High Performance Dense Linear Algebra in Intel® Math Kernel Library (Intel® MKL)

Michael Chuvelev, Intel Corporation,
michael.chuvelev@intel.com
Sergey Kazakov, Intel Corporation
sergey.kazakov@intel.com

International Conference Dedicated to the 100th Anniversary of the Birthday of Sergei L. Sobolev
Novosibirsk
November 24, 2008

Contents

- Dense Linear Algebra Functionality in Intel MKL
- Matrix-Matrix Multiplication
- High-level Algorithms

Functionality
LAPACK Contents

- Intel MKL[1] includes wide range of Dense Linear Algebra functionality:
  - Linear Equations
  - Linear Least Squares
  - Symmetric Eigenproblems
  - Singular Value Decompositions
  - Nonsymmetric Eigenproblems
- Intel MKL functionality strictly corresponds to de-facto LAPACK (Linear Algebra PACKage) standard[2]
- Reference code is freely available at NetLib site[3]

[1] *Intel® Math Kernel Library, http://www.intel.com/software/products/mkl*

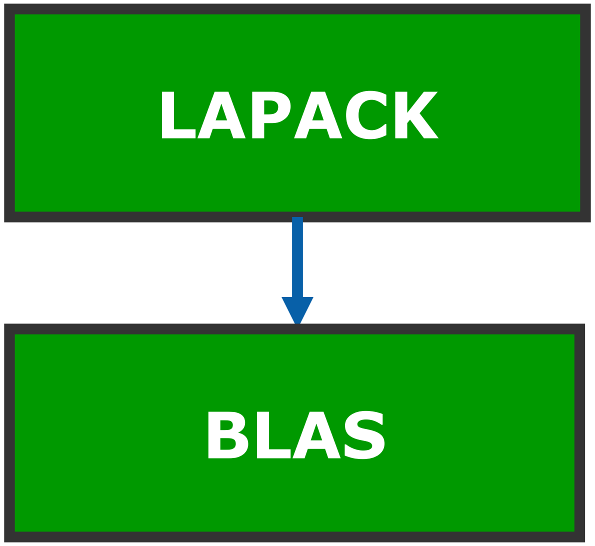
[2] *Anderson, E., Bai, Z., Bischof, C., Blackford, L. S., Demmel, J., Dongarra, J., DuCroz, J., Greenbaum, A., Hammarling, S., McKenny, A., and Sorenson, D., LAPACK User's Guide. 3rd ed., SIAM, Philadelphia, PA., 1999*

[3] *Netlib Repository: http://www.netlib.org/lapack*

Functionality

Structure

- Intel MKL follows NetLib functional structure:
  - High-level effective algorithms (LAPACK)[2]
  - High-performance computational kernels (BLAS - Basic Linear Algebra Subprograms)[4]
- Both parts are optimized on



Dense Linear Algebra Functional Structure

[2] *Anderson, E., Bai, Z., Bischof, C., Blackford, L. S., Demmel, J., Dongarra, J., DuCroz, J., Greenbaum, A., Hammarling, S., McKenny, A., and Sorenson, D., LAPACK User's Guide. 3rd ed., SIAM, Philadelphia, PA., 1999*

[4] *Dongarra, J., DuCroz, J., Duff, I., Hammarling, S., "A set of Level 3 Basic Linear Algebra Subprograms," Technical Report, ANL-MCS-TM-88, Argonne National Laboratory, Argonne, ILL, 1988*

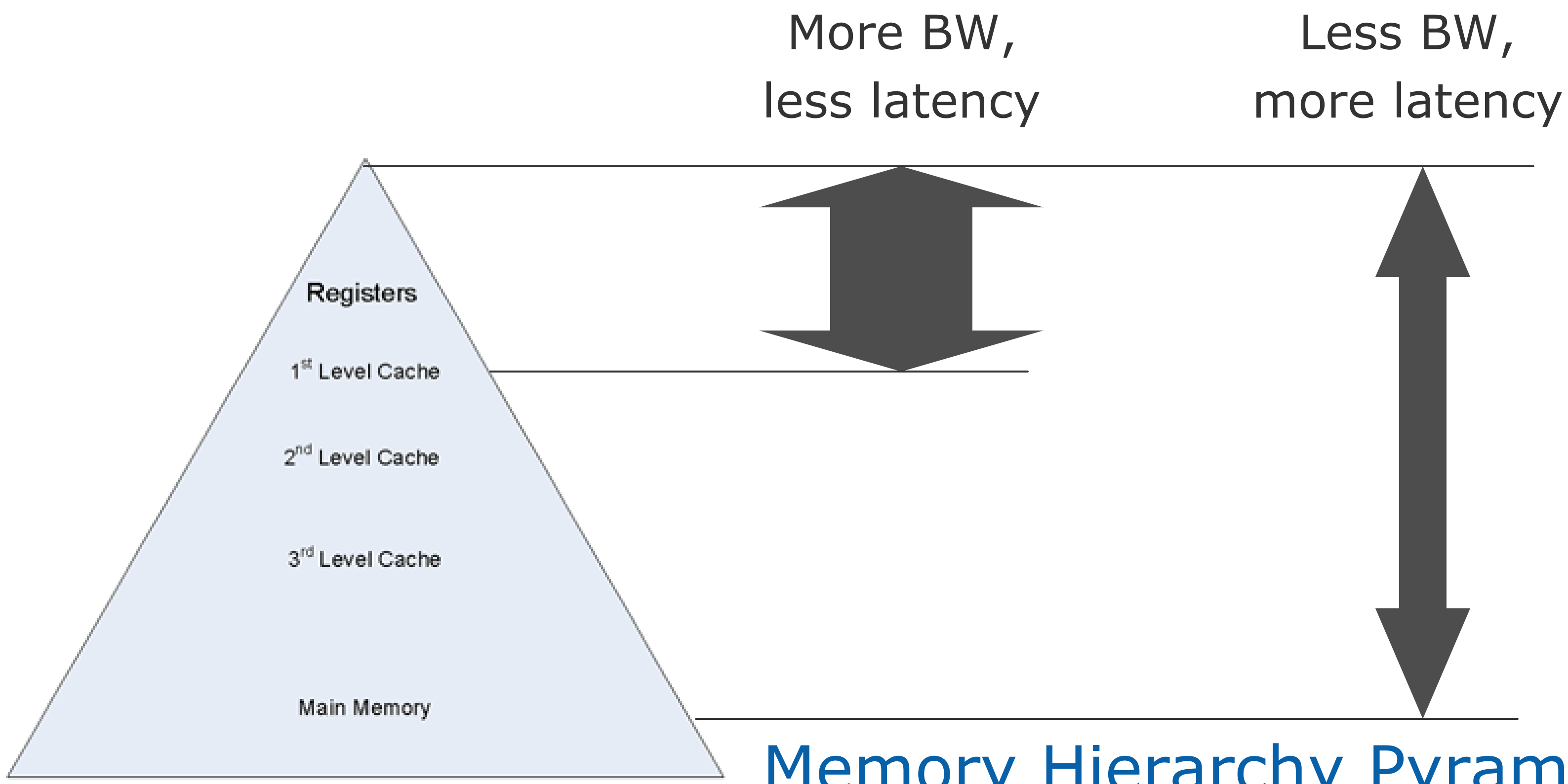
Matrix-Matrix Multiplication
Data Re-usage on Cache-based Systems

