Pacific Northwest NATIONAL LABORATORY

Proudly Operated by Battelle Since 1965

Large Scale Graph Analytics and Randomized Algorithms for Applications in Cybersecurity

EMILIE HOGAN, JOHN R. JOHNSON*, MAHANTESH HALAPPANAVAR

Pacific Northwest National Laboratory

SIAM CSE February, 2013

Outline

- Problem statement
  - “Pass the hash”
  - Network model
  - Our questions and goals
- Matrix sparsification
- Graph Minors
- Performance

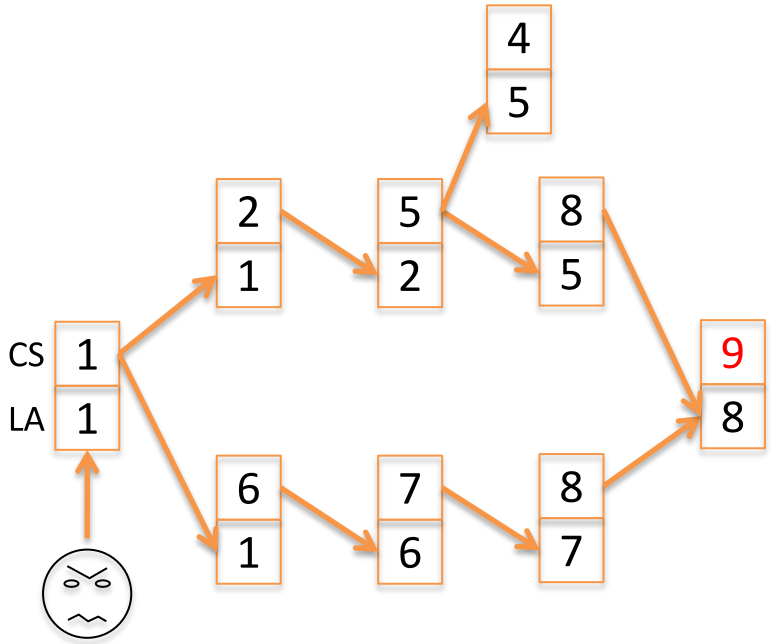
Outline

- Problem statement
  - “Pass the hash”
  - Network model
  - Our questions and goals
- Matrix sparsification
- Graph Minors
- Performance

“Pass the Hash” Hacking Technique

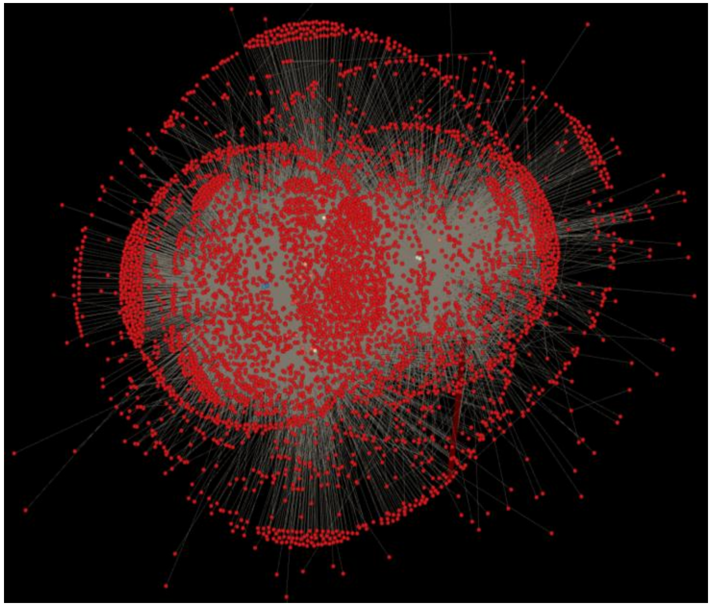
Pacific Northwest
NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

- Adversaries enter a network and obtain local administrator (LA) status on a computer.
- Can access the credential store (CS) and steal any credentials left on the computer.
- Use stolen credentials to log into other computers with LA status.
- Repeat until they obtain a high enough credential to log into any computer in the network and control it (domain controller).



