INTEL® PERFORMANCE LIBRARIES

Fast, Scalable Code with Intel® Math Kernel Library (Intel® MKL)

- Speeds computations for scientific, engineering, financial and machine learning applications by providing highly optimized, threaded, and vectorized math functions
- Provides key functionality for dense and sparse linear algebra (BLAS, LAPACK, PARDISO), FFTs, vector math, summary statistics, deep learning, splines and more
- Dispatches optimized code for each processor automatically without the need to branch code
- Optimized for single core vectorization and cache utilization
- Automatic parallelism for multi-core and many-core
- Scales from core to clusters
- Available at no cost and royalty free
- Great performance with minimal effort!

INTEL® MATH KERNEL LIBRARY OFFERS...

- DENSE & SPARSE LINEAR ALGEBRA
- FAST FOURIER TRANSFORMS
- VECTOR MATH
- VECTOR RNGS
- FAST POISSON SOLVER
- & MORE!

[1] Available only in Intel® Parallel Studio Composer Edition.

Automatic Dispatching to Tuned ISA-specific Code Paths

More cores → More Threads → Wider vectors

	Intel® Xeon® Processor 64-bit	Intel® Xeon® Processor 5100 series	Intel® Xeon® Processor 5500 series	Intel® Xeon® Processor 5600 series	Intel® Xeon® Processor E5-2600 v2 series	Intel® Xeon® Processor E5-2600 v3 series v4 series	Intel® Xeon® Scalable Processor[1]	Intel® Xeon Phi™ x200 Processor (KNL)
Up to Core(s)	1	2	4	6	12	18-22	28	72
Up to Threads	2	2	8	12	24	36-44	56	288
SIMD Width	128	128	128	128	256	256	512	512
Vector ISA	Intel® SSE3	Intel® SSE3	Intel® SSE4- 4.1	Intel® SSE 4.2	Intel® AVX	Intel® AVX2	Intel® AVX-512	Intel® AVX-512

1. Product specification for launched and shipped products available on ark.intel.com.

What's New in Intel® Math Kernel Library 2019?

Just-In-Time Fast Small Matrix Multiplication

- Improved speed of S/DGEMM for Intel® AVX2 and Intel® AVX-512 with JIT capabilities

Sparse QR Solvers

- Solve sparse linear systems, sparse linear least squares problems, eigenvalue problems, rank and null-space determination, and others

Generate Random Numbers for Multinomial Experiments

- Highly optimized multinomial random number generator for finance, geological and biological applications

What's Inside Intel® Math Kernel Library

LINEAR ALGEBRA
- BLAS
- LAPACK
- ScaLAPACK
- Sparse BLAS
- Iterative sparse solvers
- PARDISO*
- Cluster Sparse Solver

FFTS
- Multidimensional
- FFTW interfaces
- Cluster FFT

VECTOR RNGS
- Congruential
- Wichmann-Hill
- Mersenne Twister
- Sobol
- Neirderreiter
- Non-deterministic

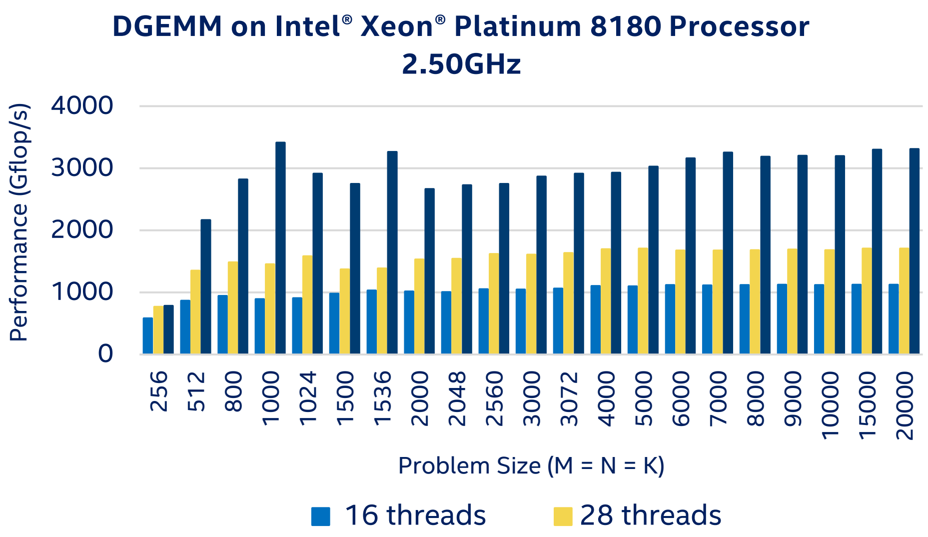
SUMMARY STATISTICS
- Kurtosis
- Variation coefficient
- Order statistics
- Min/max
- Variance-covariance

