BenchFlow
A Platform for End-to-end Automation of Performance Testing and Analysis

https://github.com/benchflow

Vincenzo Ferme, Cesare Pautasso
End-to-end (e2e) Performance Testing

Workload Modeling

SUT Configuration and Deployment
End-to-end (e2e) Performance Testing

- Workload Modeling
- SUT Configuration and Deployment

Test Execution
End-to-end (e2e) Performance Testing

Workload Modeling

SUT Configuration and Deployment

Test Execution
End-to-end (e2e) Performance Testing

- Workload Modeling
- SUT Configuration and Deployment
- Test Execution
- Data Collection
End-to-end (e2e) Performance Testing

- Workload Modeling
- SUT Configuration and Deployment
- Test Execution
- Data Collection
- Data Analysis
BenchFlow Overview
BenchFlow Overview
BenchFlow Overview

- Workload Modeling
BenchFlow Overview

BenchFlow

BF

Adapters
- Workload Modeling
BenchFlow Overview

DSL

Adapters
- Workload Modeling
- SUT Configuration

BenchFlow
BenchFlow Overview

Adapters
- Workload Modeling
- SUT Configuration
- SUT Deployment
BenchFlow Overview

BenchFlow

Adapters
- Workload Modeling
- SUT Configuration
- SUT Deployment

BF

Java

Docker
BenchFlow Overview

- DSL
- Adapters
  - Workload Modeling
  - SUT Configuration
  - SUT Deployment
  - BenchFlow

BF

BenchFlow

Java

docker
BenchFlow Overview

- DSL
- Adapters
  - Workload Modeling
  - SUT Configuration
  - SUT Deployment
  - BenchFlow

BF

BenchFlow

Java

docker
BenchFlow Overview

**Adapters**
- Workload Modeling
- SUT Configuration
- SUT Deployment
- BenchFlow

**Test Execution**

**BF**

**BenchFlow**

**Faban**

**Java**

**docker**
BenchFlow Overview

**Adapters**
- Workload Modeling
- SUT Configuration
- SUT Deployment
- BenchFlow

**Test Execution**
- Faban

**Data Collection**
- Minio

**BenchFlow**

**DSL**

Java

docker
BenchFlow Overview

**DSL**
- Workload Modeling
- SUT Configuration
- SUT Deployment
- BenchFlow

Adapters

Test Execution

Data Collection

Data Analysis

BenchFlow

**BF**

Adapters
- Faban
- Minio
- Spark
- cassandra

Technologies
- Java
- docker
A Declarative DSL to Configure and Control the e2e Process
What is Possible to Configure?

- Workload Modeling
- Test Execution
- Data Collection
- Data Analysis
- SUT Configuration and Deployment

BenchFlow
What is Possible to Configure?

Workload Modeling

YAML

SUT Configuration and Deployment

BF

Test Execution

Data Collection

Data Analysis
Workload Modeling

users: 1000

workload_execution:
  ramp_up: 30s
  steady_state: 10m
  ramp_down: 30s
Workload Modeling

workload:

name_of_the_workload:

driver_type: "http"

popularity: 100%

inter_operation_timings: "uniform"

operations:

- ...

mix:

- ...

- ...
Workload Modeling

workload:
  name_of_the_workload:
  driver_type: "http"
  popularity: 100%
  inter_operation_timings: "uniform"

operations:
  - ...

mix:
  - ...

SUT-awareness
Workload Modeling

workload:
  name_of_the_workload: ...
  driver_type: "http"
  popularity: 100%
  inter_operation_timings: "uniform"

operations:
  - ...

mix:
  - ...

Know You
SUT-awareness

Faban
workload:

```
{ String : <name of the workload > }
driver_type: { String : <"start","http"> }
popularity: { [String] : <number + "%"> }
inter_operation_timings: { String: <"negative-exponential", "uniform", "fixed-time"> }
```

operations:

```
- { String : <name of the .bpmn file with the process > }

operations:

```
{ String : <Name of operation> }
protocol: { String: <"http"> }
endpoint: { String : <path of call> }
method: { String : <"OPTIONS", "GET", "HEAD", "POST", "PUT", "DELETE", "TRACE", "CONNECT"> }
headers:
    Accept: { String : <Media Type> }
    ...

data: { String : <JSON object> }
max90th_response_time: { Number : <max response time allowed for 90th percentile> }
```

mix:

