How to run VASP

Part 2: Running big

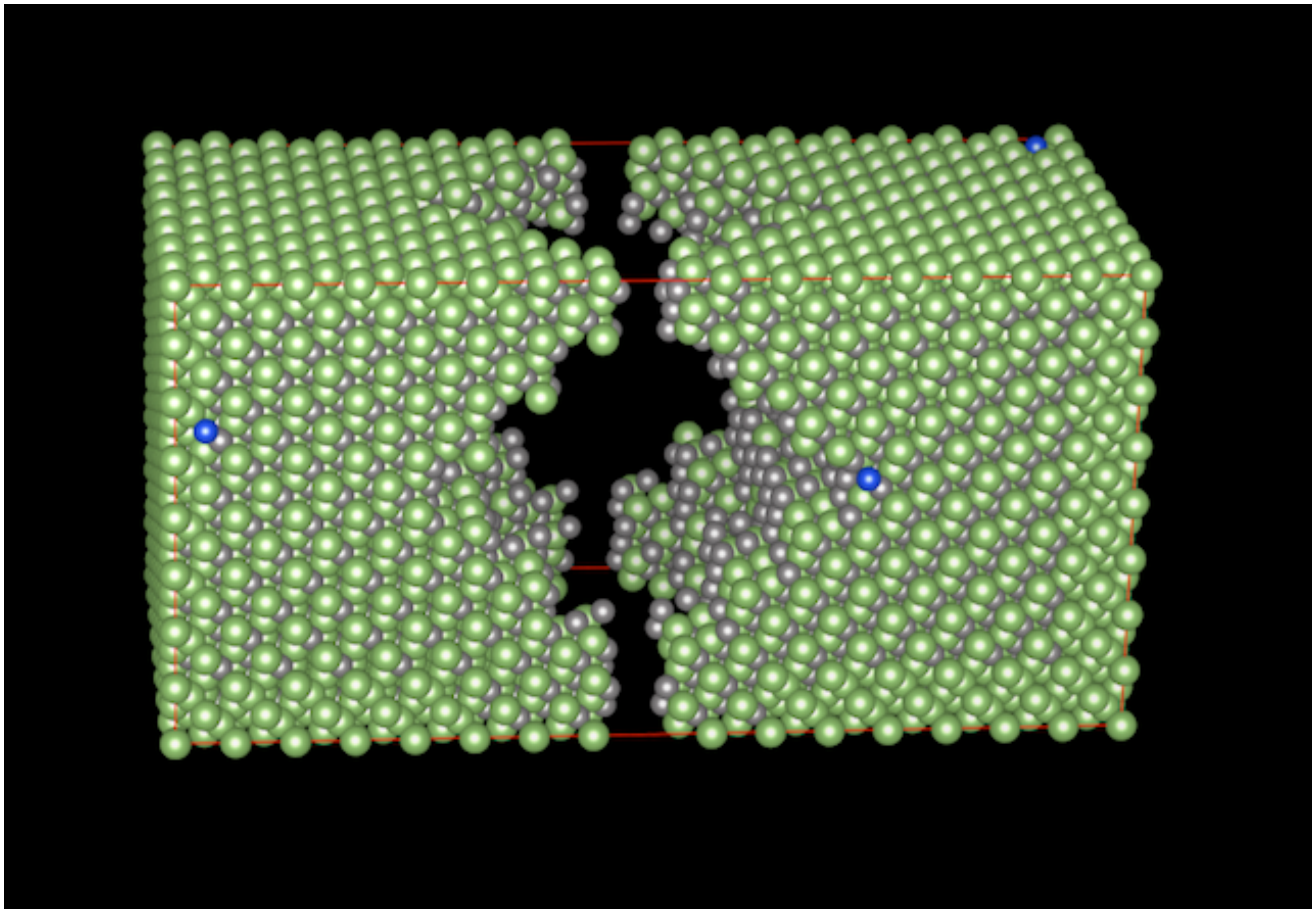
Quick summary

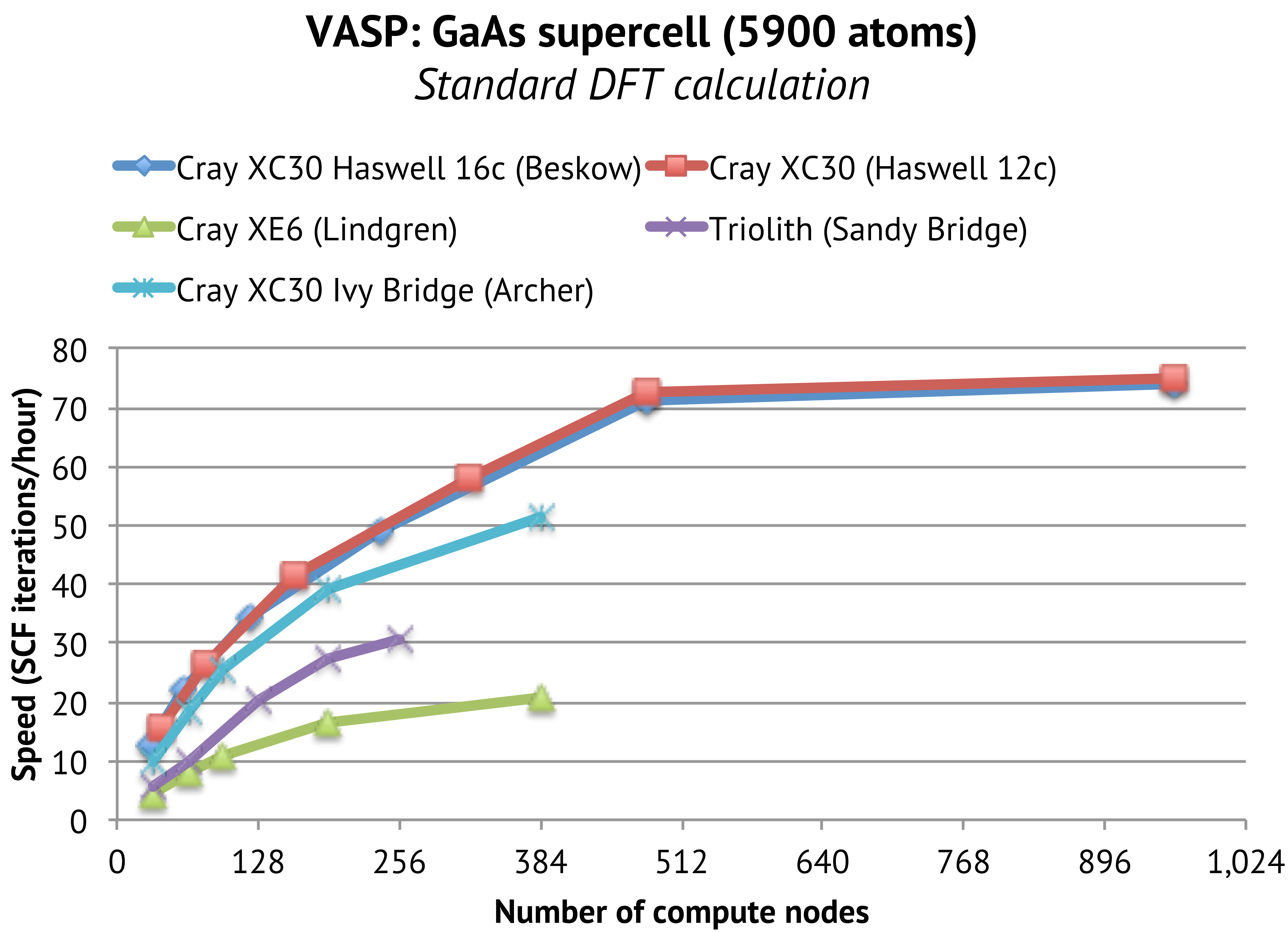
- **Everything from previous slides apply!**
- Less MPI ranks / compute node
- SCALAPACK is a must, ELPA-version of VASP even better.
- Choice of MPI-library can be influential.
- LPLANE = .TRUE. (default). Check NGZ vs cores & NPAR
- Minimize file input/output (disable STOPCAR/WAVECAR etc)

