Performance Dashboard for Continuous Benchmarking of HPC Libraries

Chingun Ariunbat, Jamil Bagga, Walter Alexander Böttcher, Darius Schefer, Maximilian Schik

2021-05-13

Contents

1 Introduction

Performance Dashboard for Continuous Benchmarking of HPC Libraries

PSE SS21

2 Goals

2.1 Required

Criteria-Template **C1**
Implemented By: FR1

template

2.2 Optional

Optional-Criteria-Template no according requirement **OC1**

template

2.3 Limitation

Non-Criteria-Template **NC1**

template

3 Usage

template

4 Product Environment

template

5 Functional Requirements

functionality template with “smthn in quotes” NOT TESTED **FR1**
Implements: C1

template

6 Nonfunctional Requirements

non-functionality template **NF1**

template

7 Product Data

Benchmark Results (Name in progress) **PD1**

Format: JSON/CSV
Description:

- saved on server
- algorithm result data (time, storage, accuracy, convergence(?))

Git Histories **PD2**

Format: ??? (WIP)

Templates **PD3**

Format: JSON (?)

8 User Interface

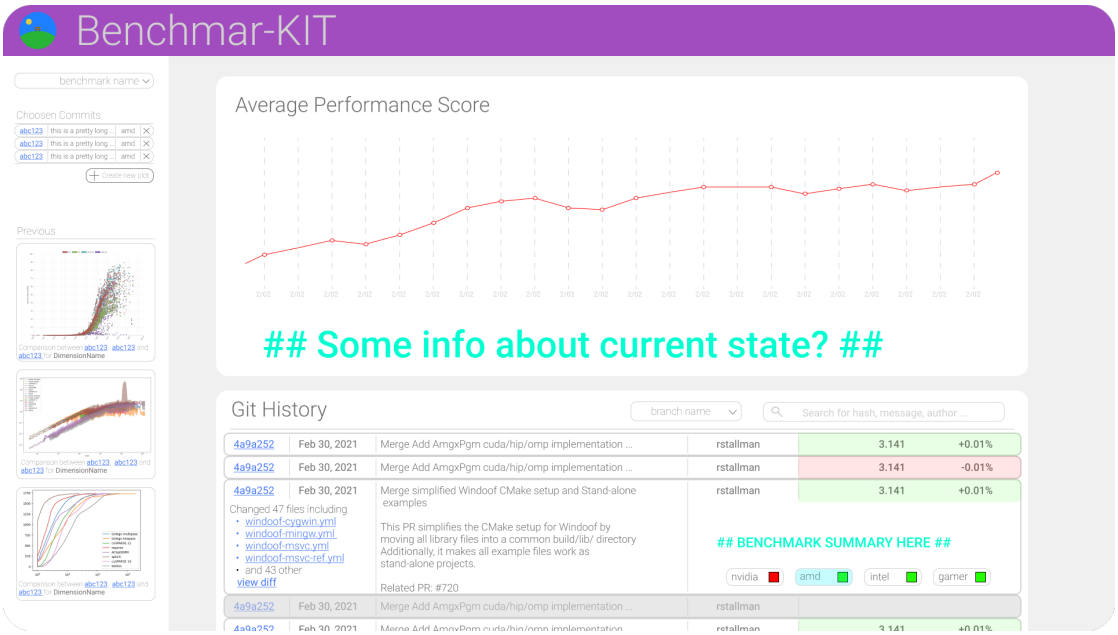

Figure 1: Main page

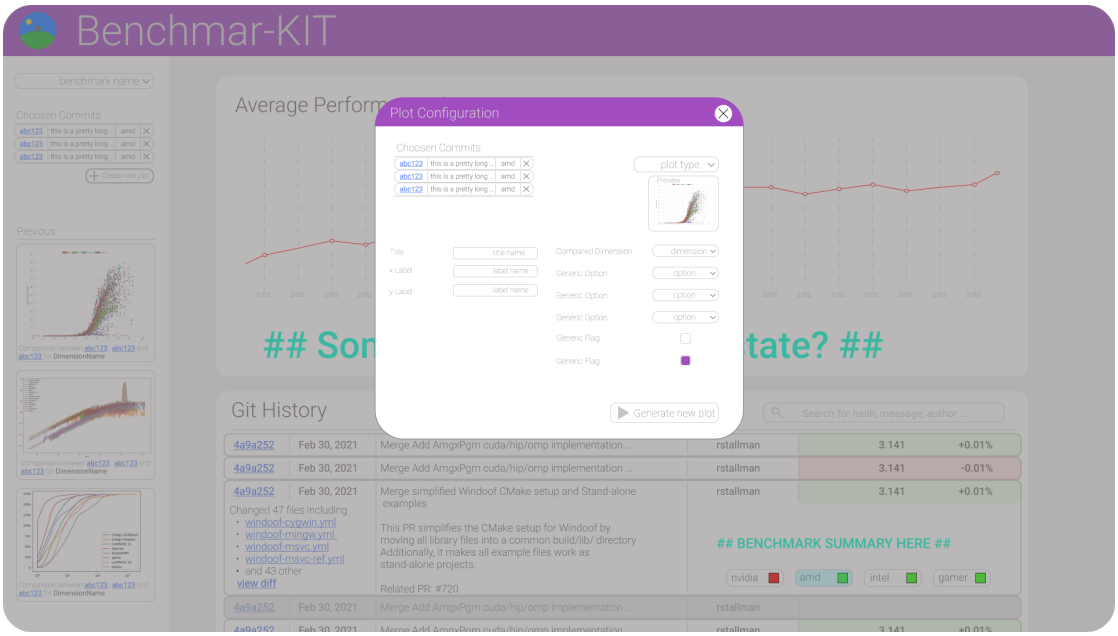

Figure 2: Configuration Popup

9 Tests

Test Template Name **T1**

Tests:

something that should be tested

10 Scenarios

SC1 Scenario name: Inspecting Last Change
Participating actor: Ted: User, CI: Benchmarking system
Flow of events: Ted works on a project that has *PROJECT NAME* set up. He makes changes on a performance critical component. After that he pushes his changes to the repository. The CI sends its benchmark results to the system, which stores it in a persistent way. Ted opens the webapp and selects a benchmark. He sees a list of all recent changes. The changes without benchmark data are greyed out. Ted selects his newest change. He selects a device to take the benchmark data from. The change appears in a list of selected changes. Ted selects the "Create New Plot" option. A popup appears. Ted chooses the dimension he wants to inspect. After he