Geophysical Research Abstracts
Vol. 17, EGU2015-6790-3, 2015
EGU General Assembly 2015

Dispel4py: An Open-Source Python library for Data-Intensive Seismology

Rosa Filgueira (1), Amrey Krause (2), Alessandro Spinuso (1,3), Iraklis Klampanos (1), Peter Danecek (4), and Malcolm Atkinson (1)

(1) School of Informatics, University of Edinburgh, Ediburgh, U.K, (2) EPCC, School of Physics, University of Edinburgh, Ediburgh, U.K, (3) The Royal Netherlands Meteorological Institute (KNMI), Utrecht, Netherelands, (4) Istituto Nazionale di Geofisica e Vulcanologia Centro Nazionale Terremoti (INGV), Roma, Italy

Scientific workflows are a necessary tool for many scientific communities as they enable easy composition and execution of applications on computing resources while scientists can focus on their research without being distracted by the computation management.

Nowadays, scientific communities (e.g. Seismology) have access to a large variety of computing resources and their computational problems are best addressed using parallel computing technology. However, successful use of these technologies requires a lot of additional machinery whose use is not straightforward for non-experts: different parallel frameworks (MPI, Storm, multiprocessing, etc.) must be used depending on the computing resources (local machines, grids, clouds, clusters) where applications are run. This implies that for achieving the best applications‘ performance, users usually have to change their codes depending on the features of the platform selected for running them.

This work presents dispel4py, a new open-source Python library for describing abstract stream-based workflows for distributed data-intensive applications. Special care has been taken to provide dispel4py with the ability to map abstract workflows to different platforms dynamically at run-time. Currently dispel4py has four mappings: Apache Storm, MPI, multi-threading and sequential.

The main goal of dispel4py is to provide an easy-to-use tool to develop and test workflows in local resources by using the sequential mode with a small dataset. Later, once a workflow is ready for long runs, it can be automatically executed on different parallel resources. dispel4py takes care of the underlying mappings by performing an efficient parallelisation.

Processing Elements (PE) represent the basic computational activities of any dispel4Py workflow, which can be a seismologic algorithm, or a data transformation process.

For creating a dispel4py workflow, users only have to write very few lines of code to describe their PEs and how they are connected by using Python, which is widely supported on many platforms and is popular in many scientific domains, such as in geosciences.

Once, a dispel4py workflow is written, a user only has to select which mapping they would like to use, and everything else (parallelisation, distribution of data) is carried on by dispel4py without any cost to the user.

Among all dispel4py features we would like to highlight the following:
* The PEs are connected by streams and not by writing to and reading from intermediate files, avoiding many IO operations.

* The PEs can be stored into a registry. Therefore, different users can recombine PEs in many different workflows.

* dispel4py has been enriched with a provenance mechanism to support runtime provenance analysis. We have adopted the W3C-PROV data model, which is accessible via a prototypal browser-based user interface and a web API. It supports the users with the visualisation of graphical products and offers combined operations to access and download the data, which may be selectively stored at runtime, into dedicated data archives.

dispel4py has been already used by seismologists in the VERCE project to develop different seismic workflows. One of them is the Seismic Ambient Noise Cross-Correlation workflow, which preprocesses and cross-correlates traces from several stations. First, this workflow was tested on a local machine by using a small number of stations as input data. Later, it was executed on different parallel platforms (SuperMUC cluster, and Terracorrelator machine), automatically scaling up by using MPI and multiprocessing mappings and up to 1000 stations as input data. The results show that the dispel4py achieves scalable performance in both mappings tested on different parallel platforms.