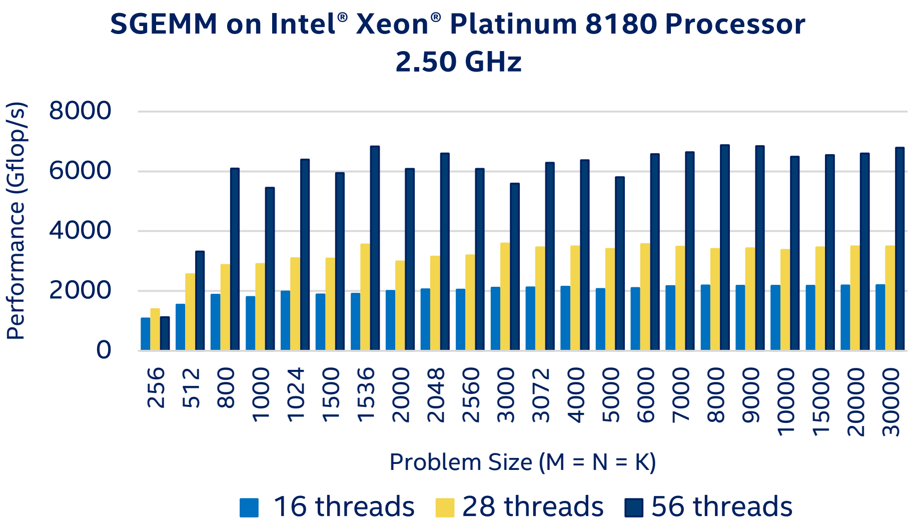
VECTOR MATH
- Trigonometric
- Hyperbolic
- Exponential
- Log
- Power
- Root

& MORE
- Splines
- Interpolation
- Trust Region
- Fast Poisson Solver

[1]Available only in Intel® Parallel Studio Composer Edition.

DGEMM, SGEMM Optimized by Intel® Math Kernel Library on Intel® Xeon® Processor





Performance results are based on testing as of July 9, 2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information, see Performance Benchmark Test Disclosure.Testing by Intel as of July 9, 2018.
Configuration: Intel® Xeon® Platinum 8180 H0 205W 2x28@2.5GHz 192GB DDR4-2666
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. For more complete information about compiler optimizations, see our Optimization Notice.

Speed Imaging, Vision, Signal, Security & Storage Apps with Intel® Integrated Performance Primitives (Intel® IPP)

Accelerate Image, Signal, Data Processing & Cryptography Computation Tasks

- Multi-core, multi-OS and multi-platform ready, computationally intensive & highly optimized functions
- Use high performance, easy-to-use, production-ready APIs to quickly improve application performance
- Reduce cost & time-to-market on software development & maintenance

What's New in 2019 Release

- Functions for ZFP floating-point data compression to help tackle large data storage challenges, great for oil/gas applications
- Optimization patch files for the bzip2 source 1.0.6
- Improved LZ4 compression & decompression performance on high entropy data
- New color conversion functions for convert RBG images to CIE Lab color models, & vice versa
- Extended optimization for Intel® AVX-512 & Intel® AVX2 instruction set
- Open source distribution of Intel® IPP Cryptography Library

Learn More: software.intel.com/intel-ipp

Intel® IPP Your Building Blocks for Image, Signal & Data Processing Applications

What is Intel® IPP?

Intel IPP provides developers with ready-to-use, processor- optimized functions to accelerate ***Image & Signal processing, Data Compression & Cryptography computation tasks***

Why should you use Intel® IPP?

- High Performance
- Easy to use API's
- Faster Time To Market (TTM)
- Production Ready
- Cross-platform API

How to get Intel® IPP?