- There's a growing gap between computational speeds and the required memory performance to support it[5]
- Matrix-matrix multiplication (MMM) has a good ratio between data movement and floating point (FP) operations
  - $O(n^2)$ data movement
  - $O(n^3)$ floating point operations
- Data re-usage is possible on cache-based systems to make MMM running at CPU speed
- MMM can be threaded effectively on multi-core

[5] *Chuvelev, M., Greer, B., Henry, G., Kuznetsov, S., Burylov, I., Sabanin, B., "Intel® Performance Libraries: Multi-Core-Ready Software for Numeric-Intensive Computation." Intel Technology Journal.* http://www.intel.com/technology/itj/2007/v11i4/4-libraries/1-abstract.htm *(November 2007)*

Matrix-Matrix Multiplication

Memory Hierarchy Pyramid



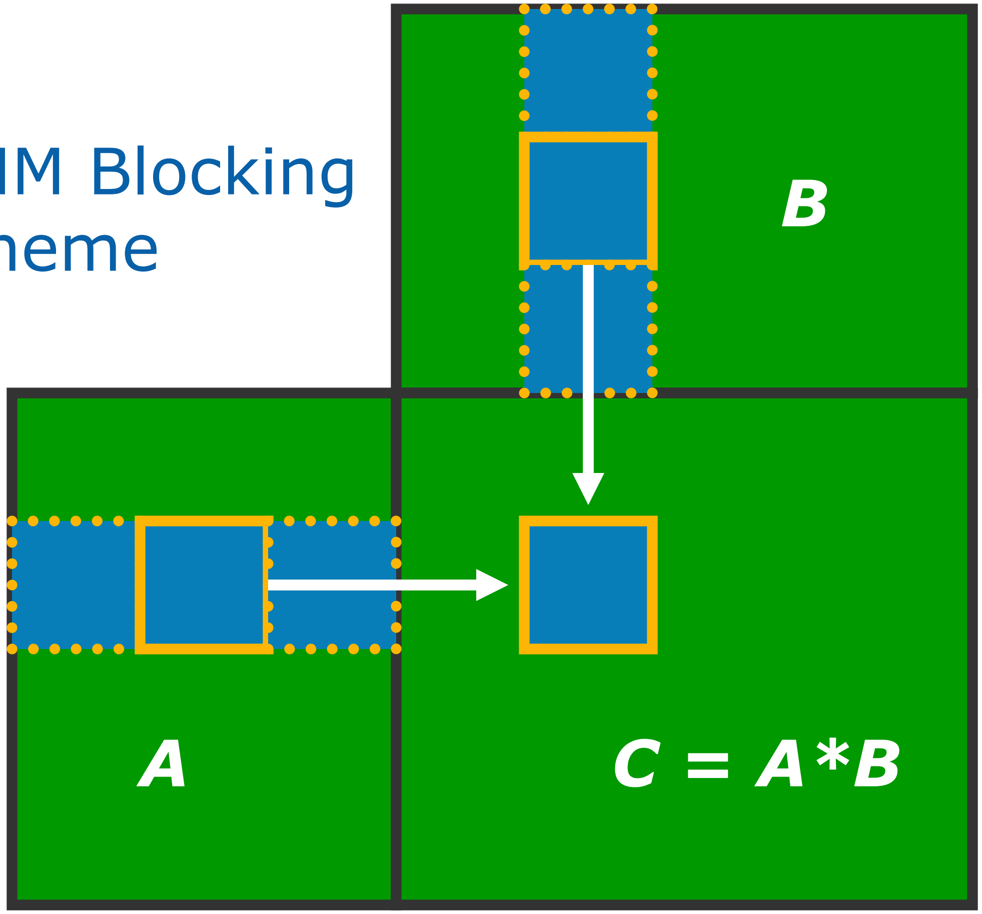
Memory Hierarchy Pyramid[5]

[5] *Chuvelev, M., Greer, B., Henry, G., Kuznetsov, S., Burylov, I., Sabanin, B., "Intel® Performance Libraries: Multi-Core-Ready Software for Numeric-Intensive Computation." Intel Technology Journal.* http://www.intel.com/technology/itj/2007/v11i4/4-libraries/1-abstract.htm *(November 2007).*

