Faster Code….
Faster

Intel® Parallel Studio XE 2016

Mike Lee

Unleash the Beast…

Create Faster Code…Faster

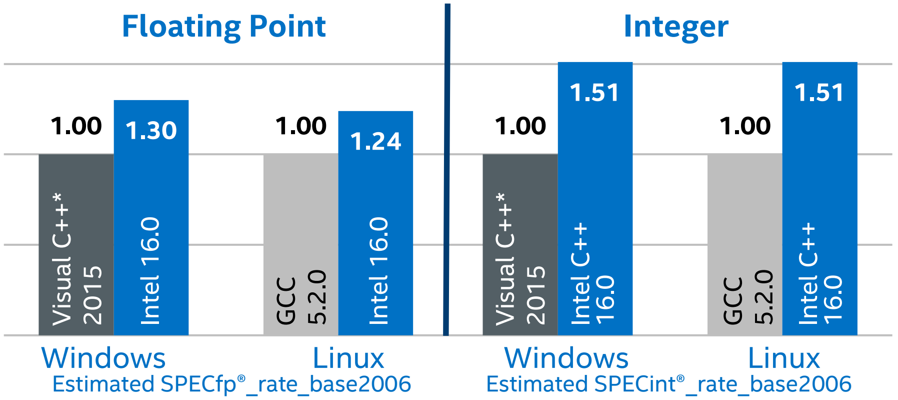
- Intel® Parallel Studio XE
  - Design, build, verify and tune
  - C++, C, Fortran and Java*
- Highlights from what’s new for “2016” edition
  - Intel® Data Analytics Acceleration Library
  - Vectorization Advisor:
    Custom Analysis and Advice
  - MPI Performance Snapshot: Scalable profiling
  - Support for the latest Standards, Operating
    Systems and Processors

http://intel.ly/perf-tools

Performance without Compromise

Intel® C++ and Fortran Compilers on Windows*, Linux* & OS X*

Boost C++ application performance on Windows* & Linux* using Intel® C++ Compiler (higher is better)



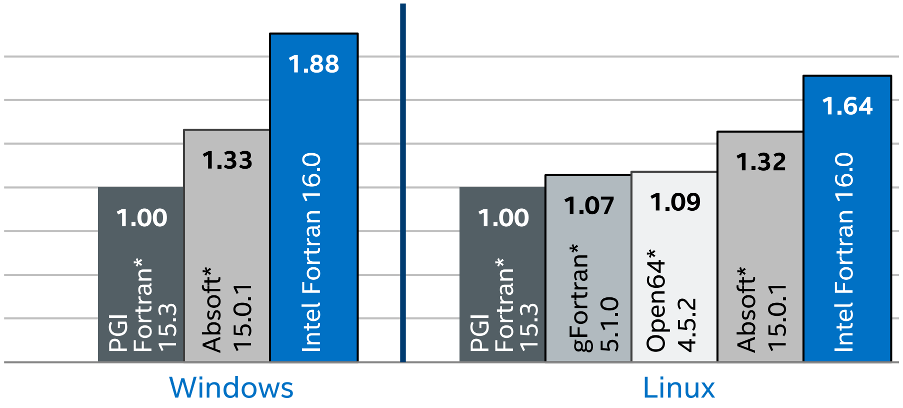
Relative geomean performance, SPEC* benchmark - higher is better

Configuration: Windows hardware: HP DL320e Gen8 v2 (single-socket server) with Intel(R) Xeon(R) CPU E3-1280 v3 @ 3.60GHz, 32 GB RAM, HyperThreading is off; Linux hardware: HP BL460c Gen9 with Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz, 256 GB RAM, HyperThreading is on. Software: Intel C++ compiler 16.0, Microsoft (R) C/C++ Optimizing Compiler Version 19.00.23026 for x86/x64, GCC 5.2.0. Linux OS: Red Hat Enterprise Linux Server release 7.1 (Maipo), kernel 3.10.0-229.el7.x86_64. Windows OS: Windows 8.1. SPEC* Benchmark (www.spec.org).

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. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

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.

