

CONTINUOUS INTEGRATION FOR HPC CODES

JLESC, April 17, 2024 | Jakob Fritz, Robert Speck | Jülich Supercomputing Centre, Forschungszentrum Jülich

Table of Contents

Continuous Integration for HPC

- Syncing with Gitlab tools

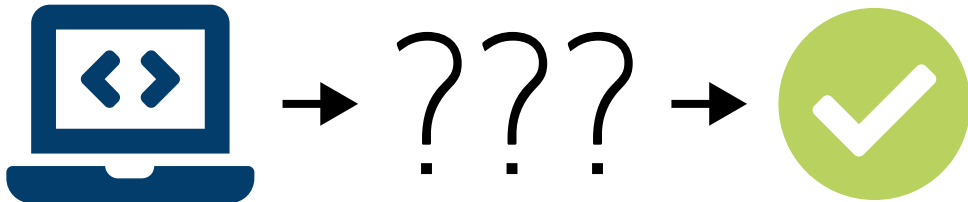
- Syncing Github to Gitlab

Continuous Benchmarking

- How to store results

- How to visualize results

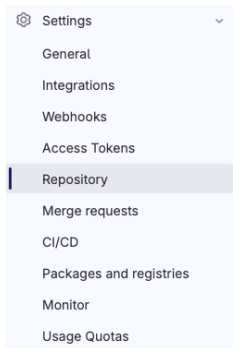
Continuous Integration for HPC



Continuous Integration for HPC

Syncing with Gitlab tools

Use Push-Mirror (can be found in the settings). Pull-Mirror only available in paid versions



Mirroring repositories

Set up your project to automatically push and/or pull changes to/from another repository. Branches, tags, and commits will be synced automatically. [How do I mirror repositories?](#)

Mirrored repositories @ 0

Add new mirror repository

Git repository URL

Input the remote repository URL.

- The repository must be accessible over `http://`, `https://`, `ssh://` or `git://`.
- When using the `http://` or `https://` protocols, please provide the exact URL to the repository. HTTP redirects will not be followed.
- Do not include the username in the URL, use the username field below if required: `https://gitlab.company.com/group/project.git`.
- When using the `ssh://` protocol, please use the following format: `ssh://username@hostname.com/group/project.git`.
- The update action will time out after 180 minutes. For big repositories, use a clone/push combination.
- Git LFS objects will be synced if LFS is [enabled for the project](#). Push mirrors will not sync LFS objects over SSH.
- In case of pull mirroring, your user will be the author of all events in the activity feed that are the result of an update, like new branches being created or new commits being pushed to existing branches.

Mirror direction

Push

Authentication method

Username and Password

Username

Password

☐ Keep divergent refs
Do not force push over diverged refs. After the mirror is created, this setting can only be modified using the API. [Learn more about this option and the API.](#)

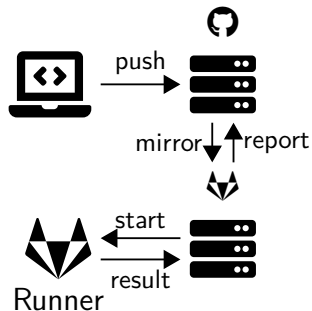
☐ Mirror only protected branches
If enabled, only protected branches will be mirrored. [Learn more.](#)

[Mirror repository](#) [Cancel](#)

Continuous Integration for HPC

Syncing Github to Gitlab

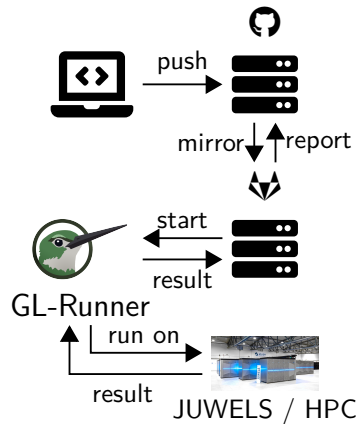
- Already shown last year



Continuous Integration for HPC

Syncing Github to Gitlab

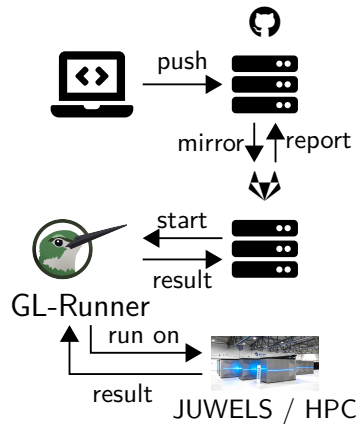
- Already shown last year
- Now used to trigger Jacamar and run on cluster with GPU usage (for testing)



