INTERVENE

INTERVENE consortium AI algorithms supported by CSC HPC environment on synthetic data

Deliverable 1.5

Dissemination level: Public

Part of:

WP 1:
Compliant and standardised data access for federated analyses

Project summary	
Project acronym:	INTERVENE
Project full title:	International consortium for integrative genomics prediction
Project Coordinator	Institute for Molecular Medicine Finland FIMM, University of Helsinki; Prof. Samuli Ripatti and Dr. Andrea Ganna
Project start date:	1.1.2021
Project end date:	31.12.2025
Project duration:	60 months
Action type:	RIA
Call identifier:	H2020-SC1-FA-DTS-2018-2020 (Trusted digital solutions and Cybersecurity in Health and Care)
Grant number	101016775

Document descriptors	
Deliverable No.	1.5
Work package	WP1
Deliverable lead	CSC
Contributors	CSC, EMBL,
Dissemination level	Public
Expected delivery date	31/12/2021
Submission date	28/12/2021

Change history log

Version	Changes made	Date	Prepared by	PC approved
0.1	First version	21.12.2021	CSC (K. Mattila)	
1.0	Technical edits	28.12.2021	Julius Anckar (UH-FIMM)	Yes

Contents

1. Background

One of the main targets of the INTERVENE project is to provide new methods and services for calculating Integrated Genetic Scores IGS (also called as Polygenic Risk Scores). New IGS methods are developed in INTERVENE WP3 while WP4 aims to build a service that allows easy evaluation of new methods developed by external users. Both of these WPs will produce a large amount of IGS calculation tasks. WP1 develops methods to enable usage of general-purpose High-Performance Clusters (HPC) for executing these computing tasks as that would provide a cost-effective way to perform these computing tasks.

In this deliverable we report the results of the first tests where the IGS calculation pipelines were installed to the HPC environment of CSC. These relate mainly to WP1 sub-objective 2:

> *To support the deployment of a multi-study analysis platform for testing and development of integrative genetic scores (in collaboration with WP2, WP3, and WP4).*

The IGS calculation pipelines developed in INTERVENE use software containers and modern workflow managers. The tests show that this approach allows easy implementation of the pipeline into a general purpose HPC cluster as well as transportability to secure computing environments with no internet access.

2. Installation tests

Installation and execution of two pipelines, **PGSC_CALC** and **PRSPIPE** were demonstrated. Both pipelines are still under development. At this stage we were doing just technical testing to ensure that these pipelines can be used in the computing environments of CSC. Installation and execution tests were run in the **Puhti cluster** that is a general purpose HPC cluster providing both CPU and GPU capacity. In addition to this, the pipelines were also installed and tested in the virtual sensitive data analysis environment of CSC: **SD Desktop.**

INTERVENE project members get access to Puhti and SD Desktop services either by joining the CSC project of INTERVENE Data Coordination Center or by setting up their own computing project at CSC.

2.1 PGSC_CALC

pgsc_calc is a bioinformatics best-practice analysis pipeline for applying scoring files from the Polygenic Score (PGS) Catalog[1] to target genotyped samples.

The pipeline is built using Nextflow[2], a workflow tool to run tasks across multiple compute infrastructures in a very portable manner. It uses Docker/Singularity containers making installation trivial and results highly reproducible. The Nextflow DSL2[3] implementation of this pipeline uses one container per process which makes it much easier to maintain and update software dependencies.

The pipeline can be downloaded from:

https://github.com/PGScatalog/pgsc_calc/

PGSC_calc has two major requirements from the computing platform. It must provide 1) Nextfow workflow engine and 2) PGSC_calc compatible container or package manager.

The workflow engine orchestrates the execution of the tasks of a pipeline while the package manager retrieves and executes the software pipeline components. In the case of CSC we use **Singularity** package manager, the other options being Docker, Podman, Shifter, Charliecloud or Conda .

2.2 PGSC_calc installation details

Puhti cluster can be accessed by ssh connections or by opening terminal session in the web interface of Puhti[4].

In Puhti Singularity it is recommended to be executed in a batch job environment. For tests we used interactive batch job that was started with command:

```
sinteractive -m 16G -c 4 -d 100
```

Once the interactive batch job session is running, the nextflow module is activated and set environment variables set to make singularity to use $TMPDIR for temporary files:

```
module load biokit
module load biopythontools/3.9.1-gcc_9.1.0
module load nextflow/21.10.4.5656
module load biojava/16
export SINGULARITY_TMPDIR=$TMPDIR
export SINGULARITY_CACHEDIR=$TMPDIR
```

[1] https://www.pgscatalog.org/
[2] https://www.nextflow.io/
[3] https://www.nextflow.io/docs/latest/dsl2.html
[4] https://www.puhti.csc.fi.

The pipeline is downloaded to the scratch area of the project from the git repository with commands below. In the example commands here use the project of INTERVENE Data Coordination Center (project_2004504). Other projects should replace the project number with their own project number.

```
$ module load git
$ cd /scratch/project_2004504
$ curl -LO
https://github.com/PGScatalog/pgsc_calc/archive/refs/tags/v0.1.1dev.zip
$ unzip v0.1.1dev.zip
$ cd pgsc_calc-0.1.1dev
```

Next, NXF_SINGULARITY_CACHEDIR variable is defined, so that the singularity containers will be stored in a directory that is located in the pgsc_calac directory. This allows us to convert the pgsc_calc directory into a package that can be exported and executed in the SD Desktop environment.

```
export NXF_SINGULARITY_CACHEDIR=$(pwd)/singularity_cache
```

After this we are ready to run a test job, that is launched with command:

```
nextflow run main.nf -profile singularity,test
```

Parameters are defined in a yaml file (*params-file*) or can be set on the command line. Other tasks can be executed by changing the settings in the *params-file* or by explicitly defining the parameters in the next flow commands line. e.g.:

```
nextflow run pgscatalog/pgsc_calc -profile singularity --input samplesheet.csv
--accession PGS001229
```

The pipeline can also be used in normal batch job files in Puhti to the SLURM scheduler. This would allow users to send large amounts of tasks to be executed in the Puhti cluster.