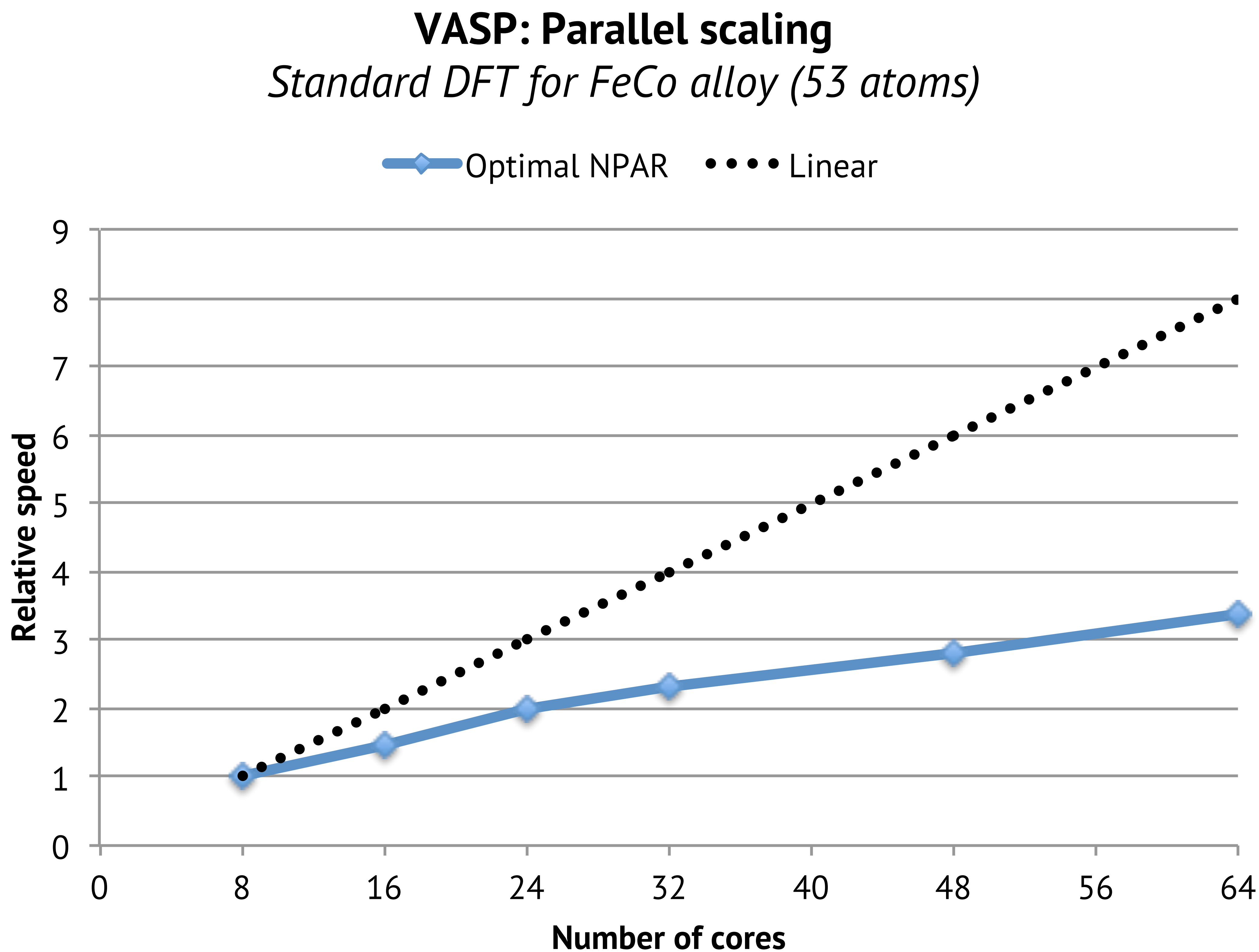
Big computer

=>

Big simulation!




VASP: GaAs supercell (5900 atoms)
Standard DFT calculation
Cray XC30 Haswell 16c (Beskow)
Cray XC30 (Haswell 12c)
Cray XE6 (Lindgren)
Triolith (Sandy Bridge)
Cray XC30 Ivy Bridge (Archer)
Speed (SCF iterations/hour)
0
10
20
30
40
50
60
70
80
0
128
256
384
512
640
768
896
1,024
Number of compute nodes


VASP: Parallel scaling
Standard DFT for FeCo alloy (53 atoms)
Optimal NPAR
Linear
Relative speed
0
1
2
3
4
5
6
7
8
9
0
8
16
24
32
40
48
56
64
Number of cores

