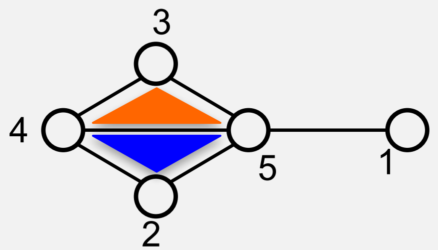
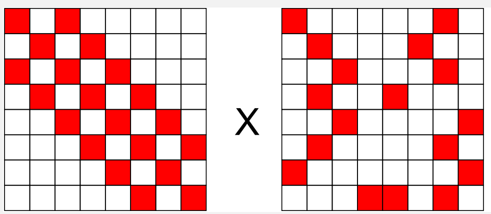
SAND2017-9701C





Sandia National Laboratories

Exceptional service in the national interest

Fast Linear Algebra-Based Triangle Counting with KokkosKernels

Michael Wolf, Mehmet Deveci, Jon Berry, Si Hammond, Siva Rajamanickam

CCR
Center for Computing Research

IEEE HPEC/DARPA/Amazon Graph Challenge

September 13, 2017

U.S. DEPARTMENT OF ENERGY

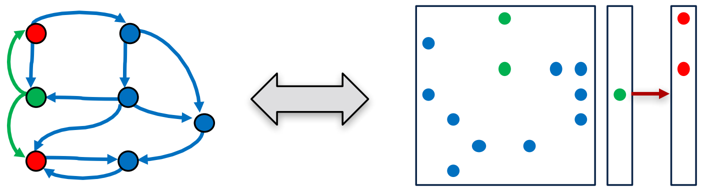
NNSA National Nuclear Security Administration

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

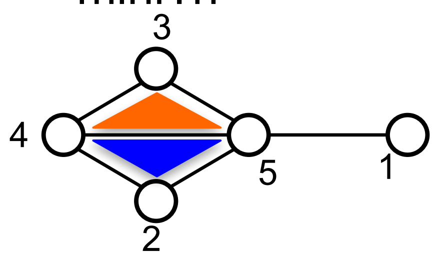
Overview of Approach

Linear Algebra Based Triangle Counting

Graph BLAS

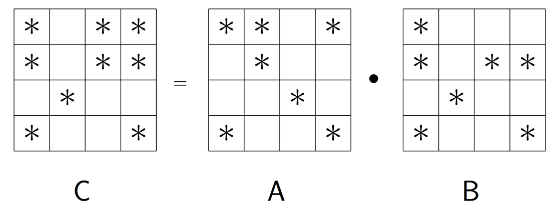


miniTri



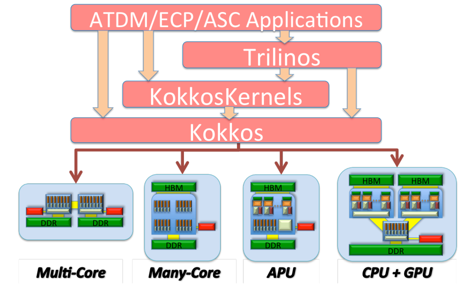
KKMEM: Highly Optimized Matrix-Matrix Multiply

SpGEMM



KokkosKernels

Performance Portability

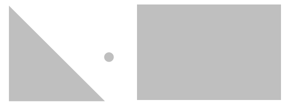


δ

Challenge: Make linear algebra-based triangle counting competitive

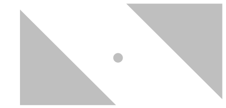
Linear Algebra-Based Triangle Counting

nnz((L*H)==2)



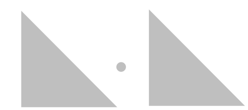
- miniTri, Challenge references
- Wolf, Berry, Stark: *2015 IEEE HPEC*.
- Pro: Good parallel scalability
- Con: high operation count

sum((L*U).*L)/2



- Azad, Buluç, Gilbert. 2015 *IPDPSW GABB*
- Linear algebra version of MinBucket (Cohen), Challenge reference (Julia)
- Pro: Low operation count (2x MinBucket)
- Con: Poor parallel scalability (1D)

sum((L*L).*L)

