GPU TECHNOLOGY CONFERENCE

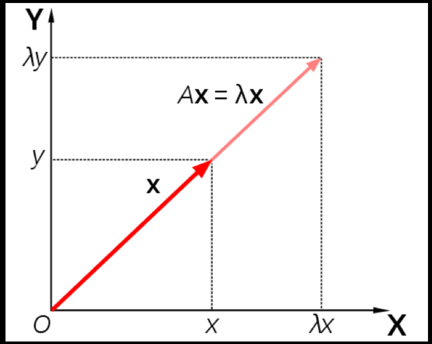
MAGMA: *A Breakthrough in Solvers for Eigenvalue Problems*

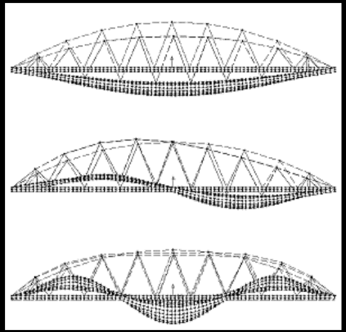
Stan Tomov w/ J. Dongarra, A. Haidar, I. Yamazaki, T. Dong
T. Schulthess (ETH), and R. Solca (ETH)
University of Tennessee

Eigenvalue and eigenvectors

- $A x = \lambda x$
  - Quantum mechanics (Schrödinger equation)
  - Quantum chemistry
  - Principal component analysis (in data mining)
  - Vibration analysis (of mechanical structures)
  - Image processing, compression, face recognition
  - Eigenvalues of graph, e.g., in Google's page rank

  . . .





- To solve it fast
  [acceleration analogy - car @ 64 mph *vs* speed of sound !]

T. Dong, J. Dongarra, S. Tomov, I. Yamazaki, T. Schulthess, and R. Solca, *Symmetric dense matrix-vector multiplication on multiple GPUs and its application to symmetric dense and sparse eigenvalue problems*, ICL Technical report, 03/2012.

J. Dongarra, A. Haidar, T. Schulthess, R. Solca, and S. Tomov, *A novel hybrid CPU- GPU generalized eigensolver for electronic structure calculations based on fine grained memory aware tasks*, ICL Technical report, 03/2012.

The need for eigensolvers

Electronic structure calculations

- Density functional theory

Many-body Schrödinger equation (exact but exponential scaling)

$$\{-\sum_i \frac{1}{2}\nabla_i^2 + \sum_{i,j} \frac{1}{|r_i - r_j|} + \sum_{i,I} \frac{Z}{|r_i - R_I|}\}\Psi(r_1,..r_N) = E\Psi(r_1,..r_N)$$

- Nuclei fixed, generating external potential (system dependent, non-trivial)
- N is number of electrons

Kohn Sham Equation: The many body problem of interacting electrons is reduced to non-interacting electrons (single particle problem) with the same electron density and a different effective potential (cubic scaling).

$$\{-\frac{1}{2}\nabla^2 + \int \frac{\rho(r')}{|r - r'|}dr' + \sum_I \frac{Z}{|r - R_I|} + V_{XC}\}\psi_i(r) = E_i\psi_i(r)$$

$$\rho(r) = \sum_i |\psi_i(r)|^2 = |\Psi(r_1,..r_N)|^2$$

- V_{XC} represents effects of the Coulomb interactions between electrons
- ρ is the density (of the original many-body system)

V_{XC} is not known except special cases ⇒ use approximation, e.g. Local Density Approximation (LDA) where V_{XC} depends only on ρ

A model leading to self-consistent iteration computation with need for HP LA (e.g, **diagonalization** and **orthogonalization**)

