CUG 2008 HELSINKI • MAY 5–8, 2008
CROSSING THE BOUNDARIES

Efficient Scaling Up of Parallel Graph Algorithms for Genome-Scale Biological Problems on Cray XT

Kevin Thomas, *Cray Inc.*

CRAY
THE SUPERCOMPUTER COMPANY

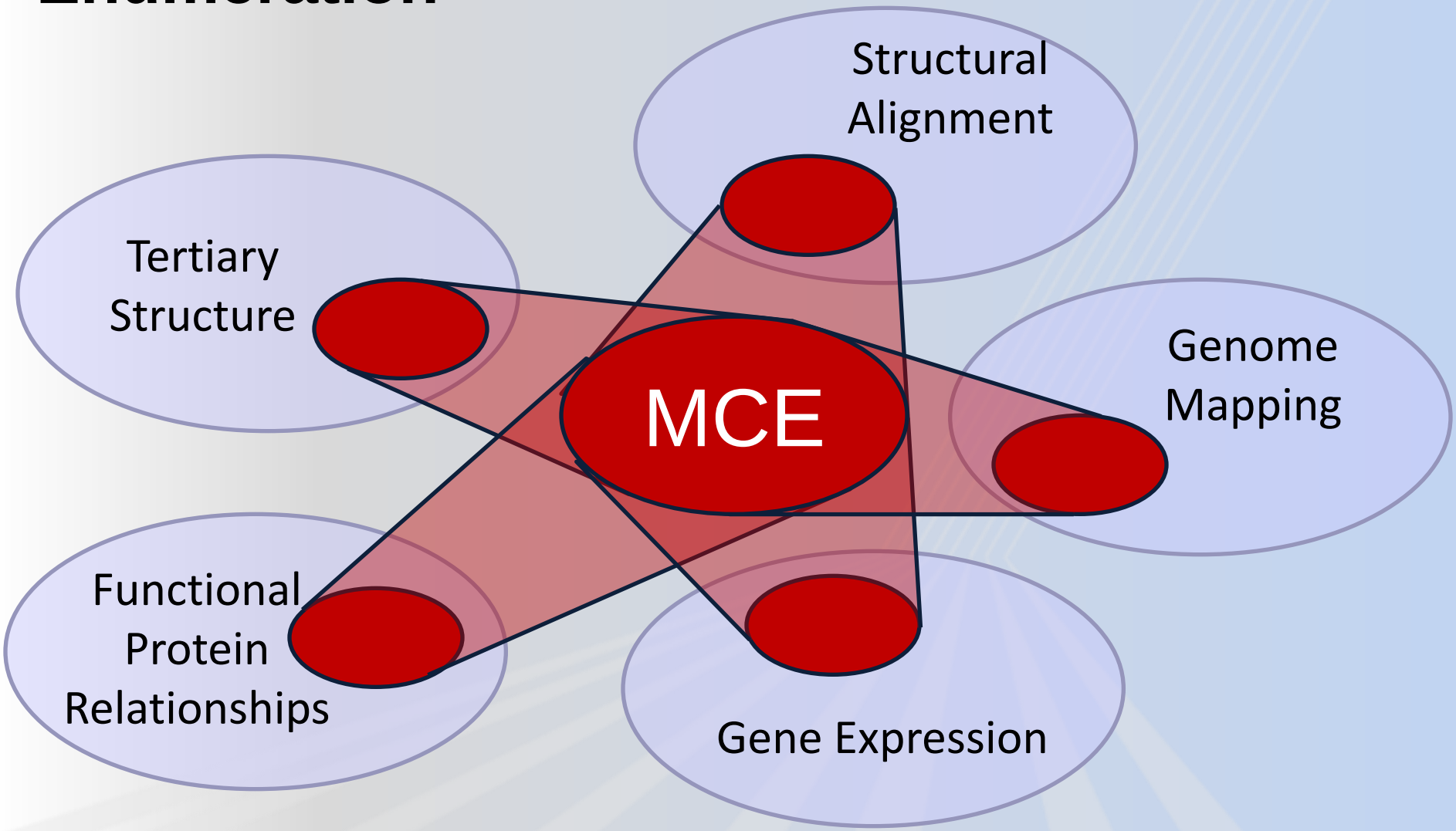
Outline

- Biological networks
- Graph algorithms and terminology
- Implementation of a parallel graph algorithm
- Optimization of single-thread performance
- Lessons learned

Analysis of Biological Networks

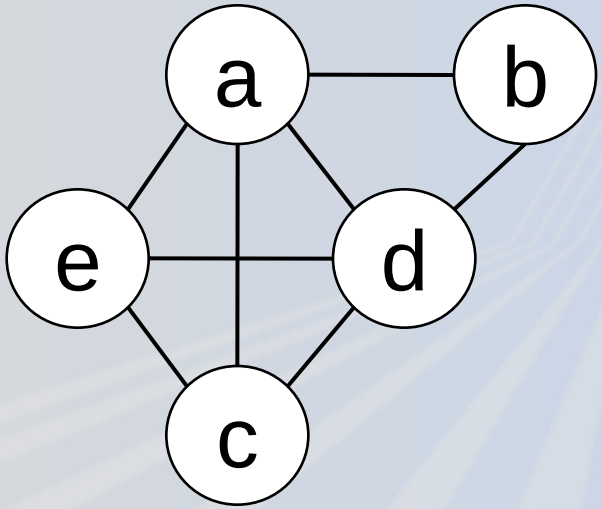
- Analysis of biological networks is increasingly an used tool in biology
- Numerous types of biological networks
  - Gene Expression
  - Protein Interaction
  - Metabolic
  - Phylogenetic
  - Signal Transduction
- Biological networks analysis requires the solution of combinatorial problems
  - Maximal and maximum clique
  - Vertex cover
  - Dominating set
  - Shortest path

