UNIVERSITAT JAUME•I
HPC&A

Impact of Multi-core and Many-core Technology on Dense Linear Algebra

Enrique S. Quintana-Ortí

HPC&A
UNIVERSITAT JAUME•I High Performance Computing and Architectures

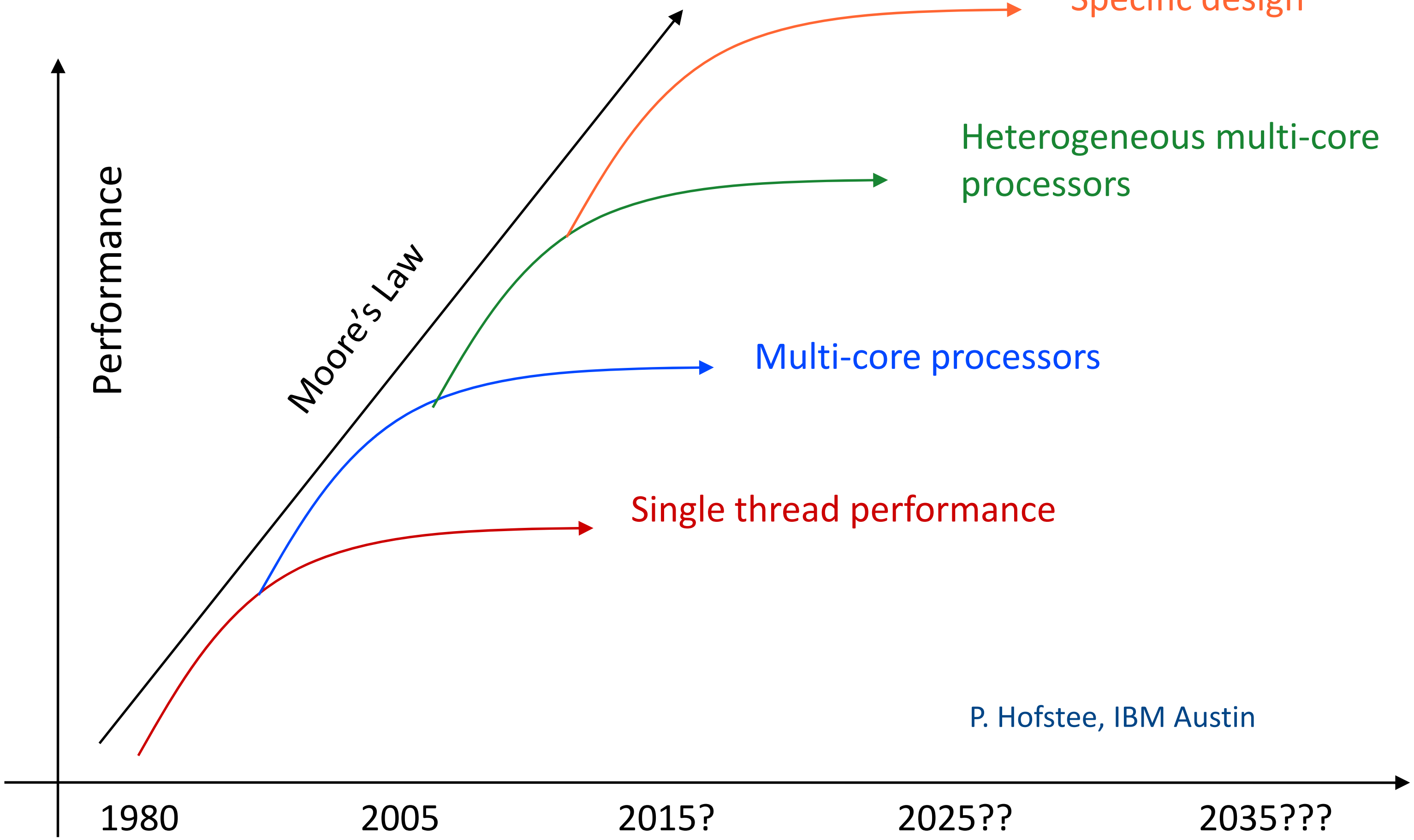
Berlin, September 2011

Multi-core and Many-core

- "The free lunch is over" (H. Sutter, 2005)
  - End of the race for GHz
    - Energy is proportional to f^3
    - Energy becomes heat
  - No more ILP
    - On average, 1 branch every five instructions
    - Also for dense linear algebra?
  - No significant improvement in memory latency
    - 1 stall ≈ 240 cycles (2008)

Multi-core and Many-core

- …but Moore's Law still holds



Multi-core and Many-core

UNIVERSITAT JAUME·I
HPC&A

- Intel Xeon Nehalem: 8-core
- AMD Opteron Interlagos (bulldozer): 16-core chip $\rightarrow$ Q3 2011

- Heterogeneous designs:
  - AMD APUs
  - Intel Sandy Bridge
  - NVIDIA+ARM

Leveraging TLP

- Libraries (dense linear algebra)
  - PLASMA
  - MAGMA
  - libflame
- Tools (general purpose)
  - StarSs
  - StarPU

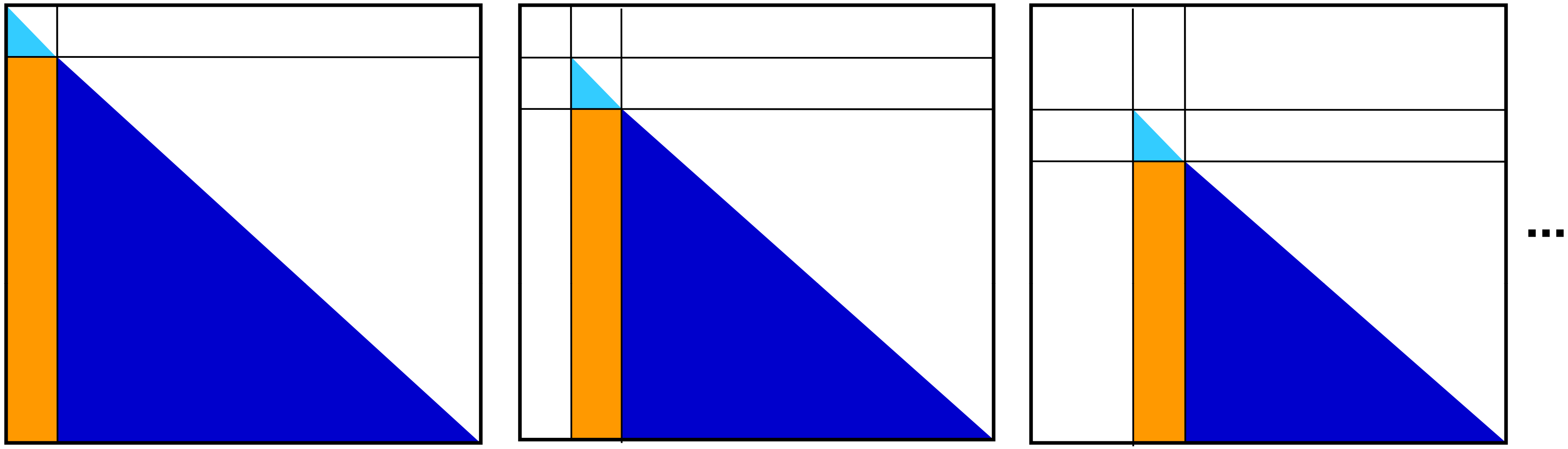
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE
INRIA
centre de recherche BORDEAUX - SUD-OUEST

Leveraging TLP

- Cholesky: LAPACK+MT-BLAS

$A_{11} = L_{11} L_{11}^T$
$A_{21} := L_{21} = A_{21} L_{11}^{-T}$
$A_{22} := A_{22} - L_{21} L_{21}^T$



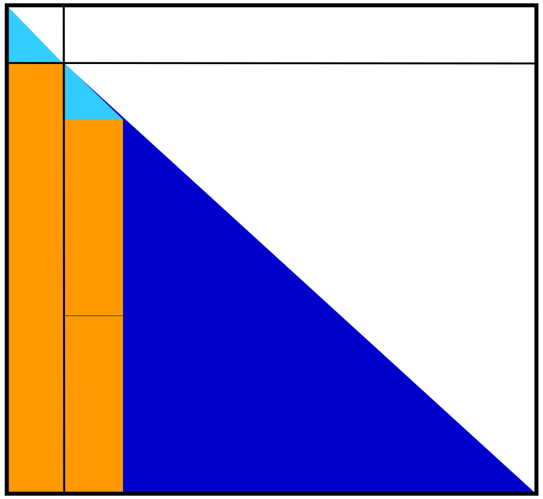
1st iteration 2nd iteration 3rd iteration

