Architecture-aware Algorithms and Software for Peta and Exascale Computing

Jack Dongarra

University of Tennessee
Oak Ridge National Laboratory
University of Manchester
Outline

• Overview of High Performance Computing

• Look at an implementation for some linear algebra algorithms on today’s High Performance Computers
  ▪ As an examples of the kind of thing needed.
State of Supercomputing in 2016

• Pflops (> $10^{15}$ Flop/s) computing fully established with 81 systems.

• Three technology architecture possibilities or “swim lanes” are thriving.
  • Commodity (e.g. Intel)
  • Commodity + accelerator (e.g. GPUs) (104 systems)
  • Special purpose lightweight cores (e.g. IBM BG, ARM, Intel’s Knights Landing)

• Interest in supercomputing is now worldwide, and growing in many new markets (around 50% of Top500 computers are used in industry).

• Exascale ($10^{18}$ Flop/s) projects exist in many countries and regions.

• Intel processors have largest share, 89% followed by AMD, 4%. 
- Listing of the 500 most powerful Computers in the World
- Yardstick: Rmax from LINPACK MPP
  \[ Ax = b, \text{ dense problem} \]
- Updated twice a year
  SC‘xy in the States in November
  Meeting in Germany in June
- All data available from www.top500.org
Performance Development of HPC over the Last 24 Years from the Top500

Performance improvements over time, with a focus on the top gains in processing power from 1994 to 2015. The graph illustrates the growth in computational speed from Gflops to EGFlops, with key milestones highlighting advancements in technology:

- 1994: 59.7 GFlops
- 2000: 1.17 TFlops
- 2002: 597 GFlops
- 2006: 420 PFlops
- 2010: 33.9 PFlops
- 2012: 206 TFlops
- 2014: 420 PFlops

Key technological advances are marked, including the performance of a laptop (70 Gflop/s) and an iPhone (4 Gflop/s), showing the relative gains in computing power over time. The N=500 line highlights the cumulative performance gain from 1994 to 2015, while the N=1 line shows the baseline performance for comparison.
# November 2015: The TOP 10 Systems

<table>
<thead>
<tr>
<th>Rank</th>
<th>Site</th>
<th>Computer</th>
<th>Country</th>
<th>Cores</th>
<th>Rmax [Pflops]</th>
<th>% of Peak</th>
<th>Power [MW]</th>
<th>MFlops/Watt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>National Super Computer Center in Guangzhou</td>
<td>Tianhe-2 NUDT, Xeon 12C + IntelXeon Phi (57c) + Custom</td>
<td>China</td>
<td>3,120,000</td>
<td>33.9</td>
<td>62</td>
<td>17.8</td>
<td>1905</td>
</tr>
<tr>
<td>2</td>
<td>DOE / OS Oak Ridge Nat Lab</td>
<td>Titan, Cray XK7, AMD (16C) + Nvidia Kepler GPU (14c) + Custom</td>
<td>USA</td>
<td>560,640</td>
<td>17.6</td>
<td>65</td>
<td>8.3</td>
<td>2120</td>
</tr>
<tr>
<td>3</td>
<td>DOE / NNSA L Livermore Nat Lab</td>
<td>Sequoia, BlueGene/Q (16c) + Custom</td>
<td>USA</td>
<td>1,572,864</td>
<td>17.2</td>
<td>85</td>
<td>7.9</td>
<td>2063</td>
</tr>
<tr>
<td>4</td>
<td>RIKEN Advanced Inst for Comp Sci</td>
<td>K computer Fujitsu SPARC64 VIIIfx (8c) + Custom</td>
<td>Japan</td>
<td>705,024</td>
<td>10.5</td>
<td>93</td>
<td>12.7</td>
<td>827</td>
</tr>
<tr>
<td>5</td>
<td>DOE / OS Argonne Nat Lab</td>
<td>Mira, BlueGene/Q (16c) + Custom</td>
<td>USA</td>
<td>786,432</td>
<td>8.16</td>
<td>85</td>
<td>3.95</td>
<td>2066</td>
</tr>
<tr>
<td>6</td>
<td>DOE / NNSA / Los Alamos &amp; Sandia</td>
<td>Trinity, Cray XC40, Xeon 16C + Custom</td>
<td>USA</td>
<td>301,056</td>
<td>8.10</td>
<td>80</td>
<td>3.95</td>
<td>2066</td>
</tr>
<tr>
<td>7</td>
<td>Swiss CSCS</td>
<td>Piz Daint, Cray XC30, Xeon 8C + Nvidia Kepler (14c) + Custom</td>
<td>Swiss</td>
<td>115,984</td>
<td>6.27</td>
<td>81</td>
<td>2.3</td>
<td>2726</td>
</tr>
<tr>
<td>8</td>
<td>HLRS Stuttgart</td>
<td>Hazel Hen, Cray XC40, Xeon 12C + Custom</td>
<td>Germany</td>
<td>185,088</td>
<td>5.64</td>
<td>76</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>KAUST</td>
<td>Shaheen II, Cray XC40, Xeon 16C + Custom</td>
<td>Saudi Arabia</td>
<td>196,608</td>
<td>5.54</td>
<td>77</td>
<td>2.8</td>
<td>1954</td>
</tr>
<tr>
<td>10</td>
<td>Texas Advanced Computing Center</td>
<td>Stampede, Dell Intel (8c) + Intel Xeon Phi (61c) + IB</td>
<td>USA</td>
<td>204,900</td>
<td>5.17</td>
<td>61</td>
<td>4.5</td>
<td>1489</td>
</tr>
</tbody>
</table>