The need for eigensolvers

- Schodinger equation:

$$H\psi = E\psi$$

- Choose a basis set of wave functions
- Two cases:
  - Orthonormal basis:

    $$H\,x = E\,x$$

    in general it needs a big basis set
  - Non-orthonormal basis:

    $$H\,x = E\,S\,x$$

Hermitian Generalized Eigenproblem

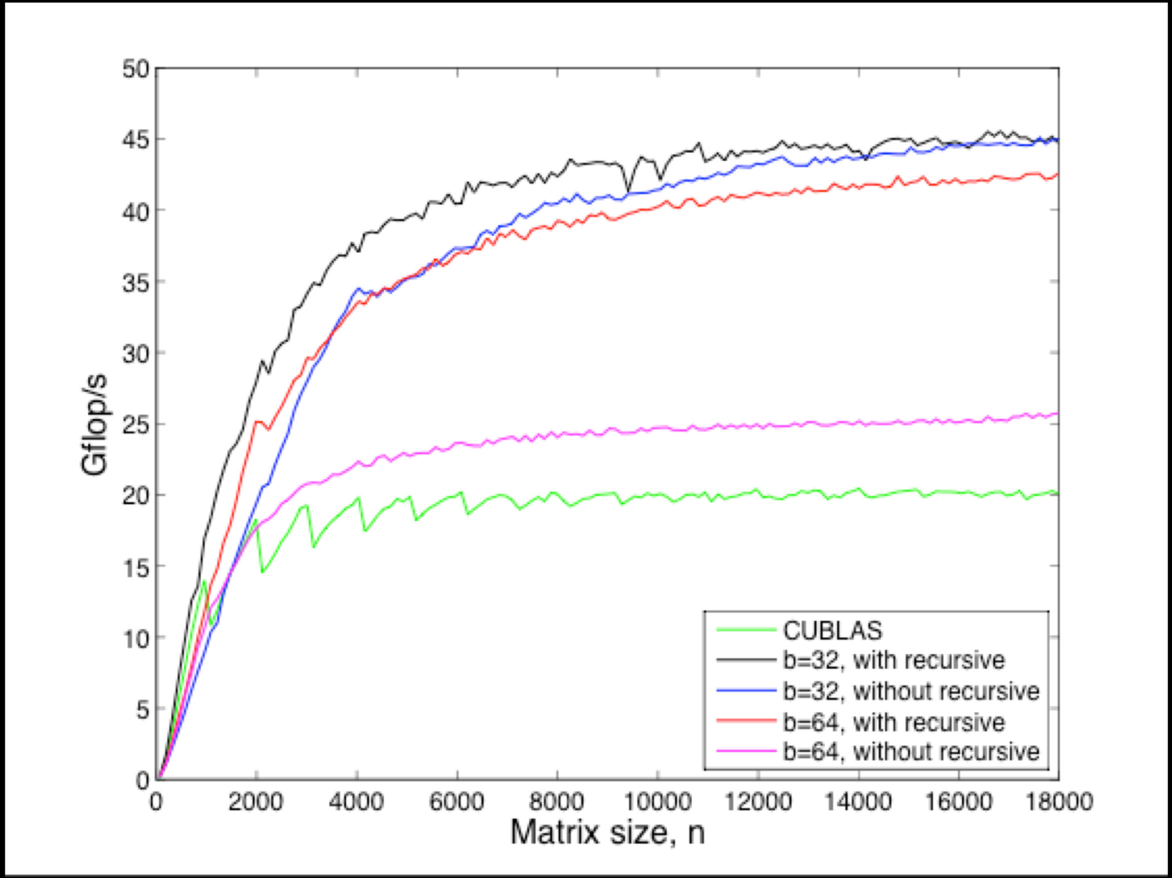
Solve $A\,x = \lambda\,B\,x$

1) Compute the Cholesky factorization of $B = LL^H$
2) Transform the problem to a standard eigenvalue problem $\tilde{A} = L^{-1}AL^{-H}$
3) Solve Hermitian standard Eigenvalue problem $\tilde{A}\,y = \lambda y$
   - Tridiagonalize $\tilde{A}$ **(50% of its flops are in Level 2 BLAS SYMV)**
   - Solve the tridiagonal eigenproblem
   - Transform the eigenvectors of the tridiagonal to eigenvectors of $\tilde{A}$
4) Transform back the eigenvectors $x = L^{-H}\,y$

