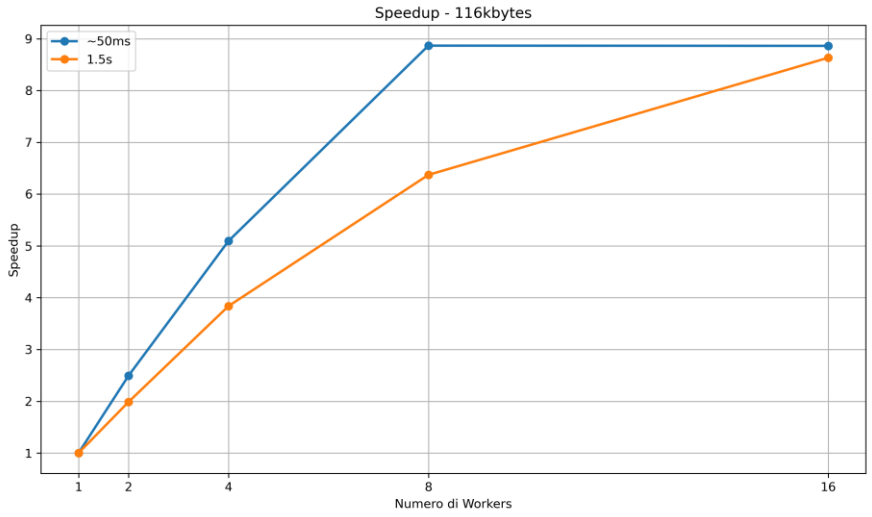


Tests: speedup on OpenWhisk

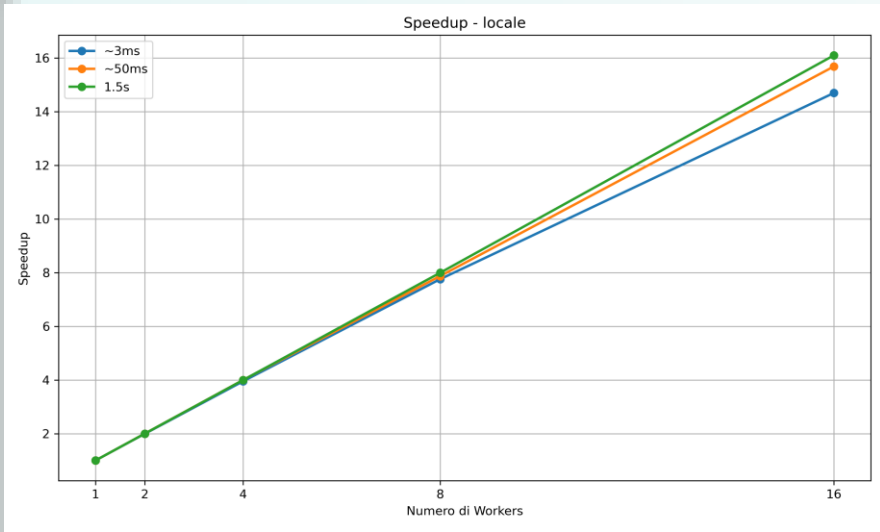


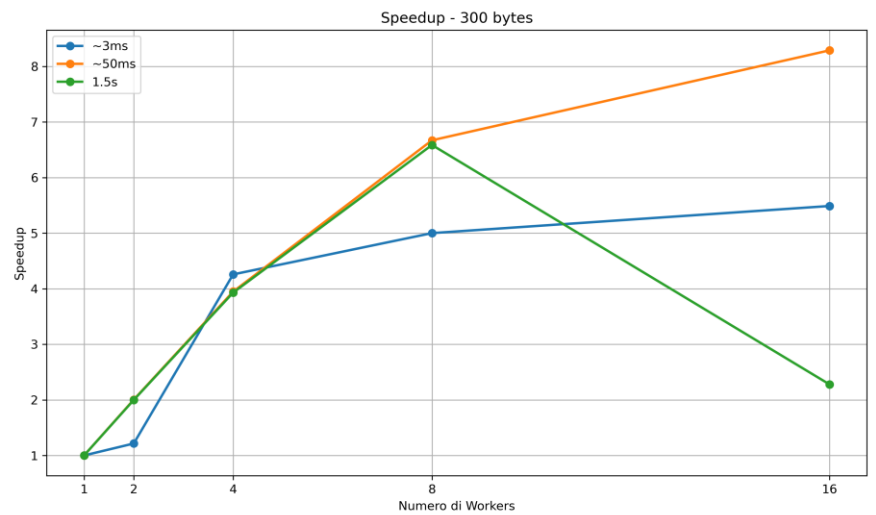


