Domain Decomposition Performance on ELMFIRE Plasma Simulation Code

F. Ogando[1,3], J. Heikkinen[2], S. Janhunen[3], T. Kiviniemi[3], S. Leerink[3], M. Nora[3]

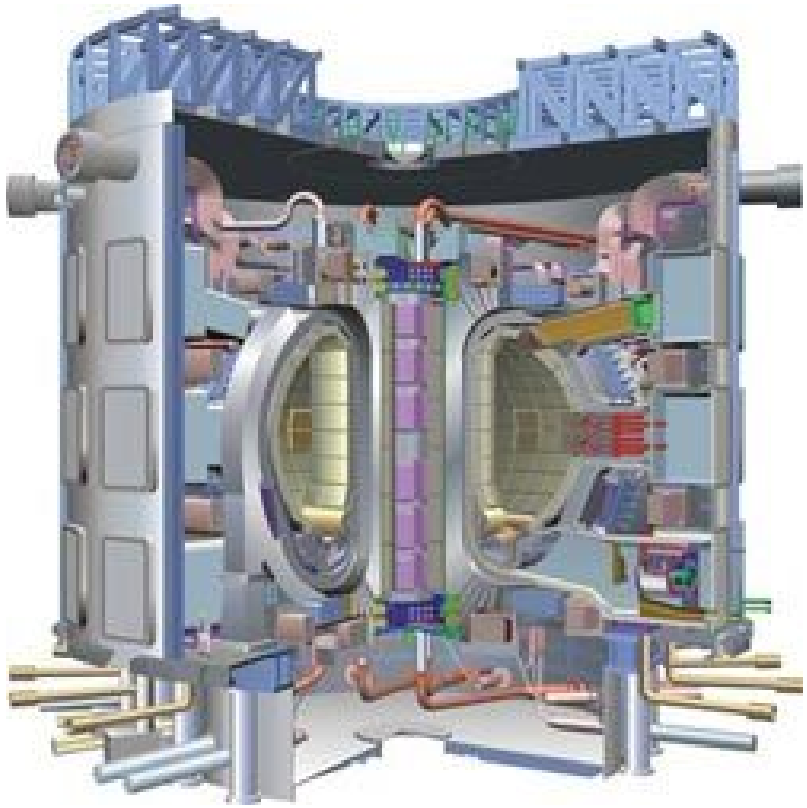
1) UNED, Spain
2) VTT - EURATOM Tekes, Finland
3) TKK - EURATOM Tekes, Finland

Supporting CUG site
CSC

Outline

- Nuclear fusion and plasma physics
- ELMFIRE simulation code
  - Some physics inside
  - Matricial problem
- Domain decomposition
  - New topology
  - Results

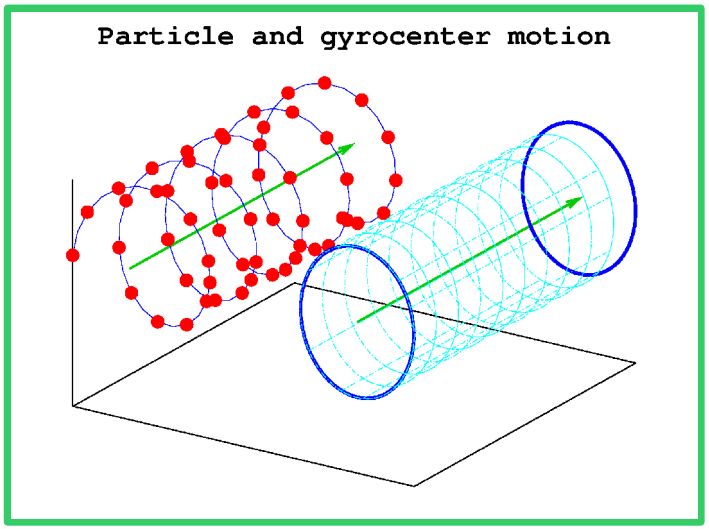
Nuclear Fusion: The energy of the stars

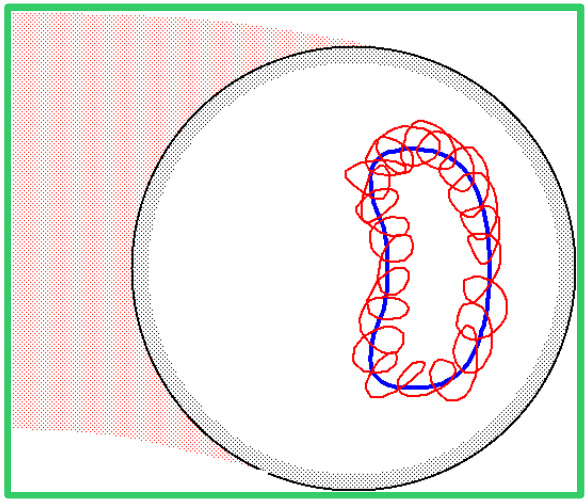


- EU is a main supporter and host of ITER, the biggest civilian fusion reactor ever.
- Keeping a hot reacting plasma confined still poses scientific and technological problems.

Gyrokinetic model for plasmas

- Plasma particles follow field lines with highly oscillating helicoidal movement.
- However their gyration centers follow smoother lines close to B-lines.
- Gyrokinetic model deals with particle gyrocenters, which present smoother transversal trajectories.





