Running Large-Scale Calculations with Hybrid MPI/OpenMP parallelism

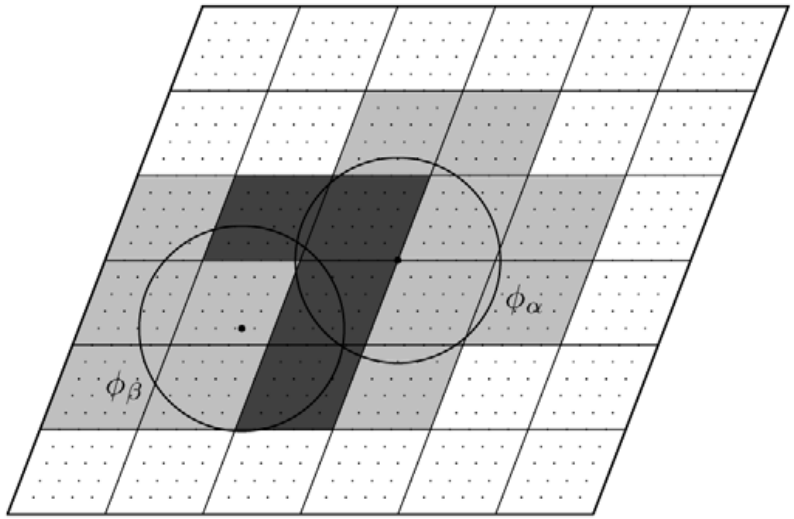
Nicholas D.M. Hine

Winton Advanced Research Fellow, Cavendish Laboratory, University of Cambridge

ONETEP Masterclass 2013, Cambridge, 28th August 2013

Computational Effort in ONETEP

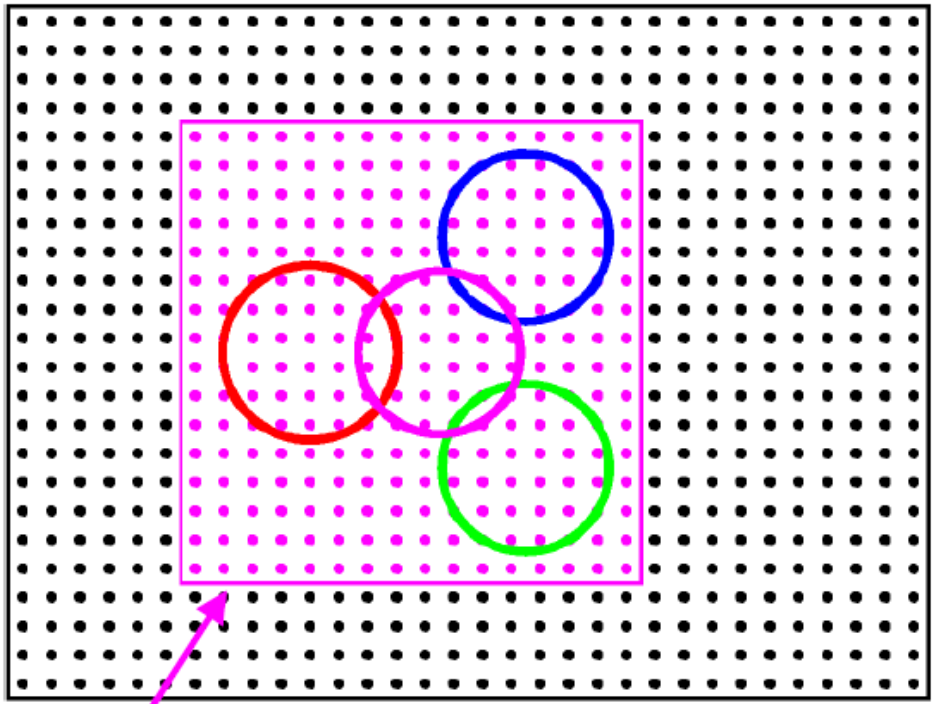
1. FFT Row Sum operations



Communicate NGWF data, deposit to large arrays

