Dense Linear Algebra on Heterogeneous Platforms: State of the Art and Trends

Paolo Bientinesi

AICES, RWTH Aachen
pauldj@aices.rwth-aachen.de

ComplexHPC Spring School 2013
Heterogeneous computing - Impact on algorithms
June 7th, 2013
Uppsala University, Sweden

RWTH AACHEN UNIVERSITY

AI ces

Deutsche Forschungsgemeinschaft DFG

1. Setting the stage
2. Part 1: blocked algorithms
3. Part 2: multithreading, fork-join
4. Part 3: multihtreading, algorithms-by-blocks
5. Part 4: streaming

Dense Linear Algebra

$$M = \begin{bmatrix} \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \end{bmatrix}$$

Dense Linear Algebra

$$M = \begin{bmatrix}
\times & \times & \times & & \times & & \times & \times & \times & & \times & \\
\times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
& \times & \times & \times & \times & \times & & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
\times & \times & \times & & \times & \times & \times & \times & \times & & \times & \times \\
\times & \times & \times & \times & & \times & \times & \times & \times & \times & & \times \\
\times & \times & \times & & \times & & \times & \times & \times & & \times & \\
\times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
& \times & \times & \times & \times & \times & & \times & \times & \times & \times & \times \\
\times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
\times & \times & \times & & \times & \times & \times & \times & \times & & \times & \times \\
\times & \times & \times & \times & & \times & \times & \times & \times & \times & & \times
\end{bmatrix}$$

Dense Linear Algebra

$$M = \begin{bmatrix}
\times & \times & & & \times & & \times & & \times & & \times & \\
\times & & \times & & \times & \times & \times & & \times & \times & & \times \\
& & \times & \times & & & & \times & & \times & \times & \\
& \times & & \times & \times & & & & \times & \times & & \times \\
& \times & & & \times & & \times & \times & & & \times & \\
\times & & & & & \times & \times & \times & & \times & & \times \\
\times & & \times & & \times & & \times & & \times & & \times & \\
\times & \times & & \times & \times & \times & \times & \times & & & \times & \times \\
& \times & & & & \times & & & & & \times & \times \\
& \times & \times & \times & & \times & & \times & \times & \times & & \\
\times & & \times & & \times & & \times & \times & & & \times & \times \\
\times & & \times & \times & & \times & & & \times & \times & & \times
\end{bmatrix}$$

Dense Linear Algebra

$$M = \begin{bmatrix}
\times & & & & & & & & & & & \\
\times & \times & & & & & & & & & & \\
\times & \times & \times & & & & & & & & & \\
\times & \times & \times & \times & & & & & & & & \\
 & \times & \times & \times & \times & & & & & & & \\
\times & \times & \times & \times & & \times & & & & & & \\
\times & \times & \times & \times & \times & \times & \times & & & & & \\
\times & \times & & \times & \times & \times & \times & \times & & & & \\
\times & \times & \times & \times & \times & \times & \times & \times & \times & & & \\
\times & \times & \times & \times & \times & \times & \times & & & \times & & \\
 & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \\
\times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times
\end{bmatrix}$$

Dense Linear Algebra

$$M = \left[\begin{array}{cccccccccccc}
\times & \times & & & & & & & & & & \\
\times & \times & \times & & & & & & & & & \\
 & \times & \times & \times & & & & & & & & \\
 & & \times & \times & \times & & & & & & & \\
 & & & \times & \times & \times & & & & & & \\
 & & & & \times & \times & \times & & & & & \\
 & & & & & \times & \times & \times & & & & \\
 & & & & & & \times & \times & \times & & & \\
 & & & & & & & \times & \times & \times & & \\
 & & & & & & & & \times & \times & \times & \\
 & & & & & & & & & \times & \times & \times \\
 & & & & & & & & & & \times & \times
\end{array}\right]$$

Dense **Linear Algebra**

Linear systems

$Ax = b$, $AX = B$, least squares, ...

Eigenproblems

$Ax = \lambda x$, $AX = BX\Lambda$, SVD, ...

Dense **Linear Algebra**

Linear systems

$Ax = b$, $AX = B$, least squares, ...

Eigenproblems

$Ax = \lambda x$, $AX = BX\Lambda$, SVD, ...

Support routines

factorizations, reductions, ...

Dense **Linear Algebra**

Matrix equations

$AX + XB = C$, $A = \frac{A+A^{-1}}{2}$, ...

Linear systems

$Ax = b$, $AX = B$, least squares, ...

Eigenproblems

$Ax = \lambda x$, $AX = BX\Lambda$, SVD, ...

Support routines

factorizations, reductions, ...