```
fixed_sequence: { [String] : <name of operations in the wanted order> }
flat: { [String] : <number + "%"> }
# IF FlatSequenceMix
flat: { [String] : <number + "%"> }
sequences:
    - { [String] : <name of operations in the wanted order> }
    ...
matrix:
    - { [String] : <number + "%"> }
    ...
max_deviation: { String : <number + "%"> }
```
SUT Configuration and Deployment

services:
  service_a:
    image: 'service_a'
    environment:
      - DB_URL=jdbc:mysql://${BF_db_IP}:${BF_db_PORT}/${BF_db_MYSQL_DATABASE}
    network_mode: host
  ports:
    - 8080:8080

db:
  image: 'mysql:5.7.18'
  network_mode: host
  environment:
    - MYSQL_DATABASE="test_database"
  ports:
    - 3306:3306
SUT Configuration and Deployment

sut:
  name: example_name
  version: 1.0.0
  type: "http"

configuration:
  target_service:
    name: service_a
    endpoint: /
    sut_ready_log_check: "Started in (.*)"

deployment:
  service_a: server_1
  db: server_2
SUT Configuration and Deployment

sut:
  name: example_name
  version: 1.0.0
  type: "http"

configuration:
  target_service:
    name: service_a
    endpoint: /
    sut_ready_log_check: "Started in (.*)"

deployment:
  service_a: server_1
  db: server_2
SUT Configuration and Deployment

sut:

...  

...  

service_a:

resources:

- memory: 1GB
- cores: 1

...  

...  

environment:

...
SUT Configuration and Deployment

Docker Machine

provides

Docker Engine

Containers
SUT Configuration and Deployment

Docker Compose
SUT’s Deployment Conf.

Docker Engine
Containers

Docker Machine
provides

Docker Swarm
manages

Servers
BenchFlow and Test Execution

termination_criteria:

  test:

    max_time = 1h

  ...

  ...

  ...

  ...
Data Collection

data_collection:
  client_side:
    faban:
      ...
      interval: 1s
      ...

server_side:
  service_a: logs
  db:
    mysql:
      environment:
        SETTING_A: example_setting
      ...

Server-side Data and Metrics Collection

Test Execution

harness

Faban Drivers

Servers

Containers

Serv. N

Serv. A

Serv. B
Server-side Data and Metrics Collection

Test Execution

Faban Drivers

harness

 Servers Containers

Serv. A

Serv. B

Serv. N

MONITOR
Server-side Data and Metrics Collection

Monitors’ Characteristics:
- RESTful services
- Lightweight (written in Go)
- Can embed any existing tool

Examples of Monitors:
- CPU usage
- Database state
Server-side Data and Metrics Collection

Monitors’ Characteristics:
- RESTful services
- Lightweight (written in Go)
- Can embed any existing tool

Examples of Monitors:
- CPU usage
- Database state
Server-side Data and Metrics Collection

- Faban Drivers
- Servers
- Containers
- Serv. A
- Serv. N
- DBMS

Test Execution
Server-side Data and Metrics Collection
Server-side Data and Metrics Collection

Collectors’ Characteristics:
- RESTful services
- Lightweight (written in Go)
- Two types: online and offline
- Can embed any existing tool
- Buffer data locally

Examples of Collectors:
- Container's Stats (e.g., CPU usage)
- Database dump
- Applications Logs
Server-side Data and Metrics Collection

Collectors’ Characteristics:
- RESTful services
- Lightweight (written in Go)
- Two types: online and offline
- Can embed any existing tool
- Buffer data locally

Examples of Collectors:
- Container’s Stats (e.g., CPU usage)
- Database dump
- Applications Logs
Data Analysis

1. Analysis to Perform
2. Metrics to Compute
3. Settings for the Analysis Logic
Data Analysis in BenchFlow

Test Execution

Analyses

harness

DBMS

Serv. A

Stats Collector

Serv. N

DB Collector

kafka

Spark

Minio

Instance Database
Data Analysis in BenchFlow
Data Analysis in BenchFlow

Test Execution

Analyses

harness

Serv. A

Serv. N

Stats Collector

DB Collector

DBMS

kafka

Spark

Minio

Instance Database

COLLECT

Publish
Data Analysis in BenchFlow

- Test Execution
- Servers
- Containers
- Serv. A
- Stats Collector
- DBMS
- Serv. N
- DB Collector
- kafka
- Spark
- Minio
- Instance Database
- Faban Drivers

Subscribe
COLLECT
Data Analysis in BenchFlow

Test Execution

harness