The ELMFIRE group

Founded in 2000

International group
Finland
Spain
Netherlands

Main affiliations
VTT
TKK

... but also ...
CSC
Åbo Akademi
UNED (Spain)

ELMFIRE code

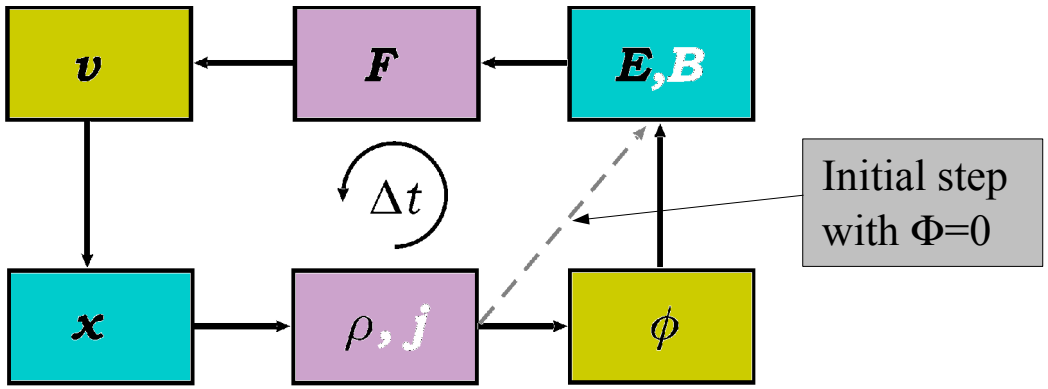
- Full-f nonlinear gyrokinetic particle-in-cell approach for global plasma simulation.
- Parallelized using MPI with very good scalability.
  - Based on free and optionally propietary software: PETSc, GSL or PESSL, ACML, MKL ...
- Benchmarked against other gyrokinetic codes.

Calculation flux in ELMFIRE

Acceleration and increment of velocity

Calculation of forces from fields and velocity

Computation of electric field. Magnetic is given.



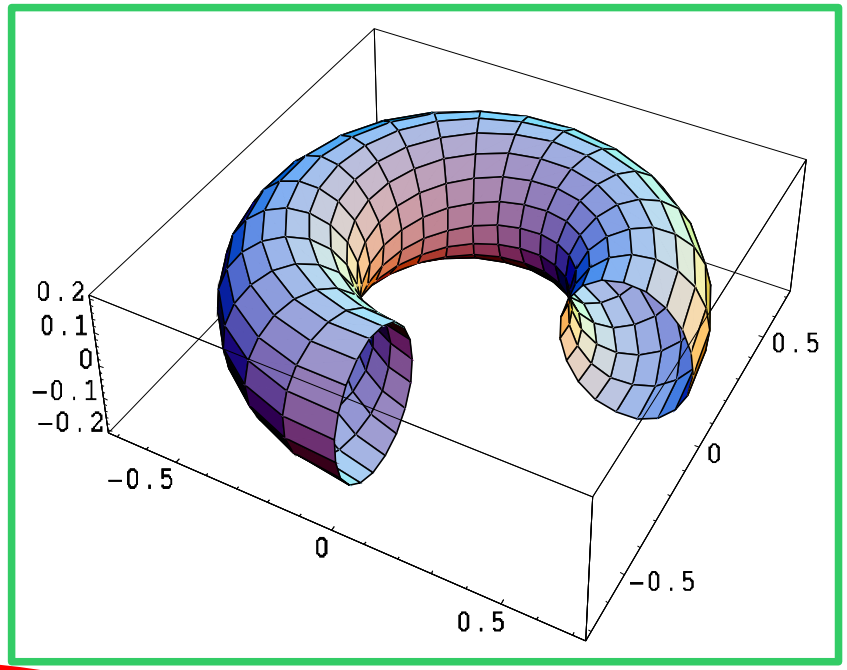
Displacements and new positions. Boundary conditions.

Calculation of density. Current profile fixed.

Resolution of Poisson equation for the electrostatic potential.

Poisson equation

- Particles move in a electrostatic field.
- Field is calculated on a field-aligned 3D mesh.
  - Lines are twisted along the azimuthal direction.



$$\nabla^2\Phi + \frac{q^2}{mB\epsilon_0}\int\left[(\Phi - \langle\Phi\rangle)\frac{\partial\langle f\rangle}{\partial\mu} + \frac{m}{q\Omega}\langle f\rangle\nabla_\perp^2\langle\Phi\rangle\right]\mathrm{d}v = -\frac{1}{\epsilon_0}(q\tilde{n}_i(\vec{r}) - en_e(\vec{r}))$$

