Streaming Graph Analytic Algorithms in DARPA's HIVE Challenge

Steven T. Smith

LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Shameless website plug → **graphchallenge.org**

DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

This material is based upon work supported by the Defense Advanced Research Projects Agency under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency.

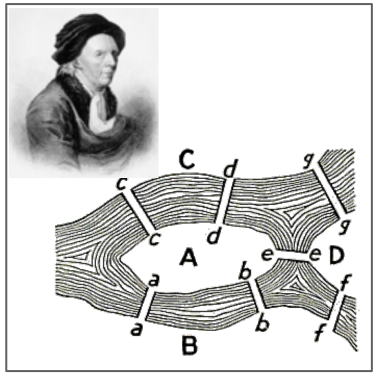
Outline

- Graph Challenge overview
- Static graph challenge: Subgraph isomorphism
- Streaming graph challenge: Stochastic block partitioning
- Challenge data sets and metrics
- Summary

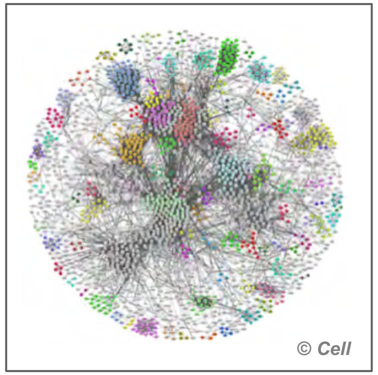
Scales of Graph Problems

Small Graphs

Euler's Bridges of Königsberg, 7*

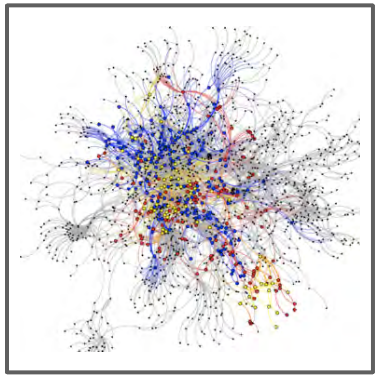


Drosophila protein interaction map, 10^4



Large Graphs

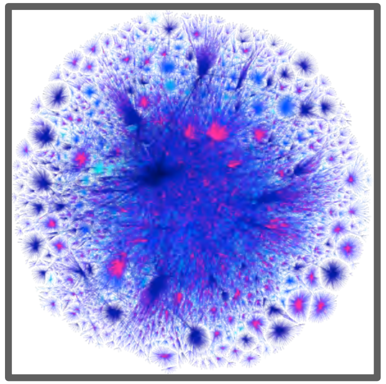
Scientific Publications, 10^8



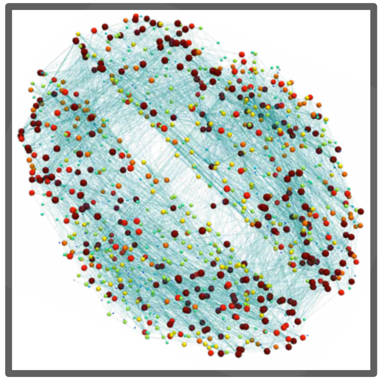
Social Media, 10^{11}

Enormous Graphs

The Internet, 10^{12}



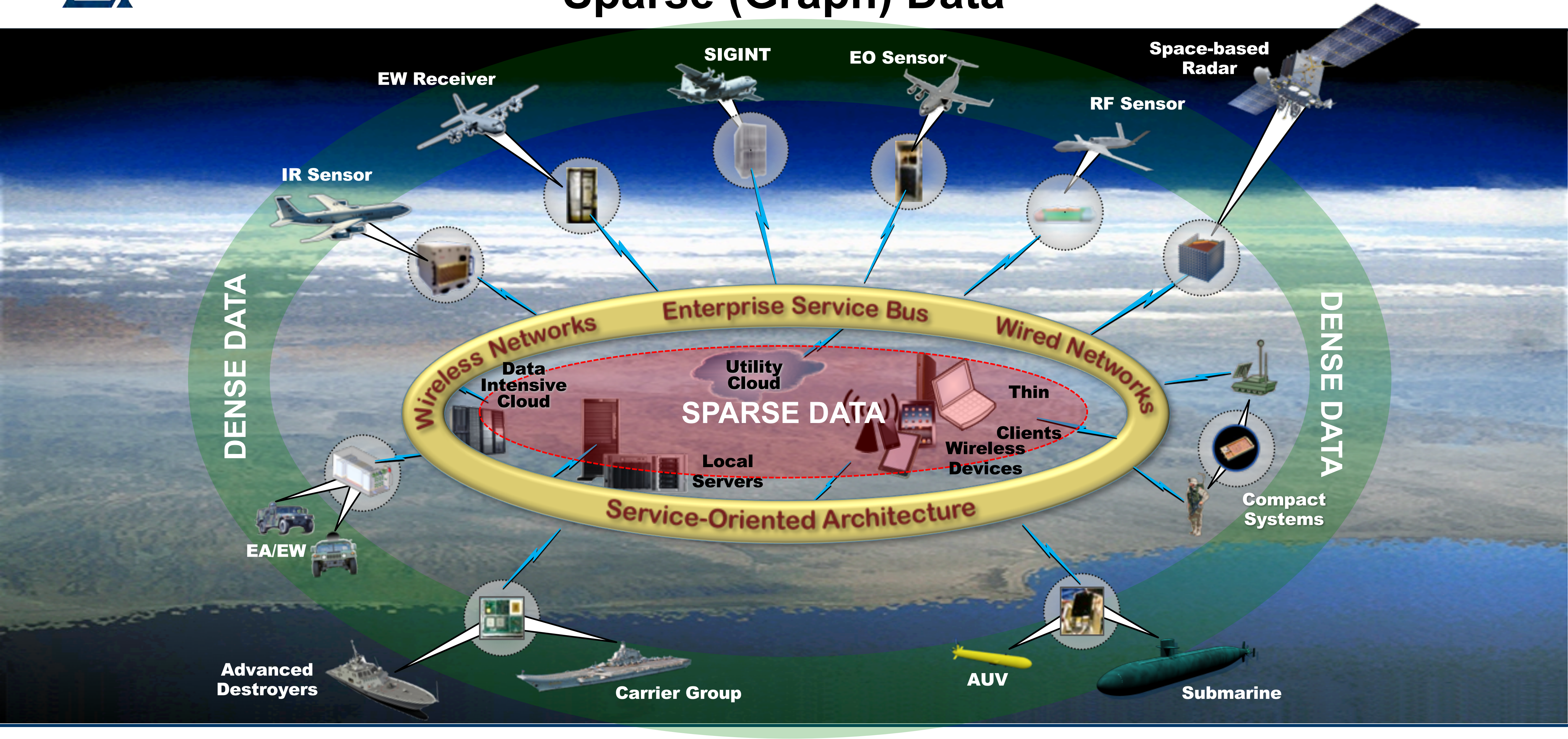
Human Brain, 10^{14}



Real-world graph applications range from thousands to millions to trillions and beyond