Maintaining a Network

- Given a snapshot in time of a computer network including local administrator and credential store data
  - What are all the paths an adversary could take?
  - Can we quantify the risk level of the network?
- Given a stream of network data
  - Answer the above questions in real-time
  - Identify adversaries as they make their attack



Network model and questions

- Model network as a graph
  - Vertices are IP addresses
  - Detection graph
    - Edges indicate when an event takes place
  - Reachability graph
    - Edges indicate common credential between two computers
    - For a given set of credentials, what are all the paths that could lead to that credential
    - Constraints on the graph require the communicating system to use a credential that has local administrator privilege on the target machine
- Static graph
  - Take all data from a time period (e.g., one day) and look at that graph
- Evolving graph
  - As events occur edges are created
  - When credentials expire the edge is removed
- Risk metric / Cross section
  - For a randomly selected node in the network, what is the probability having a path to a certain credential?
  - How does this number change over time (i.e. as hashes expire in the credential store, and new credentials are deposited?
- **Can signatures of path traversal along the reachability graph be detected in existing data?**

What we are looking for

- Find paths from outside a network to high level computer
- Too many paths = network at risk
- How to find paths
  - Use graph adjacency matrix, A
  - A^k counts walks of length k between all pairs of vertices

A^k counts walks of length k in the graph

Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

$$(A^k)_{i,j} = \left(\underbrace{\begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,n} \end{pmatrix}}_{= A^{k-1}} \cdot \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \right)_{i,j}$$

$$= (w_{i,1} \quad w_{i,2} \quad \cdots \quad w_{i,n}) \cdot \begin{pmatrix} a_{1,j} \\ a_{2,j} \\ \vdots \\ a_{n,j} \end{pmatrix} = w_{i,1}a_{1,j} + w_{i,2}a_{2,j} + \cdots + w_{i,n}a_{n,j}$$

$$= \sum_{\ell=1}^{n} w_{i,\ell}a_{\ell,j}$$

Number of walks of length $k-1$ from i to ℓ ($w_{i,\ell}$) times number of edges from ℓ to j ($a_{\ell,j}$) yields the number of walks of length k from i to j in which the second to last vertex in the walk is ℓ.

What we are looking for

- Find paths from outside a network to high level computer
- Too many paths = network at risk
- How to find paths
  - Use graph adjacency matrix, A
  - A^k counts walks of length k between all pairs of vertices
  - Use symbolic adjacency matrix:

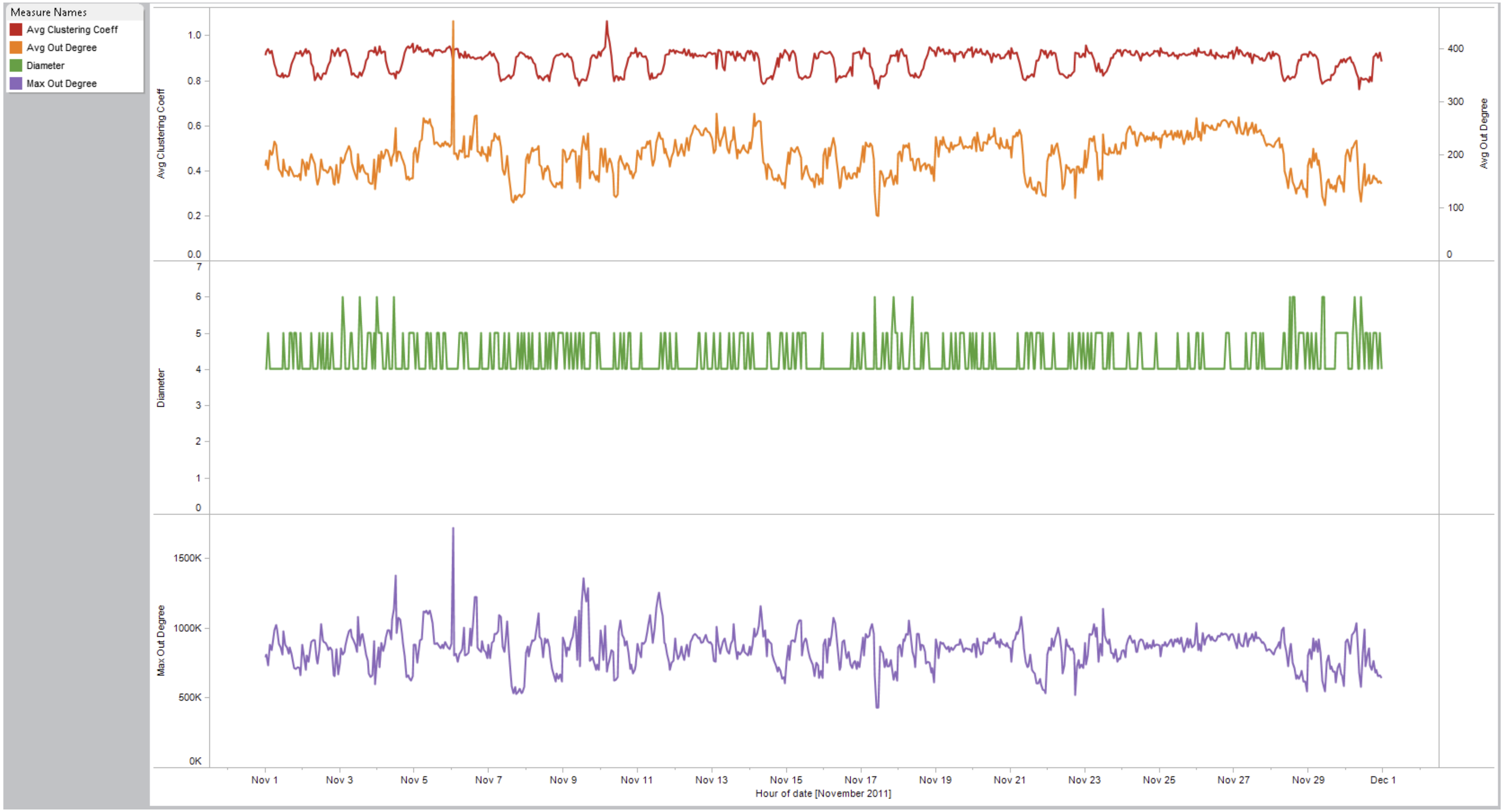
$$S = (s_{i,j}) \text{ where } s_{i,j} = \begin{cases} x_{i,j} & \text{if } (i,j) \text{ is an edge} \\ 0 & \text{otherwise} \end{cases}$$