Cores/node: VASP and memory bandwidth

- VASP is limited by memory bandwidth
- Can't utilize all cores on a multi-core machine!

CPU

A breakdown of how the 76.8% total CPU time was spent:

Scalar numeric ops	24.2%
Vector numeric ops	15.8%
Memory accesses	60.0%
Other	0.0%

The per-core performance is memory-bound. Use a profiler to identify time-consuming loops and check their cache performance.

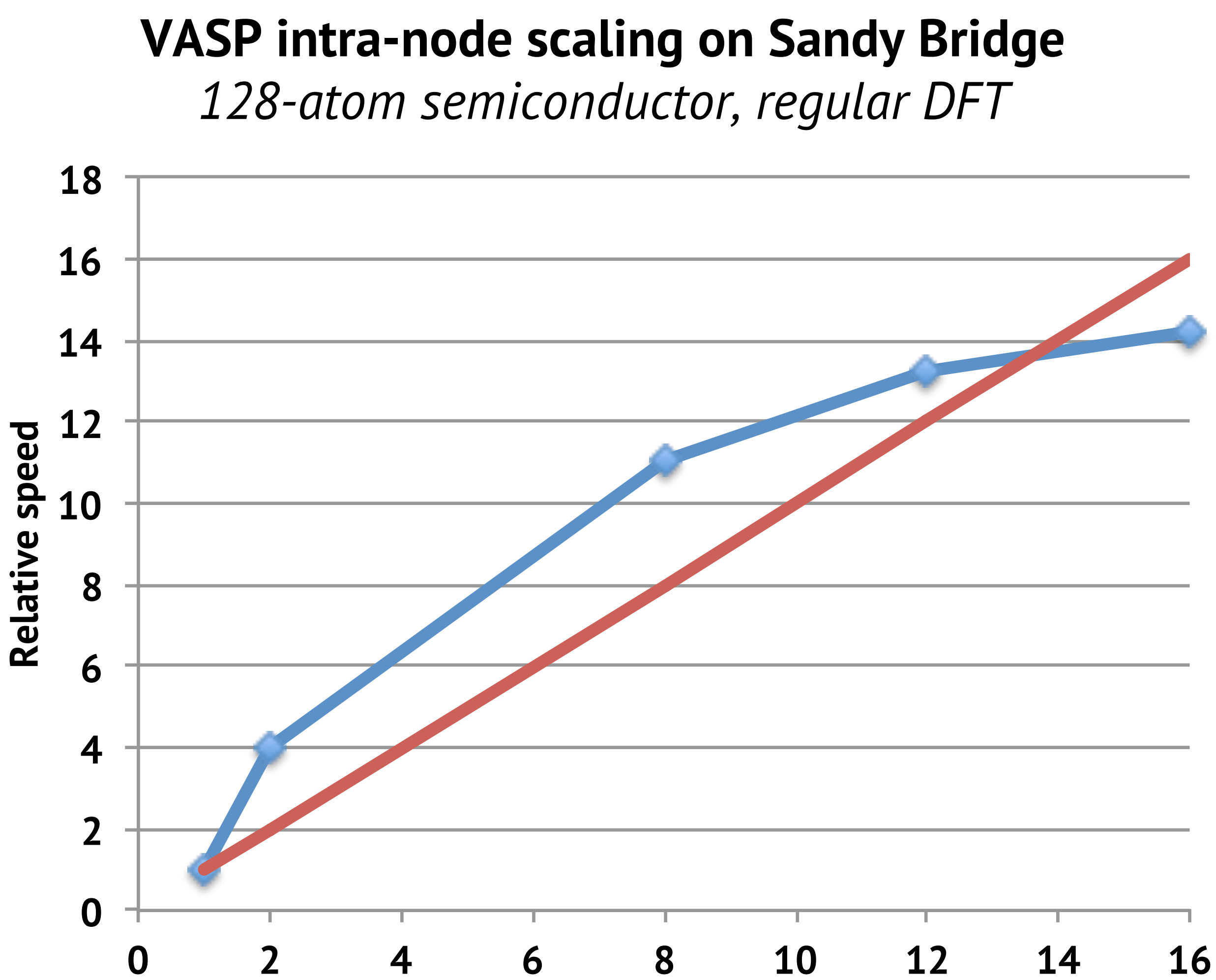
Little time is spent in vectorized instructions. Check the compiler's vectorization advice to see why key loops could not be vectorized.