Leveraging TLP

UNIVERSITAT JAUME•I HPC&A

- Cholesky: LAPACK+MT-BLAS

$$A_{11} = L_{11} L_{11}^{T}$$
$$A_{21} := L_{21} = A_{21} L_{11}^{-T}$$
$$A_{22} := A_{22} - L_{21} L_{21}^{T}$$



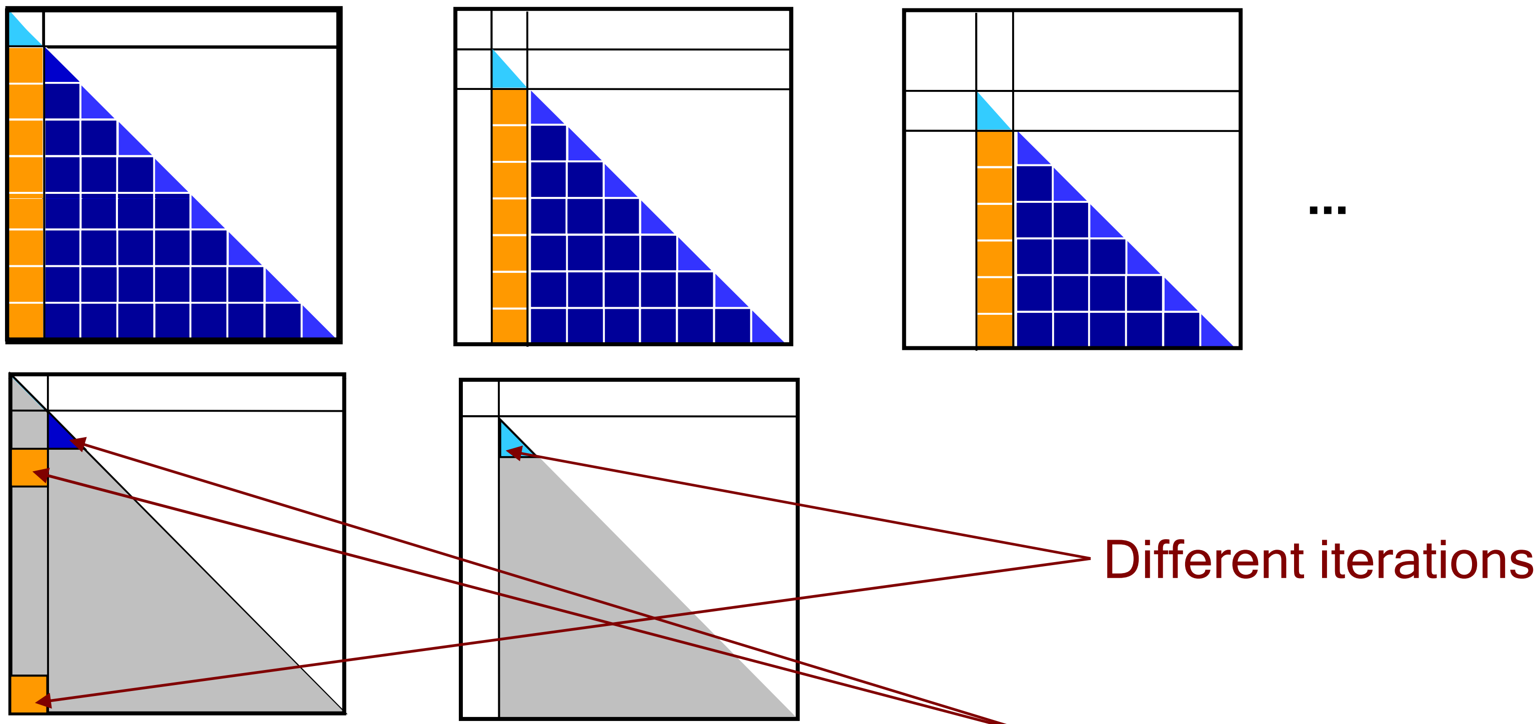
1st iteration

Some parallelism can be regained with a look-ahead but:

- It complicates programming
- It hampers portability

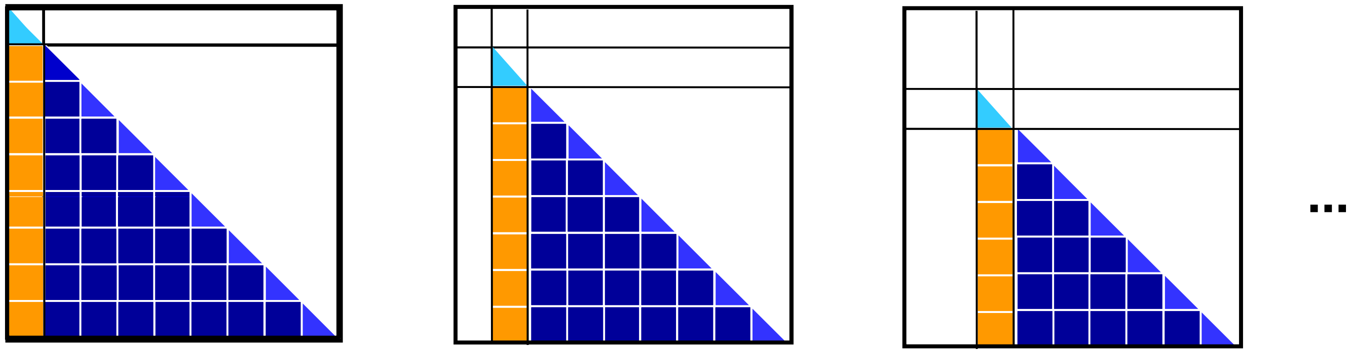
Leveraging TLP

- Cholesky: More concurrency available



Leveraging TLP

- Cholesky: More concurrency available



1. Decompose the problem into tasks
2. Scheduler (run-time) executes the tasks:
    1. Keep track of dependencies
    2. Maximize concurrency (with priorities)
    3. Balance the load (possibly dynamically)

PLASMA: Overview

- Univ. Tennessee, Univ. California @ Berkeley, Univ. Colorado @ Denver
- Support for multi-core platforms
- Support for complex/real, single/double
- Covers BLAS-3, linear system solvers, and partially SVD and EVD
- Column-major order accepted, internally converted to tile layout
- Static or dynamic scheduler (QUARK)
- Licensed: free redistribution and reuse

PLASMA: Functionality

- Linear systems
  - LU w/partial pivoting, Cholesky
  - Overdetermined, underdetermined linear systems via QR and LQ
  - Explicit matrix inversion
  - Mixed precision iterative refinement

- Symmetric Eigenvalue Problem (only eigenvalues, no eigenvectors yet)
  - reduction to “block tridiagonal” (band)
  - band reduction (bulge chasing)
  - QR iteration (LAPACK)

- Singular Value Problem (only singular values, no singular vectors yet)
  - reduction to “block bidiagonal” (band)
  - band reduction (bulge chasing)
  - QR algorithm (LAPACK)

- Generalized Eigenvalue Problem (s.p.d. matrices only)
  - Cholesky factorization of B and application to A
  - reduction to band & band reduction (bulge chasing)
  - QR algorithm (LAPACK)

