CASTEP: plane-wave DFT for exascale HPC

Matt Smith, Ben Durham and Phil Hasnip

PAX Project Meeting
April 2024

First-principles materials modelling

- Aim: predict materials' behaviour from *first principles* i.e. no knowledge of what they'll do beforehand
- Materials are made from atoms
- Most behaviour depends on the electrons
- We can use quantum mechanics to model electrons

The material's behaviour should emerge from the simulation

First-Principles Modelling

Most common approach is Density Functional Theory (DFT); we solve a form of Schrödinger equation:

$$\hat{H}_k[\rho]\psi_{bk}(\mathbf{r}) = E_{bk}\psi_{bk}(\mathbf{r})$$

First-Principles Modelling

Most common approach is Density Functional Theory (DFT); we solve a form of Schrödinger equation:

$$\hat{H}_k[\rho]\psi_{bk}(\mathbf{r}) = E_{bk}\psi_{bk}(\mathbf{r})$$

where

$$\hat{H}_k[\rho] = -\frac{\hbar^2}{2m}\nabla^2 + V_{\mathrm{loc}}[\rho](\mathbf{r}) + \hat{V}_{\mathrm{nl}}$$

First-Principles Modelling

Most common approach is Density Functional Theory (DFT); we solve a form of Schrödinger equation:

$$\hat{H}_k[\rho]\psi_{bk}(\mathbf{r}) = E_{bk}\psi_{bk}(\mathbf{r})$$

where

$$\hat{H}_k[\rho] = -\frac{\hbar^2}{2m}\nabla^2 + V_{\mathrm{loc}}[\rho](\mathbf{r}) + \hat{V}_{\mathrm{nl}}$$

and

$$\rho(\mathbf{r}) = \sum_{bk} |\psi_{bk}(\mathbf{r})|^2.$$

First-Principles Modelling

Most common approach is Density Functional Theory (DFT); we solve a form of Schrödinger equation:

$$\hat{H}_k[\rho]\psi_{bk}(\mathbf{r}) = E_{bk}\psi_{bk}(\mathbf{r})$$

where

$$\hat{H}_k[\rho] = -\frac{\hbar^2}{2m}\nabla^2 + V_{\mathrm{loc}}[\rho](\mathbf{r}) + \hat{V}_{\mathrm{nl}}$$

and

$$\rho(\mathbf{r}) = \sum_{bk} |\psi_{bk}(\mathbf{r})|^2.$$

So to find ψ_{bk} we need $\hat{H}_k$, which depends on ρ, which depends on ψ_{bk}... solve *iteratively*.

First-Principles Modelling

$$\hat{H}_k[\rho]\psi_{bk}(\mathbf{r}) = E_{bk}\psi_{bk}(\mathbf{r}).$$

Main basis set choices to represent ψ_{bk}:

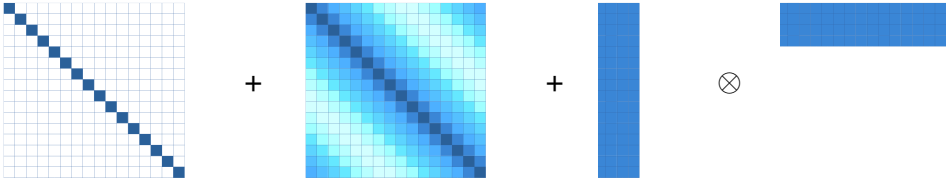
- Local basis set (e.g. CRYSTAL, CP2K; recall Marcello's talk)
  - Hamiltonian is compact
  - Constructing Hamiltonian resource-intensive
  - E.g. CRYSTAL
- Plane waves (e.g. CASTEP)
  - Hamiltonian is large
  - Constructing Hamiltonian is simple

The plane-wave Hamiltonian

Typical HPC simulation might be:

- 100,000 plane-waves
- 1000 electrons
- 10 k-points

$$-\frac{\hbar^2}{2m}\nabla^2 \quad + \quad V_{\mathrm{loc}}[\rho](\mathbf{r}) \quad + \quad \hat{V}_{\mathrm{nl}}$$



Where does CASTEP spend its time?

- Applying $\hat{H}_k$ to ψ_{bk}

$$\hat{H}_k[\rho] = -\frac{\hbar^2}{2m}\nabla^2 + V_{\text{loc}}[\rho](\mathbf{r}) + \hat{V}_{\text{nl}}.$$

  - The kinetic energy is applied in reciprocal-space
  - The local potential is applied in real-space
  - The non-local potential can be applied in either space

  We need to Fourier transform between the two spaces.