Non-local values

ELMFIRE requirements

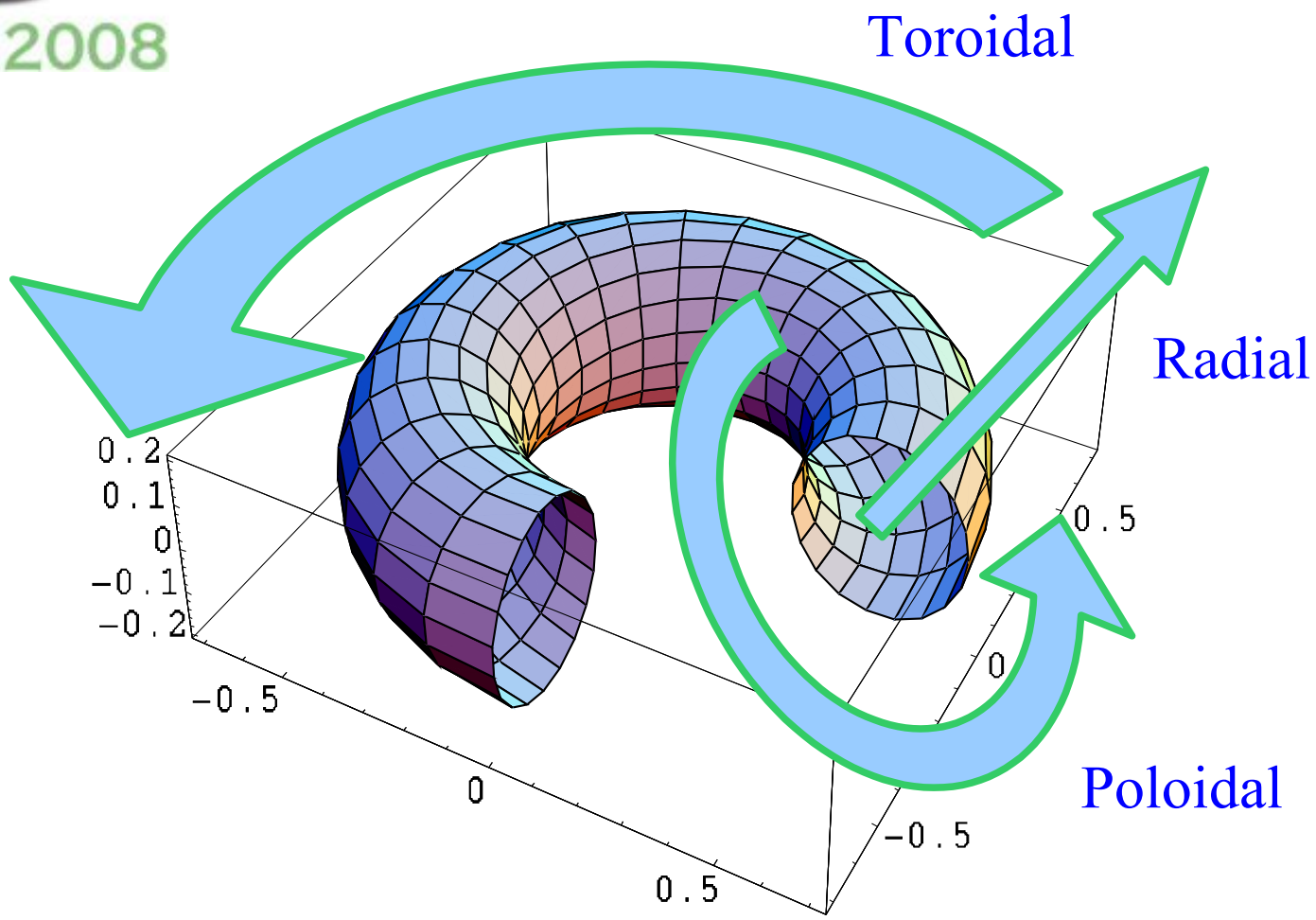
- ELMFIRE is has excellent parallelization in most tasks. Particles are splitted among processors.
- CPU-time (T) is directly related to the number of markers being treated in a single processor (N_P/P).
- Memory usage (M) is proportional to the size of grid (G), since it is not properly splitted among processors.
- The number of particles per cell lies in certain limits.

$$T \propto \frac{N_P}{P};\ \ M \propto G;\ \ N_P \propto G \ \rightarrow\ \ M \propto P \cdot T$$

GK-Poisson problem in ELMFIRE

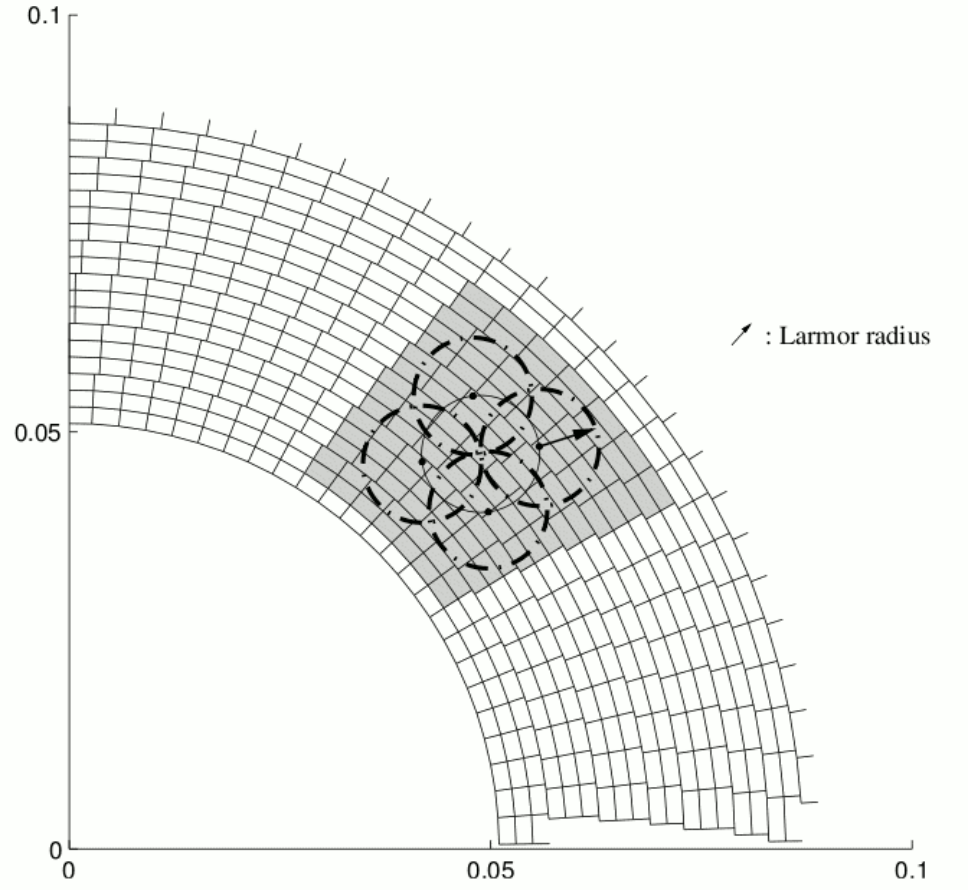
- The code computes electrostatic field so that the calculated trajectories keep the plasma neutral
- The most sensitive part of the dynamics is computed implicitly
  - Future potential changes trajectories, which change densities, which change potential ...
- A linear system is build with implicit drifts
  - Matrix element A_{ij} contains the effect of j-cell potential into i-cell density ($A_{ij} = \partial n_i / \partial \Phi_j$).

