USC Viterbi
School of Engineering

Accelerating the Multifrontal Method

Information Sciences Institute

22 June 2012
Bob Lucas, Gene Wagenbreth, Dan Davis, Roger Grimes
{rflucas,genew,ddavis}@isi.edu and grimes@lstc.com

USC Viterbi School of Engineering

3D Finite Element Test Problem

ISI Information Sciences Institute

Test Problem: cylinders cyl1e6

USC

USC Viterbi School of Engineering

LS-DYNA Implicit Half Million Elements

ISI Information Sciences Institute

T i m i n g i n f o r m a t i o n

	CPU(seconds)	%CPU	Clock(seconds)	%Clock
Initialization	2.6513E+01	0.24	2.7598E+01	1.53
Element processing ...	8.1698E+01	0.73	7.6144E+01	4.22
Binary databases	2.0471E+00	0.02	4.8657E-01	0.03
ASCII database	1.3164E-04	0.00	1.5000E-05	0.00
Contact algorithm	2.9563E+01	0.26	5.7157E+00	0.32
Interf. ID 1	7.6828E+00	0.07	1.2124E+00	0.07
Interf. ID 2	4.8080E+00	0.04	9.7007E-01	0.05
Interf. ID 3	4.4934E+00	0.04	9.8310E-01	0.05
Interf. ID 4	5.5574E+00	0.05	1.2091E+00	0.07
Interf. ID 5	7.0113E+00	0.06	1.3380E+00	0.07
Contact entities	0.0000E+00	0.00	0.0000E+00	0.00
Rigid bodies	8.5901E-04	0.00	3.1600E-04	0.00
Implicit Nonlinear ...	8.7434E+00	0.08	6.5756E+00	0.36
Implicit Lin. Alg. ...	1.1063E+04	98.67	1.6898E+03	93.55
T o t a l s	1.1212E+04	100.00	1.8063E+03	100.00

USC

Half million elements, eight Intel Nehalem cores

USC Viterbi School of Engineering

Major software components

ISI Information Sciences Institute

- All of LS-DYNA
  - Linear Solver
    - Reordering (Metis) & symbolic factorization
    - Redistribute & permute sparse matrix
    - Factor, i.e., for all supernodes:
      - Assemble the frontal matrix
      - Factor the frontal matrix
      - Stack its Schur complement
    - Solve

USC

USC Viterbi School of Engineering

Time in solver (sec)
Eight Nehalem Threads

ISI Information Sciences Institute

grep WCT mes0000 =>

```
WCT: symbolic factorization =    18.952
WCT: matrix redistribution  =     5.221
WCT: factorization          =   795.354
WCT: numeric solve          =     2.705
WCT: total imfslv_mf2       =   826.735
WCT: symbolic factorization =    16.560
WCT: matrix redistribution  =     5.087
WCT: factorization          =   804.471
WCT: numeric solve          =     2.819
WCT: total imfslv_mf2       =   829.281
```


USC

USC Viterbi School of Engineering

Start by accelerating factorization

ISI Information Sciences Institute

- All of LS-DYNA
  - Multifrontal Linear Solver
    - Reordering (Metis) & symbolic factorization
    - Redistribute & permute sparse matrix
    - Factor, i.e., for all supernodes
      - Assemble the frontal matrix
      - Factor the frontal matrix with accelerator
      - Stack its Schur complement
    - Solve

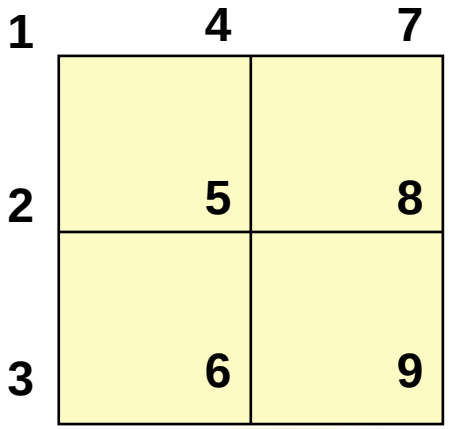
USC

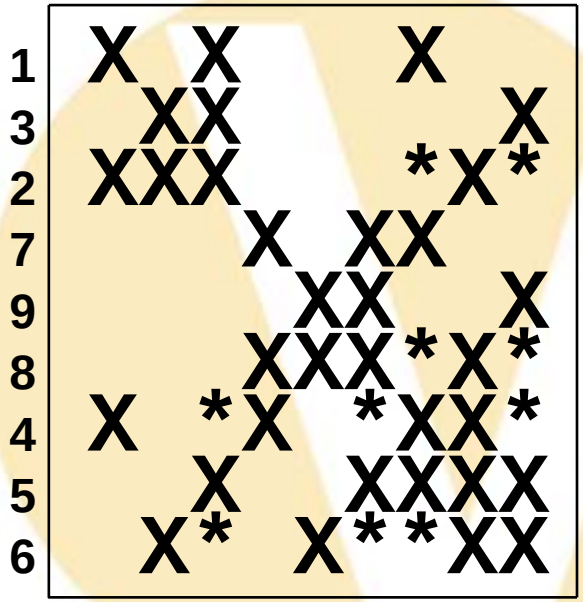
USC Viterbi School of Engineering

Multifontal Method Toy Problem

Information Sciences Institute

```
      do 4 k = 1, 9
        do 1 i = k + 1, 9
          a(i, k) = a(i,k) / a(k,k)
1       continue
        do 3 j = k + 1, 9
          do 2 i = k + 1, 9
            a(i,j) = a(i,j) -
     1                 a(i,k) *
     2                 a(k,j)
2         continue
3       continue
4     continue
```



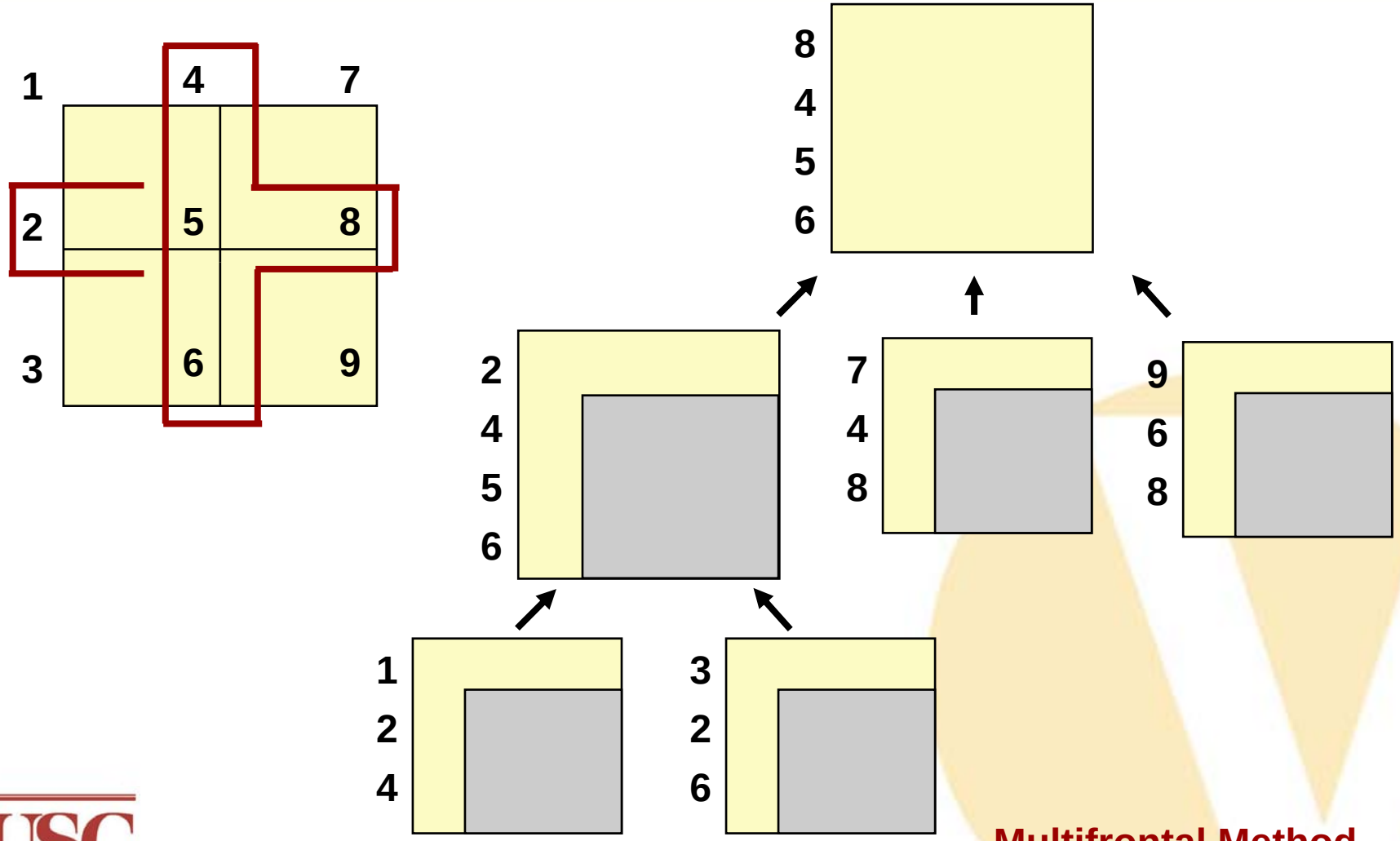


USC

USC Viterbi School of Engineering

Multifrontal View of a Toy Matrix

ISI Information Sciences Institute



USC

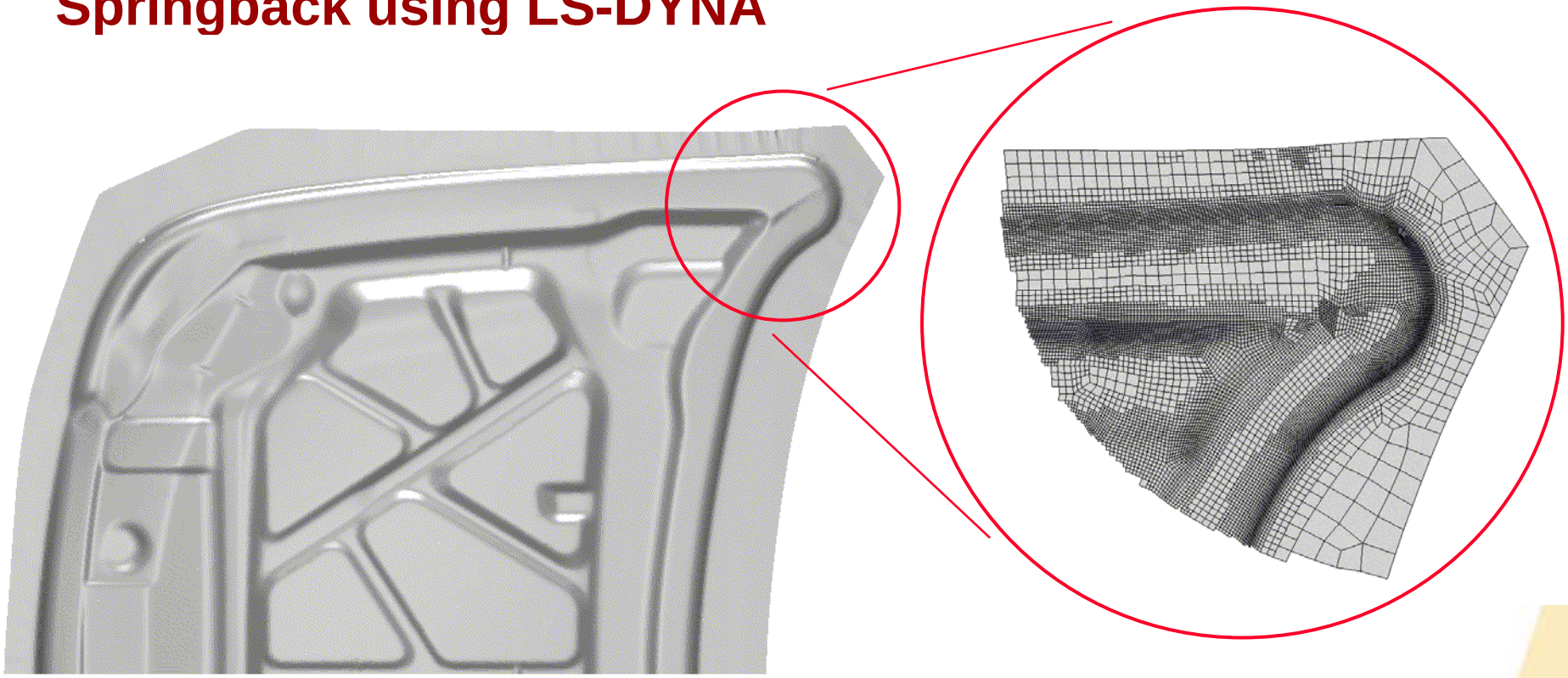
Multifrontal Method
Duff and Reid, ACM TOMS 1983

