GPU computing and R

Willem Ligtenberg[1,⋆]

1. OpenAnalytics BVBA
⋆Contact author: willem.ligtenberg@openanalytics.eu

Keywords: R, GPU, OpenCL, High-performance computing

Modern CPUs currently have 4 to 8 cores. In contrast an affordable GPU (200 euro) has 384 cores. This makes them ideally suited for parallel processing. Examples of problems that can benefit from processing on the GPU are Fast Fourier Transforms, the k-nearest neighbour algorithm or sequence alignment. If the problem can be cut into small pieces, it might be interesting to port it to the GPU.

In order to make this compute power more accessible there a few options. One is to make some of the basic *R* functionality use the GPU directly, a second option is to wrap *C(++)* applications that make use of the GPU with *R*. Finally, we can build an interface between *R* and *OpenCL*, which leaves all possibilities open for the user and therefore provides maximum flexibility.

In the first part of our presentation we will provide an overview of current *R* packages that use the GPU. In the second part, we will present our progress on the **ROpenCL** package. **ROpenCL** provides functions to transfer data frames to and from the GPU and to push kernel functions to the GPU, thereby unlocking the full potential of the GPU. These kernel functions still need to be written in *OpenCL*, however the interfacing with the GPU can all be done with *R*. The **ROpenCL** package is similar in spirit and based on the PyOpenCL *Python* library Klöckner (Klöckner).

References

Klöckner, A. PyOpenCL. http://mathema.tician.de/software/pyopencl.