MPGOS: A modular and general-purpose program package to solve a large number of independent ODE systems

Ferenc Hegedűs[1]

[1] Department of Hydrodynamic Systems, Budapest University of Technology and Economics, Budapest, Hungary

In many fields of science, the governing equations of a physical problem can be described by an ordinary differential equation system (ODEs). The application of the massively parallel environment of GPUs on such systems is not trivial, since an initial value problem of an ODE is serial in nature. However, if large number of parameters or initial conditions are involved during a research study, the overall problem can be decomposed into a large number of decoupled sub-problems where each parameter combination and/or initial conditions represent an individual and independent task. Therefore, the whole computational problem became suitable for professional GPGPUs using the SIMT (single instruction multiple thread) paradigm.

The main aim of the present study is to introduce a modular and general-purpose program package capable to handle the aforementioned problem. The name of the package is Massively Parallel GPU-ODE Solver (MPGOS) and it is free to use under an MIT license, for details see the official website *www.gpuode.com*. The parallelization strategy is very simple: a GPU thread is assigned to each system having different parameters and/or initial conditions (the individual systems themselves are independent). The program package is written in C++ and CUDA C. Event handling is incorporated to detect special points of a trajectory. As a special feature, the user can specify parameters shared among each system (e.g. a differentiation matrix originated from the discretization of a PDE), which are automatically loaded into the shared memory of the GPU. Moreover, user-programmable parameters can also be defined to store the special features of the solutions. They can be updated after every successful event detection or time steps by the user through pre-declared device functions (the control flow is absolutely under the control of the user). In this way, for instance, non-smooth dynamics can be efficiently handled. In the first version of the code, only explicit solvers are implemented: the 4th order explicit and adaptive Runge–Kutta–Cash–Karp method with 5th embedded error estimation, and the well-known 4th order explicit Runge–Kutta technique with fixed time step.

One of the main strengths of the package is the user friendly framework that automatically manages the allocation of the data structure and the data transfer between the CPU side and the GPU side. Thus, it can almost completely hide GPU programming form the user. In case of multi-GPU systems, the distribution of the task to different GPUs can also be easily done.