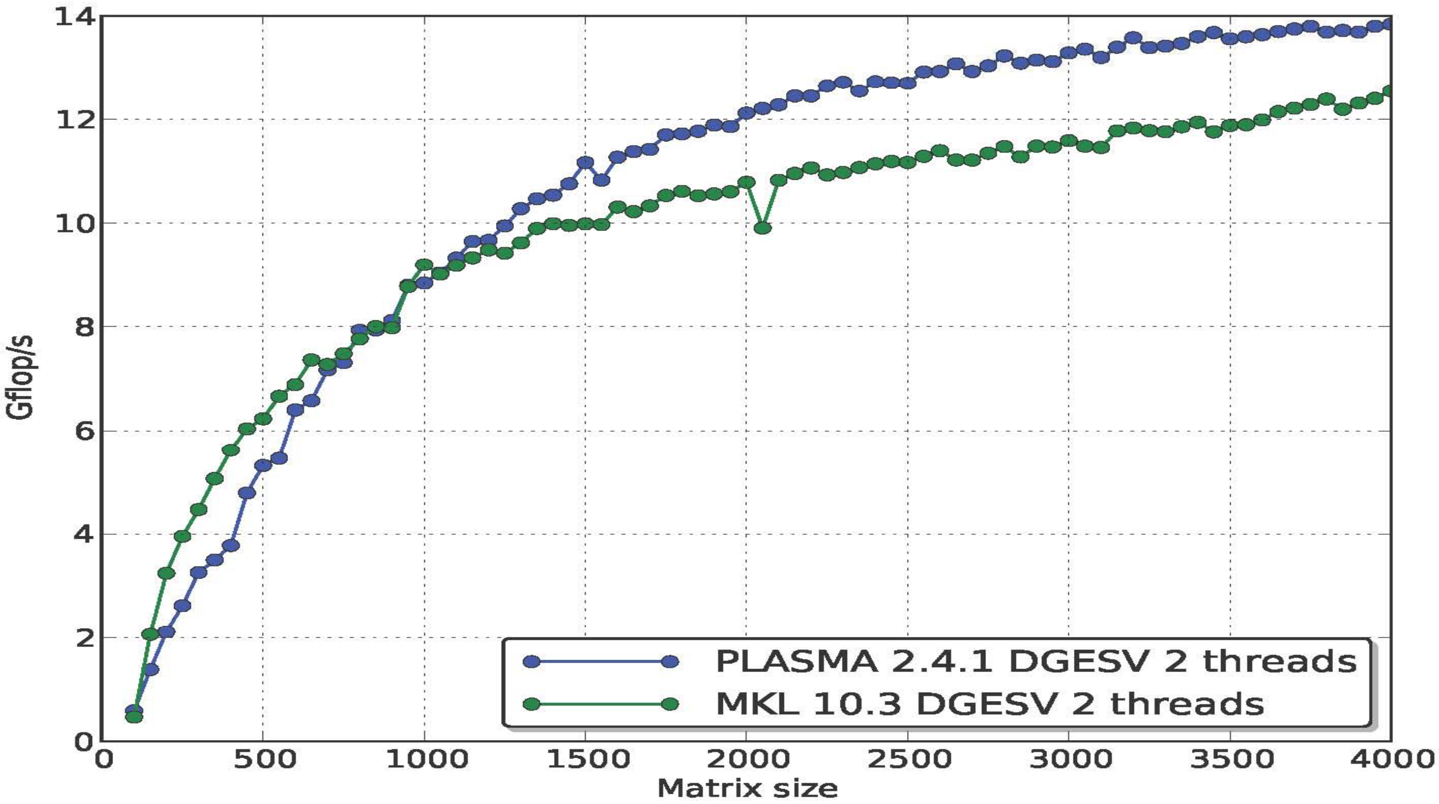
Thanks to J. Kurzak, ICL - UTK

PLASMA: Interface

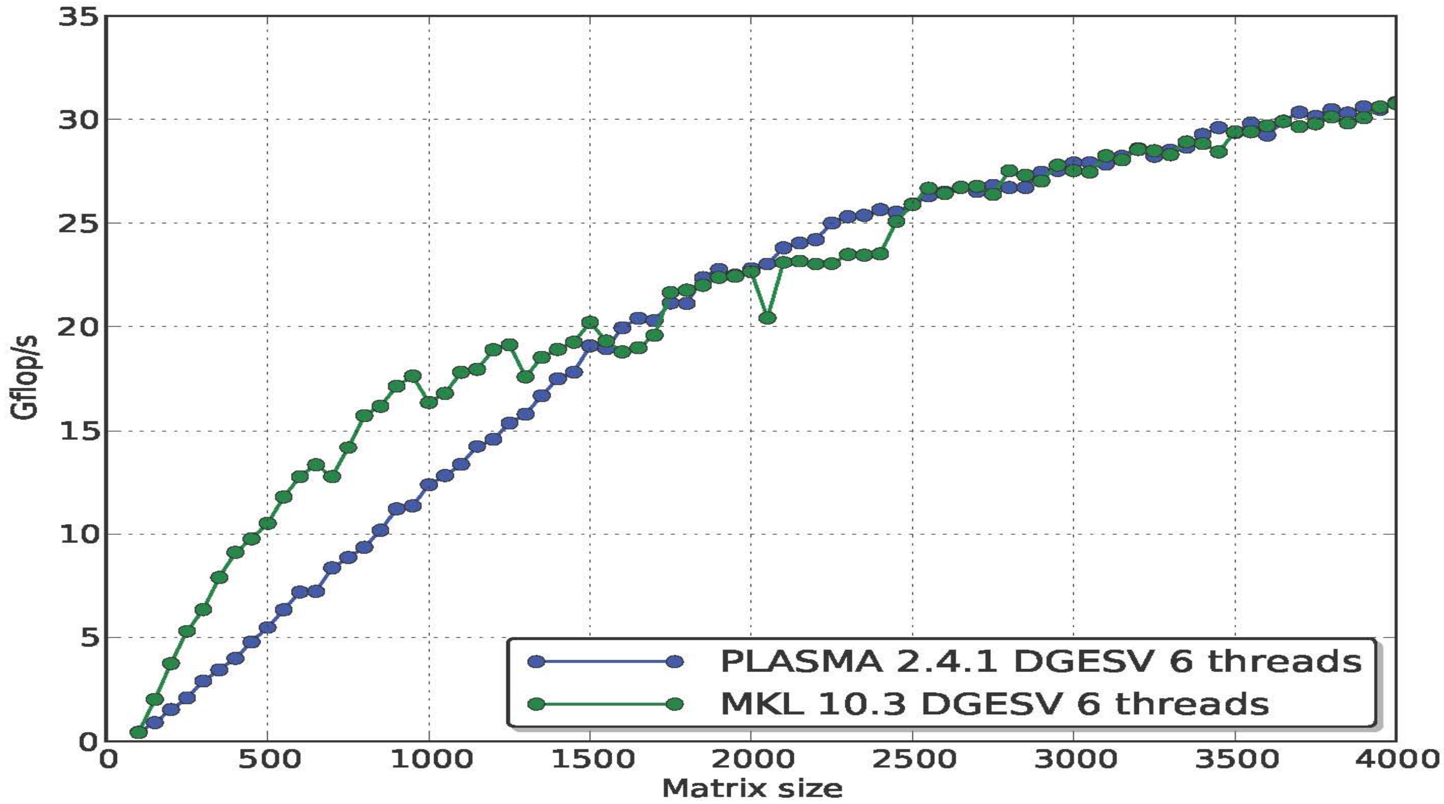

```
PLASMA_Init(0);

PLASMA_Desc_Create(&descA, ...);
PLASMA_Desc_Create(&descB, ...);

PLASMA_dLapack_to_Tile(A, lda, descA);
PLASMA_dLapack_to_Tile(B, ldb, descB);

PLASMA_dgesv_Tile(descA, IPIV, descB);

PLASMA_Desc_Destroy(&descA);
PLASMA_Desc_Destroy(&descB);

PLASMA_Finalize();
```