GPU TECHNOLOGY CONFERENCE

Fast BLAS development

Performance of MAGMA DSYMVs vs CUBLAS



$$y = \alpha A x + \beta y$$

Keeneland system, using one node

3 NVIDIA GPUs (M2090@ 1.55 GHz, 5.4 GB)

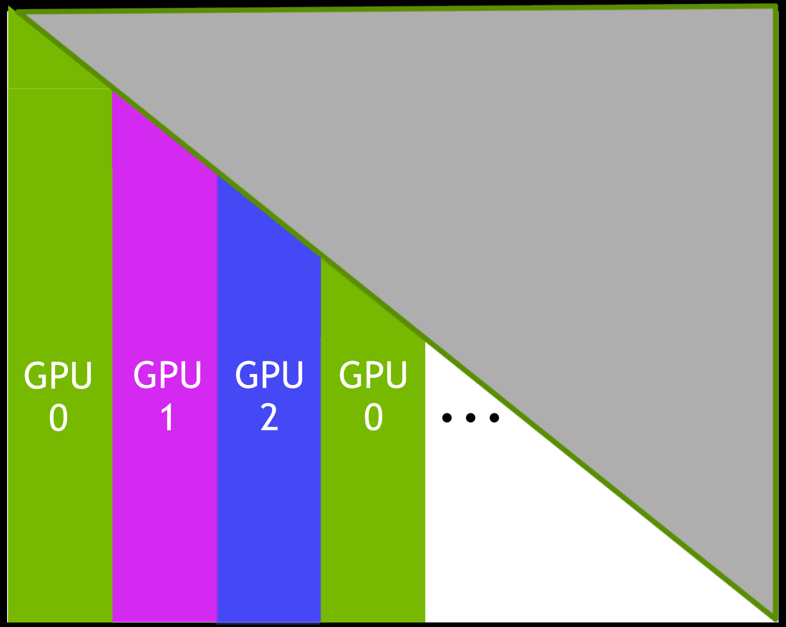
2 x 6 Intel Cores (X5660 @ 2.8 GHz, 23 GB)

Parallel SYMV on multiple GPUs

- Multi-GPU algorithms were developed
  - 1-D block-cyclic distribution
  - Every GPU
    - has a copy of x
    - Computes $y_i = \alpha A_i$ where A_i is the local for GPU i matrix
    - Reuses the single GPU kernels
  - The final result

