# MAGMA

**M**atrix **A**lgebra on **G**PU and **M**ulticore **A**rchitectures

Mark Gates

February 2012

# Hardware trends

- Scale # cores instead of clock speed
- Hardware issue became software issue
- Multicore
- Hybrid

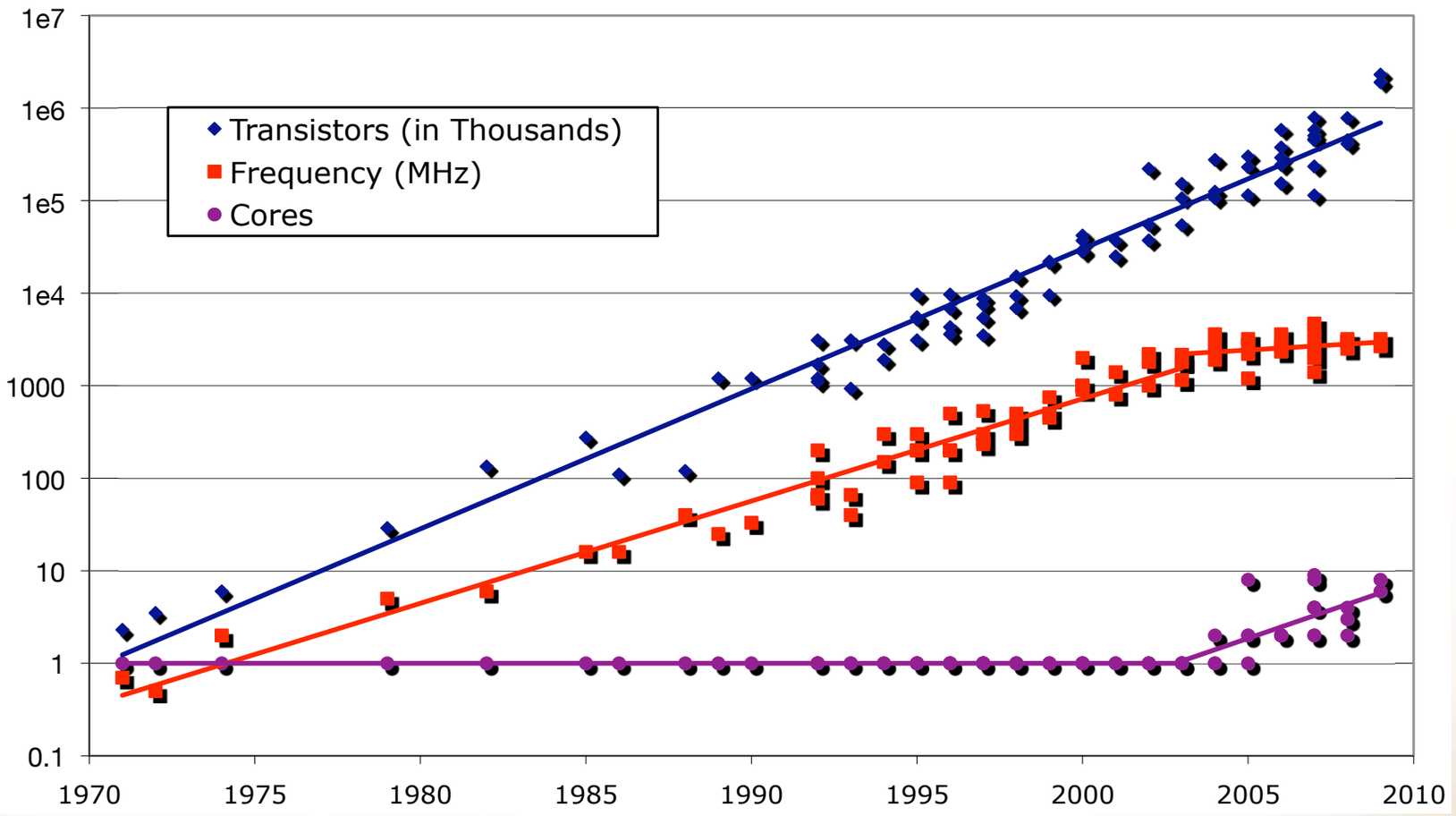
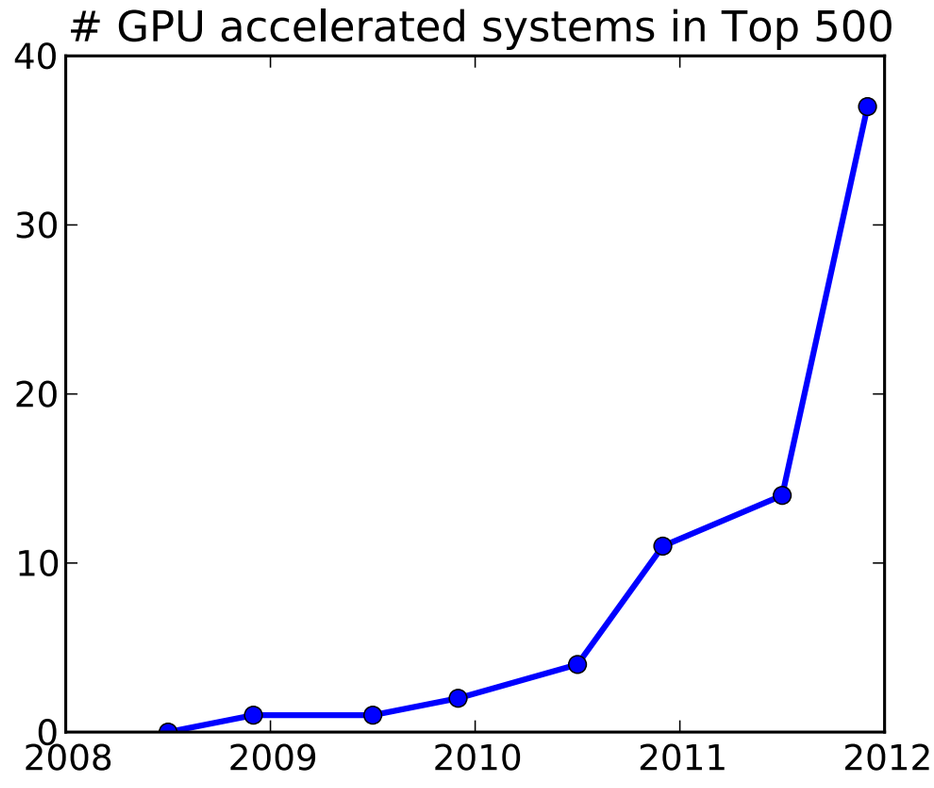



Figure from Kathy Yelick, “Ten Ways to Waste a Parallel Computer.” Data from Kunle Olukotun, Lance Hammond, Herb Sutter, Burton Smith, Chris Batten, and Krste Asanoviç.

# Future systems

- Most likely hybrid design
  - Multicore + GPU accelerators
- Today accelerators attached
- Future accelerators integrated
  - Intel's MIC Knight's Corner
  - AMD's Fusion
  - Nvidia's Project Denver





ICL UT

# Challenges of GPUs

- High levels of parallelism
- Hybrid architecture
  - Small, non-parallelizable tasks on CPU
  - Large, parallel tasks on GPU
- Compute vs. communication gap growing
  - Tesla C2070 has 515 Gflops, 8 GB/s PCIe

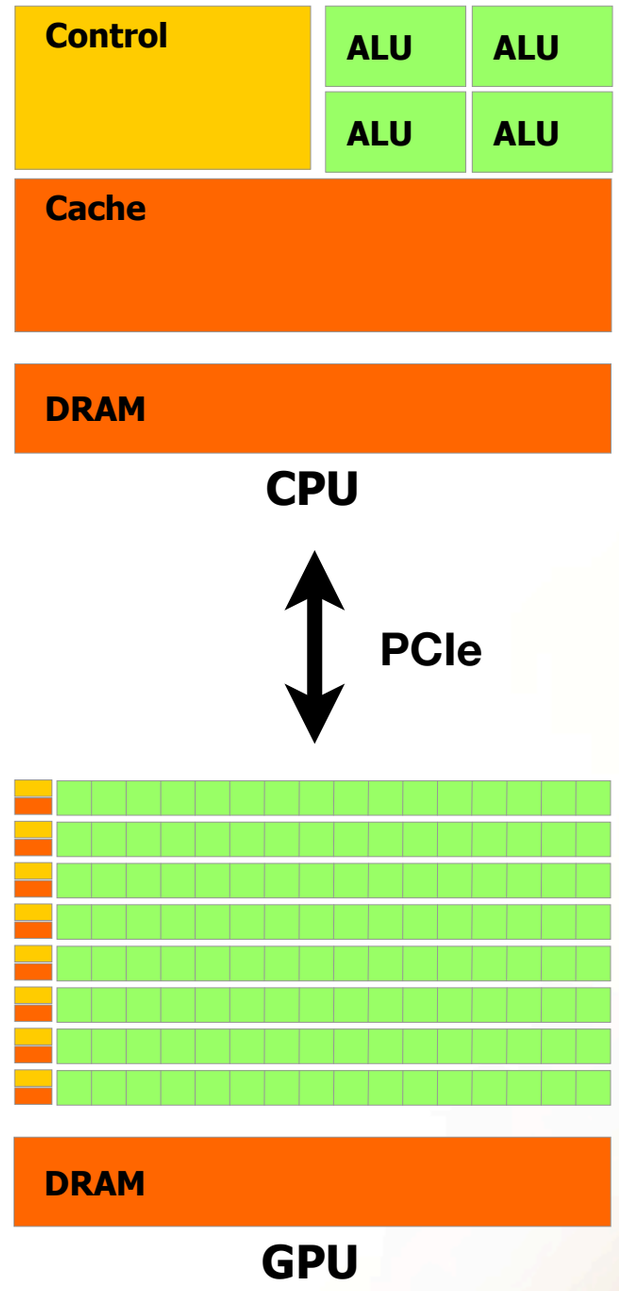
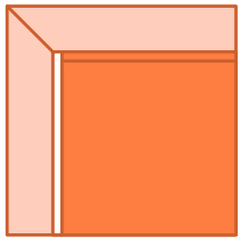