Thanks to J. Kurzak, ICL - UTK

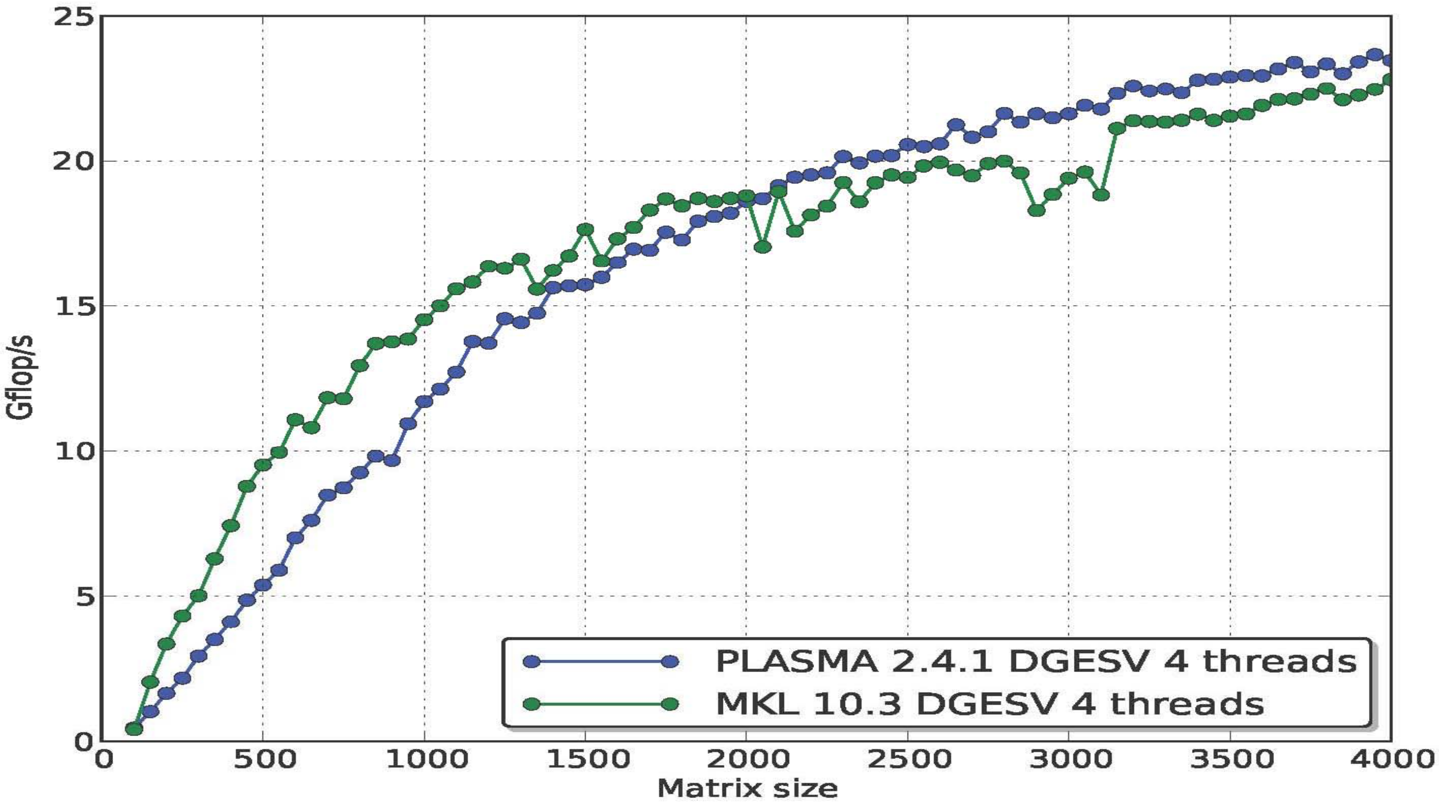
PLASMA: Performance



PLASMA: Performance



PLASMA: Performance



PLASMA: Performance

- AMD 48 cores – J. Kurzak ISC'11

UNIVERSITAT JAUME•I
HPC&A

MAGMA: Overview

- Univ. Tennessee, Univ. California @ Berkeley, Univ. Colorado @ Denver
- Support for hybrid platforms: one or more multi-core processors + 1 GPU (no multi-GPU)
- Support for complex/real, single/double
- Static scheduler
- Licensed: free redistribution and reuse

MAGMA: Functionality

UNIVERSITAT JAUME•I
HPC&A

- Linear systems
  - LU, QR, and Cholesky factorizations
  - Mixed-precision iterative refinement

- Eigenproblems and singular value problems
  - Hessenberg, bidiagonal, and tridiagonal reductions

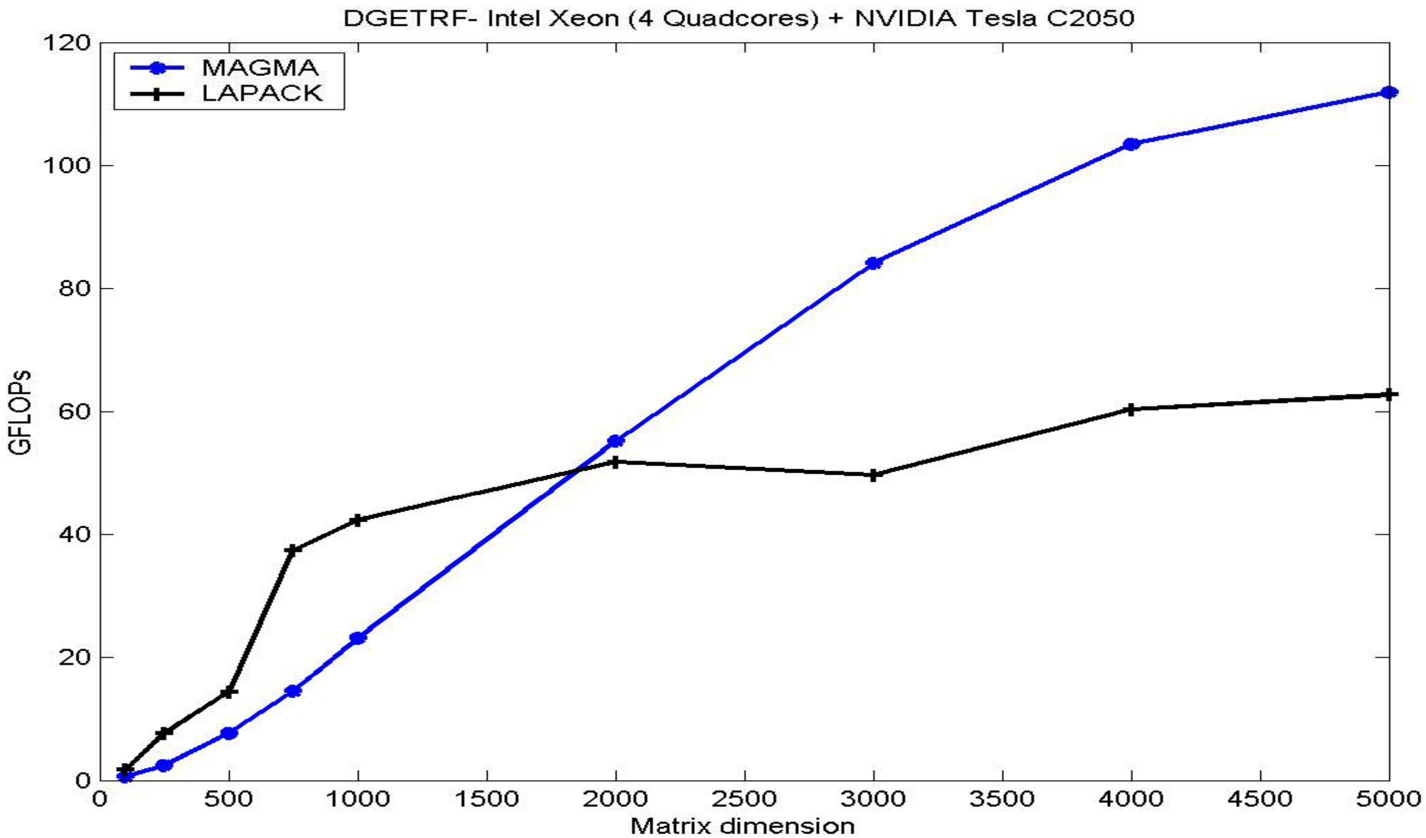
- Generalized Hermitian-definite eigenproblem solvers

MAGMA: Interface (F90)

