Intel® Performance Libraries, the latest Updates, upcoming features

March 2015

Gennady Fedorov

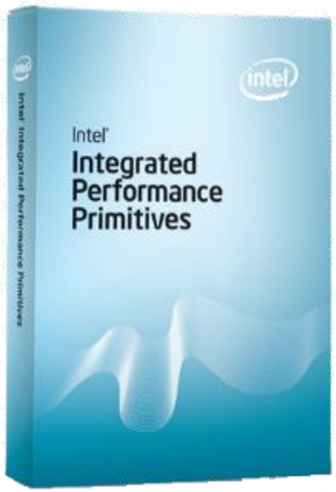
Outline

- **Intel® IPP**
  - Overview
  - IPP 9.0 beta – new features
- Intel® MKL
  - Overview
  - MKL 11.3 beta - new Features
- Intel® DAAL
- References

Intel® Integrated Performance Primitives v.8.2

This is the basic algorithms for:

- Signal and String Processing
- Image Processing
- Computer Vision
- Data Compression
- Cryptography*
- Color Conversion
- Vector Operations (add, pow, sin, inv, dev, erf, rounding…)
- Small Matrix Math (transpose, mul, inverse, LU, Cholesky ….)
- Audio coding,
- String processing and
- many others …

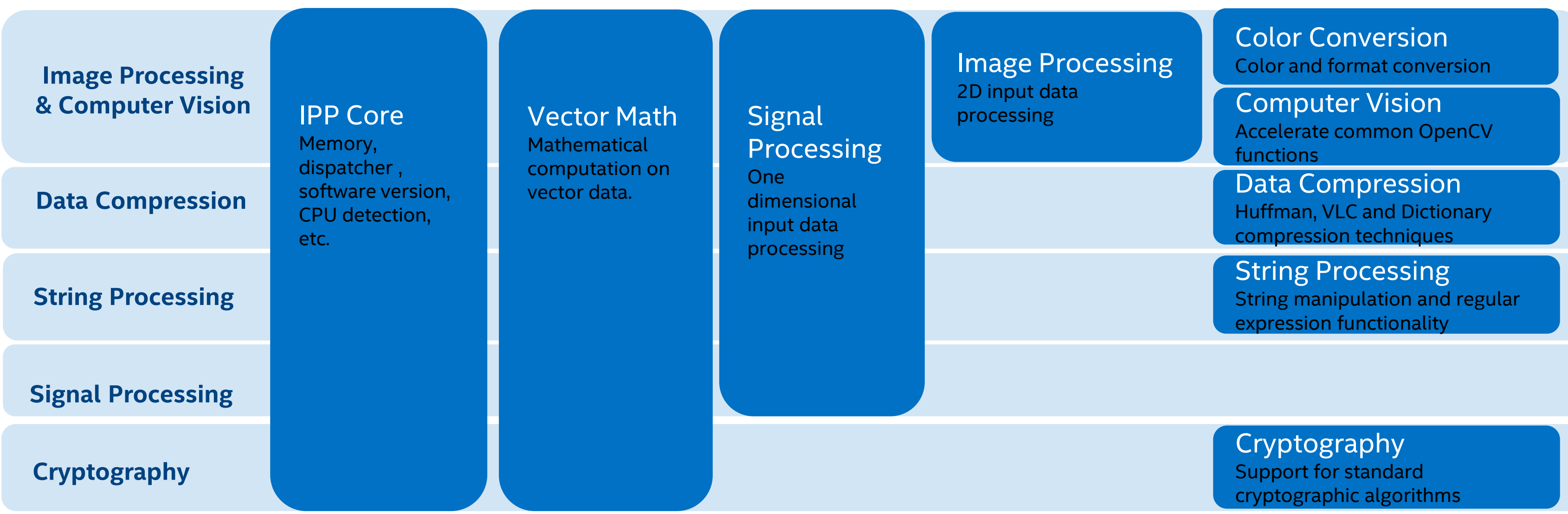


* Cryptography domain may not be available in all geographies

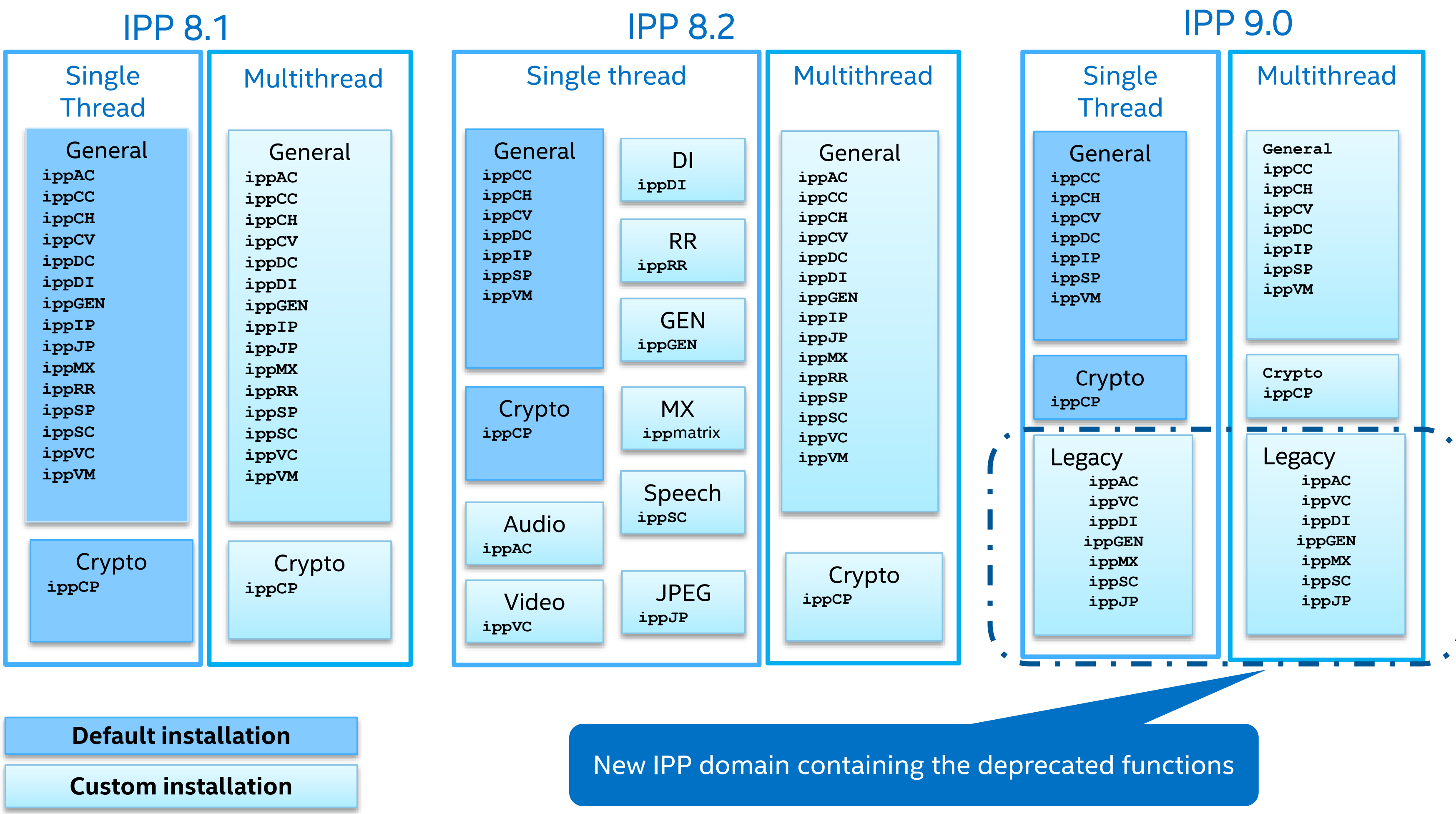
Intel® IPP 9.0 Focused Areas and Domains

Optimized performance primitives focused on Image Processing, Signal Processing, String Processing, Data Compression, Cryptography & Computer Vision

- Domains previously marked for deprecation are in the legacy library. Please see the IPP documentation for additional details.



