Intel Parallel Computing Center
The Innovative Computing Laboratory
The University of Tennessee, Knoxville

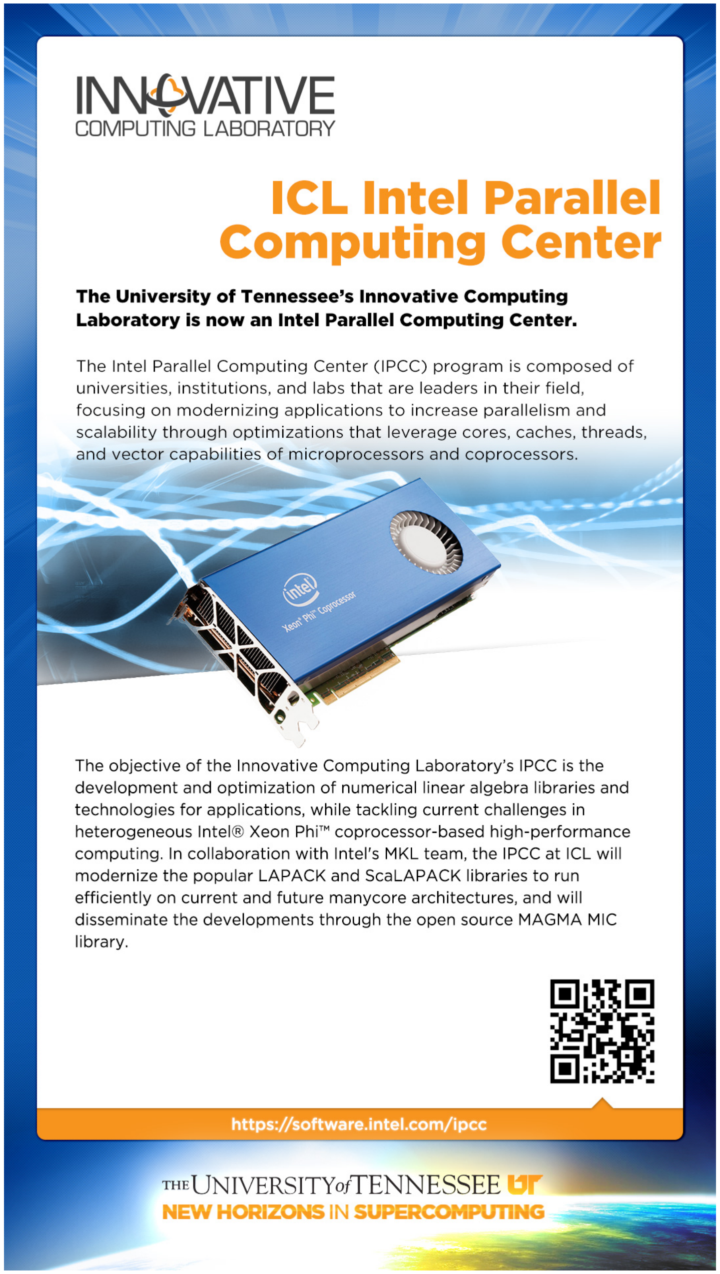
MAGMA MIC 1.3 Release
Optimizing Linear Algebra for Applications on Intel Xeon Phi Coprocessors

J. Dongarra, M. Gates, A. Haidar, K. Kabir, P. Luszczek, S. Tomov, and I. Yamazaki

Innovative Computing Laboratory
Department of Computer Science
University of Tennessee, Knoxville

November 15, 2014

IPCC at ICL


INNOVATIVE
COMPUTING LABORATORY
ICL Intel Parallel Computing Center
The University of Tennessee's Innovative Computing Laboratory is now an Intel Parallel Computing Center.
The Intel Parallel Computing Center (IPCC) program is composed of universities, institutions, and labs that are leaders in their field, focusing on modernizing applications to increase parallelism and scalability through optimizations that leverage cores, caches, threads, and vector capabilities of microprocessors and coprocessors.
The objective of the Innovative Computing Laboratory's IPCC is the development and optimization of numerical linear algebra libraries and technologies for applications, while tackling current challenges in heterogeneous Intel® Xeon Phi™ coprocessor-based high-performance computing. In collaboration with Intel's MKL team, the IPCC at ICL will modernize the popular LAPACK and ScaLAPACK libraries to run efficiently on current and future manycore architectures, and will disseminate the developments through the open source MAGMA MIC library.
https://software.intel.com/ipcc
THE UNIVERSITY of TENNESSEE UT
NEW HORIZONS IN SUPERCOMPUTING


APPLICATIONS
ScaLAPACK
LAPACK
BLAS
Hardware
intel Atom inside
CORE i7 inside
Xeon inside
Xeon Phi inside

IPCC at ICL

LAPACK and **ScaLAPACK**

- Standard dense linear algebra (DLA) libraries
- Many applications rely on DLA
- Designed in 80/90's for cache-based architectures

Must be redesigned for modern heterogeneous systems with multi/many-core CPUs and coprocessors.