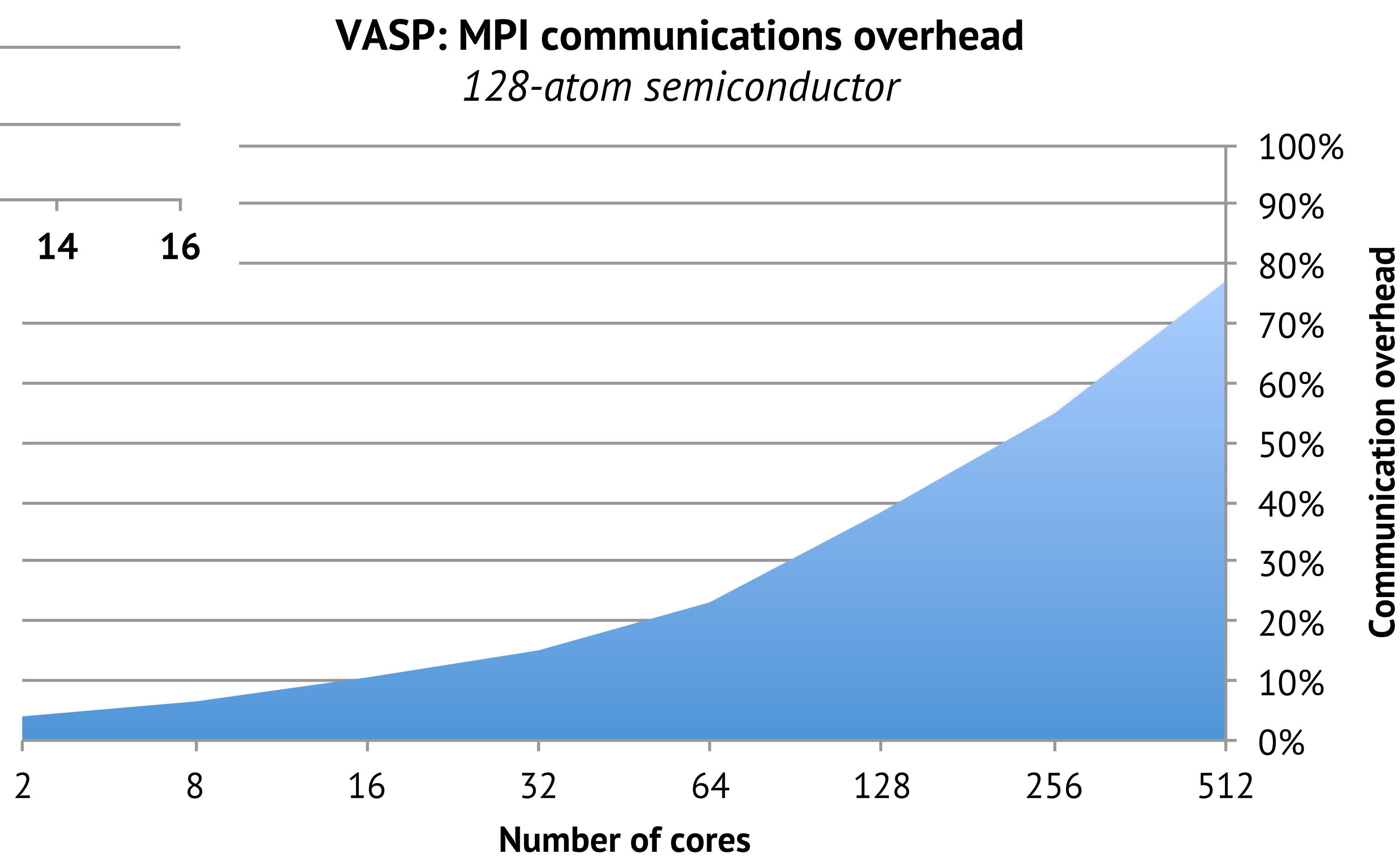
MPI

Of the 23.2% total time spent in MPI calls:

Time in collective calls	95.0%
Time in point-to-point calls	5.0%
Effective process collective rate	416 MB/s
Effective process point-to-point rate	59.8 MB/s

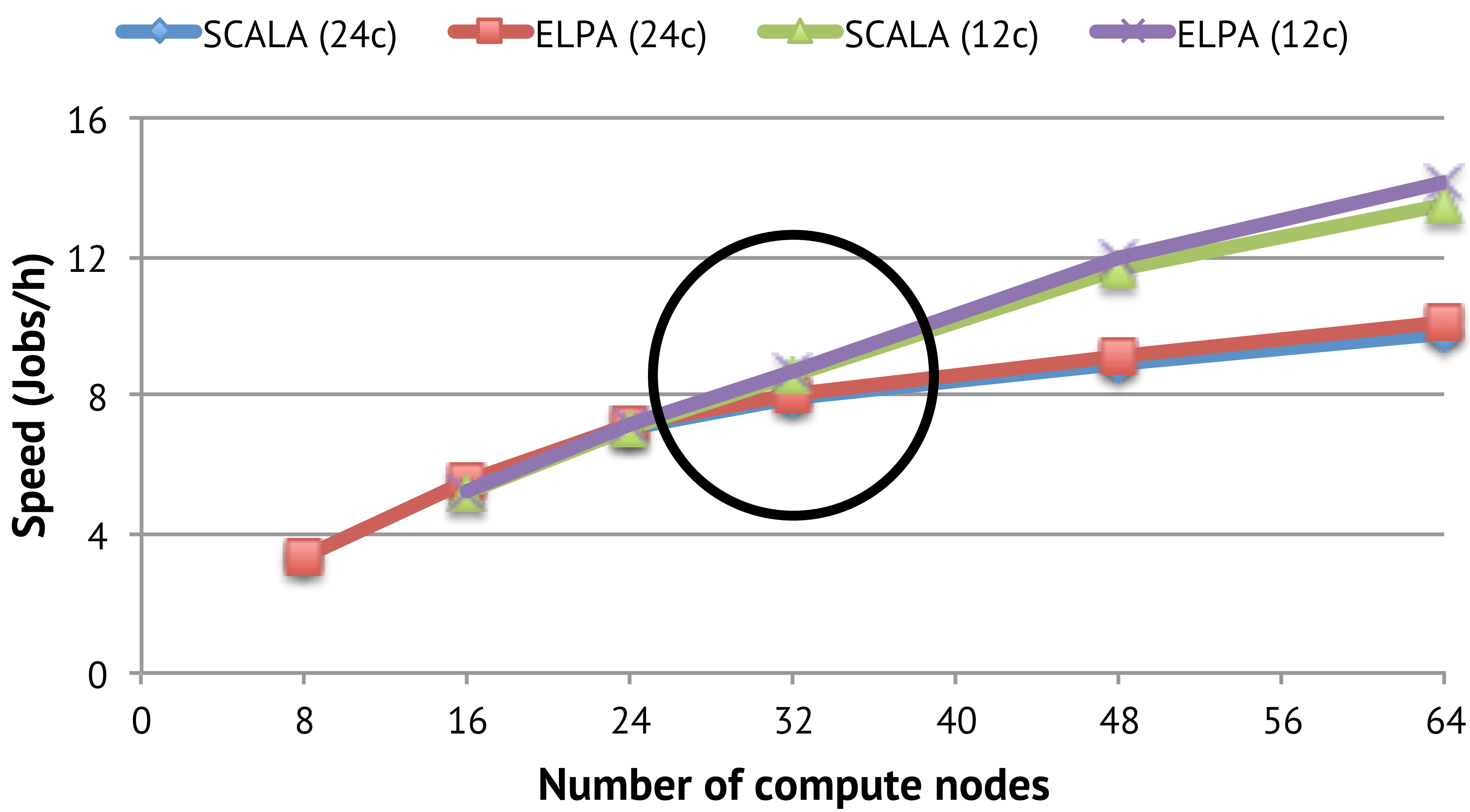
Most of the time is spent in collective calls with an average transfer rate. Using larger messages and overlapping communication and computation may increase the effective transfer rate.


