GRAPH ANALYSIS BEYOND LINEAR ALGEBRA

E. Jason Riedy

DMML, 24 October 2015

HPC Lab, School of Computational Science and Engineering
Georgia Institute of Technology

MOTIVATION AND APPLICATIONS

(Insert prefix here)-scale data analysis

Cyber-security	Identify anomalies, malicious actors
Health care	Finding outbreaks, population epidemiology
Social networks	Advertising, searching, grouping
Intelligence	Decisions at scale, regulating algorithms
Systems biology	Understanding interactions, drug design
Power grid	Disruptions, conservation
Simulation	Discrete events, cracking meshes

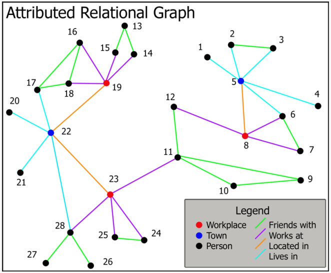
- Graphs are a motif / theme in data analysis.
  - Changing and *dynamic* graphs are important!

OUTLINE

1. Motivation and background
2. Linear algebra leads to a better graph algorithm: incremental PageRank
3. Sparse linear algebra techniques lead to a scoop: community detection
4. And something else: connected components

WHY GRAPHS?

Another tool, like dense and sparse linear algebra.



- Combine *things* with *pairwise relationships*
- Smaller, more generic than raw data.
- Taught (roughly) to all CS students...
- Semantic attributions can capture essential *relationships*.
- Traversals can be faster than filtering DB joins.
- Provide clear phrasing for queries about *relationships*.

POTENTIAL APPLICATIONS

- Social Networks
  - Identify *communities*, influences, bridges, trends, anomalies (trends *before* they happen)...
  - Potential to help social sciences, city planning, and others with large-scale data.
- Cybersecurity
  - Determine if new connections can access a device or represent new threat in $<$ 5ms...
  - Is the transfer by a virus / persistent threat?
- Bioinformatics, health
  - Construct gene sequences, analyze protein interactions, map brain interactions
- Credit fraud forensics $\Rightarrow$ detection $\Rightarrow$ monitoring
  - Integrate all the customer's data, identify in real-time

STREAMING GRAPH DATA

Networks data rates:

- Gigabit ethernet: 81k – 1.5M packets per second
- Over 130 000 flows per second on 10 GigE ($< 7.7\ \mu$s)

Person-level data rates:

- 500M posts per day on Twitter (6k / sec)[1]
- 3M posts per minute on Facebook (50k / sec)[2]

We need to analyze only *changes* and not *entire* graph.

Throughput & latency trade off and expose different levels of concurrency.

---

1 www.internetlivestats.com/twitter-statistics/

2 www.jeffbullas.com/2015/04/17/21-awesome-facebook-facts-and-statistics-you-need-to-check-out/

INCREMENTAL PAGERANK

Everyone's "favorite" metric: PageRank.

- Stationary distribution of the *random surfer* model.
- Eigenvalue problem can be re-phrased as a linear system

$$\left(I - \alpha A^T D^{-1}\right) x = kv,$$

with

α	*teleportation constant*, much < 1
A	adjacency matrix
D	diagonal matrix of out degrees, with $x/0 = x$ (self-loop)
v	*personalization* vector, here $\mathbf{1}/\lvert V\rvert$
k	irrelevant scaling constant

- Amenable to analysis, etc.

INCREMENTAL PAGERANK

- Streaming data setting, update PageRank without touching the entire graph.
- Existing methods maintain databases of walks, *etc*.
- Let $A_\Delta = A + \Delta A$, $D_\Delta = D + \Delta D$ for the new graph, want to solve for $x + \Delta x$.
- Simple algebra:

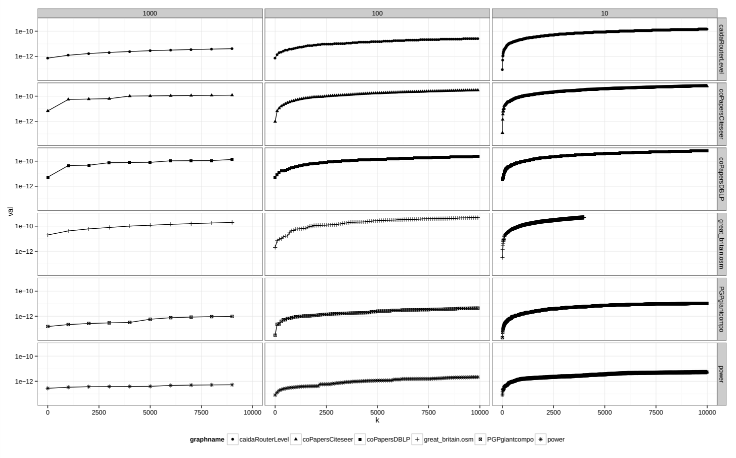
$$\left(I - \alpha A_\Delta^T D_\Delta^{-1}\right) \Delta x = \alpha \left(A_\Delta D_\Delta^{-1} - A D^{-1}\right) x,$$