***Order of magnitude of connections/interactions**

DoD Systems Produces Large Amounts of Sparse (Graph) Data

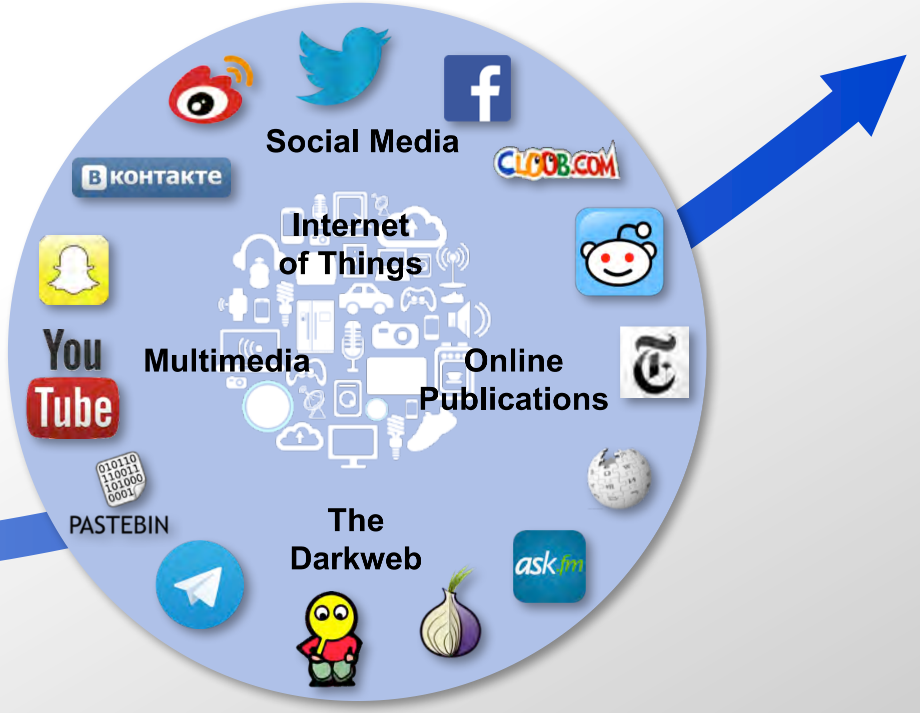


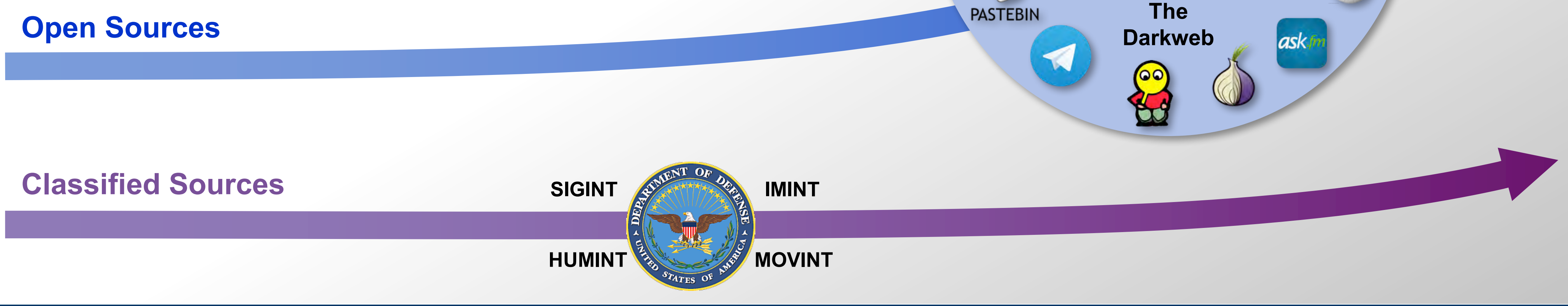
AUV = Autonomous Underwater Vehicle
IR = Infra Red
EA/EW = Electronic Attack/Electronic Warfare

SIGINT = Signals Intelligence
EO = Electro Optic
RF = Radio Frequency

The ISR Challenge and Opportunity in Data*

- From 2013 to 2020, the digital universe will grow by a factor of 10—from 4.4 trillion gigabytes to 44 trillion gigabytes[1]
- Internet of Things, will grow from 2% of the digital universe in 2013 to 10% in 2020 and estimates are for trillions of sensors[1] within 10 years!
- Open sources will far outstrip classified ones, but the combination of the two will provide key leverage for the DoD/IC

