Introduction To The Elk Code

Kay Dewhurst[1], Sangeeta Sharma[1], Lars Nordström[2],
Francesco Cricchio, Fredrik Bultmark, Hardy Gross[1]

[1]Max-Planck-Institut für Mikrostrukturphysik, Halle
[2]Division of Materials Theory, Uppsala University

What is Elk?

- Elk is an all-electron LAPW+l.o. code
- In development for about 8 years (originally the EXCITING code)
- Simple as possible ($\sim 40k$ lines of Fortran 90)
- Aims to be trustworthy
- Aims to be feature-complete
- Aims to be fully documented
- Released under the GPL
- Website: http://elk.sourceforge.net/

Contributors

JKD, Sangeeta Sharma, Lars Nordström, Francesco Cricchio, Fredrik Bultmark, Hardy Gross, Claudia Ambrosch-Draxl, Clas Persson, Christian Brouder, Rickard Armiento, Andrew Chizmeshya, Per Anderson, Igor Nekrasov, Sushil Auluck, Frank Wagner, Fateh Kalarasse, Jürgen Spitaler, Stefano Pittalis, Nektarios Lathiotakis, Tobias Burnus, Stephan Sagmeister, Christian Meisenbichler, Sébastien Lebègue, Yigang Zhang, Fritz Körmann, Alexey Baranov, Anton Kozhevnikov, Shigeru Suehara, Frank Essenberger, Antonio Sanna, Tyrel McQueen, Tim Baldsiefen, Marty Blaber, Anton Filanovich, Torbjörn Björkman, Martin Stankovski, Jerzy Goraus, Markus Meinert, Daniel Rohr and Vladimir Nazarov

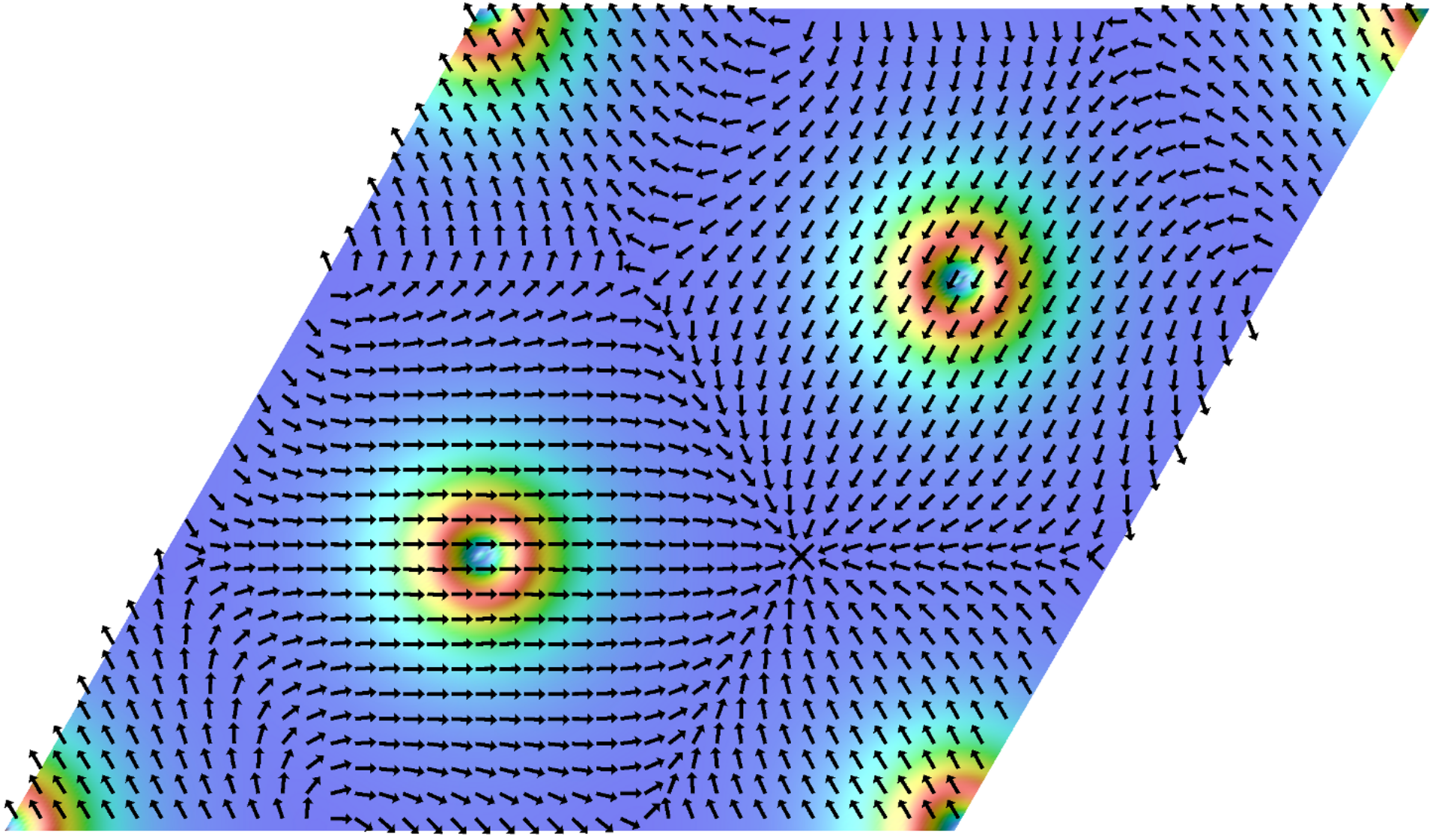
D. J. Singh, *Planewaves, Pseudopotentials and the LAPW Method* (Kluwer Academic Publishers, Boston, 1994)