Figure from NVIDIA CUDA C Programming Guide

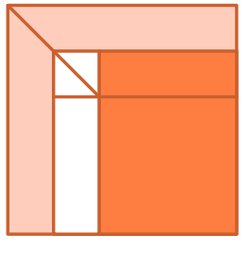
# Software generations

**LINPACK** (70's)
vector operations



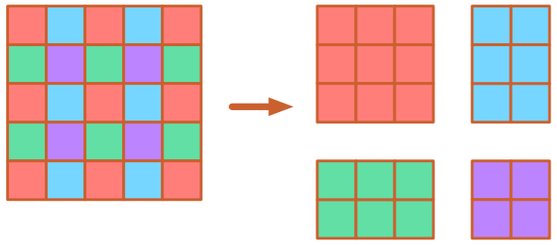
Level-1 BLAS

**LAPACK** (80's)
blocked, cache friendly



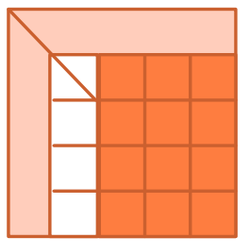
Level-3 BLAS

**ScaLAPACK** (90's)
distributed memory



PBLAS
message passing

**PLASMA**
tiled, multicore



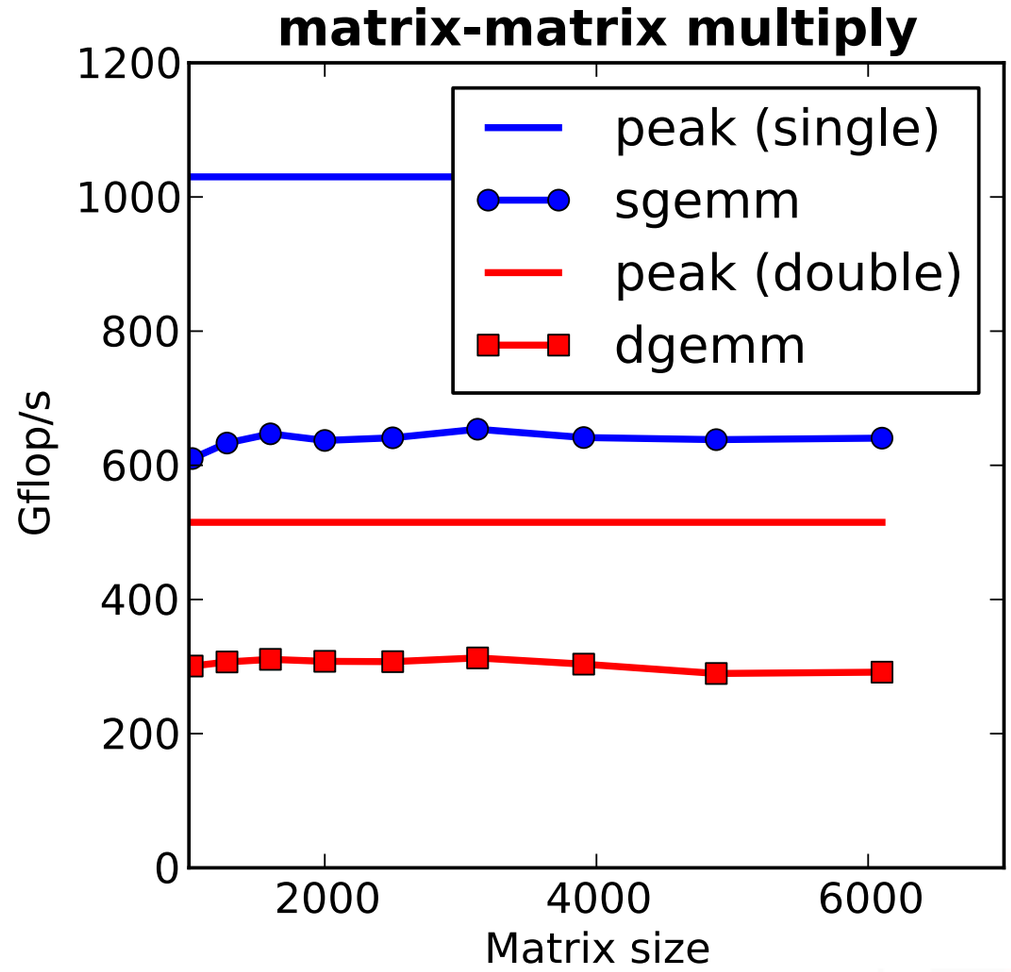
DAG + scheduler

**MAGMA**
hybrid


hybrid kernels

# Nvidia's CUBLAS

- Level 1, 2, 3 BLAS kernels
  - cublasDgemv
  - cublasDgemm
- Copy between CPU ↔ GPU
  - cublasSetMatrix
  - cublasGetMatrix
- Stream support
  - cublasSetKernelStream, magmablasSetKernelStream



# MAGMA 1.1

- LAPACK column-wise layout
- LAPACK-like C and Fortran interfaces
- CPU and GPU interfaces

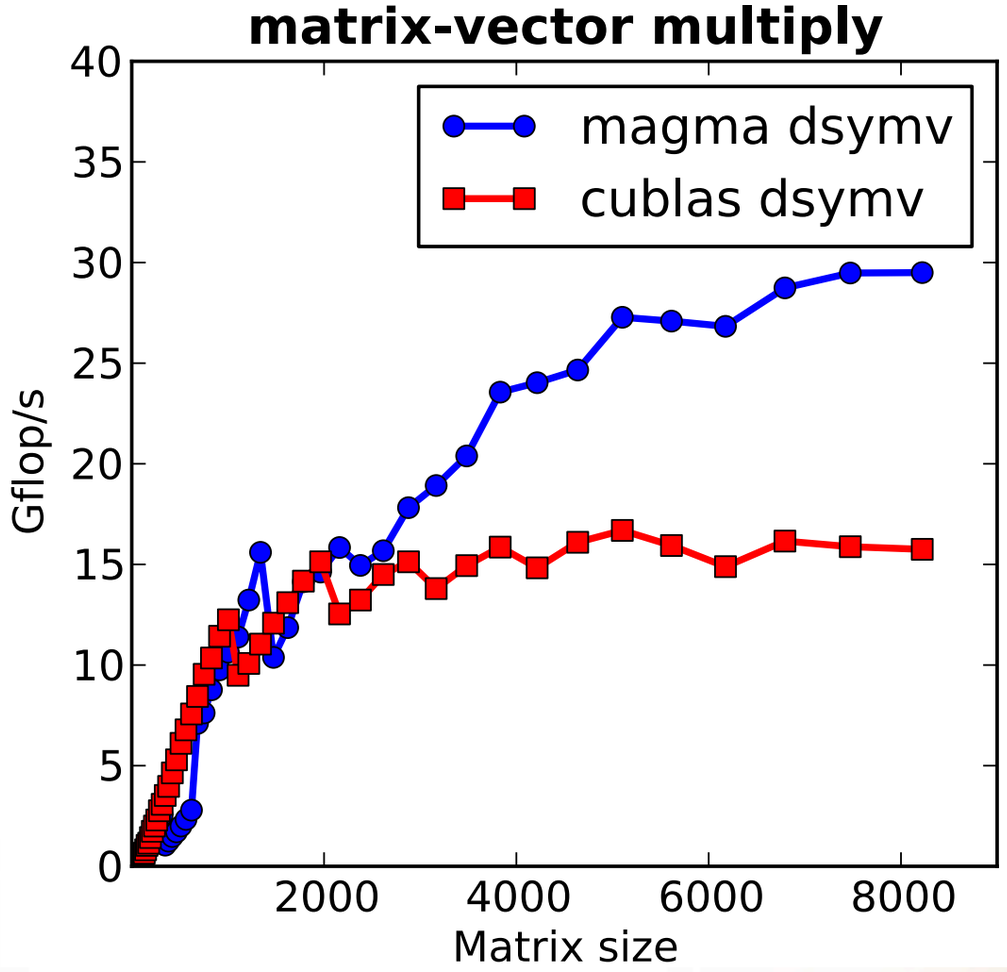
| MAGMA 1.1 ROUTINES & FUNCTIONALITIES | SINGLE GPU | MULTI-GPU STATIC | MULTI-GPU DYNAMIC |
|---|---|---|---|
| One-sided Factorizations (LU, QR, Cholesky) | ✓ | ✓ | ✓ |
| Linear System Solvers | ✓ | | ✓ |
| Linear Least Squares (LLS) Solvers | ✓ | | ✓ |
| Matrix Inversion | ✓ | | ✓ |
| Singular Value Problem (SVP) | ✓ | | |
| Non-symmetric Eigenvalue Problem | ✓ | | |
| Symmetric Eigenvalue Problem | ✓ | | |
| Generalized Symmetric Eigenvalue Problem | ✓ | | |

| | |
|---|---|
| **SINGLE GPU** | Hybrid LAPACK algorithms with static scheduling and LAPACK data layout |
| **MULTI-GPU STATIC** | Hybrid LAPACK algorithms with 1D block cyclic static scheduling and LAPACK data layout |
| **MULTI-GPU DYNAMIC** | Tile algorithms with StarPU scheduling and tile matrix layout |