USC Viterbi School of Engineering

A Real Problem : “Hood”

ISI Information Sciences Institute

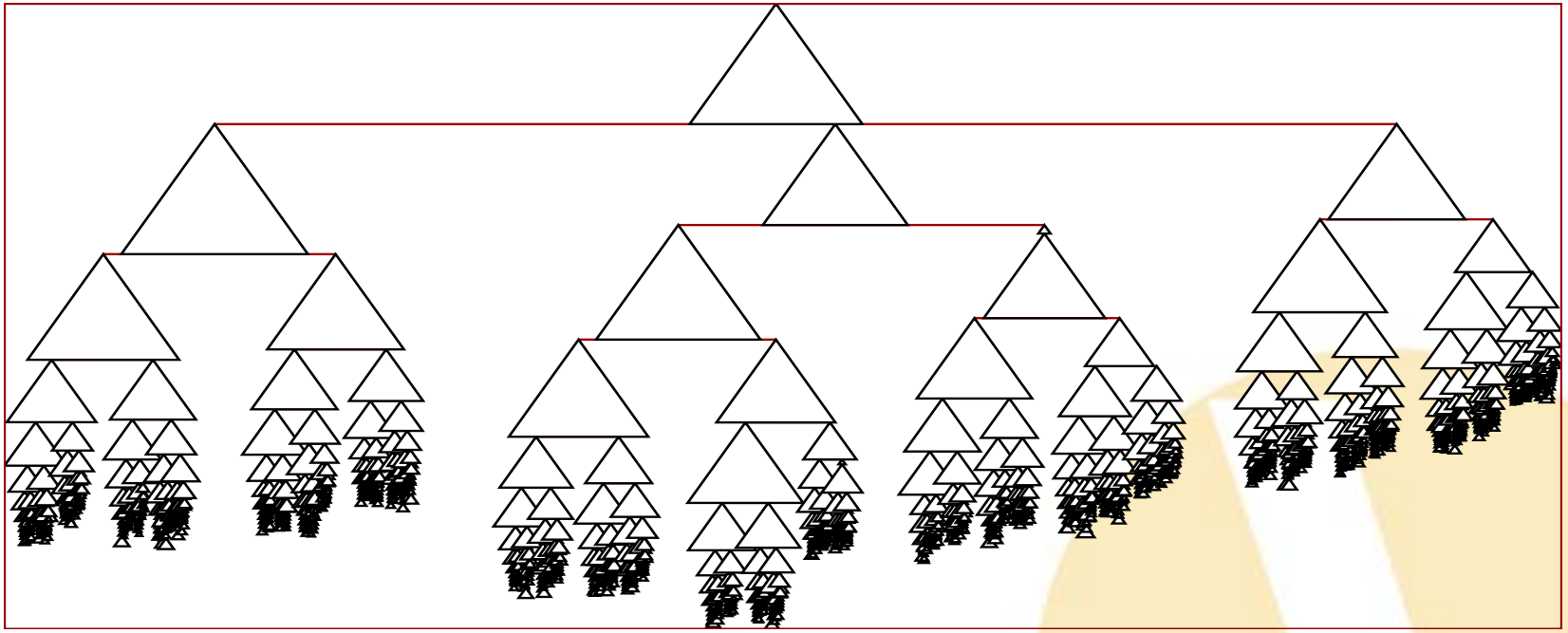
Automotive Hood Inner Panel Springback using LS-DYNA



USC Viterbi School of Engineering

“Hood” Elimination Tree

ISI Information Sciences Institute



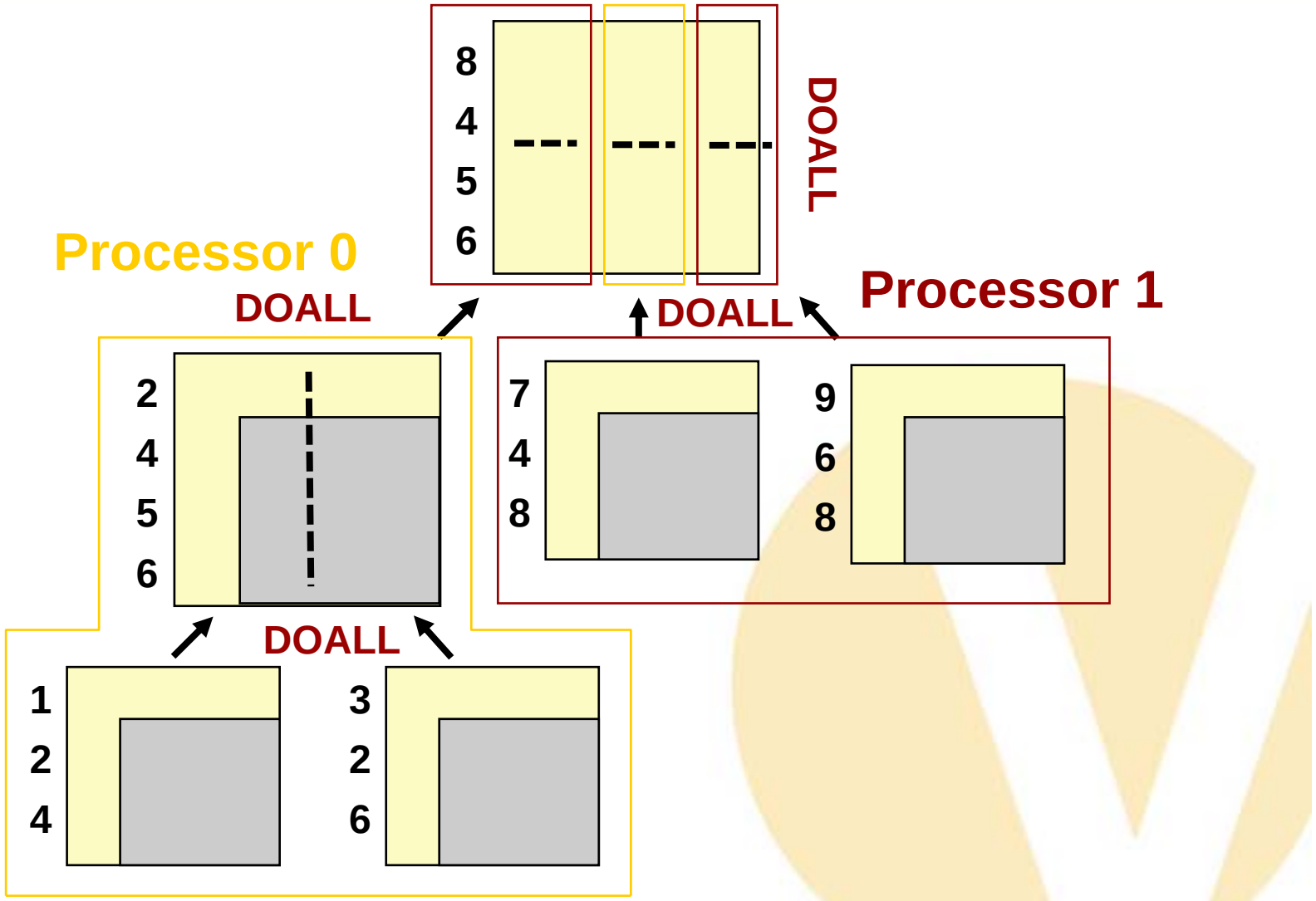
Each frontal matrix’s triangle scaled by operations required to factor it.

USC

USC Viterbi School of Engineering

Exploiting Concurrency Toy with MPI & OpenMP

ISI Information Sciences Institute

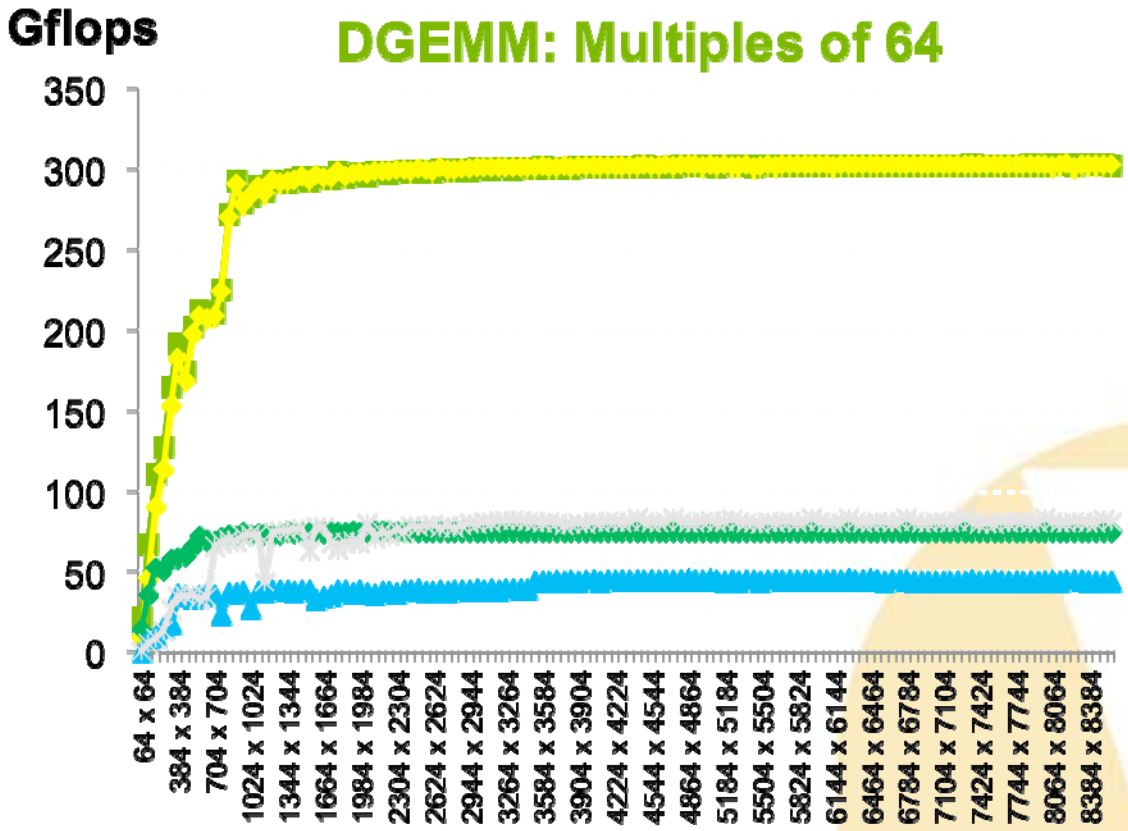


USC

USC Viterbi School of Engineering

Why Accelerators?