VASP intra-node scaling on Sandy Bridge
128-atom semiconductor, regular DFT
Relative speed
0
2
4
6
8
10
12
14
16
18
0
2
4
6
8
10
12
14
16


VASP: MPI communications overhead
128-atom semiconductor
Communication overhead
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
2
8
16
32
64
128
256
512
Number of cores

VASP on Lindgren: ELPA vs SCALAPACK

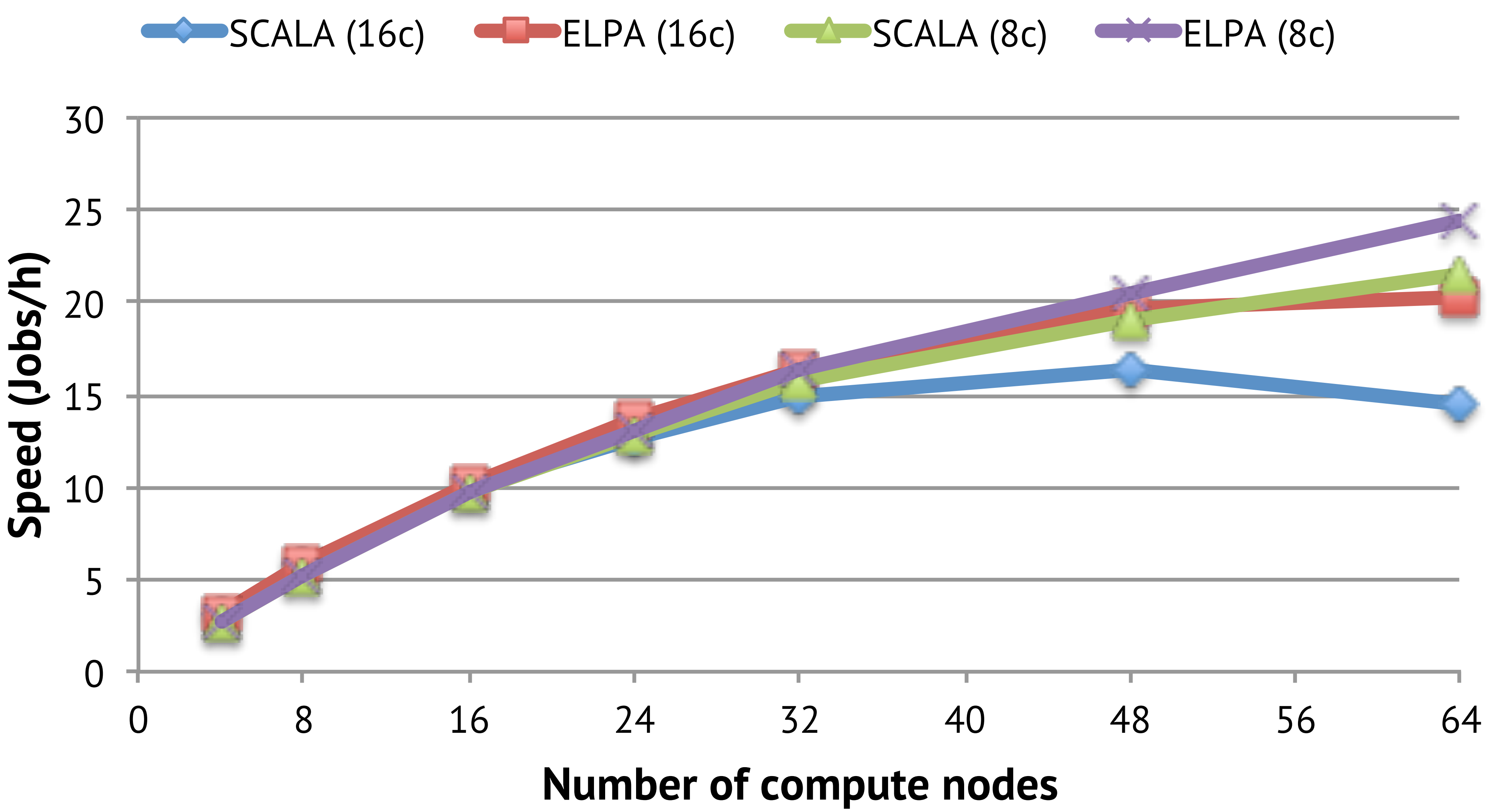
MgH2 (1269 atoms)