Boost Fortran application performance on Windows* & Linux* using Intel® Fortran Compiler (higher is better)



Relative geomean performance, Polyhedron* benchmark– higher is better

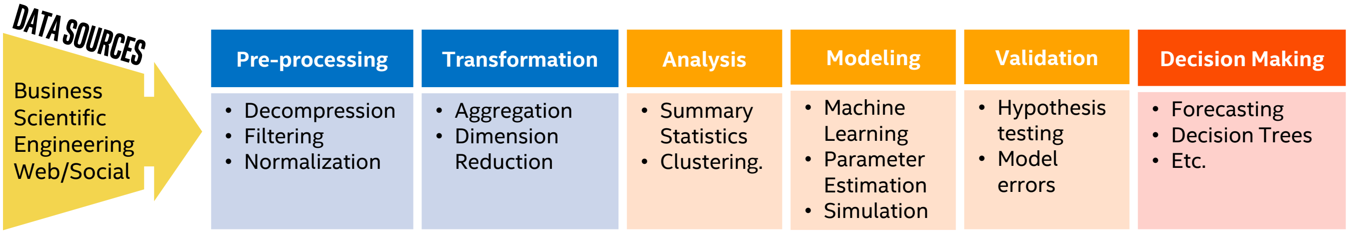
Configuration: Hardware: Intel(R) Core(TM) i7-4770K CPU @ 3.50GHz, HyperThreading is off, 16 GB RAM. Software: Intel Fortran compiler 16.0, Absoft*15.0.1,. PGI Fortran* 15.3, Open64* 4.5.2, gFortran* 5.1.0. Linux OS: Red Hat Enterprise Linux Server release 7.0 (Maipo), kernel 3.10.0-123.el7.x86_64. Windows OS: Windows 7, Service pack 1. Polyhedron Fortran Benchmark (www.fortran.uk). Windows compiler switches: Absoft: -m64 -O5 -speed_math=10 -fast_math -march=core -xINTEGER -stack:0x80000000. Intel® Fortran compiler: /fast /Qparallel /link /stack:64000000. PGI Fortran: -fastsse -Munroll=n:4 -Mipa=fast,inline -Mconcur=numa. Linux compiler switches: Absoft -m64 -mavx -O5 -speed_math=10 -march=core -xINTEGER. Gfortran: -Ofast -mfpmath=sse -flto -march=native -funroll-loops -ftree-parallelize-loops=4. Intel Fortran compiler: -fast –parallel. PGI Fortran: -fast -Mipa=fast,inline -Msmartalloc -Mfprelaxed -Mstack_arrays -Mconcur=bind. Open64: -march=bdver1 -mavx -mno-fma4 -Ofast -mso –apo.

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. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

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.

Turn Big Data Into Information Faster with Intel® Data Analytics Acceleration Library

Advanced analytics algorithms supporting all data analysis stages.



DATA SOURCES

Business
Scientific
Engineering
Web/Social

Pre-processing
- Decompression
- Filtering
- Normalization

Transformation
- Aggregation
- Dimension Reduction

Analysis
- Summary Statistics
- Clustering.

Modeling
- Machine Learning
- Parameter Estimation
- Simulation

Validation
- Hypothesis testing
- Model errors

Decision Making
- Forecasting
- Decision Trees
- Etc.

Designed and Built by Intel to

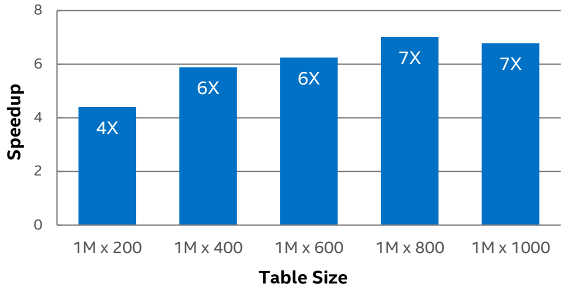
Delight Data Scientists

Simple to incorporate object-oriented APIs for C++ and Java

Easy connections to:

- Popular analytics platforms (Hadoop, Spark)
- Data sources (SQL, non-SQL, files, in-memory)

PCA Performance Boost Using Intel® DAAL vs. Spark* MLLib



Configuration Info - Versions: Intel® Data Analytics Acceleration Library 2016, CDH v5.3.1, Apache Spark* v1.2.0; Hardware: Intel® Xeon® Processor E5-2699 v3, 2 Eighteen-core CPUs (45MB LLC, 2.3GHz), 128GB of RAM per node; Operating System: CentOS 6.6 x86_64. PCA normalized input.

