Elmer on Intel Xeon Phi
Elmer on Intel Xeon Phi

CSC-IT Center for Science Ltd.
Mikko Byckling (mikko.byckling@csc.fi),
Olli-Pekka Lehto (olli-pekka.lehto@csc.fi),
Elmer Team
Contents

- Introduction to Elmer
- Porting Elmer to MIC
- Current status and performance
- Threading legacy code
- Future developments for Elmer
- Conclusions
Elmer: Finite element software for multiphysical problems

- Developed and maintained by CSC
- Used by thousands of researchers worldwide
- Licensed under (L)GPLv2
- Contains a large set of ready-made physical models
- Readily extensible by end user

http://www.csc.fi/elmer
Elmer components

- **Elmer** is a suite of several programs
- Components can be used independently
- **ElmerGUI**: Pre- and Postprocessing
- **ElmerGrid**: structured meshing and mesh import
- **ElmerSolver**: Solution
- **ElmerPost**: Postprocessing
- **Others**: ElmerFront, ElmerParam, MATC, Mesh2D
Elmer on Intel Xeon Phi (MIC)

- CPU: Preprocessing and mesh generation
- CPU/MIC: Solution of the physical problem
- CPU: Postprocessing of the results

Porting effort:
ElmerSolver and associated libraries
Elmer programming languages

- Fortran90 (and newer)
  - ElmerSolver (~210,000 lines, ~50% in DLLs)
- C++
  - ElmerGUI (~18,000 lines)
  - ElmerSolver (~10,000 lines)
- C
  - ElmerPost
  - ElmerGrid (~30,000 lines)
  - MATC (~11,000 lines)
Elmer: Physical Models

- **Heat transfer**
  - Heat equation
  - Radiation with view factors
  - Convection and phase change

- **Fluid mechanics**
  - Navier-Stokes (2D & 3D)
  - RANS: SST k-ω, k-ε, \(v^2-f\)
  - LES: VMS
  - Thin films: Reynolds (1D & 2D)

- **Structural mechanics**
  - General Elasticity (unisotropic, lin & nonlin)
  - Plate, Shell

- **Acoustics**
  - Helmholtz
  - Linearized time-harmonic N-S

- **Species transport**
  - Generic convection-diffusion equation

- **Electromagnetics**
  - Steady-state and harmonic analysis
  - Whitney element formulation for magnetic fields

- **Mesh movement (Lagrangian)**
  - Extending displacements in free surface problems
  - ALE formulation
  - Mortar finite elements

- **Level set method (Eulerian)**
  - Free surface defined by a function

- **Electrokinetics**
  - Poisson-Boltzmann

- **Thermoelectricity**

- **Quantum mechanics**
  - DFT (Kohn Scham)

- **Particle Tracker**

…
Elmer: Numerical Methods

- **Time-dependency**
  - Static, transient, eigenmode, scanning

- **Discretization**
  - Element families: nodal, edge, face, and p-elements, DG
  - Formulations: Galerkin, stabilization, bubbles

- **Linear system solvers**
  - Direct: Lapack, Umfpack, SuperLU, Mumps, Pardiso
  - Iterative Krylov subspace methods (Internal, Hypre)
  - Preconditioners: ILU, AINV, Multigrid (Internal, Hypre, Trilinos)
  - Multigrid solvers (GMG, AMG) (Internal, Hypre, Trilinos)
  - FETI (with Mumps)

- **Parallelism (MPI / OpenMP)**
  - Mesh multiplication, parallel finite element assembly
  - Linear system solution (Krylov methods, Multigrid)
Elmer: Multiphysics features

Solver is an abstract dynamically loaded object
  – May be developed and compiled using an API to the main library
  – No upper limit to the number of Solvers (currently ~50 available)

Solvers may be active in different domains and meshes
  – Automatic mapping of field values

Solvers may be weakly coupled without any *a priori* defined manner

Tailored methods difficult strongly coupled problems
  – Consistent modification of equations (e.g. artificial compressibility in FSI, pull-in analysis)
  – Monolithic solvers (e.g. Linearized time-harmonic Navier-Stokes)
Porting Elmer to MIC

- Porting work started Q2/12
- Focus to build ElmerSolver on a MIC
- Build process not entirely trivial
  - Initially tricks to fool automake
  - Manual editing of some resulting config-files
- ElmerSolver consistency tests
  - Initially 152 of 215 tests passed successfully
  - After a few hours of work 198 of 215 tests passed successfully
Elmer build process is based on automake

Short term solution (current)
- Trap execve to redirect configure test with ssh
  `LD_PRELOAD=./xmatic.so ./configure`
- Manual editing of some Makefiles

Long term solution(s) (in progress)
- Using `binfmt_misc` from Linux kernel
- Permanently switch to using cmake
Automake with `binfmt_misc`