Biological Applications of Maximal Clique Enumeration



Graphs and Cliques

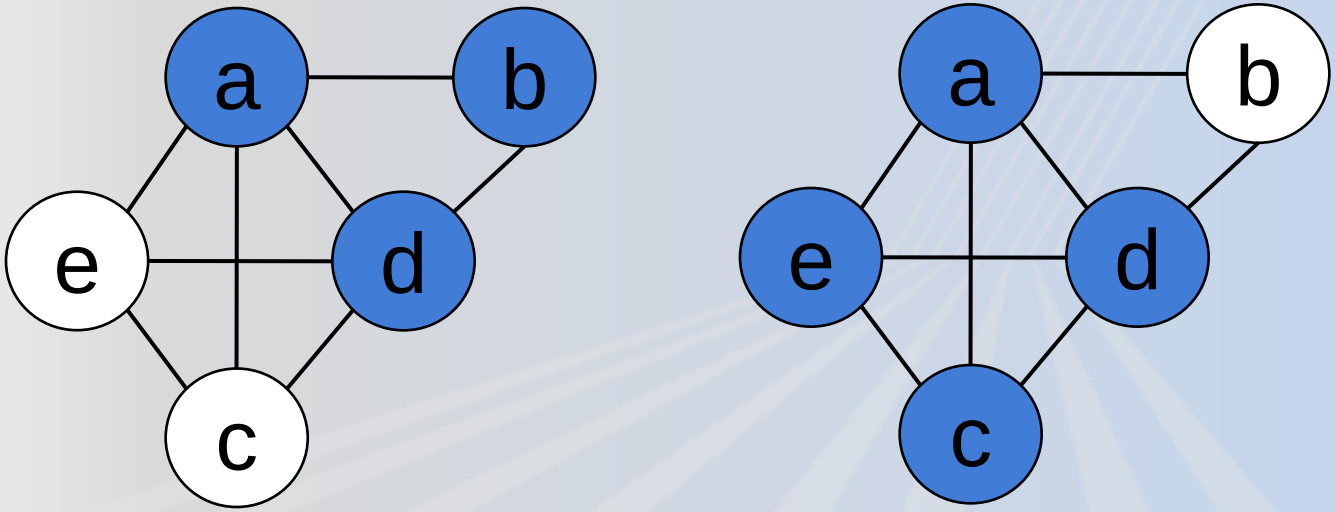
- Graphs are composed of vertices connected by edges
- A clique is a set of vertices which are pair-wise connected
- A maximal clique cannot include any additional vertex and still remain a clique
- (a,c,d,e) is a maximal clique



Step 1 of 2

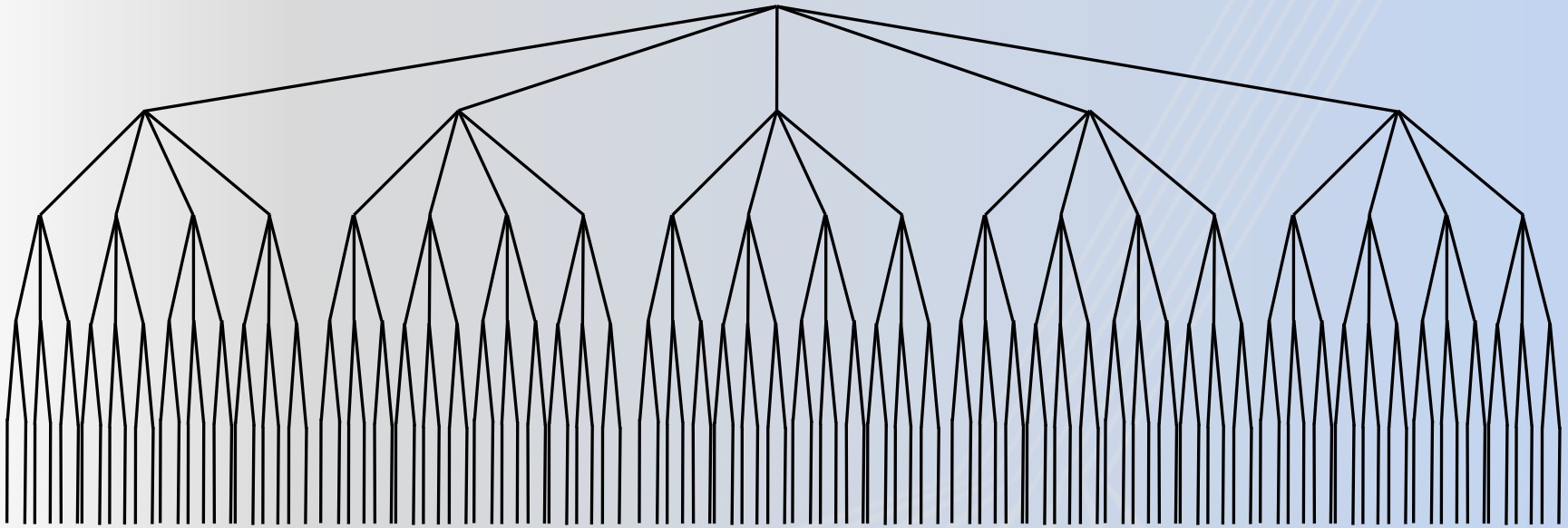
Maximal Clique Enumeration

- Finding all of the maximal cliques of a graph
  - (a,b,d)
  - (a,c,d,e)