Mesh geometry



CUG 2008

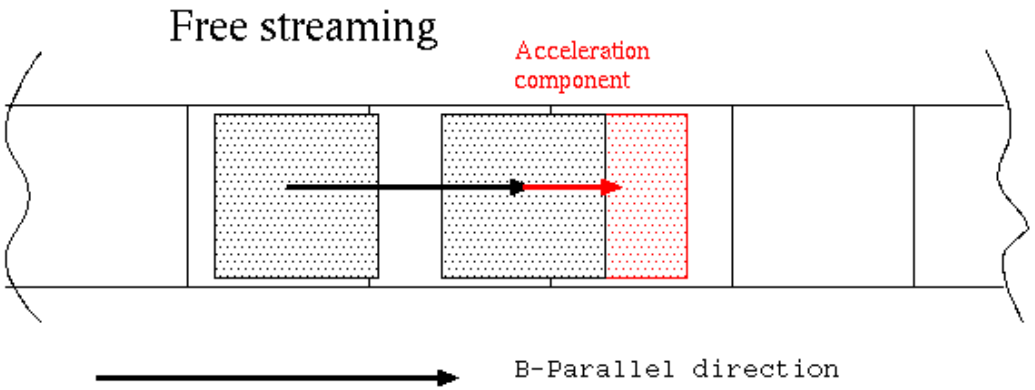
Matrix coefficients from polarization



When particles spin around B-field, they cross several cells surrounding the gyrocenter

Electron parallel treatment

$$\Delta x_{fs} = v \cdot \Delta t \qquad \Delta x_a = \frac{1}{2} a \Delta t^2 = \frac{qE_{\parallel}}{2m} \cdot \Delta t^2$$



A.B Langdon et al (1983)

- E field in Δx_a is calculated at advanced time but at the position after free streaming
  - We demand that $|\Delta x_{fs}| >> |\Delta x_a|$. It constrains Δt

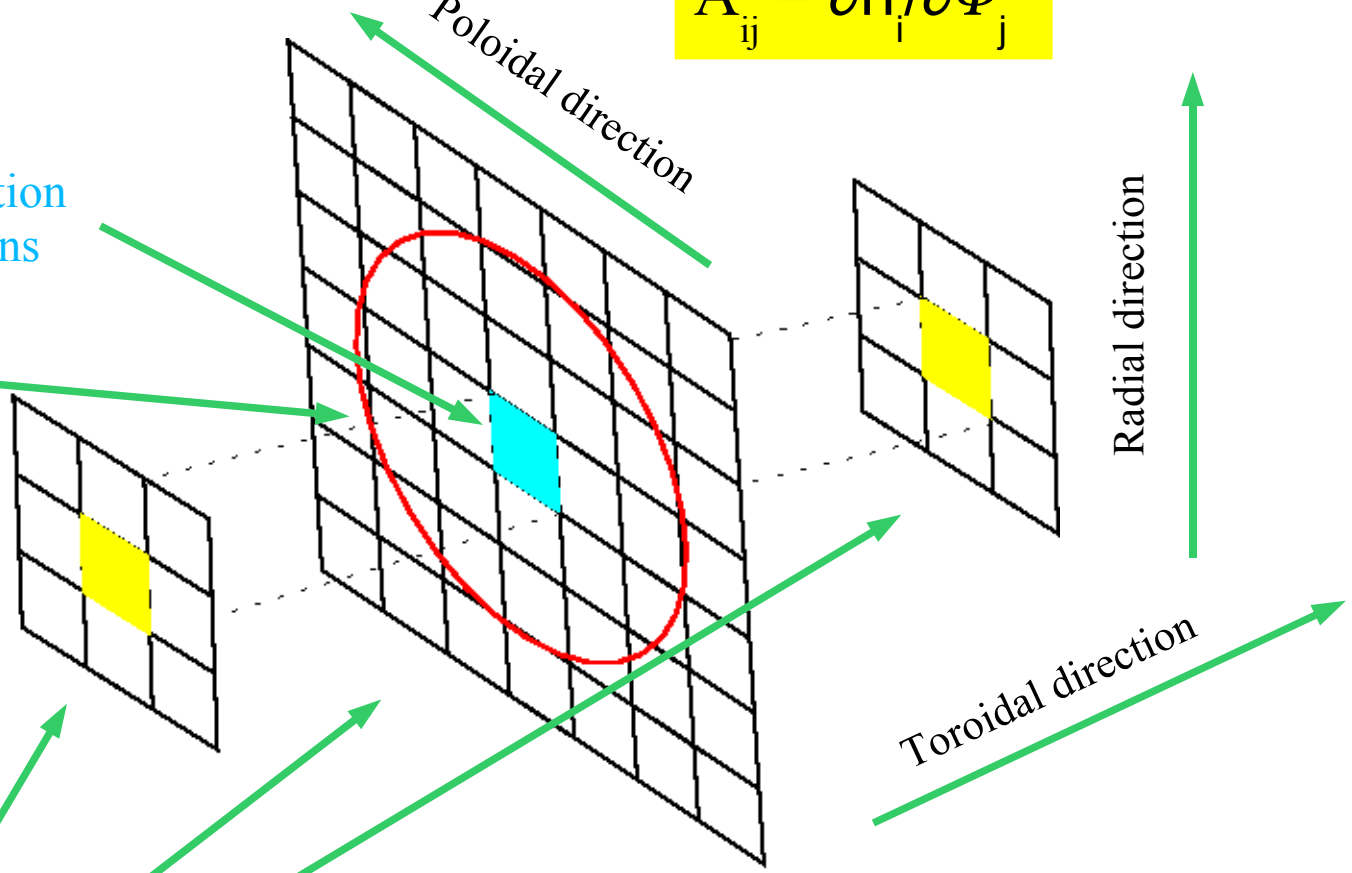
$$\Delta t \ll \frac{\sqrt{8 m_e kT}}{e\,|E_{\parallel}|} \qquad \begin{cases} kT = 100\ \mathrm{eV} \\ E_{\parallel} = 50\ \mathrm{V/m} \end{cases} \quad \Rightarrow \quad \Delta t \ll 1\ \mu\mathrm{s}$$

Overall matrix coefficients

$$A_{ij} = \partial n_i / \partial \Phi_j$$

i-cell under consideration suffer density variations

Larmor radius for calculation of gyroaverages



j-cells whose potential produce variations of i-cell density

And this for every single cell in the system in every processor.