- Intel Parallel Studio XE
- Intel System Studio
- Free Tools Program

Optimized for

Supports

Windows   Linux   Android   MacOS

Addresses

Data Center   Internet of Things   Embedded Systems   Cloud Computing

Image Processing Uses

- Medical Imaging
- Computer Vision
- Digital Surveillance
- ADAS
- Automated Sorting
- Biometric Identification
- Visual Search

Signal Processing Uses

- Games (sophisticated audio content or effects)
- Echo cancellation
- Telecommunications
- Energy

Data Compression & Cryptography Uses

- Data centers
- Enterprise data management
- ID verification
- Smart Cards/wallets
- Electronic Signature
- Information security/cybersecurity

Find out more at: http://software.intel.com/intel-ipp
Contact us through our forum: http://software.intel.com/en-us/forums/intel-integrated-performance-primitives

What's Inside Intel® Integrated Performance Primitives

High Performance, Easy-to-Use & Production Ready APIs

Image Processing

Computer Vision

Color Conversion

Signal Processing

Vector Math

Data Compression

Cryptography

String Processing

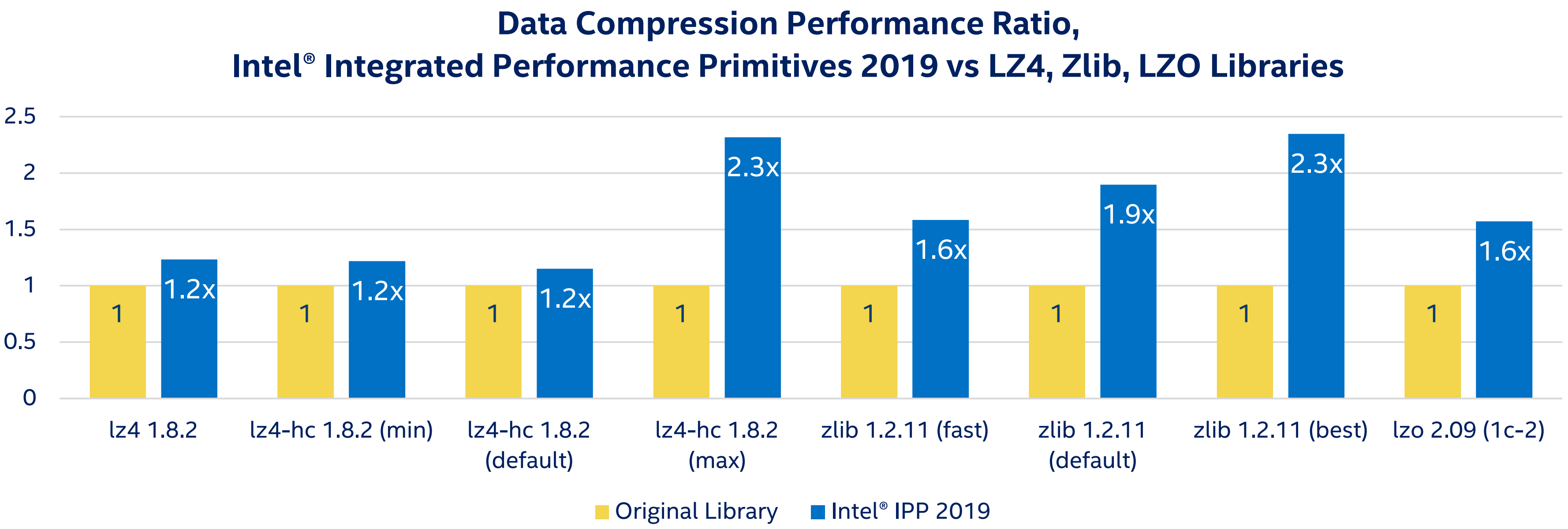
Image Domain | Signal Domain | Data Domain

Operating Systems: Windows*, Linux*, MacOS[1]*

Intel® Architecture Platforms

[1]Available only in Intel® Parallel Studio Composer Edition.

