JÜLICH
FORSCHUNGSZENTRUM

Parallel Tree Methods

Paul Gibbon

Institute for Advanced Simulation
Jülich Supercomputing Centre (JSC)

Heraeus Summer School on Long-Range Interactions 6-10 September 2010

Outline

1. Basic multipole tree algorithm (Barnes-Hut method)
2. Parallel tree code
3. Scaling and performance issues
4. Applications

1. Basic multipole tree algorithm (Barnes-Hut method)
2. Parallel tree code
3. Scaling and performance issues
4. Applications

N-body electrostatics

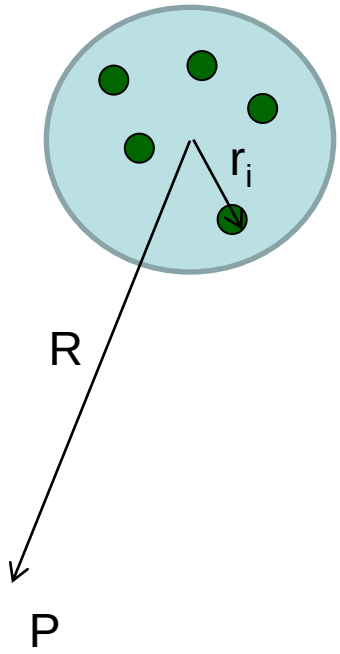
- Multipole expansion for general vector field:

$$\mathbf{E}(\mathbf{R}-\mathbf{r}_i) \simeq \mathbf{E}(\mathbf{R}) \qquad \text{monopole}$$

$$-\mathbf{r}_i \nabla \mathbf{E}(\mathbf{R}) \qquad \text{dipole}$$

$$+\tfrac{1}{2}\mathbf{r}_i\mathbf{r}_i :: \nabla\nabla\mathbf{E}(\mathbf{R}) \qquad \text{quadrupole}$$

$$+\ldots,$$



Multipole Algorithms

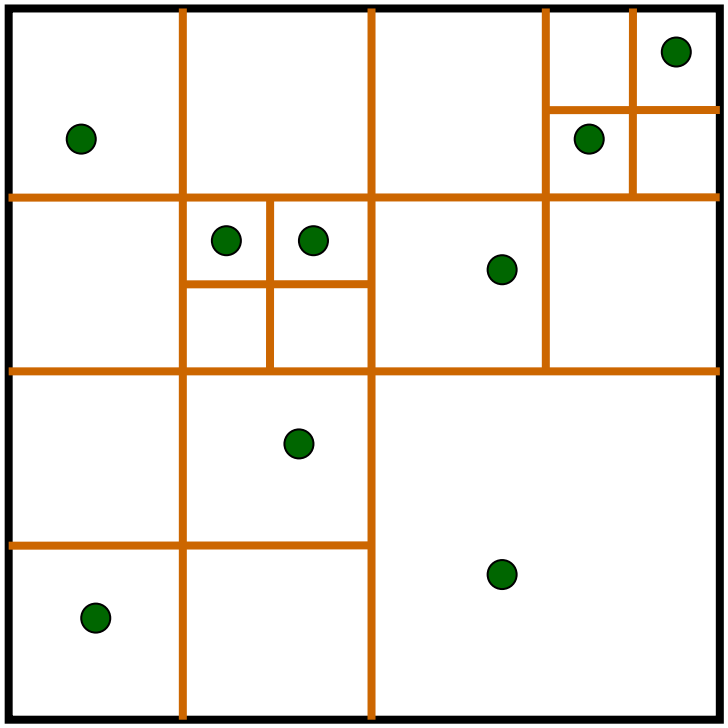
- Tree Code O(N log N)

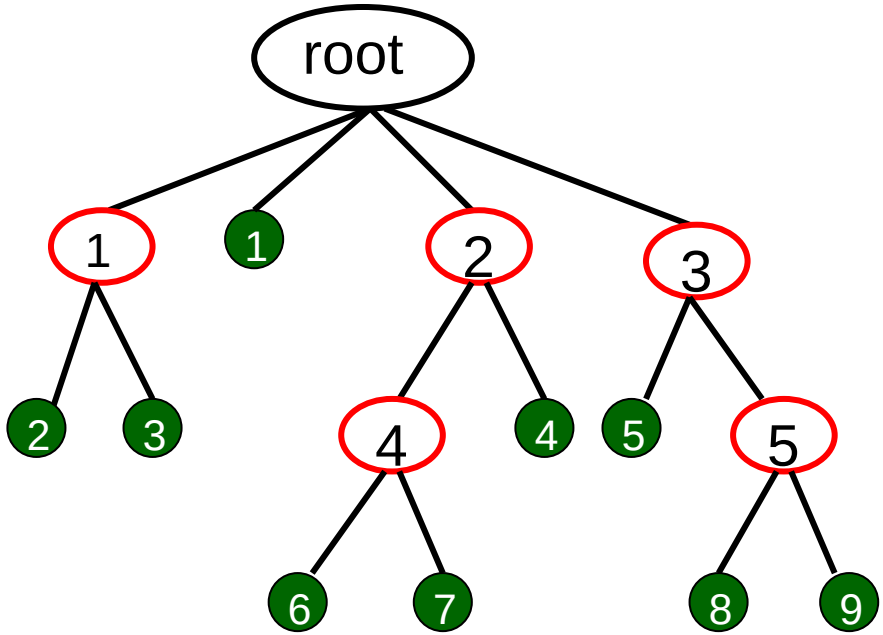
  Josh Barnes (1986)

- Fast Multipole Method O(N)

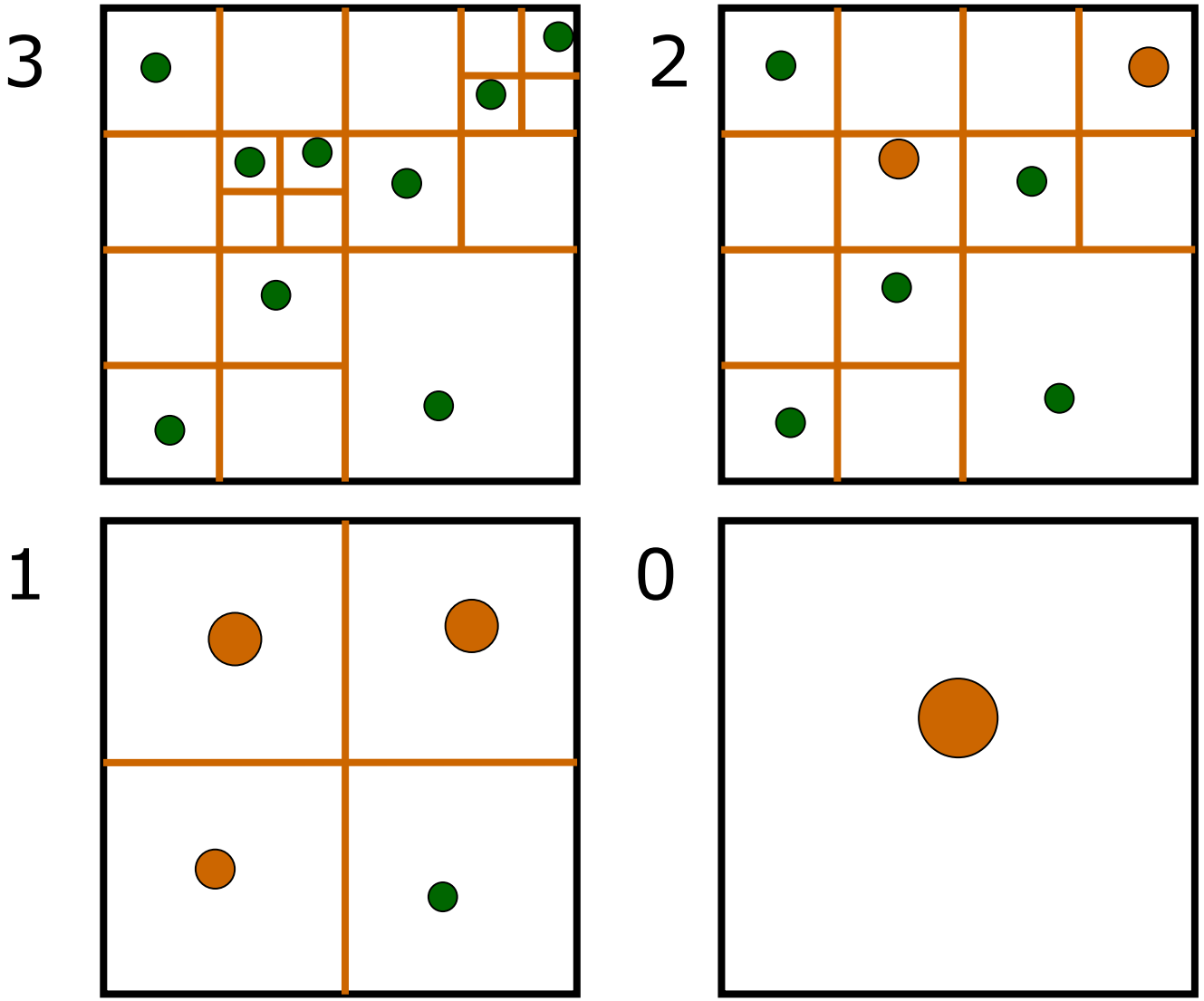
  Leslie Greengard (1988)

