Towards an efficient implementation of the Particle-Mesh-Ewald (PME) method on low-bandwidth linux clusters

Workshop on Fast Algorithms for Long-Range Interactions
Forschungszentrum Jülich, 7 - 8 April 2005

Carsten Kutzner

ckutzne@gwdg.de

Max-Planck-Institute for Biophysical Chemistry
Theoretical and Computational Biophysics Department
Göttingen

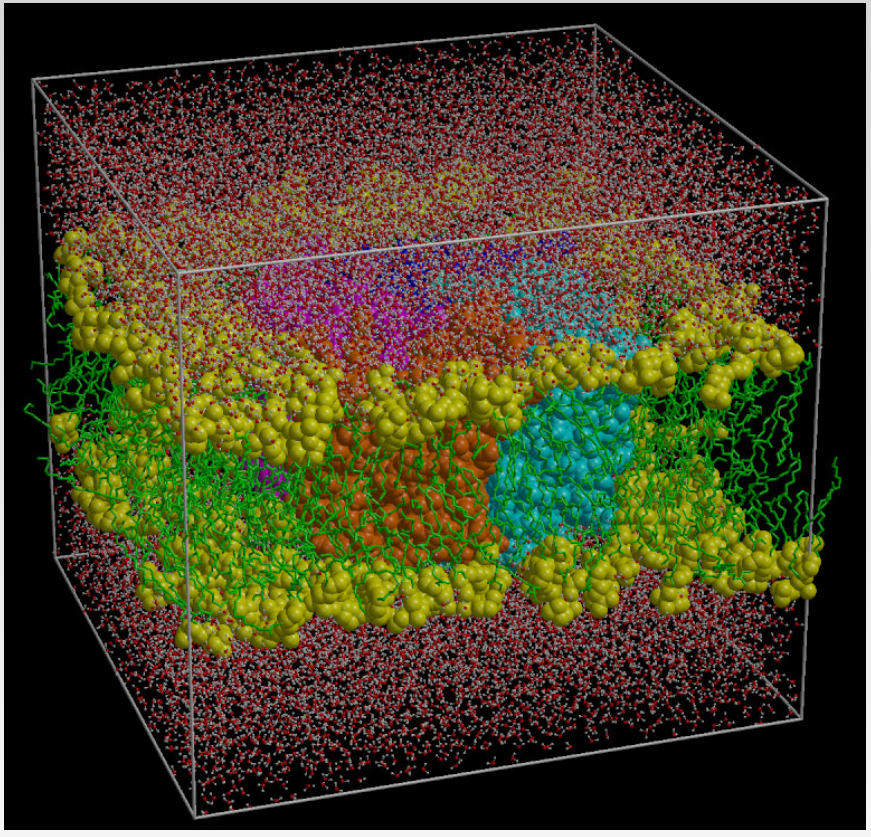
Goal of project

better performance of GROMACS on parallel machines

- David van der Spoel, Uppsala University (GMX developer)
- Erik Lindahl, Stockholm (GMX developer)
- Jakob Pichlmeier, IBM Munich (**domain decomposition**)
- Renate Dohmen, RZ Garching (**load balancing**)
- Carsten Kutzner (**PME/PP node splitting**)

Molecular dynamics simulations

- molecular dynamics (MD) simulations of proteins in water
- 1 000 – 1 000 000 atoms
- GROMACS 3.2.1 MD simulation package
- **long-range electrostatic forces** are evaluated with Particle-Mesh-Ewald (PME)
- **Aquaporin-1**, $\approx$ 80 000 atoms, protein (tetramer) embedded in a lipid bilayer membrane surrounded by water



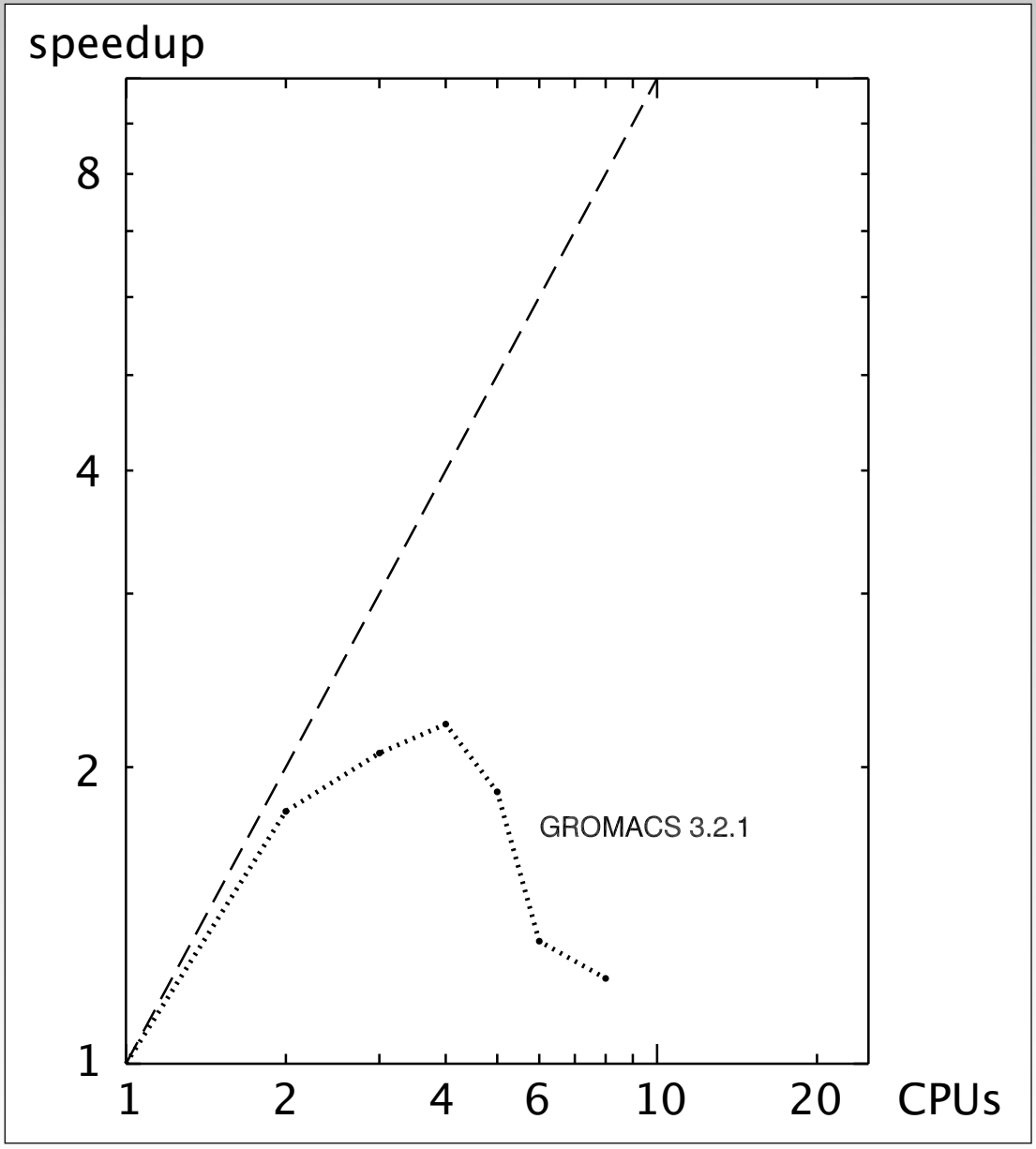
Speedup of GROMACS 3.2.1

- N Intel Xeon CPUs 3.06GHz (Orca1)
- LAM 7.1.1 MPI
- Gigabit Ethernet
- MPI_Wtime hi-res t counter
- time step length variation typ. ≈ 5 %
- 9 step average