IPCC at ICL

- **Develop**
  - **Next generation LAPACK / ScaLAPACK**
  - **Programming models, and**
  - **Technologies**

  for heterogeneous

  Intel Xeon Phi-based platforms

- **Disseminate developments through the MAGMA MIC library**

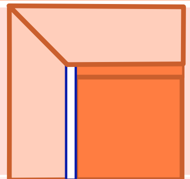
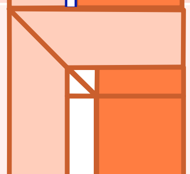
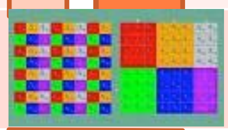
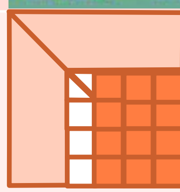
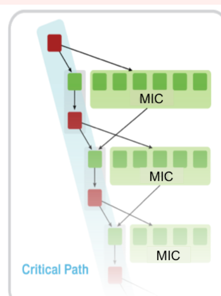
- **High value proposition**

  MAGMA MIC enables ease of use and adoption of Intel Xeon Phi architectures in applications as linear algebra is fundamental to scientific computing

ICL UT INNOVATIVE COMPUTING LABORATORY

THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

A New Generation of Dense Linear Algebra Libraries

Software/Algorithms follow hardware evolution in time		
LINPACK (70's) (Vector operations)		Rely on - Level-1 BLAS operations
LAPACK (80's) (Blocking, cache friendly)		Rely on - Level-3 BLAS operations
ScaLAPACK (90's) (Distributed Memory)		Rely on - PBLAS Mess Passing
PLASMA (00's) New Algorithms (many-core friendly)		Rely on - a DAG/scheduler - block data layout - some extra kernels
MAGMA Hybrid Algorithms (heterogeneity friendly)		Rely on - hybrid scheduler - hybrid kernels

ICL INNOVATIVE COMPUTING LABORATORY

THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

MAGMA MIC
LAPACK for heterogeneous systems

- **MAGMA MIC**
  - Project on the development of a new generation of HP Linear Algebra Libraries
  - To provide LAPACK/ScaLAPACK on heterogeneous Intel Xeon Phi-based systems
  - Well established project with product disseminated through the MAGMA MIC libraries:

MAGMA MIC 0.3 (2012-11-13)

MAGMA MIC 1.0 (2013-05-03)

MAGMA MIC 1.1 (2014-01-07)

MAGMA MIC 1.2 (2014-09-17)

MAGMA MIC 1.3 (2014-11-15)

- For heterogeneous, shared memory systems
- Included are the main factorizations, linear system and eigen-problem solvers
- Open Source Software (http://icl.cs.utk.edu/magma)

- **Collaborators**
  - Intel MKL Team
  - UC Berkeley, UC Denver, INRIA (France), KAUST (Saudi Arabia)
  - Community effort, similar to LAPACK/ScaLAPACK