Features

Basic aim of Elk is to solve the Kohn-Sham equations for external fields

External field	Conjugate field
$V_{\text{ext}}(\mathbf{r})$	$\rho(\mathbf{r})$
$\mathbf{B}_{\text{ext}}(\mathbf{r})$	$\mathbf{m}(\mathbf{r})$
$\mathbf{E}(\mathbf{r})$ $(= -\nabla V(\mathbf{r}))$	$\rho(\mathbf{r})$
$\mathbf{A}$	$\mathbf{j}$

Magnetisation $\mathbf{m}(\mathbf{r})$ of a Cr monolayer



Features

- Kohn-Sham equations solved in two-step process:

  First-variational step: scalar potential and $\mathbf{E}$ only

  $$\hat{H} = \hat{T}_s + \hat{V}_{\text{ext}} + \mathbf{E} \cdot \hat{\mathbf{r}} + \hat{V}_{\text{xc}}$$

  and solve $\hat{H}\phi_i = \epsilon_i \phi_i$

  Second-variational step: add magnetic fields, spin-orbit coupling and $\mathbf{A}$ field

  $$H_{ij} = \varepsilon_i \delta_{ij} + \langle \phi_i | \boldsymbol{\sigma} \cdot (\hat{\mathbf{B}}_{\text{ext}} + \hat{\mathbf{B}}_{\text{xc}}) + \boldsymbol{\sigma} \cdot \hat{L} + \mathbf{A} \cdot \hat{\nabla} | \phi_j \rangle$$

Features

- Full spinor wavefunctions:

$$\phi_{i\mathbf{k}}(\mathbf{r}) = \begin{pmatrix} U^{\uparrow}_{i\mathbf{k}}(\mathbf{r}) \\ U^{\downarrow}_{i\mathbf{k}}(\mathbf{r}) \end{pmatrix} \times e^{i\mathbf{k}\cdot\mathbf{r}}$$

- Spin-spiral states also available:

$$\phi_{i\mathbf{k}}(\mathbf{r}) = \begin{pmatrix} U^{\uparrow}_{i\mathbf{k}}(\mathbf{r}) e^{i(\mathbf{k}+\mathbf{q}/2)\cdot\mathbf{r}} \\ U^{\downarrow}_{i\mathbf{k}}(\mathbf{r}) e^{i(\mathbf{k}-\mathbf{q}/2)\cdot\mathbf{r}} \end{pmatrix}$$

(Frank Essenberger's talk)

Features

- APW radial derivative matching to arbitrary orders at muffin-tin surface
- Arbitrary number of local-orbitals allowed
- LSDA and GGA functions available
- Interface to `libxc`, $>$ 100 functionals (Miguel Marques)
- Fully non-collinear implementation using Kübler's 'trick'
- fixed spin moment or direction
- spin-orbit coupling
- LDA+U available (Lars Nördstrom, Francesco Chricchio)
- Spin-current tensor decomposition (LN, FC)

