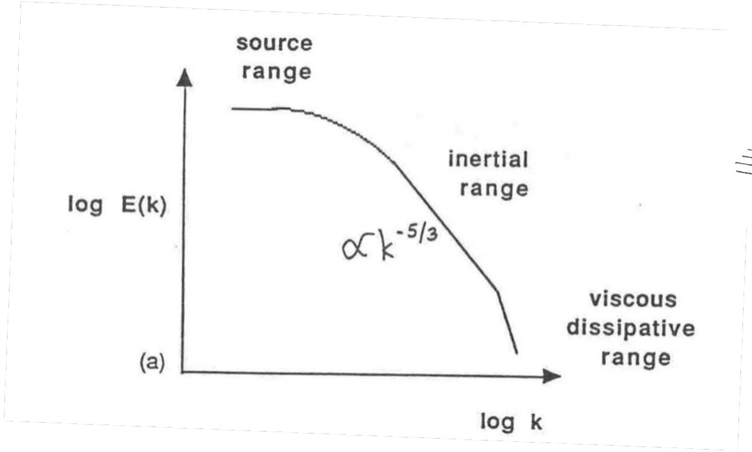
Russian Supercomputing Days
international supercomputing conference

На пути к атомистическому моделированию турбулентного потока жидкости: повышение эффективности параллельных молекулярно-динамических расчетов

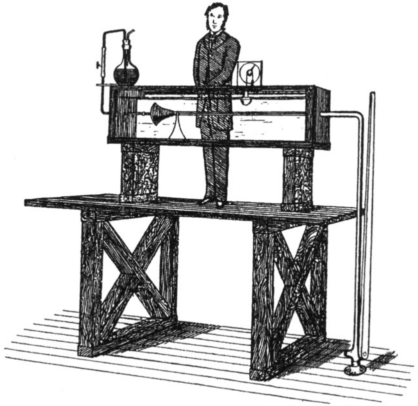
Галигеров Владислав Сергеевич[1,3],
Павлов Даниил Глебович[1,2], Стегайлов Владимир Владимирович[1,2,3]

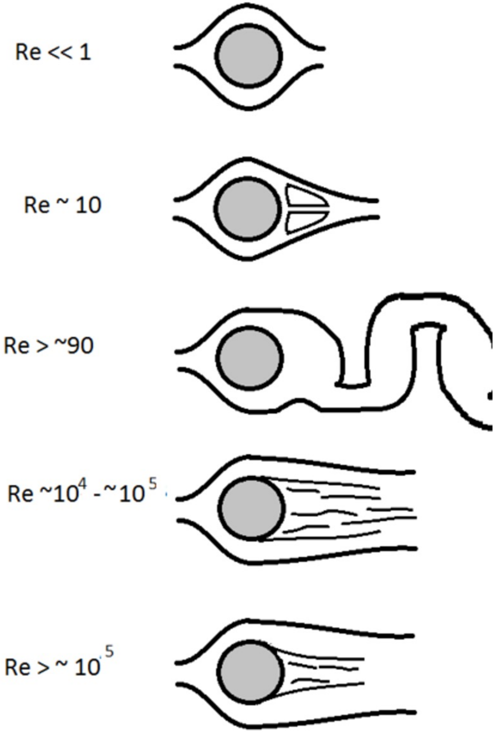
[1] ОИВТ РАН
[2] МФТИ (НИУ)
[3] НИУ ВШЭ

Введение



Источник: astronomy.ohio-state.edu/ryden.1/ast825/ch7.pdf





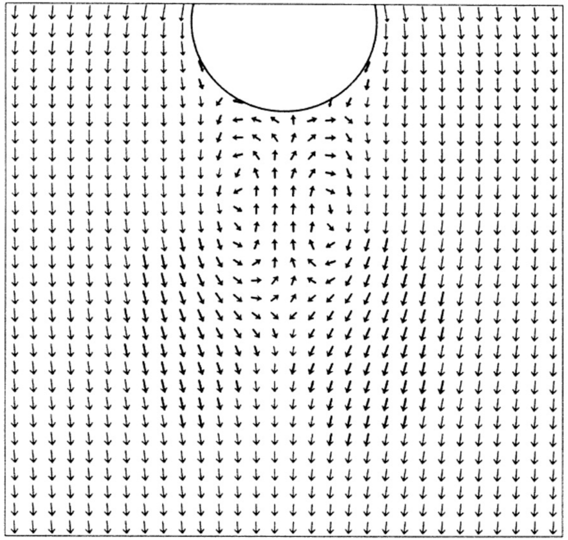
Big whirls have little whirls that feed on their velocity,

and little whirls have lesser whirls and so on to viscosity.