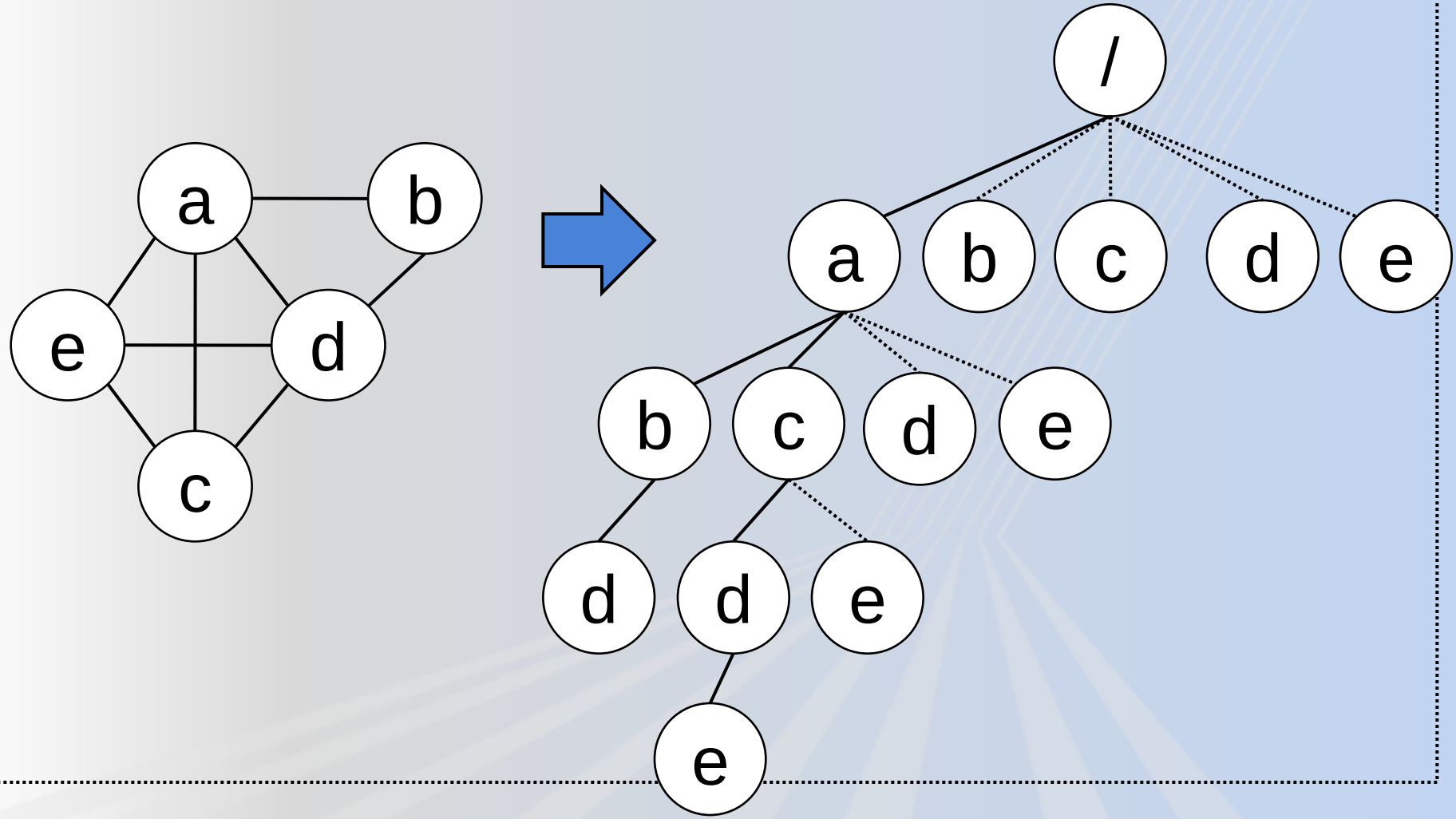
Maximal Clique Enumeration

- Brute Force Search



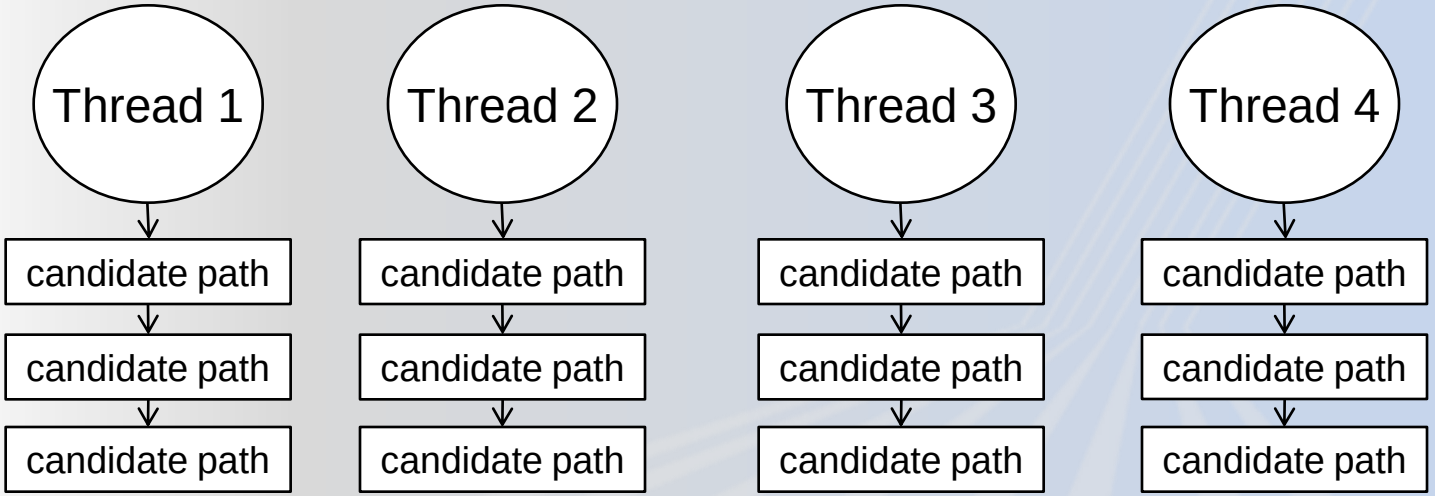
Maximal Clique Enumeration

- Applying a backtracking algorithm results in a search tree



