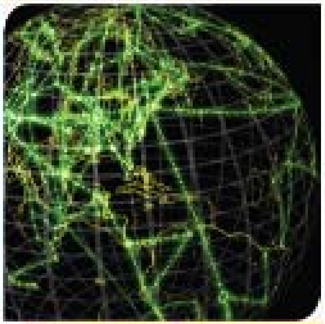
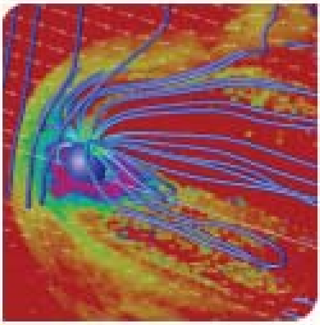
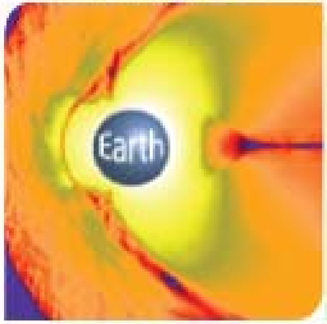
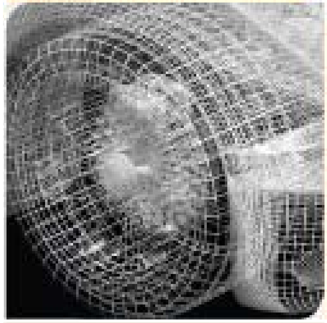


Petascale Computing for Large-Scale Graph Problems

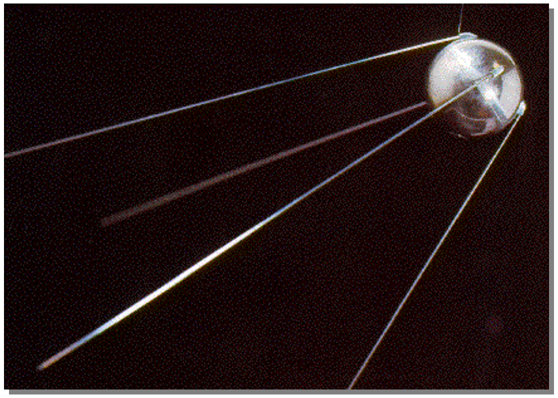
David A. Bader



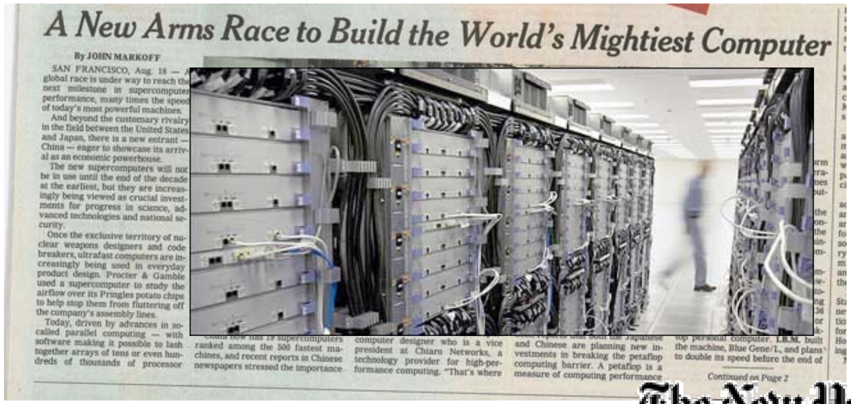
Georgia Tech College of Computing

Computational Science and Engineering

A global race is under way …



Sputnik (1957)



A New Arms Race to Build the World's Mightiest Computer

By JOHN MARKOFF

SAN FRANCISCO, Aug. 18 — A global race is under way to reach the next milestone in supercomputer performance, many times the speed of today's most powerful machines.

And beyond the customary rivalry in the field between the United States and Japan, there is a new entrant — China — eager to showcase its arrival as an economic powerhouse.

The new supercomputers will not be in use until the end of the decade at the earliest, but they are increasingly being viewed as crucial investments for progress in science, advanced technologies and national security.

Once the exclusive territory of nuclear weapons designers and code breakers, ultrafast computers are increasingly being used in everyday product design. Procter & Gamble used a supercomputer to study the airflow over its Pringles potato chips to help stop them from fluttering off the company's assembly lines.

Today, driven by advances in so-called parallel computing — with software making it possible to lash together arrays of tens or even hundreds of thousands of processor

China now has 19 supercomputers ranked among the 500 fastest machines, and recent reports in Chinese newspapers stressed the importance

computer designer who is a vice president at Chiaro Networks, a technology provider for high-performance computing. "That's where

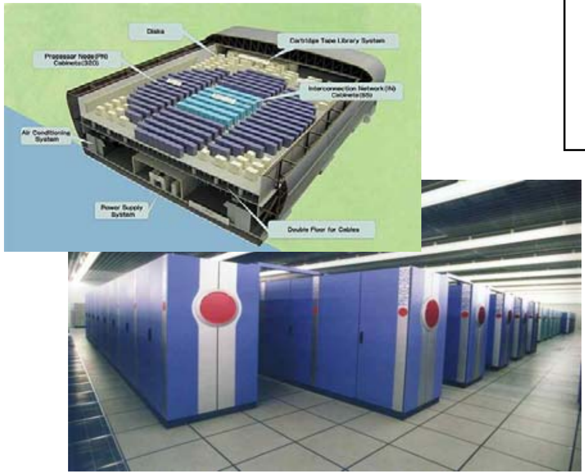
and Chinese are planning new investments in breaking the petaflop computing barrier. A petaflop is a measure of computing performance

top personal computer. I.B.M. built the machine, Blue Gene/L, and plans to double its speed before the end of

Continued on Page 2

The New York Times

China joins U.S. and Japan in global race to build the fastest computer
- John Markoff, Aug 19, 2005


Disks
Cartridge Tape Library System
Processor Node (PN) Cabinets (320)
Interconnection Network (IN) Cabinets (65)
Air Conditioning System
Power Supply System
Double Floor for Cables

Japanese Earth Simulator (2002)

DARPA HPCS

NSF Leadership-Class System Acquisition - Creating a Petascale Computing Environment for Science and Engineering

U.S. Petascale (2008-2010)

Aiming for Petascale at Georgia Tech!

- **6th** ranked academic institution in the most recent June 2006 **Top100** List of most capable supercomputers in the world

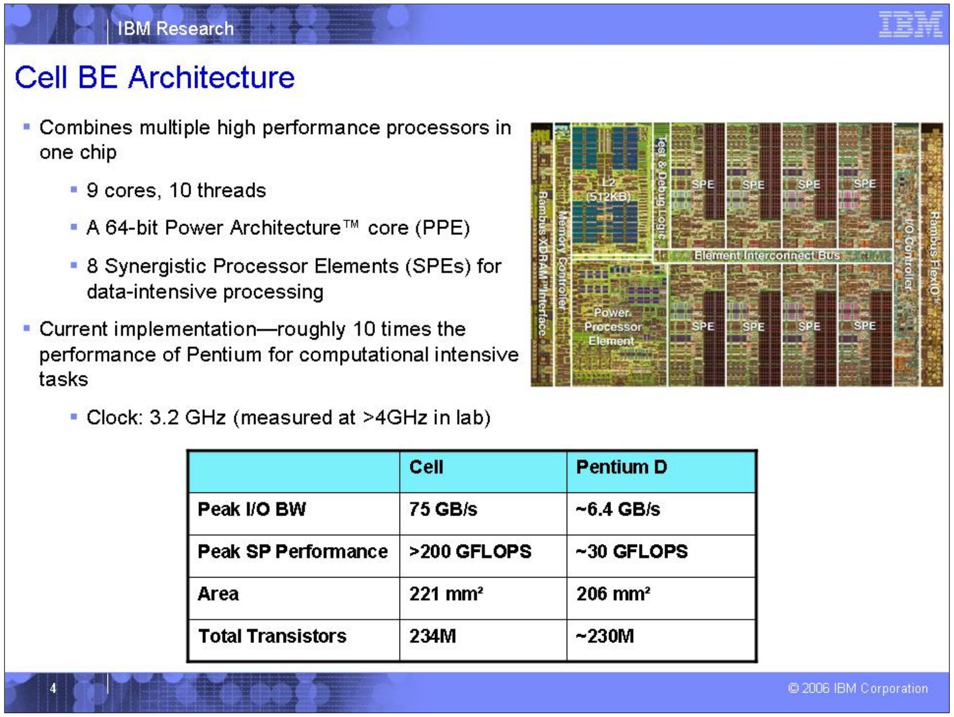
TOP500 SUPERCOMPUTER SITES

- Georgia Tech's high-end computing resources include approximately 7,000 processors in 35 clusters along with about 100 processors across several SMP systems. Recent HPC system acquisitions include:
  - IBM Skolnick System Biology Center system: a 4020-processor IBM eServer BladeCenter with 1,005 blades of 2x2 Opteron cores/blade
  - Dell PowerEdge 1850 system: a 512-node supercomputing cluster with Intel Xeons and InfiniBand interconnect.

- **Klaus Advanced Computing Building** (most advanced the world!) opens 26 October 2006

Created a **Computational Science & Engineering** Department in Fall 2005.

HPC for Multicore Processors



IBM Research | IBM

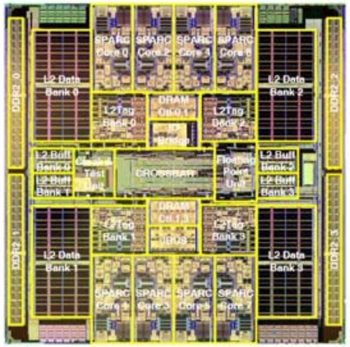
Cell BE Architecture

- Combines multiple high performance processors in one chip
  - 9 cores, 10 threads
  - A 64-bit Power Architecture™ core (PPE)
  - 8 Synergistic Processor Elements (SPEs) for data-intensive processing
- Current implementation—roughly 10 times the performance of Pentium for computational intensive tasks
  - Clock: 3.2 GHz (measured at >4GHz in lab)

	Cell	Pentium D
Peak I/O BW	75 GB/s	~6.4 GB/s
Peak SP Performance	>200 GFLOPS	~30 GFLOPS
Area	221 mm²	206 mm²
Total Transistors	234M	~230M

4 | © 2006 IBM Corporation

- Sun Fire T2000 Servers
- UltraSPARC T1 “Niagara” processor
- “the highest-throughput and most **eco-responsible** processor ever created”®

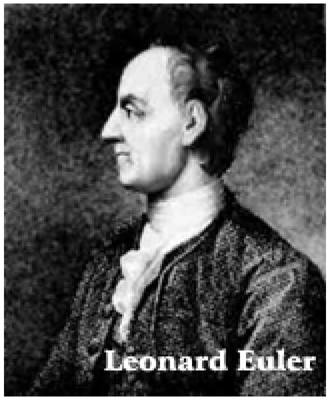


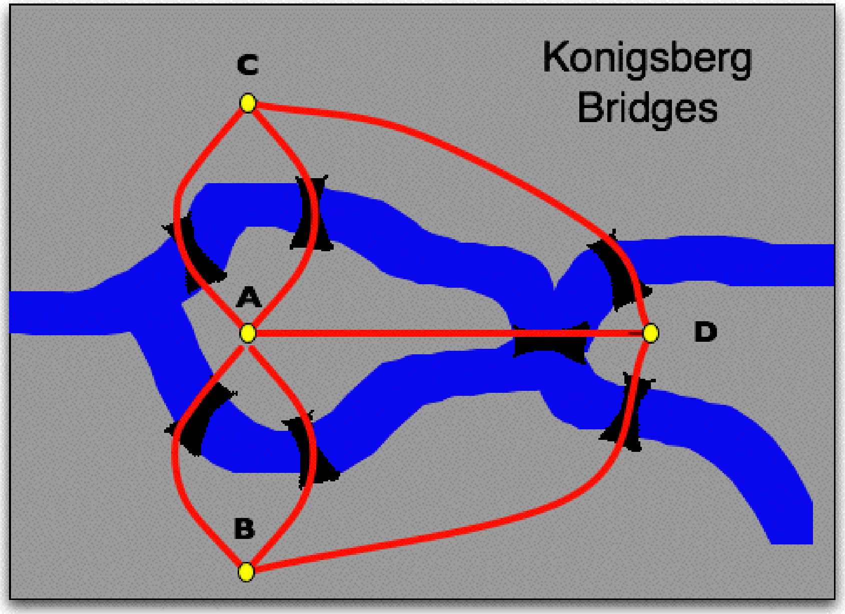
Georgia Tech multicore research includes:

- IBM Shared University Research Award for Cell processors
- Sun Academic Equipment Grant for Sun Fire T2000 servers

Germany: The birthplace of graph theory

- In Konigsberg, Germany, a river ran through the city such that in its center was an island, and after passing the island, the river broke into two parts. Seven bridges were built so that the people of the city could get from one part to another.
- The people wondered whether or not one could walk around the city in a way that would involve crossing each bridge exactly once.
- Leondard Euler, circa 1735

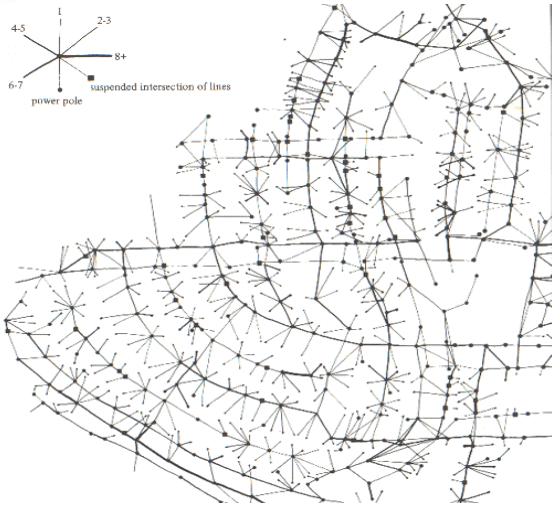




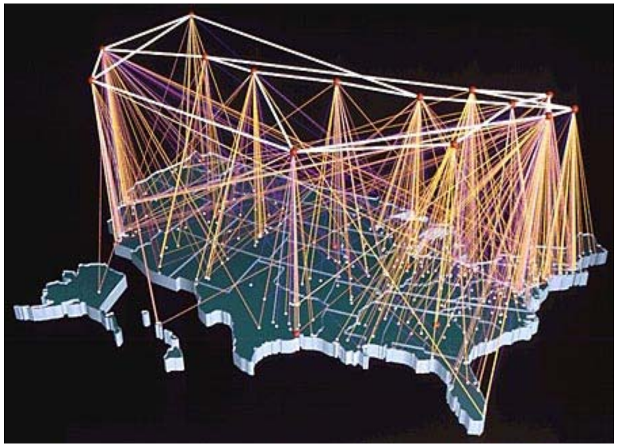
Source: The Math Forum

Graph problems arise from a variety of sources

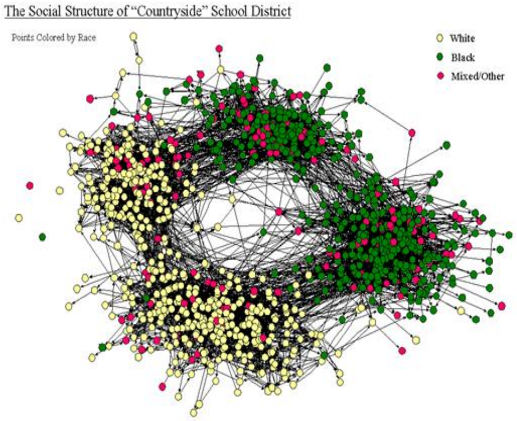
Power Distribution Networks



Internet backbone

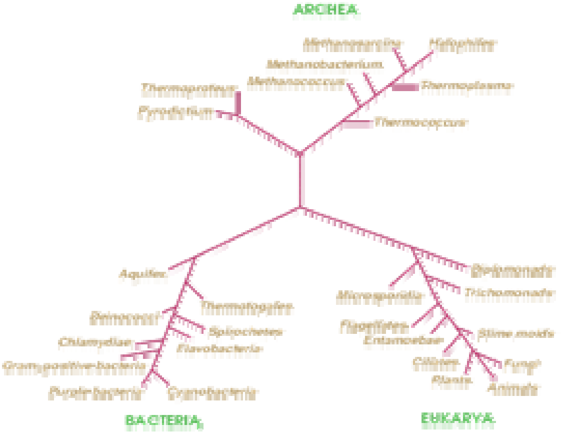


Social Networks

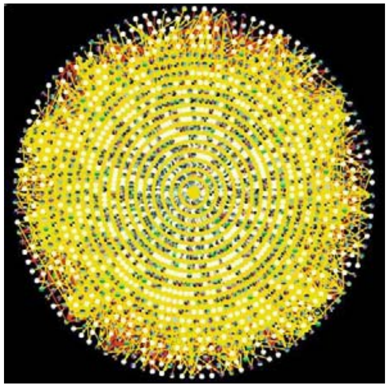


Graphs are everywhere!

Ground Transportation



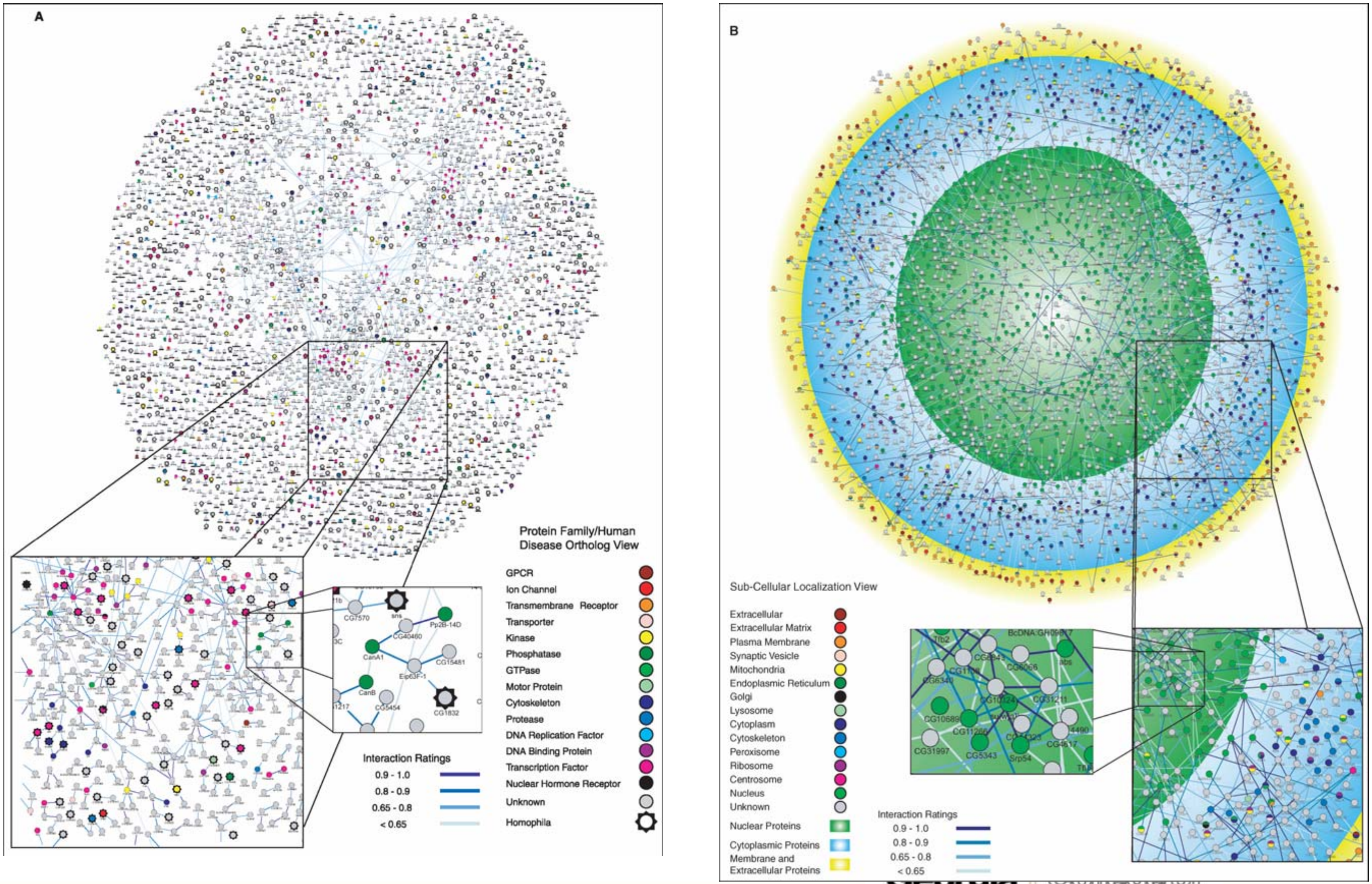
Tree of Life



Protein-interaction networks

Sources: C. Faloutsos talk [IPAM 05]

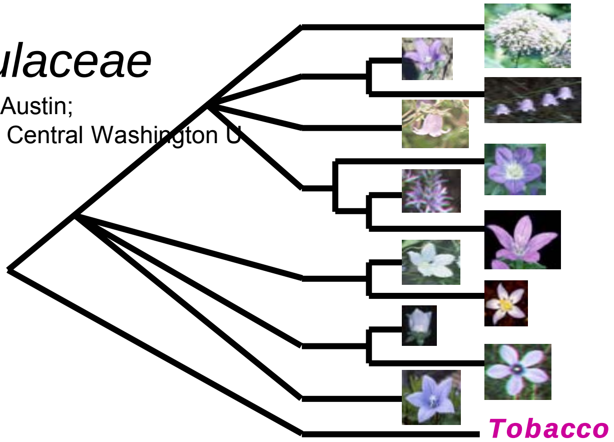
Giot L, **Bader JS**, …, Rothberg JM,
A protein interaction map of *Drosophila melanogaster*
Science 302: 1727-1736, 2003.

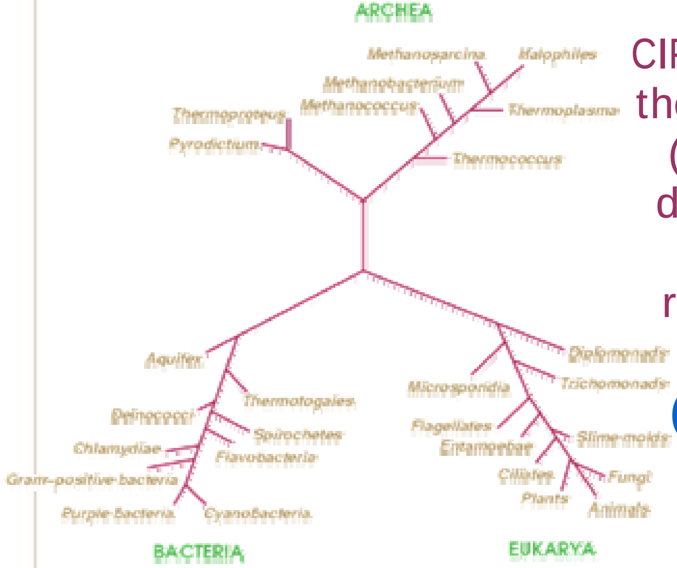


Computational Phylogeny GRAPPA

Campanulaceae

- Bob Jansen, UT-Austin;
- Linda Raubeson, Central Washington U





The Tree of Life

CIPRES aims to establish the cyber infrastructure (platform, software, database) required to attempt a reconstruction of the Tree of Life

(10-100M organisms)

- Genome Rearrangements Analysis under Parsimony and other Phylogenetic Algorithm
  - Freely-available, open-source, GNU GPL
  - already used by other computational phylogeny groups, Caprara, Pevzner, LANL, FBI, Smithsonian Institute, Aventis, GlaxoSmithKline, PharmCos.
- Gene-order Phylogeny Reconstruction
  - Breakpoint Median
  - Inversion Median
- over one-billion fold speedup from previous codes
- Parallelism scales linearly with the number of processors

