FLUPS: a versatile and performant FFT-based library of unbounded Poisson solvers

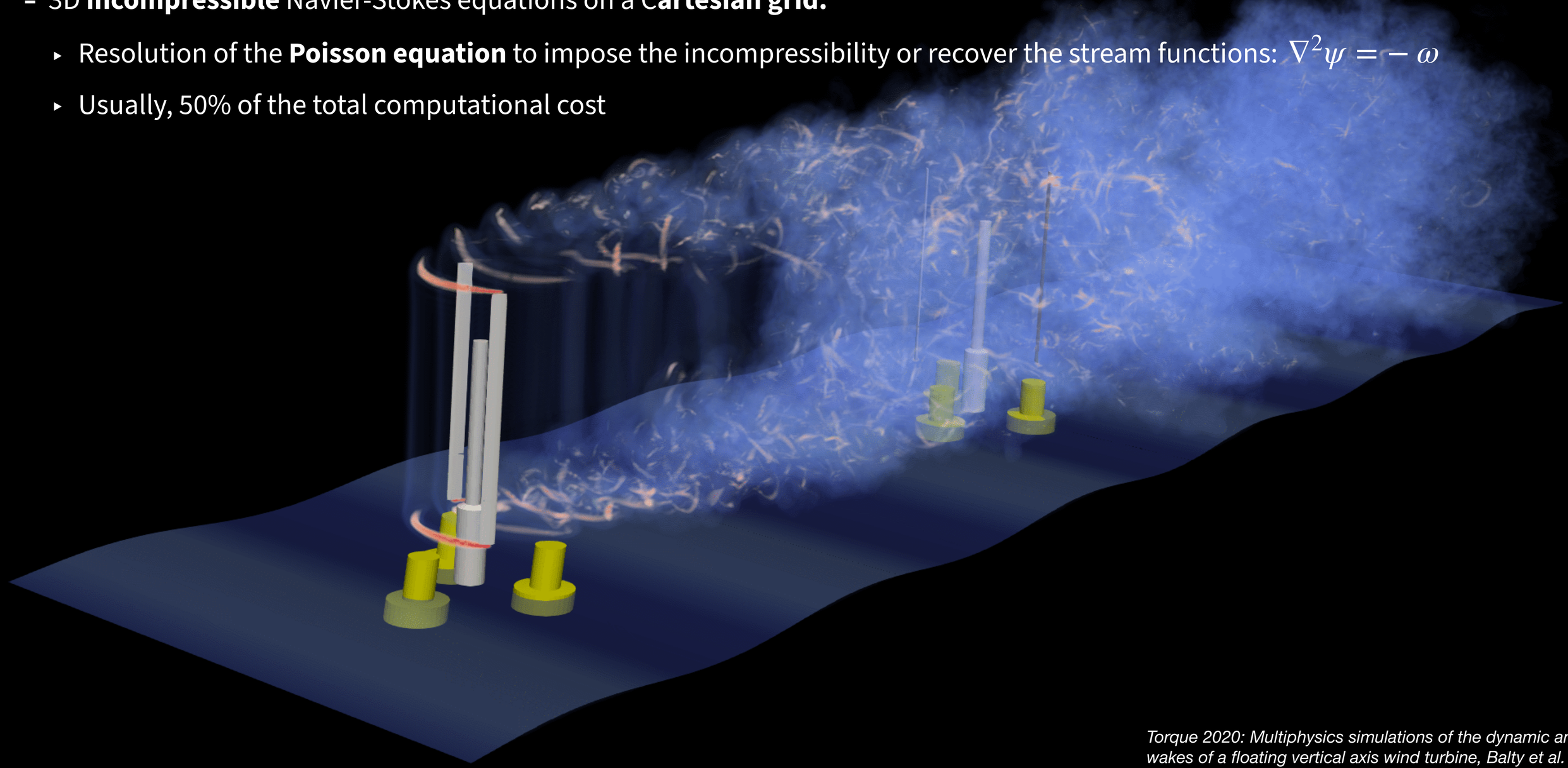
P. Balty, D.-G. Caprace, P. Chatelain, T. Gillis

UCLouvain
Institute of Mechanics,
Materials and Civil Engineering

SIAM - CSE 2023

Context

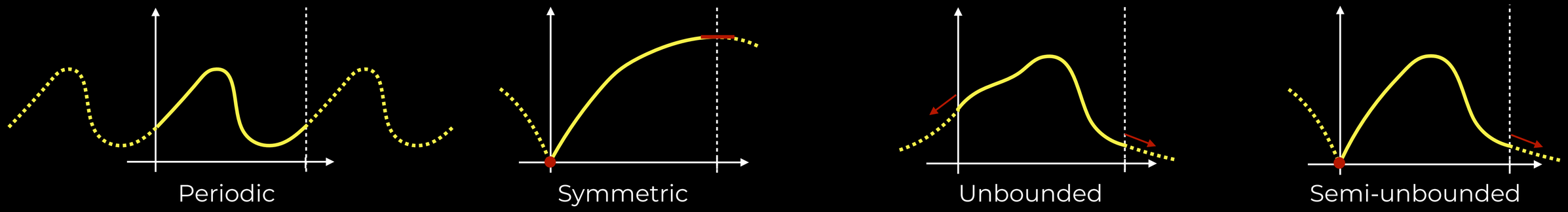
- 3D **incompressible** Navier-Stokes equations on a C**artesian grid:**
  - Resolution of the **Poisson equation** to impose the incompressibility or recover the stream functions: $\nabla^2 \psi = -\omega$
  - Usually, 50% of the total computational cost



Torque 2020: Multiphysics simulations of the dynamic and wakes of a floating vertical axis wind turbine, Balty et al.

Context

- 3D **incompressible** Navier-Stokes equations on a C**artesian grid:**
  - Resolution of the **Poisson equation** to impose the incompressibility or recover the stream functions: $\nabla^2\psi = -\omega$
  - Usually, 50% of the total computational cost
- Specific need:
  - Performance on massively parallel systems
  - Flexibility in the data layout: cell-centered or node-centered
  - Various boundary conditions (BC)**:**



- FFT-based method is compatible with most of the BCs and are the fastest on uniform rectangular grid [1].

[1] Gholami:2016

Plan for today

I. Context

II. Methodology

III. Implementation for massively distributed systems

IV. Results

V. Conclusion

An unbounded FFT-based Poisson Solver…

- FFT-based method:

$$\nabla^2 \mathbf{\Psi} = -f \longrightarrow \psi = G \star f \longrightarrow \hat{\psi} = \hat{G}\hat{f}$$

  - 3D FFT obtained as a succession of 1D FFTs: $\psi_{x,y,z} \rightarrow \tilde{\psi}_{x,ky,z} \rightarrow \tilde{\psi}_{x,ky,kz} \rightarrow \hat{\psi}_{kx,ky,kz}$
  - Analytical expression of the Green's functions (provided some regularisation of a given order)
  - Different transforms for each boundary condition and data layout

An unbounded FFT-based Poisson Solver…

- FFT-based method:

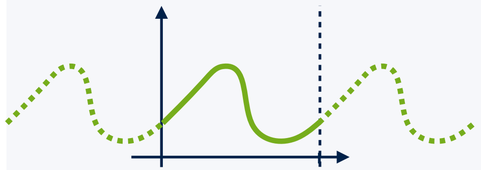
$$\nabla^2 \mathbf{\Psi} = -f \longrightarrow \psi = G \star f \longrightarrow \hat{\psi} = \hat{G}\hat{f}$$

  - 3D FFT obtained as a succession of 1D FFTs: $\psi_{x,y,z} \rightarrow \tilde{\psi}_{x,ky,z} \rightarrow \tilde{\psi}_{x,ky,kz} \rightarrow \hat{\psi}_{kx,ky,kz}$
  - Analytical expression of the Green's functions (provided some regularisation of a given order)
  - Different transforms for each boundary condition and data layout

Periodic or symmetric boundary condition

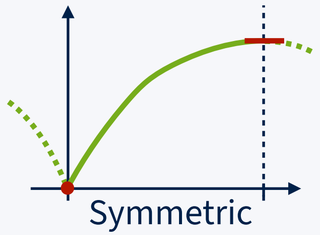
$$\nabla^2 \mathbf{\Psi} = -f \Leftrightarrow \psi = G \star f \Leftrightarrow \hat{\psi} = \hat{G}\hat{f}$$

where $\hat{G} = -1/k^2$ is the Green's function in a spectral domain



Periodic

$f(x) \rightarrow \tilde{f}(k)$ → *1D Discrete Fourier transform (real to complex or complex to complex)*



Symmetric

$f(x) \rightarrow \tilde{f}(k)$ → *1D Discrete cosine/sine transform* depending on the symmetry

… with various boundary conditions

- FFT-based method:

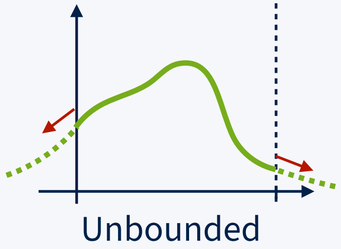
$$\nabla^2 \mathbf{\Psi} = -f \longrightarrow \psi = G \star f \longrightarrow \hat{\psi} = \hat{G}\hat{f}$$

  - 3D FFT obtained as a succession of 1D FFTs: $\psi_{x,y,z} \rightarrow \tilde{\psi}_{x,ky,z} \rightarrow \tilde{\psi}_{x,ky,kz} \rightarrow \hat{\psi}_{kx,ky,kz}$

Unbounded and semi-unbounded boundary condition

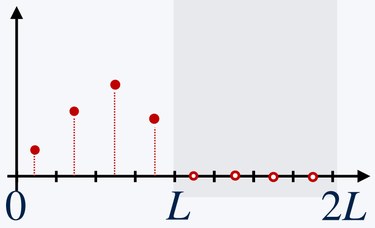
$$\nabla^2 \mathbf{\Psi} = -f \Leftrightarrow \psi = G_\delta \star f \Leftrightarrow \hat{\psi} = \hat{G}_\delta \hat{f}$$

where $\hat{G}_\delta$ is the regularized Green's function



$f(x) \rightarrow \tilde{f}(k)$

→ *Domain doubling technique[2,3]:*



- *f is extended to a domain of 2N with 0 padding*
- *Discrete Fourier transform on the padded function*