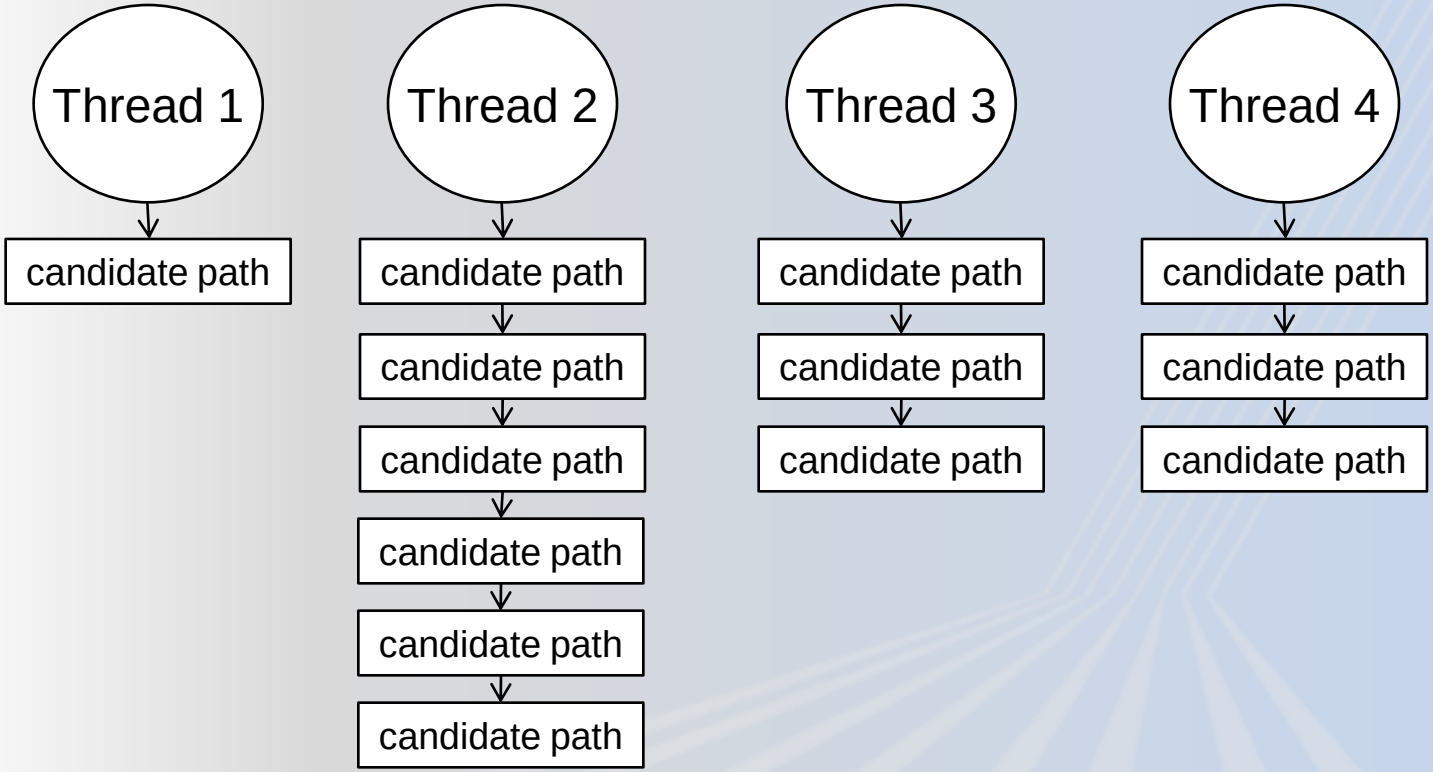
Parallel Maximal Clique Enumeration

- The search tree is divided into independent sub-trees
- Unexplored sub-trees are represented as candidate paths
- The candidate paths are placed into per-thread work pools



