Eos Compilers

OLCF
OAK RIDGE LEADERSHIP COMPUTING FACILITY

Fernanda Foertter
HPC User Assistance Specialist

Available Compilers

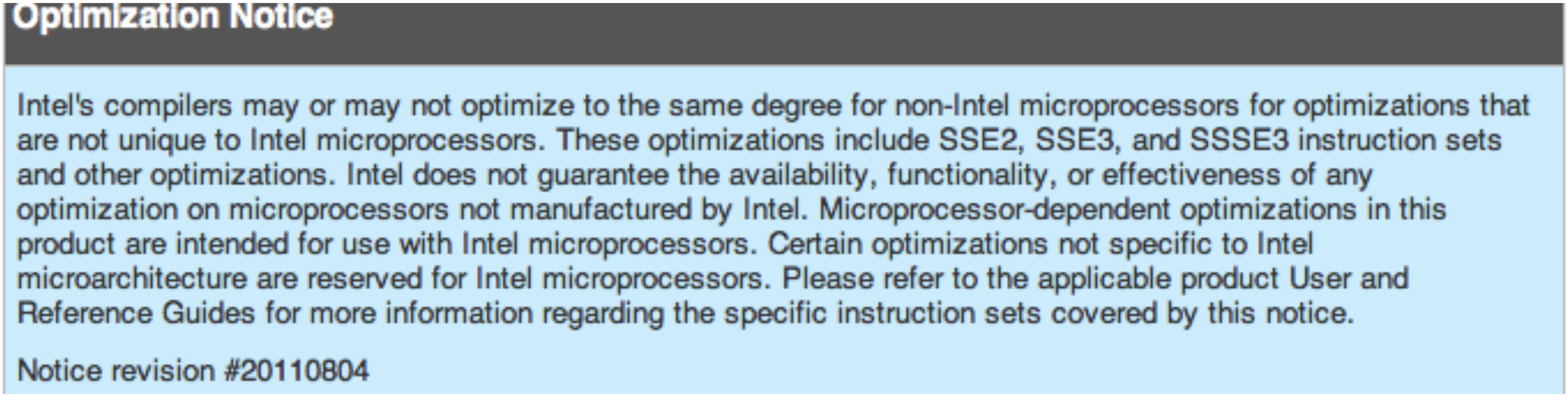
- Cray, Intel, GNU and PGI
- Cray compiler wrappers ***Always use the wrapper!***
  - Same command (ftn/cc/CC) regardless of compiler
  - Links in MPI/Libsci libraries automatically
  - Intel is the default
  - Static linking is default

```
module swap PrgEnv-cray PrgEnv-intel
```

Compiler tips

- Some are more aggressive then others at optimization and standards
- Safe to assume vendor knows its architecture best (e.g. Intel) and thus better at optimizing for their own
- No compiler is superior for all codes
- Try different compilers, good way of catching bugs
- Test your application against lower optimization levels and included tests
- Use system provided libraries