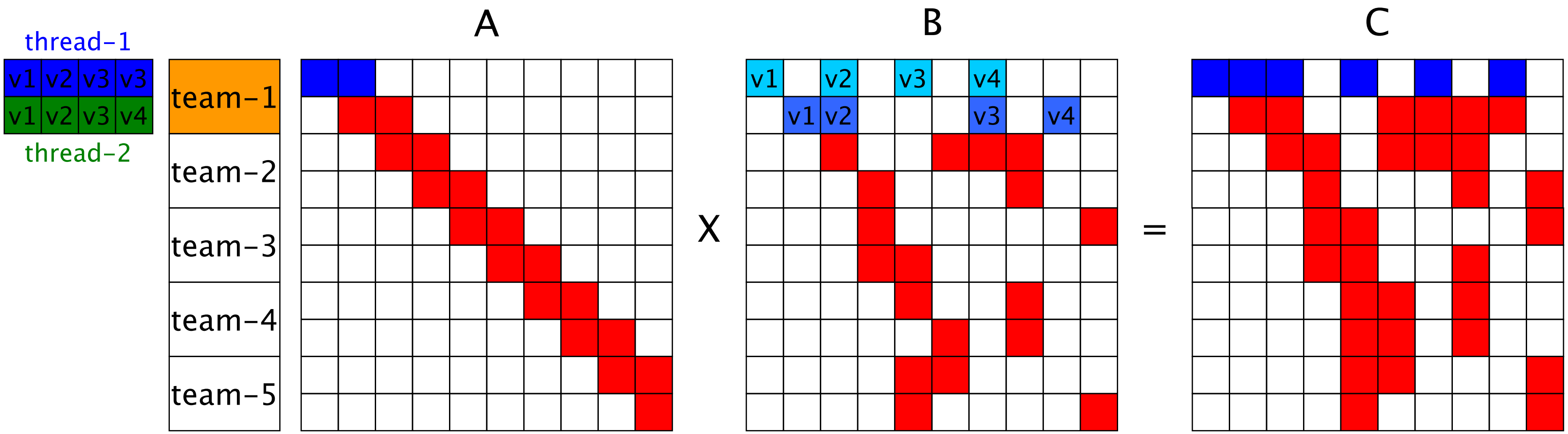


- Extension of (LU).*L method
- "Visits" each triangle/wedge once (extra constraint placed on vertex order for wedges visited)
- Method chosen for challenge
- Pro: Good parallel scalability
- Pro/Con: reasonable operation count (better than LH, slightly worse than LU)

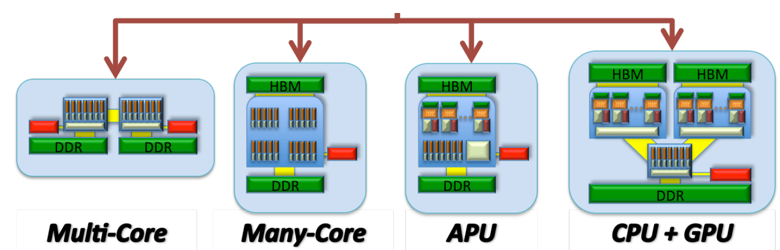
LL method gives good balance of low operation count and good parallel scalability

L, U – lower, upper triangular part of adjacency matrix   H– incidence matrix

KKMEM: KokkosKernels-Based SpGEMM



- KKMEM*
  - Parallel version of Gustavson algorithm: C(i,*) = A(i,*) x B
  - Portable SpGEMM method that runs on CPUs, Xeon Phis, GPUs
  - 2-phases: symbolic and numeric (triangle counting only needs symbolic)
- Hierarchical thread parallelism/data structures
  - Maps hierarchical algorithmic parallelism to SPMD/SIMD computational units
  - Tens/hundreds/thousands of threads

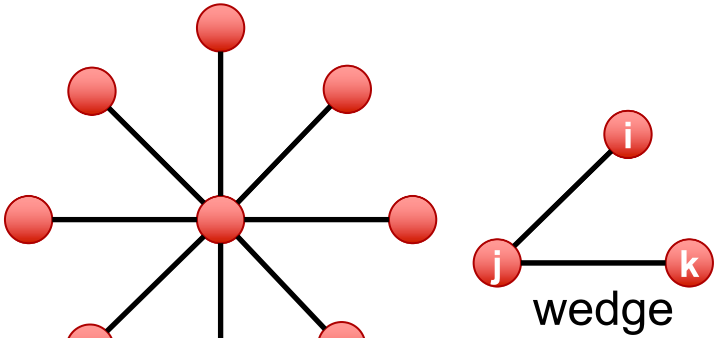


*Deveci, Trott, Rajamanickam: "Performance-Portable Sparse Matrix-Matrix Multiplication for Many-Core Architectures", ASHES Workshop, IPDPSW' 17.

