IT Platform

May 20, 2021

Utz Uwe-Haus, Alfio Lazzaro
HPE Switzerland

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 773897

Overview

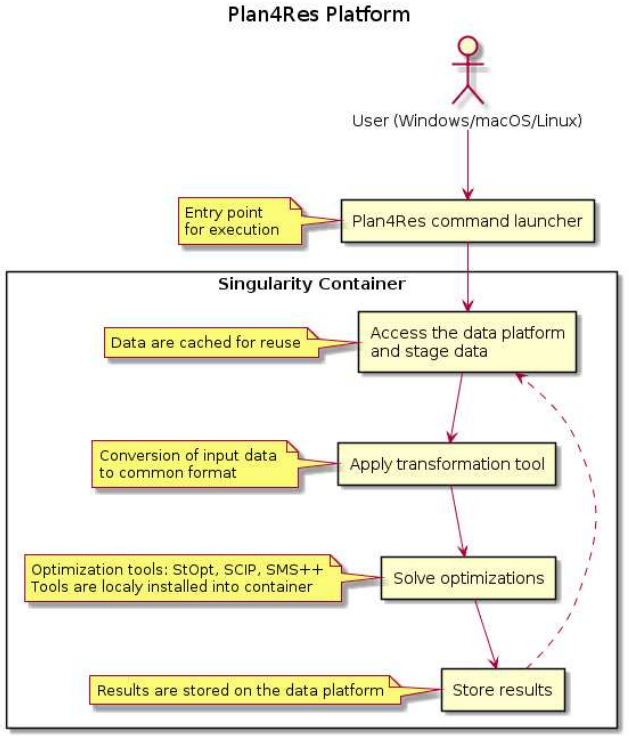
- Requirements
- Workflow
- Containerized Compute Environment
- Plan4res Singularity Container
- Plan4res Environment
- Add-ons

Requirements

- Suitable to be used on all partner site's compute infrastructure
- Suitable to be used on a wide range of hardware performance classes
  - From Laptops to Supercomputers
- Usable to obtain repeatable results over time
- Suitable to run the necessary data transformations and optimization codes
- Permitting exchange of data and code as well as executables between partners

Workflow

- Containerized Compute Environment
  - Everything executed within the Compute Environment with a command launcher as entry point
  - Staging ➔ Transformations ➔ Solve ➔ Store phases
  - Eventually store the results back to the data platform



Containerized Compute Environment

- A container consists of an entire runtime environment, i.e. Linux OS + your applications + all their dependencies, libraries and other binaries and configuration files needed to run
  - Everything bundled in one package
- Same executables run everywhere
- No dependency issues
- Add-on software (license restricted) can be locally added in a standardized way
  - Eg.: CPLEX, SCIP, StOpt
- Directory structure layout predefined so software can rely on it cross-site

Plan4res Singularity Container

- Singularity is a well established technology for HPC containers
  - https://github.com/sylabs/singularity
  - https://sylabs.io/singularity/
- Single file based container images
  - Facilitates distribution, archiving, and sharing
- Plan4res container is automatically built via a GitLab CI execution and published as an artifact
  - Repository at https://gitlab.com/cerl/plan4res/p4r-exec-singularity
  - Any push of the Singularity image definition file triggers a new build of the image

Supported Host Operating Systems

- Singularity natively runs on Linux system
  - We have tested the Plan4res Environment on several Linux systems
- For Windows and macOS, we use a virtualized Linux, e.g. VirtualBox (https://www.virtualbox.org/)
  - Requires to install a virtual machine, e.g., through Vagrant to install and manage a minimal Linux box (https://www.vagrantup.com/)
- The Plan4res environment commands do not depend on the host Operating System
  - Users have a common set of commands for all Operating Systems

Container Dependencies

- Base container is Debian 10 ("Buster")
- We install the following main additional packages
  - Octave, Python3, Clang
- Optimized source compilation installation of the following packages
  - Eigen, MPICH, HDF5, NETCDF, Boost
- Image size is 901MB

Plan4res Environment

- Available at https://gitlab.com/cerl/plan4res/p4r-env
  - Ensure good usability for non-technical users
- Directory structure allows to run across multiple systems
  - `bin`: contains all software that is needed for the workflow execution
  - `config`: contains the configuration files to set the plan4res environment
  - `data`: staging of the input/output data
  - `scripts`: contains advanced script for the installation of the add-ons and the execution of the workflows