configured his plot he decides to save the template for later use. He selects the "Save Template" option. He enters a name for the template and selects the "Save" option. After that he selects the "Generate Plot" option. Ted gets redirected to a new site where he can inspect his plot. He decides to send this plot to a coworker. He selects the "Share" option and a link gets displayed. He copies the link and sends it to his coworker.

SC2 Scenario name: Comparing Benchmarks
Participating actor: Greta: User
Flow of events: Greta opens the webapp. She selects a benchmark. She selects two benchmarks by first picking a specific change and then a specific device. She opens the configuration popup by selecting the "Create New Plot" option. She wants to use a previously created template. She selects the "Use Template" option and chooses her template from a list of available ones. The settings specified in the template get applied to the current configuration. Greta makes final adjustments and then selects the "Generate Plot" option. After that she gets redirected to a new site where she can inspect her plot. Ted also wants to download the plot for use in his publication. He selects the "Export" option. He gets to choose between a seleciton of filetypes. He picks his preferred one. A link gets displayed leading to a download with the selected filetype.

SC3 Scenario name: Performance Tracking
Participating actor: CI: Benchmarking system
Flow of events: The CI runs a specific benchmark for a specific change on a specific device. It sends a POST request to the system containing the results and the benchmark type, change identification and device name. The system receives the results and stores them in a persistent database. It recognizes that the benchmark performance has dropped by a significant factor. The system triggers a hook that sends a message to the developer slack channel informing about the performance drop. It also publishes a comment under the change on GitHub.