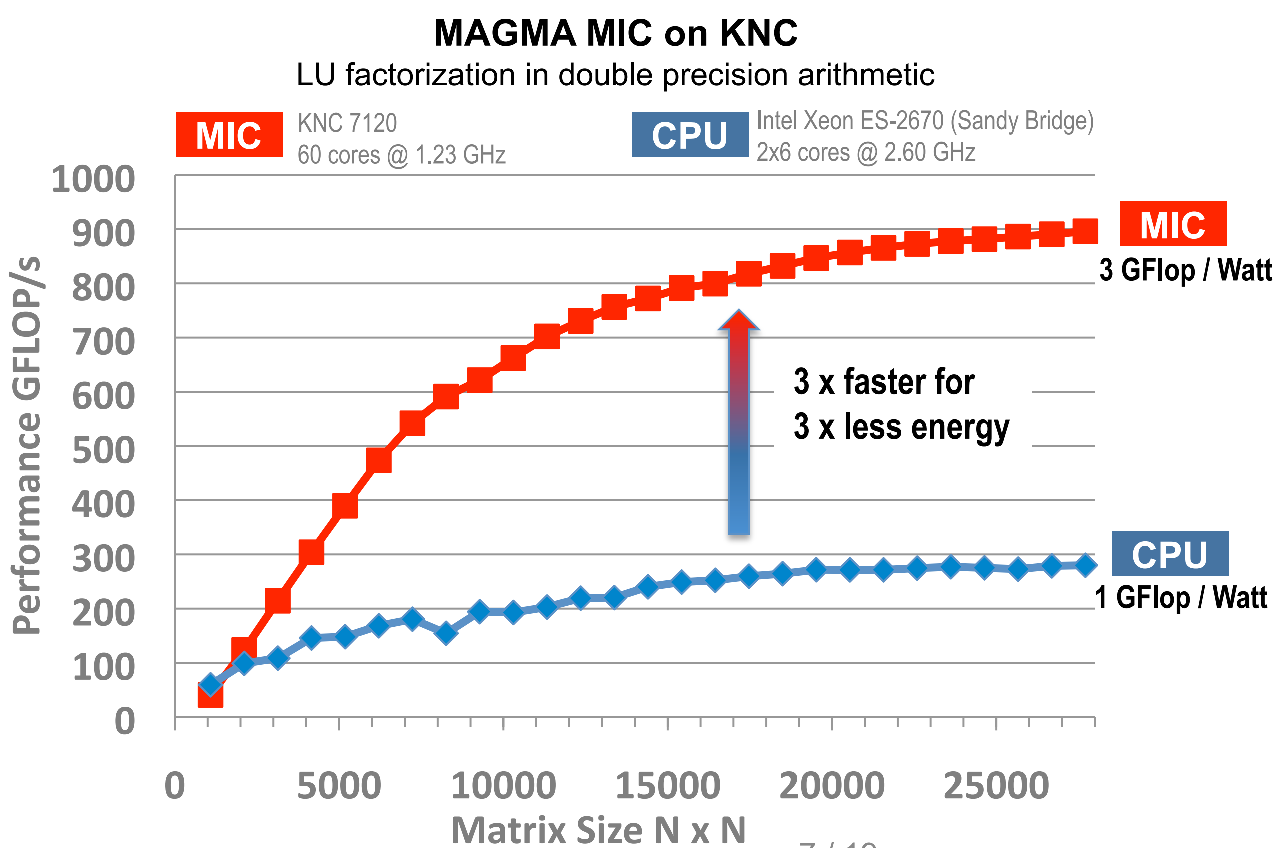
ICL UT INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Key Features of MAGMA MIC

HIBRID ALGORITHMS

MAGMA MIC uses hybrid algorithms where the computation is split into tasks of varying granularity and their execution scheduled over the hardware components. Scheduling can be static or dynamic. In either case, small non-parallelizable tasks, often on the critical path, are scheduled on the CPU, and larger more parallelizable ones, often Level 3 BLAS, are scheduled on the MICs.

PERFORMANCE & ENERGY EFFICIENCY



FEATURES AND SUPPORT

MAGMA MIC 1.3

- Linear system solvers
- Eigen-problem solvers
- SVD
- CPU/AO interface
- MIC/Native interface
- Multiple precision support
- Mixed-precision iter. refinement solvers
- Multicore and multi-MIC support
- LAPACK testing
- Linux

ICL INNOVATIVE COMPUTING LABORATORY

THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Methodology overview

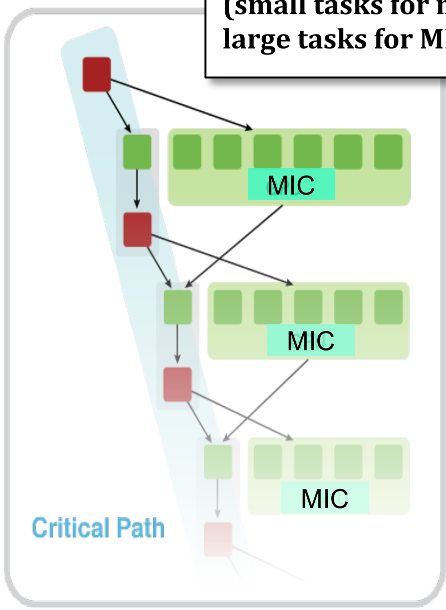
A methodology to use all available resources:

- MAGMA MIC uses hybrid algorithms
  - Representing linear algebra algorithms as collections of tasks and data dependencies among them
  - Properly scheduling tasks' execution over multicore CPUs and manycore coprocessors

- Successfully applied to fundamental linear algebra algorithms
  - One- and two-sided factorizations and solvers
  - Iterative linear and eigensolvers

- Productivity
  1) High level;
  2) Leveraging prior developments;
  3) Exceeding in performance homogeneous solutions

Hybrid CPU+MIC algorithms (small tasks for multicores and large tasks for MICs)