ISI Information Sciences Institute



USC

Tesla vs Nehalem, courtesy of Stan Posey
Similar expectations for Intel Phi

USC Viterbi School of Engineering

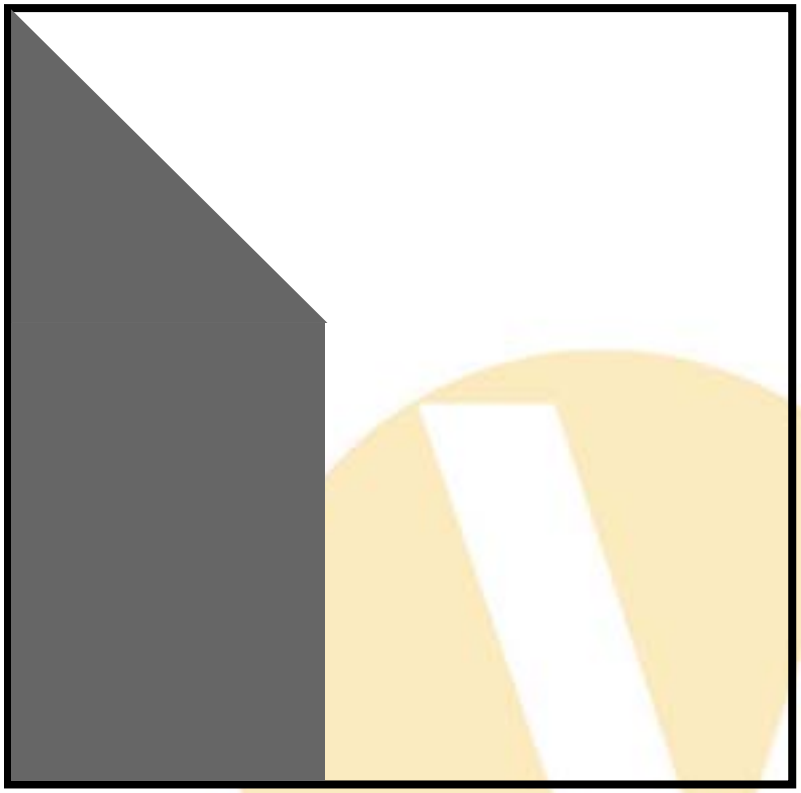
Left-Looking Accelerator Factorization of a Frontal Matrix

ISI Information Sciences Institute

Assemble frontal matrix on host CPU

Initialize by sending panel of assembled frontal matrix

Only large frontal matrices due to high cost of sending data to and from accelerator