  and the right-hand side is sparse.
- Re-arrange for Jacobi,

$$\Delta x^{(k+1)} = \alpha A_\Delta^T D_\Delta^{-1} \Delta x^{(k)} + \alpha \left(A_\Delta D_\Delta^{-1} - A D^{-1}\right) x,$$

  iterate, …

INCREMENTAL PAGERANK: WHOOPS



- And fail. The updated solution wanders away from the true solution. Top *rankings* stay the same…

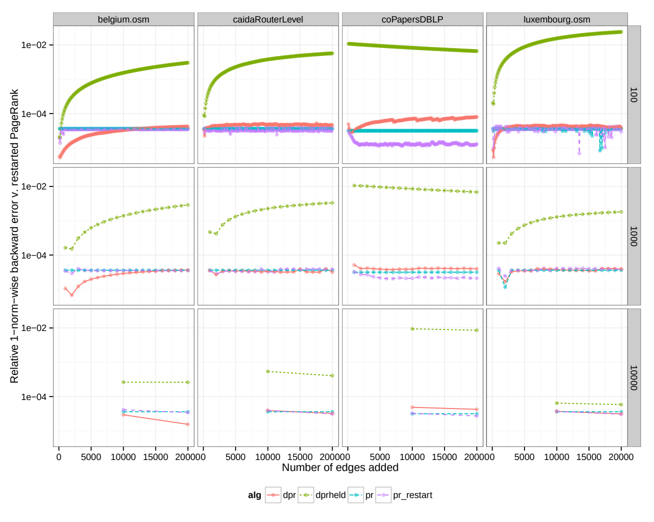
INCREMENTAL PAGERANK: *THINK* INSTEAD

- The old solution x is an ok, not exact, solution to the original problem, now a nearby problem.
- How close? Residual:

$$\begin{aligned} r' &= kv - x + \alpha A_{\Delta} D_{\Delta}^{-1} x \\ &= r + \alpha \left(A_{\Delta} D_{\Delta}^{-1} - A D^{-1} \right) x. \end{aligned}$$

- Solve $(I - \alpha A_{\Delta} D_{\Delta}^{-1}) \Delta x = r'$.
- Cheat by not refining *all* of r', only region growing around the changes.
- (Also cheat by updating r rather than recomputing at the changes.)

INCREMENTAL PAGERANK: WORKS

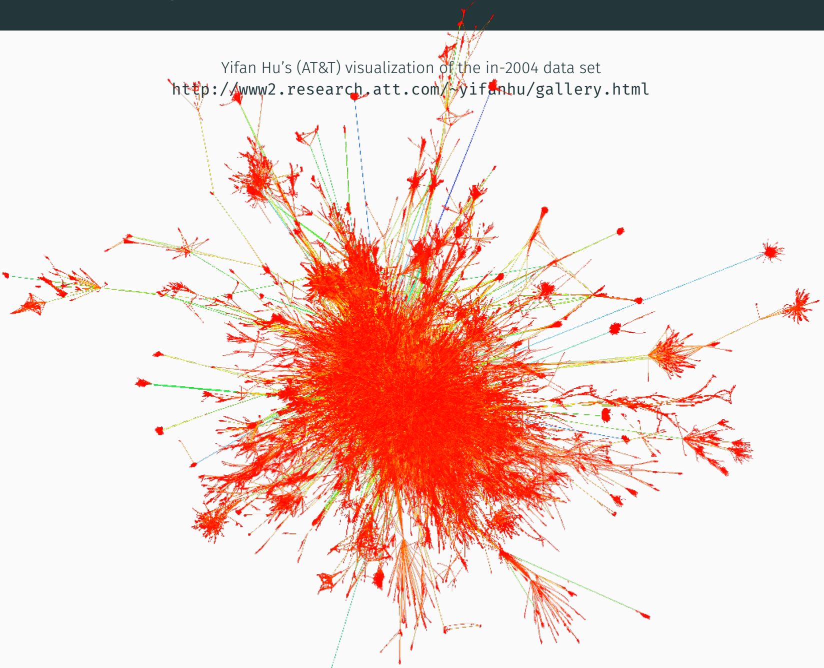


- Thinking about the numerical linear algebra issues can lead to better graph algorithms.