ICL INNOVATIVE COMPUTING LABORATORY

THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

A Hybrid Algorithm Example

Left-looking hybrid Cholesky

From sequential
LAPACK

```
for( j=0, j<n; j+=nb) {
   jb = min(nb, n-j);
   zherk( MagmaUpper
          jb, j, one, dA(0

   if (j+jb < n)
      zgemm( MagmaCo
             dA(0,j), ldd

   zpotrf( MagmaUpper
   if (info != 0)
      *info += j;

   If (j+jb) < n) {

      ztrsm( MagmaLeft,
             jb, n
   }
}
```

to parallel hybrid
MAGMA

```
for( j=0, j<n; j+=nb) {
   jb = min(nb, n-j);
   magma_zherk( MagmaUpper, MagmaConjTrans,
                jb, j, one, dA(0,j), ldda, one, dA(j,j), ldda, queue);
    magma_zgetmatrix_async( jb, jb, dA(j,j), ldda, work, jb, queue, &event);
    if (j+jb < n)
      magma_zgemm( MagmaConjTrans, MagmaNoTrans, jb, n-j-jb, j, one,
                   dA(0,j), ldda, dA(0,j+jb), ldda, one, dA(j, j+jb), ldda, que
    magma_event_sync( event );
    zpotrf( MagmaUpperStr, &jb, work, &jb, info);
    if (info != 0)
       *info += j;
    magma_zsetmatrix_async(jb, jb, work, jb, dA(j, j), ldda, queue, &event);
    If (j+jb) < n) {
      magma_event_sync( event );
      magma_ztrsm( MagmaLeft, MagmaUpper, MagmaConjTrans, MagmaNo
                   jb, n-j-jb, one, dA(j,j), ldda, dA(j,j+jb), ldda, queue);
    }
}
```

Note:

- MAGMA and LAPACK look similar
- Difference is lines in red, specifying data transfers and dependencies
- Differences can be hidden in a dynamic scheduler making the top level representation of MAGMA MIC algorithms almost identical to LAPACK

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

A Hybrid Algorithm Example

Left-looking hybrid Cholesky

From sequential LAPACK

```
for( j=0, j<n; j+=nb) {
   jb = min(nb, n-j);
     zherk( MagmaUpper
            jb, j, one, dA(0

     if (j+jb < n)
       zgemm( MagmaCo
              dA(0,j), ldd

     zpotrf( MagmaUpper
     if (info != 0)
        *info += j;

     If (j+jb) < n) {

        ztrsm( MagmaLeft,
                   jb, n
     }
}
```

to parallel hybrid MAGMA

```
for( j=0, j<n; j+=nb) {
   jb = min(nb, n-j);
   magma_zherk( MagmaUpper, MagmaConjTra
                  jb, j, one, dA(0,j), ldda, one, dA
    magma_zgetmatrix_async( jb, jb, dA(j,j), ldd
    if (j+jb < n)
       magma_zgemm( MagmaConjTrans, Magma
                  dA(0,j), ldda, dA(0,j+jb), ldd
    magma_event_sync( event );
    zpotrf( MagmaUpperStr, &jb, work, &jb, info);
    if (info != 0)
       *info += j;
    magma_zsetmatrix_async(jb, jb, work, jb, dA
    If (j+jb) < n) {
       magma_event_sync( event );
       magma_ztrsm( MagmaLeft, MagmaUpper, 
                  jb, n-j-jb, one, dA(j,j), ldda, d
    }
}
```

MAGMA MIC runtime environment

- Scheduling can be static or dynamic
- Dynamic is based on QUARK
- Uses CUDA streams to offload computation to the GPU

Note:
- MAGMA and LAPACK look similar
- Difference is lines in red, specifying data transfers and dependencies
- Differences can be hidden in a dynamic scheduler making the top level representation of MAGMA MIC algorithms almost identical to LAPACK