Signaling networks: activating potentials through space and time

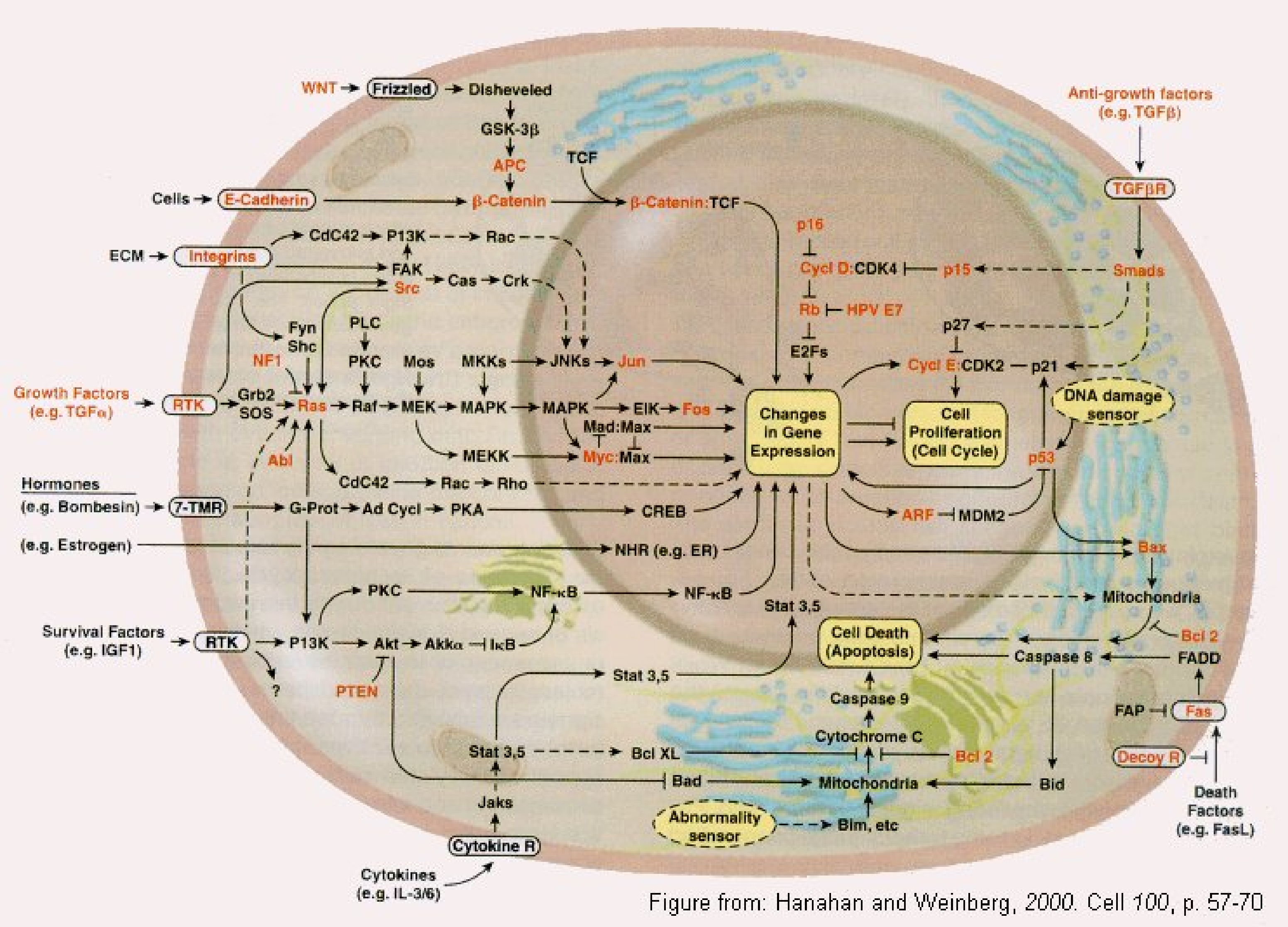
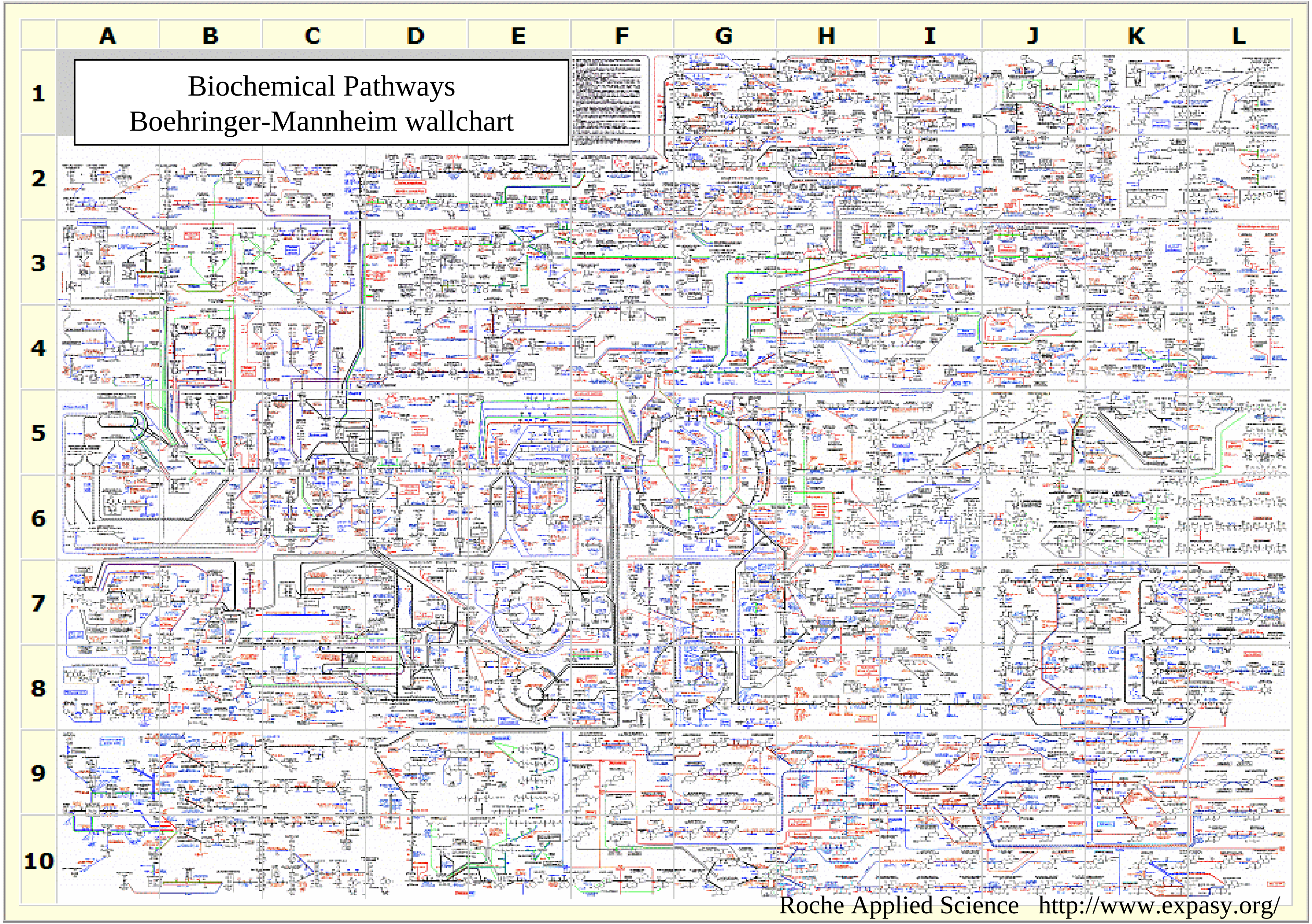


Figure from: Hanahan and Weinberg, *2000*. Cell *100*, p. 57-70

Biochemical Pathways
Boehringer-Mannheim wallchart



Roche Applied Science http://www.expasy.org/

Homeland Security: Terrorist Networks

- Certain activities are often suspicious not because of the characteristics of a single actor, but because of the interactions among a group of actors.
- Interactions are modeled through a graph abstraction where the entities are represented by vertices, and their interactions are the directed edges in the graph.

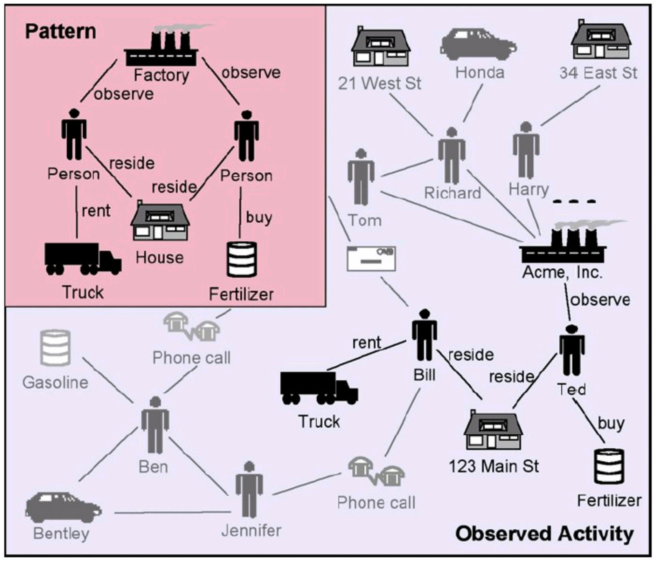


Figure Credit: *Graph-based technologies for intelligence analysis,*T. Coffman, S. Greenblatt, S. Marcus, Commun. ACM, 47(3):45-47, 2004.

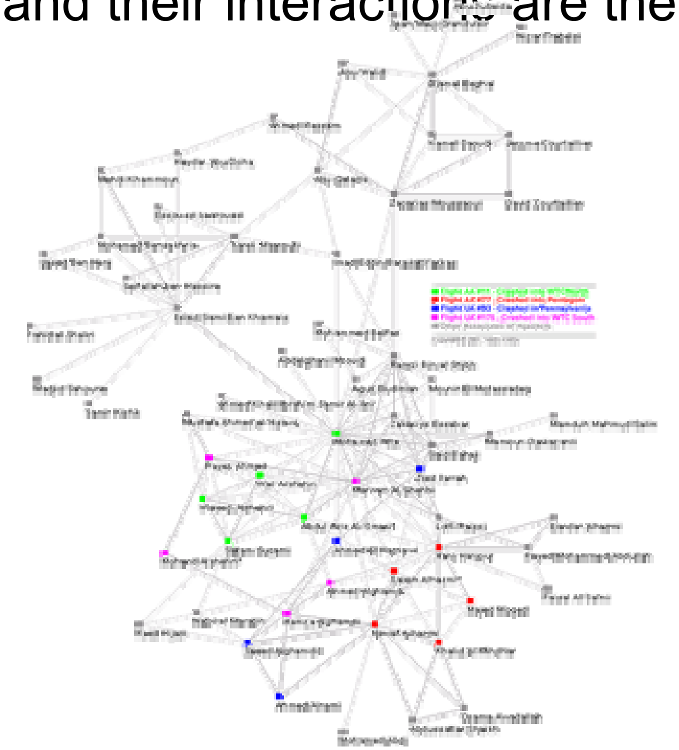
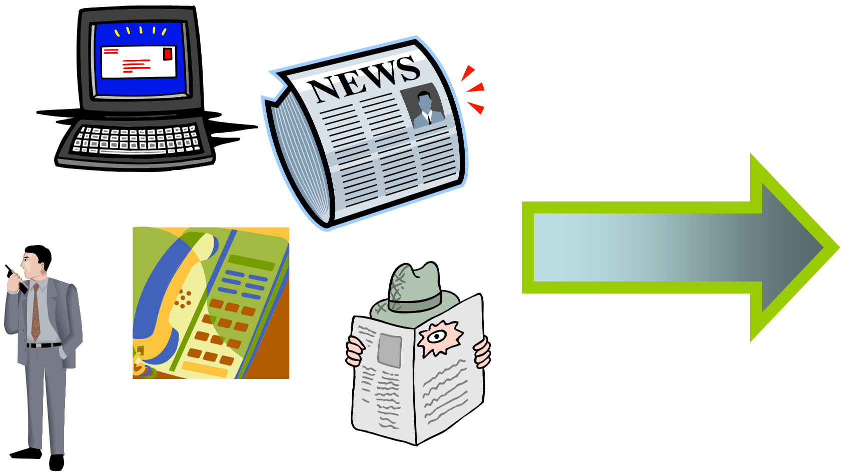
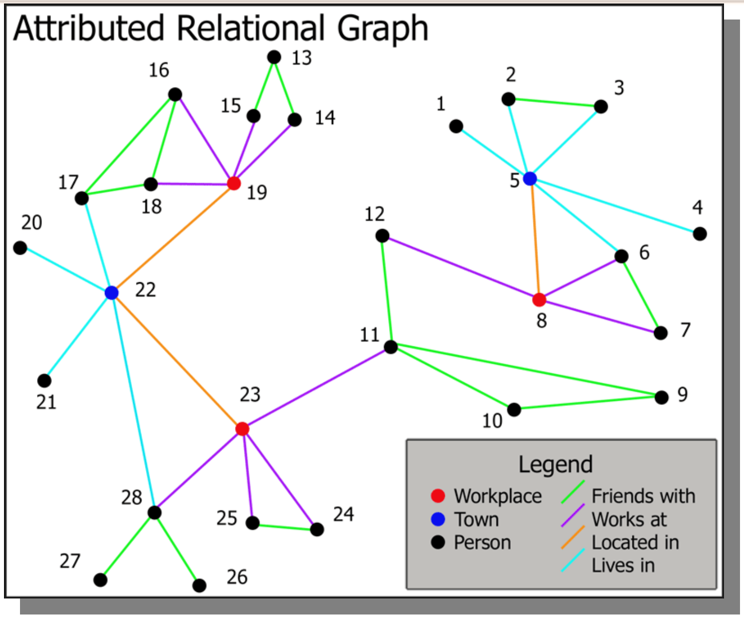


Figure Credit: *Uncloaking Terrorist Networks*, V.E. Krebs, First Monday, 7(4), April 2002.

