Task-based parallelization of a transport Discontinuous Galerkin solver

Applications to fluid models

Philippe Helluy

Inria ToNuS, IRMA Strasbourg

DFG-CNRS Workshop, Paris, 2 February 2016

Outlines

Kinetic schemes

High order implicit kinetic scheme

Implicit DG solver for transport

1) Kinetic Schemes

Why is it important to solve transport equations ?

Kinetic framework

- Distribution function: $f(x, v, t)$, $x \in \mathbb{R}^d$, $v \in \mathbb{R}^d$, $t \in [0, T]$.
- Microscopic "collision vector" $K(v) \in \mathbb{R}^m$. Macroscopic conserved data

$$w(x,t) = \int_v f(x,v,t) K(v) dv.$$

- Microscopic entropy s and associated Maxwellian $M_w(v)$:

$$\int_v M_w K = w, \quad \int_v s(M_w) = \min_{\int_v fK = w} \left\{ \int_v s(f) \right\}.$$

- Kinetic-BGK equation ($a = a(x,t)$ is the acceleration):

$$\partial_t f + v \cdot \nabla_x f + a \cdot \nabla_v f = \eta \left(M_w - f \right).$$

Kinetic schemes

When the relaxation parameter η is big, the kinetic equation provides an approximation of the hyperbolic conservative system

$$\partial_t w + \nabla \cdot F(w) + \Pi(w) = 0,$$

with

$$F^i(w) = \int_v v^i M_w(v) K(v) dv.$$

$$\Pi(w) = a \cdot \int_v \nabla_v M_w(v) K(v) = -a \cdot \int_v M_w(v) \nabla_v K(v).$$

Main idea: numerical solvers for the linear scalar transport equation lead to natural solvers for the non-linear hyberbolic system [Deshpande, 1986]. Micro or macro approach.

Euler equations

The Maxwellian M_w has not necessarily a physical meaning.
Famous example [Perthame, 1990]

$$w = \begin{pmatrix} \rho \\ \rho u \\ \rho e + \rho u^2/2 \end{pmatrix}, \quad F(w) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ (\rho e + \rho u^2/2 + p)\, u \end{pmatrix}, \quad p = 2\rho e.$$

It is possible to find a convex entropy and a kinetic interpretation with

$$a = 0, \quad K(v) = \begin{pmatrix} 1 \\ v \\ v^2/2 \end{pmatrix}, \quad M_w(v) = \frac{\rho}{2\sqrt{6e}} \chi_{[-1,1]} \left(\frac{v-u}{\sqrt{6e}} \right),$$

where $\chi_{[-1,1]}$ is the indicator function of $[-1,1]$.

Isothermal flow

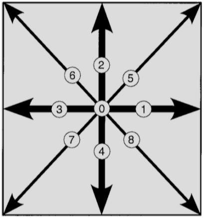
We would like to solve a transport equation for each v. How to reduce as much as possible the velocity space ?
Answer: lattice Boltzmann. Example for an isothermal inviscid fluid.

- Density ρ, velocity u
- $w = (\rho, u^T)^T$
- pressure $p = c^2\rho$ (c is the sound speed)

$$\partial_t \rho + \nabla \cdot (\rho u) = 0,$$
$$\partial_t(\rho u) + \nabla \cdot (\rho u \otimes u + pI) = 0.$$

Lattice kinetic interpretation

Lattice kinetic interpretation [Qian et al., 1992] under a low Mach hypothesis:



- In 2D, $N = 9$, $v \in \{v_0 \dots v_{N-1}\}$.
- $\int_v = \sum_{i=0}^{N-1}$
- $w = (\rho, \rho u)$, $K(v) = (1, v^T)^T$, $a = 0$,

$$M_w(v_i) = \rho \omega_i \left(1 + \frac{v_i \cdot u}{\theta} + \frac{(v_i \cdot u)^2}{\theta} - \frac{u^2}{2\theta}\right),$$

$$\theta = 1/3, \quad \omega_0 = 4/9, \quad \omega_{1-4} = 1/9, \quad \omega_{5-8} = 1/36.$$