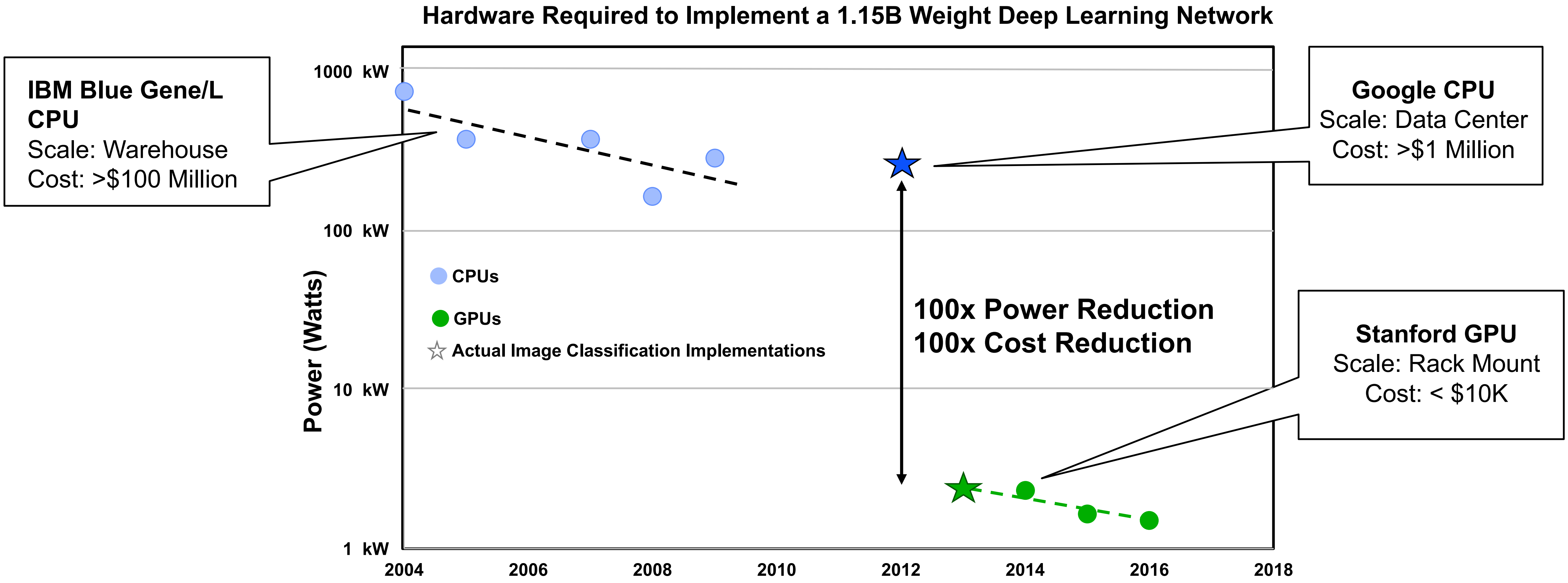




*Intelligence, Surveillance & Reconnaissance

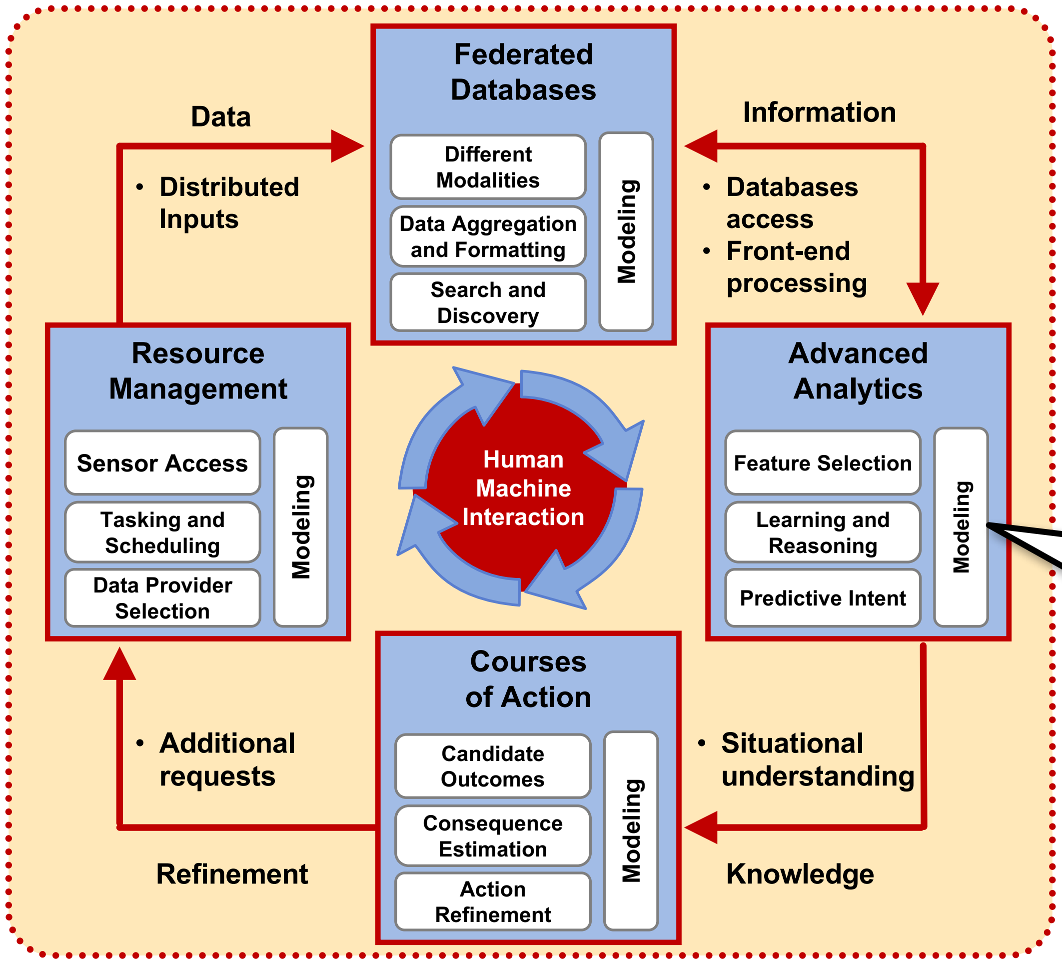
[1] *Source: IDC White Paper April 2014*

Hardware has played a vital role in advancing AI



The GPU made Deep Learning functional and accessible by dramatically reducing the cost and processing power relative to CPU's

DARPA HIVE Goal: Enable Insight Computing for Enabling Autonomous Sensors and Systems



- Parallel processing
- Parallel memory access
- Fastest (TB/s) to memory
- Higher scalability (TB/s)
- Optimized for Graphs

HIVE Challenge

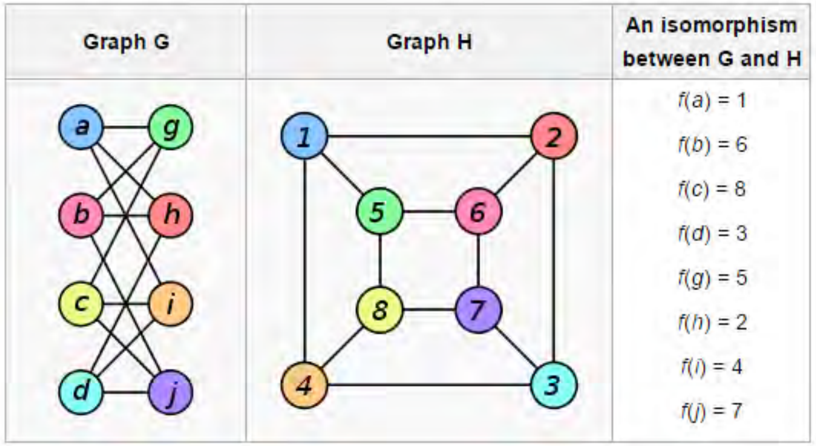
- The HIVE Graph Challenge seeks input from diverse communities to develop graph challenges that take the best of what has been learned from groundbreaking efforts such as GraphAnalysis, Graph500, FireHose, MiniTri, and GraphBLAS to create a new set of challenges to move the community forward
- Initial Graph Challenges
  - Static Graph Challenge: Sub-Graph Isomorphism

