CINECA
consorzio interuniversitario

Large Scale Parallelism

Carlo Cavazzoni, HPC department, CINECA

Parallel Architectures

Two basic architectural scheme:

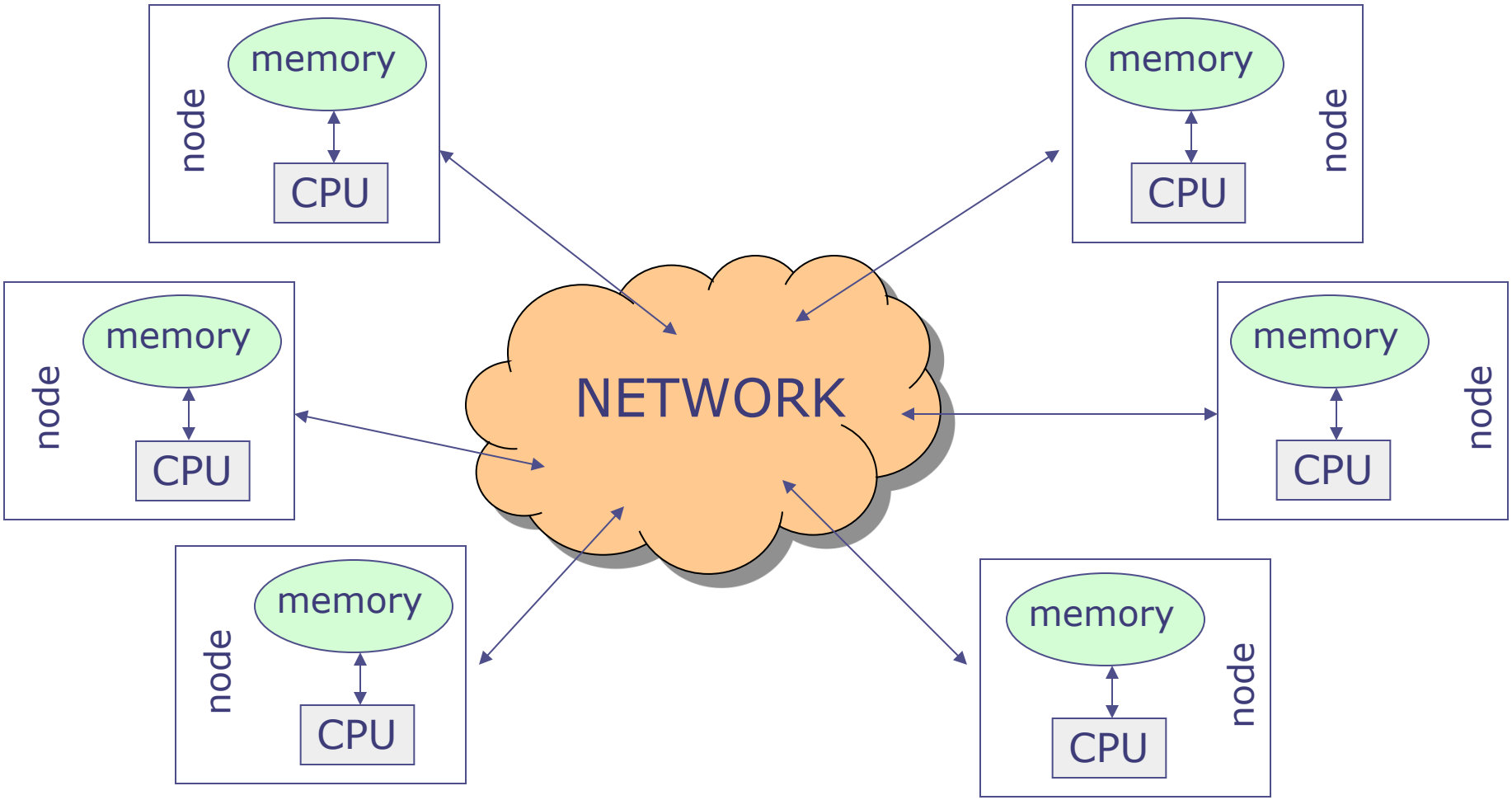
Distributed Memory

Shared Memory

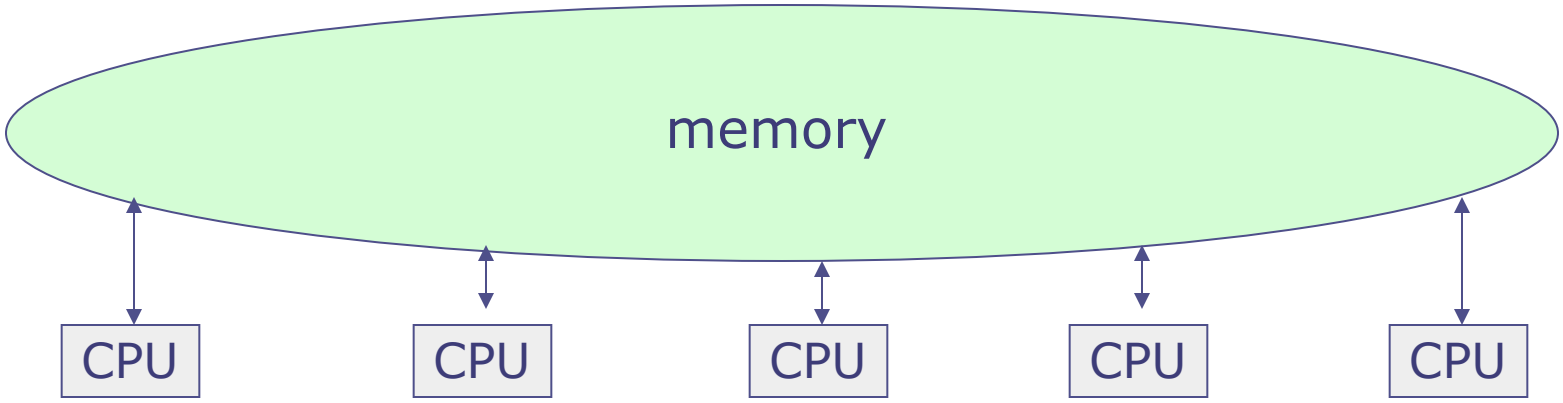
Now most computers have a mixed architecture