ICL UT INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

A Hybrid Algorithm Example

Left-looking hybrid Cholesky

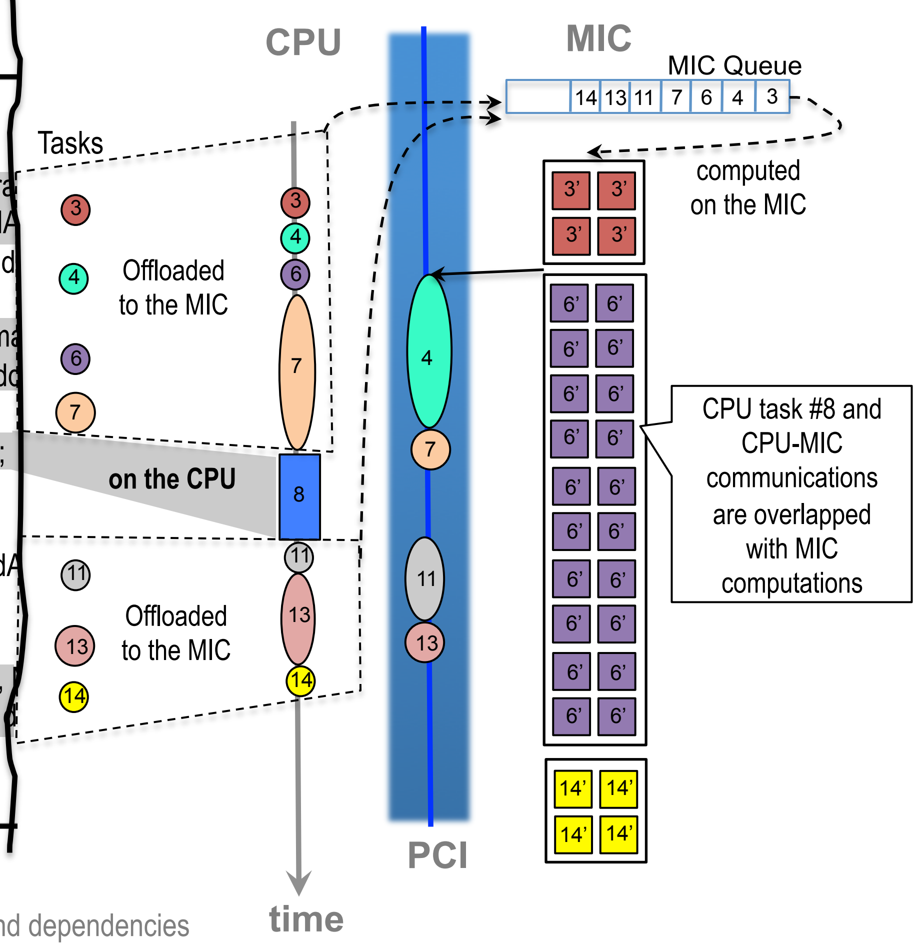
From sequential LAPACK

```
for( j=0, j<n; j+=nb) {
   jb = min(nb, n-j);
   zherk( MagmaUpper
          jb, j, one, dA(0

    if (j+jb < n)
      zgemm( MagmaCo
             dA(0,j), ldd

    zpotrf( MagmaUpper
    if (info != 0)
       *info += j;

    If (j+jb) < n) {

      ztrsm( MagmaLeft,
             jb, n
    }
}
```

to parallel hybrid **MAGMA**

```
1   for( j=0, j<n; j+=nb) {
2      jb = min(nb, n-j);
3      magma_zherk( MagmaUpper, MagmaConjTra
                    jb, j, one, dA(0,j), ldda, one, dA
4      magma_zgetmatrix_async( jb, jb, dA(j,j), ldd
5      if (j+jb < n)
6        magma_zgemm( MagmaConjTrans, Magma
                      dA(0,j), ldda, dA(0,j+jb), ldd
7      magma_event_sync( event );
8      zpotrf( MagmaUpperStr, &jb, work, &jb, info);
9      if (info != 0)
10        *info += j;
11     magma_zsetmatrix_async(jb, jb, work, jb, dA
12     If (j+jb) < n) {
13       magma_event_sync( event );
14       magma_ztrsm( MagmaLeft, MagmaUpper,
                      jb, n-j-jb, one, dA(j,j), ldda, d
       }
   }
```

MAGMA MIC runtime environment

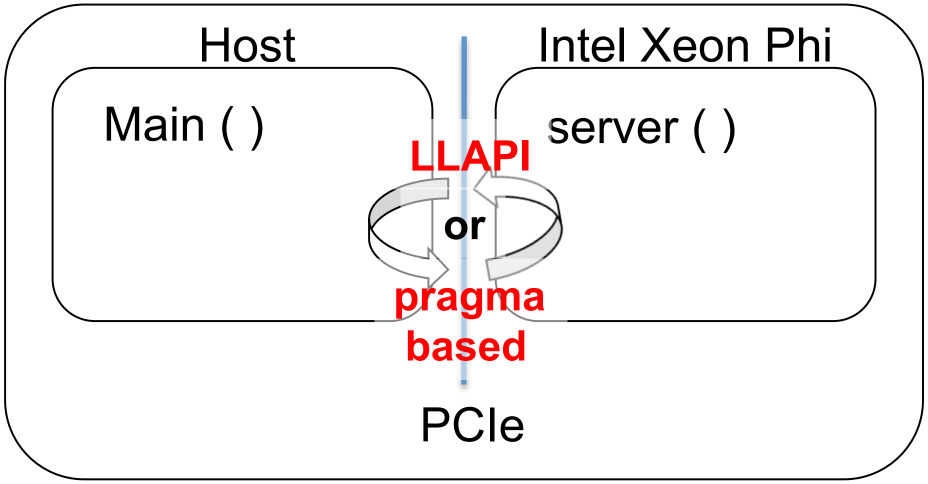


Note:
- MAGMA and LAPACK look similar
- Difference is lines in red, specifying data transfers and dependencies
- Differences can be hidden in a dynamic scheduler making the top level representation of MAGMA MIC algorithms almost identical to LAPACK

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Programming models

- **We developed two APIs for offloading work to MIC:**



Both APIs have the same interface and abstract low level programming details

LLAPI based

- A server runs on the MIC
- Communications are implemented through LLAPI using SCIF

Compiler pragma offload based

- API is using Phi-specific offload directives
- Enhancements for CPU-MIC communications

ICL UT INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Scheduling strategies

No need to explicitly code data dependencies and data transfers. This is hidden in the runtime system.