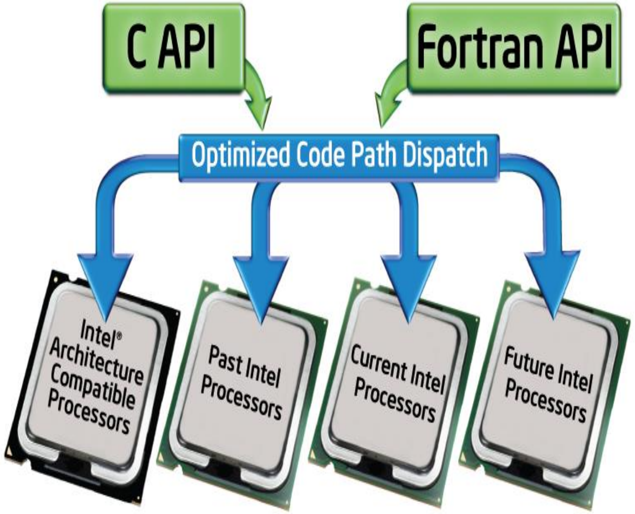
Intel® IPP 9.0 Packaging



Intel® Math Kernel Library

- The fastest and most used math library for Intel and compatible processors**
- De-facto industry standard APIs
- Supports math problems of many scientific

  applications
- Highly optimized threaded math routines
- The component of Intel® Parallel Studio XE and Intel® Composer XE
- Works with Intel, gcc, MSFT*, PGI compilers
- Windows*, Linux*, Mac OS*



**Source: Evans Data Software Developer surveys 2011-2013

Optimized Mathematical Building Blocks - Intel® Math Kernel Library

Linear Algebra

- BLAS
- LAPACK
- Sparse Solvers
  - Iterative
  - Pardiso*, Cluster_sparse_solver
  - ScaLAPACK

Fast Fourier Transforms

- Multidimensional
- FFTW interfaces
- Cluster FFT

Vector Math

- Trigonometric
- Hyperbolic
- Exponential, Log
- Power / Root

Vector RNGs

- Congruential
- Wichmann-Hill
- Mersenne Twister
- Sobol
- Neiderreiter
- Non-deterministic

Summary Statistics

- Kurtosis
- Variation coefficient
- Order statistics
- Min/max
- Variance-covariance

And More

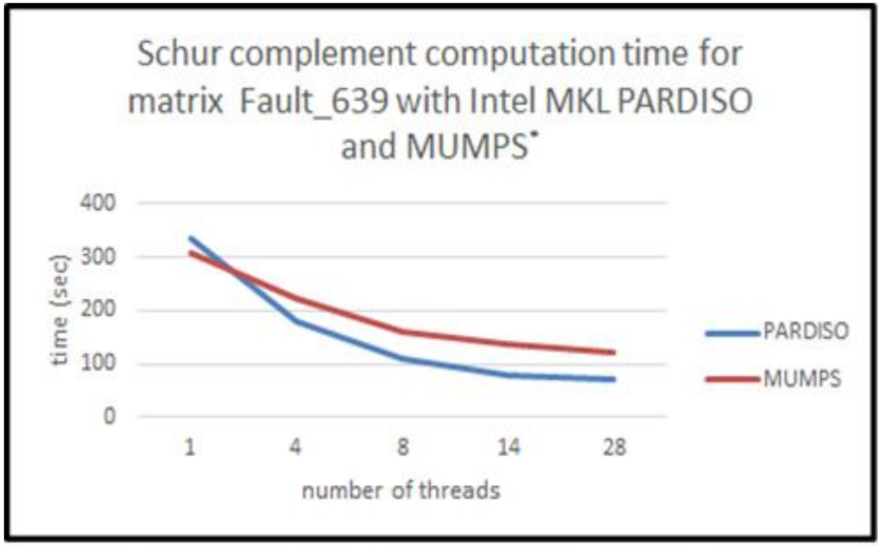
- Splines
- Interpolation
- Trust Region
- Fast Poisson Solver

