Science & Technology Facilities Council

# Boosting the scaling performance of CASTEP:

## Enabling next generation HPC for next generation science

Dominik Jochym

STFC Rutherford Appleton Laboratory

4th October 2011

# Outline

1. MPI collective optimisations of I/O

2. Optimisation of error reporting

# CASTEP is…

- A general-purpose ‘first principles’ atomistic modeling code
- Based on density functional theory

Written in

- Fortran 95 + extensions
- BLAS/LAPACK for linear algebra
- FFT libraries (where available)
- MPI for parallel communication

# CASTEP can…

- Compute the electronic density
- Determine the atomic configuration and cell
- Simulate molecular dynamics (Born-Oppenheimer, path-integrals, variable cell)
- Calculate band-structures and density of states
- Compute various spectra (optical, IR, Raman, NMR, XANES...)
- plus linear response, population analysis, ELF, TDDFT and more…

# Key CASTEP components

- Kohn-Sham equations

$$H_k[n]\psi_{bks}(r) = \epsilon_{bks}\psi_{bks}(r)$$

- In a plane-wave basis

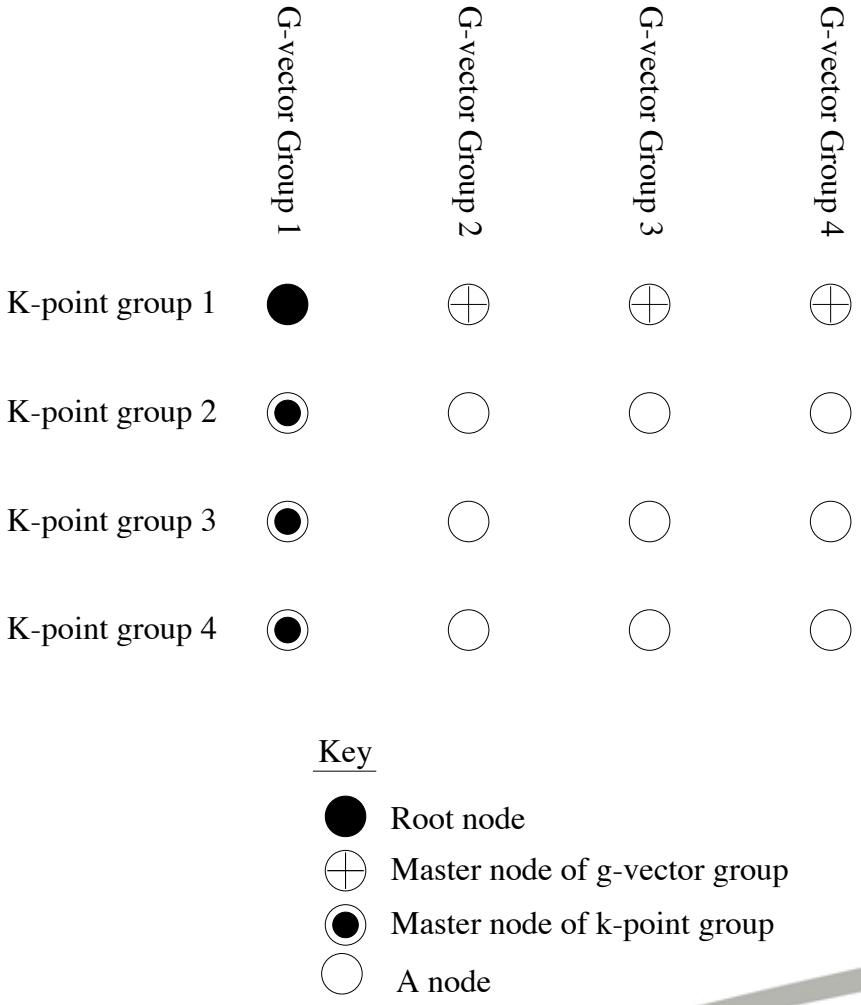
$$\psi_{bks}(r) = \sum_G c_{Gbks} e^{i(G+k)\cdot r}$$

- Wavefunction coefficients

```
wvfn%coeffs(1:nG,1:nbands,1:nkpts,1:nspins)
```

Science & Technology Facilities Council

# Parallel distribution




Three data distribution strategies

1. k-points
2. g-vectors
3. bands

Science & Technology Facilities Council

# Checkpoints and wavefuntions

- HECToR limited to a 12 hour run time, checkpoint and restart mechanism required
- Wavefunction manipulation can require collection and redistribution of data

Problems

- Long checkpoint write/read times
- MPI ‘unexpected buffer’ error

# Cause of the problem?

- Many point-to-point MPI send/receive calls
- When it works, comms can get expensive
- When it doesn't, crash with MPI 'unexpected buffer error'
- Temporary measures in place that blocked bands together

# How to improve?

- CDG requested that backwards compatibility with existing checkpoint files was kept, including post-processing tools
- So MPI-IO is not an option
- Our approach: use MPI collectives instead of point-to-point communications

# Wave_write

- Use MPI collective over g-vectors to gather each band on gv-masters
- Pass each band to its band-master
- Band-masters pass data to root to write out
- Data is written “as is”, with grid data

# Wave_write timings

| No. processing elements | Version 5.5 write | Collectives write |
|---|---|---|
| 24 | 7.09 | 7.63 |
| 48 | 7.23 | 7.72 |
| 96 | 9.85 | 7.88 |
| 192 | 19.49 | 8.13 |
| 384 | 70.48 | 8.42 |

Table I: Benchmark times (in seconds) for wave_write.