ICL UT

# MAGMA 1.1

- 50+ hybrid LAPACK algorithms
  - 4 precisions (single, double, single complex, double complex)
  - 3 mixed precision algorithms
- MAGMA BLAS
  - Supplements CUBLAS
  - Improves some routines



# Naming: magma_zgesv_gpu

- **magma_** or **magmablas_** prefix
- Precision (**s**ingle, **d**ouble, single **c**omplex, "**z**" double complex).
  Mixed precision (**ds** and **zc**)
- Matrix type
  **ge**neral **sy**mmetric **he**rmetian **po**sitive definite
  **or**thogonal **un**itary **tr**iangular
- Operation
  **sv** solve
  **trf** triangular factorization
  **ev** eigenvalue problem
  **gv** generalized eigenvalue problem
- Interface **_gpu** suffix

# Driver routines

- Solve entire problem

| | | | | Interface | |
|---|---|---|---|---|---|
| **Matrix type** | **Operation** | **Routine** | | **CPU** | **GPU** |
| General | Solve | dgesv | dsgesv | ✓ | ✓ |
| SPD | Solve | dposv | dsposv | ✓ | ✓ |
| Least squares | Solve | dgeqrs | dsgeqrsv | | ✓ |
| General | SVD | dgesvd | | ✓ | |
| | Eigenvalues | dgeev | | ✓ | |
| Symmetric | Eigenvalues | dsyevd* / zheevd* | | ✓ | ✓ |
| | Generalized eigenvalues | dsygvd* / zhegvd* | | ✓ | |

* Additional variants; complete list at http://icl.eecs.utk.edu/magma/

# Computational routines

- Solve one part of problem

| | | | Interface | |
|---|---|---|---|---|
| **Matrix type** | **Operation** | **Routine** | **CPU** | **GPU** |
| General | LU | dgetrf | ✓ | ✓ |
| | Solve (given LU) | dgetrs | | ✓ |
| | Inverse | dgetri | | ✓ |
| SPD | Cholesky | dpotrf | ✓ | ✓ |
| | Solve (given $LL^T$) | dpotrs | | ✓ |
| | Inverse | dpotri | | ✓ |
| General | QR | dgeqrf | ✓ | |
| | Generate Q | dorgqr / zungqr | ✓ | ✓ |
| | Multiply by Q | dormqr / zunmqr | ✓ | ✓ |

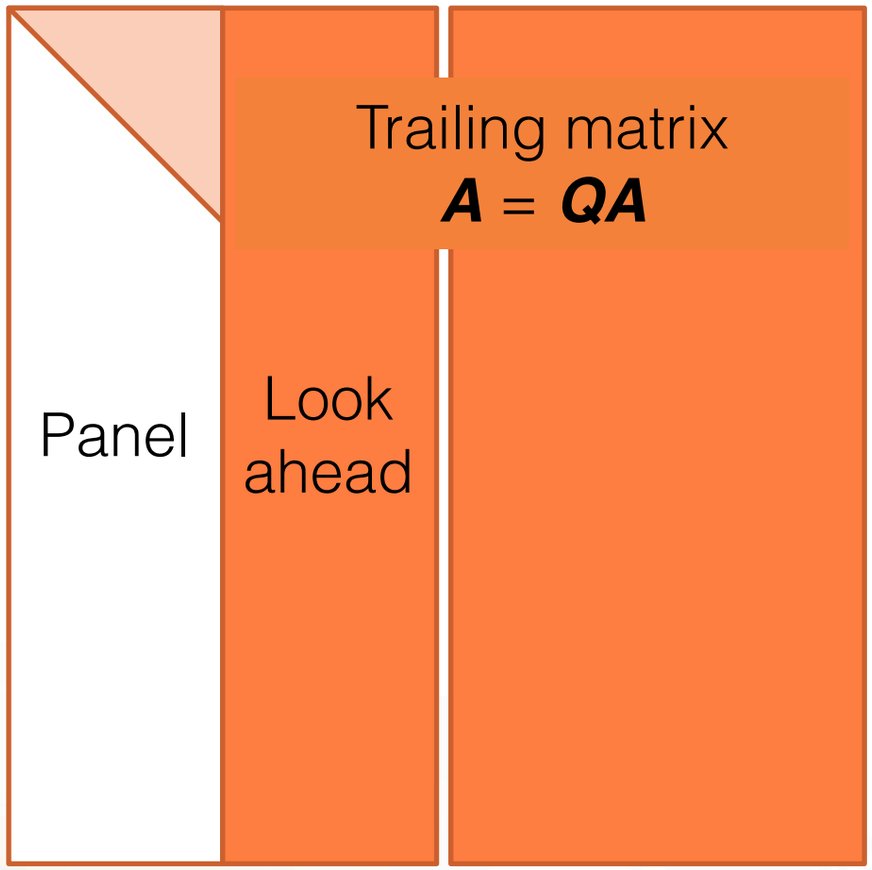
Selected routines; complete list at http://icl.eecs.utk.edu/magma/

# One-sided factorization

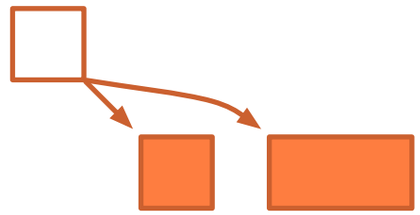
- LU, Cholesky, QR factorizations for solving linear systems

Level 2 BLAS on CPU

Level 3 BLAS on GPU




DAG

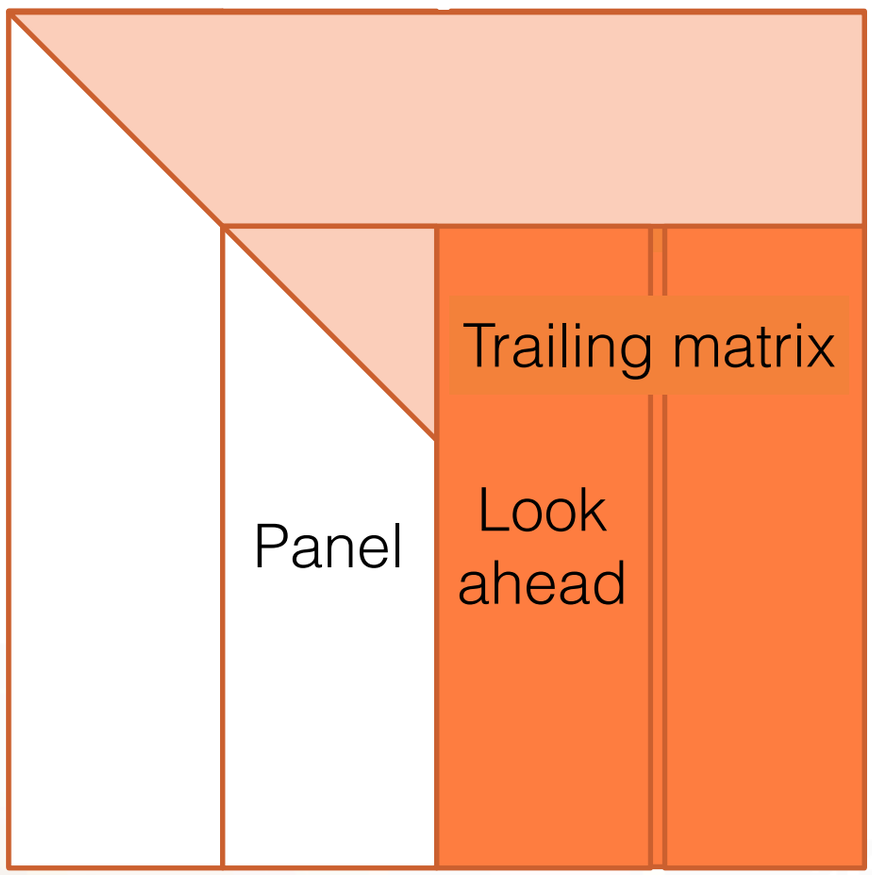


# One-sided factorization

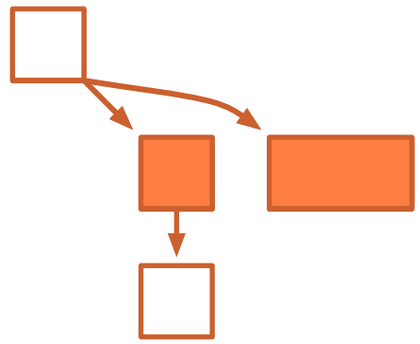
- LU, Cholesky, QR factorizations for solving linear systems