+ accelerators -> hybrid architectures

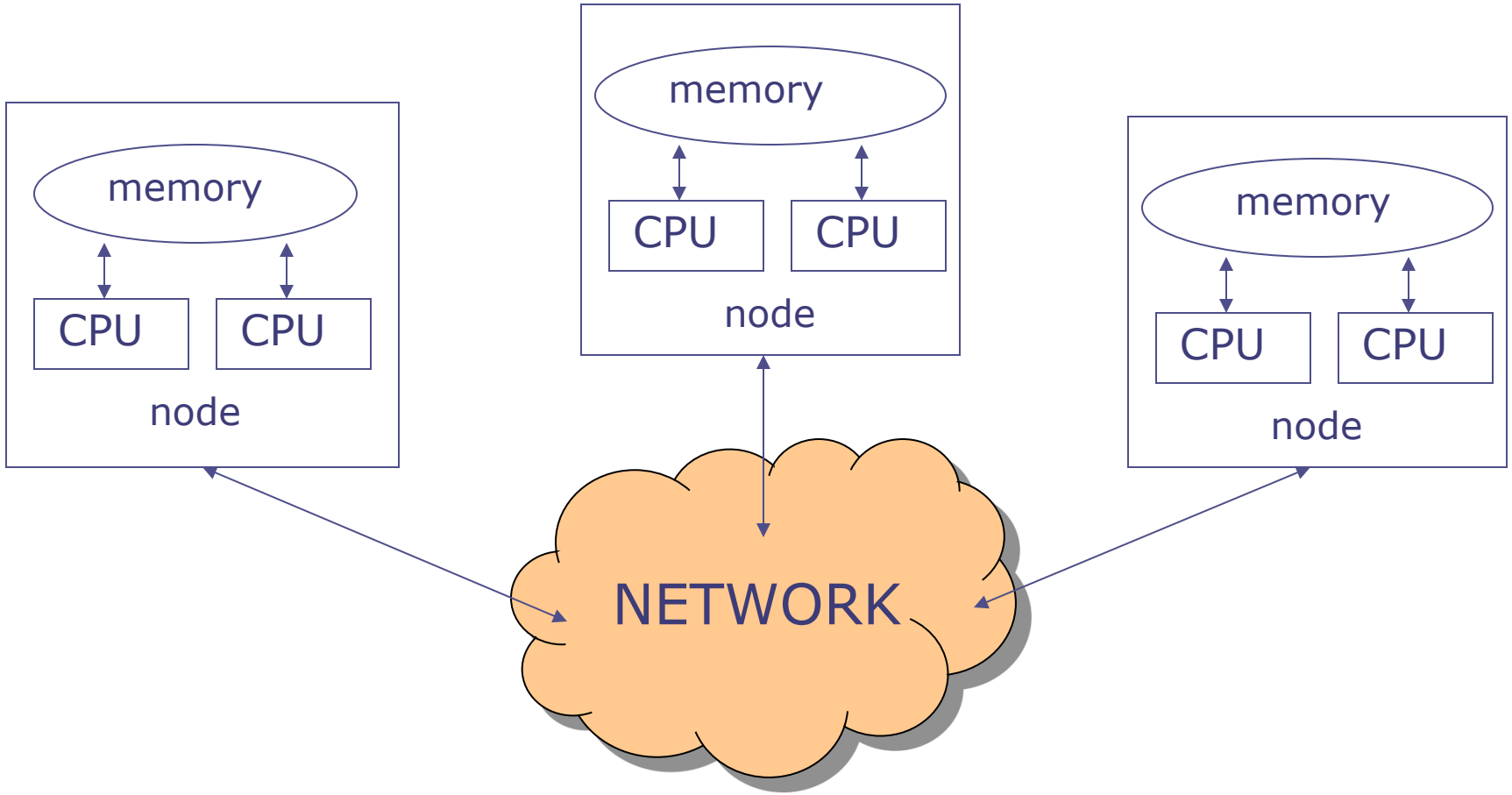
Distributed Memory



Shared Memory

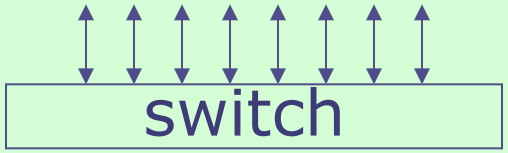


Mixed Architectures

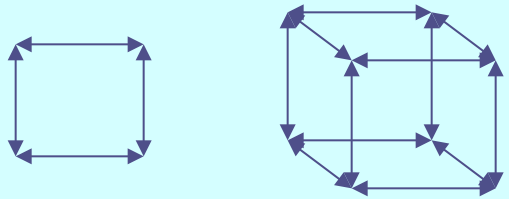


Most Common Networks

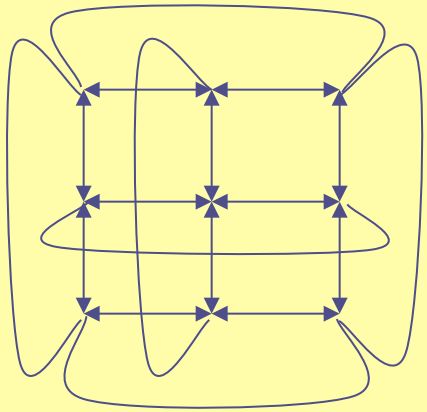
switched



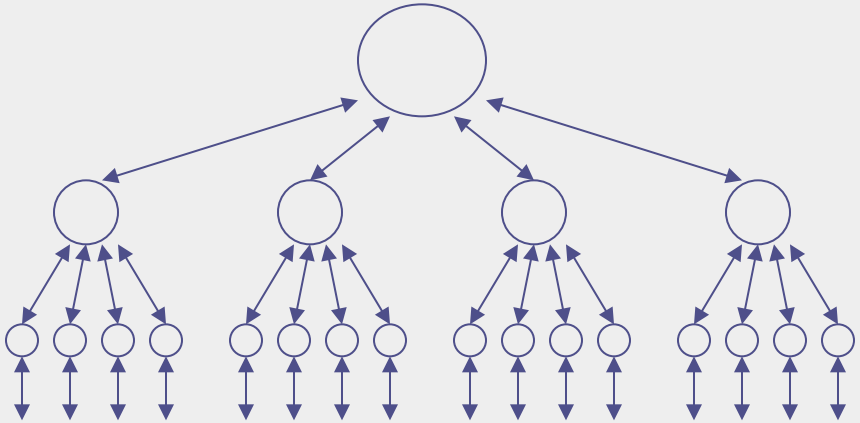
Cube, hypercube, n-cube



Torus in 1,2,...,N Dim



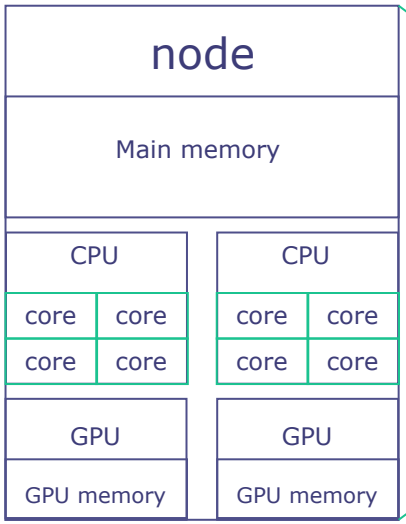
Fat Tree

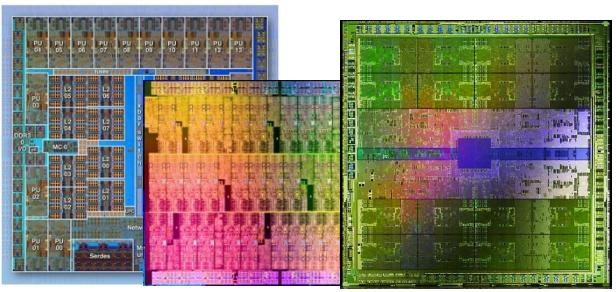


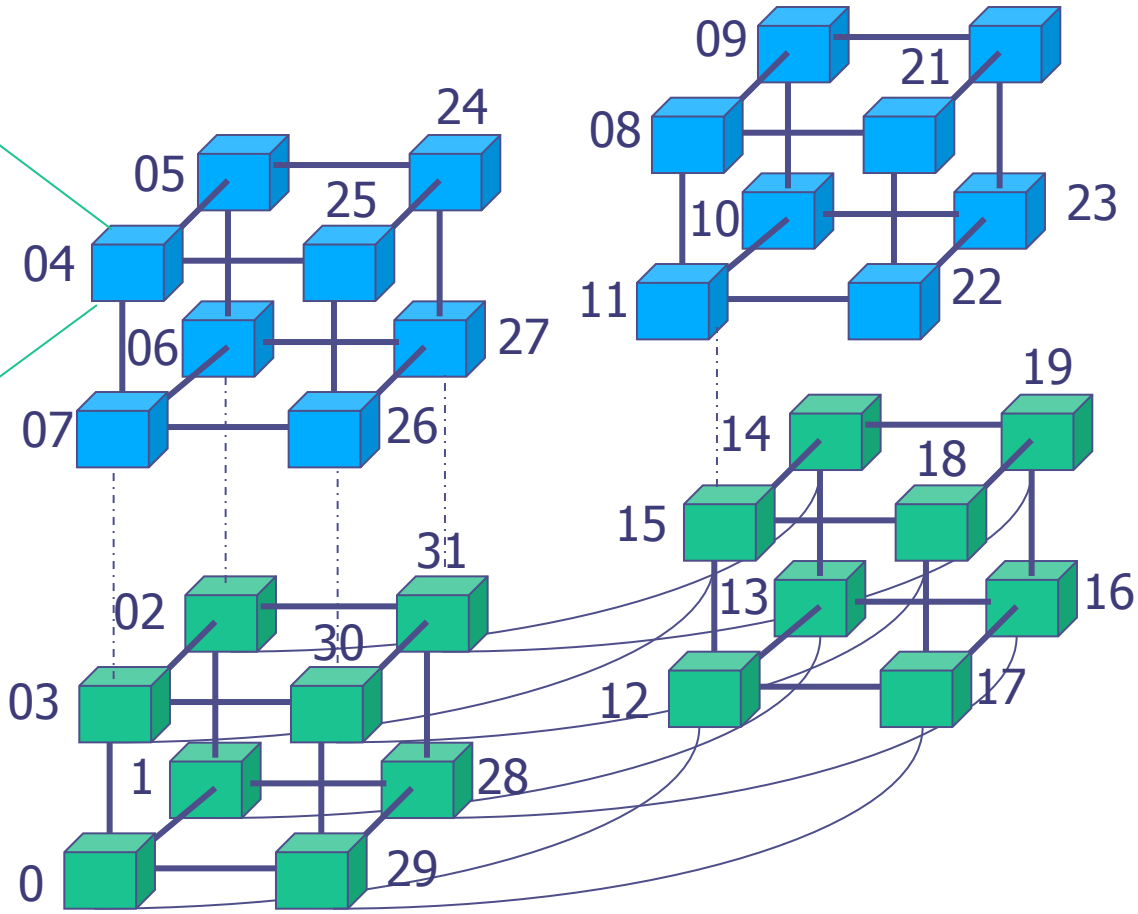
Roadmap to Exascale
(architectural trends)

Systems	2009	2011	2015	2018
System Peak Flops/s	2 Peta	20 Peta	100-200 Peta	1 Exa
System Memory	0.3 PB	1 PB	5 PB	10 PB
Node Performance	125 GF	200 GF	400 GF	1-10 TF
Node Memory BW	25 GB/s	40 GB/s	100 GB/s	200-400 GB/s
Node Concurrency	12	32	O(100)	O(1000)
Interconnect BW	1.5 GB/s	10 GB/s	25 GB/s	50 GB/s
System Size (Nodes)	18,700	100,000	500,000	O(Million)
Total Concurrency	225,000	3 Million	50 Million	O(Billion)
Storage	15 PB	30 PB	150 PB	300 PB
I/O	0.2 TB/s	2 TB/s	10 TB/s	20 TB/s
MTTI	Days	Days	Days	O(1Day)
Power	6 MW	~10 MW	~10 MW	~20 MW

Exascale architecture







CPU ~ 16 cores / 16 threads

Co-processor ~ 128 cores / 512 threads

GPU ~ 1024 cores / 10^4 threads

Dennard scaling law

new VLSI gen.

old VLSI gen.

$L' = L / 2$

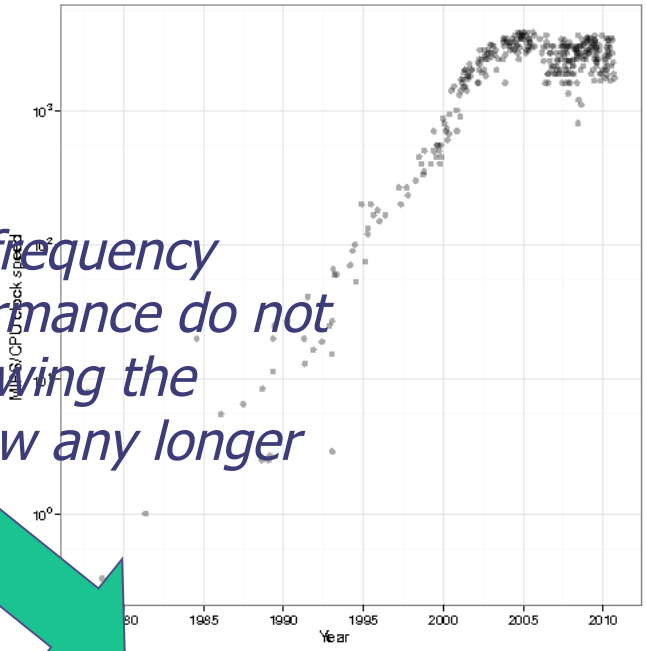
$V' = V / 2$

$F' = F * 2$

$D' = 1 / L^2 = 4D$

$P' = P$

do not hold anymore!