$$\text{speedup} = t_1/t_N$$

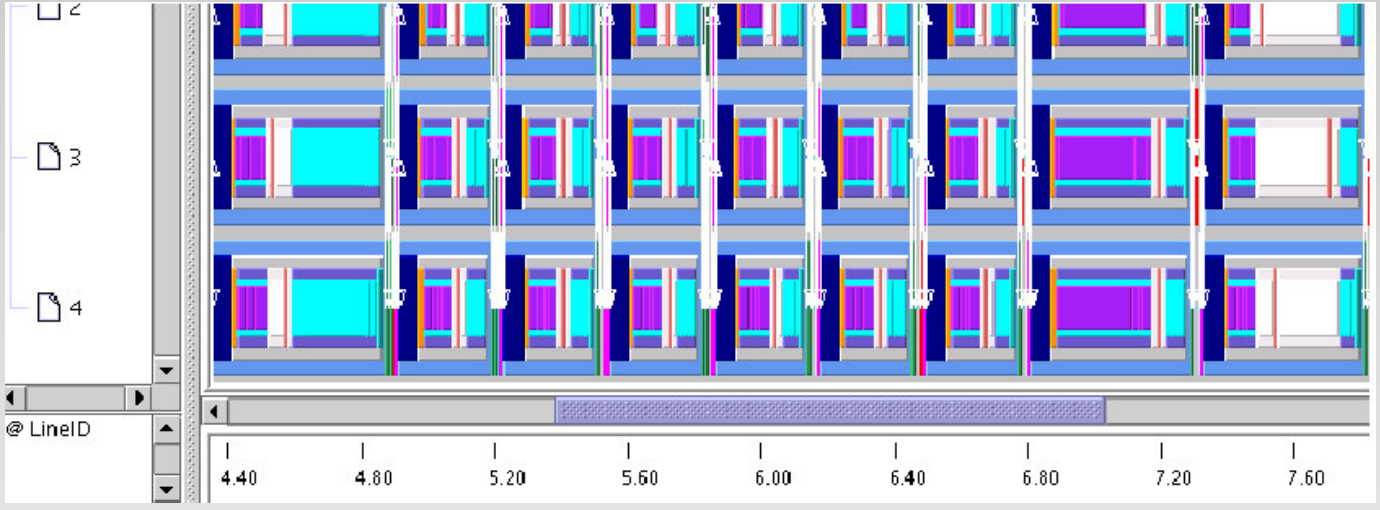
$$\text{scaling} = \text{speedup}\ /N$$

- **Max. speedup: 2.2 @ 4 CPUs, scaling 0.55**



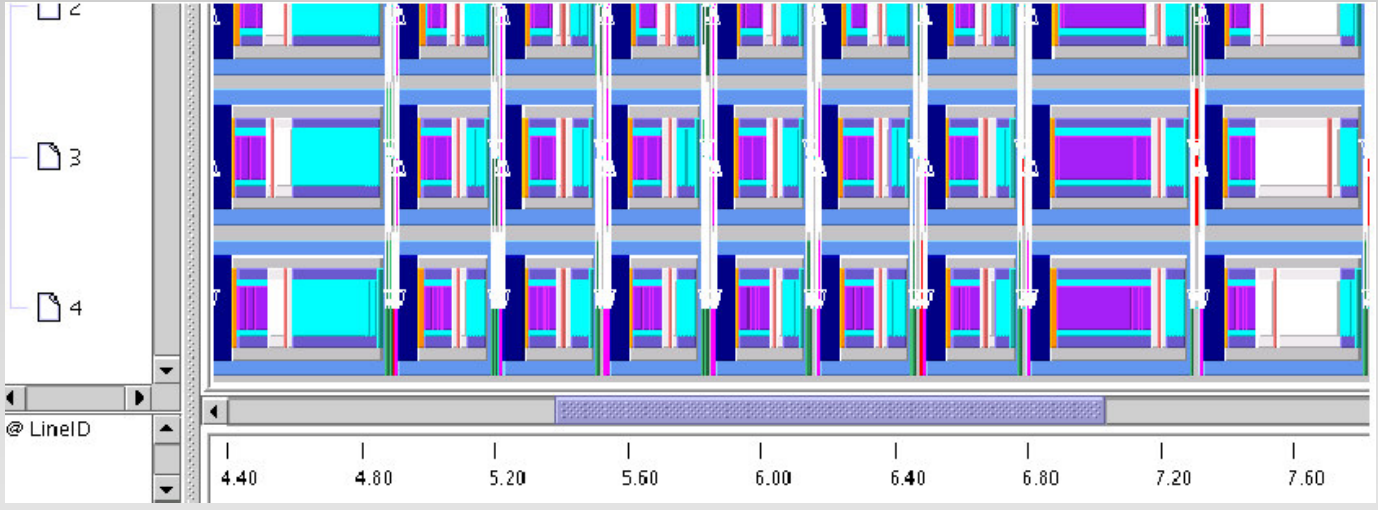
Switch congestion

Default switch settings: switch congestion prevents good scaling

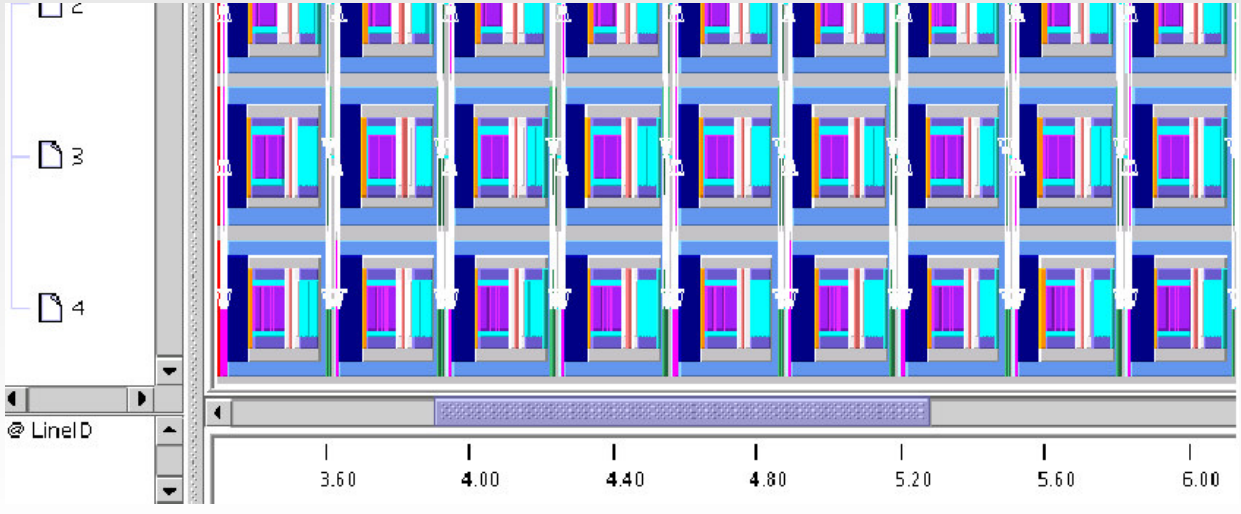


Switch congestion

Default switch settings: switch congestion prevents good scaling

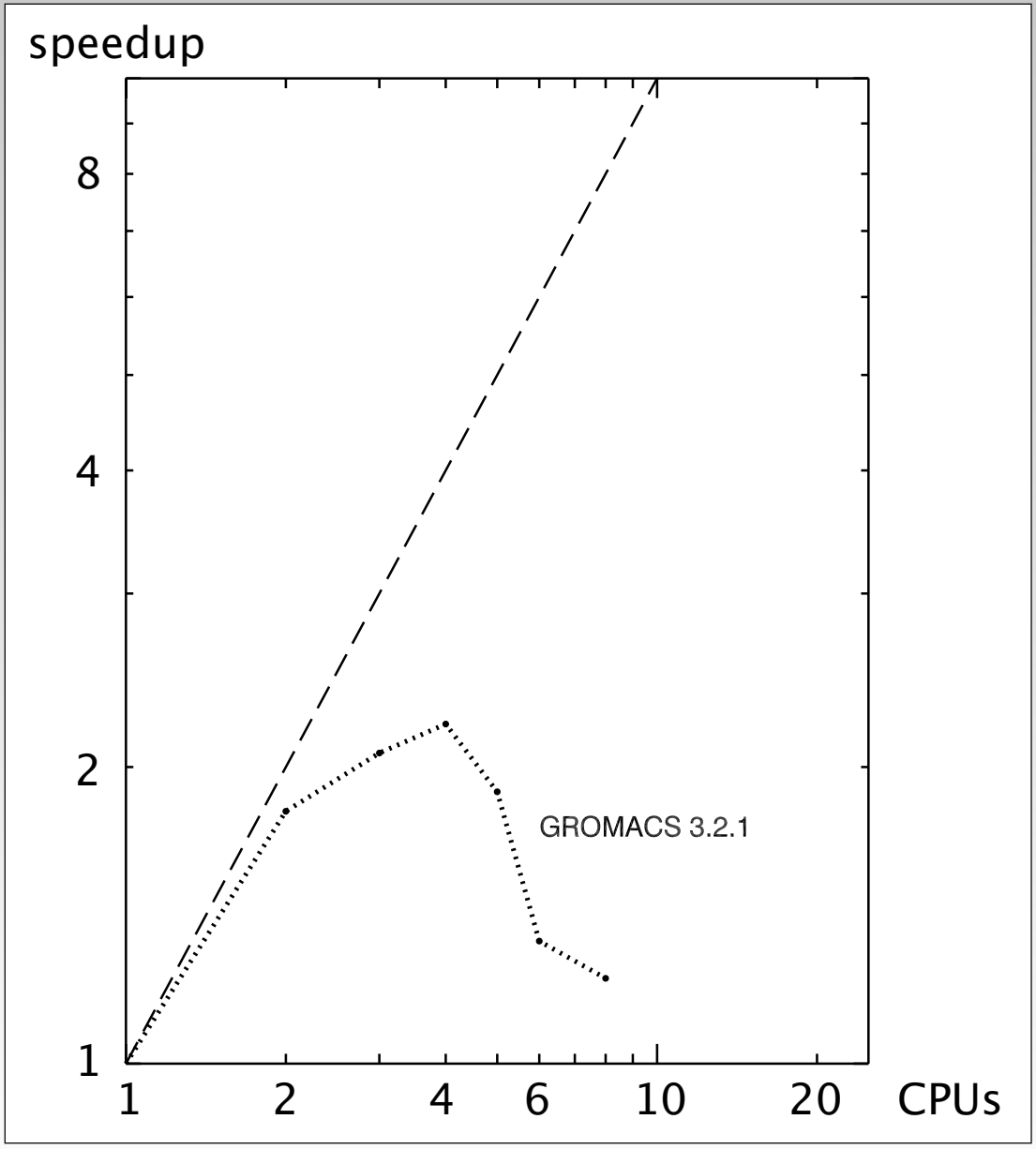


With switch flow control: no lost messages any more



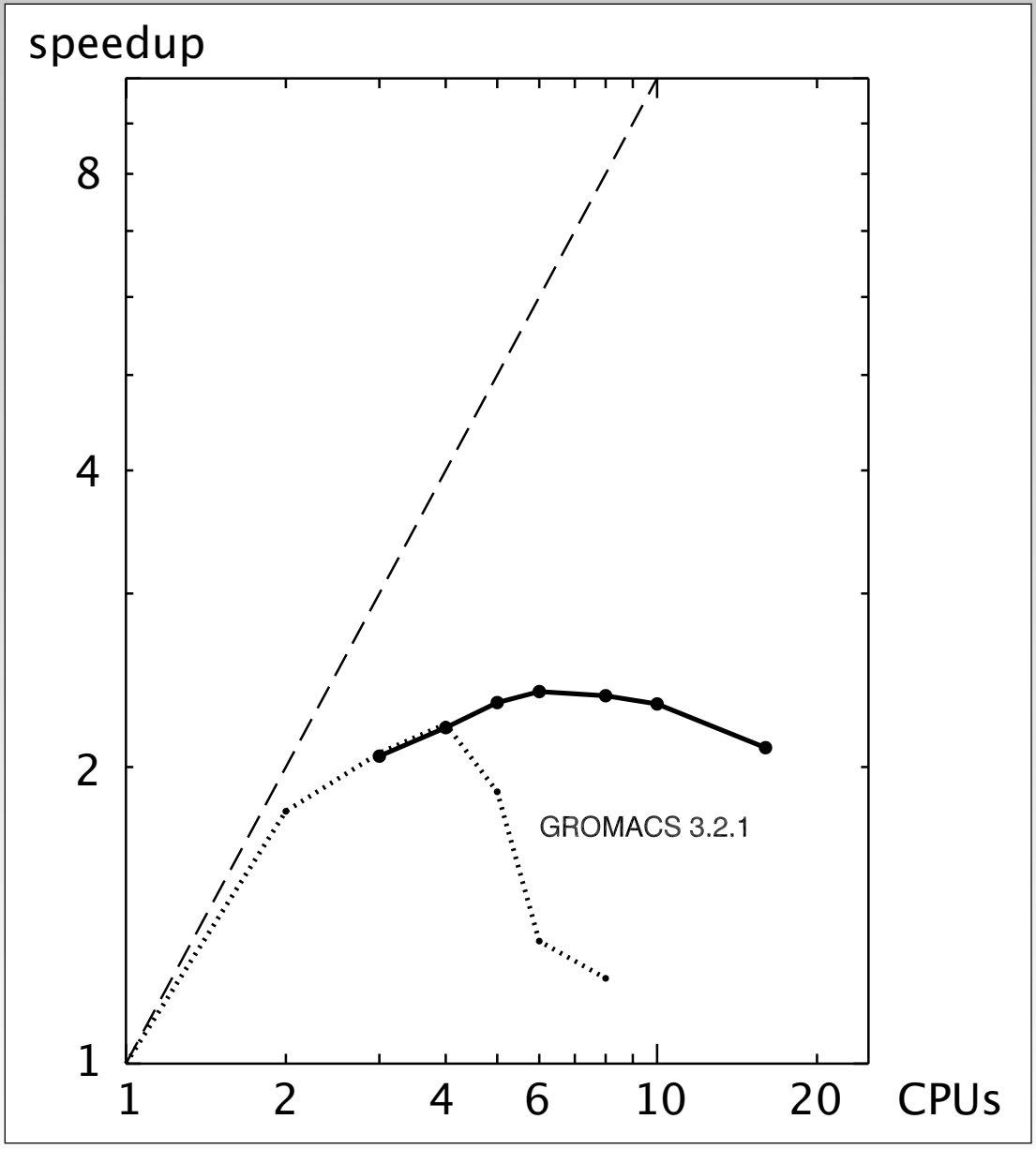
Speedup of GROMACS 3.2.1

- default switch settings
- Max. speedup:
  2.2 @ 4 CPUs
- scaling (4 CPUs):
  0.55



Speedup of GROMACS 3.2.1

- with **flow control**:
- Max. speedup:
  2.4 @ 6 CPUs
- scaling (6 CPUs):
  0.39

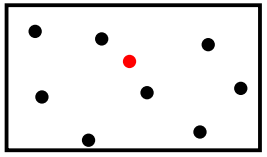


Particle Mesh Ewald

Coulomb forces on N particles, charges q_i, positions $\mathbf{r}_i$, box length L, periodic b.c.

- electrostatic potential

$$V = \frac{1}{2} \sum_{i,j=1}^{N} {\sum_{\mathbf{n}\in\mathbb{Z}^3}}' \frac{q_i q_j}{|\mathbf{r}_{ij} + \mathbf{n}L|}$$

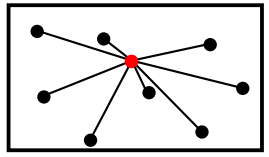


Particle Mesh Ewald

Coulomb forces on N particles, charges q_i, positions $\mathbf{r}_i$, box length L, periodic b.c.

- electrostatic potential

$$V = \frac{1}{2} \sum_{i,j=1}^{N} {\sum_{\mathbf{n}\in\mathbb{Z}^3}}' \frac{q_i q_j}{|\mathbf{r}_{ij} + \mathbf{n}L|}$$



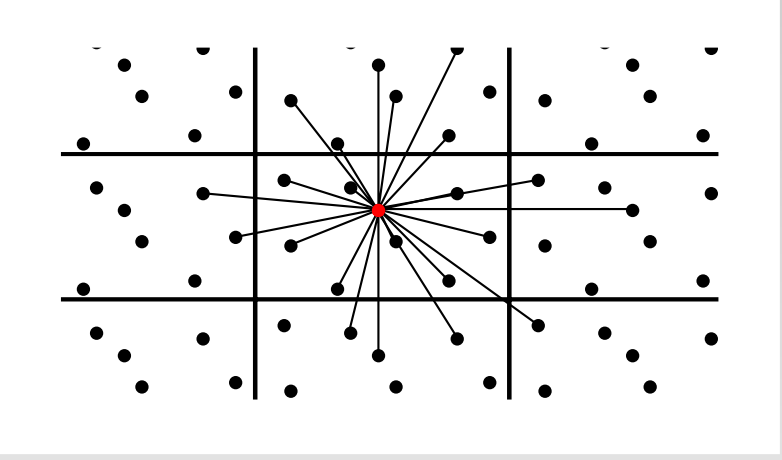
Particle Mesh Ewald

Coulomb forces on N particles, charges q_i, positions $\mathbf{r}_i$, box length L, periodic b.c.

- electrostatic potential

