Intel® Math Kernel Library (Intel® MKL) Parallel Direct Sparse Solver for Clusters

Alexander Kalinkin
Anton Anders
Roman Anders

Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, reference www.intel.com/software/products.

BunnyPeople, Celeron, Celeron Inside, Centrino, Centrino Atom, Centrino Atom Inside, Centrino Inside, Centrino logo, Cilk, Core Inside, FlashFile, i960, InstantIP, Intel, the Intel logo, Intel386, Intel486, IntelDX2, IntelDX4, IntelSX2, Intel Atom, Intel Atom Inside, Intel Core, Intel Inside, Intel Inside logo, Intel. Leap ahead., Intel. Leap ahead. logo, Intel NetBurst, Intel NetMerge, Intel NetStructure, Intel SingleDriver, Intel SpeedStep, Intel StrataFlash, Intel Viiv, Intel vPro, Intel XScale, Itanium, Itanium Inside, MCS, MMX, Oplus, OverDrive, PDCharm, Pentium, Pentium Inside, skoool, Sound Mark, The Journey Inside, Viiv Inside, vPro Inside, VTune, Xeon, and Xeon Inside are trademarks of Intel Corporation in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

http://intel.com/software/products

Agenda

- Problem statement
- Algorithm description
- Interface description
- Experiments
- Conclusions

Problem statement

$$Ax=b$$

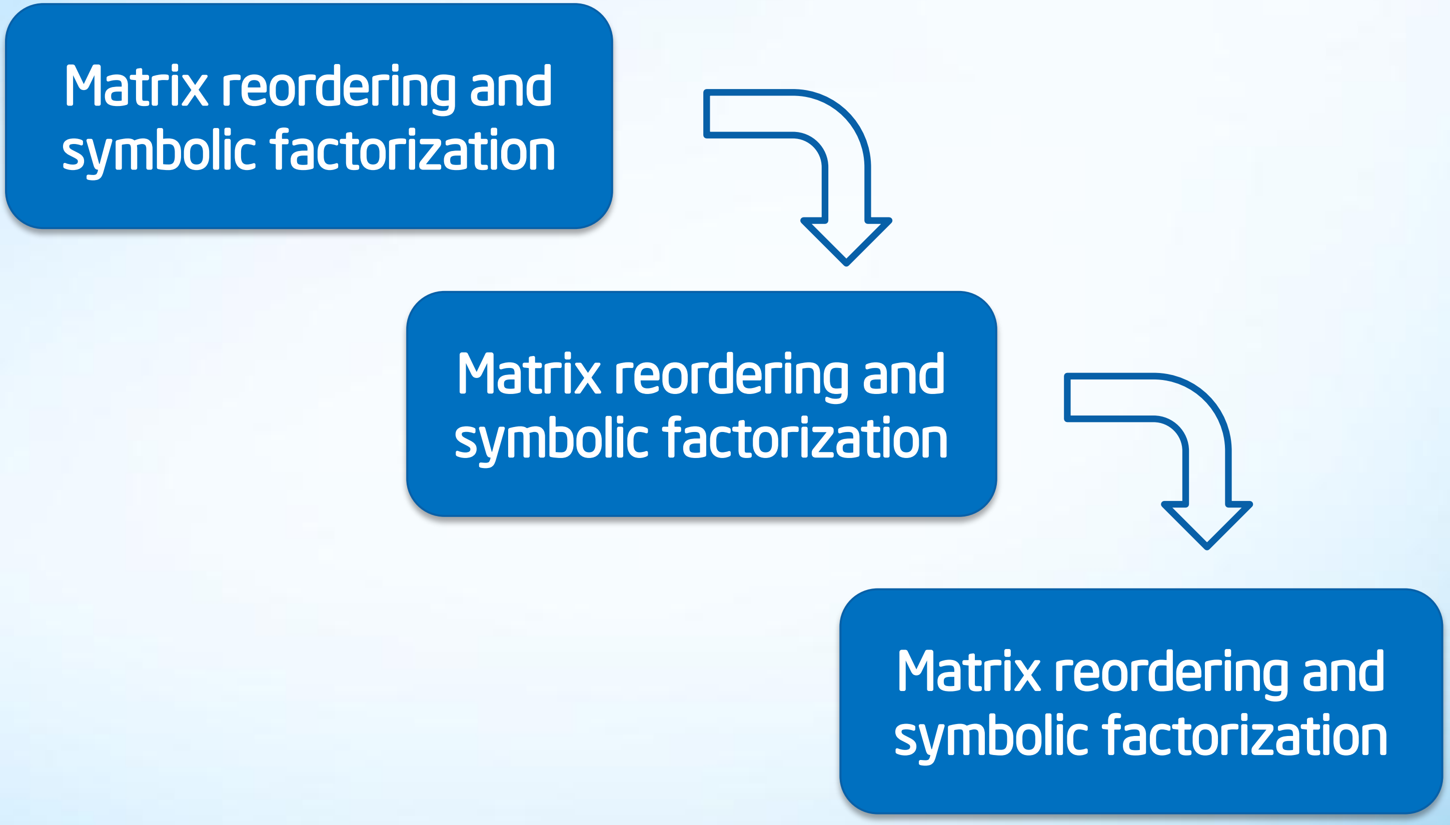
✓ **Cons**

- No extra info available on the matrix, only few generic properties (positive definite, Hermitian,…)
- Huge size