The core frequency and performance do not grow following the Moore's law any longer

$L' = L / 2$

$V' = \sim V$

$F' = \sim F * 2$

$D' = 1 / L^2 = 4 * D$

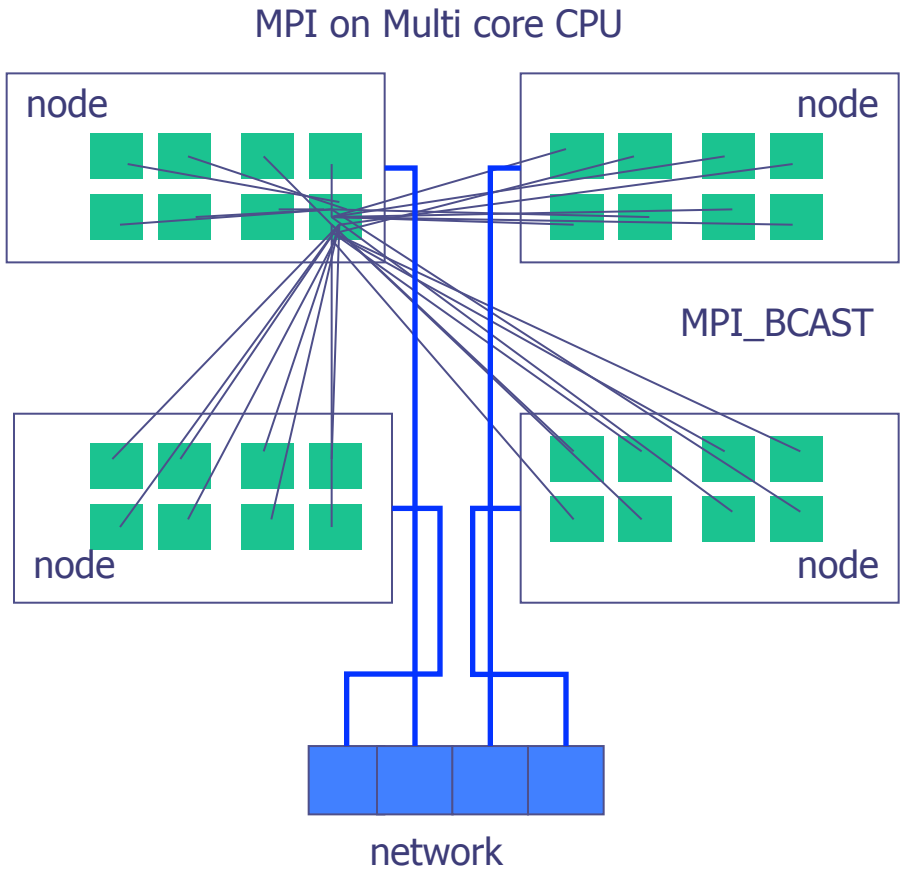
$P' = 4 * P$

The power crisis!

Increase the number of cores to maintain the architectures evolution on the Moore's law

Programming crisis!

MPI inter process communications



1 MPI proces / core
Stress network
Stress OS

Many MPI codes (QE) based on ALLTOALL
Messages = processes * processes

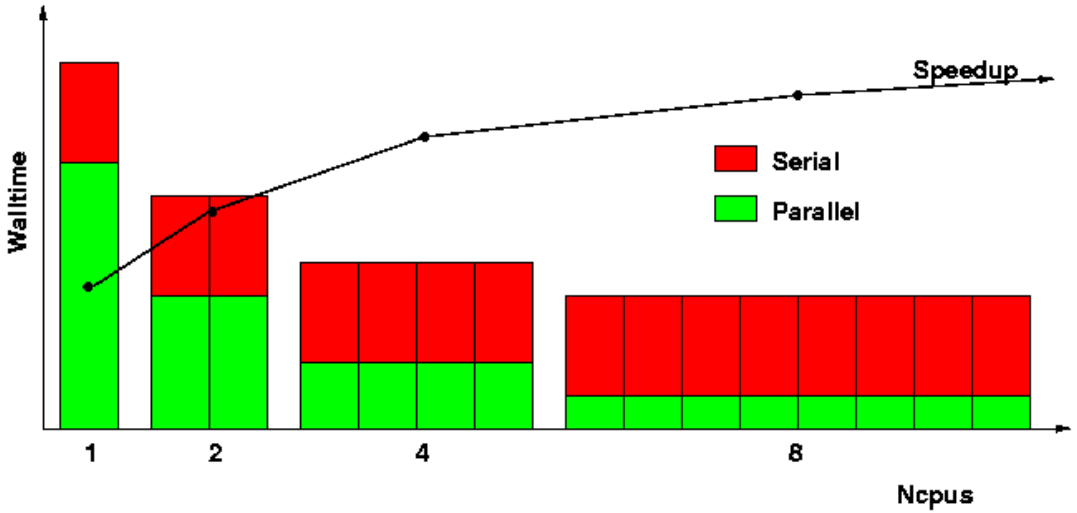
We need to exploit the hierarchy

Re-design applications

Mix message passing And multi-threading

What about Applications?

In a massively parallel context, an upper limit for the scalability of parallel applications is determined by the fraction of the overall execution time spent in non-scalable operations (Amdahl's law).



maximum speedup tends to
$1 / (1 - P)$
P= parallel fraction

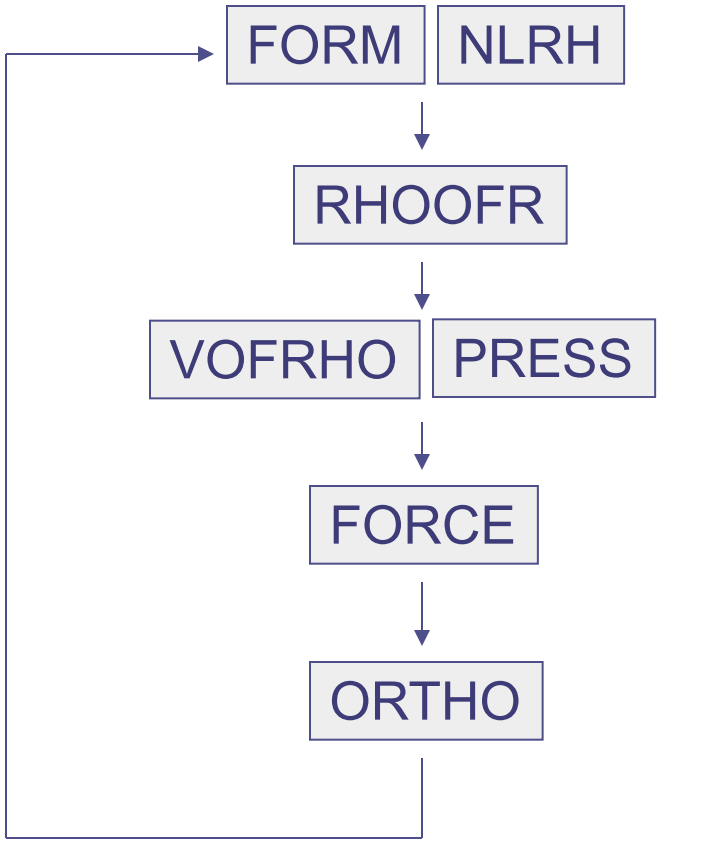
1000000 core

$P = 0.999999$

serial fraction= 0.000001

What about QE?

CP Flow chart



Pseudopotential Form factors

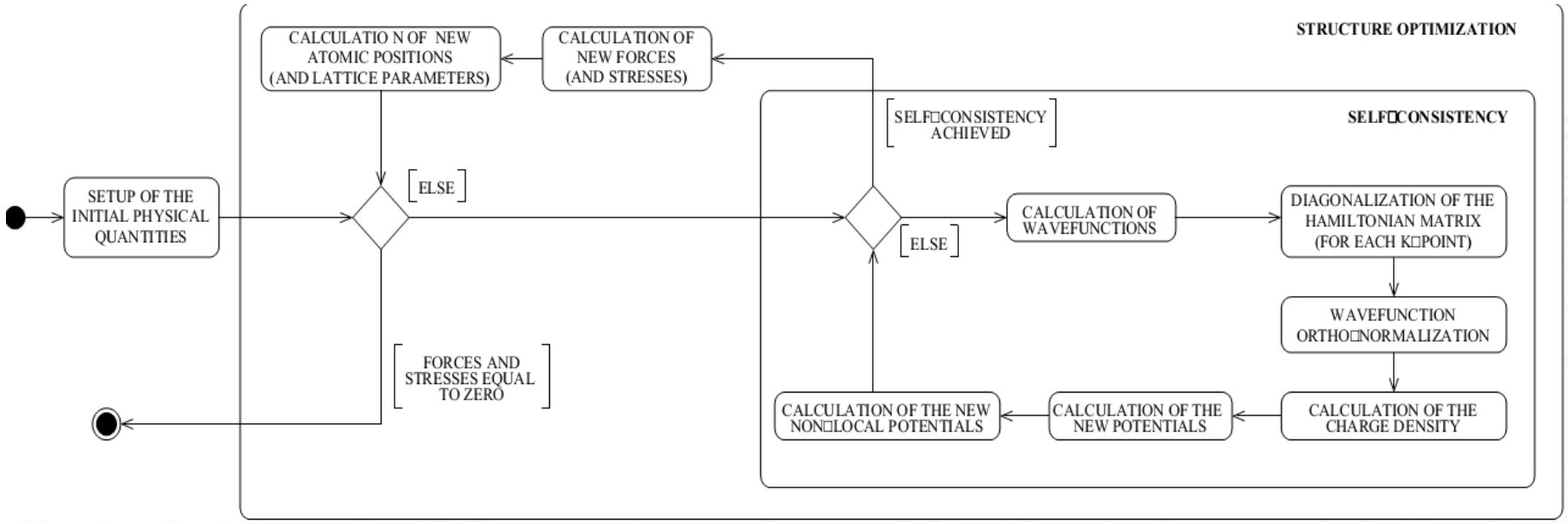
$\rho(r) = \Sigma\ |\psi(r)|^2$

$V(R, \psi) = V^{r}_{DFT}(R, \rho(r)) + V^{G}_{DFT}(R, \rho(G))$

Forces on the electrons: $\mathbf{F}_{\psi}$

Orthogonalize wave functions: ψ

PW flow chart

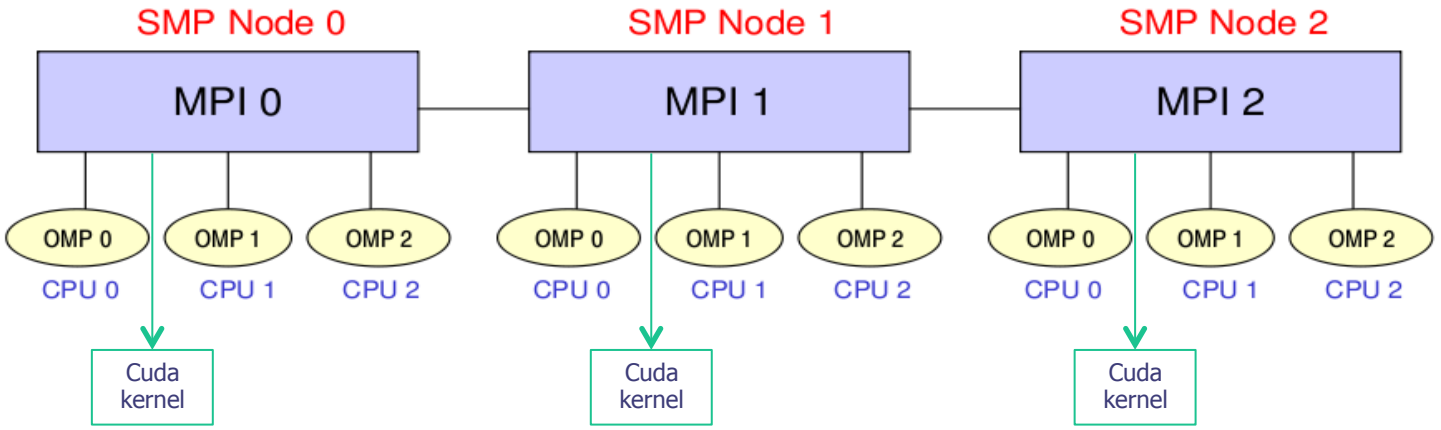


Main Algorithms in QE

- 3D FFT
- Linear Algebra
  - Matrix Matrix Multiplication
  - less Matrix-Vector and Vector-Vector
  - Eigenvalues and Eigenvectors computation
- Space integrals
- Point function evaluations

