**Motivation:** Propagating interfaces occur in wide variety of fields, including fluid mechanics, optimal path planning, computer vision and graphics. The Eikonal equation can be used to compute the distance field over a complex terrain region. This information is used in physical parameterization of the atmospheric boundary layer. For complex terrain represented with a fine resolution serial computations can be time consuming and may not fit to the device memory of modern GPUs. Although fast algorithms are available to compute the Eikonal equations, their parallel implementation on modern GPU clusters are not straightforward.

**Objective:** Implement an efficient parallel distance field solver for GPU accelerator platforms and perform software verification for concurrent programs with a mixed (e.g. MPI-CUDA/OpenACC) implementation. Efficiency of the solver depends on both the performance and the amount of development effort spent on writing parallel code. The resulting software element will serve as the blueprint for the entire suite of software elements.

**Fast Sweeping Method (FSM)**
- Zhao proposed the Fast Sweeping Method, with an algorithmic complexity of O(n) for n unknown points.
- Solves the Eikonal equation using Gauss-Seidel iterations in alternate directions and upwind difference scheme to calculate the shortest distance to any geometry.
- Propagates the initial distance values in the grid to other points. Initial distance field is calculated using the algorithm described in Senocak et al. (2015)

\[ \lVert \nabla \phi \rVert = 1 \]

- Converges in \(2^n\) sweeps in \(\mathbb{R}^n\).
  - \(\mathbb{R}^2 = 2^2 = 4\) (2D)
  - \(\mathbb{R}^3 = 2^3 = 8\) (3D)

**Parallelization of FSM**
- Zhao (2007)
  - Simultaneous sweep
  - Reduce minimum
  - Update solution
  - Pros
    - Simple to implement
  - Cons
    - Requires more memory
    - Limited threads (4 in 2D and 8 in 3D)
    - GPU parallelization not feasible due to thread limitations

- Detrixhe, Gibou, Min (2013)
  - Cuthill-McKee type ordering
  - Define node levels as sum of coordinates
  - Update nodes on each level simultaneously
  - Pros
    - Uses newly computed values for update
    - Same convergence as serial
  - Cons
    - Not so straightforward implementation

**Discussion:** The performance of the parallel solver implemented using the algorithm described in Detrixhe et al (2013) is suited for GPU architecture and scales well with the domain size. GPU parallelization with CUDA for NVIDIA devices reaches speed-up of up to 35x the serial implementation. However, advanced features of CUDA has to be used for performance. OpenACC provides easy to use APIs that boost performance of the code by adding few lines, making it a better choice for a sustainable software infrastructure.

**Concurrent Software Verification**
- Goal is to verify hybrid parallel software (e.g. MPI-CUDA/OpenACC)
  - CIVL: verification tool for hybrid programs
  - MPI, OpenMP, CUDA
  - C++ and OpenACC support not available yet
  - Converts program to its own concurrent language CIVL-C
  - Symbolic execution of the translated program
  - Reports concurrency errors

**Acknowledgment:** This material is based upon work supported by the National Science Foundation under grant 1440638.

Project url: [https://github.com/GEM3D](https://github.com/GEM3D)

Contact: senocak@boisestate.edu