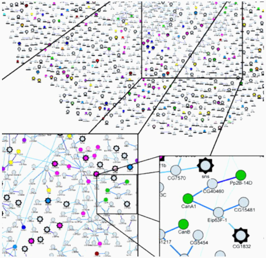
COMMUNITY DETECTION

GRAPHS: BIG, NASTY HAIRBALLS

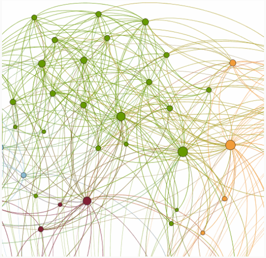
Yifan Hu's (AT&T) visualization of the in-2004 data set
`http://www2.research.att.com/~yifanhu/gallery.html`



BUT NO SHORTAGE OF STRUCTURE…



Protein interactions, Giot *et al.*, “A Protein Interaction Map of Drosophila melanogaster”, Science 302, 1722-1736, 2003.

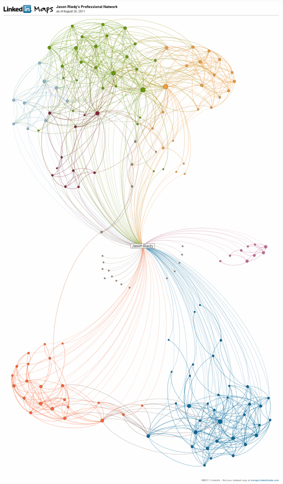


Jason’s network via LinkedIn Labs

- Locally, there are clusters or *communities*.
- Until 2011, no parallel method for *community detection*.
- But, gee, the problem looks familiar…

COMMUNITY DETECTION

- Partition a graph's vertices into disjoint communities.
- A community locally optimizes some metric, NP-hard.
- Trying to capture that vertices are *more similar* within one community than between communities.



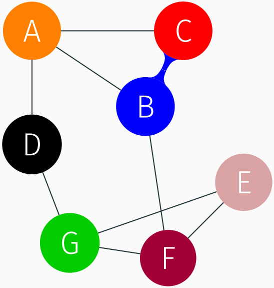
Jason's network via LinkedIn Labs

- **Modularity:** Deviation of connectivity in the community induced by a vertex set S from some expected background model of connectivity.
- Newman's uniform model, modularity of a cluster is

  fraction of edges in the community −
  fraction expected from uniformly sampling graphs
  with the same degree sequence

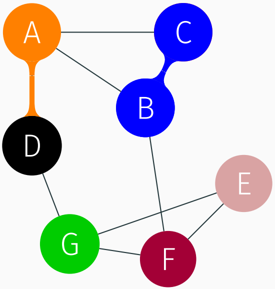
  $$Q_S = (m_S - x_S^2/4m)/m$$
- Modularity: sum of cluster contributions
- "Sufficiently large" modularity ⇒ *some* structure
- Known issues: Resolution limit, NP, *etc.*

SEQUENTIAL AGGLOMERATIVE METHOD



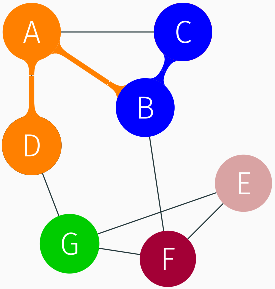
- A common method (*e.g.* Clauset, Newman, & Moore, 2004) *agglomerates* vertices into communities.
- Each vertex begins in its own community.
- An edge is chosen to contract.
  - Merging maximally increases modularity.
  - *Priority queue.*
- Known often to fall into an $O(n^2)$ performance trap with modularity (Wakita & Tsurumi '07).

SEQUENTIAL AGGLOMERATIVE METHOD



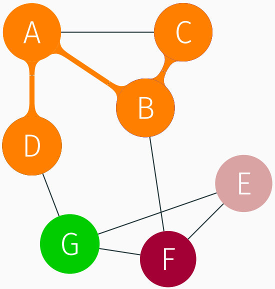
- A common method (*e.g.* Clauset, Newman, & Moore, 2004) *agglomerates* vertices into communities.
- Each vertex begins in its own community.
- An edge is chosen to contract.
    - Merging maximally increases modularity.
    - *Priority queue.*
- Known often to fall into an $O(n^2)$ performance trap with modularity (Wakita & Tsurumi '07).