- Orthogonalisation of ψ_{bk}
  We need to ensure our trial bands are orthogonal to each other.
  We compute the overlap matrix between all pairs of bands, and invert it.

Large calculations

As we simulate larger and larger systems, N_G and N_b increase and N_k decreases (for very large cells $N_k = 1$).

- Time for Fourier transforms scales as $N_G \log N_G N_b N_k$.
  For very large systems $\sim N_G N_b$.
- Time for orthogonalisation scales as $N_G N_b^2 N_k$.
  For very large systems $\sim N_G N_b^2$.

$\longrightarrow$ Orthogonalisation dominates in large calculations.

k-point parallelism

- Equations at different **k**-points are almost entirely independent of each other
  $\longrightarrow$ give each core a subset of the **k**-points.
  $\longrightarrow$ each core solves a subset of Kohn-Sham equations
- Cores only communicate when constructing the density

$$\rho(\mathbf{r}) \quad = \quad \sum_{bk} |\psi_{bk}(\mathbf{r})|^2$$

TiN Benchmark

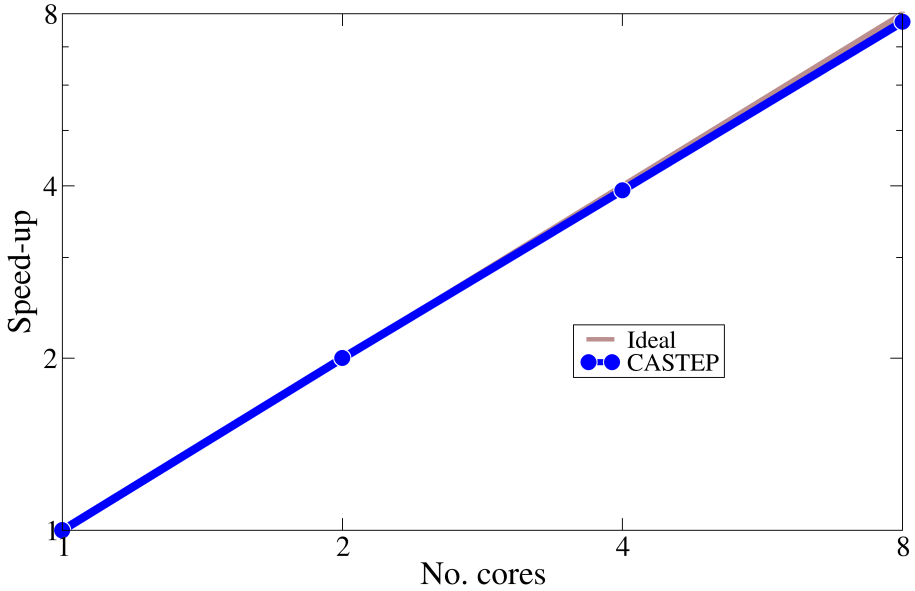
There are some benchmark calculations at:

`http://www.castep.org/CASTEP/Benchmarks`

The TiN simulation is a small standard benchmark

- 33 atoms
- 8 **k**-points
- 164 bands
- 10,972 **G**-vectors

k-point parallelism in action



Exascale HPC

Materials modelling software typically based on:

- Modern Fortran
- MPI
- OpenMP

Exascale machines characterised by:

- GPUs
- Massive parallelism

Enter the GPU

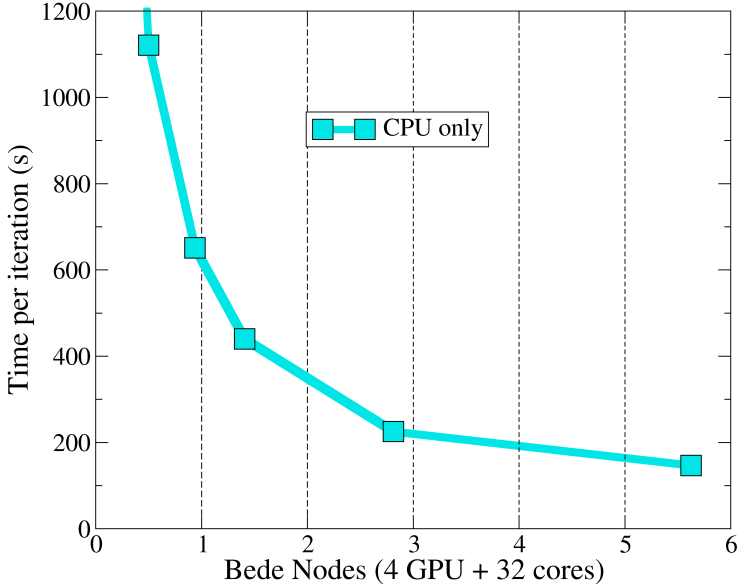
$$\hat{H}_k[\rho] \quad = \quad -\frac{\hbar^2}{2m}\nabla^2 + V_{\mathrm{loc}}[\rho](\mathbf{r}) + \hat{V}_{\mathrm{nl}}.$$