Faban Drivers

Servers

Containers

Serv. N

Stats Collector

Serv. A

DB Collector

DBMS

Analyses

kafka

Subscribe

Spark

Read

Minio

Instance Database

COLECT

Publish
Data Analysis: Metrics

1. Client-side Metrics from Faban:
   - Response Time, Throughput, Errors, ...
Data Analysis: Metrics

1. Client-side Metrics from Faban:
   - Response Time, Throughput, Errors, ...

2. Resource Utilisation Metrics:
   - RAM, CPU, IO, NETWORK Utilisation
Data Analysis: Metrics

1. Client-side Metrics from Faban:
   - Response Time, Throughput, Errors, ...

2. Resource Utilisation Metrics:
   - RAM, CPU, IO, NETWORK Utilisation

3. SUT-aware Metrics:
   - Execution Time of Process Instances, ...
Data Analysis: Metrics

1. Client-side Metrics from Faban:
   - Response Time, Throughput, Errors, ...

2. Resource Utilisation Metrics:
   - RAM, CPU, IO, NETWORK Utilisation

3. SUT-aware Metrics:
   - Execution Time of Process Instances, ...

... AND Statistics on Them:
   - Descriptive Statistics (max, min, avg, ci, ...), Little’s Law, Covariance, Coefficient of Variation, ...
BenchFlow Today

Test Execution

Data Collection

Data Analysis
BenchFlow in DevOps

Test Execution

Data Collection

Data Analysis

BenchFlow
BenchFlow in DevOps

Test Execution
Data Collection
Data Analysis

Repo

BenchFlow

[api]
BenchFlow in DevOps

Repo

Test Execution

Data Collection

Data Analysis

BenchFlow
BenchFlow in DevOps

- CI Server
- Test Execution
- Data Collection
- Data Analysis

BenchFlow

Repo
BenchFlow in DevOps
BenchFlow in DevOps

BenchFlow

Test Execution

Data Collection

Data Analysis
BenchFlow in DevOps

BenchFlow

Test Execution

Data Collection

Data Analysis

Continuous Deployment
BenchFlow in DevOps

BenchFlow

Test Execution

Data Collection

Data Analysis

Continuous Deployment

Performance Regressions

Versions

Metric
BenchFlow in DevOps

BenchFlow

Test Execution

Data Collection

Data Analysis

Continuous Deployment

Capacity Planning

Performance Regressions

Metric

Versions
BenchFlow in DevOps

- Test Execution
- Data Collection
- Data Analysis
- Continuous Deployment
- Capacity Planning
- Performance Regressions
- Versioning

BenchFlow
Objective-Driven Performance Testing
Objectives Taxonomy

Base Objectives (Test Types)
standard performance tests, e.g., load test, stress test, spike test, and configuration test
Objectives Taxonomy

Base Objectives (Test Types)
standard performance tests, e.g., load test, stress test, spike test, and configuration test

Objectives
specific types of performance engineering activities, e.g., capacity planning and performance tradeoffs
Objectives Taxonomy

Base Objectives (Test Types)
standard performance tests, e.g., load test, stress test, spike test, and configuration test

Objectives
specific types of performance engineering activities, e.g., capacity planning and performance tradeoffs

Meta-Objectives
defined from already collected performance knowledge, e.g., regressions detection, comparing different systems (or versions of the same system) using a benchmark
Example: Configuration Test

**objective:**

**type:** configuration

**observation:**

- ...

**exploration_space:**

- ...

**termination_criterion:**

- ...
Example: Configuration Test

observation:

service_A:
- ram_avg
- cpu_avg
- response_time_90th_p

service_B:
- ram_avg
Example: Configuration Test

exploration_space:

service_A:

resources:

- memory:
  range: 1GB... 5GB
  step: +1GB

- cpus:
  range: 1...4

environment:

- SIZE_OF_THREADPOOL:
  range: 5...100
  step: +5

- ...

service_B

service_A
Example: Configuration Test

termination_criteria:

exploration:
  ...

experiment:
  number: 5
  confidence: 90%

  test:
    max_time = 1h

  ...

BF
Example: Configuration Test
Example: Configuration Test
Example: Configuration Test
Example: Configuration Test

MARS
Kriging
...

observation:
  service_A:
    - ram_avg
    - cpu_avg
    - response_time_90th_p
  service_B:
    - ram_avg
Objectives

- Capacity planning (also based on some constraints)
  e.g., CPU, RAM
