Domain Decomposition and Memory Footprint Reduction of an Eikonal Solver

Daniel Ganellari[1] Gundolf Haase[2] Gerhard Zumbusch[3]

The basis equations in cardiac electrophysiology are the bidomain equations describing the intercellular and the extracellular electrical potential via a system of two PDEs coupled nonlinearly by a bunch of ODEs. Its difference, the transmembrane potential, is responsible for the excitation of the heart and its steepest gradients form an excitation wavefront propagating in time.

This arrival time $\varphi(x)$ of the wavefront at some point $x \in \Omega$ can be approximated by the Eikonal equation [1]

$$\sqrt{\left(\nabla\varphi(x)\right)^T M(x)\nabla\varphi(x)} = 1 \qquad x \in \Omega$$

with given heterogeneous, anisotropic velocity information M. The domain $\Omega \subset \mathbb{R}^3$ is discretized by planar-sided tetrahedrons with a piecewise linear approximation of the solution $\varphi(x)$ inside each of them. The numerical solution of the Eikonal equation follows the fast iterative method [2] with its application for tetrahedral meshes [3]. Therein the main operations in each discretization element τ contain various inner products in the M-metric as $\langle \vec{e}_{k,s}, \vec{e}_{s,\ell} \rangle_{M^\tau} \equiv \vec{e}^{\,T}_{k,s} \cdot M^\tau \cdot \vec{e}_{s,\ell}$ with $\vec{e}_{s,\ell}$ as connecting edge between vertices s and ℓ in element τ. While the authors of [3] pass all coordinates of the tetrahedron together with the 6 entries of M^τ we precompute these inner products and use only them in the wave front computation. This first change requires less memory transfers for each tetrahedron.

The second change is caused by the fact that $\langle \vec{e}_{k,s}, \vec{e}_{s,\ell} \rangle_{M^\tau}$ $(k \neq \ell)$ represents an angle of a surface triangle whereas $\langle \vec{e}_{k,s}, \vec{e}_{k,s} \rangle_{M^\tau}$ represents the length of an edge in the M-metric. Basic geometry as well as vector arithmetics yield to the conclusion that the angle information can be expressed by the combination of three edge lengths. Therefore we only have to precompute the 6 edge lengths of a tetrahedron and compute the remaining 12 angle data on-the-fly which reduces the memory footprint per tetrahedron to 6 numbers.

The efficient implementation of the two changes requires a local Gray-code numbering of edges in the tetrahedron and a bunch of bit shifts to assign the appropriate data. Numerical experiments on CPUs and GPUs show that the reduced memory footprint approach is faster by 40% than the original implementation.

Additionally, we will present our very recent domain decomposition algorithm for the Eikonal equation. For large scale problems, the task based parallel model will run into difficulties: There might be not enough (shared) memory on a single host or on a GPU, the computing power of a single compute unit is not sufficient, or the parallel efficiency

[1] Karl Franzens University of Graz, Institute for Mathematics and Scientific Computing, 8010, Graz, Austria, daniel.ganellari@uni-graz.at

[2] Karl Franzens University of Graz, Institute for Mathematics and Scientific Computing, 8010 , Graz, , Austria, gundolf.haase@uni-graz.at

[3] Friedrich-Schiller-Universität Jena, Institut für Angewandte Mathematik, 07743 Jena, Germany, gerhard.zumbusch@uni-jena.de

is not satisfactory. In all cases, a distributed memory model is needed. Hence a coarser decomposition of the algorithm is needed, namely a domain decomposition approach.

The domain Ω is statically partitioned into a number of non-overlapping sub-domains Ω_i. Each of them is assigned to a single processor. Synchronization and communication of the processors is to be reduced to a minimum. In our case, a single processor i can efficiently solve the Eikonal equation on Ω_i, as long as its boundary data on $\partial\Omega_i$ is correct. However, this data may belong to the outer boundary $\partial\Omega$ or to other processors. Hence inter-processor communication is needed.

We present two different strategies on load balancing in CUDA in order to achieve to run the domain decomposition approach in one GPU. The first approach maps simply one sub-domain to one thread block. Its scaling improves with an increased number of sub-domains by reducing the overall host synchronization together with the preallocation of the global memory. The second approach takes better advantage of the GPU shared memory since it shares the workload of one sub-domain between many thread blocks exploiting in this way the total shared memory space. This allows to overcome the shared memory limitation with sufficient sub-domains which improves the performance significantly. This works very well if enough GPU memory is available. Otherwise we have to preallocate data for each block in each iteration which drops the performance significantly with increased number of sub-domains. This GPU memory limitations can be relaxed by allowing memory allocations only by the active blocks computing for one active sub-domain in the wave front. Again this preallocation is performed in each iteration but only from those blocks who are currently run on one SM. As soon as the blocks finish their execution the memory is freed and ready to be used by other active blocks waiting to be distributed on the idle SMs.

The domain decomposition approach is the first step towards the inter-process communication implementation where the limitation of the global memory will be overcome completely by using multiple accelerator cards and cluster computing. As a future work, it will allow the preallocation of global memory which will enable the scalability on large scale problems.

Supported in part by the FWF project F32-N18 and the JoinEU-SEE scholarship.

References:

[1] J. P. Keener, An Eikonal-curvature equation for action potential propagation in myocardium, J. Math. Biol., 29 (1991), pp. 629-651.

[2] W.-K. Jeong and R. T. Whitaker, A fast iterative method for eikonal equations, SIAM J. Sci. Comput., 30(5), pp. 2512-2534, 2008.

[3] Z. Fu , R. M. Kirby and R. T. Whitaker, A fast iterative method for solving the eikonal equation on tetrahedra domains, Sci. Comput. 35(5), pp. C473-C494, 2013.