Льюис Ричардсон

Источник: https://en.wikipedia.org/wiki/Reynolds_number

Молекулярная динамика

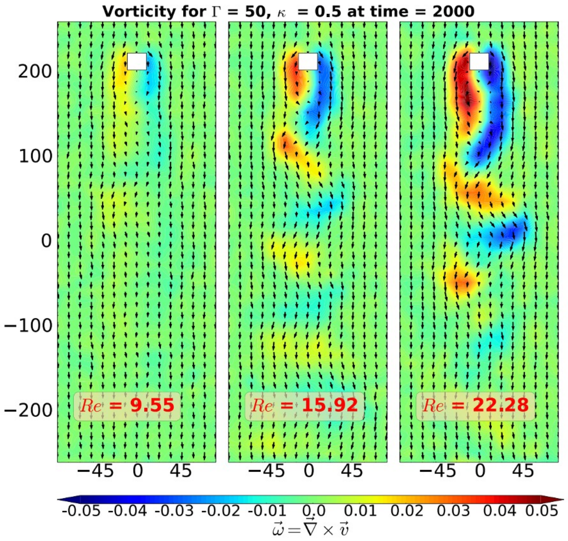


Rapaport, 1986

N ≈ 10^4

Re ≈ 25

2D

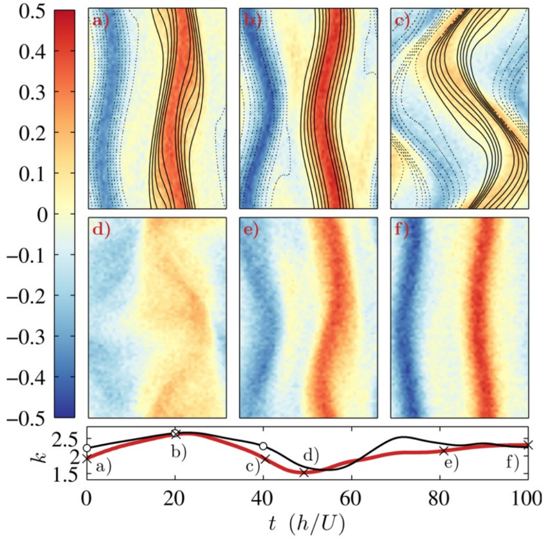


Charan, Harish, 2016

N ≈ 6*10^4

Re ≈ 30

2D



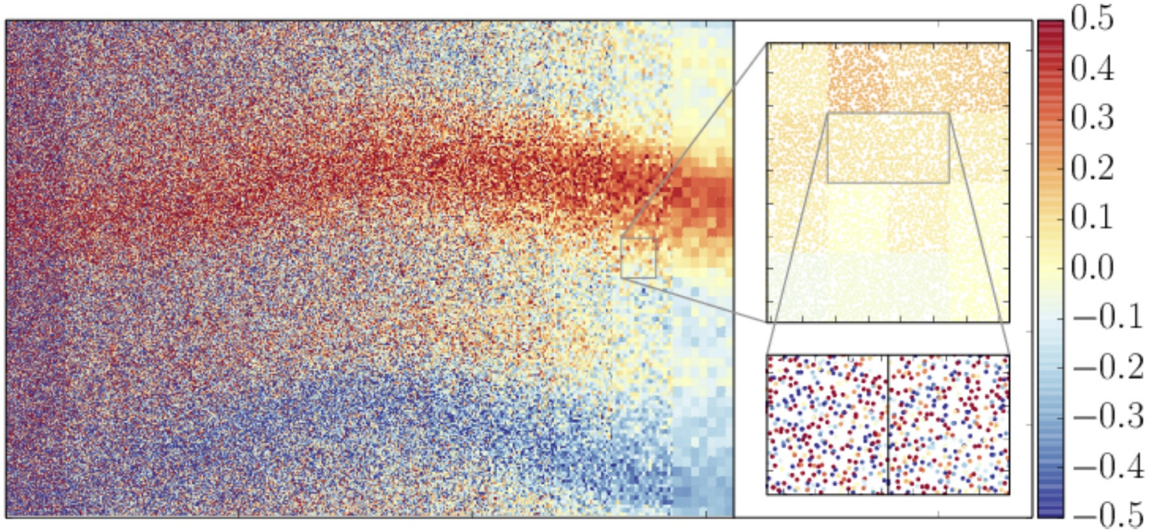
Smith, 2015

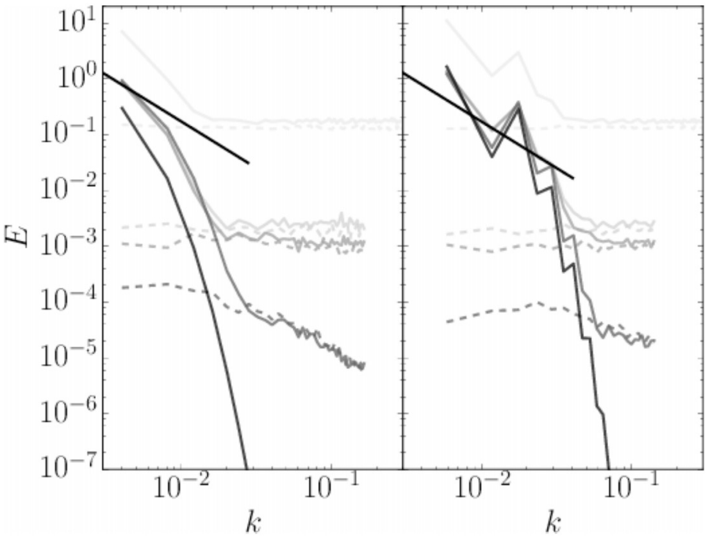
N ≈ 3*10^8

Re ≈ 450

3D

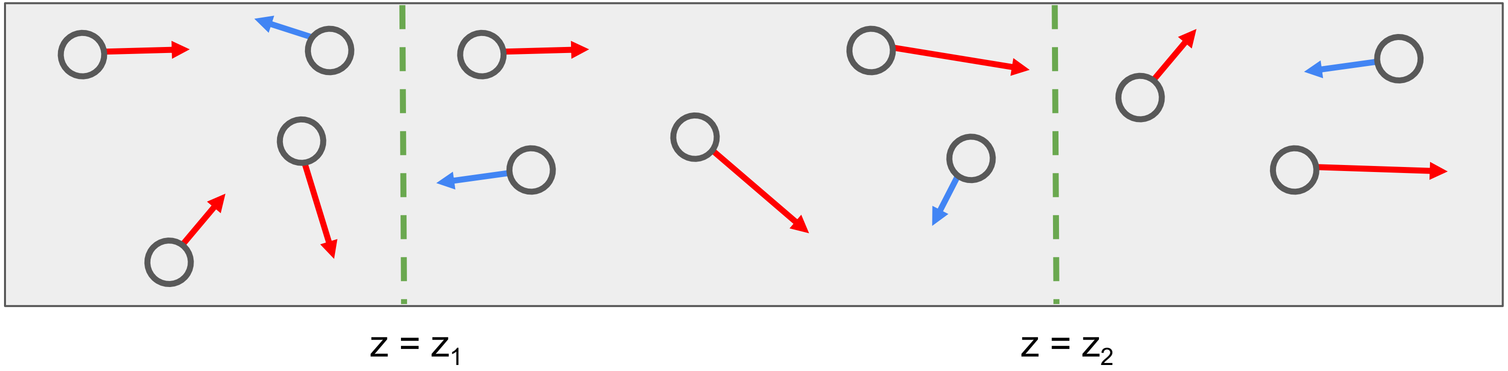
Молекулярная динамика





E. R. Smith, “A molecular dynamics simulation of the turbulent Couette minimal flow unit” // Physics of Fluids 27, 115105 (2015)