2) High order implicit kinetic scheme

CFL condition is not a fatality

One-dimensional lattice kinetic interpretation

- Lattice $v \in \{-1, 0, 1\}$.
- $w = (\rho, u)^T$.
- $f(x, v, t)$ is solution of

$$\partial_t f + v \partial_x f = \eta (M_w - f).$$

- $\rho = \int_v f = f(\cdot, -1, \cdot) + f(\cdot, 0, \cdot) + f(\cdot, 1, \cdot)$,
  $\rho u = \int_v fv = f(\cdot, 1, \cdot) - f(\cdot, -1, \cdot)$,
  $\rho u^2 + c^2 \rho = \int_v fv^2 = f(\cdot, 1, \cdot) + f(\cdot, -1, \cdot)$.
- sound speed c.
- $M_w(\pm 1) = \rho u (u \pm 1)/2 + c^2 \rho / 2$,
  $M_w(0) = \rho (1 - u^2 - c^2)$.
- Validity: $1/2 < c < 1$, $|u| < \sqrt{1 - c^2}$.

First order splitting algorithm

For each time step of duration Δt,

- free transport: solve

$$\partial_t f + v \partial_x f = 0;$$

- relaxation: compute $w = (\rho, u)^T$ and return to

$$f(x, v, t) = M_{w(x,t)}(v).$$

The resulting scheme is $O(\Delta t)$ with high numerical viscosity

Second order extension

Spatial approximation leads to a differential equation

$$v'(t) = g(v(t)), \quad v(0) = v_0.$$

First order numerical method

$$\varphi(\Delta t)v_0 = v_0 + g(v_0)\Delta t + E(v_0)\Delta t^2 + O(\Delta t^3).$$

More precise method $\psi(\Delta t) = \varphi(\alpha\Delta t)\varphi(\beta\Delta t)$.
Second order is attained iff

$$\alpha + \beta = 1, \quad \alpha\beta = \frac{1}{2}, \quad \alpha^2 + \beta^2 = 0.$$

Small miracle: $\alpha = \frac{1+i}{2}, \quad \beta = \frac{1-i}{2}$ works !

Comments and possible generalization

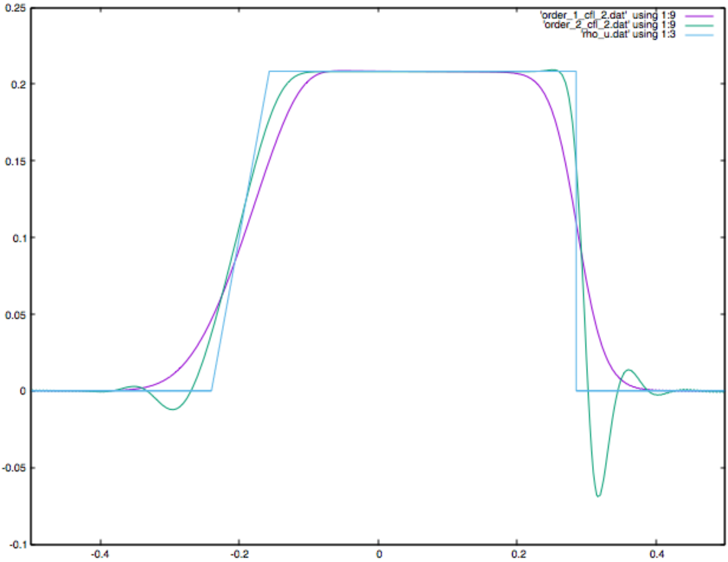
- Bibliography: [Fung, 1998, McLachlan and Quispel, 2002]
- Very easy to program: consider complex time-step Δt. At the end of the time-step change Δt to $\overline{\Delta t}$.
- Transport: each sub-step is unstable. The global step is stable.
- Complexity: storage$\times 2$, CPU time approximately$\times 2.5$.
- Possible generalization to higher orders. Rely on the BCH formula in the Lie algebra of vector fields