$$V = \frac{1}{2} \sum_{i,j=1}^{N} {\sum_{\mathbf{n}\in\mathbb{Z}^3}}' \frac{q_i q_j}{|\mathbf{r}_{ij} + \mathbf{n}L|}$$

straightforward summation impracticable!



Particle Mesh Ewald

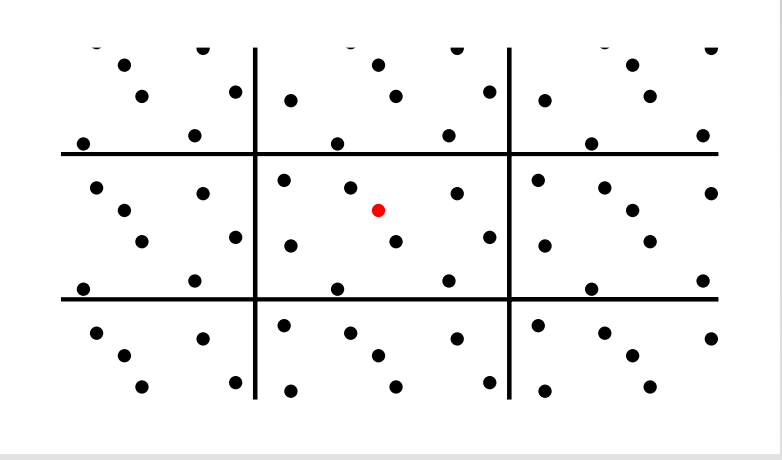
Coulomb forces on N particles, charges q_i, positions $\mathbf{r}_i$, box length L, periodic b.c.

- electrostatic potential

$$V = \frac{1}{2} \sum_{i,j=1}^{N} {\sum_{\mathbf{n}\in\mathbb{Z}^3}}' \frac{q_i q_j}{|\mathbf{r}_{ij} + \mathbf{n}L|}$$

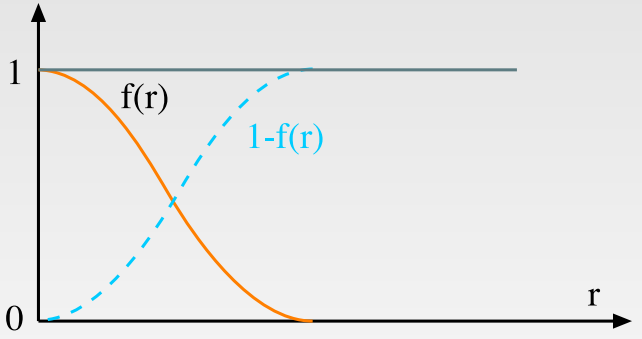
straightforward summation

impracticable



- **Trick 1:** Split problem into 2 parts with help of:

$$\frac{1}{r} = \underbrace{\frac{f(r)}{r}}_{\text{short range}} + \underbrace{\frac{1-f(r)}{r}}_{\text{long range}}$$



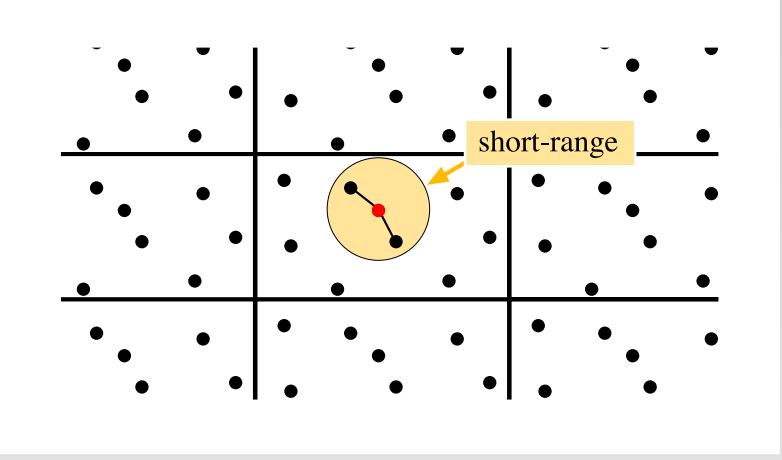
Particle Mesh Ewald

Coulomb forces on N particles, charges q_i, positions $\mathbf{r}_i$, box length L, periodic b.c.

- electrostatic potential

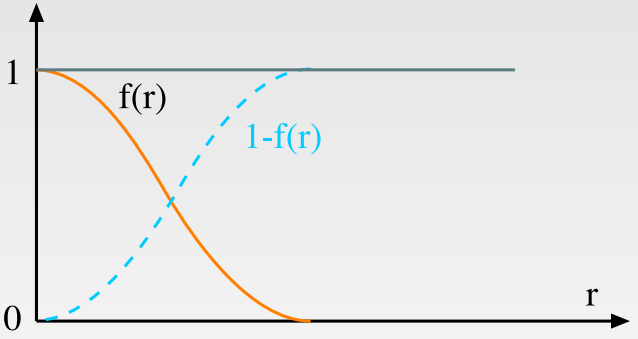
$$V = \frac{1}{2} \sum_{i,j=1}^{N} {\sum_{\mathbf{n}\in\mathbb{Z}^3}}' \frac{q_i q_j}{|\mathbf{r}_{ij} + \mathbf{n}L|}$$

straightforward summation impracticable



- **Trick 1:** Split problem into 2 parts with help of:

$$\frac{1}{r} = \underbrace{\frac{f(r)}{r}}_{\text{short range}} + \underbrace{\frac{1-f(r)}{r}}_{\text{long range}}$$



- $V = \underbrace{V_{dir}}_{\text{direct space}} + \underbrace{V_{rec}}_{\text{fourier space}}$

smooth V_{rec} needs only few $\mathbf{k}$

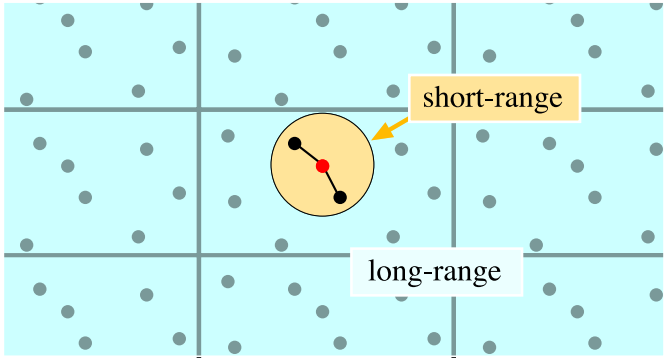
Particle Mesh Ewald

Coulomb forces on N particles, charges q_i, positions $\mathbf{r}_i$, box length L, periodic b.c.