Граничные условия



$$\frac{\partial f(\mathbf{r},\mathbf{v},t)}{\partial t} + \mathbf{v}\cdot\frac{\partial f(\mathbf{r},\mathbf{v},t)}{\partial \mathbf{r}} + \frac{\mathbf{F}_{\mathrm{ext}}}{m}\cdot\frac{\partial f(\mathbf{r},\mathbf{v},t)}{\partial \mathbf{v}} = \left.\frac{\partial f}{\partial t}\right|_{\mathrm{coll}}$$

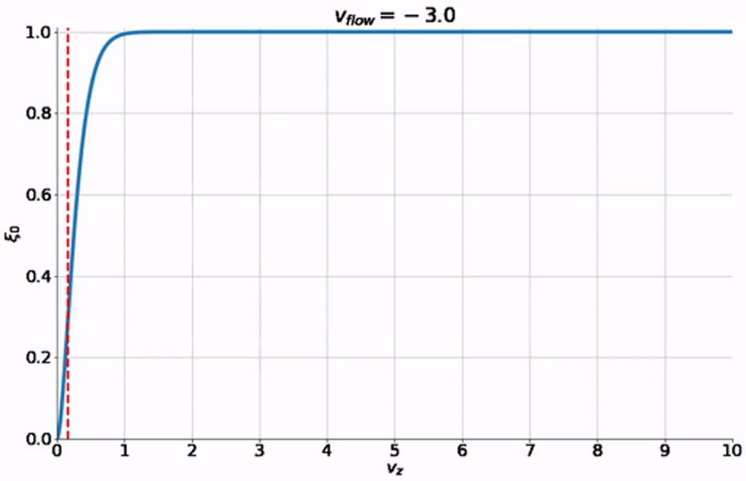
$$s(\mathbf{r},\mathbf{v}) = \left.\frac{\partial f}{\partial t}\right|_{\mathrm{coll}}$$

Граничные условия

$$f_z(v_z) \propto \exp\left(-\frac{m(v_z - v_{\text{flow}})^2}{2kT}\right)$$

$$F(v_z) = \int_0^{v_z} v'_z f_z(v'_z)\, dv'_z$$

$$\xi_0 F(\infty) = F(v_z)$$



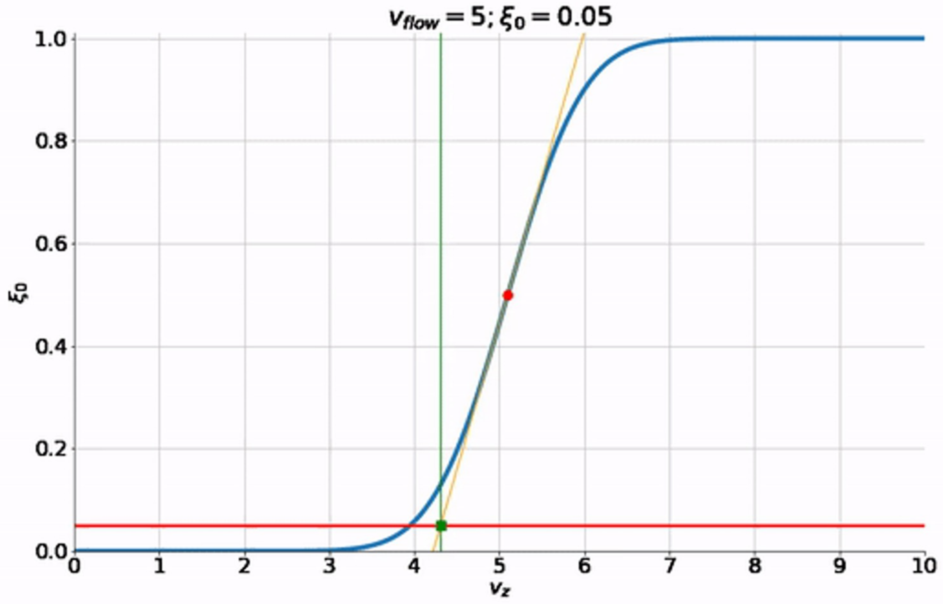
Соотношение ξ_0 и v_z для различных скоростей потока

Красным показана точка перегиба

Граничные условия

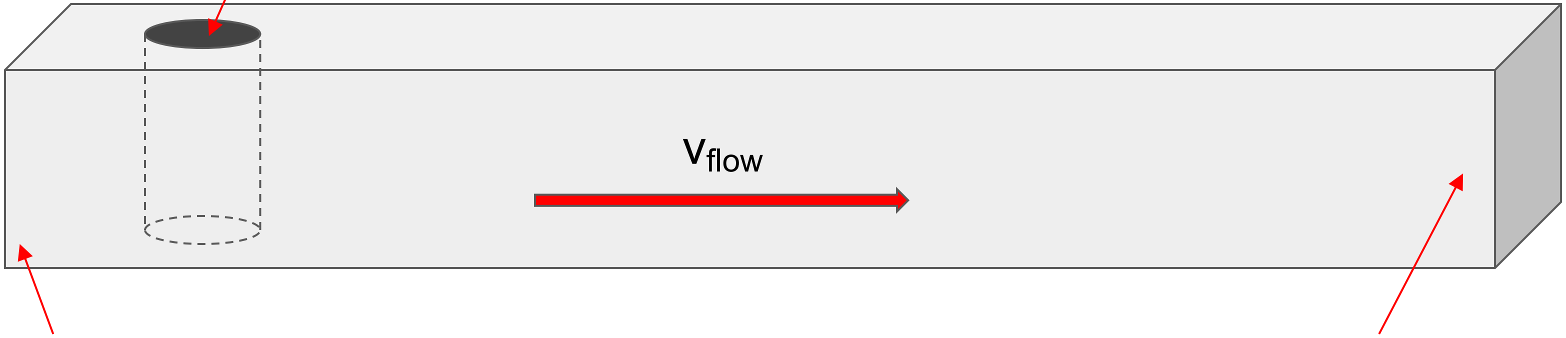
$$v_{i+1} = v_i - \frac{F(v_i) - \xi_0 F(\infty)}{F'(v_i)}$$

$$v_0 = \frac{1}{2}\left(v_{\text{flow}} - \sqrt{v_{\text{flow}}^2 + \frac{4kT}{m}}\right)$$



Сходимость метода Ньютона для различных возможных комбинаций v_{flow} и ξ_0

Система (вид сбоку)