Programming Models in QE

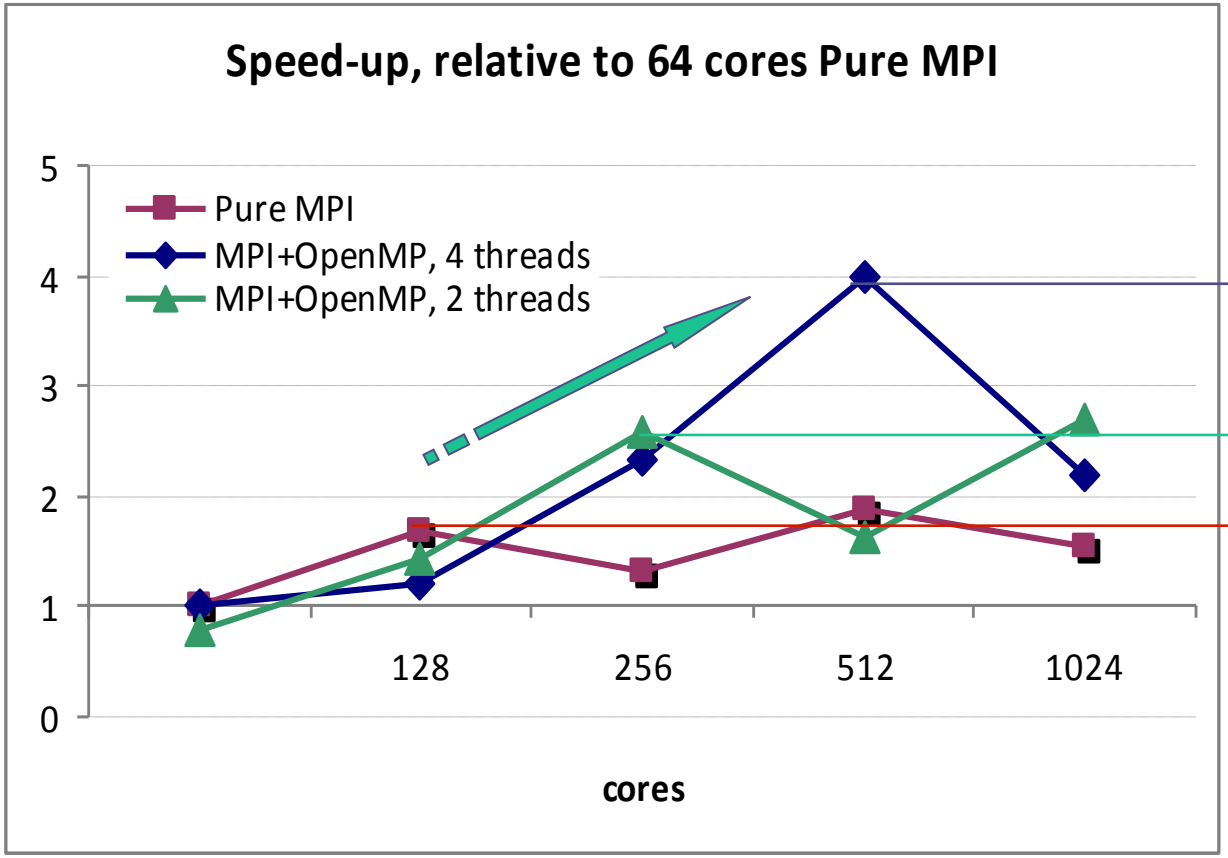
Message Passing (MPI)
Shared Memory (OpenMP)
Languages and Paradigm for Hardware Accelerators (CUDA)
Hybrid: MPI + OpenMP + CUDA



OpenMP

```
!$omp parallel do
do i = 1 , nsl
   call 1DFFT along z ( f [ offset( threadid ) ] )
end do
!$omp end parallel do
call fw_scatter ( . . . )
!$omp parallel
do i = 1 , nzl
!$omp parallel do
  do j = 1 , Nx
    call 1DFFT along y ( f [ offset( threadid ) ] )
  end do
!$omp parallel do
  do j = 1, Ny
    call 1DFFT along x ( f [ offset( threadid ) ] )
  end do
end do
!$omp end parallel
```

Improve scalability with OpenMP



MPI+4OMP Threads saturate at 512 cores

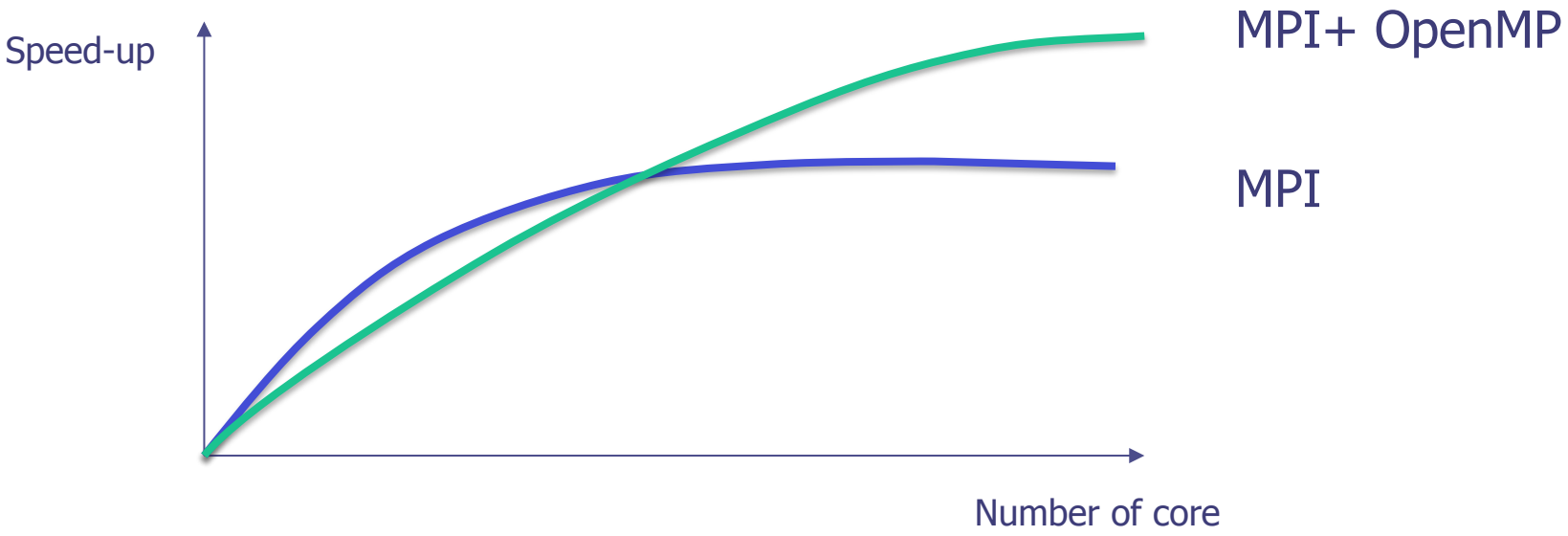
MPI+2OMP Threads saturate at 256 cores

Pure MPI saturate at 128 cores

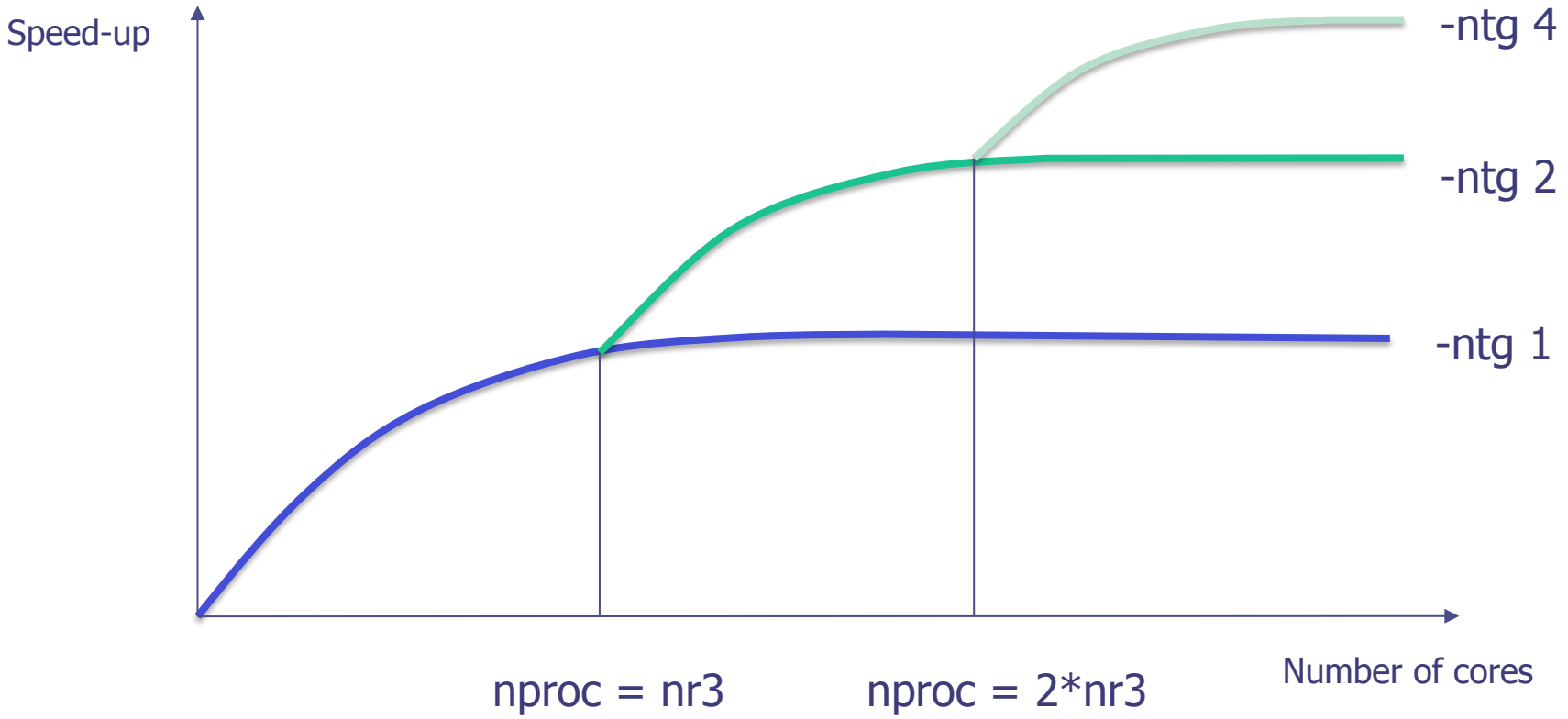
We observe the same behaviour
But at an higher number of cores

CP simulation of 256Water molecules

when I should use OpenMP?