Vertex Ordering and Triangle Counting

Sandia National Laboratories

Vertex Ordering Matters



Wedges with d(i) < d(j) > d(k) : 56
Wedges with d(i) > d(j) < d(k) : 0

Ordering Impacts # operations (# wedges visited)

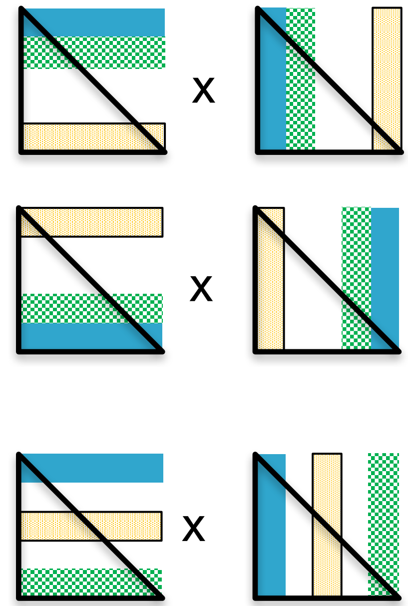
LxL Method Ordering Challenging

Heuristic

Decreasing degree — Good load-balance

Increasing degree

"Interleaved" — Best operation Count (of 3)

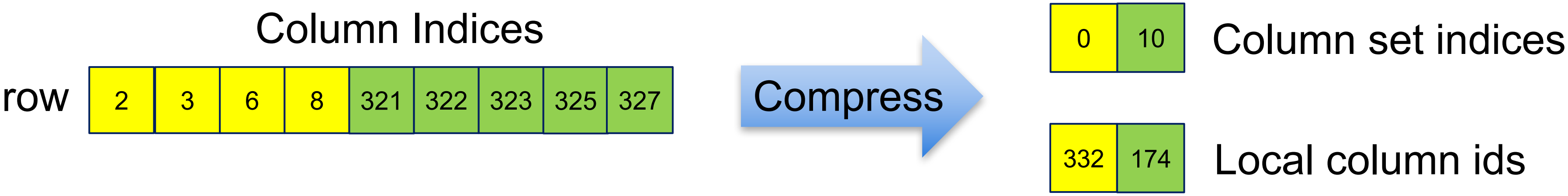


- Densest row
- 2nd Densest row
- Sparsest row

Avoiding computation matters for triangle counting!

Wedges = paths of length 2

Matrix Compression



- Compression used on right hand side matrix
  - Encodes columns using fewer integers
  - Reduces number of operations and memory required in symbolic phase
  - Allows ”vectorized” bitwise union/intersection of different rows
- Effectiveness of compression varies greatly with data
  - Large random graphs compress poorly (R-Mat <1% compression storage)
  - However, still helpful for many random graphs (e.g. power-law) – effective for dense rows (improves load balance, operation count)

Fused Masking

- Computing SpGEMM operation of (L x L).*L or (L x U).*L is inefficient
  - Requires creation of all wedges in graph (large memory requirement since many more wedges than triangles)
- Azad, et al. (2015) showed element-wise multiplication can be combined with SpGEMM into 1 function MaskedSpGEMM(L,L,L)
  - Third argument (L) is output mask
  - GraphBLAS C language API supports masks
- Reduces memory (not storing all wedges)
- Can reduce operation counts (masking unnecessary operations)
  - We chose not to do this)