Continuous Integration for HPC

Syncing Github to Gitlab

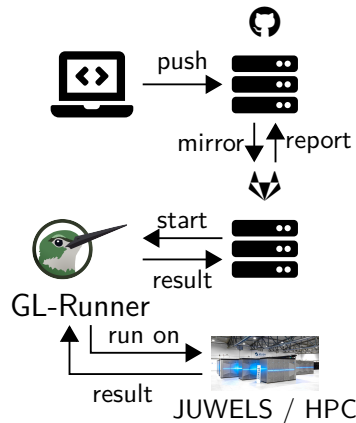
- Already shown last year
- Now used to trigger Jacamar and run on cluster with GPU usage (for testing)
- Returning results back to repository (Gitlab & Github) is of different difficulty. Success/Failure is easy; numbers are more difficult.
- Numbers are not only for benchmarking, but already for testing, (e.g. test coverage)



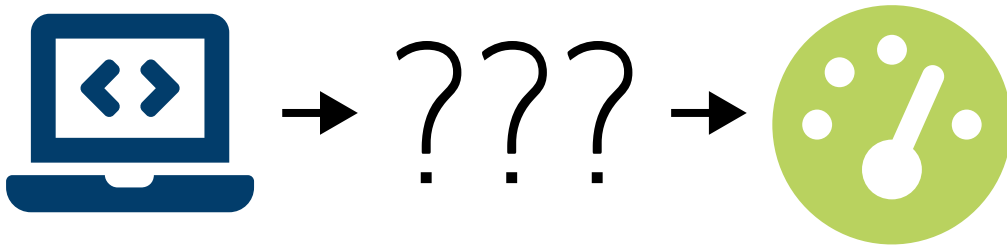
Continuous Integration for HPC

Syncing Github to Gitlab

- Already shown last year
- Now used to trigger Jacamar and run on cluster with GPU usage (for testing)
- Returning results back to repository (Gitlab & Github) is of different difficulty. Success/Failure is easy; numbers are more difficult.
- Numbers are not only for benchmarking, but already for testing, (e.g. test coverage)
- Artifacts created in Jacamar returned to Gitlab. Transfer to Github already present, but transfer between Github workflows of the same commit still needs development



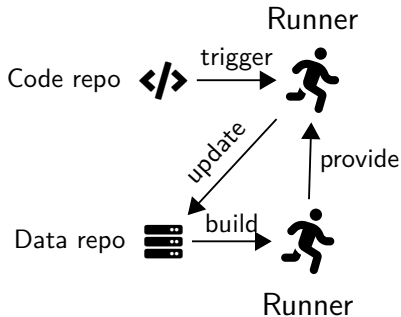
Continuous Benchmarking



Continuous Benchmarking

How to store results

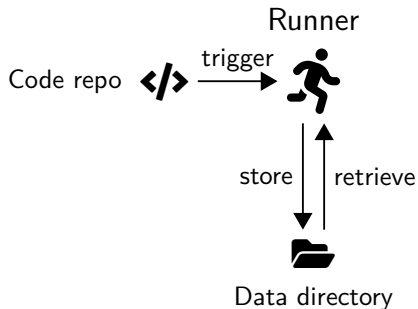
- Storage of results from benchmarks can be in separate repo
- For few metrics and/or meta-data a database can be used (e.g. sqlite) in a repository (therefore, versioned)



Continuous Benchmarking

How to store results

- Storage of results from benchmarks can be in separate repo
- For few metrics and/or meta-data a database can be used (e.g. sqlite) in a repository (therefore, versioned)
- For larger amounts of data, a file-dump or git-annex or datalad could be used (not done myself)



Continuous Benchmarking

How to visualize results

For large visualizations, a server with e.g. Grafana could be used

- ⇒ Backend server needed
- ⇒ Faster to provide updated data

Continuous Benchmarking

How to visualize results

For large visualizations, a server with e.g. Grafana could be used

- ⇒ Backend server needed
- ⇒ Faster to provide updated data

For smaller versions, static html with embedded JS is an alternative (e.g. via plotly)

- ⇒ Simple webserver sufficient
- ⇒ Easily scalable, as only static html-files are served

Continuous Benchmarking

How to visualize results

For large visualizations, a server with e.g. Grafana could be used

- ⇒ Backend server needed
- ⇒ Faster to provide updated data

For smaller versions, static html with embedded JS is an alternative (e.g. via plotly)

- ⇒ Simple webserver sufficient
- ⇒ Easily scalable, as only static html-files are served

This could be combined with Jupyter NB for easy development and with mkdocs if multiple pages shall be deployed.