- electrostatic potential

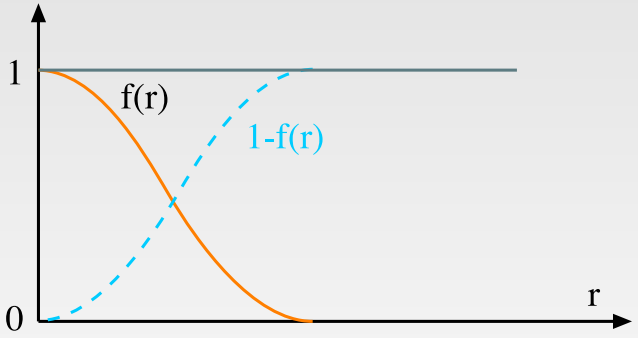
$$V = \frac{1}{2} \sum_{i,j=1}^{N} {\sum_{\mathbf{n}\in\mathbb{Z}^3}}' \frac{q_i q_j}{|\mathbf{r}_{ij} + \mathbf{n}L|}$$

straightforward summation impracticable



- **Trick 1:** Split problem into 2 parts with help of:

$$\frac{1}{r} = \underbrace{\frac{f(r)}{r}}_{\text{short range}} + \underbrace{\frac{1-f(r)}{r}}_{\text{long range}}$$

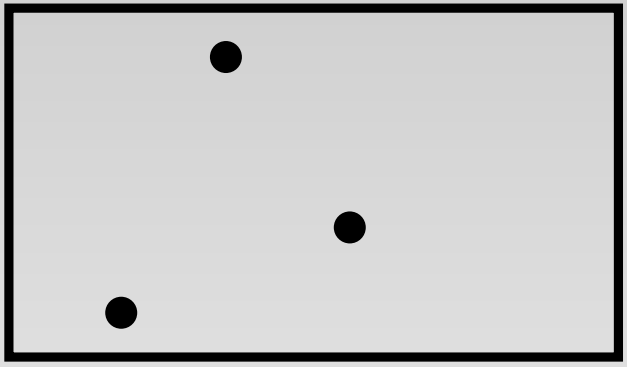


- $V = \underbrace{V_{dir}}_{\text{direct space}} + \underbrace{V_{rec}}_{\text{fourier space}}$

smooth V_{rec} needs only few $\mathbf{k}$

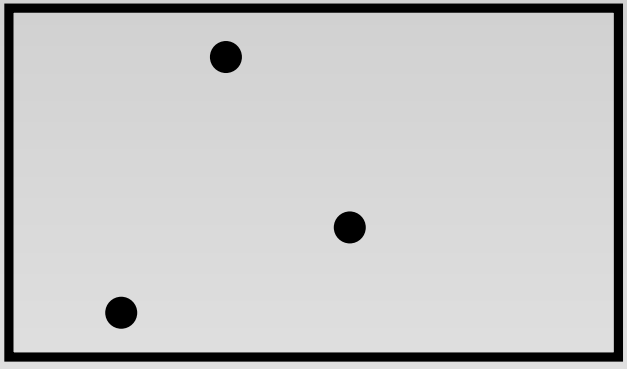
Particle Mesh Ewald

V_{rec} needs FT charge density



Particle Mesh Ewald

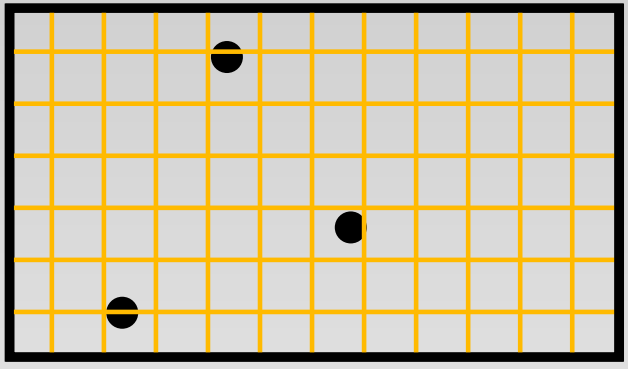
V_{rec} needs FT charge density



- **Trick 2: discretize charges** $\rightarrow$ use discrete FT

Particle Mesh Ewald

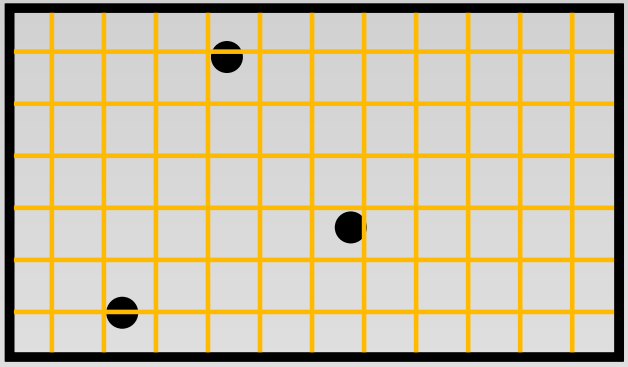
V_{rec} needs FT charge density



- **Trick 2: discretize charges** $\rightarrow$ use discrete FT

Particle Mesh Ewald

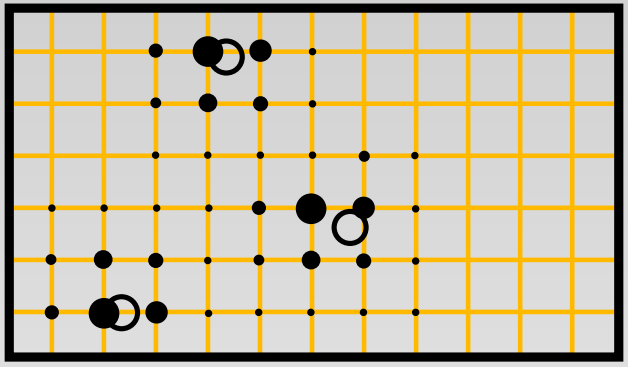
V_{rec} needs FT charge density



- **Trick 2: discretize charges** $\rightarrow$ use discrete FT
- Each charge is spread on $64 = 4x4x4$ neighbouring grid points, grid spacing 0.12 nm
- mesh-based charge density:
  approximation to Σ of charges at atom positions
- Aquaporin-1: 80 000 atoms, grid size $90x88x80 = 633\,600$ points

Particle Mesh Ewald

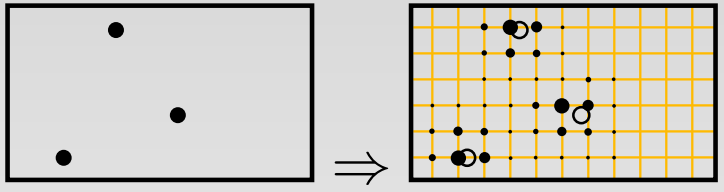
V_{rec} needs FT charge density



- **Trick 2: discretize charges** $\rightarrow$ use discrete FT
- Each charge is spread on $64 = 4x4x4$ neighbouring grid points, grid spacing 0.12 nm
- mesh-based charge density:
  approximation to Σ of charges at atom positions
- Aquaporin-1: 80 000 atoms, grid size $90x88x80 = 633\,600$ points

Evaluation of Coulomb forces

1. **Short-range** $\mathbf{F}_{i,dir}$
2. **Long-range (PME):**
   - Put charges on grid