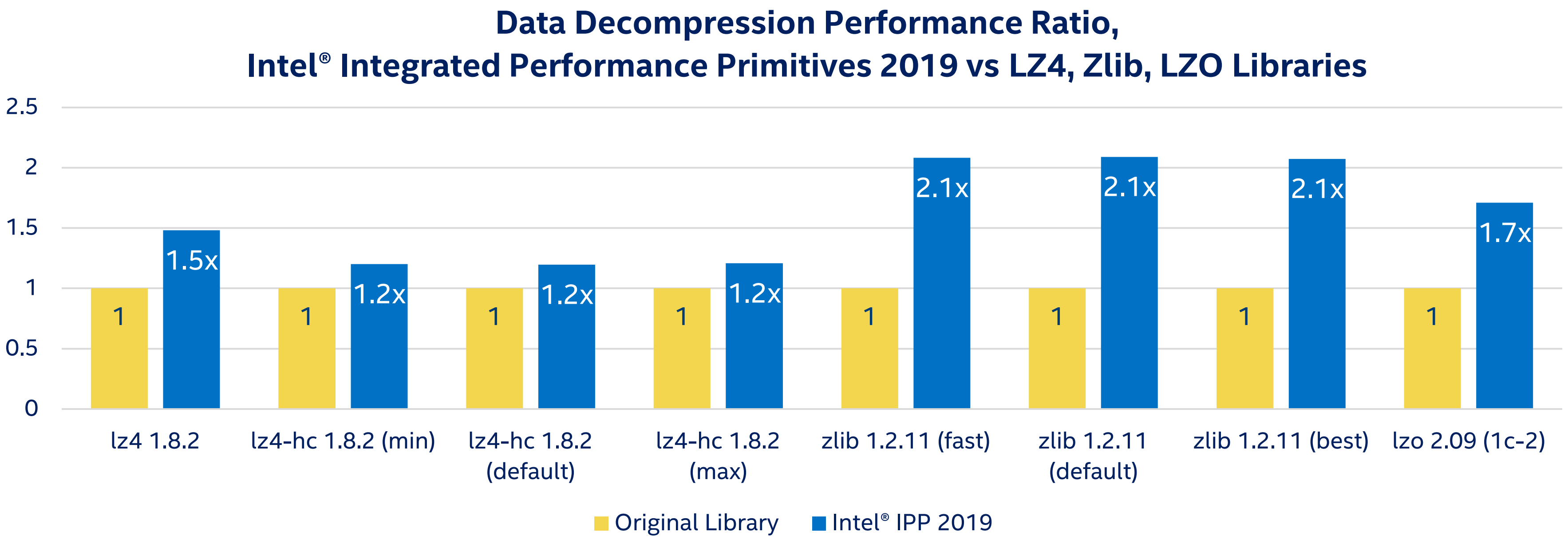
Performance Improvement for Data Compression



Performance results are based on testing as of Aug. 15, 2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information, see Performance Benchmark Test Disclosure.
Testing by Intel as of August 15, 2018. Configuration: Intel® Core™ i5-7600 CPU @3.50GHz, 4 cores, hyper-threading off; Cache: L1=32KB, L2=256KB, L3=6MB; Memory: 64GB; OS: RH EL Server 7.2
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. For more complete information about compiler optimizations, see our Optimization Notice.

Optimization Notice

Performance Improvement for Data Decompression



Performance results are based on testing as of Aug. 15, 2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information, see Performance Benchmark Test Disclosure.
Testing by Intel as of Aug. 15, 2018. Configuration: Intel® Core™ i5-7600 CPU @3.50GHz, 4 cores, hyper-threading off; Cache: L1=32KB, L2=256KB, L3=6MB; Memory: 64GB; OS: RH EL Server 7.2
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.
For more complete information about compiler optimizations, see our Optimization Notice.

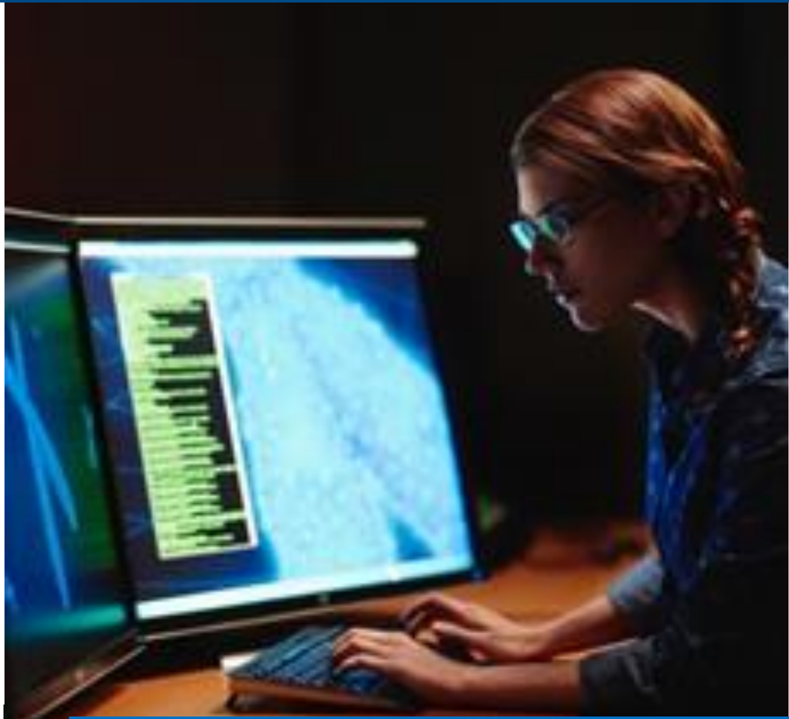
Get the Benefits of Advanced Threading with Threading Building Blocks