Tree building

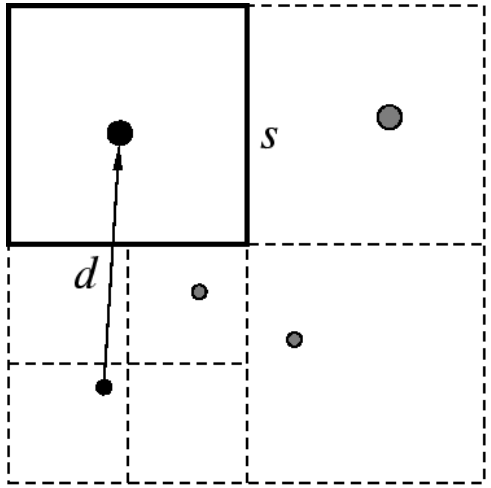




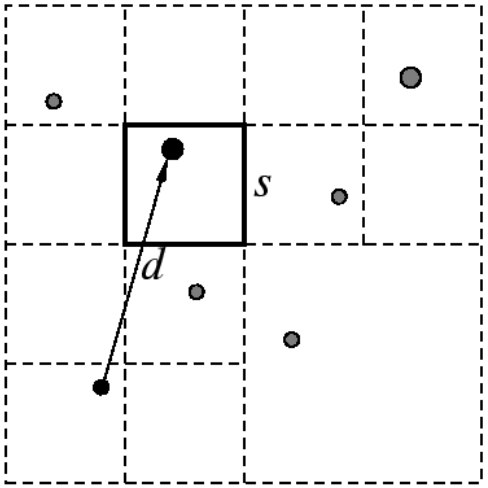
Multipole moments at each tree level



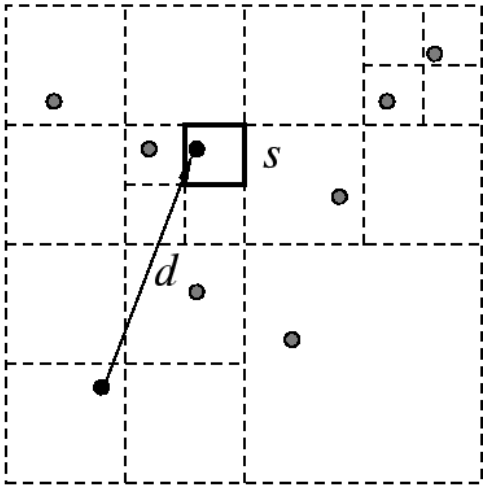
Tree-walk: multipole acceptance



$s/d = 0.94$



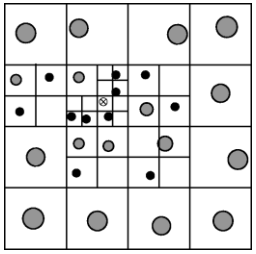
$s/d = 0.48$

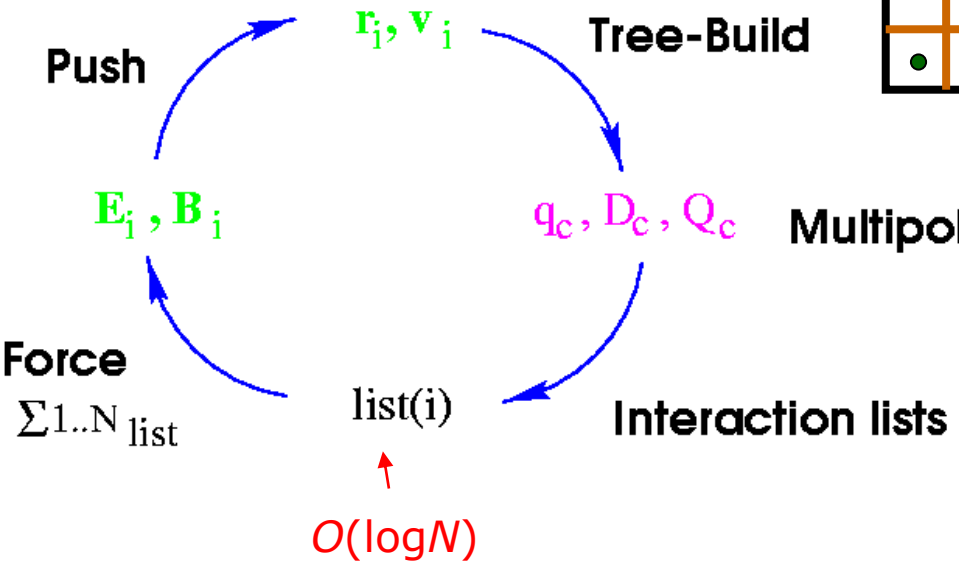


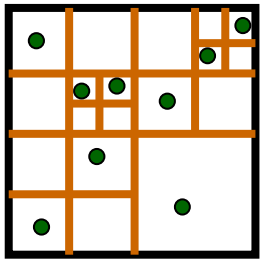
$s/d = 0.23$

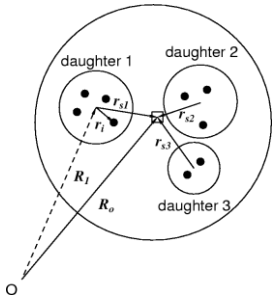
Put node on list if s/d < θ

Barnes-Hut tree algorithm



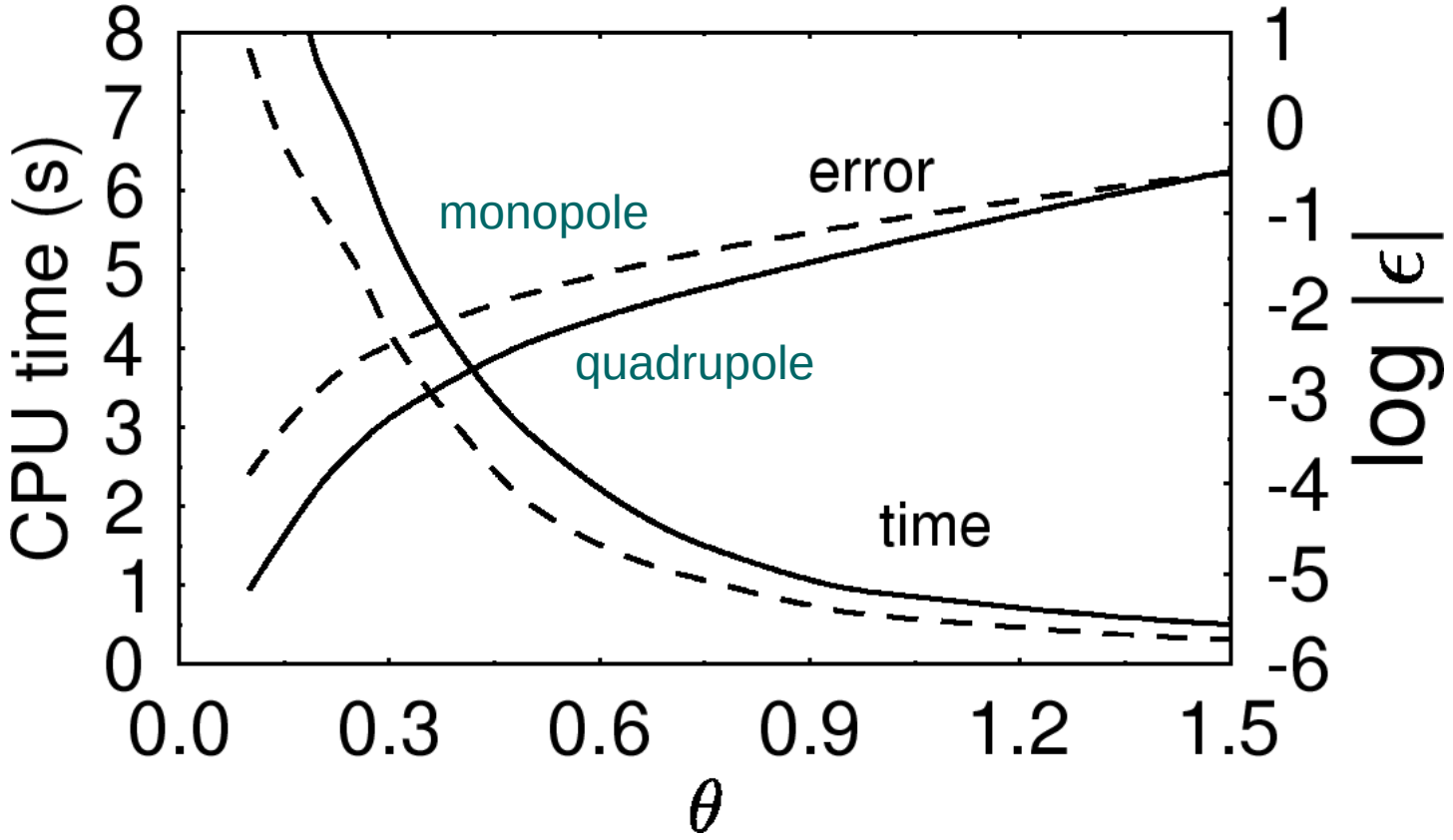






For a tutorial introduction, see
Pfalzner & Gibbon, *Many Body Tree Methods in Physics*,
(Cambridge University Press, 1996)

Speed-accuracy trade-off



JÜLICH
FORSCHUNGSZENTRUM

1. Basic multipole tree algorithm (Barnes-Hut method)
2. Parallel tree code
3. Scaling and performance issues
4. Applications

Algorithm structure

domain decomposition:
    convert local particle coords to keys
    perform global key-sort

build tree:
    construct local nodes
    define global branches
    fill in top-level nodes
    compute multipole properties

define particle chunks
for each chunk **do**
    get interaction lists via *tree-walk*
    compute forces
end do

Theoretical algorithm scaling

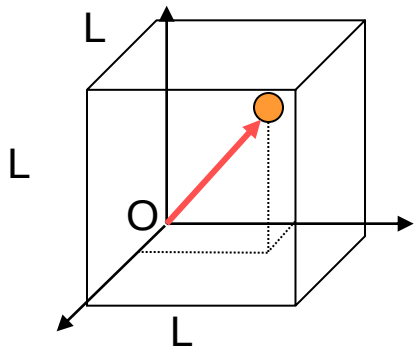
Function	Complexity	Routine name
Particle properties (**r**,**v**,q,m)	N/P	`configure`
Construct keys (x,y,z) $\Rightarrow$ p	N/P	`tree_domains`
Sort keys + load balance $p_1 \ldots p_n$	< N/P log N	`sl_sort`
Build local tree	N/P	`tree_build`
Construct branch nodes	P log N/P	`tree_branches`
Fill in top-level nodes	log P	`tree_fill`
Multipole moments	log N/P	`tree_properties`

Parallelization strategies

- Equal volume (uniform systems)
- Orthogonal recursive bisection (ORB)
- Hashed Oct Tree (Warren & Salmon, 1994):

  map coordinates onto *space-filling curve*
  cut into equal (load-balanced) segments

Converting particle coords into binary keys



r=(2.5, 3.5, 4.0)

L = 8.0, nlevel = 4 $\Rightarrow$ $s=L/2^{nlevel} = 0.5$

$i_z = z/s = 8$ $i_y=7$ $i_x=5$

Binary: 01000 00111 00101

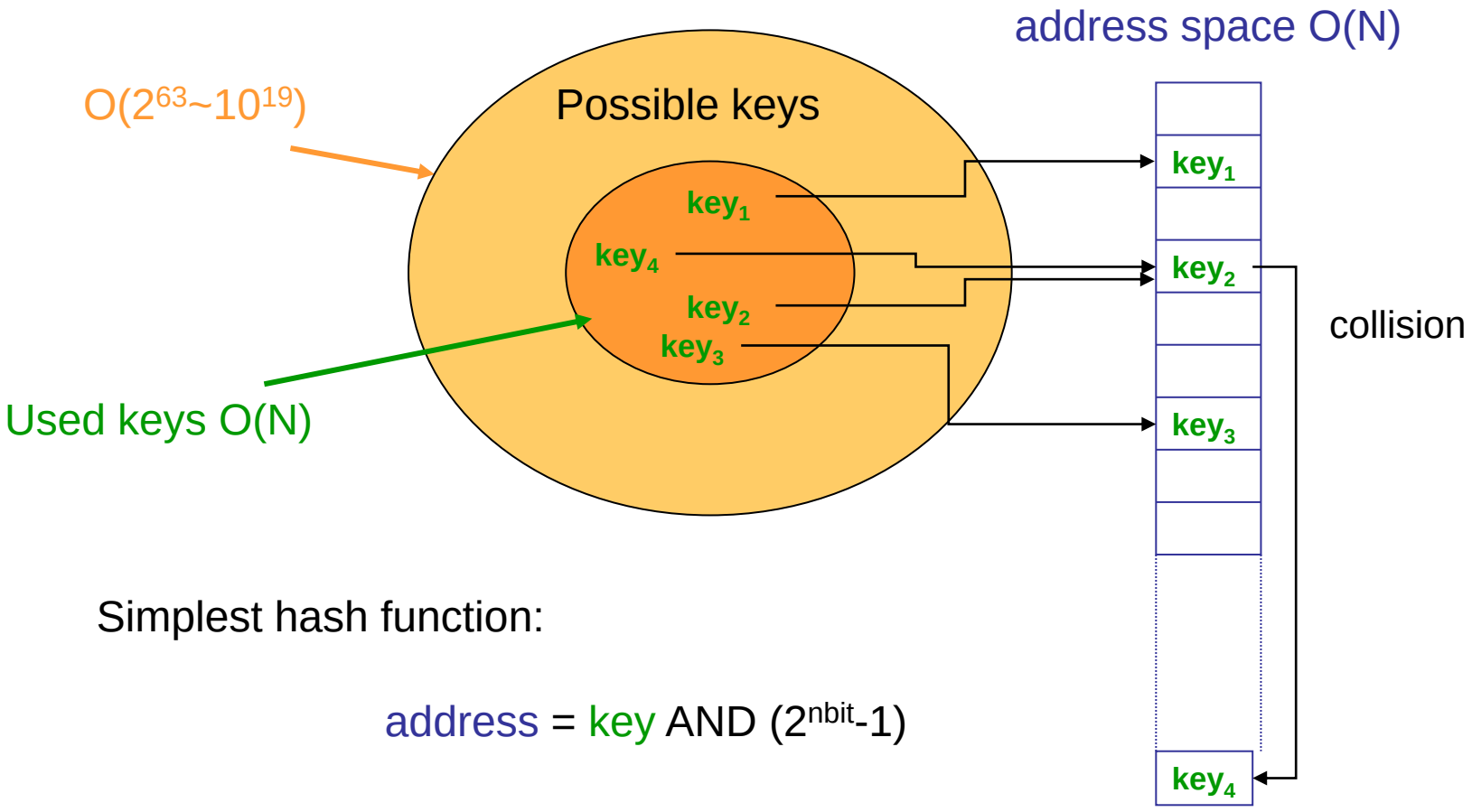
Interleave: 1 000 100 011 010 011

placeholder bit

64-bit machine $\Rightarrow$ 21 bits/coord + placeholder

$\Rightarrow$ max 20 refinement levels

Hashing: mapping keys to memory locations

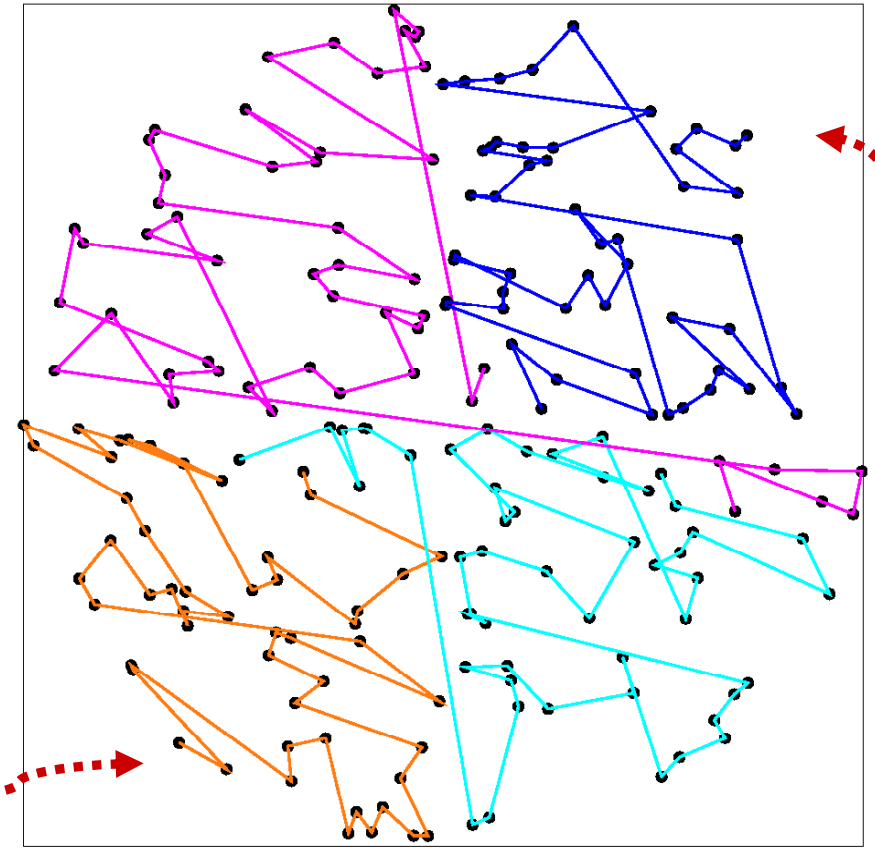


Simplest hash function:

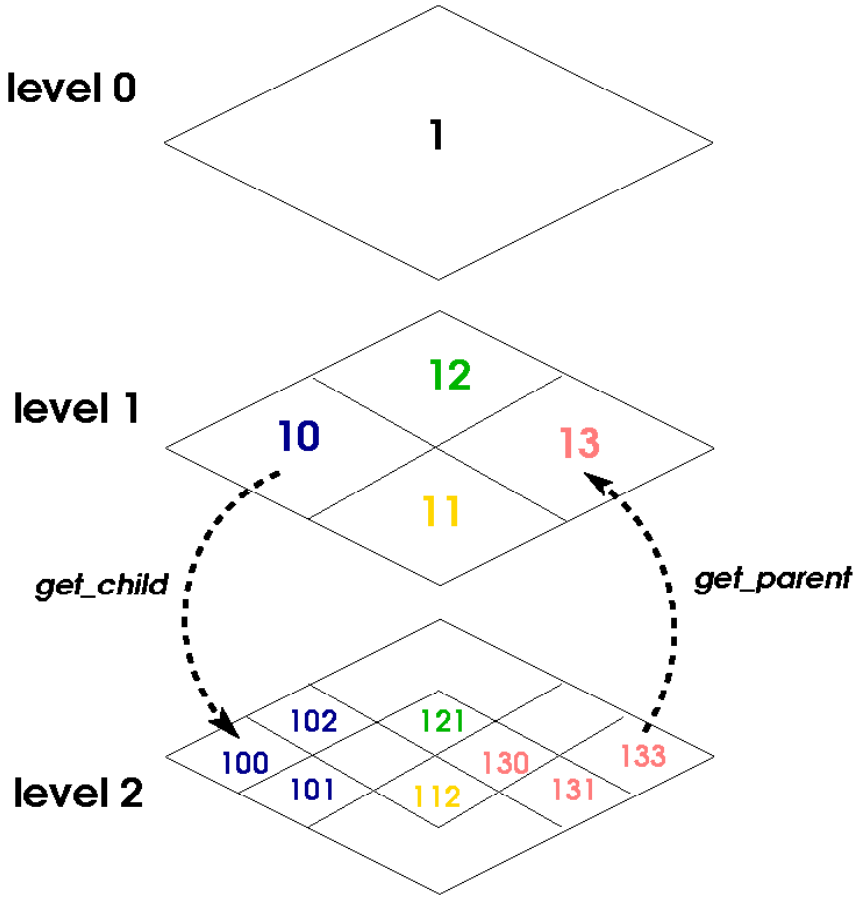
address = key AND (2^{nbit}-1)

Sorted list of keys: particle coordinates mapped onto 1D space-filling curve

Morton (Z) order:

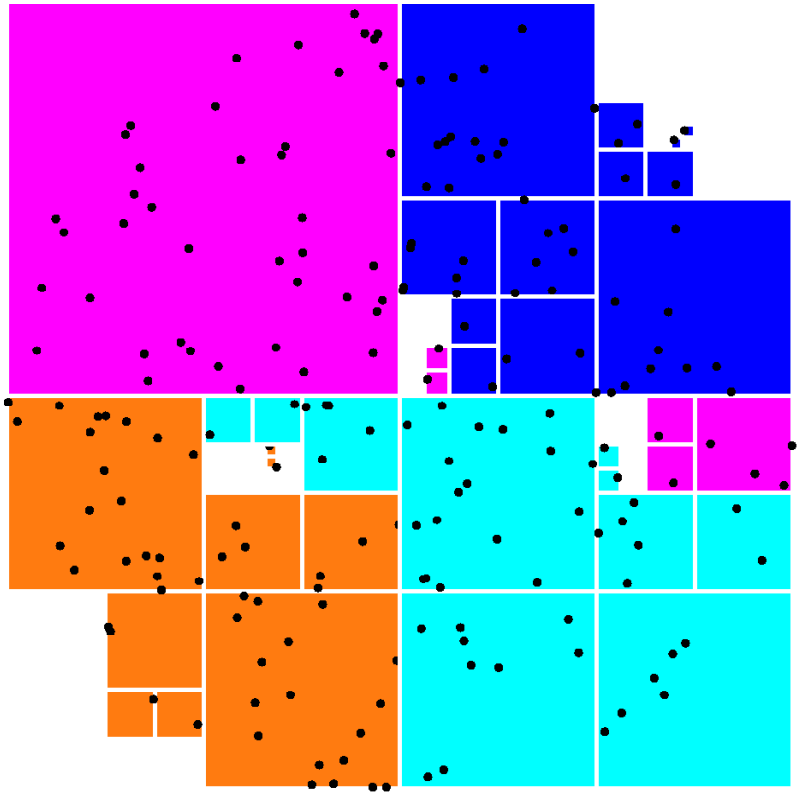


Local tree construction: parent nodes derived from particle keys

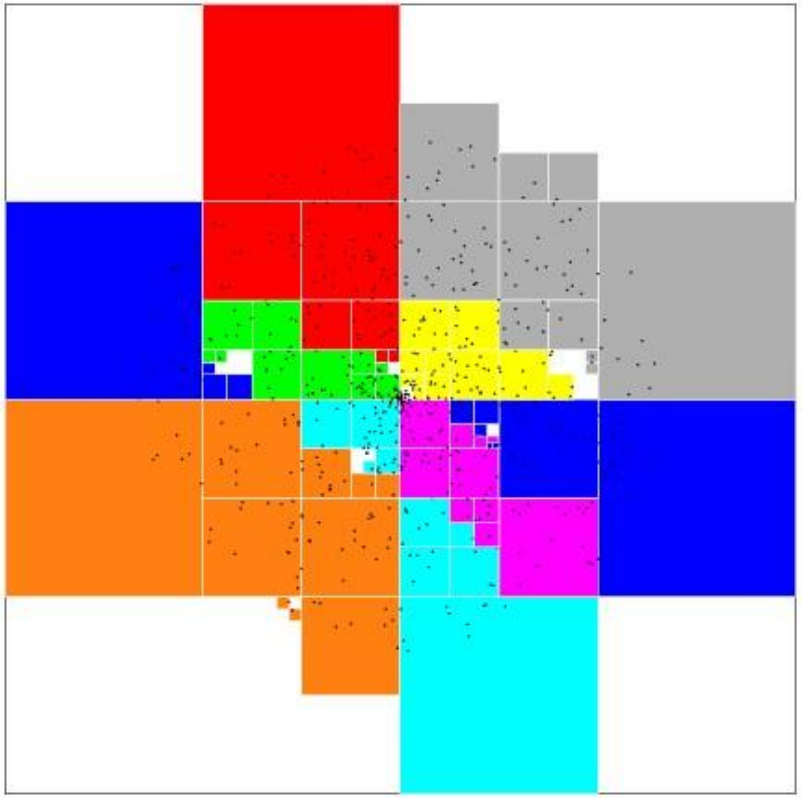


Domain decomposition: 2D

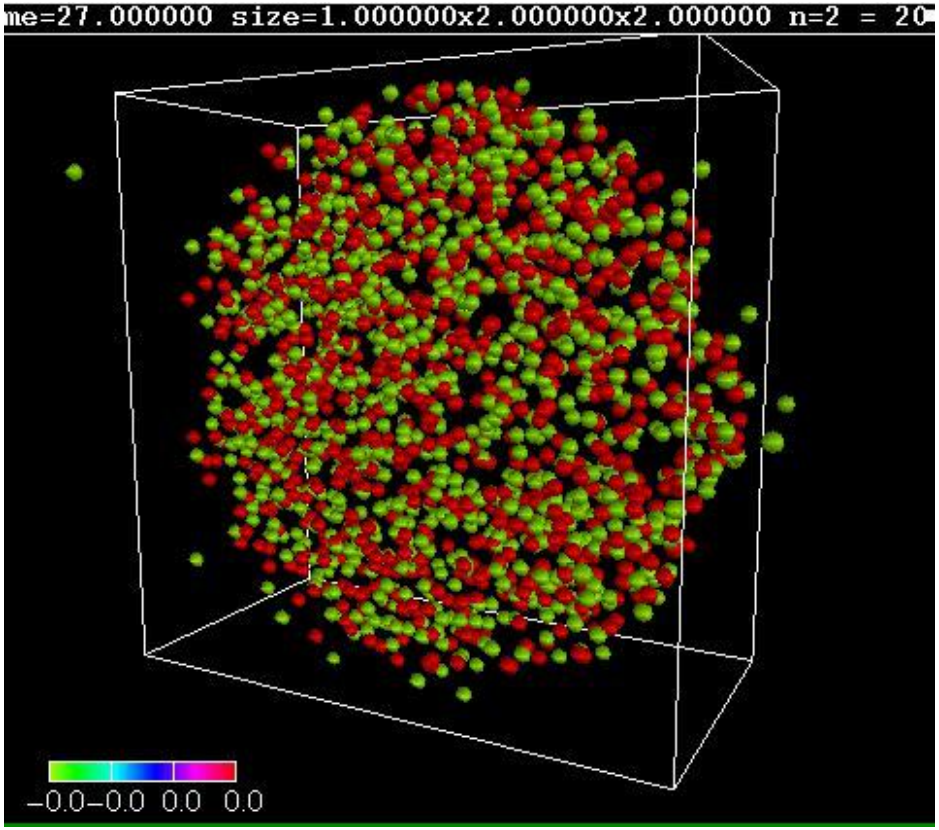
100 particles, 4 procs



200 particles, 8 procs



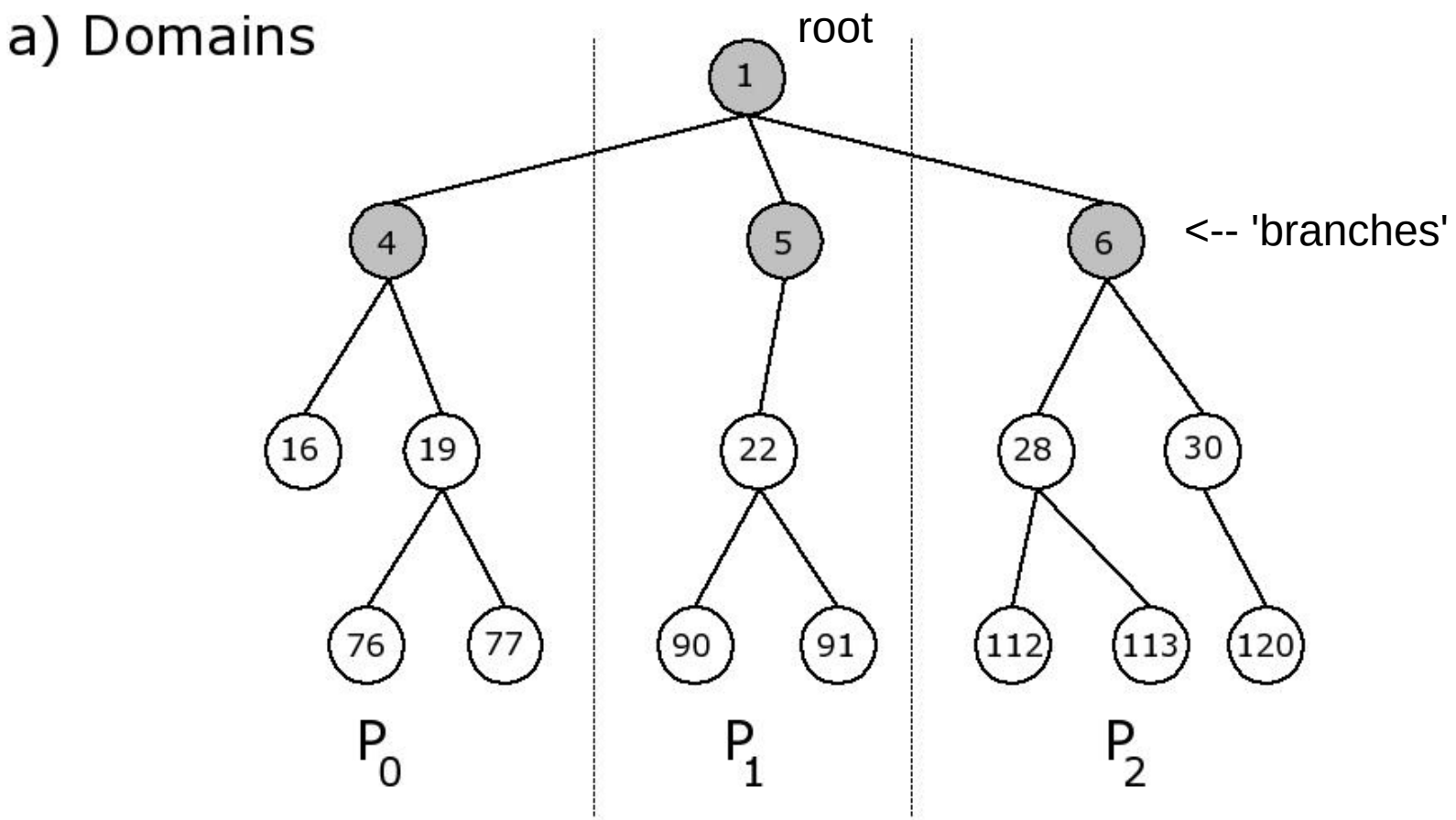
Domain decomposition: 3D



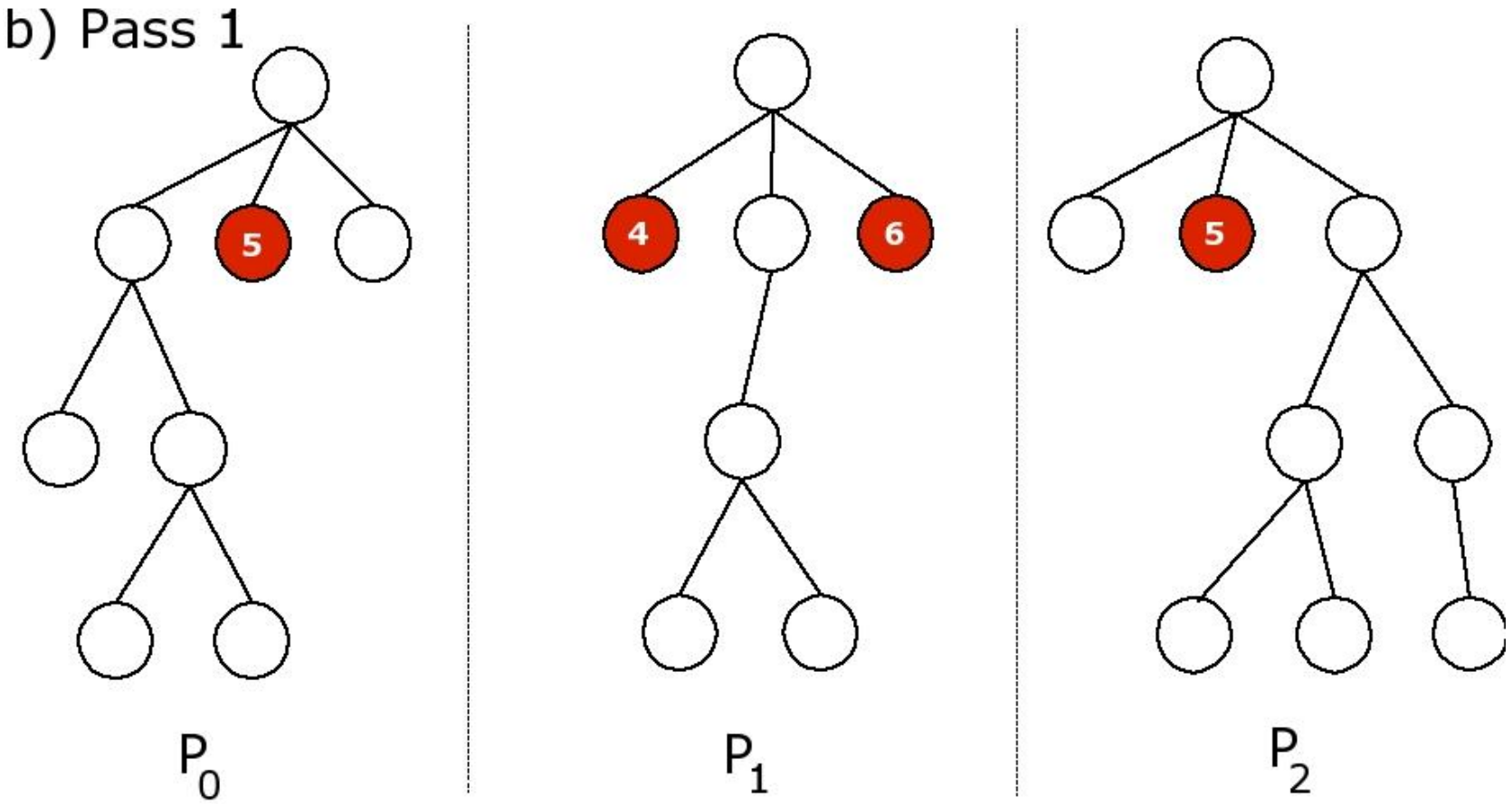
Algorithm structure: local tree-walk

```
do while (active particles)

        if (MAC_OK) then
                put walk_node on interaction list
                walk_node=next_node

        else if  (.not. MAC_OK & node local)
                walk_node = first_child_node

        else if  (.not. MAC_OK & node non-local)
                put walk_node on request list
                walk_node = next_node
        end if

end do
```

Tree-walk example: domains on 3 CPUs



Local walks: pass 1



Algorithm structure: multipole exchange

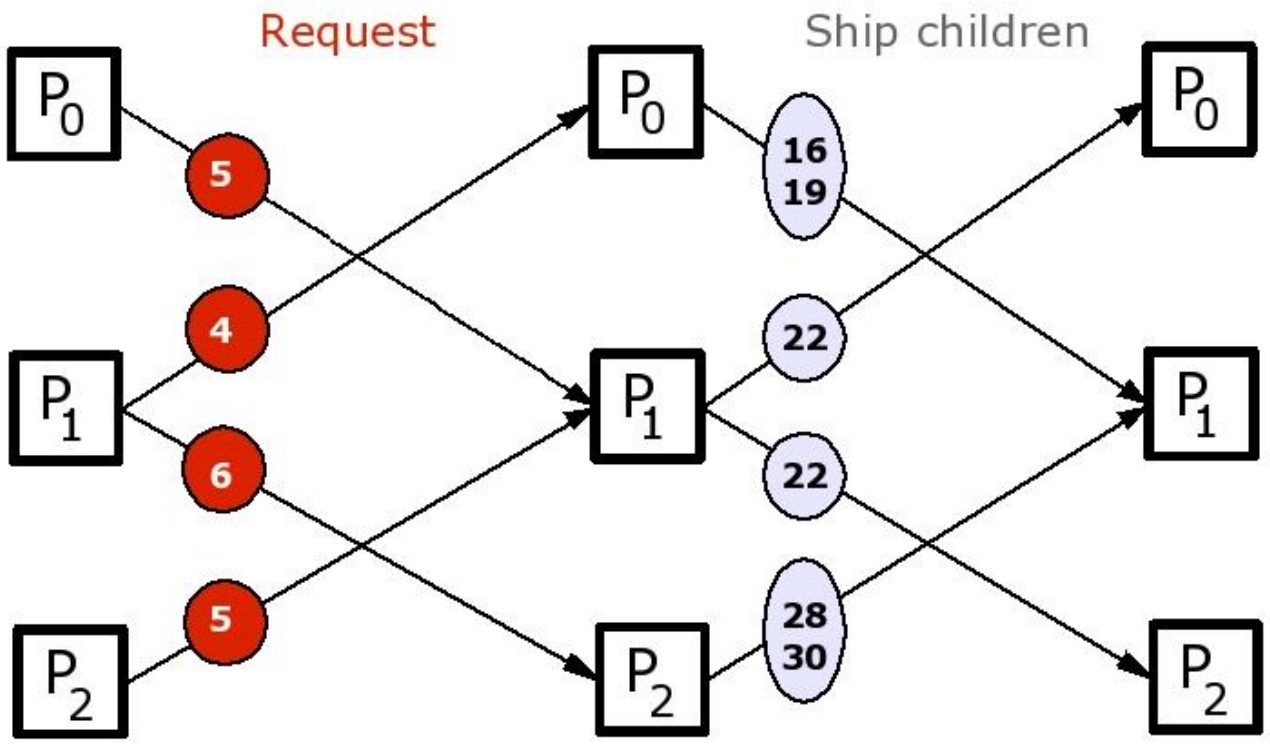
initiate receives for incoming requests
initiate receives for incoming multipole data
send out own request list to owners of missing multipole data

do while (requests pending)
    test for incoming requests
    package and ship back multipole data of child nodes
end do