Features

- Automatic determination of spacegroup symmetry
- Automatic reduction to primitive cell
- Generation of unit cell with spacegroup and Wyckoff coordinates (with `spacegroup` utility)
- Inversion symmetry exploited for real diagonalisation
- Iterative diagonalisation
- Automatic determination and reduction of k-point set
- Symmetrisation of density, potential and fields

Features

- Forces (works with magnetism, etc)
- Structural optimisation
- Various mixing schemes (adaptive, Pulay, Broyden)
- Total DOS, partial DOS in irreducible representations
- Electron localisation function (ELF)
- Electron localisability indicator (ELI) plots with DGrid (Alexey Baranov)
- Plots of wavefunctions, band structures, potentials, etc.
- STM plots

Features

- Core states solved with the Dirac equation
- c is a global constant in the code
- Polarised cores
- Mössbauer isomer shift, EFG and hyperfine fields
- Effective masses
- Fermi surface plots
- Equation of state fitting (with `eos` utility)

Features

- Momentum matrix elements with s.o. correction
- Macroscopic RPA dielectric function
- Kerr angle and Magneto-Optic Kerr Effect (MOKE) output
- Energy loss near edge structure (ELNES) for arbitrary q-vectors
- Non-linear optical (NLO) second-harmonic generation
- Core-state optical spectra (XPS, XAS)

Features

- Hartree-Fock
- Optimised effective potential (OEP) with exact exchange functional
- Reduced density matrix functional theory (RDMFT) (Sangeeta Sharma, Tim Baldseifen, Florian Eich)
- Microscopic RPA function $\epsilon(\mathbf{G}, \mathbf{G}', \mathbf{q}, w)$ in retarded, advance, time-ordered and Matsubara form
- Bethe-Salpeter equation (BSE) beyond Tamm-Dankoff (Matteo Gatti)
- Linear-response time-dependent density functional theory (TDDFT) (SS)

Features

- Phonons for arbitrary **q**-vectors
- Thermodynamic quantities: zero point, free energy, entropy, heat capacity
- Electron-phonon coupling matrices
- Phonon linewidths
- Eliashberg function $\alpha^2 F(\omega)$
- Electron-phonon coupling constant λ
- McMillan-Allen-Dynes superconducting T_c
- Eliashberg equations (Antonio Sanna)

Features

- Everything is interoperable, so you can run:

  A structural optimisation calculation with $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{E}$ fields, in conjunction with a non-collinear, spin-spiral, fixed spin moment, GGA+U calculation, whilst constraining $\nabla \cdot \mathbf{B}_{\mathrm{xc}} = 0$, and then compute a BSE core-state spectra beyond the Tamm-Dankoff approximation on top of this.

  This interoperability makes for robust code

Limitations

Can't (yet) perform

- Molecular dynamics
- Automatic stress/strain/elastic tensor
- Automatic isobaric/isovolumetric optimisation
- *GW* approximation
- Dynamical mean field theory (DMFT)
- Superconducting density functional theory (SCDFT)
- True non-collinear functional for spin-dynamics (F. Eich)
- TDDFT in real-time for attosecond electron dynamics
- Berry phase
- Meta-GGA and van der Waals functionals
- ...

Requirements

Minimum requirements:

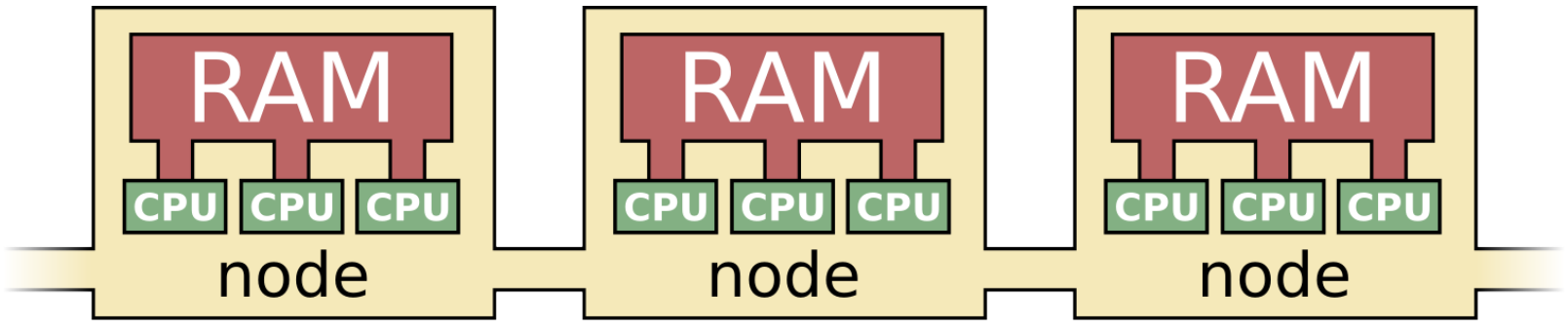
- UNIX/Linux based computer
- Optimising Fortran 90 compiler