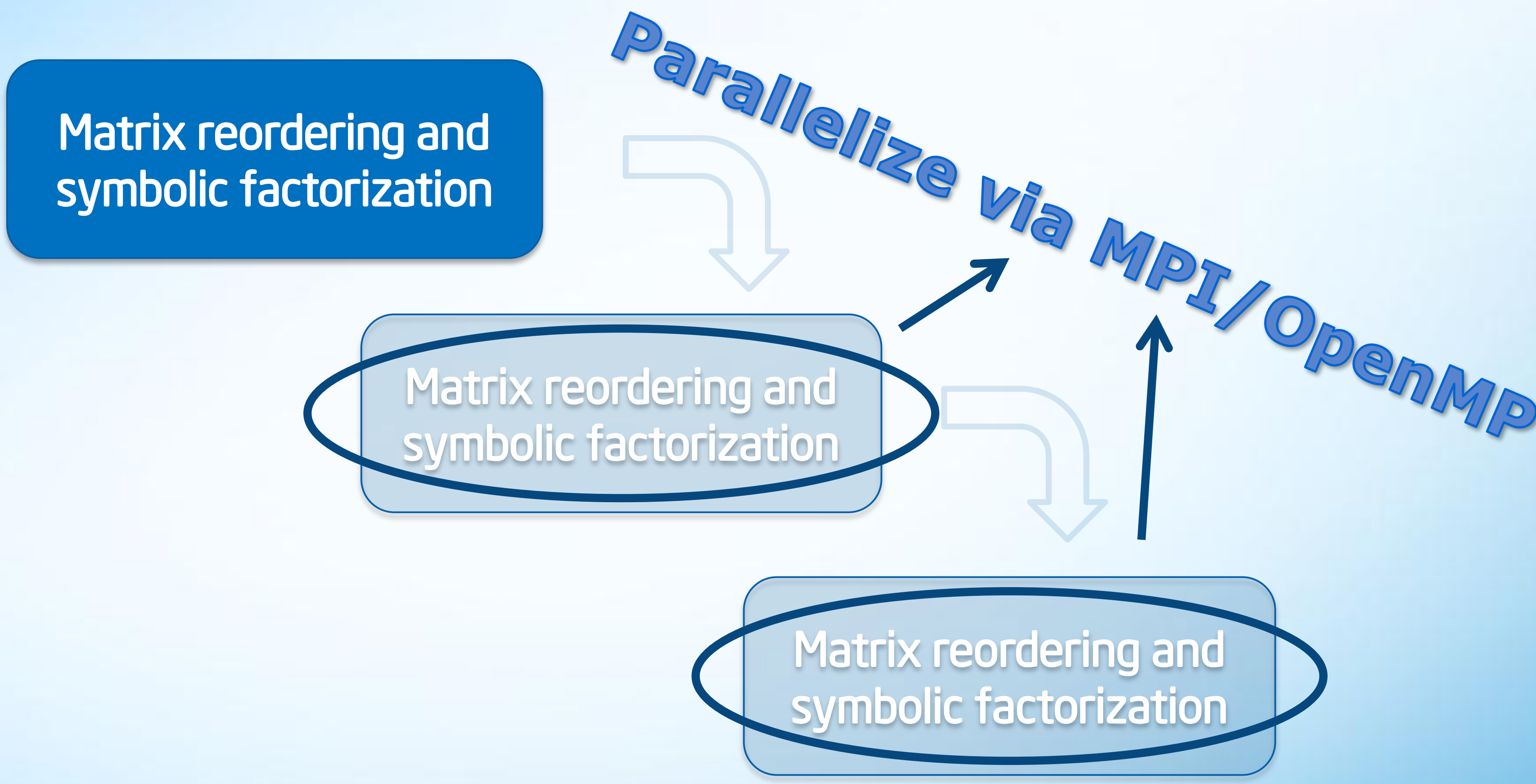
✓ **Pros**

- Clusters with modern CPUs
- Intel® Math Kernel Library (Intel® MKL) with optimized BLAS, LAPACK, PARDISO functionality

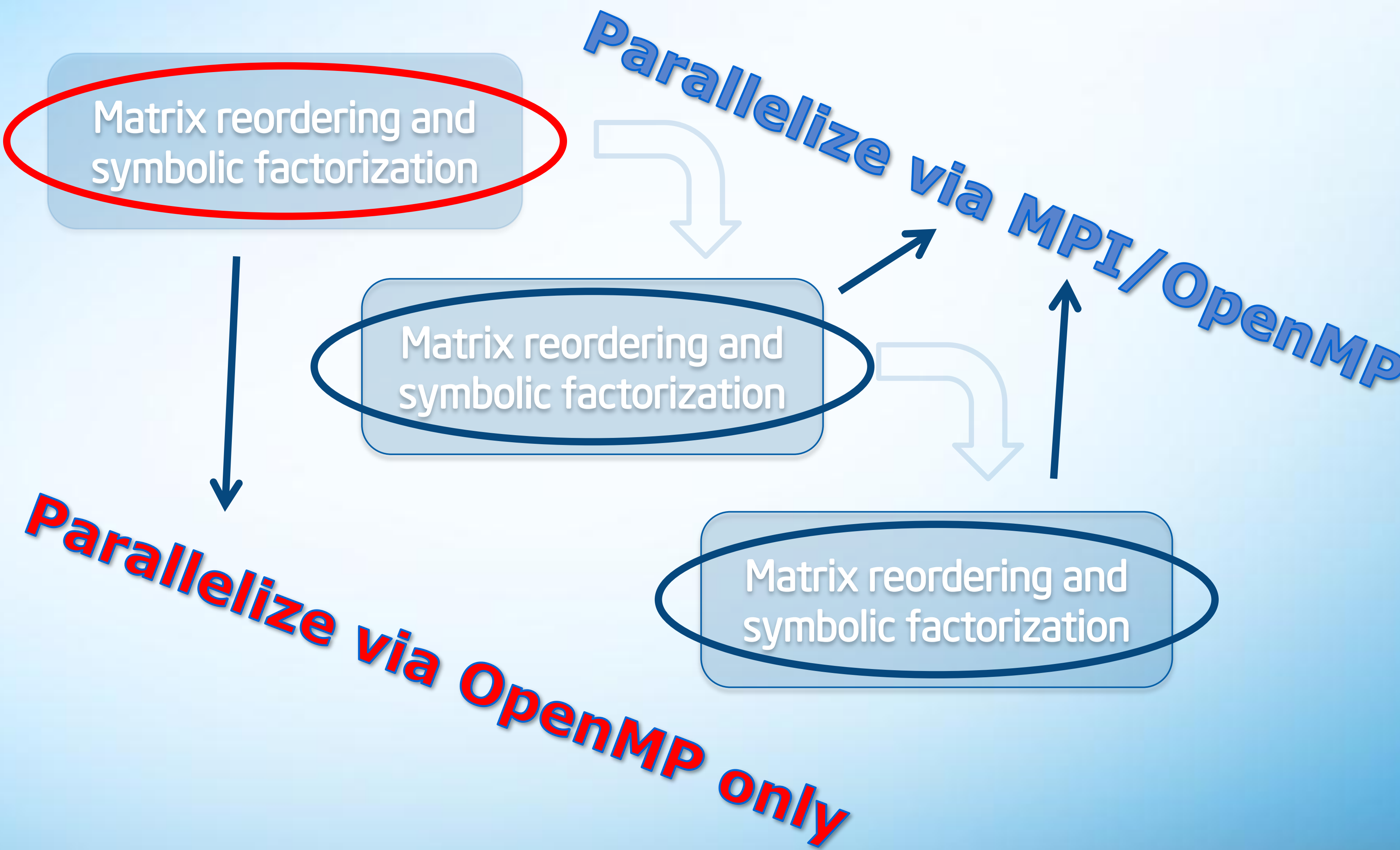
Algorithm (Ax=b)



Algorithm (Ax=b)

