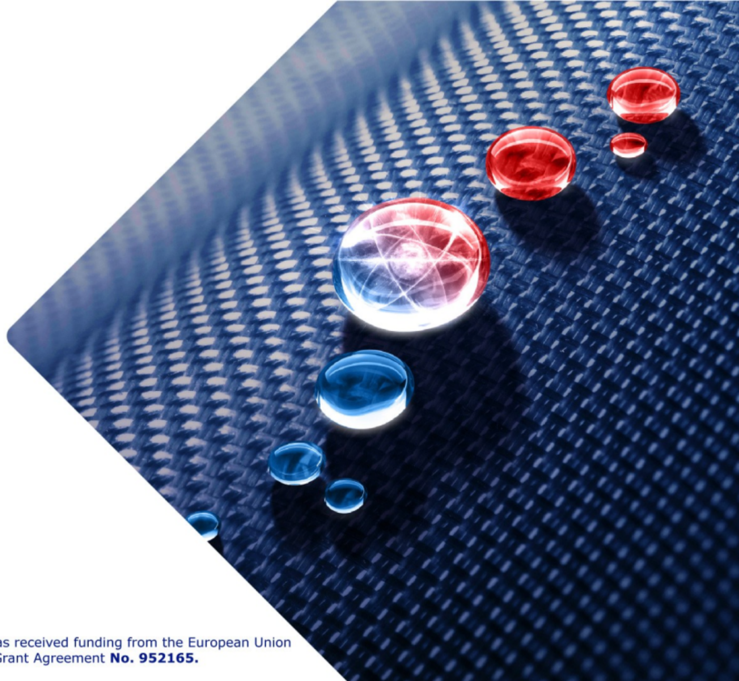
Targeting Real chemical accuracy at the EXascale

Introduction to Verificarlo CI

Aurélien Delval

November 21, 2022



Targeting Real Chemical Accuracy at the Exascale project has received funding from the European Union Horizoon 2020 research and innovation programme under Grant Agreement **No. 952165.**

Principle of Verificarlo CI

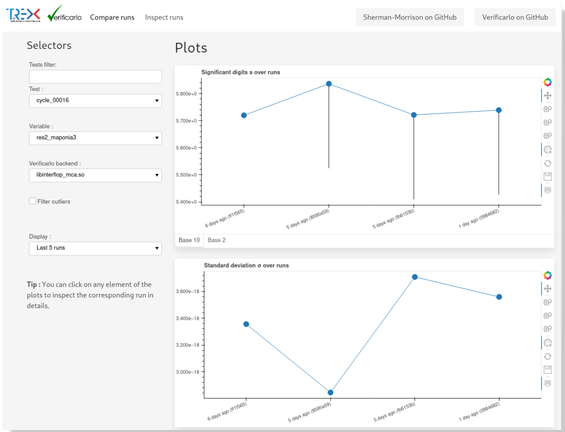
Verificarlo CI is a tool built on top of Verificarlo that lets its user :

- run
- automatize
- visualize

… custom Verificarlo tests.

These tests can be automatized with **Github Actions** or **Gitlab CI/CD**, and their results can then be visualized and shared in **HTML reports**.

Interpreting the report : ”Compare runs” mode



Selectors

Tests filter:

Test : cycle_00016

Variable : res2_maponia3

Verificarlo backend : libinterflop_mca.so

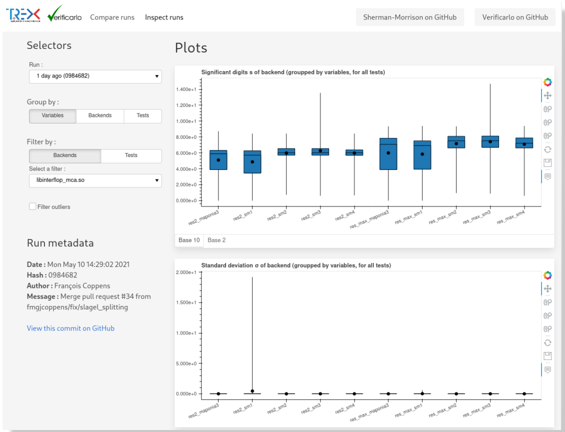
Filter outliers

Display : Last 5 runs

Tip : You can click on any element of the plots to inspect the corresponding run in details.