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. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

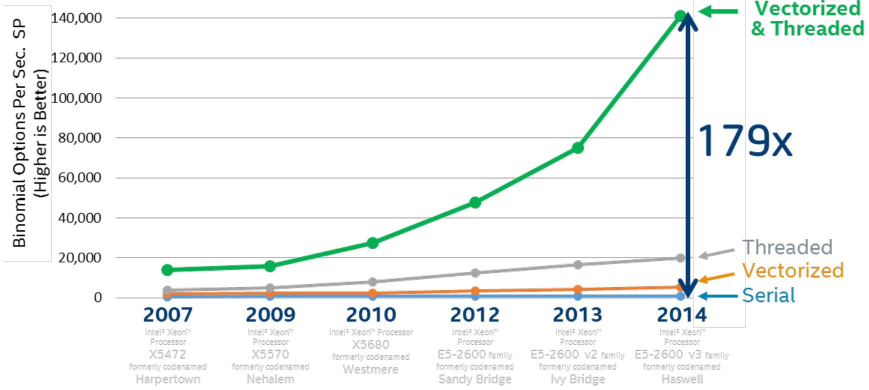
Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, 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 .

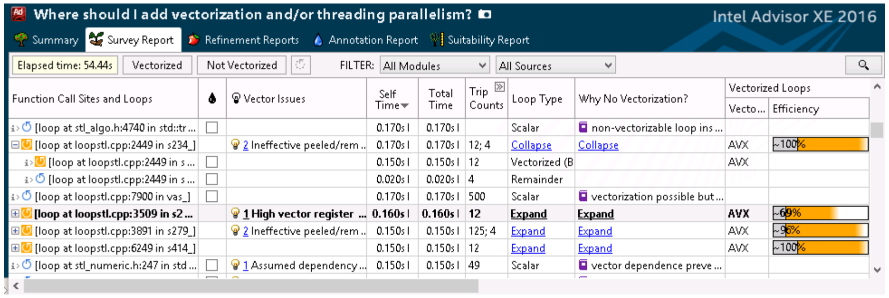
Intel® Advisor

Vectorization Optimation & Threading Prototyping

- Vectorizing and threading your code is a MUST on modern processors
- Easy optimization workflow to support development of faster code
- Prioritize, prototype & predict performance gain

Benefits of Vectorization & Threading





Function Call Sites and Loops	💧	Vector Issues	Self Time	Total Time	Trip Counts	Loop Type	Why No Vectorization?	Vectorized Loops	
								Vecto...	Efficiency
[loop at stl_algo.h:4740 in std::tr...	□		0.170s	0.170s		Scalar	non-vectorizable loop ins...		
[loop at loopstl.cpp:2449 in s234_]		2 Ineffective peeled/rem...	0.170s	0.170s	12; 4	Collapse	Collapse	AVX	~100%
[loop at loopstl.cpp:2449 in s...	□		0.150s	0.150s	12	Vectorized (B		AVX	
[loop at loopstl.cpp:2449 in s...	□		0.020s	0.020s	4	Remainder			
[loop at loopstl.cpp:7900 in vas_]	□		0.170s	0.170s	500	Scalar	vectorization possible but...		
[loop at loopstl.cpp:3509 in s2...		**1 High vector register ...**	**0.160s**	**0.160s**	**12**	**Expand**	**Expand**	**AVX**	~69%
[loop at loopstl.cpp:3891 in s279_]		2 Ineffective peeled/rem...	0.150s	0.150s	125; 4	Expand	Expand	AVX	~96%
[loop at loopstl.cpp:6249 in s414_]			0.150s	0.150s	12	Expand	Expand	AVX	~100%
[loop at stl_numeric.h:247 in std ...	□	1 Assumed dependency...	0.150s	0.150s	49	Scalar	vector dependence preve...		