Organization in layers

Organization in layers

BLAS

Organization in layers

BLAS

BLAS-1: $y := y + \alpha x \qquad x, y \in \mathbb{R}^n,\ \alpha \in R$

$dot := \alpha + x^T y$

Organization in layers

BLAS

BLAS-2: $y := y + Ax$ $A, L \in R^{n \times n},\; x, y \in R^n$

$y := L^{-1}x$

BLAS-1: $y := y + \alpha x$ $x, y \in \mathbb{R}^n,\; \alpha \in R$

$dot := \alpha + x^T y$

Organization in layers

BLAS

BLAS-3:	$C := C + AB$ $C := L^{-1}B$	$A, B, C, L \in R^{n \times n}$
BLAS-2:	$y := y + Ax$ $y := L^{-1}x$	$A, L \in R^{n \times n},\ x, y \in R^n$
BLAS-1:	$y := y + \alpha x$ $dot := \alpha + x^T y$	$x, y \in \mathbb{R}^n,\ \alpha \in R$

Organization in layers

LAPACK

$$LU = A, \quad LL^T = A, \quad QR = A, \quad Q^T T Q = A, \quad \ldots$$

BLAS

BLAS-3:	$C := C + AB$ $C := L^{-1} B$	$A, B, C, L \in R^{n \times n}$
BLAS-2:	$y := y + Ax$ $y := L^{-1} x$	$A, L \in R^{n \times n}, \; x, y \in R^n$
BLAS-1:	$y := y + \alpha x$ $dot := \alpha + x^T y$	$x, y \in \mathbb{R}^n, \; \alpha \in R$

Organization in layers

other libraries

ScaLAPACK, Elemental, PETSc, . . .

LAPACK

$LU = A, \quad LL^T = A, \quad QR = A, \quad Q^T TQ = A, \quad \ldots$

BLAS

BLAS-3: $C := C + AB$ $\quad A, B, C, L \in R^{n \times n}$

$C := L^{-1}B$

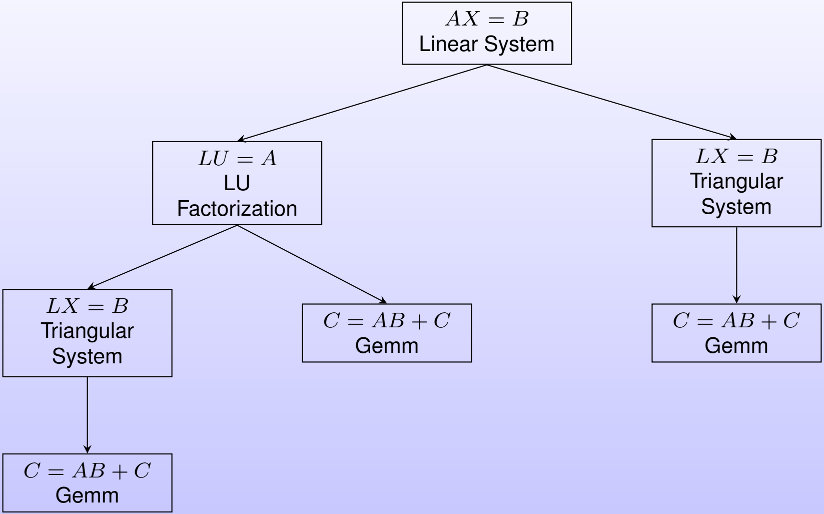
BLAS-2: $y := y + Ax$ $\quad A, L \in R^{n \times n},\ x, y \in R^n$

$y := L^{-1}x$

BLAS-1: $y := y + \alpha x$ $\quad x, y \in \mathbb{R}^n,\ \alpha \in R$

$dot := \alpha + x^T y$

Example: $AX = B$ (full A)



Why BLAS-3? Why GEMM?

Why BLAS-3? Why GEMM?

BLAS	#FLOPS	Mem. refs.	Ratio

BLAS-1: $y := y + \alpha x$ $x, y \in \mathbb{R}^n, \ \alpha \in R$

Why BLAS-3? Why GEMM?

BLAS	#FLOPS	Mem. refs.	Ratio
Level 1	$2n$	$3n$	$2/3$

BLAS-1: $y := y + \alpha x \qquad x, y \in \mathbb{R}^n, \ \alpha \in R$

Why BLAS-3? Why GEMM?

BLAS	#FLOPS	Mem. refs.	Ratio
Level 1	$2n$	$3n$	$2/3$

BLAS-2: $y := y + Ax$ $\quad A \in R^{n \times n},\ x, y \in R^n$

BLAS-1: $y := y + \alpha x$ $\quad x, y \in \mathbb{R}^n,\ \alpha \in R$

Why BLAS-3? Why GEMM?

BLAS	`#FLOPS`	Mem. refs.	Ratio
Level 2	$2n^2$	n^2	2
Level 1	$2n$	$3n$	$2/3$

BLAS-2: $y := y + Ax$ $\quad A \in R^{n \times n},\ x, y \in R^n$

BLAS-1: $y := y + \alpha x$ $\quad x, y \in \mathbb{R}^n,\ \alpha \in R$

Why BLAS-3? Why GEMM?

BLAS	#FLOPS	Mem. refs.	Ratio
Level 2	$2n^2$	n^2	2
Level 1	$2n$	$3n$	$2/3$

BLAS-3: $C := C + AB$ $\quad A, B, C, \in R^{n \times n}$

BLAS-2: $y := y + Ax$ $\quad A \in R^{n \times n},\ x, y \in R^n$

BLAS-1: $y := y + \alpha x$ $\quad x, y \in \mathbb{R}^n,\ \alpha \in R$

Why BLAS-3? Why GEMM?