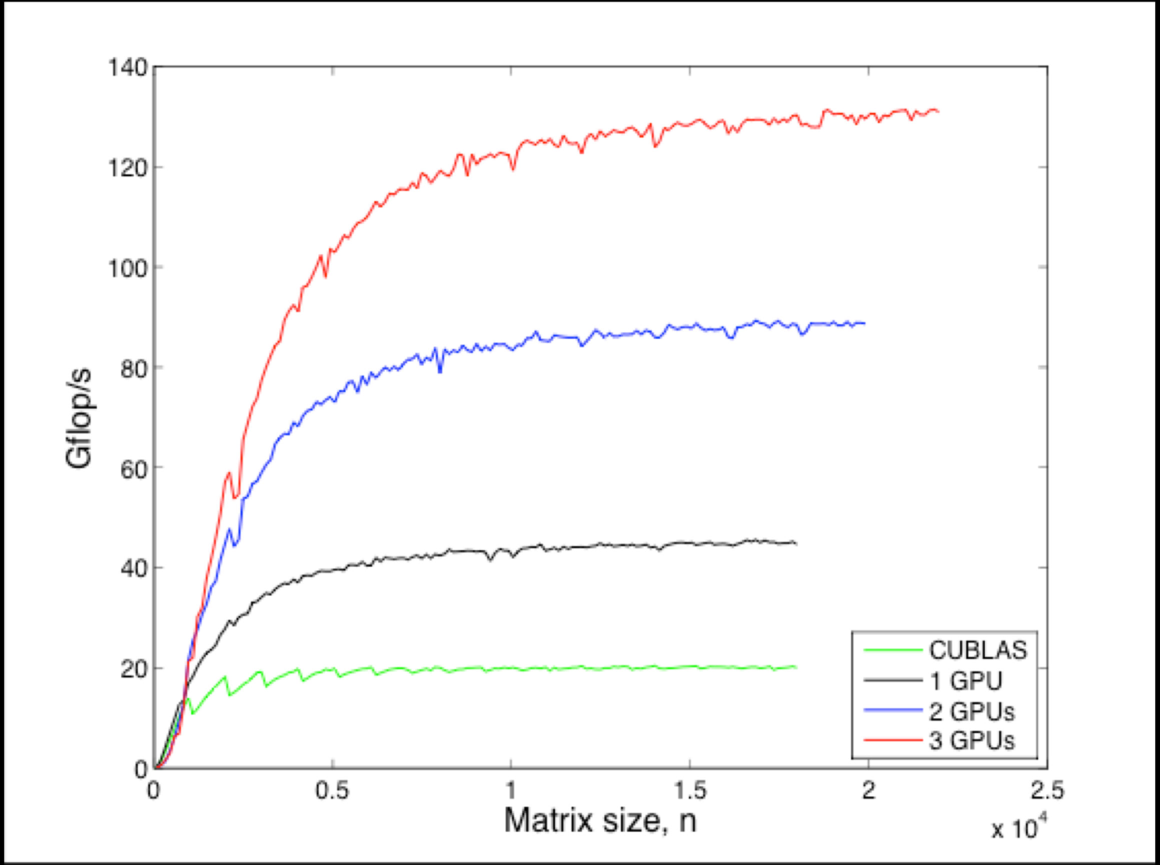
$$y = \sum_{0}^{\#GPUs-1} y_i + \beta y$$