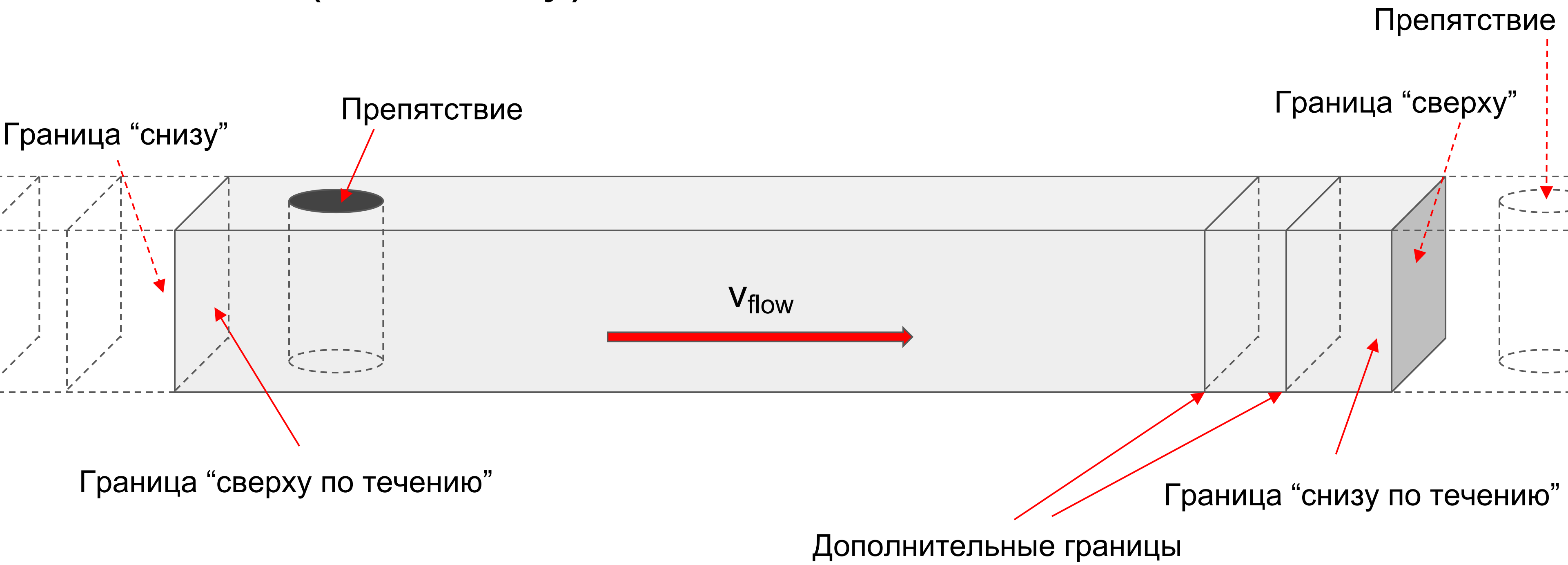
Граница “сверху по течению” (в основном испускает частицы)

Граница “снизу по течению” (в основном поглощает частицы)

Завихренность $\omega = \nabla \times \mathbf{v} = \begin{pmatrix} \partial_x \\ \partial_y \\ \partial_z \end{pmatrix} \times \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix},$

$$\omega(\mathbf{r}) = \left\langle \begin{pmatrix} 1/\Delta r_x \\ 1/\Delta r_y \\ 1/\Delta r_z \end{pmatrix} \times (\overline{\mathbf{v}}(\mathbf{r}+\Delta\mathbf{r}) - \overline{\mathbf{v}}(\mathbf{r})) \right\rangle_{\Delta\mathbf{r}\in U(\mathbf{r})}$$

Система (вид сбоку)

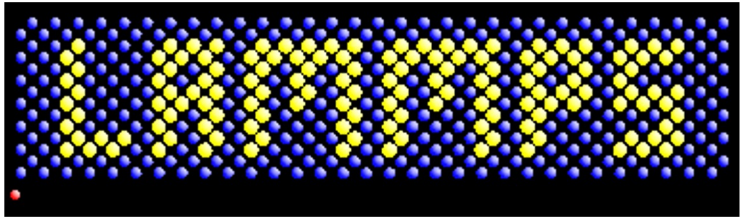


OpenMM

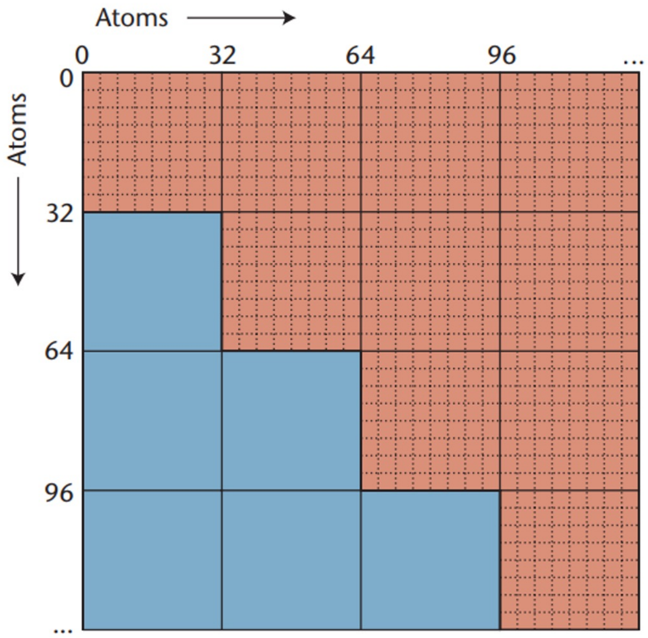
- High performance molecular dynamics library
- Is able to very efficiently utilize a single GPUs
- Convenient C++ and Python APIs
- Easy to customize and extend

LAMMPS

- High performance molecular dynamics library
- Is able to efficiently utilize multiple GPUs
- Custom scripting interface
- Possible to customize and extend