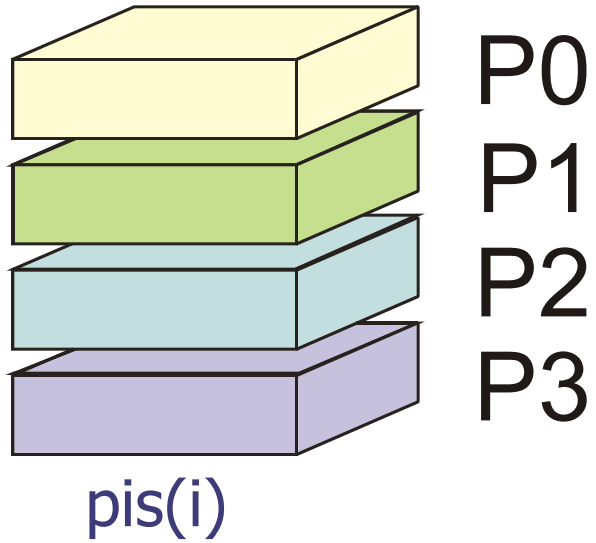
when I should use TaskGroups?



Tasks Group

parallel 3D FFT

```
do i = 1, n
   compute parallel 3D FFT( psi(i) )
end do
```



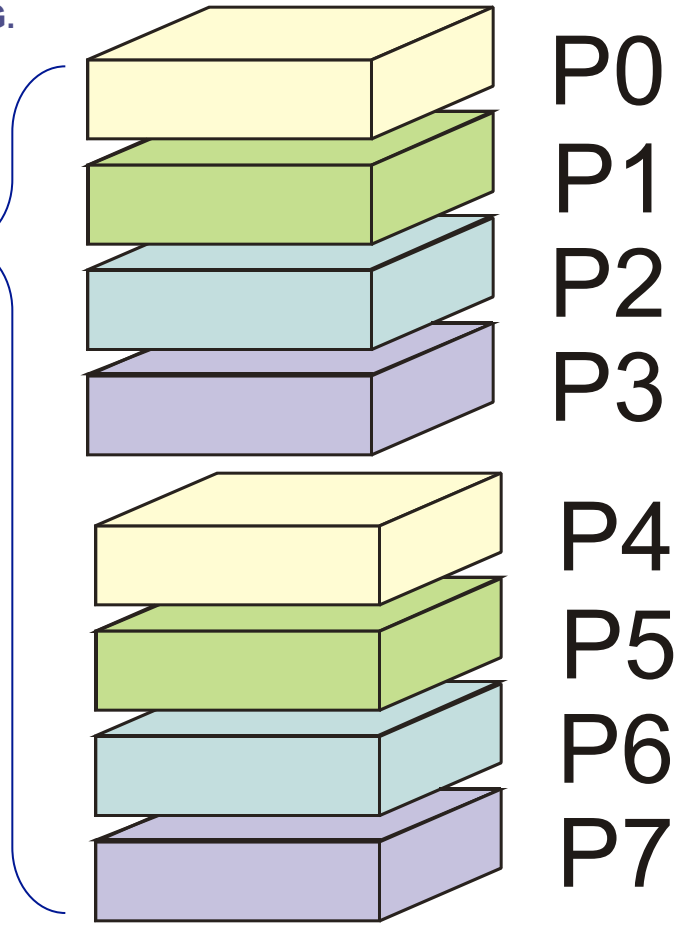
the parallelization is limited to the number of planes in the 3D FFT (NX x NY x NZ)
there is little gain to use more than NZ proc

Tasks Group II

The goal is to use more processors than **NZ**.
The solution is to perform FFT not one by one but in group of **NG.**

```
redistribute the n FFT
do i = 1, nb, ng
  compute ng parallel 3D FFT
  (at the same time)
end do
```

2 - 3D FFT
In one shot

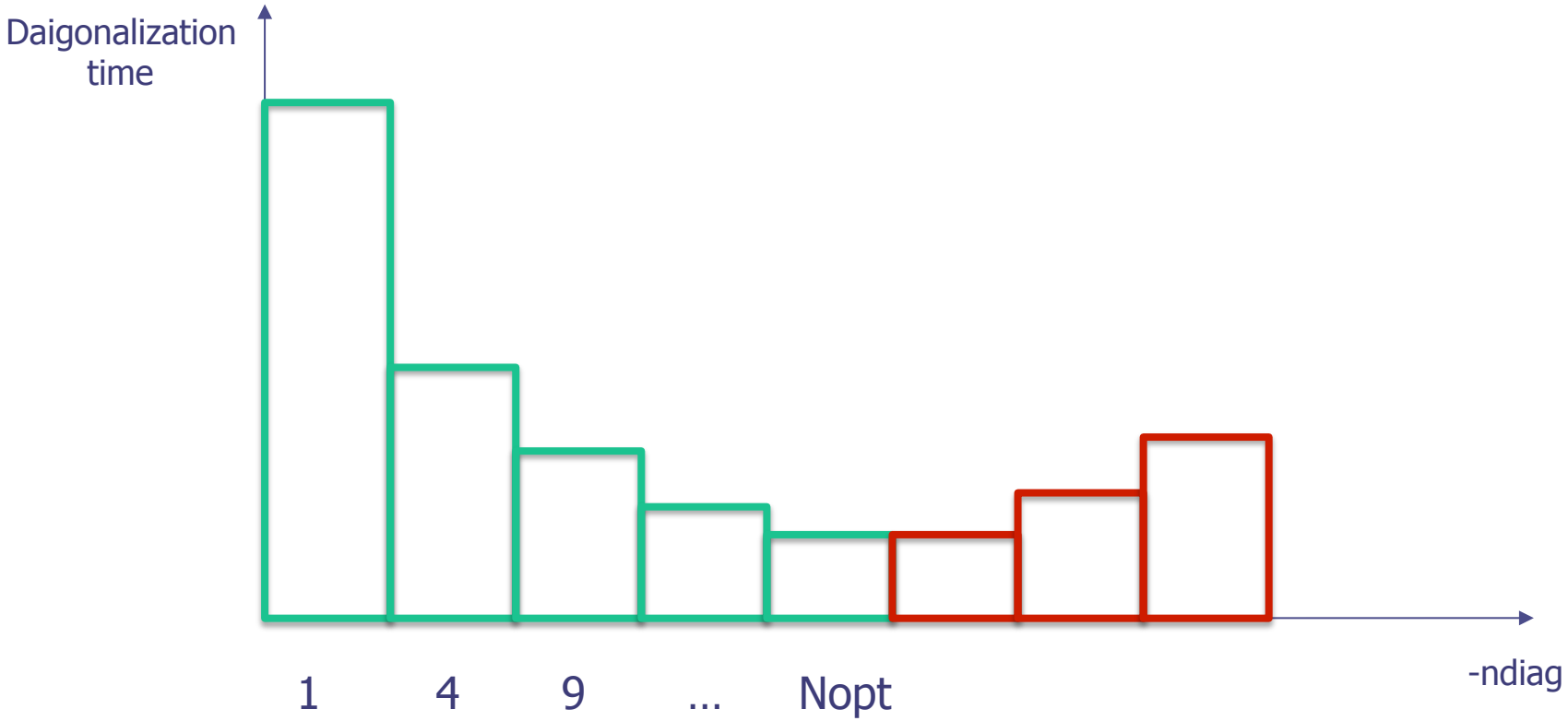


we can scaleup to **NZ x NG processor**.

This cost an additional ALLTOALL
and memory (NG times the size of the 3D vector).

But we have half the number of Loop cycle!

Diagonalization: how to set -ndiag



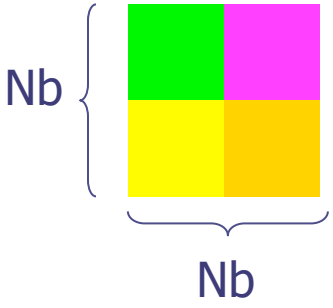
Nopt: depend on the number of electrons and the communication performance

Diagonalization/Orthogonalization Group

when incrasing the number of cores, not all part of the code scale with the same efficiency.

Hermitian matrixes are square matrixes, and a square grid of processors can give to optimal performance (communication/computatio)

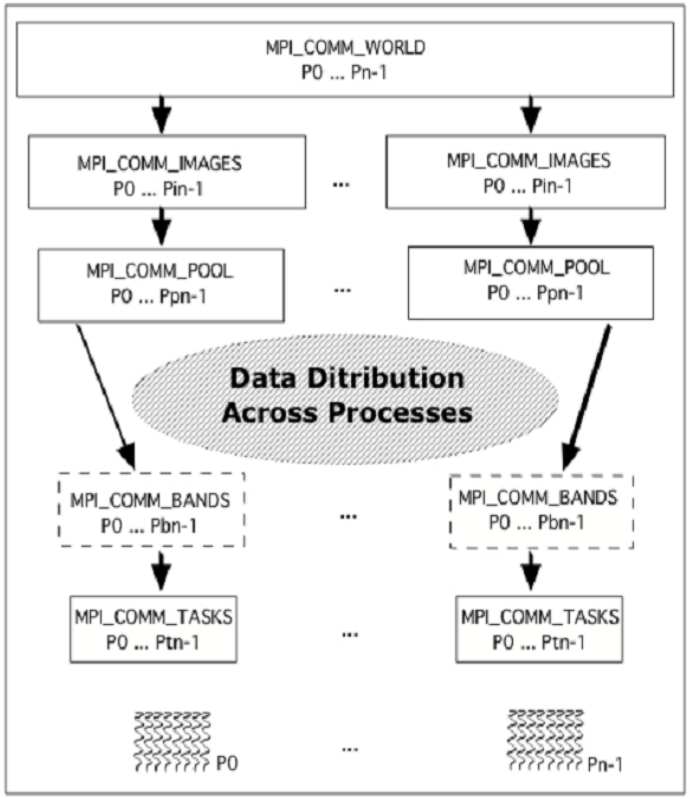
in a run with 10 processors,
the diag. group use 4 procrs (2x2)



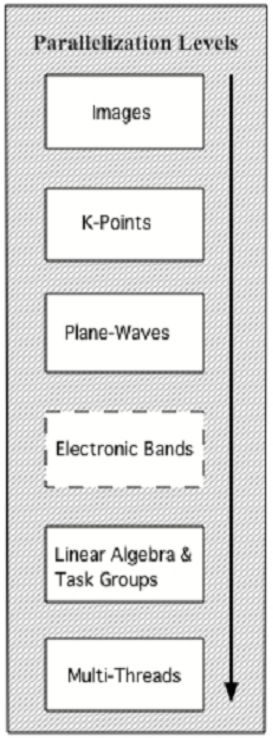
Matrixes are block distributed to the diag group.