Caution



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

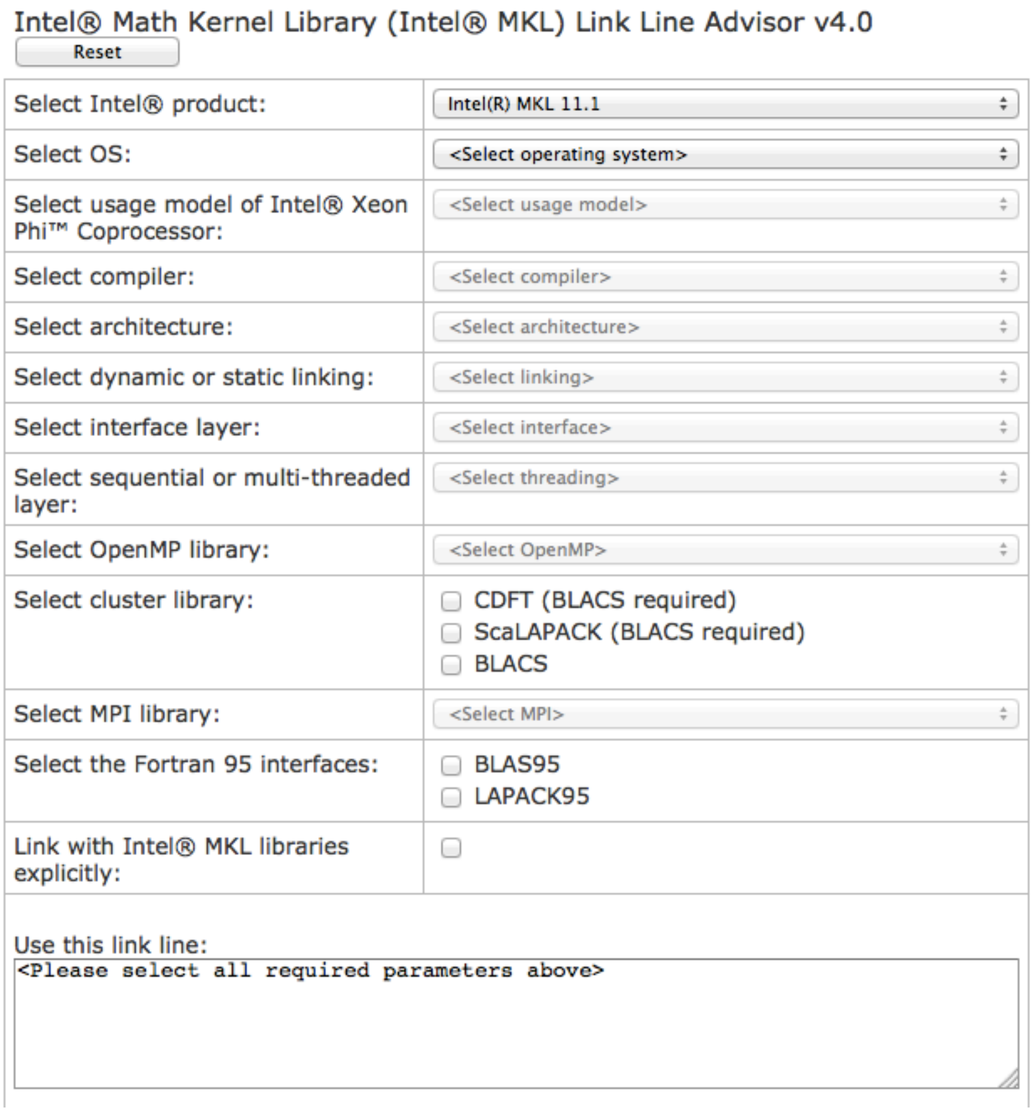
Intel specific tips

Pros	Cons
• Compatibility with Sandybrige • Scalar optimization • Vectorization • Intel MKL • Intel specific OpenMP extensions • AVX (speeds up vector + FP) • OpenMP 4.0 support	• ~~No Cray math libraries available~~

Some Intel Compiler Options

Flag	Description
none	Optimizes for speed; no flags, approx –O2 level
-fast	Maximizes speed across the entire program. Aggressive. Includes -ipo -O3 -no-prec-div -static -xHost
-fast -no-ipo	No IPO
-O0	Disables optimizations
-O1	Optimizes for speed but keeps binary small
-O2	Speed + vectorizaiton, inlining
-O3	O2 + aggressive loop transformations
-mkl=cluster	Single threaded MKL (BLAS/LAPACK/scaLAPACK/FFTW)
-mkl=parallel	Multi-Threaded MKL (No ScaLAPACK)
-xAVX	Default on Eos

MKL link line builder



Intel® Math Kernel Library (Intel® MKL) Link Line Advisor v4.0

Select Intel® product:	Intel(R) MKL 11.1
Select OS:	<Select operating system>
Select usage model of Intel® Xeon Phi™ Coprocessor:	<Select usage model>
Select compiler:	<Select compiler>
Select architecture:	<Select architecture>
Select dynamic or static linking:	<Select linking>
Select interface layer:	<Select interface>
Select sequential or multi-threaded layer:	<Select threading>
Select OpenMP library:	<Select OpenMP>
Select cluster library:	□ CDFT (BLACS required) □ ScaLAPACK (BLACS required) □ BLACS
Select MPI library:	<Select MPI>
Select the Fortran 95 interfaces:	□ BLAS95 □ LAPACK95
Link with Intel® MKL libraries explicitly:	□