Внутренний алгоритм OpenMM



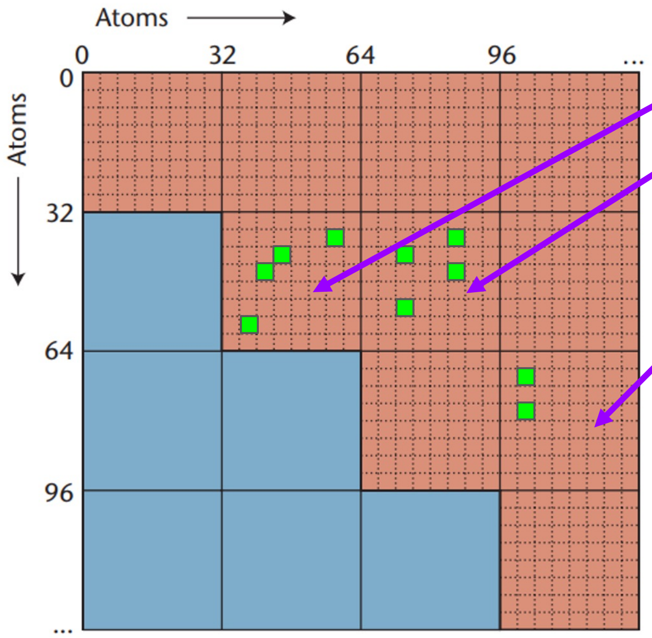
Eastman, Pande
(Computing in science and engineering, 2010)

```
class OPENMM_EXPORT OpenBoundary : public Force {
public:
    /**
     * Create an OpenBoundary.
     *
     * @param defaultTemperature         the default temperature at the boundary (in Kelvin)
     * @param defaultVelocity  the default velocity at the boundary (in nm/ps)
     * @param axis  the axis at which to apply open boundary
     * @param internalBoundaries  an array of extra internal boundaries in (0, 1)
     */
    OpenBoundary(double defaultTemperature, double defaultVelocity, char axis, const std::ve
```

➔ Force objects define the behavior of the particles in a System. The Force class is actually slightly more general than its name suggests. A Force can, indeed, apply forces to particles, but it can also directly modify particle positions and velocities in arbitrary ways.

Внутренний алгоритм OpenMM

Помечены как “взаимодействующие”



Eastman, Pande
(Computing in science and engineering, 2010)

+650 −586

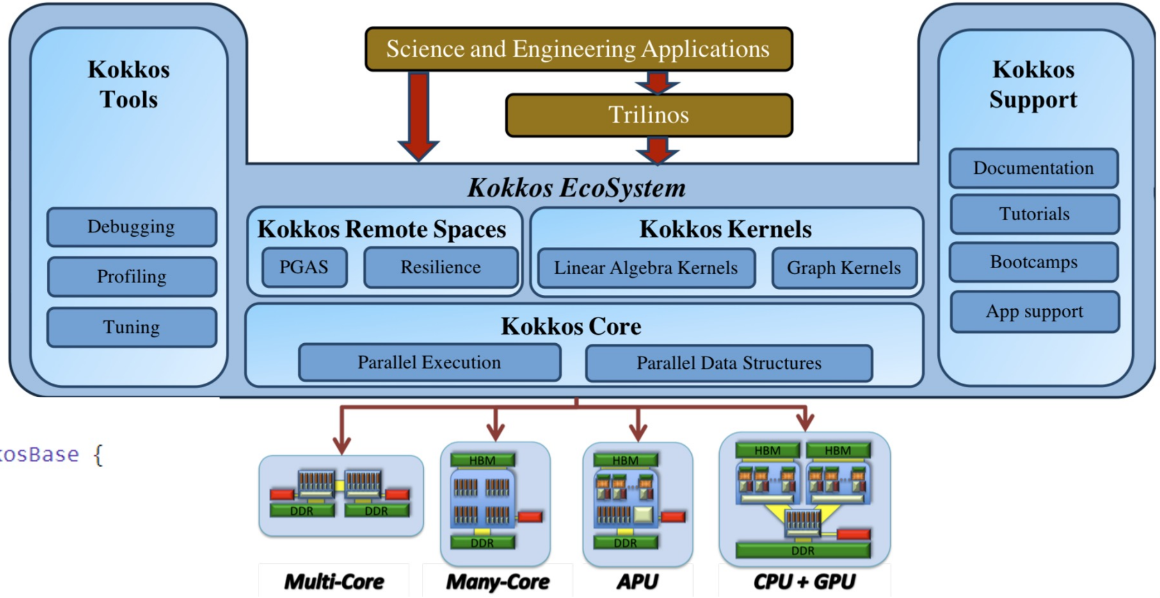
```
unsigned int* __restrict__ interactionCount
```

LAMMPS

- In LAMMPS, a “fix” is any operation that is applied to the system during timestepping or minimization.
- `fix wall/flow` для поддержания стационарного потока
- `compute property/grid` для обработки на лету

```
class FixWallFlow : public Fix {
 public:
  enum FlowAxis {AX_X = 0, AX_Y = 1, AX_Z = 2};

  FixWallFlow(class LAMMPS *, int, char **);
  ~FixWallFlow() override;
  int setmask() override;
  void init() override;
  void end_of_step() override;

  void grow_arrays(int) override;
  void copy_arrays(int, int, int) override;

  int pack_exchange(int, double *) override;
  int unpack_exchange(int, double *) override;
```

LAMMPS/Kokkos




```
template<class DeviceType>
class FixWallFlowKokkos : public FixWallFlow, public KokkosBase {
 public:
  typedef DeviceType device_type;
  typedef ArrayTypes<DeviceType> AT;
  struct MassTag{};
  struct RMassTag{};
  FixWallFlowKokkos(class LAMMPS *, int, char **);
  ~FixWallFlowKokkos();

  void init() override;
  void end_of_step() override;
  void grow_arrays(int) override;
  void copy_arrays(int, int, int) override;
  void sort_kokkos(Kokkos::BinSort<KeyViewType, BinOp> &Sorter) override;
  int pack_exchange(int, double *) override;
  int unpack_exchange(int, double *) override;
```