In this case is possible to use a mixed parallelization MPI+OpenMP using SMP library

QE parallelization hierarchy



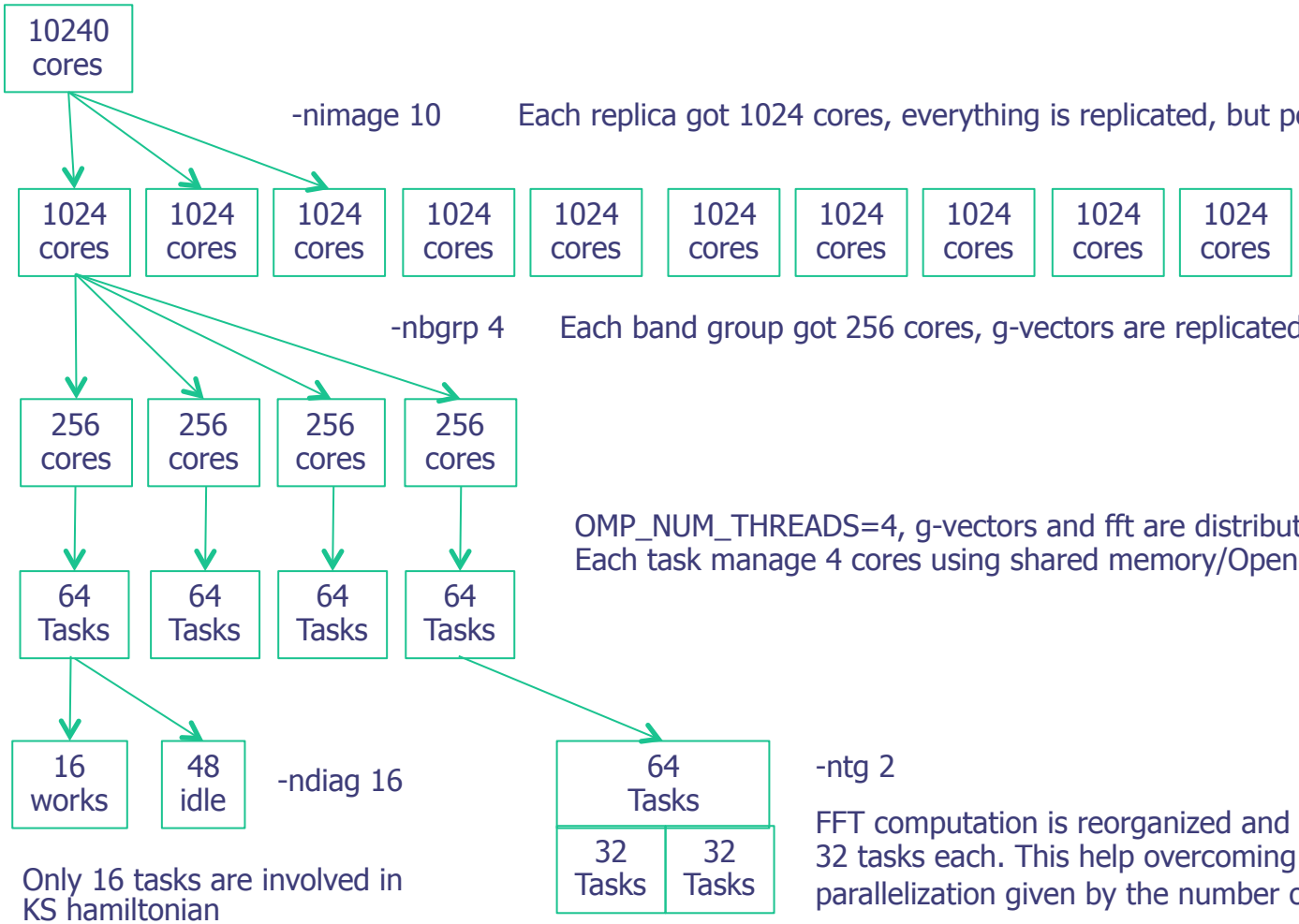
MPI Communicators Hierarchy



Parallelization Strategy

- 3D FFT — ad hoc MPI & OpenMP driver
- Linear Algebra — ScalaPACK + blas multithread
- Space integrals — MPI & OpenMP loops parallelization and reduction
- Point function evaluations — MPI & OpenMP loops parallelization

How to deal with extreme parallelism

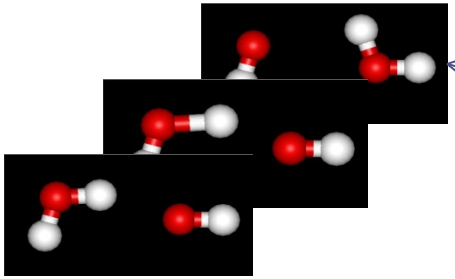


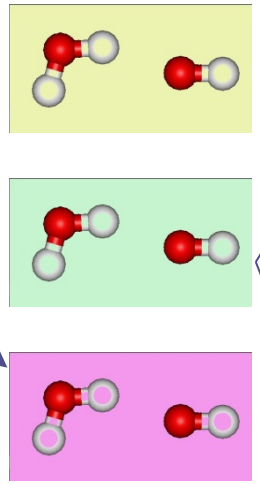
G-vectors are replicated
(across groups)

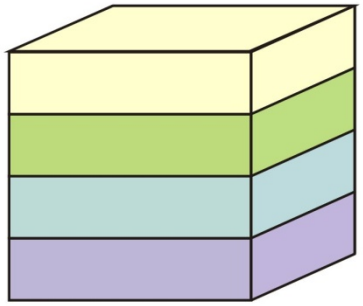
G-vectors are distributed
(within each group)

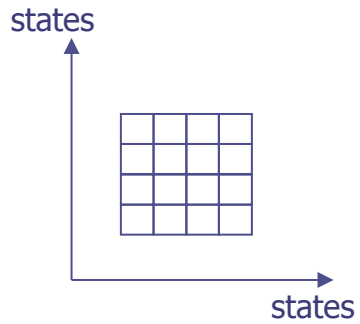
1x1xN proc grid

√N x √N proc grid

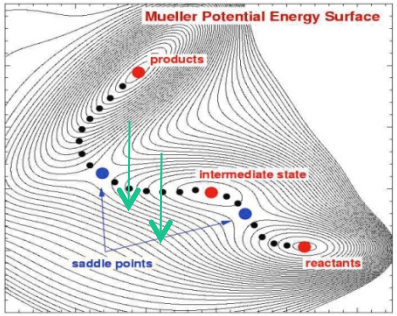








Parallelization over Images
Energy barrier evaluation



Parallelization over Pools:
K-points sempling of
Brillouin Zone.
Electronic band structure

Charge density

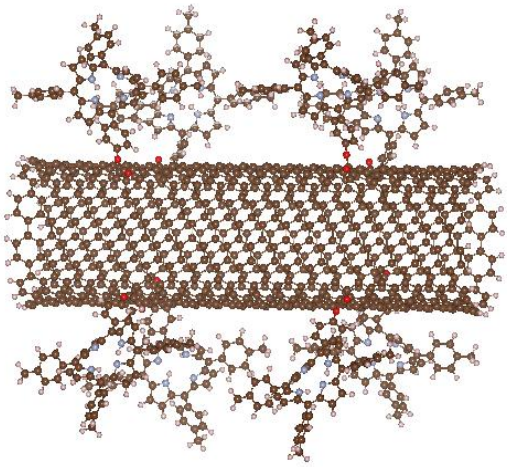
Hamiltonian

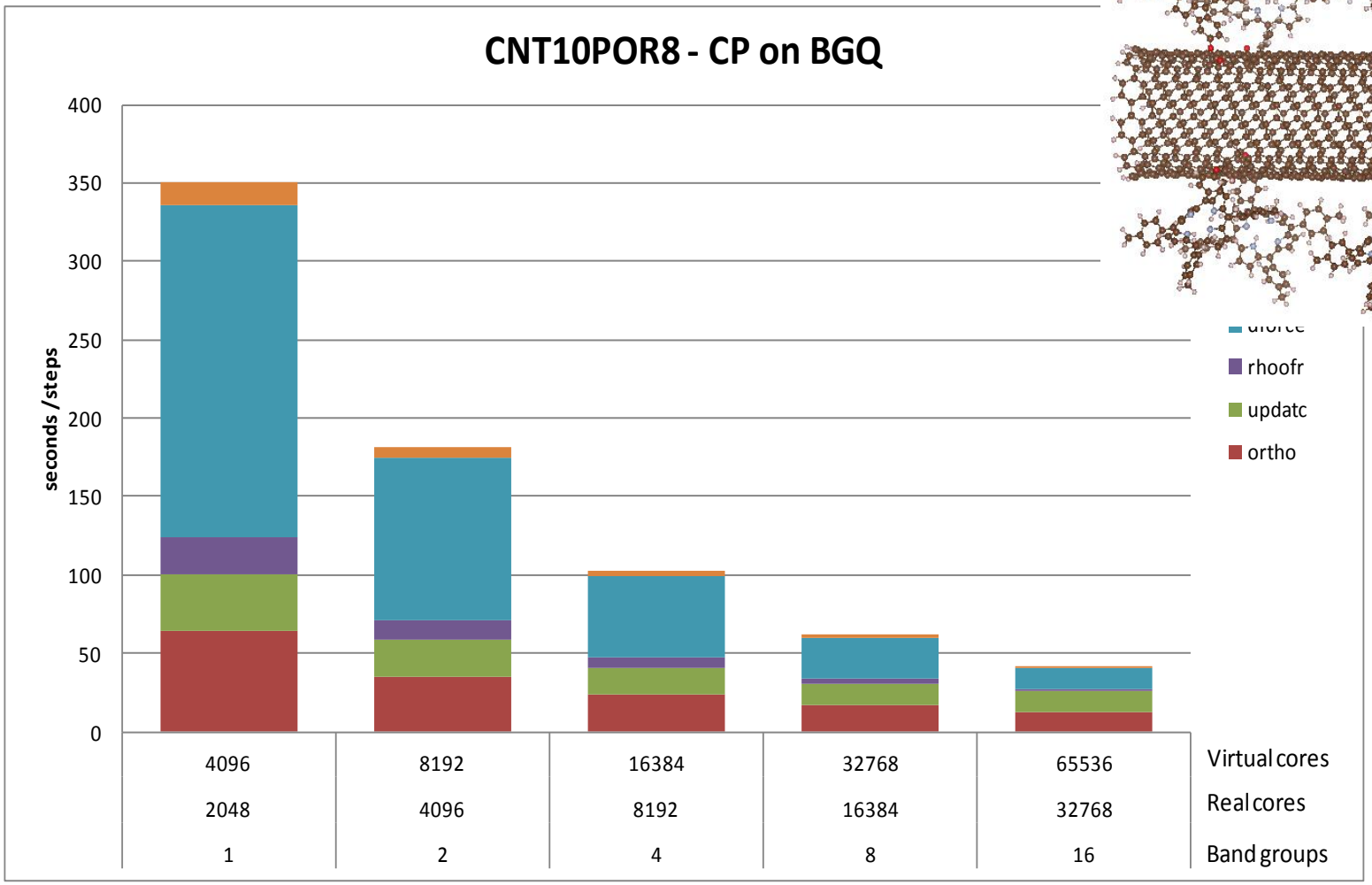
Parallelization of the FFT
Real and Reciprocal Space
Decomposition.
One FFT for each electron