Computational Effort in ONETEP

2. FFT box Fourier Transforms

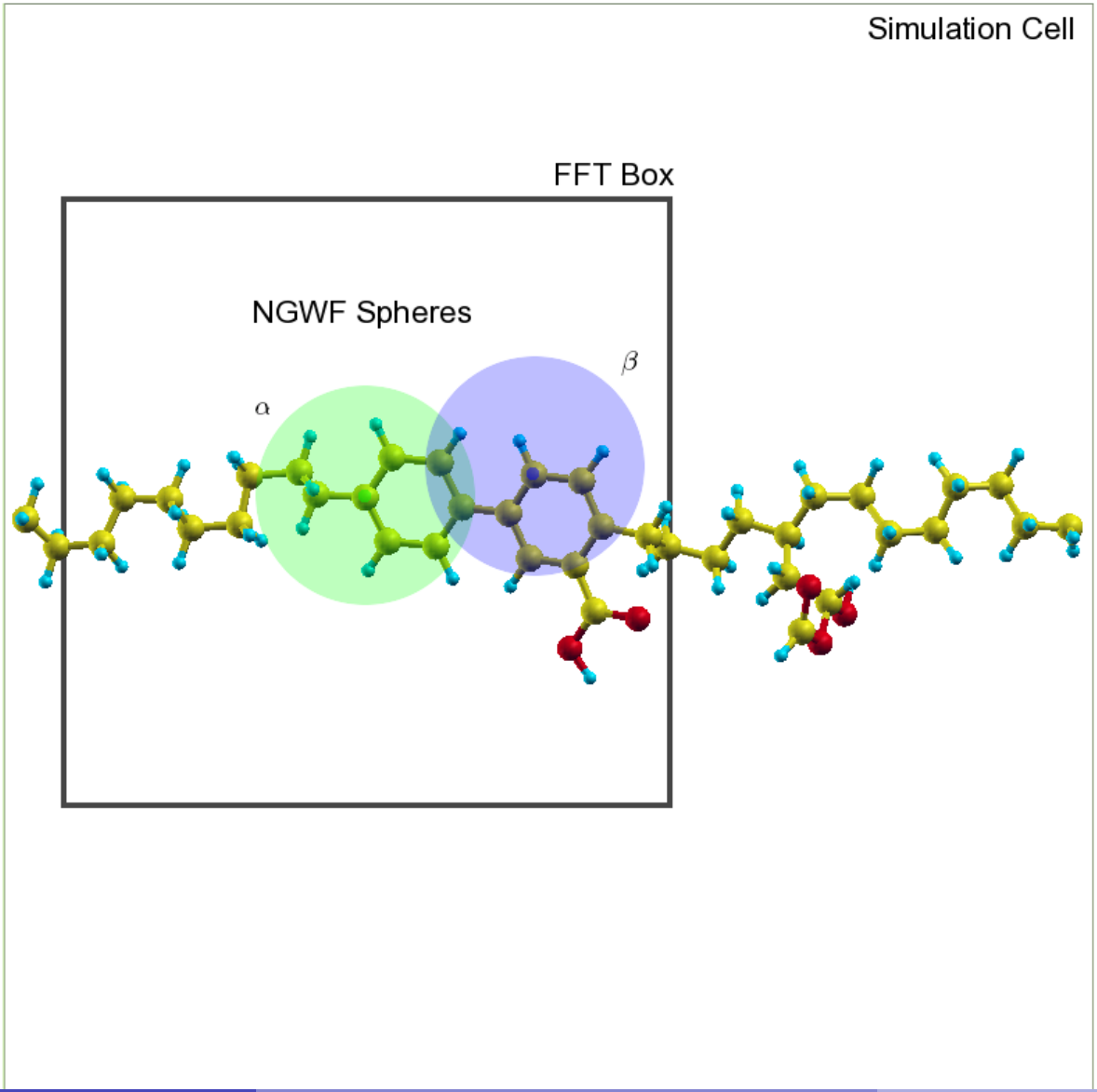


C.-K. Skylaris, A. A. Mostofi, P. D. Haynes, C. J. Pickard & M. C. Payne, *Comp. Phys. Comm.* **140**, 315 (2001)
A. A. Mostofi, C.-K. Skylaris, P. D. Haynes & M. C. Payne, *Comp. Phys. Comm.* **147**, 788 (2002)

Local FFTs on large box of data: density, local potential, nonlocal projectors, NGWF gradient