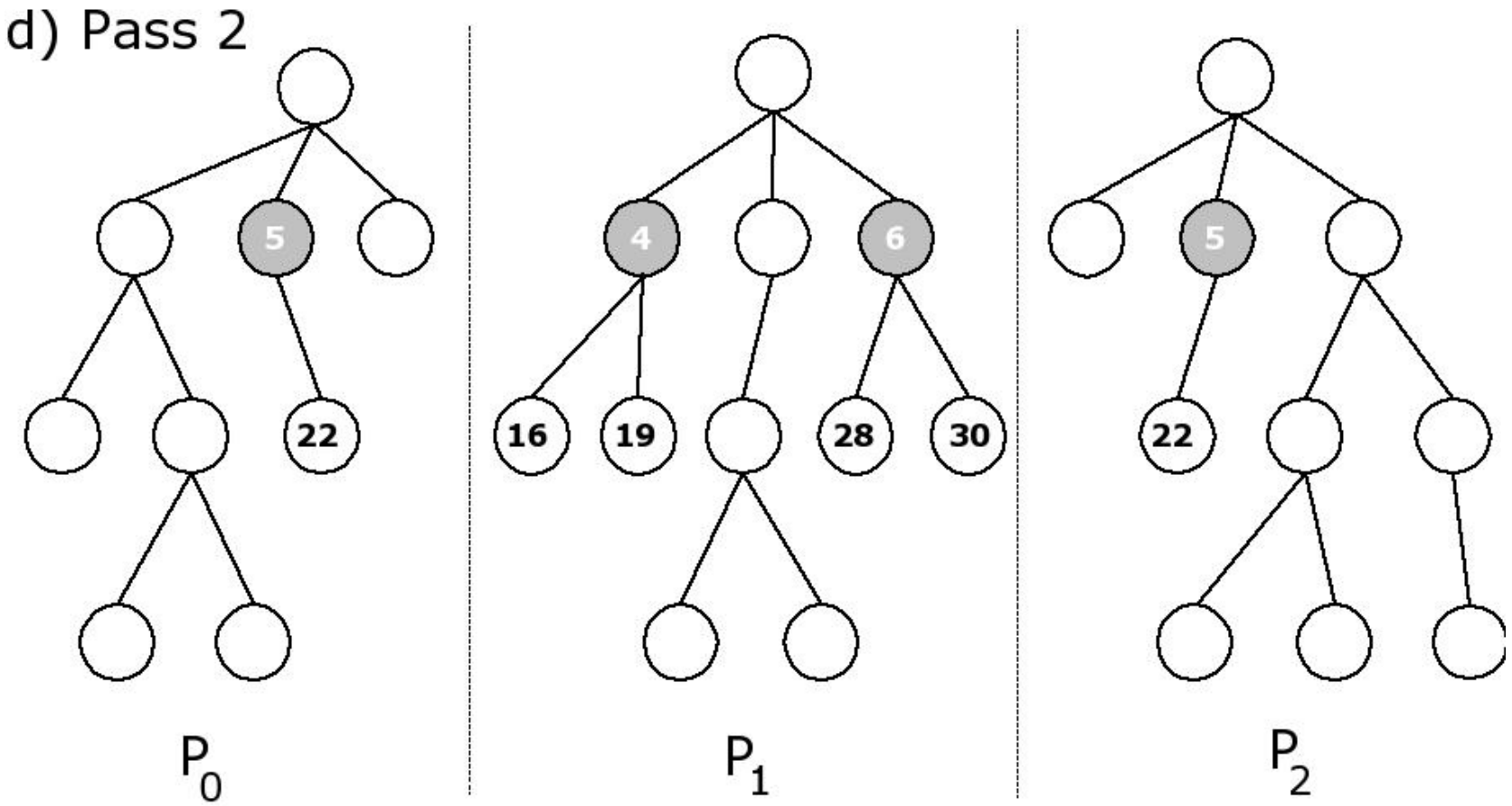
do while (receives pending)
    test for incoming multipole data
    create local copy of multipole for each new node
end do

Exchange nonlocal multipole info

c) Exchange 1

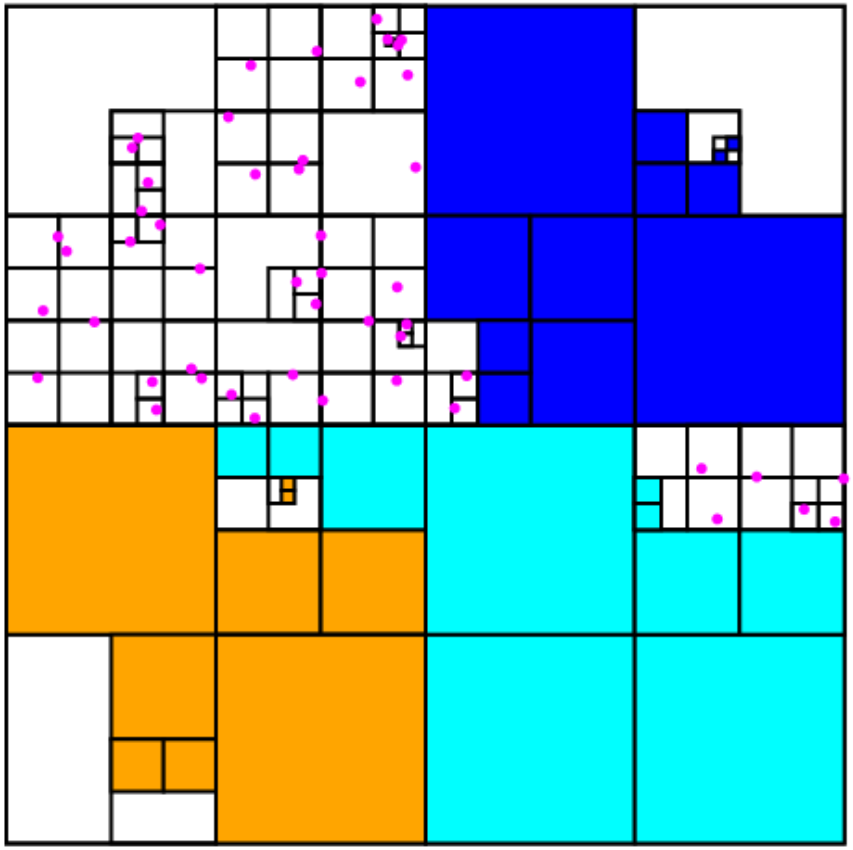


Local walks: pass 2 => Locally Essential Trees

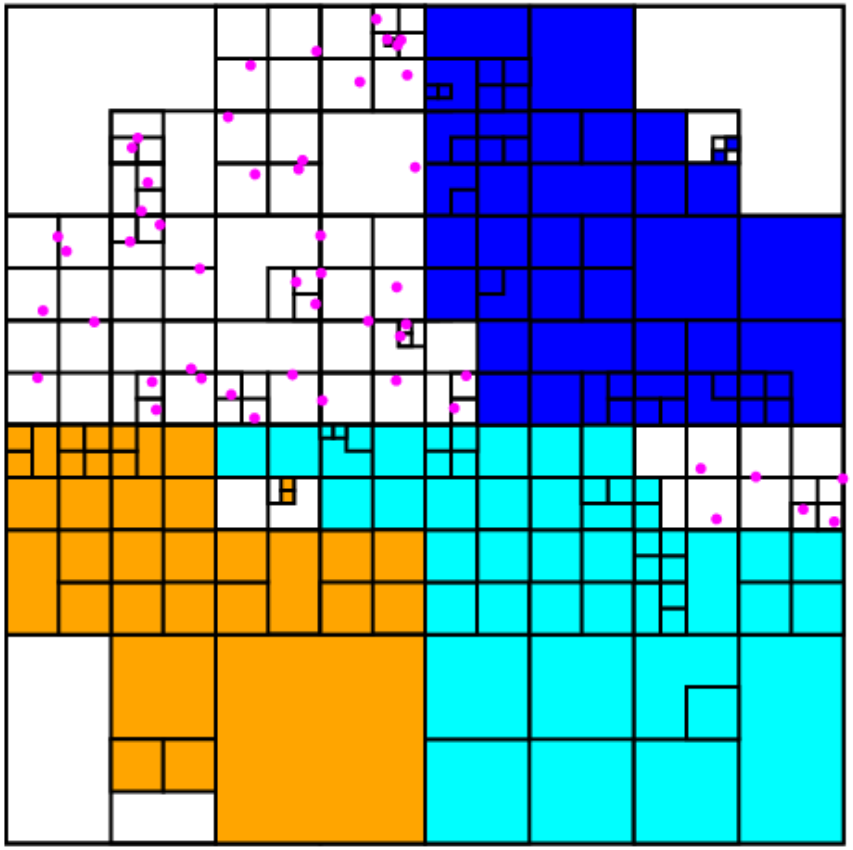


Locally essential trees

Before tree-walk

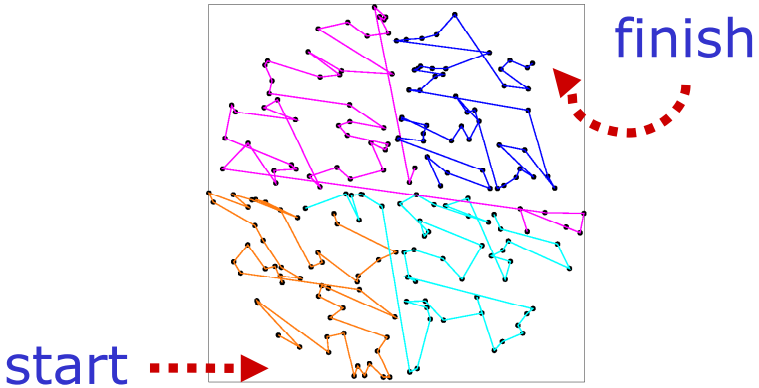


After tree-walk

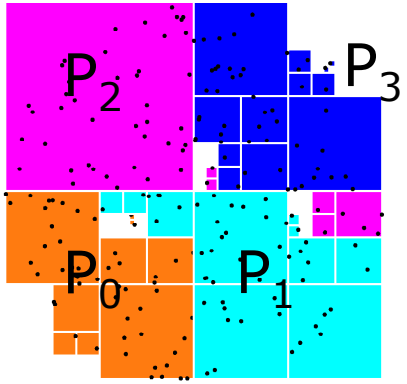


Parallel tree code: summary

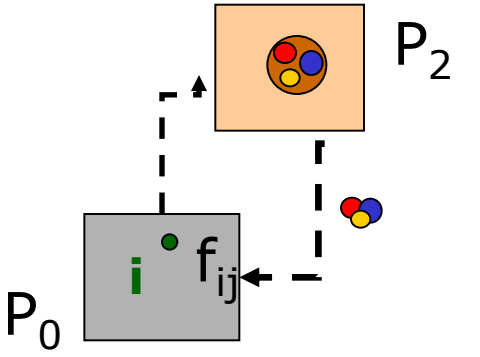
1. Domain decomposition

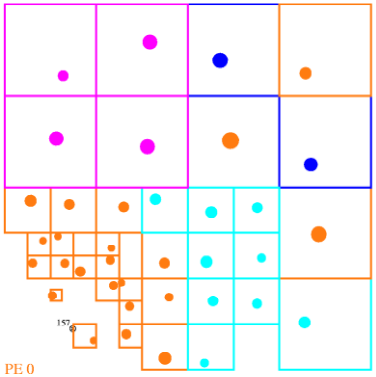


2. Local trees



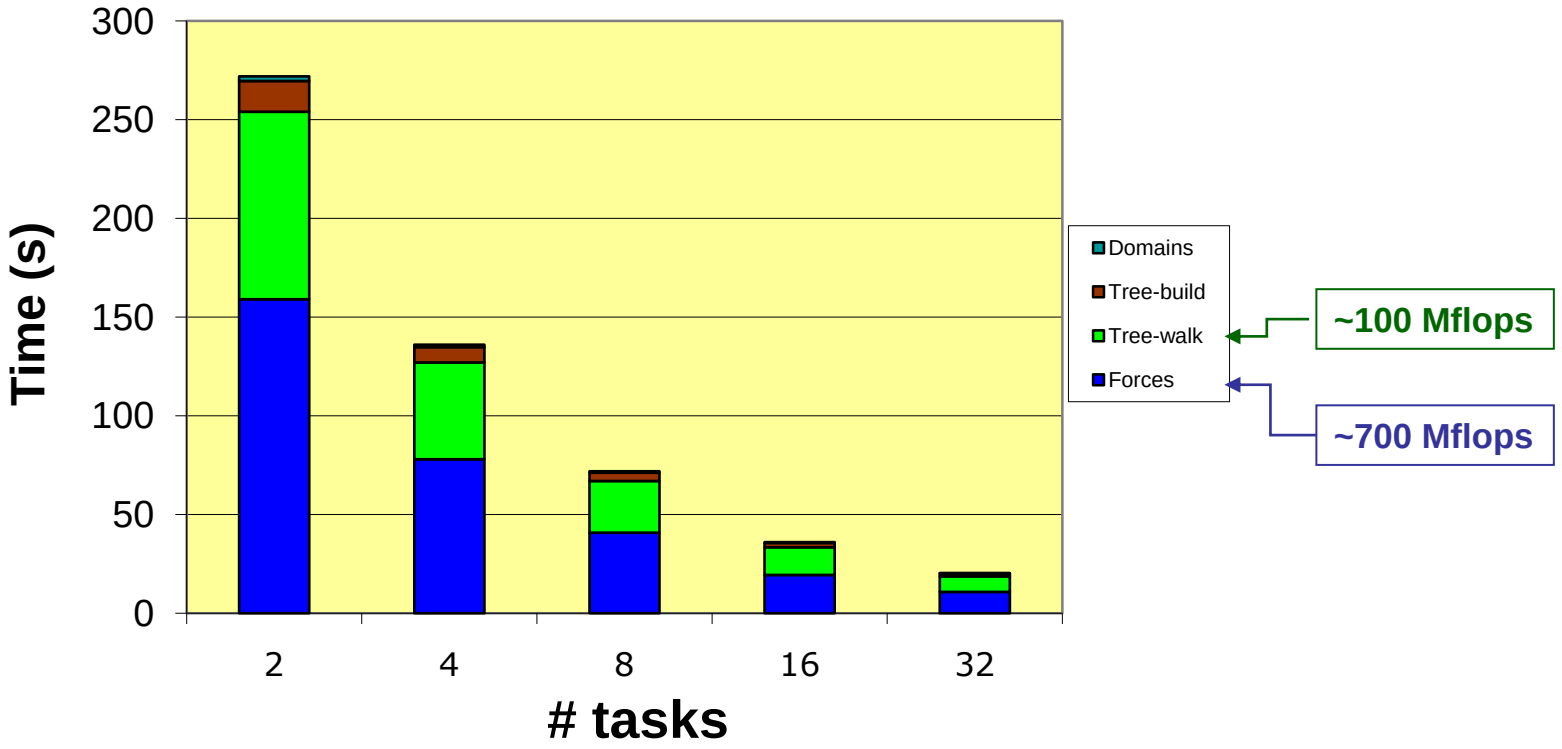
3. Non-local multipoles + interaction lists



