What to measure in a graph stream?

C. Seshadhri
Sandia National Labs, Livermore

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.
Understanding large graphs

• Start with the simple, static problem

• I get a large graph from some <INSERT NAME> application

• “You’re a graph expert. Tell us something about this graph.”
Things to measure

- Decade of network science research gives enough leads
- Degree distributions
- Clustering coefficients
- Eigenvalues of matrix
- Hop-plots
- Core decompositions
- Community structure

We “know” the space of real networks

• Social/biological/communication networks are contained in a “tiny slice” of all graphs
• By no means a solved problem, but we have a good sense of what this slice is
• Nice collection of properties across domains: degree distributions, triangles, cores, etc.
Measurements help

- Having measurements makes the modeling discussion sane
- Let me give you a demonstration
Measurements to inference

Random graph:
1. Formed according to CL Model
2. “High” clustering coefficient

\[ G = (V, E) \quad \{d_i\}_{i \in V} \text{ (prescribed)} \]
\[ \text{Prob} ((i, j) \in E \mid i, j, \in V) \propto d_i \cdot d_j \]

**Thm: Must** contain a “substantive” subgraph that is a **dense Erdös-Rényi graph**.

A heavy-tailed network with a high clustering coefficient contains many Erdös-Rényi **affinity blocks**. (The distribution of the block sizes is also heavy tailed.)

**CL Model**

\[ c = \frac{3 \times \# \text{ triangles in graph}}{\# \text{ wedges in graph}} \]

**Global Clustering Coefficient**

\[ \tilde{V} \subset V, \tilde{E} \subset E \]
\[ \text{Prob} ((i, j) \in \tilde{E} \mid i, j \in \tilde{V}) \propto \text{constant} \]

Dense Erdös-Rényi Subgraph

Some modeling ability

- Far from solved, but models are reasonable
- [Kolda et al 14] And we can scale these models to get decent synthetic graphs
- At the very least, not hard to generate some test cases (ER, CL, SKG, ForestFire, Hyperbolic, etc.)
But what about a graph stream?

- Time: a complete new dimension to worry about
- Standard approach is to just aggregate over windows
- [Macskassy 14] “Mining dynamic networks: The importance of pre-processing on downstream analytics”
  - The choice of time window affects results
Temporal degree distributions

• Degree distribution is not one object any more
• Degrees vary over time
  – Is there some pattern that is relevant across domains?
  – How to represent information?
• [Shmueli et al 14] Degrees in social trading network over time
Time in subgraphs

- A subgraph is a temporal object
- Are there any trends/patterns over time?
Measures for temporal graphs

• Area is wide wide open

• Few scattered results, but nothing compelling

• Lack of good datasets...?
Now for an actual result

- Not directly related to the measurement problem
- But nice (?) story on how thinking about streaming algorithms could lead to ideas
  - Result with Madhav Jha and Ali Pinar (2014)
Triangle information

- $W =$ no. of wedges (paths of length 2)
  - “Center” of wedge is middle vertex
- $T =$ no. of triangles
- Transitivity = $\tau = 3T/W =$ fraction of closed wedges

**Wedge Sampling**: Sample a few wedges (uniformly). Check if each is closed.
$\tau = \#\text{ closed sampled wedges} / \#\text{ sampled wedges}$
Streaming Triangle Counting

Triangles so far: 4
Graph seen so far:

- Data streams important for situational awareness
  - Streaming algorithms also useful for large data sets
- Algorithmically
  - See each edge only once
  - Either take action or lose that piece of information forever
Real-world messiness

- Real-world streams are multigraphs: edges can be repeated
  - Consider communication network. Obvious repeats
- There is no true “graph”. It depends on how you aggregate
  - Different time intervals give different graphs

Standard approaches

- There are no repeats. Assume graph is simple
- Aggregate every edge seen. The “window” is all of history
Drawbacks of ignoring repeats

- Assumptions useful for algorithmic progress, but avoids real-world complexities
  - Algorithms cannot be deployed in “wild”

- Removing repeated edges requires extra pass over edges
  - Assumption of no repeats is expensive to enforce

- Not clear how to store information of various time-windows simultaneously
Our result

- Algorithm for approximating triangle counts and transitivity of underlying simple graph
  - No preprocessing. Works with raw stream
- Maintain information on multiple time windows with same data structures
- Provable bounds on accuracy, excellent empirical behavior
- Based on [Jha-S-Pinar 13] approach, but needs new ideas to debias counts
Past art

- Much work on triangle counting in data streams
- Good theory and empirical behavior
- [Jha-S-Pinar 13], [Pavan-Tangwongsan-Tirthapura-Wu 13], [Ahmed-Duffield-Neville-Kompella 14]

- Work on idealized stream with no repeats, and only aggregate all of history
Just to make my point...
Case study: DBLP graph

- DBLP co-authorship graph: all paper records over 50 years gives graph stream
  - Naturally repeated edges. Colleagues work together for many papers
  - Size = 3600K, non-repeated edges = 254K
- For graph $G[t:t+\Delta t]$, there is associated transitivity and triangle count
  - How does this vary with $t$ and $\Delta t$?
Triangle trends in DBLP graph

- Stream size = 3600K, non-repeated edges = 254K
- Results obtained with storing 30K edges
- Enron email network: stream size 1100K, non-repeated 300K
- Storage used = 8K
- Trends “opposite” to DBLP graph
• Natural question for affiliation network like DBLP
Algorithm Sketch

Edge stream

Hashing based sampling (add if $h(e) < \alpha$)

Edge pool

Hash sampling again (add if $h(w) < \beta$)

Wedge pool

part of triangle?
Streaming Algorithm Features

- Only two parameters $\alpha, \beta$
  - No knowledge of graph required
- Provable guarantee on expectation
  - Provable variance bound (though not useful in practice)
- Space around 1% of total stream
- Accuracy always within 5%
We need
- Metrics for temporal structure
- Data to try things out
- Algorithms to compute these metrics efficiently
- Domain expertise to guide us

What is normal, what is abnormal?

Generating realistic data?

Back to the beginning: Could give insight into the right things to measure...?