Matrix-Matrix Multiplication
Optimization Technique

- MMM optimization technique:
  - Blocking for data re-usage, depending on the cache characteristics
  - Low-level (processor-specific) state-of-art kernel fully utilizes FP unit
  - Threading

MMM Blocking Scheme



Matrix-Matrix Multiplication

Performance Model

- MMM (double precision) single-threaded efficiency peak is well described by the formula

$$E = 1 - \frac{16 \cdot P}{B} \cdot \sqrt{\frac{6}{L}}$$

97.7% on Intel® Xeon® Processor L5400 Series

where

E – efficiency (ratio of actual to system peak performance)

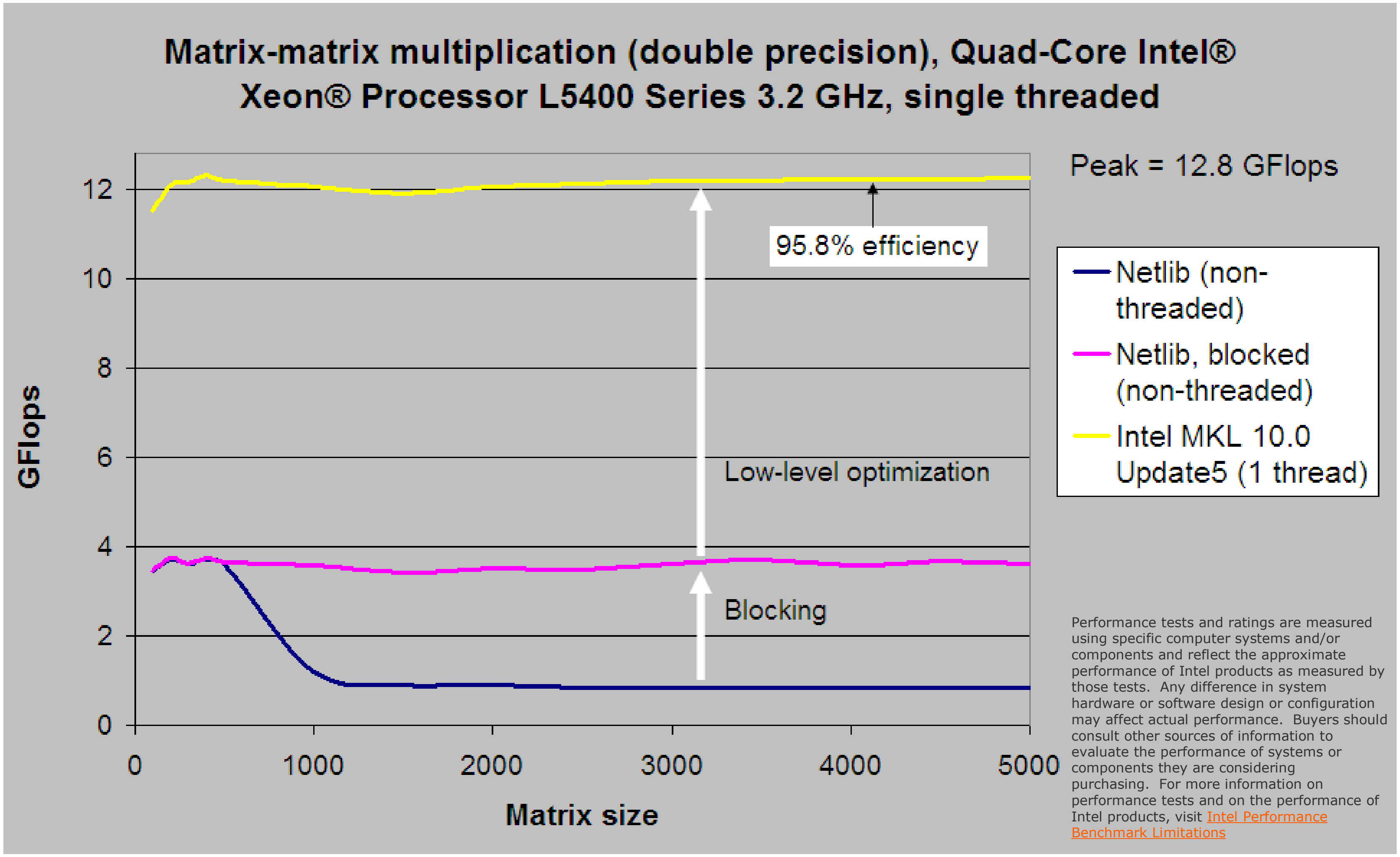
P – system peak performance (Floating-point operations per sec)

B – system memory bandwidth (Bytes per sec)

L – last level cache size (Bytes)

