Graphs and HPC:
Lessons for Future Architectures

Bruce Hendrickson

Senior Manager for Computer Science & Mathematics

Sandia National Labs, Albuquerque

Outline

- An introduction to combinatorial scientific computing
- Architectures and programming models for high performance graph algorithms
- Lessons for multi-core machines

Combinatorial Algorithms Enable Computational Science

- "Computational Science & Engineering" brings to mind…
  - Differential equations
  - Numerical methods
- But combinatorial algorithms have long played a key enabling role
  - Sparse direct methods and preconditioning
  - Load balancing and architecture exploitation
  - Optimization and uncertainty quantification
  - Mesh generation, etc.
- Graphs feature strongly in emerging application areas
  - Biological networks
  - Chemistry
  - Advanced data analysis

Sparse Matrix Methods

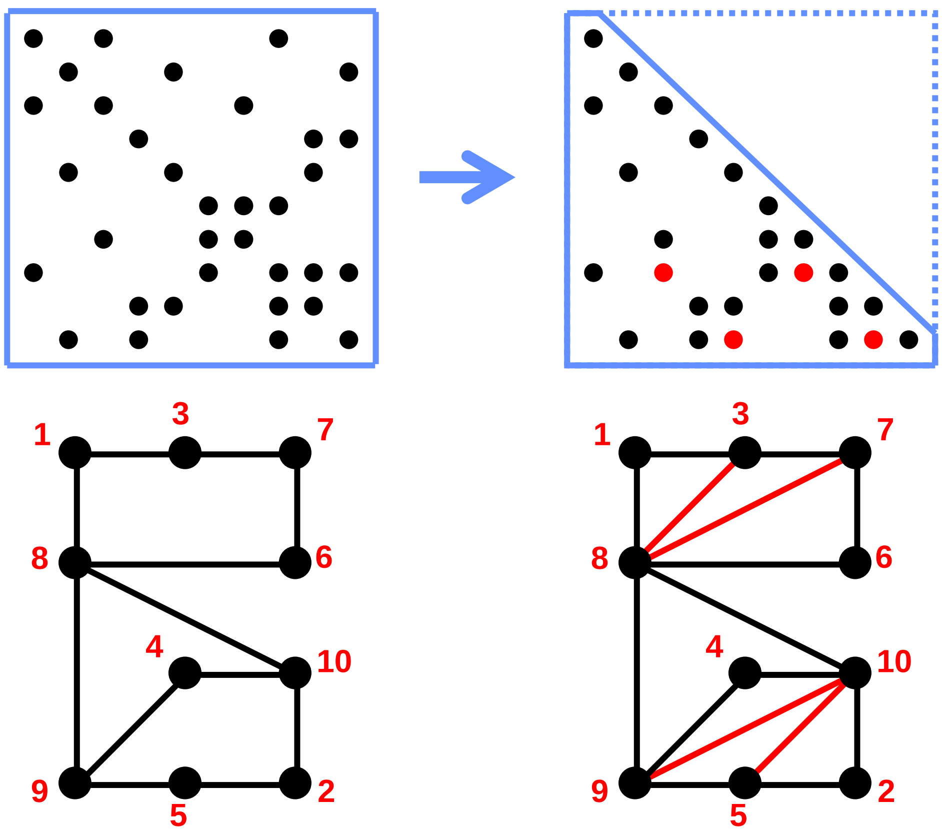
- Reorderings for sparse solvers
  - Powerfully phrased as graph problems
  - Fill reducing orderings
    - Graph partitioning, graph traversals, graph eigenvectors
  - Heavy diagonal to reduce pivoting (matching)

- Data structures for efficient exploitation of sparsity

- Derivative computations for optimization
  - Matroids, graph colorings, spanning trees

- Preconditioning
  - Incomplete Factorizations
  - Partitioning for domain decomposition
  - Graph techniques in algebraic multigrid
    - Independent sets, matchings, etc.
  - Support Theory
    - Spanning trees & graph embedding techniques

Graphs and Sparse Gaussian Elimination



G(A)

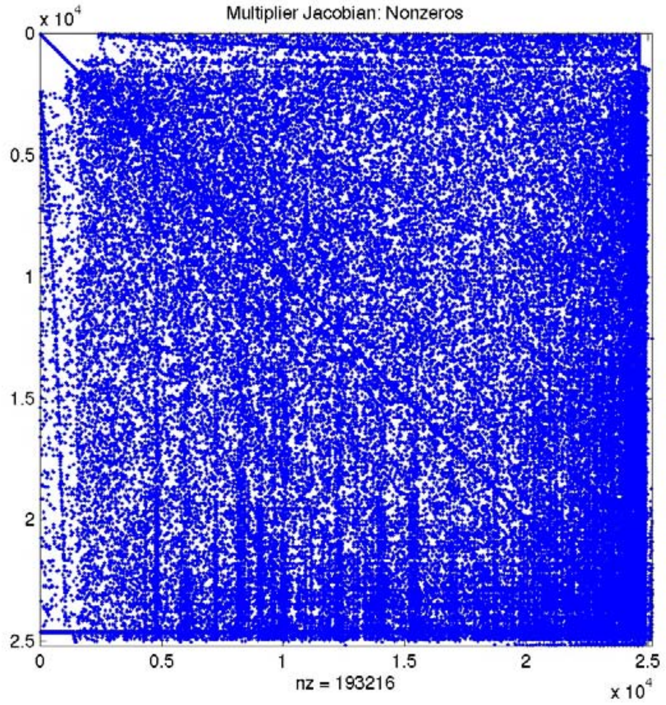
$G^+(A)$
[chordal]