- Visualize evolution of **one variable over time** (commits)
- Shows significant digits s, standard deviation σ, variable distribution

Interpreting the report : ”Inspect runs” mode



Run : 1 day ago (0984682)

Group by : Variables | Backends | Tests

Filter by : Backends | Tests

Select a filter : libinterflop_mca.so

Filter outliers

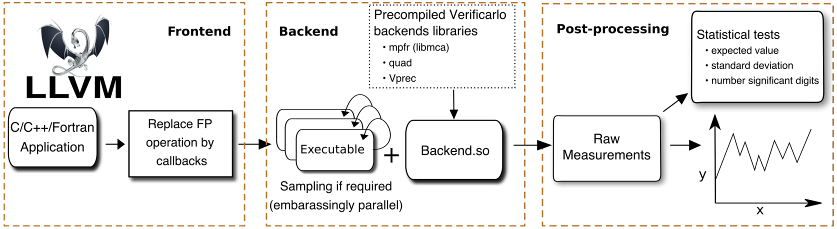
Run metadata

Date : Mon May 10 14:29:02 2021
Hash : 0984682
Author : François Coppens
Message : Merge pull request #34 from fmgjcoppens/fix/slagel_splitting

View this commit on GitHub

- Focus in depth on **one particular run**
- Different group by / filter by options
- For each group, shows distribution of s, σ and compute aggregated variable average

The Verificarlo CI pipeline



LLVM
Frontend
C/C++/Fortran Application
Replace FP operation by callbacks
Backend
Precompiled Verificarlo backends libraries
• mpfr (libmca)
• quad
• Vprec
Executable
+
Backend.so
Sampling if required (embarassingly parallel)
Post-processing
Raw Measurements
Statistical tests
• expected value
• standard deviation
• number significant digits
y
x

Verificarlo CI is integrated into Verificarlo as a **post-processing tool**.

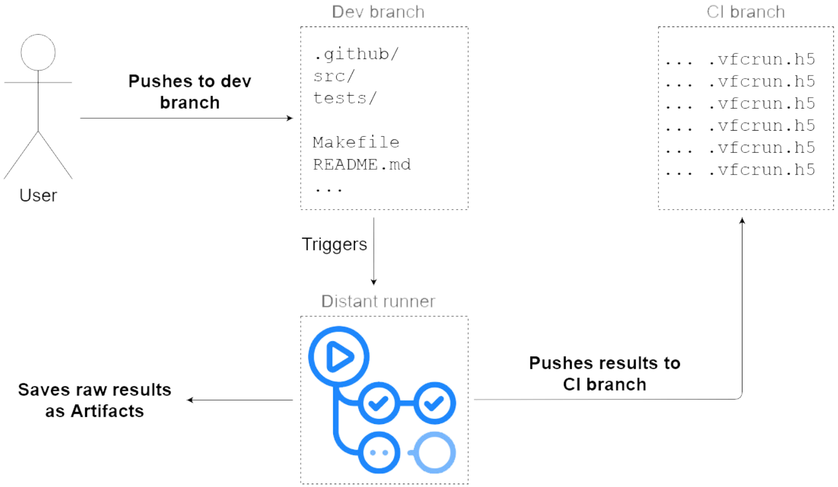
The Verificarlo CI pipeline

The CI integration pipeline relies on two different git branches :

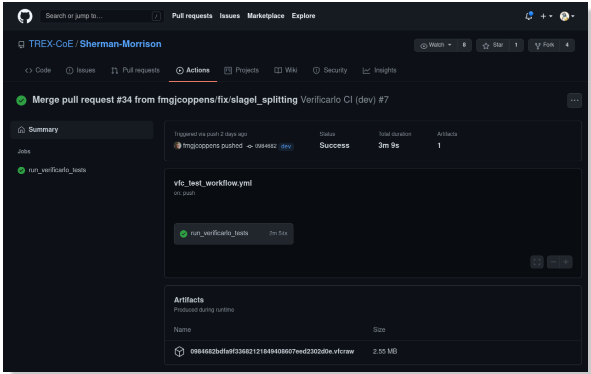
- **the ”dev” branch**, which is pre-existent and contains the project’s source code
- **the ”CI” branch**, which will be automatically be updated with result files, and is completely orphan of the dev branch

Note that while the tool has been **designed to integrate with git and CI systems**, it can still be used **completely independently**, simply as a way to visualize test results.