Load Balancing

- The work pools can become unbalanced over time



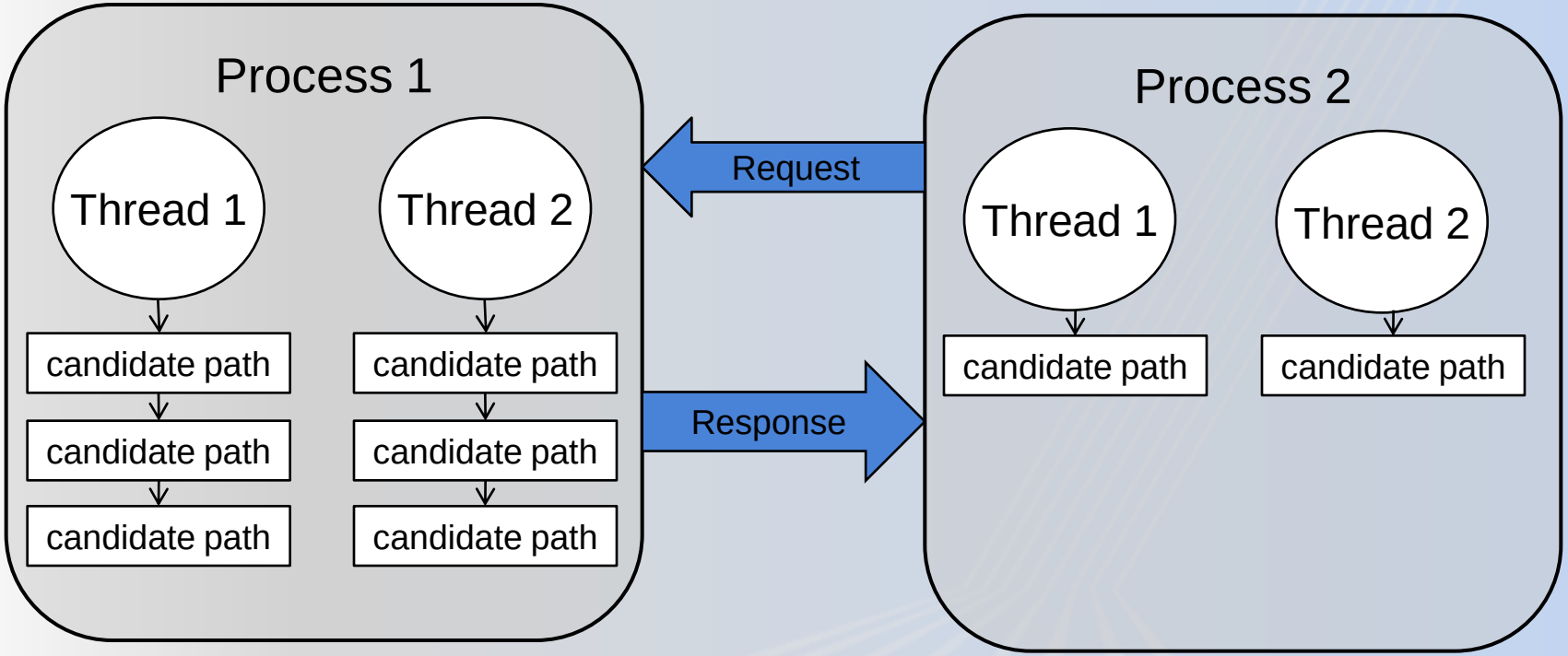
- Dynamic load balancing through work stealing

Step 3 of 3

Two levels of load balancing

- Thread level
  - Used when one thread of a process becomes idle
  - Balances work within a single process
  - Each thread acts on its own to steal work from other threads
  - Locks are used to prevent race conditions
- Process level
  - Used when all threads of a process become idle
  - Local master thread sends a request to another process
  - Remote master thread responds to the request
  - Master thread must poll for incoming requests while performing the main computation

Load balancing between processes



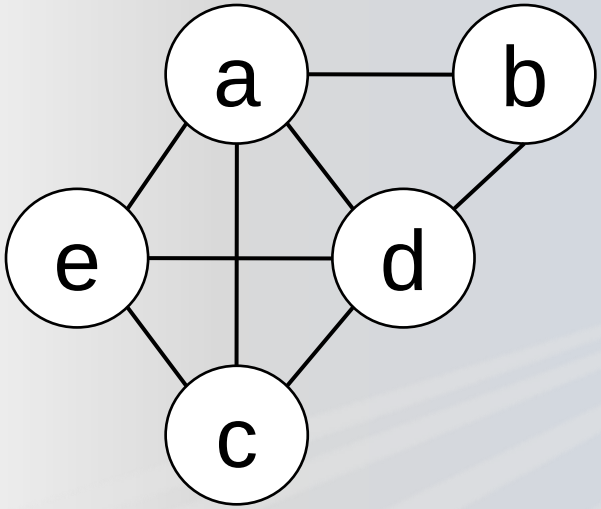
Step 2 of 3

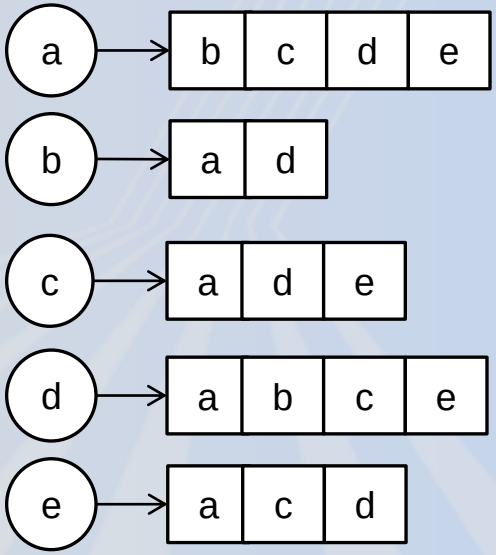
Termination

- Process-level load balancing attempts are made until all processes have been checked
- When no process has work to share, then the idle state is entered
- To synchronize globally, an idle notification is sent to each process
- When all processes are idle, the job can terminate
- $2(N-1)^2$ messages are required for termination