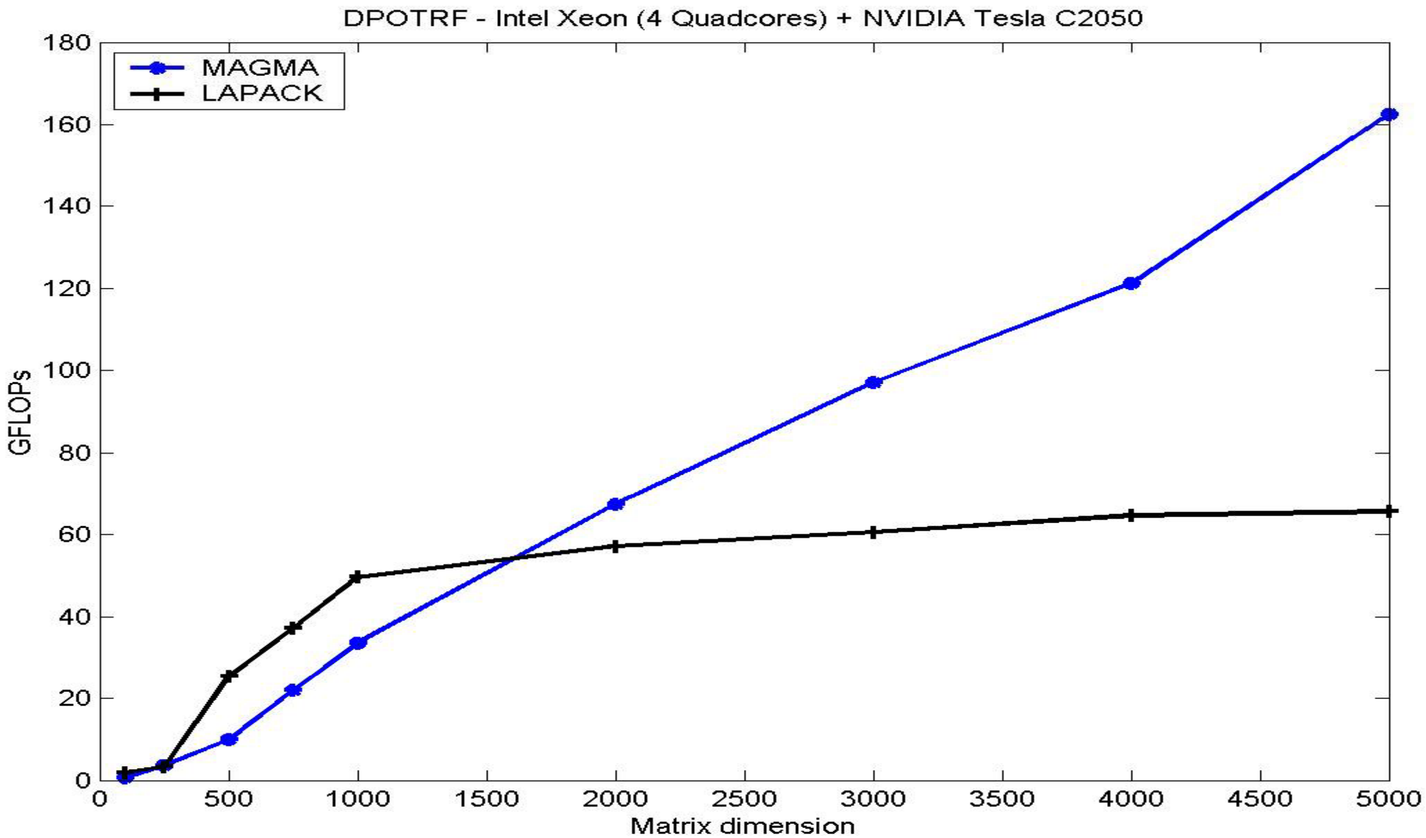
UNIVERSITAT JAUME•I
HPC&A

```
magma_devptr_t :: devptrA, devptrB

!------ Allocate GPU memory
stat = cublas_alloc(ldda*n, sizeof_double, devPtrA)

!---- devPtrA = h_A
call cublas_set_matrix(n, n, size_of_elt, h_A, lda,
                       devptrA, ldda)

call magmaf_dgetrf_gpu(n, n, devptrA, ldda, ipiv, info)
```