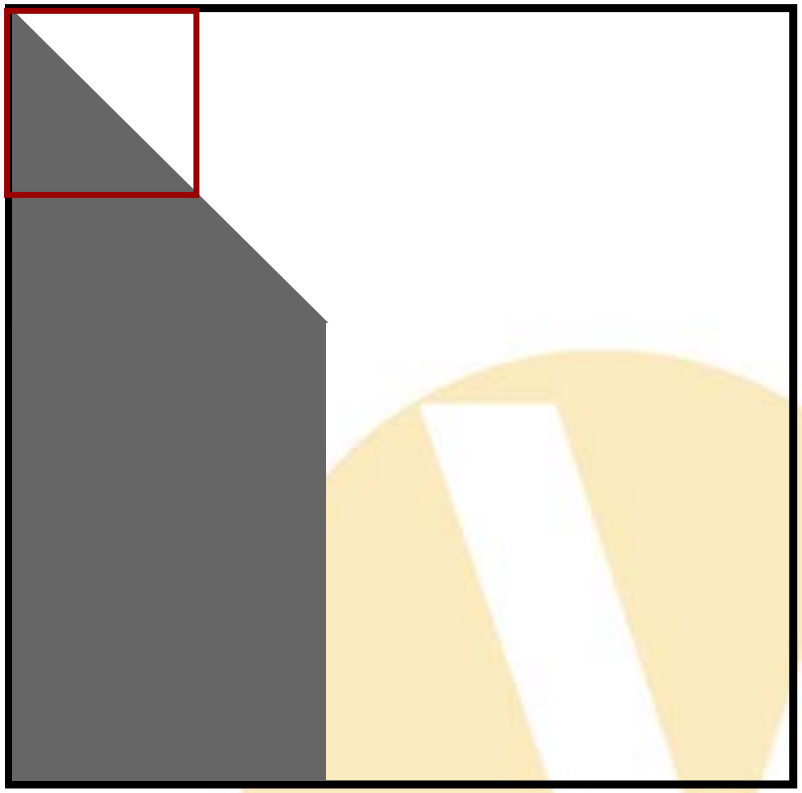
USC

USC Viterbi School of Engineering

Eliminate panels

ISI Information Sciences Institute

Factor diagonal block



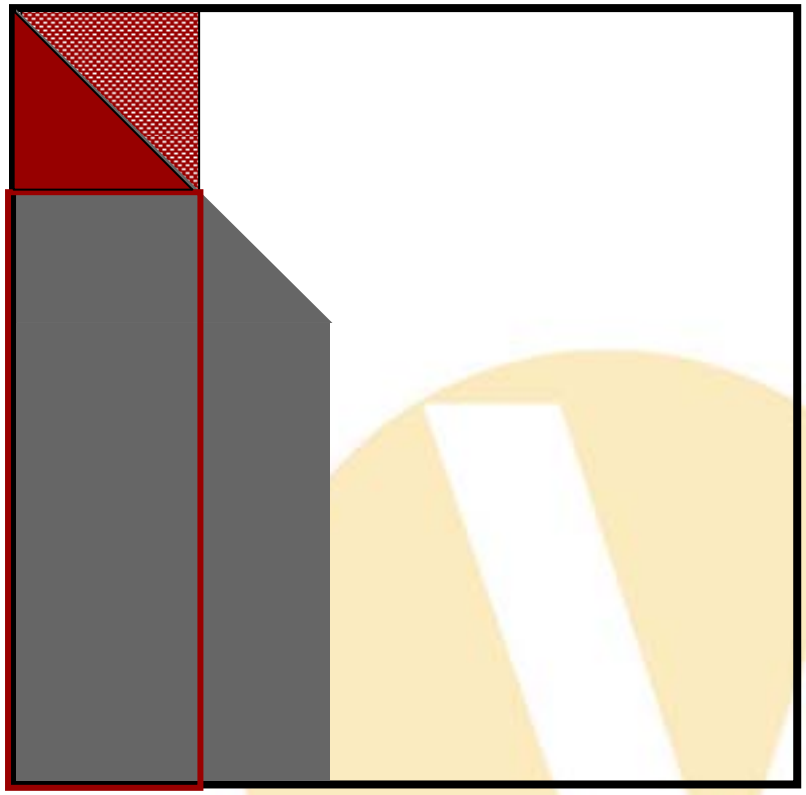
USC

USC Viterbi School of Engineering

Eliminate panels

Information Sciences Institute

Eliminate off-diagonal panel

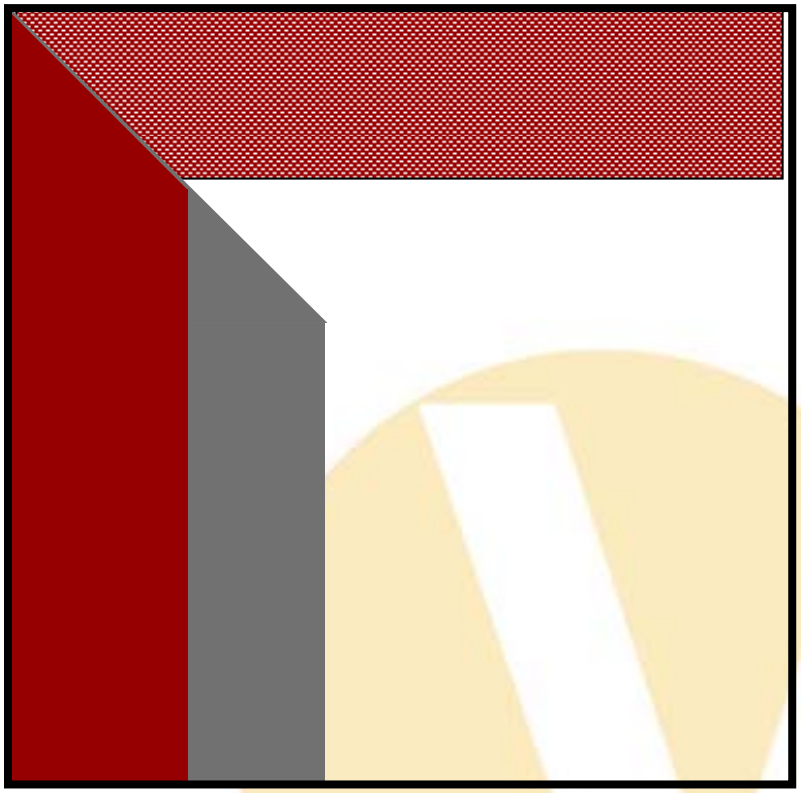


USC

USC Viterbi
School of Engineering

Fill Upper Triangle

ISI
Information Sciences Institute



USC

USC Viterbi School of Engineering

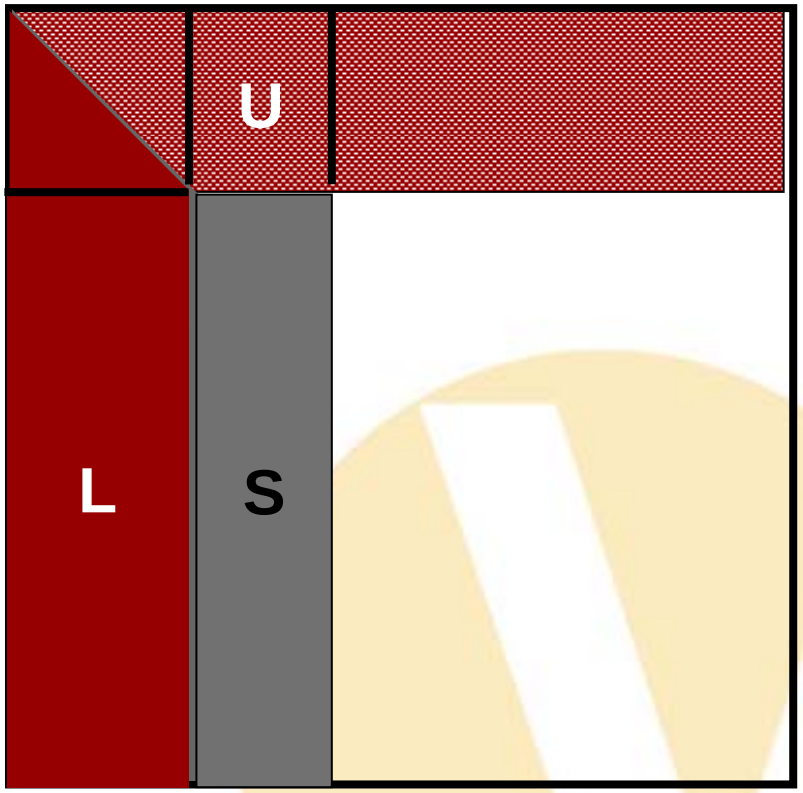
Update Schur Complement

ISI Information Sciences Institute

Update panels with DGEMM

$$S = S - L * U$$

Assumes vendor math library DGEMM is extremely fast



USC

USC Viterbi School of Engineering

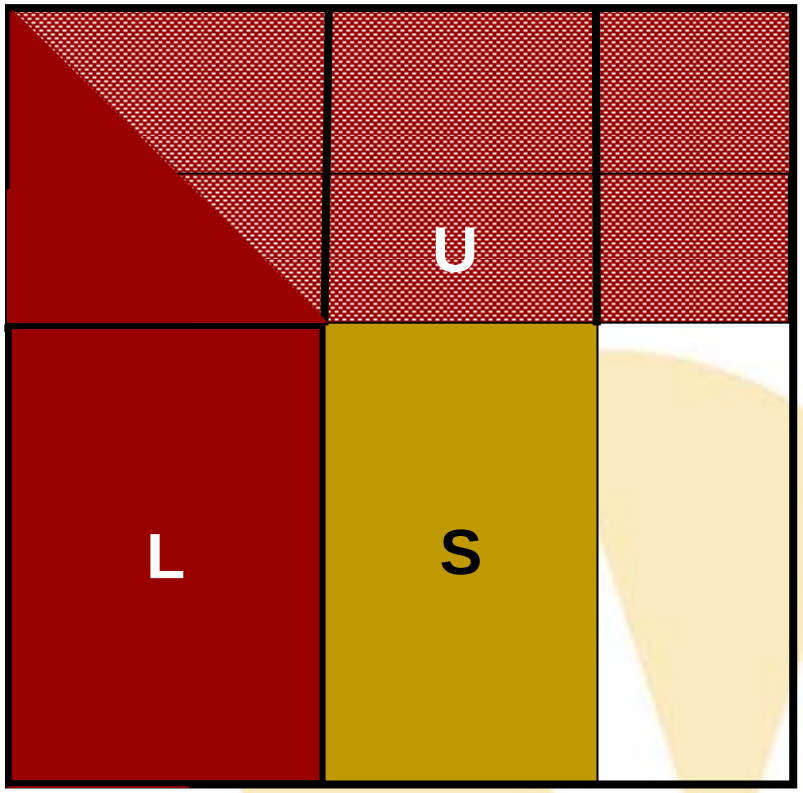
Update Schur Complement

ISI Information Sciences Institute

Wider panels in Schur complement

DGEMM is even faster

$$S = S - L * U$$



USC

USC Viterbi School of Engineering

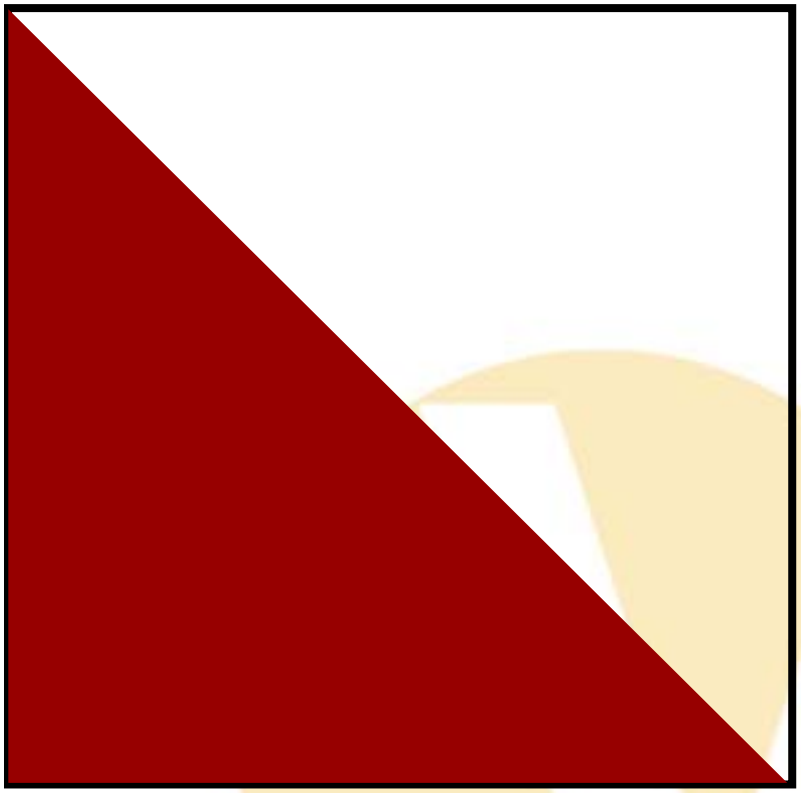
Return Entire Frontal Matrix

ISI Information Sciences Institute

Return error if diagonal of 0.0 encountered or pivot threshold exceeded

Otherwise complete frontal matrix is returned

Schur complement added to initial values on host CPU



USC

USC Viterbi School of Engineering

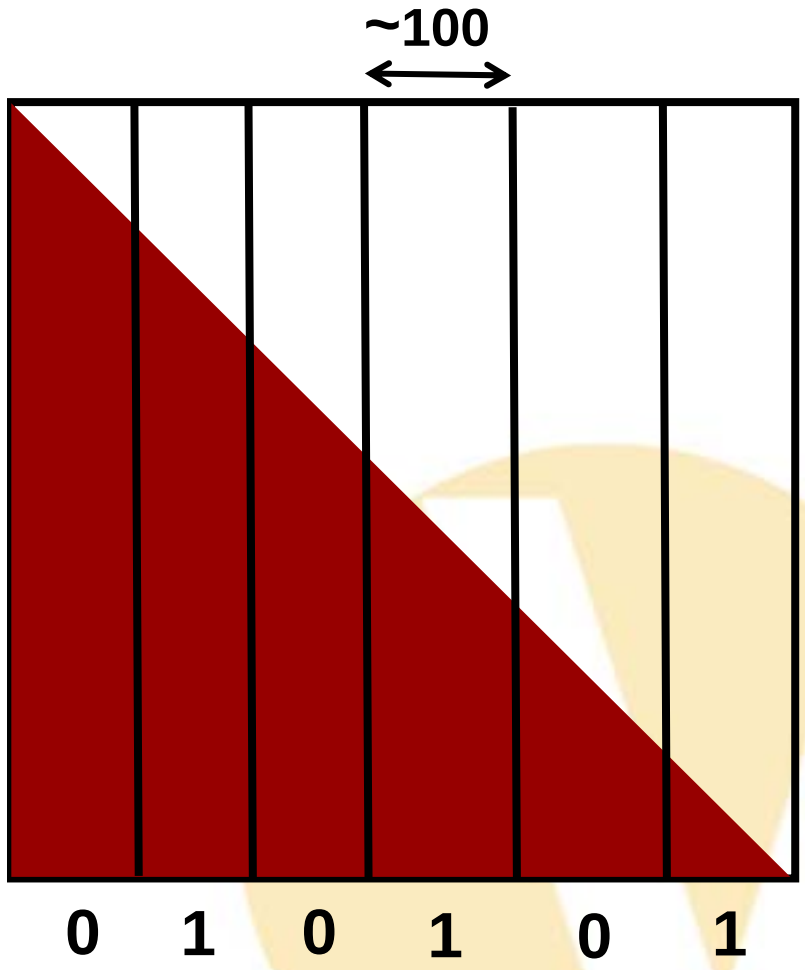
Panels distributed with MPI

ISI Information Sciences Institute

Factor panels broadcast and sent to GPU.

GPU eliminates factor panel and performs large-rank updates to the Schur-complement.

Load imbalance is major bottleneck, so this needs revisiting



USC

USC Viterbi School of Engineering

OpenMP on MIC (and on Xeon host)


```
do j = jl, jr
  do i = jl + 1, ld
    x = 0.0
    do k = jl, j - 1
      x = x + s(i, k) * s(k, j)
    end do
    s(i, j) = s(i, j) - x
  end do
end do
```

```
      sq = jr – jl + 1
c$omp  parallel do
c$omp& shared (jl, jr, sq, ld, s)
c$omp& private (ii, j, i, il, ir)
      do ii = jl + 1, ld, sq
        do j = jl, jr
          il = ii
          ir = min(ii + sq - 1, ld)
          do i = il, ir
            x = 0.0
            do k = jl, j - 1
              x = x + s(i, k) * s(k, j)
            end do
            s(i, j) = s(i, j) - x
          end do
        end do
      end do
```

USC Viterbi School of Engineering

Fortran vs CUDA

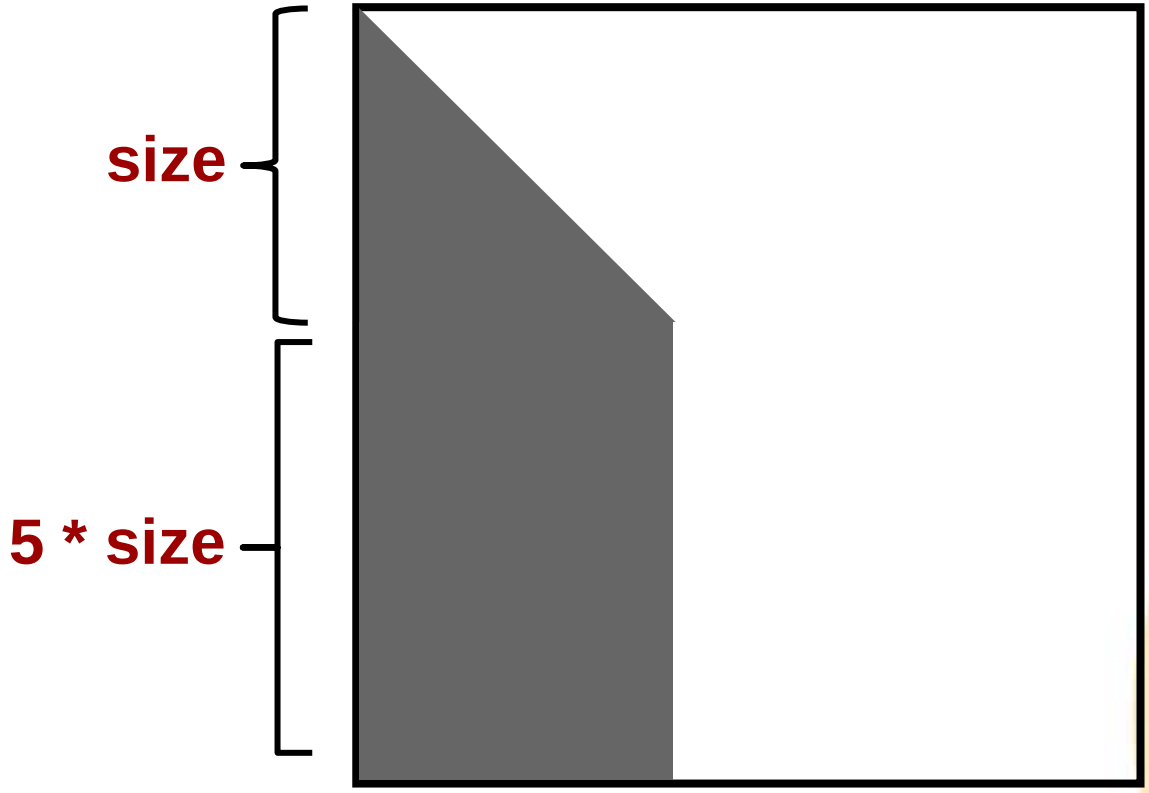
Information Sciences Institute

```
do j = jl, jr
  do i = jr + 1, ld
    x = 0.0
    do k = jl, j - 1
      x = x + s(i, k) * s(k, j)
    end do
    s(i, j) = s(i, j) - x
  end do
end do
```

```
ip=0;
for (j = jl; j <= jr; j++) {
  if(ltid <= (j-1)-jl){
    gpulskj(ip+ltid) = s[IDXS(jl+ltid,j)];
    }
  ip = ip + (j - 1) – jl + 1;
  }

__syncthreads();

for (i  = jr + 1 + tid; i <= ld;
     i += GPUL_THREAD_COUNT) {
  for (j = jl; j <= jr; j++) {
    gpuls(j-jl,ltid) = s[IDXS(i,j)];
    }
  ip=0;
  for (j = jl; j <= jr; j++) {
    x = 0.0f;
    for (k = jl; k <= (j-1); k++) {
      x  = x  + gpuls(k-jl,ltid) * gpulskj(ip);
      ip = ip + 1;
      }
      gpuls(j-jl,ltid) -= x;
    }
  for (j = jl; j <= jr; j++) {
    s[IDXS(i,j)] = gpuls(j-jl,ltid);
    }
  }
```