Plan4res Command Launcher

- bin/p4r [command] [<options>]
  - Single entry point to launch commands within the containerized environment
  - It reads the configuration files, download any new version of the container (it checks for changes first to avoid long downloads), and run the command
  - If no command or options are provided, it opens a shell within the container
- Eg., open a shell:

```
> bin/p4r
  Updating image - if you want to avoid this set
  'P4R_SINGULARITY_IMAGE_PRESERVE=1' in plan4res.conf
  ;; No tool specified to run. Starting a shell inside the
  ;; container.
  ;; If this is not what you intended, specify the command
  ;; to invoke on the command line of the p4r script.
  ;; Run 'p4r -h' to get help.
  [P4R-ENV] ~/Work/p4r-env >
```

Add-ons

- Common extra tools that are not inserted in the container
  - Due to licensing issues
  - Due to frequent updates (to not require a rebuild of the container)
- Installation of the add-ons based on a recipe file, which can be executed via the command

  `> bin/p4r add-on <add-on name> [<target>]`
- Available add-ons:
  - SCIP
  - StOpt
  - SMS++
  - Transformation tools

Add-ons Targets

- Each add-on has a set of targets, eg.

```
> bin/p4r add-on stopt help
Targets for stopt add-on (first target as default):
     install : Install stopt add-on
      update : Update and re-install stopt
       check : Check stopt add-on
      status : Print stopt version
       clean : Clean stopt build directory
   uninstall : Remove stopt build and installation directories
        help : This help
```

- Some of the targets (eg. install, update, status) are common to all add-ons
  - Very easy to interact with add-ons, uers no need to know low-level installation details

Parallel Execution

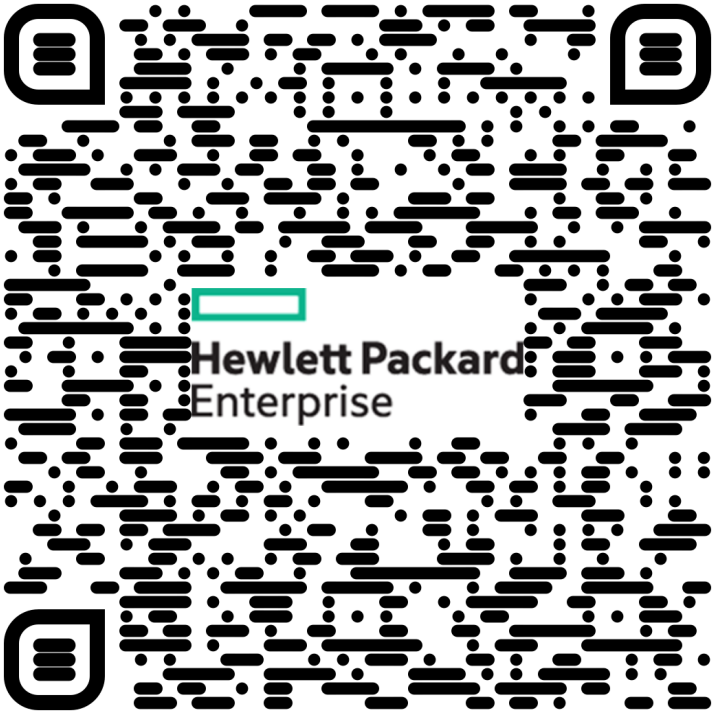
- Parallel execution via MPI for StOpt and SMS++
- Integration with batch systems, eg. SLURM
  - Tested on AWS (access kindly provided by EDF) and HPE/Cray Supercomputers
- Example: SDDP execution on the AWS cloud with different MPI processors, timings:
  - 1 processor: 903s
  - 2 processors: 638s
  - 4 processors: 325s
  - 8 processors: 231s


MINGW64:/c/Users/alazzaro/plan4res
alazzaro@ALFIOLAZZAR8D9D MINGW64 ~/plan4res
$
alfio.lazzaro@ip-10-0-83-10:~/fsx/demo
[alfio.lazzaro@ip-10-0-83-10 demo]$

Conclusion

- Containerized environment is a convenient way to distribute codes
  - Easy to deploy, run anywhere
  - Support of Windows, Linux, macOS
  - Run on Laptops, Workstations, Supercomputers, AWS instances
- Plan4res environment ensures good usability for non-technical users
- Easily extendable with add-ons for specific user codes

Thank you



uhaus@hpe.com
alfio.lazzaro@hpe.com
nina.mujkanovic@hpe.com