High-productivity with Dynamic Runtime Systems

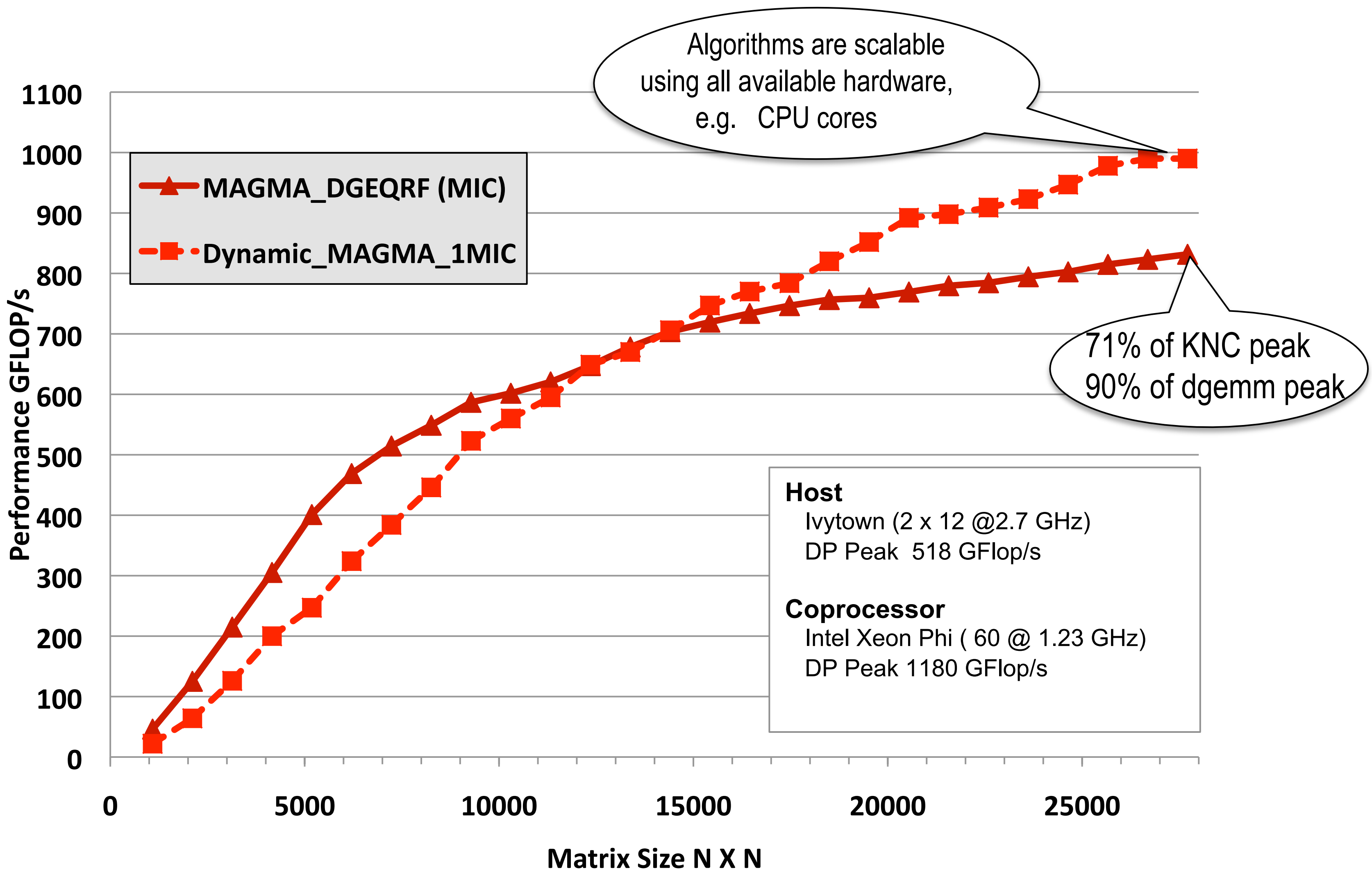
From sequential code → **Parallel execution**

```
for (k = 0; k < min(MT, NT); k++){
  zgeqrt(A[k;k], ...);
  for (n = k+1; n < NT; n++)
    zunmqr(A[k;k], A[k;n], ...);
  for (m = k+1; m < MT; m++){
    ztsqrt(A[k;k],,A[m;k], ...);
    for (n = k+1; n < NT; n++)
      ztsmqr(A[m;k], A[k;n], A[m;n], ...);
  }
}
```