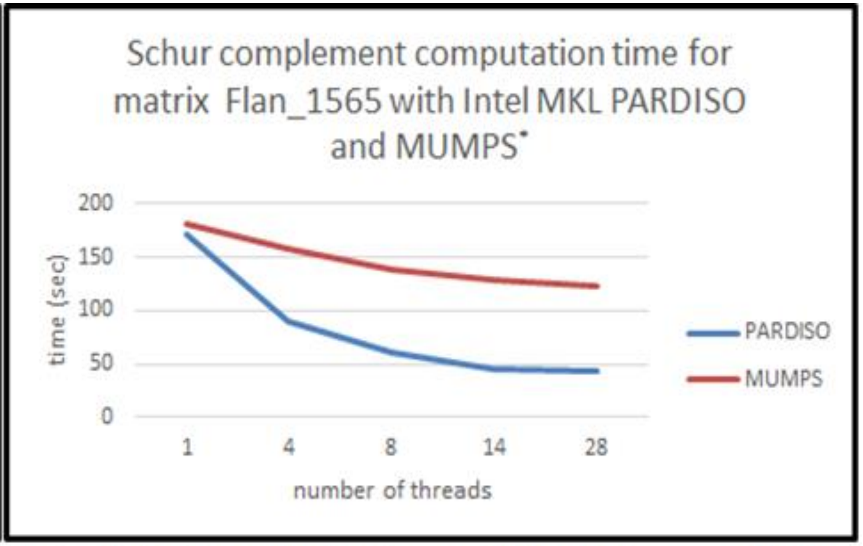
Intel® MKL 11.2 - new Features

- Parallel Direct Sparse Solvers for Clusters
- Verbose mode for BLAS and LAPACK
- S/C/Z/DGEMM improvements on small matrix sizes
- Significant SVD and Eigen Solvers performance improvements
- Cookbook recipes
- Other features and optimizations

Using Intel® MKL Parallel Direct Sparse Solvers for Clusters vs. MUMPS*

- Available since MKL 11.2. update 1
- iparm(36) = 1
- Comparison Intel MKL PARDISO with MUMPS in term of time needed for calculating the Schur complement





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.

Configuration Info - Versions: Intel® Math Kernel Library (Intel® MKL) 11.2, Intel® Xeon® E5-2697 v3 processors (35M Cache, 2.60 GHz) with 64Gb RAM memory, KMP_AFFINITY set to "compact", MUMPS version 4.10.0, and Intel MKL 11.2 update 1

Intel® MKL 11.3 beta - new Features

- Additional Sparse Matrix Vector Multiplication API
- MKL MPI wrappers
- Optimized HPCG benchmark
- Support For Small Matrix multiplication (Batch mode)
- Support for Philox4x35 and ARS5 RNG
- Sparse Solver SMP improvements

Additional Sparse Matrix Vector API

- The new API as a part of Sparse BLAS
- The building blocks for iterative sparse solvers
- Introduces inspector-executor pipeline:
  - Inspect step – analyze matrix to choose best strategy
    - Computational kernels for portrait
    - Balancing strategy for parallel execution
  - Execute step – use analysis data to get better performance
    - Several execution steps required to repay analysis overhead