   - FFT charge grid | convolution | FFT back

     yields potential V_{rec} at grid points

   - interpolate grid to derive forces at atom positions

     $$\mathbf{F}_{i,rec} = -\frac{\partial}{\partial \mathbf{r}_i} V$$

GMX 3.2.1 parallel PME

Put charges on grid

Sum q grid: MPI_Reduce

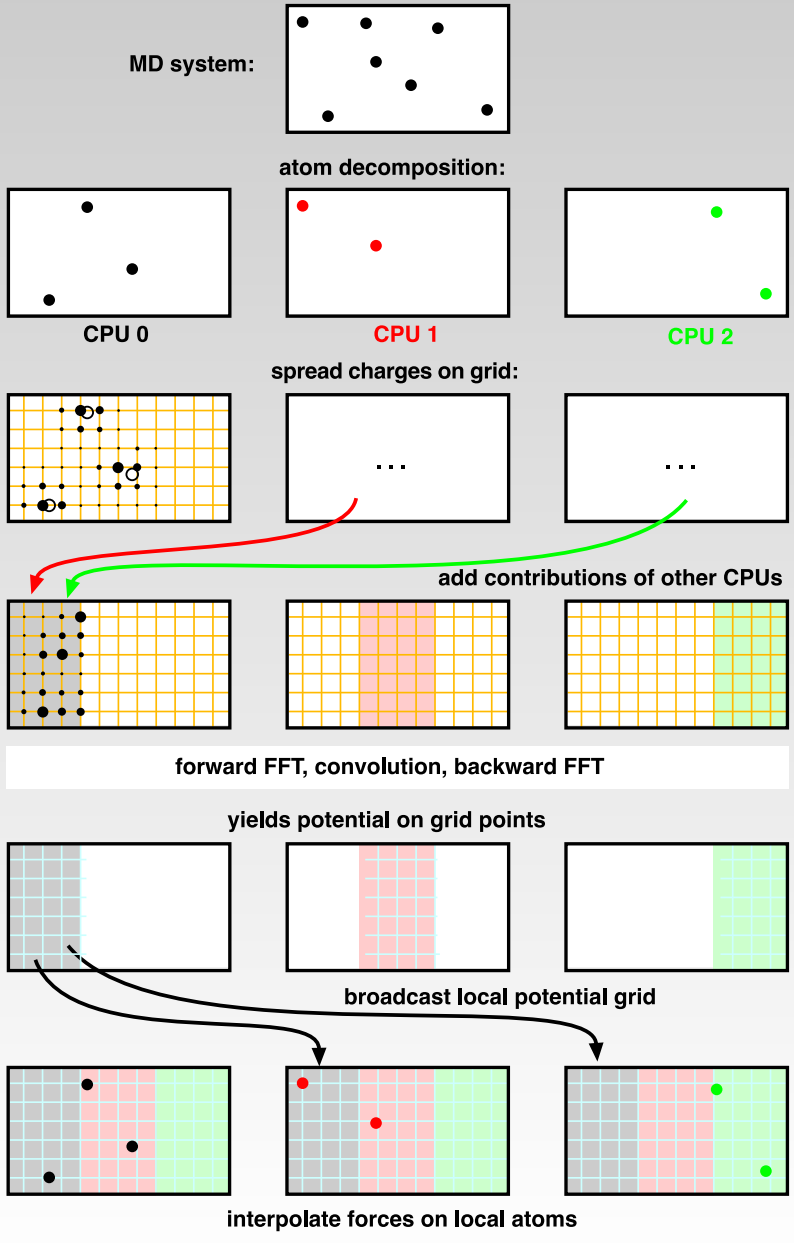
FFT

convolution

FFT back