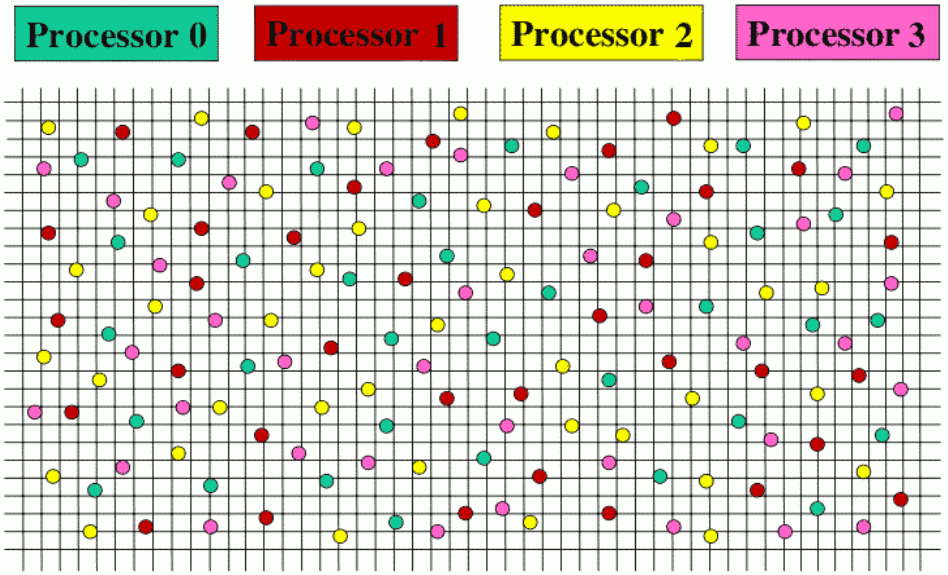
Memory usage

- The storage of matrix coefficients in those boxes takes most memory of the system, posing a real limit.
  - The boxsizes are equal for all cells while Larmor radius is not.
  - Matrix is not distributed across processors. Poor memory scalability.
- Typical memory requirements for real case:
  - ncell=2×10^5, boxsize=500 → 800 MB. Unacceptable!!

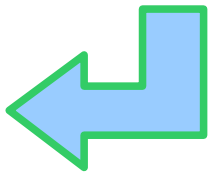
Domain decomposition

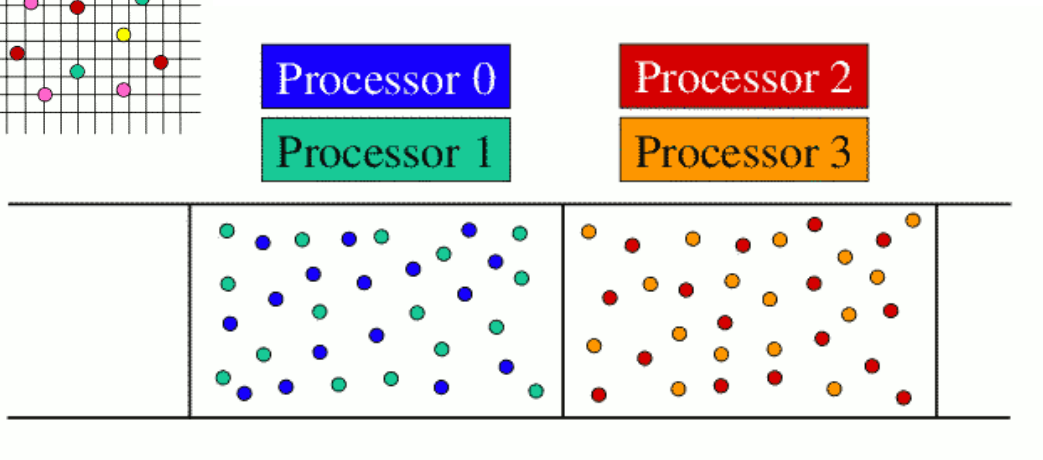
- Key question for DD: to what matrix coefficients is a given particle contributing?
  - Polarization calculations are contained in its Z-plane, both density variations (i-cells) and potentials (j-cells).
  - Electron parallel movement includes also neighbouring toroidal planes (both i- and j-cells), around particle.
- At last a certain particle only affects its toroidal plane and locally the neighbouring ones.
  - If we keep process particles inside a toroidal domain, their coefficients will NOT span the whole torus.