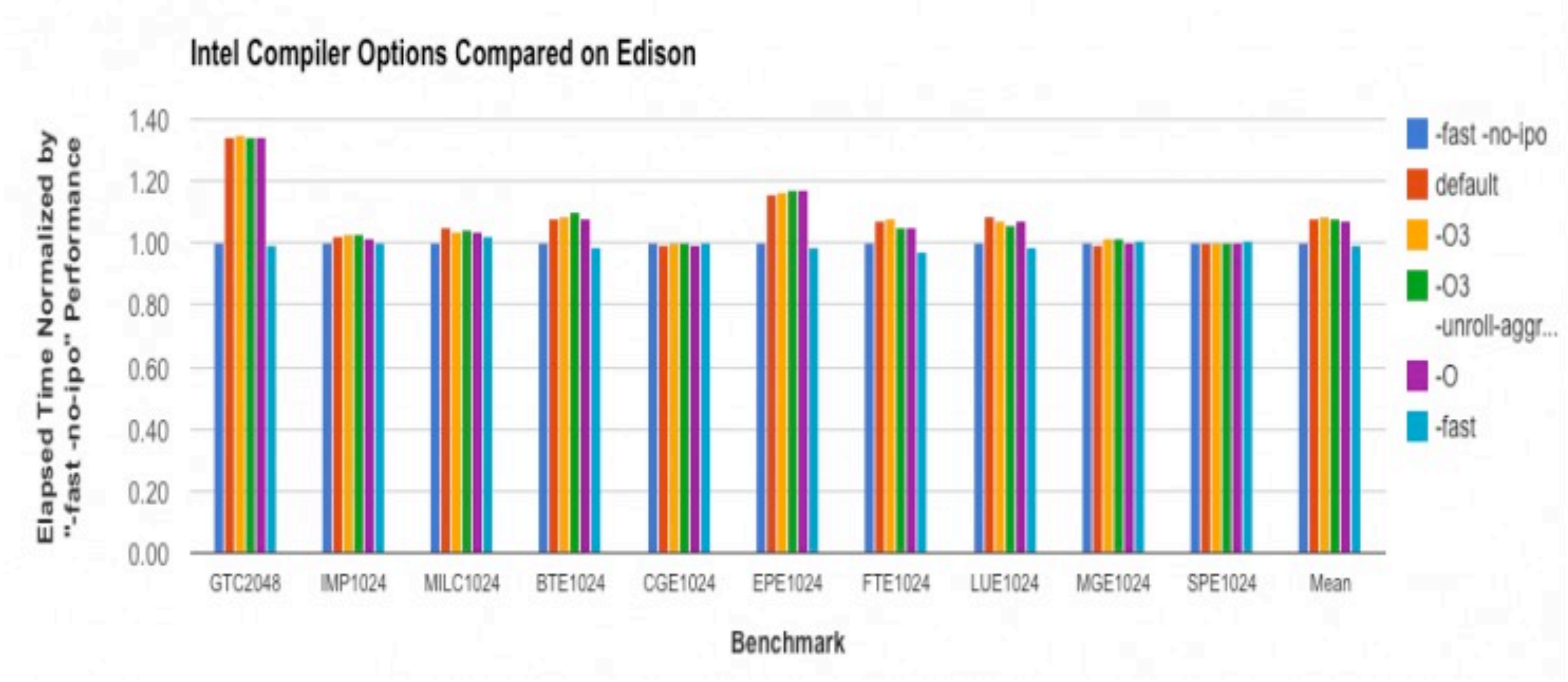
Use this link line:

```
<Please select all required parameters above>
```