- ⇒ Simple creation of sites and automatic creation of menu and footers

Continuous Benchmarking

How to visualize results

Example for website with Jupyter Notebooks and mkdocs

- Separate sqlite-files for each branch

```
/
├── cb-testing
│   └── results.sqlite
├── master
│   └── results.sqlite
├── .gitlab-ci.yml
├── mkdocs.yml
├── plot-results.ipynb
└── ...
```

Continuous Benchmarking

How to visualize results

Example for website with Jupyter Notebooks and mkdocs

- Separate sqlite-files for each branch
- Usage of template file of Jupyter NB
- This get copied for each branch automatically in CI

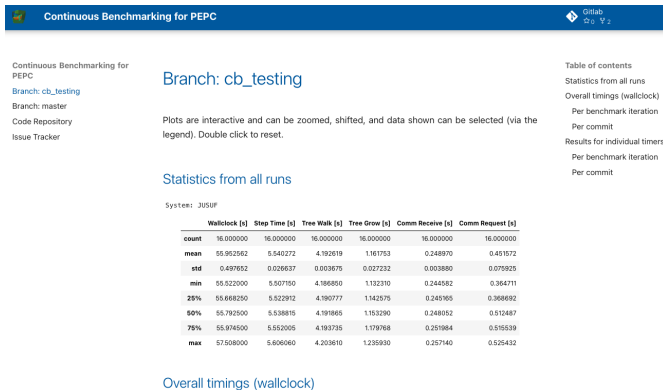
```
/
├── cb-testing
│   └── results.sqlite
├── master
│   └── results.sqlite
├── .gitlab-ci.yml
├── mkdocs.yml
├── plot-results.ipynb
└── ...
```

Continuous Benchmarking

How to visualize results

Example for website with Jupyter Notebooks and mkdocs

- Separate sqlite-files for each branch
- Usage of template file of Jupyter NB
- This get copied for each branch automatically in CI
- Created website



Continuous Benchmarking

How to visualize results

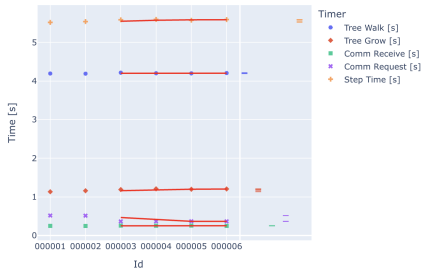
Example for website with Jupyter Notebooks and mkdocs

- Separate sqlite-files for each branch
- Usage of template file of Jupyter NB
- This get copied for each branch automatically in CI
- Created website
- With interactive graphs

Per benchmark iteration

The solid red line represents a trend in form of a sliding average of 3 values.

JUSUF



Thank you for your attention!

Link to GitHub2Lab:

https://github.com/jakob-fritz/github2lab_action



Link to CB-Website of PEPC:

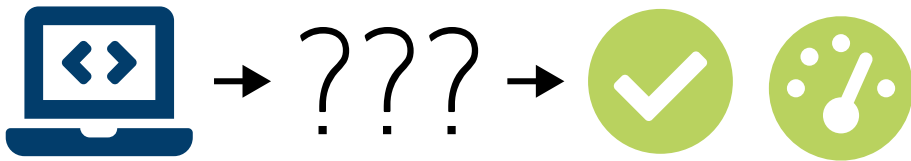
<https://slpp.pages.jsc.fz-juelich.de/pepc/pepc/>



Thank you for your attention!

If you are interested in the details of the codes or have questions,
feel free to contact me!

`j.fritz@fz-juelich.de`



CONTINUOUS INTEGRATION FOR HPC CODES

JLESC, April 17, 2024 | Jakob Fritz, Robert Speck | Jülich Supercomputing Centre, Forschungszentrum Jülich