Use Threading to Leverage Multicore Performance & Heterogeneous Computing

- Parallelize computationally intensive work across CPUs, GPUs & FPGAs,—deliver higher-level & simpler solutions using C++
- Most feature-rich & comprehensive solution for parallel programming
- Highly portable, composable, affordable, approachable, future-proof scalability

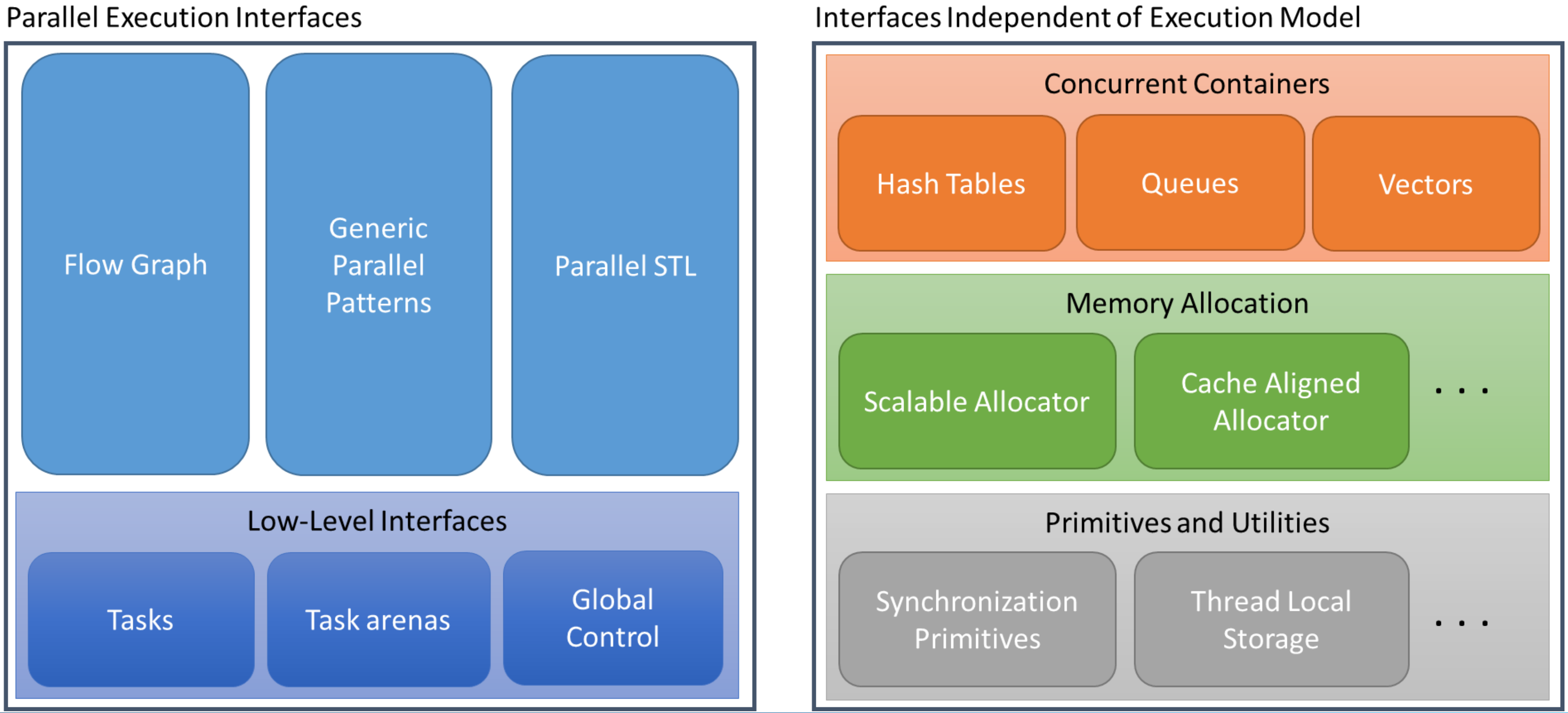
What's New in 2019 Release

- New capabilities in Flow Graph improve concurrency & heterogeneity through improved task analyzer & OpenCL* device selection
- New templates to optimize C++11 multidimensional arrays
- C++17 Parallel STL, OpenCL*, & Python* Conda language support
- Expanded Windows*, Linux*, Android*, MacOS* support



Learn More: software.intel.com/intel-tbb

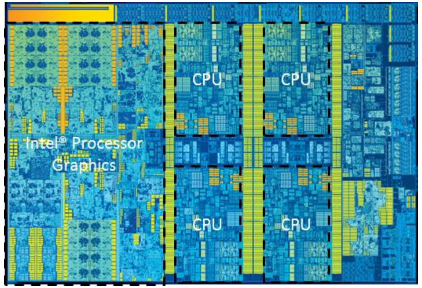
What's Inside Threading Building Blocks



Heterogeneous Support

Threading Building Blocks (TBB)

TBB flow graph as a coordination layer for heterogeneity—retains optimization opportunities & composes with existing models



CPUs, integrated GPUs, etc.