Recommended requirements:

- Multi-node computer cluster
- OpenMP + Message Passing Interface (MPI) environment
- Optimised BLAS/LAPACK and FFT libraries

Parallelism in Elk

Parallelism in Elk



Memory bus

Bandwidth: ~35 GB/s

Latency: ~10 ns

Infiniband

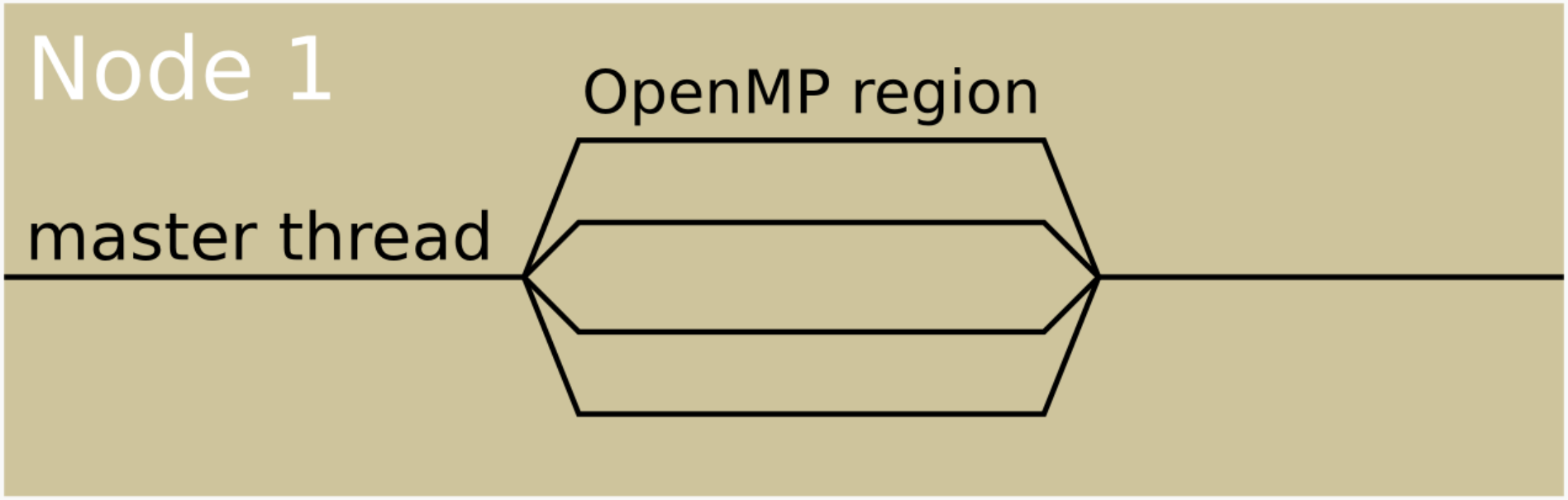
Bandwidth: 1-5 GB/s

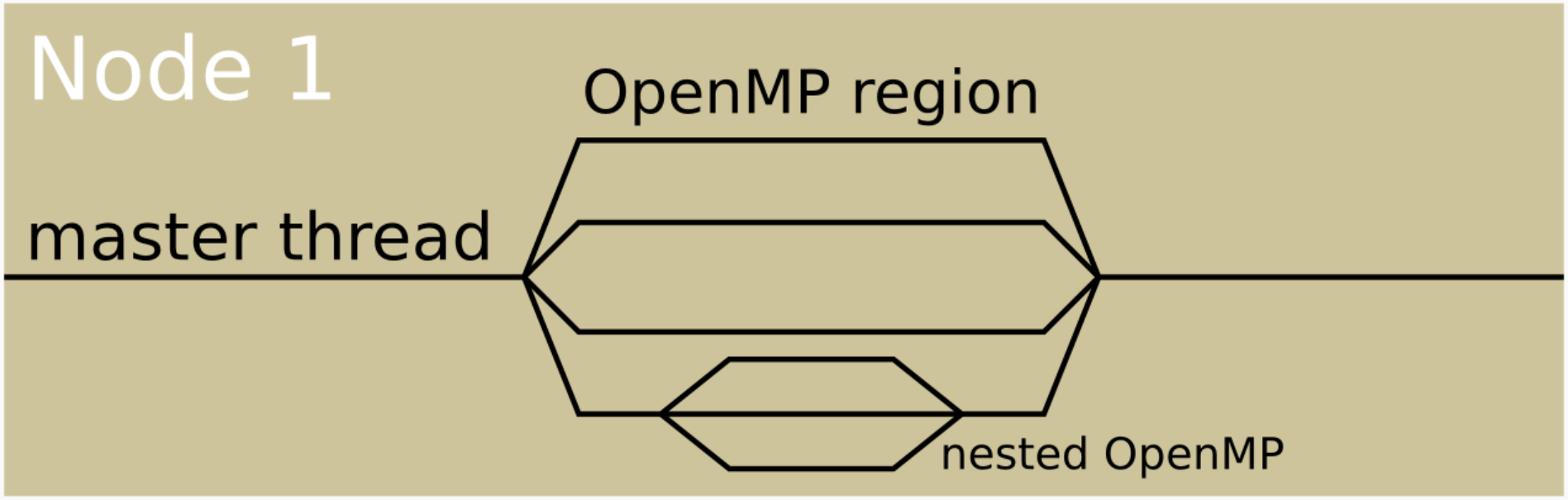