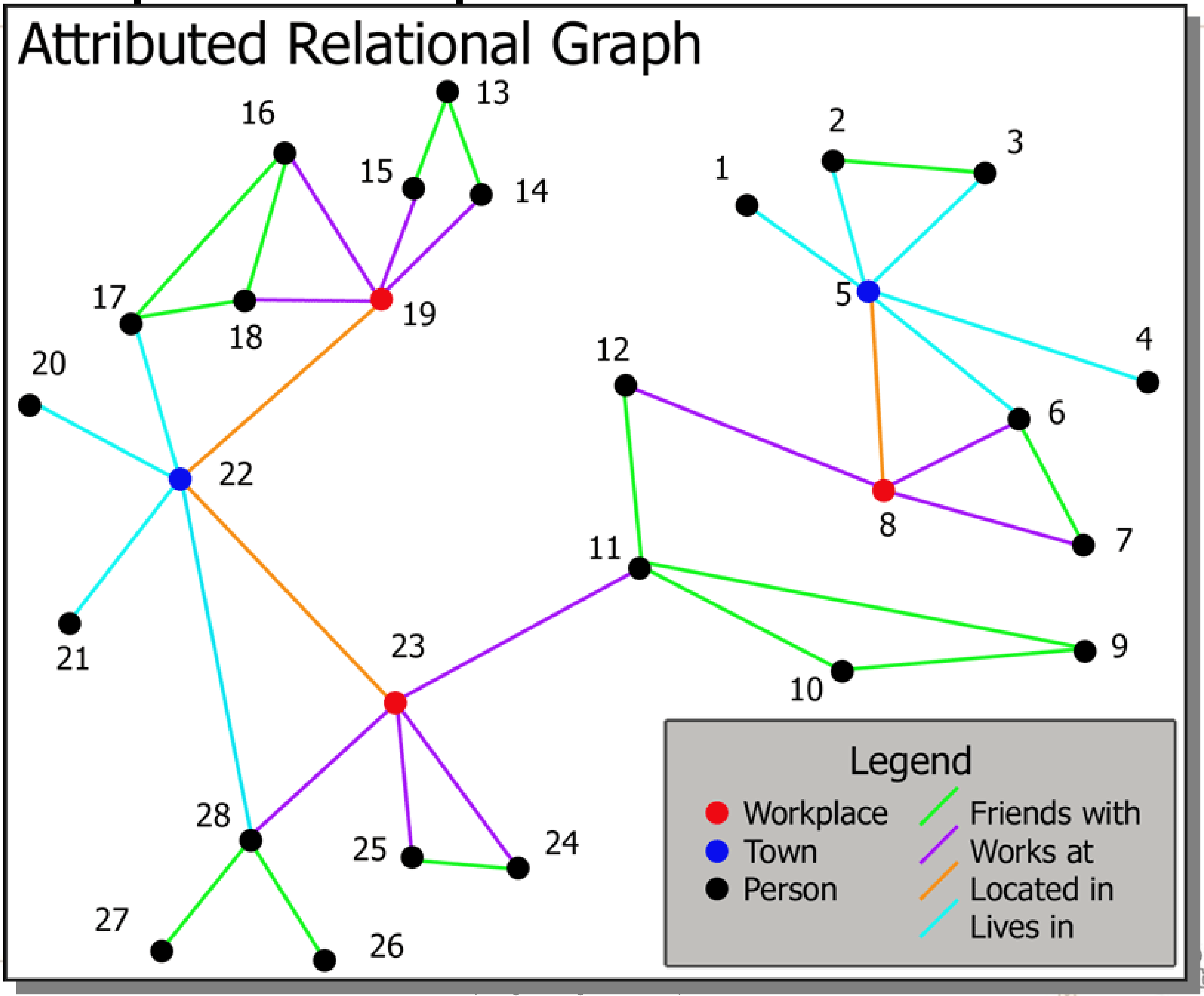
Information Overload





- *Challenge*: Piecing the data together and extracting critical, relevant information in a timely manner
- Semantic Graphs (or Attributed Relational Graphs) are one way to integrate data from disparate sources
  - Vertices represent people, places, locations, events, etc.
  - Edges represent the relationships between the vertices
  - Semantic graph encodes web of relationships

Simple Example



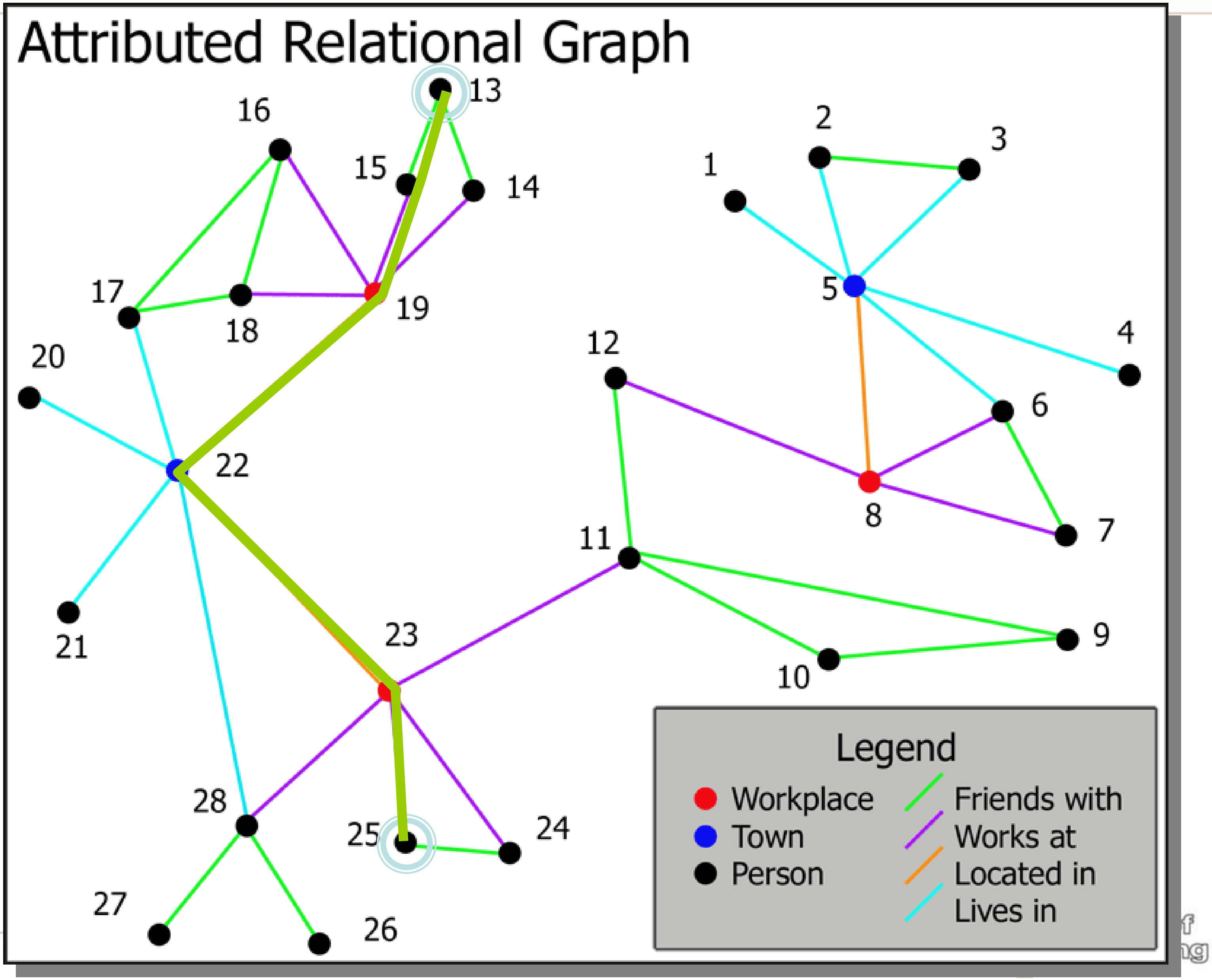
Advantages of Semantic Graphs

- Much smaller than raw data. Can fit in memory of large computer
  - Fast response to queries
  - Pre-join of database

- Combine data from different sources and of different types

- Some common intelligence and law enforcement queries are naturally posed on graphs
  - Particularly for the terrorist threat