500 (368) Regensburg Eurotech Intel Germany 15,872 .206 95
Commodity plus Accelerator
Today 104 of the Top500 Systems

<table>
<thead>
<tr>
<th>Commodity</th>
<th>Accelerator (GPU)</th>
<th>192 Cuda cores/SMX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Xeon</td>
<td>Nvidia K20X “Kepler”</td>
<td>2688 “Cuda cores”</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gives 14 cores</td>
</tr>
</tbody>
</table>

**Commodity**
- Intel Xeon

**Accelerator (GPU)**
- Nvidia K20X “Kepler”

**Interconnect**
- PCI-e Gen2/3 16 lane
- 64 Gb/s (8 GB/s)
- 1 GW/s

**Processor**
- 8 cores
- 3 GHz
- 8*4 ops/cycle
- 96 Gflop/s (DP)

**GPU**
- 2688 “Cuda cores”
- 732 GHz
- 2688*2/3 ops/cycle
- 1.31 Tflop/s (DP)

**Video Memory**
- 6 GB
- 192 Cuda cores/SMX
- Gives 14 cores
Core Counts in the Top500 Systems

#1, Max, Mean, Min
Recent Developments

- US DOE planning to deploy three $O(100)$ Pflop/s systems for 2017-2018 - $525M$ hardware
- Oak Ridge Lab and Lawrence Livermore Lab to receive IBM and Nvidia based systems
- Argonne Lab to receive Intel based system
  - After this Exaflops

- US Dept of Commerce is preventing some China groups from receiving Intel
  - Citing concerns about nuclear
  - After this Exaflops

- For the first time, < 50% of Top500 are in the U.S.
  - 201 of the systems are U.S.-based, China #2 w/109.
Yutong Lu from NUDT at ISC Last Week

### Status of Tianhe System

<table>
<thead>
<tr>
<th>System</th>
<th>Tianhe-1A</th>
<th>Tianhe-2</th>
<th>Tianhe-2A</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Peak (PF)</td>
<td>4.7</td>
<td>54.9</td>
<td>~100</td>
</tr>
<tr>
<td>Peak Power (MW)</td>
<td>4.04</td>
<td>17.8</td>
<td>~18</td>
</tr>
<tr>
<td>Total System Memory</td>
<td>262 TB</td>
<td>1.4 PB</td>
<td>~3PB</td>
</tr>
<tr>
<td>Node Performance (TF)</td>
<td>0.655</td>
<td>3.431</td>
<td>~6</td>
</tr>
<tr>
<td>Node processors</td>
<td>Xeon X5670, Nvidia M2050</td>
<td>Xeon E5 2692, Xeon Phi</td>
<td>Xeon E5 2692, China Accelerator</td>
</tr>
<tr>
<td>System size (nodes)</td>
<td>7,168 nodes</td>
<td>16,000 nodes</td>
<td>~18,000</td>
</tr>
<tr>
<td>System Interconnect</td>
<td>TH Express-1</td>
<td>TH Express-2</td>
<td>H2FS+Lustre</td>
</tr>
<tr>
<td>File System</td>
<td>2 PB Lustre</td>
<td>12.4PB H2FS+Lustre</td>
<td>~30PB H2FS+TDM</td>
</tr>
</tbody>
</table>
China Accelerator

Matrix2000 GPDSP

- **High Performance**
  - 64bit Supported
  - ~2.4/4.8TFlops(DP/SP)
  - 1GHz, ~200W

- **High Throughput**
  - High-bandwidth Memory
  - 32~64GB
  - PCIe 3.0, 16x

[Diagram of Matrix2000 GPDSP with labeled components]
Countries Share

<table>
<thead>
<tr>
<th>Country</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>US</td>
<td>201</td>
</tr>
<tr>
<td>China</td>
<td>109</td>
</tr>
<tr>
<td>Japan</td>
<td>38</td>
</tr>
<tr>
<td>Germany</td>
<td>32</td>
</tr>
<tr>
<td>UK</td>
<td>18</td>
</tr>
<tr>
<td>France</td>
<td>18</td>
</tr>
</tbody>
</table>

China nearly tripled the number of systems on the latest list, while the number of systems in the US has fallen to the lowest point since the TOP500 list was created.