    Then S^k keeps track of what the walks are
  - $W_k(G) = \sum_{i=1}^{k} A^i$ is a matrix which counts walks of length $\leq k$ **(recall for later)**
  - $\sum_{i=1}^{k} S^i$ keeps track of the walks of length $\leq k$
    - Takes up a lot of memory

Our data

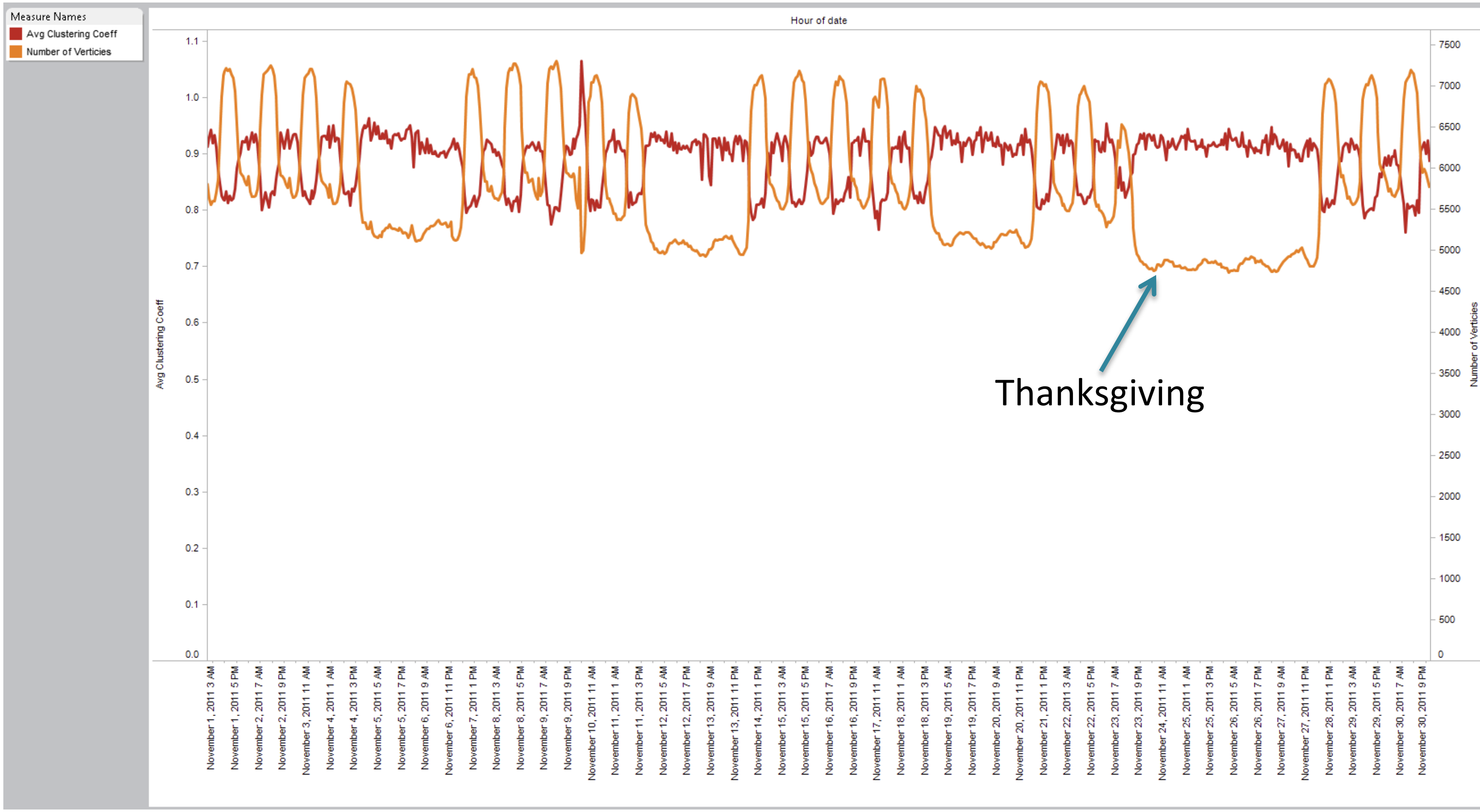
- Have network traffic data in the form of Windows event logs
  - Source IP
  - Host IP
  - Event ID (logon, logoff, error, password change, …)
  - Timestamp
  - Username
  - Etc.
- One day of network data
  - Nodes – |V| = 4,661
    - Including perimeter data can introduce millions of vertices
  - Edges – |E| = 15,466
    - Began with 4,433,142 events and threw away parallel edges
  - Average degree = 6.6
  - Network diameter = 7

Clustering coefficient, average/max out degree, and diameter



Clustering coefficient vs. Number of vertices

Pacific Northwest
NATIONAL LABORATORY
Proudly Operated by **Battelle** *Since 1965*



Outline

- Problem statement
  - “Pass the hash”
  - Network model
  - Our questions and goals
- Matrix sparsification
- Graph Minors
- Performance

Matrix Sparsification – version 1

- Input