Matrix-Matrix Multiplication

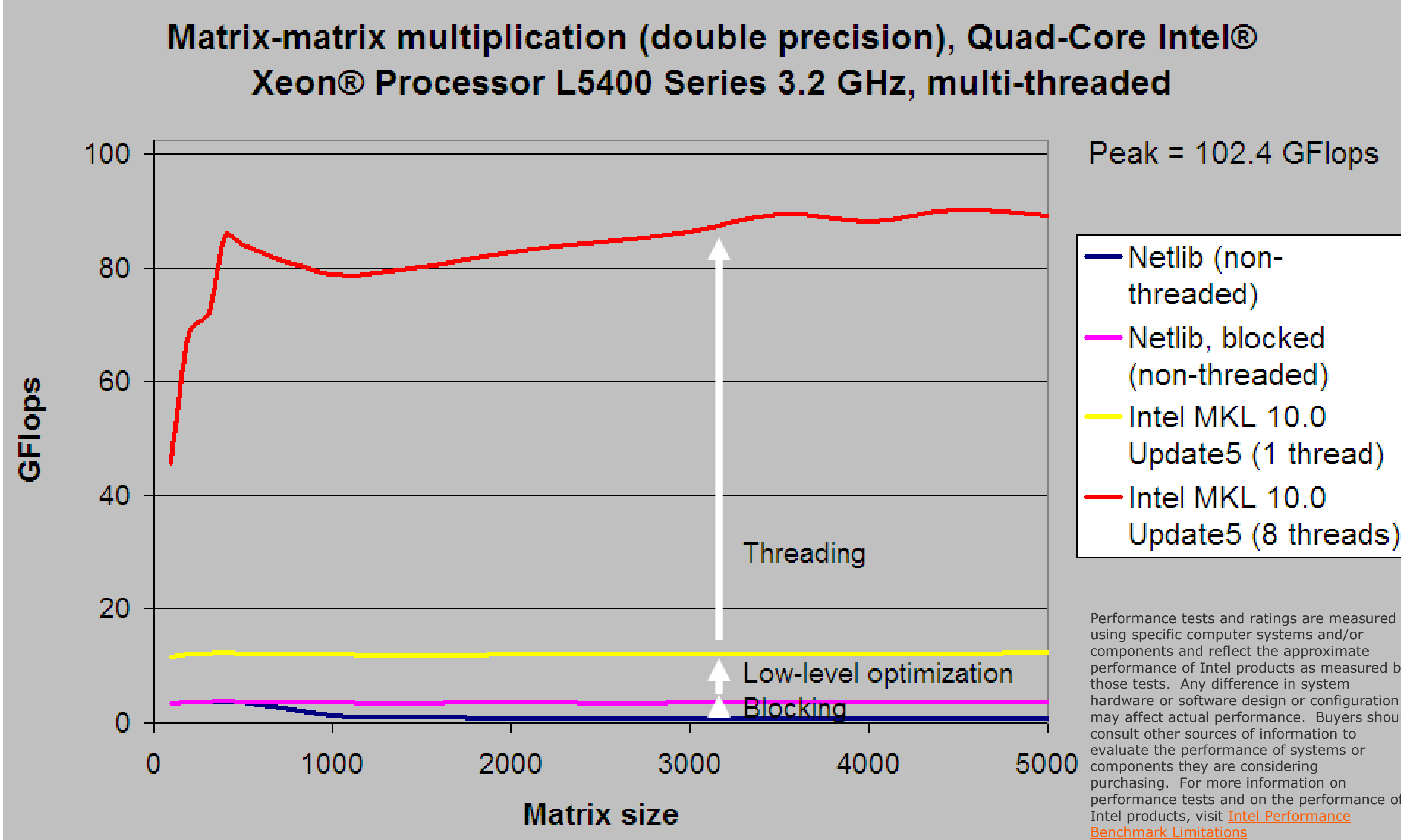
Single-core Performance



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, visit Intel Performance Benchmark Limitations

Matrix-Matrix Multiplication

Multi-core Performance



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, visit Intel Performance Benchmark Limitations

High-level Algorithms
Two Performance Factors

- Many Linear Algebra functions have the same order of data movement and FP operations as MMM:
  - $O(n^2)$ data movement
  - $O(n^3)$ floating point operations
- Two factors giving performance advantage:
  - Algorithms based on Matrix-matrix multiplication take advantage over vector operation based algorithms
  - Algorithms may rely on BLAS or LAPACK-level parallelism, threading at the highest level gets higher performance (NetLib LAPACK code[3] isn't threaded, may get use of BLAS parallelism only)
- Intel MKL[1] has a set of functionality optimized on LAPACK level

[1] *Intel® Math Kernel Library*, http://www.intel.com/software/products/mkl
[3] *Netlib Repository:* http://www.netlib.org/lapack

High-level Algorithms

Multiple Vector Rotation

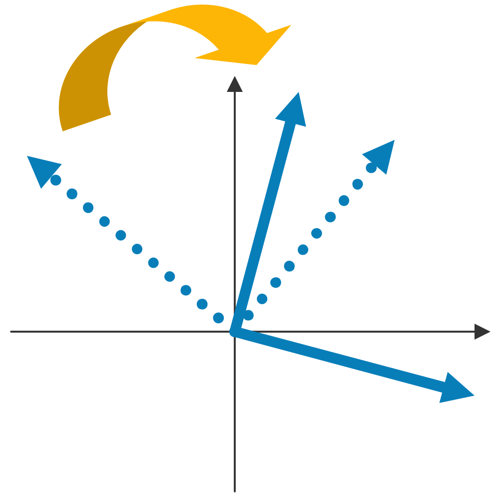
- Symmetric Eigenproblem

  $A = U^*\Lambda^*U'$

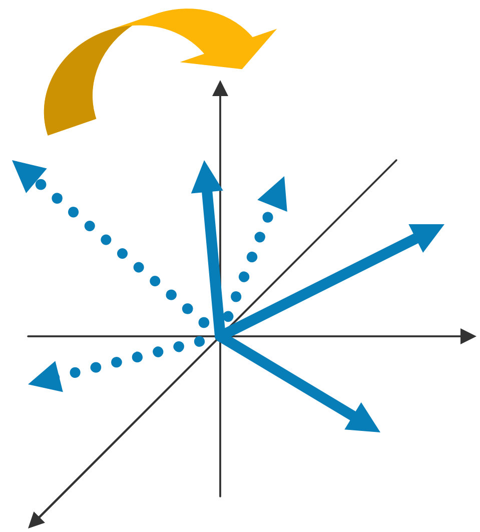
- Singular Value Decomposition

  $A = U^*\Sigma^*V'$

- Traditional QR algorithm[6] uses plane rotations – slow on memory
- Multiple vector rotation (MMM based) algorithm[7] outperforms plane rotations