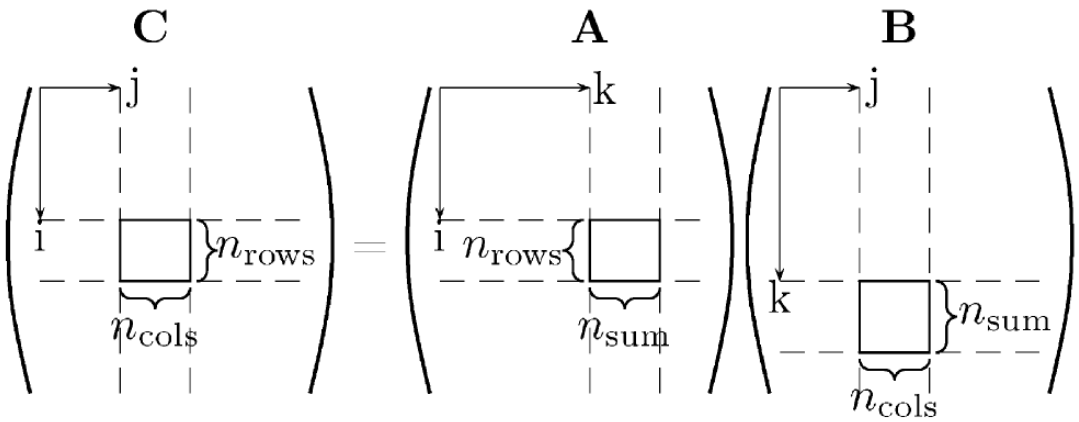
Computational Effort in ONETEP

3. Whole Cell Grid Extract/Deposit

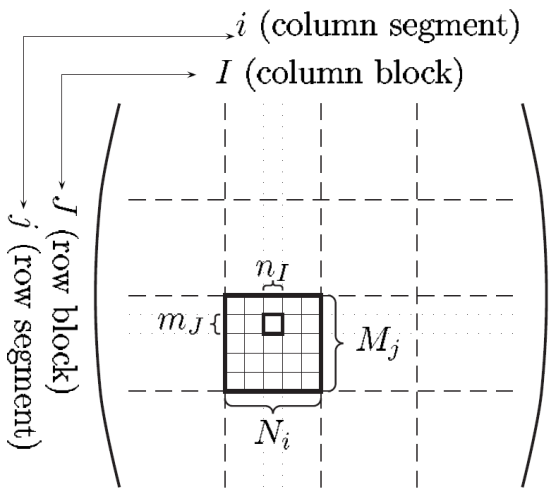


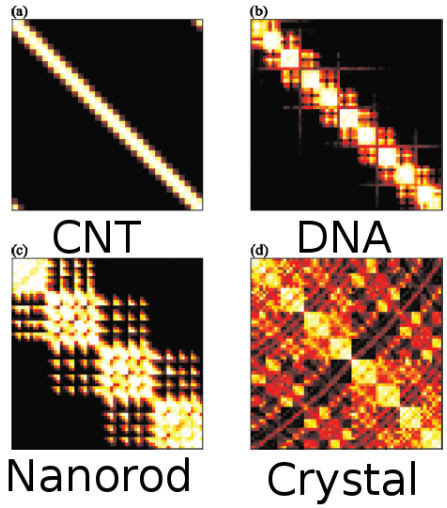
Computational Effort in ONETEP

4. Sparse Matrix Algebra



Communicates matrix data, multiplies segments.





Computational Effort in ONETEP

5. Other
   - Initialisation (Sparse Algebra Setup, preparing local potential, core density etc)
   - Whole-Cell FFTs (usually negligible, slightly less so for GGAs, considerably less so for van der Waals DFs)
   - Multigrid Hartree

Main Data Structures

- FFT boxes: number controlled (in 3.4 and beyond) by fftbox_batch_size
- Whole cell grids (3-10 stored at any one time, dependent on options), parallelised over slabs in 12-direction (real space)
- Sparse Matrices (SPAM3 type) parallelised over columns
- Workspaces (300-500MB, depending on options)

Grouped Communications: nodes share data. Default group size is closest power of two to square-root of number of processes (can adjust with comms_group_size)

All-MPI Parallelism needs more memory: 2GB/core minimum
With OpenMP, can go down to 1GB/core

Hybrid Parallelism

MPI Parallelism

- Message Passing Interface
- Splits code into lots of MPI 'processes', each running the same code
- Performance dependent on interconnect speed between nodes
- Uses shared memory for messages between processes on same node, but still copies between memory locations

OpenMP Parallelism

- Open Multi-Processing
- Shared-Memory multithreaded model - direct access by one thread to memory of another
- Runs one 'master' thread, which splits into multiple threads inside PARALLEL regions
- Only acts within a node

Tips for Large Jobs

- Aim for balanced load: run VERBOSE and check for balanced min/max NGWFs per node
- Aim for 2-10 atoms per core (can go lower if heavily using OpenMP: see KAW talk).
- Increase fftbox_batch_size if you can