Improving the Performance of an AstroPhi Code for Massively Parallel Supercomputers Using Roofline Analysis

I. Kulikov[1], **I. Chernykh**[1], B. Glinskiy[1]
[1]Institute of Computational Mathematics and Mathematical Geophysics SB RAS

Overview

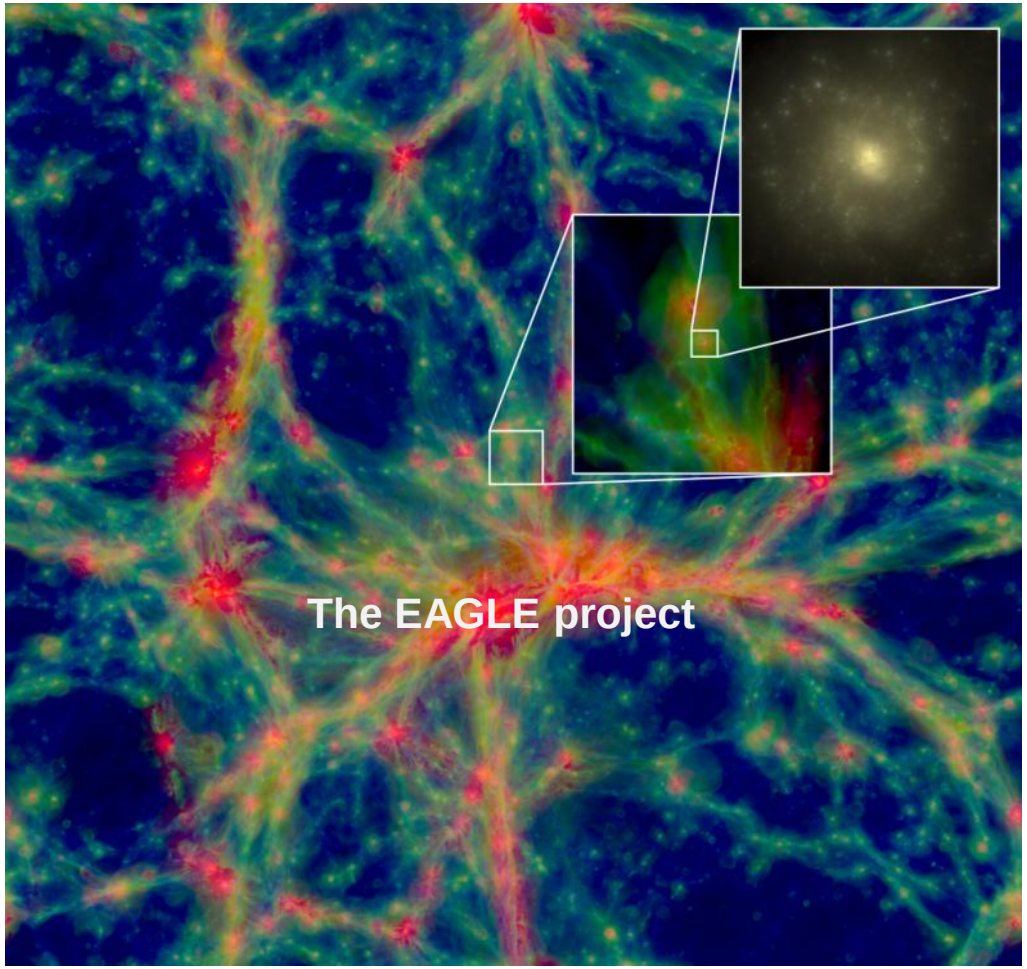
1) Astrophysics
2) Mathematical model
3) Numerical method
4) Parallel realization and co-design
5) Numerical simulation results

Astrophysics

Astrophysics is the branch of astronomy that employs the principles of physics and chemistry "to ascertain the nature of the heavenly bodies, rather than their positions or motions in space.“[1]

[1]Keeler, James E. (November 1897), "The Importance of Astrophysical Research and the Relation of Astrophysics to the Other Physical Sciences", The Astrophysical Journal 6 (4): 271–288

Cosmological simulation



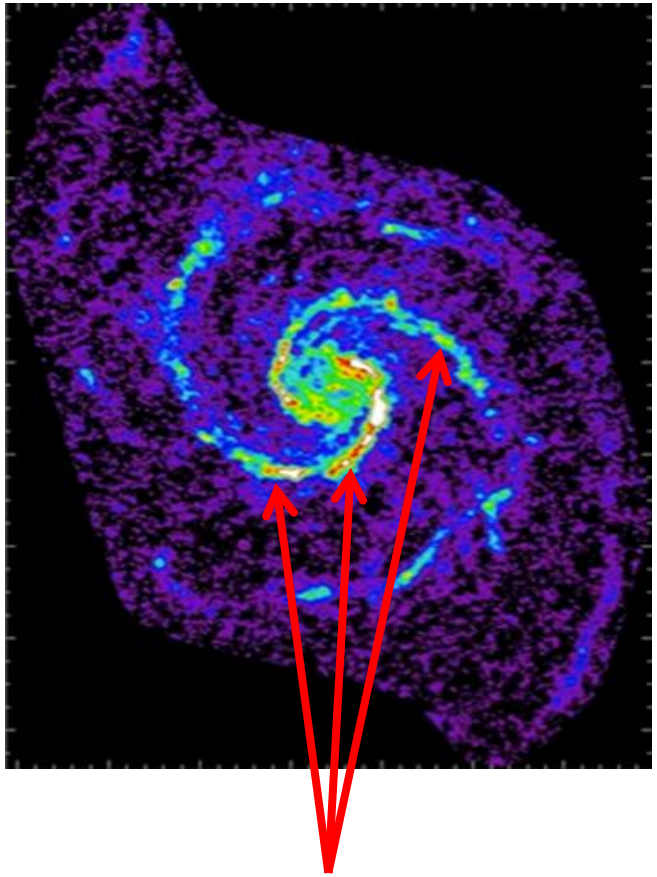
We were inspired by EAGLE project.