In Italy:
2 - Exploration & Production - Eni S.p.A.
2 - CINECA
Technology Trends:
Microprocessor Capacity

Number of devices/chip doubles every 18 months

2X transistors/Chip Every 1.5 years
Called “Moore’s Law”

The future of integrated electronics is the future of electronics itself. The advantages of integration will bring about a proliferation of electronics, pushing this science into many new areas.

Integrated circuits will lead to such wonders as home computers— or at least terminals connected to a central computer—automatic controls for automobiles, and personal portable communications equipment. The electronic wristwatch need only a display to be feasible today.

But the biggest potential lies in the production of large systems. In telephone communications, integrated circuits in digital filters will separate channels on multiplex equipment. Integrated circuits will also switch telephone circuits and perform data processing.

Computers will become powerful, and will be organized in completely different ways. For example, memories built of integrated electronics may be distributed throughout the machine instead of being concentrated in a central unit. In addition, the improved reliability made possible by integrated circuits will allow the construction of larger processing units. Machines similar to those in existence today will be built at lower costs and with faster turn-around.

Present and future

By integrated electronics, I mean all the various technologies which are referred to as microelectronics today as well as any additional ones that result in electronics functions supplied to the user as irreplaceable units. These technologies were first investigated in the late 1940’s. The object was to miniaturize electronics equipment to include increasingly complex electronic functions in limited space with minimum weight. Several approaches evolved, including monolithic techniques for individual components, thin-film structures and semiconductor integrated circuits.

Each approach evolved rapidly and converged so that each borrowed techniques from another. Many researchers believe the way of the future to be a combination of the various approaches.

The advocates of semiconductor integrated circuitry are already using the improved characteristics of thin-film resistors by applying such films directly to an active semiconductor substrate. Those advocating a technology based upon
Moore’s Secret Sauce: Dennard Scaling

Moore’s Law put lots more transistors on a chip… but it’s Dennard’s Law that made them useful.

Dennard Scaling:
- Decrease feature size by a factor of $\lambda$ and decrease voltage by a factor of $\lambda$; then
- # transistors increase by $\lambda^2$
- Clock speed increases by $\lambda$
- Energy consumption does not change

2x transistor count
40% faster
50% more efficient

Design of Ion-Implanted MOSFET’s with Very Small Physical Dimensions

ROBERT H. DENNARD, MEMBER, IEEE, FRITZ H. GAENSSLEN, HWA-NIEN YU, MEMBER, IEEE, V. LEO RIDEOUT, MEMBER, IEEE, ERNEST BASSOUS, AND ANDREI R. LEBLANC, MEMBER, IEEE

Abstract—This paper considers the design, fabrication, and characterization of ion-implant MOSFET’s having channel dimensions of the order of $\lambda$. Scaling relationships are presented which show how a conventional MOSFET can be reduced in size. An improved small device structure is proposed in which the implanted induced bandgap helps to provide shallow source and drain regions and a nullsubthreshold slope channel profile. One-dimensional models are used to predict the ultimate doping profile and the corresponding threshold voltage versus source voltage characteristic. A two-dimensional current transport model is used to predict current characteristics for different device geometry combinations. Polycrystalline MOSFET’s with channel lengths as short as 0.5 $\mu$m fabricated, and the device characteristics measured and compared with predicted values. The performance improvement expected from using these very small devices in highly miniaturized integrated circuits is projected.

Moore’s Secret Sauce
Inverse exponential slopes of subthreshold characteristic Width of idealized step function profile for channel implant Work function difference between gate and substrate Dilute dopant concentrations for silicon and silicon dioxide Drain current Bohman’s constant Unilith scaling constant MOSFET channel length Effective surface mobility Intrinsic carrier concentration Substrate acceptor concentration Bond breaking efficiency of silicon at the onset of strong inversion for zero substrate voltage

[Denard, Gaensslen, Yu, Rideout, Bassous, Leblanc, IEEE JSSC, 1974]
Unfortunately Dennard Scaling is Over: What is the Catch?

Powering the transistors without melting the chip

Breakdown is the result of small feature sizes, current leakage poses greater challenges, and also causes the chip to heat up

Power Cost of Frequency

- Power \( \propto \) Voltage\(^2\) x Frequency \((V^2F)\)
- Frequency \( \propto \) Voltage
- Power \( \propto \) Frequency\(^3\)

<table>
<thead>
<tr>
<th></th>
<th>Cores</th>
<th>V</th>
<th>Freq</th>
<th>Perf</th>
<th>Power</th>
<th>PE (Bops/watt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superscalar</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>“New” Superscalar</td>
<td>1X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>3.3X</td>
<td>0.45X</td>
</tr>
</tbody>
</table>
## Power Cost of Frequency