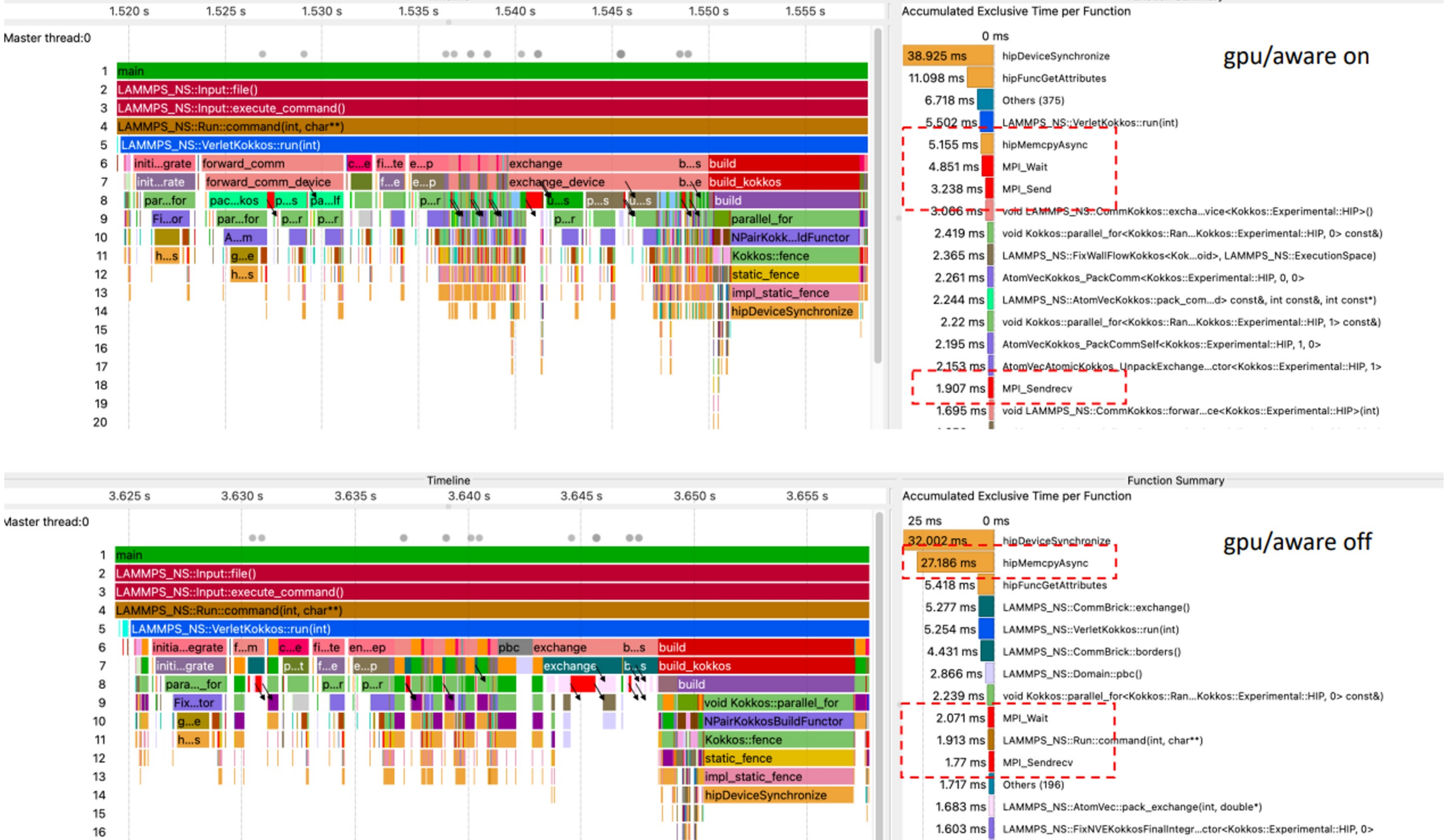
LAMMPS/Kokkos

Kokkos exchange comm for fixes #1394

Merged akohlmey merged 69 commits into `lammps:develop` from `valleymouth:granular-kokkos` on Apr 12

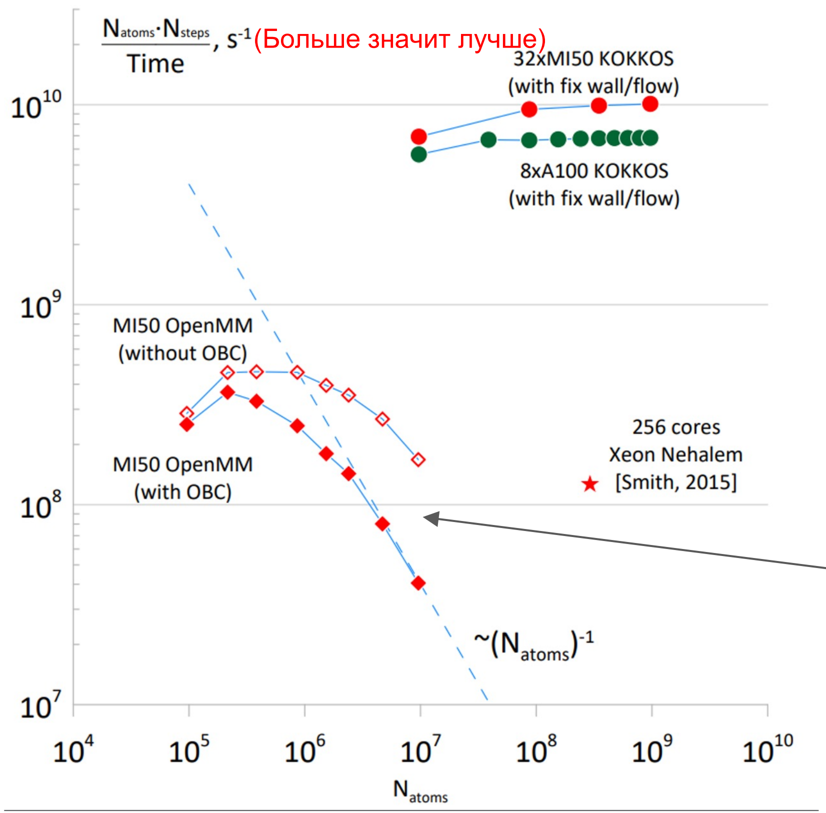
```
template<class DeviceType>
class FixWallFlowKokkos : public FixWallFlow, public KokkosBase {
 public:
  typedef DeviceType device_type;
  typedef ArrayTypes<DeviceType> AT;
  struct MassTag{};
  struct RMassTag{};
  FixWallFlowKokkos(class LAMMPS *, int, char **);
  ~FixWallFlowKokkos();

  void init() override;
  void end_of_step() override;
  void grow_arrays(int) override;
  void copy_arrays(int, int, int) override;
  void sort_kokkos(Kokkos::BinSort<KeyViewType, BinOp> &Sorter) override;
  int pack_exchange(int, double *) override;
  int unpack_exchange(int, double *) override;
```