Adjacency Test – Linear List

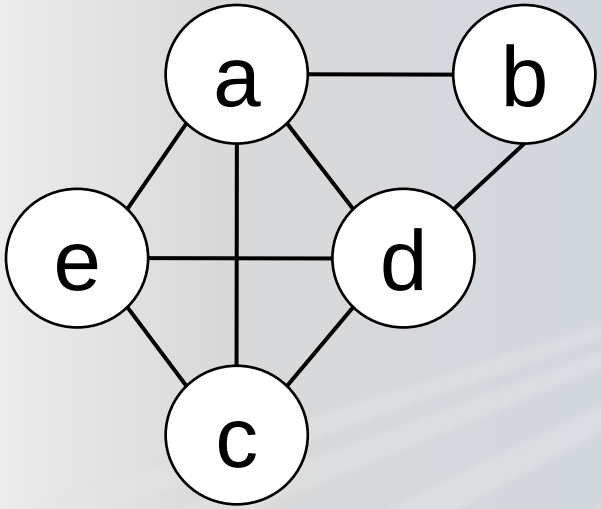
- An important MCE operation is testing two vertices for adjacency
- Graph representation uses a vertex adjacency list
  - Each vertex has a list of adjacent vertices
  - An adjacency test requires a list traversal
  - A linked list is easy to build, but slow to search
  - A linear list (array) is faster to search





Adjacency Test – Bit Matrix

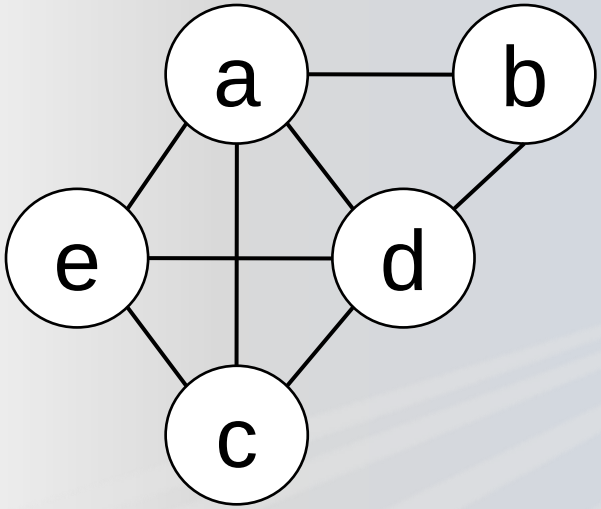
- Adjacency bit matrix has a fast, constant time lookup
- Memory requirement is N^2