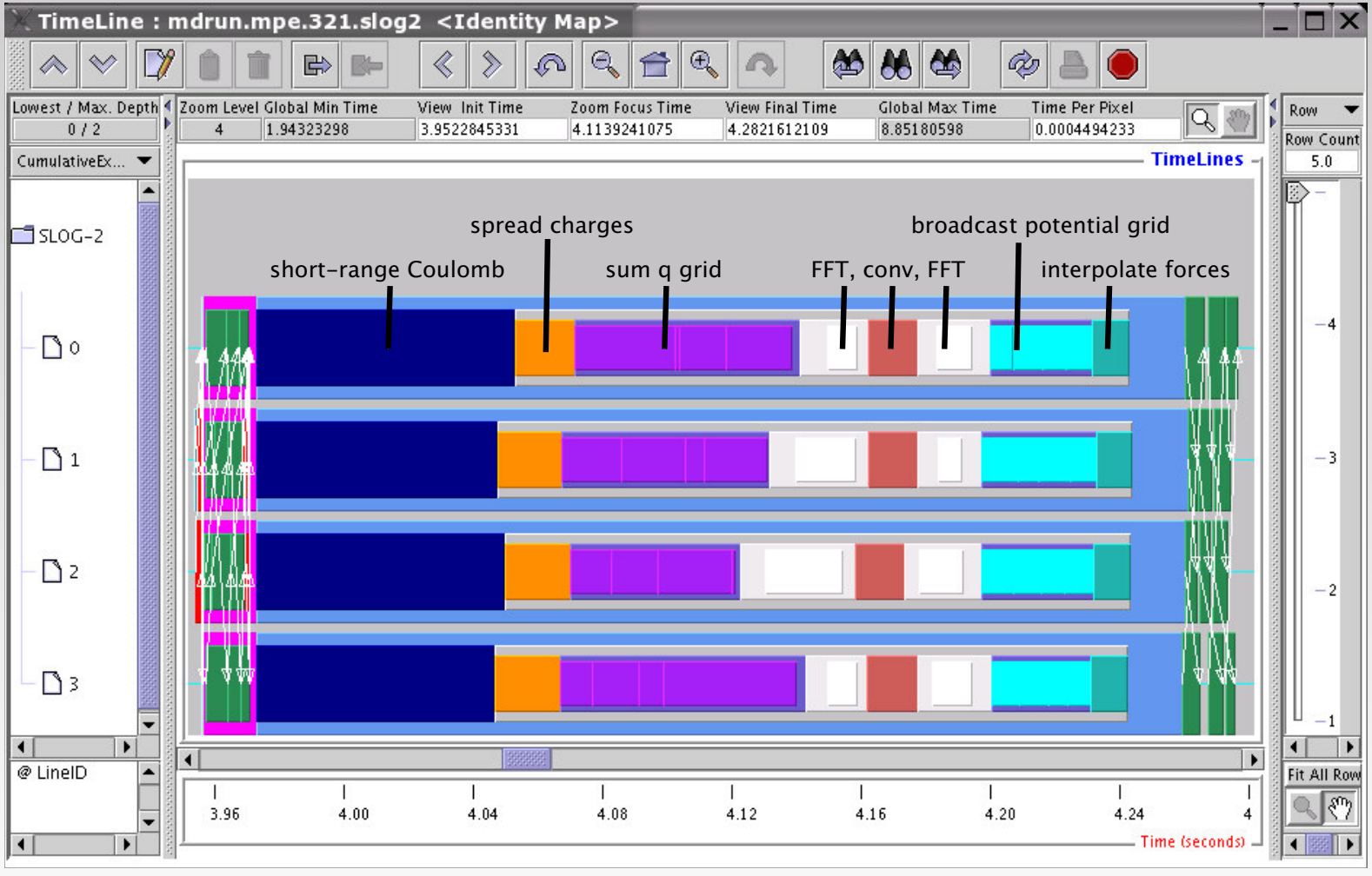
MPI_Bcast

Interpolate forces



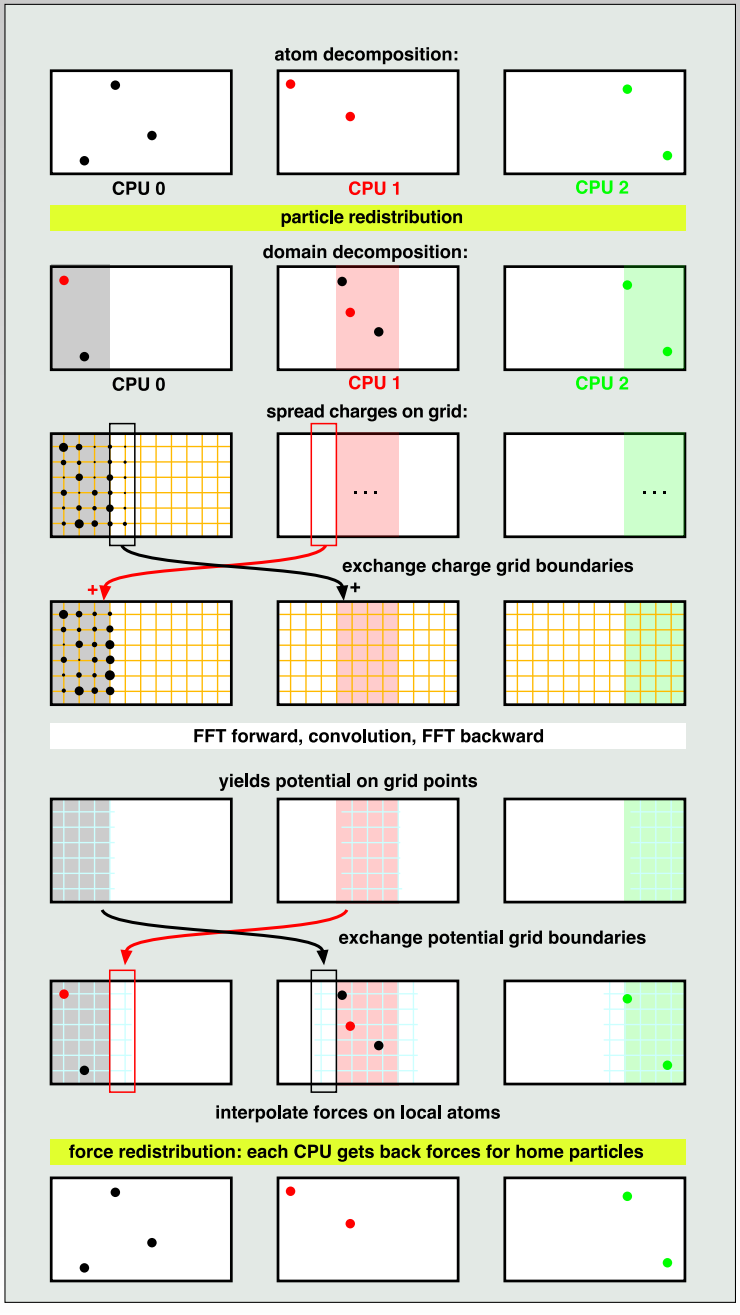
PME time step

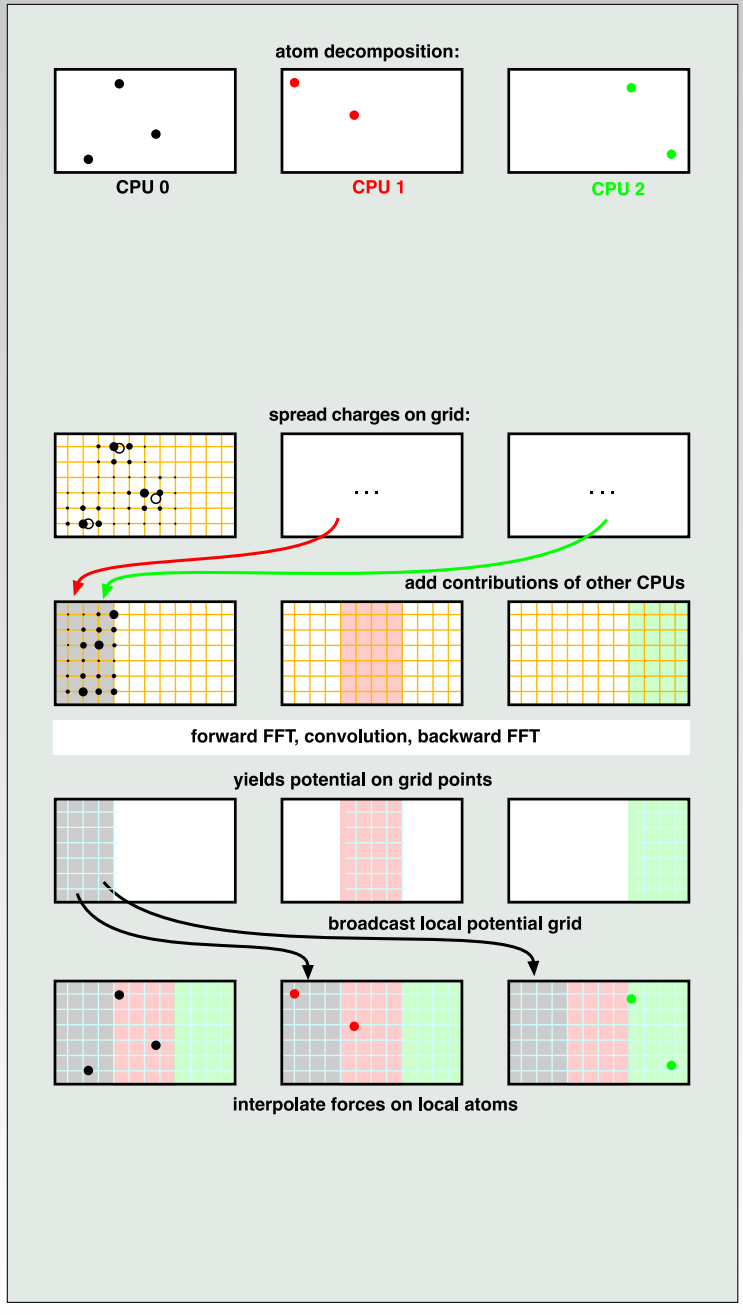
GROMACS 3.2.1 time step on 4 CPUs (Orca1)



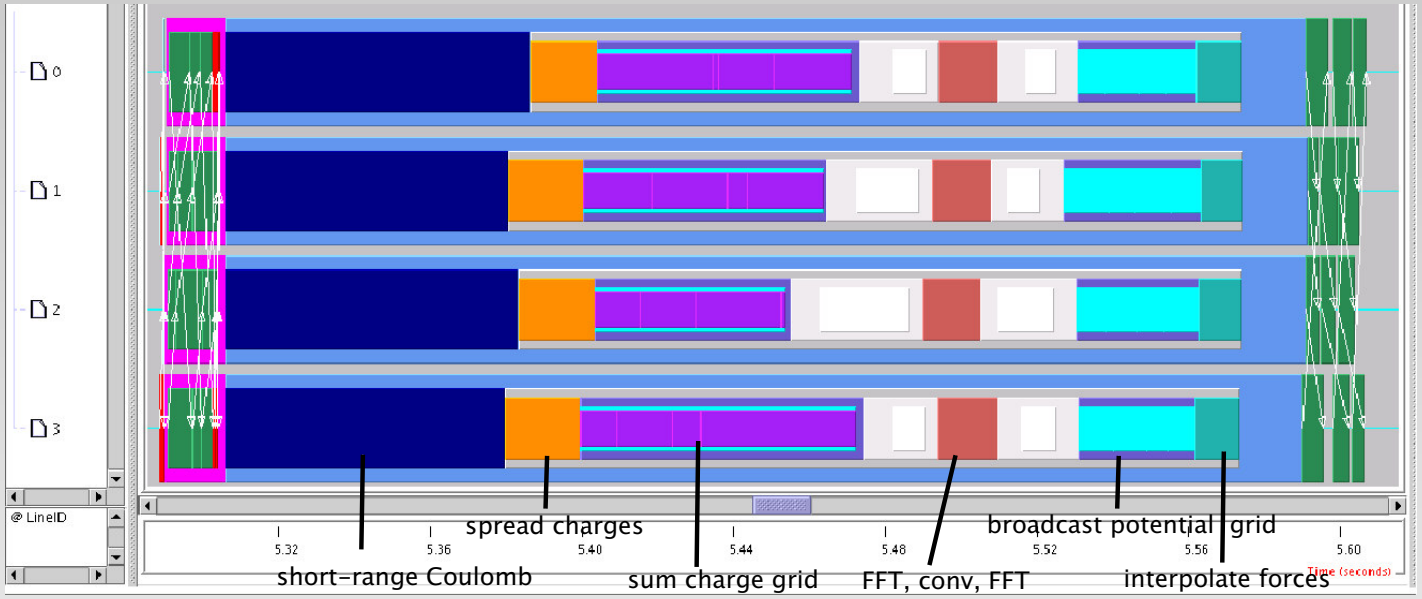
Sum charge grid: N × **MPI_Reduce**, Broadcast potential grid: N × **MPI_Bcast**

Domain decomp. ↔ atom dec.