BLAS	#FLOPS	Mem. refs.	Ratio
Level 3	$2n^3$	$4n^2$	$\boldsymbol{n/2}$
Level 2	$2n^2$	n^2	2
Level 1	$2n$	$3n$	$2/3$

BLAS-3: $C := C + AB$ $\quad A, B, C, \in R^{n \times n}$

BLAS-2: $y := y + Ax$ $\quad A \in R^{n \times n},\ x, y \in R^n$

BLAS-1: $y := y + \alpha x$ $\quad x, y \in \mathbb{R}^n,\ \alpha \in R$

Why BLAS-3? Why GEMM?

BLAS	#FLOPS	Mem. refs.	Ratio
Level 3	$2n^3$	$4n^2$	$\boldsymbol{n/2}$
Level 2	$2n^2$	n^2	2
Level 1	$2n$	$3n$	$2/3$

BLAS-3: $C := C + AB$ $\quad A, B, C, \in R^{n\times n}$

BLAS-2: $y := y + Ax$ $\quad A \in R^{n\times n},\ x, y \in R^n$

BLAS-1: $y := y + \alpha x$ $\quad x, y \in \mathbb{R}^n,\ \alpha \in R$

Morale *BLAS-3: The larger the problem the better, as long as it fits in memory.*
GEMM is the building block for all the other BLAS-3 kernels, and for LAPACK.

Part 1: Blocked algorithms

Simple example: Cholesky factorization

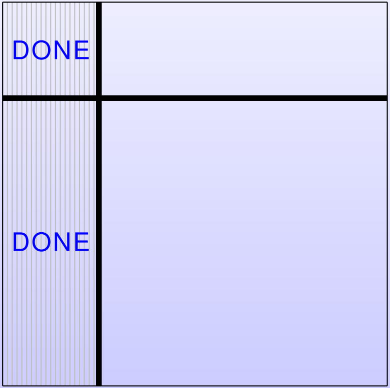
Input: Matrix A, symmetric and positive definite.

Goal: Determine L (lower triangular matrix) such that $LL^T = A$

$$L = \left(\begin{array}{c|c} L_{TL} & \\ \hline L_{BL} & L_{BR} \end{array} \right) = ?$$