- **Power ∝ Voltage^2 x Frequency** \((V^2F)\)
- **Frequency ∝ Voltage**
- **Power ∝ Frequency^3**

<table>
<thead>
<tr>
<th></th>
<th>Cores</th>
<th>V</th>
<th>Freq</th>
<th>Perf</th>
<th>Power</th>
<th>PE (Bops/watt)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Superscalar</strong></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>“New” Superscalar</strong></td>
<td>1X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>1.5X</td>
<td>3.3X</td>
<td>0.45X</td>
</tr>
<tr>
<td><strong>Multicore</strong></td>
<td>2X</td>
<td>0.75X</td>
<td>0.75X</td>
<td>1.5X</td>
<td>0.8X</td>
<td>1.88X</td>
</tr>
</tbody>
</table>

(Bigger # is better)

50% more performance with 20% less power

Preferable to use multiple slower devices, than one superfast device
The primary reason cited for the breakdown is that at small sizes, current leakage poses greater challenges, and also causes the chip to heat up, which creates a threat of thermal runaway and therefore further increases energy costs.
## High Cost of Data Movement

<table>
<thead>
<tr>
<th>Operation</th>
<th>Energy consumed</th>
<th>Time needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>64-bit multiply-add</td>
<td>200 pJ</td>
<td>1 nsec</td>
</tr>
<tr>
<td>Read 64 bits from cache</td>
<td>800 pJ</td>
<td>3 nsec</td>
</tr>
<tr>
<td>Move 64 bits across chip</td>
<td>2000 pJ</td>
<td>5 nsec</td>
</tr>
<tr>
<td>Execute an instruction</td>
<td>7500 pJ</td>
<td>1 nsec</td>
</tr>
<tr>
<td>Read 64 bits from DRAM</td>
<td>12000 pJ</td>
<td>70 nsec</td>
</tr>
</tbody>
</table>

Notice that 12000 pJ @ 3 GHz = 36 watts!

Algorithms & Software: minimize data movement; perform more work per unit data movement.
Peak Performance - Per Core

Floating point operations per cycle per core

+ Most of the recent computers have FMA (Fused multiple add): (i.e. \(x \leftarrow x + y*z\) in one cycle)
+ Intel Xeon earlier models and AMD Opteron have SSE2
  + 2 flops/cycle DP & 4 flops/cycle SP
+ Intel Xeon Nehalem (’09) & Westmere (’10) have SSE4
  + 4 flops/cycle DP & 8 flops/cycle SP
+ Intel Xeon Sandy Bridge (’11) & Ivy Bridge (’12) have AVX
  + 8 flops/cycle DP & 16 flops/cycle SP
+ Intel Xeon Haswell (’13) & (Broadwell (’14)) AVX2
  + 16 flops/cycle DP & 32 flops/cycle SP
  + Xeon Phi (per core) is at 16 flops/cycle DP & 32 flops/cycle SP
+ Intel Xeon Skylake (server) (’15) AVX 512
  + 32 flops/cycle DP & 64 flops/cycle SP
In 167 cycles can do 2672 DP Flops

- Main memory: 167 cycles
- L3 Cache Full Random access: 38 cycles
- L3 Cache In Page Random access: 18 cycles
- L3 Cache sequential access: 14 cycles
- L2 Cache Full Random access: 11 cycles
- L2 Cache In Page Random access: 11 cycles
- L2 Cache sequential access: 11 cycles
- L1 Cache In Full Random access: 4 cycles
- L1 Cache In Page Random access: 4 cycles
- L1 Cache sequential access: 4 cycles
Level 1, 2 and 3 BLAS

1 core Intel Haswell i7-4850HQ, 2.3 GHz (Turbo Boost at 3.5 GHz);
Peak = 56 Gflop/s

- dgemm Level-3 BLAS
- dgemv Level-2 BLAS
- daxpy Level-1 BLAS

C = C + A * B

Performance GFLOP/s

y = y + A * x

y = a*x + y

1 core Intel Haswell i7-4850HQ, 2.3 GHz, Memory: DDR3L-1600MHz
6 MB shared L3 cache, and each core has a private 256 KB L2 and 64 KB L1.
The theoretical peak per core double precision is 56 Gflop/s per core.
Compiled with gcc and using Veclib
Main points

- Factorization column (zero) mostly sequential due to memory bottleneck
- Level 1 BLAS
- Divide pivot row has little parallelism
- Rank -1 Schur complement update is the only easy parallelize task
- Partial pivoting complicates things even further
- Bulk synchronous parallelism (fork-join)
  - Load imbalance
  - Non-trivial Amdahl fraction in the panel
  - Potential workaround (look-ahead) has complicated implementation
The Standard LU Factorization LAPACK
1980’s HPC of the Day: Cache Based SMP