[2] Hockney:1988
[3] Caprace : 2021

… with various boundary conditions

- FFT-based method:

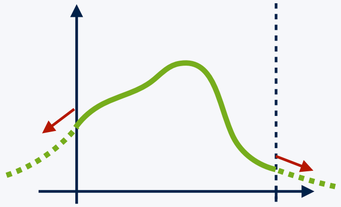
$$\nabla^2 \boldsymbol{\Psi} = -f \longrightarrow \psi = G \star f \longrightarrow \hat{\psi} = \hat{G}\hat{f}$$

– 3D FFT obtained as a succession of 1D FFTs: $\psi_{x,y,z} \rightarrow \tilde{\psi}_{x,ky,z} \rightarrow \tilde{\psi}_{x,ky,kz} \rightarrow \hat{\psi}_{kx,ky,kz}$

Unbounded and semi-unbounded boundary condition

$$\nabla^2 \boldsymbol{\Psi} = -f \Leftrightarrow \psi = G_\delta \star f \Leftrightarrow \hat{\psi} = \hat{G}_\delta \hat{f}$$

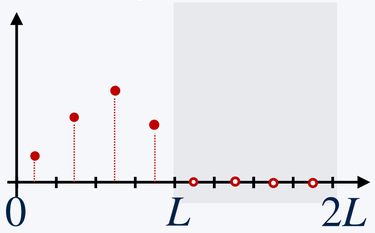
where $\hat{G}_\delta$ is the regularized Green's function



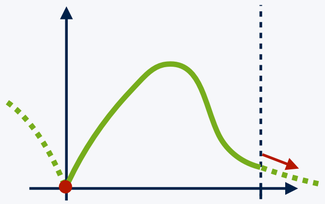
Unbounded

$f(x) \rightarrow \tilde{f}(k)$

→ *Domain doubling technique[2,3]:*

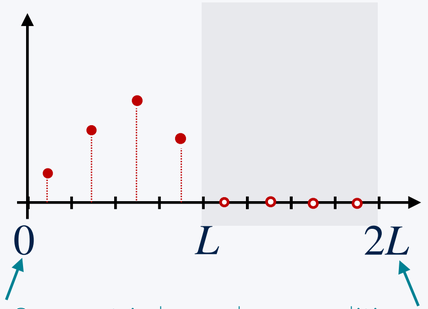


- *f is extended to a domain of 2N with 0 padding*
- *Discrete Fourier transform on the padded function*



Semi-unbounded

$f(x) \rightarrow \tilde{f}(k)$



- *f is extended to a domain of 2N with 0 padding*
- *Discrete sine/cosine transform on the padded function*
  → *DST/DCT is equivalent to impose symmetric boundary conditions on the new domain*

Very convenient for various problems:
-Inflow - outflow
- Wall bounded (wake in ground effects)

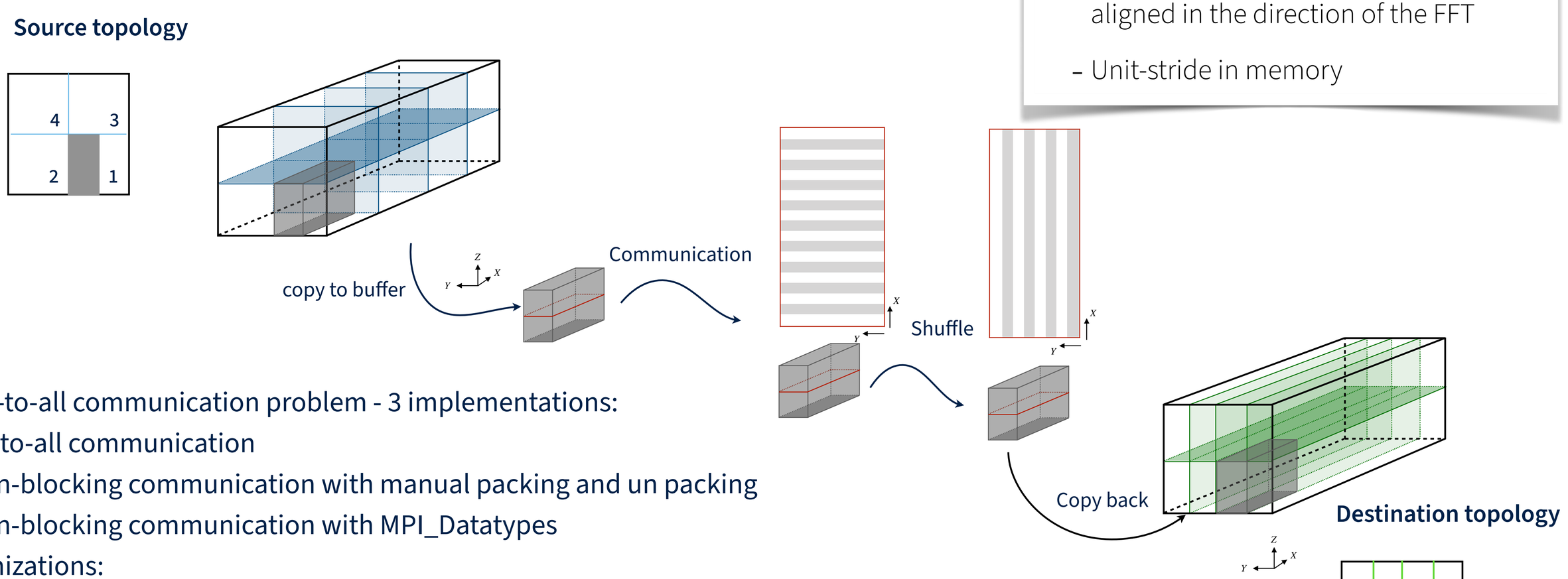
[2] Hockney:1988
[3] Caprace : 2021

3D FFT on massively parallel systems

- Pencil decomposition
- The topology switches:

Distributed 1D FFTs require:

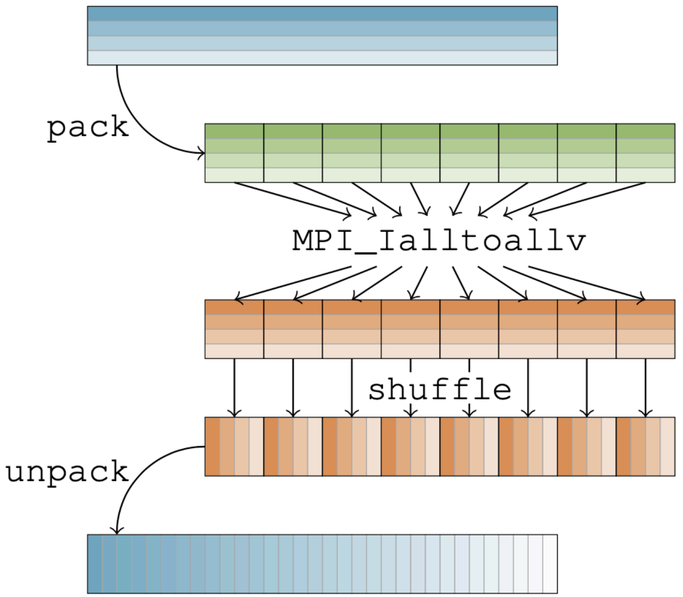
- All data on the same processor must be aligned in the direction of the FFT
- Unit-stride in memory



- An all-to-all communication problem - 3 implementations:
  - All-to-all communication
  - Non-blocking communication with manual packing and un packing
  - Non-blocking communication with MPI_Datatypes
- Optimizations:
  - Order of the 1D FFTs determined to reduce the memory footprint and the computational cost
  - Creation of sub-communicators to reduce the memory footprint on large partition

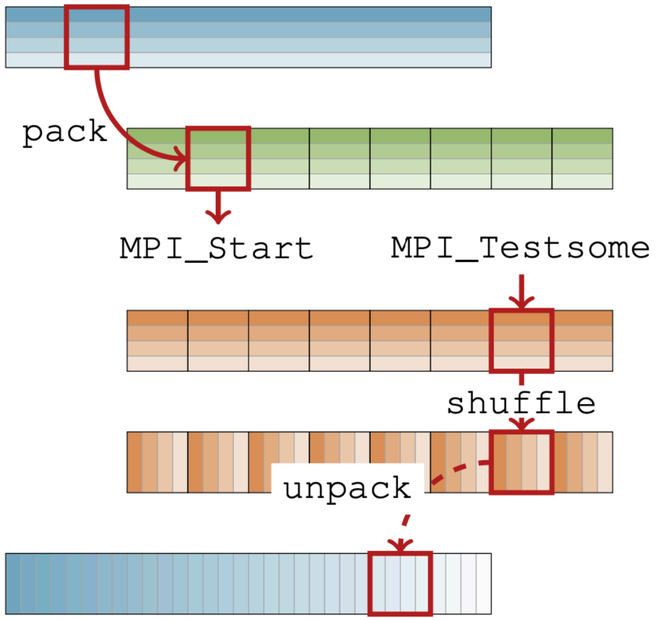
Switching between pencils - 3 implementations