- Supported Formats:
  - CSR
  - CSC
  - BCR *
  - COO

* no explicit ESB formats support

Optimization Notice

New SpMV API – Example, iterative computation

```
mkl_sparse_d_create_csr(&A, SPARSE_INDEX_BASE_ZERO,
                        rows, cols, rowsStart, rowsEnd, colIndx, values );
mkl_sparse_set_mv_hint(A,
    SPARSE_OPERATION_NON_TRANSPOSE,
    SPARSE_FULL,
    n_iter);
mkl_sparse_set_memory_hint(A, SPARSE_MEMORY_AGRESSIVE);
mkl_sparse_optimize(A);

for (int i=0;i<n_iter;i++) {
mkl_sparse_d_mv(SPARSE_OPERATION_NON_TRANSPOSE,
                alpha, A, SPARSE_FULL, x, beta, y);
…
}
mkl_sparse_destroy(A);
```

Optimization Notice

MKL MPI wrappers

- MKL supports intel® MPI, MPICH2*,MPICH3**, OpenMPI* and MS MPI
- Motivation: All MPI implementations (Intel MPI, MSMPI, MPICH, ...) are API-compatible but MPI implementations are **not ABI-compatible**
- MKL BLACS highly depended on concrete MPI implementation
- For MKL it means (since MKL is distributed in binary form):
  - have to compile all MPI specific functions with all supported MPI implementations
  - distribute the pre-built libraries:

`libmkl_blacs_lp64.a`,`libmkl_blacs_openmpi_lp64.a`,`libmkl_blacs_sgimpt_lp64.a` and `libmkl_blacs_intelmpi_lp64.a`

* - MPICH2 version 1.5 and MPICH3 version 3.1 (http://www-unix.mcs.anl.gov/mpi/mpich)
** - Open MPI 1.6 and 1.7 (http://www.open-mpi.org)

Optimization Notice

MKL MPI wrappers

- The MKLMPI wrapper solves this problem by providing an MPI-independent ABI to MKL
- all MPI specifics are in one file (mklmpi-impl.c), with one function that returns a structure of all needed MPI functions
- Customer have to recompile **mklmpi-impl.c** with his specific version of MPI and `put` it into one of installed MKL Blacs library
- Available for users: <mklroot>/interfaces/mklmpi/mklmpi-impl.c
- Use makefile to build the Custom BLACS library
- Once built, custom blacs libraries can be used in application linking with Intel MKL just like the pre-built ones
- LIMITATIONS : The MKLMPI wrappers are not supported with MKL single dynamic library.

Optimization Notice

Optimized HPCG benchmark

- HPCG is the Intel® Optimized benchmark of the HPCG benchmark (https://software.sandia.gov/hpcg)
- Optimized for Intel® AVX, Intel® AVX2 and Intel® Xeon Phi™ instruction sets
- The HPCG benchmark implementation is based on a 3D regular 27-point discretization of an elliptic partial differential equation
- HPCG contains the follow kernels:
  - sparse matrix vector multiplication (SpMV)
  - symmetric Gauss-Seidel smoother (SYMGS) and
  - Gauss-Seidel preconditioner (GS) kernels
- SpMV and GS kernels are implemented using an inspector-executor model