  why? cost of resources -> important for the business
Objectives

- Capacity planning (also based on some constraints)
e.g., CPU, RAM
why? cost of resources -> important for the business

- Performance tradeoffs based on some (resource) constraints
e.g., which configuration is good enough?
why? responsiveness -> important for the user
Example: Performance Tradeoff

tradeoff_target:

service_A:
  - \text{min}(\text{response\_time\_90th\_p})

service_B:
  - \text{min}(\text{memory})

...
Example: Performance Tradeoffs

\text{tradeoff_target:}

\text{service}_A: \quad \text{min}(\text{response}\_\text{time}\_90\text{th}\_p)

\text{service}_B: \quad \text{min}(\text{memory})

\text{MARS}

\text{Kriging}

\ldots
Example: Performance Tradeoffs

tradeoff_target:
  service_A:
    - min(response_time_90th_p)
  service_B:
    - min(memory)

MARS
Kriging
...
Example: Performance Tradeoffs

tradeoff_target:

- service_A:
  - min(response_time_90th_p)

- service_B:
  - min(memory)

MARS

Kriging

...
How Can I Extend BenchFlow?

1. New SUT Types:
   - Add Adapters to Interact with the SUT (Java)
How Can I Extend BenchFlow?

1. New SUT Types:
   - Add Adapters to Interact with the SUT (Java)

2. Support new Objectives:
   - Extend the DSL (Scala) and BenchFlow (Java)
1. New SUT Types:
   - Add Adapters to Interact with the SUT (Java)

2. Support new Objectives:
   - Extend the DSL (Scala) and BenchFlow (Java)

3. New Data Collectors:
   - Add Collectors Services (docker)
1. New SUT Types:
   - Add Adapters to Interact with the SUT (Java)

2. Support new Objectives:
   - Extend the DSL (Scala) and BenchFlow (Java)

3. New Data Collectors:
   - Add Collectors Services (docker)

4. New Data Analyses:
   - From Dedicated Services
   - From Collectors, or with Spark Tasks
BenchFlow Adoption: Last Two Years

5 Total Developers (2-3 Active)
BenchFlow Adoption: Last Two Years

5 Total Developers (2-3 Active)

3 Adopters

Universität Stuttgart

Facoltà di scienze informatiche

Università della Svizzera italiana
BenchFlow Adoption: Last Two Years

5 Total Developers (2-3 Active)

3 Adopters

200+ Unique Tests DSL (3 SUTs / 12 Vers.)
BenchFlow Adoption: Last Two Years

5 Total Developers (2-3 Active)

3 Adopters

200+ Unique Tests DSL (3 SUTs / 12 Vers.)

600+ Executed Test Runs
BenchFlow Adoption: Last Two Years

5 Total Developers  (2-3 Active)

3 Adopters

200+ Unique Tests DSL  (3 SUTs / 12 Vers.)

600+ Executed Test Runs

450GB+ Crunched Data
BenchFlow Adoption: Last Two Years

5 Total Developers (2-3 Active)

3 Adopters

Universität Stuttgart

200+ Unique Tests DSL (3 SUTs / 12 Vers.)

600+ Executed Test Runs

450GB+ Crunched Data

20’000+ Computed Metrics and Statistics
Where is BenchFlow?

Platform
Where is BenchFlow?

Platform

🔗 https://github.com/benchflow
Where is BenchFlow?

Platform

🔗 https://github.com/benchflow

🚀 https://hub.docker.com/u/benchflow/
Where is BenchFlow?

Platform

🔗 https://github.com/benchflow

🚀 https://hub.docker.com/u/benchflow/

When a Release?  September 2017
Where is BenchFlow?

Platform

- https://github.com/benchflow
- https://hub.docker.com/u/benchflow/

When a Release? September 2017

Research Publications

http://benchflow.inf.usi.ch
Call for Internship

1. Academic / Industry or Hybrid
Call for Internship

1. Academic / Industry or Hybrid

2. Use Cases for BenchFlow
Call for Internship

1. Academic / Industry or Hybrid

2. Use Cases for BenchFlow

3. Open for Extending BenchFlow for the Use Cases
Call for Project Collaboration

1. Performance Testing in DevOps
Call for Project Collaboration

1. Performance Testing in DevOps

2. End-to-end Automation of Performance Testing
Call for Project Collaboration

1. Performance Testing in DevOps
2. End-to-end Automation of Performance Testing
3. Objective-driven Performance Testing
Call for Project Collaboration

1. Performance Testing in DevOps
2. End-to-end Automation of Performance Testing
3. Objective-driven Performance Testing
4. …Other Ideas?
Thank You!

benchflow

https://github.com/benchflow

vincenzo.ferme@usi.ch