$$e^Z = e^X e^Y, \quad Z = X + Y + \frac{1}{2}[X,Y] + \cdots$$

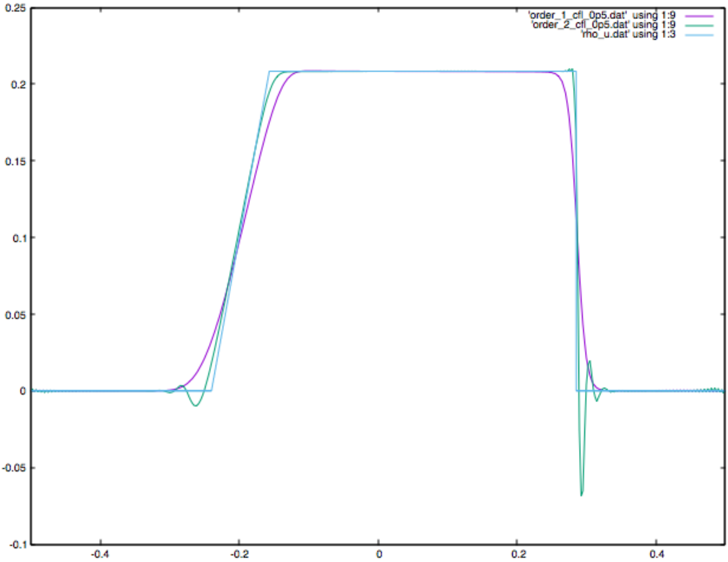
Numerical results

- Riemann problem $\rho_L = 2$, $\rho_R = 1$, $u_L = u_R = 0$.
- Implicit Continuous Galerkin scheme with second order Lagrange interpolation.
- 200 finite elements (401 nodes).
- Compare first and second order time integration for various CFL.