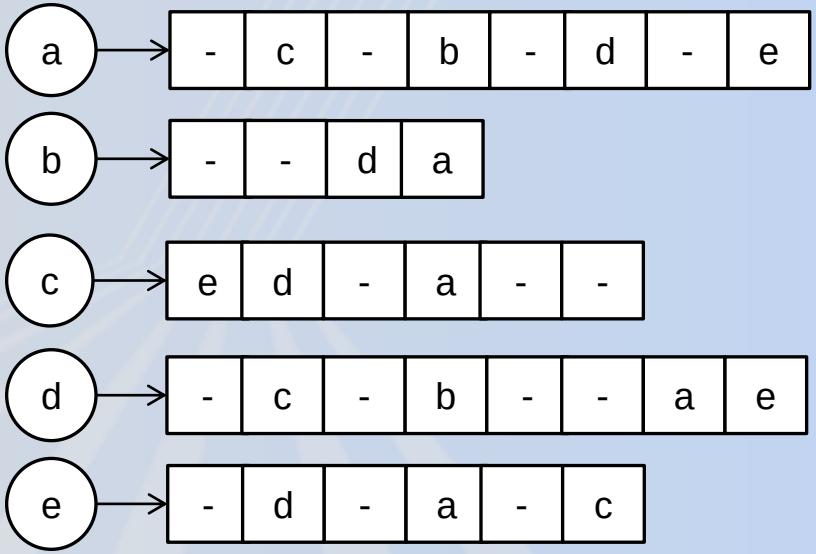


	a	b	c	d	e
a	-	1	1	1	1
b	1	-	0	1	0
c	1	0	-	1	1
d	1	1	1	-	1
e	1	0	1	1	-

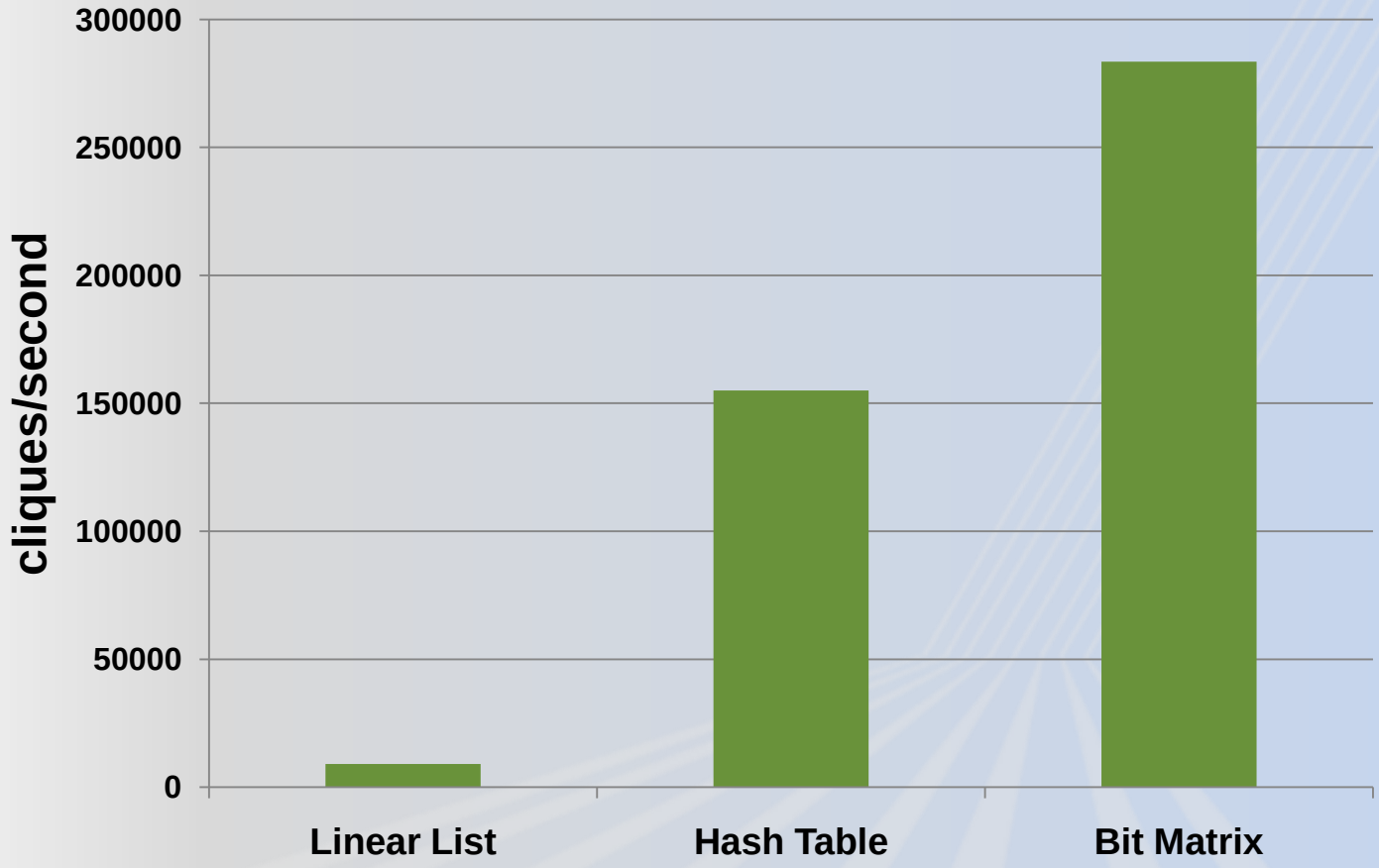
Adjacency Test – Hash Table

- Adjacency hash table has a fast, constant time lookup
  - But not as fast as bit matrix
- Memory requirement is cN (2N in this example)
- Data structure is a sparse linear list
- But access is through direct through key hashing