Query Example I: Short Paths



Attributed Relational Graph

Query Example II: Motif Finding

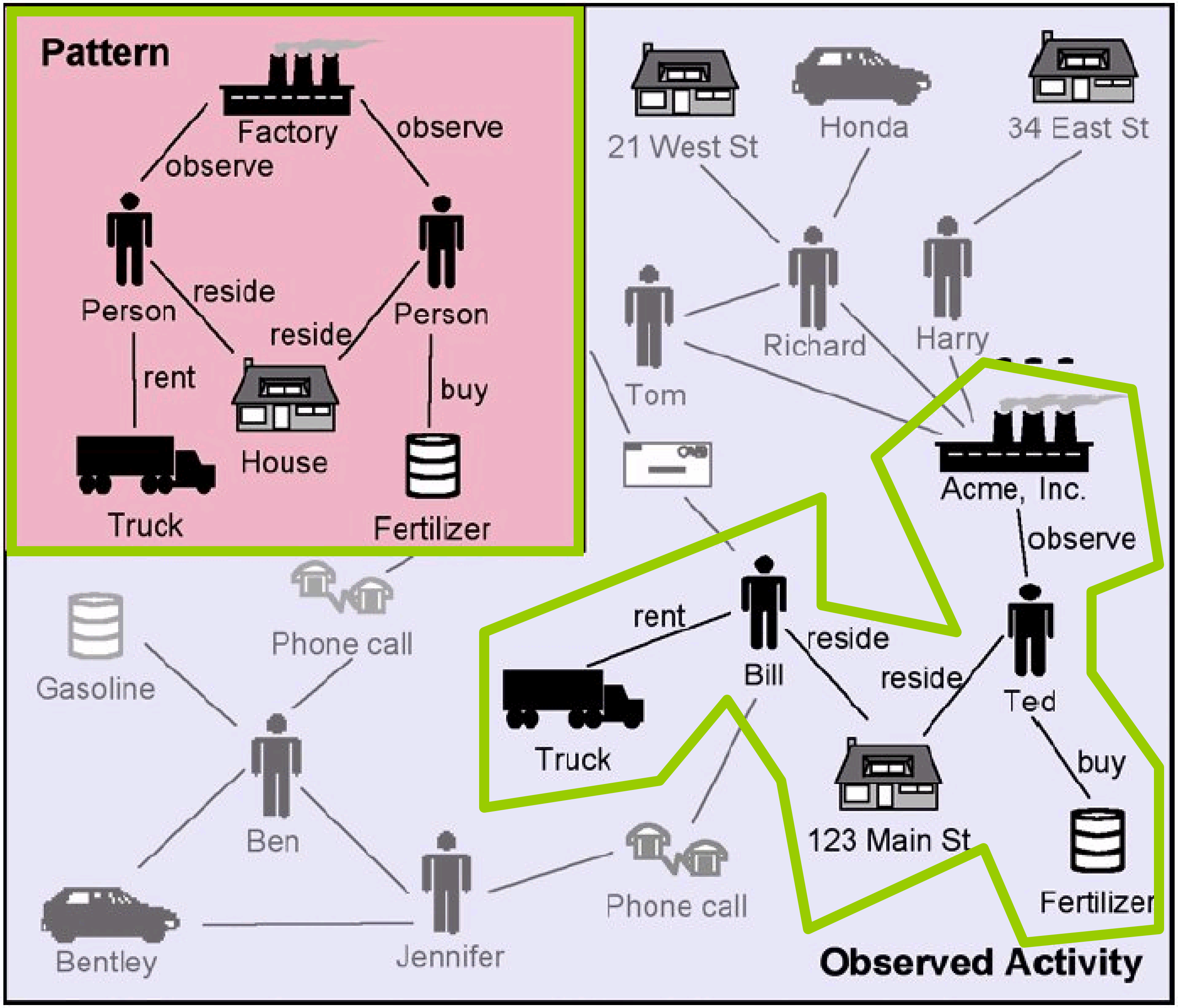
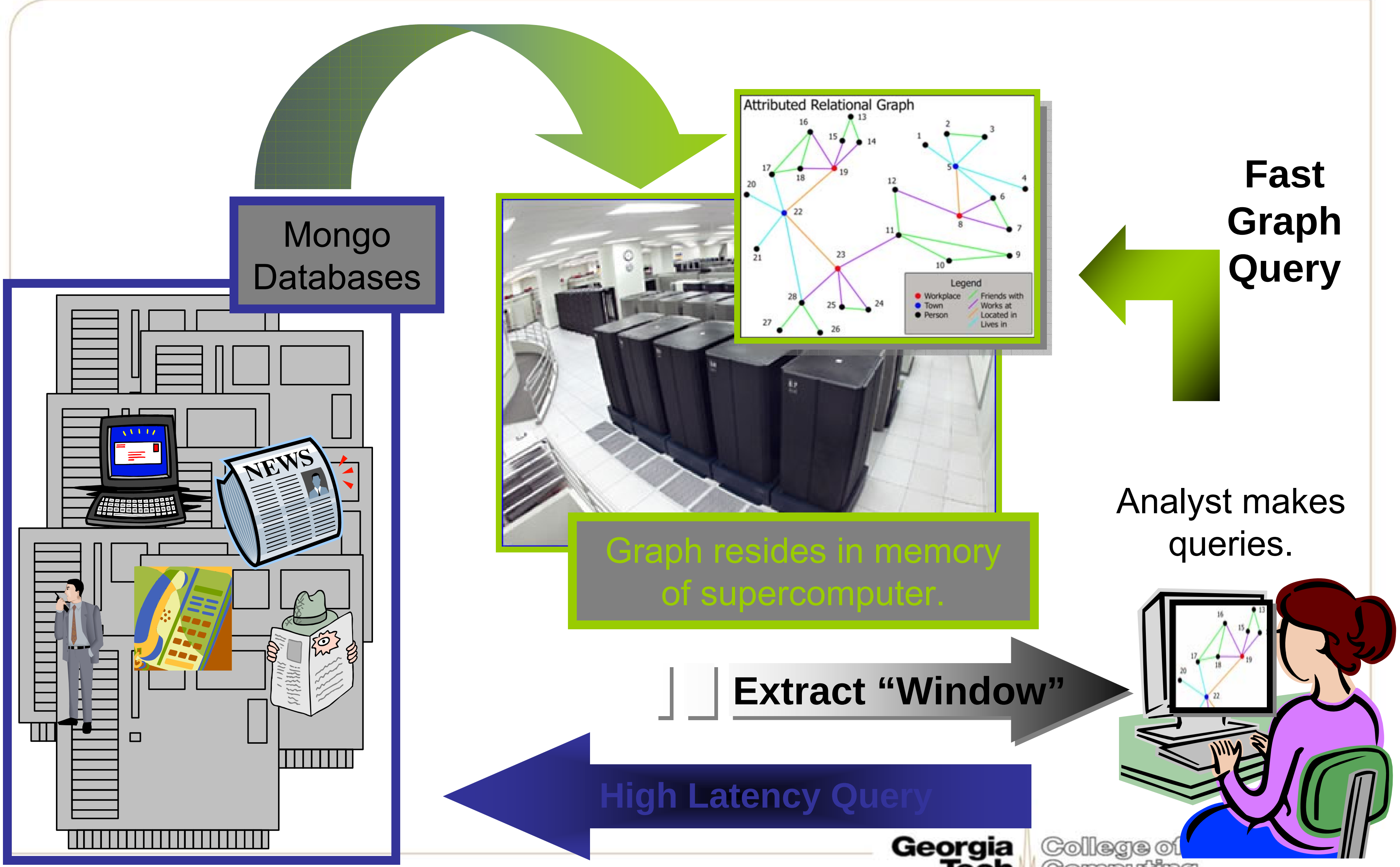


Image Source:
T. Coffman,
S. Greenblatt,
S. Marcus,
Graph-based technologies for intelligence analysis,
CACM, 47
(3, March 2004):
pp 45-47

The Big Picture



Mongo Databases

Graph resides in memory of supercomputer.

Fast Graph Query

Analyst makes queries.

Extract “Window”

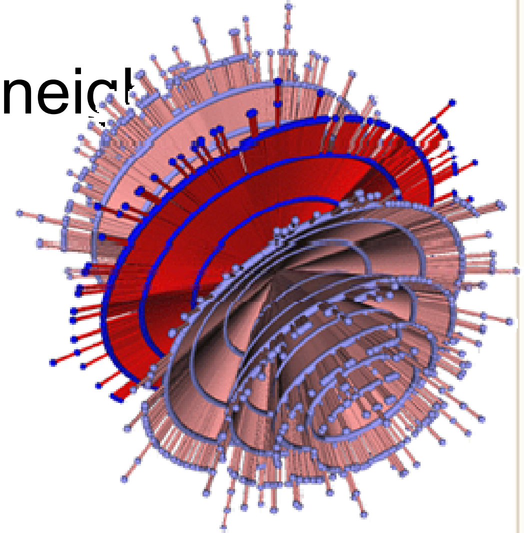
High Latency Query

Graph algorithms

- Driving applications are not traditional HPC:
  - health care, proteomics, security, informatics, …
- Fundamental abstraction
  - Standard introductory material covered in a computer science course on data structures and algorithms, but...
- Why have there been so few (or no) efficient distributed memory implementations of even the simplest algorithm for sparse, arbitrary graphs?

Informatics Graphs are Tough

- Very different from graphs in scientific computing!
  - Graphs can be enormous
  - Power-law distribution of the number of neig
  - Small world property – no long paths
  - Very limited locality, not partitionable
  - Highly unstructured
  - Edges and vertices have types



Six degrees of Kevin Bacon
Source: Seokhee Hong

- Experience in scientific computing applications provides only limited insight.

Architectural Challenges

- Runtime is dominated by latency
  - Random accesses to global address space
  - Perhaps many at once
- 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

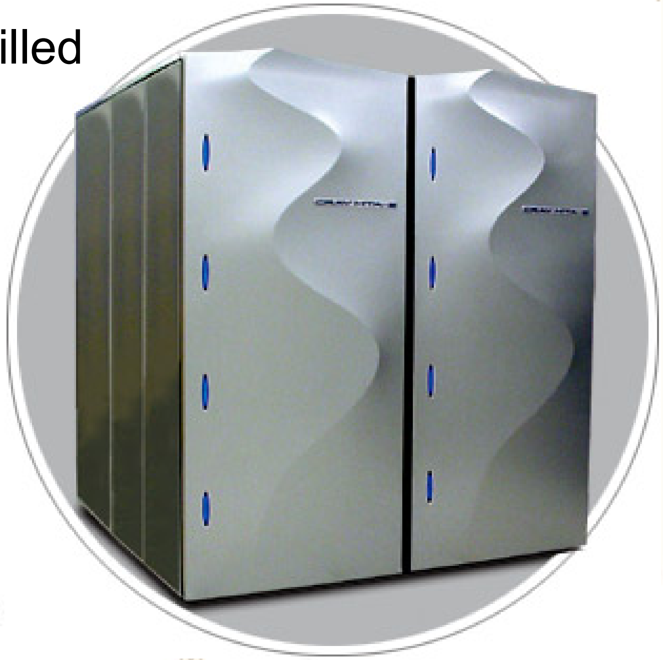
Desirable Architectural Features

- Low latency / high bandwidth
  - For small messages!
- Latency tolerant
- Light-weight synchronization mechanisms
- Global address space
  - No graph partitioning required
  - Avoid memory-consuming profusion of ghost-nodes
  - No local/global numbering conversions

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

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 does not 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 5 years 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, Erdős-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: 0.7 seconds on one processor, 0.09 seconds on 10 procs
- Conclusion: 4 MTA-2 procs = 32K BlueGene/L procs

But Speed Isn’t Everything

- Unlike MTA code, MPI code limited to Erdős-Renyi graphs
  - Can’t support power-law graphs; pervasive in informatics
- MPI code is 3 times larger than MTA-2 code
  - Took considerably longer to develop
- MPI code can only solve this very special problem
  - MTA code is part of general and flexible infrastructure
- MTA easily supports multiple, simultaneous users
- But … MPI code runs everywhere
  - MTA code runs only on MTA/Eldorado and on serial machines

Lessons & Challenges

- Massively multithreaded architectures:
  - Are highest performing for graph algorithms
  - Are boutique and rare
  - Have specialized programming model

- Distributed memory machines:
  - Are a very poor fit for graph informatics applications
  - Are commodity and ubiquitous
  - MPI provides extremely portable programming model

What is easy on the MTA-2

- No need to place data near computation
- No performance concerns with modifying shared data
- Can access data in any order
- Using indirection or linked data-structures, and pointer-chasing
- No need to partition program into independent, balanced computations
- No need to use adaptive or dynamic computations for load balancing
- No laborious task needed to minimizing synchronization operations

Source: Cray, Inc.

Our development on MTA-2 includes

- Data Structures
  - Treaps (randomized binary trees)
    - Fast set operations – parallel algorithms run in optimal O(*m* log(*n*/*m*)) work and O(log *n*) expected time
    - Used for representing neighbors of high-degree nodes in the graph
    - Used for compacting edge sets in BFS, MST algorithms
  - Van Emde Boas trees
    - Recursive data structure, set operations
  - Fibonacci Heaps and Pairing Heaps
    - Dijkstra-based Shortest paths implementations

- List ranking and connected components.
  - List ranking runs 40 times faster
  - Connected components runs 6 times faster
  - on 220MHz Cray MTA-2 processors compared with a commodity 400MHz Sun SMP.
  - [Bader, Cong, Feo; ICPP 2005]

- Graph theory applications
  - Parallel breadth-first search; approximate clique extraction; DARPA SSCA2 [Bader, Madduri, Feo, in progress]
  - st-connectivity [Bader, Madduri; ICPP 2006]
  - Betweenness Centrality [Bader, Madduri; ICPP 2006]

Two Case Studies

- Breadth-First Search (BFS)
- Betweenness Centrality

Case Study 1: Breadth-First Search (BFS)

- Sequential BFS: $O(m+n)$ using a FIFO queue
- Recent algorithms and implementations for handling large-scale graphs:
  - graph partitioning [Yoo et. al. 2005]
  - external memory [Meyer et. al. 2006]
- Our design is a fine-grained algorithm, suited for multithreaded architectures
  - All vertices at a given *level* in the graph can be processed simultaneously, instead of just picking the vertex at the head of the queue
  - The adjacencies of each vertex can be inspected in parallel