All-to-all



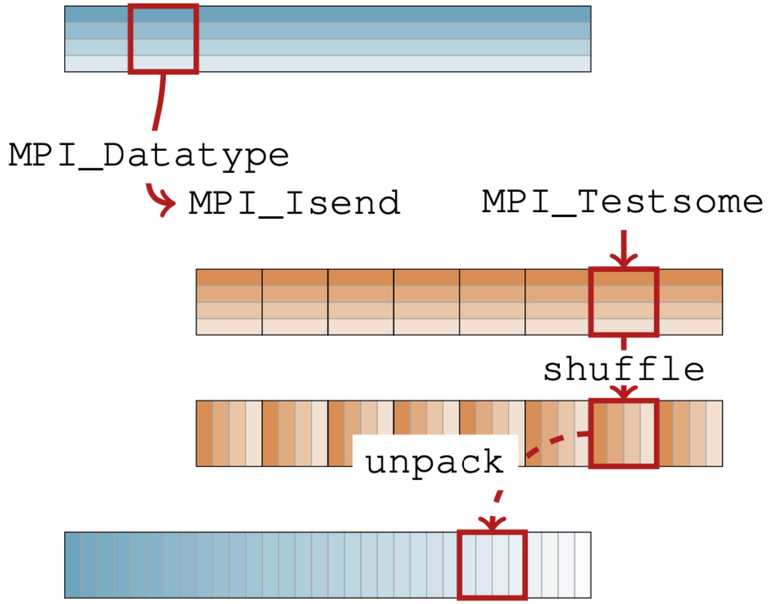
- **Simpler implementation** using MPI_Ialltoallv
- **Implicit barrier**
- Transposition of the data (based on FFTW)

Non-blocking with manual packing



- Use **persistent MPI_Send_Init, MPI_Recv_Init** and **MPI_Testsome**
- **Overlap** the **data packing** and the **transposition** with the **communication**
- Transposition of the data based on FFTW

Non-blocking with MPI_Datatype



- Use **non-blocking Send/Recv request**
- **Avoid manual packing** by using **MPI_Datatype**
- **Overlap** the **data packing** and the **transposition** with the **communication**
- Transposition of the data based on FFTW

1D FFTs reordering

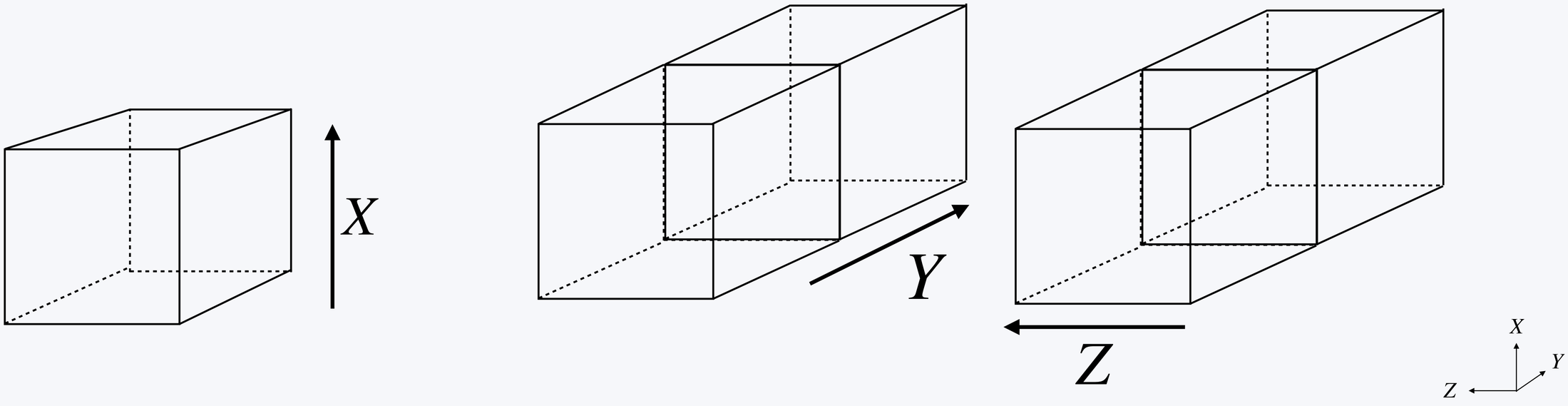
- 3D FFT obtained as a succession of 1D FFTs: $\psi_{x,y,z} \rightarrow \tilde{\psi}_{x,ky,z} \rightarrow \tilde{\psi}_{x,ky,kz} \rightarrow \hat{\psi}_{kx,ky,kz}$

Example 1 - (Periodic, Unbounded, Periodic)

Three 1D FFTs have to be performed:

Without rendering of the transform $X \rightarrow Y \rightarrow Z$:

1) $\psi_{x,y,z} \rightarrow \tilde{\psi}_{k_x,y,z}$ A real-to-complex DFT
2) $\tilde{\psi}_{k_x,y,z} \rightarrow \tilde{\psi}_{k_x,k_y,z}$ A complex-to-complex DFT on an extended and padded domain
3) $\tilde{\psi}_{k_x,k_y,z} \rightarrow \hat{\psi}_{k_x,k_y,k_z}$ A complex-to-complex DFT (on an extended domain)



→ FFT in Z costs $2N^3 log(N)$

1D FFTs reordering

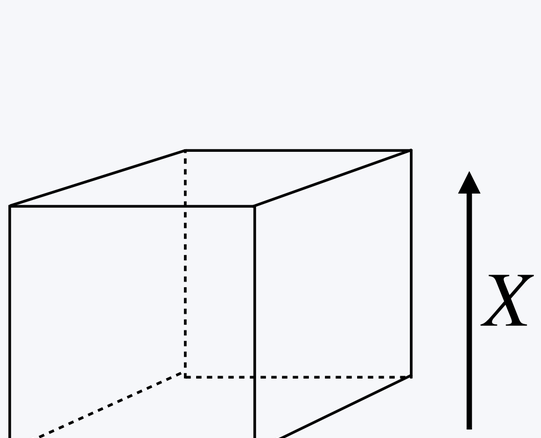
- 3D FFT obtained as a succession of 1D FFTs: $\psi_{x,y,z} \rightarrow \tilde{\psi}_{x,ky,z} \rightarrow \tilde{\psi}_{x,ky,kz} \rightarrow \hat{\psi}_{kx,ky,kz}$

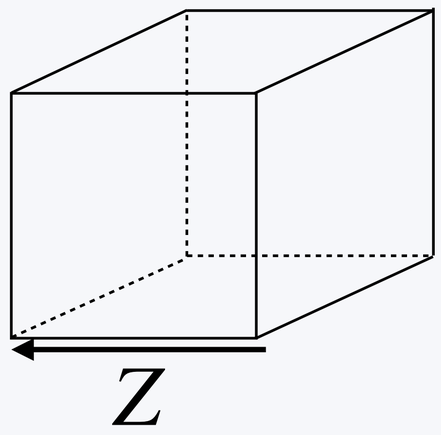
Example 1 - (Periodic, Unbounded, Periodic)

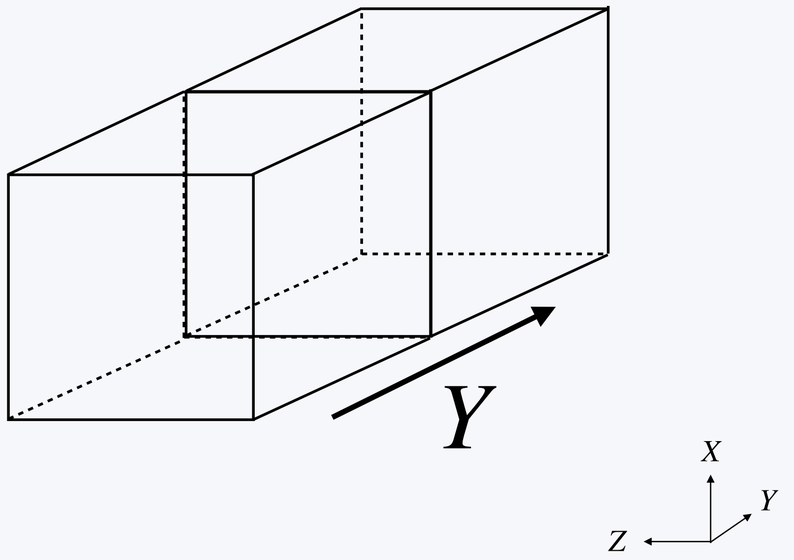
Three 1D FFTs have to be performed:

With rendering of the transform $X \rightarrow Z \rightarrow Y$:

1) $\psi_{x,y,z} \rightarrow \tilde{\psi}_{k_x,y,z}$ A real-to-complex DFT
2) $\tilde{\psi}_{k_x,y,z} \rightarrow \tilde{\psi}_{k_x,y,k_z}$ A complex-to-complex DFT
3) $\tilde{\psi}_{k_x,y,k_z} \rightarrow \hat{\psi}_{k_x,k_y,k_z}$ A complex-to-complex DFT on an extended domain







→ FFT in Z costs $N^3 log(N)$

Applications - Biot-Savart solver: Electromagnetism, Fluid mechanics,…

$$\nabla^2 \mathbf{u} = -\nabla \times \omega$$

Methodology

I. Forward transform of the rhs: $\omega \rightarrow \hat{\omega}$

II. Computation of the curl in the spectral space

III. Multiplication with the spectral Green's function $\hat{G}$

Testcase

- Compact vortex tube: $\omega(x,y,z) = \left\{0,0,-\omega_z(r)\right\}$

with $\omega_z(r) = \begin{cases} \frac{1}{2\pi}\frac{2}{R^2}\frac{1}{\mathrm{E}_2(1)}\exp\left(-\frac{1}{\left(1-\left(\frac{r^2}{R}\right)\right)}\right) & \text{If } r <= R \\ 0 & \text{otherwise} \end{cases}$