Intel® Parallel Studio XE - Try it Today! bit.ly/ipsxe-try

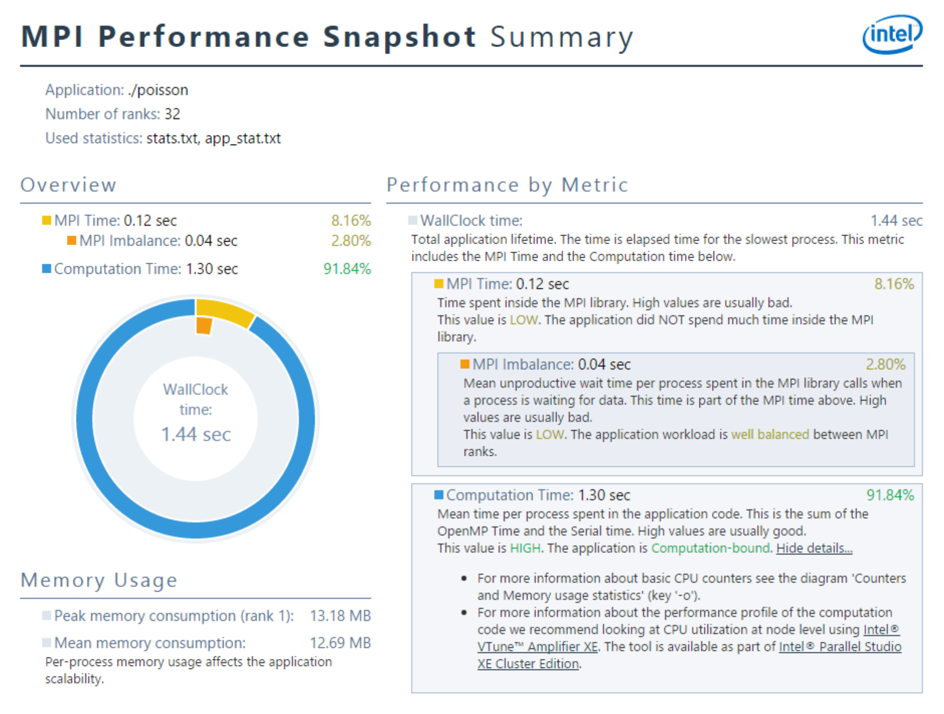
Optimization Notice

Scalable Profiling for MPI and Hybrid Clusters with MPI Performance Snapshot

Lightweight – Low overhead profiling up to 32K Ranks

Scalability- Performance variation at scale can be detected sooner

Identifying Key Metrics – Shows PAPI counters and MPI/OpenMP* imbalances



MPI Performance Snapshot Summary

Application: ./poisson
Number of ranks: 32
Used statistics: stats.txt, app_stat.txt

Overview

🟨 MPI Time: 0.12 sec	8.16%
🟧 MPI Imbalance: 0.04 sec	2.80%
🟦 Computation Time: 1.30 sec	91.84%

Memory Usage

Peak memory consumption (rank 1):	13.18 MB
Mean memory consumption:	12.69 MB

Per-process memory usage affects the application scalability.

Performance by Metric

WallClock time: 1.44 sec
Total application lifetime. The time is elapsed time for the slowest process. This metric includes the MPI Time and the Computation time below.

🟨 MPI Time: 0.12 sec 8.16%
Time spent inside the MPI library. High values are usually bad.
This value is LOW. The application did NOT spend much time inside the MPI library.