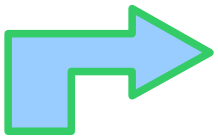
Particle distribution



Original particle distribution



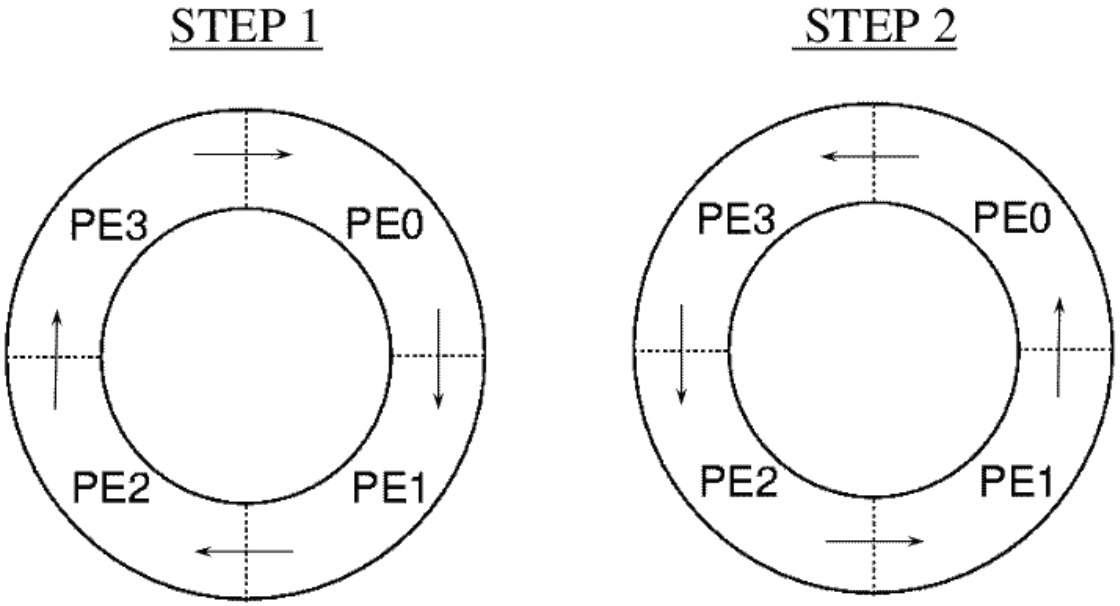




Distribution under toroidal domain decomposition

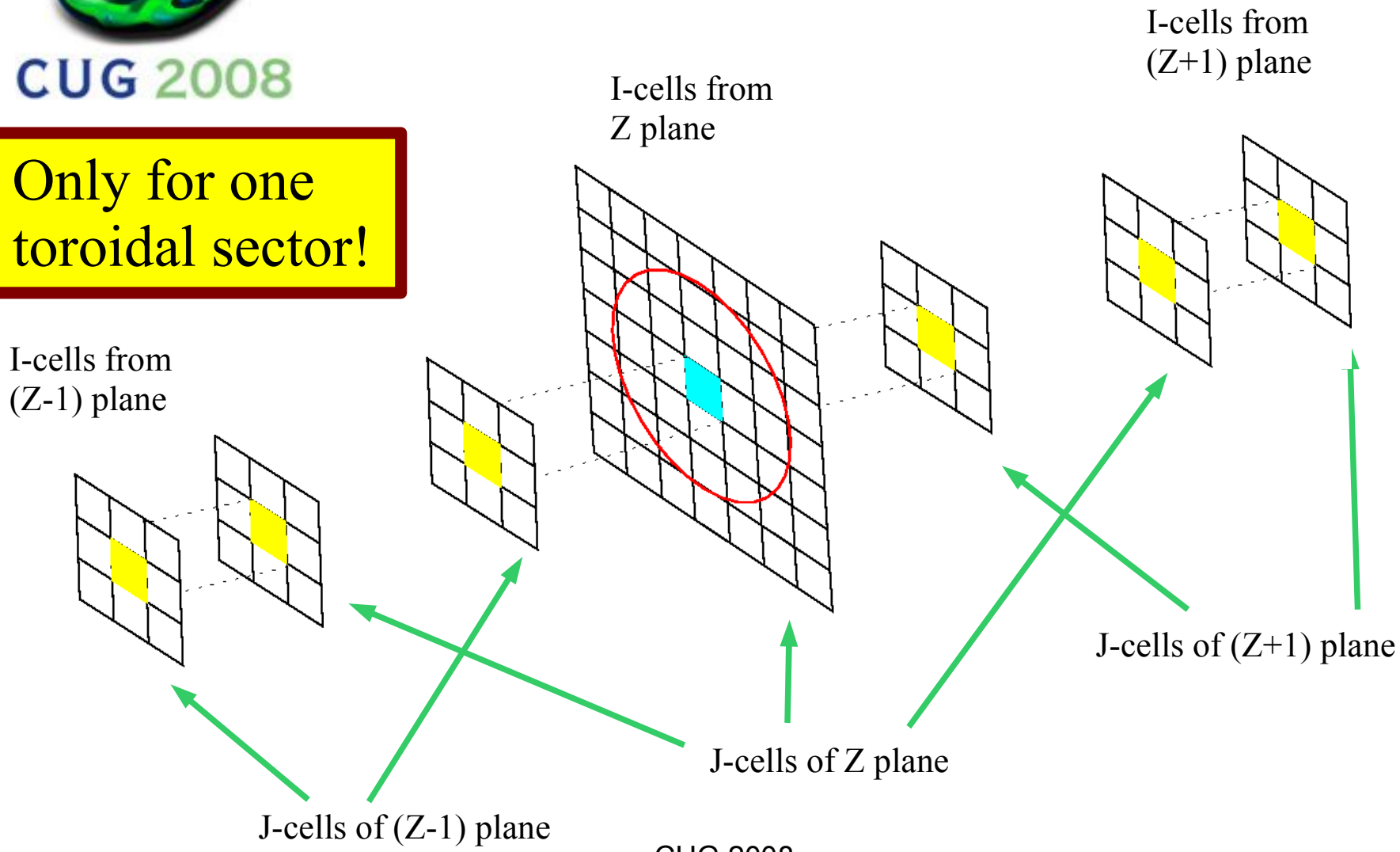
Particle transfer

- Particles have to be transferred to the proper domain every time they cross toroidal domain boundaries.
  - Simultaneous transfer (MPI_SENDRECV) in few steps.
  - Particle number per processor is bounded in practice.