EAGLE (Evolution and Assembly of GaLaxies and their Environments) is a simulation aimed at understanding how galaxies form and evolve. This computer calculation models the formation of structures in a cosmological volume, 100 Megaparsecs on a side (over 300 million light-years). This is large enough to contain 10,000 galaxies of the size of the Milky Way or bigger, enabling a comparison with the whole zoo of galaxies visible in the Hubble Deep field for example.

http://icc.dur.ac.uk/Eagle/
Durham University

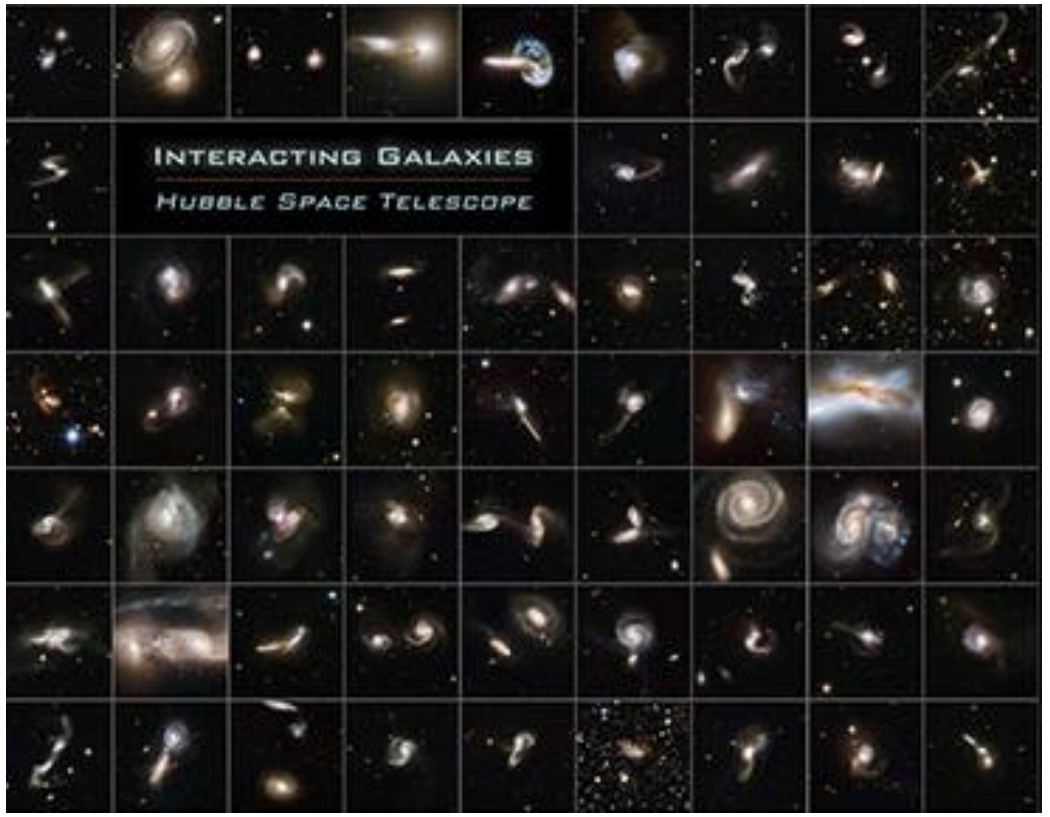
Astrophysics

M 51 Galaxy



Stars formation processes

Interacting galaxies



Mathematical model

Gas component + Collisionless component + Poisson equation + H2 formation

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\vec{u}) = \mathrm{S} - \mathrm{D}$$

$$\frac{\partial \rho_H}{\partial t} + \nabla\cdot(\rho_H\vec{u}) = -s_{H,H_2} + \mathrm{S}\,\frac{\rho_H}{\rho} - \mathrm{D}\,\frac{\rho_H}{\rho}$$

$$\frac{\partial \rho_{H_2}}{\partial t} + \nabla\cdot(\rho_{H_2}\vec{u}) = s_{H,H_2} + \mathrm{S}\,\frac{\rho_{H_2}}{\rho} - \mathrm{D}\,\frac{\rho_{H_2}}{\rho}$$

$$\frac{\partial \rho\vec{u}}{\partial t} + \nabla\cdot(\rho\vec{u}\vec{u}) = -\nabla p - \rho\nabla(\Phi) + \vec{v}\mathrm{S} - \vec{u}\mathrm{D}$$

$$\frac{\partial \rho E}{\partial t} + \nabla\cdot(\rho E\vec{u}) = -\nabla\cdot(p\vec{v}) - (\rho\nabla(\Phi),\vec{u}) - \Lambda + \Gamma - \varepsilon\frac{\mathrm{D}}{\rho}$$

$$\frac{\partial \rho\varepsilon}{\partial t} + \nabla\cdot(\rho\mathrm{T}\vec{u}) = -(\gamma-1)\rho\mathrm{T}\nabla\cdot\vec{u} - \Lambda + \Gamma - \varepsilon\frac{\mathrm{D}}{\rho}$$

$$\rho E = \frac{1}{2}\rho\vec{u}^2 + \rho\varepsilon \qquad p = (\gamma-1)\rho\varepsilon$$

$$\frac{\partial n}{\partial t} + \nabla\cdot(n\vec{v}) = \mathrm{D} - \mathrm{S}$$

$$\frac{\partial n\vec{v}}{\partial t} + \nabla\cdot(n\vec{v}\vec{v}) = -\nabla\Pi - n\nabla(\Phi) + \vec{u}\mathrm{D} - \vec{v}\mathrm{S}$$

$$\frac{\partial \rho W}{\partial t} + \nabla\cdot(\rho W\vec{v}) = -\nabla\cdot(\Pi\vec{v}) - (n\nabla(\Phi),\vec{v}) - \Gamma + \varepsilon\frac{\mathrm{D}}{\rho}$$

$$\frac{\partial \Pi_{\xi\xi}}{\partial t} + \nabla\cdot(\Pi_{\xi\xi}\vec{v}) = -2\Pi\nabla\cdot\vec{u} - \Gamma + \varepsilon\frac{\mathrm{D}}{3\rho}$$

$$\rho W = \frac{1}{2}\rho\vec{v}^2 + \frac{\Pi_{xx}+\Pi_{yy}+\Pi_{zz}}{2}$$

$$\Delta\Phi = 4\pi G(\rho + n)$$

Mathematical model

H_2 formation (*Khoperskov et al., 2013; Glover & Mac Low, 2007*)

$$\frac{dn_{H_2}}{dt} = R_{gr}(T) n_H (n_H + 2n_{H_2}) - \left(\xi_H + \xi_{diss}(N_{H_2}, A_V)\right) n_{H_2}$$