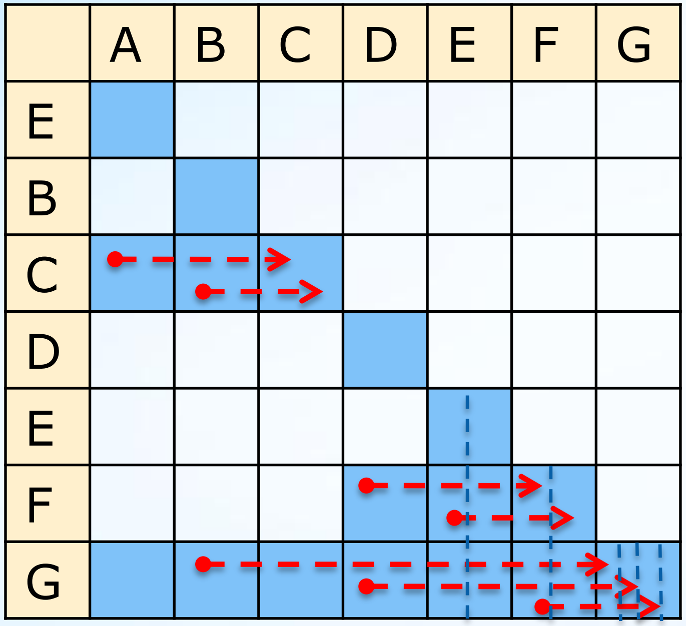


Algorithm (Ax=b)

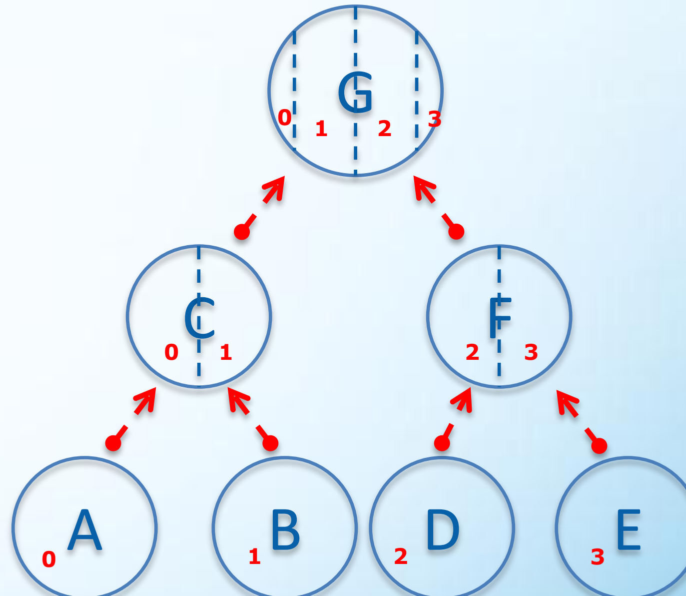


Factorization step

Matrix A after reordering
(example of 4 leafs/processes)



Tree representation of
matrix A after reordering



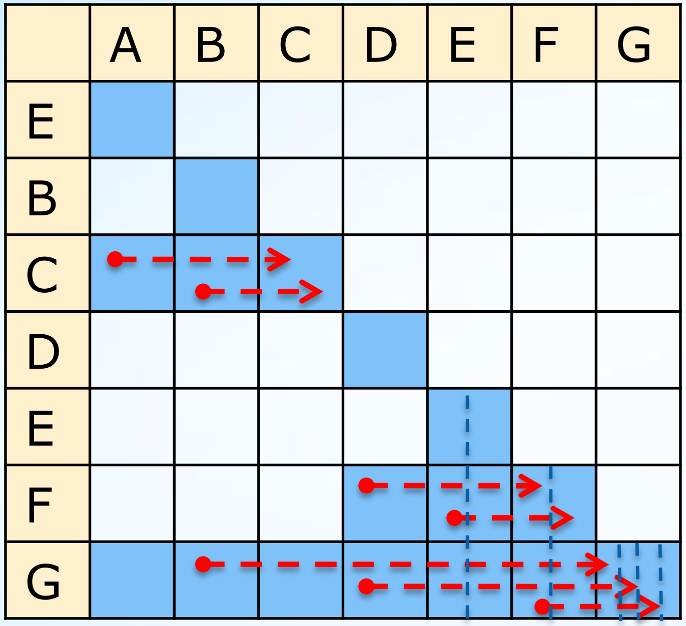
- non-zero block

- L-block updates R-block
(or Right depends on Left)

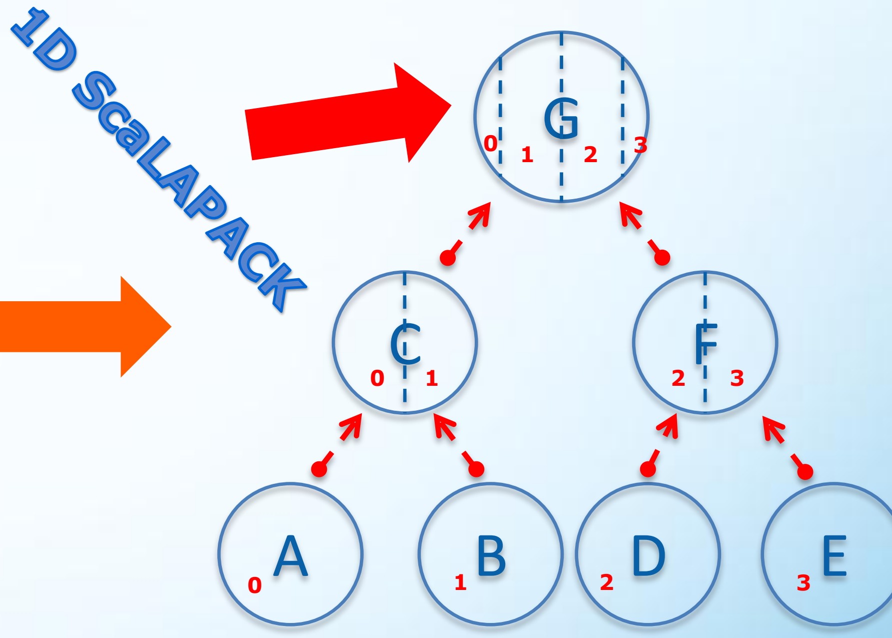
- Both tree and tree-node parallelization are used
- All computations within the node are based on functionality from Intel MKL
- Computation of leafs & updates of a block are independent on each process
- Data is distributed between processes uniformly

Factorization step

Matrix A after reordering
(example of 4 leafs/process)