Plane rotation



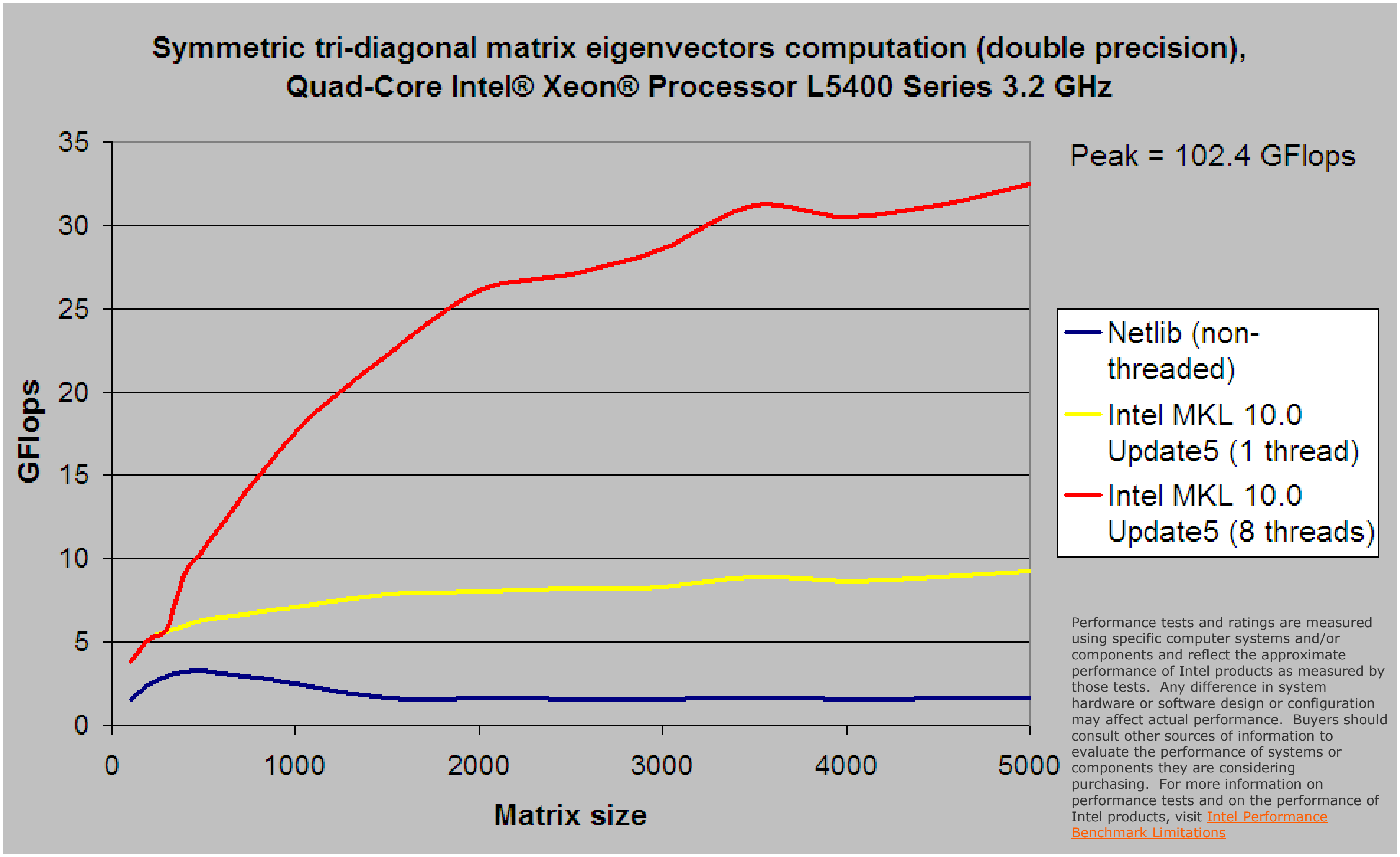
Multiple vector rotation

[6] *Lloyd N. Trefethen, David Bau, III, "Numerical Linear Algebra," SIAM, 1997*

[7] *Lang, B., "Using Level 3 BLAS in Rotation-Based Algorithms," SIAM Journal on Scientific Computing, Volume 19, Number 2, pp. 626–634, 1998*

High-level Algorithms

Multiple Vector Rotation Performance



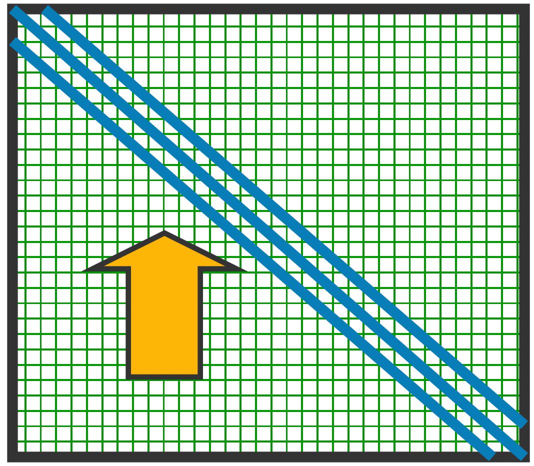
High-level Algorithms

Successive Band Reduction

- Reduction to tri-diagonal form:

  A = U*T*U′

- Traditional one-step reduction[6] requires $O(n^3)$ matrix-vector operations – slow on memory
- Two-step reduction[8] (MMM based) requires only $O(n^2)$ matrix-vector operations

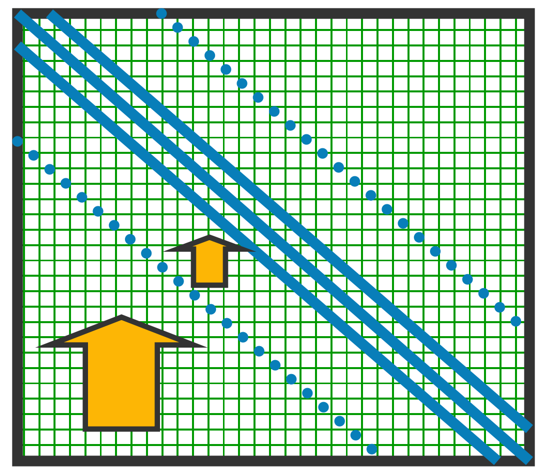


One-step Reduction

Two-step Reduction:

1) full-to-banded;

2) banded-to-trid.