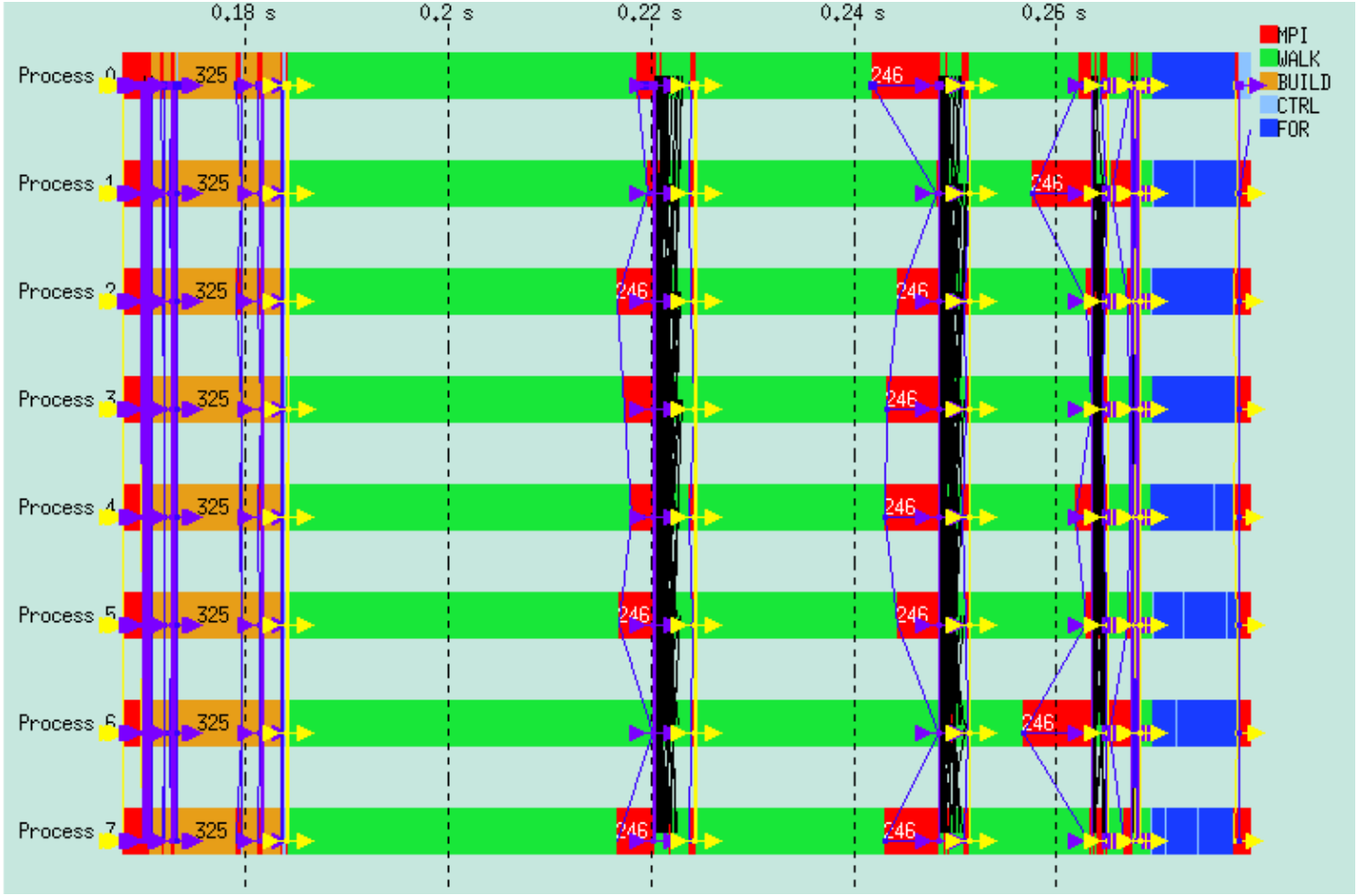


1. Basic multipole tree algorithm (Barnes-Hut method)
2. Parallel tree code
3. Scaling and performance issues
4. Applications