Multithreaded BFS

Input : $G(V, E)$, source vertex s

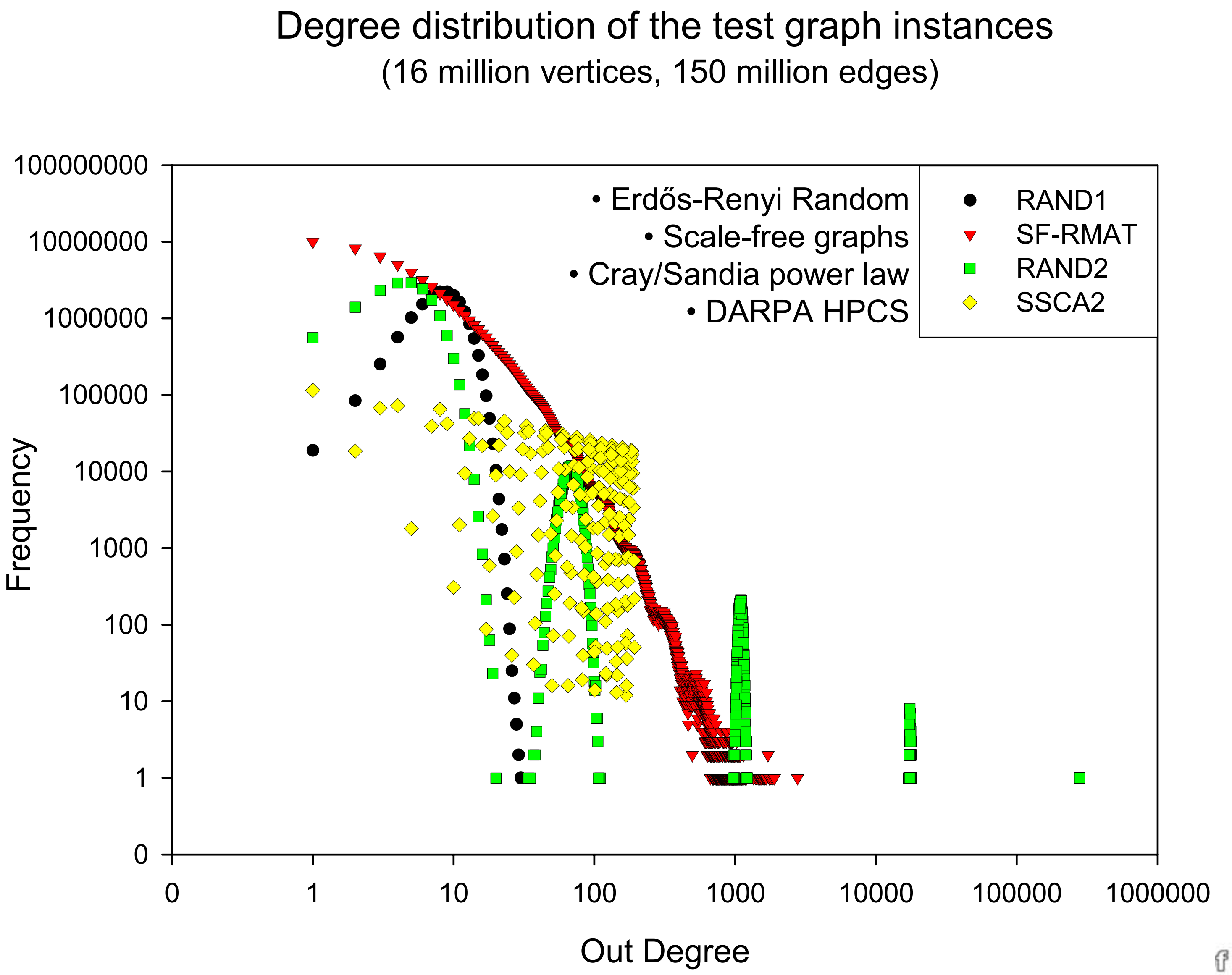
Output : Array $d[1..n]$ with $d[v]$ holding the length
of the shortest path from s to $v \in V$,
assuming unit - weight edges

```
for all $v \in V$ in parallel do
    $d[v] \leftarrow -1$;
$d[s] \leftarrow 0$;
$Q \leftarrow \phi$;
Enqueue $s \rightarrow Q$;
while $Q \neq \phi$ do
    for all $u \in Q$ in parallel do
        Delete $u \leftarrow Q$;
        for each $v$ adjacent to $u$ in parallel do
            if $d[v] = -1$ then
                $d[v] \leftarrow d[u] + 1$;
                **Enqueue** $v \rightarrow Q$;
```

MTA-2 BFS Implementation details

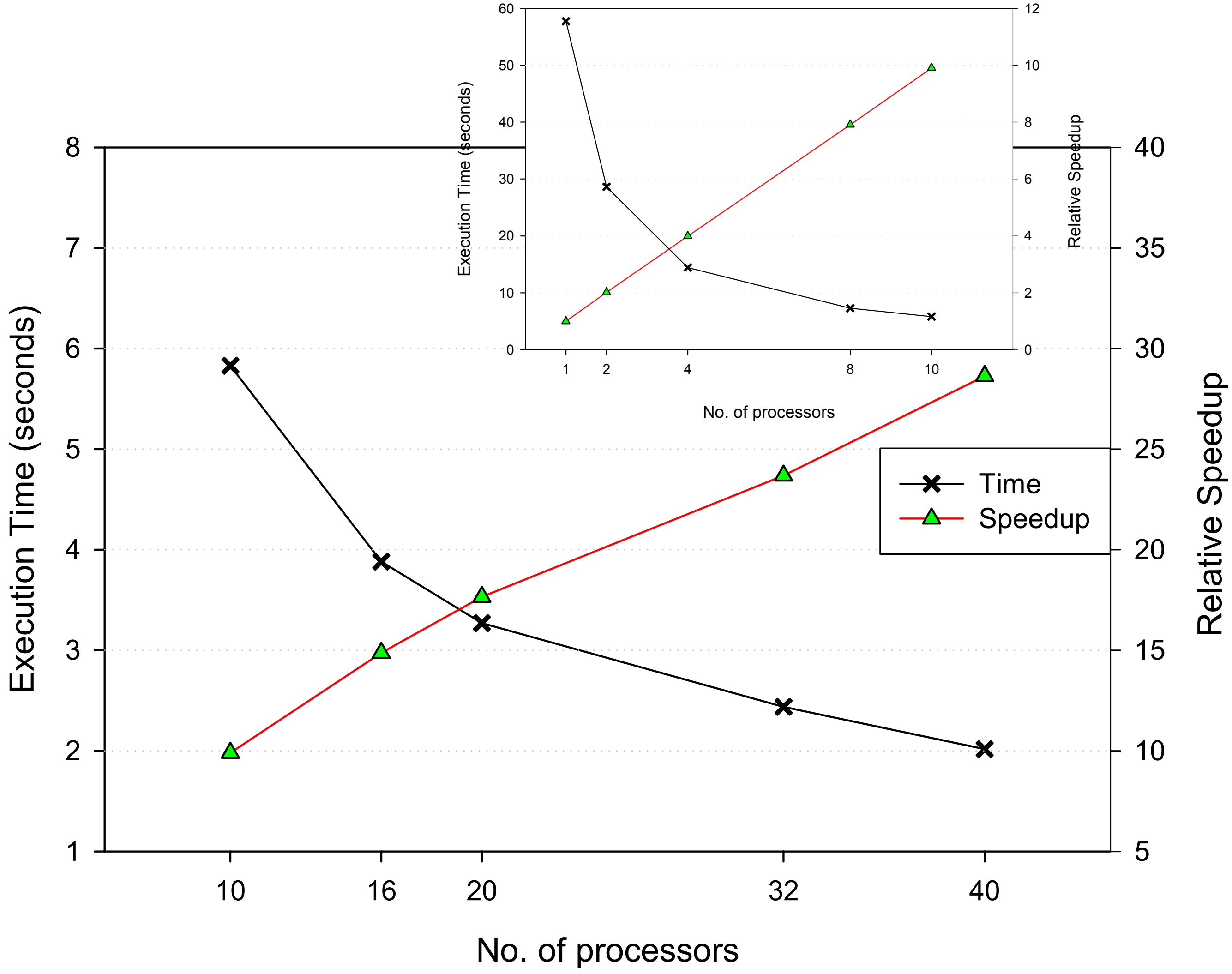
- Only requires a simple shared queue
  - efficient due to the low-overhead synchronization primitives on MTA-2
- We easily exploit nested parallelism in the algorithm
  - MTA compiler automatically collapses the inner loop (visiting adjacencies)
- Unbalanced degree distributions (scale free graphs) do not pose a problem
  - Loop iterations are dynamically scheduled