- The kinetic energy is applied in reciprocal-space
- The local potential is applied in real-space
  Needs FFTs $\longrightarrow$ optimised cuFFT library
- The non-local potential is a dense matrix-matrix multiplication
  Standard BLAS/LAPACK $\longrightarrow$ optimised cuBLAS library

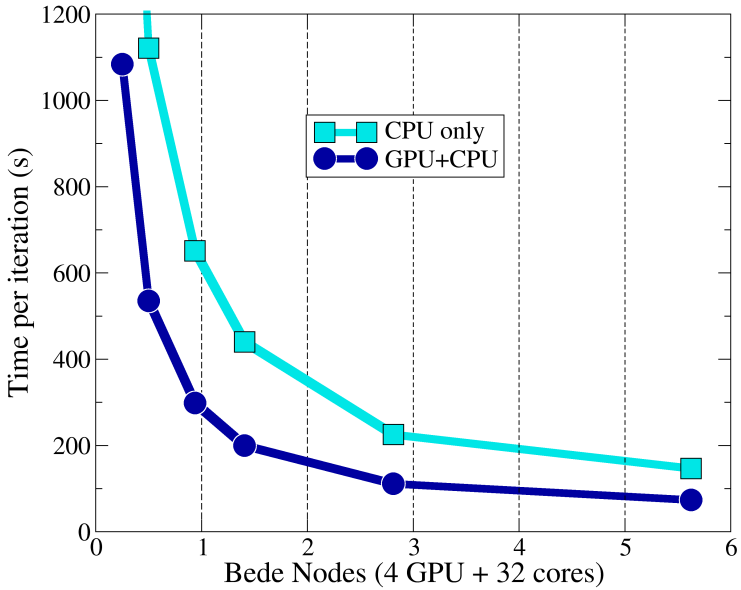
Focus on:

- Single code base
- Directives-based data movement (OpenACC)
- Use of optimised GPU libraries

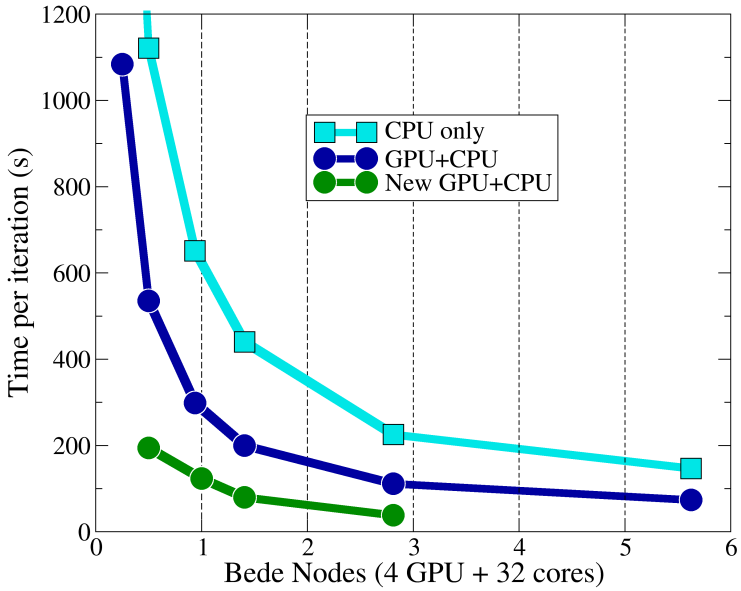
CASTEP-GPU on Bede (UK Tier-2 HPC)