Cholesky factorization

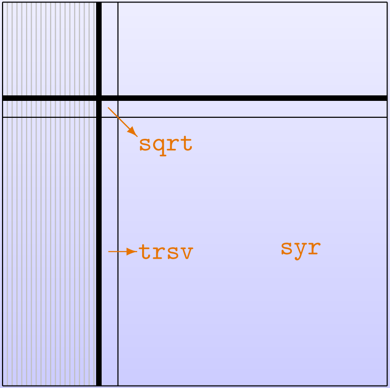
iteration i



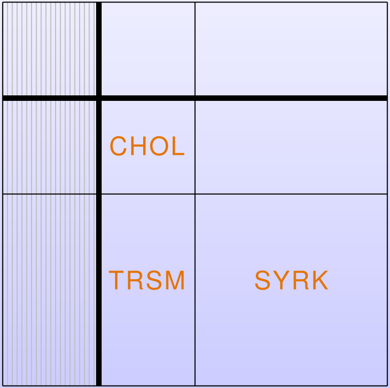
Cholesky factorization

iteration $i + 1$

unblocked algorithm



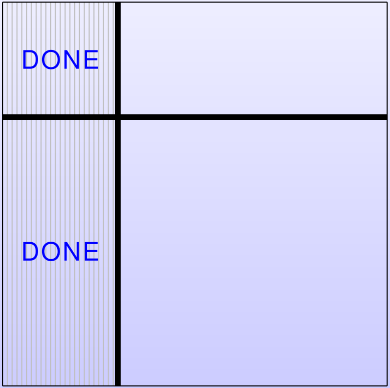
blocked algorithm



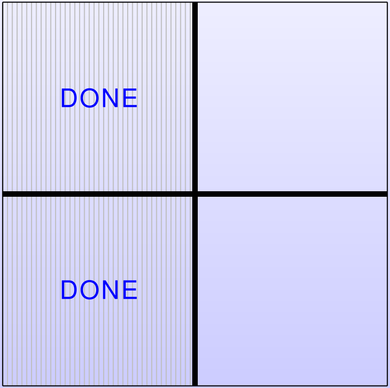
Cholesky factorization

iteration $i + 1$

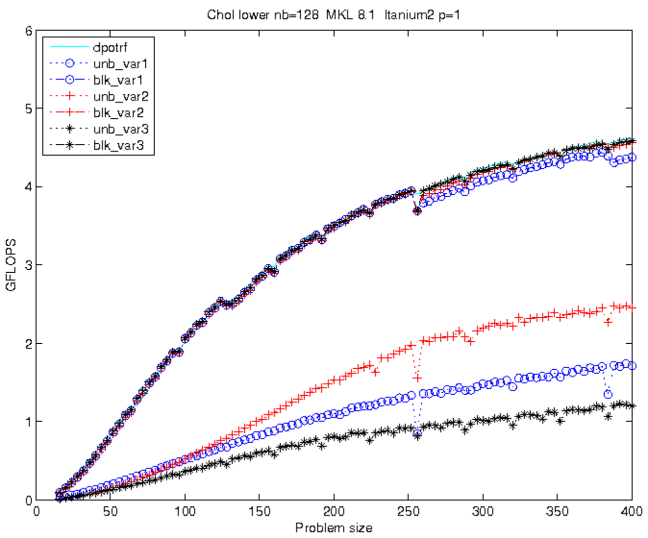
unblocked algorithm



blocked algorithm

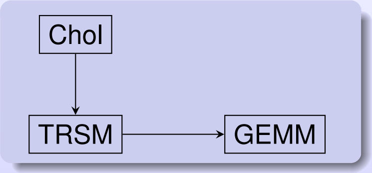


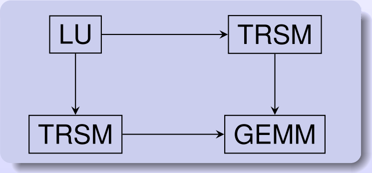
Cholesky: unblocked vs. blocked algorithms



Part 2: Parallelism? fork-join

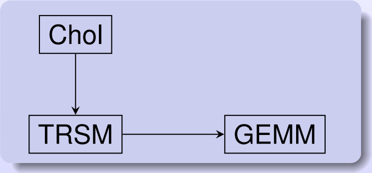
- Solution #1: Multithreaded BLAS (+ vector instructions)

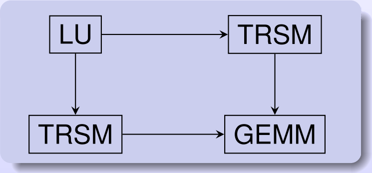




Part 2: Parallelism? fork-join

- Solution #1: Multithreaded BLAS (+ vector instructions)

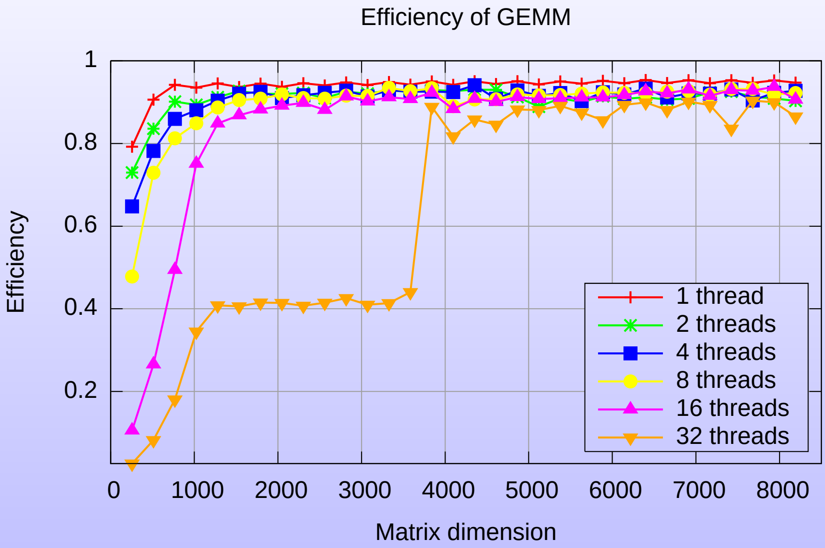




- Advantage: ease of use. Legacy code!
- Drawback: unnecessary synchronization points
- OpenBLAS, ATLAS, BLIS, old versions of MKL, ...

Multithreaded BLAS

Xeon, 32 physical cores, MKL



Development of LA libraries

- New architecture / new architectural features

Development of LA libraries

- New architecture / new architectural features
- GEMM

Development of LA libraries

- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]

Development of LA libraries

- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]
- BLAS3, factorizations, AX=B

Development of LA libraries

- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]
- BLAS3, factorizations, AX=B $\longrightarrow$ LINPACK benchmark

Development of LA libraries

- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]
- BLAS3, factorizations, AX=B $\longrightarrow$ LINPACK benchmark
- BLAS3, factorizations, AX=B

Development of LA libraries

- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]
- BLAS3, factorizations, AX=B $\longrightarrow$ LINPACK benchmark
- BLAS3, factorizations, AX=B
- factorizations, AX=B

Development of LA libraries

- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]
- BLAS3, factorizations, AX=B $\longrightarrow$ LINPACK benchmark
- BLAS3, factorizations, AX=B
- factorizations, AX=B
- factorizations, AX=B, support routines

Development of LA libraries

- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]
- BLAS3, factorizations, AX=B $\longrightarrow$ LINPACK benchmark
- BLAS3, factorizations, AX=B
- factorizations, AX=B
- factorizations, AX=B, support routines