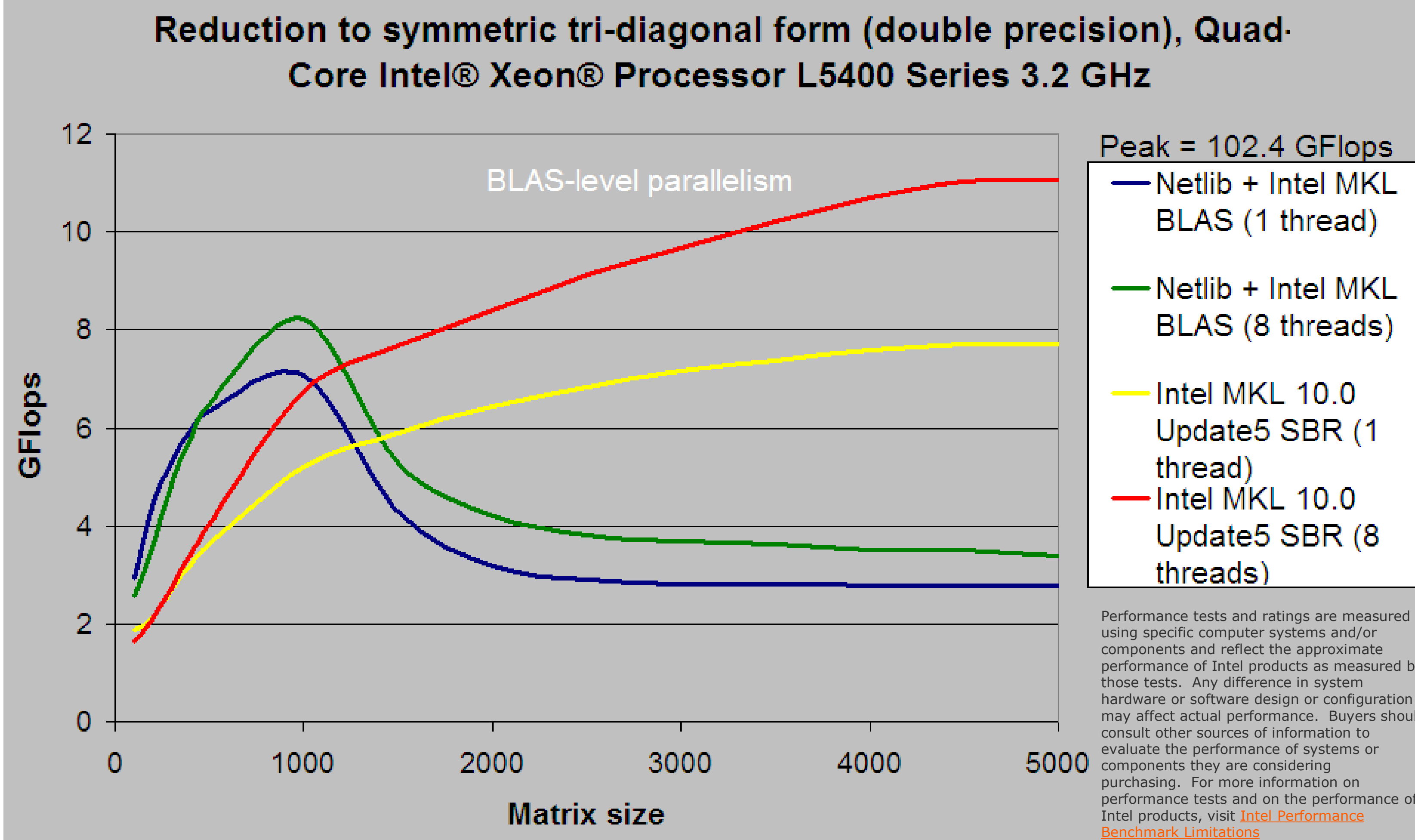


[6] *Lloyd N. Trefethen, David Bau, III, "Numerical Linear Algebra," SIAM, 1997*

[8] *Bischof, C., Lang, B., Sun, X., "Parallel tridiagonalization through two-step band reduction." In Proceedings of the Conference on Scalable High-Performance Computing (Washington, D.C.). IEEE Press, Piscataway, NJ, 23-27, 1994*

High-level Algorithms

Successive Band Reduction Performance



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, visit Intel Performance Benchmark Limitations

High-level Algorithms

DAG Technique

- LU/QR factorization:

  A = P*L*U

  A = Q*R

- Traditional algorithm relies on BLAS-level parallelism[9], therefore includes sequential parts and requires too many synchronizations
- DAG (Direct Acyclic Graph) based algorithm[10,11] is threaded on LAPACK level, has better workload balance, much more effective on multi-core

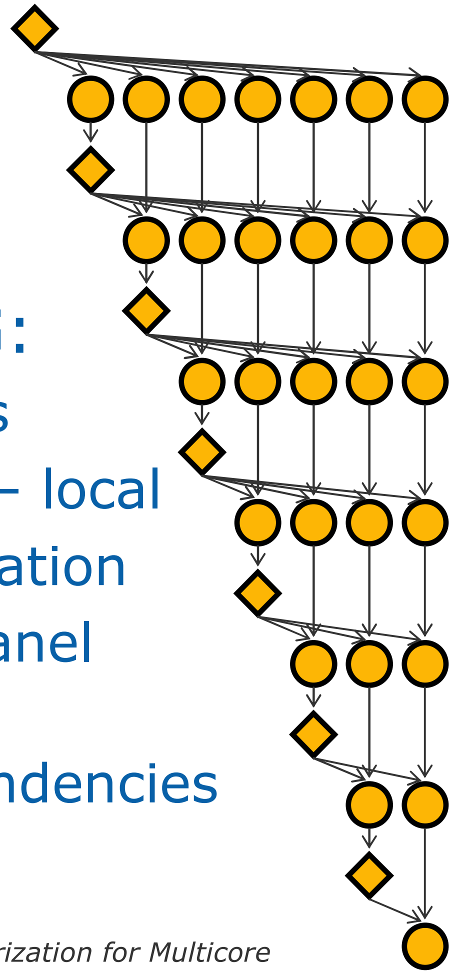
LU/QR DAG:

Nodes – tasks

1) diamonds – local panel factorization

2) circles – panel updates

Edges - dependencies

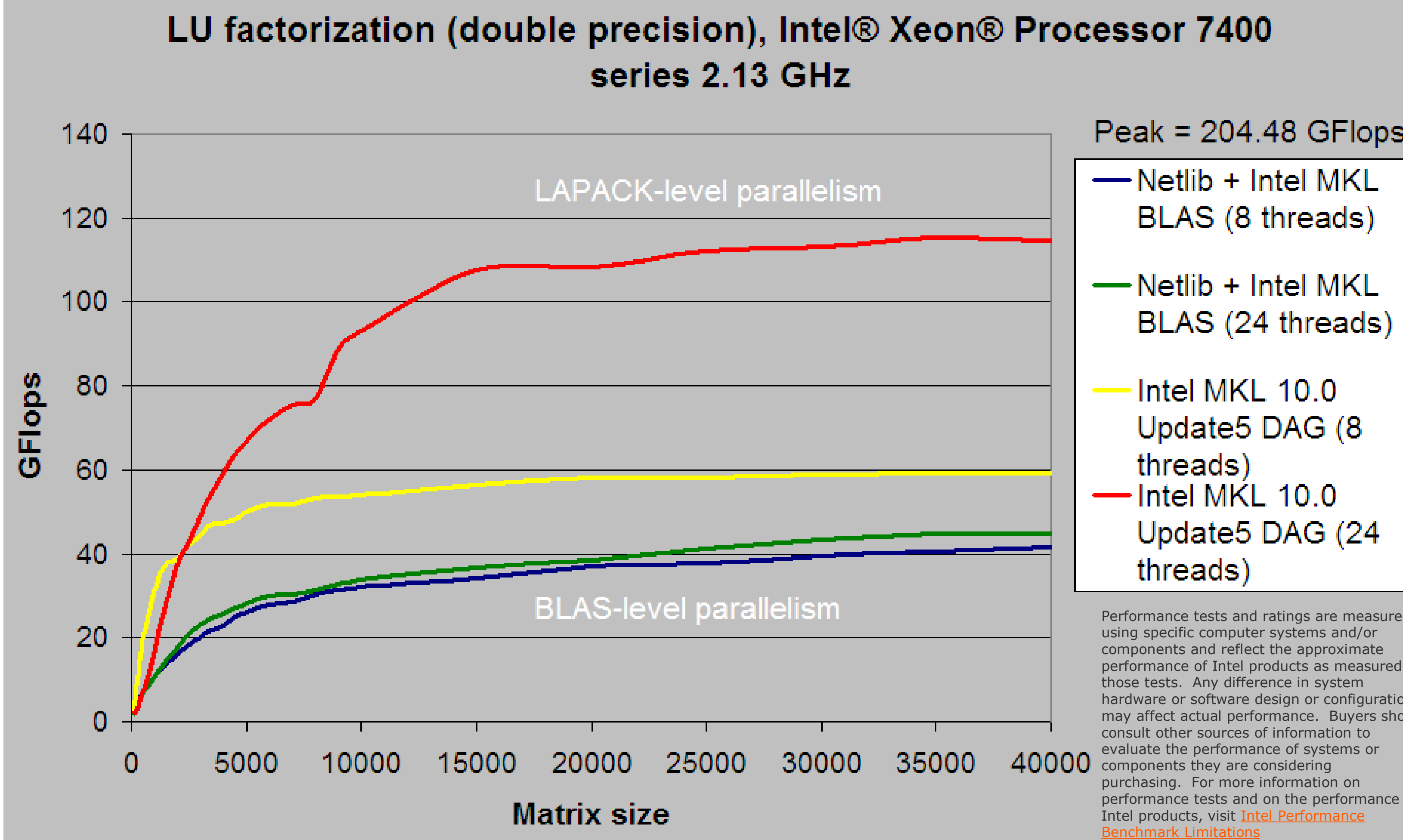


[9] *James W. Demmel, "Applied Numerical Linear Algebra," SIAM, 1997*

[10] *Alfredo Buttari, Julien Langou, Jakub Kurzak, Jack Dongarra, "Parallel Tiled QR Factorization for Multicore Architectures," 2007* http://www.netlib.org/lapack/lawnspdf/lawn190.pdf