Masked SpGEMM to avoid storing all wedges

Visitor Pattern

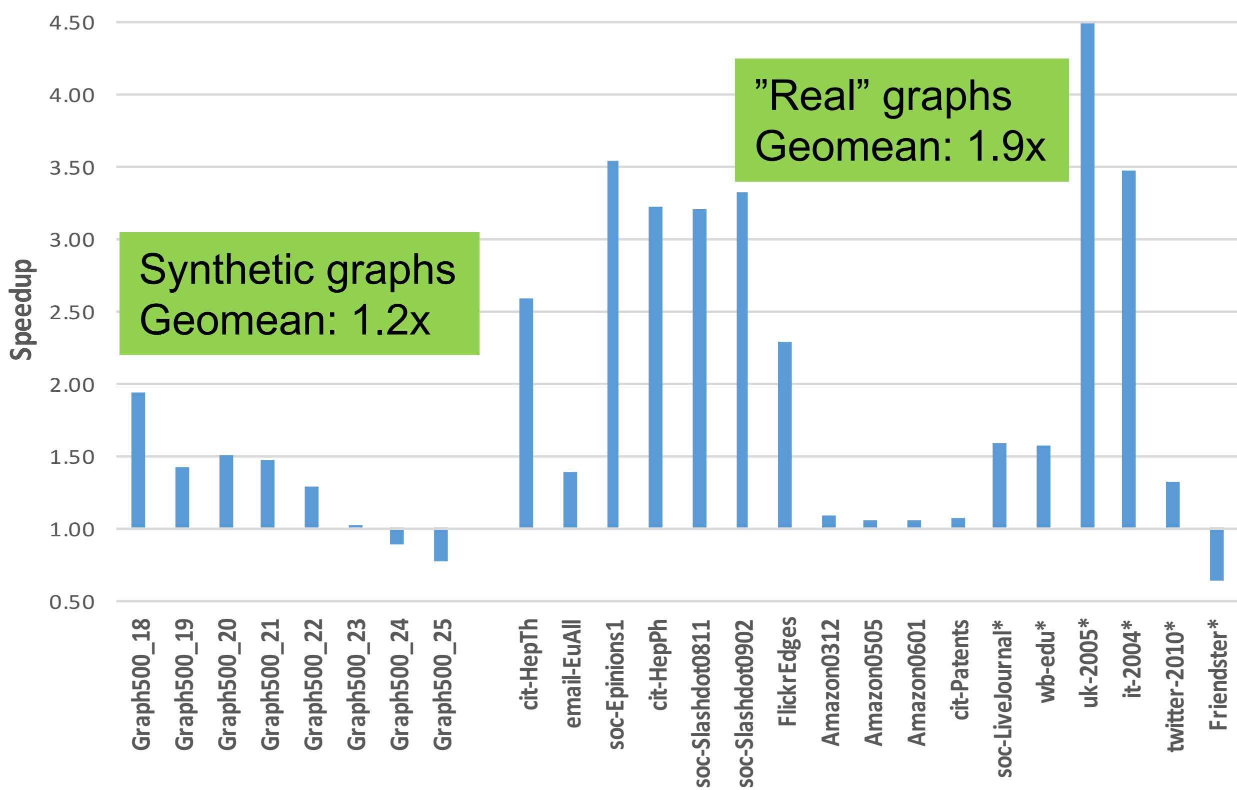
- KKMEM based triangle counting supports visitor pattern
  - Concept fundamental to BGL and MTGL
- Functor passed to triangle identification function, which allows method to be run once triangle is found
  - For triangle counting: triangleCount++;
  - Flexibility allows for more complex analysis of triangles, miniTri

Visitor pattern support provides additional flexibility to analysts

BGL = Boost Graph Library, MTGL = MultiThreaded Graph Library

Results: Speedup Relative to TCM**

Comparison with Ligra's merge based method with Cilk++ (TCM**)



Intel Haswell
32 cores
64 threads

Linear algebra-based triangle counting competitive with (arguably better than) state-of-the-art method

*Additional large graphs from SuiteSparse Matrix Collection

** Shun, Tangwongsan, 2015

Future Work

- Better vertex ordering for LL method
  - Faster parallel implementation (TCM significantly)
  - More optimal orderings
    - Better heuristic for LL method (parallelism, # operations)
    - Nonzero pattern based orderings
- Improved Kokkos scalability
  - TCM's Cilk++ nested parallelism outperformed Kokkos (threads>cores)
- GPU results
  - Excellent KKMEM performance on CS&E applications
  - Minor changes needed for KKMEM-based triangle counting
- miniTri
  - KokkosKernels-based miniTri data analytics miniapp
  - Expect 3-5 order improvement over original miniTri