Main points
• Panel factorization mostly sequential due to memory bottleneck
• Triangular solve has little parallelism
• Schur complement update is the only easy parallelize task
• Partial pivoting complicates things even further
• Bulk synchronous parallelism (fork-join)
  • Load imbalance
  • Non-trivial Amdahl fraction in the panel
  • Potential workaround (look-ahead) has complicated implementation
## Last Generations of DLA Software

Software/Algorithms follow hardware evolution in time

<table>
<thead>
<tr>
<th>Software/Algorithms</th>
<th></th>
<th>Rely on</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINPACK (70's)</td>
<td></td>
<td>- Level-1 BLAS operations</td>
</tr>
<tr>
<td>(Vector operations)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAPACK (80's)</td>
<td></td>
<td>- Level-3 BLAS operations</td>
</tr>
<tr>
<td>(Blocking, cache friendly)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ScaLAPACK (90's)</td>
<td></td>
<td>- PBLAS Mess</td>
</tr>
<tr>
<td>(Distributed Memory)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 2D Block Cyclic Layout

<table>
<thead>
<tr>
<th>Matrix point of view</th>
<th>Processor point of view</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 2 4 0 2 4 0 2 4</td>
<td>0 0 0 2 2 2 4 4 4</td>
</tr>
<tr>
<td>1 3 5 1 3 5 1 3 5</td>
<td>0 0 0 2 2 2 4 4 4</td>
</tr>
<tr>
<td>0 2 4 0 2 4 0 2 4</td>
<td>0 0 0 2 2 2 4 4 4</td>
</tr>
<tr>
<td>1 3 5 1 3 5 1 3 5</td>
<td>0 0 0 2 2 2 4 4 4</td>
</tr>
<tr>
<td>0 2 4 0 2 4 0 2 4</td>
<td>0 0 0 2 2 2 4 4 4</td>
</tr>
<tr>
<td>1 3 5 1 3 5 1 3 5</td>
<td>0 0 0 2 2 2 4 4 4</td>
</tr>
<tr>
<td>0 2 4 0 2 4 0 2 4</td>
<td>1 1 1 3 3 3 5 5 5</td>
</tr>
<tr>
<td>1 3 5 1 3 5 1 3 5</td>
<td>1 1 1 3 3 3 5 5 5</td>
</tr>
<tr>
<td>0 2 4 0 2 4 0 2 4</td>
<td>1 1 1 3 3 3 5 5 5</td>
</tr>
</tbody>
</table>

- **LINPACK (70's)**: Vector operations
- **LAPACK (80's)**: Blocking, cache friendly
- **ScaLAPACK (90's)**: Distributed Memory
Parallelization of LU and QR.

Parallelize the update:
- Easy and done in any reasonable software.
- This is the $2/3n^3$ term in the FLOPs count.
- Can be done efficiently with LAPACK+multithreaded BLAS
Synchronization (in LAPACK LU)

- Fork-join
- Bulk synchronous processing
Numerical program generates tasks and run time system executes tasks respecting data dependences.

PLASMA LU Factorization

Dataflow Driven

Sparse / Dense Matrix System

\[
\begin{pmatrix}
A_{11} & A_{12} & A_{13} & A_{14} \\
A_{21} & A_{22} & A_{23} & A_{24} \\
A_{31} & A_{32} & A_{33} & A_{34} \\
A_{41} & A_{42} & A_{43} & A_{44}
\end{pmatrix}
\]

DAG-based factorization

Batched LA

- LU, QR, or Cholesky on small diagonal matrices
- TRSMs, QRs, or LUs
- TRSMs, TRMMs
- Updates (Schur complement) GEMMs, SYRKs, TRMMs

And many other BLAS/LAPACK, e.g., for application specific solvers, preconditioners, and matrices
OpenMP Tasking

- Added with OpenMP 3.0 (2009)
- Allows parallelization of irregular problems
- OpenMP 4.0 (2013) - Tasks can have dependencies
  - DAGs
Tiled Cholesky Decomposition

```c
#pragma omp parallel
#pragma omp master
{   CHOLESKY( A ); }
CHOLESKY( A ) {
    for (k = 0; k < M; k++) {
        #pragma omp task depend(inout:A(k,k)[0:tilesize])
        {
            POTRF( A(k,k) );
        }
        for (m = k+1; m < M; m++) {
            #pragma omp task depend(in:A(k,k)[0:tilesize]) depend(inout:A(m,k)[0:tilesize])
            {
                TRSM( A(k,k), A(m,k) );
            }
        }
        for (m = k+1; m < M; m++) {
            #pragma omp task depend(in:A(m,k)[0:tilesize]) depend(inout:A(m,m)[0:tilesize])
            {
                SYRK( A(m,k), A(m,m) );
            }
        }
        for (n = k+1; n < m; n++) {
            #pragma omp task depend(in:A(m,k)[0:tilesize], A(n,k)[0:tilesize]) depend(inout:A(m,n)[0:tilesize])
            {
                GEMM( A(m,k), A(n,k), A(m,n) );
            }
        }
    }
}
```
Dataflow Based Design