⋮

Development of LA libraries

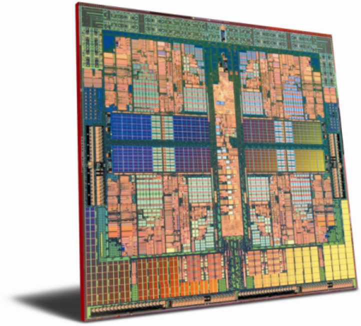
- New architecture / new architectural features
- GEMM $\rightarrow$ peak performance [FFT, SpMV]
- BLAS3, factorizations, AX=B $\longrightarrow$ LINPACK benchmark
- BLAS3, factorizations, AX=B
- factorizations, AX=B
- factorizations, AX=B, support routines

⋮

⋮

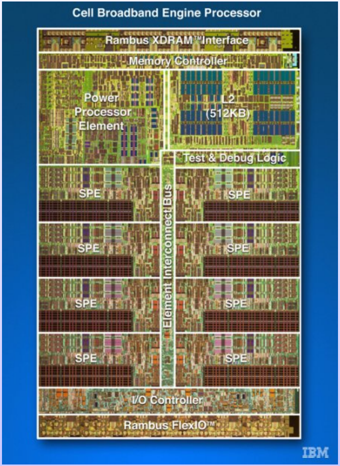
- eigenproblems

2005–2006: multicores



- GEMM, mtBLAS
- HPL
  2005-7: **Blue Gene L**
  dual cores (PowerPC A2)
  2008-9: **Roadrunner**
  dual cores (Opteron)
  2009: **Jaguar**
  6-cores (Opteron)
  2010-11: **K Computer**
  8-cores SPARC64
  ...
- libFLAME, PLASMA, MKL
- Eigensolvers: 2010

2005–2006: Cell

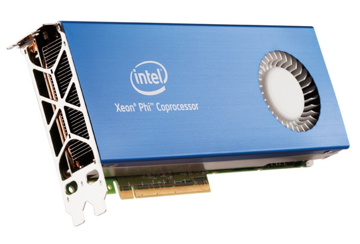


- GEMM: 99%   [FFT] no BLAS
- HPL
  2008-9: **Roadrunner**
  >1 PetaFLOP
  6k * (Opteron + 2 Cell)
- no libs
- 2009: discontinued
- Eigensolvers: –

2005: GPGPUs

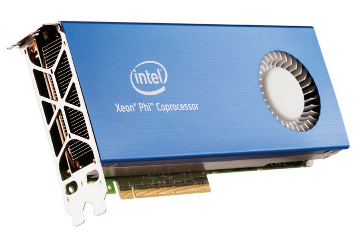
- cuBLAS (*)
- HPL 2010: **Tianhe-1A**
  2.5k dual-GPU (ATI Radeon)
- CULA (*)
- libFLAME, MAGMA (multiGPU)
- Eigensolvers: 2010-11

Intel Xeon Phi (MIC)



- GEMM (*), MKL
- HPL (predicted)
  2013: **Tianhe-2**
  16k * (2 Ivy Bridge + 3 Phi)
- -
- -
- -

Present Intel Xeon Phi (MIC)



- GEMM (*), MKL
- HPL (predicted)
  2013: **Tianhe-2**
  16k * (2 Ivy Bridge + 3 Phi)
- -
- -
- -

Also ARM, FPGAs, DSPs, . . .

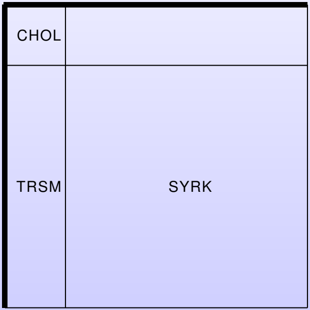
2017: Exascale (?)

???

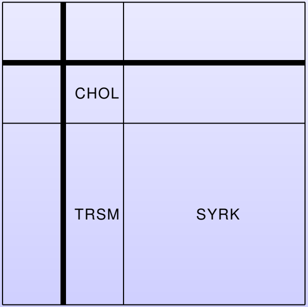
- heterogeneous architecture

Back to the algorithms: Chol. fact

Fork-join $\Rightarrow$ unnecessary synchronizations

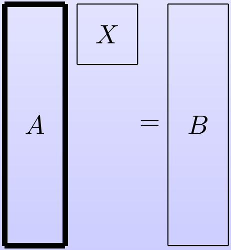


Iteration 1



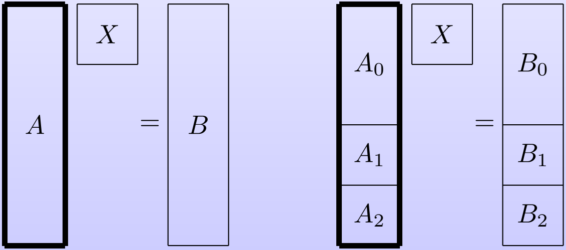
Iteration 2

- Idea: decompose the tasks
- Solution #2: dependent tasks + scheduling



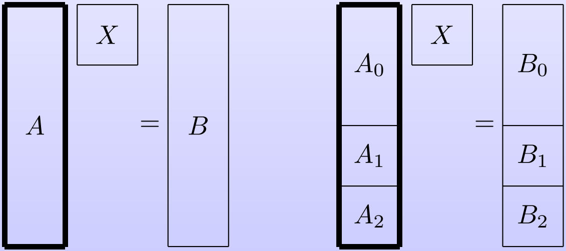
Part 3: Parallelism? algorithms-by-blocks