Coefficients in memory

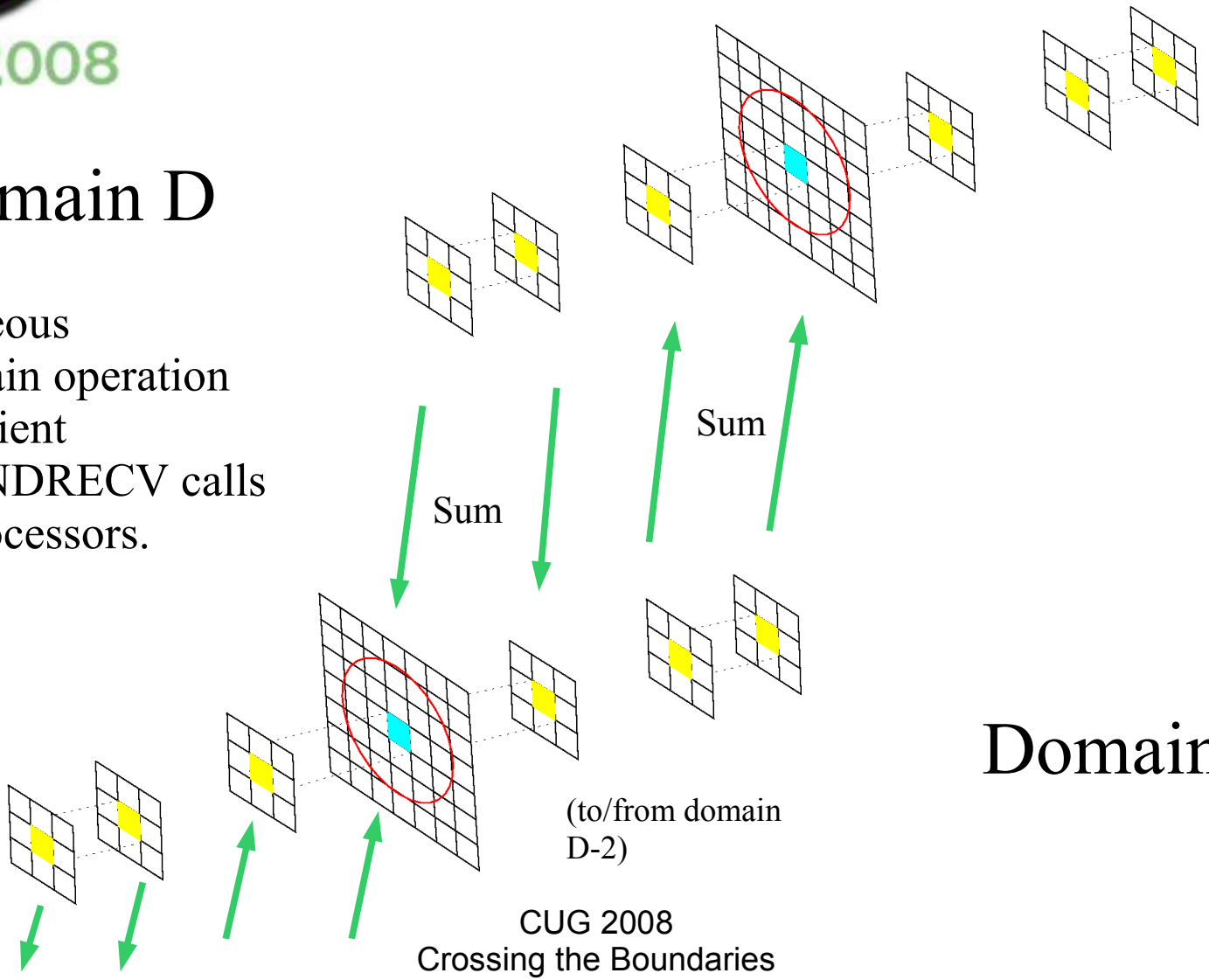
Only for one toroidal sector!