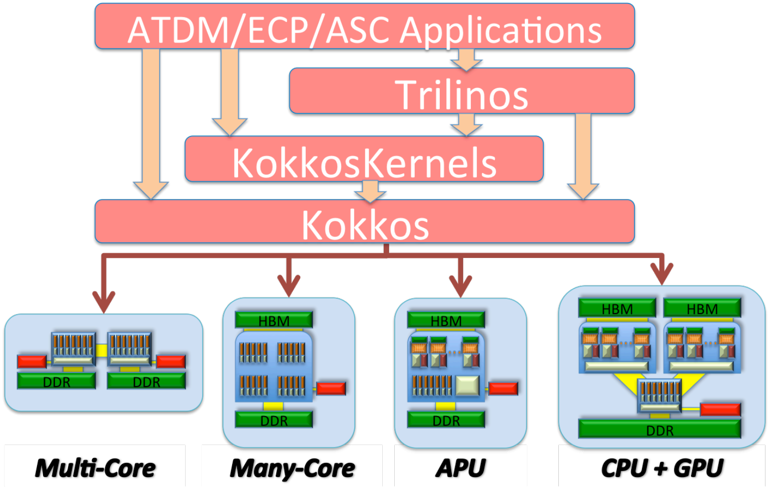
Additional Information

- KKMEM based triangle counting found in KokkosKernels
  - **https://github.com/kokkos/kokkos-kernels**
- Our triangle counting driver and reference implementations distributed with miniTri data analytics miniapp
  - **https://github.com/Mantevo/miniTri** (triangleCounting subdir)
- Additional related publications
  - Deveci, Trott, Rajamanickam: "Performance-Portable Sparse Matrix-Matrix Multiplication for Many-Core Architectures", ASHES Workshop, IPDPSW' 17.
  - Azad, Buluç, Gilbert. "Parallel triangle counting and enumeration using matrix algebra", *Proc. of the IPDPSW, GABB Workshop*, 2015. (LU method)
  - Wolf, Berry, Stark: "A Task-Based Linear Algebra Building Blocks Approach for Scalable Graph Analytics," *2015 IEEE HPEC*. (LH method)
  - Cohen, Jonathan. "Graph twiddling in a mapreduce world." *Computing in Science & Engineering* 11.4 (2009): 29-41. (MinBucket algorithm)
  - Shun, Julian, and Kanat Tangwongsan. "Multicore triangle computations without tuning." *Data Eng. (ICDE), 2015 IEEE 31st Intern. Conf. on*. IEEE, 2015. (TCM method)

Extra

Performance Portability

- Kokkos:
  - Layered collection of template C++ libraries
  - Manages data access patterns
  - Execution spaces, Memory spaces
- Kokkos provides tools for portability
  - Performance portability does not come for free.
  - Not trivial for sparse matrix and graph algorithms

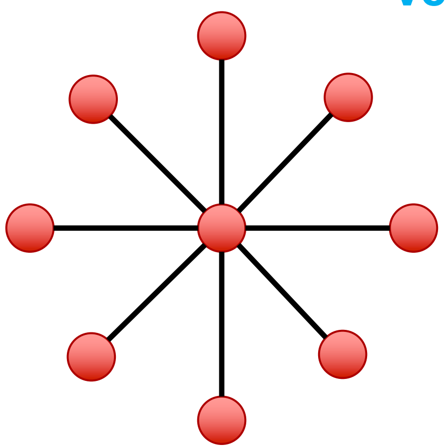


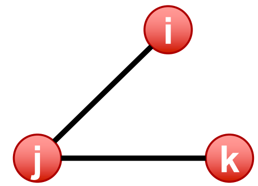
- KokkosKernels:
  - Layer of performance-portable kernels
- We study design decisions for achieving portability for sparse matrix algorithms
  - In this work our application problem: SPGEMM

Vertex Ordering and Triangle Counting

Sandia National Laboratories

Vertex Ordering Matters





\# Wedges with d(j) > d(i), d(j) > d(k): 56
\# Wedges with d(j) > d(i), d(j) > d(k) : 0

- Vertex ordering impacts # of operations (# wedges visited)
  - Linear algebra: impacts # of in SpGEMM, nnz in resulting matrix
- LL method ordering challenging
  - Avoiding dense rows in L -> dense columns in L
  - Reasonable heuristics: **decreasing**/increasing degree, “interleaved”

Avoiding computation matters for triangle counting!

Wedges = paths of length 2