The Verificarlo CI pipeline



Description of the CI workflow



TREX-CoE / Sherman-Morrison

Merge pull request #34 from fmgjcoppens/fix/slagel_splitting Verificarlo CI (dev) #7

Triggered via push 2 days ago
fmgjcoppens pushed 0984682 dev

Status: Success
Total duration: 3m 9s
Artifacts: 1

Jobs
run_verificarlo_tests

vfc_test_workflow.yml
on: push

run_verificarlo_tests 2m 54s

Artifacts
Produced during runtime

Name	Size
0984682bdfa9f33682121849408607eed2302d0e.vfcraw	2.55 MB

- Workflow runs will be triggered **after each push** by default
- The CI branch will be **updated automatically**
- Raw results will be accessible as **Github Artifacts**
- The CI setup can be automated with the `vfc_ci setup` command

The probes system

In order to export test data from the programs to the tool itself, Verificarlo CI uses a **probes** system which allows the user to :

- **define test variables** (or *probes*) identified by a unique test name / variable name combination
- associate them with an **optional accuracy target** (absolute or relative)

... and **interfaces with Verificarlo CI** itself.

The probes system

- Here is a basic example where we want to store the value of var in a probe called varName belonging to a test testName in **C** :

```
vfc_probe(&probes, "testName", "varName", var);
```

- A **Fortran** interface is also provided using the *ISO_C_BINDING* module:

```
vfc_probe(probes,
"testName//C_NULL_CHAR",
"varName//C_NULL_CHAR",
 var);
```

The `vfc_ci` CLI

Verificarlo CI is accessed through a single command-line interface, *`vfc_ci`*. It provides 3 different subcommands :

- *`setup`* : an helper script to initialize the **CI workflow**
- *`test`* : start a **test run**
- *`serve`* : launch a server giving access to the **HTML report**

Tests configuration file

The *vfc_ci test* command manages the tests runs. It is configured with the *vfc_tests_config.json* file :

```
{
  "make_command": "make tests",
  "executables": [
    {
      "executable": "bin/test",
      "parameters" : "foo",
      "vfc_backends": [
        {
          "name": "libinterflop_mca.so --precision=53",
          "repetitions": 20
        }
    }, [...]
}
```

The `vfc_ci test` command

After executing the tests and gathering the results, the **postprocessing pipeline** is executed. For stochastic backends, we compute for each probe :

- the **empirical average** μ and the *standard deviation* σ
- the **number of significant digits** s_2, s_{10}
- the distribution's **quantiles**

The accuracy checks (with a target t) are evaluated as follows :

- if the check is **absolute**, $\sigma < t$?
- if the check is **relative**, $\frac{\sigma}{|\mu|} < t$?

To reproduce the tutorial: Links

- The **Verificarlo CI tutorial**, can be found at *`https://github.com/verificarlo/vfc_ci_tutorial`*
- For more details, the **Verificarlo CI documentation** can be found at *`https://github.com/verificarlo/verificarlo/blob/master/doc/06-Postprocessing.md#verificarlo-ci`*
- If you plan to use either Singularity or Docker, you will need to run the following script just before starting your server to update some Python packages (follow the instructions) : *`https://gist.github.com/PurplePachyderm/2ae9fedf7f458bdae7534adc0d5862e0`*

To reproduce the tutorial: Singularity

- If you plan to use the **Singularity image on CALMIP** while connecting with the VPN, you can set up port forwarding with :

```
> ssh -L [port]:127.0.0.1:[port]
-p 11300 [username]@127.0.0.1
```

- Then, to load the Verificarlo Singularity image:

```
> module load singularity
> singularity run /usr/local/trex/verificarlo/ver [...] .sif
```

To reproduce the tutorial: Docker

- If you plan to use the **Docker image**, you can download it and setup a container with port forwarding :

```
> docker pull verificarlo/verificarlo
> docker run -it -p [port]:[port]
verificarlo/verificarlo /bin/bash
```

- You can optionally set up a shared directory when creating the container:

```
-v [/path/to/host/dir]:[/path/to/docker/dir]
```