11 Use Cases

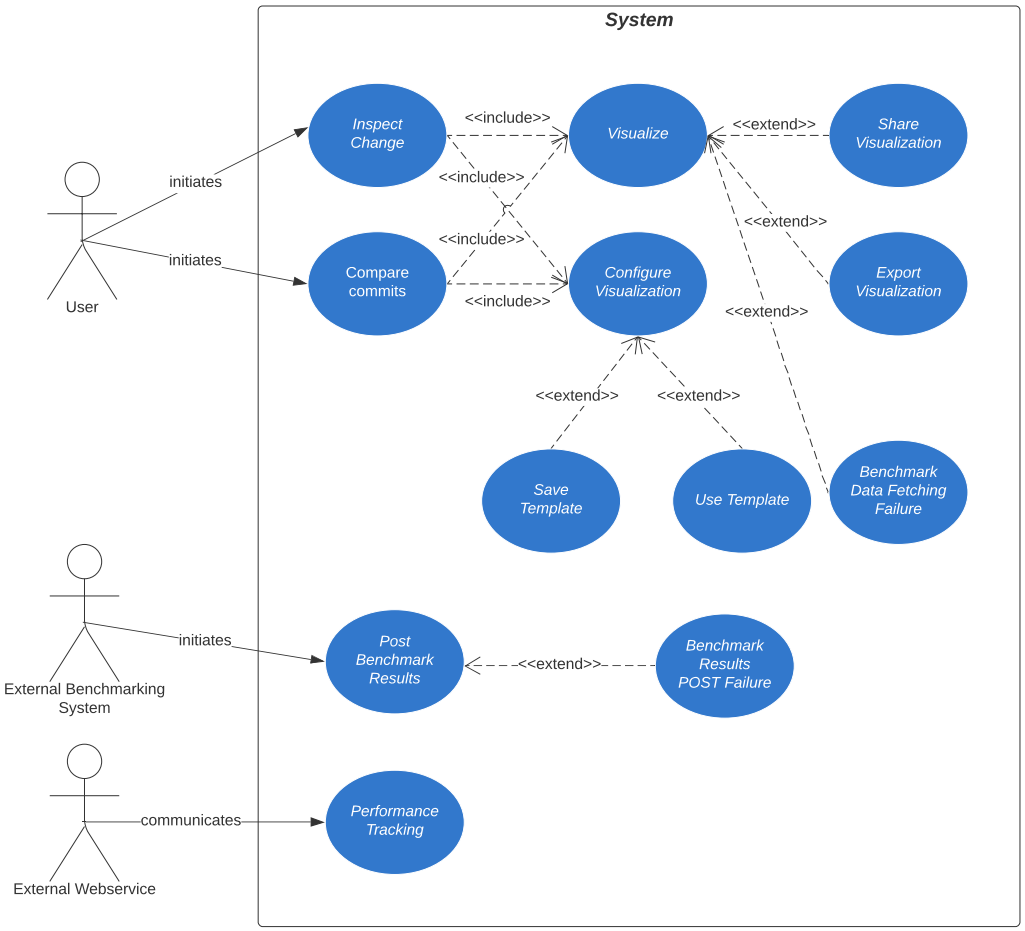

Figure 3: Use case diagram

U1 Use case name: Visualize
Participating actors: initiated by User
Entry conditions: configuration is available
Flow of events:

1. The web app sends a request to the backend containing the configuration.
2. The backend fetches the specified data from a databank.
3. The backend does the calculations specified in the configuration (mean, median, standard deviation).
4. The backend sends the data back to the webapp.
5. The webapp takes the data and generates the plot specified in the configuration.
6. The user gets redirected to a new site where he can inspect the generated plot.

Exit conditions: The plot specified by the configuration gets shown to the User.
Quality requirements: Shouldn't take more than 10 seconds

U2 Use case name: Configure Visualization
Participating actors: initated by User
Entry conditions: User selected the "Create New Plot" option
Flow of events:

1. A popup appears.
2. The user chooses a plot type.
3. The user chooses between certain options that are specific to the plot type.

Exit conditions: A configuration gets created.
Quality requirements: The available options should be understandable without any previous knowledge or otherwise described by a short text.

U3 Use case name: Inspect Change
Participating actors: initated by User
Entry conditions: benchmark data for commit is available
Flow of events:

1. The user selects a single commit.
2. The user initiates the `Configure Visualization` use case by selecting the “Create New Plot” option.
3. Once the user is satisfied with his configuration, he initiates the `Visualize` use case by selecting the “Create New Plot” option in the popup.

Exit conditions: Visualization is displayed to the user
Quality requirements: ???