Objectives
- High utilization of each core
- Scaling to large number of cores
- Synchronization reducing algorithms

Methodology
- Dynamic DAG scheduling
- Explicit parallelism
- Implicit communication
- Fine granularity / block data layout

Arbitrary DAG with dynamic scheduling

Fork-join parallelism
Notice the synchronization penalty in the presence of heterogeneity.
### Pipelining: Cholesky Inversion

**3 Steps:** Factor, Invert L, Multiply L’s

<table>
<thead>
<tr>
<th>POTRF</th>
<th>TRTRI</th>
<th>LAUUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>Cholesky Factorization alone: 3t-2</td>
<td></td>
</tr>
</tbody>
</table>

Pipelined: 18 (3t+6)

- **48 cores**
- **POTRF, TRTRI and LAUUM.**
- The matrix is 4000 x 4000, tile size is 200 x 200,

**POTRF+TRTRI+LAUUM:** 25 (7t-3)

Pipelined: 18 (3t+6)
### Other Systems

<table>
<thead>
<tr>
<th></th>
<th>PARSEC</th>
<th>SMPss</th>
<th>StarPU</th>
<th>Charm</th>
<th>FLAME</th>
<th>QUARK</th>
<th>Tblas</th>
<th>PTG</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language</strong></td>
<td>Internal or Seq. w/ Affine Loops</td>
<td>Seq. w/ add_task</td>
<td>Seq. w/ add_task</td>
<td>Msg-Driven Objects</td>
<td>Internal (LA DSL)</td>
<td>Seq. w/ add_task</td>
<td>Seq. w/ add_task</td>
<td>Internal</td>
</tr>
<tr>
<td><strong>Accelerator</strong></td>
<td>GPU</td>
<td>GPU</td>
<td>GPU</td>
<td>GPU</td>
<td>GPU</td>
<td>GPU</td>
<td>GPU</td>
<td>GPU</td>
</tr>
<tr>
<td><strong>Availability</strong></td>
<td>Public</td>
<td>Public</td>
<td>Public</td>
<td>Public</td>
<td>Public</td>
<td>Public</td>
<td>Public</td>
<td>Not Avail.</td>
</tr>
</tbody>
</table>

Early stage: ParalleX
Non-academic: Swarm, MadLINQ, CnC

All projects support Distributed and Shared Memory (QUARK with QUARKd; FLAME with Elemental)
Confessions of an Accidental Benchmarker

- Appendix B of the Linpack Users’ Guide
  - Designed to help users extrapolate execution Linpack software package
- First benchmark report from 1977;
  - Cray 1 to DEC PDP-10
Started 38 Years Ago
Have seen a Factor of $10^9$ - From 14 Mflop/s to 34 Pflop/s

- In the late 70’s the fastest computer ran LINPACK at 14 Mflop/s
- Today with HPL we are at 34 Pflop/s
- Nine orders of magnitude
doubling every 14 months
- About 6 orders of magnitude increase in the number of processors
- Plus algorithmic improvements

Began in late 70’s
time when floating point operations were expensive compared to other operations and data movement
TOP500

- In 1986 Hans Meuer started a list of supercomputer around the world, they were ranked by peak performance.
- Hans approached me in 1992 to put together our lists into the “TOP500”.
- The first TOP500 list was in June 1993.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Site</th>
<th>System</th>
<th>Cores</th>
<th>Rmax (GFlop/s)</th>
<th>Rpeak (GFlop/s)</th>
<th>Power (kW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Los Alamos National Laboratory United States</td>
<td>CM-5/1024</td>
<td>1,024</td>
<td>59.7</td>
<td>131.0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Minnesota Supercomputer Center United States</td>
<td>CM-5/544</td>
<td>544</td>
<td>30.4</td>
<td>69.6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>National Security Agency United States</td>
<td>CM-5/512</td>
<td>512</td>
<td>30.4</td>
<td>65.5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>NCSA United States</td>
<td>CM-5/512</td>
<td>512</td>
<td>30.4</td>
<td>65.5</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>NEC Japan</td>
<td>SX-3/44R</td>
<td>4</td>
<td>23.2</td>
<td>25.5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Atmospheric Environment Service (AES)</td>
<td>SX-3/44</td>
<td>4</td>
<td>20.0</td>
<td>22.0</td>
<td></td>
</tr>
</tbody>
</table>
HPL - Bad Things

- LINPACK Benchmark is 37 years old
  - TOP500 (HPL) is 24 years old
- Floating point-intensive performs $O(n^3)$ floating point operations and moves $O(n^2)$ data.
- No longer so strongly correlated to real apps.
- Reports Peak Flops (although hybrid systems see only 1/2 to 2/3 of Peak)
- Encourages poor choices in architectural features
- Overall usability of a system is not measured
- Used as a marketing tool
- Decisions on acquisition made on one number
- Benchmarking for days wastes a valuable resource
Proposal: HPCG