Source: http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor

Edison Benchmark Results at NERSC

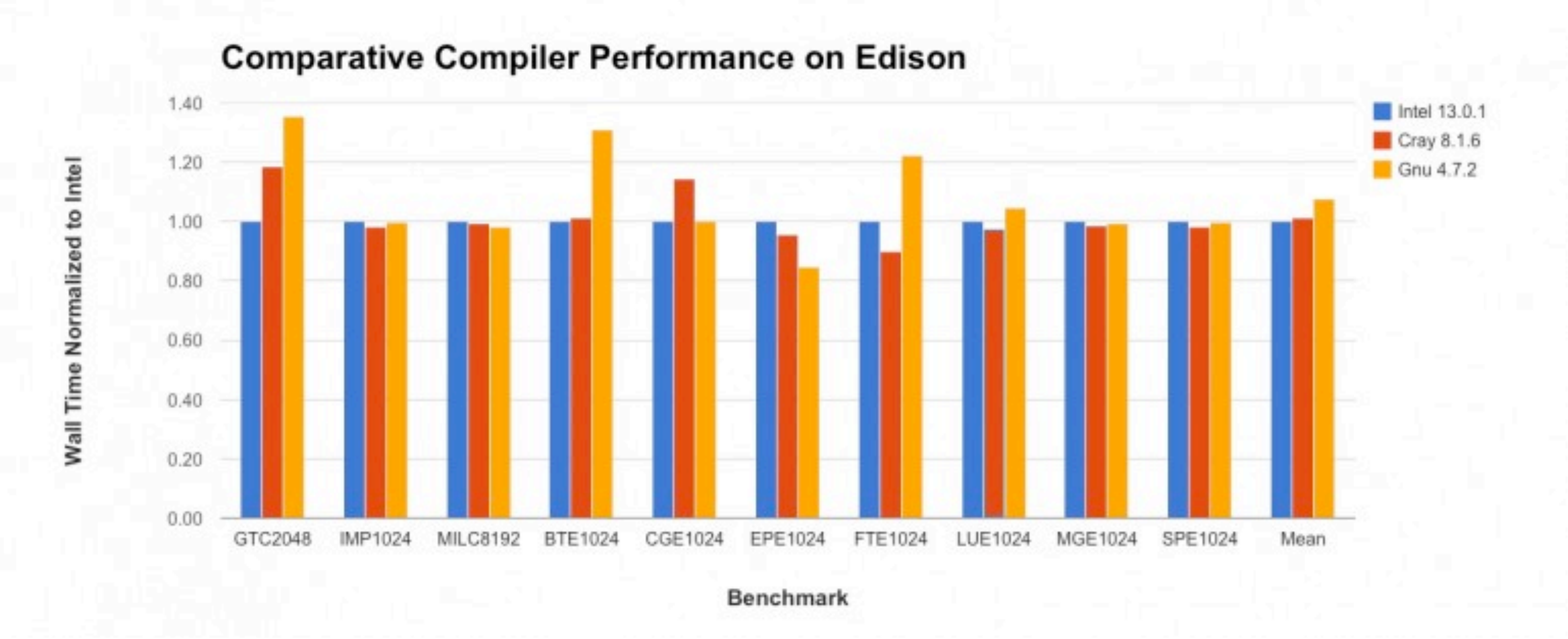
Not a lot of difference (lower is better)



Source: http://www.nersc.gov/users/computational-systems/edison/performance-and-optimization/compiler-comparisons/

Edison Benchmark Results at NERSC

Not a lot of difference (lower is better)



Source: http://www.nersc.gov/users/computational-systems/edison/performance-and-optimization/compiler-comparisons/

Known Issue: Intel OpenMP Hybrid

- The Cray thread affinity settings and Intel's run time OpenMP environment conflict.
- Intel has an extra thread at run time, so 2 threads are scheduled on the same core and the job takes twice as long as it should.

Source: Introducing the Intel Compiler on Edison, Michael Stuart, NUG Feb 2013

Intel OpenMP Hybrid

- CPU-binding should be turned off
  - Allows user compute threads to spread out over available resources
  - Helper thread will no longer impact performance

NERSC's Edison Solution

- OMP_NUM_THREADS <=8
  - set KMP_AFFINITY compact + aprun -cc numa_node
- OMP_NUM_THREADS >8 and <=16
  - set KMP_AFFINITY scatter + aprun -cc none

Source: Introducing the Intel Compiler on Edison, Michael Stuart, NUG Feb 2013

OLCF.ORNL.GOV