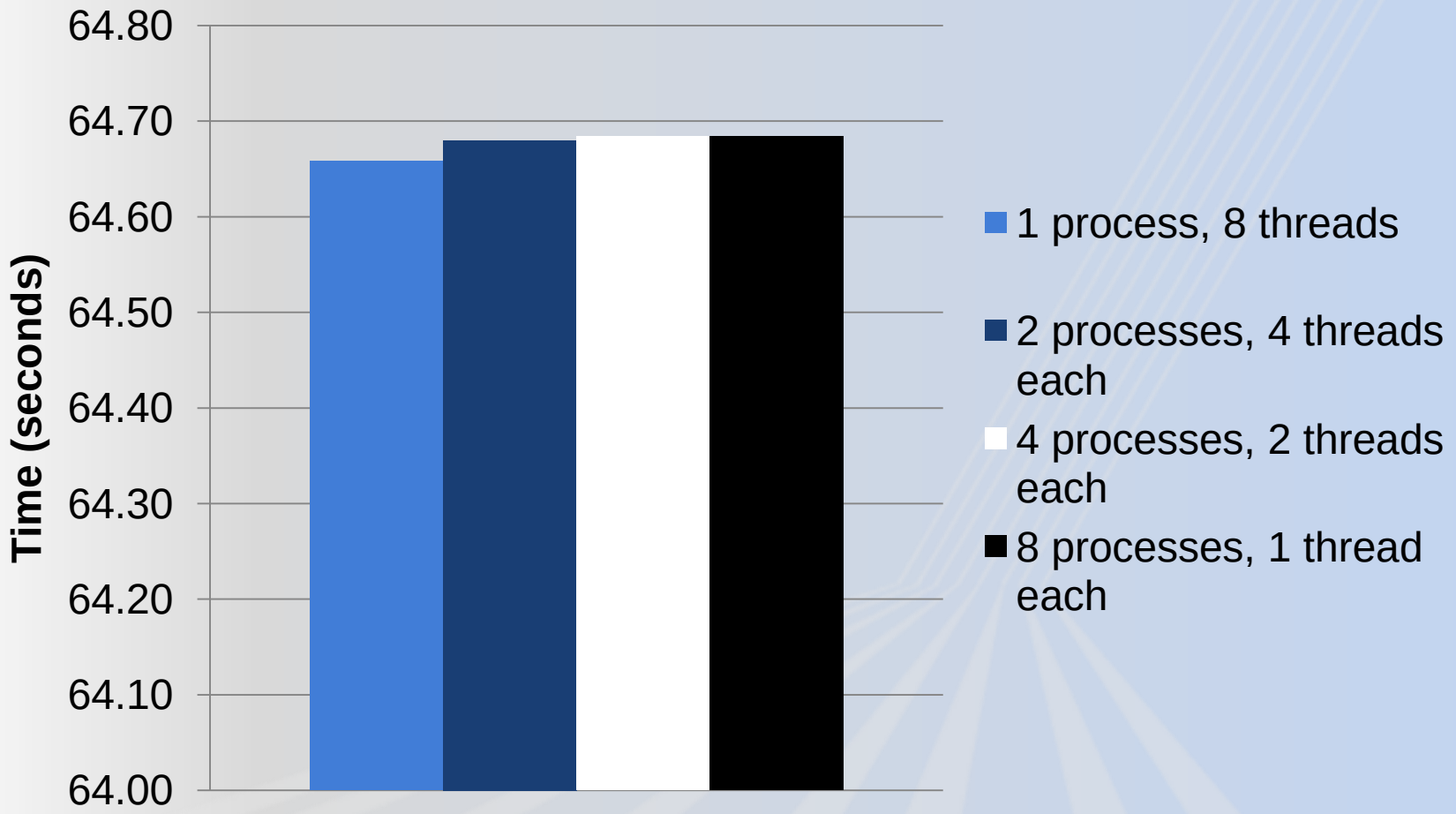




Adjacency Test Performance Comparison

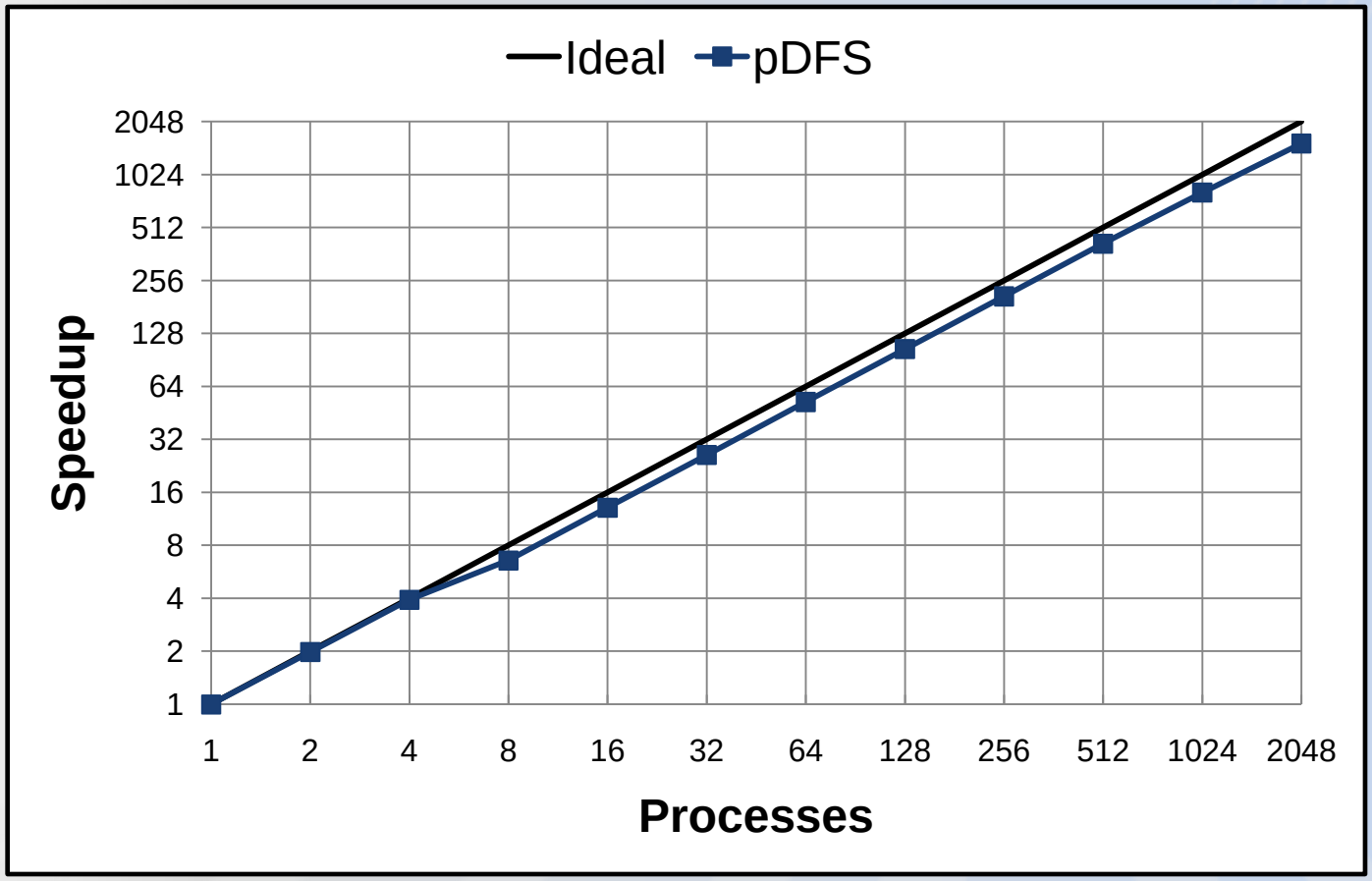


SMP Versus DMP Programming



Parallel Scaling on Cray XT4 quad core

- At 2048 processes, compute time is 2.1 seconds
  - Overhead due to message passing is 0.43 seconds
  - Graph contains 3472 vertices, found 2.6 billion maximal cliques



Conclusion

- Explicit decomposition at the thread level enabled easier implementation of MPI
  - Independent work already identified
  - Compact representation of units of work
- Additional work
  - Improved load balancing by grouping processes
  - Parallel I/O optimization

Conclusion

- Research group members
  - Nagiza F. Samatova, *North Carolina State University and Oak Ridge National Laboratory*
  - Matthew Schmidt, *North Carolina State University and Oak Ridge National Laboratory*
  - Byung-Hoon Park, *Oak Ridge National Laboratory*
  - Kevin Thomas, *Cray Inc.*
- Thank you!
- Questions?