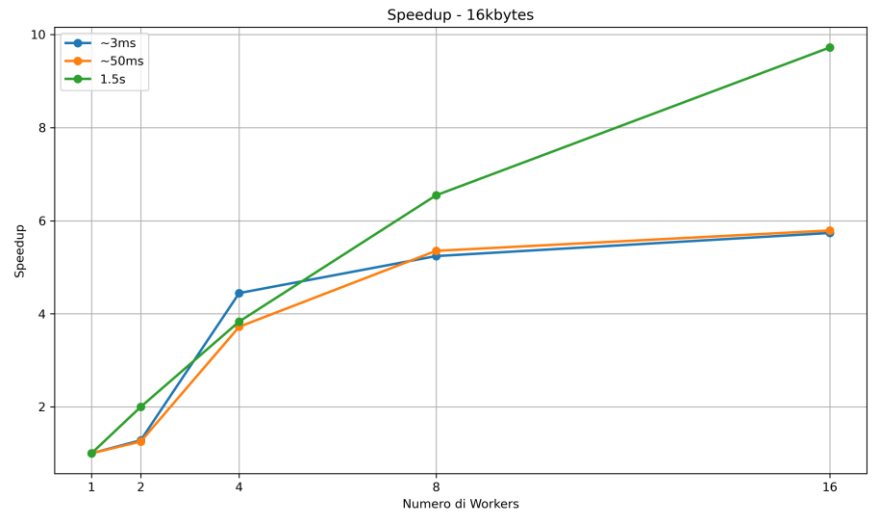


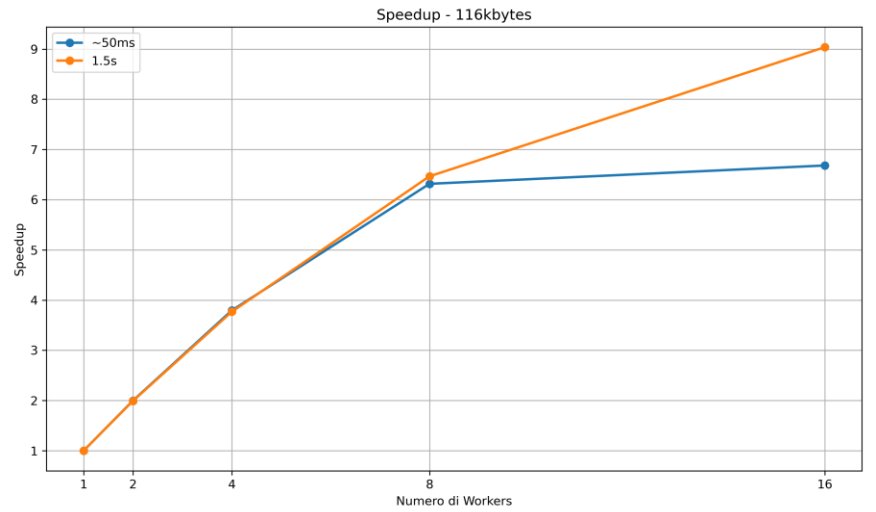


Tests: speedup on Serverledge

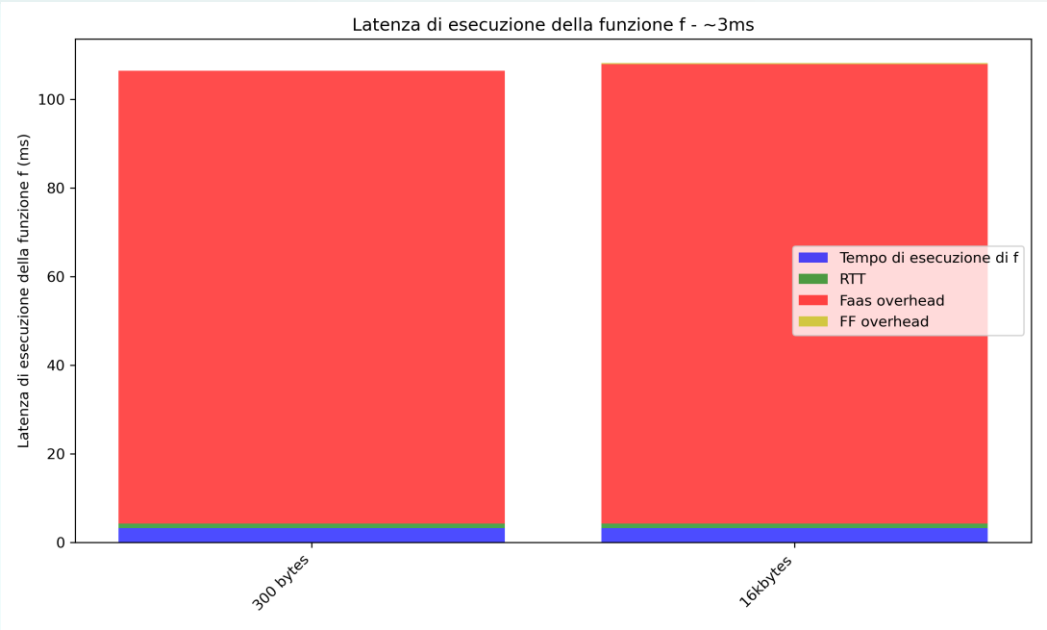


