Challenges in Data Mining and Optimization at Sandia National Laboratories

Cynthia A. Phillips (Sandia National Laboratories)
(and many others)
Sandia National Labs (and other national labs, industry, and universities) face many Big Data problems

- Fast streaming data (e.g. cybersecurity)
  - Sampling, fast ingestion
- Managing massive scientific simulations
  - What to store? How to visualize? Managing workflows at exascale, etc.
- Decision support under uncertainty
  - Optimization, quantification of uncertainty, ranking, solution diversity, human element, informative non-biased visualization
- Missing data, adversarial tampering
- Low-dimensional representations
  - Compressive sensing, tensor decomposition
Geospatial Data Mining

- Many sensors with different location, technology, uncertainty
- Manage data collection and interpretation to
  - Aid decision making
  - Minimize uncertainty

Sample applications:
- Image analysis (e.g. overhead)
  - Object identification
- Seismic data: eg. Source location

(wikipedia)
Geospatial Data Mining

• Challenge 1: Statistical analysis of examples
  – Integration of multi-sensor data
  – Uncertainty extraction and propagation
  – Analysis of individual source contributions (what, if anything, does each source contribute to the final result?)

• Challenge 2: use this knowledge to create an objective function for sensor scheduling optimization
  – Which sensors most helpful in reducing decision uncertainty (which are best for answering the target question?)
  – Combine with existing considerations such as weather, availability, priority, etc

• Similar issues in other settings
  – Collecting dispersed data over low-bandwidth channels
  – Allocating computational resources

PoC: David Stracuzzi (SNL)
Streaming

- Information generated “outside”
  - Sensors, computer simulation, network traffic
  - (key,value) pair, graph edges, etc.
- Lots of research on sampling
- National Labs concern: keep as much as you can
  - Fast write-optimized data structures
    - Minimize drops, maximize accuracy
  - Light-weight streaming infrastructure
  - architectures: SSDs, multicore, distributed
- Benchmarks
  - Firehose
  - Streaming graph benchmarks
Connected Components

- Input: stream of edges (learn nodes from edge)
- Output: (node, label) pairs
- Two nodes have the same label if there is a path between them
- Can’t output 2 pairs with different labels until seen all of (finite) graph
What Analysts Want/Need

Queries:

\((v_1, v_2)\) connected?  
All components < k  
Neighbors of v

Infinite stream of edges

Query responses

• How do we approximate this in a real streaming setting?
Parallel Distributed Algorithm

- Store whole graph
- Exact query answers with latency
- Age oldest edges when too full
- Queries disabled during data structure repair. No edges dropped

Preliminary experiments:
- 1.1M edges/sec real data.
- 350k edges/sec RMAT streams.

Feedback loop, for maintenance

Feedback (≤ k -1)

Edges, Queries, Commands

Query responses
Community Detection

One approach: Treat as an optimization problem

Metrics: modularity, conductance

Lots of algorithms: CNM, wCNM, Bader-McCloskey, Louvain, Ruan and Zhang, .... (hundreds of papers in physics and CS)

Some complications:

• How can we tell if communities are significant?
  – Distinguish from random fluctuations
  – Not resolved enough? Too resolved?

• Hierarchy

• Overlapping

Relational Data Mining: Our Goal

Issues for finding structure in social and biological networks
- Lack of rigorous measures for optimization
  - No graph theoretic function seems to always capture what humans perceive are the best communities
- Inherent randomness
- Uncertainty in observations

Makes it difficult to
- Judge quality of a solution
- Compare solutions/methods

Challenges:
- Statistically informed algorithms
- Benchmarks
- Means to compare algorithms
Bayesian Community Detection

- Use a prior to capture community structure for an application
- Get a distribution of community partitions as output

A Bayesian CD method:
Jiqiang Guo, Alyson G. Wilson, Daniel J. Nordman
A New Distributed Computing Model

Alice and Bob (or more) independently create social graphs $G_A$ and $G_B$.

- Alice and Bob each know nothing of the other’s graph.
- Shared namespace. Overlap at nodes.

Goal: Cooperate to compute algorithms over $G_A \cup G_B$ with limited sharing (polylog) or secure multiparty computation.
Autonomous Data Center Model

Motivation

- Company mergers
- National security: connect-the-dots for counterterrorism

Nodes are people

- Exploit structure of social networks
- We must because there are bad lower bounds otherwise
Triangles Important in Social Networks

– Strong triadic closure (Easley, Kleinberg): two strong edges in a wedge implies (at least weak) closure.
  • Reasons: opportunity, trust, social stress

– Converse of strong triadic closure: not (both edges strong) implies coincidental closures
  • experimental evidence: Kossinets, Watts 2006

– Online social networks can have non-meaningful triangles (Beiber phenomenon)
Maximum Triangle Density Subgraph (MTDS)

- Algorithmic tool
- Find subgraph that maximizes

\[
\text{Triangle density} = \frac{\text{# triangles in subgraph}}{\text{# vertices in subgraph}} \quad \frac{7}{5}
\]

Triangle density = \( \frac{7}{5} \)
Maximum Triangle Density Subgraph (MTDS)

- Find subgraph that maximizes \[ \frac{\text{# triangles in subgraph}}{\text{# vertices in subgraph}} \]

Generalization of Charikar’s LP for maximum edge density:
\[ x_i = \text{fraction node } i \text{ in subgraph. } t_{ijk} = \text{fraction triangle } (i,j,k) \text{ in subgraph} \]

\[
\begin{align*}
\max & \quad \sum_{i,j,k} t_{ijk} \\
\text{s.t.} & \quad t_{ijk} \leq x_i \\
& \quad t_{ijk} \leq x_j \\
& \quad t_{ijk} \leq x_k \\
& \quad \sum_i x_i = 1 \\
& \quad 0 \leq x_i, t_{ijk} \leq 1
\end{align*}
\]

Naively, too much space
Maximum Triangle Density Subgraph (MTDS)

• Algorithmic tool
  - Find subgraph that maximizes
    \[ \frac{\text{# triangles in subgraph}}{\text{# vertices in subgraph}} \]

Triangle density = \( \frac{7}{5} \)

• Greedy 3-approximation (from Charikar’s 2-approx for edge density)
  - Find triangles each node participates in
  - While graph not empty
    • Remove a node of minimum triangle count, update counts
    • Choose graph with maximum triangle density among n choices
  • The approximation works when triangles have static weights.
Edge strength

- A notion somewhat like Easley and Kleinberg 2010, and Berry et al., 2011

\[ s(u, v) = \frac{2 \times \# \text{ triangles on}(u, v)}{d_u + d_v - 2} \]

\[ s(u, v) = \frac{2 \times 2}{5 + 6 - 2} = \frac{4}{9} \]

- Edge strength a continuum, not just strong/weak
- Depends only on topology
- How do you efficiently compute maximum triangle density subgraph when edge strength and triangle weight depends on triangles?
Final Comments

- This is a small sampling of just what I’ve been involved in
- There are a lot more problems like this
- Pitch: Sandia is a great place to work
  - We need more statisticians and statistics-saavy optimizers
  - Sandia is a great place to do an internship or post doc