Scaling on JUMP: 10^6-particle sphere

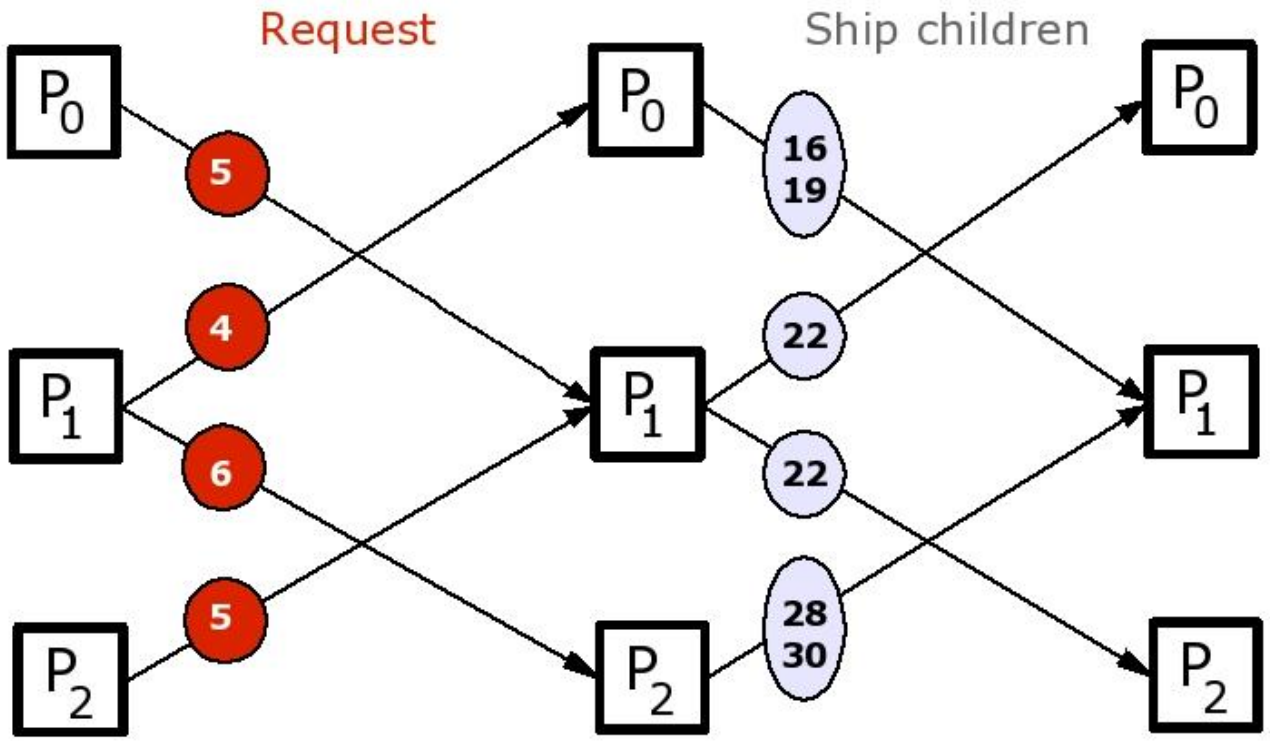


Analysis with VAMPIR tool

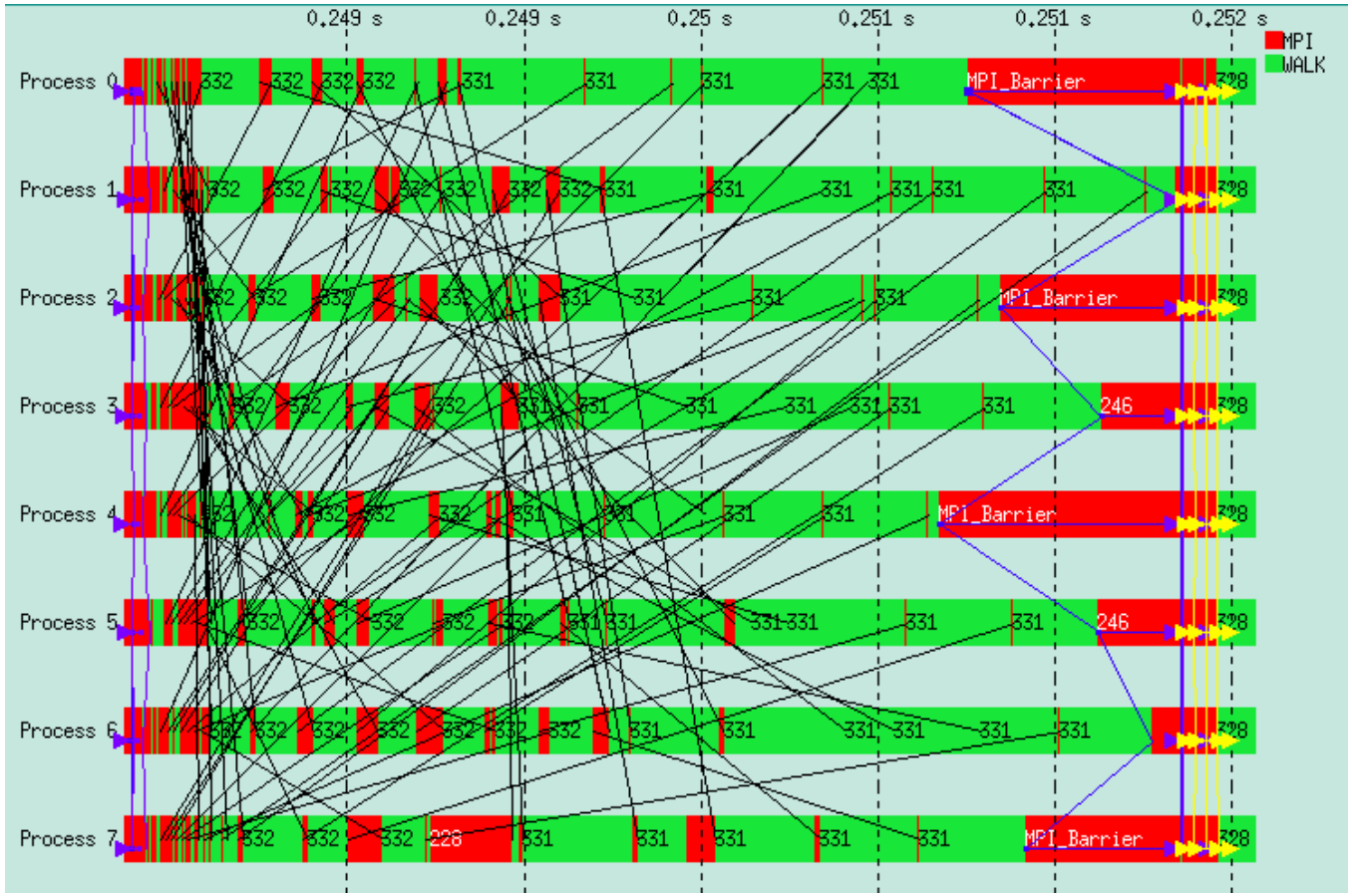


Exchange nonlocal multipole info

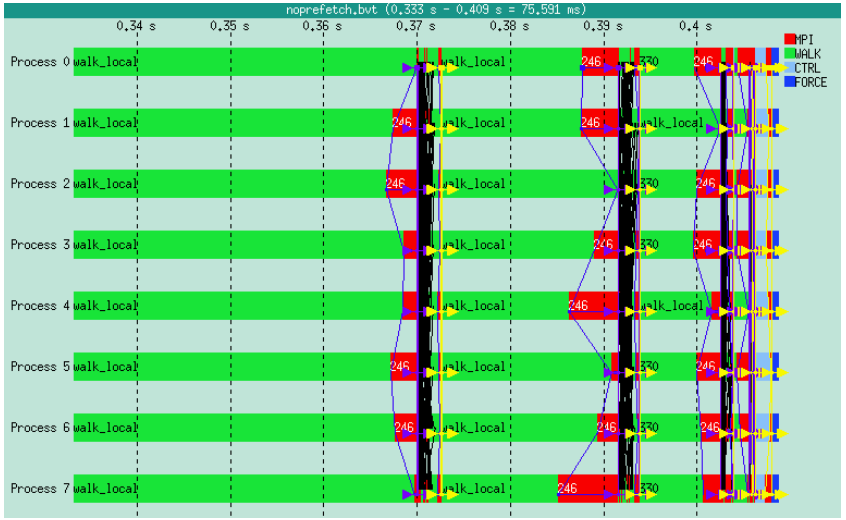
c) Exchange 1



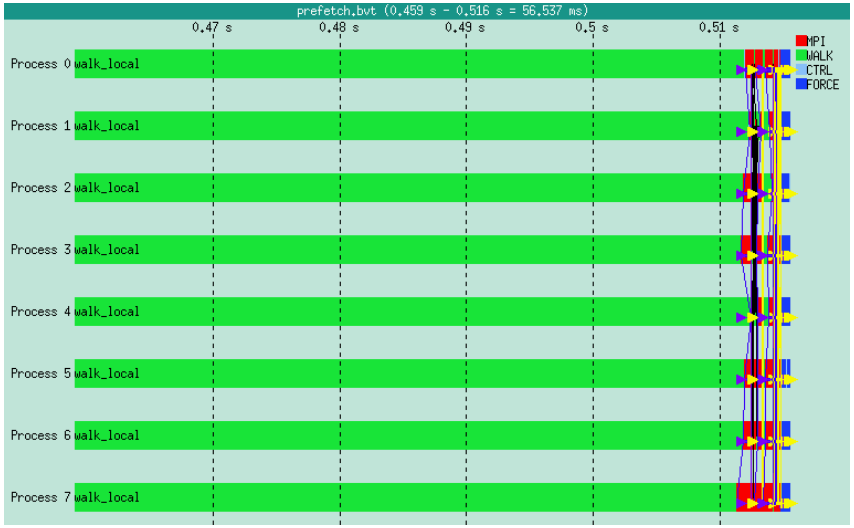
Tree walk: multipole exchange



Multipole prefetching

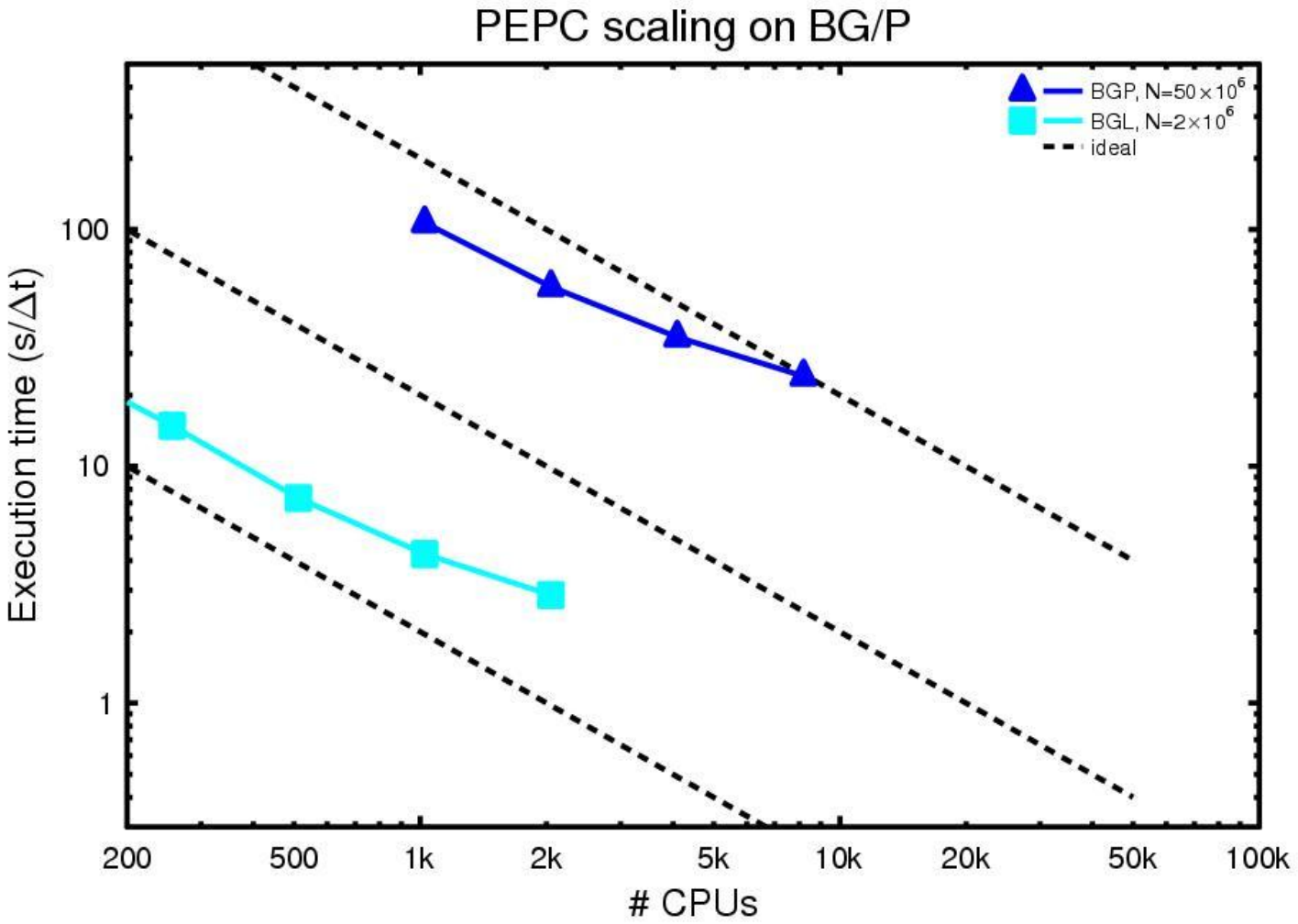


normal walk

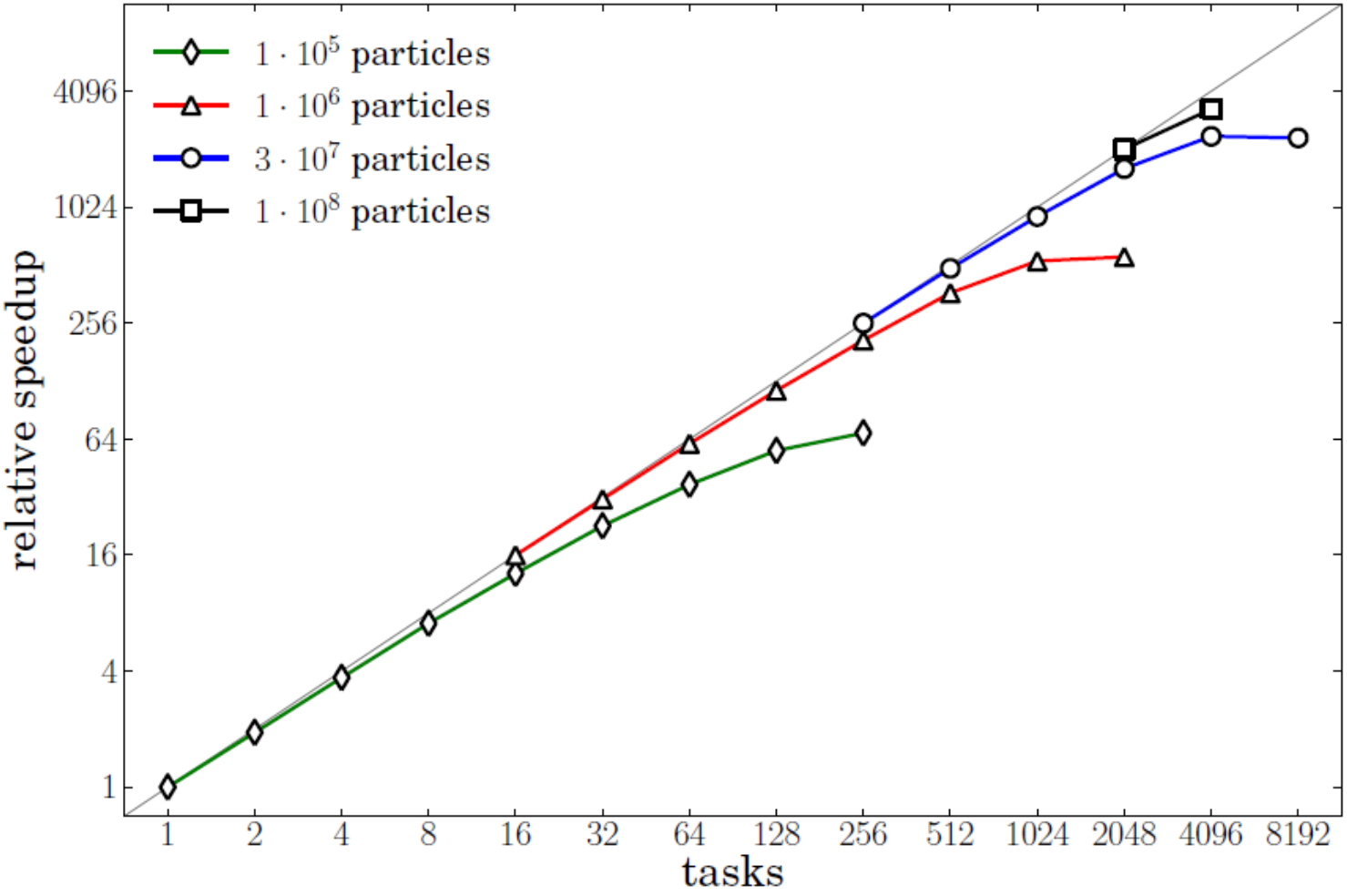


walk after prefetch

Parallel Scalability

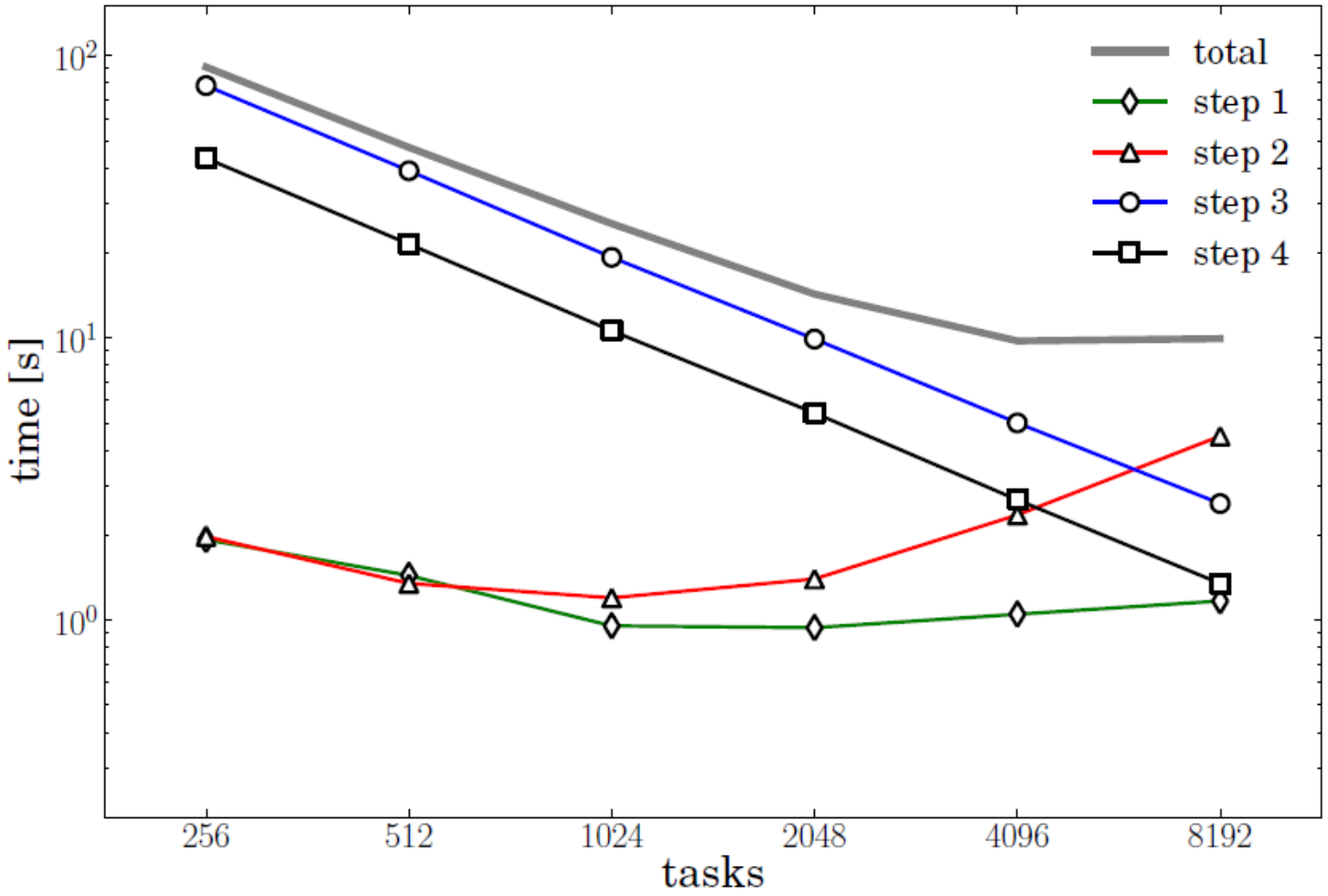


Algorithm scaling (measured)

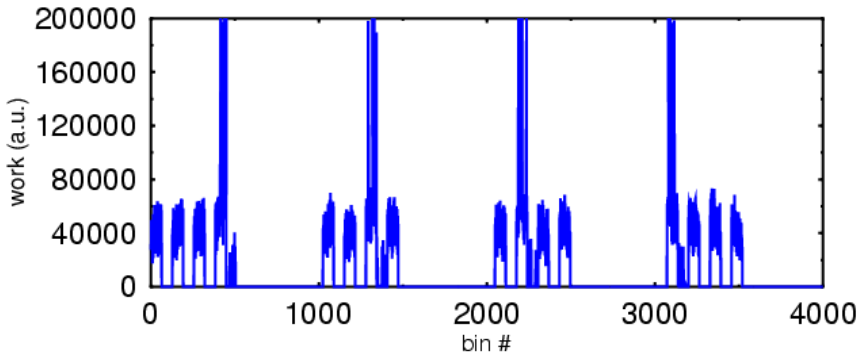


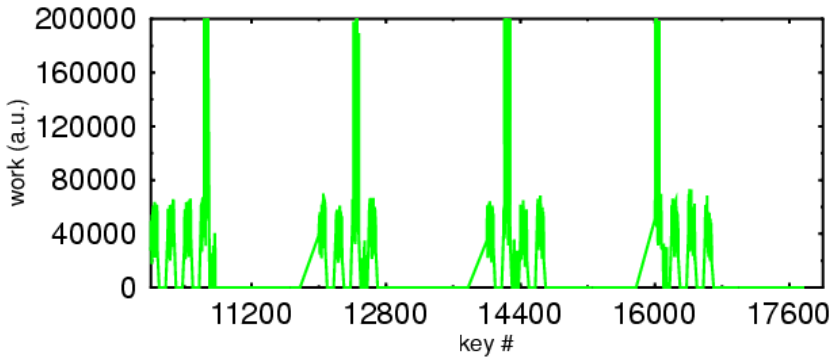
36

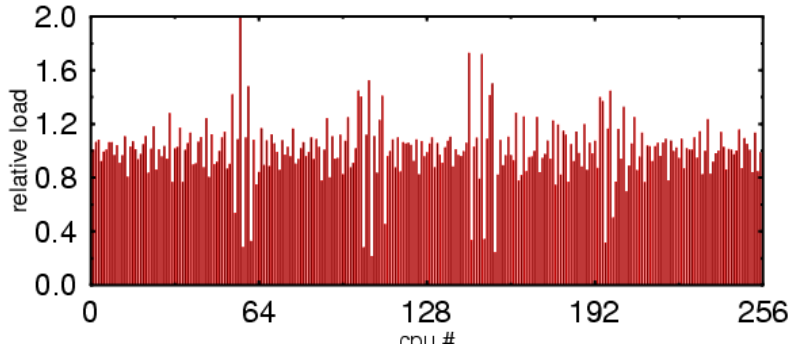
Bottleneck hunt: tree-build with O(P) dependence



Dynamic load balance with oct-sort

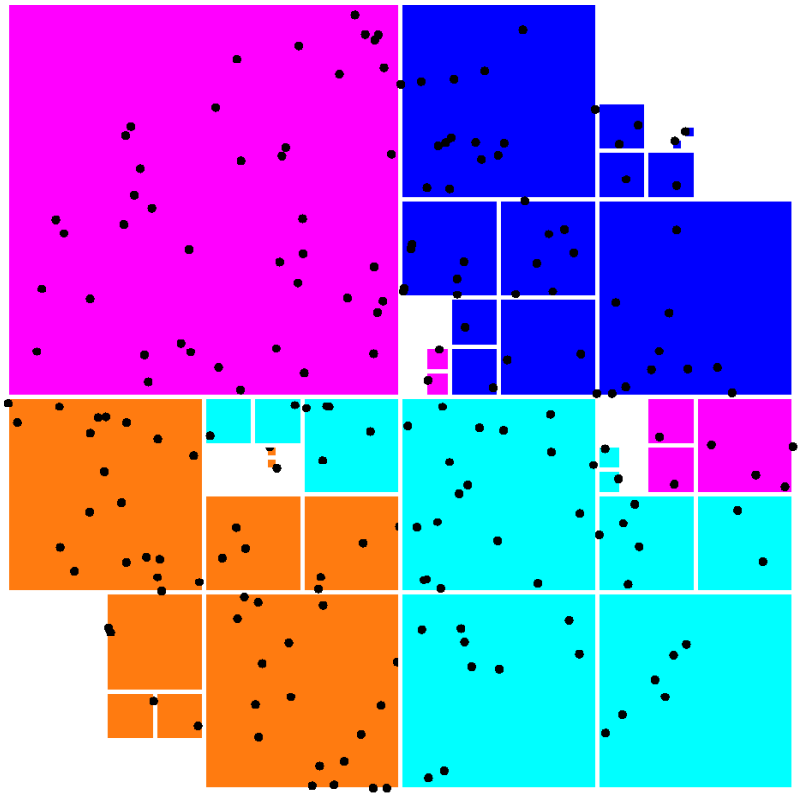




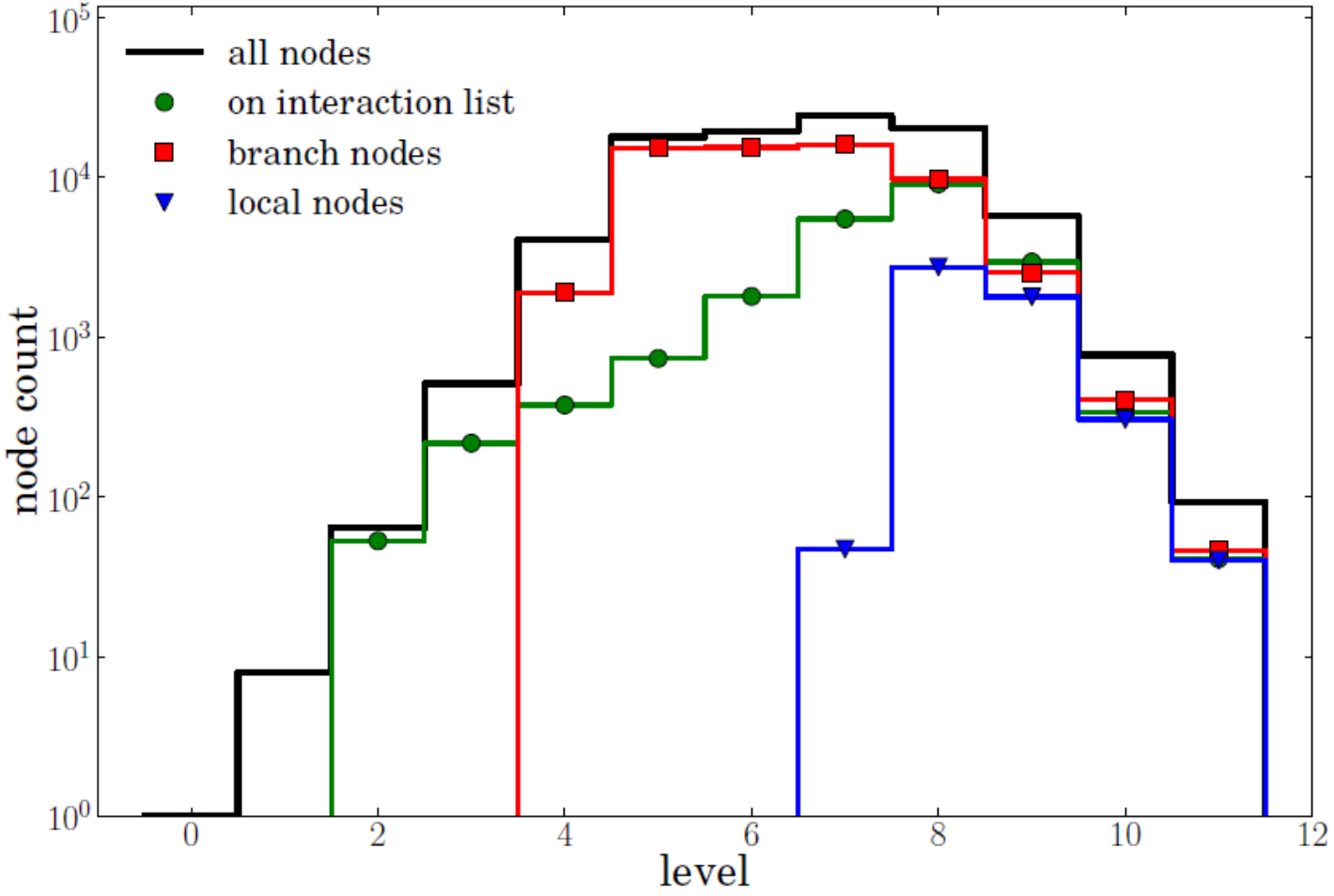


=> Need recursive analysis of particle key distribution in oct-tree structure (adaptive parallel sort)

branch nodes per task roughly constant



Bottleneck analysis: the trouble with branches



JÜLICH
FORSCHUNGSZENTRUM

1. Basic multipole tree algorithm (Barnes-Hut method)
2. Parallel tree code
3. Scaling and performance issues
4. Applications