Level 2 BLAS on CPU

Level 3 BLAS on GPU




DAG

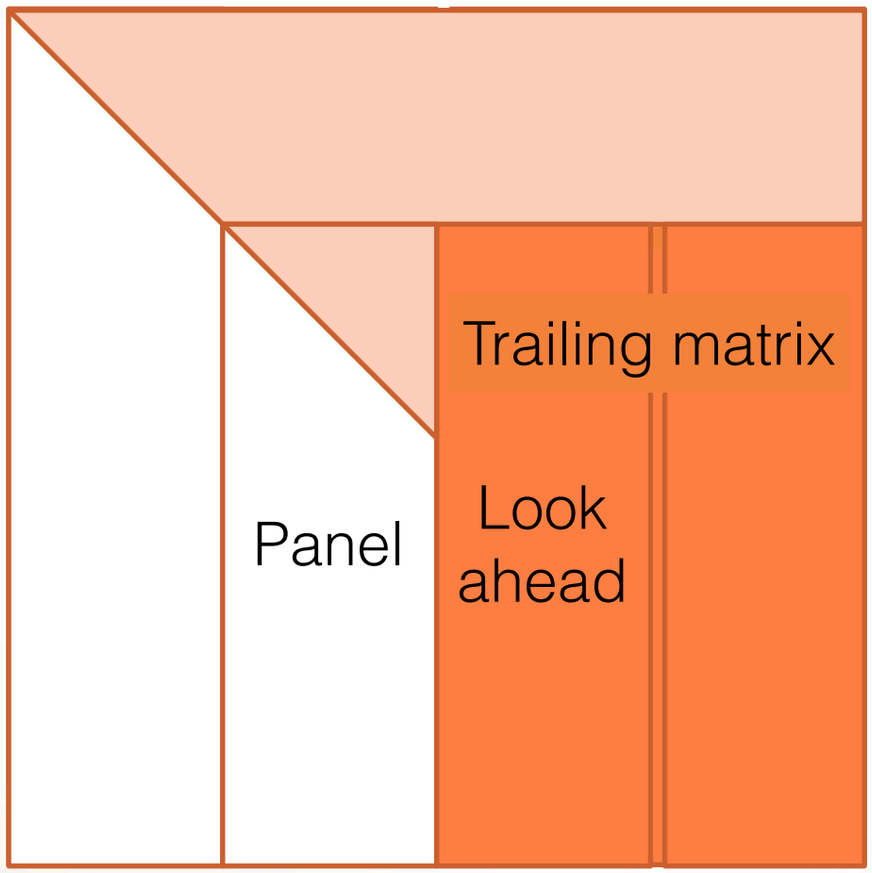


# One-sided factorization

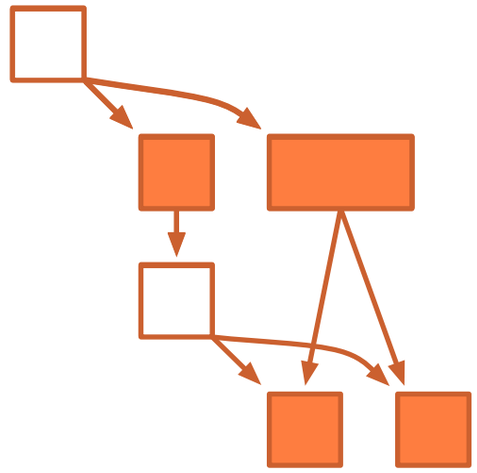
- LU, Cholesky, QR factorizations for solving linear systems