Optimization Notice

Top500 – Intel Optimized HPL and HPCG

Site	Computer	Cores	HPL Rmax (Pflops)	HPL Rank	HPCG (Pflops)	HPCG/ HPL
NSCC / Guangzhou	Tianhe-2 NUDT, Xeon 12C 2.2GHz + Intel Xeon Phi 57C + Custom	3,120,000	33.9	1	.580	1.7%
RIKEN Advanced Inst for Comp Sci	K computer Fujitsu SPARC64 VIIIfx 8C + Custom	705,024	10.5	4	.427	4.1%
DOE/OS Oak Ridge Nat Lab	Titan, Cray XK7 AMD 16C + Nvidia Kepler GPU 14C + Custom	560,640	17.6	2	.322	1.8%
DOE/OS Argonne Nat Lab	Mira BlueGene/Q, Power BQC 16C 1.60GHz + Custom	786,432	8.59	5	.101#	1.2%
Swiss CSCS	Piz Daint, Cray XC30, Xeon 8C + Nvidia Kepler 14C + Custom	115,984	6.27	6	.099	1.6%
Leibniz Rechenzentrum	SuperMUC, Intel 8C + IB	147,456	2.90	12	.0833	2.9%
CEA/TGCC-GENCI	Curie tine nodes Bullx B510 Intel Xeon 8C 2.7 GHz + IB	79,504	1.36	26	.0491	3.6%
Exploration and Production Eni S.p.A.	HPC2, Intel Xeon 10C 2.8 GHz + Nvidia Kepler 14C + IB	62,640	3.00	11	.0489	1.6%
DOE/OS L Berkeley Nat Lab	Edison Cray XC30, Intel Xeon 12C 2.4GHz + Custom	132,840	1.65	18	.0439 #	2.7%
Texas Advanced Computing Center	Stampede, Dell Intel (8c) + Intel Xeon Phi (61c) + IB	78,848	.881*	7	.0161	1.8%
Meteo France	Beaufix Bullx B710 Intel Xeon 12C 2.7 GHz + IB	24,192	.469 (.467*)	79	.0110	2.4%
Meteo France	Prolix Bullx B710 Intel Xeon 2.7 GHz 12C + IB	23,760	.464 (.415*)	80	.00998	2.4%
U of Toulouse	CALMIP Bullx DLC Intel Xeon 10C 2.8 GHz + IB	12,240	.255	184	.00725	2.8%
Cambridge U	Wilkes, Intel Xeon 6C 2.6 GHz + Nvidia Kepler 14C + IB	3584	.240	201	.00385	1.6%
TiTech	TUSBAME-KFC Intel Xeon 6C 2.1 GHz + IB	2720	.150	436	.00370	2.5%

Jack Dongarra & Piotr Luszczek
University of Tennessee/ORNL
Michael Heroux
Sandia National Labs,
available on netlib.org

* scaled to reflect the same number of cores
\# unoptimized implementation

Optimization Notice

Support for Philox4x35-10 and ARS-5 RNG

Philox4x351-10 and ARS-5 are counter-based pseudorandom number generators with a period of 2^128

ARS5 (Advanced randomization system) [1]:

- Output function is based on AES encryption algorithm, AES instructions are required
- Positioned as “fastest Crush-resistant[2] random number generator on CPUs”
- Limitation: run on IA only. Into another cases, VSR RNG routines return “VSL_RNG_ERROR_ARS5_NOT_SUPPORTED” status code