Parallelization of
Bands structure.
Matrix Diagonalization
At least one state for
each electron

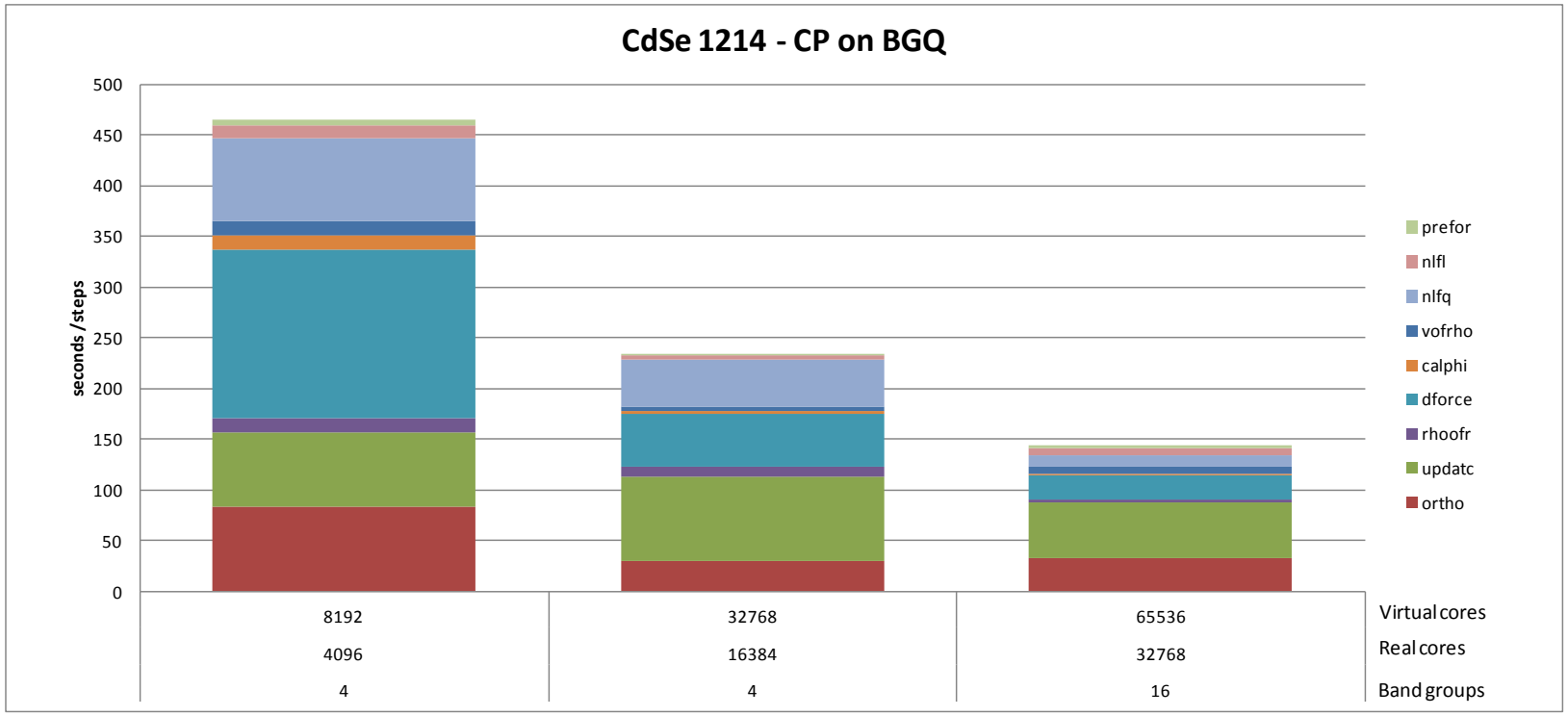
Loosely coupled ⟷ **Tightly coupled**

Bands parallelization scaling





CdSe 1214 - FERMI							
virtual core	real core	MPI task	OpenMP T	Band Grou	Task Grou	Ortho procs	Time/step
8192	4096	1024	8	4	4	256	472
32768	16384	4096	8	4	4	1024	241
65536	32768	8192	8	16	4	512	148.835



Typical CP command line on massively parallel supercomputers

```
export WORKDIR=`pwd`/.
export TASKS=16384
export PREFIX=cp
export TASK_PER_NODE=8
export THREADS=4
export NTG=4
export NDIAG=512
export NBG=16
export RUNID=4
export INPUT_FILE=$WORKDIR/$PREFIX.in
export OUTPUT_FILE=$WORKDIR/$PREFIX.${TASKS}t.${TASK_PER_NODE}tpn.${THREADS}omp.${NBG}bg.${NTG}tg.${NDIAG}d.${RUNID}.out

runjob --np $TASKS --ranks-per-node $TASK_PER_NODE --envs OMP_NUM_THREADS=$THREADS : \
            $WORKDIR/cp.x -nbgrp $NBG -ntask_groups $NTG -ndiag $NDIAG < $INPUT_FILE > $OUTPUT_FILE
```

Input parameters

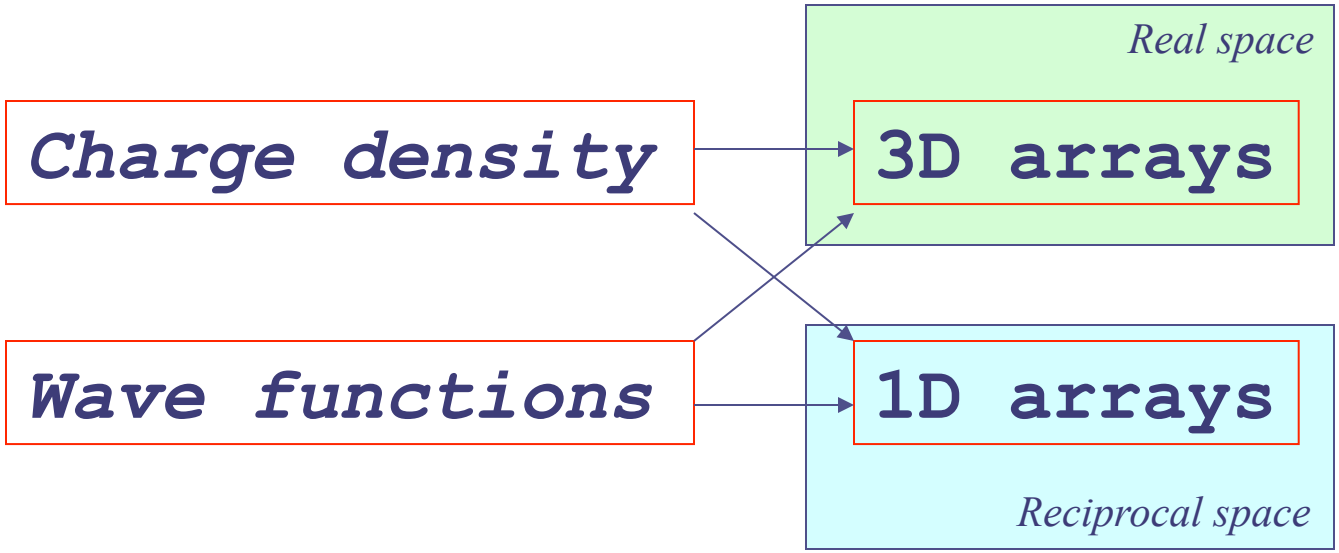
```
&control
    title='Prace bench',
    calculation = 'cp',
    restart_mode='restart',
    ndr=53,
    ndw=52,
    prefix='nano',
    nstep=500,
    iprint=10,
    isave=200,
    dt=5.0d0,
    etot_conv_thr = 1.d-8,
    pseudo_dir = './'
    outdir = './'
    tstress = .false.
    tprnfor = .true.
    wf_collect=.false.
    saverho=.false.
    memory="small"

/
```

can be critical for performance,
use only when really needed

Reading the output… CNT10POR8

```
Program CP v.5.0.1 (svn rev. 9250M) starts on  7Aug2012 at 23: 8:40

     This program is part of the open-source Quantum ESPRESSO suite
     for quantum simulation of materials; please cite
         "P. Giannozzi et al., J. Phys.:Condens. Matter 21 395502 (2009);
          URL http://www.quantum-espresso.org",
     in publications or presentations arising from this work. More details at
     http://www.quantum-espresso.org/quote.php

     Parallel version (MPI & OpenMP), running on  131072 processor cores
     Number of MPI processes:               16384
     Threads/MPI process:                       8
     band groups division:  nbgrp     =      16
     R & G space division:  proc/pool =   16384
     wavefunctions fft division:  fft/group =       4

…

   Matrix Multiplication Performances
   ortho mmul, time for parallel driver      =   0.02369 with 1024 procs

   Constraints matrixes will be distributed block like on
   ortho sub-group =   32*  32 procs
```