Level 2 BLAS on CPU

Level 3 BLAS on GPU

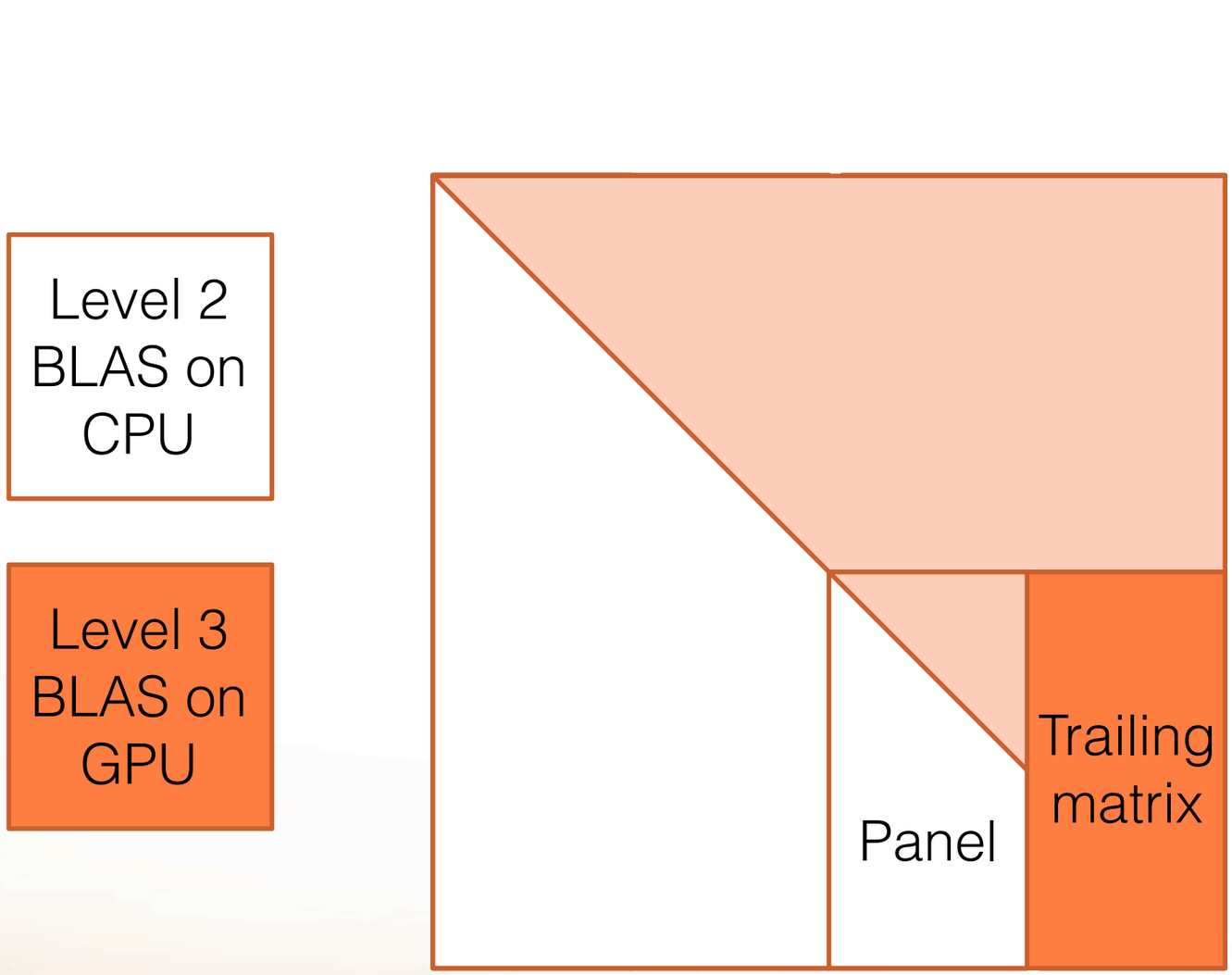


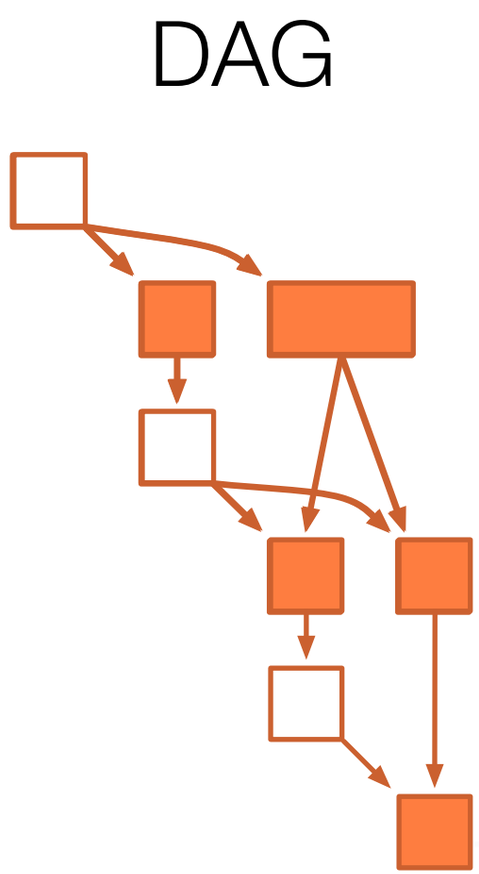

DAG



# One-sided factorization

- LU, Cholesky, QR factorizations for solving linear systems



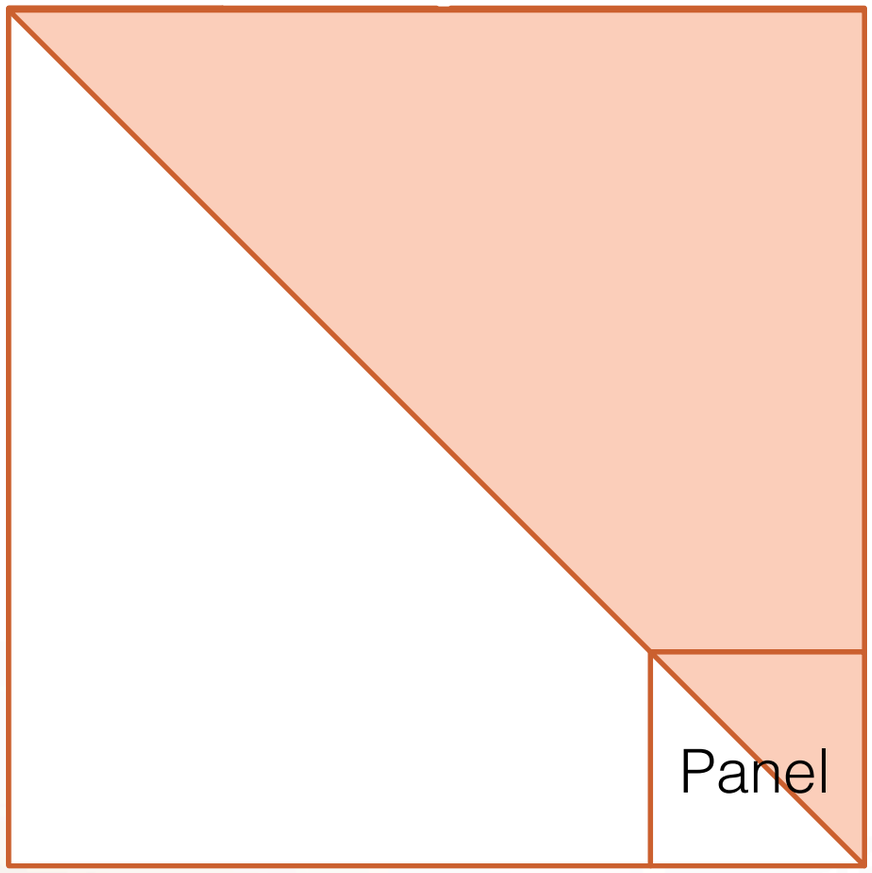



# One-sided factorization

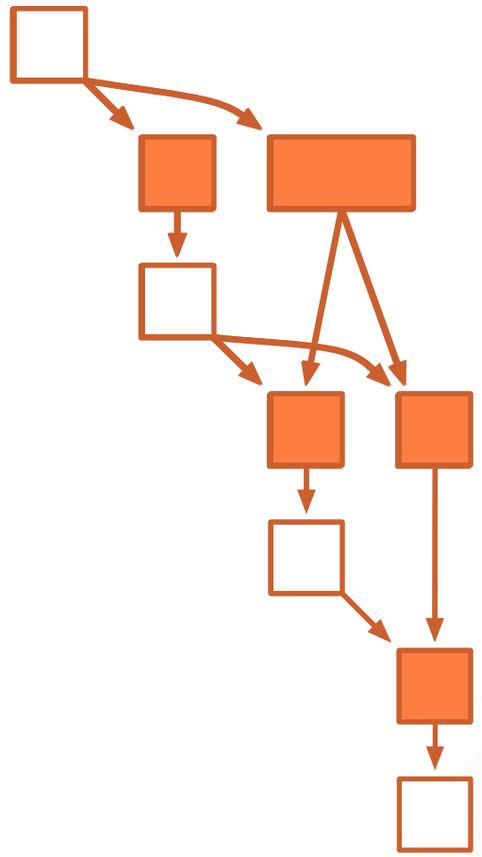
- LU, Cholesky, QR factorizations for solving linear systems

Level 2 BLAS on CPU

Level 3 BLAS on GPU




DAG

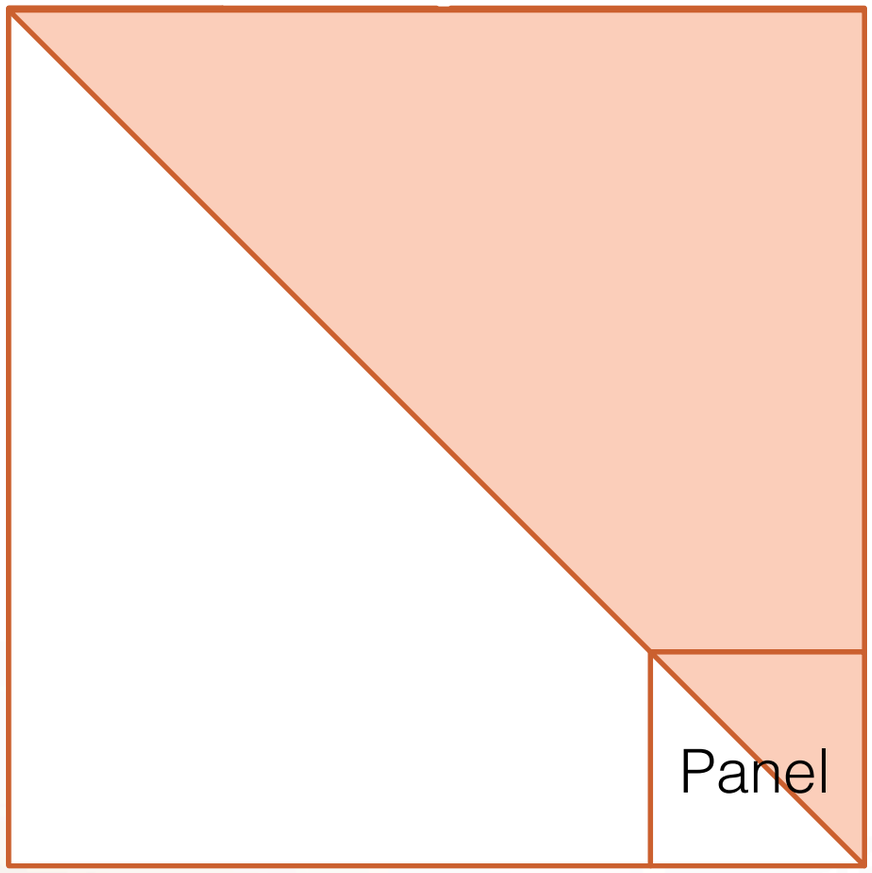


# One-sided factorization

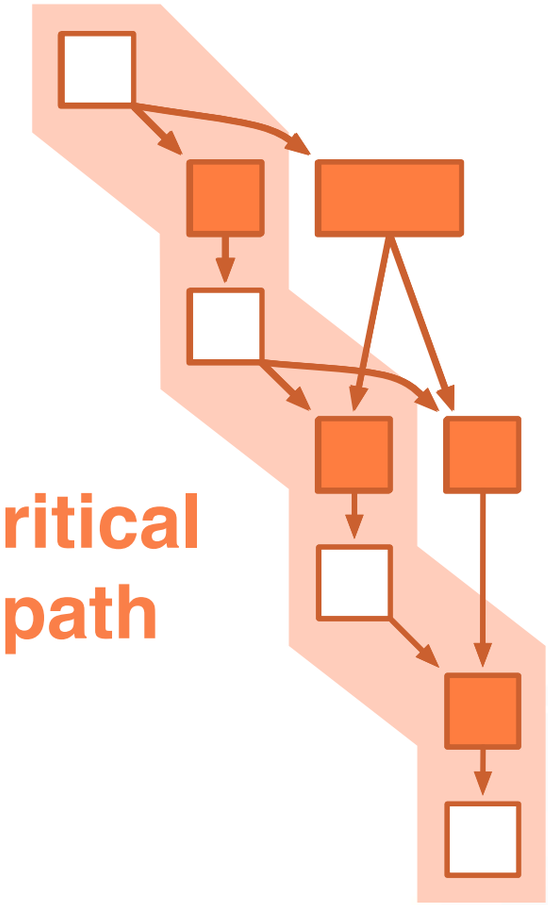
- LU, Cholesky, QR factorizations for solving linear systems