+

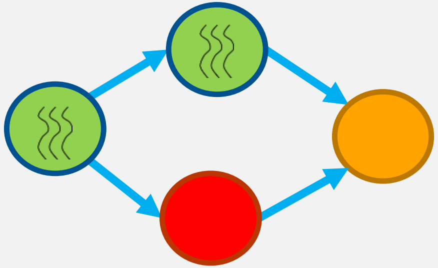
Threading Building Blocks
OpenVX*
OpenCL*
COI/SCIF
....

TBB as a **composability layer** for library implementations

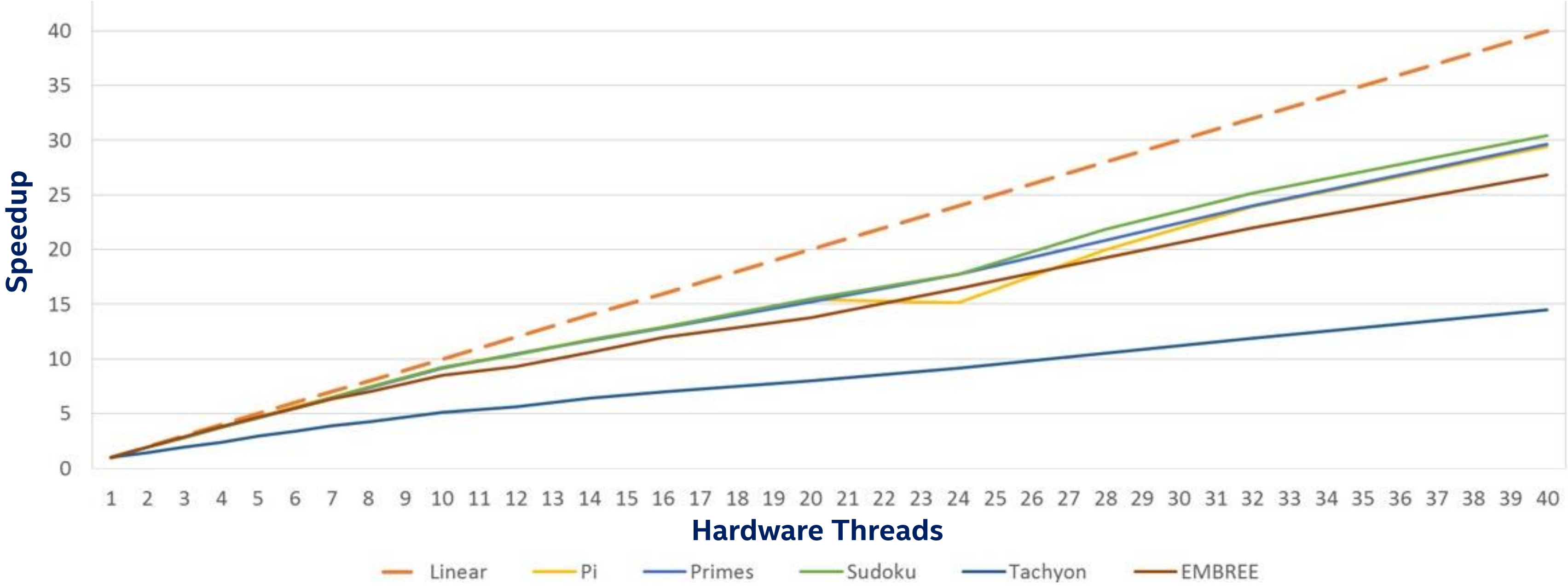
- One threading engine ***underneath*** all CPU-side work

TBB flow graph as a **coordination layer**

- Be the glue that connects heterogeneous hardware & software together
- Expose parallelism between blocks—simplify integration



Excellent Performance Scalability with Threading Building Blocks on Intel® Xeon® Processor



Performance results are based on testing as of July 31, 2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks. Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.**Configuration:** Testing by Intel as of July 31, 2018. Software versions: Intel® C++ Intel® 64 Compiler, Version 18.0, Threading Building Blocks (TBB) 2019; Hardware: 2x Intel® Xeon® Gold 6152 CPU @ 2.10GHz, 192GB Main Memory; Operating System: CentOS Linux* release 7.4 1708 (Core), kernel 3.10.0-693.e17.x86_64; Note: sudoku, primes and tachyon are included with TBB.