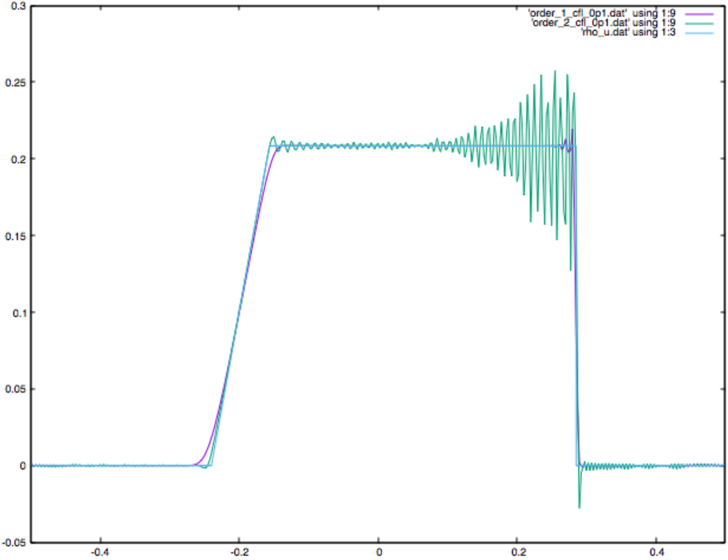
velocity, CFL=2



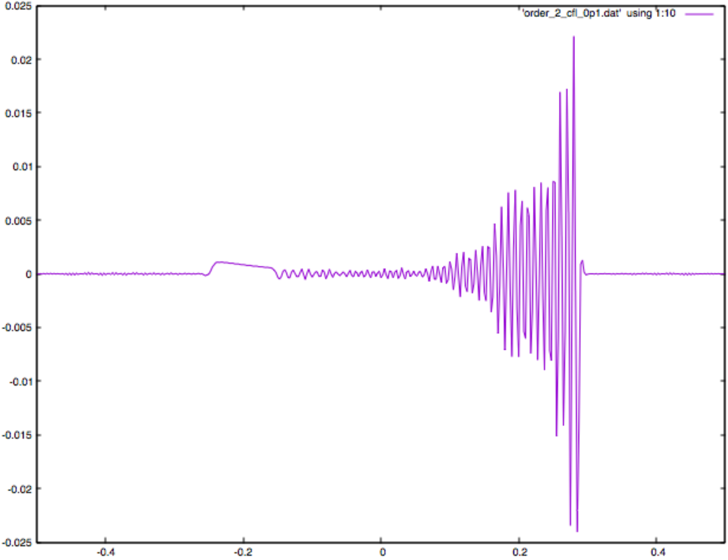
velocity, CFL=0.5



velocity, CFL=0.1



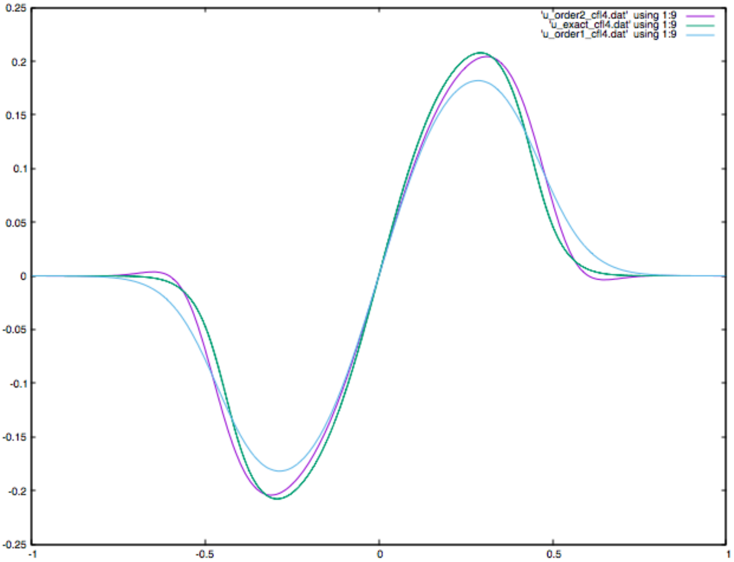
velocity, imaginary part, CFL=0.1



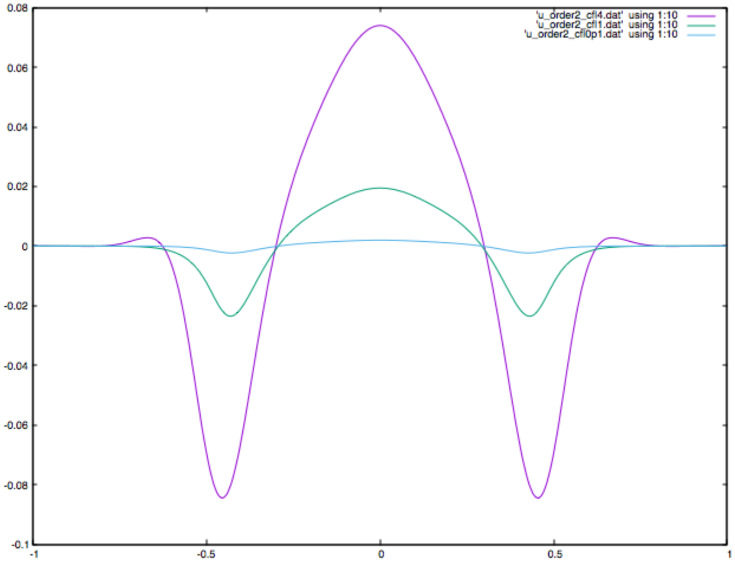
Smooth solution

- Smooth initial data.
- Implicit Continuous Galerkin scheme with second order Lagrange interpolation.
- 200 finite elements (401 nodes).
- Compare first and second order time integration for various CFL.

velocity, CFL$= 4$



velocity, imaginary part



3) Implicit DG solver for transport

Explicit Discontinuous Galerkin (DG) are constrained by an annoying CFL condition. Empirical stability condition

$$\Delta t \leq \frac{\Delta x}{2d(2p+1)V_{\max}}$$

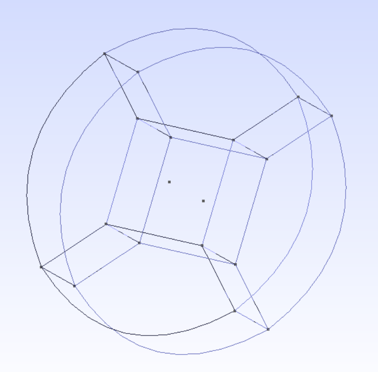
with:

- Δx: cell size.
- d: space dimension
- p : polynomial degree
- $V_{\max}$: maximal speed
- Can be worse...

Macromesh approach

We consider a coarse mesh made of hexahedral curved macrocells

- Each macrocell is itself split into smaller subcells of size h.
- In each subcell L we consider polynomial basis functions ψ_i^L of degree p.
- Possible non-conformity in "h" and "p".
- We need a conservative scheme. We want to avoid CFL condition.