Level 2 BLAS on CPU

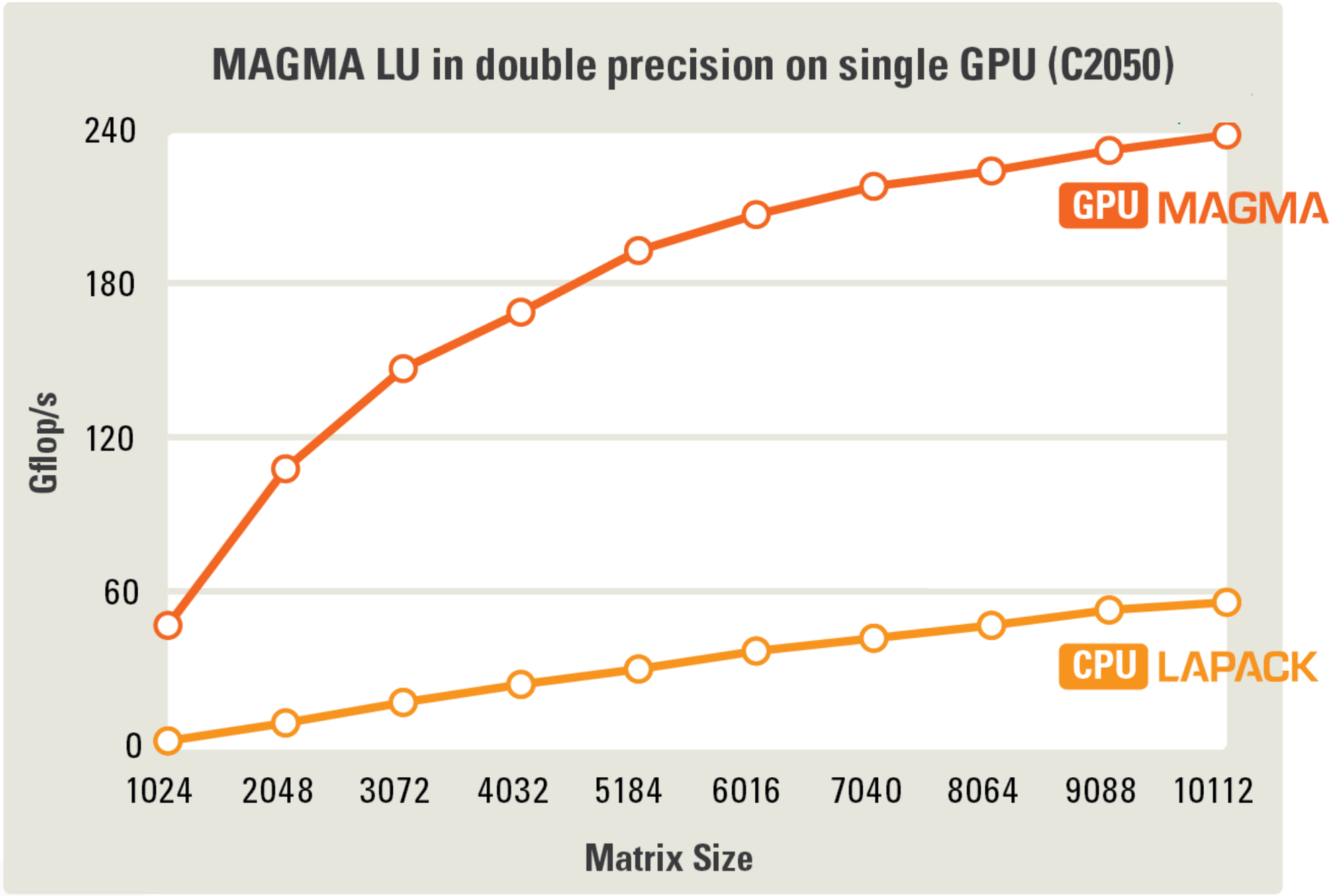
Level 3 BLAS on GPU




DAG



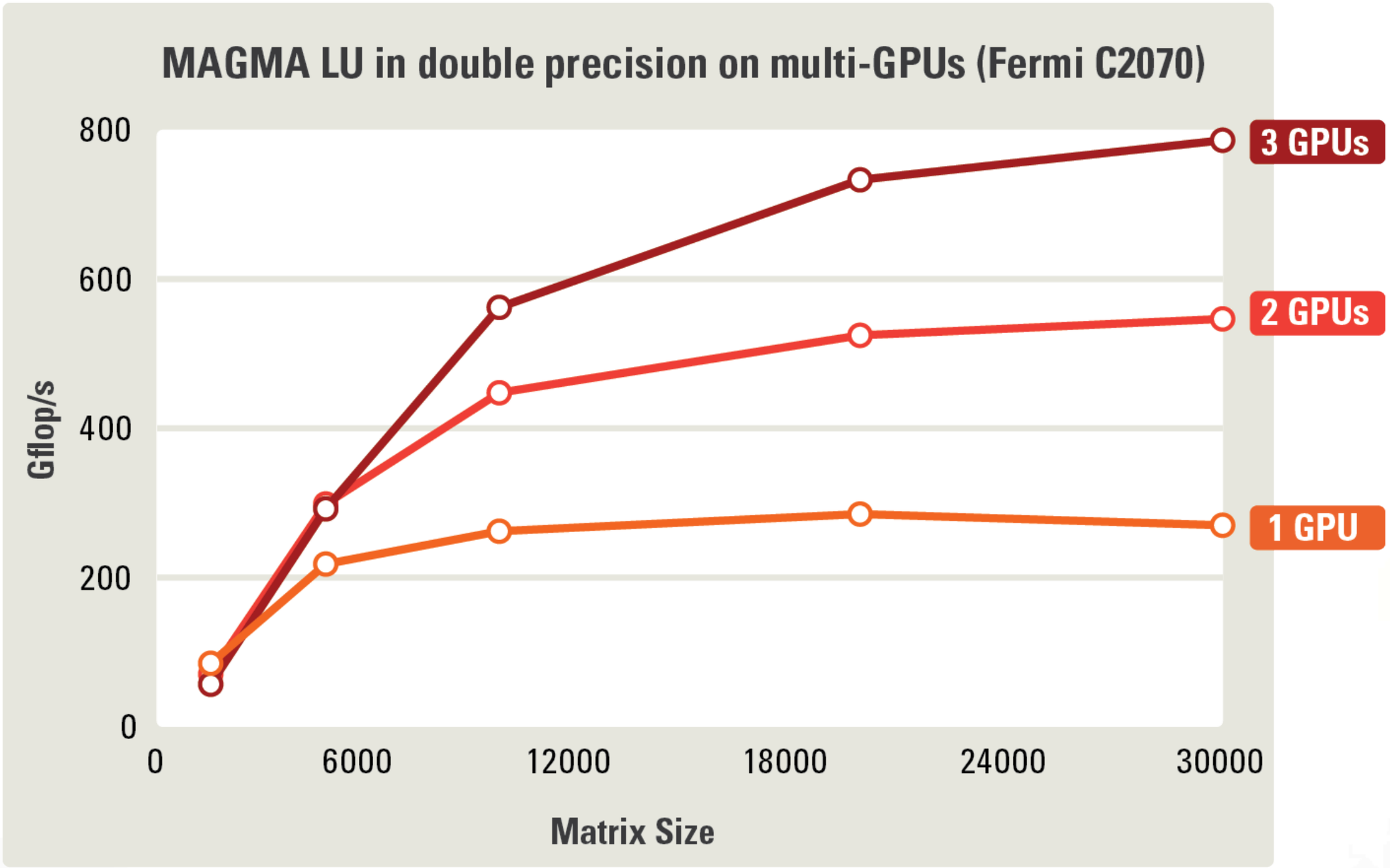
critical path




**GPU** Fermi C2050 (448 CUDA Cores @ 1.15 GHz)
+ Intel Q9300 (4 cores @ 2.50 GHz)
DP peak **515** + **40** GFlop/s
Power * ~**220** W

**CPU** AMD Istanbul
[ 8 sockets x 6 cores (48 cores) @2.8GHz ]
DP peak **538** GFlop/s
Power * ~**1,022** W

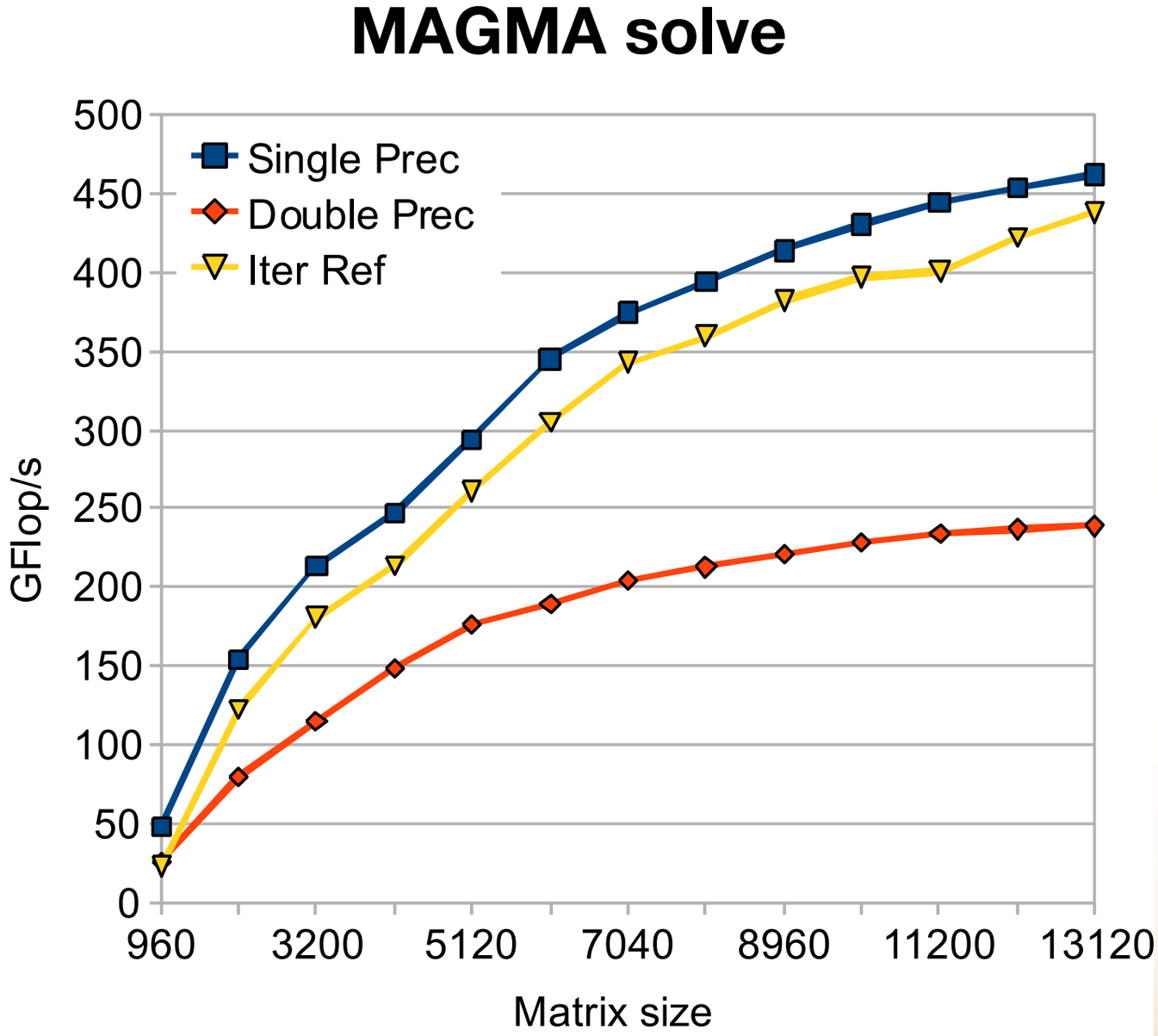
* Computation consumed power rate (total system rate minus idle rate), measured with KILL A WATT PS, Model P430