- Cubic domain of size $[0,L]^3$
- X and Y: fully unbounded
- Z: periodic

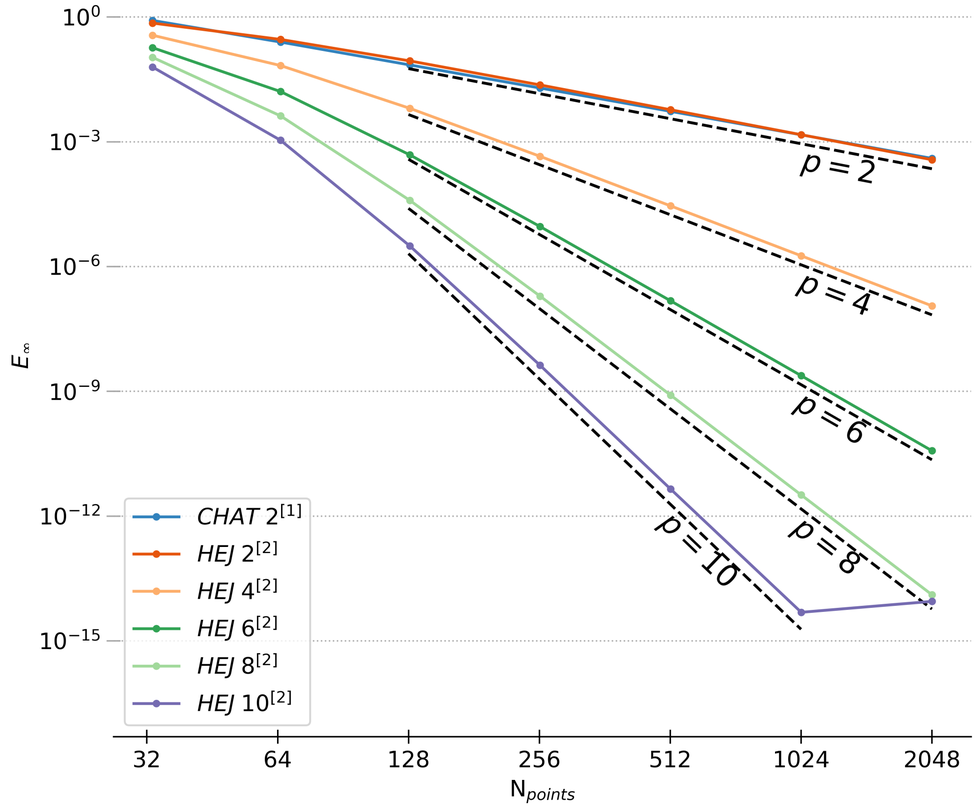
- Corresponding analytical velocity:

$$\mathbf{u}(x,y,z) = \left\{-\sin(\theta)u_\theta(r),\ \cos(\theta)u_\theta(r),\ 0\right\}$$

with $u_\theta(r) = \begin{cases} \frac{1}{2\pi r}\left[1-\frac{1}{\mathrm{E}_2(1)}\left(1-\left(\frac{r^2}{R}\right)\right)\mathrm{E}_2\left(\frac{1}{1-\left(\frac{r^2}{R}\right)}\right)\right] & \text{If } r <= R \\ \frac{1}{2\pi r} & \text{otherwise} \end{cases}$

$$E_\infty = \sup_{x,y,z}\{\,|\,\mathbf{u}(x,y,z) - \mathbf{u}_{ref}(x,y,z)\,|\,\}$$

Infinite norm of the error



[1] Green's function from Chatelain et al., 2010

[2] Green's function from Hejlesen et al., 2013

Results: Comparison with accFFT[4]

Comparison with accFFT, one of the fastest distributed FFT libraries on CPU[5]

Testcase

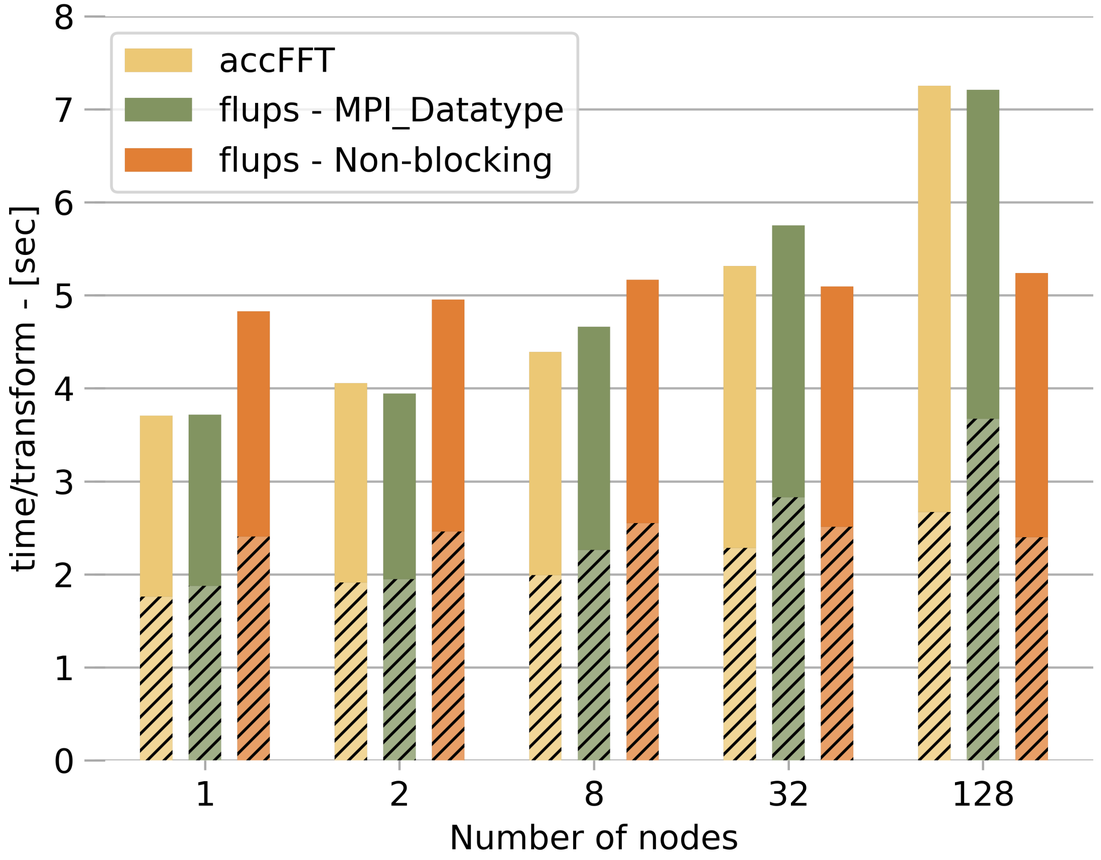
- Weak scaling
- Fully periodic
- Rectangular domain
- unknowns per rank: 256^3

accFFT

- Pencils are aligned in the Z direction
- Does $Z \rightarrow Y \rightarrow X$
- Opt. Flag: ACCFFT_MEASURE

Flups

- Pencils are aligned in the X direction
- Does $X \rightarrow Y \rightarrow Z$
- Opt. Flag: FFTW_MEASURE



MeluXina:

- *CPU:* AMD EPYC 7H12
- *Interconnect:* 200 Gb/s Infiniband HDR
- *MPI:* MPICH 4.1a1
- *Transport Layer:* UCX 1.13.1

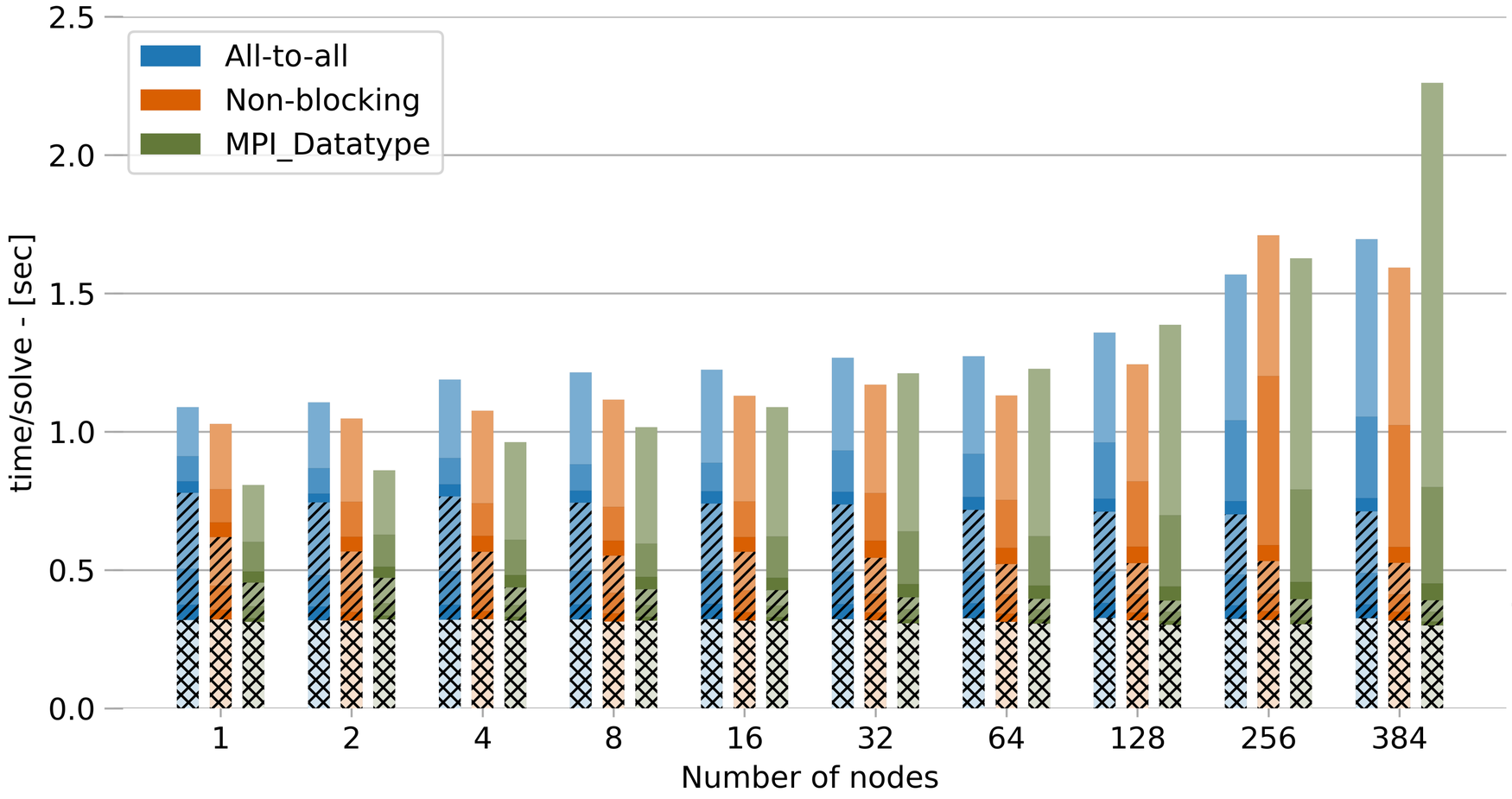
	Px	Py	Pz
1	1	8	16
2	1	16	16
8	1	32	32
32	1	64	64
128	1	128	128