Tree representation of
matrix A after reordering

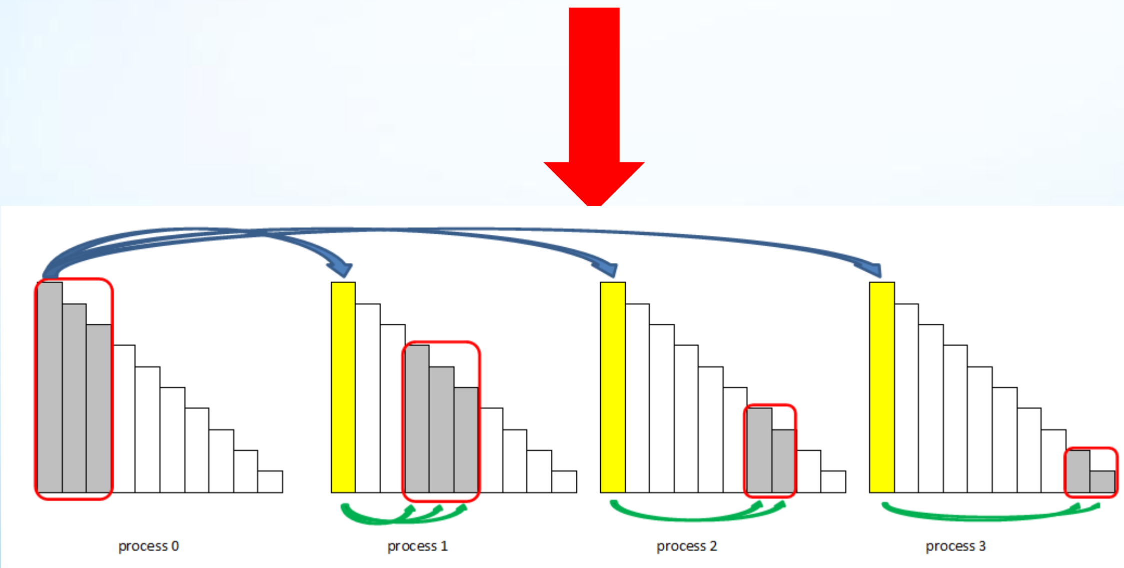


- non-zero block

- L-block updates R-block
(or Right depends on Left)

- Both tree and tree-node parallelization are used
- All computations within the node are based on functionality from Intel MKL
- Computation of leafs & updates of a block are independent on each process
- Data is distributed between processes uniformly

Implementation of LU decomposition within a “node”



Selecting one thread per process allows us to “mask” data transfers behind computations

Current status/interface

Supports various MPI implementations via BLACs in MKL

C:

```
{
….
PARDISO (pt, &maxfct, &mnum, &mtype,
     &phase, &n, a, ia, ja, &idum, &nrhs,
               iparm, &msglvl, b, x, &error);
…
}
```

→

```
{
….
comm =  MPI_Comm_c2f(MPI_COMM_WORLD);
CPARDISO (pt, &maxfct, &mnum, &mtype,
     &phase, &n, a, ia, ja, &idum, &nrhs,
     iparm, &msglvl, b, x, comm, &error);
…
}
```

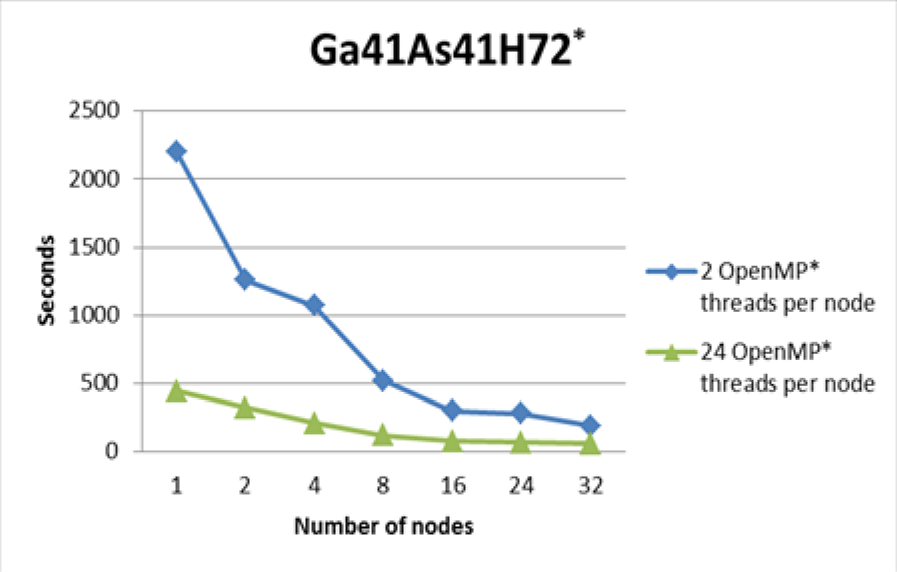
Fortran:

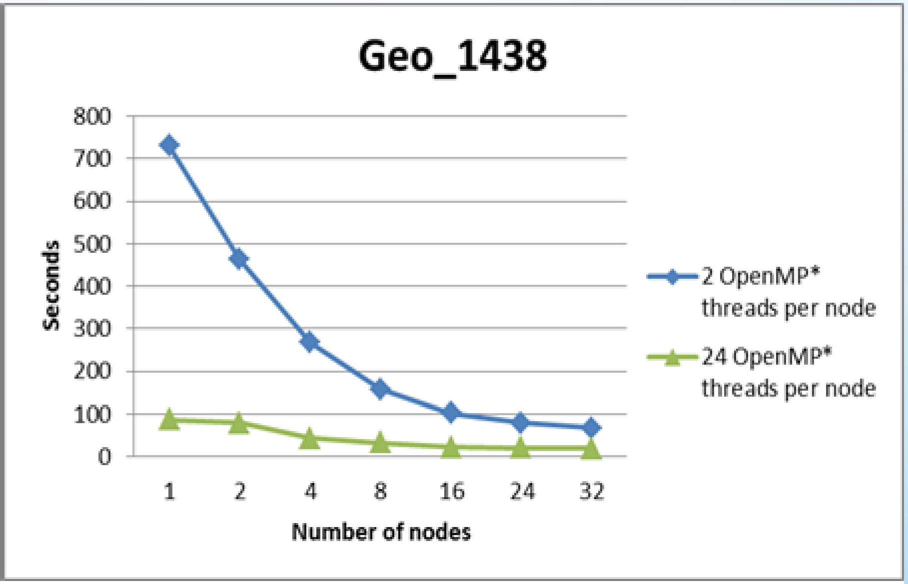
```
….
call PARDISO(pt, maxfct, mnum, mtype,
     phase, n, a, ia, ja, idum, nrhs,
               iparm, msglvl, b, x, error)
…
```

→

```
…
call CPARDISO(pt, maxfct, mnum, mtype,
     phase, n, a, ia, ja, idum, nrhs,
     iparm, msglvl, b, x, comm, &error)
…
```