Latency: ~1 µs

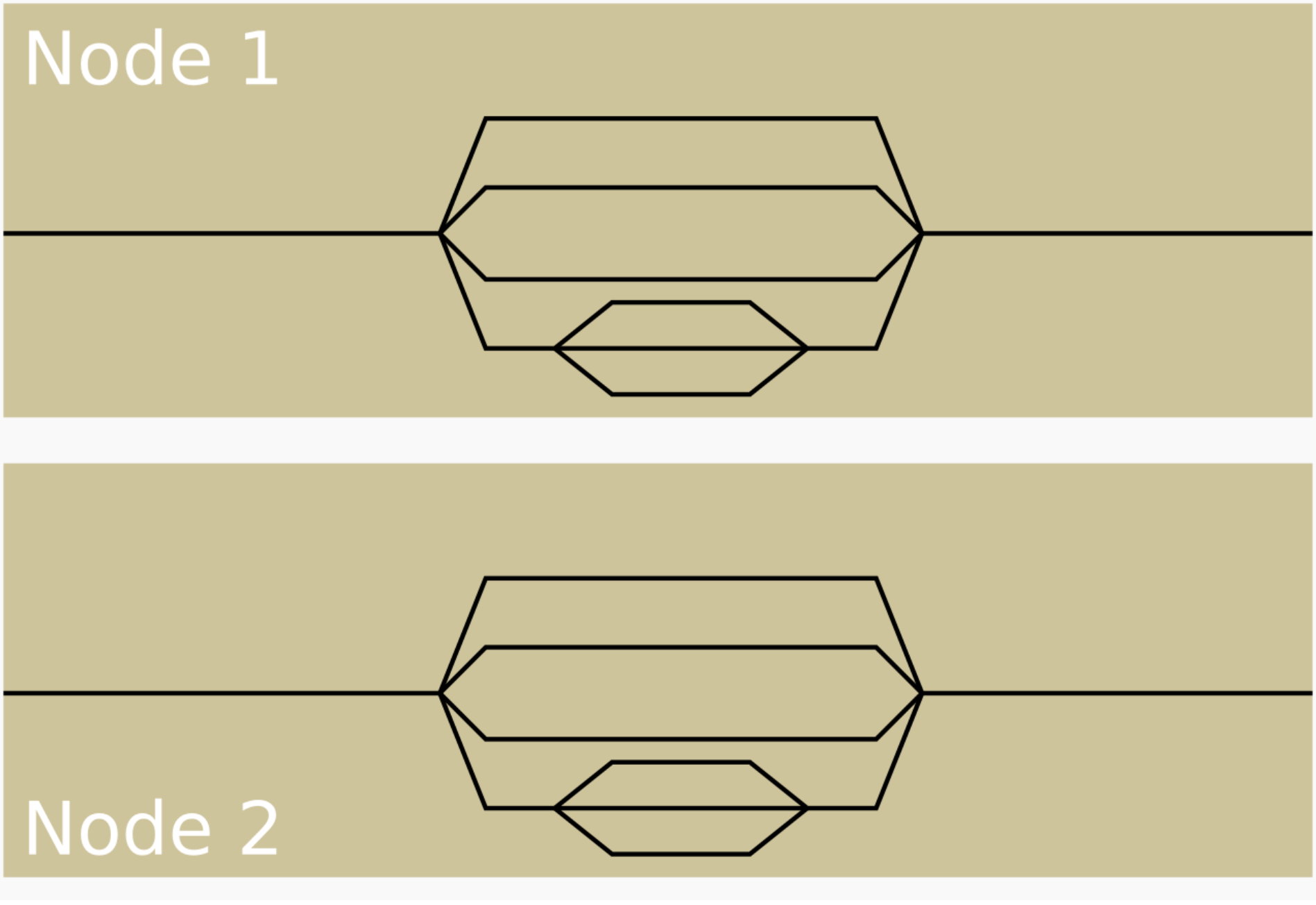
Parallelism in Elk

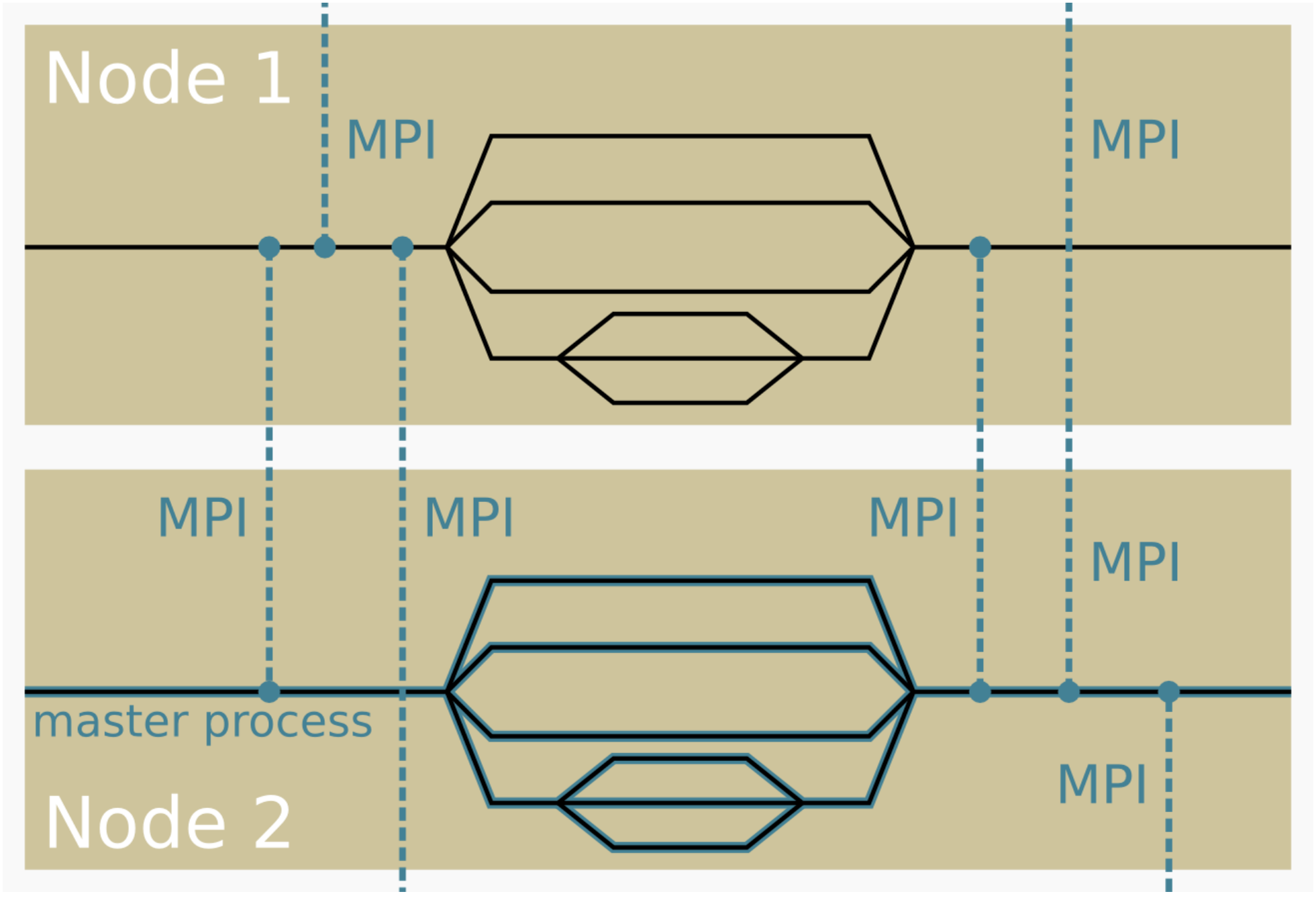
Code parallelised in three ways:

- OpenMP
  (also in MKL/ACML BLAS/LAPACK/FFT libraries)
- Message Passing Interface (MPI)
- Phonon calculations via networked filesystem


Node 1
OpenMP region
master thread


Node 1
OpenMP region
master thread
nested OpenMP


Node 1
Node 2


Node 1
MPI
MPI
MPI
MPI
MPI
MPI
MPI
master process
Node 2

- Parallelised over k-points with MPI and OpenMP
- Parallelised over states, atoms, $\mathbf{G}$-vectors with OpenMP
- Parallelised over q-points, atoms and polarisations for phonon calculations with NFS
- BLAS/LAPACK/FFT parallelised with MKL/ACML

Compiling Elk

1. Download and unpack `elk-1.3.24.tgz`
2. Run `setup`
3. Edit `make.inc` for optimimal performance
4. Run `make`

The executable is `elk` and in the directory `elk/src/`

Running Elk

Normally, only one input file is required: `elk.in`

All output files are in uppercase and have the extension `.OUT`
Some are binary, others are human-readable

General information in the file `INFO.OUT`

Elk.in

```
tasks
  0   :  ground-state run
  10  :  total and partial DOS
  120 :  momentum matrix elements
  180 :  RPA ϵ^{-1}(G, G', q, w)
  185 :  Bethe-Salpeter equation (BSE)

avec
  5.13 5.13 0.00
  5.13 0.00 5.13
  0.00 5.13 5.13

atoms
  1
  'Si.in'
  2
  0.0 0.0 0.0
  0.25 0.25 0.25

ngridk
  5 5 5
```

Species files

Species files are required to tell the code things like muffin-tin radius, type of (L)APW, local-orbitals and which states are in the core

Elements from H to Rf available in the `elk/species/` directory

Running Elk

Full test suite, just run

```
make test
```

Wide range of examples available in `elk/examples/` which demonstrate many features of the code

Also available is the Elk manual and a very active forum on the Elk website

Units

All input and output units are atomic: Units obtained by setting

$$m_e = 1 \qquad e = 1 \qquad \hbar = 1 \qquad 4\pi\epsilon_0 = 1$$

Dervied units:

- Length: 1 Bohr $a_0 = 4\pi\epsilon_0/m_e e^2 \sim 0.52918$ Å
- Energy: 1 Hartree $E_H = m_e e^4/(4\pi\epsilon_0\hbar)^2 \sim 27.211$ eV
- Time: $\hbar/E_h \sim 2.418 \times 10^{-17}$ s
- Magnetic field: $e/a_0^2 \sim 1715.255$ Tesla

Also note $\alpha = e^2/(4\pi\epsilon_0)\hbar c$, thus

$$c = \frac{1}{\alpha} \sim 1/137$$

Globally adjustable in Elk

Beware

It's easy to generate rubbish with electronic structure codes

Always check for convergence with respect to relevant variables:

- `rgkmax`: planewave cut-off for APWs
- `lmaxapw`, `lmaxvr`, `lmaxmat`: angular momentum cut-offs
- `nempty`: number of first-variational states, and therefore second-variational basis size
  (essential for magnetism, optics, ...)
- `ngridk`: k-point grid
- ...

Also compare with other codes: ABINIT, VASP, PWSCF, Wien2k, EXCITING, FLEUR, RSPt, SIESTA, CASTEP, etc.