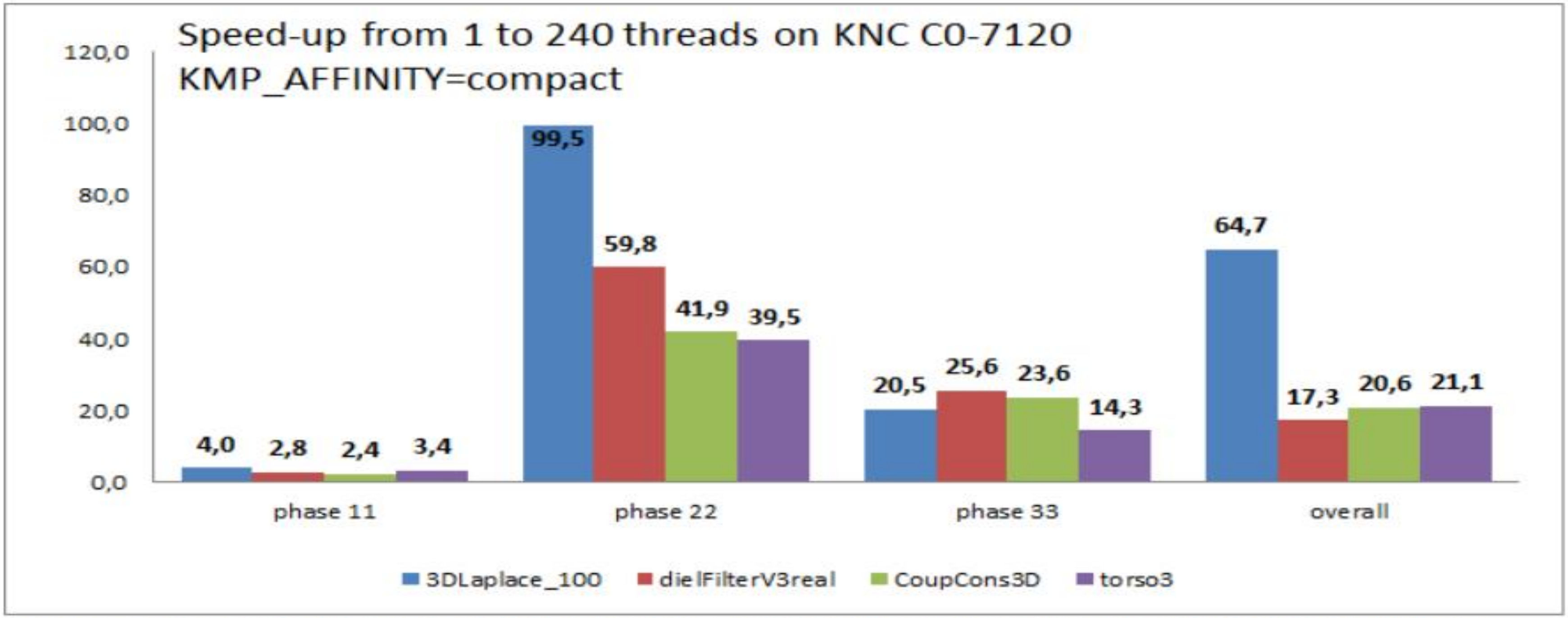
Philox4x32-10[1]:

- No complicated instructions are required, the RNG can be easily vectorized
- Positioned as “fastest Crush-resistant random number generator on GPUs”

[1]John K. Salmon, Mark A. Moraes, Ron O. Dror, and David E. Shaw. Parallel Random Numbers: As Easy as 1, 2, 3
[2]passing SmallCrush, Crush and BigCrush test batteries from TestU01

Optimization Notice

Sparse Solver SMP improvements



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

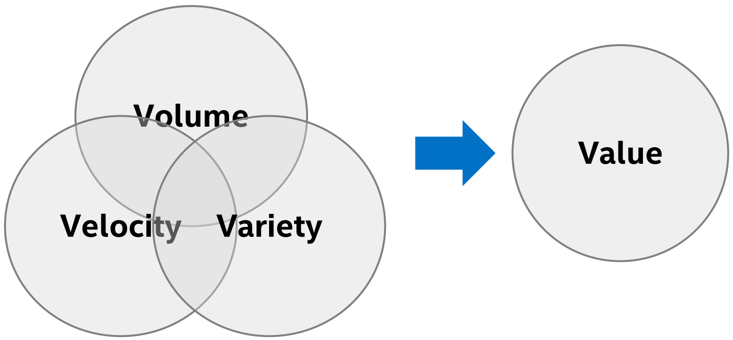
Speed-up from 1 to 240 threads, Intel® Xeon Phi™ Coprocessor 7120P (16GB, 1.238 GHz, 61 core, 244 threads), Native mode, Problem sizes ~ 250Kx250k… 1.1Mx1.1M

Optimization Notice

Outline

- Intel® IPP
  - Overview
  - IPP 9.0 beta – new features
- Intel® MKL
  - Overview
  - MKL 11.3 beta - new Features
- **Intel® DAAL**
- References

Data Analytics in the Age of Big Data



	Intel® Xeon® processor 64-bit	Intel® Xeon® processor 5100 series	Intel® Xeon® processor 5500 series	Intel® Xeon® processor 5600 series	Intel® Xeon® processor code-named Sandy Bridge EP	Intel® Xeon® processor code-named Ivy Bridge EP	Intel® Xeon® processor code-named Haswell EP	Intel® Xeon Phi™ coprocessor Knights Corner	Intel® Xeon Phi™ processor & coprocessor Knights Landing[1]
Core(s)	1	2	4	6	8	12	18	61	72
Threads	2	2	8	12	16	24	36	244	288
SIMD Width	128	128	128	128	256	256	256	512	512

*Product specification for launched and shipped products available on ark.intel.com.
1. Not launched or in planning.