Stars formation (*Katz et al., 1996*)

$$T < 10^4 K \qquad \nabla \cdot \vec{u} < 0 \qquad \rho > 1.64 \frac{M_\odot}{pc^{-3}}$$

$$\mathcal{D} = C\rho^{3/2}\sqrt{\frac{32G}{3\pi}}$$

Supernova + Heating function (*Springel & Hernquist, 2003*)

$$\mathcal{S} = \beta C n^{3/2}\sqrt{\frac{32G}{3\pi}}$$

$$\Gamma = 10^{51}\frac{M^{SN}}{M_\odot} erg$$

Cooling function (*Sutherland & Dopita, 1993*)

$$\Lambda \simeq 10^{-22} n_H^2 cm^{-3} erg$$

Numerical Method

$$\frac{\partial \mathbb{Q}}{\partial t} + \nabla \bullet \left(\mathbb{Q} \cdot \vec{v} \right) = \mathfrak{I}\left(\mathbb{Q}, \nabla \mathbb{Q} \right)$$

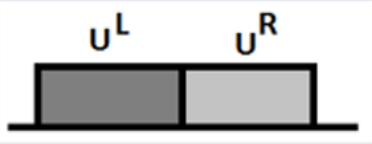
(1)

$$\frac{\partial \mathbb{Q}}{\partial t} = \mathfrak{I}\left(\mathbb{Q}, \nabla \mathbb{Q} \right)$$

(2)

$$\frac{\partial \mathbb{Q}}{\partial t} + \nabla \bullet \left(\mathbb{Q} \cdot \vec{v} \right) = 0$$

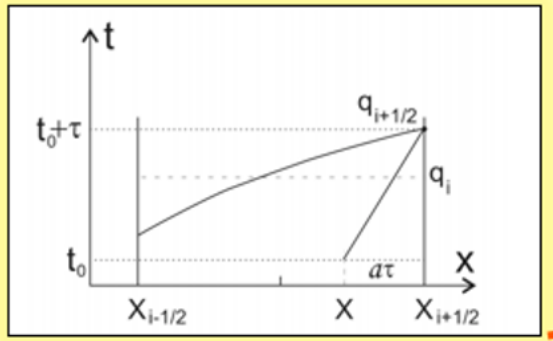
Riemann problem



$$\frac{\partial u}{\partial t} + B\frac{\partial u}{\partial x} = 0 \qquad B = R\Lambda L \qquad LR = I$$

$$L\frac{\partial u}{\partial t} + LR\Lambda L\frac{\partial u}{\partial x} = 0 \qquad w = Lu$$

$$\frac{\partial w}{\partial t} + \Lambda\frac{\partial w}{\partial x} = 0 \qquad w(x,t) = w(x - \Lambda t) \qquad u = Rw$$



piecewise-parabolic functions

(*) Kulikov, et al., LNCS, 2009; APJS, 2011, 2014; AAABS, 2013; CPC 2015; NewA 2016

(**) Ustyugov, Popov, Comp. Math. & Math. Phys., 2007, 2008, Comp. Phys., 2009

Highlights of Numerical method

- High-order method
- The absence of artificial viscosity
- Galilean-invariant solution
- Entropy nondecrease guarantee
- Possible extension of the model by other hyperbolic equations (for example MHD equations)
- Simple parallelization
- Potentially “infinite” scalability

Numerical simulation (method validation)

- **One-dimensional shock tube tests**
- **One-dimensional Aksenov's test with continuous periodic solution**
- **Sedov explosion problem**
- **Two-dimensional Rayleigh-Taylor instability**
- **Two-dimensional Kelvin-Helmholtz instability**
- **Three-dimensional Evrard's collapse**

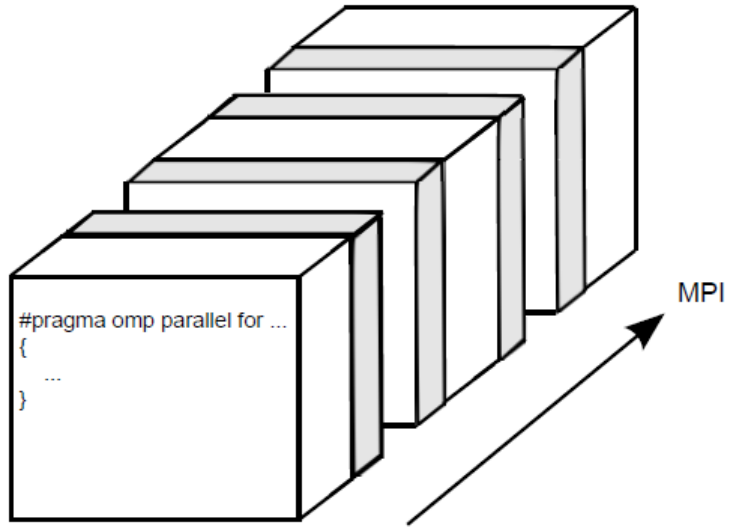
1) I.M. Kulikov, *GPUPEGAS: A new GPU-accelerated hydrodynamic code for numerical simulations of interacting galaxies*, Astrophysical Journal. Supplement Series, vol. 214(12), pp. 1-12, 2014.
2) I.M. Kulikov, I.G. Chernykh, A.V. Snytnikov, B.M. Glinskiy, A.V. Tutukov, *AstroPhi: a code for complex simulation of dynamics of astrophysical objects using hybrid supercomputers*, Comp. Phys. Comm., vol. 186, pp. 71-80, 2015.
3) I. Kulikov, E.Vorobyov, *Using the PPML approach for constructing a low-dissipation, operator-splitting scheme for numerical simulations of hydrodynamic flows*, The Journal of Computational Physics, vol. 317, pp. 318-346, 2016.

Parallel realization

134x speedup on 240 Intel Xeon Phi threads

RSC PetaStream (JSCC RAS)

MICs 64 x Intel Xeon Phi 7120D
Threads 15 360 (64 x 240)



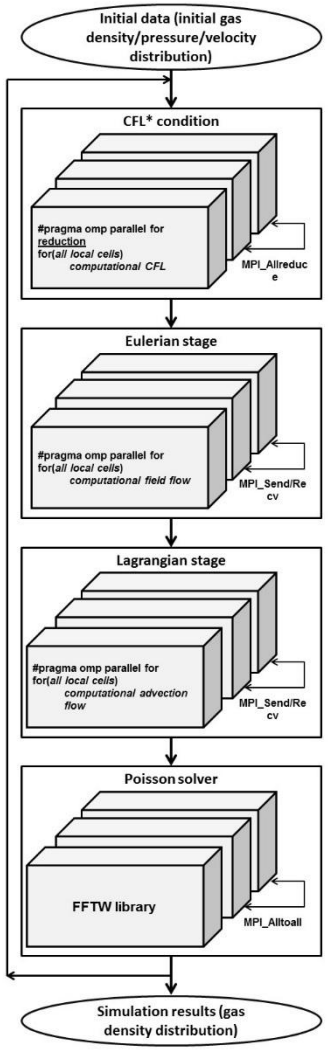
RSC PetaStream (Polytechnic State University)

