Performance Assessment of Cloud Applications

Author: Mgr. Gábor Sándor
Supervisor: prof. RNDr. Tomáš Bureš, PhD.

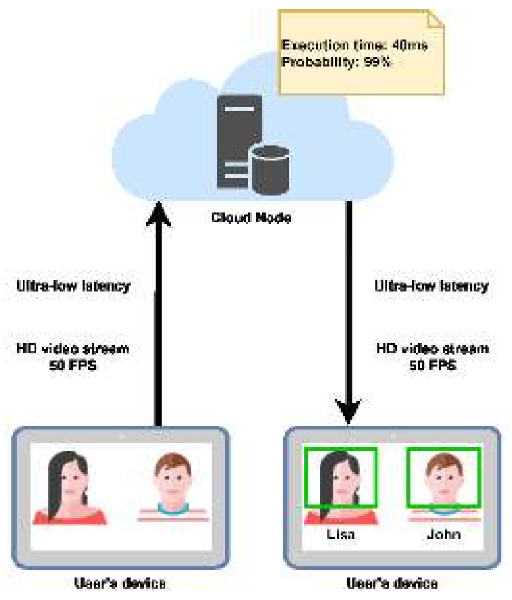
Charles University, Faculty of mathematics and physics

Motivation

Modern Cyber-physical systems and mobile applications like augmented reality or coordinated driving, etc. are envisioned to combine edge-cloud processing with real-time requirements. The real-time requirements however create a brand new challenge for cloud processing which has traditionally been best-effort. A key to guaranteeing real-time requirements is the understanding of how services sharing resources in the cloud interact on the performance level.

Objectives

The current cloud technologies cannot ensure hard real-time guarantees, therefore we targeted applications composed of multiple microservices that require soft real-time responses. In this frame, the general goal was to develop a new approach that provides soft real-time guarantees on the response time of microservices running in a container-based cloud e.g., Kubernetes, where microservices are developed in high-level programming languages e.g., Java. In this context, soft real-time guarantee means that, for example, in 99% of cases, the response time will not be more than 40 ms and in 99.9% of cases, the request demand will be less than 70 ms.



Method

Microservice profiling

- Treat microservices as black-boxes
- Assess microservice behavior in isolation & with colocated workloads
  - Analyze attributes of main system resources
  - Build model for prediction
- Design and implement **Deployment Framework**
  - Automatically execute the microservice and assess its resource usage

Predicting the response-time

- Optimal planning of colocated workloads
- Satisfy microservice resource demand
- Optimize server utilization

Characterizing the Microservice Behavior

Research for resource usage indicators

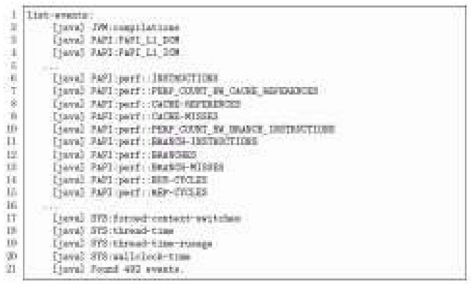
- Linux built-in features
- Third-party solutions

Observability levels

- System-wide
- Per-process

Methodologies

- Counters
- Event-based profiling



```
list-events:
        [java] JVM:compilations
        [java] PAPI:PAPI_L1_DCM
        [java] PAPI:PAPI_L1_ICM
    ...
        [java] PAPI:perf::INSTRUCTIONS
        [java] PAPI:perf::PERF_COUNT_HW_CACHE_REFERENCES
        [java] PAPI:perf::CACHE-REFERENCES
        [java] PAPI:perf::CACHE-MISSES
        [java] PAPI:perf::PERF_COUNT_HW_BRANCH_INSTRUCTIONS
        [java] PAPI:perf::BRANCH-INSTRUCTIONS
        [java] PAPI:perf::BRANCHES
        [java] PAPI:perf::BRANCH-MISSES
        [java] PAPI:perf::BUS-CYCLES
        [java] PAPI:perf::REF-CYCLES
    ...
        [java] SYS:forced-context-switches
        [java] SYS:thread-time
        [java] SYS:thread-time-rusage
        [java] SYS:wallclock-time
        [java] Found 492 events.
```

Deployment Framework

Assesses the microservice performance in predefined workload combinations. Combines the following three main components.