Basic Data Type



Reciprocal Space Representation

Wave Functions

$$\psi_i(r) = \frac{1}{\sqrt{\Omega}} \sum_G C_i(G) \exp(iGr)$$

$$|G|^2 / 2 \le E_{cut}$$

To truncate the infinite sum

Charge Density

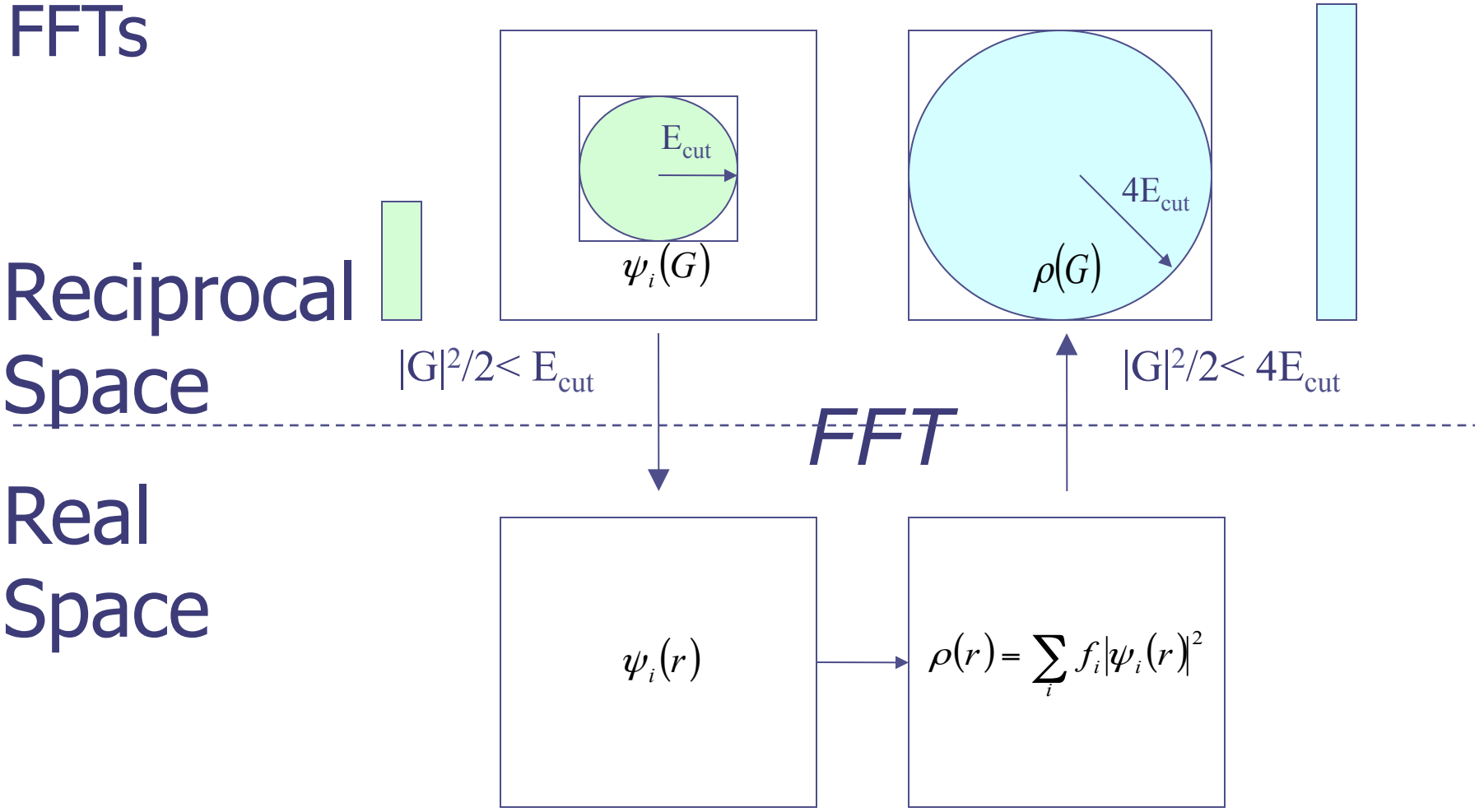
$$\rho(r) = \sum_i f_i |\psi_i(r)|^2$$

$$\rho(G) = \frac{1}{\Omega} \sum_i f_i \sum_{G'} C_i(G') C_i(G - G') \exp(i(G - G')r)$$

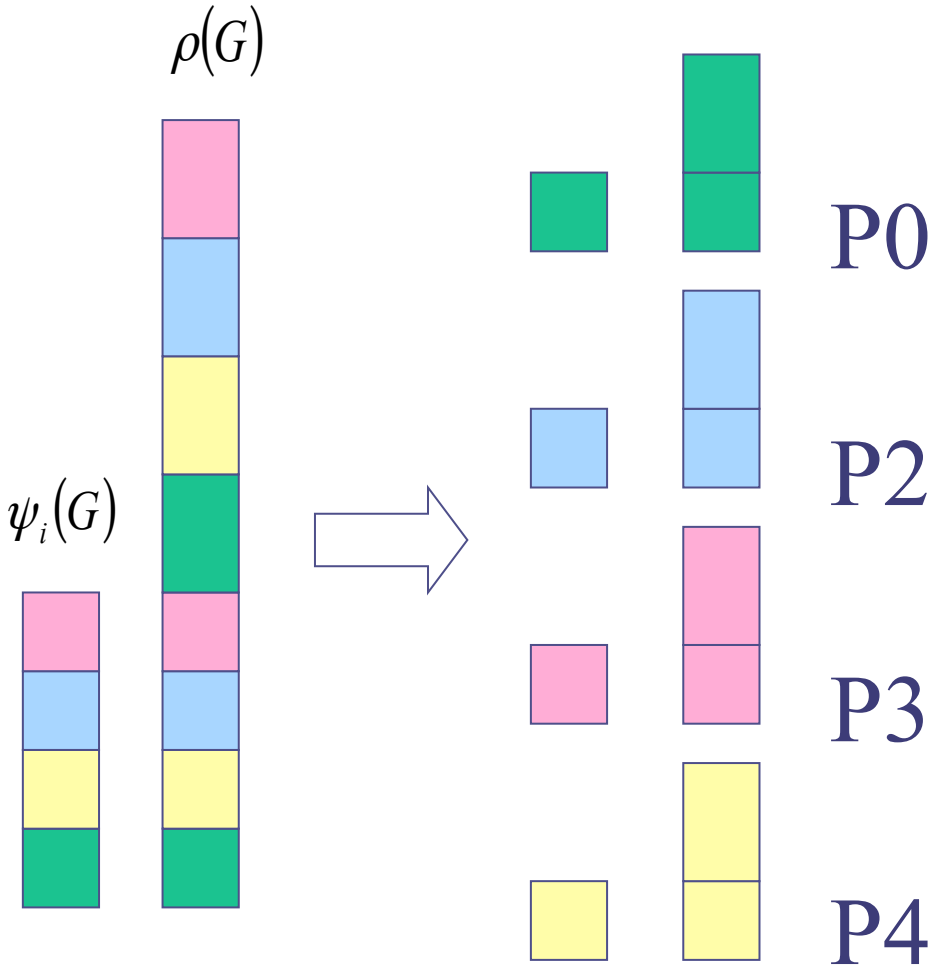
$$|G|^2 / 2 \le 4E_{cut}$$

To retain the same accurancy as the wave function

FFTs

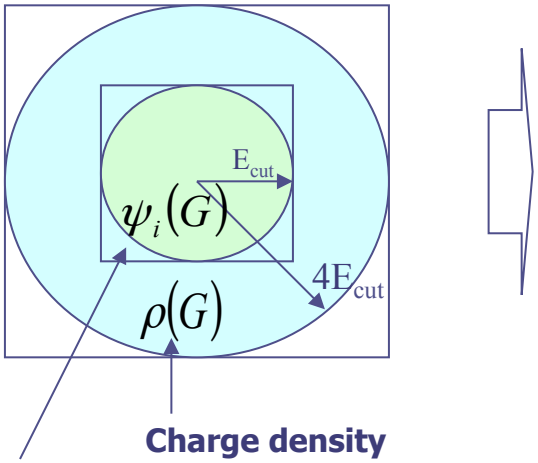

Reciprocal Space
Real Space
E_{cut}
$\psi_i(G)$
$4E_{cut}$
$\rho(G)$
$|G|^2/2 < E_{cut}$
$|G|^2/2 < 4E_{cut}$
FFT
$\psi_i(r)$
$\rho(r) = \sum_i f_i |\psi_i(r)|^2$

Reciprocal Space distribution



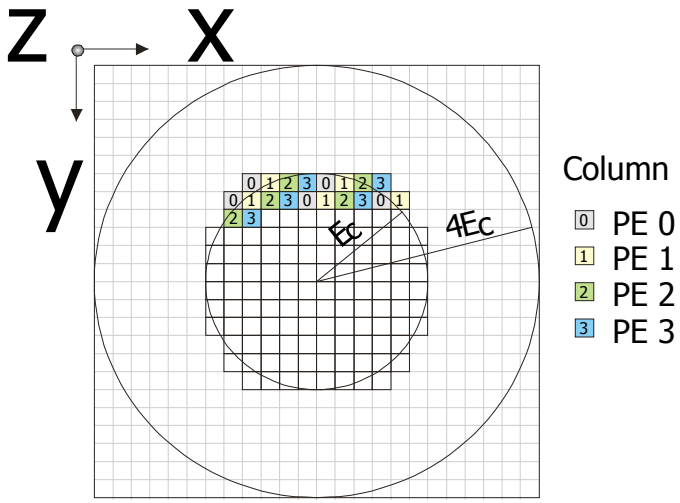
Understanding QE 3DFFT, Parallelization of Plane Wave

Reciprocal Space
G ⇔Plane Wave vectors

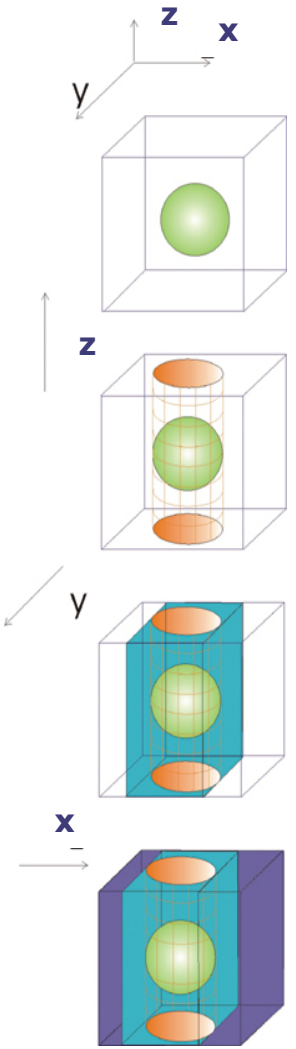


Charge density

Single state electronic wave function



~ Nx Ny / 5 FFT along z



Similar 3DFFT are present in most ab-initio codes like CPMD

Conclusion

Number of cores double every two years -> parallel vs serial

Memory per core decreases -> parallelism at all level

Multi/many core nodes -> MPI and OpenMP

Communicator hierarchy -> tune command line parameters

I/O will be critical -> avoid it when not required

Power consumption will drive CPU/Computer design