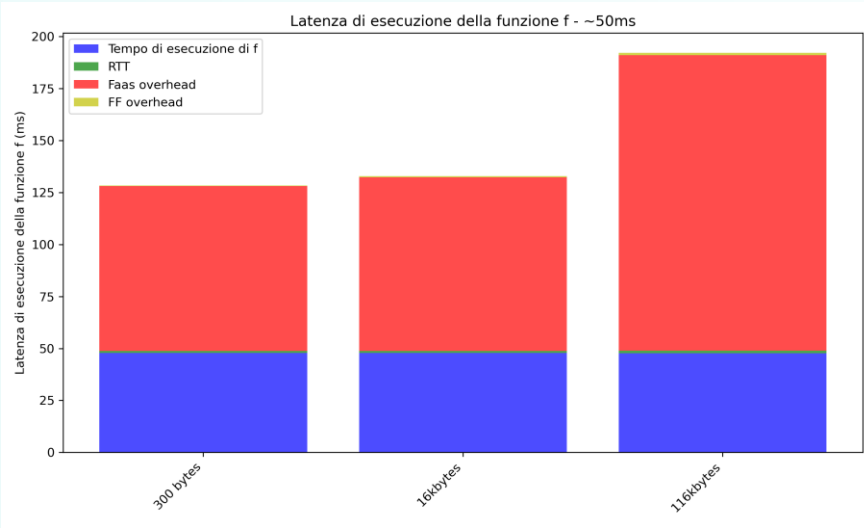


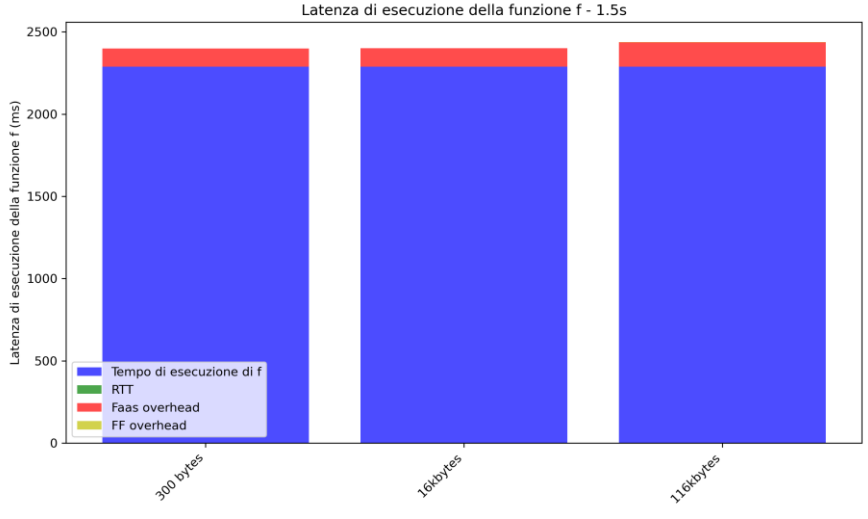




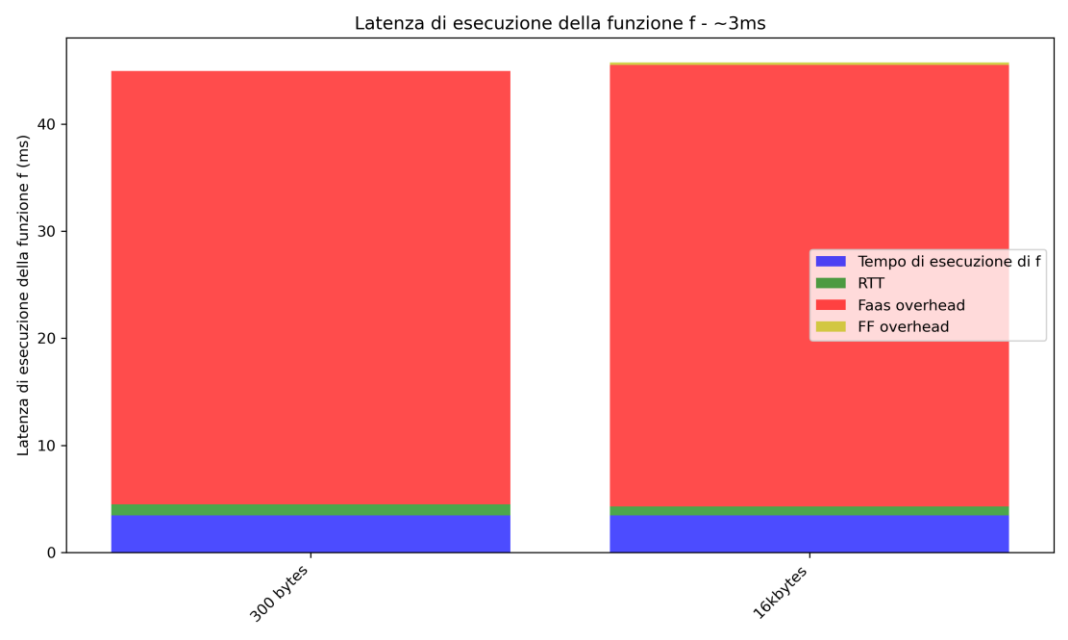
Risultati: latenza in OpenWhisk