→ At 128 nodes, non-blocking version of flups is 27% faster than accFFT

[4] Gholami:2015
[5]Ayala:2021a

Weak scaling - from 128 to 49,152 processes

Time-to-solution



MeluXina:

- *CPU:* AMD EPYC 7H12
- *Interconnect:* 200 Gb/s Infiniband HDR
- *MPI:* MPICH 4.1a1
- *Transport Layer:* UCX 1.13.1

Test case:

- Poisson equation in a fully unbounded domain
- 96^3 unknowns per rank
- From 1 to 384 nodes

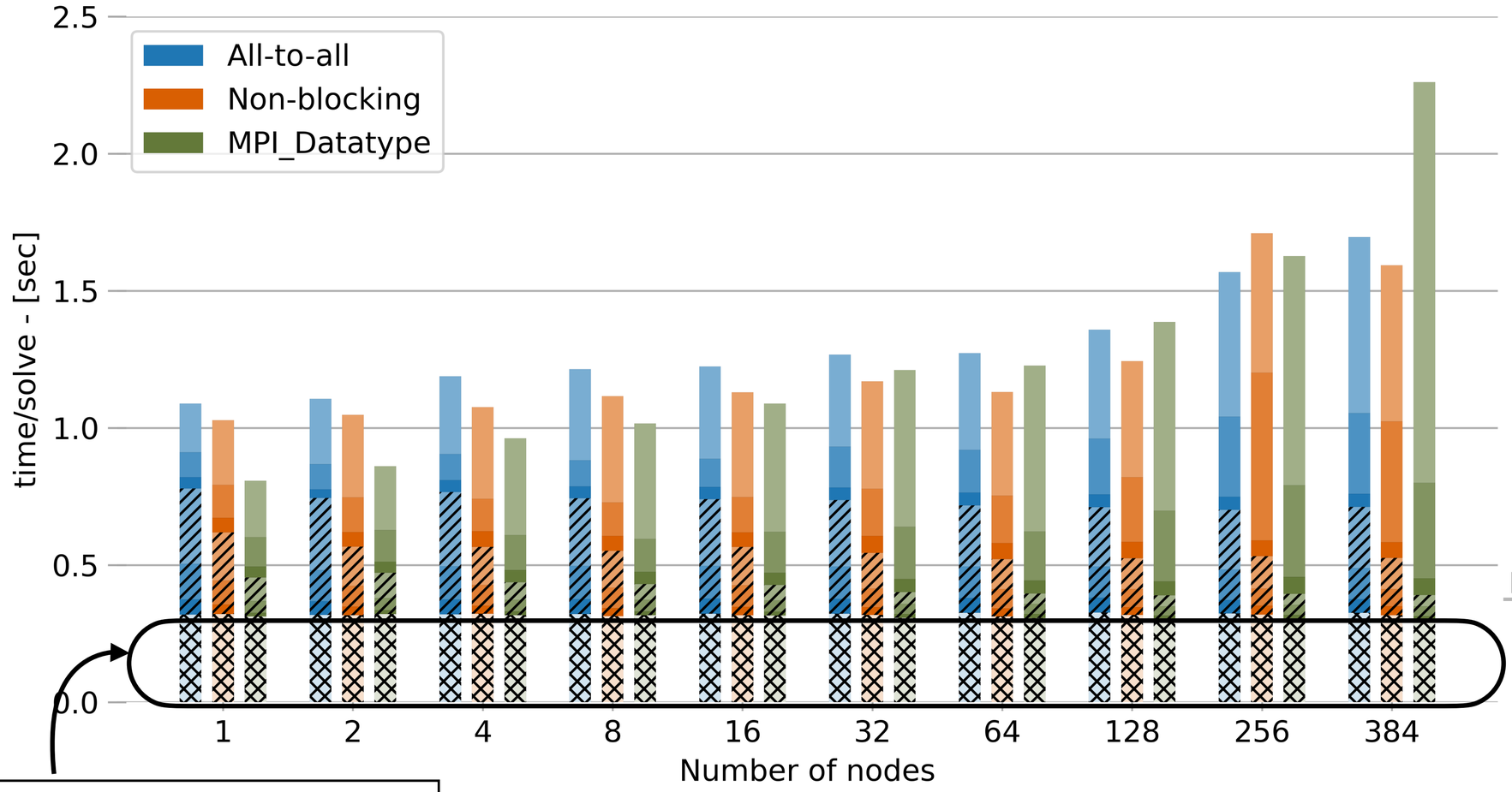
Process distribution

Nodes	Px	Py	Pz
1	4	4	8
2	4	8	8
4	8	8	8
8	8	8	16
16	8	16	16
32	16	16	16
64	16	16	32
128	16	32	32
256	32	32	32
384	32	32	48

12

Weak scaling - from 128 to 49,152 processes

Time-to-solution



Fixed computational cost: `ffts`, `mult`, `copy` of the rhs

- *CPU:*
- *Interconnect:* 200 Gb/s InfiniBand HDR
- *MPI:* MPICH 4.1a1
- *Transport Layer:* UCX 1.13.1

Test case:

- Poisson equation in a fully unbounded domain
- 96^3 unknowns per rank
- From 1 to 384 nodes

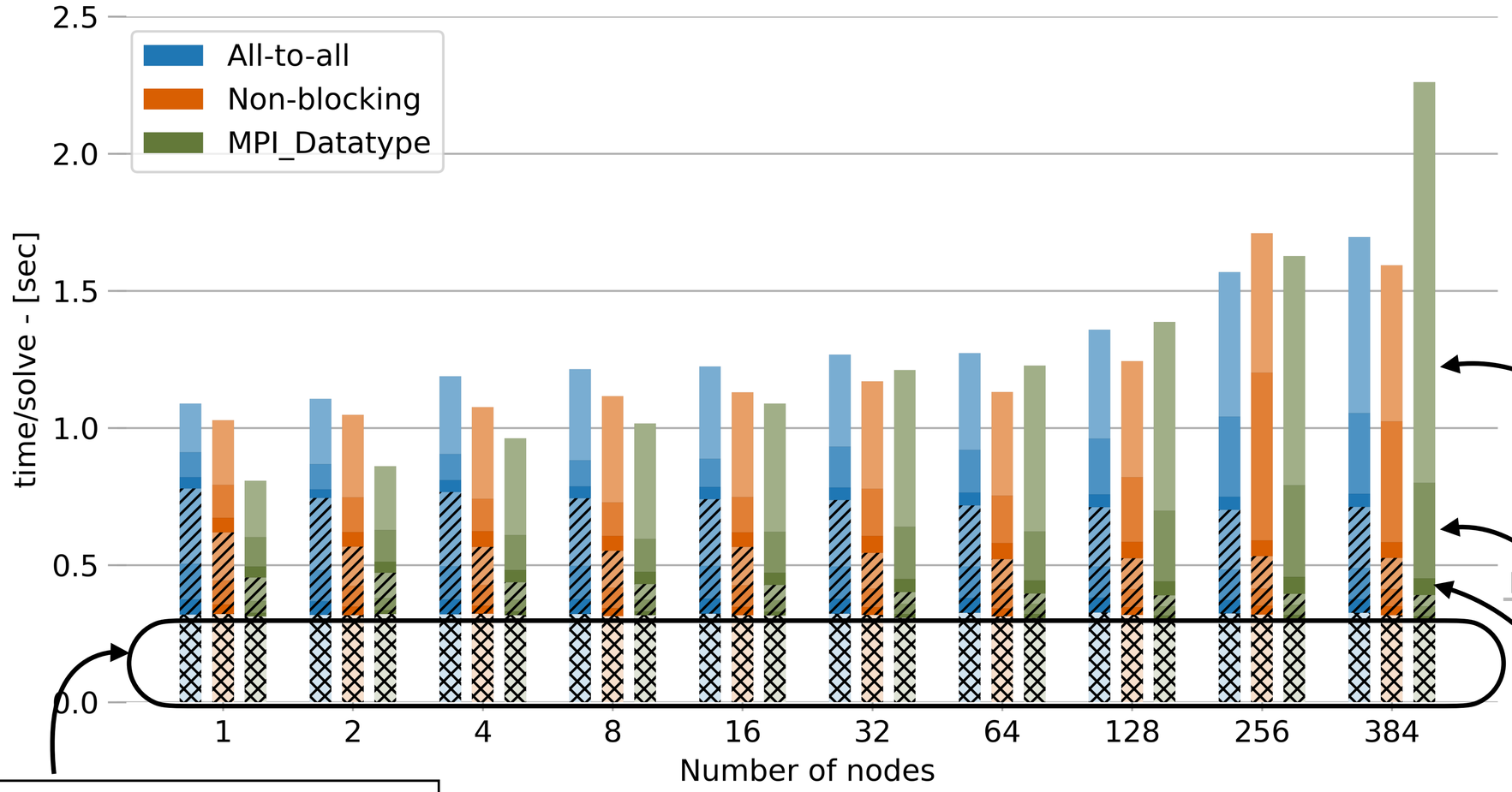
Process distribution

Nodes	Px	Py	Pz
1	4	4	8
2	4	8	8
4	8	8	8
8	8	8	16
16	8	16	16
32	16	16	16
64	16	16	32
128	16	32	32
256	32	32	32
384	32	32	48

12

Weak scaling - from 128 to 49,152 processes

Time-to-solution



Process distribution

Nodes	Px	Py	Pz
1	4	4	8
2	4	8	8
4	8	8	8
8	8	8	16
16	8	16	16
256	32	32	32
384	32	32	48