is computed on the CPU



Parallel SYMV on multiple GPUs

Performance of MAGMA DSYMV on multi M2090 GPUs



Keeneland system, using one node

3 NVIDIA GPUs (M2090@ 1.55 GHz, 5.4 GB)

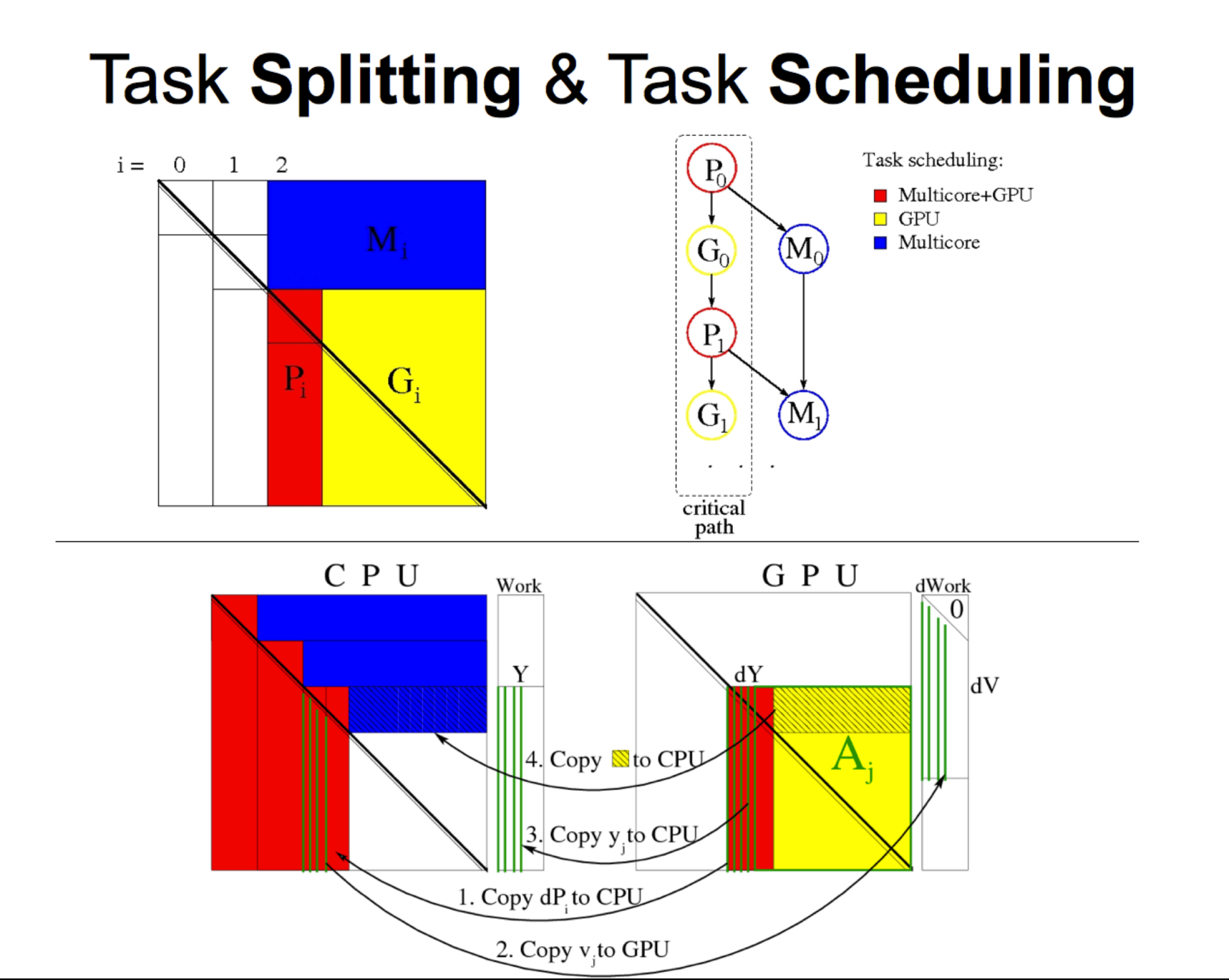
2 x 6 Intel Cores (X5660 @ 2.8 GHz, 23 GB)

Hybrid Algorithms

Two-sided factorizations (to bidiagonal, tridiagonal, and upper Hessenberg forms) for eigen- and singular-value problems