SEQUENTIAL AGGLOMERATIVE METHOD



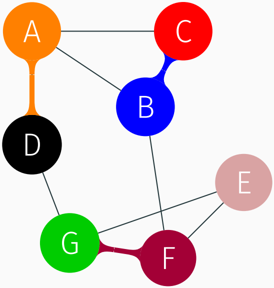
- A common method (*e.g.* Clauset, Newman, & Moore, 2004) *agglomerates* vertices into communities.
- Each vertex begins in its own community.
- An edge is chosen to contract.
  - Merging maximally increases modularity.
  - *Priority queue.*
- Known often to fall into an $O(n^2)$ performance trap with modularity (Wakita & Tsurumi ’07).

SEQUENTIAL AGGLOMERATIVE METHOD



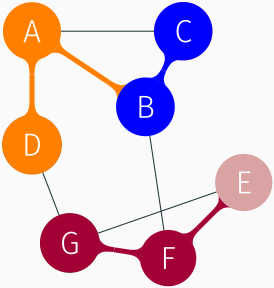
- A common method (*e.g.* Clauset, Newman, & Moore, 2004) *agglomerates* vertices into communities.
- Each vertex begins in its own community.
- An edge is chosen to contract.
    - Merging maximally increases modularity.
    - *Priority queue.*
- Known often to fall into an $O(n^2)$ performance trap with modularity (Wakita & Tsurumi '07).

PARALLEL AGGLOMERATIVE METHOD



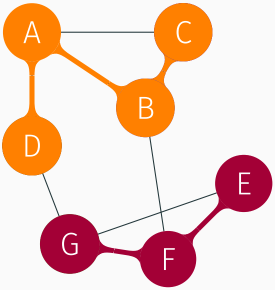
- Use a matching to avoid the queue.
- Compute a heavy weight matching.
  - Simple greedy, maximal algorithm.
  - Within factor of 2 from heaviest.
- Merge all communities at once.
- Maintains some balance.
- *Produces different results.*
- Agnostic to weighting, matching
- Up until 2011, no one tried this...

PARALLEL AGGLOMERATIVE METHOD



- Use a matching to avoid the queue.
- Compute a heavy weight matching.
  - Simple greedy, maximal algorithm.
  - Within factor of 2 from heaviest.
- Merge all communities at once.
- Maintains some balance.
- *Produces different results.*
- Agnostic to weighting, matching
- Up until 2011, no one tried this…

PARALLEL AGGLOMERATIVE METHOD



- Use a matching to avoid the queue.
- Compute a heavy weight matching.
  - Simple greedy, maximal algorithm.
  - Within factor of 2 from heaviest.
- Merge all communities at once.
- Maintains some balance.
- *Produces different results.*
- Agnostic to weighting, matching
- Up until 2011, no one tried this…

PARALLEL AGGLOMERATIVE COMMUNITY DETECTION

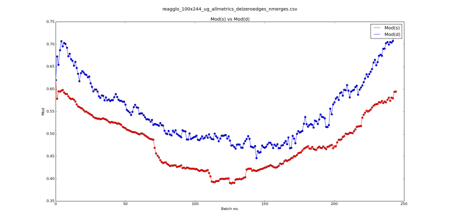
Graph	$\lvert V \rvert$	$\lvert E \rvert$	Reference
soc-LiveJournal1	4 847 571	68 993 773	"SNAP"
uk-2007-05	105 896 555	3 301 876 564	Ubicrawler

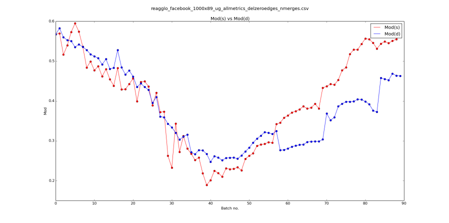
Peak processing rates in edges/second:

Platform	Mem	soc-LiveJournal1	uk-2007-05
E7-8870	256GiB	6.90×10^6	6.54×10^6
XMT2	2TiB	1.73×10^6	3.11×10^6

Clustering: Sufficiently good. Won 10^{th} DIMACS Implementation Challenge's mix category in 2012. Later: Fagginger Auer & Bisseling (2012), add *star* detection. LaSalle and Karypis (2014), add m-l "refinement."

WHAT ABOUT STREAMING?