**Keeneland system, using one node**
3 Nvidia GPUs (M2070 @ 1.1 GHz, 5.4 GB)
2 x 6 Intel Cores (X5660 @ 2.8 GHz, 23 GB)

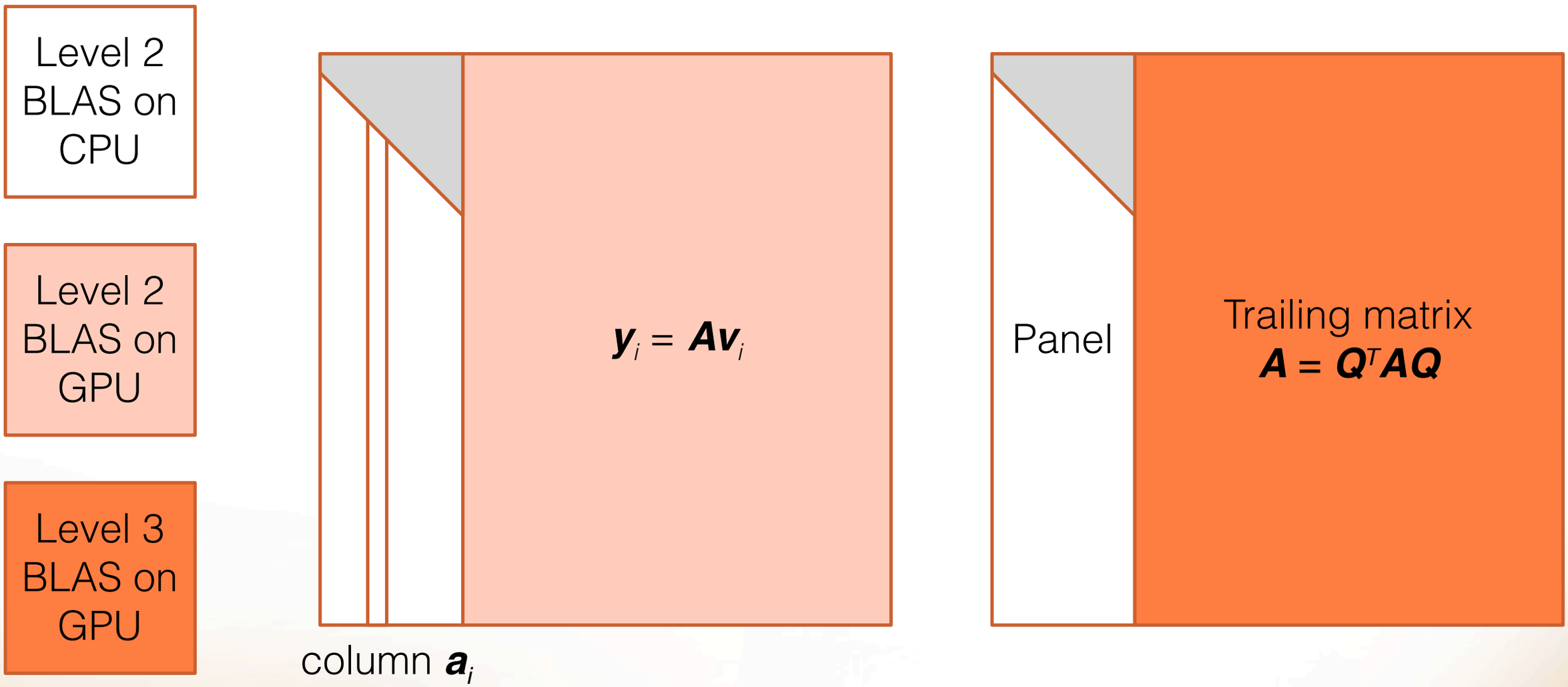
# Mixed-precision solvers

- Factor in single precision
- Iterative refinement yields double precision accuracy