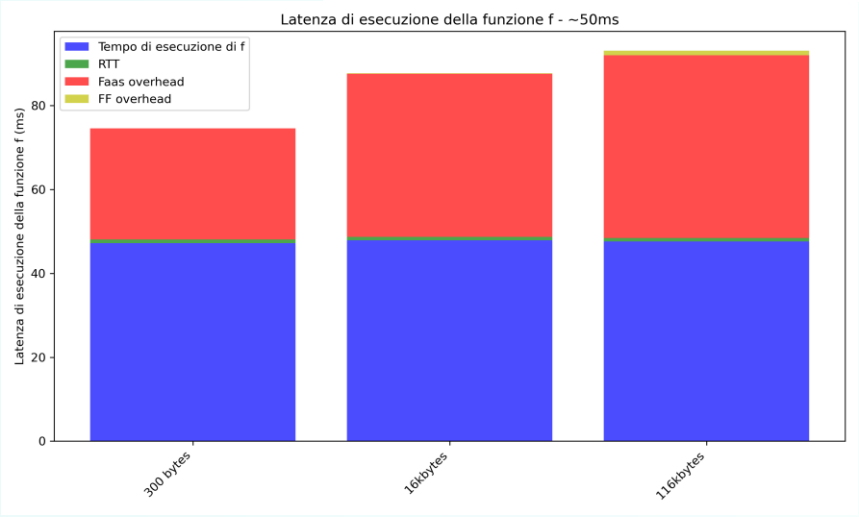


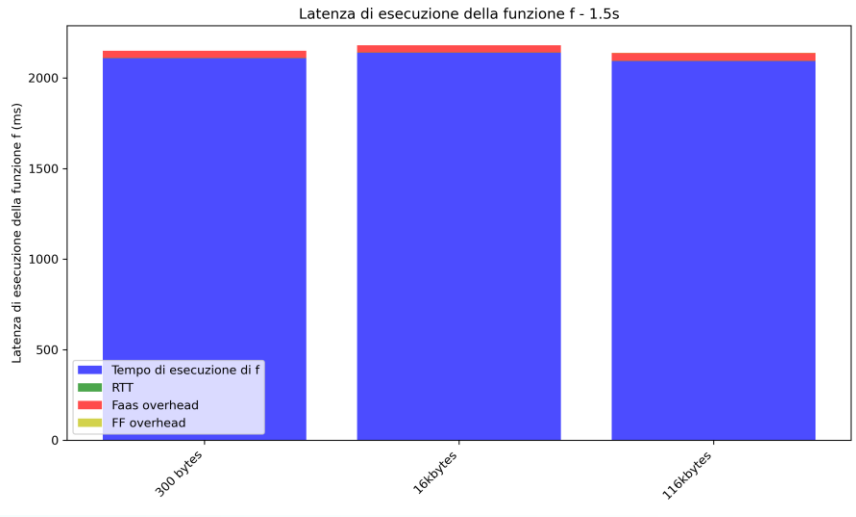




Risultati: latenza in Serverledge







Questions?