Fill: new nonzeros in factor

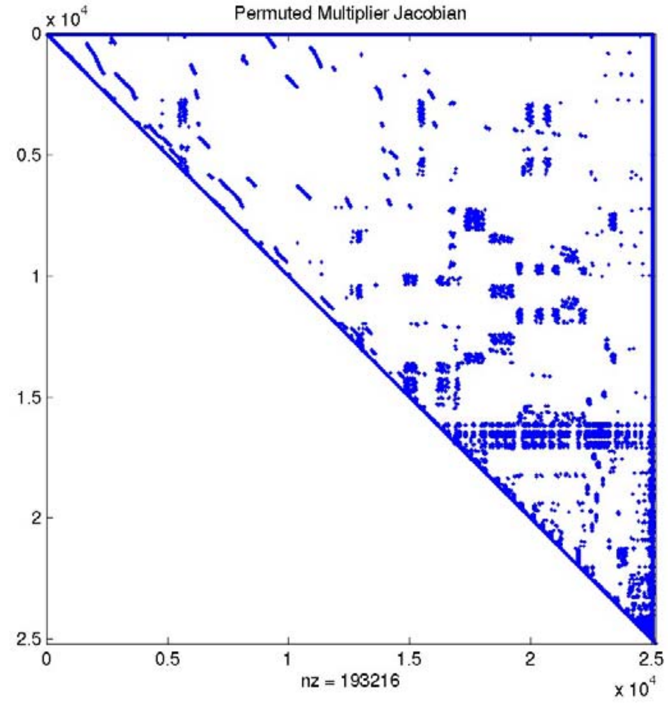
Cholesky factorization:
for j = 1 to n
add edges between j's
higher-numbered neighbors

Sandia National Laboratories

Matrix Reordering: Strongly Connected Components



Before



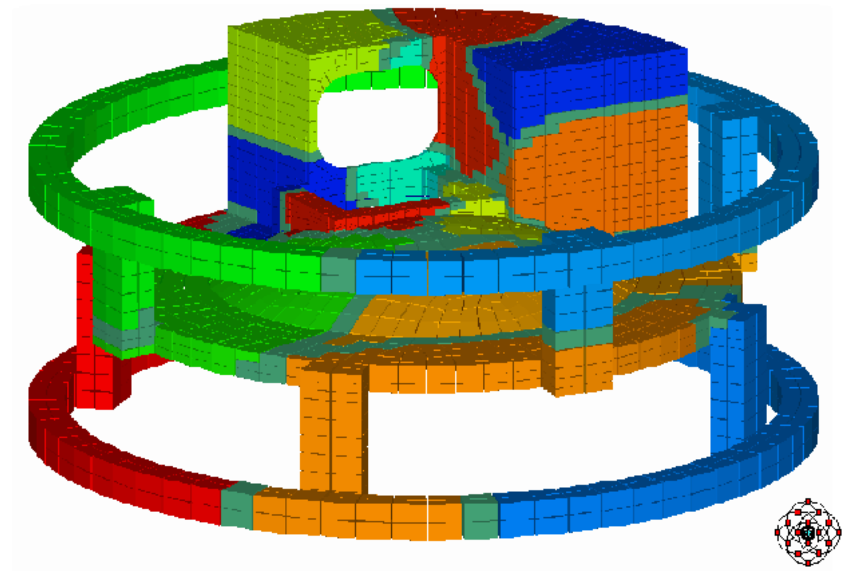
After

Sparse Matrix Methods

- Reorderings for sparse solvers
  - Powerfully phrased as graph problems
  - Fill reducing orderings
    - Graph partitioning, graph traversals, graph eigenvectors
  - Heavy diagonal to reduce pivoting (matching)
- Data structures for efficient exploitation of sparsity
- Derivative computations for optimization
  - Matroids, graph colorings, spanning trees
- Preconditioning
  - Incomplete Factorizations
  - Partitioning for domain decomposition
  - Graph techniques in algebraic multigrid
    - Independent sets, matchings, etc.
  - Support Theory
    - Spanning trees & graph embedding techniques

Sandia National Laboratories

Parallelizing Scientific Computations



- Graph Algorithms
  - Partitioning
  - Coloring
  - Independent sets, etc.
- Geometric algorithms
  - Space-filling curves & octrees for particles
  - Geometric partitioning
- Reordering for memory locality

Sandia National Laboratories

Combinatorial Scientific Computing

- New scientific community formed around this theme in in 2002
  - *Development, application and analysis of combinatorial algorithms to enable scientific and engineering computations*

- Four international workshops (first one supported by ASCR/AMR), special issues of journals, etc.