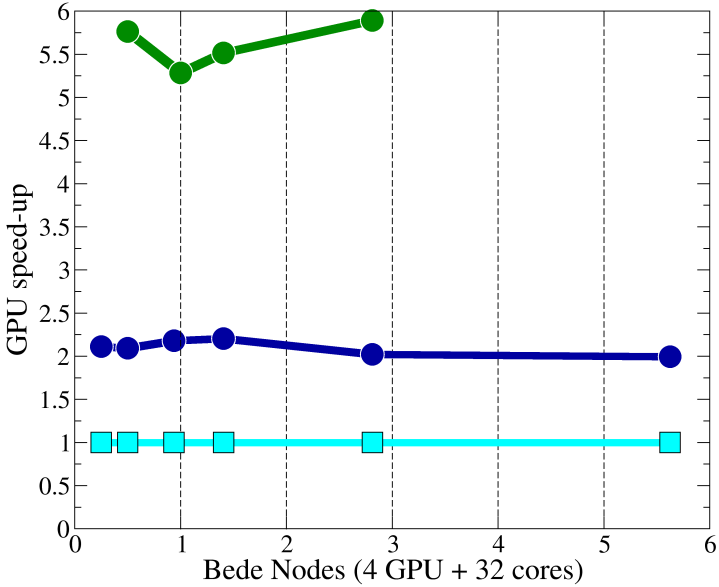
CASTEP-GPU on Bede



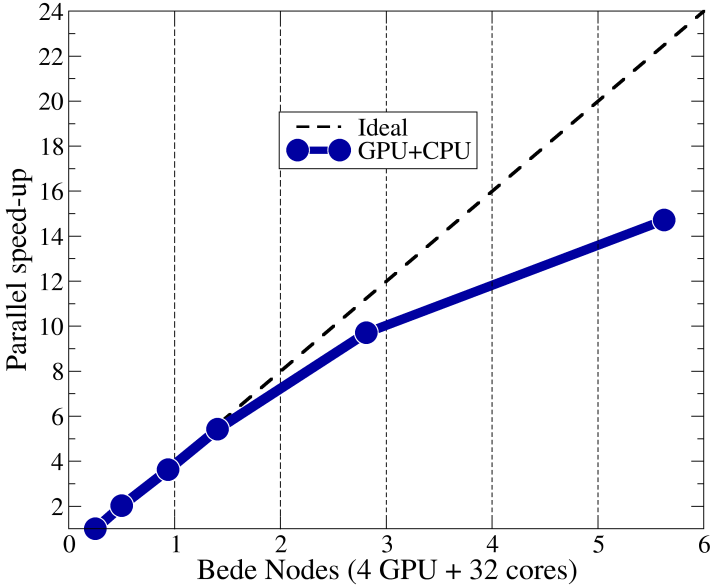
CASTEP-GPU on Bede (UK Tier-2 HPC)



CASTEP-GPU on Bede (UK Tier-2 HPC)



CASTEP-GPU on Bede (UK Tier-2 HPC)



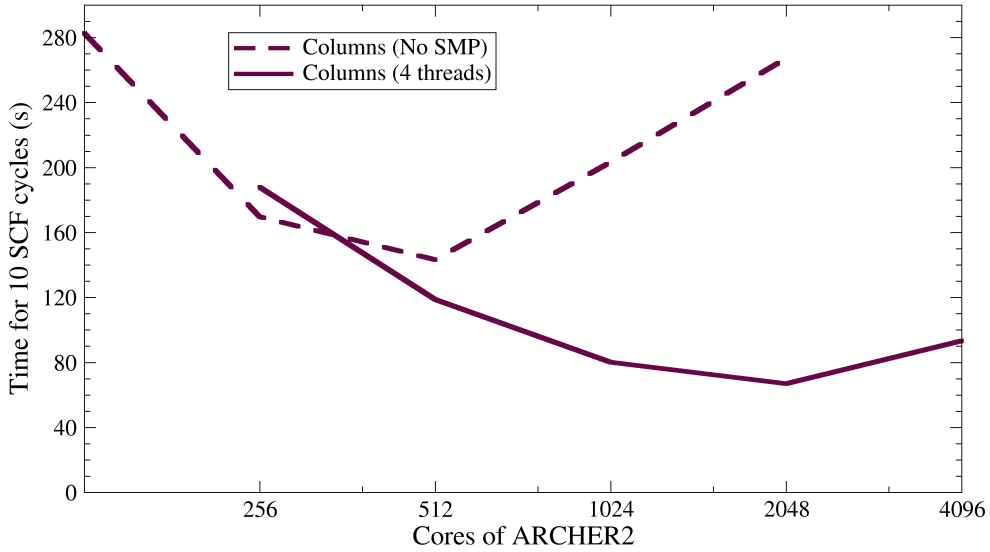
Accelerating or breaking?

