Programming Model
INTERoperability ToWards Exascale

http://www.intertwine-project.eu
Interoperability between programming models for scalable performance on extreme-scale supercomputers

- **Co-design methodology**
  - Define interoperability requirements, implement and evaluate, drive new requirements
  - Work with real applications

- **Computational Resource Management**
  - Coordinated resource sharing for interoperability between runtime systems, libraries

- **Distributed Data Management**
  - Scalable, transparent data sharing on heterogeneous, distributed memory hierarchies

- **Engagement with HPC community**
  - Standards bodies: OpenMP, MPI, GASPI
  - Courses, workshops and Best Practice Guides

Follow INTERTWinE on Twitter: @intertwine_eu

http://www.intertwine-project.eu
Interoperable node-level resource sharing

Computational Resource Sharing

- Multiple codes compete for CPU cores, accelerator devices on cluster nodes
- Application threads
- Numerical libraries threads
- Runtime systems threads
- Communication library threads

- **Interference leads to resource over-subscription or under-subscription on cluster nodes**
- Interoperability?

- **Need coordinated resource sharing:**
  - Ability to express general resource needs
  - Ability to express dynamic resource requirements:
    - computational-heavy periods, idleness periods

→ INTERTWinE Resource Management APIs
Sequential application

PLASMA linear algebra on OpenMP

Intel MKL

CPU USAGE

Interoperable node-level resource sharing

Classical fork-join scheme
Enable runtime systems to dynamically negotiate and adjust resource usage
Maximize resource usage:
\(\rightarrow\) task-based application + task-based libraries on multiple runtime systems

- task-based application on OmpSs
- PLASMA task-based linear algebra on OpenMP
- Chameleon task-based linear algebra on StarPU
Directory/cache: For distributed shared memory

Programmer’s view of memory

0xFFF0000
Node 2
Node 1
Node 0

Array A

0x000000
Node 2
Node 1
Node 0

Array B

Node 0
Array A
Array B

Node 1
Array B
Array A

Node 2
Array B
Array A

Physical view of memory
It is intended to be integrated into the runtimes of task based models (but can be used directly)
- Such as OmpSs, StarPU and PaRSEC
- So its use (and existence!) is transparent to the programmer

Transport layers are provided that implement the underlying data movement
- GASPI, MPI RMA, BeeGFS
- Trivial to switch in and out
Early performance measures

- Block Cholesky matrix factorisation
- 16384 * 16384 elements, block size of 16 * 16
- BLAS for computation, concentrating here on the cost of data movement
- On ARCHER, Cray XC30
INTERTWinE: Programming Model INTERoperability ToWards Exascale

Visit http://www.intertwine-project.eu to find out about our:

• Courses and workshops:
  • Advanced OpenMP: 12-14 December *(London, UK)*

• Best Practice Guides:
  • Writing GASPI-MPI Interoperable Programs
  • MPI + OpenMP Programming
  • MPI + OmpSs Interoperable Programs
  • Open MP/OmpSs/StarPU + Multi-threaded Libraries Interoperable Programs

• “Developer Hub” of resources for developers & application users

...and to sign up for the latest news from INTERTWinE at http://www.intertwine-project.eu/newsletter