- SciDAC CSCAPES Institute
  - Combinatorial Scientific Computing and Petascale Simulations

Architectural Challenges for Graphs

- Runtime is dominated by latency
  - Potentially random accesses to global address space
  - Perhaps many at once, but parallelism is fine-grained
- Essentially no computation to hide memory costs
- Access pattern is data dependent
  - Prefetching unlikely to help
  - Usually only want small part of cache line
- Potentially abysmal locality at all levels of memory hierarchy

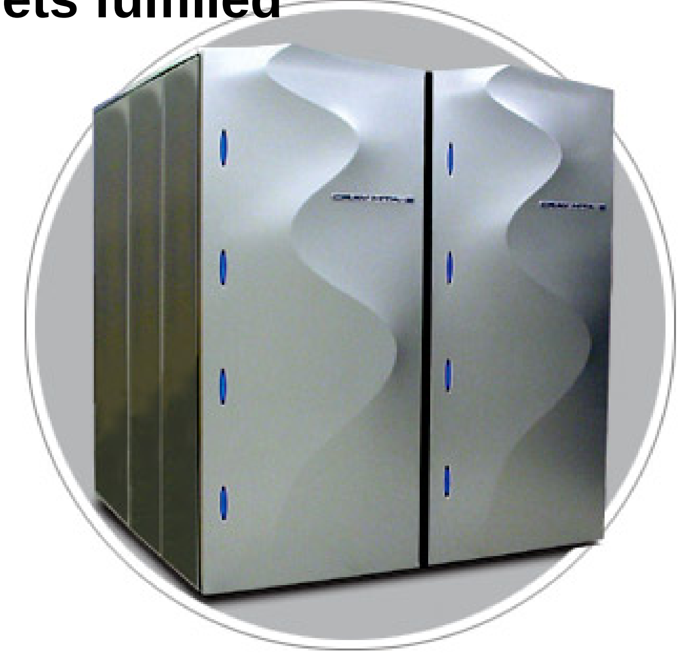
Desirable Architectural Features

- Low latency / high bandwidth
  - For small messages!
- Latency tolerant
- Light-weight synchronization mechanisms
- Global address space
  - Obviate the need for partitioning
  - Avoid memory-consuming profusion of ghost-nodes
  - No local/global numbering conversions
  - Support fine-grained parallelism

- One machine with these properties is the Cray MTA-2
  - And successor *XMT*

How Does the MTA Work?

- Latency tolerance via massive multi-threading
  - Each processor has hardware support for 128 threads
  - Context switch in a single tick
  - Global address space, hashed to reduce hot-spots
  - No cache or local memory. Context switch on memory request.
  - Multiple outstanding loads
- Remote memory request doesn't stall processor
  - Other streams work while your request gets fulfilled
- Light-weight, word-level synchronization
  - Minimizes access conflicts
- Flexibly supports dynamic load balancing
- Notes:
  - MTA-2 is old
  - Clock rate is 220 MHz
  - Largest machine is 40 processors



Case Study: MTA-2 vs. BlueGene/L

- With LLNL, implemented S-T shortest paths in MPI
- Ran on IBM/LLNL BlueGene/L, world's fastest computer
- Finalist for 2005 Gordon Bell Prize
  - 4B vertex, 20B edge, Erdös-Renyi random graph
  - Analysis: touches about 200K vertices
  - Time: 1.5 seconds on 32K processors

- Ran similar problem on MTA-2
  - 32 million vertices, 128 million edges
  - Measured: touches about 23K vertices
  - Time: .7 seconds on one processor, .09 seconds on 10 processors

- **Conclusion: 4 MTA-2 processors = 32K BlueGene/L processors**

Sandia National Laboratories

Broader HPC Relevance

- Existing HPC applications are getting more complex
  - Unstructured and adaptive grids
  - Multiscale and multiphysics
  - Complex data structures and dependencies
- Emerging applications are even more demanding
  - Data analysis, biological networks, decision support
- Architectural ramifications
  - Very high demands on memory system
    - Latency will be increasingly important
  - Extremely difficult micro-load balancing problems

Lessons for Multi-Core Machines

- MTA suggests an alternative model for multi-core node programming
  - Shared memory with simple programming model
  - Latency tolerance
  - Fine grained parallelism & dynamic load balancing

- Many open questions at interface of math and CS
  - How best to build and program multiple cores?
  - Is there a unified programming model that achieves high inter- and intra-node performance?
  - How do we get from here to there?

- Graph algorithms can serve as a “canary in a coal mine” for new architectures, languages, & programming environments
  - Stress systems in ways that anticipate the needs of future applications

Sandia National Laboratories

Acknowledgements

- Collaborators
  - Jon Berry, Rich Murphy, Keith Underwood
  - Alex Pothen, Erik Boman, Karen Devine

- Support from
  - ASCR Applied Math Research Program
  - Sandia LDRD program

- Contact
  - bah@sandia.gov
  - www.sandia.gov/~bahendr

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the US DOE under contract DE-AC-94AL85000

Sandia National Laboratories