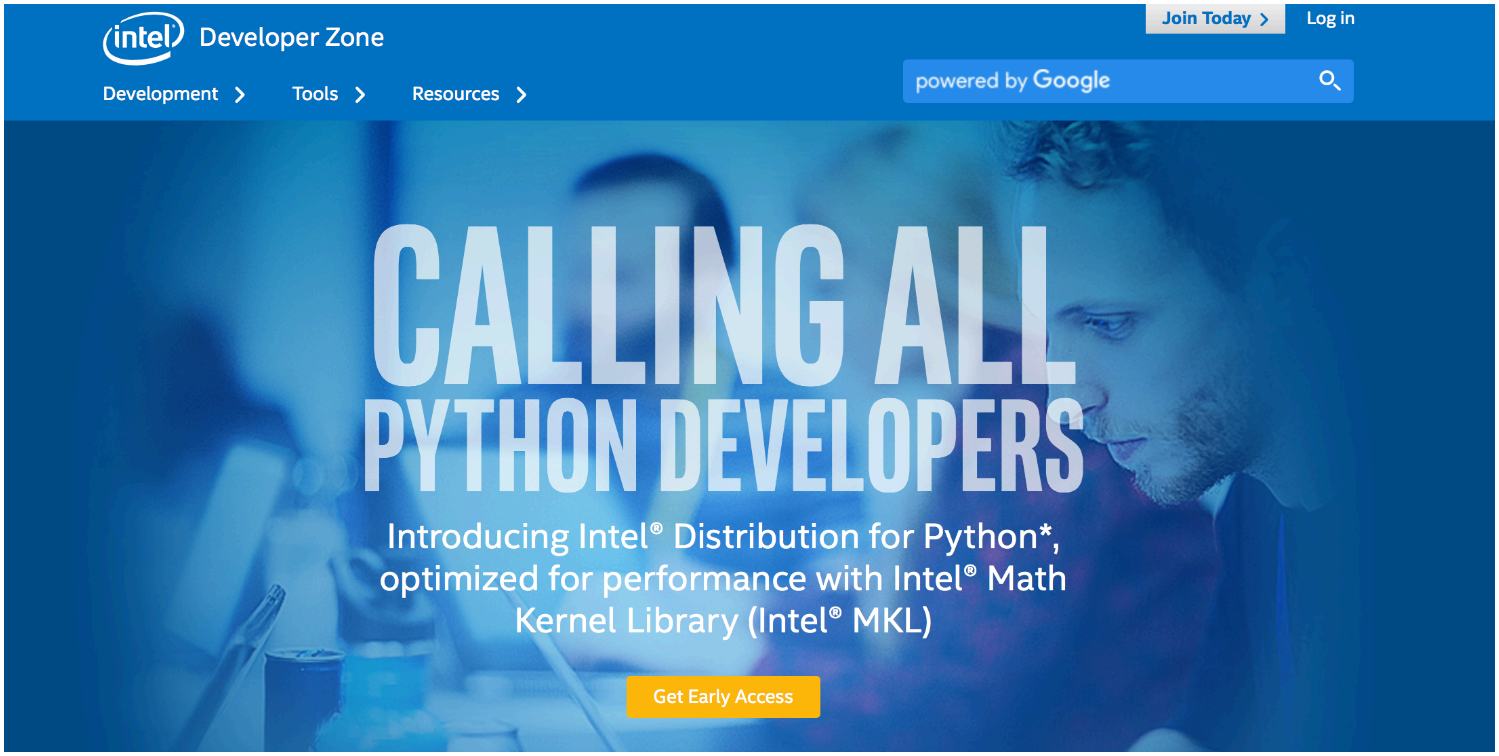
🟧 MPI Imbalance: 0.04 sec 2.80%
Mean unproductive wait time per process spent in the MPI library calls when a process is waiting for data. This time is part of the MPI time above. High values are usually bad.
This value is LOW. The application workload is well balanced between MPI ranks.

🟦 Computation Time: 1.30 sec 91.84%
Mean time per process spent in the application code. This is the sum of the OpenMP Time and the Serial time. High values are usually good.
This value is HIGH. The application is Computation-bound. Hide details...

- For more information about basic CPU counters see the diagram 'Counters and Memory usage statistics' (key '-o').
- For more information about the performance profile of the computation code we recommend looking at CPU utilization at node level using Intel® VTune™ Amplifier XE. The tool is available as part of Intel® Parallel Studio XE Cluster Edition.