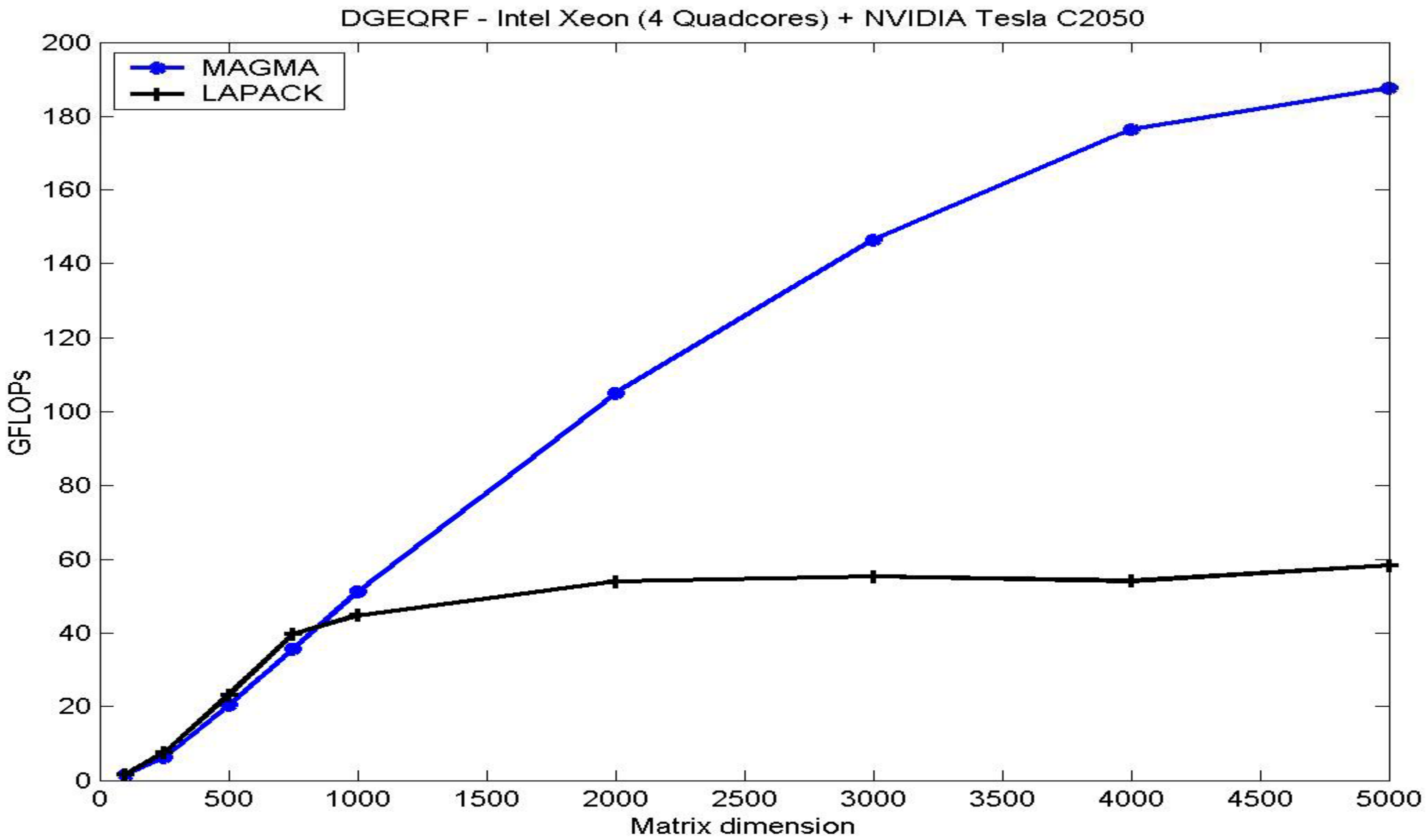
MAGMA: Performance



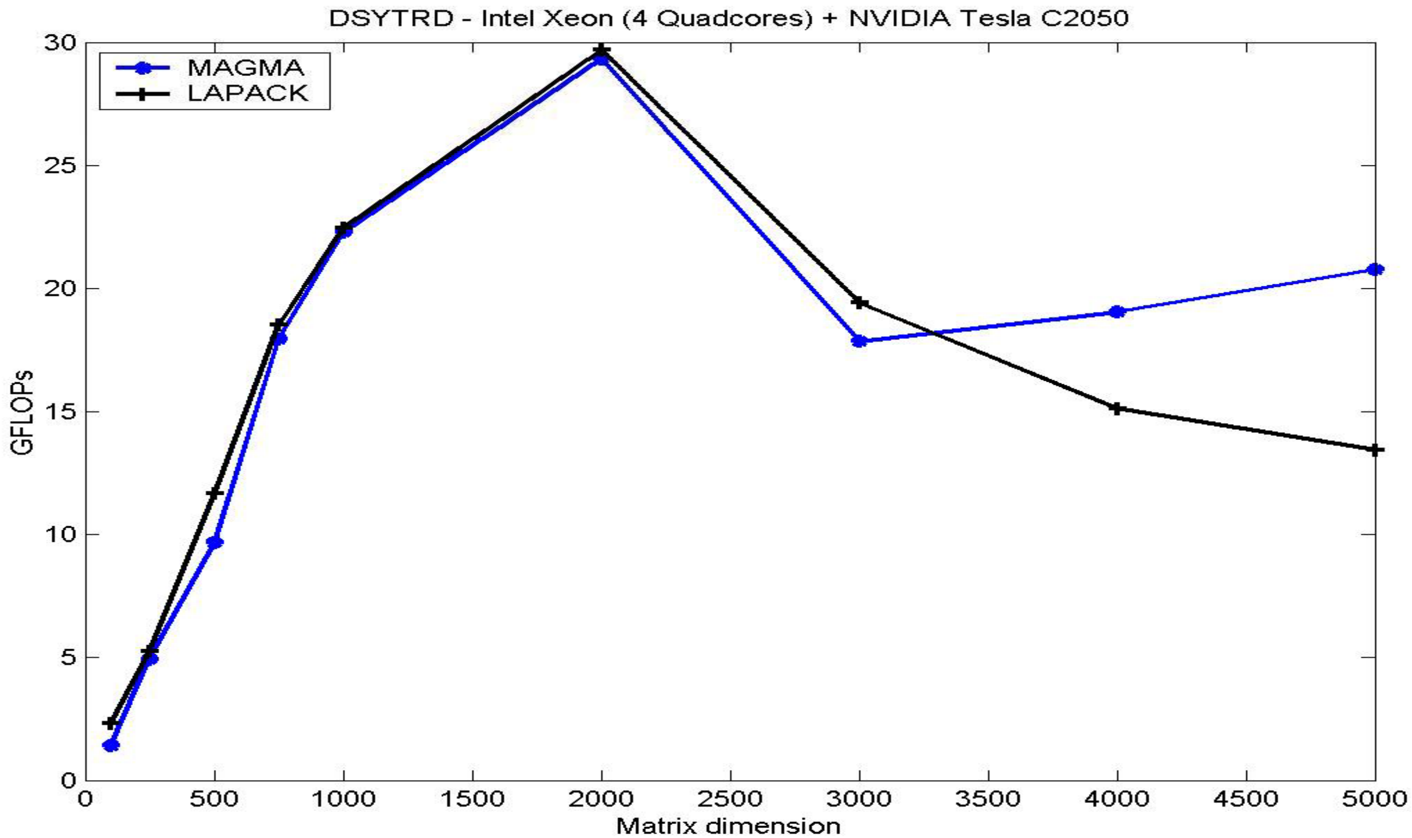
MAGMA: Performance



MAGMA: Performance



MAGMA: Performance



libflame: Overview

UNIVERSITAT JAUME·I
HPC&A

- Univ. Texas @ Austin
- Object-oriented API
- Support for multi-core and multi-GPU platforms
- Support for complex/real, single/double
- Covers BLAS-1, -2, -3 and linear system solvers. No SVD, EVD
- lapack2flame compatibility layer
- Column-major order or storage-by-blocks
- MT-BLAS or dynamic scheduling (SuperMatrix)
- Licensed as LGPL

libflame: Funcionality

UNIVERSITAT JAUME•I HPC&A

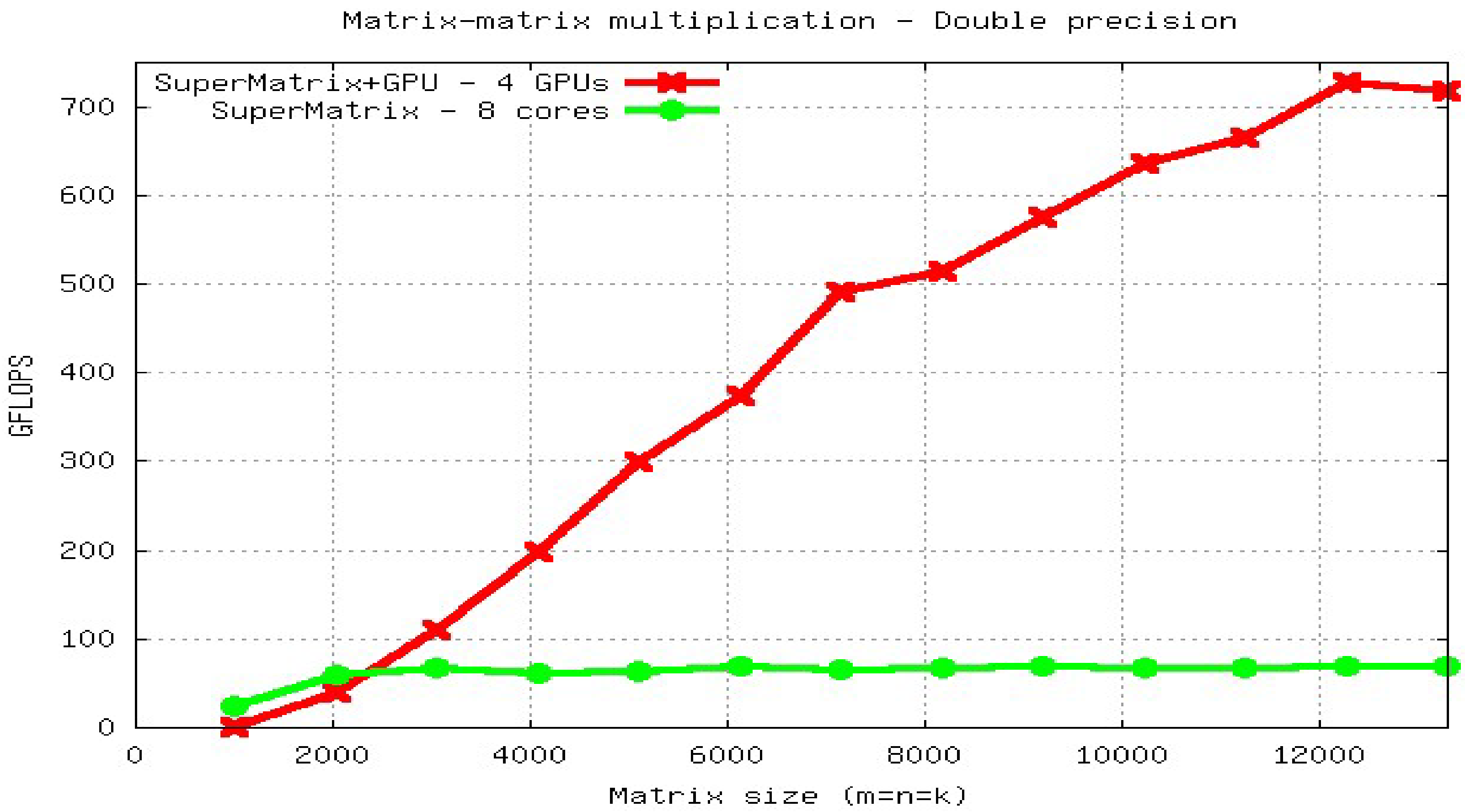
operation	Classic FLAME	FLASH/SM	lapack2flame
Level-3 BLAS	y	y	n/a
Cholesky	y	y	y
LU with partial pivoting	y	y	y
LU with incremental pivoting	y	y	*
QR (UT)	y	y	y
LQ (UT)	y	y	y
SPD/HPD inversion	y	y	y
Triangular inversion	y	y	y
Triangular Sylvester	y	y	y
Lyapunov	y	y	y
Up-and-downdate (UT)	y	y	*
SVD	planned		
EVD	planned		