U4 Use case name: Compare commits
Participating actors: intiated by User
Entry conditions: benchmark results for all selected commits are available
Flow of events:

1. The user selects multiple commits.
2. The user initiates the `Configure Visualization` use case by selecting the “Create New Plot” option.
3. Once the user is satisfied with his configuration, he initiates the `Visualize` use case by selecting the "Generate New Plot" option in the popup.

Exit conditions: Visualization is displayed to the user
Quality requirements: ???

U5 Use case name: Share Visualization
Participating actors: initiated by User
Entry conditions: A visualization has been generated
Flow of events:

1. The user selects the “Share Visualization” option.
2. A link gets displayed.
3. The link redirects any visitors to the same visualization.

Exit conditions: A link is shown which redirects to the visualization
Quality requirements: ???

U6 Use case name: Export Visualization
Participating actors: initiated by User
Entry conditions: A visualization has been generated
Flow of events:

1. The user selects the “Export Visualization” option.
2. A popup appears.
3. The user chooses a filetype for the export.
4. The user confirmes and downloads the visualization in the choosen file format.

Exit conditions: The User is offered a download of an export of the visualization
Quality requirements: Support for the filetypes png, pdf and pgf (what is the preferred latex format?)

U7 Use case name: Save Template
Participating actors: initiated by User, (maybe web browser as well? the cookies get stored on the web browser)
Entry conditions: The user is in the `Configure Visualization` use case
Flow of events: The `Save Template` use case extends the `Configure Visualization` use case.

1. The User selects the “save template” option.
2. The User enters a name for the template.
3. The webapp stores the template locally (cookies).

Exit conditions: Template is stored on the locally (might add global templates for later?)
Quality requirements: Requires less than 1kB of memory

U8 Use case name: Use Template
Participating actors: initiated by User (maybe web browser as well? the cookies get stored on the web browser)
Entry conditions: The user is in the `Configure Visualization` use case and a template is available locally
Flow of events: The `Use Template` use case extends the `Configure Visualization` use case.

1. The User selects the "use template" option.
2. User is shown a list of all available templates.
3. User selects a template from the list.
4. The current configuration options get set to the values specified in the template.

Exit conditions: The template is applied to the current configuration.
Quality requirements: ???

U9 Use case name: Post Benchmark Results
Participating actors: initiated by External Benchmarking System
Entry conditions: The external benchmarking system ran the benchmarks
Flow of events:

1. The benchmarking system makes a POST request to the backend containing the new benchmark data in JSON format.
2. The backend converts the received data into the correct format.
3. The backend stores the received data in a database.

Exit conditions: The received performance data is stored in a database
Quality requirements: ???

U10 Use case name: Performance Tracking
Participating actors: communicates with External Webservice
Entry conditions: New benchmark data has been posted to the backend
Flow of events:

1. The backend evaluates the performance of the new benchmark data.
2. The backend compares the performance of the new benchmark with the performance of the corresponding benchmark of the last commit.
3. The backend relays the results to a configured number of hooks.
4. The hooks contact their external webservices according to how they have been configured.

Exit conditions: The server fires a POST Request to all webhook subscribers
Quality requirements:

U11 Use case name: Benchmark Results POST Failure
Participating actors: return error code to External Benchmarking System
Entry conditions:
Flow of events: This use case extends the `Post Benchmark Results` use case if an error occurs.

1. The backend identifies the error.
2. The backend creates a response with the correct error code.

Exit conditions:
Quality requirements:

U12 Use case name: Benchmark Data Fetching Failure
Participating actors: displays error to User
Entry conditions: This use case extends the `Visualization` use case if an error occurs.
Flow of events:

1. The backend identifies the error.
2. The backend creates a response with the correct error code.
3. The webapp displays an error message.

Exit conditions:
Quality requirements:

U13 Use case name: UseCaseTemplate
Participating actors: Actors
Entry conditions: Entry cond.
Flow of events:

1. Flow 1
2. Flow 2

Exit conditions: Exit cond.
Quality requirements: Quality Requirements(?)

Glossary

benchmarking system System that runs benchmark for a code base..

configuration A complete description of a visualization. It contains all the necessary information except the benchmark data.

JSON JavaScript Object Notation.

template A partial configuration of a visualization, It contains preconfigured values, but leaves others blank for the user to costumize.

user Person working on the project that gets benchmarked.

visualization A graphical representation of benchmark data.