Optimization Notice

Intel® Distribution for Python! – Technical Preview

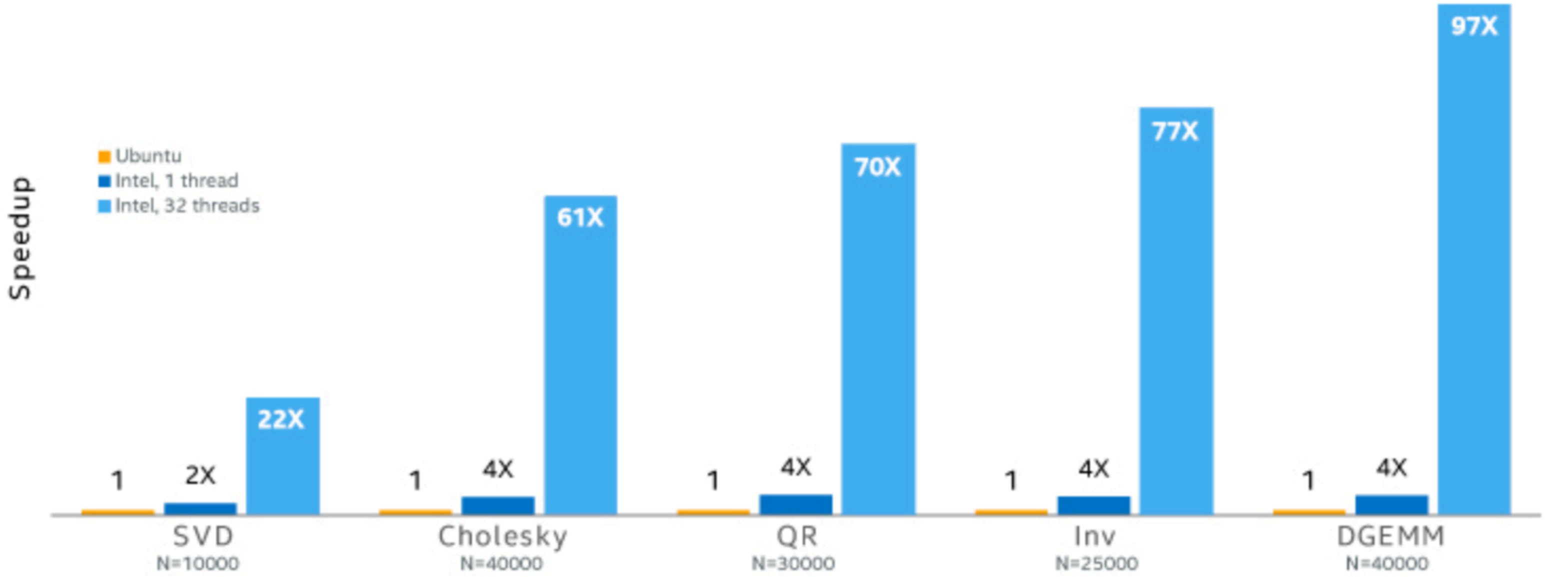


Introducing Intel® Distribution for Python*, optimized for performance with Intel® Math Kernel Library (Intel® MKL)

Sign Up at - http://bit.ly/intel-python

Python Performance Boost on Select Numerical Functions

Intel Distribution for Python (Technical Preview) vs. Ubuntu* Python

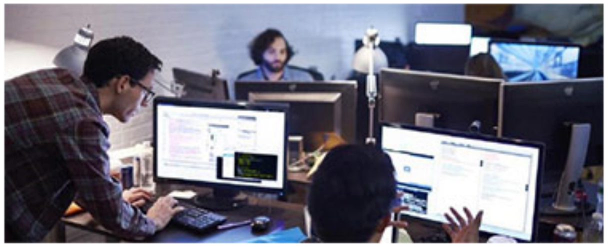


Configuration info: - Versions: Intel® Distribution for Python 2.7.10 Technical Preview 1 (Aug 03, 2015), Ubuntu* built Python*: Python 2.7.10, NumPy 1.9.2 built with gcc 4.8.4; Hardware: Intel® Xeon® CPU E5-2698 v3 @ 2.30GHz (2 sockets, 16 cores each, HT=OFF), 64 GB of RAM, 8 DIMMS of 8GB@2133MHz; Operating System: Ubuntu 14.04 LTS.

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. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

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 .

Free Intel® Software Development Tools for:

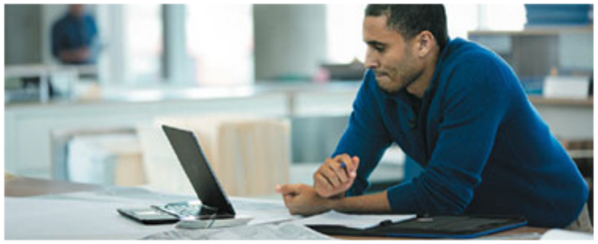


Academic Researcher ›

Intel Performance Libraries for academic research

Everyone ›

Community Licenses for Intel® Performance Libraries



Student ›

For current students at degree-granting institutions.

Educator ›

For use in teaching curriculum.

Open Source Contributor ›

For developers actively contributing to open source projects.

Visit us at https://software.intel.com/en-us/qualify-for-free-software

Legal Disclaimer & Optimization Notice

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.

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.

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