Algorithm Engineering Lab Assignment 11

Brian Zahoransky (brian.zahoransky@uni-jena.de)

February 16, 2021

1. What is Cython?

Cython is applied to run Python code faster. It is a compiled language and allows to use C/C++ code and libraries to speed up a program. Most Python code is also valid in Cython. Thus, Cython may be seen as a superset of Python.

2. Describe an approach how Python programs can be accelerated with the help of Cython.

The acceleration process includes the following steps.

First, bottlenecks need to be located. Therefore, the programmer may use software such as pyinstruments or cProfile. Pyinstruments shows the time of the program parts and also marks bottleneck functions red.

Second, the bottleneck function is outsourced in a .pyx Cython-file. Now, a Cython-module can be compiled and imported into the Python program. Running the Python program should deliver the same results as before, maybe a first acceleration can be seen. The preparation is completed, and the actual optimization can start.

Third, the developer should start with defining types for all variables and recompile the module afterwards.

Fourth, the programmer should check if the program runs fast enough. If not, the program should be analyzed again with pyinstruments. When the bottleneck is still the same function, the programmer may optimize the algorithm and repeat step four. In case that another bottleneck appears, the developer can proceed with the second step.

3. Describe two ways for compiling a .pyx Cython module.

First, the Cython module can be compiled **before** used in a Python program. The Cythonize command creates a C/C++ file from a .pyx file. It then compiles the C/C++ file into a module which is importable to Python. The Cython command creates only the C/C++ file. Thus, the programmer has to build an importable Python module.

Second, the Cython module can be compiled **during the run-time**. The easiest way to achieve this is importing the pyximport module, followed by invoking its install() routine. Now, a .pyx file can be imported as a Cython module. Note that the file extension has to be removed. The Python interpreter checks if the Cython module needs to be recompiled due to changes since the last compilation, whenever the program is interpreted.

However, if the programmer likes to have more control over the compilation process, he/she may use the modules "setuptools" and "Cython.Build". Further information about this way, can be found on the website, mentioned in the lecture. (`https://Cython.readthedocs.io/en/latest/src/userguide/source_files_and_compilation.html`) The website also explains how to include Cython in jupyter notebooks.

4. Name and describe two compiler directives in Cython.

There are several compiler directives for the Cython compiler. Directives can be set by writing "# **Cython: <directive>**" at the beginning of a .pyx file. Replacing <**directive**> by "language_level = 3" defines the portable Python version as Python 3. Adding the directive "cdivision = true" disables the check if a divisor is equal to null. That may speed up the program.

There are are also directives for the Cythonize compiler. Two of them are the "-h" and the "j" flag. The "-h" directive shows a complete list of all provided options while the "-j" flag enables building the module with multiple threads.

5. What is the difference between def, cdef and cpdef when declaring a Cython function?

The signal word "def" defines a Cython function as a Python function. That means a Python program can call the function. On the other hand, a "cdef" procedure is Cython only. Thus, it can just be invoked within the Cython code. It is faster than a "def" function since it has a defined output type. A "cpdef" function can be used by both Python and Cython. As a "cdef" function, its output type is pre-defined. When used by a Cython module, it is faster than a "def" but slower than a "cdef" function.

6. What are typed memory-views especially useful for in Cython?

A typed memory-view means transforming a Python or NumPy array into a typed Cython array. This cast often occurs during the parameter transfer between a Python program and the Cython module. In other words, the Python program invokes a function of a Cython module giving an untyped array while the input array of the Cython function is a typed array. That is especially useful in image processing or other applications which work on big matrices.