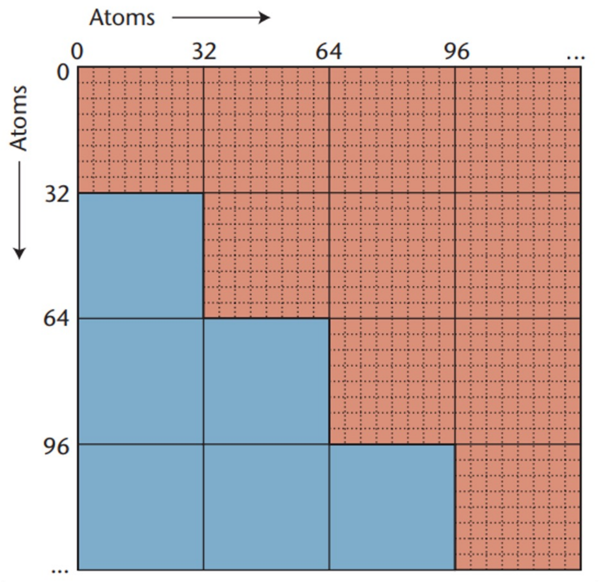
```
int pack_exchange_kokkos(const int &nsend,DAT::tdual_xfloat_2d &buf,
                         DAT::tdual_int_1d k_sendlist,
                         DAT::tdual_int_1d k_copylist,
                         ExecutionSpace space) override;

void unpack_exchange_kokkos(DAT::tdual_xfloat_2d &k_buf,
                            DAT::tdual_int_1d &indices,int nrecv,
                            ExecutionSpace space) override;
```



1.520 s
1.525 s
1.530 s
1.535 s
1.540 s
1.545 s
1.550 s
1.555 s
Master thread:0
main
LAMMPS_NS::Input::file()
LAMMPS_NS::Input::execute_command()
LAMMPS_NS::Run::command(int, char**)
LAMMPS_NS::VerletKokkos::run(int)
forward_comm
forward_comm_device
exchange
exchange_device
build
build_kokkos
parallel_for
NPairKokk...ldFunctor
Kokkos::fence
static_fence
impl_static_fence
hipDeviceSynchronize
Accumulated Exclusive Time per Function
gpu/aware on
0 ms
38.925 ms hipDeviceSynchronize
11.098 ms hipFuncGetAttributes
6.718 ms Others (375)
5.502 ms LAMMPS_NS::VerletKokkos::run(int)
5.155 ms hipMemcpyAsync
4.851 ms MPI_Wait
3.238 ms MPI_Send
3.066 ms void LAMMPS_NS::CommKokkos::excha...vice<Kokkos::Experimental::HIP>()
2.419 ms void Kokkos::parallel_for<Kokkos::Ran...Kokkos::Experimental::HIP, 0> const&)
2.365 ms LAMMPS_NS::FixWallFlowKokkos<Kok...oid>, LAMMPS_NS::ExecutionSpace)
2.261 ms AtomVecKokkos_PackComm<Kokkos::Experimental::HIP, 0, 0>
2.244 ms LAMMPS_NS::AtomVecKokkos::pack_com...d> const&, int const&, int const*)
2.22 ms void Kokkos::parallel_for<Kokkos::Ran...Kokkos::Experimental::HIP, 1> const&)
2.195 ms AtomVecKokkos_PackCommSelf<Kokkos::Experimental::HIP, 1, 0>
2.153 ms AtomVecAtomicKokkos_UnpackExchange...ctor<Kokkos::Experimental::HIP, 1>
1.907 ms MPI_Sendrecv
1.695 ms void LAMMPS_NS::CommKokkos::forwar...ce<Kokkos::Experimental::HIP>(int)
Timeline
Function Summary
3.625 s
3.630 s
3.635 s
3.640 s
3.645 s
3.650 s
3.655 s
Master thread:0
main
LAMMPS_NS::Input::file()
LAMMPS_NS::Input::execute_command()
LAMMPS_NS::Run::command(int, char**)
LAMMPS_NS::VerletKokkos::run(int)
pbc
exchange
build
build_kokkos
void Kokkos::parallel_for
NPairKokkosBuildFunctor
Kokkos::fence
static_fence
impl_static_fence
hipDeviceSynchronize
Accumulated Exclusive Time per Function
gpu/aware off
25 ms 0 ms
32.002 ms hipDeviceSynchronize
27.186 ms hipMemcpyAsync
5.418 ms hipFuncGetAttributes
5.277 ms LAMMPS_NS::CommBrick::exchange()
5.254 ms LAMMPS_NS::VerletKokkos::run(int)
4.431 ms LAMMPS_NS::CommBrick::borders()
2.866 ms LAMMPS_NS::Domain::pbc()
2.239 ms void Kokkos::parallel_for<Kokkos::Ran...Kokkos::Experimental::HIP, 0> const&)
2.071 ms MPI_Wait
1.913 ms LAMMPS_NS::Run::command(int, char**)
1.77 ms MPI_Sendrecv
1.717 ms Others (196)
1.683 ms LAMMPS_NS::AtomVec::pack_exchange(int, double*)
1.603 ms LAMMPS_NS::FixNVEKokkosFinalIntegr...ctor<Kokkos::Experimental::HIP, 0>