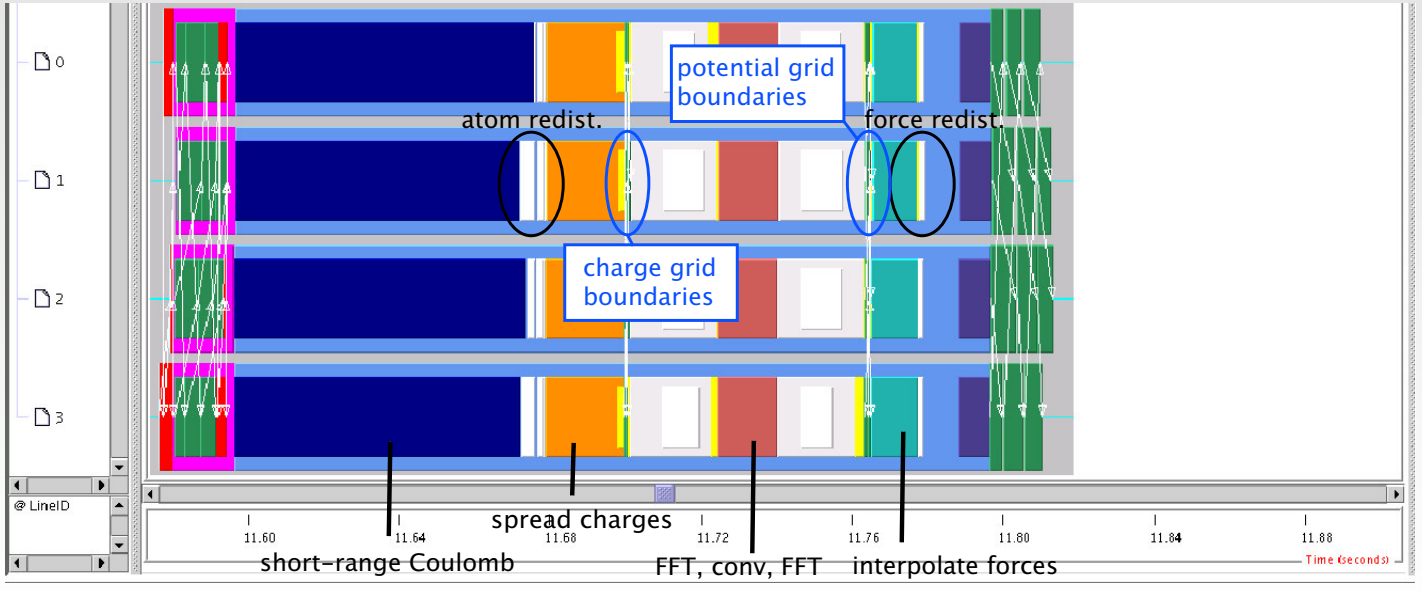




PME time step with DD

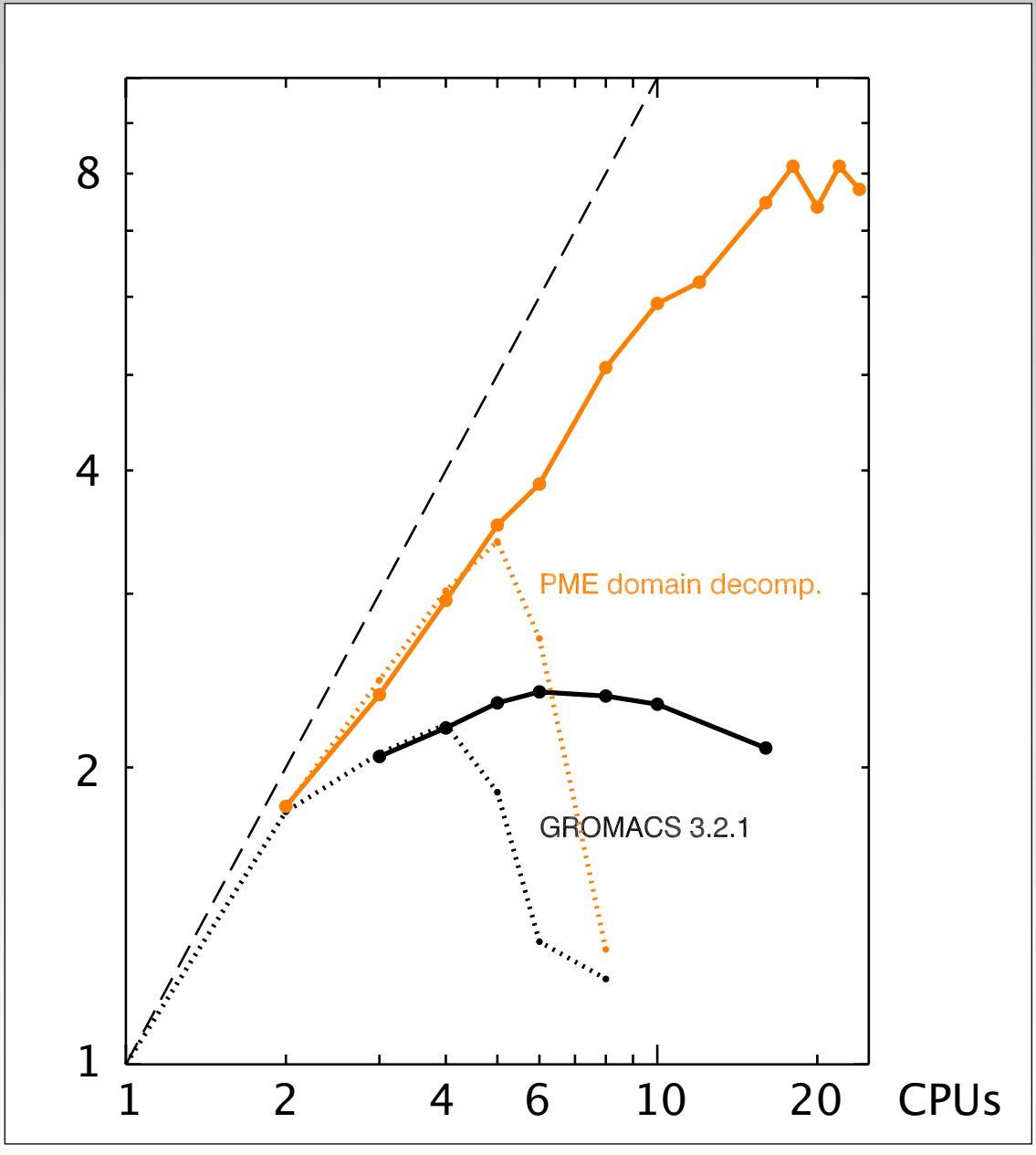


3.2.1



Domain decomp. speedups

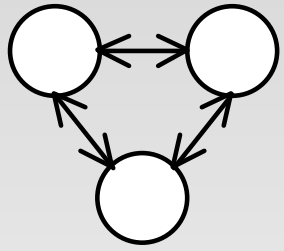
- broken lines: standard switch settings
- solid lines: with switch flow control
- Max. speedup:
  8.0 @ 18 CPUs
  (was: 2.4 @ 6 CPUs)
- scaling (4 CPUs):
  0.74 (was: 0.54)



PME/PP splitting

Separate a part of the CPUs to do PME only. Expect speed gains in FFT part!

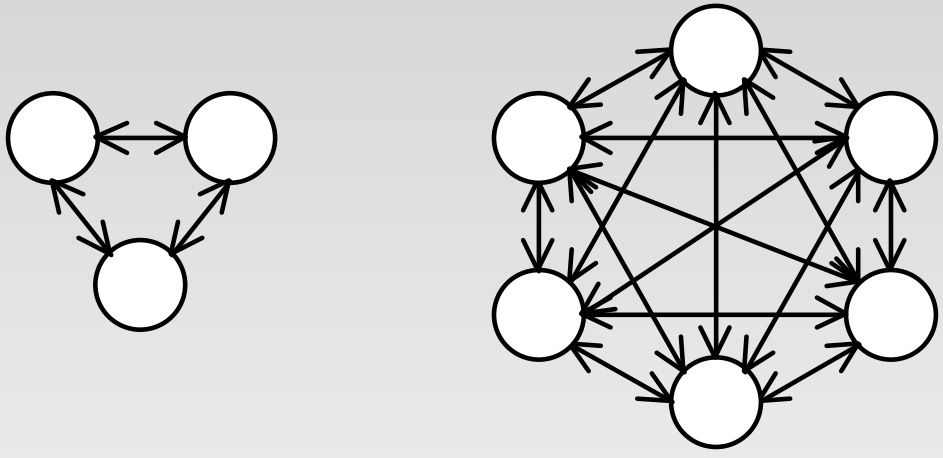
1. because parallel FFT expensive on high number of CPUs (All-to-all) (latency)



PME/PP splitting

Separate a part of the CPUs to do PME only. Expect speed gains in FFT part!

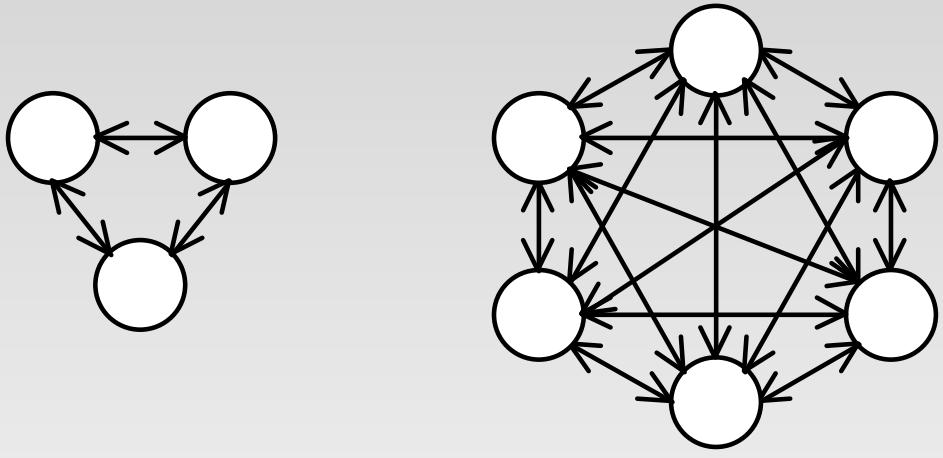
1. because parallel FFT expensive on high number of CPUs (All-to-all) (latency)



PME/PP splitting

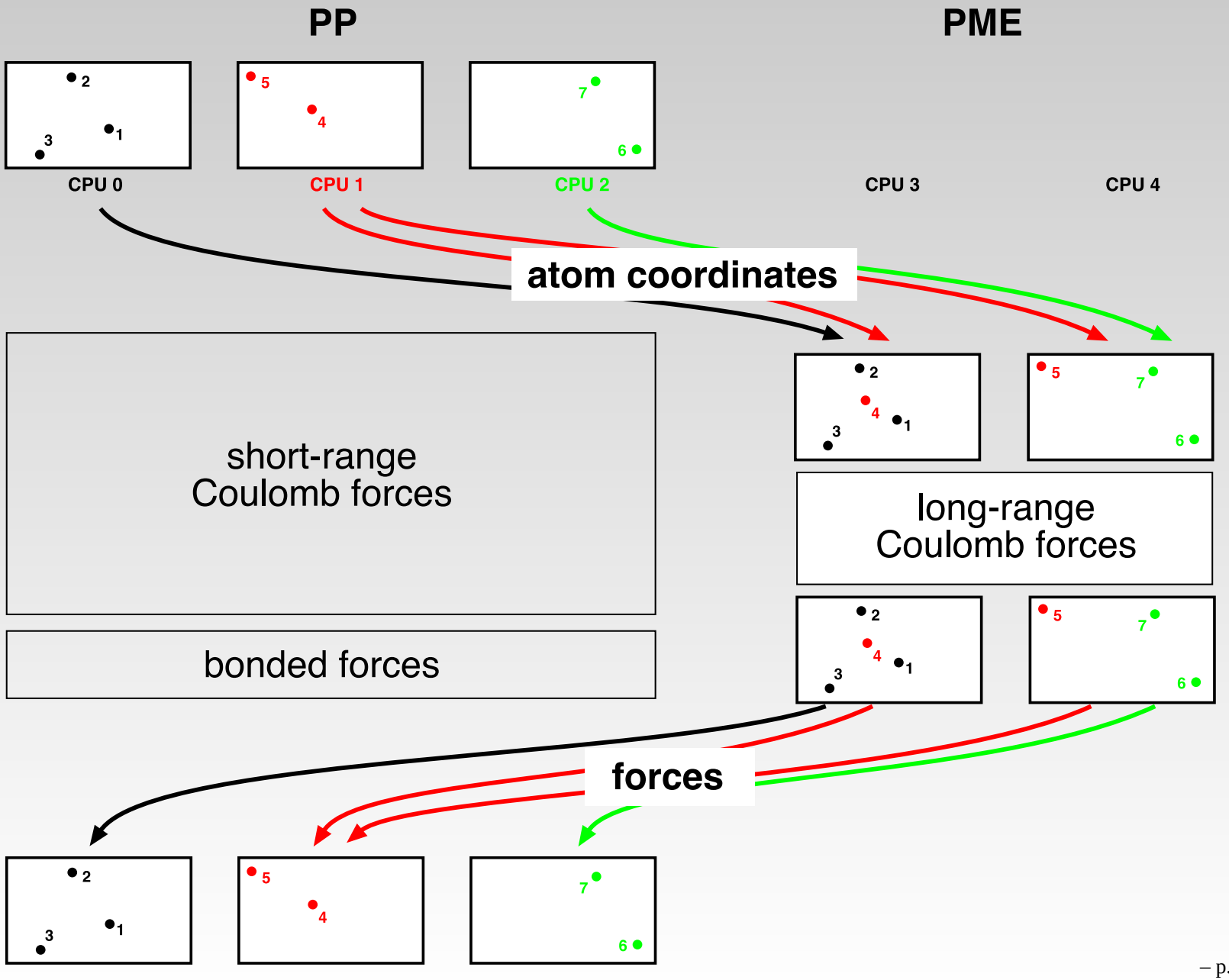
Separate a part of the CPUs to do PME only. Expect speed gains in FFT part!

1. because parallel FFT expensive on high number of CPUs (All-to-all) (latency)



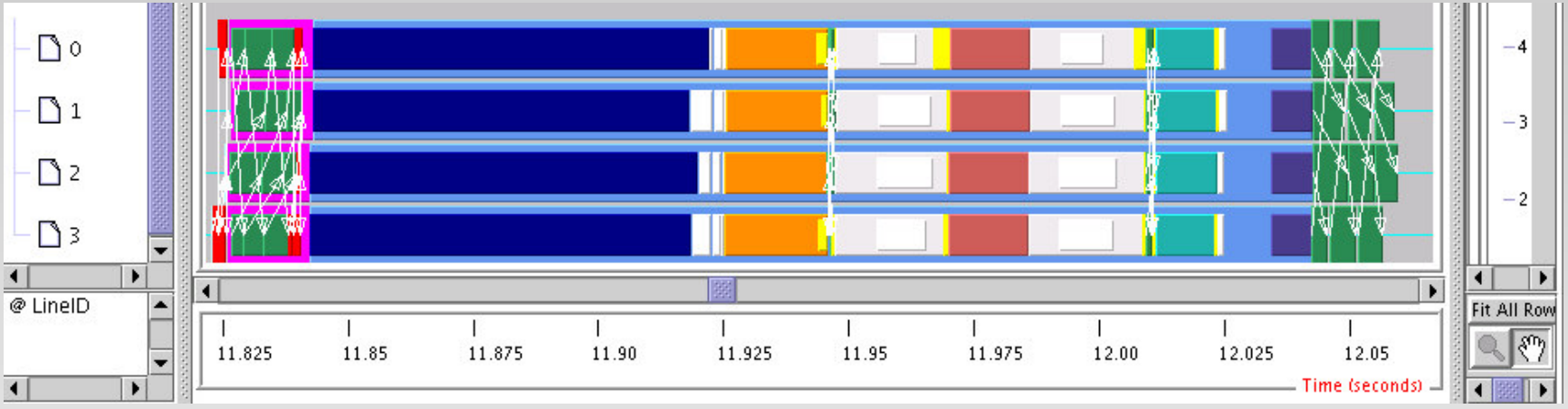
2. because x-size of FFT grid must be multiple of nCPU
   4 000 atoms $\rightarrow$
   30*x*30*x*21 = 18 900 FFT grid points on 2 CPUs
   40*x*40*x*21 = 33 600 FFT grid points on 20 CPUs.