- Idea: decompose the tasks
- Solution #2: dependent tasks + scheduling



Part 3: Parallelism? algorithms-by-blocks

- Idea: decompose the tasks
- Solution #2: dependent tasks + scheduling

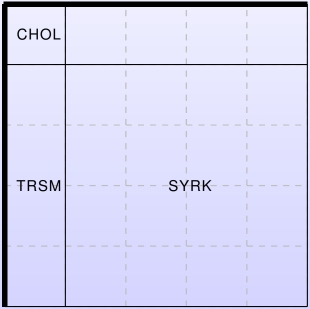


$$A_0 X = B_0$$

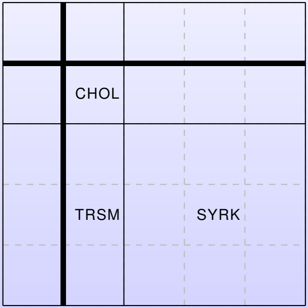
$$A_1 X = B_1$$

$$A_2 X = B_2$$

Storage by blocks



Iteration 1



Iteration 2

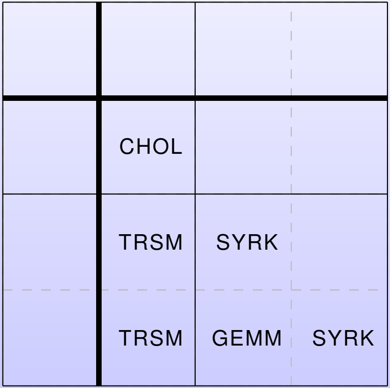
Creating tasks

Iteration 1

CHOL			
TRSM	SYRK		
TRSM	GEMM	SYRK	
TRSM	GEMM	GEMM	SYRK

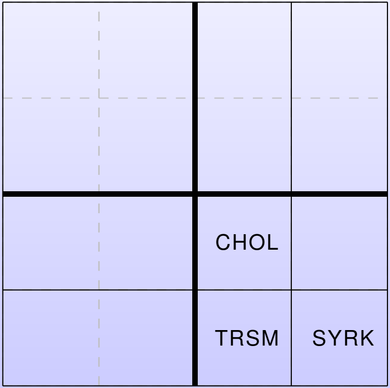
Creating tasks

Iteration 2



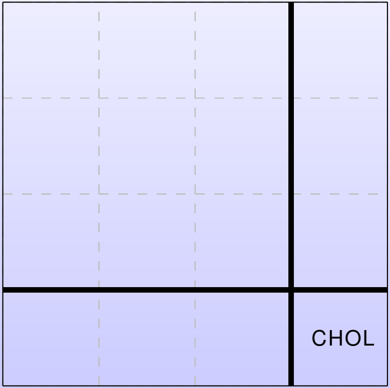
Creating tasks

Iteration 3



Creating tasks

Iteration 4



Dependencies

CHOL			
TRSM	SYRK		
TRSM	GEMM	SYRK	
TRSM	GEMM	GEMM	SYRK

Dependencies

CHOL			
TRSM	SYRK		
TRSM	GEMM	SYRK	
TRSM	GEMM	GEMM	SYRK

Dependencies

CHOL			
TRSM	SYRK		
TRSM	GEMM	SYRK	
TRSM	GEMM	GEMM	SYRK

Dependencies

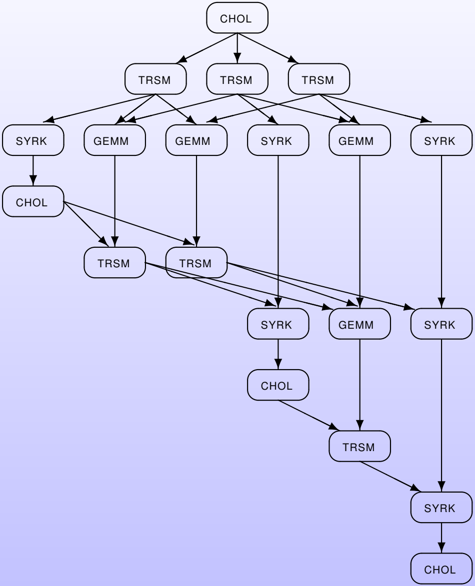
CHOL			
TRSM	SYRK		
TRSM	GEMM	SYRK	
TRSM	GEMM	GEMM	SYRK

Dependencies

CHOL			
TRSM	SYRK		
TRSM	GEMM	SYRK	
TRSM	GEMM	GEMM	SYRK

DAG - Dependencies

4×4-tile matrix



Task Execution

4×4-tile matrix

Stage	Scheduled Tasks			
1	CHOL			
2	TRSM	TRSM	TRSM	TRSM
3	SYRK	GEMM	SYRK	GEMM
4	GEMM	SYRK	GEMM	GEMM
5	GEMM	SYRK	CHOL	
6	TRSM	TRSM	TRSM	
7	SYRK	GEMM	SYRK	GEMM
8	GEMM	SYRK	CHOL	
9	TRSM	TRSM		
10	SYRK	GEMM	SYRK	
11	CHOL			
12	TRSM			
13	SYRK			
14	CHOL			