* Not present in LAPACK

libflame: Interface

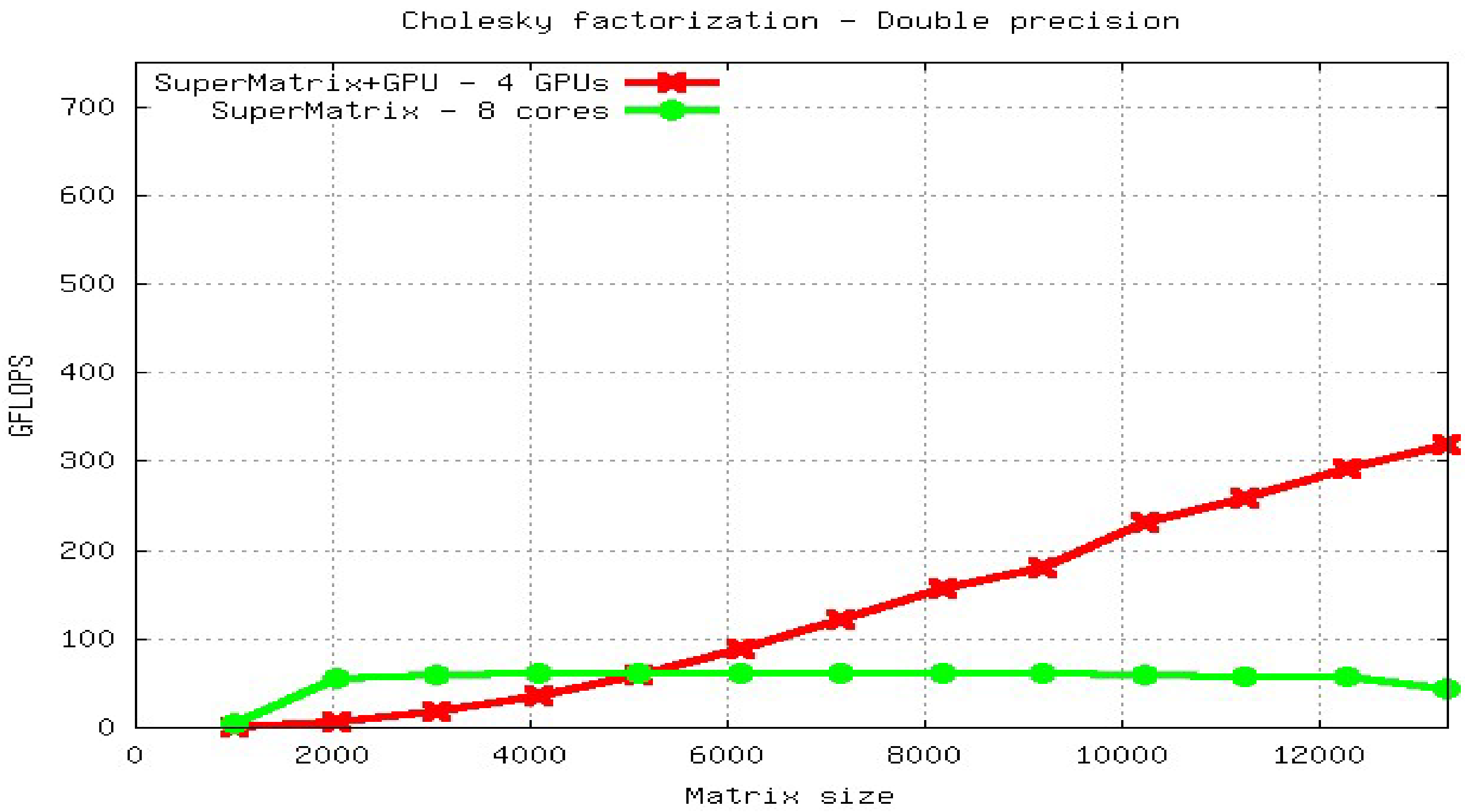
```
FLA_Obj A, p, b, x;

FLA_Init();

FLA_Obj_create( FLA_DOUBLE, m, m, 0, 0, &A );
FLA_Copy_buffer_to_object( FLA_NO_TRANSPOSE, m, m,
                           buffer, rs, cs, 0, 0, A );
...

// Initialize A, b, x.

FLA_LU_piv( A, p );
FLA_LU_piv_solve( A, p, b, x );

FLA_Obj_free( &A );
...
```