Scalability of Intel MKL Parallel Direct Sparse Solver for Clusters

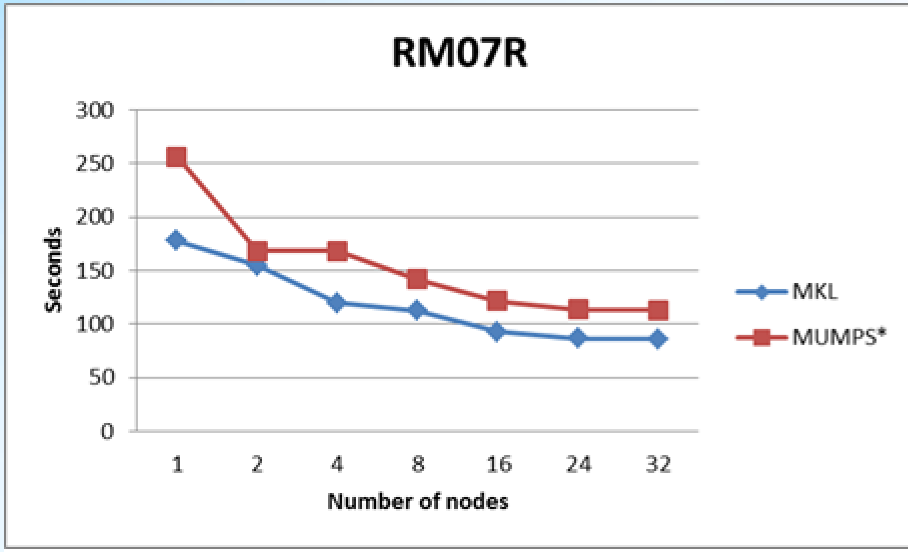


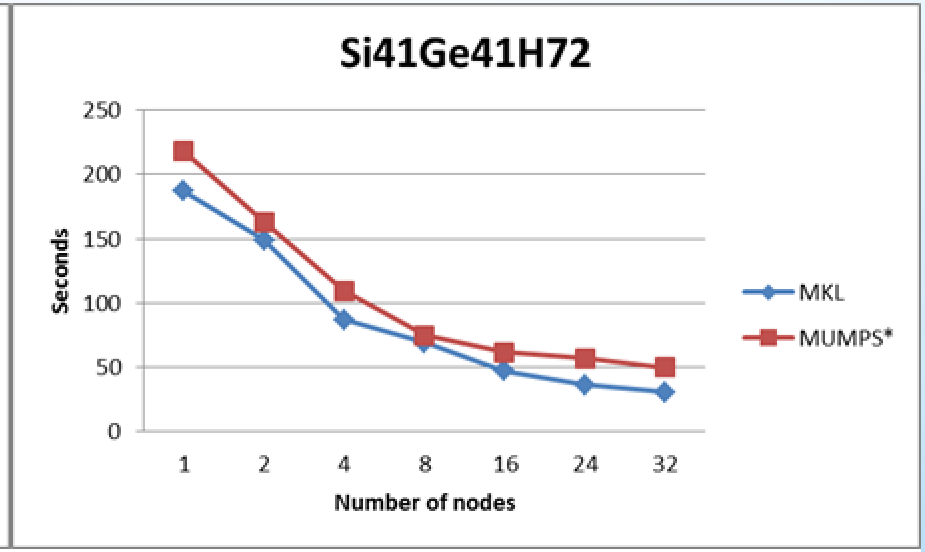


- Each node contains two Intel® Xeon® E5-2697 v2 processors (24 cores in total), 64GB RAM
- Intel® MKL 11.2 Beta

*Here and further: The University of Florida Sparse Matrix Collection T. A. Davis and Y. Hu, ACM Transactions on Mathematical Software, Vol 38, Issue 1, 2011, pp 1:1 - 1:25. http://www.cise.ufl.edu/research/sparse/matrices.

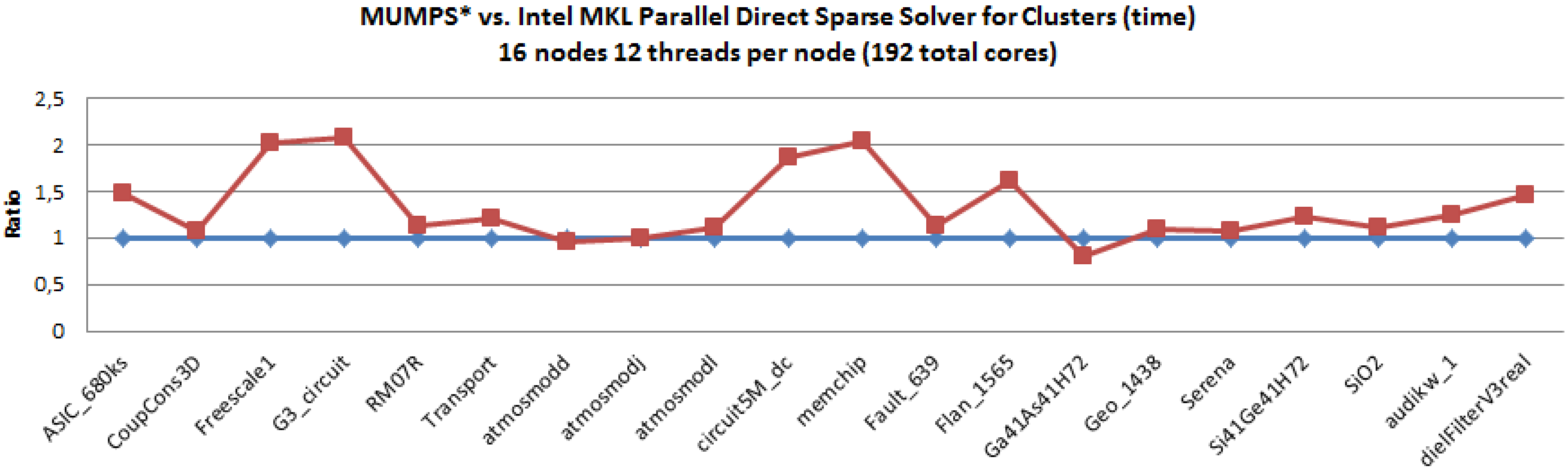
Scalability comparison





- Each node contains two Intel® Xeon® E5-2697 v2 processors (24 cores in total), 64GB RAM
- Intel® MKL 11.2 Beta
- MUMPS* version 4.10.0

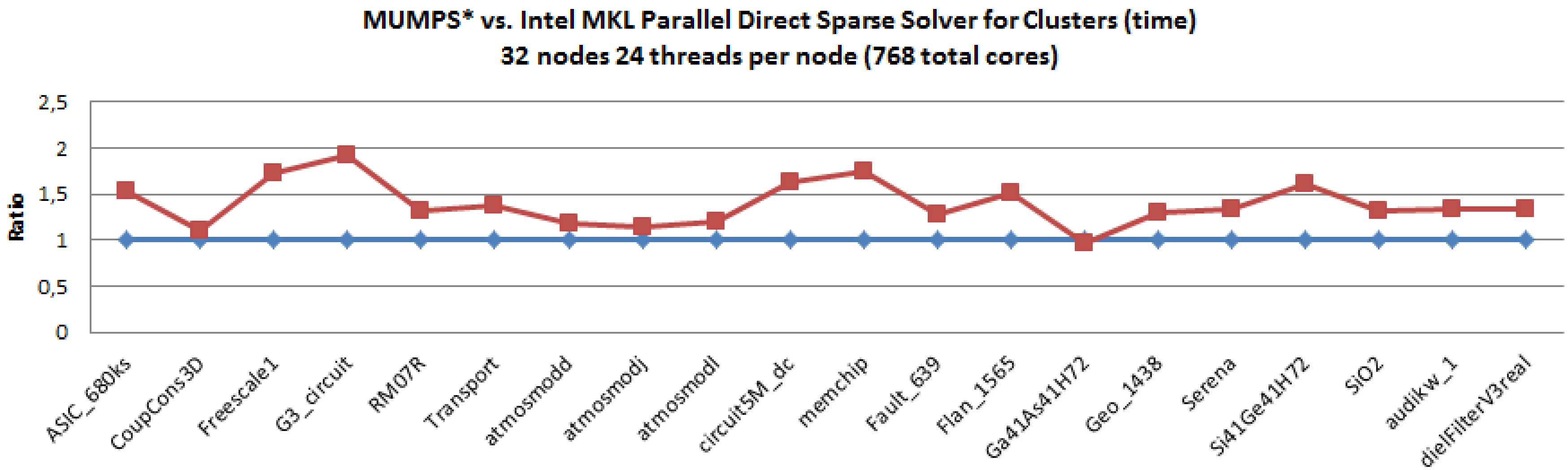
Performance comparison – Intel MKL speedup over MUMPS*



Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, refer to http://www.intel.com/content/www/us/en/benchmarks/resources-benchmark-limitations.html
Refer to our Optimization Notice for more information regarding performance and optimization choices in Intel software products at: http://software.intel.com/en-ru/articles/optimization-notice/
*Other brands and names are the property of their respective owners.

- Each node contains two Intel® Xeon® E5-2697 v2 processors (24 cores in total), 64GB RAM
- Intel® MKL 11.2 Beta
- MUMPS* version 4.10.0

Performance comparison – Intel MKL speedup over MUMPS*



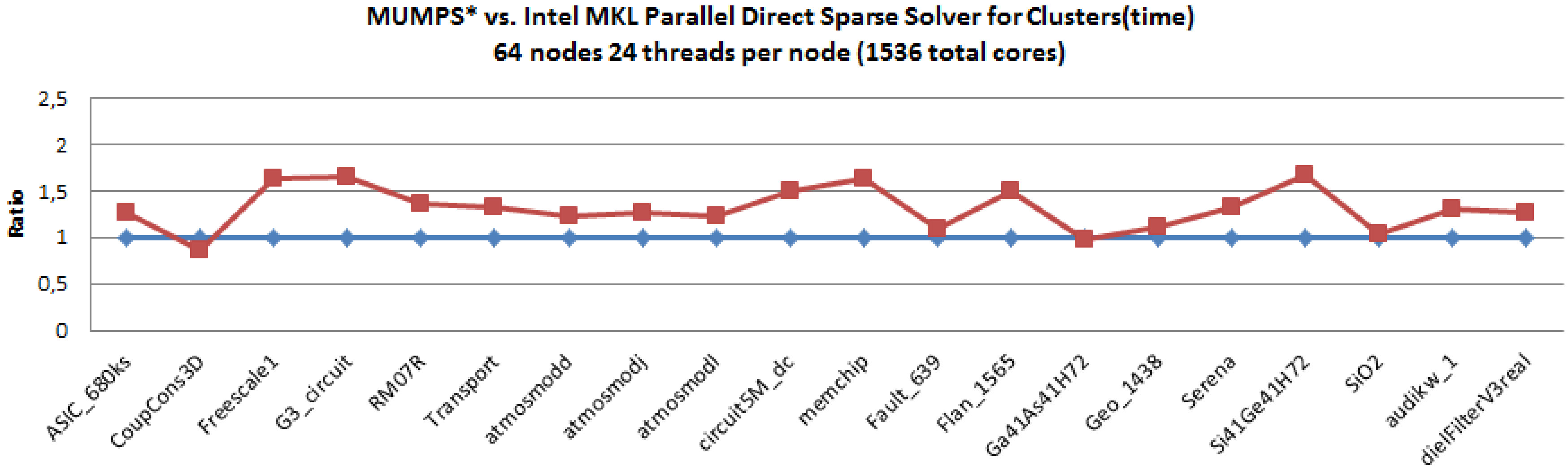
Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, refer to http://www.intel.com/content/www/us/en/benchmarks/resources-benchmark-limitations.html

Refer to our Optimization Notice for more information regarding performance and optimization choices in Intel software products at: http://software.intel.com/en-ru/articles/optimization-notice/

*Other brands and names are the property of their respective owners.

- Each node contains two Intel® Xeon® E5-2697 v2 processors (24 cores in total), 64GB RAM
- Intel® MKL 11.2 Beta
- MUMPS* version 4.10.0

Performance comparison – Intel MKL speedup over MUMPS*

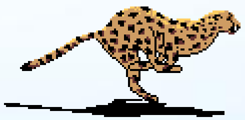


- Each node contains two Intel® Xeon® E5-2697 v2 processors (24 cores in total), 64GB RAM
- Intel® MKL 11.2 Beta
- MUMPS* version 4.10.0

Conclusion

- Intel® Direct Sparse Solver for Clusters included in Intel MKL 11.2 Beta functionality results in
    - Good scaling of computational time
    - Good scaling of memory per node
- On the Roadmap:
    - Implement pure MPI version
    - Parallelize reordering step
    - Implement Intel® Xeon Phi™ version