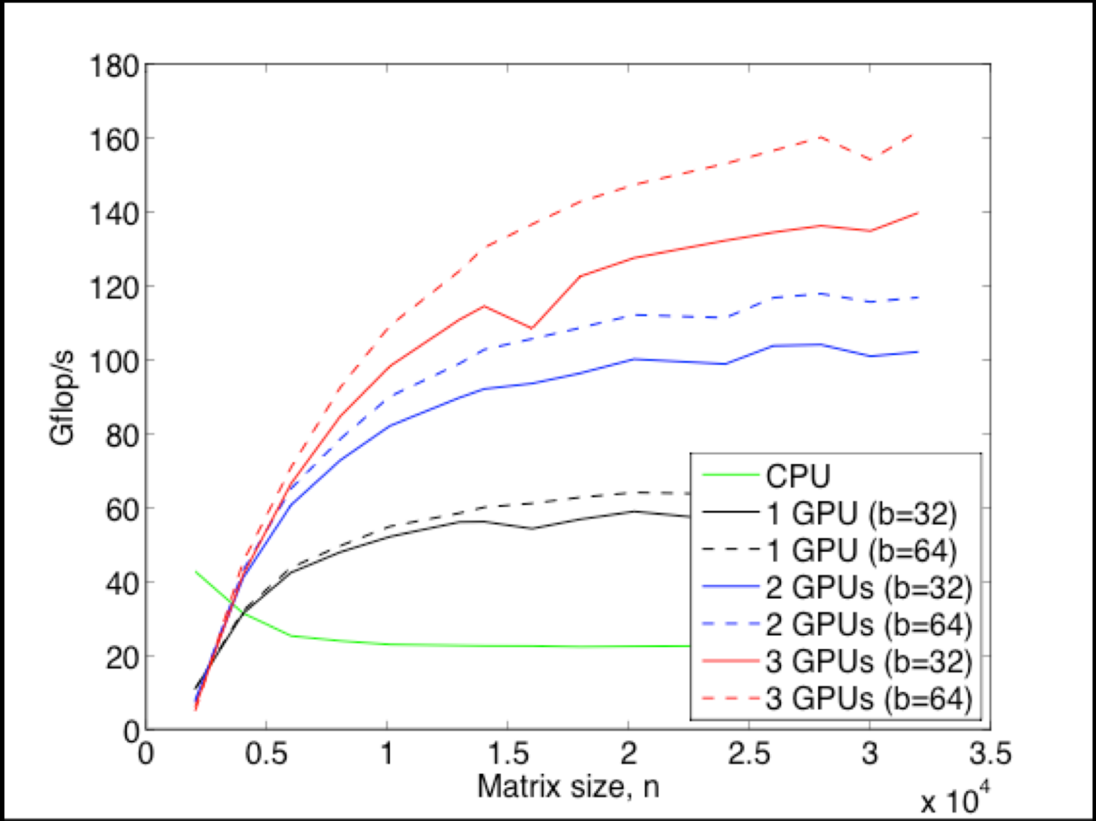
- Hybridization
  - Trailing matrix updates (Level 3 BLAS) are done on the GPU (similar to the one-sided factorizations)
  - Panels (Level 2 BLAS) are hybrid
    - operations with memory footprint restricted to the panel are done on CPU
    - The time consuming matrix-vector products involving the entire trailing matrix are done on the GPU

Hybrid Two-Sided Factorizations



From fast BLAS to fast tridiagonalization

Performance of MAGMA DSYTRD on multi M2090 GPUs



- 50 % of the flops are in SYMV
- Memory bound, i.e. does not scale well on multicore CPUs
- Use the GPU's high memory bandwidth and optimized SYMV
- 8 x speedup over 12 Intel cores (X5660 @2.8 GHz)