- Multiple MPI ranks share each GPU (keeps occupancy high)
- Minimising data movement is crucial to performance
- k-point and band-parallel
- Not quite single code-base yet

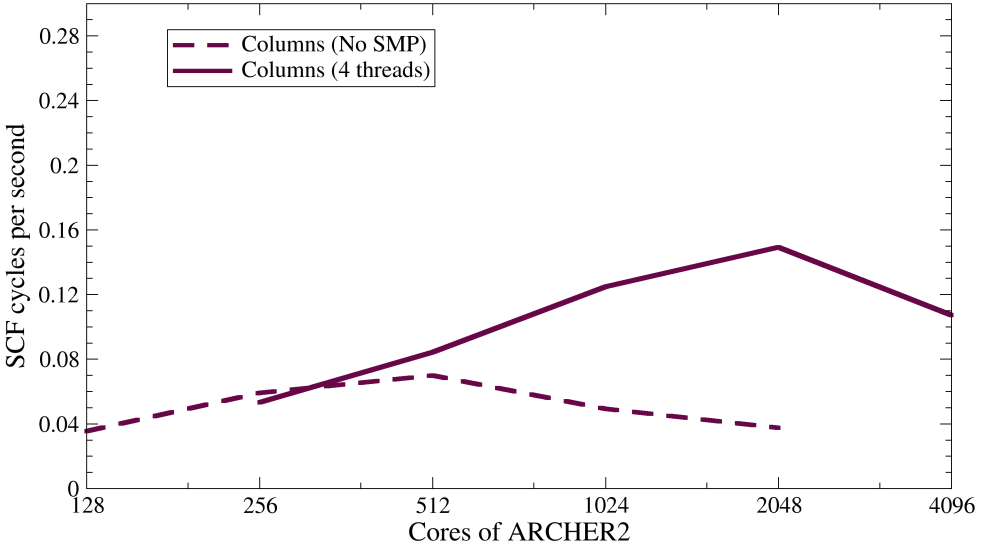
Parallelisation

- Exascale HPC machines are massively parallel
- How does plane-wave DFT scale?
- Can pure MPI get us to exascale?

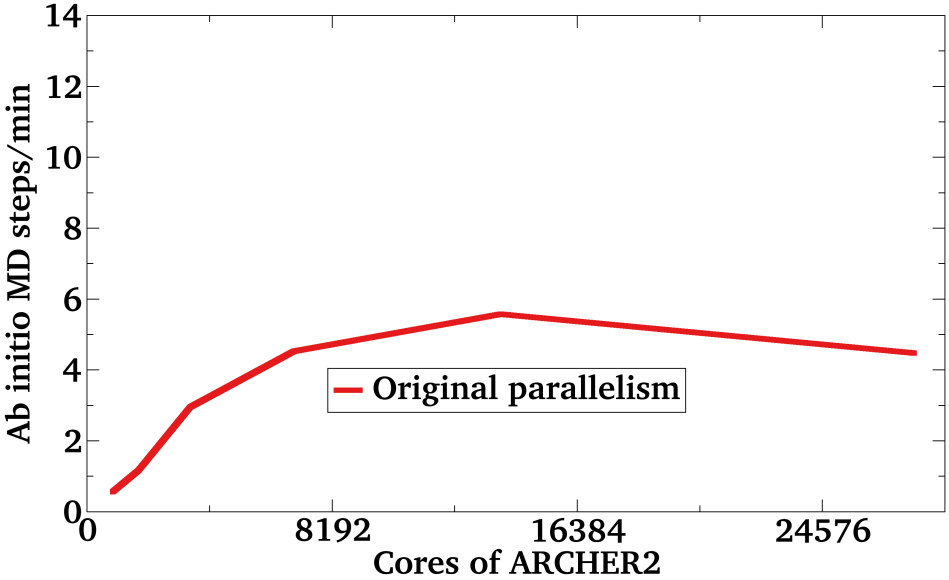
Performance: Time (Al_2O_3 on ARCHER2)



Performance: Speed (Al_2O_3 on ARCHER2)

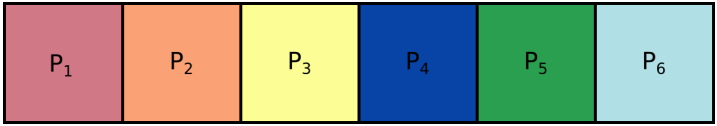


Performance: Ab initio MD



Plane-wave parallelism problems

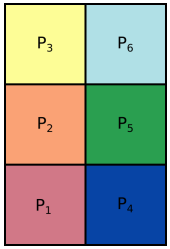
- For plane-wave DFT, the parallel FFT limits the scaling
- FFTs need all-to-all comms
- For P processes, comms time scales as P^2
- Need to re-think the data distribution...