SPD Inverse: Chol+Inv+GEMM

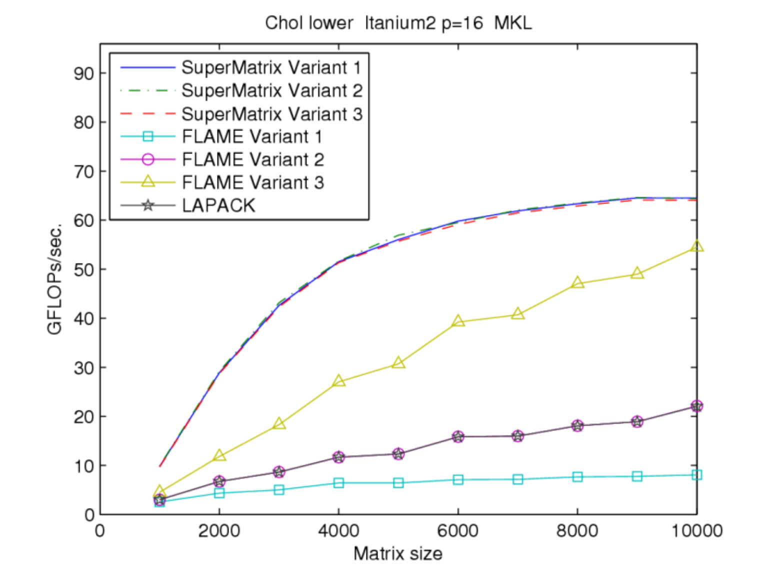
5×5-tile matrix

SPD Inverse: Chol+Inv+GEMM

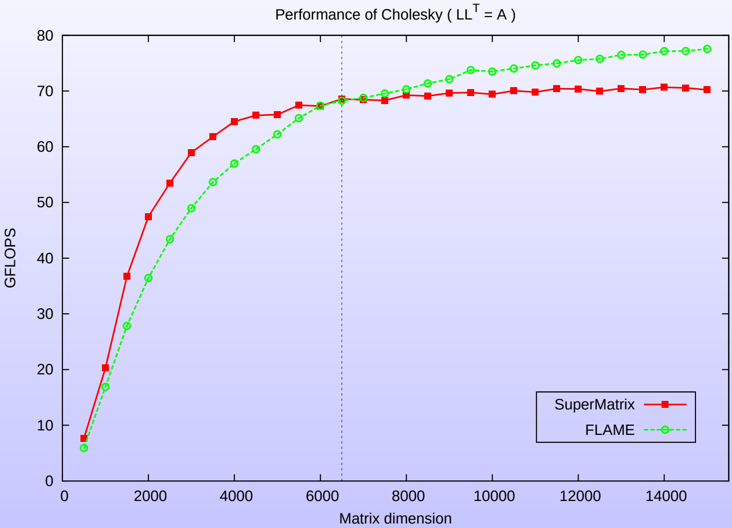
5×5-tile matrix

Stage	Scheduled Tasks			
1	CHOL			
2	TRSM	TRSM	TRSM	TRSM
3	SYRK	GEMM	SYRK	GEMM
4	GEMM	SYRK	GEMM	GEMM
5	GEMM	SYRK	CHOL	TRSM
6	TRSM	TRSM	TRSM	TRSM
7	TRSM	TRSM	TRINV	SYRK
8	GEMM	SYRK	GEMM	GEMM
9	SYRK	TTMM	CHOL	TRSM
10	TRSM	TRSM	TRSM	TRSM
11	GEMM	GEMM	GEMM	SYRK
12	GEMM	SYRK	TRSM	CHOL
13	TRSM	TRSM	TRINV	SYRK
14	TRSM	GEMM	GEMM	GEMM
15	GEMM	TRMM	SYRK	TRSM
16	TRSM	TTMM	CHOL	TRSM
17	SYRK	TRINV	GEMM	SYRK
18	GEMM	GEMM	GEMM	TRMM
19	TRMM	TRSM	TRSM	TRSM
20	TRSM	TRSM	TRSM	TRSM
21	TTMM	SYRK	GEMM	SYRK
22	TRINV	GEMM	GEMM	TRINV
23	SYRK	SYRK	GEMM	SYRK
24	TRMM	GEMM	TRMM	GEMM
25	TRMM	SYRK	GEMM	GEMM
26	TTMM	GEMM	TRMM	TRMM
27	SYRK	TRMM		
28	TRMM			
29	TTMM			

Blocked vs. By-blocks



Blocked vs. By-blocks: crossover



Multithreaded BLAS vs. Algorithms-by-blocks

No absolute winner: crossover!