    This challenge seeks to identify a given sub-graph in a larger graph
  - Streaming Graph Challenge: Stochastic Block Partition

    This challenge seeks to identify optimal blocks (or clusters) within a graph

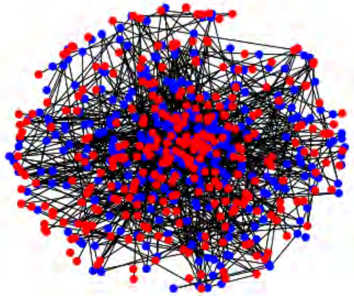
Challenges of the Graph Challenge Challenge

- **Static Graph Challenge: Subgraph Isomorphism**
  - Find a specific subgraph within a larger graph
  - Addresses topological performance

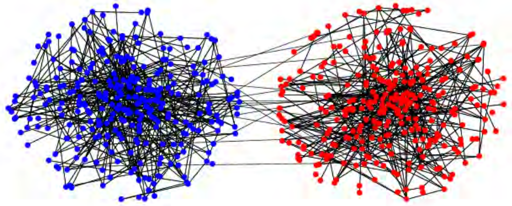


- **Streaming Graph Challenge: Stochastic Block Partitioning**
  - Identify statistically relevant blocks (or clusters) within a larger graph
  - Exploit state for Streaming updates
  - Addresses property-based performance

Streaming unstructured graph



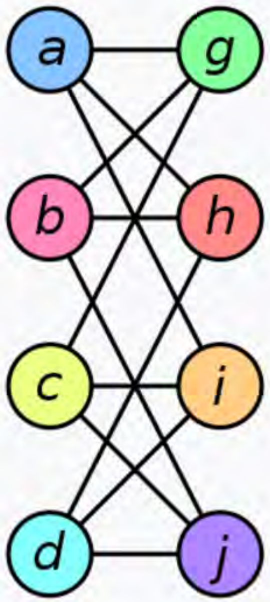
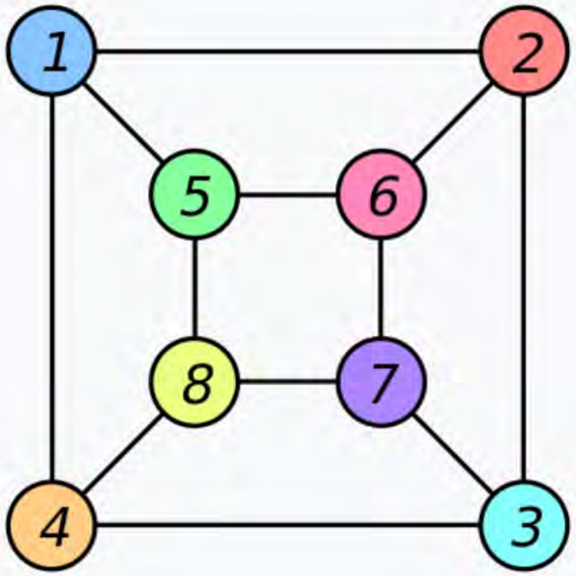
Block optimized graph



Outline

- Graph Challenge overview
- Static graph challenge: Subgraph isomorphism
- Streaming graph challenge: Stochastic block partitioning
- Challenge data sets and metrics
- Summary

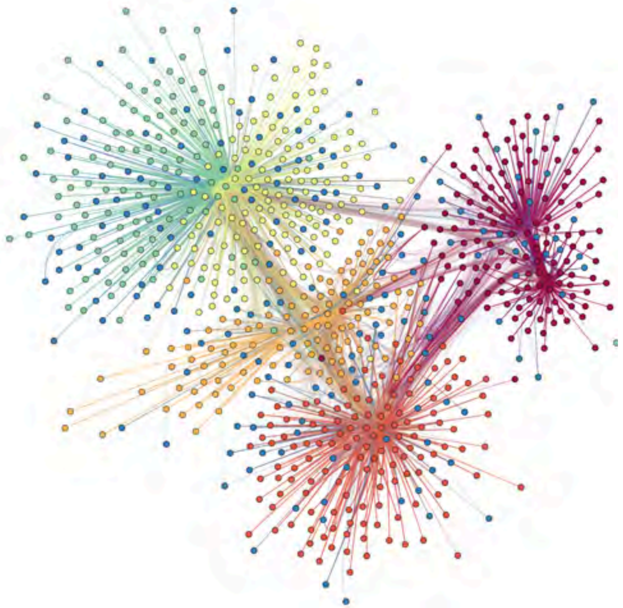
Static Graph Challenge: Sub-Graph Isomorphism

Graph G	Graph H	An isomorphism between G and H
		$f(a) = 1$ $f(b) = 6$ $f(c) = 8$ $f(d) = 3$ $f(g) = 5$ $f(h) = 2$ $f(i) = 4$ $f(j) = 7$

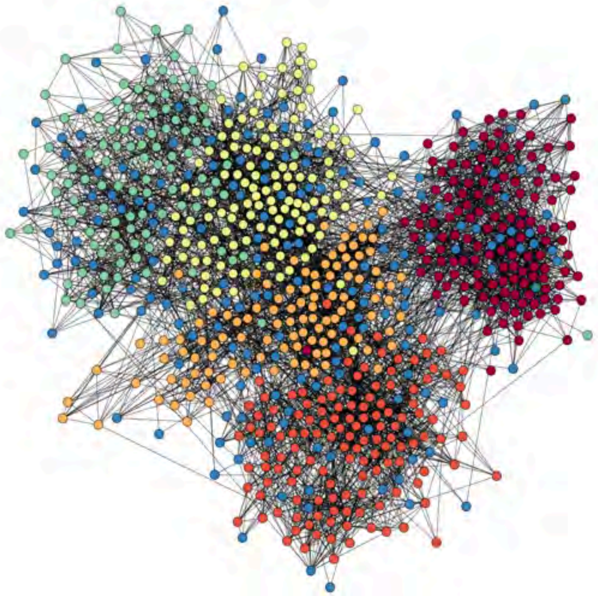
Is there a 1-to-1 mapping of the vertices in Graph H to vertices in Graph G such that every edge in H is also in G ?