$O(P^2)$

Rethinking the distribution

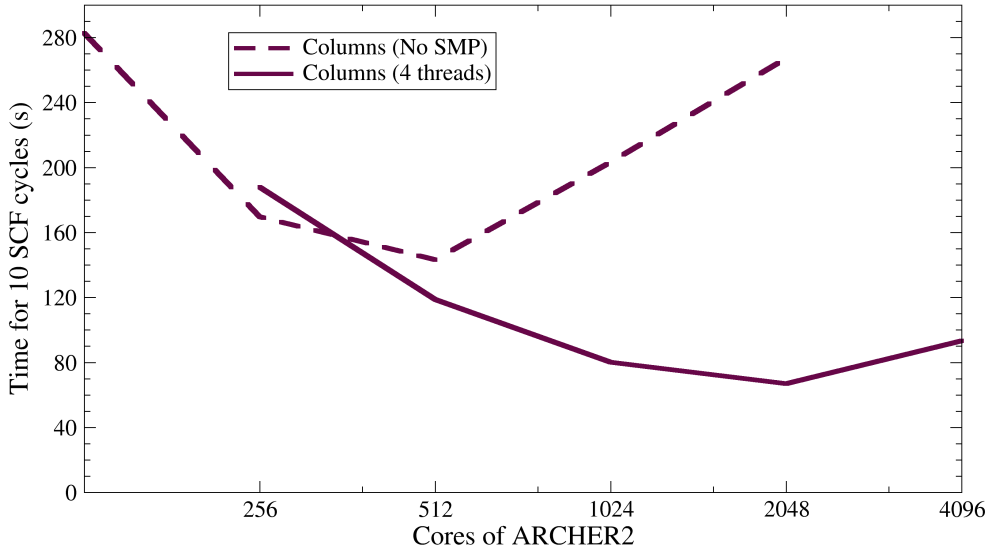
- Key idea: arrange processes in a logical process grid, side $\sim \sqrt{P}$
- Each data transposition only involves processes in either the same process row or column