More cores → More Threads → Wider vectors

Problem:

- Big data needs high performance computing.
- Many big data applications leave performance at the table – Not optimized for underlying hardware.

Solution:

- A performance library provides building blocks to be easily integrated into big data analytics workflow.

What Are We Releasing?

Intel DAAL 2016 Beta

- Available to selected partners in Feb 2015.
- Public beta starting in April 2015.

Intel DAAL 2016 product release

- Available in Q3 2015.

- Support IA-32 and Intel64 architectures.
- C++, Java APIs.
- Static and dynamic linking.
- A standalone library, and also bundled in Intel PSXE Cluster Edition 2016.

Note: Bundled version is not available on OS* X.

Intel® DAAL

New library targeting data analytics market

- **Customers:** analytics solution providers, system integrators, and application developers (FSI, Telco, Retail, Grid, etc.)
- **Key benefits:** improved time-to-value, forward-scaling performance and parallelism on IA, advanced analytics building blocks

Key features

- Building blocks highly optimized for IA to support all data analysis stages.
- Support batch, streaming, and distributed processing with easy connectors to popular platforms (Hadoop, Spark) and tools (R, Python, Matlab).
- Flexible interfaces for handling different data sources (CSV, MySQL, HDFS, RDD (Spark)).
- Rich set of operations to handle sparse and noisy data.
- C++ and Java APIs.

- 6 releases of Tech Preview in 2014. Received feedbacks and feature requests from multiple customers.
- First Beta to start in WW06 2015.
- Gold release in Aug 2015.

Intel® Data Analytics Acceleration Library – a C++ and Java API library of optimized analytics building blocks for all data analysis stages, from data acquisition to data mining and machine learning. Essential for engineering high performance Big Data applications.

Data Management
Interfaces for data representation and access. Connectors to a variety of data sources and data formats, such HDFS, SQL, CSV, ARFF, and user-defined data source/format.

Data Sources | Numeric Tables | Missing Values Handling, Outliers Detection | Compression / Decompression | Serialization / Deserialization

Data Processing
Parallel and optimized algorithms for statistical analysis, data transformation, model training, and decision making. Engines for batch, streaming, and distributed processing.

Data Modeling
Data structures for model representation, and operations to derive model-based predictions and conclusions.

Important features offered in the initial Beta

Analysis
- PCA
- Variance-Covariance Matrix
- Distances
- Matrix decompositions (SVD, QR, Cholesky)
- EM for GMM
- Uni-/multi-variate outlier detection
- Statistical moments

Machine learning
- Linear regression
- Apriori
- K-Means clustering
- Naïve Bayes
- LogitBoost, BrownBoost, AdaBoost
- SVM

- Data layouts: AOS, SOA, homogeneous, CSR
- Data sources: csv, MySQL, HDFS/RDD
- Compression/decompression: ZLIB, LZO, RLE, BZIP2
- Serialization/deserialization

References

Intel® MKL and MKL Forum pages

- http://software.intel.com/en-us/articles/intel-mkl/
- http://software.intel.com/en-us/articles/intel-math-kernel-library-documentation/
- http://software.intel.com/en-us/forums/intel-math-kernel-library/

Intel® IPP and IPP Forum pages:

- https://software.intel.com/en-us/intel-ipp
- https://software.intel.com/en-us/forums/intel-integrated-performance-primitives/

Intel® DAAL Forum page:

- https://software.intel.com/en-us/forums/intel-data-analytics-acceleration-library

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