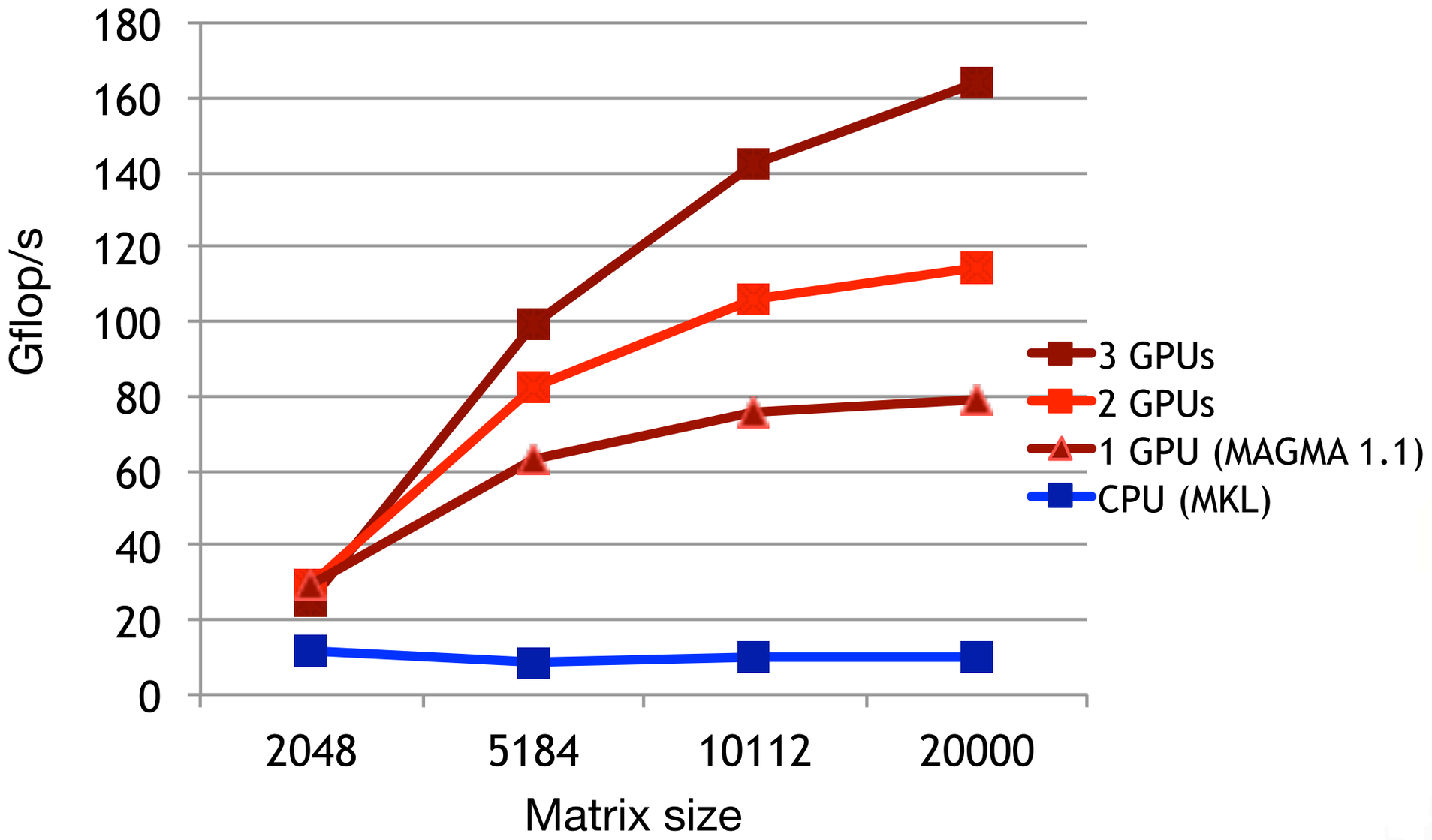


# Two-sided factorization

- Hessenberg, tridiagonal factorizations for eigenvalue problems



# MAGMA Hessenberg in double precision



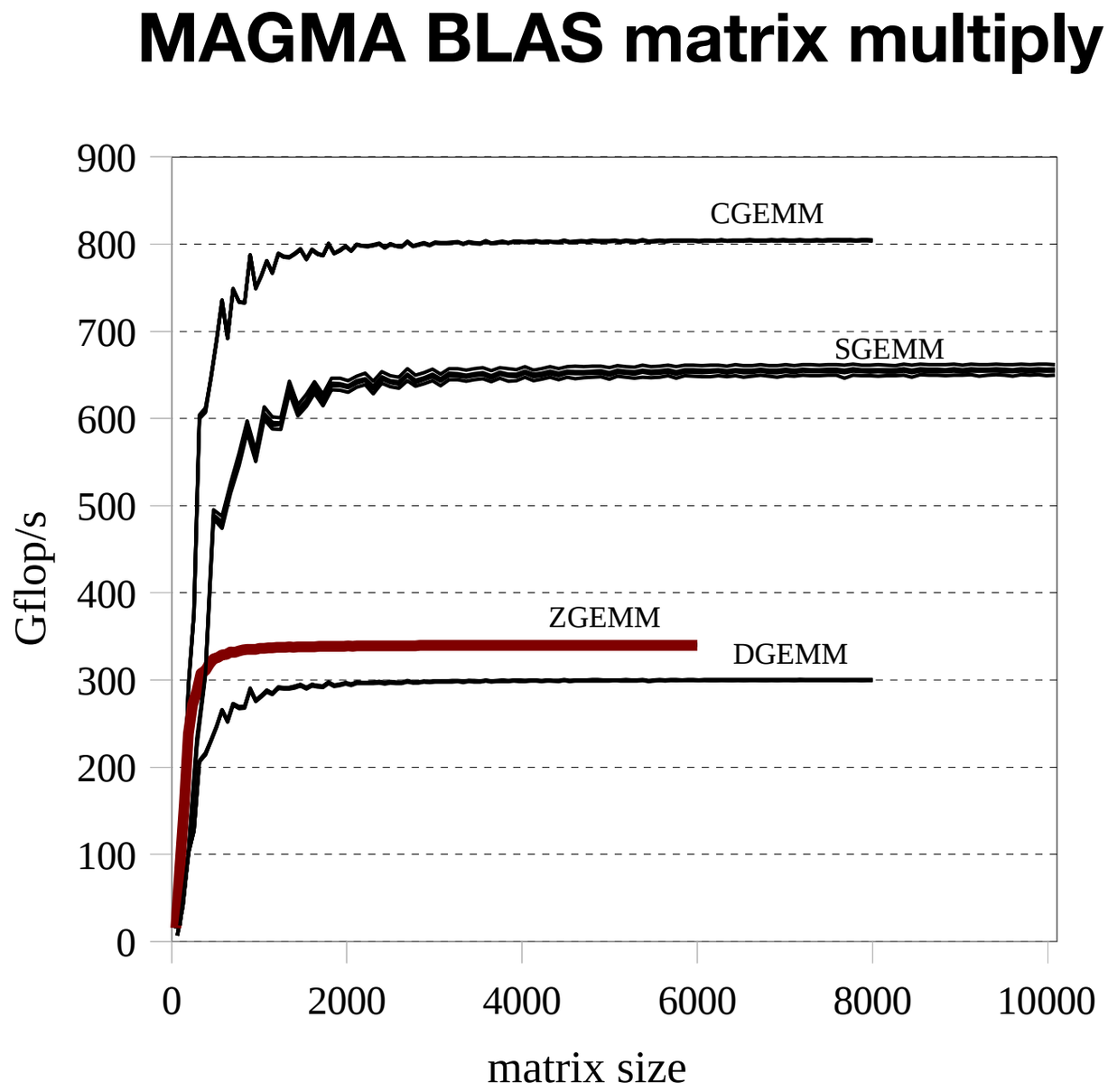

**Keeneland system, using one node**

3 Nvidia GPUs (M2070 @ 1.1 GHz, 5.4 GB)

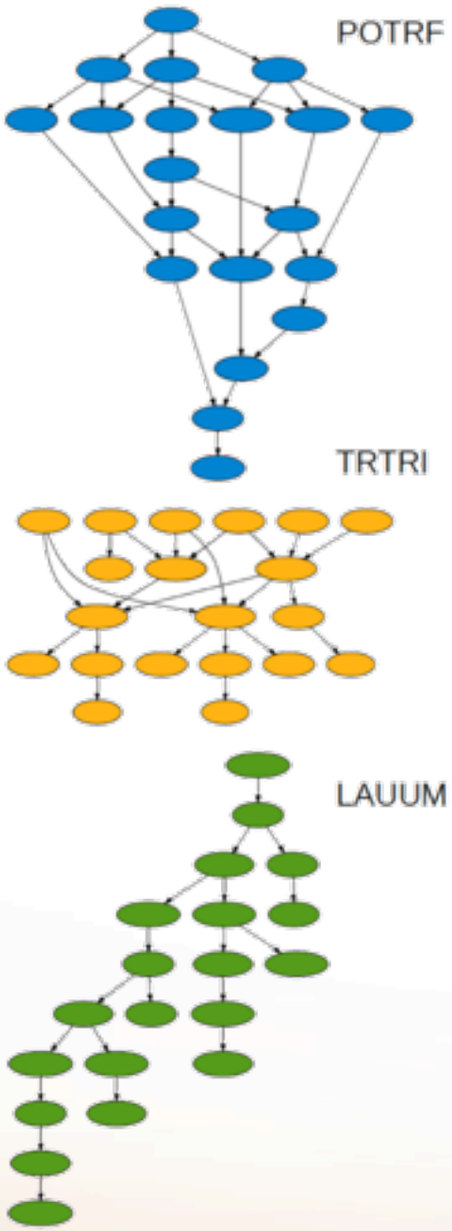
2 x 6 Intel Cores (X5660 @ 2.8 GHz, 23 GB)

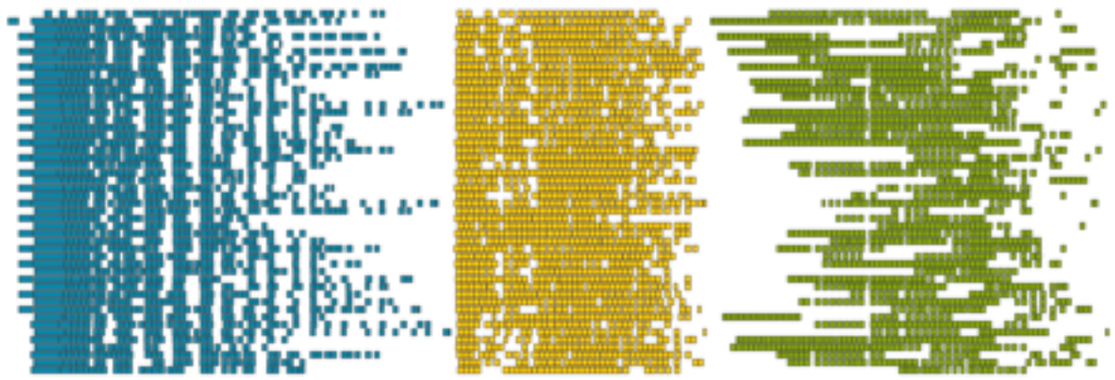
# Autotuning kernels

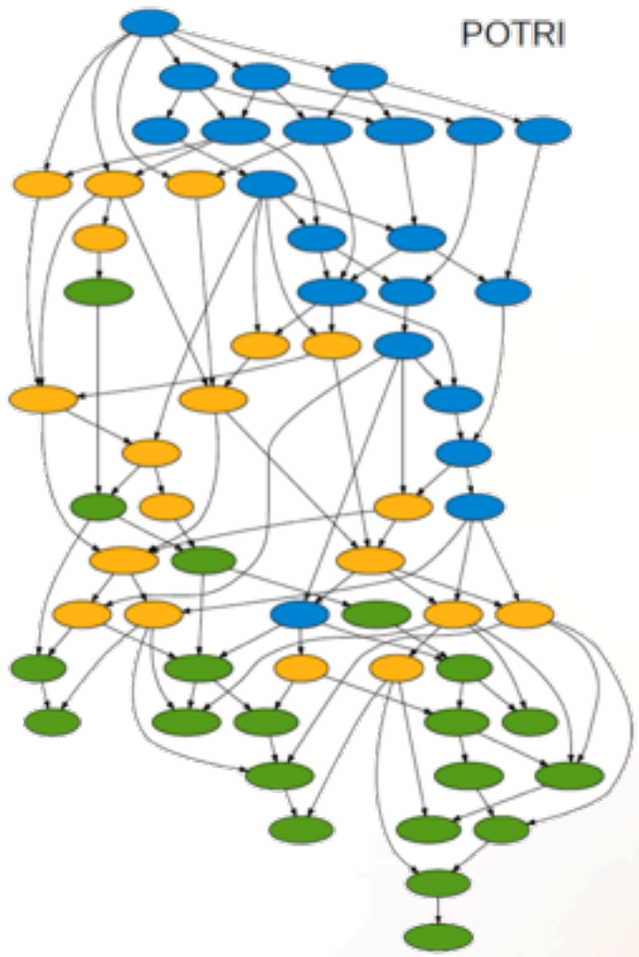
- Parameterize code, e.g., blocksize
- Test 100 – 500 kernels
- Improved zgemm from 308 to 341 Gflops/s
- Improved up to 2x on specific rectangular shapes



# Scheduling DAGs






48 cores
Matrix is 4000 x 4000, tile is 200 x 200.

# Dynamic scheduling

- Parallelism using DAG-based runtime scheduler
- One-sided factorizations and solvers using StarPU

```
// Sequential Tile Cholesky
for k = 1 .. ntiles
   dpotrf( Akk )
   for i = k+1 .. ntiles
      dtrsm( Akk, Aik )
   for i = k+1 .. ntiles
      dsyrk( Aik, Aii )
      for j = i+1 .. ntiles
         dgemm( Ajk, Aik, Aij )
```

```
// Hybrid Tile Cholesky
for k = 1 .. ntiles
   Insert_Task( dpotrf, ... )
   for i = k+1 .. ntiles
      Insert_Task( dtrsm, ... )
   for i = k+1 .. ntiles
      Insert_Task( dsyrk, ... )
      for j = i+1 .. ntiles
         Insert_Task( dgemm, ... )
```

# Documentation

- Nvidia Guides
  http://developer.nvidia.com/nvidia-gpu-computing-documentation
- MAGMA
  http://icl.eecs.utk.edu/magma
- PLASMA
  http://icl.eecs.utk.edu/plasma
- BLAS and LAPACK index
  http://web.eecs.utk.edu/~mgates3/docs/lapack.html

# Collaborators / Support

- University of Tennessee, Knoxville
- University of California, Berkeley
- University of Colorado, Denver
- INRIA, France
- KAUST, Saudi Arabia

INNOVATIVE COMPUTING LABORATORY

THE UNIVERSITY of TENNESSEE

http://icl.eecs.utk.edu/