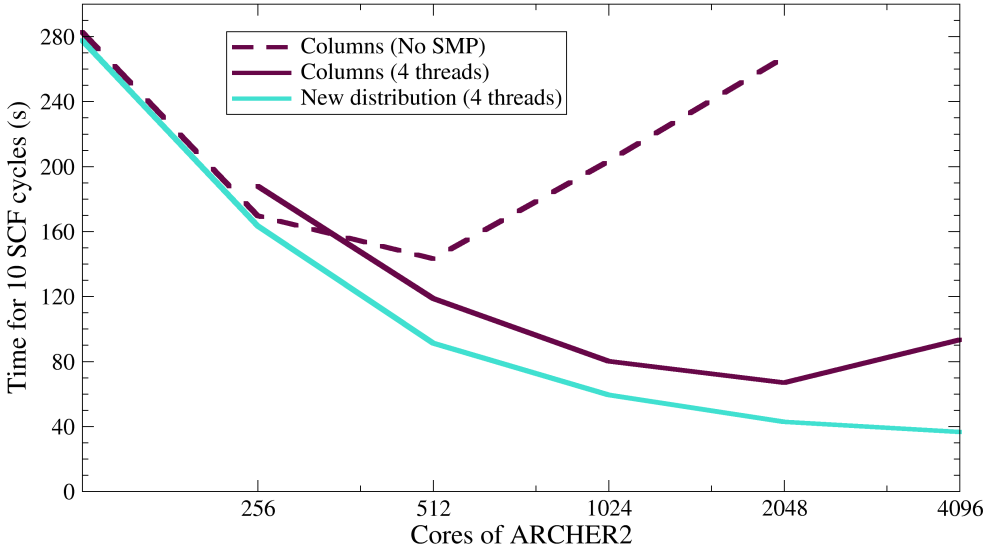
$O(P)$

- Restrict communications to sub-communicator rows and columns of process grid

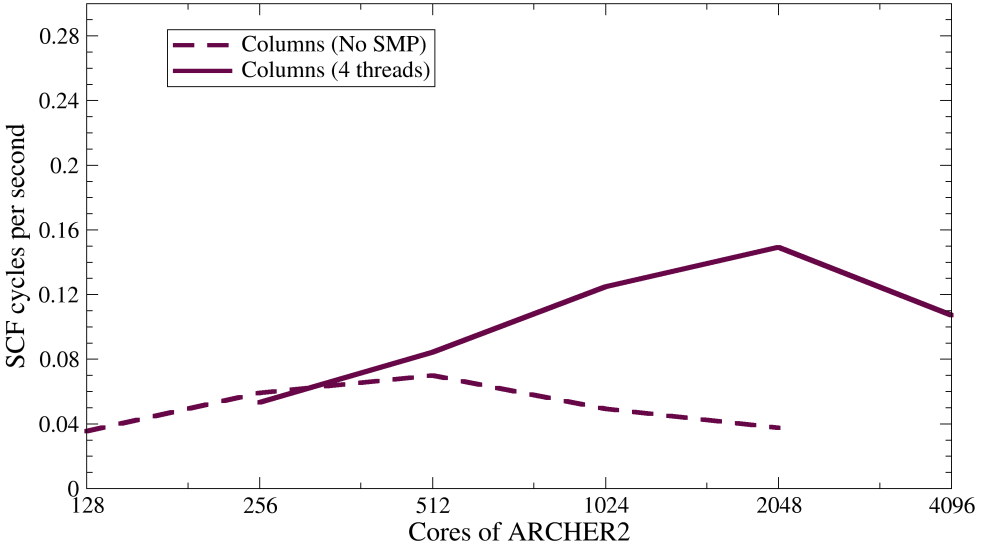
Performance: Time (Al_2O_3 on ARCHER2)



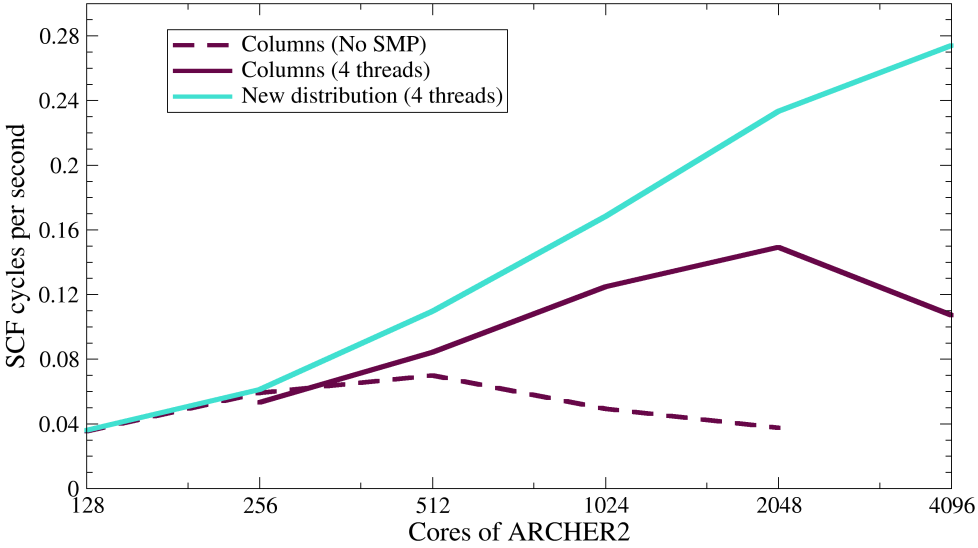
Performance: Time (Al_2O_3 on ARCHER2)



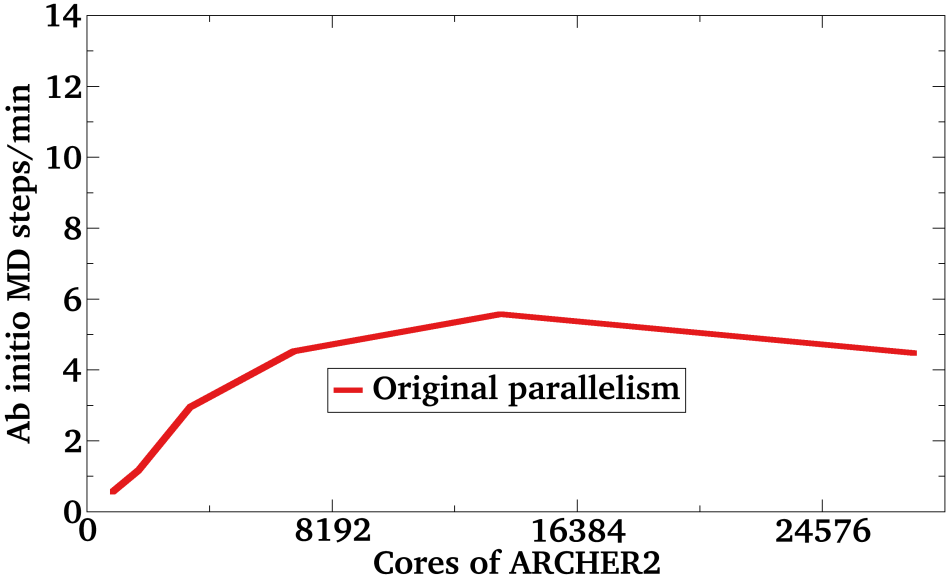
Performance: Speed (Al_2O_3 on ARCHER2)