USC

Factorization Performance on Individual Frontal Matrices



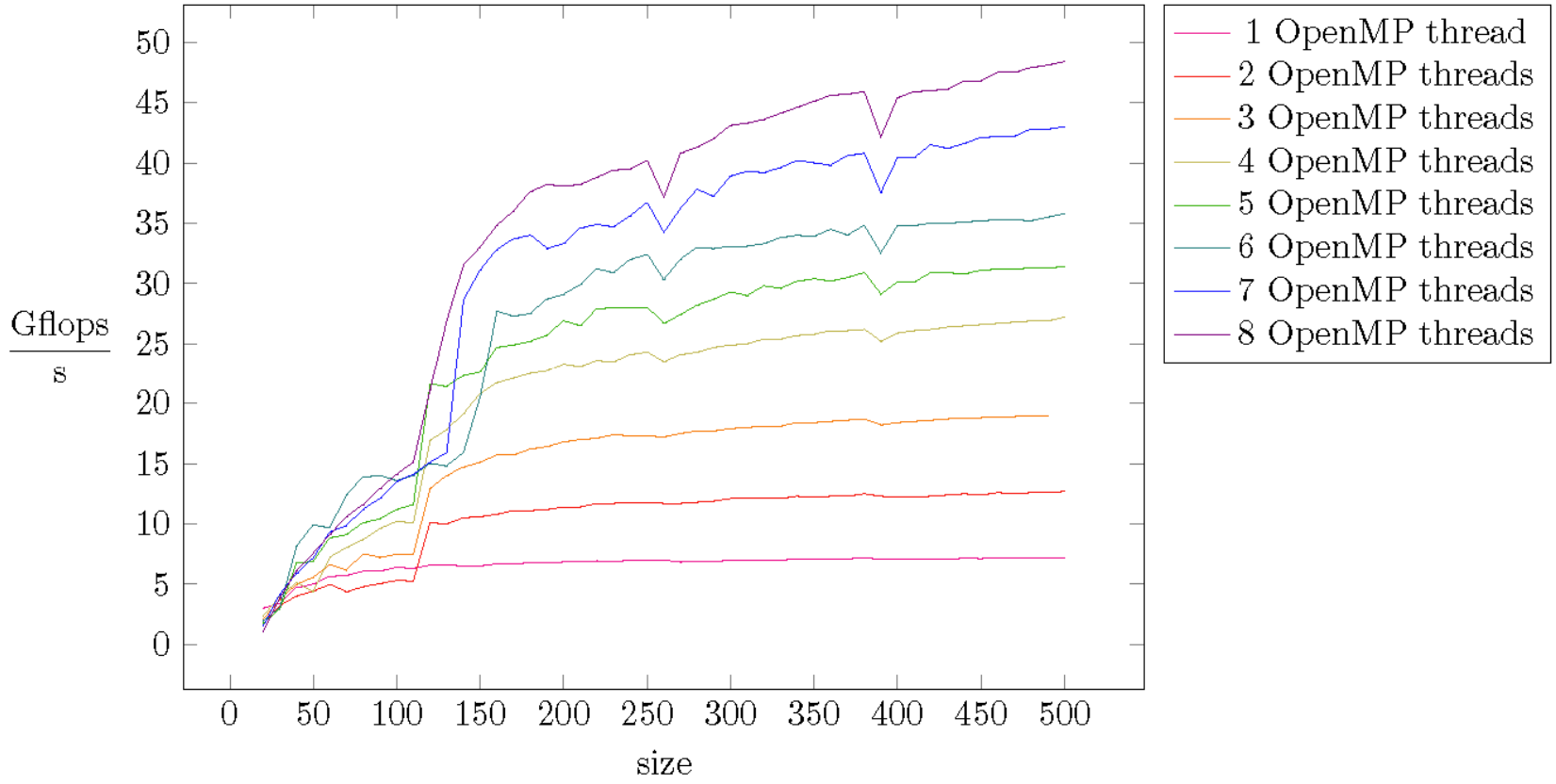
Construct a family of model frontal matrices

Eliminate “size” equations from 5X their number

USC Viterbi School of Engineering

Multithreaded Host Throughput Factoring a Model Frontal Matrix

Information Sciences Institute



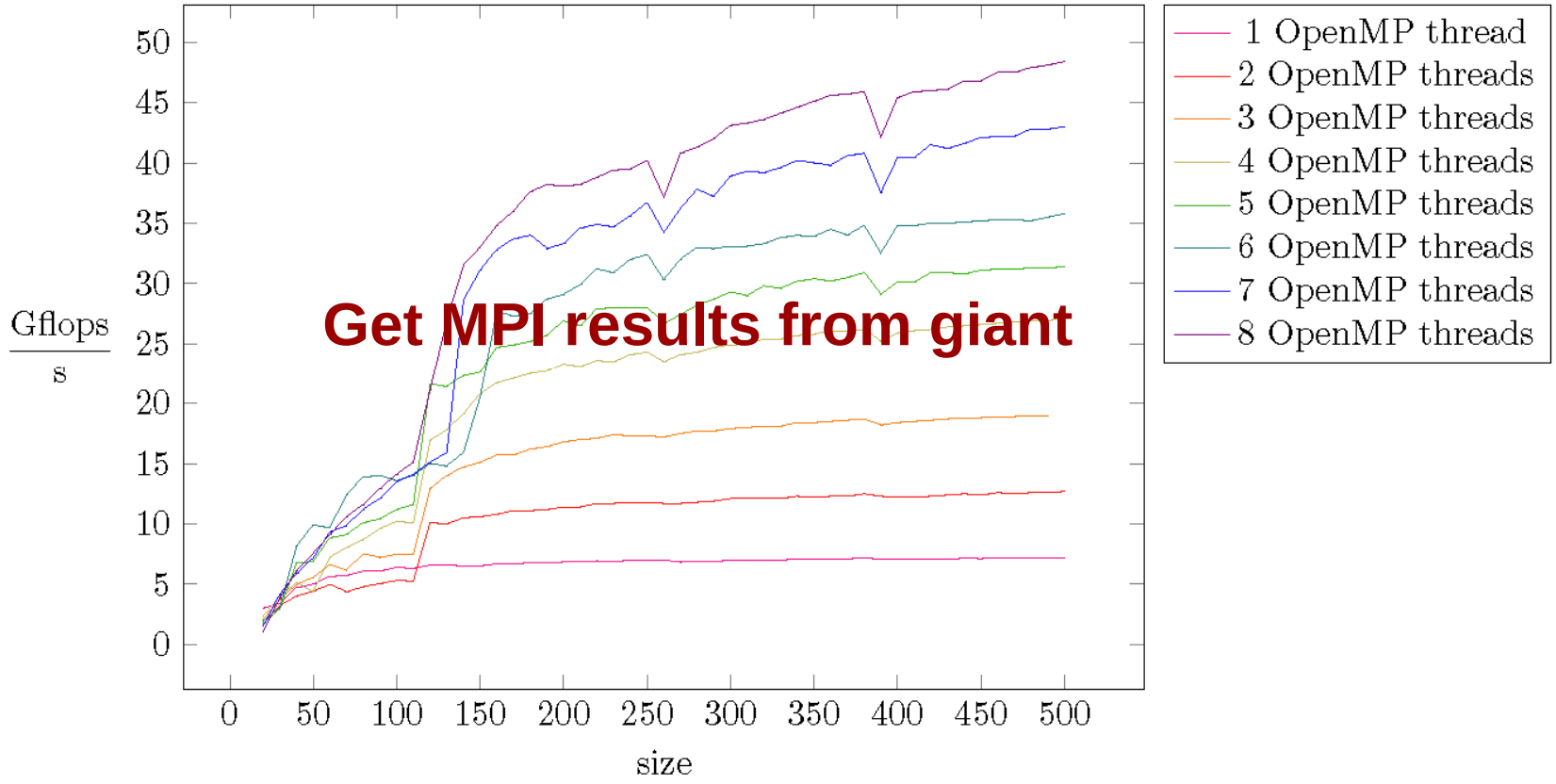
Intel Nehalem

USC

USC Viterbi School of Engineering

Host Throughput with MPI Factoring a Model Frontal Matrix

ISI Information Sciences Institute



Get MPI results from giant

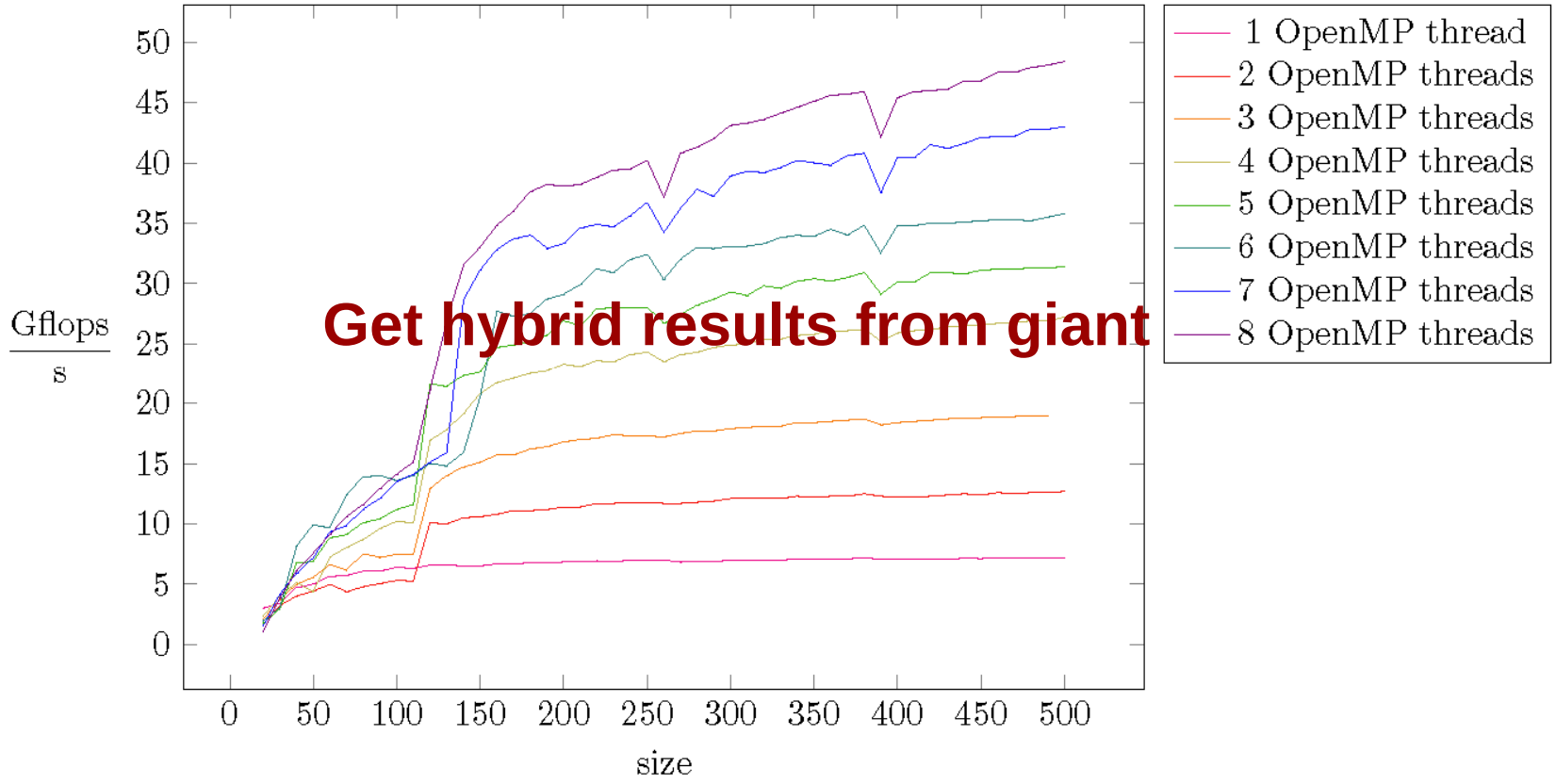
For larger frontal matrices, XX Gflop/s have been observed