Scaling Considerations

Large G and Large H: Too complex

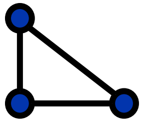


H

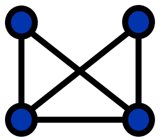


G

Large G and |G| >> |H| : More interesting

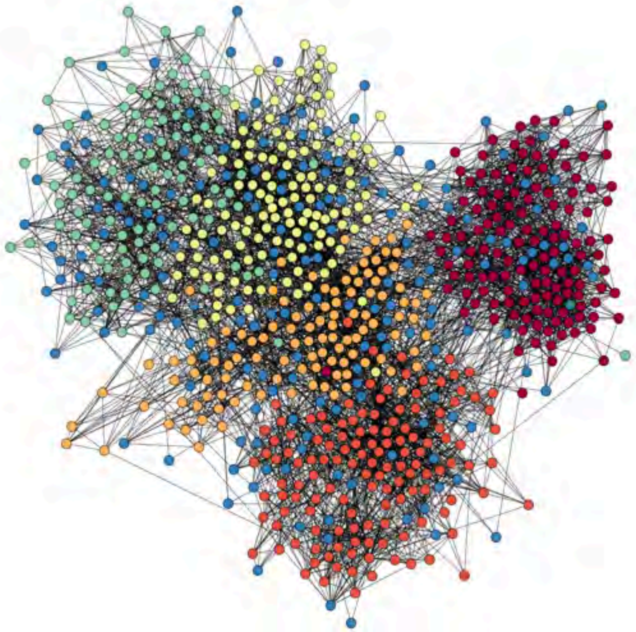


Triangle



k-truss

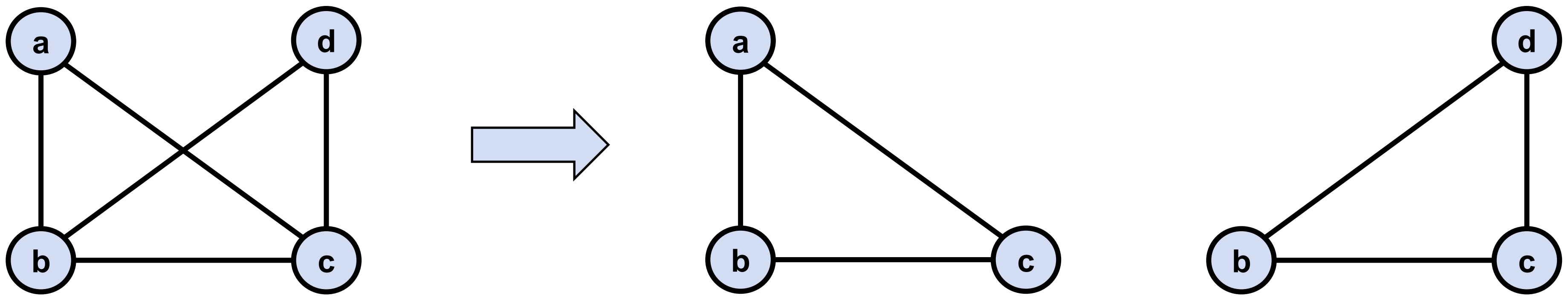
H



G

Complexity of subgraph isomorphism is determined by the size of graphs G and H

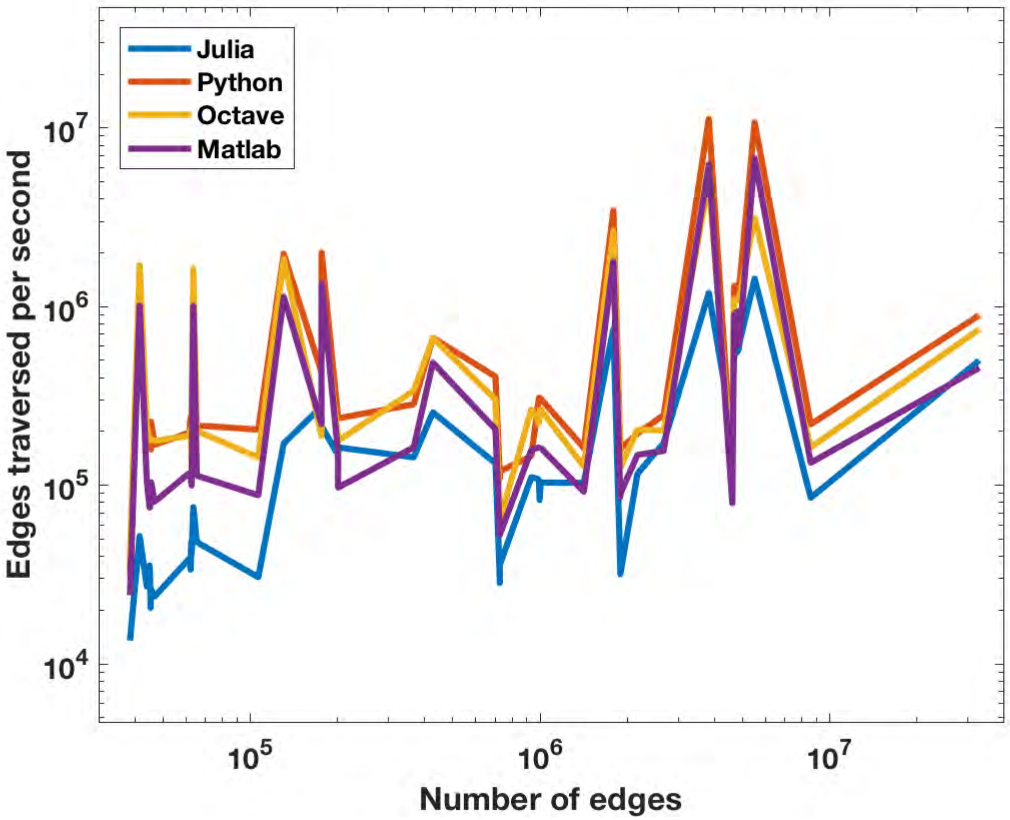
Static Graph Challenge : Triangle Counting