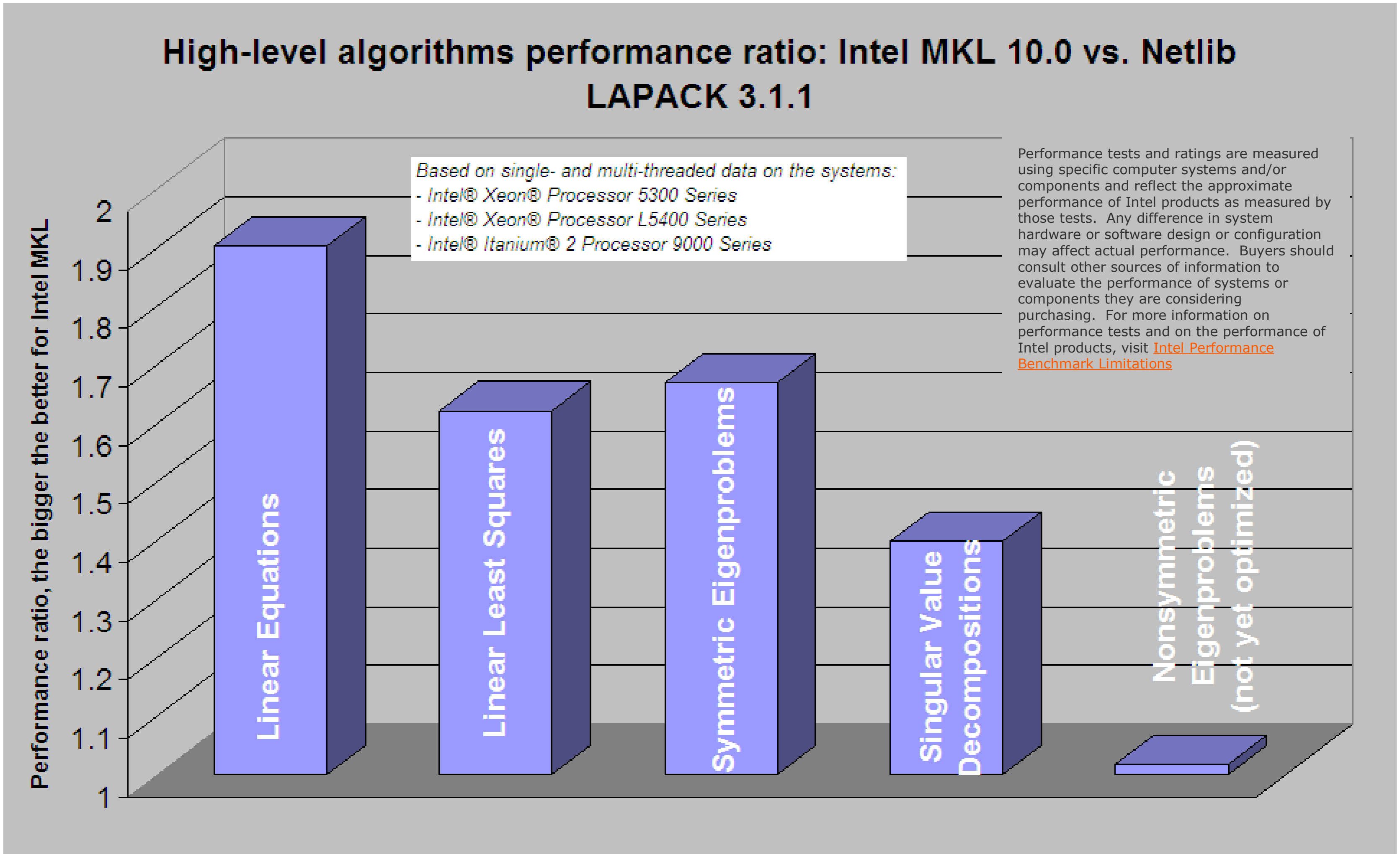
[11] *Jack Dongarra, "An Overview of High Performance Computing and Challenges for the Future," 2007,* http://www.cresco.enea.it/Documenti/web/presentazioni/Jack-Dongarra-0907.ppt

High-level Algorithms
DAG Technique Performance



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, visit Intel Performance Benchmark Limitations

High-level Algorithms
Performance Summary



Summary

- Two-level design (LAPACK-BLAS) defines high-level algorithmic and low-level processor-specific parts
- Matrix-matrix multiplication (MMM) gets nearly peak performance on the cache-based systems
- Using MMM is a key performance factor for high-level algorithms
- Threading at the highest level is very effective on multi-core

References

[1] *Intel® Math Kernel Library, http://www.intel.com/software/products/mkl*

[2] *Anderson, E., Bai, Z., Bischof, C., Blackford, L. S., Demmel, J., Dongarra, J., DuCroz, J., Greenbaum, A., Hammarling, S., McKenny, A., and Sorenson, D., LAPACK User's Guide. 3rd ed., SIAM, Philadelphia, PA., 1999*

[3] *Netlib Repository: http://www.netlib.org/lapack*

[4] *Dongarra, J., DuCroz, J., Duff, I., Hammarling, S., "A set of Level 3 Basic Linear Algebra Subprograms," Technical Report, ANL-MCS-TM-88, Argonne National Laboratory, Argonne, ILL, 1988*

[5] *Chuvelev, M., Greer, B., Henry, G., Kuznetsov, S., Burylov, I., Sabanin, B., "Intel® Performance Libraries: Multi-Core-Ready Software for Numeric-Intensive Computation." Intel Technology Journal. http://www.intel.com/technology/itj/2007/v11i4/4-libraries/1-abstract.htm (November 2007)*

[6] *Lloyd N. Trefethen, David Bau, III, "Numerical Linear Algebra," SIAM, 1997*

[7] *Lang, B., "Using Level 3 BLAS in Rotation-Based Algorithms," SIAM Journal on Scientific Computing, Volume 19, Number 2, pp. 626–634, 1998*

[8] *Bischof, C., Lang, B., Sun, X., "Parallel tridiagonalization through two-step band reduction." In Proceedings of the Conference on Scalable High-Performance Computing (Washington, D.C.). IEEE Press, Piscataway, NJ, 23-27, 1994*

[9] *James W. Demmel, "Applied Numerical Linear Algebra," SIAM, 1997*

[10] *Alfredo Buttari, Julien Langou, Jakub Kurzak, Jack Dongarra, "Parallel Tiled QR Factorization for Multicore Architectures," 2007 http://www.netlib.org/lapack/lawnspdf/lawn190.pdf*

[11] *Jack Dongarra, "An Overview of High Performance Computing and Challenges for the Future," 2007, http://www.cresco.enea.it/Documenti/web/presentazioni/Jack-Dongarra-0907.ppt*