- High Performance Conjugate Gradient (HPCG).
- Solves $Ax=b$, $A$ large, sparse, $b$ known, $x$ computed.
- An optimized implementation of PCG contains essential computational and communication patterns that are prevalent in a variety of methods for discretization and numerical solution of PDEs

- Patterns:
  - Dense and sparse computations.
  - Dense and sparse collective.
  - Multi-scale execution of kernels via MG (truncated) V cycle.
  - Data-driven parallelism (unstructured sparse triangular solves).
- Strong verification and validation properties (via spectral properties of PCG).
Model Problem Description

- Synthetic discretized 3D PDE (FEM, FVM, FDM).
- Single heat diffusion model.
- Zero Dirichlet BCs, Synthetic RHS s.t. solution = 1.
- Local domain: $(n_x \times n_y \times n_z)$
- Process layout: $(np_x \times np_y \times np_z)$
- Global domain: $(n_x \times np_x) \times (n_y \times np_y) \times (n_z \times np_z)$
- Sparse matrix:
  - 27 nonzeros/row interior.
  - 7 – 18 on boundary.
  - Symmetric positive definite.
HPL vs. HPCG: Bookends

- Some see HPL and HPCG as “bookends” of a spectrum.
  - Applications teams know where their codes lie on the spectrum.
  - Can gauge performance on a system using both HPL and HPCG numbers.
- Problem of HPL execution time still an issue:
  - Need a lower cost option. End-to-end HPL runs are too expensive.
  - Work in progress.

