Internship proposal: Contributions to the Pyccel Open Source Projects

A. Ratnani[1], I. Kissami[1], N. Ouhaddou[1], N. Hamid [1]
[1] Mohammed VI Polytechnic University

Information

- Supervisor : Dr. Ahmed Ratnani
- Co-Supervisor : Imad Kissami
- Tutors : Nouredine Ouhaddou, Noureddine Hamid.
- Host laboratory : MSDA / Al-Khwarizmi
- Host Team : Distributed and High Performance Computing
- Host University : Mohammed VI Polytechnic University, Lot 660, Hay Moulay Rachid, Benguerir 43150, Morocco
- Number of Interns: 2
- Compensation: 3000 DH
- Contact mail: Simlab-Recruite@um6p.onmicrosoft.com

General Context

A challenging problem in modern computing is how to ensure scientific codes portability so that they can benefit from the current and future High Performance Computing (HPC) resources while increasing their maintainability. The challenge became apparent with the advent of heterogeneous architectures offering massive amounts of multithreading and computing resources as well as the different architectures (CPUs, GPUs, FPGA, . . .).

On the other hand, the Python language has gained significant popularity as a language for scientific computing and data science, mainly because it is easy to learn and provides many scientific libraries, including parallel ones. However, the Python interpreter/compiler does not perform any optimization and is not meant for writing fast codes. Therefore, different approaches have been proposed to accelerate computation-intensive parts of Python codes.

In order to simplify the process of going from a prototype to a production code, it is important that the *compiler* allows the use of legacy codes or some Python scientific libraries such as numpy, scipy (including Blas, Lapack and FFTW), mpi4py, etc . . . Moreover, this should be presented in a user-friendly way while ensuring that the new compiler is not difficult to maintain; it is then necessary to rely on some well established Python libraries.

For this purpose, Pyccel was designed to be used in two cases: (1) accelerate Python code by converting it to one of C or Fortran and wrapping it into Python, (2) generate portable HPC Fortran codes from a DSL using the Python syntax.

The aim of *Pyccel* is to allow computational scientists to get a low level Fortran/C code from a (valid) Python code. The input Python code is converted into an Abstract Representation (AST) for which we provide the associated Fortran/C printers. In addition, Pyccel provides simple ways to use legacy codes and third party Python libraries such as numpy, scipy and mpi4py. Extending the covered libraries is not difficult; following the idea of Haskell's [**?**] design, extending Pyccel can be done by using the Pyccel language, which has the syntax of Python; Pyccel language is then an extension of Python.

Junior Software developer

We are looking for a high motivated software developer to join Pyccel team. You will be working in an open and agile environment by leveraging the power of open-source projects as well as the power of a large organization's cross-team work.

What You'll Do

- You will work with open-source community
- Develop efficient code, participate in code reviews, creating tests and documentation
- Develop and design new features based off detailed technical designs
- Collaborate with team members to plan new features and analyze requirements.

Required Skills

- Proficiency in the use of Python and C.
- Self-starter and be ready to take on challenges and readiness to dig into old code.
- Knowledge of coding standards and best practices.
- Experience with version control systems like git.

Bonus Skills

- Experience in parallel programming for CPU and/or GPU.
- Knowledge of parallel programming frameworks (OpenMP, OpenACC, MPI ...)
- Experience in performance optimization
- Experience in Fortran
- Experience with Agile, or Scrum methodologies for software development.