Fixed computational cost: `ffts`, `mult`, `copy` of the rhs

Unbounded domain:
cost of the topology switch increases

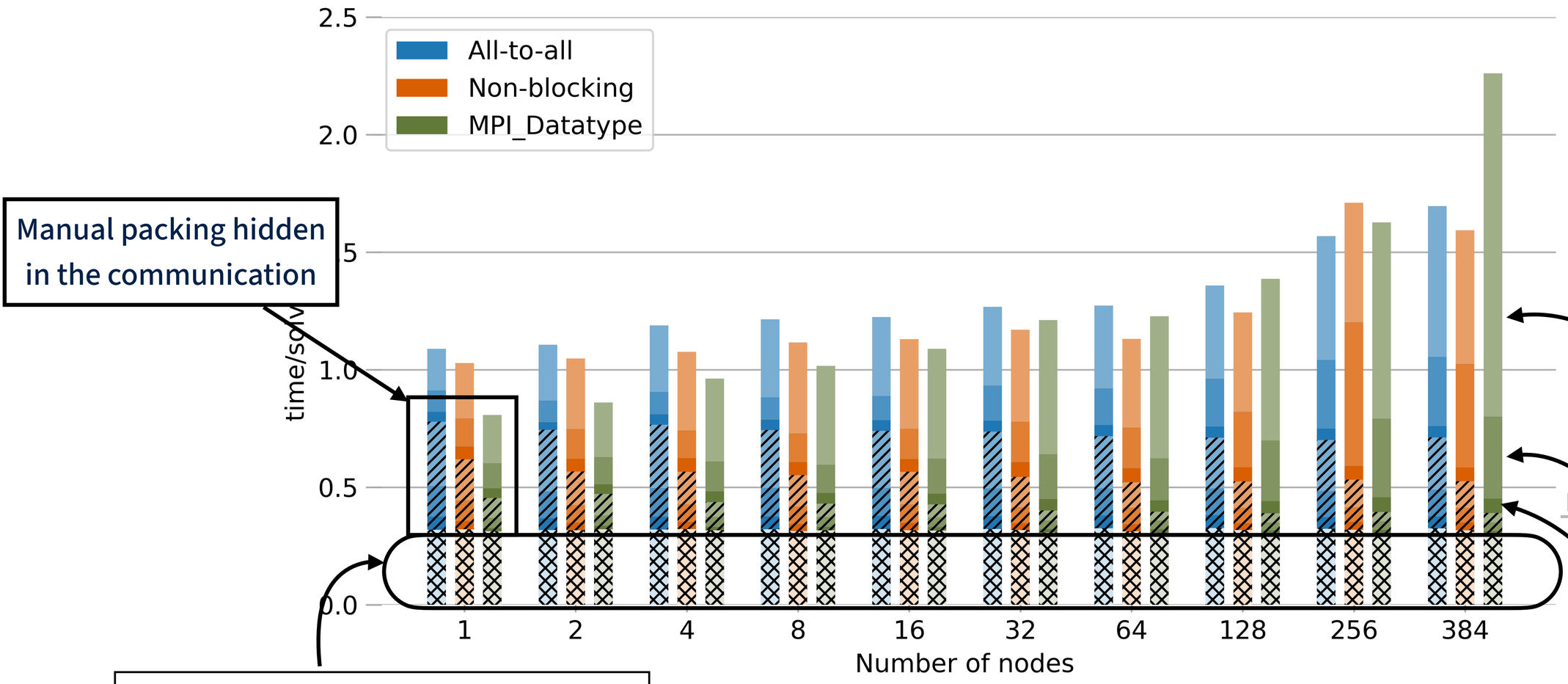
- *CPU:*
- *Interconnect:* 200 Gb/s InfiniBand HDR
- *MPI:* MPICH 4.1a1
- *Transport Layer:* UCX 1.13.1

Test case:

- Poisson equation in a fully unbounded domain
- 96^3 unknowns per rank
- From 1 to 384 nodes

Weak scaling - from 128 to 49,152 processes

Time-to-solution



Manual packing hidden in the communication

Fixed computational cost: `ffts`, `mult`, `copy` of the rhs

Process distribution

Nodes	Px	Py	Pz
1	4	4	8
2	4	8	8
4	8	8	8
8	8	8	16
16	8	16	16
256	32	32	32
384	32	32	48

Unbounded domain:
cost of the topology switch increases

- *CPU:*
- *Interconnect:* 200 Gb/s InfiniBand HDR
- *MPI:* MPICH 4.1a1
- *Transport Layer:* UCX 1.13.1

Test case:

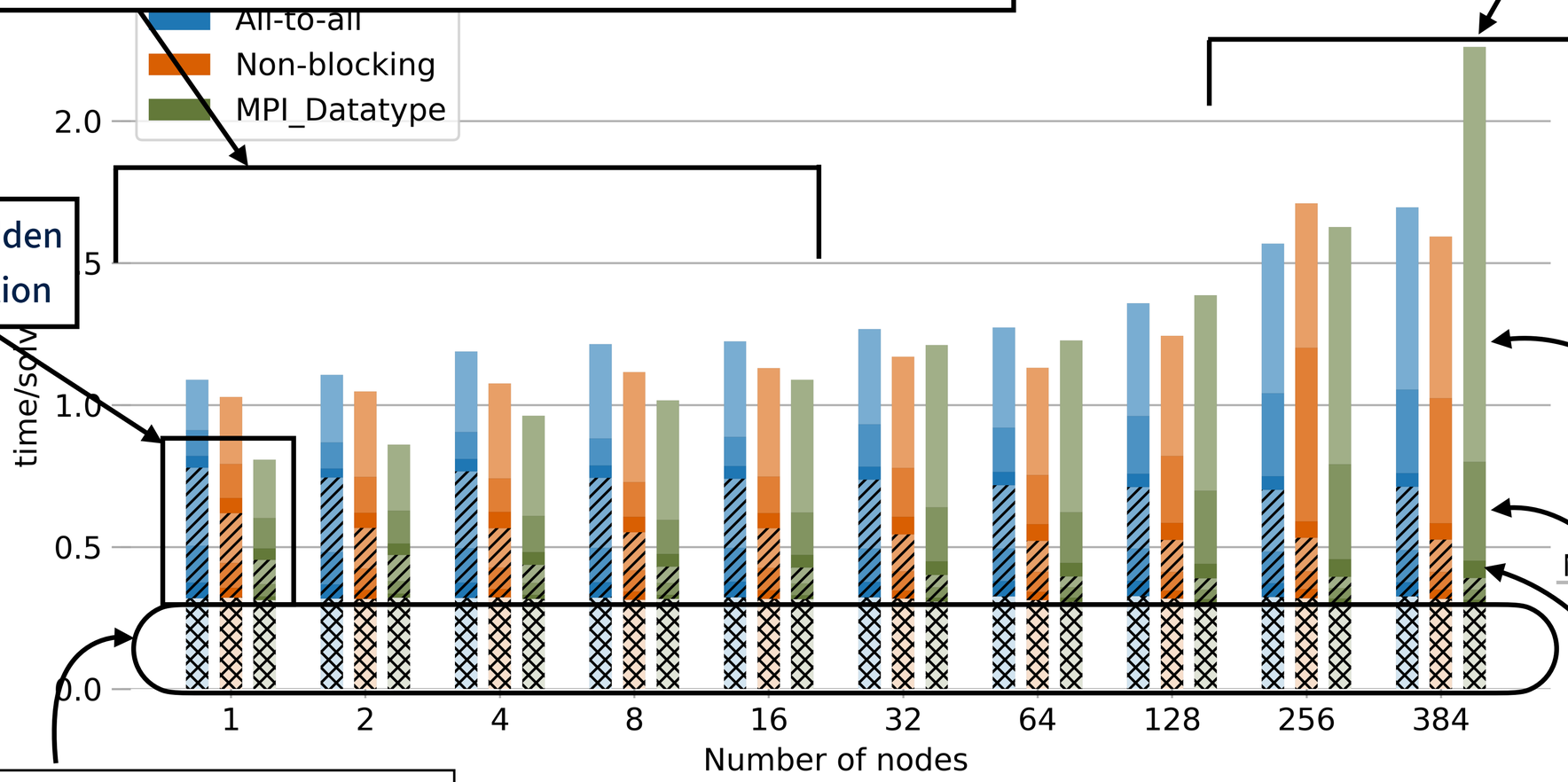
- Poisson equation in a fully unbounded domain
- 96^3 unknowns per rank
- From 1 to 384 nodes

Weak scaling - from 128 to 49,152 processes

Time-to-solution:
`MPI_Datatype > all to all > Non-Blocking`

Time-to-solution

Time-to-solution: `all-to-all > Non-Blocking > MPI_Datatype`



Manual packing hidden in the communication

Fixed computational cost: `ffts, mult, copy` of the rhs

Process distribution

Nodes	Px	Py	Pz
1	4	4	8
2	4	8	8
4	8	8	8
8	8	8	16
16	8	16	16
256	32	32	32
384	32	32	48

Unbounded domain:
cost of the topology switch increases

- *CPU:*
- *Interconnect:* 200 Gb/s InfiniBand HDR
- *MPI:* MPICH 4.1a1
- *Transport Layer:* UCX 1.13.1

Test case:

- Poisson equation in a fully unbounded domain
- 96^3 unknowns per rank
- From 1 to 384 nodes