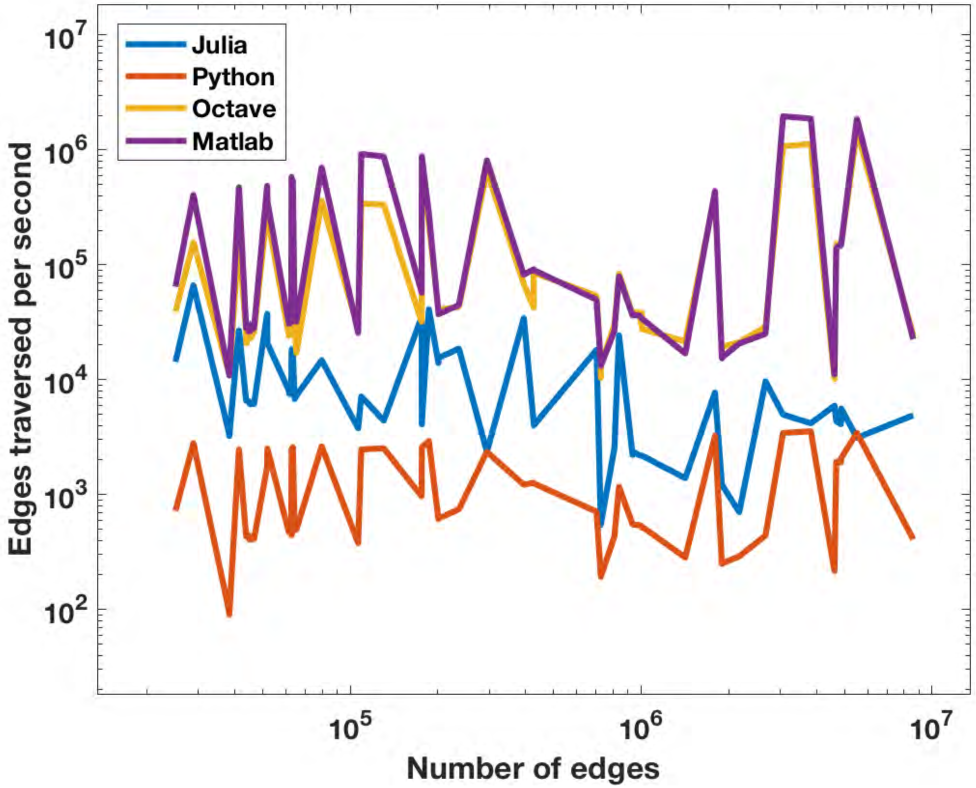
- **Triangles are the most basic non-trivial subgraph**
  - **Set of three mutually adjacent vertices in a graph**
- **Key uses: dense subgraph detection, characterizing graphs, improving community detection, generating graphs**
- **Related applications in cyber security, intelligence, functional biology**

Performance Results
Triangle Count and K-Truss

Triangle Counting



K-Truss



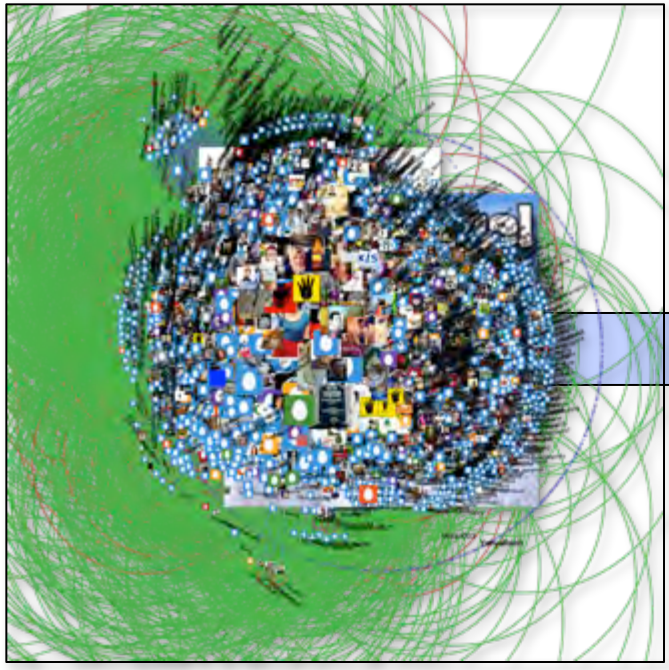
Outline

- Graph Challenge overview
- Static graph challenge: Subgraph isomorphism
- Streaming graph challenge: Stochastic block partitioning
- Challenge data sets and metrics
- Summary

Streaming Graph Challenge
Stochastic Block Partition

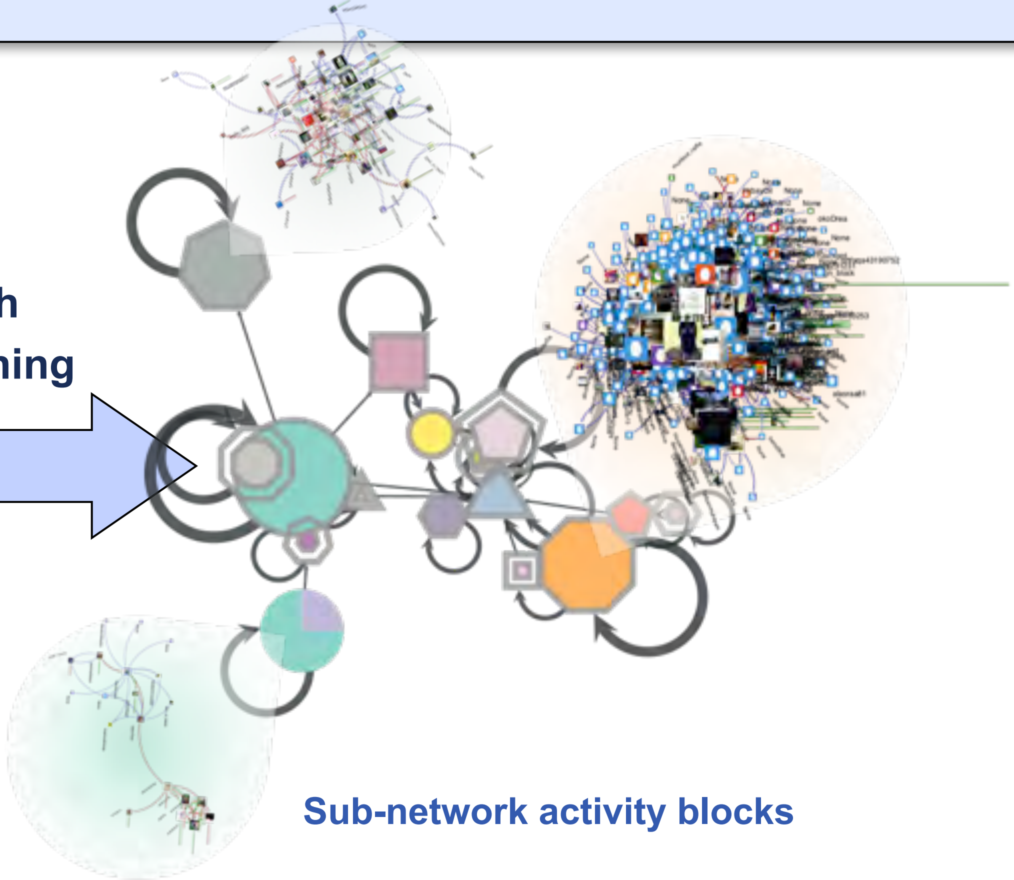
Problem: Discover community structure in graph topology and interaction data