**Prequisities**
- Passwordless `ssh` access to MIC
- Home directories mounted with `nfs`

**Set up** `micrun -script (ssh wrapper)`

**Add K1OM architecture definition to `binfmt_misc` dictionary to execute native MIC binaries via `micrun`

Any application using automake can be cross-compiled to MIC with this approach
Elmer OpenMP status

ElmerSolver library routines are generally thread safe

Environment variable `OMP_NUM_THREADS` must be set, the default is to use a single thread

ElmerSolver internal tests run with `OMP_NUM_THREADS>1`
  - 228 of 231 tests pass successfully
  - Test failures are due to lack of tolerances
With **OMP_NUM_THREADS** undefined

> unset OMP_NUM_THREADS
> mpirun -np 2 ElmerSolver_mpi

Elmer SOLVER (v 7.0) STARTED AT: 2013/04/02 15:46:43
Elmer SOLVER (v 7.0) STARTED AT: 2013/04/02 15:46:43
ParCommInit: Initialize #PEs: 2

**WARNING:: MAIN:** OMP_NUM_THREADS not set. Using only 1 thread.

**WARNING:: MAIN:** OMP_NUM_THREADS not set. Using only 1 thread.

MAIN: ElmerSolver finite element software, Welcome!
MAIN: This program is free software licensed under (L)GPL
MAIN: Copyright 1st April 1995 – , CSC – IT Center for Science Ltd.
MAIN: Webpage http://www.csc.fi/elmer, Email elmeradm@csc.fi
MAIN: Library version: 7.0 (Rev: 6103M)
MAIN: Running in parallel using 2 tasks.
Elmer OpenMP status (cont.)

With `OMP_NUM_THREADS=4`

> export OMP_NUM_THREADS=4
> mpirun -np 2 ElmerSolver_mpi
ELMER SOLVER (v 7.0) STARTED AT: 2013/04/02 15:57:54
ELMER SOLVER (v 7.0) STARTED AT: 2013/04/02 15:57:54
ParCommInit: Initialize #PEs: 2
MAIN:
MAIN: ElmerSolver finite element software, Welcome!
MAIN: This program is free software licensed under (L)GPL
MAIN: Copyright 1st April 1995 - , CSC - IT Center for Science Ltd.
MAIN: Webpage http://www.csc.fi/elmer, Email elmeradm@csc.fi
MAIN: Library version: 7.0 (Rev: 6103M)
MAIN: Running in parallel using 2 tasks.
MAIN: Running in parallel with 4 threads per task.
Elmer OpenMP status (cont.)

Internally OpenMP threading supported by
- Solver API routines related to element assembly
- Time integration routines
- Sparse matrix vector products
- Element assembly loop of some solvers (MagnetoDynamics2D, ShallowWaterNS, StatElecSolve, ThermoElectricSolver)

Library support for OpenMP exists in
- External BLAS routines
- External LAPACK routines
- Direct solvers such as Cholmod, SPQR and Pardiso
Finite element assembly

- Up to 20% of the runtime
- Linear workload growth with problem size
- Critical section needed in final step

Pseudocode:

```c
for each Element in Elements in parallel do
    compute basis for Element
    compute local matrix
    glue local matrix to global matrix
end do
```
Finite element assembly

Sandy Bridge E5, parallel scaling and efficiency

- **Speedup**
  - Y-axis: Speedup
  - X-axis: CPU cores

- **Efficiency**
  - Y-axis: Efficiency
  - X-axis: CPU cores
Finite element assembly

Xeon Phi, parallel scaling and efficiency

Graphs showing speedup and efficiency on the y-axis for different numbers of MIC threads on the x-axis.
**Sparse matrix-vector product, y=Ax**

- Up to 80% of the total runtime
- Required by Krylov subspace methods
- Linear system solution is often the most challenging part as the model size increases

**Pseudocode:**

```plaintext
for i from 1 to n in parallel do
    y(i)=0
    for nonzero elements of A(i,:) do
        y(i)=y(i)+A(i,j)*x(j)
    end do
end do
```
SpDGEMv

Sandy Bridge E5, parallel scaling and efficiency
SpDEGEMv

Xeon Phi, parallel scaling and efficiency

![Graph showing speedup and efficiency with increasing MIC threads.]
Threading legacy code

- Single core performance of Xeon Phi is low => be aware of Amdahl’s law
- Perform disruptive changes if necessary
- Use tools
  - Intel Inspector XE / Intel IDB (to find threading bugs)
  - Intel Vtune (to find hotspots)
Future developments for Elmer

- Modify most important solvers to fully support OpenMP
- Modify ElmerSolver kernels to better support SIMD processing
- Expand ElmerSolver kernels to fully support OpenMP
- Experiment with offloading
- Implement parallel preconditioners
Conclusions

ElmerSolver libraries have been ported to Intel Xeon Phi

Porting effort was relatively easy

Performance optimizations are in development

Added benefit: code improvements and optimizations will also benefit CPUs
Elmer on Intel Xeon Phi

Thank you!
Questions / Comments?