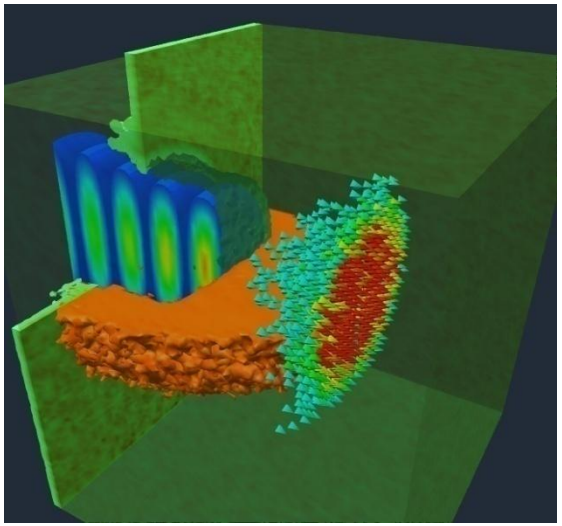
Parallel tree code PEPC:
Pretty Efficient Parallel Coulomb-solver

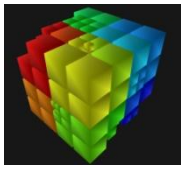
- Hashed-oct-tree algorithm
  using multipole expansions

- Scaling to 8192 cores on BlueGene/P

- Applications:
  - laser-plasma acceleration
  - Periodic (strongly coupled) systems
  - magnetic fusion edge physics
  - stellar accretion discs
  - vortex-fluid simulation

- Future version:
  - magnetic fields
  - implicit integration scheme





PEPC code family

Front end

‘B’ Laser-plasma interactions

‘E’ Strongly coupled Coulomb systems

‘G’ Gravitational systems

‘V’ Vortex fluids

Tree code kernel

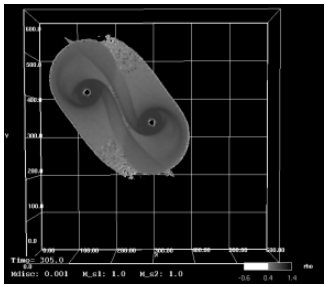
Ipepc

I/O

Particle data

3rd-party applications

Scafacos Library

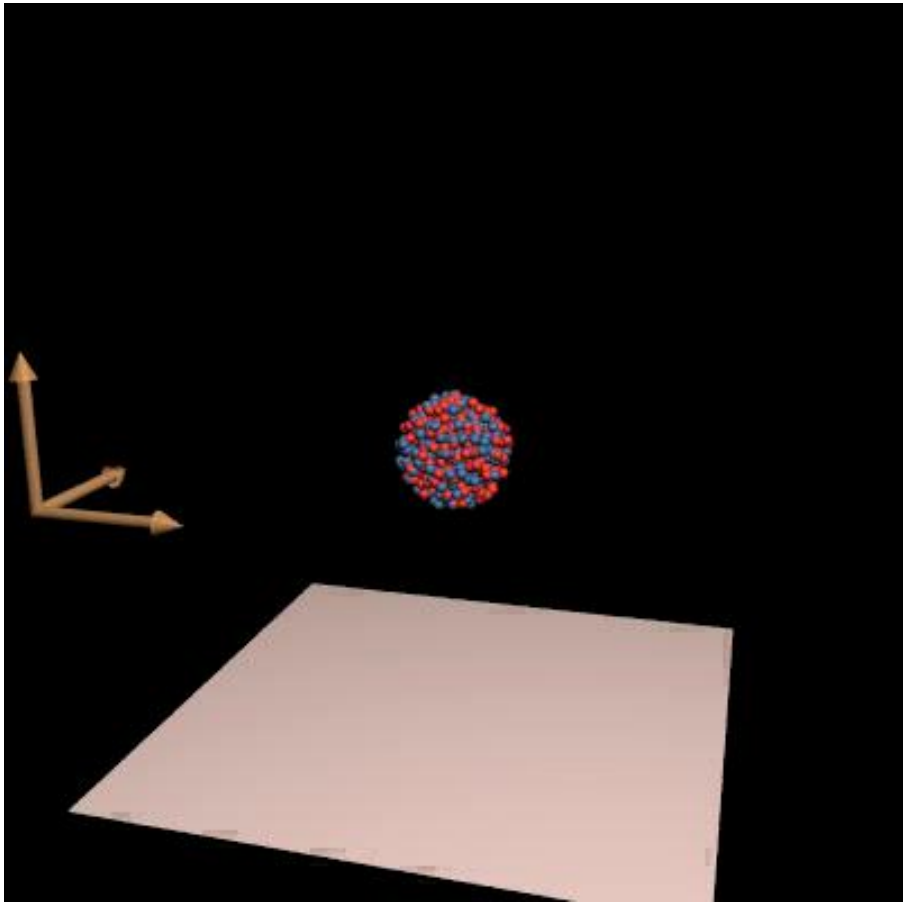


Visualisation

SVN Repository

http://www.fz-juelich.de/jsc/pepc

Coulomb explosion of laser-heated atomic cluster



$\lambda = 0.8\ \mu m,$

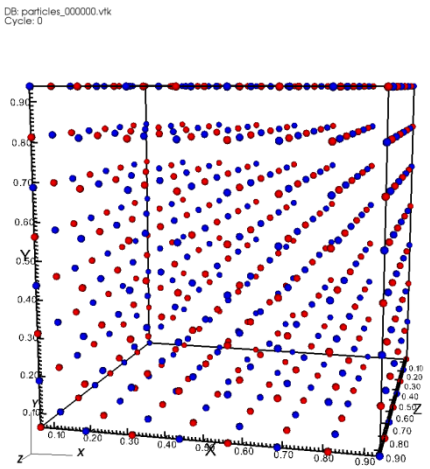
$I = 10^{16}\ Wcm^{-2}$

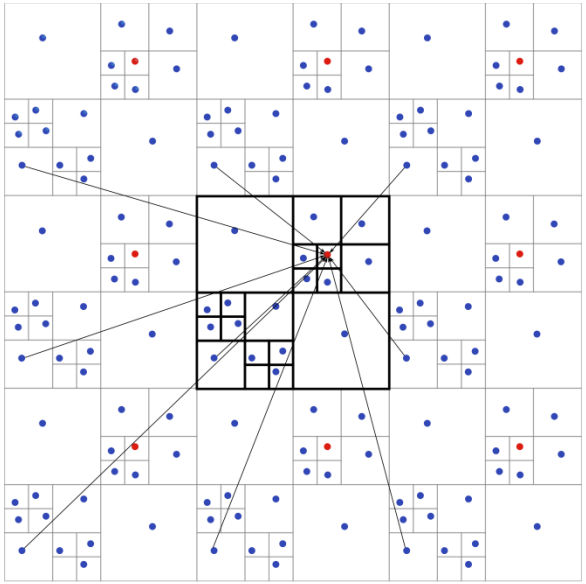
$n_e/n_c = 13$

Extension to periodic systems

Mathias Winkel

- approach stolen from FMM: bipolar expansion of inverse distance in terms of LEGENDRE polynomials
- Pre-calculation of geometry dependent lattice-coefficients using real-space renormalization scheme
- Can treat non-cubic systems as well as periodicity in 1D, 2D & 3D
- computational overhead:
  - O(N) far field
  - O(N log N) near field

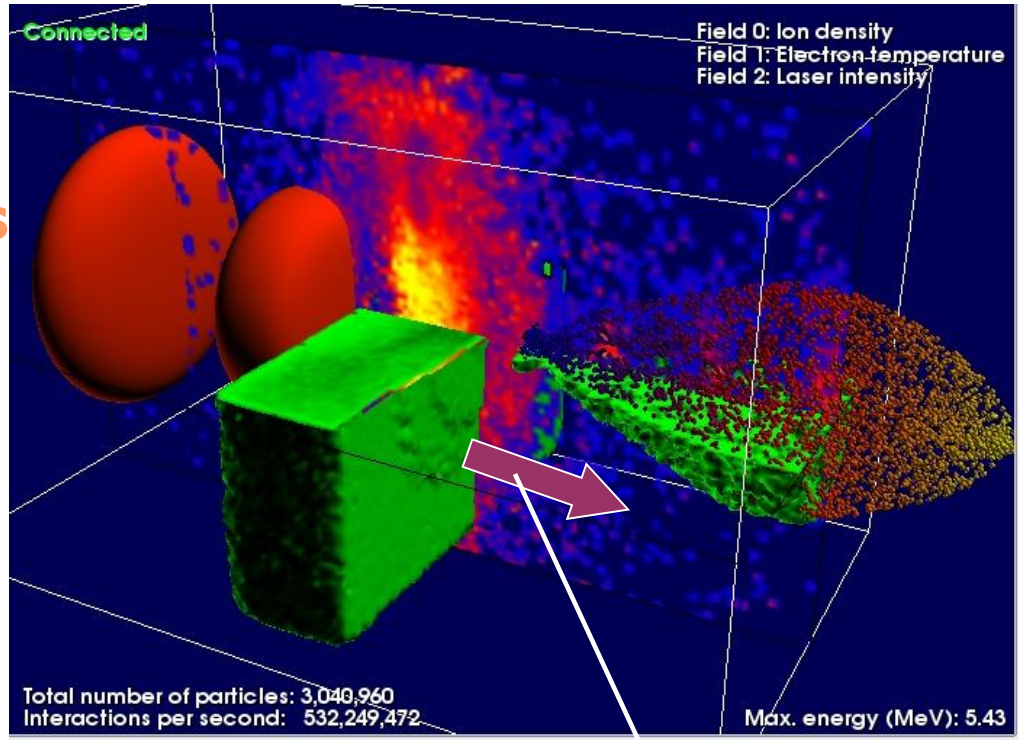




Laser-produced proton beams

Hot electron cloud
T~ MeV

PW Laser:
100 J/100 fs

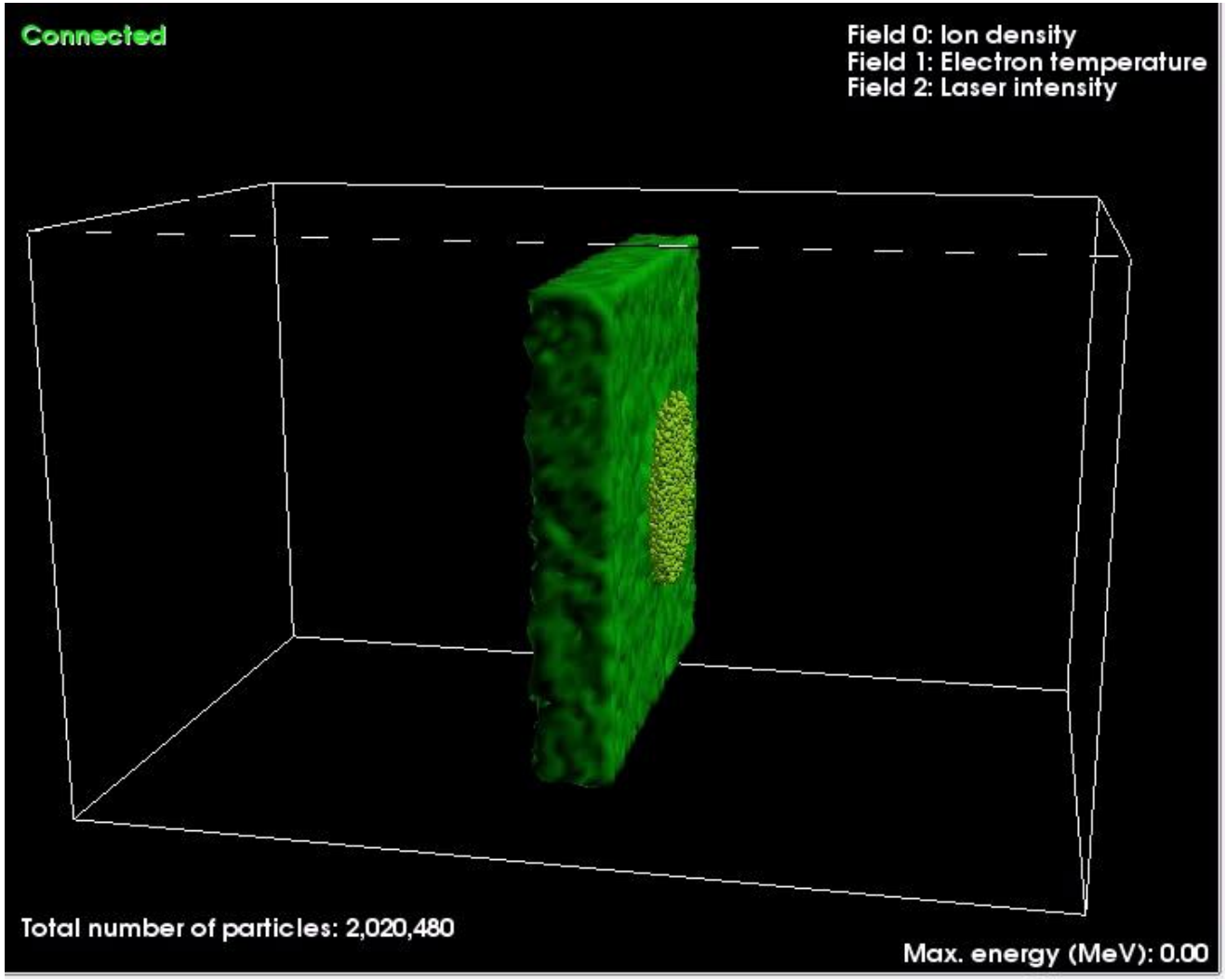


MeV protons

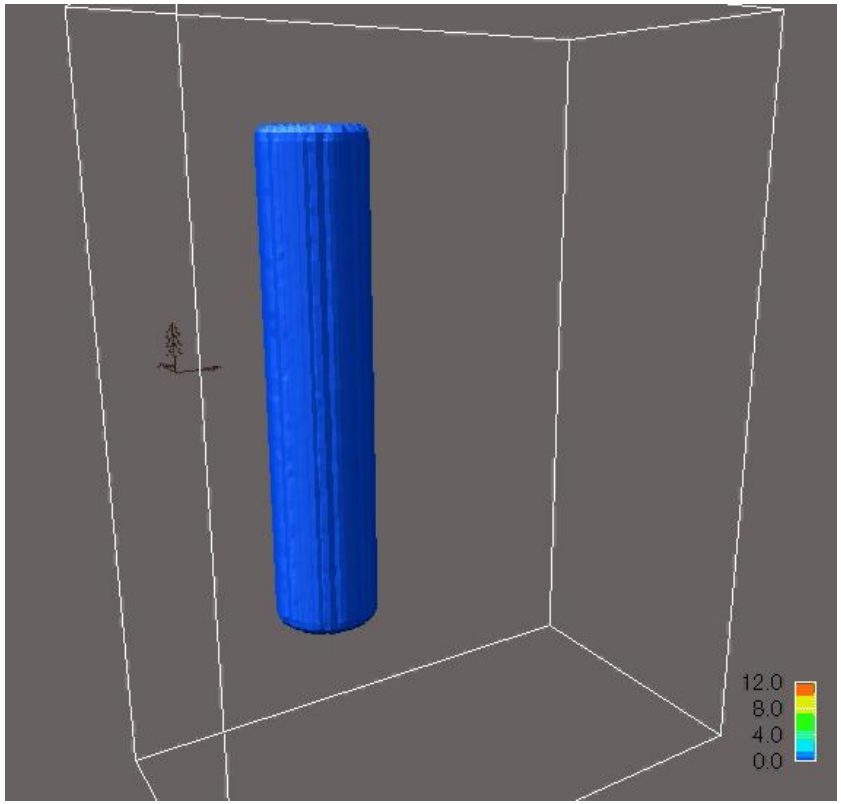
Solid target
(Al, Au foil)