Application: Identify significant activities within large graphs



Social Media Network

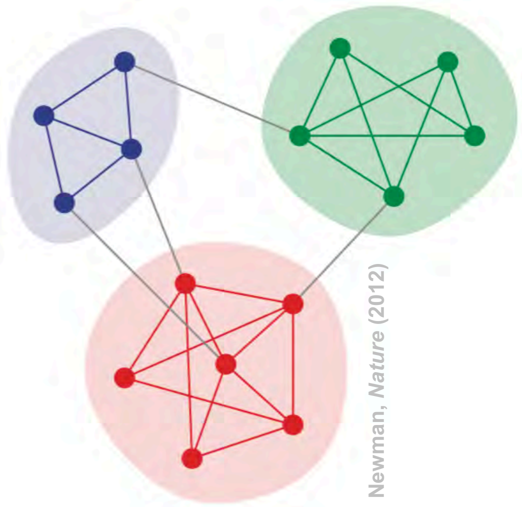
Graph Partitioning



Sub-network activity blocks

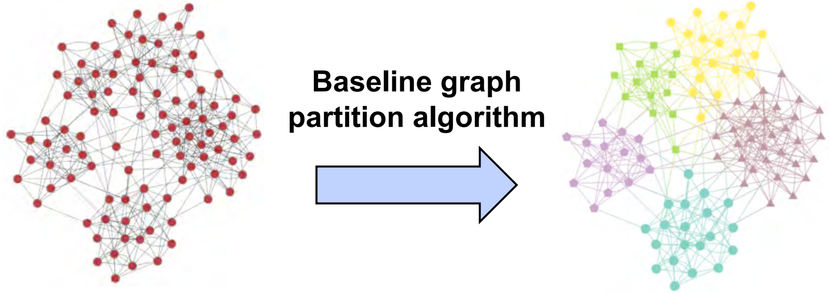
Blockmodel Graph Partitioning

- **Blockmodels capture the community structure of graphs:**
  - Block membership for each node
  - Block matrix describes interaction between blocks
  - Typically, nodes in the same block interact more than those between different blocks
- **Rigorous statistical inference on blockmodels is a principled way to partition graphs**
  - Inferred block membership on each node provides the partition
  - Model selection determines the optimal number of blocks (i.e., partition resolution)
  - Bayesian methods capture uncertainty of the partition
  - Produces better partition than spectral and modularity maximization methods



Blockmodel Partitioning Baseline Algorithm

- Based on degree corrected stochastic blockmodel (Karrer and Newman 2010)
  - Realistic model obtained by varying degrees across vertices
  - Likelihood a function of edge counts between blocks (simple, parallel)
- Returns partition with shortest graph description length (Peixoto 2012, 2013)
- Markov chain Monte Carlo (MCMC) inference approach amenable to parallelism
- Challenging computational complexity for large graphs
- Participants may enter with a different partition algorithm
  - All entries should report performance using the Challenge metrics and data sets
  - The true number of blocks is NOT given to the algorithm



Degree Corrected Blockmodel

Network model with Poisson Interactions:

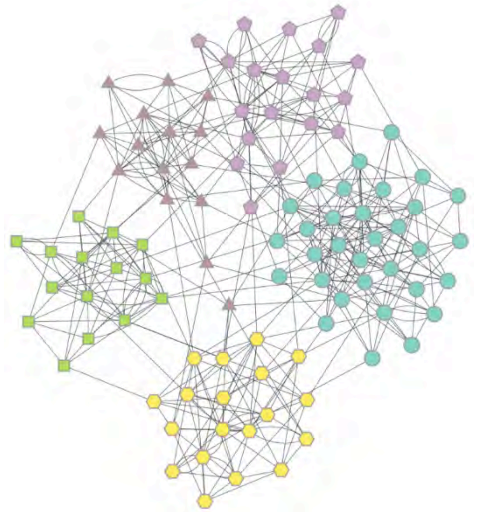
Edge interaction from i to j ~ Poisson(λ_{ij})

Rate $\lambda_{ij} = \boxed{d_i d_j} \times \boxed{\text{block interaction rate}(b_i, b_j)}$

Node degrees

Stochastic blockmodel

Block membership



- **Realistic degree distribution accounts for features like "six degrees of Kevin Bacon" (aka small world) and power-law scaling**
- **Realistic community structure captured by stochastic blockmodel**
- **A combination of two well-known random graph models**

- Stochastic blockmodels and community structure in networks, Karrer and Newman (2011)
- A random graph model for power law graphs, Aiello, Chung, and Lu (2001)
- Estimation and prediction for stochastic blockmodels …, Snijders and Nowicki (1997)

Parallel MCMC for Blockmodel Graph Partitioning

- **The distribution on the graph partition does not have a closed form and has high dimension (#vertices)**

$$p(\mathrm{b}|\mathrm{G}) \propto \sum_{ij} M_{ij} \log\left(\frac{M_{ij}}{\mathrm{d}_i \mathrm{d}_j}\right)$$