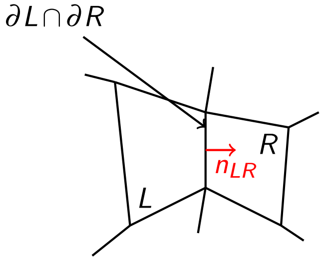


DG approximation of the transport equation

Implicit DG approximation scheme: $\forall L, \forall i$

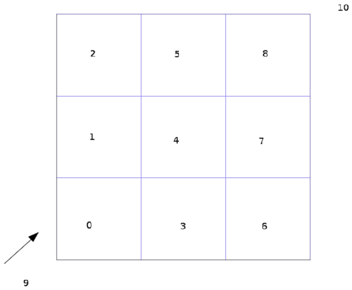
$$\int_L \frac{f_L^n - f_L^{n-1}}{\Delta t} \psi_i^L - \int_L v \cdot \nabla \psi_i^L f_L^n + \int_{\partial L} \left(v \cdot n^+ f_L^n + v \cdot n^- f_R^n \right) \psi_i^L = 0.$$

- R denotes the neighbor cells along ∂L.
- $v \cdot n^+ = \max(v \cdot n, 0)$, $v \cdot n^- = \min(v \cdot n, 0)$.
- n_{LR} is the unit normal vector on ∂L oriented from L to R.



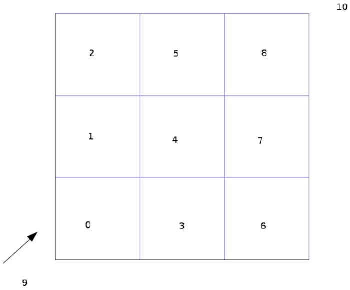
Upwind numbering

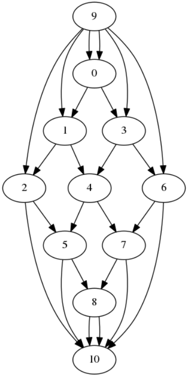
- L is *upwind* with respect to R if $v \cdot n_{LR} > 0$ on $\partial L \cap \partial R$.
- In a macrocell L, the solution depends only on the values of f in the upwind macrocells.
- No assembly and factorization of the global system.



Dependency graph

For a given velocity v we can build a dependency graph. Vertices are associated to macrocells and edges to macrocells interfaces or boundaries. We consider two fictitious additional vertices: the "upwind" vertex and the "downwind" vertex.





Algorithm

Bibliography: [Duff and Reid, 1978, Johnson et al., 1984, Wang and Xu, 1999, Natvig and Lie, 2008]

- Topological ordering of the dependency graph (it has to be a DAG...).
- First time step: Assembly and LU decomposition of the local macrocell matrices.
- For each macrocell (in topological order):
  - Compute volume terms.
  - Compute upwind fluxes.
  - Solve the local linear system.
  - Extract the results to the downwind cells.

Parallelization ?

StarPU parallelization

- StarPU is a library developed at Inria Bordeaux [Augonnet et al., 2012]: `http://starpu.gforge.inria.fr`
- Task-based parallelism.
- Task description: codelets, inputs (R), outputs (W or RW).
- The user submits tasks in a correct sequential order.
- StarPU schedules the tasks in parallel if possible.

StarPU implementation

- We start from a working sequential code
  `http://schnaps.gforge.inria.fr`
- StarPU implementation was smooth: incremental migrations task by task.
- Several implementations of the same task are possible (CPU, optimized CPU, GPU I, GPU II, MIC, etc.)