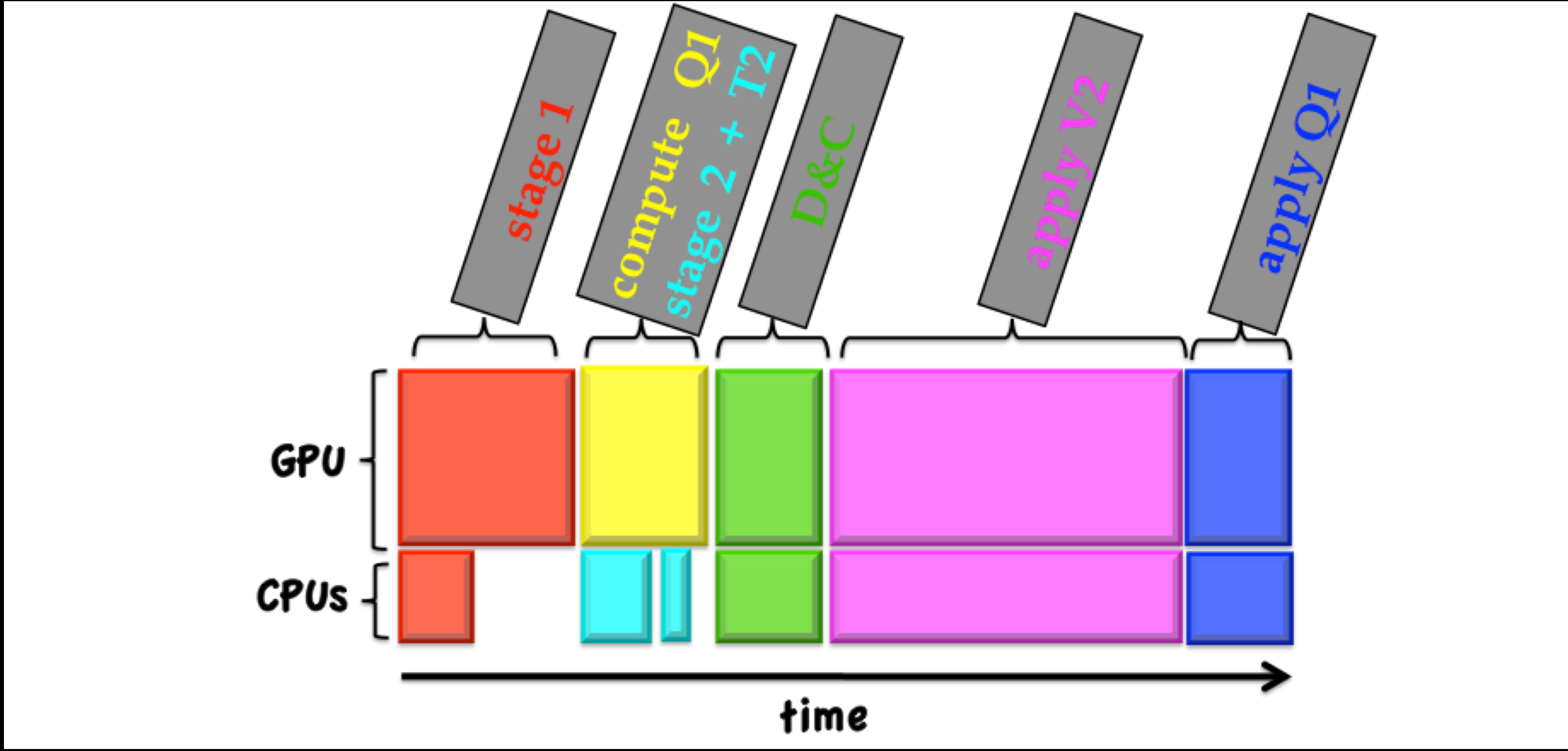
Keeneland system, using one node
3 NVIDIA GPUs (M2090@ 1.55 GHz, 5.4 GB)
2 x 6 Intel Cores (X5660 @ 2.8 GHz, 23 GB)

Can we accelerate 4 x more ?