 # edges between block *i* and *j* according to partition *b* on graph *G*

 # edges in block *i and j*

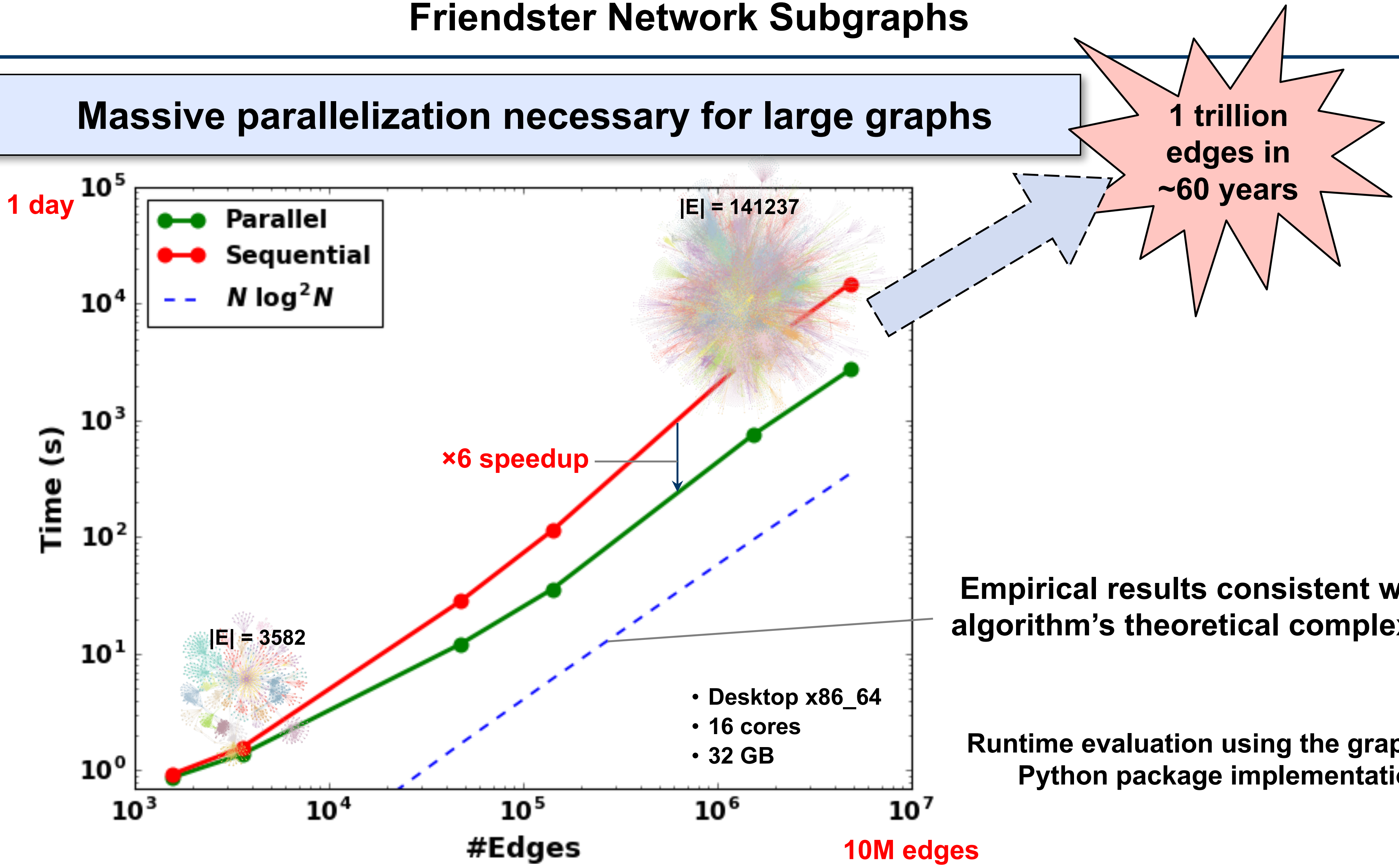
  - **MCMC sampling is ideal for computing this distribution**
  - **Efficient partition updates in parallel, one node at a time**

- **Updating each node can be massively parallelized**
  - **Sequential update gives exact graph partition distribution**
  - **Parallel update gives a reasonable approximation and produces comparable result in our evaluation over simulated and real graphs**

Processing Time vs. Graph Size

Friendster Network Subgraphs

Massive parallelization necessary for large graphs



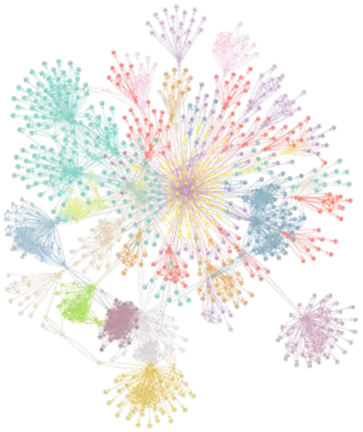
Empirical results consistent with algorithm's theoretical complexity

Runtime evaluation using the graph-tool Python package implementation

Parallel MCMC Partitioning Closely Approximates "Exact" Sequential MCMC

- **Partitions on a range of graphs sizes show differences within the random variability of the partitioning algorithm itself**
- **Quantitative metrics used for comparing partitions (correctness metrics)**
- **Recent theoretical development in parallel MCMC suggests good performance under sparse correlation (De Sa 2016)**
  - **Nodal block assignments correlate sparsely since graphs are typically sparse**

Sequential, $O(10^3)$ vertices



Parallel, $O(10^3)$ vertices

Sequential, $O(10^4)$ vertices

Parallel, $O(10^4)$ vertices

Outline

- Graph Challenge overview
- Static graph challenge: Subgraph isomorphism
- Streaming graph challenge: Stochastic block partitioning
- Challenge data sets and metrics
- Summary

Graph Datasets

Wikipedia

Twitter

Stanford Network Analysis Project

Amazon AWS Public Datasets

Road Networks

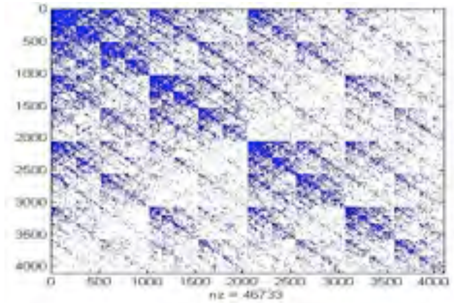
Autonomous Systems

Yahoo!

WEBSCOPE

Datasets

Graph500.org



- **Public, open source data available from a variety of sources**
- **Packaged in standardized, basic triple-store and hosted on AWS S3**

Graph Partition Performance Metrics

- Correctness metrics
  - Objective comparison between performer results with common datasets
  - Diagnostic performance tools for performer development
  - Baseline recommendation: Pairwise Precision-Recall
- Computational metrics
  - Total number of edges in graph partitioned
  - Execution time
  - Rate (edges/second)
  - Energy (Watts)
  - Rate per energy (edges/second/Watt)
  - Memory and processor requirement (amount and type used)
- For streaming graphs, evaluation should be done at each stage using these metrics

Summary

- Graph partitioning challenge on embedded real-world and simulated graphs
  - Partitioning algorithm uses realistic stochastic blockmodel and statistically rigorous inference
  - Streaming versions defined
  - Correctness and computational performance metrics defined
- Performer implementations will compute the block partition on provided data sets and report the given metrics
- A submission to the IEEE HPEC Graph Challenge consists of a conference style paper describing the approach, implementation, innovations, and results
- Both hardware and software should be described in solution
  - Innovative hardware solutions are of interest (in addition to best algorithm for hardware)
  - Special interest in performance for large scale data sets (1 billion edges)

Acknowledgements

- Bill Arcand
- David Bestor
- Chansup Byun
- Vijay Gadepally
- Michael Jones
- Matt Hubbell
- Jeremy Kepner
- David Martinez
- Pete Michaleas
- Lauren Milechin
- Julie Mullen
- Paul Monticello
- Sanjeev Mohindra
- Edward Kao
- Michael Hurley
- Andy Prout
- Albert Reuther
- Tony Rosa
- Sid Samsi
- William Song
- Charles Yee