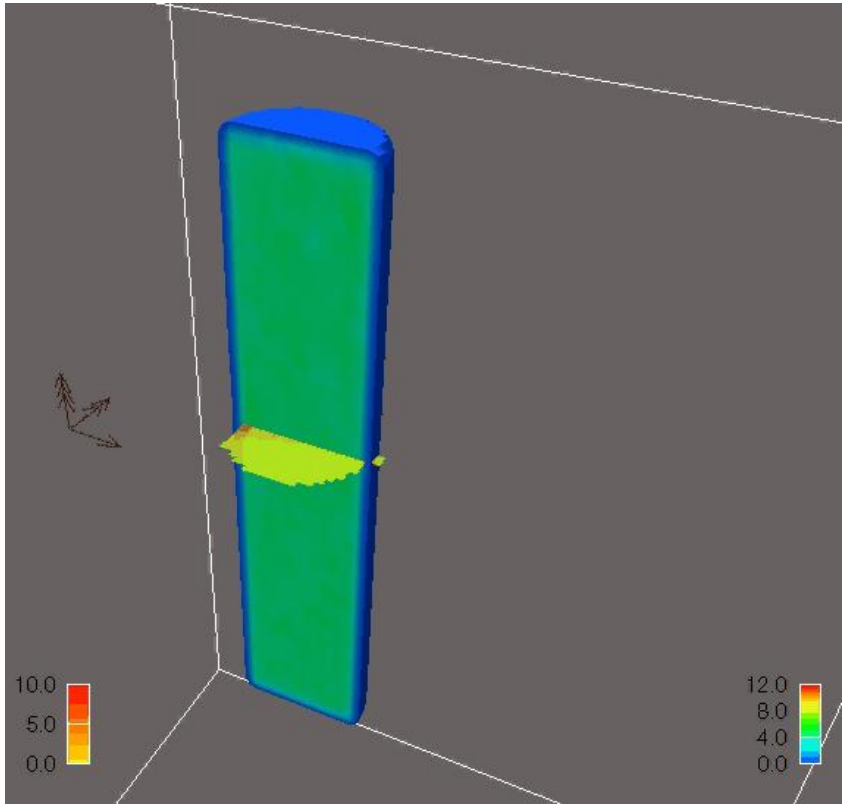
Electric field
~ 10^{12} Vm^{-1}

Proton acceleration (2)



Laser-heated wire: ion density isovolume ($\geq n_c/4$)

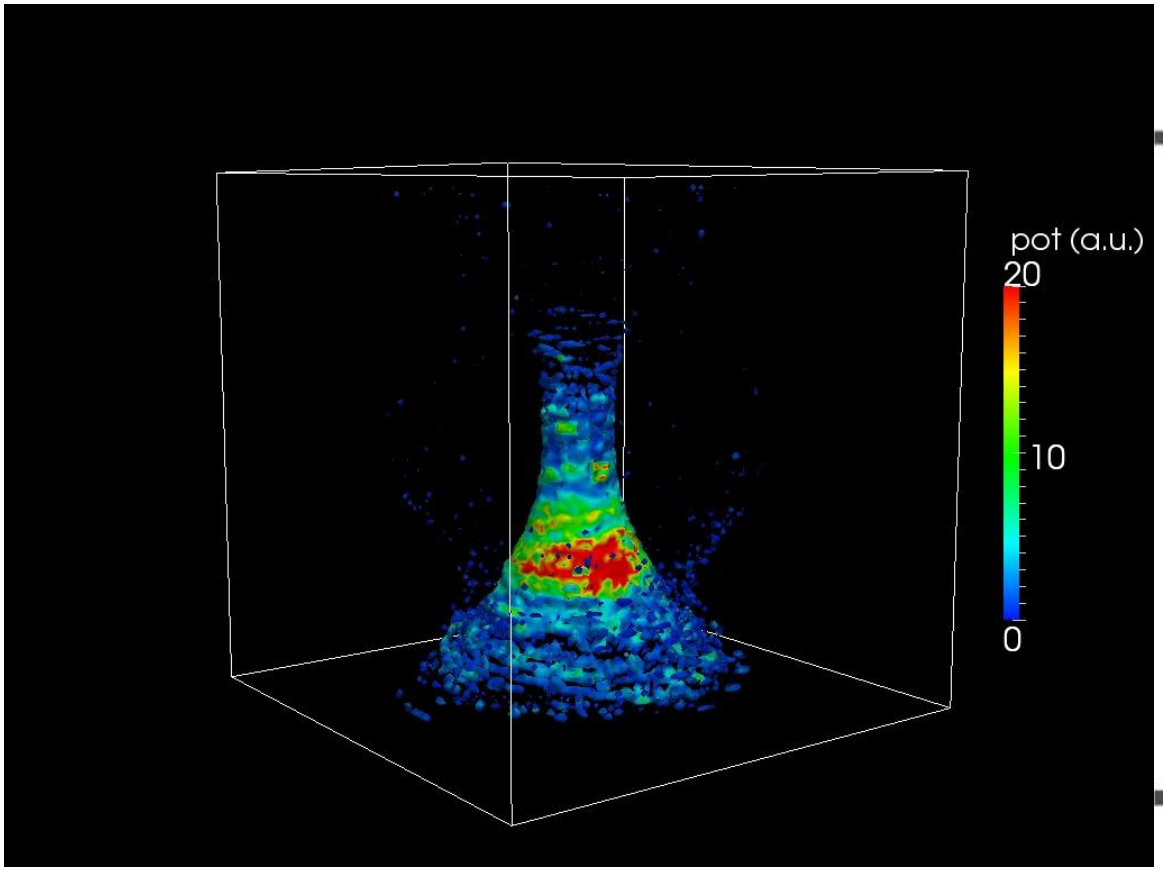


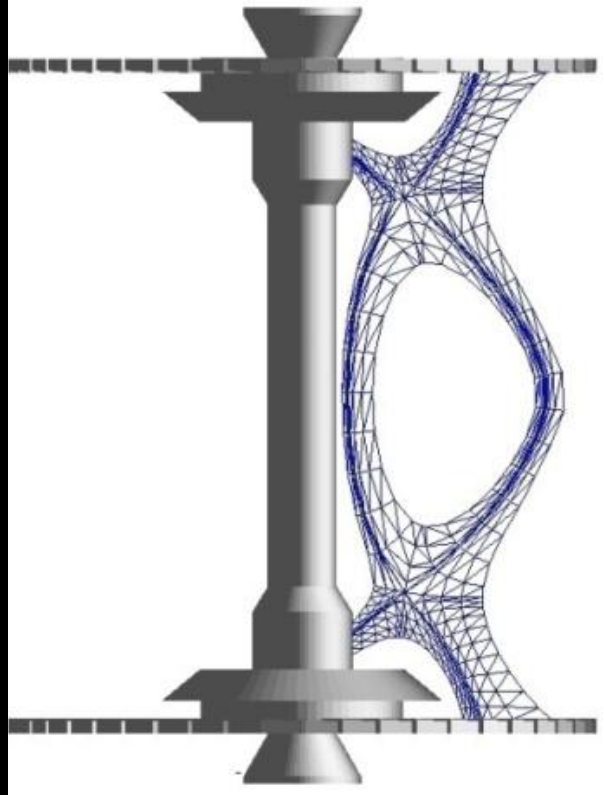


Magnetically confined edge plasmas

Benjamin Berberich

Gas puff simulation

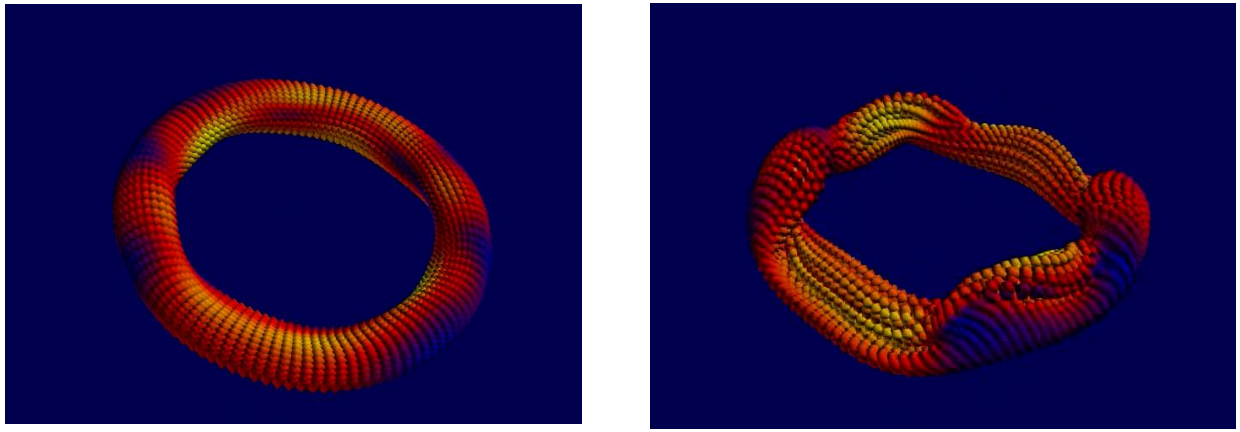




Vortex fluids

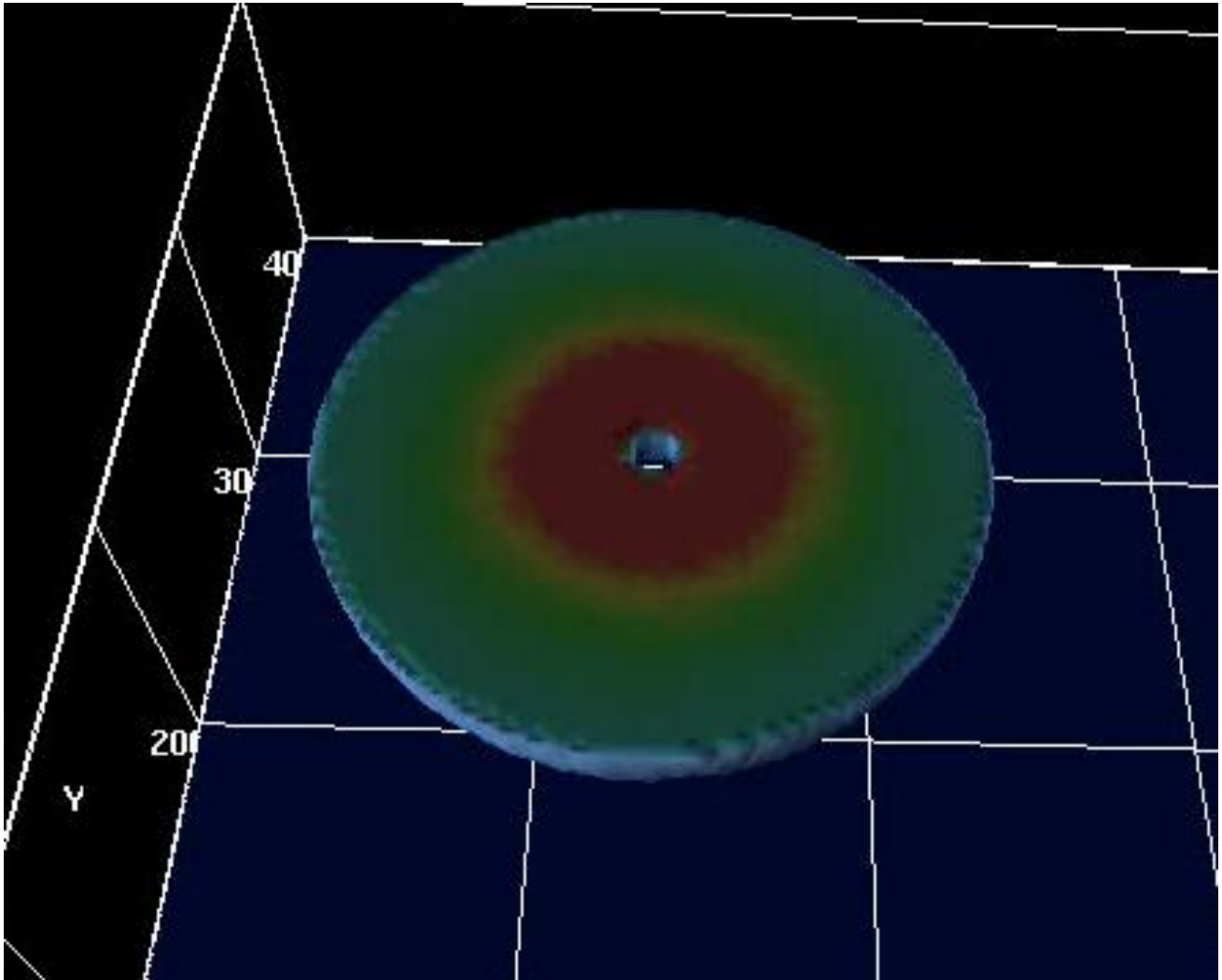
R. Speck

- Mathematical equivalence between Navier-Stokes vortex equations and magnetostatics
- Vortex ‘particles’ must overlap (cf. SPH) → *remeshing*
- Multipole expansion with smoothing kernel satisfying convergence criteria



Example: smoke ring benchmark problem

Astrophysics: self gravitating accretion disc



PEPC: prêt à porter

Credits

Robert Speck	black-box interface; vortex model
Mathias Winkel	periodic boundaries
Lukas Arnold	petascaling
Helge Hübner	Hilbert curve, memory
Benjamin Berberich	gyrokinetics, collisions
Andreas Breslau	neighbour search; SPH
Wolfgang Frings	parallel I/O
…	