PME/PP splitting scheme



Time step AP-1 / 80k atoms

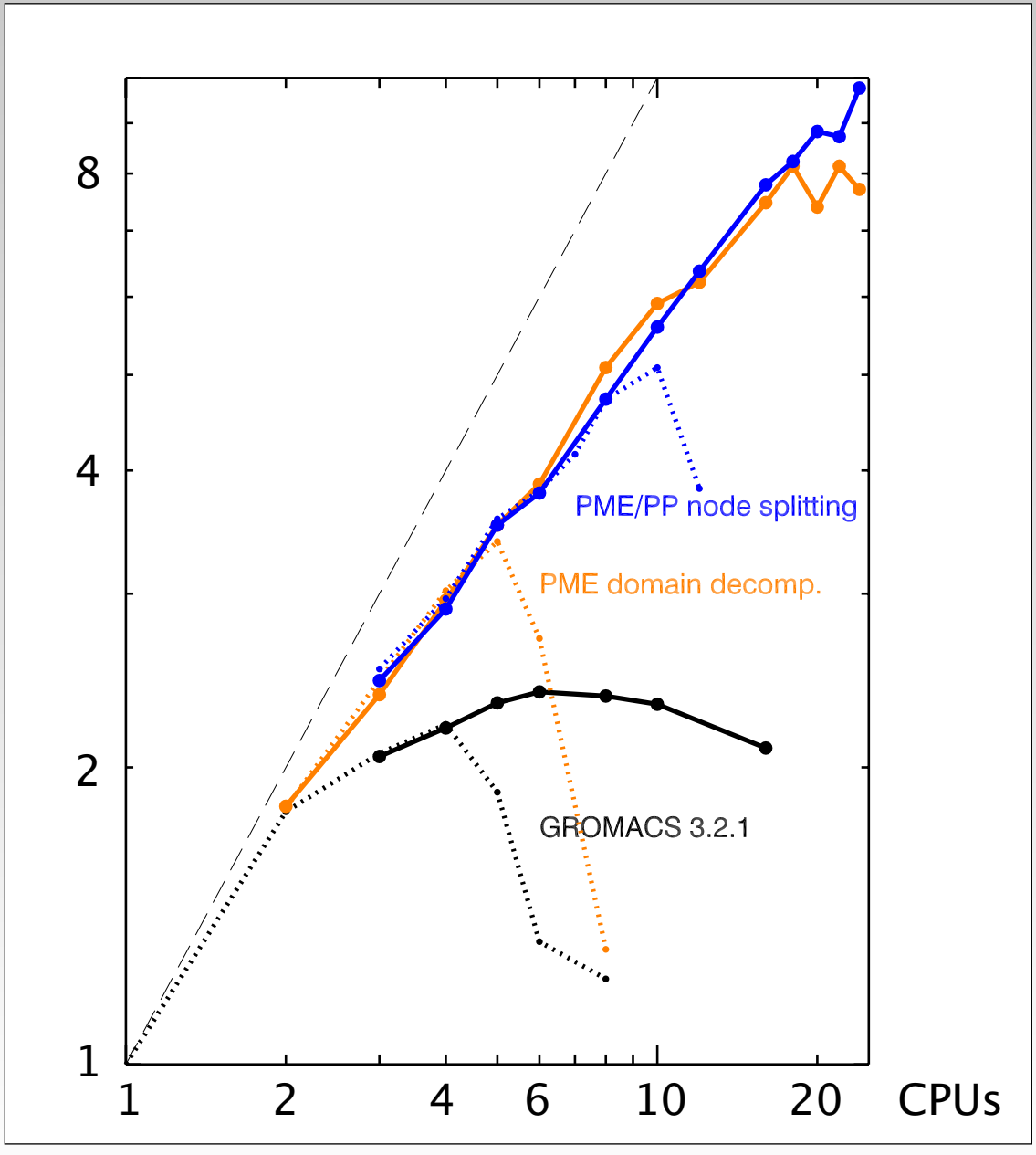
Domain decomposition in PME part (speedup = 2.9 @ 4 CPUs, scaling 0.74):



DD + node splitting (speedup = 2.9 @ 4 CPUs, scaling 0.74):

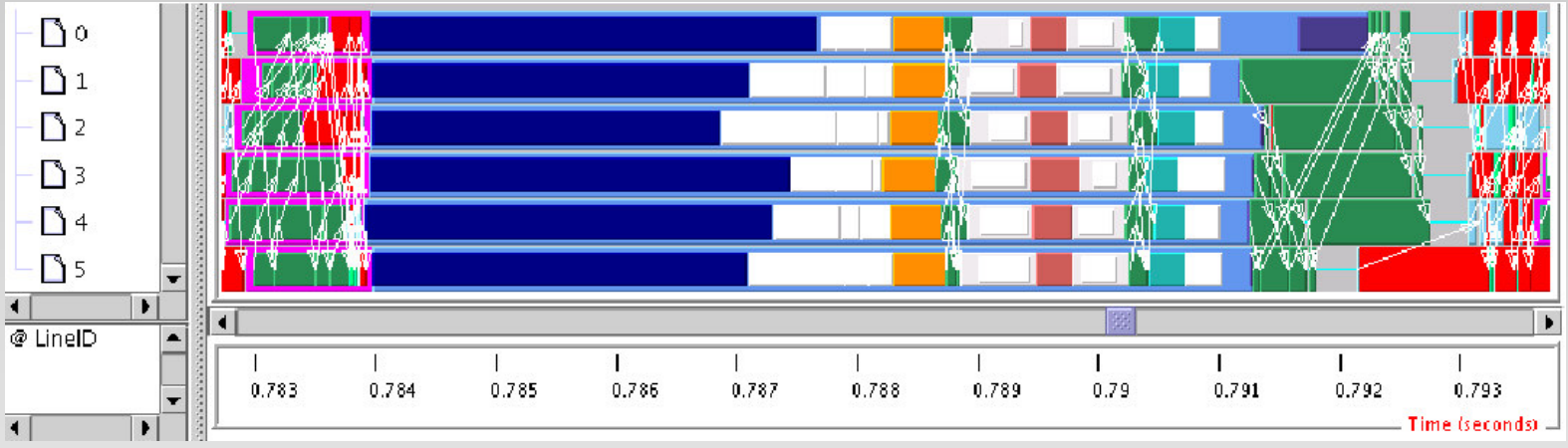
PME/PP splitting speedups

- Max. speedup:
  $>$ **9.6** @ 24 CPUs
- scaling (4 CPUs):
  0.74

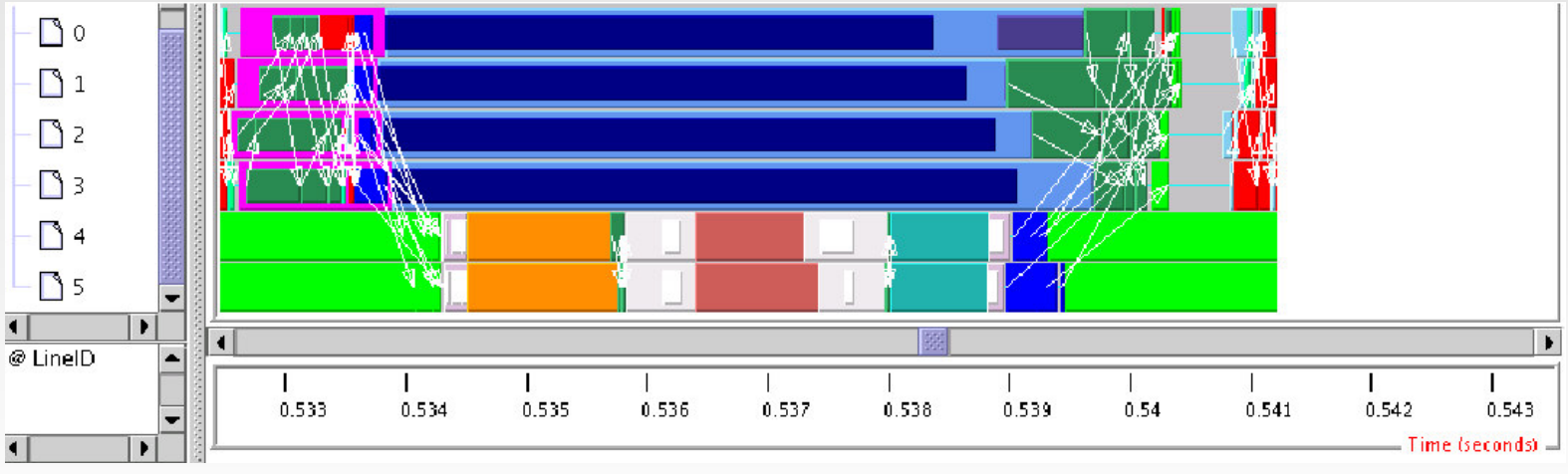


Time step Guanylin / 4k atoms

Domain decomposition in PME part (speedup = 2.8 @ 6 CPUs, scaling 0.47):



DD + node splitting (**speedup = 3.6** @ 6 CPUs, **scaling 0.60**):



Summary

1. DD only: scaling @ 4 CPUs & 80 000 atoms
   0.54 → 0.74 !
2. DD + node splitting: **speedup of 9.6+ possible** (was: 2.4)
3. Node splitting needed for **small systems** and/or **many CPUs**

Outlook:

Optimize splitting code for speed (non-blocking communications)