Orchestrator

Enables the communication between the microservice developer and the distributed components of the Deployment Framework.

Measuring Agent

Executes the microservice in an isolated environment and records the microservice resource usage pattern.

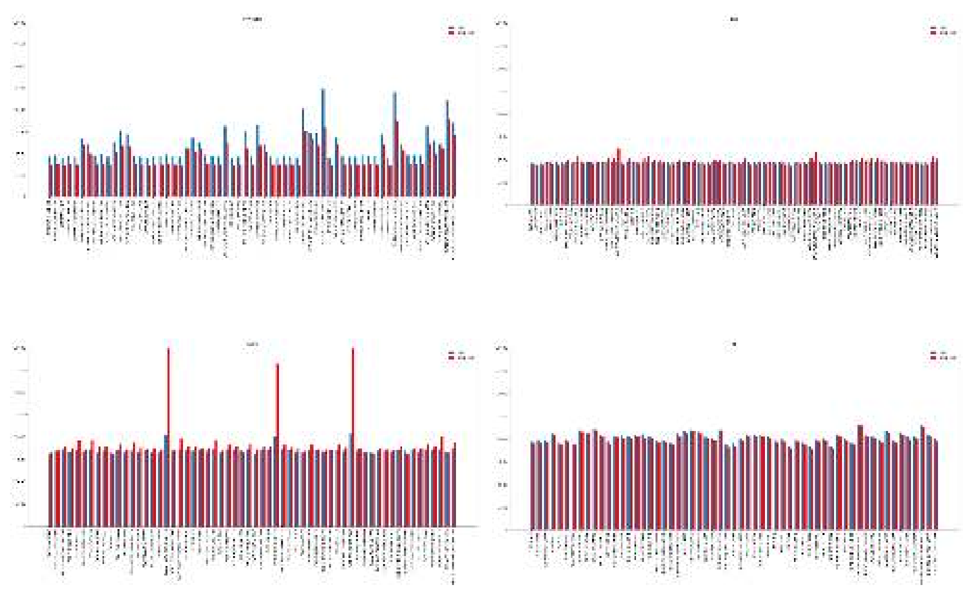
Predictor

Manages the prediction process to predict the microservice execution time using the data measured by the Measuring Agents.

Evaluation

17 artificial microservices

- Distributed into group of single, double, triple and quadruple workload combinations
- **5907** combinations executed in total



Microservice	Error rate								
	Double			Triple			Quadruple		
	Min	Max	Average	Min	Max	Average	Min	Max	Average
CYPHERD	0.069103	0.219291	0.185672	0.036139	0.288659	0.195137	0.007625	0.359415	0.197647
EGG	0.036147	0.074182	0.057391	0.000921	0.15564	0.041813	0.000237	0.276021	0.046179
FACE	0.026932	0.117689	0.075302	0.001154	1.401065	0.076166	0.001862	1.278915	0.103014
RB	0.00684	0.035495	0.019303	0.004248	0.055577	0.021804	0.003426	0.031801	0.019649
ZB	0.564034	0.88302	0.738443	0.210829	0.832983	0.535543	0.084218	0.627021	0.426497

Conclusion

In this work, we developed a new approach that provides soft real-time guarantees on the response time of microservices running in a container-based cloud. In this context, we implemented a prototype of a Deployment Framework, which measures the resource usage of the submitted microservice and uses this data to predict the microservice response time in various workload combinations.

Contrary to existing solutions, our solution uses the CPU, disk, and memory resources to characterize the microservice resource demand and to build a model for prediction.

Furthermore, we target a private cloud environment e.g., mobile network operators, which enables us to apply constraints that would not be possible to implement in public cloud service environments.