# Wave_read

- Not just a simple reverse of wave_write, also need to cater for changes in parallel distribution and Γ to all-k-point conversion
- Approach:
  - Read in grid data
  - Read in (block of) bands
  - Distribute for correct k-point and band to gv-masters
  - Re-order data for current g-vector distribution
  - gv-masters scatter the data

# Wave_read timings

| No. processing elements | Version 5.5 read | Prototype collectives read | Collectives write |
|---|---|---|---|
| 24 | 16.78 | 14.74 | 7.63 |
| 48 | 22.03 | 15.97 | 7.72 |
| 96 | 32.90 | 18.57 | 7.88 |
| 192 | 57.61 | 23.94 | 8.13 |
| 384 | 113.47 | 34.80 | 8.42 |

Table II: Benchmark times (in seconds) for wave_read and, for comparison, wave_write.

# Further wave_read optimisation

- Used CASTEP's trace module to profile code and identified two bottlenecks
  1. Read of each band data
  2. Reordering of g-vector distributed data
- Solutions
  1. Array index order on read of band data
  2. Use a many-one vector subscript to store map between old and new g-vector distribution
     An indirect index is then used to prepare the data for scattering

# Wave_read/write summary

| No. processing elements | Version 5.5 read | Prototype collectives read | Optimised collectives read | Version 5.5 write | Collectives write |
|---|---|---|---|---|---|
| 24 | 16.78 | 14.74 | 9.15 | 7.09 | 7.63 |
| 48 | 22.03 | 15.97 | 9.20 | 7.23 | 7.72 |
| 96 | 32.90 | 18.57 | 9.23 | 9.85 | 7.88 |
| 192 | 57.61 | 23.94 | 9.36 | 19.49 | 8.13 |
| 384 | 113.47 | 34.80 | 9.60 | 70.48 | 8.42 |

Table III: Benchmark times (in seconds) for wave_read and wave_write.

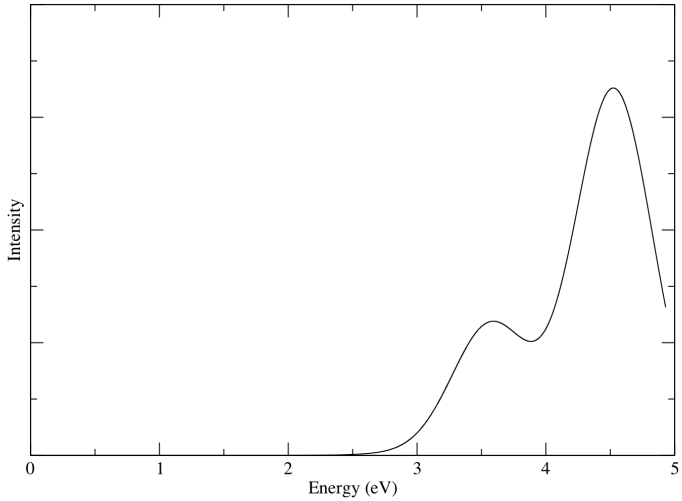
# More of the same…

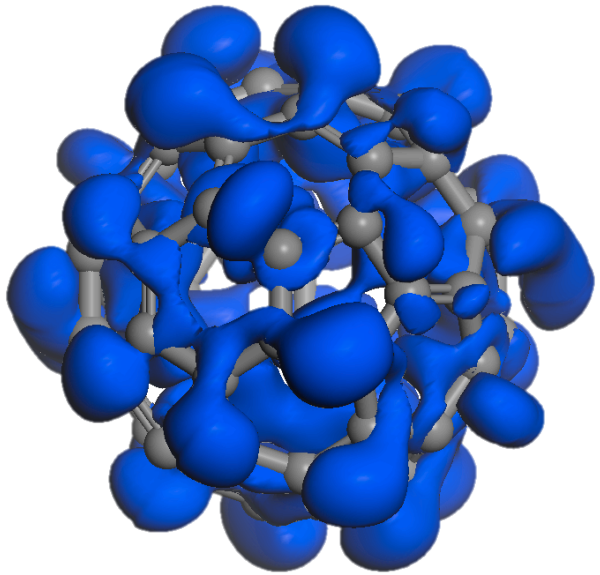
- Apply the same principles to
  - Wave_apply_symmetry – phonon calculations
  - Wave_reassign – variable basis set calculations
  - Density_write/read
  - Pot_write/read

# Part 1 Summary

Optimisations allow

- Larger phonon and TDDFT calculations
- Restart of band parallel runs



Available in current 5.5.2 release version

Science & Technology Facilities Council

# CASTEP error reporting

- Creates empty <seed>.nnnn.err files
- On error, all processes write to their .err

Problems

- Load on filesystem at start of run
- Clean up of empty files at end of run
- Slow ‘ls’ command on some systems
- Redundant information from repeated error messages

# How to improve?

- Open .err files only when an error condition is reached
- Move .err setup from io_initialise to its own routine, io_open_stderr, called from error reporting routines io_abort and io_allocate_abort
- Occasionally extra crash information placed in .err – make io_open_stderr public and check if .err unit is open

# More control

- Still need to address message duplication
- Extra argument to io_abort to allow developer control over which processes report an error
  - ‘A’ – all
  - ‘F’ – farm master
  - ‘R’ – calculation root
  - ‘K’ – k-point masters
  - ‘G’ – g-vector masters
  - ‘B’ – band masters

# Summary

Two successful optimisation projects

- Collectives for data I/O and manipulation
  - ✓ Available in current CASTEP release
- Enhanced error reporting
  - ✓ Included in developer CVS, ready for upcoming version 6.0 release