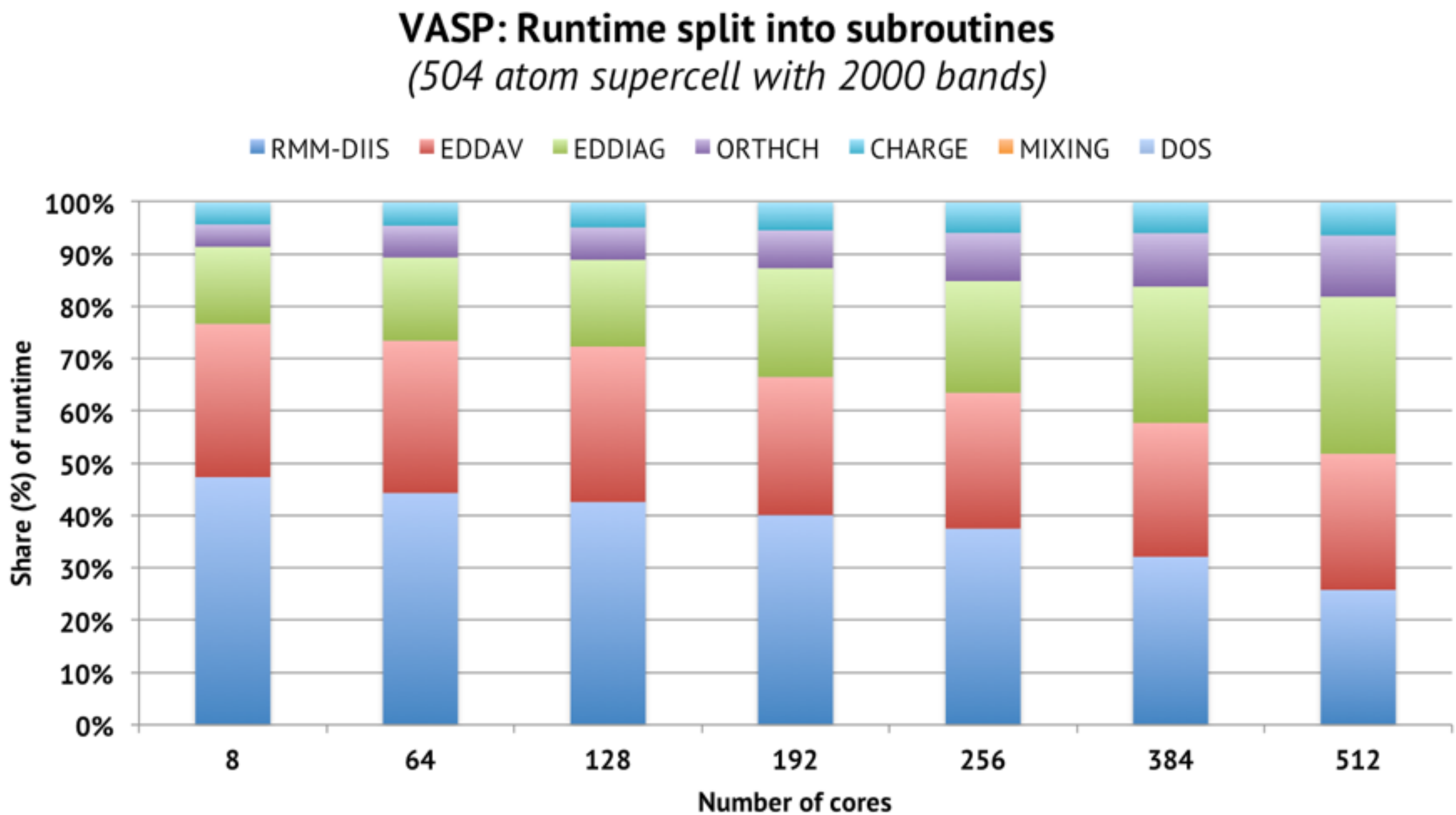


http://elpa.rzg.mpg.de/

VASP on Triolith: ELPA vs SCALAPACK (Intel MPI + MKL)

MgH2 (1269 atoms)





EDIAG/ORTCH starts to dominate (SCALAPACK)

LPLANE

- Activates "plane-wise data distribution in real-space". This affects parallelization, especially of Fast-Fourier transforms.

- Default value is .TRUE.

- From the manual:
  make sure that **NGZ > 3*cores/NPAR**

- Best load balance with **NGZ=n*NPAR**

- **Example: (NGZ 64) == 16*(NPAR 4)**

Check for LPLANE = .TRUE.

```
$ grep "minimum data exchange" OUTCAR
 parallel 3D FFT for wavefunctions:
    minimum data exchange during FFTs selected (reduces bandwidth)
 parallel 3D FFT for charge:
    minimum data exchange during FFTs selected (reduces bandwidth)
```

Check load balance in OUTCAR

32 compute nodes:

```
…
real space projection operators:
  total allocation    :     638386.16 KBytes
  max/ min on nodes   :      40667.10       38042.62
…
```

128 compute nodes:

```
…
 real space projection operators:
  total allocation    :     638386.16 KBytes
  max/ min on nodes   :      59273.56       41218.42
…
```

NGZ: case study

- 128-atom semiconductor
- PREC=Accurate / ENCUT=400 eV gives NGZ = 60
- PREC=Accurate / ENCUT=500 eV gives NGZ = 70
- Linear parallel scaling from 5 to 6 compute nodes = 120% speed.
- 400 eV gives 120%.
- 500 eV gives 108% (11% loss). Why? 500 eV has more compute work per core. Should scale better!
- 6 nodes means NPAR=6 with NGZ=70. **MOD(70,6) ≠ 0**