MICs 256 x Intel Xeon Phi 5120D
Threads 61 440 (256 x 240)

RSC Tornado-F node
Intel Xeon Phi 7250 (KNL)

http://www.rscgroup.ru/en

Parallel realization



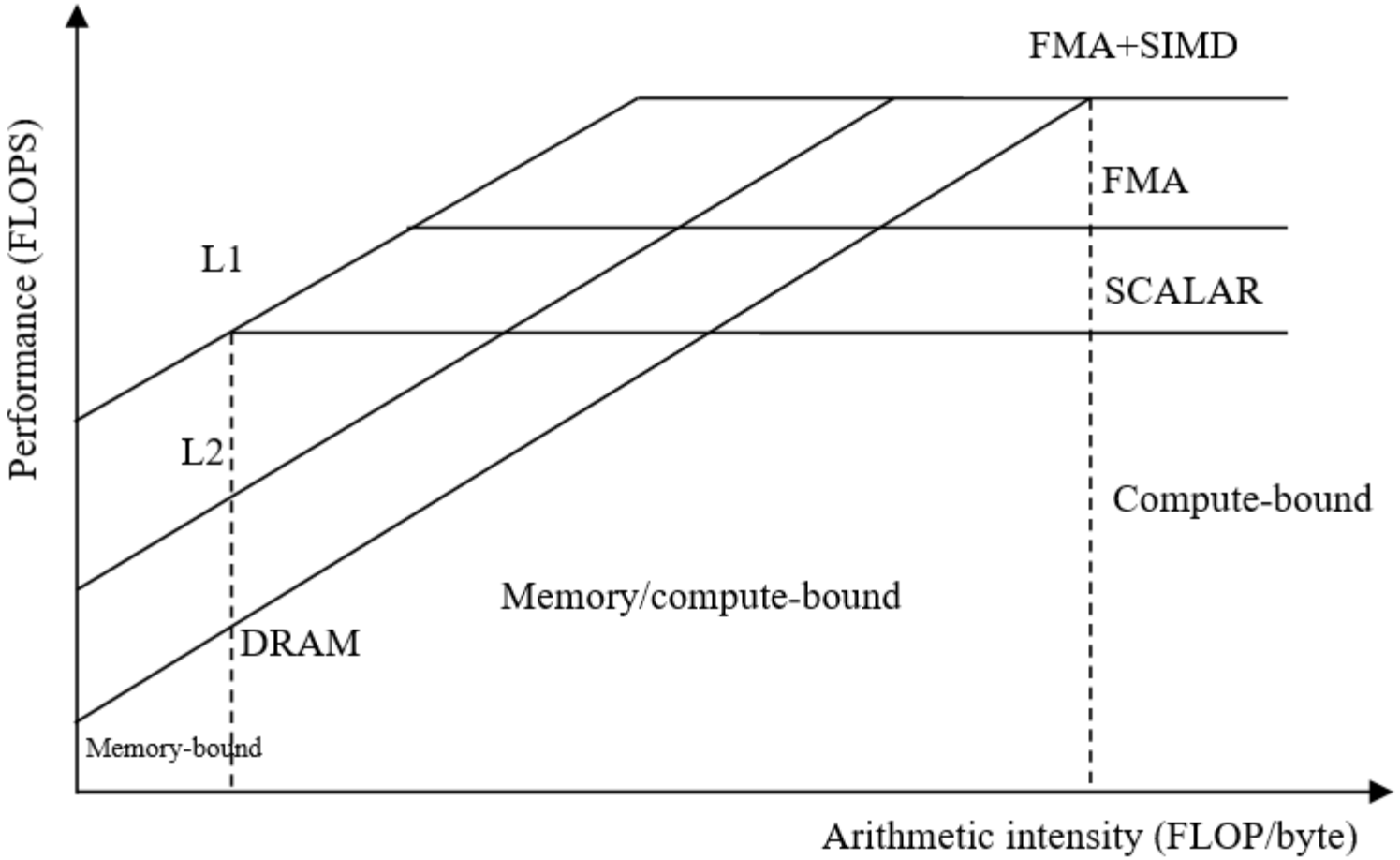
Number of accelerators	Efficiency (MVS-10P)	Efficiency (Polytechnic)
1	1,0000	1,0000
2	1,0345	1,0058
4	1,0335	0,9864
8	0,9771	0,9857
16	0,9417	0,9591
32	0,9345	0,9101
64	0,9235	0,8659
96	—	0,8326
128	—	0,8101
192	—	0,7711
224	—	0,7565

TABLE 2
Efficiency on a different accelerators number for MVS-10P and Polytechnic supercomputers.

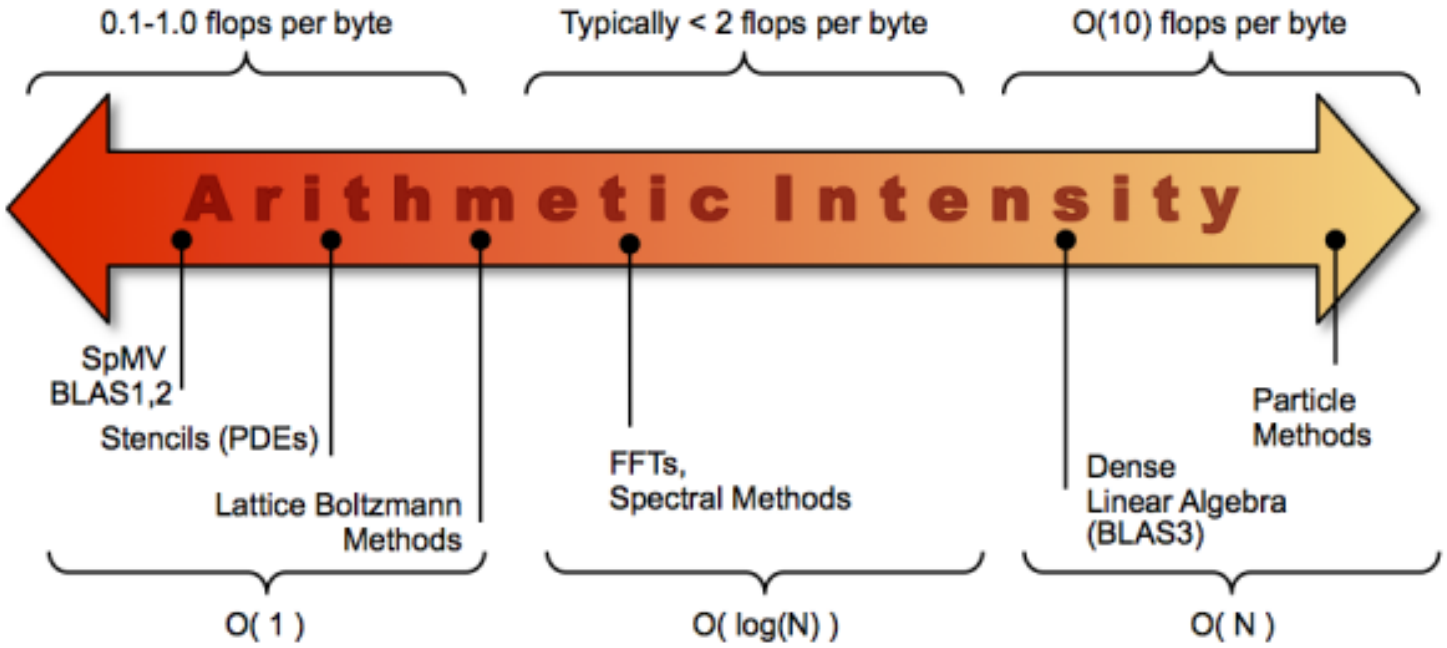
73% efficiency (weak scalability) on 256x Intel Xeon Phi accelerators, 15360 cores (61440 threads) used