USC

USC Viterbi School of Engineering

Hybrid (MPI + OpenMP) Throughput Factoring a Model Frontal Matrix

ISI Information Sciences Institute



Get hybrid results from giant

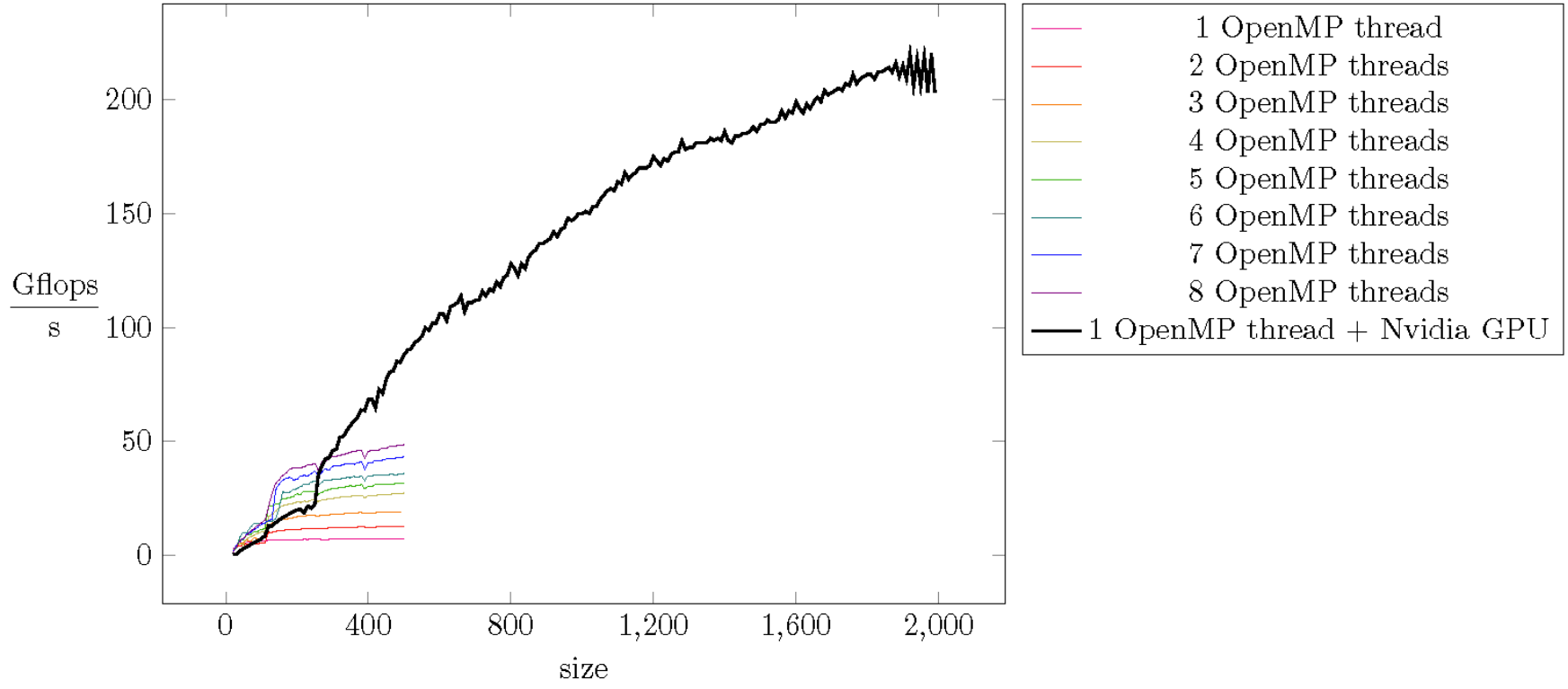
For larger frontal matrices, XX Gflop/s have been observed