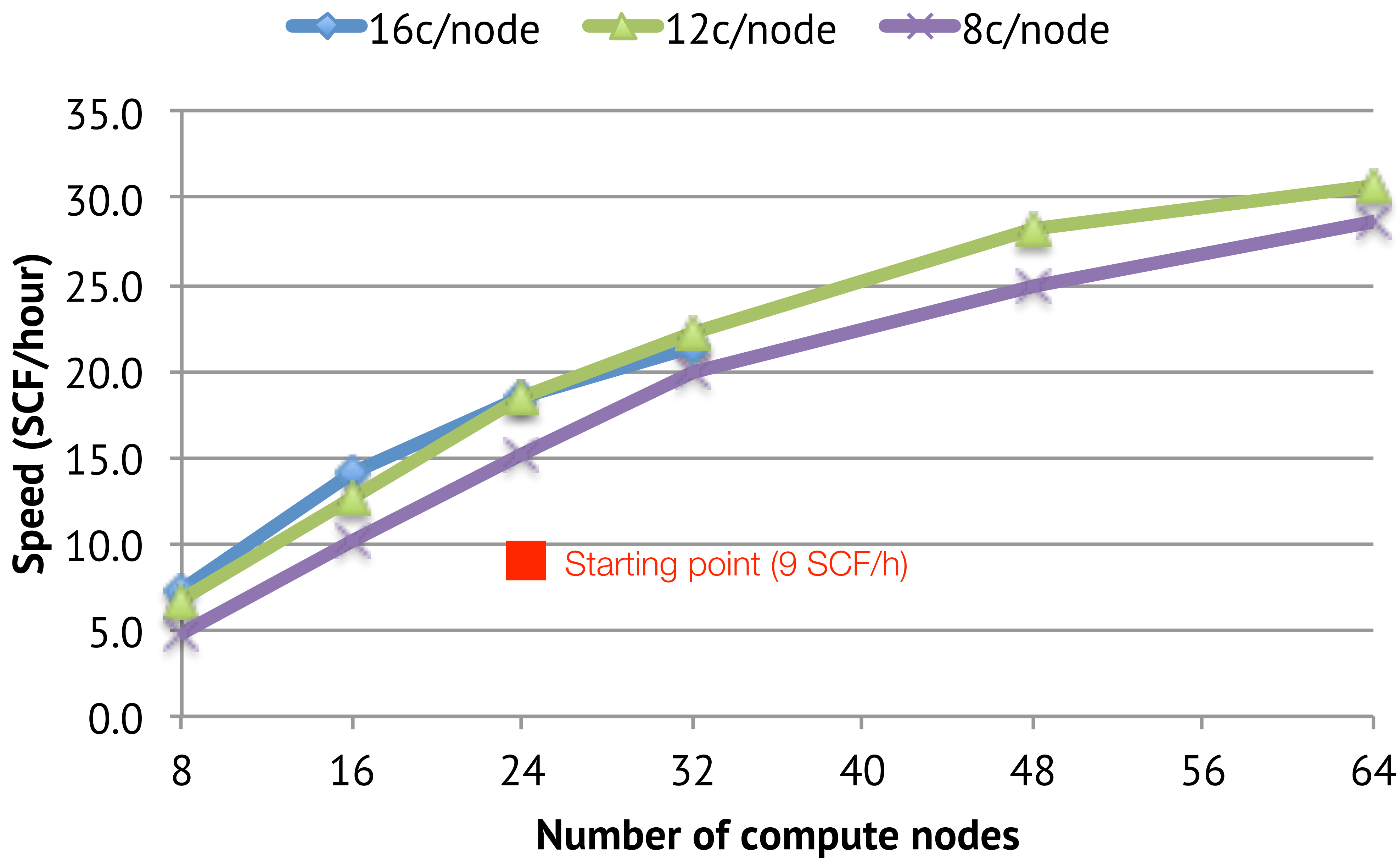
MPI-libraries

- Don't compile your own OpenMPI from source, e.g. on a Cray machine, use CrayMPI!
- At NSC in Sweden, we rely on Intel MPI.
- Intel MPI 4.0.3 works fine with VASP. Version 4.1+ have problems with hybrid-DFT calculations. You need to set **I_MPI_COMPATIBILITY=4**.
- MVAPICH 1.8-2.0 also ok.
- OpenMPI usually slower than IMPI/Mvapich for VASP.
- **Process pinning / core-binding is essential (10-20% faster).** IMPI pins automatically, OpenMPI does not.

Hybrid-DFT case study

- A 128-atom supercell with HSE06.
- Need high accuracy, 4-8 k-points.
- The research groups typically runs this simulation on 8-24 nodes (sometimes with 8 cores/node).
- Speed about 9 SCF steps/hour.
- Is there a better way to run? Can k-point parallelization help?

VASP: 128 atom hybrid-DFT job (HSE06) k-point parallelization with KPAR=4



Hybrid-DFT case study

- Switching to VASP 5.3.5 from 5.3.3 (faster hybrids) ≈ 5%
- Recommended better, already existing binaries. ≈ 5%
- K-point parallelization ≈ 50%
- Changed NGZ (extra basis functions for free!) ≈ 10%
- Boosted cores/node from 8 to 12. ≈ 10%
- **Final outcome was 2x performance boost** at their current job size (24 compute nodes).
- Proof-of-concept for further scaling up to at least 64 compute nodes