Data and plots from Pushkar Godbolé.

Preliminary experiments...

Simple re-agglomeration: Fast, decreasing modularity.

“Backtracking” appears to work, but carries more data (see also Görke, *et al.* at KIT).

Clusterings are very sensitive.

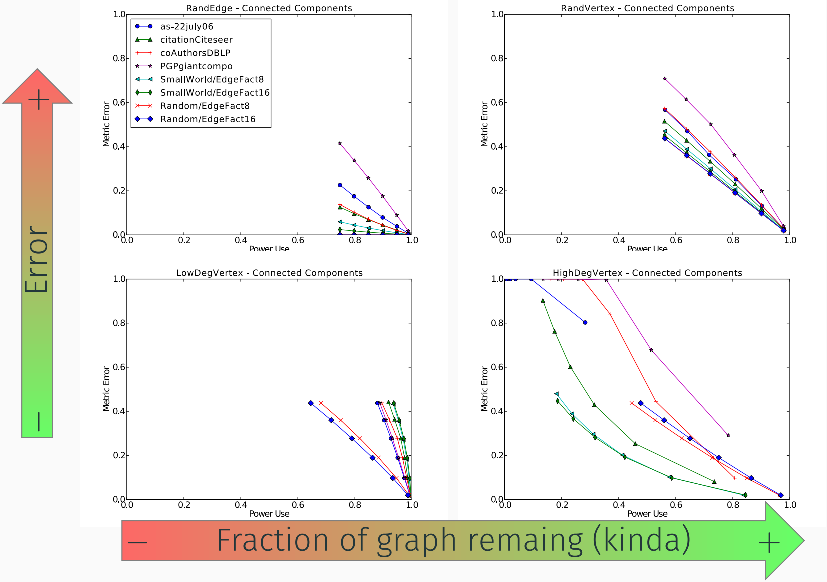
CONNECTED COMPONENTS

SENSITIVITY AND COMPONENTS

- Ok, clusterings are optimizing over a bumpy surface, *of course* they're sensitive... (Streaming exacerbates.)
- Pick a clean problem: connected components
- Where could errors occur?
    - Streaming: Dropped, forgotten information
    - Computing: Stop for energy or time, thresholds
    - Real-life: Surveys not returned
- How do you even *measure* errors?
    - Pairwise co-membership counts
    - Empirical distributions from vertex membership
    - No one measure...
    - All need the true solution...

SENSITIVITY OF CONNECTED COMPONENTS

From Zakrzewska & Bader, “Measuring the Sensitivity of Graph Metrics to Missing Data,” PPAM 2013



QUESTIONS / FUTURE

Can graph analysis learn from linear algebra and numerical analysis?

- Are there relevant concepts of backward error?
  - Don't need the true solution to evaluate (or estimate) some distance.
- Should graph analysis look more in the statistical direction?
  - *Moderate* graphs hit converging limits.
- Are there other easy analogies / low-hanging fruit?
- Environments for playing with large graphs?
  - Sane threading and atomic operations (not data)

Feel free to join in...

ACKNOWLEDGEMENTS

HPC LAB PEOPLE

Faculty:

- David A. Bader
- Oded Green (was student)

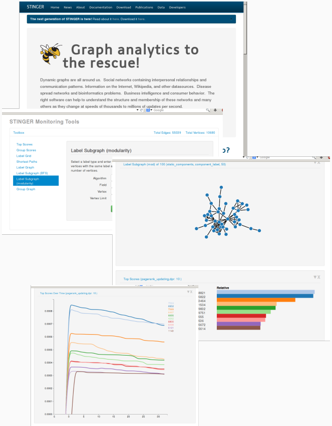
Data:

- Pushkar Godbolé
- Anita Zakrzewska

STINGER:

- Robert McColl,
- James Fairbanks,
- Adam McLaughlin,
- Daniel Henderson,
- David Ediger (now GTRI),
- Jason Poovey (GTRI),
- Karl Jiang, and
- *feedback from users in industry, government, academia*

STINGER: WHERE DO YOU GET IT?



Home: www.cc.gatech.edu/stinger/

Code: git.cc.gatech.edu/git/u/eriedy3/stinger.git/

Gateway to

- code,
- development,
- documentation,
- presentations...

Remember: Academic code, but maturing with contributions.

Users / contributors / questioners: Georgia Tech, PNNL, CMU, Berkeley, Intel, Cray, NVIDIA, IBM, Federal Government, Ionic Security, Citi, ...