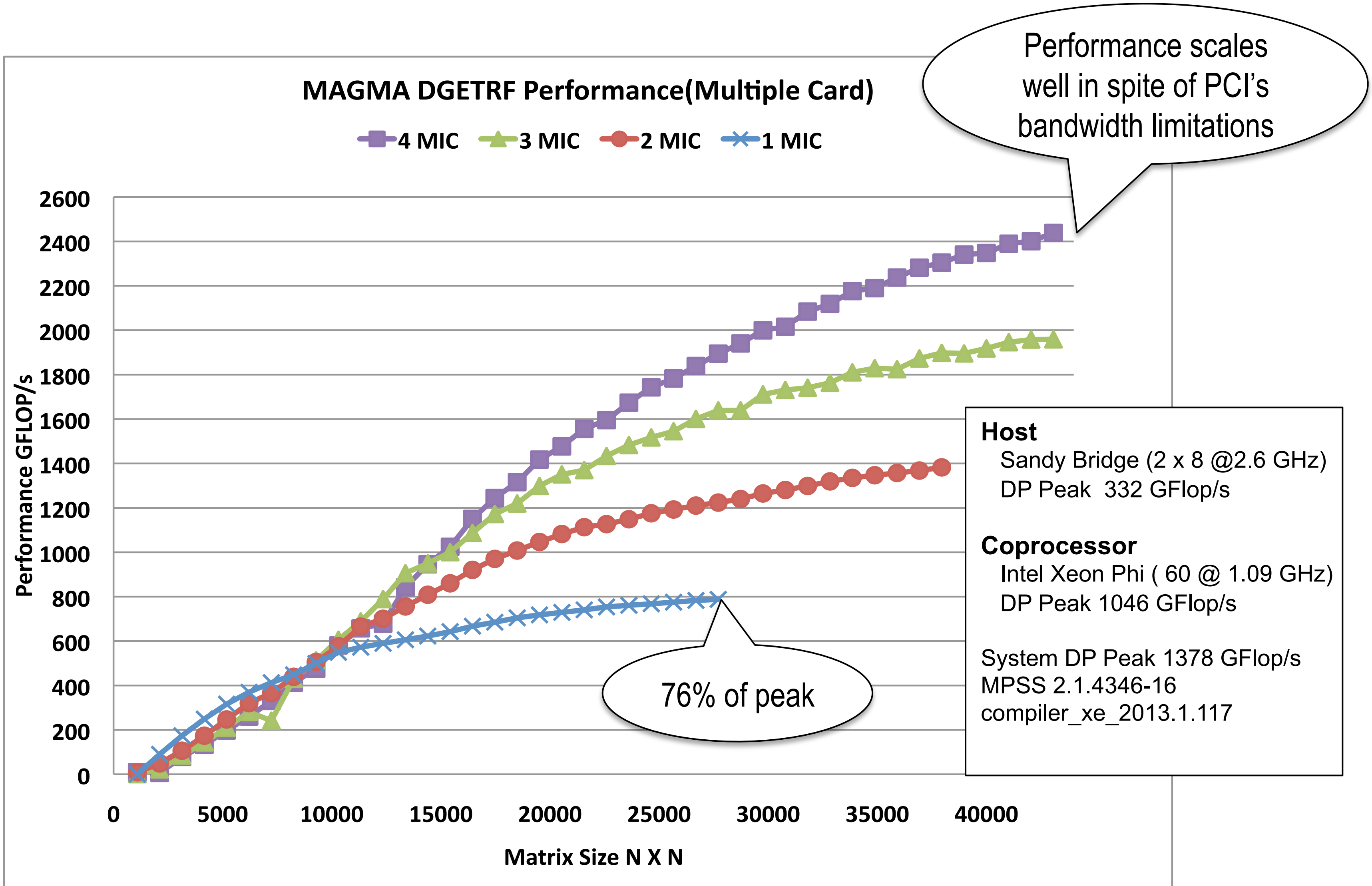
```
for (k = 0; k < min(MT, NT); k++){
  Insert_Task(&zgeqrt, k , k, ...);
  for (n = k+1; n < NT; n++)
   Insert_Task(&zunmqr, k, n, ...);
  for (m = k+1; m < MT; m++){
   Insert_Task(&ztsqrt, m, k, ...);
   for (n = k+1; n < NT; n++)
        Insert_Task(&ztsmqr, m, n, k, ...);
  }
}
```