Weak scalability is how the solution time varies with the number of processors for a fixed problem size *per processor*

Roofline Analysis



Roofline Analysis



Roofline Performance Model // https://crd.lbl.gov/departments/computer-science/PAR/research/roofline/

Parallel realization

Roofline analysis with using of Intel Advisor consists of 3 steps: survey collection, trip count collection, visualization and/or extraction of the collected data into a report. Before analysis, the application should be compiled in debug mode.

1. Survey collection by command line with advisor:

mpirun -n <number of KNL nodes> advixe-cl -collect survey –-trace-mpi -- ./<app_name>

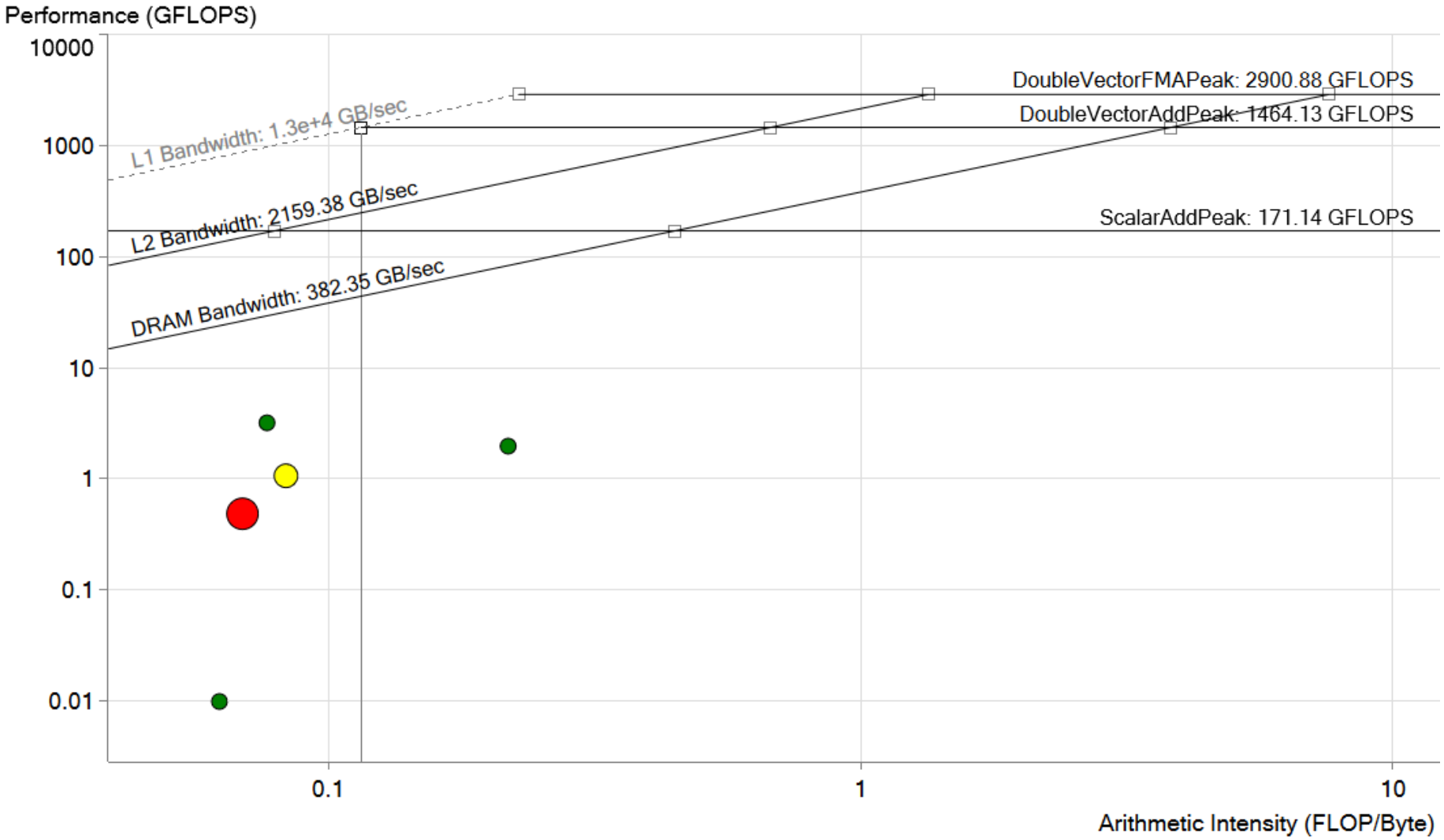
2. Trip count collection by command line with advisor:

mpirun -n <number of KNL nodes> advixe-cl -collect tripcounts -flops-and-masks –-trace-mpi -- ./<app_name>