Weak scaling - from 128 to 49,152 processes

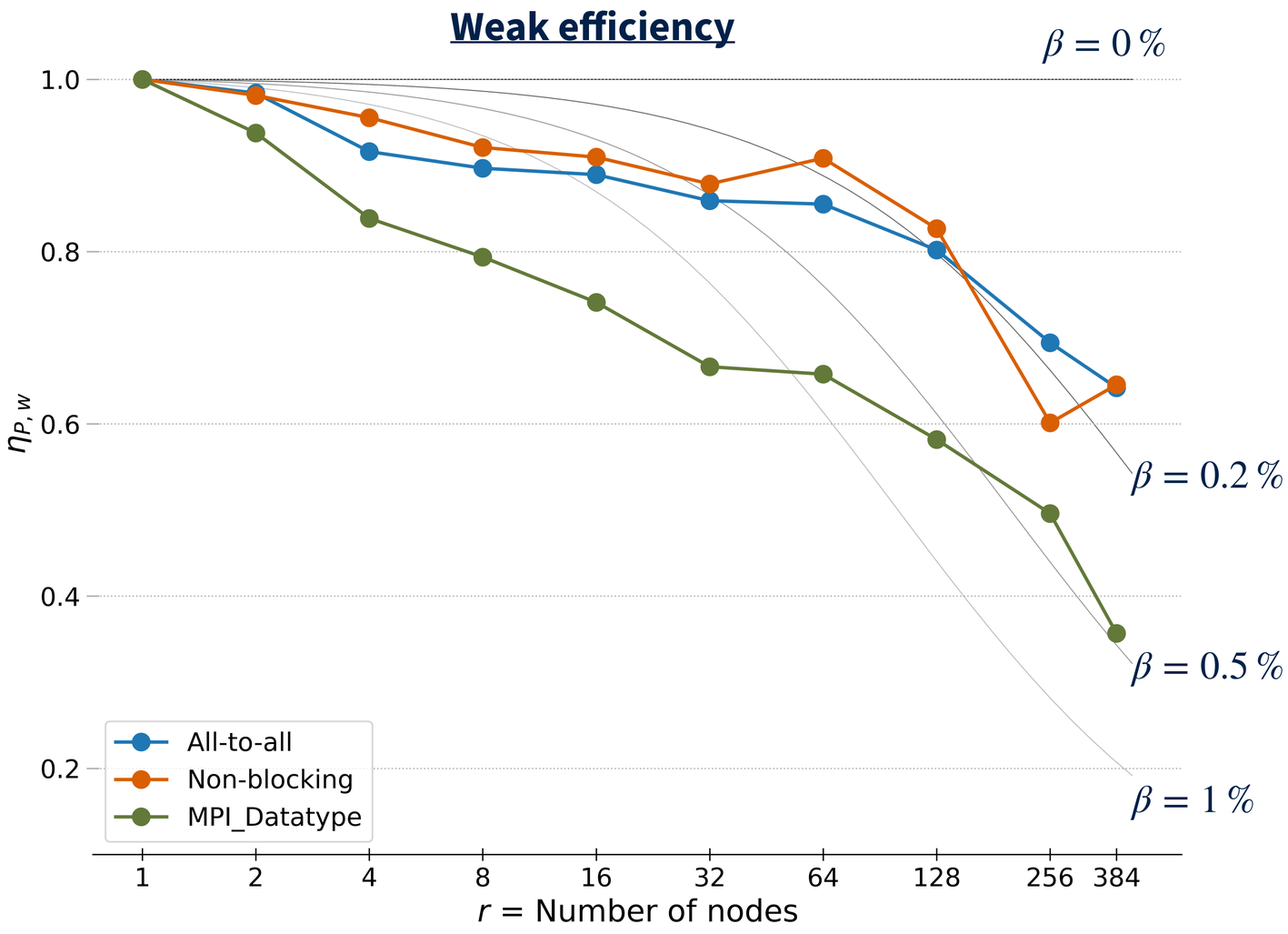
Gustafson's law:

efficiency η is

$$\eta_{P,w} = \frac{1}{1 + (r-1)\beta}$$

where:

- $r = N/N_0$ is the ratio of the computational resource
- β is the sequential percentage of the program



Weak scaling - from 128 to 49,152 processes

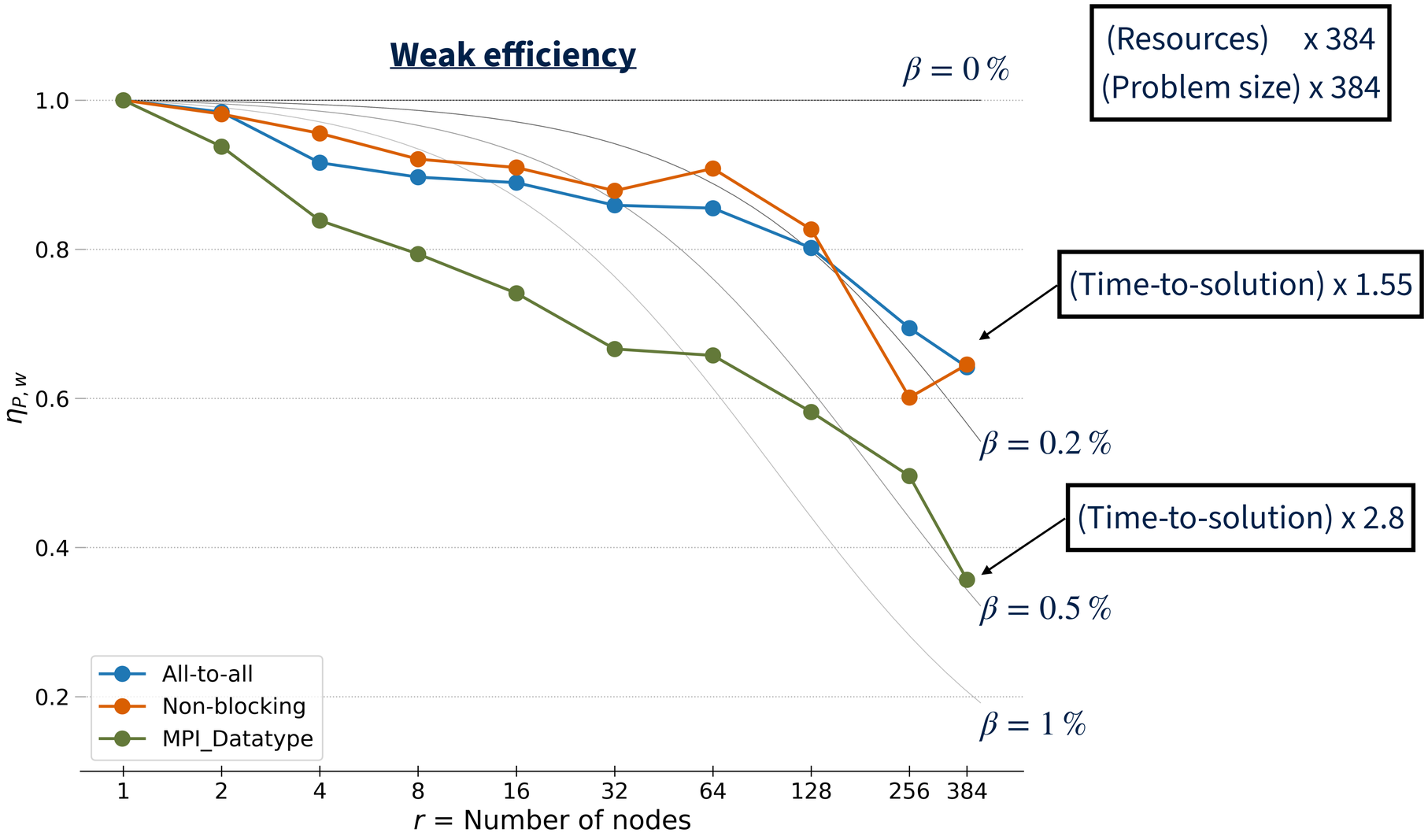
Gustafson's law:

efficiency η is

$$\eta_{P,w} = \frac{1}{1 + (r-1)\beta}$$

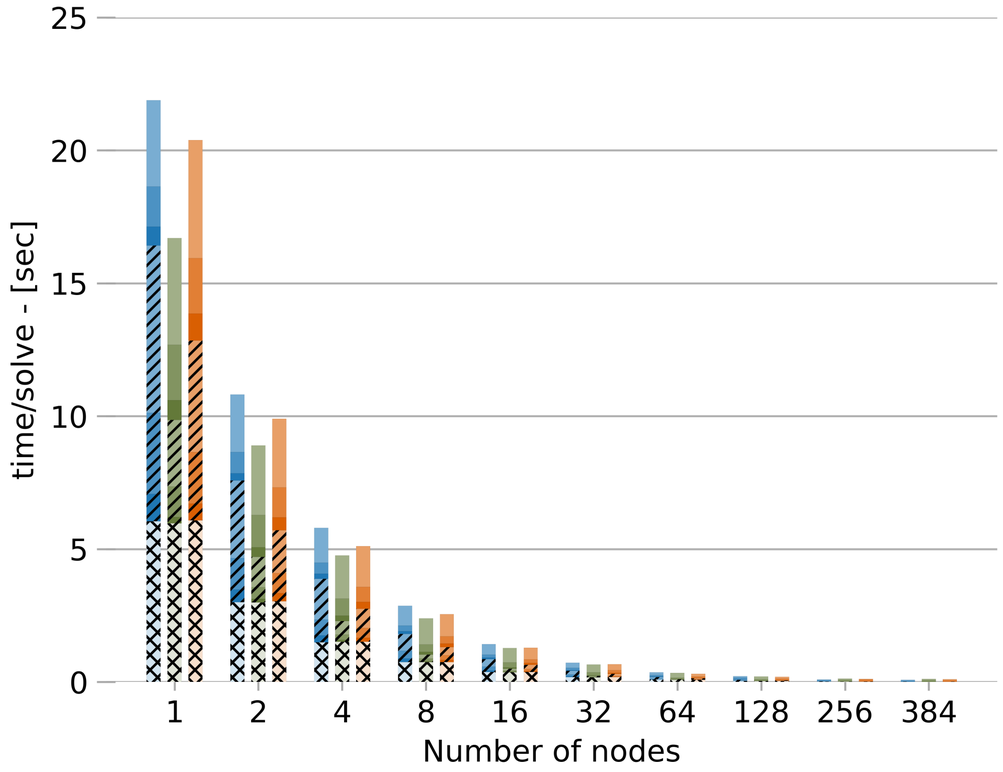
where:

- $r = N/N_0$ is the ratio of the computational resource
- β is the sequential percentage of the program