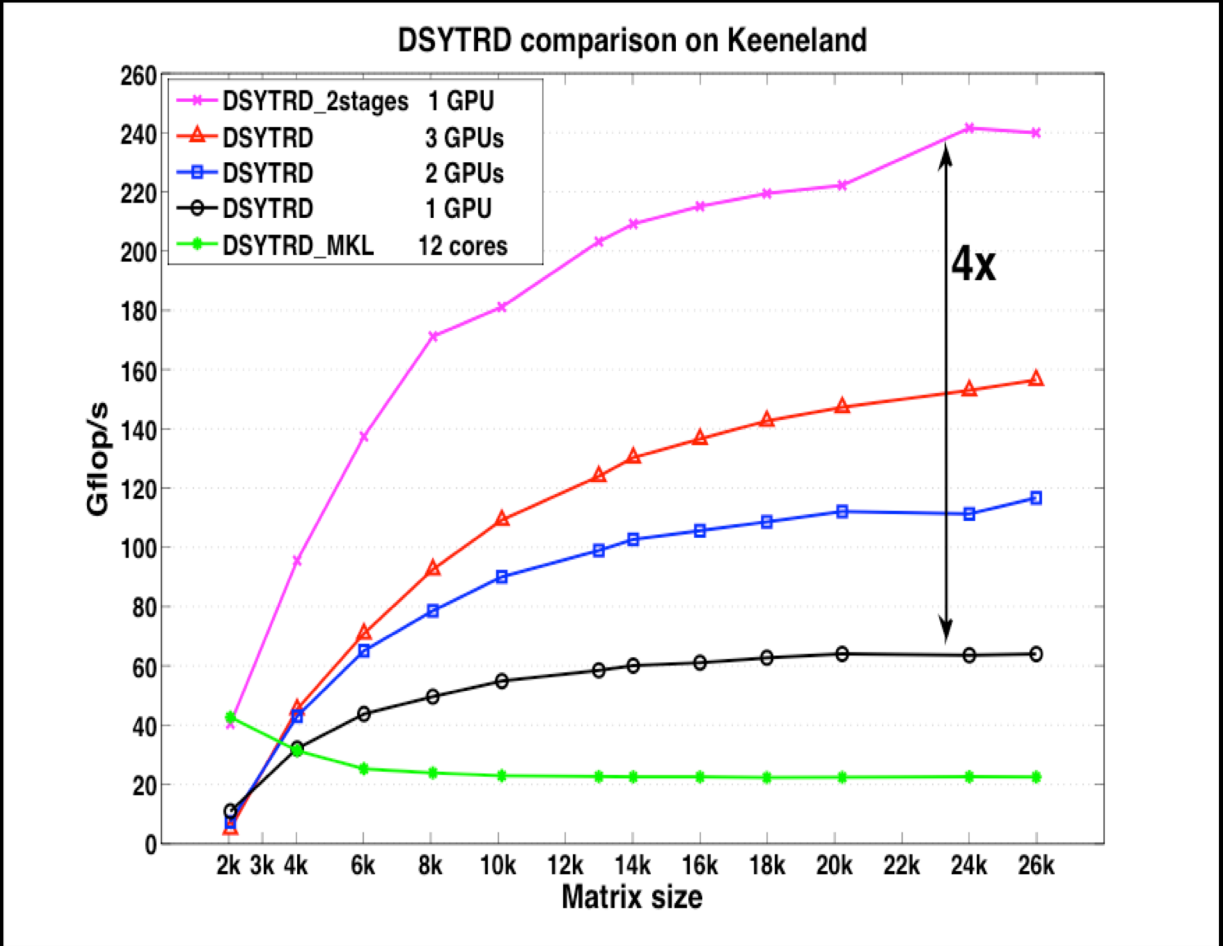
A two-stages approach

- Increases the computational intensity by introducing
  - 1st stage: reduce the matrix to band
    [Level 3 BLAS; implemented very efficiently on GPU using “look-ahead”]
  - 2nd stage: reduce the band to tridiagonal
    [memory bound, but we developed a very efficient “bulge” chasing algorithm with memory aware tasks for multicore to increase the computational intensity]

Schematic profiling of the eigensolver



An additional 4 x speedup !



- 12 x speedup over 12 Intel cores (X5660 @2.8 GHz)

Keeneland system, using one node
3 NVIDIA GPUs (M2090@ 1.55 GHz, 5.4 GB)
2 x 6 Intel Cores (X5660 @ 2.8 GHz, 23 GB)

Conclusions

- Breakthrough eigensolver using GPUs
- Number of fundamental numerical algorithms for GPUs (BLAS and LAPACK type)
- Released in MAGMA 1.2
- Enormous impact in technical computing and applications
- 12 x speedup w/ a Fermi GPU vs state-of-the-art multicore system (12 Intel Core X5660 @2.8 GHz)
  - From a speed of car to the speed of sound !

Colloborators / Support

- MAGMA [Matrix Algebra on GPU and Multicore Architectures] team
  http://icl.cs.utk.edu/magma/
- PLASMA [Parallel Linear Algebra for Scalable Multicore Architectures] team
  http://icl.cs.utk.edu/plasma
- Collaborating partners

  University of Tennessee, Knoxville
  University of California, Berkeley
  University of Colorado, Denver

  INRIA, France
  KAUST, Saudi Arabia

NVIDIA.
The MathWorks
NSF
U.S. DEPARTMENT OF ENERGY