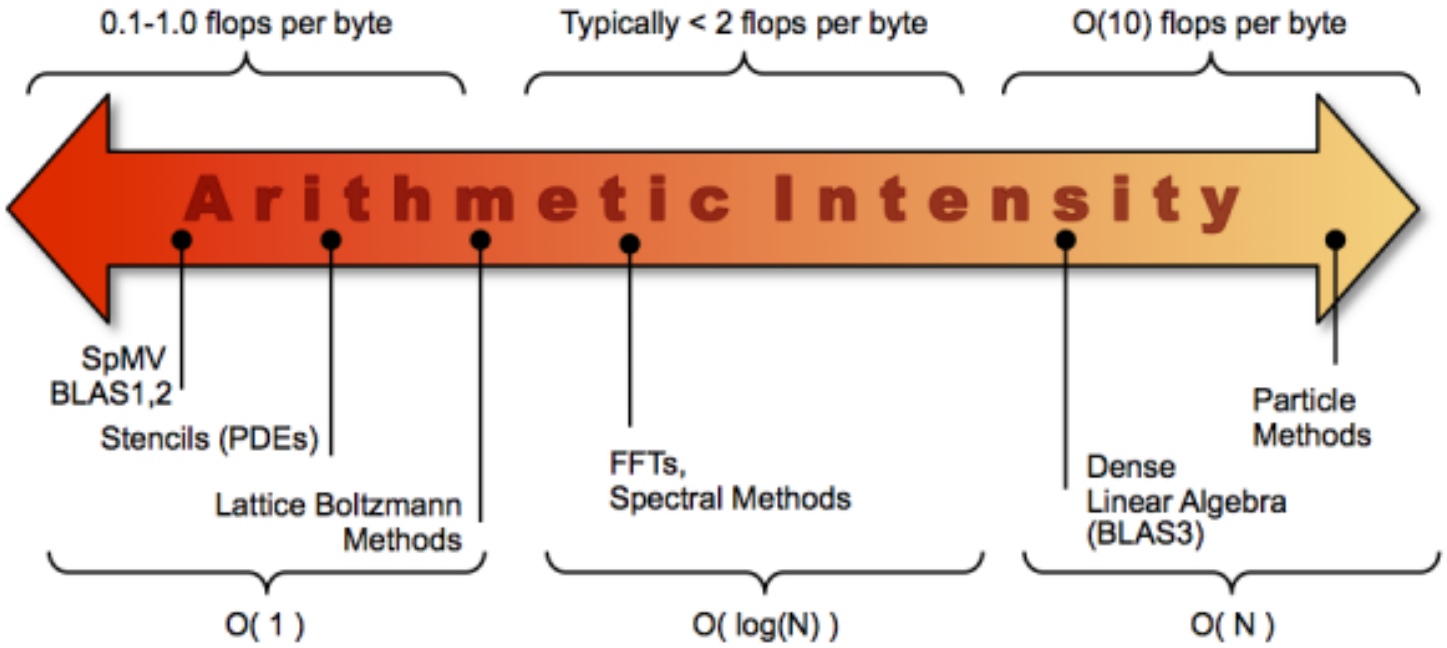
3. Extraction of the data in a report:

advixe-cl –report survey –show-all-columns –-format=text -– report-output report.txt

Parallel realization

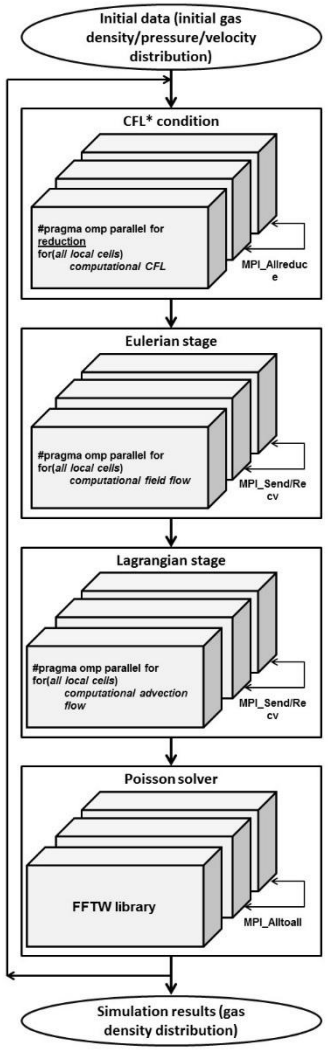


Roofline Analysis



Roofline Performance Model // https://crd.lbl.gov/departments/computer-science/PAR/research/roofline/

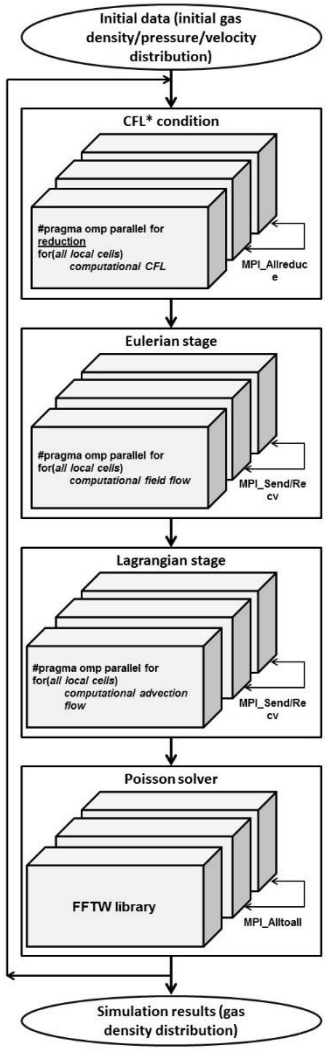
Low level parallel code



AstroPhi algorithm change history:

1) Vector dependencies removed
2) Memory load operations changed
3) Vectors size, arrays size adapted for KNL