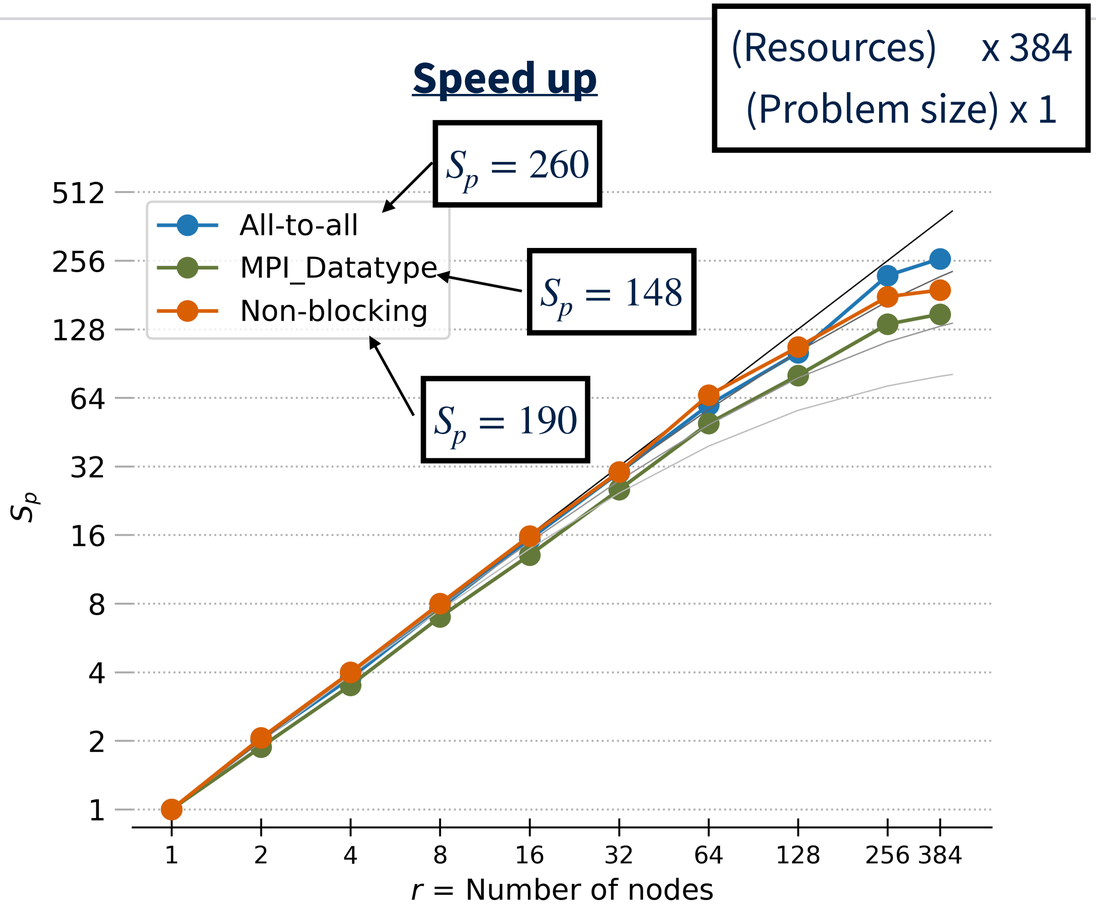


Strong scaling - from 128 to 49,152 processes

Time-to-solution



Speed up



MeluXina:

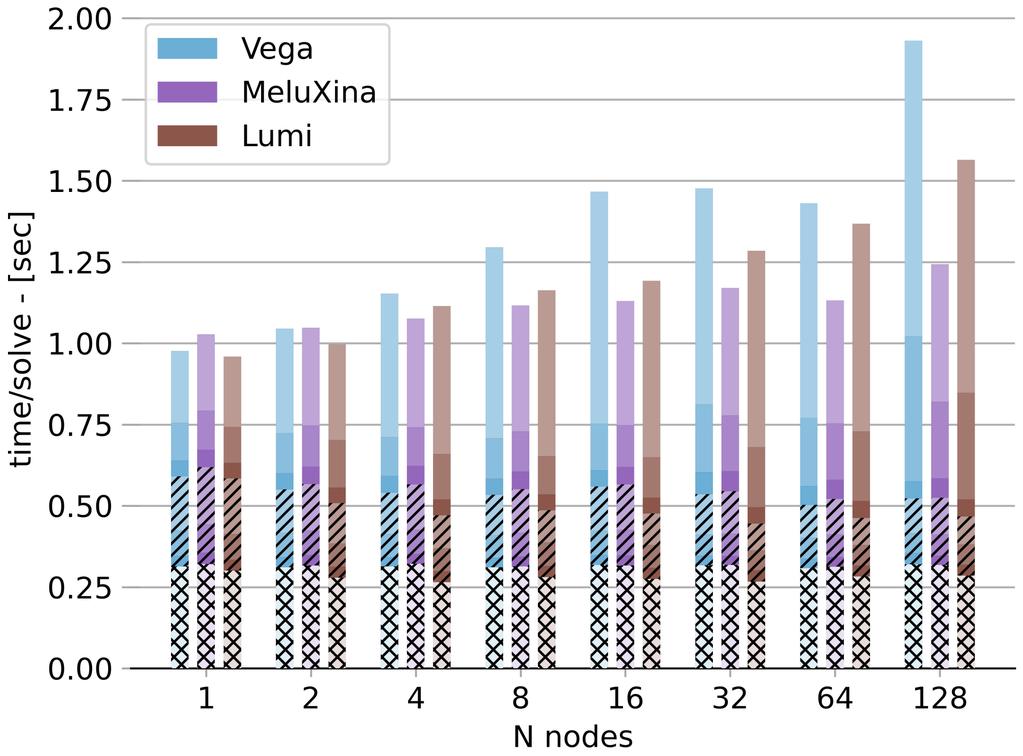
- *CPU:* AMD EPYC 7H12
- *Interconnect:* 200 Gb/s Infiniband HDR
- *MPI:* MPICH 4.1a1
- *Transport Layer:* UCX 1.13.1

Test case:

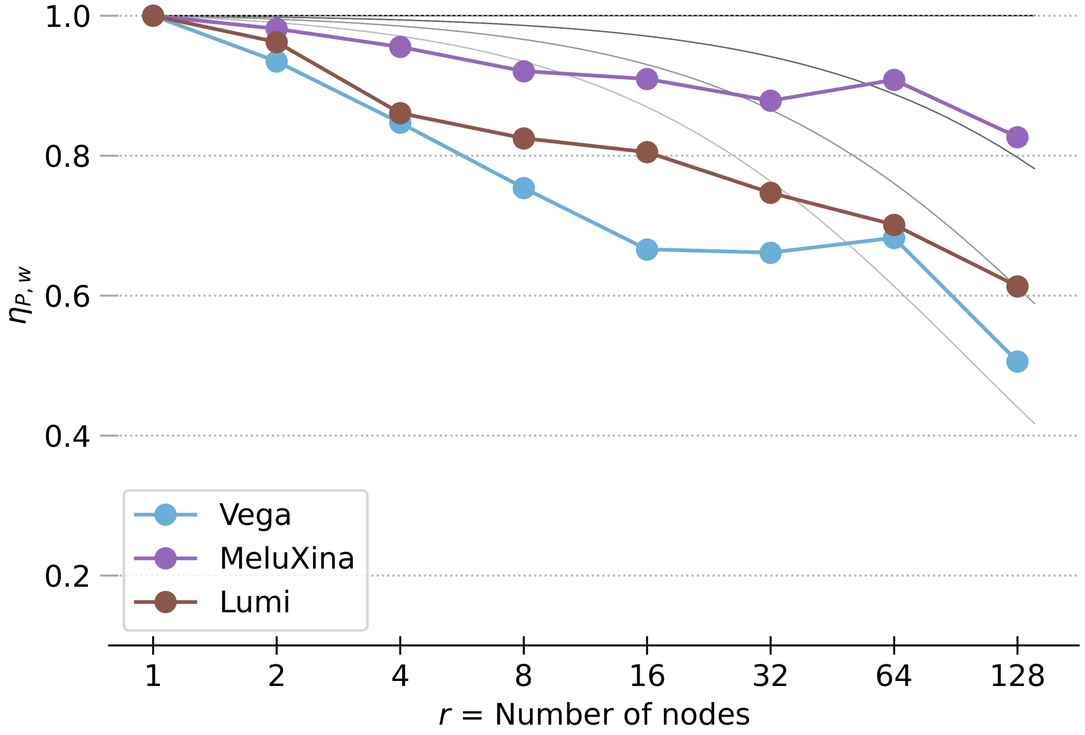
- Poisson equation in a fully unbounded domain
- 1281^3 unknowns
- From 1 to 384 nodes

European cluster comparison - Weak scaling test

Time-to-solution



Weak efficiency



Test case:

- Poisson equation in a fully unbounded domain
- 96^3 unknowns per rank
- From 1 to 128 nodes

Name	Location	CPU	Interconnect	Transport Layer	OSU Latency
Lumi	Finland	AMD EPYC 7763	200 Gb/s Slingshot-11	`Libfabric 15.0.0`	2.05 us
MeluXina	Luxembourg	AMD EPYC 7H12	200 Gb/s Infiniband HDR	ucx 1.13.1	1.45 us
Vega	Slovenia	AMD EPYC 7H12	100 Gb/s Infiniband HDR	ucx 1.13.1	1.99 us

Conclusions

Flups offers highly efficient distributed FFT-based Poisson solvers

Methodology

- Handle arbitrary cartesian topology
- 1000 combinations of boundary conditions
- 8 different Green's function
- 2 data layouts

Parallel performance

- Faster time-to-solution compared to accFFT on large FFTs
- Weak and Strong scalability with a parallel percentage $\approx 96\% - 98\,\%$
- Scalability on three EuroCC systems

Thanks for your attention!

Any questions?

- *FLUPS* - a flexible and performant massively parallel Fourier transform library, Balty et al., IEEE - Transactions on Parallel and Distributed Systems, 2023 (*forthcoming*)

- *FLUPS - A Fourier-based Library of Unbounded Poisson Solvers*, Caprace et al., SIAM Journal on Scientific Computing, vol. 43, no. 1, pp. C31–C60, January 2021

- https://github.com/vortexlab-uclouvain/flups