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.
For more complete information about compiler optimizations, see our Optimization Notice.

Speedup Analytics & Machine Learning with Intel® Data Analytics Acceleration Library (Intel® DAAL)

- Highly tuned functions for classical machine learning & analytics performance from datacenter to edge running on Intel® processor-based devices
- Simultaneously ingests data & computes results for highest throughput performance
- Supports batch, streaming & distributed usage models to meet a range of application needs
- Includes Python*, C++, Java* APIs, & connectors to popular data sources including Spark* & Hadoop

What's New in the 2019 Release

New Algorithms

- **Logistic Regression**, most widely-used classification algorithm
- **Extended Gradient Boosting Functionality** for inexact split calculations & user-defined callback canceling for greater flexibility
- **User-defined Data Modification Procedure** supports a wide range of feature extraction & transformation techniques

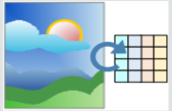
Learn More: software.intel.com/daal

Pre-processing → **Transformation** → **Analysis** → **Modeling** → **Validation** → **Decision Making**

Pre-processing

Decompression, Filtering, Normalization

Transformation



Aggregation, Dimension Reduction

Analysis

Summary Statistics Clustering, etc.

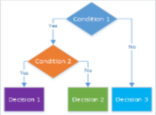
Modeling

Machine Learning (Training) Parameter Estimation Simulation

Validation

Hypothesis Testing Model Errors

Decision Making



Forecasting Decision Trees, etc.

Algorithms, Data Transformation & Analysis

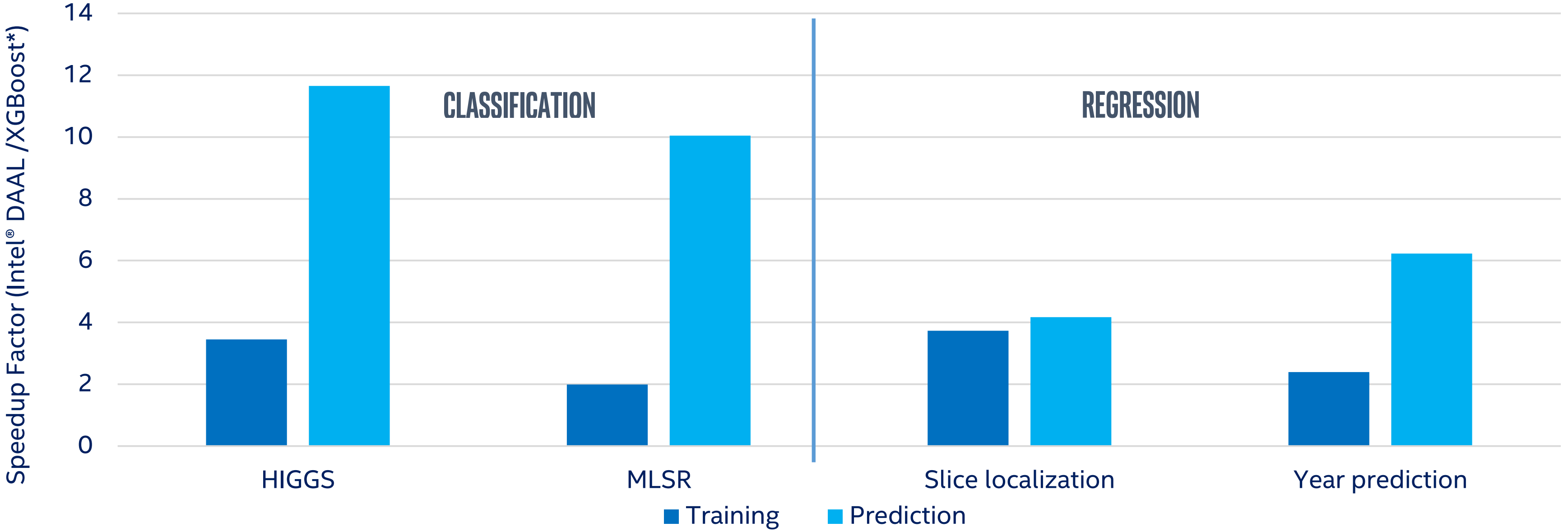
Intel® Data Analytics Acceleration Library