Бенчмаркинг

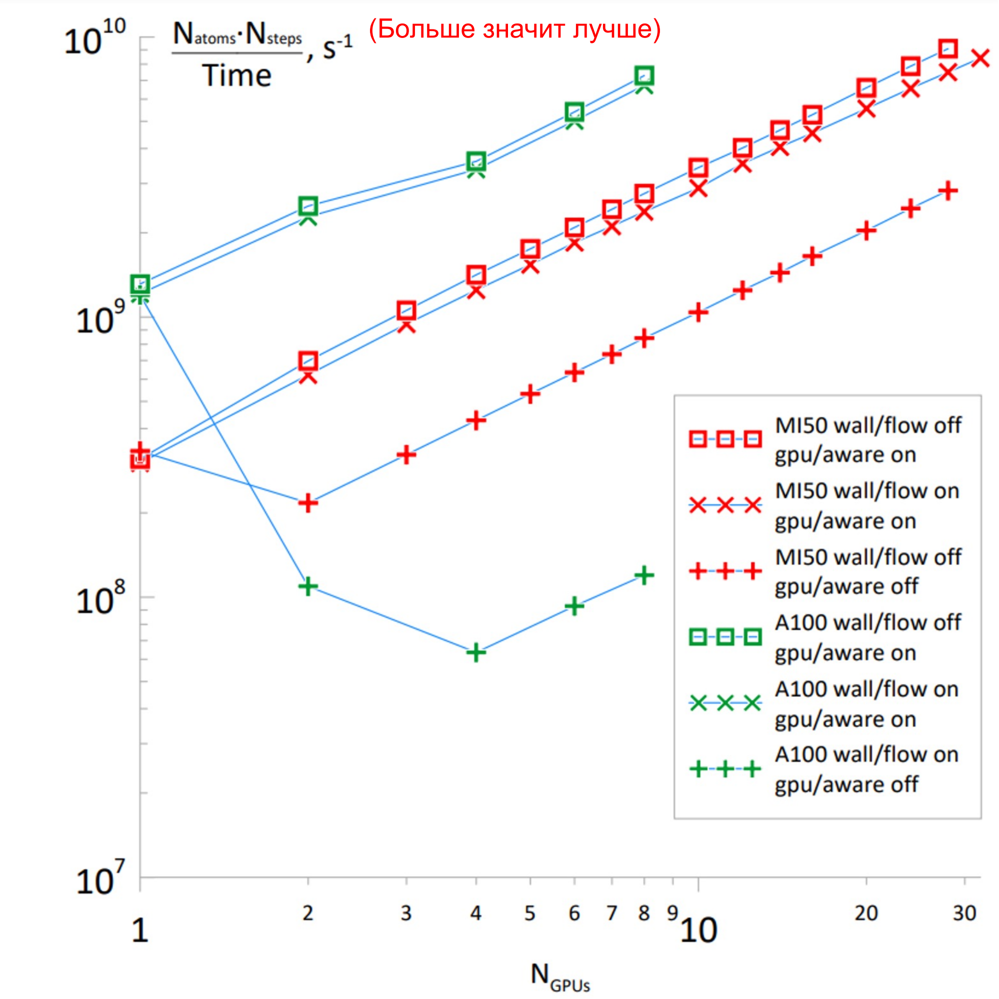




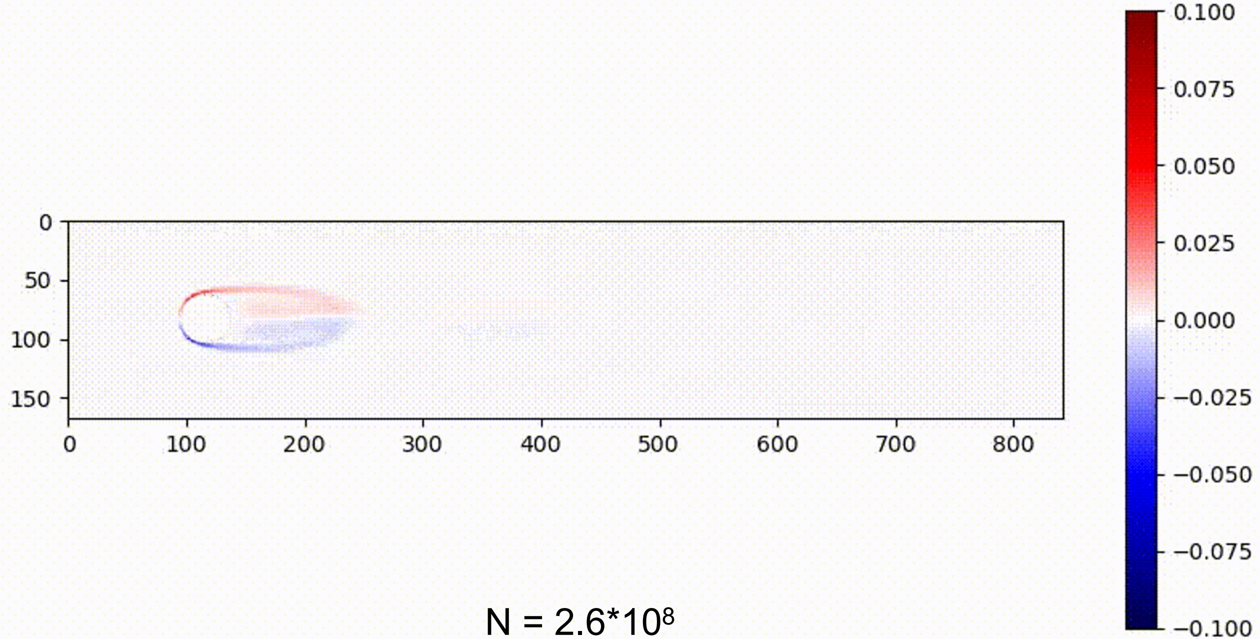
$O(N^2)$

Сильное масштабирование

Nsteps = 10000
Natoms = 60480000



Демонстрация



$N = 2.6*10^8$
Re = 400

Спасибо за внимание!

Итого:

- ➔ Были созданы две реализации предложенного метода поддержания стационарного потока жидкости.
- ➔ Реализации были портированы на GPU ускорители
- ➔ Благодаря анализу узких мест была достигнута масштабируемость реализации для LAMMPS/Kokkos

Исходный код находится в открытом доступе:

- https://github.com/vladgl/lammps/tree/develop
- https://github.com/Dann239/openmm/tree/open-boundary

Публикации по результатам работы:

Pavlov D., Kolotinskii D., Stegailov V. **GPU-Based Molecular Dynamics of Turbulent Liquid Flows with OpenMM** // International Conference on Parallel Processing and Applied Mathematics. – Cham : Springer International Publishing, 2022. – С. 346-358.

D. Pavlov, V. Galigerov, D. Kolotinskii, V. Nikolskiy and V. Stegailov, “**GPU-based Molecular Dynamics of Fluid Flows: Reaching for Turbulence**” (in press)