USC

USC Viterbi School of Engineering

Nehalem & Tesla Throughput Factoring a Model Frontal Matrix

Information Sciences Institute



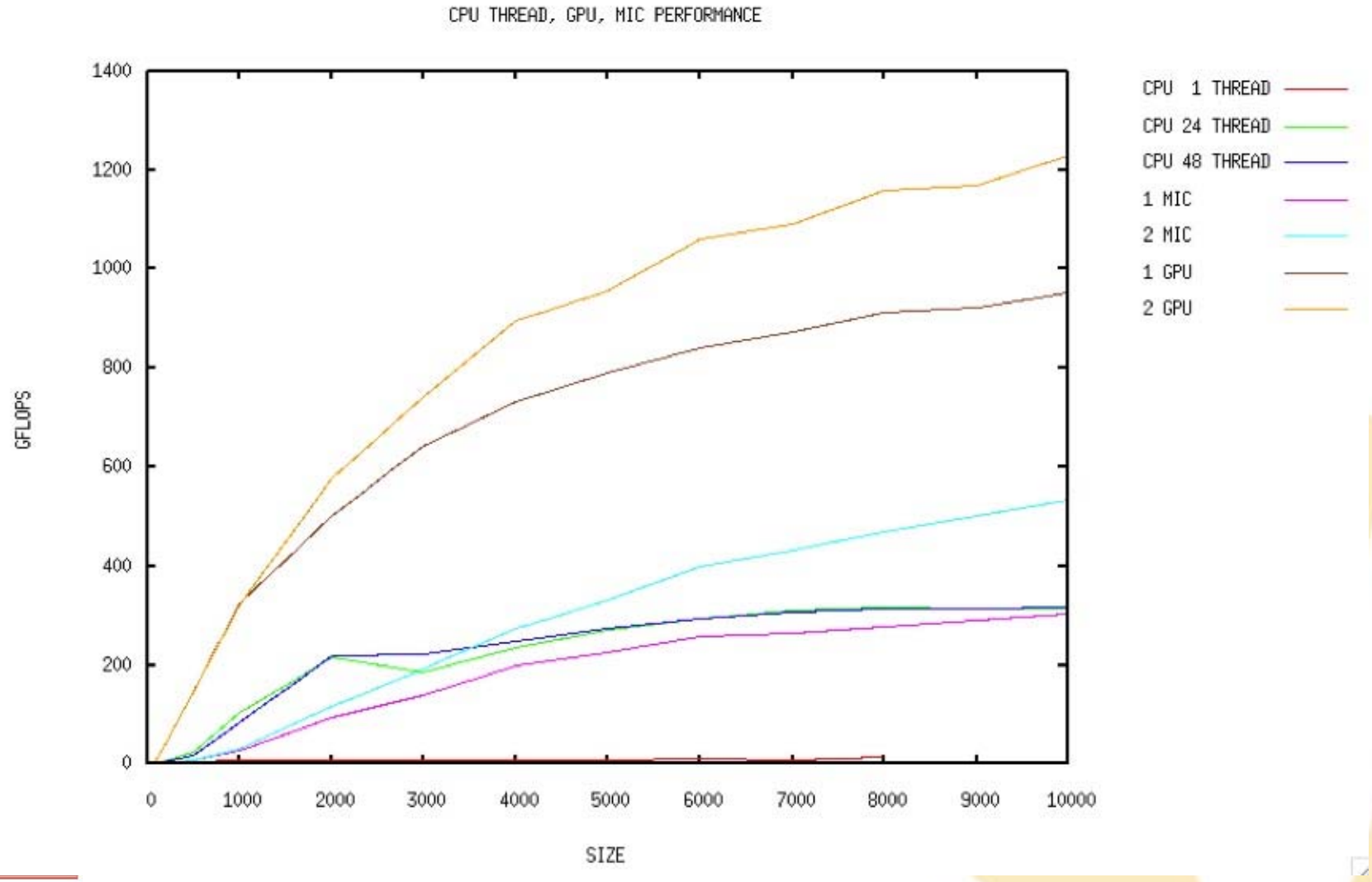
USC

For larger frontal matrices, >250 Gflop/s have been observed

USC Viterbi School of Engineering

Ivy Bridge & Accelerator Factoring a Model Frontal Matrix

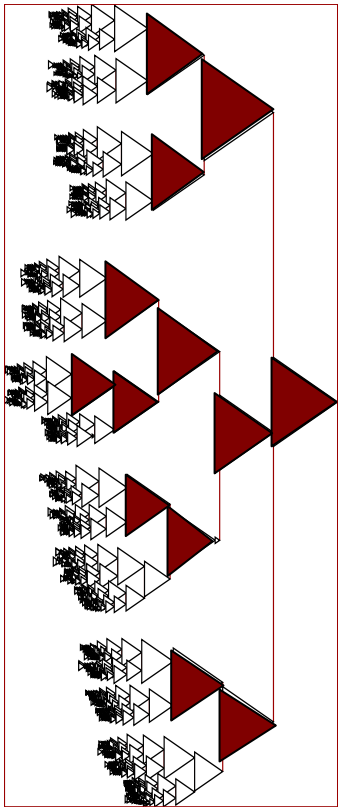
Information Sciences Institute



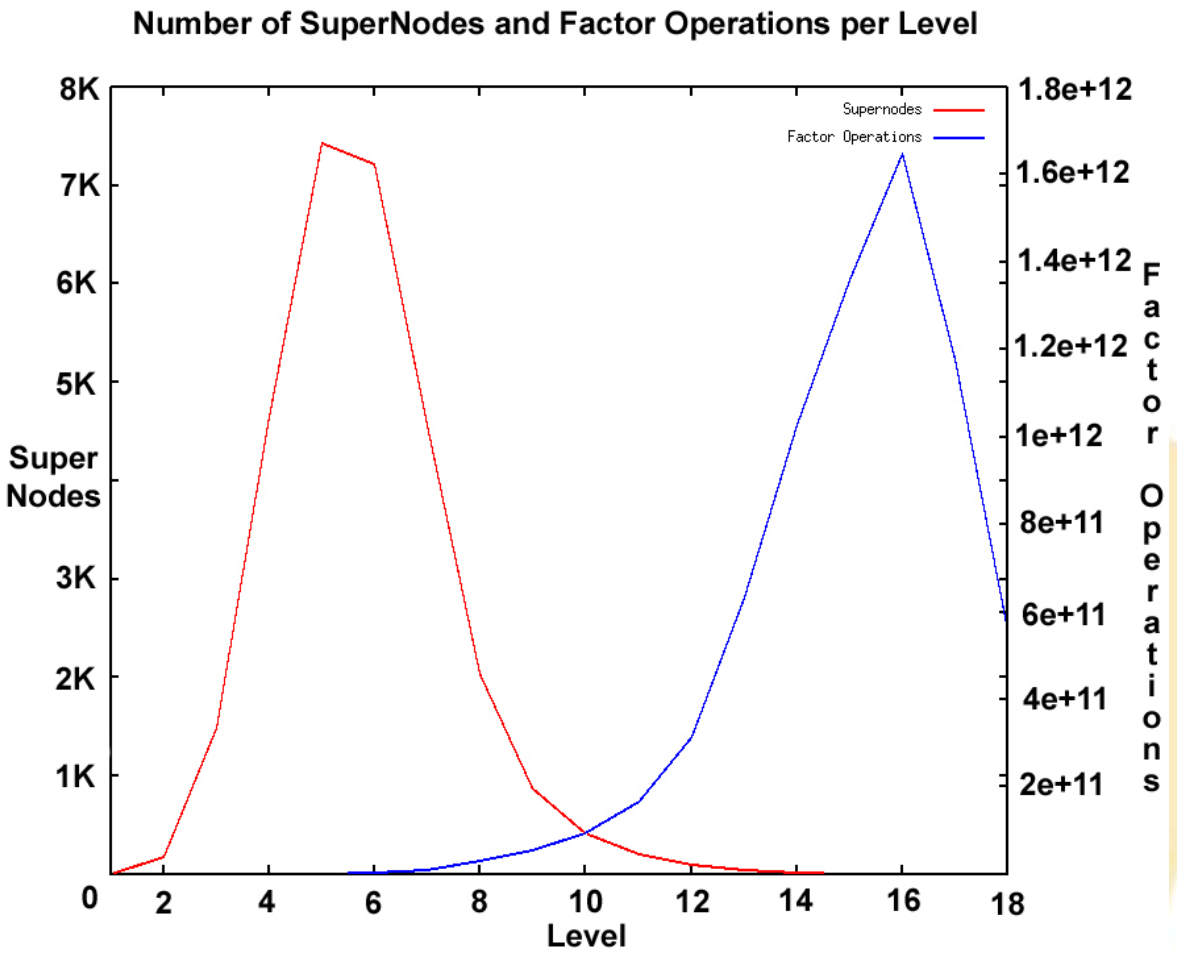
USC Viterbi School of Engineering

GPUs Get the Larger Supernodes

ISI Information Sciences Institute



Hood's tree



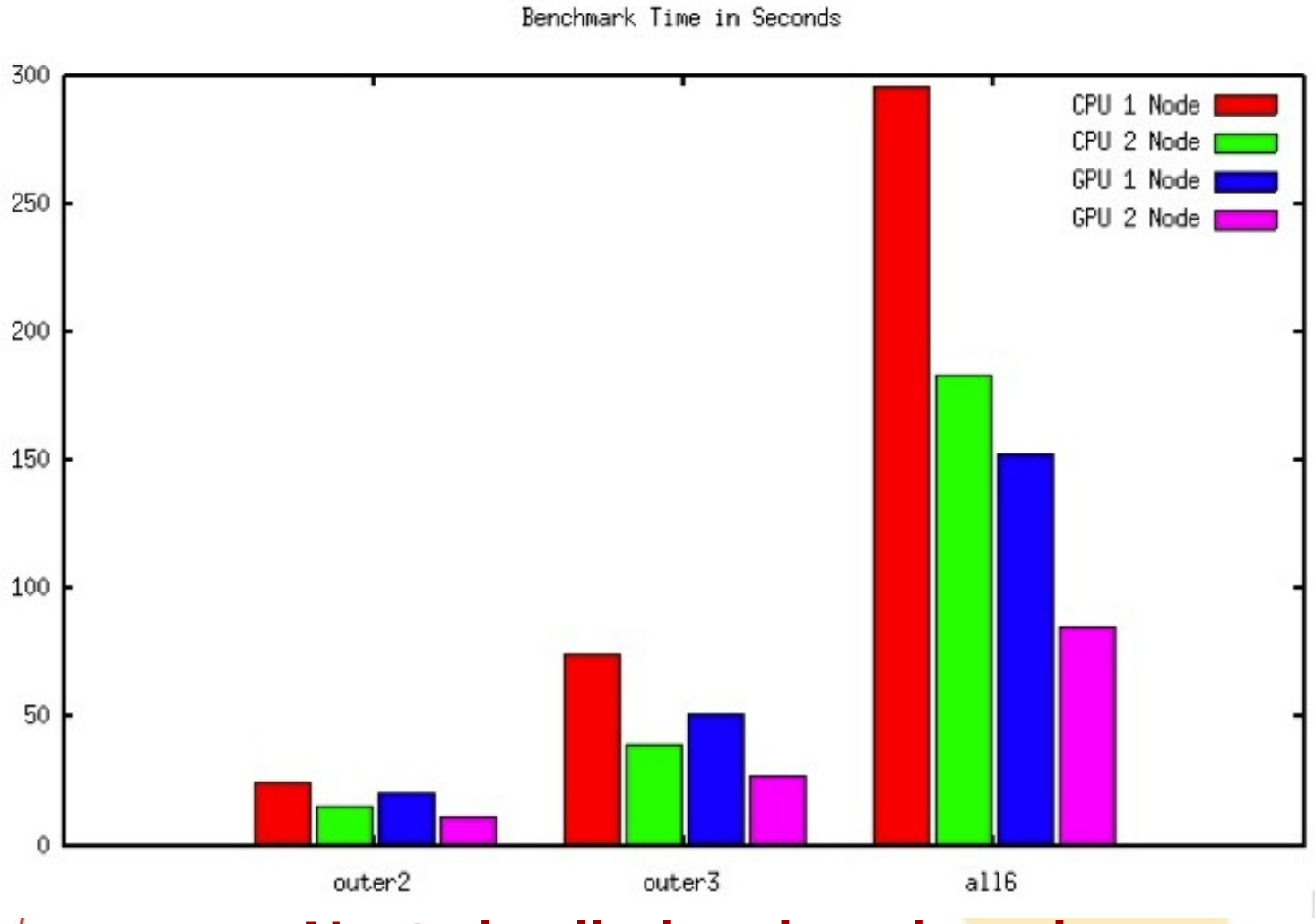
Three cylinders benchmark

USC

USC Viterbi School of Engineering

Relative Time (sec)

Information Sciences Institute



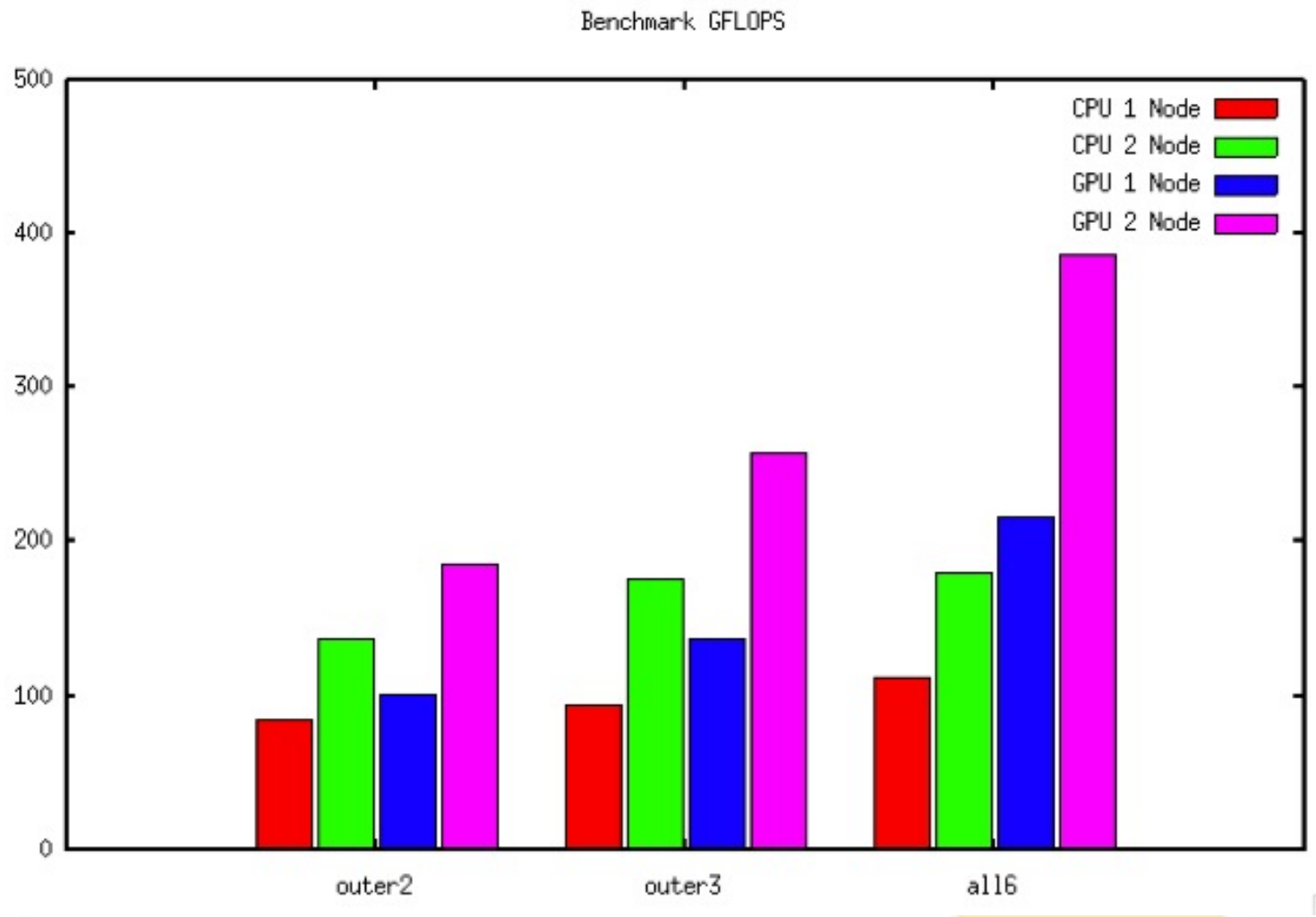
USC

Nested cylinders benchmarks

USC Viterbi School of Engineering

Relative Performance (GFlop/s)

Information Sciences Institute



USC

Nested cylinders benchmarks

USC Viterbi School of Engineering

Work in Progress

ISI Information Sciences Institute

- **Near Term**
  - **Further accelerator performance optimization**
  - **Better MPI + accelerators**
  - **Pivoting on accelerators for numerical stability**
- **Longer Term**
  - **Factor many small frontal matrices on the accelerators**
    - **Assemble initial values**
    - **Maintain real stack**
  - **If the entire matrix fits on the accelerator**
    - **Forward and back solves**
    - **Exploit GDRAM memory bandwidth**

USC

USC Viterbi School of Engineering

Research Initially Funded by US JFCOM and AFRL

ISI Information Sciences Institute

This material is based on research sponsored by the U.S. Joint Forces Command via a contract with the Lockheed Martin Corporation and SimIS, Inc., and on research sponsored by the Air Force Research Laboratory under agreement numbers F30602-02-C-0213 and FA8750-05-2-0204. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government. Approved for public release; distribution is unlimited.

USC

USC Viterbi
School of Engineering

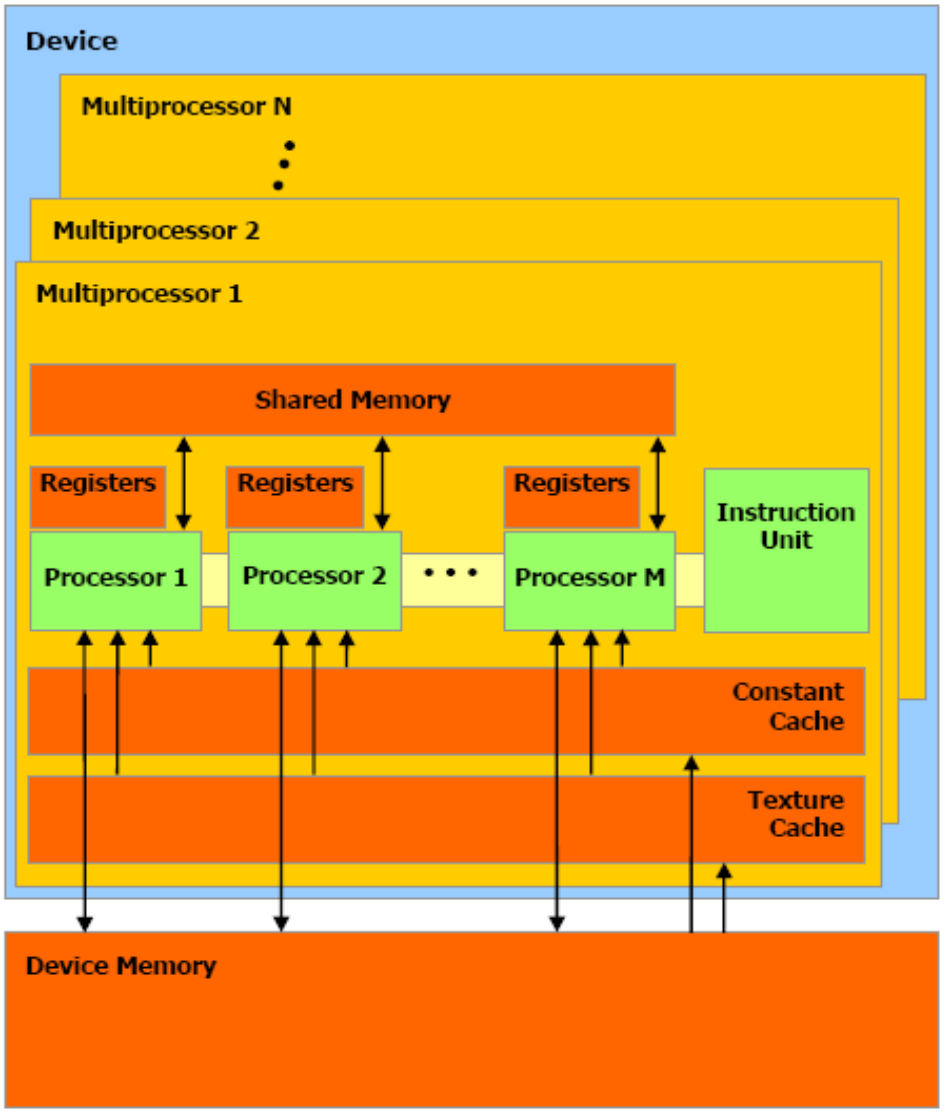
Bonus Slides

ISI
Information Sciences Institute

USC

USC Viterbi School of Engineering

GPU Architecture



A set of SIMD multiprocessors with on-chip shared memory.

