Getting Started Guide

The folder framework contains the source code of the modified COMPSs runtime implementing the contribution withing the article. Although the results presented in the paper have been conducted in the MareNostrum4 Supercomputer, the tests can be executed in other computing infrastructures ranging from one single computer to any other cluster.

Installation

Baremetal

The prototype build on COMPSs 3.1; hence, it inherits all its dependencies which are described in the following **page** [1]. To setup the testing environment on your own computer, it is necessary to install the modified version of the framework following the next instructions:

```
> cd framework/builders/
> sudo -E ./buildlocal  -K -J -T -D -C -M --skip-tests /opt/COMPSs
```

For installing the software in a cluster with queue system, please refer to the corresponding **section** [2] in the official COMPSs guide.

Container image (Recommended option)

To simplify the installation process, the artifact provides a mechanism to build a docker image and run the tests in a containerized environment. The image can be created running the following the next commands in the root folder of the artifact

```
> docker build -t compss_nested .
```

Execution

The test folder contains all the necessary code and scripts to run the different experiments in the Evaluation section of the paper. Inside it there is one folder for each application (gridsearch and random_forest) containing the source code of the application, the used datasets and scripts for launching the execution locally or to submit the execution to a queue system.

1 https://compss-doc.readthedocs.io/en/3.1/Sections/01_Installation/01_Dependencies.html

2 https://compss-doc.readthedocs.io/en/3.1/Sections/01_Installation/04_Supercomputers.html#

Local Execution

Besides, in the root folder of the artifact there is the `launch_test` script that simplifies the execution of all the test presented in the Evaluation section of the paper using the baremetal installation. This script is the entrypoint of the Docker image created in the previous section; hence, running `docker run --rm compss_nested` would start the exact same execution but in a docker container.

The first parameter of the script selects the experiment which of the applications is to be executed. To run the Gridsearch experiment (Subsection 5.1), run
`launch_test gridsearch`
or
`docker run --rm compss_nested gridsearch`;
for the tests executing the RandomForest application, run
`launch_test random_forest`
or
`docker run --rm compss_nested random_forest`.

GridSearch (Section 5.1)

Here is a list of the different parameters to pass in to to the execution step

1) **VERSION** (Mandatory): accepts two values `FLAT` or `NESTED`. Selects which of the two versions described in the article is executed.

2) **NUM_COMBINATIONS** (Mandatory): Indicates the number of combinations of CSVM parameters to compare. It only accepts 25 and 50.

3) **NUM_GRIDSEARCH** (Default value:1): Indicates how many executions of the application (with their respective time measurements) will be done in the same run.

4) **DATASET**: (Default value IRIS): accepts three different values `IRIS`, `digits` and `AT`. Selects which of the datasets will be used to run the gridsearch. In the article only IRIS (small dataset) and AT are used.

The following commands are two equivalent commands to run the 5 executions of the NESTED version of the gridsearch to optimize the CSVMs using the Atrial Fibrillation dataset considering 25 combinations. The first runs using the baremetal installation and the second one the container image created before.

```
> ./launch_test gridsearch NESTED 25 5 AT
> docker run --rm compss_nested gridsearch NESTED 25 5 AT
```

Random Forest (Section 5.2)

Here is a list of the different parameters to pass in to to the execution step

1. **VERSION** (Mandatory): accepts two values `FLAT` or `NESTED`. Selects which of the two versions described in the article is executed.
2. **NUM_MODELS** (Default value:1): Indicates how many executions of the application (with their respective time measurements) will be done in the same run.
3. **NUM_ESTIMATORS** (Default value: 1) number of estimators to compute for each model

4. **NUM_ESTIMATORS_PER_BATCH** (Default value 48) amount of estimators that are grouped within an intermediate task in the NESTED version. In the FLAT version this argument is ignored.

The following commands are two equivalent commands to train 5 random forest models with 12288 estimators using the NESTED version of the algorithm creating batches of 1024 estimators. The first runs using the baremetal installation and the second one the container image created before.

```
> ./launch_test random_forest NESTED 5 12288 1024
> docker run --rm compss_nested random_forest NESTED 5 12288 1024
```

Queue System

There is a `enqueue` script within the folder of each test (tests/gridsearch and tests/random_forest) to launch the execution in a multi-node setup. The scripts accept one additional parameter that is the number of nodes to be used in the execution.

The following command uses 16 nodes to run 5 executions of the NESTED version of the Gridsearch using the Atrial Fibrillation dataset

```
> cd tests/gridsearch
> ./enqueue 16 NESTED 25 5 AT
```

Step-by-step Instructions

To reproduce the results of the tests, first you need to install the application on your system as explained in the **Installation** section of this document. The tests were conducted in the MareNostrum4 Supercomputer and to reproduce the results it is necessary to have access to it and use the enqueue scripts provided in the application-specific folders.

Since the tests can be executed also locally, we will provide a list of the commands used for executing each test using the container image to simplify the installation process.

To create the docker image, download the ZIP file of this artifact, decompress it within a specific folder and navigate with your favorite shell command interface change the working directory to where the file has been decompressed (we will use *<wdir>* to indicate path to this folder). Then, create your docker image with the following command:

```
> cd <wdir>
> docker build -t compss_nested .
```

The following subsections contains the exact commands that were used to run all the tests within the article. For each execution, we trained 5 different models and averaged the execution time. To reproduce its execution, just run the corresponding command in the list.

GridSearch

- 25 parameter combinations with Small dataset (IRIS)
  - FLAT version (seconds to train a model with 4 nodes: 116.27)

```
> docker run --rm compss_nested gridsearch FLAT 25 5 IRIS
```

  - NESTED version (seconds to train a model with 4 nodes: 9.33)

```
> docker run --rm compss_nested gridsearch NESTED 25 5 IRIS
```

- 25 parameter combinations with Large dataset (AT)
  - FLAT version (seconds to train a model with 4 nodes: 27,887)

```
> docker run --rm compss_nested gridsearch FLAT 25 5 AT
```

  - NESTED version (seconds to train a model with 4 nodes: 4,315)

```
> docker run --rm compss_nested gridsearch NESTED 25 5 AT
```

- 50 parameter combinations with Large dataset (AT)
  - FLAT version (estimated 55,774 seconds to train a model)

```
> docker run --rm compss_nested gridsearch FLAT 50 5 AT
```

  - NESTED version (seconds to train a model with 16 nodes: 4,315; with 50, 1,824)

```
> docker run --rm compss_nested gridsearch NESTED 50 5 AT
```

Random forest

- FLAT version:

```
> docker run --rm compss_nested random_forest FLAT 5 <ESTIMATORS>
```

  Replace <ESTIMATORS> by the desired number of estimators (in the paper,we used 1, 1024, 3072, 6144, 12288)
- NESTED version

```
> docker run compss_nested random_forest FLAT 5 <ESTIMATORS> <EST_PER_BATCH>
```

  Replace <ESTIMATORS> by the desired number of estimators (in the paper,we used 1, 1024, 3072, 6144, 12288)
  Replace <EST_PER_BATCH> with the desired number. Since we used several nodes, in the paper, the results of the paper were using MAX(48, <ESTIMATORS>/NUM_NODES)