Parallel realization



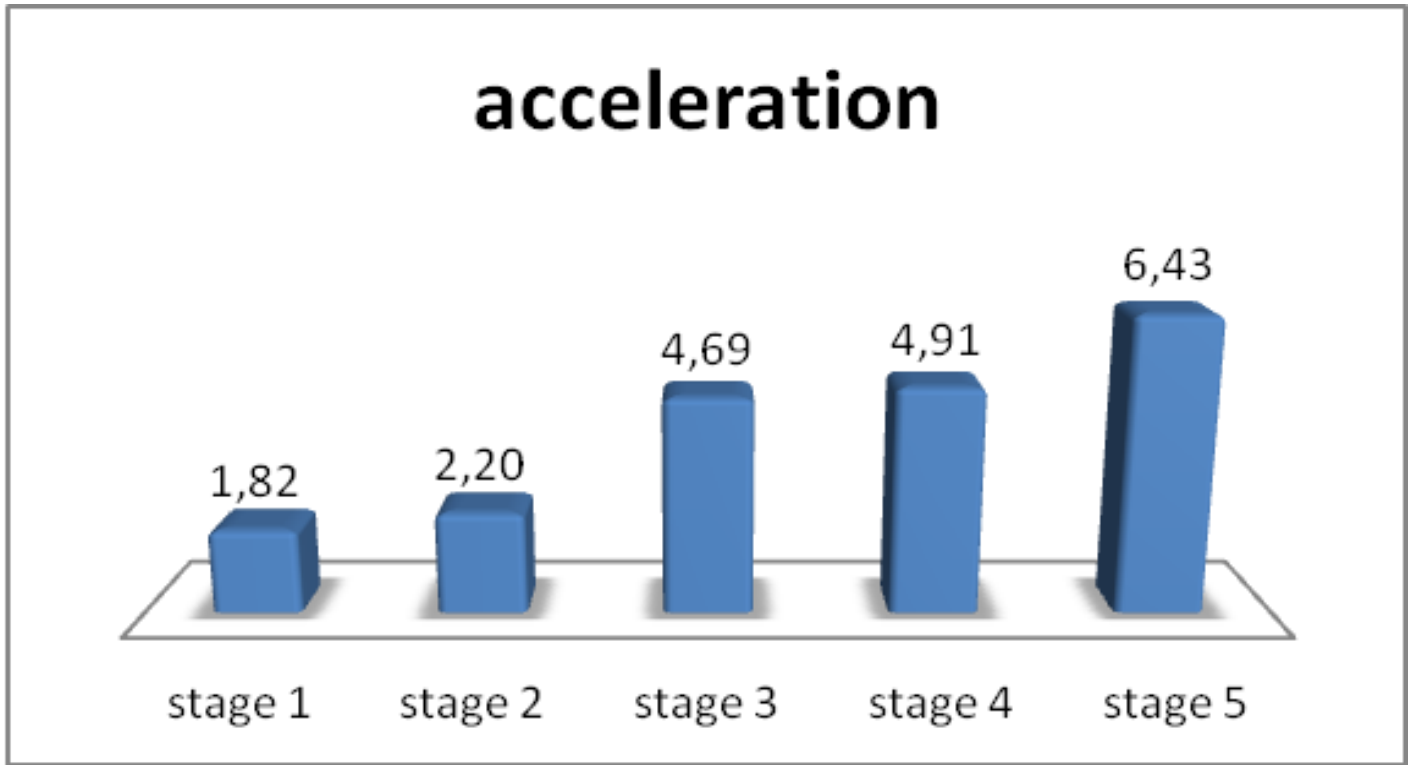
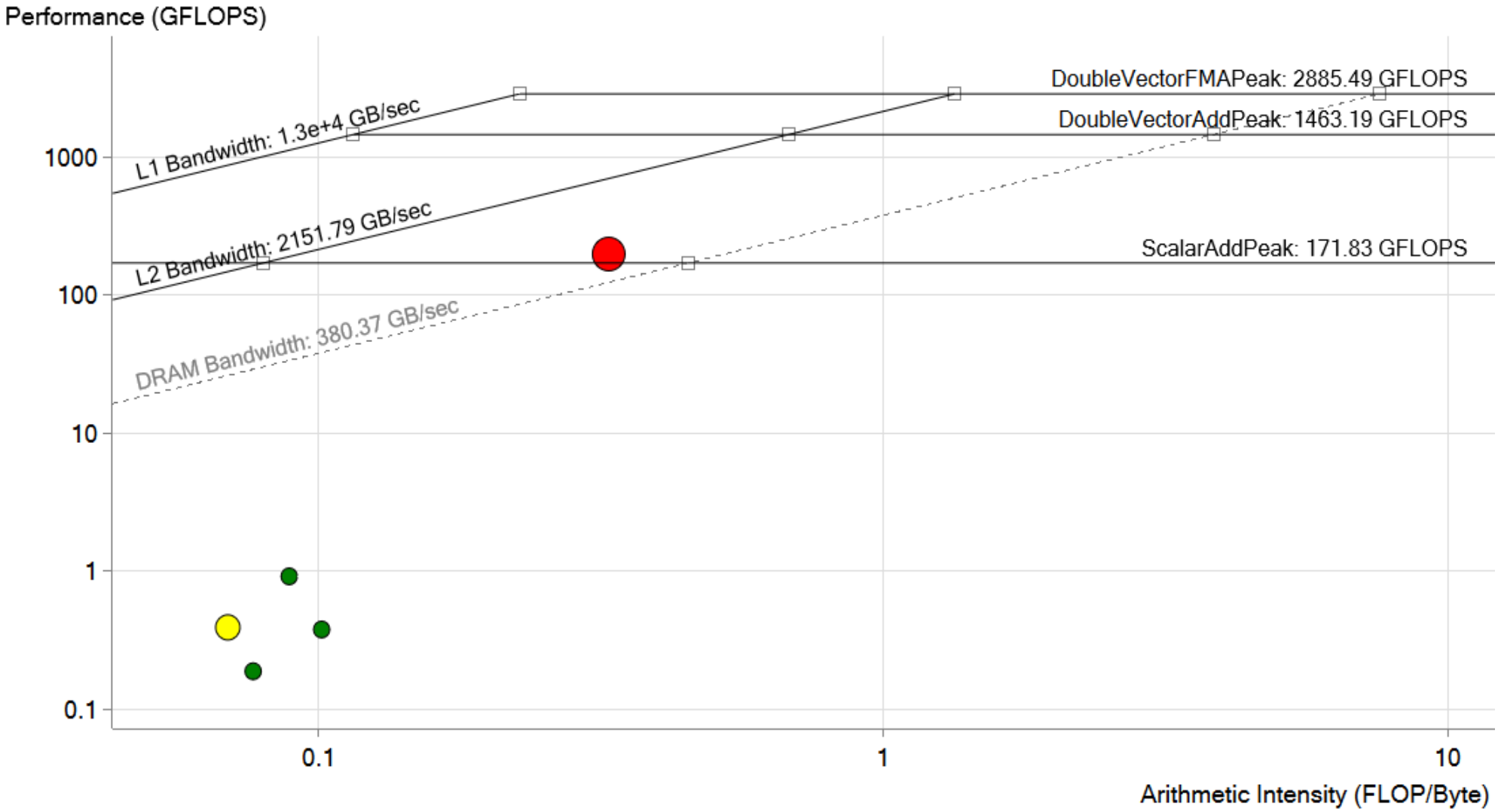