ICL UT INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE Department of Electrical Engineering and Computer Science

Performance on single MIC
QR AO with static and dynamic MAGMA



Scalability on multiple MICs



Host
Sandy Bridge (2 x 8 @2.6 GHz)
DP Peak 332 GFlop/s

Coprocessor
Intel Xeon Phi (60 @ 1.09 GHz)
DP Peak 1046 GFlop/s

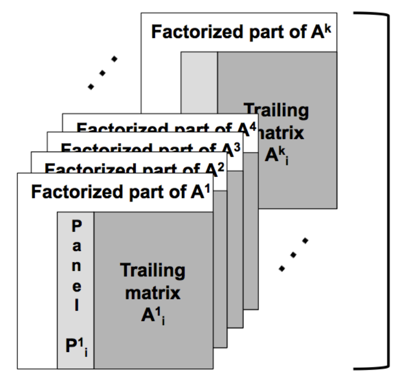
System DP Peak 1378 GFlop/s
MPSS 2.1.4346-16
compiler_xe_2013.1.117

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Plans & Goals: Dense Linear Algebra

- **Derive new methods and algorithmic improvements**
  - **Eigensolvers and SVD using two-stage reductions**
    [remove the memory-bound limitations of the LAPACK algorithms, and depending on hardware show an order of magnitude improvement]
  - **Factorizations and solvers for symmetric indefinite problems**

- **Develop linear algebra on small matrices**
  - **Batched linear algebra operations to provide support for various applications**
  - **Batched LU, QR, and Cholesky**
    [for the simultaneous factorization of many very small dense matrices]



Plans & Goals: Sparse Linear Algebra (SLA)

- **While extremely important for applications, SLA is notorious for running only at a fraction of the peak of modern architectures.**
- **Develop a highly optimized MAGMA MIC Sparse package**

  [include the standard CG, BiCGSTAB, GMRES, and preconditioned versions]
- **Incorporate communication-avoiding algorithms to significantly exceed in performance the standard memory and latency bound algorithms.**

  [include s-step methods, CA-GMRES, and blocked eigensolvers, e.g., LOBPCG]

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

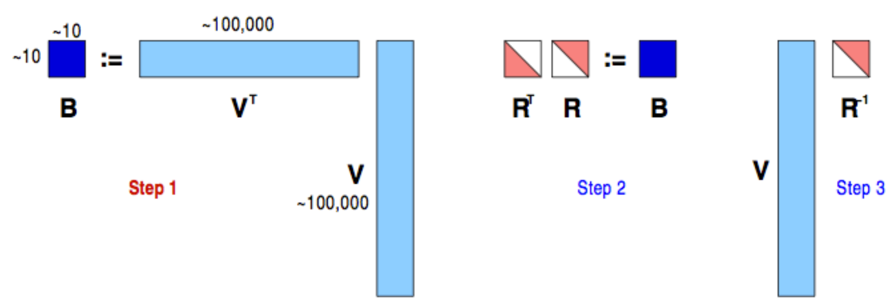
Plans & Goals: Mixed-Precision Methods

- **Develop numerical algorithms that recognize and exploit the presence of mixed-precision mathematics:**
  - **Show 2x acceleration using mixed-precision iterative refinement solvers for dense problems;**
  - **Mixed-precision orthogonalization schemes to accelerate applications, sparse iterative linear system and eigenproblem solvers:**

Step 1 Gram-matrix formation $B := V^T V$
on **MICs** in **extended precision**

Step 2 Cholesky factorization $R^T R := B$
on **CPUs** in **extended precision**

Step 3 Backward-substitution $Q := VR^{-1}$
on **MICs** in standard-precision.



ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Plans & Goals: Benchmarks

- **Develop a set of benchmarks for both performance and energy consumption. Include the**
  - **Newly proposed HPCG, optimized for Intel Xeon Phi architectures**
  - **Benchmarks for main communication and computation patterns**
    [e.g., CPU-MIC communication, MIC copy, MIC broadcast, latencies, representative BLAS 1/2/3, SpMV, SpMV, LU, SVD, etc.]
- **Show essential communication and computation patterns in various applications**
- **Goal is to encourage the focus of both hardware and software developers on architecture features and application needs; incorporate in performance analysis tools**

ICL INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Collaborators and Support

MAGMA team
http://icl.cs.utk.edu/magma

PLASMA team
http://icl.cs.utk.edu/plasma

Intel MKL team

Collaborating partners
University of Tennessee, Knoxville
University of California, Berkeley
University of Colorado, Denver
INRIA, France (StarPU team)
KAUST, Saudi Arabia