The pipeline outputs polygenic scores for each individual in the target genomic data, calculated from a polygenic scoring file. The polygenic scoring file can be fetched from the PGS Catalog or a local file path can be specified. A summary report is also made to describe the process that calculated the polygenic scores.

2.3 Transporting workflow to SD Desktop

In the example above, the pgsc_pipe was executed in Puhti cluster. Puhti can be used for processing non sensitive data e.g. like running pgsc_pipe with synthetic data, but in the cases of real human genetic data, the tasks need to be executed in a more secure computing environment like the SD Desktop[5] at CSC.

When the above command was executed in Puhti for the first time, it downloaded the singularity containers it needs to the *singularity_cache* directory. After that, *pgsc_pipe* directory contains all the components needed to execute the task. This directory can be packed to a one tar file and uploaded to the CSC sensitive data repository: SD Connect[6]. This can be done with commands:

```
module load allas
allas-conf
tar cvf pgsc_calc.tar pgsc_calc
a-put –nc –sdx pgsc_calc.tar -b 2004504_sd-pipelines
```

The commands above pack the contents of pgsc_calc directory into a tar file and then store the pipeline to a project specific bucket (2004504_sd-pipelines in this case). If needed, the pipeline copy can be encrypted so that it can be opened only in the SD Desktop environment. In this case this is achieved by using *a-pu*t with option *–sdx.*

From SD Connect service the packed pipeline can be downloaded to a virtual machine running in the SD Desktop environment. The virtual machines running in SD Desktop include both Nextflow and Snakemake workflow mangers. After opening the downloaded tar file the, *NXF_SINGULARITY_CACHEDIR* is set to point to the current location in the virtual machine.

```
tar xvf pgsc_calc.tar
cd pgsc_calc
export NXF_SINGULARITY_CACHEDIR=$(pwd)/singularity_cache
```

Now the pipeline could be executed just like in Puhti.

```
nextflow run pgscatalog/pgsc_calc -profile singularity --input\
samplesheet.csv --accession PGS001229
```

SD Desktop has limited resources, so it cannot be used for running massive amounts of heavy pgsc_calc tasks. However, **this test demonstrates that the pgsc_calc pipeline can easily be converted into a file that can be executed in an isolated secure computing environment. Thus, it will be compatible with the sensitive data HPC environment currently developed at CSC.**

[5] https://sd-desktop.csc.fi/
[6] https://sd-connect.csc.fi/

3. prspipe

Prspipe is Snakemake pipeline to run Polygenic Risk Score (PRS) prediction. It implements and extends the GenoPred pipeline, i.e. a reference standardized framework for the prediction of PRS using different state of the art methods using summary statistics.

Prspipe is available in GitHub in repository:
https://github.com/intervene-EU-H2020/prspipe

The full pipeline can roughly be divided into two stages: (1) Download and adjustment of summary statistics using pre-computed LD reference panels where available and data from the 1000 Genomes project (public data), and (2) prediction and evaluation of PRS using the adjusted summary statistics, which includes hyper-parameter tuning using cross validation. The second step uses sensitive biobank data. As biobank data can't yet be processed in Puhti, only the first step of the pipeline was tested.

3.1 Installing prspipe to Puhti

In Puhti the installation was done using Conda package manager and Singularity software container environment. In the approach used here, a group specific conda environment was first created with the commands below. The conda environment was in this test named as prspipe. In the test we used the project of INTERVENE Data Coordination Center (project_2004504). Other projects should replace the project number with their own project number.

```
export PROJAPPL=/scratch/project_2004504
module load bioconda
conda create -n prspipe
conda activate prspipe
conda install snakemake
```

In addition users needed to add following rows to their .bashrc.

```
# Make these functions available to scripts.
 export -f conda
 export -f __conda_activate
 export -f __conda_reactivate
 export -f __add_sys_prefix_to_path
 export -f __conda_hashr
 export -f __conda_exe
```

By default, singularity uses home directory for temporary files. In Puhti, the home directory is very small however. To avoid singularity to fill up the home directory, users need to define following environment variables:

```
export SINGULARITY_TMPDIR=$TMPDIR
export SINGULARITY_CACHEDIR=$TMPDIR
```

After these settings, installation was executed with following commands.

```
cd /scratch/project_2004504
git clone https://github.com/intervene-EU-H2020/prspipe
cd prspipe
bash ./install_software.sh
```

At the time of test installation, the link to the plink2 file in the install script was outdated.
After adding plink2 manually to ./bin directory the installation could be executed with commands:

```
bash run.sh --use-singularity get_plink_files_chr_all
download_hapmap3_snplist
bash run.sh --use-singularity all_setup
bash run.sh cleanup_after_setup
bash run.sh download_test_data
bash run.sh -n validate_setup_ext
```

4. Summary

The first versions of PGSC_calc and PRSpipe were successfully installed to Puhti HPC cluster at CSC. Usage of PGSC_calc was also tested in the CSC sensitive data platform: SD Desktop. The results show that these tools can utilize the computing resources provided by CSC. Both pipelines used in the tests are still under rapid development. Thus, the procedures described above may not be fully compatible with the newer releases of these pipelines. Internal communication and regular testing will however ensure that the pipelines will be compatible with the available computing environments in the future too.