Figure 3-1. Hardware Model

Courtesy NVIDIA

Multiple SIMD cores

Multithreaded
- **O(1000) per GPU**

Banked shared memory
- **32 Kbytes Tesla**
- **48 Kbytes Fermi**

Simple thread model
- **Only sync at host**

USC

USC Viterbi School of Engineering

Fortran vs CUDA

Information Sciences Institute

```
do j = jl, jr
  do i = jr + 1, ld
    x = 0.0
    do k = jl, j - 1
      x = x + s(i, k) * s(k, j)
    end do
    s(i, j) = s(i, j) - x
  end do
end do
```

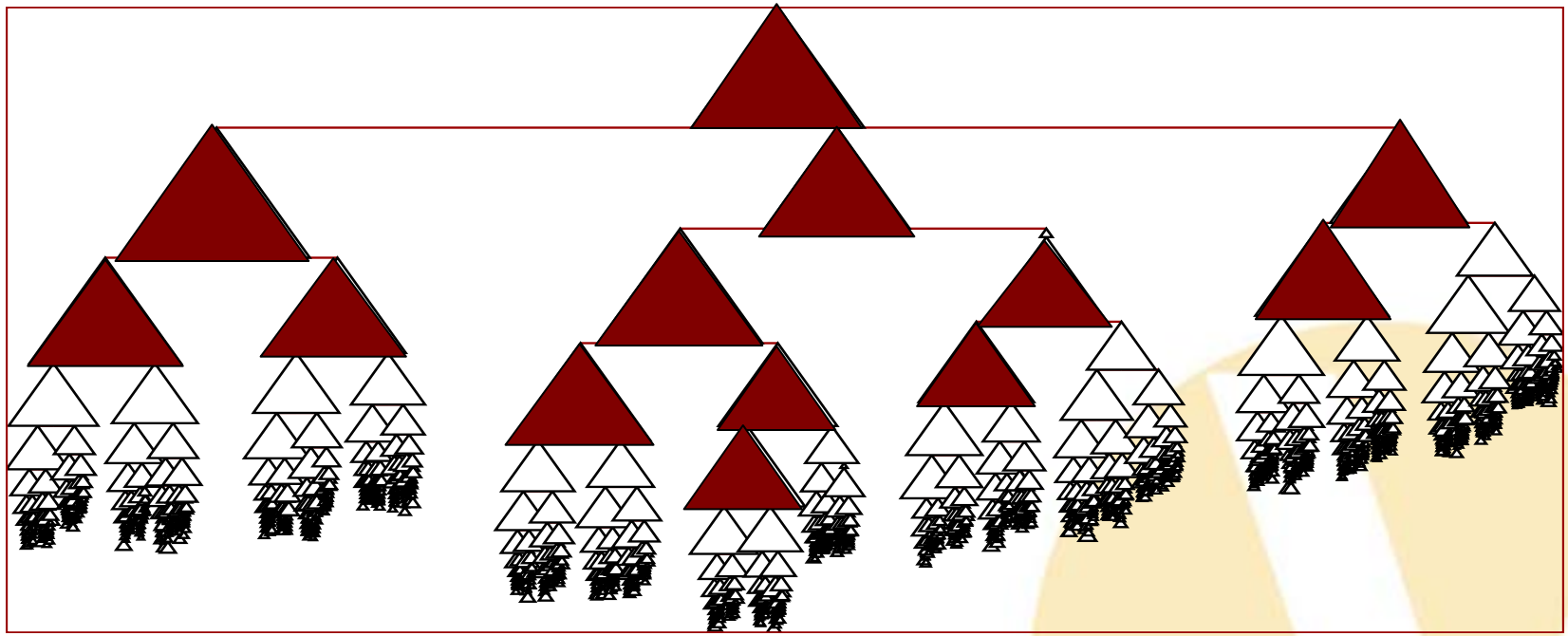
```
ip=0;
for (j = jl; j <= jr; j++) {
  if(ltid <= (j-1)-jl){
    gpulskj(ip+ltid) = s[IDXS(jl+ltid,j)];
    }
  ip = ip + (j - 1) – jl + 1;
  }

__syncthreads();

for (i  = jr + 1 + tid; i <= ld;
     i += GPUL_THREAD_COUNT) {
  for (j = jl; j <= jr; j++) {
    gpuls(j-jl,ltid) = s[IDXS(i,j)];
    }
  ip=0;
  for (j = jl; j <= jr; j++) {
    x = 0.0f;
    for (k = jl; k <= (j-1); k++) {
      x  = x  + gpuls(k-jl,ltid) * gpulskj(ip);
      ip = ip + 1;
      }
      gpuls(j-jl,ltid) -= x;
    }
  for (j = jl; j <= jr; j++) {
    s[IDXS(i,j)] = gpuls(j-jl,ltid);
    }
  }
```

USC

USC Viterbi School of Engineering

GPU gets larger Supernodes

ISI Information Sciences Institute

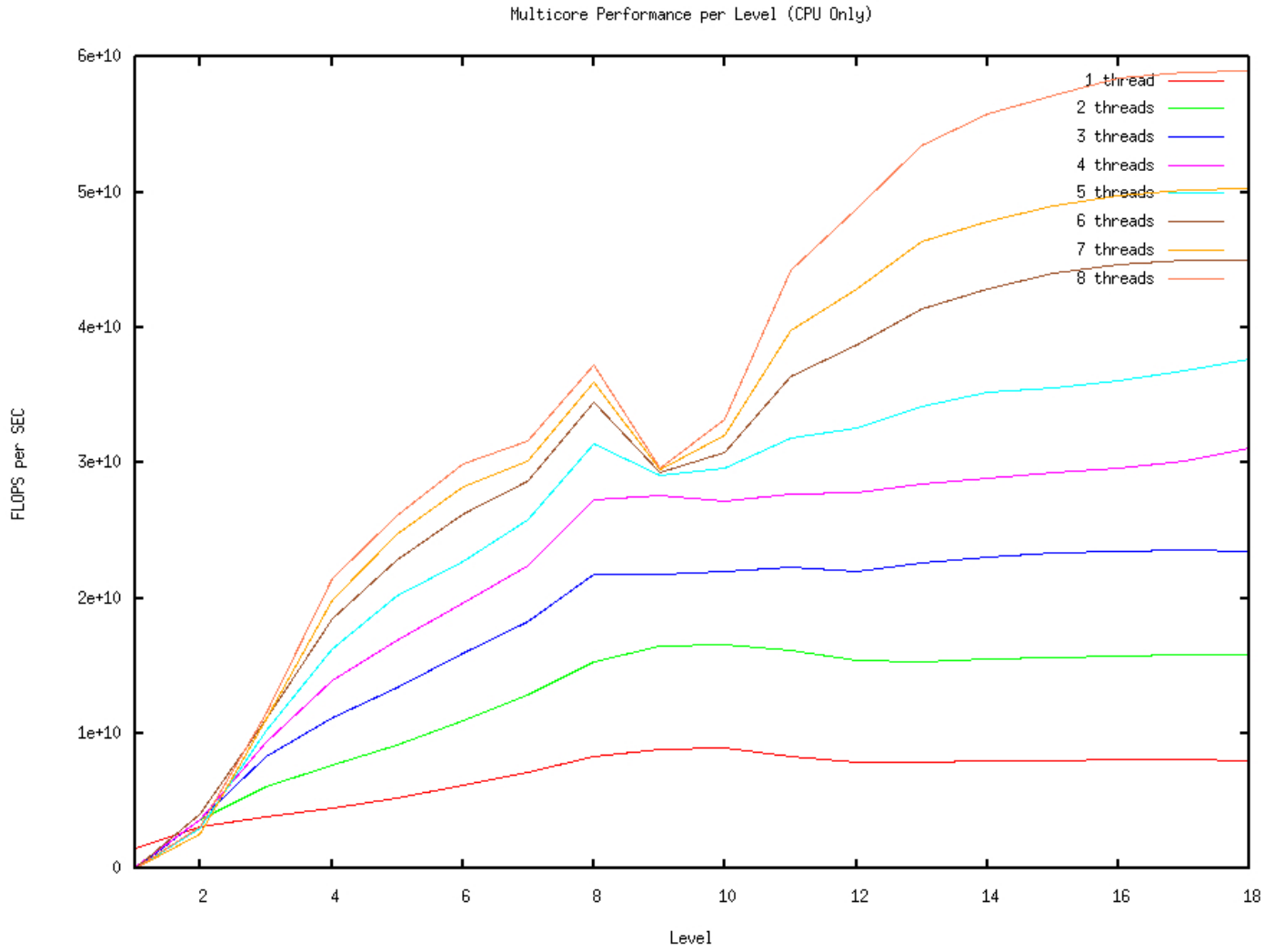


USC

USC Viterbi School of Engineering

Multicore Performance (i4r4) vs. the Elimination Tree

Information Sciences Institute



USC

USC Viterbi School of Engineering

Relative Performance (sec) half-million elements, in-core

MPI, OpenMP, GPU	1st Factorization	LS-DYNA
8 threads	795	1806
8 threads + GPU	283	775
2 MPI x 4 threads	772	1659
2 MPI x 4 + 2 GPU	211	537

USC

Nested cylinders benchmark

USC Viterbi School of Engineering

Relative Performance (sec) million elements, out-of-core

ISI Information Sciences Institute

MPI, OpenMP, GPU	1st Factorization	LS-DYNA
8 threads	6131	15562
8 threads + GPU	2828	9143
2 MPI x 4 threads	6248	15287
2 MPI x 4 + 2 GPU	2567	8214

USC

Nested cylinders benchmark

Time in out-of-core solver (sec)
Eight threads plus GPU

grep WCT mes0000 =>

```
WCT: symbolic factorization =    51.764
WCT: matrix redistribution  =    10.875
WCT: factorization          =  2828.933
WCT: numeric solve          =  1409.523
WCT: total imfslv_mf2       =  4316.063
WCT: symbolic factorization =    40.244
WCT: matrix redistribution  =    10.685
WCT: factorization          =  2679.422
WCT: numeric solve          =  1543.953
WCT: total imfslv_mf2       =  4346.260
```

Note: almost half the elapsed time for LS-DYNA (~4400 sec. out of 9143) is spent in disk I/O.