Input Graphs for Testing on MTA-2

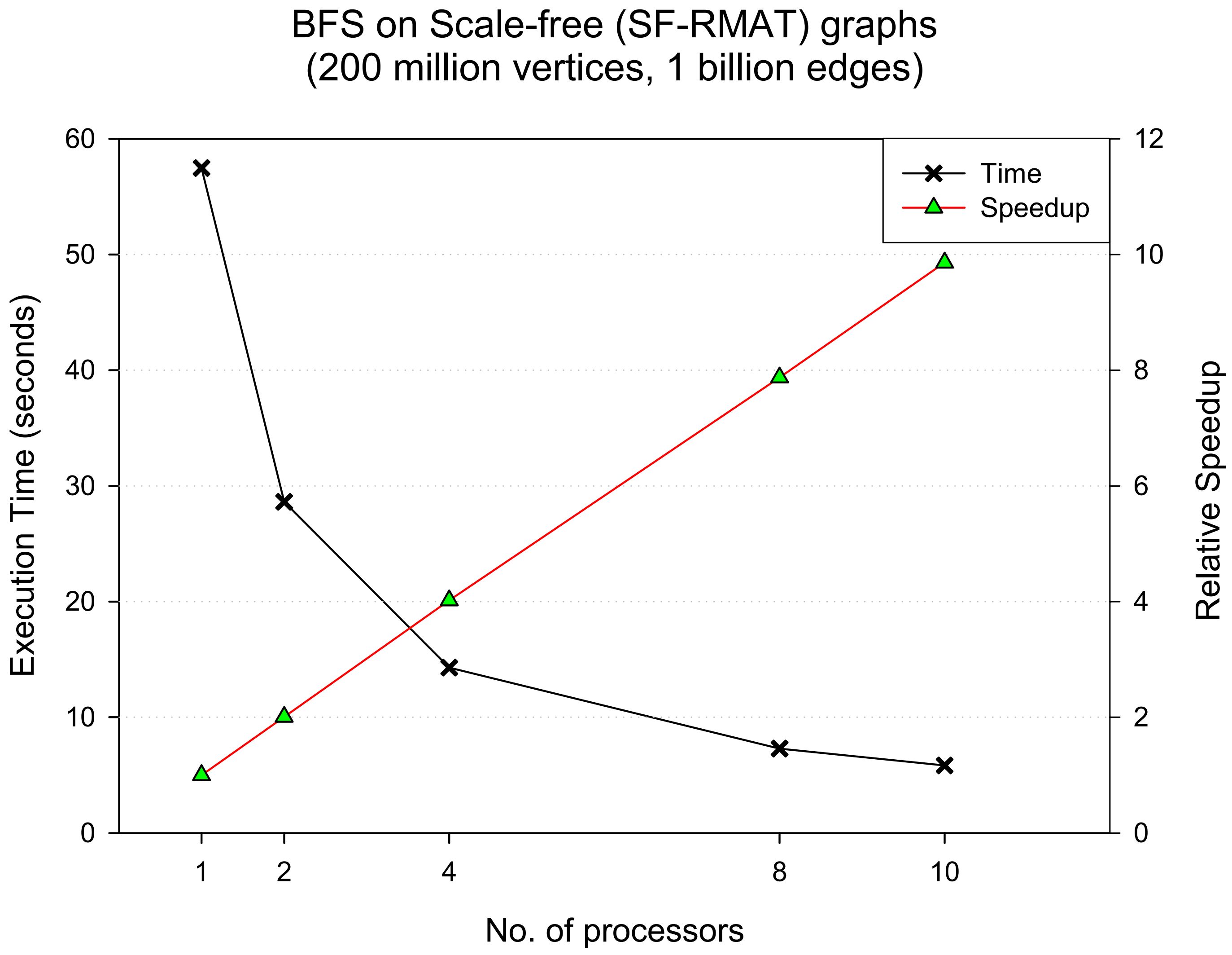


Scaling of BFS on RAND1

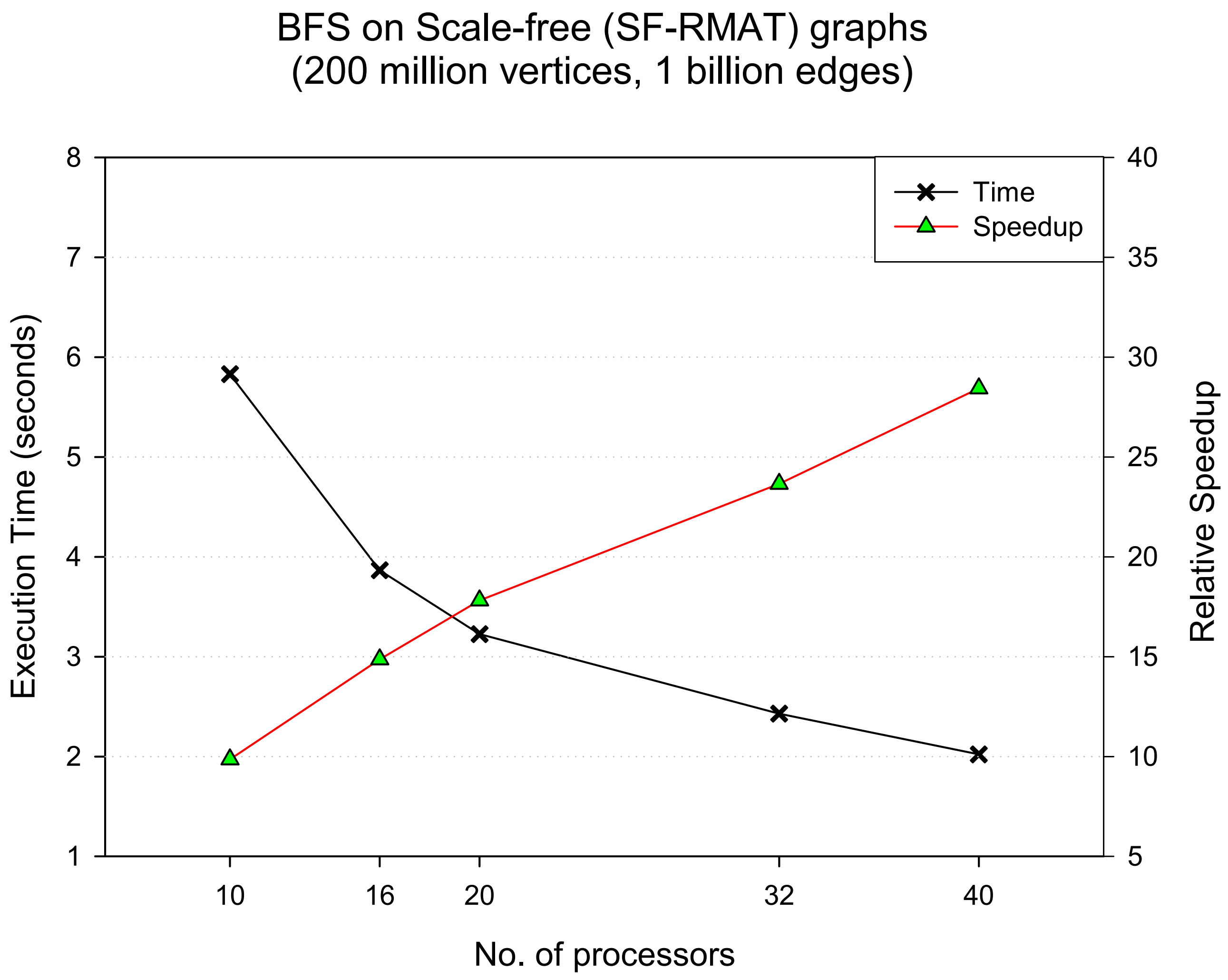
BFS on Random (RAND1) graphs
(200 million vertices, 1 billion edges)



Scaling of BFS on SF-RMAT (1 to 10 processors)



Scaling of BFS on SF-RMAT (10 to 40 processors)



Comparison

- Cray MTA-2: 40 processors
  - BFS, graph of 528M vertices, 2.1 Billion edges
  - Scale-free graph: 17.32 seconds
  - Random graph: 13.74 seconds
- [1] IBM BlueGene/L: 32K processors
  - BFS on a random Poisson graph of 3.2 Billion vertices, average degree of 10
  - 4.9 seconds
- [2] Parallel Boost Graph Library: performance on a 128–node cluster
  - BFS Random graph, 1M vertices, 15M edges
  - 1 processors: 40 seconds
  - 20 processors: 10 seconds
  - 70 processors: 3 seconds
  - 100 processors : 10 seconds

[1] Chow '04, Eliassi-Rad IPAM 2005 talk

[2] http://www.osl.iu.edu/research/pbgl/performance/

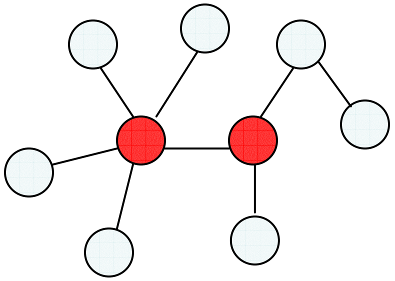
Case Study 2: Social Network Analysis

- Centrality metrics: Quantitative measures to capture the importance of a node/vertex/actor in a graph
  - Degree, Closeness, Stress, **Betweenness**
- Applications include:
  - Biological networks, protein-protein interactions
  - Sexual networks and AIDS
  - Identifying key actors in terrorist networks
  - Organizational behavior
  - Supply chain management
  - Transportation networks

Betweenness Centrality (BC)

- Key metric in social network analysis
  [Freeman ’77, Goh ’02, Newman ’03, Brandes ’03]

$$\boxed{BC(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}}$$



- σ_{st} -- No. of shortest paths between vertices *s* and *t*
- $\sigma_{st}(v)$ -- No. of shortest paths between vertices *s* and *t* passing through *v*

- Betweenness Centrality is compute-intensive

Our Contributions

- Design and implementation of the *first parallel algorithm* for evaluating Betweenness Centrality, *optimized for scale-free sparse graphs*
  - [Bader, Madduri; ICPP 2006]

- Capability to solve real-world instances more than three orders of magnitude larger than current SNA packages!

- We have analyzed *several large-scale real datasets*: patent citation networks, movie-actor, and protein-interaction networks

BC Previous Results

- *Traditional Algorithm* for BC computation:
  - Compute the length and number of shortest paths between all pairs
  - Sum all pair-dependencies ($\frac{\sigma_{st}(v)}{\sigma_{st}}$)
    - $O(n^3)$ time summation, $O(n^2)$ storage of pair dependencies
- Current Social network analysis packages (UCINET, Pajek, InFlow etc.) use this straight-forward $O(n^3)$ algorithm for implementing Betweenness Centrality
  - They cannot compute BC for graphs larger than 10,000 vertices
- Brandes [2003] proposed a faster sequential algorithm for BC on sparse graphs
  - $O(mn+n^2\log n)$ time and $O(m+n)$ space for weighted graphs
  - $O(mn)$ time for unweighted graphs

Overview of Our BC Parallel Algorithm

- Our parallel algorithm is motived by Brandes' sequential algorithm
  - Augments BFS/SSSP and maintains a running sum of BC
- Compute *n* shortest paths trees in parallel, one for each vertex in the graph
- During these computations, also maintain the predecessor sets
- The dependencies can be computed by traversing the vertices in non-increasing order of their distance from the source vertex
- Individual BFS/SSSP computations are also parallelized

BC Implementation Details

- We have designed and implemented parallel betweenness centrality for two shared memory platforms:
  - Symmetrical multiprocessors (SMPs)
    - Modest number of processors
    - Coarse-grained implementation, BFS/SSSP computations are done concurrently
    - Implemented on IBM p570
  - multithreaded architectures
    - Thousands of hardware threads
    - Individual BFS/SSSP computation is parallelized
    - Implemented on Cray MTA-2

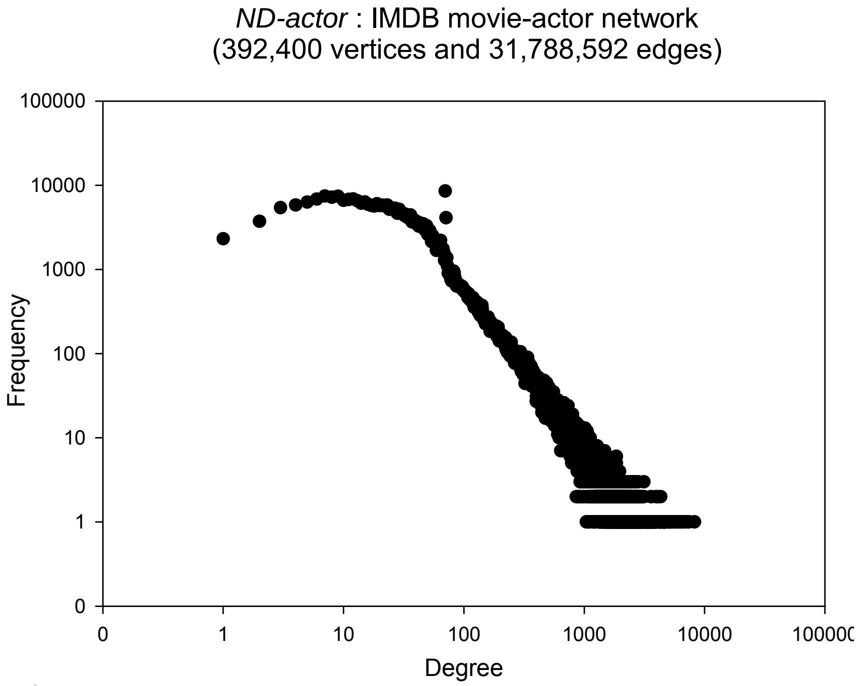
IBM p5 570

- 16-way Power5 symmetric multiprocessor
- 1.9 GHz processor
- 256 GB physical memory
- 32KB L1D, 1.9MB L2, 32MB L3
- 8-way superscalar
- SMT on each core

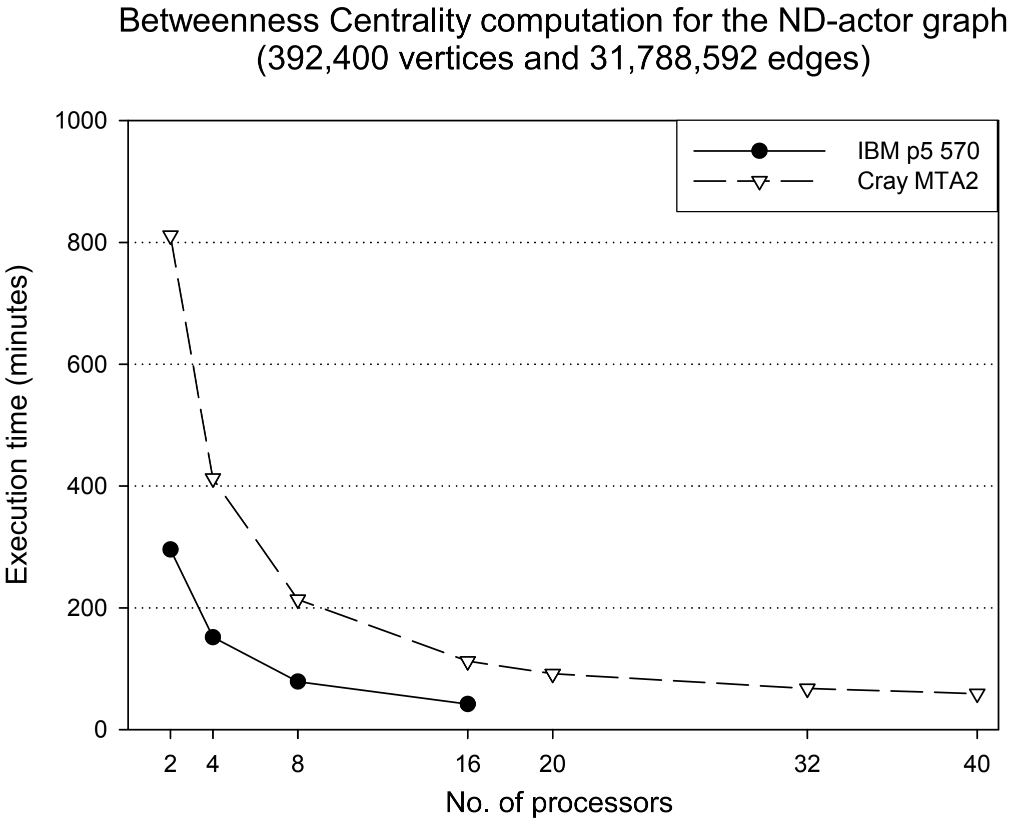
- Supports a C and POSIX threads parallel implementation