  - Optimized versions for Intel and Nvidia
Comparison Peak, HPL
Comparison Peak, HPL, & HPCG
<table>
<thead>
<tr>
<th>Rank</th>
<th>Site</th>
<th>Computer</th>
<th>Cores</th>
<th>Rmax Pflops</th>
<th>HPCG Pflops</th>
<th>HPCG /HPL</th>
<th>% of Peak</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NSCC / Guangzhou</td>
<td>Tianhe-2 NUDT, Xeon 12C 2.2GHz + Intel Xeon Phi 57C + Custom</td>
<td>3,120,000</td>
<td>33.86</td>
<td>0.580</td>
<td>1.7%</td>
<td>1.1%</td>
</tr>
<tr>
<td>2</td>
<td>RIKEN Advanced Institute for Computational Science</td>
<td>K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect</td>
<td>705,024</td>
<td>10.51</td>
<td>0.460</td>
<td>4.4%</td>
<td>4.1%</td>
</tr>
<tr>
<td>3</td>
<td>DOE/SC/Oak Ridge Nat Lab</td>
<td>Titan - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x</td>
<td>560,640</td>
<td>17.59</td>
<td>0.322</td>
<td>1.8%</td>
<td>1.2%</td>
</tr>
<tr>
<td>4</td>
<td>DOE/NNSA/LANL/SLN</td>
<td>Trinity - Cray XC40, Intel E5-2698v3, Aries custom</td>
<td>301,056</td>
<td>8.10</td>
<td>0.182</td>
<td>2.3%</td>
<td>1.6%</td>
</tr>
<tr>
<td>5</td>
<td>DOE/SC/Argonne National Laboratory</td>
<td>Mira - BlueGene/Q, Power BQC 16C 1.60GHz, Custom</td>
<td>786,432</td>
<td>8.58</td>
<td>0.167</td>
<td>1.9%</td>
<td>1.7%</td>
</tr>
<tr>
<td>6</td>
<td>HLRS/University of Stuttgart</td>
<td>Hazel Hen - Cray XC40, Intel E5-2680v3, Infiniband FDR</td>
<td>185,088</td>
<td>5.64</td>
<td>0.138</td>
<td>2.4%</td>
<td>1.9%</td>
</tr>
<tr>
<td>7</td>
<td>NASA / Mountain View</td>
<td>Pleiades - SGI ICE X, Intel E5-2680, E5-2680V2, E5-2680V3, Infiniband FDR</td>
<td>186,288</td>
<td>4.08</td>
<td>0.131</td>
<td>3.2%</td>
<td>2.7%</td>
</tr>
<tr>
<td>8</td>
<td>Swiss National Supercomputing Centre (CSCS)</td>
<td>Piz Daint - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x</td>
<td>115,984</td>
<td>6.27</td>
<td>0.124</td>
<td>2.0%</td>
<td>1.6%</td>
</tr>
<tr>
<td>9</td>
<td>KAUST / Jeda</td>
<td>Shaheen II - Cray XC40, Intel Haswell 2.3 GHz 16C, Cray Aries</td>
<td>196,608</td>
<td>5.53</td>
<td>0.113</td>
<td>2.1%</td>
<td>1.6%</td>
</tr>
<tr>
<td>10</td>
<td>Texas Advanced Computing Center/Univ. of Texas</td>
<td>Stampede - PowerEdge C8220, Xeon E5-2680 8C 2.7GHz, Infiniband, Phi SE10P</td>
<td>522,080</td>
<td>5.16</td>
<td>0.096</td>
<td>1.9%</td>
<td>1.0%</td>
</tr>
<tr>
<td>Rank</td>
<td>Site</td>
<td>Computer</td>
<td>Cores</td>
<td>Rmax Pflops</td>
<td>HPCG Pflops</td>
<td>HPCG/HPL % of Peak</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>----------</td>
<td>-------</td>
<td>-------------</td>
<td>-------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Forschungszentrum Jülich</td>
<td>JUQUEEN - BlueGene/Q</td>
<td>458,752</td>
<td>5.0089</td>
<td>0.095</td>
<td>1.9% 1.6%</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Information Technology Center, Nagoya University</td>
<td>ITC, Nagoya - Fujitsu PRIMEHPC FX100</td>
<td>92,160</td>
<td>2.91</td>
<td>0.086</td>
<td>3.0% 2.7%</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Leibniz Rechenzentrum</td>
<td>SuperMUC - iDataPlex DX360M4, Xeon E5-2680 8C 2.70GHz, Infiniband FDR</td>
<td>147,456</td>
<td>2.897</td>
<td>0.083</td>
<td>2.9% 2.6%</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>EPSRC/University of Edinburgh</td>
<td>ARCHER - Cray XC30, Intel Xeon E5 v2 12C 2.700GHz, Aries interconnect</td>
<td>118,080</td>
<td>1.643</td>
<td>0.081</td>
<td>4.9% 3.2%</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>DOE/SC/LBNL/NERSC</td>
<td>Edison - Cray XC30, Intel Xeon E5-2695v2 12C 2.4GHz, Aries interconnect</td>
<td>133,824</td>
<td>1.655</td>
<td>0.079</td>
<td>4.8% 3.1%</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>National Institute for Fusion Science</td>
<td>Plasma Simulator - Fujitsu PRIMEHPC FX100, SPARC64 Xifx, Custom</td>
<td>82,944</td>
<td>2.376</td>
<td>0.073</td>
<td>3.1% 2.8%</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>GSIC Center, Tokyo Institute of Technology</td>
<td>TSUBAME 2.5 - Cluster Platform SL390s G7, Xeon X5670 6C 2.93GHz, Infiniband QDR, NVIDIA K20x</td>
<td>76,032</td>
<td>2.785</td>
<td>0.073</td>
<td>2.6% 1.3%</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>HLRS/Universitaet Stuttgart</td>
<td>Hornet - Cray XC40, Xeon E5-2680 v3 2.5 GHz, Cray Aries</td>
<td>94,656</td>
<td>2.763</td>
<td>0.066</td>
<td>2.4% 1.7%</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Max-Planck-Gesellschaft MPI/IPP</td>
<td>iDataPlex DX360M4, Intel Xeon E5-2680v2 10C 2.800GHz, Infiniband</td>
<td>65,320</td>
<td>1.283</td>
<td>0.061</td>
<td>4.8% 4.2%</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>CEIST / JAMSTEC</td>
<td>Earth Simulator - NEC SX-ACE</td>
<td>8,192</td>
<td>0.487</td>
<td>0.058</td>
<td>11.9% 11.0%</td>
<td></td>
</tr>
</tbody>
</table>
Critical Issues at Peta & Exascale for Algorithm and Software Design

- **Synchronization-reducing algorithms**
  - Break Fork-Join model

- **Communication-reducing algorithms**
  - Use methods which have lower bound on communication

- **Mixed precision methods**
  - 2x speed of ops and 2x speed for data movement

- **Autotuning**
  - Today’s machines are too complicated, build “smarts” into software to adapt to the hardware

- **Fault resilient algorithms**
  - Implement algorithms that can recover from failures/bit flips

- **Reproducibility of results**
  - Today we can’t guarantee this. We understand the issues, but some of our “colleagues” have a hard time with this.
Summary

• **Major Challenges are ahead for extreme computing**
  ▪ **Parallelism** $O(10^9)$
    • Programming issues
  ▪ **Hybrid**
    • Peak and HPL may be very misleading
    • No where near close to peak for most apps
  ▪ **Fault Tolerance**
    • Today Sequoia BG/Q node failure rate is 1.25 failures/day
  ▪ **Power**
    • 50 Gflops/w (today at 2 Gflops/w)

• **We will need completely new approaches and technologies to reach the Exascale level**
Collaborators / Software / Support

- **PLASMA**

- **MAGMA**
  - [http://icl.cs.utk.edu/magma/](http://icl.cs.utk.edu/magma/)

- **Quark (RT for Shared Memory)**

- **PaRSEC** *(Parallel Runtime Scheduling and Execution Control)*

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