Q & A



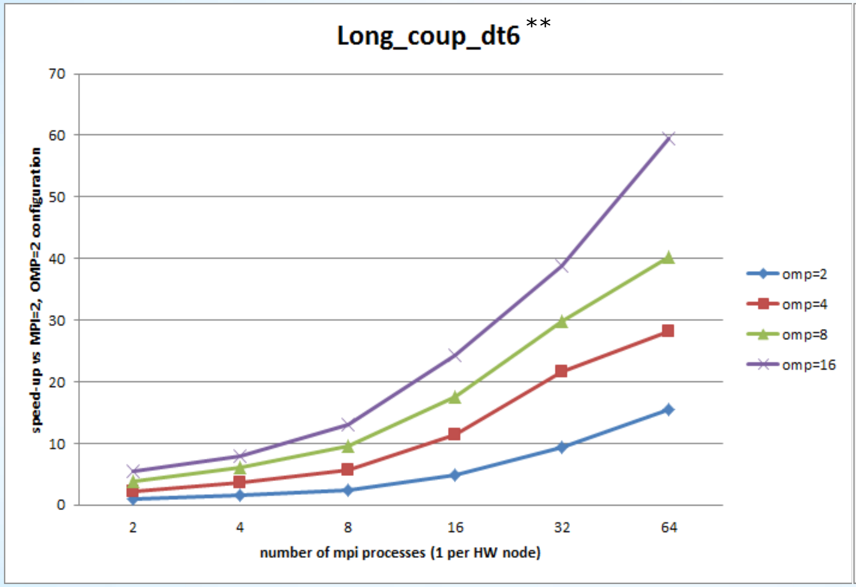
Legal Disclaimer & Optimization Notice

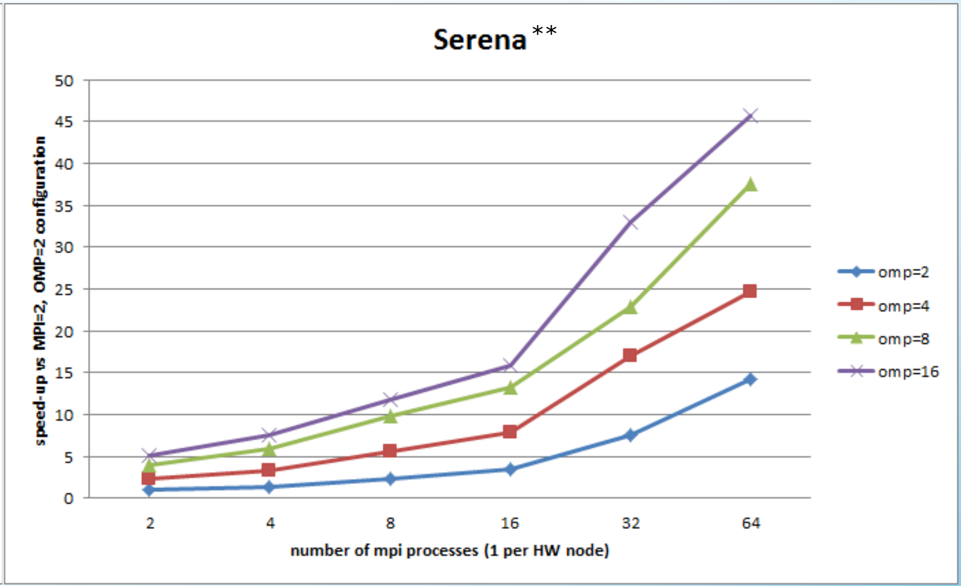
Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

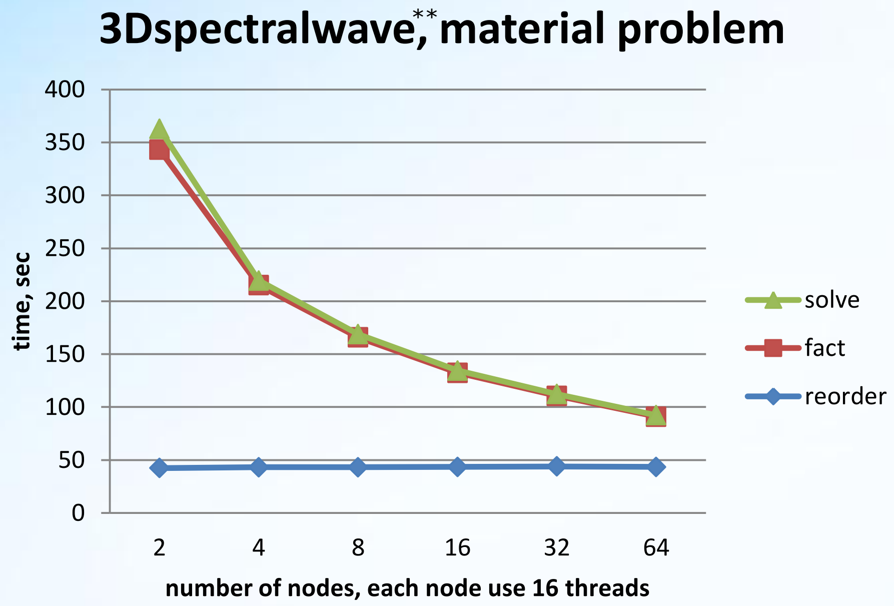
Experiments (scalability in time)

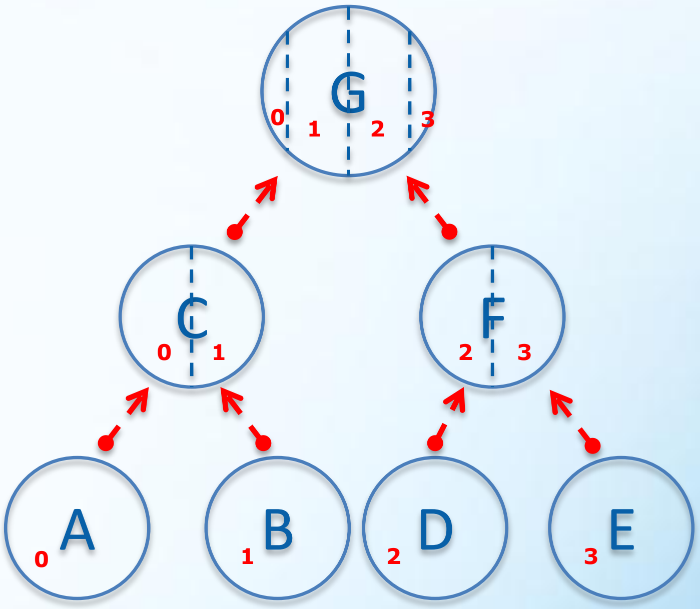




**Here and further: The University of Florida Sparse Matrix Collection T. A. Davis and Y. Hu, ACM Transactions on Mathematical Software, Vol 38, Issue 1, 2011, pp 1:1 - 1:25. http://www.cise.ufl.edu/research/sparse/matrices.

Experiments (scalability in time)