BC for IMDB movie actor network

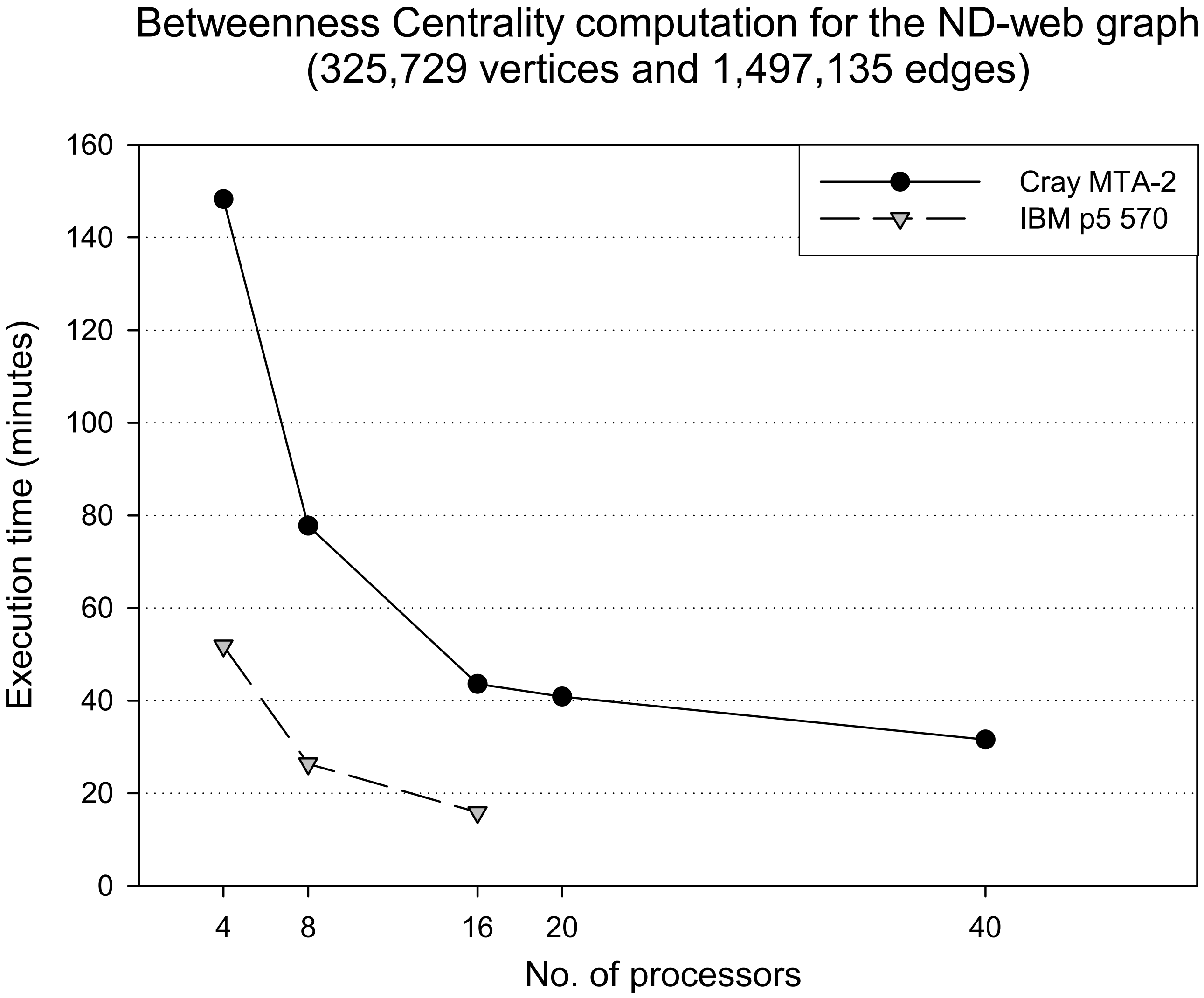
Real-world instance: an undirected graph of 392,400 vertices (movie actors) and 31,788,592 edges. An edge corresponds to a link between two actors, if they have acted together in a movie. The dataset includes actor listings from 127,823 movies.



Degree Distribution: Scale-free



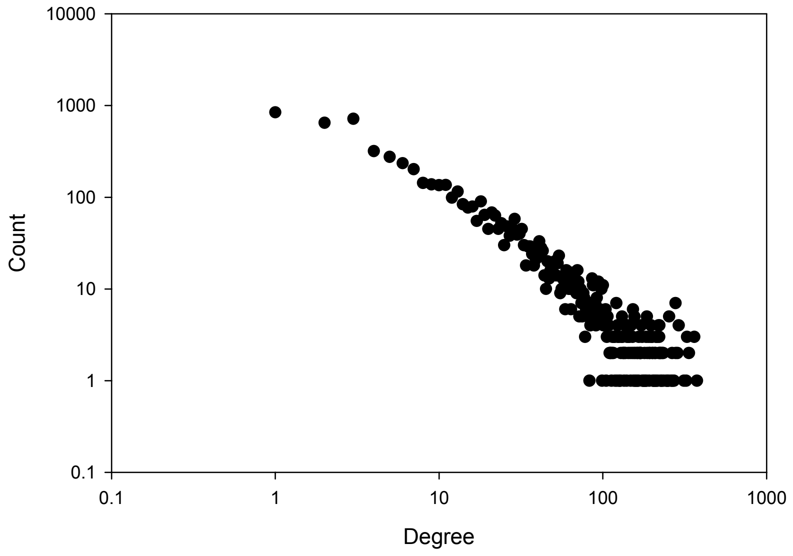
BC for web graph



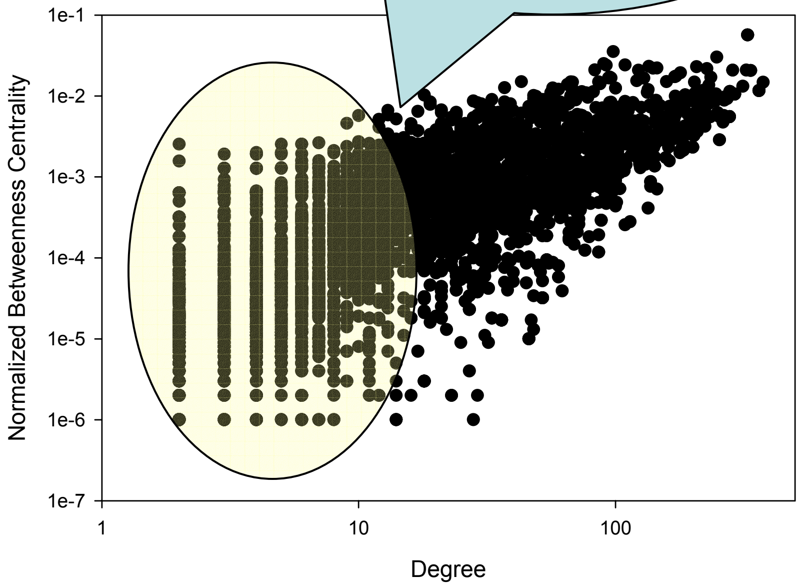
BC Analysis: Protein-protein interactions

- We recently computed betweennes the human genome[1] protein intera

Human Genome Protein Interactions
degree distribution
(undirected graph, 6228 vertices and 71803 edges)



H
Deg

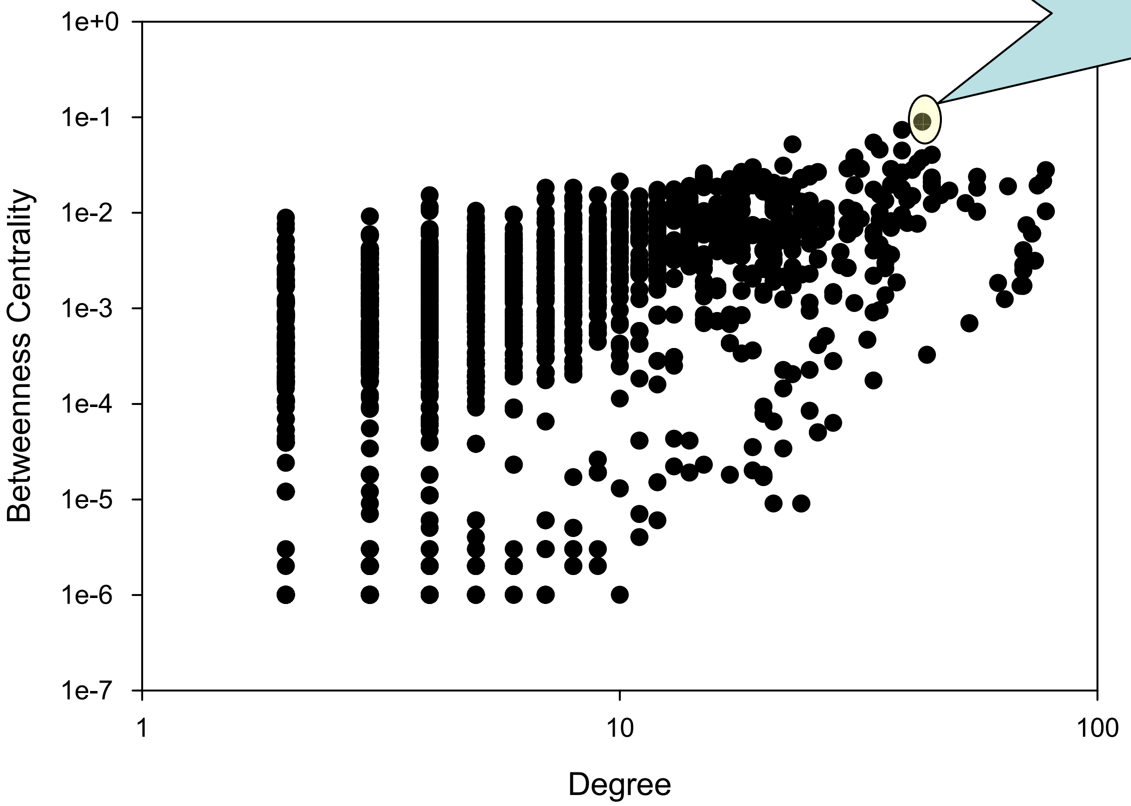


Low degree vertices can have high centrality scores

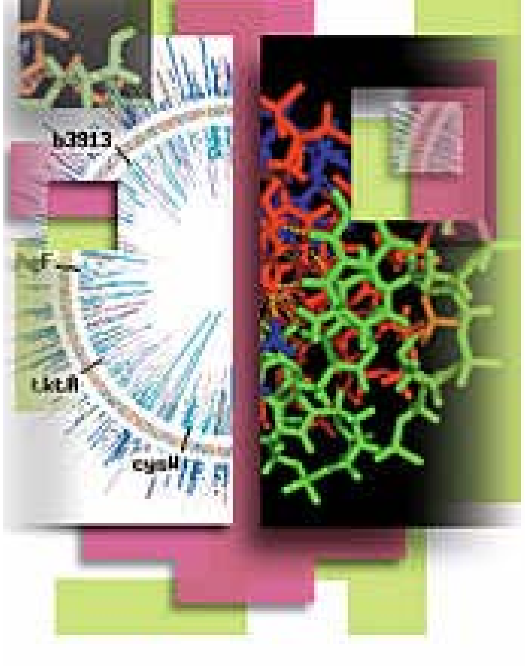
[1] Lehner, Fraser. A first draft human protein interaction map, http://genomebiology.com/2004/5/9/R63

BC Analysis: Protein-protein interactions

Human Genome core protein interactions
Degree vs. Betweenness Centrality



43 interactions
Protein Ensembl ID
ENSG00000145332.2
Kelch-like protein 8



Collaborators

- **Kamesh Madduri** (Georgia Tech)
- Bruce Hendrickson (Sandia National Laboratories)
- Jon Berry (Sandia National Laboratories)
- **Vipin Sachdeva** (IBM Austin Research Lab)
- **Guojing Cong** (IBM TJ Watson Research Center)
- John Feo (Cray, Inc.)

Acknowledgment of Support

- National Science Foundation
  - **CSR**: A Framework for Optimizing Scientific Applications (06-14915)
  - **CAREER**: High-Performance Algorithms for Scientific Applications (06-11589; 00-93039)
  - **ITR**: Building the Tree of Life -- A National Resource for Phyloinformatics and Computational Phylogenetics (EF/BIO 03-31654)
  - **ITR/AP:** Reconstructing Complex Evolutionary Histories (01-21377)
  - **DEB** Comparative Chloroplast Genomics: Integrating Computational Methods, Molecular Evolution, and Phylogeny (01-20709)
  - **ITR/AP(DEB):** Computing Optimal Phylogenetic Trees under Genome Rearrangement Metrics (01-13095)
  - **DBI:** Acquisition of a High Performance Shared-Memory Computer for Computational Science and Engineering (04-20513).

- IBM PERCS / DARPA High Productivity Computing Systems (HPCS)
  - DARPA Contract NBCH30390004

Petascale Computing for Graph Theory: Conclusions

- Need to move from FP-centric to data-centric computing
  - Impact to emerging areas such as life sciences and informatics
- Several architectural features reduce the programmer’s burden and enable high-performance large-scale applications with irregular data structures
- How will we program multicore processors, especially for these applications?
- Will Microsoft / Intel reach this before the HPC community? ☺