✓ Ease of use	✓ Out of order execution
✖ Synchronization	✓ Parallelism dictated by data dependencies
	✖ Plateux

libFLAME, MKL, PLASMA, ...

Heterogeneous systems $\leftrightarrow$ schedulers

Part 4: Streaming

The entire problem does not fit even in main memory

Example $b := \left(X^T M^{-1} X\right)^{-1} X^T M^{-1} y$

The entire problem does not fit even in main memory

Example $b := \left(X^T M^{-1} X\right)^{-1} X^T M^{-1} y$

Linear regression with non-independent outcomes

Inputs

- $M \in \mathcal{R}^{n \times n}, \quad SPD(M), \quad n \in [10^3, \ldots, 10^4]$
- $X \in \mathcal{R}^{n \times p}, \quad p \in [1, \ldots, 20],$ full rank
- $y \in \mathcal{R}^n$

Output

- $b \in \mathcal{R}^p$

$\star$**Sequence of thousands of problems**$\star$

Algorithm

$$LL^T = M$$
$$X := L^{-1}X$$
$$S := X^TX$$
$$GG^T = S$$
$$y := L^{-1}y$$
$$b := X^Ty$$
$$b := G^{-1}b$$
$$b := G^{-T}b$$

Algorithm – bottleneck?

$LL^T = M$

$X := L^{-1}X$ → to accelerator (`TRSM`)

$S := X^T X$

$GG^T = S$

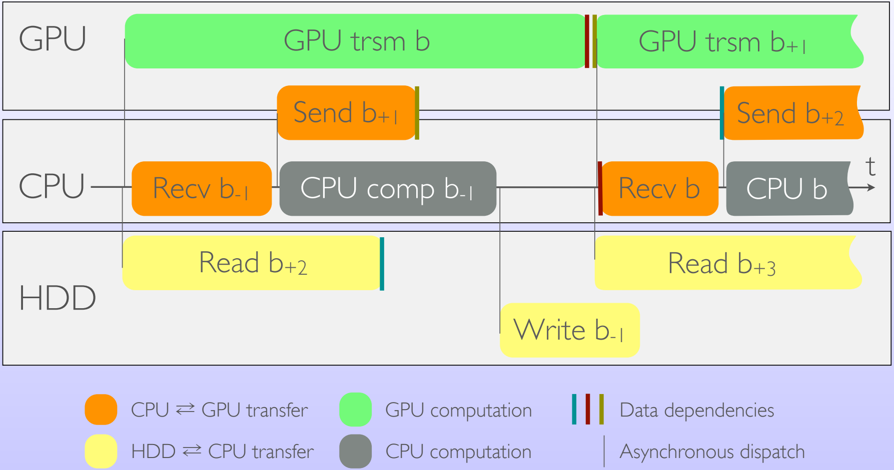
$y := L^{-1}y$

$b := X^T y$

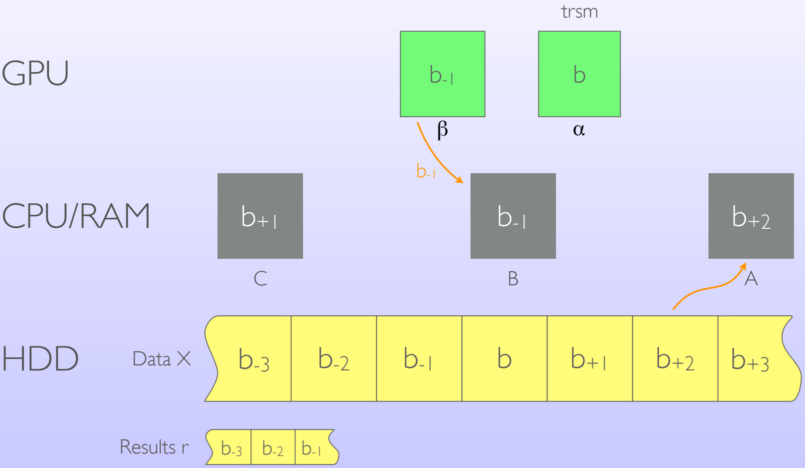
$b := G^{-1}b$

$b := G^{-T}b$

double+triple buffering



double+triple buffering



Summary

- Dense linear algebra: it's matter of layers
  GEMM = foundations, THE building block

- Irrespective of the architecture, need for highly optimized BLAS

- How to use threads/cores?
  Multithreaded BLAS vs. algorithms-by-blocks

- Large/many problems but limited memory $\Rightarrow$ streaming

References

- MKL
  http://software.intel.com/en-us/intel-mkl
- cuBLAS, CULA
  https://developer.nvidia.com/cublas
- libFLAME
  http://wiki.cs.utexas.edu/flame
- BLIS
  http://code.google.com/p/blis/
- OpenBLAS
  http://xianyi.github.io/OpenBLAS/
- Magma
  http://icl.cs.utk.edu/magma/
- Plasma
  http://icl.cs.utk.edu/plasma/