libflame: Performance

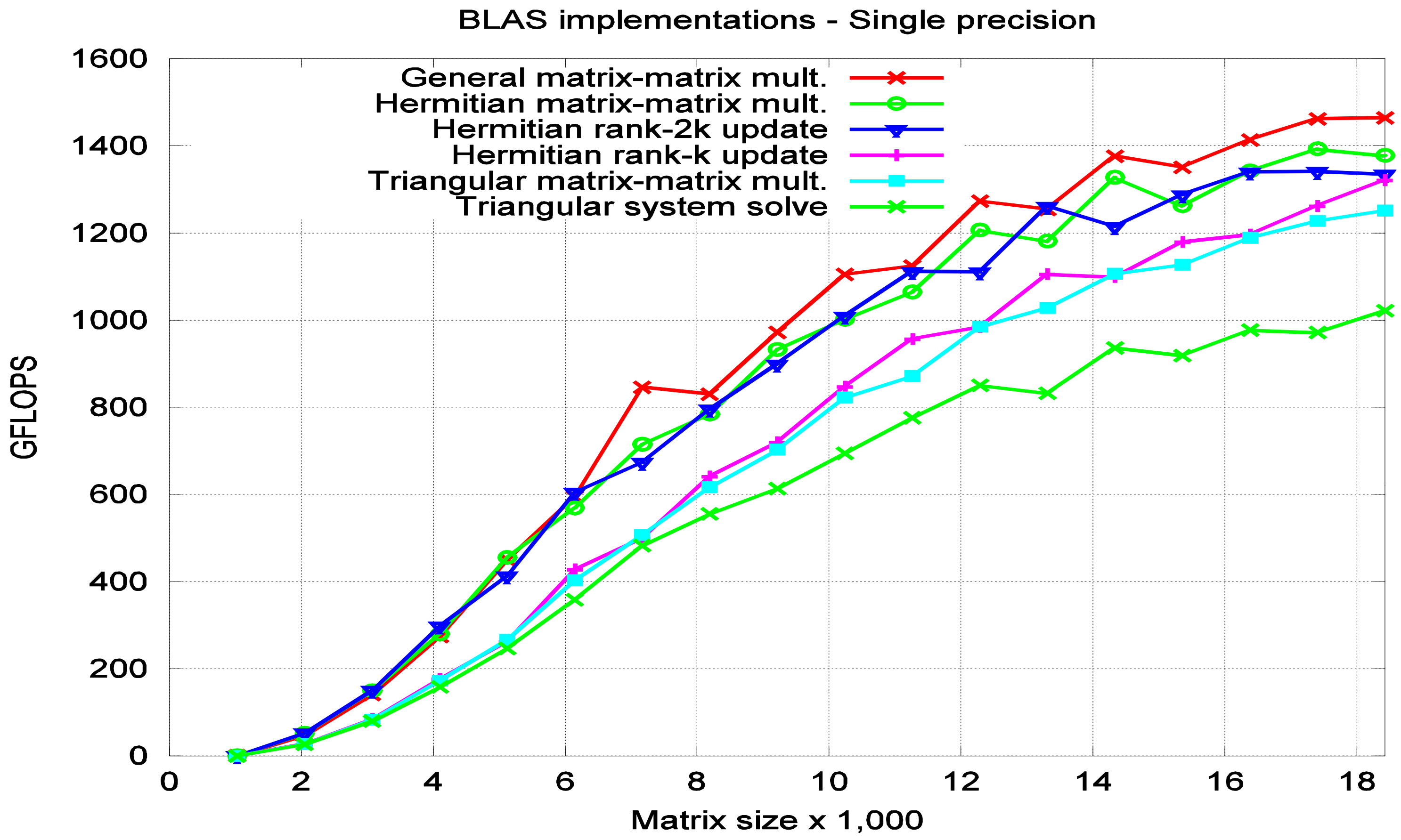


libflame: Performance

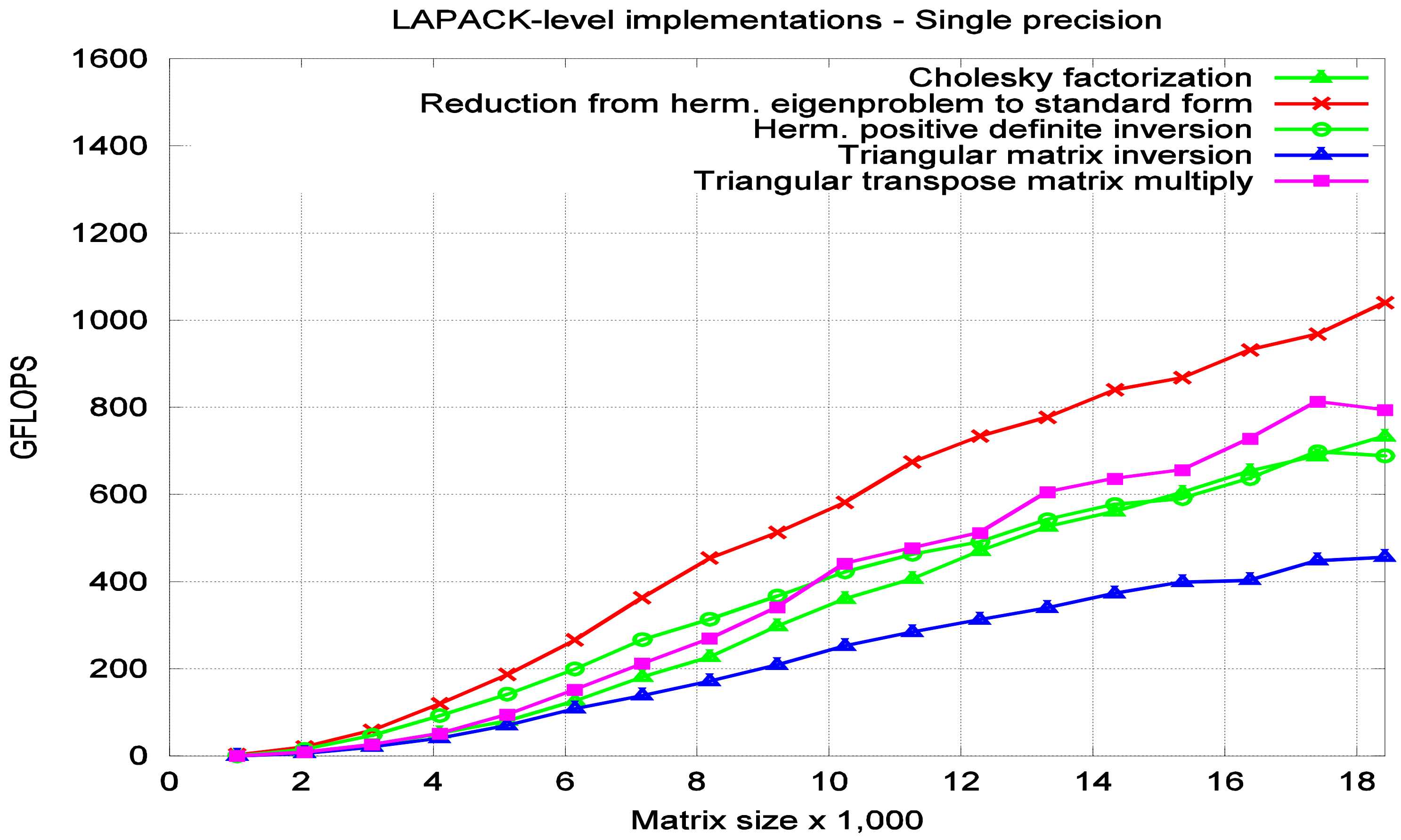
UNIVERSITAT JAUME•I
HPC&A



libflame: Performance



libflame: Performance

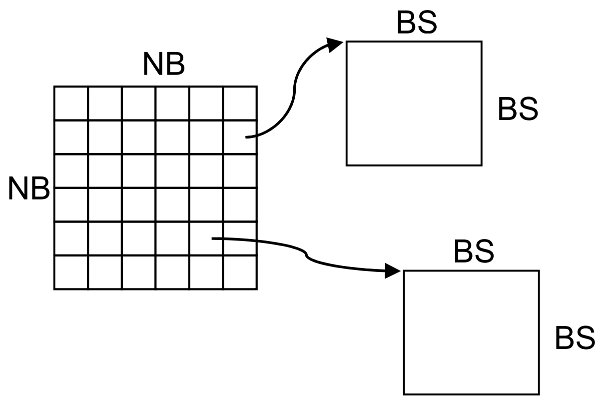


StarSs: Overview

- Barcelona Supercomputing Center
- A parallelization framework (tool), not a library:
  - OpenMP-like pragmas
  - Source-to-source compiler
  - Run-time library
- Support for multi-core, multi-accelerator (Cell, GPU) etc. in different instances: SMPSs, CellSs, GPUSs, etc.
- Licensed as LGPL

SMPSs: Interface

```
int main  (int argc, char **argv) {
  int i, j, k;
  ...
  initialize(A, B, C);

  for (i=0; i < NB; i++)
    for (j=0; j < NB; j++)
      for (k=0; k < NB; k++)
        mm_tile( C[i][j], A[i][k], B[k][j]);
}
```




```
#pragma omp task input([BS][BS]A,[BS][BS]B)\
inout([BS][BS]C)
static void mm_tile ( float C[BS][BS],
                      float A[BS][BS],
                      float B[BS][BS]) {
  int i, j, k;

  for (i=0; i< BS; i++)
    for (j=0; j< BS; j++)
      for (k=0; k< BS; k++)
        C[i][j] += A[i][k] * B[k][j];
}
```

GPUSs: Interface

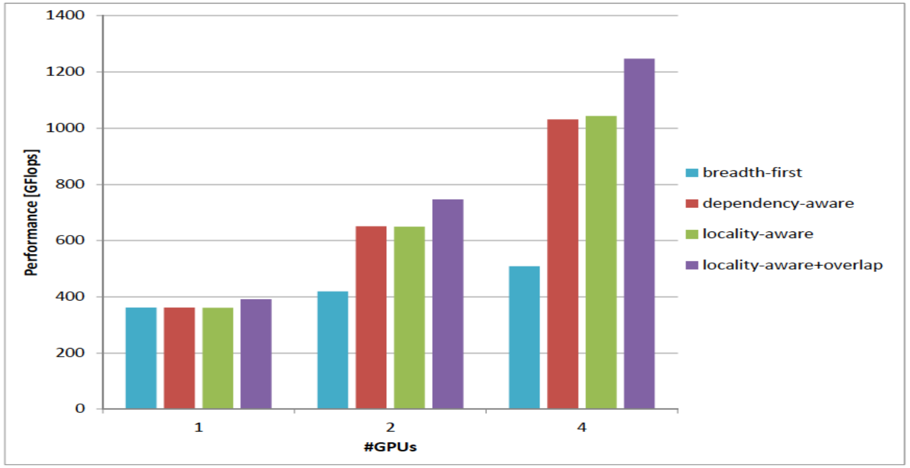
```
void blocked_cholesky( int NB, float *A ) {
  int i, j, k;
  for (k=0; k<NB; k++) {
    spotrf (A[k*NB+k]);
    for (i=k+1; i<NB; i++)
      strsm (A[k*NB+k], A[k*NB+i]);
      for (i=k+1; i<NB; i++) {
        for (j=k+1; j<i; j++)
          sgemm( A[k*NB+i], A[k*NB+j], A[j*NB+i]);
        ssyrk (A[k*NB+i], A[i*NB+i]);
      }
  }
  #pragma omp task wait
}
```

```
#pragma omp target device (cuda) copy_deps
#pragma omp task inout([BS][BS]A)
void spotrf (float *A);
#pragma omp target device (cuda) copy_deps
#pragma omp task input ([BS][BS]A) inout([BS][BS]C)
void ssyrk (float *A, float *C);
#pragma omp target device (cuda) copy_deps
#pragma omp task input ([BS][BS]A, [BS][BS]B) inout([BS][BS]C)
void sgemm (float *A, float *B, float *C);
#pragma omp target device (cuda) copy_deps
#pragma omp task input ([BS][BS]T) inout([BS][BS]B)
void strsm (float *T, float *B);
```

GPUSs: Performance

Blocked

Matrix-matrix multiply



Cholesky

Control Problems in GPUs

UNIVERSITAT JAUME•I
HPC&A

- Matrix inversion via GJE: general and s.p.d.
- Lyapunov and Riccati sign-function solvers
- Mixed precision/iterative refinement for Lyapunov equations
- Differential Riccati equations
- Model reduction via BT and BST based on those

UNIVERSITAT JAUME•I
HPC&A

Remarks

- Impressive potential of new architectures for solving large-scale problem… but likely of interest only to a few SLICOT users
- Libraries immature to be adopted (e.g., MAGMA and PLASMA may merge)… but they will likely incorporate an LAPACK interface
- No significant change of functionality (decrease?)
- Change in some of the underlying algorithms:
    - Incremental QR factorization
    - LU with incremental pivoting
    - Two stage reduction to condensed forms
- Iterative refinement?
- GPUs may be integrated with "regular" cores, but it will be at the expense of #cores

Acks

- Thanks to J. Kurzak and P. Luszczek -The University of Tennessee at Knoxville- for the results with PLASMA and MAGMA
- Thanks to F. Igual -The University of Texas at Austin- for the results with libflame