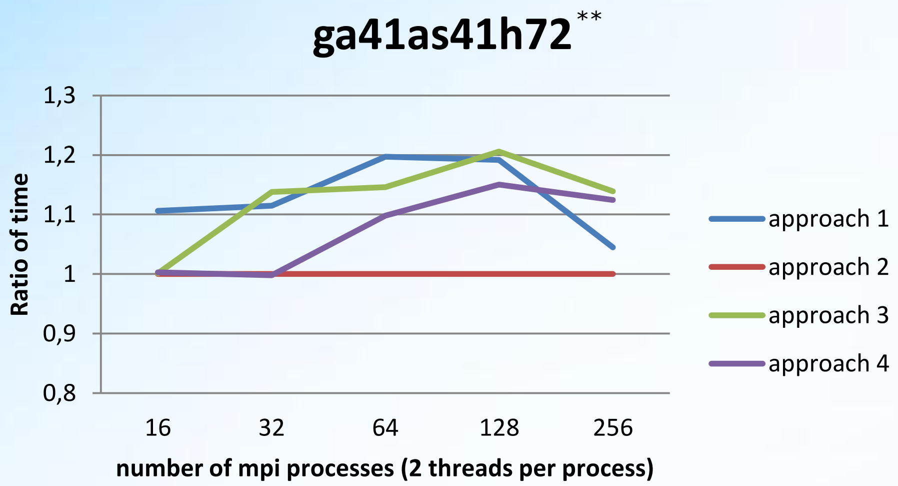


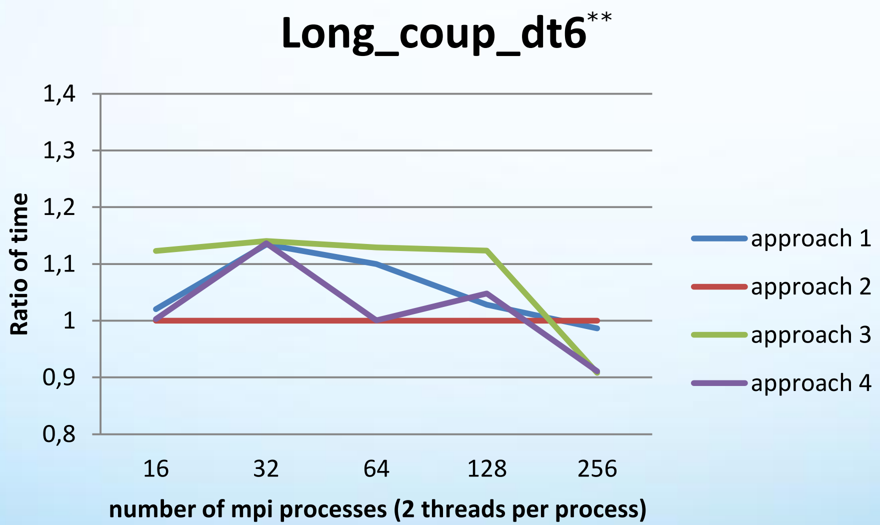


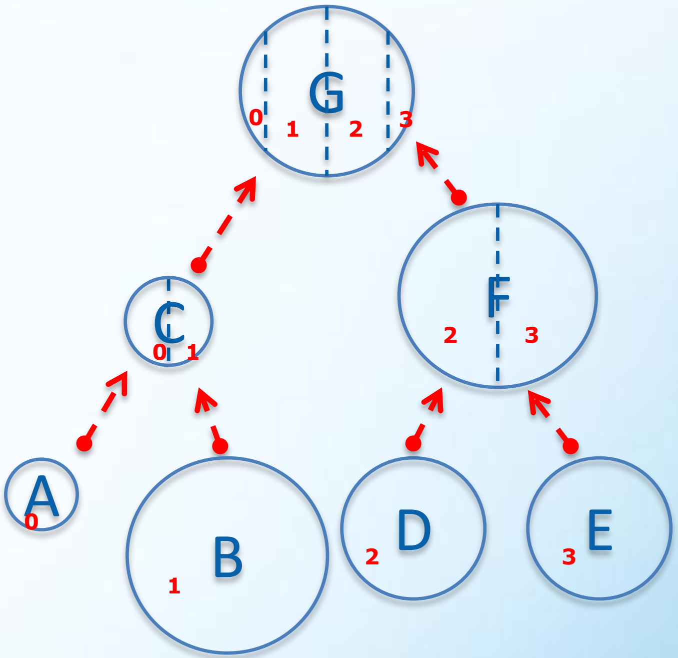
Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, refer to http://www.intel.com/content/www/us/en/benchmarks/resources-benchmark-limitations.html
Refer to our Optimization Notice for more information regarding performance and optimization choices in Intel software products at: http://software.intel.com/en-ru/articles/optimization-notice/
*Other brands and names are the property of their respective owners.

Factorization and solving steps scale well in terms of memory and performance.
Parallelization of reordering step might lead to "worse" reordering affecting overall time… Deeper investigation is needed here.

Experiments (balancing)

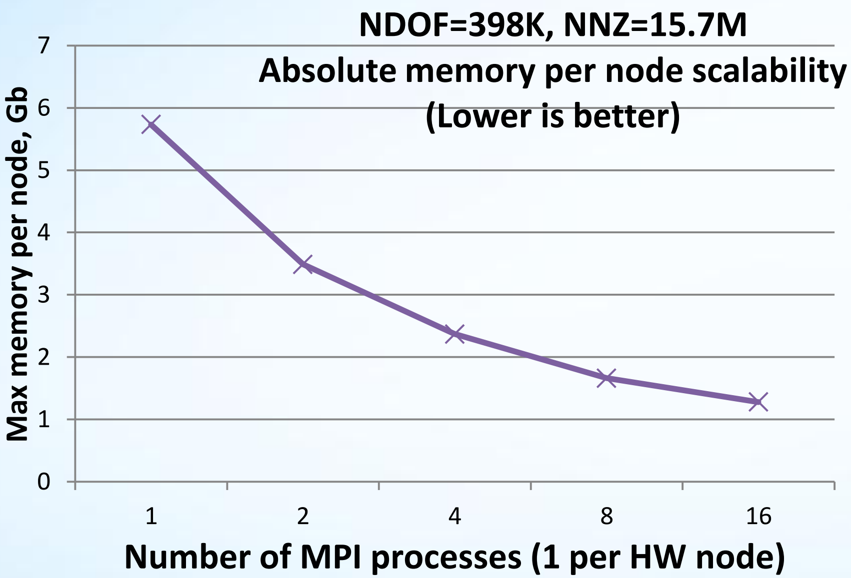


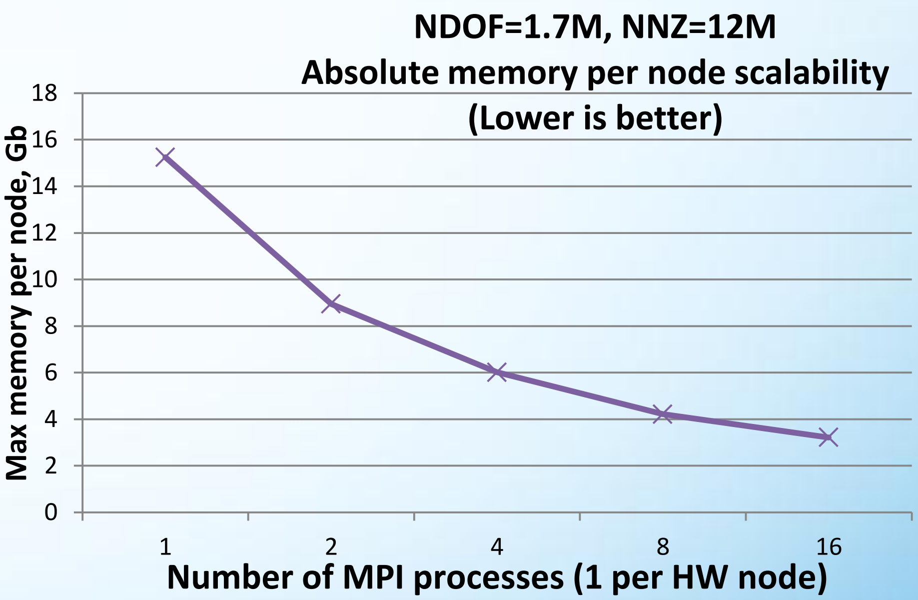




In case of non-uniform "tree", there are a few approaches to divide nodes of the tree between computational nodes. But there is no "best" approach, so to achieve good performance we switch between them at reordering step.

Experiments (scalability on memory)





Experiments (scalability on memory)

Additional processes decrease memory size per host!!!