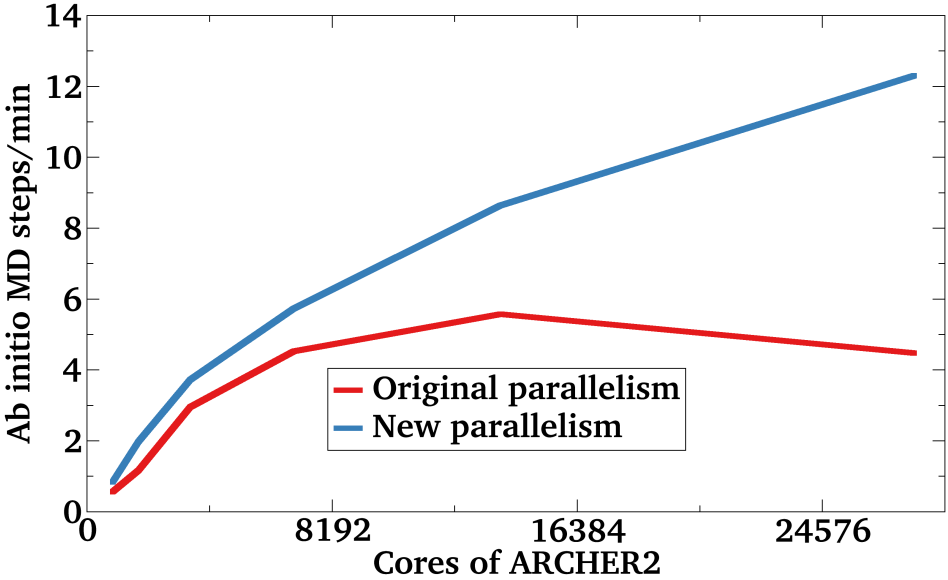
Performance: Speed (Al_2O_3 on ARCHER2)



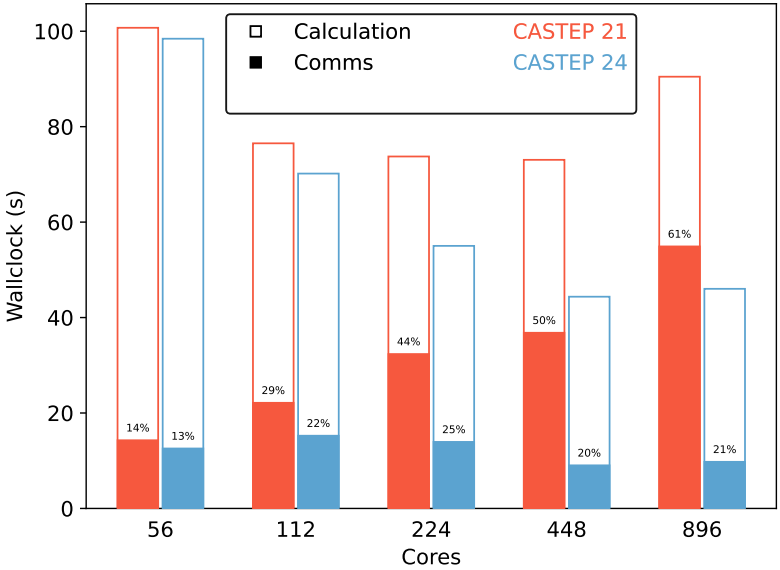
Performance: Ab initio MD



Performance: Ab initio MD



Performance analysis (Solid benzene on CSD3)



Re-imagined parallelism

- Emphasis on scaling (improved 4x)
- Faster on all core counts
- Gives a 1.6x speed-up on 2048 cores
  $\longrightarrow$ 63% of the time-to-science
- Other communications are now the bottleneck!

Conclusions

- Many challenges... but significant performance improvements
- Re-engineering for GPUs: 6-8X
- Re-thinking parallel decompositions: 4X scaling
- Now need to combine the two!

CASTEP GPU:
M.J. Smith et al, *Comput. Sci. Eng.* **24**(1) 46-55 (Jan-Feb 2022); doi: 10.1109/MCSE.2022.3141714

New CASTEP parallelism:
B. Durham, M.J. Smith & P. Hasnip, eCSE Technical Report