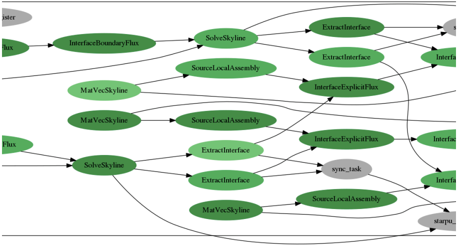
Preliminary results

We compare a global direct solver to the upwind StarPU solver with several meshes.
Weak scaling. "dmda" scheduler. AMD Opteron 16 cores, 2.8 Ghz.
Timing in seconds for 200 iterations.

nb cores	0	1	2	4	8	16
$10 \times 10 \times 8 \times 8$ direct	30	144	-	-	-	-
$10 \times 10 \times 8 \times 8$ upwind	-	32	19	12	7	6
$20 \times 20 \times 4 \times 4$ upwind	-	41	26	17	12	17
$20 \times 20 \times 8 \times 8$ upwind	-	120	72	40	28	20

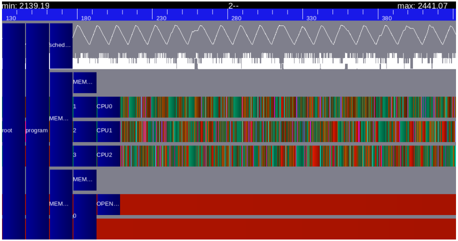
Task graph

Zoom of the task graph generated by StarPU



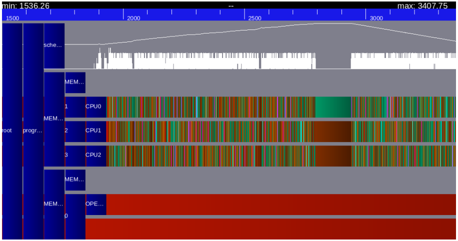
Gantt diagram

Gantt diagram generated by StarPU: sync point at the end of each time step



Gantt diagram

Gantt diagram generated by StarPU: without sync point at the end of each time step



Conclusion

- Kinetic framework: avoiding CFL condition and costly linear solvers.
- Migration of a transport DG solver to StarPU.

TODO list:

- Apply downwind numbering also in macrocells.
- Detect cycles in the upwind graph (Tarjan algorithm).
- Migrate GPU codelets (OpenCL).
- MPI + StarPU.
- Parallelize on several velocities.
- Kinetic schemes, Vlasov.

Bibliography I

[Augonnet et al., 2012] Augonnet, C., Aumage, O., Furmento, N., Namyst, R., and Thibault, S. (2012).
StarPU-MPI: Task Programming over Clusters of Machines Enhanced with Accelerators.
In Jesper Larsson Träff, S. B. and Dongarra, J., editors, *EuroMPI 2012*, volume 7490 of *LNCS*. Springer.
Poster Session.

[Deshpande, 1986] Deshpande, S. (1986).
Kinetic theory based new upwind methods for inviscid compressible flows.
In *24th AIAA Aerospace Sciences Meeting*, volume 1.

[Duff and Reid, 1978] Duff, I. S. and Reid, J. K. (1978).
An implementation of tarjan's algorithm for the block triangularization of a matrix.
ACM Transactions on Mathematical Software (TOMS), 4(2):137–147.

[Fung, 1998] Fung, T. (1998).
Complex-time-step newmark methods with controllable numerical dissipation.
International Journal for numerical methods in Engineering, 41(1):65–93.

[Johnson et al., 1984] Johnson, C., Nävert, U., and Pitkäranta, J. (1984).
Finite element methods for linear hyperbolic problems.
Computer methods in applied mechanics and engineering, 45(1):285–312.

[McLachlan and Quispel, 2002] McLachlan, R. I. and Quispel, G. R. W. (2002).
Splitting methods.
Acta Numerica, 11:341–434.

[Natvig and Lie, 2008] Natvig, J. R. and Lie, K.-A. (2008).
Fast computation of multiphase flow in porous media by implicit discontinuous galerkin schemes with optimal ordering of elements.
Journal of Computational Physics, 227(24):10108–10124.

Bibliography II

[Perthame, 1990] Perthame, B. (1990).
Boltzmann type schemes for gas dynamics and the entropy property.
SIAM Journal on Numerical Analysis, 27(6):1405–1421.

[Qian et al., 1992] Qian, Y., d'Humières, D., and Lallemand, P. (1992).
Lattice bgk models for navier-stokes equation.
EPL (Europhysics Letters), 17(6):479.

[Wang and Xu, 1999] Wang, F. and Xu, J. (1999).
A crosswind block iterative method for convection-dominated problems.
SIAM Journal on Scientific Computing, 21(2):620–645.