- **Basic Statistics for Datasets**
  - Low Order Moments
  - Quantiles
  - Order Statistics
- **Correlation & Dependence**
  - Cosine Distance
  - Correlation Distance
  - Variance-Covariance Matrix
- **Matrix Factorizations**
  - SVD
  - QR
  - Cholesky
- **Dimensionality Reduction**
  - PCA
- Association Rule Mining (Apriori)
- Optimization Solvers (SGD, AdaGrad, lBFGS)
- **Outlier Detection**
  - Univariate
  - Multivariate
- Math Functions (exp, log,…)

Algorithms supporting batch processing

Algorithms supporting batch, online and/or distributed processing

Intel® Data Analytics Acceleration Library 2019 Speedup vs XGBoost*



XGBoost Open Source Project
Performance results are based on testing as of **July 9, 2018** and may not reflect all publicly available security updates. See configuration disclosure for details. No product can be absolutely secure.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information, see Performance Benchmark Test Disclosure.
Testing by Intel as of July 9, 2018. Configuration: Intel® Xeon® Platinum 8180 H0 205W, 2x28@2.50GHz, 192GB, 12x16gb DDR4-2666, Intel® Data Analytics Acceleration Library (Intel® DAAL 2019), RHEL 7.2
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. For more complete information about compiler optimizations, see our Optimization Notice.

BACKUP

Intel® MKL BLAS (Basic Linear Algebra Subprograms)

De-facto Standard APIs since the 1980s	
100s of Basic Linear Algebra Functions	Level 1 – vector vector operations, O(N) Level 2 – matrix vector operations, $O(N^2)$ Level 3 – matrix matrix operations, $O(N^3)$
Precisions Available	Real – Single and Double Complex - Single and Double
BLAS-like Extensions	Direct Call, Batched, Packed and Compact
Reference Implementation	*http://netlib.org/blas/*

Intel® MKL LAPACK (Linear Algebra PACKage)

De-facto Standard APIs since the 1990s

1000s of Linear Algebra Functions	Matrix factorizations - LU, Cholesky, QR Solving systems of linear equations Condition number estimates Symmetric and non-symmetric eigenvalue problems Singular value decomposition and many more …
Precisions Available	Real – Single and Double, Complex – Single and Double
Reference Implementation	*http://netlib.org/lapack/*

Intel® MKL Fast Fourier Transforms (FFTs)

FFTW Interfaces support	C, C++ and FORTRAN source code wrappers provided for FFTW2 and FFTW3. FFTW3 wrappers are already built into the library
Cluster FFT	Perform Fast Fourier Transforms on a cluster Interface similar to DFTI Multiple MPIs supported
Parallelization	Thread safe with automatic thread selection
Storage Formats	Multiple storage formats such as CCS, PACK and Perm supported
Batch support	Perform multiple transforms in a single call
Additional Features	Perform FFTs on partial images Padding added for better performance Transform combined with transposition mixed-language usage supported

Intel® MKL Vector Math

Example:	y(i) = $e^{x(i)}$ for *i =1 to n*
Broad Function Support	Basic Operations – add, sub, mult, div, sqrt Trigonometric– sin, cos, tan, asin, acos, atan Exponential – exp,, pow, log, log10, log2, Hyperbolic – sinh, cosh, tanh Rounding – ceil, floor, round And many more
Precisions Available	Real – Single and Double Complex - Single and Double
Accuracy Modes	High - almost correctly rounded Low - last 2 bits in error Enhanced Performance - 1/2 the bits correct

Intel® MKL Sparse Solvers

PARDISO - Parallel Direct Sparse Solver	Factor and solve Ax = b using a parallel shared memory *LU*, *LDL*, or LL^T factorization Supports a wide variety of matrix types including real, complex, symmetric, indefinite, … Includes out-of-core support for very large matrix sizes
Parallel Direct Sparse Solver for Clusters	Factor and solve Ax = b using a parallel distributed memory *LU*, *LDL*, or LL^T factorization Supports a wide variety of matrix types (real, complex, symmetric, indefinite, …) Supports A stored in 3-array CSR3 or BCSR3 formats
DSS – Simplified PARDISO Interface	An alternative, simplified interface to PARDISO
ISS – Iterative Sparse Solvers	Conjugate Gradient (CG) solver for symmetric positive definite systems Generalized Minimal Residual (GMRes) for non-symmetric indefinite systems Rely on Reverse Communication Interface (RCI) for matrix vector multiply

intel®