Combining the whole matrix

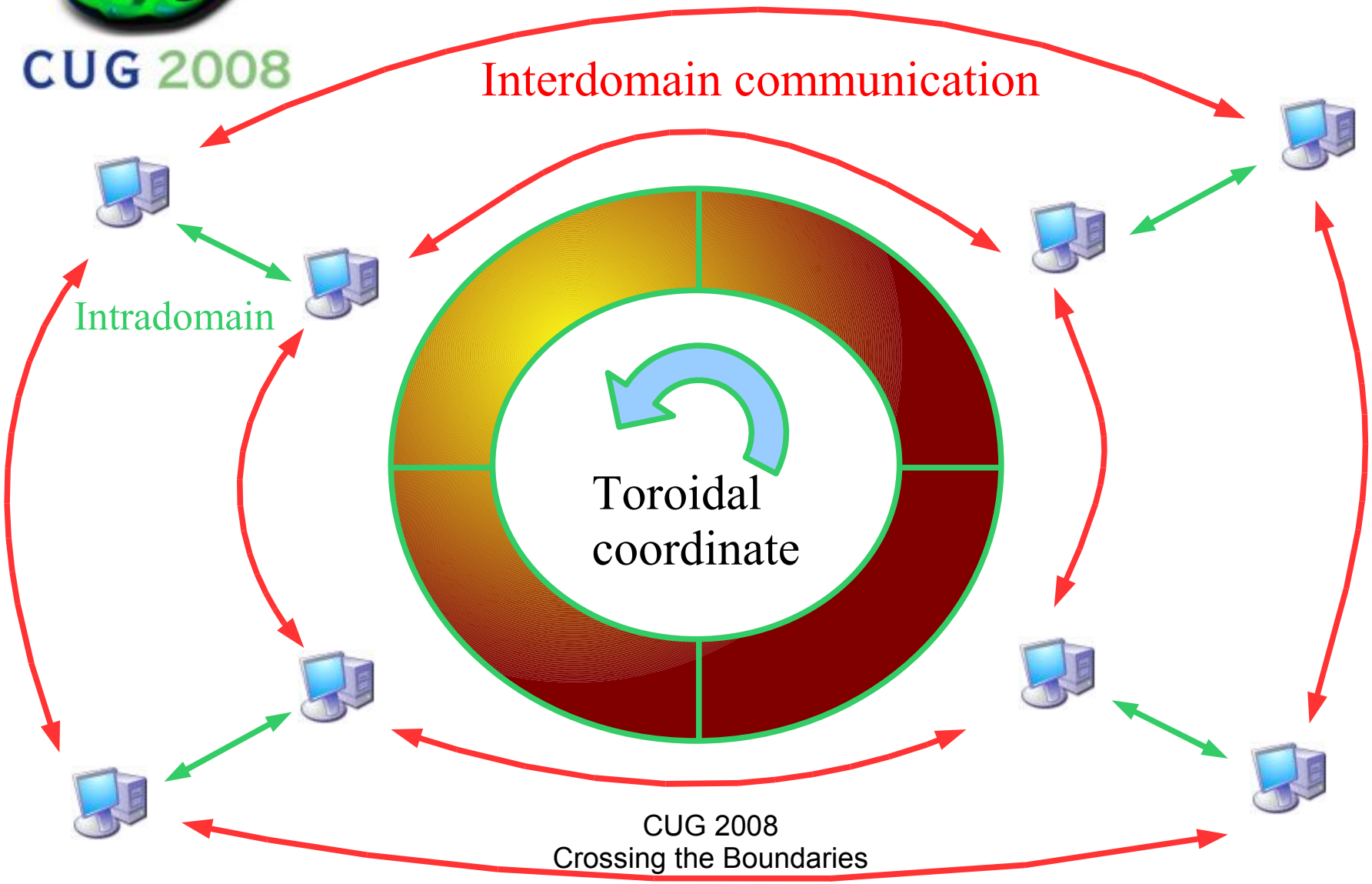
Domain D

Simultaneous interdomain operation with efficient MPI_SENDRECV calls for all processors.



Domain D-1

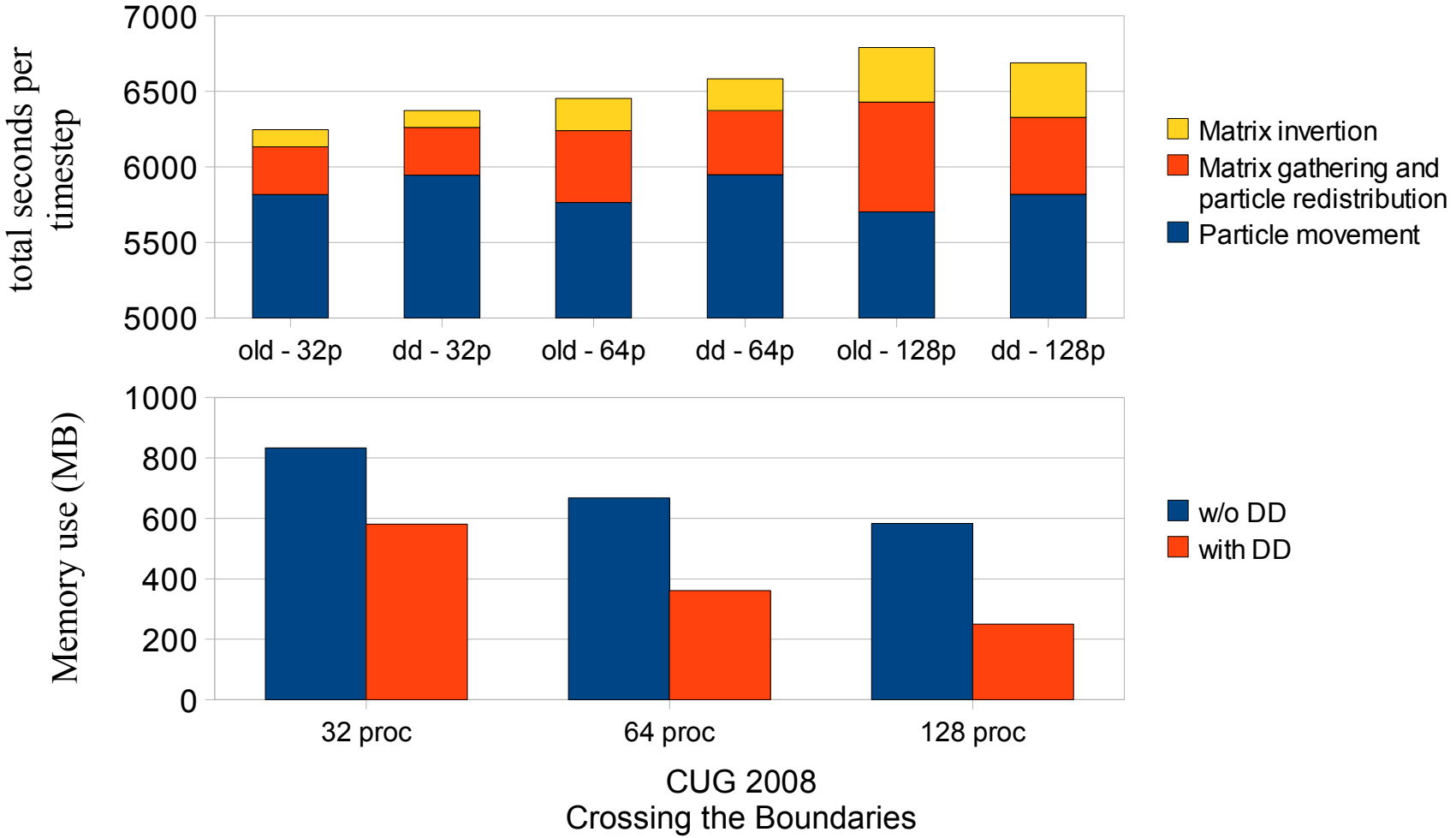
MPI Process topology



Performance results

- Test runs were performed on louhi with a variety of processor number.
- A case was selected with reasonable and favourable parameters
  - High cell number → most memory taken by matrix
  - Fine toroidal division → more domains

Results: computation time and max memory use



Conclusions

- A domain decomposition algorithm has been developed and implemented into ELMFIRE
- Memory consumption has been strongly reduced, extending the code capabilities.
  - Specially true in low node memory systems like new supercomputers (Cray XT4, Blue Gene...)
  - Computation speed is not affected.
- Algorithm is transparent to matrix inversion.

Acknowledgements

- Thanks to all members of the project and their institutions.
  - VTT leading the project
  - TKK participating and hosting me
  - UNED supporting my secondment
- Special thanks to the supporting institutions.
  - Funded by the European Commission
  - Supported by CSC and Finnish Ministry of Education