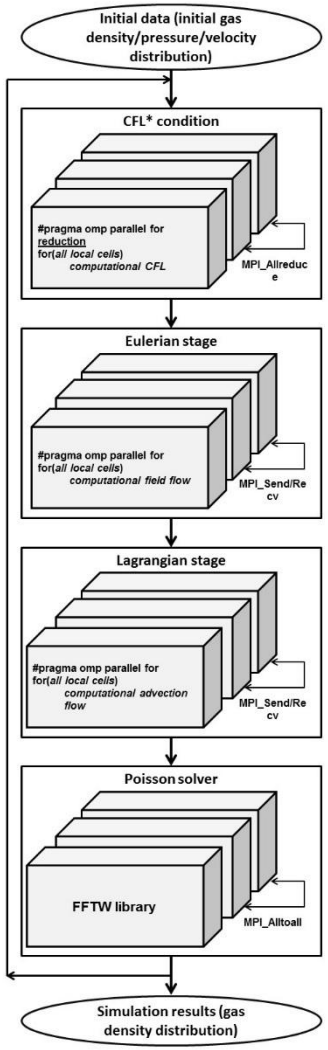
Fig. 3. Acceleration of the AstroPhi code via vectorization. Stage 1 – algorithm optimization. Stage 2 – optimization of arithmetic operations. Stage 3 – optimization of memory operations. Stage 4 – reducing a number of memory load operations. Stage 5 – transition to FMA commands.

Kulikov, Chernykh, Glinskiy, Nenashev, Shmelev, Andreev, Egunov, Kharkov. LNCS 10049

Parallel realization



Low level parallel code



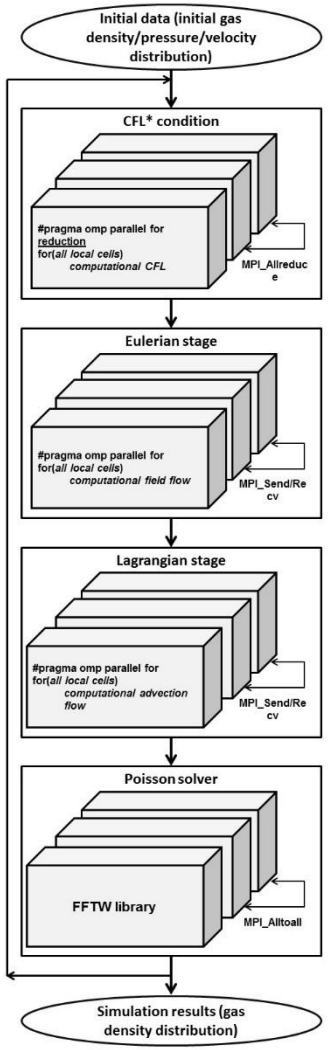

```
FYP = _mm512_mul_pd(vecincs,
                _mm512_add_pd(_mm512_sub_pd(_mm512_mul_pd(vecsr,vecfm),_mm512_mul_pd(vecsl,vecfp)),

_mm512_mul_pd(_mm512_sub_pd(vecup,vecum),_mm512_mul_pd(vecsl,vecsr))));
// down interface
vecsl = _mm512_set1_pd(Sound[index(i,k-1,l,3)]);
vecsr = _mm512_set1_pd(Sound[index(i,k,l,2)]);
vecincs = _mm512_set1_pd(1.0/(Sound[index(i,k,l,2)]-Sound[index(i,k-1,l,3)]));
vecfp = _mm512_load_pd(FY+index(i,k,l,0));
vecfm = _mm512_load_pd(FY+index(i,k-1,l,0));
vecup = _mm512_load_pd(U+index(i,k,l,0));
vecum = _mm512_load_pd(U+index(i,k-1,l,0));
FYM = _mm512_mul_pd(vecincs,
                _mm512_add_pd(_mm512_sub_pd(_mm512_mul_pd(vecsr,vecfm),_mm512_mul_pd(vecsl,vecfp)),

_mm512_mul_pd(_mm512_sub_pd(vecup,vecum),_mm512_mul_pd(vecsl,vecsr))));
```

Kulikov, Chernykh, Glinskiy, Nenashev, Shmelev, Andreev, Egunov, Kharkov.LNCS 10049

Low level parallel code




```
rvppp = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vppp, zero, _CMP_LT_OS), rvppp, next);
rvpmp = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vpmp, zero, _CMP_LT_OS), rvpmp, next);
rvmpp = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vmpp, zero, _CMP_LT_OS), rvmpp, next);
rvmmp = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vmmp, zero, _CMP_LT_OS), rvmmp, next);
rvppm = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vppm, zero, _CMP_GT_OS), rvppm, prev);
rvpmm = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vpmm, zero, _CMP_GT_OS), rvpmm, prev);
rvmpm = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vmpm, zero, _CMP_GT_OS), rvmpm, prev);
rvmmm = _mm512_mask_blend_pd(_mm512_cmp_pd_mask(vmmm, zero, _CMP_GT_OS), rvmmm, prev);
FZP = _mm512_mul_pd(_mm512_add_pd(_mm512_add_pd(_mm512_mul_pd(vppp, rvppp), _mm512_mul_pd(vpmp,
rvpmp)), _mm512_add_pd(_mm512_mul_pd(vmpp, rvmpp), _mm512_mul_pd(vmmp, rvmmp))), four);//(vppp*rvppp +
vppm*rvppm + vpmp*rvpmp + vpmm*rvpmm) / 4.0;
FZM = _mm512_mul_pd(_mm512_add_pd(_mm512_add_pd(_mm512_mul_pd(vppm, rvppm),
_mm512_mul_pd(vpmm, rvpmm)), _mm512_add_pd(_mm512_mul_pd(vmpm, rvmpm), _mm512_mul_pd(vmmm,
rvmmm))), four);//(vmpp*rvmpp + vmpm*rvmpm + vmmp*rvmmp + vmmm*rvmmm) / 4.0;
__m512d aVect = _mm512_loadu_pd(a + i*NZ*NY + k*NZ + l);
__m512d dmvVect = _mm512_set1_pd(dmv);
__m512d result = _mm512_sub_pd(aVect,
_mm512_mul_pd(_mm512_add_pd(_mm512_add_pd(_mm512_sub_pd(FXP, FXM), _mm512_sub_pd(FYP, FYM)),
_mm512_sub_pd(FZP, FZM)), dmvVect));
```

Kulikov, Chernykh, Glinskiy, Nenashev, Shmelev, Andreev, Egunov, Kharkov.LNCS 10049

Advantages and disadvantages of parallelization approaches

Classical approach:

Serial Code-> Parallel code ->

Optimization

1. Cross-platform code
2. Upgradeable code
3. Dependence on compiler quality

Advantages and disadvantages of parallelization approaches

Alternative approach:
Low level parallel code

1. Maximum performance
2. Unreadable code
3. Dependence on the architecture

Thank you for your attention