  - $A \in \mathbb{R}^{m\times n}, B \in \mathbb{R}^{n\times p}$, constant $1 \le c \le n$, probability distribution $\{p_i\}_{i=1}^{n}$
- Output
  - $C \in \mathbb{R}^{m\times c}$ (columns selected from A), $R \in \mathbb{R}^{c\times p}$ (rows selected from B)
- Procedure
  - For $t = 1, \dots, c$ choose $i_t \in \{1, \dots, n\}$ with probability $P(i_t = k) = p_k$ independently *with replacement*
  - Let $C_{j,t} = \frac{A_{j,i_t}}{\sqrt{c\, p_{i_t}}}$ for $j = 1, \dots, m$ and $R_{t,k} = \frac{B_{i_t,k}}{\sqrt{c\, p_{i_t}}}$ for $k = 1, \dots, p$
    - Column t of C is multiple of column i_t of A, row t of R is multiple of row i_t of B
- Assuming we chose good p_i, the resulting $C \cdot R$ can provide a good approximation for $A \cdot B$

Drineas, Petros; Kannan, Ravi; and Mahoney, Michael W.: (2006) *Fast Monte Carlo Algorithms for Matrices I: Approximating Matrix Multiplication*, SIAM Journal on Computing, vol. 36(1): pp. 132 – 157

Matrix Sparsification – version 1 (cont.)

- Approximating $A \cdot B$ with $C \cdot R$
  - Assuming nearly optimal probabilities (β depends on $\{p_i\}$)

$$\mathbb{E}[\|AB - CR\|_F^2] \leq \frac{1}{\beta c}\|A\|_F^2\|B\|_F^2$$

  - For $\delta \in (0,1), \eta = 1 + \sqrt{\frac{8}{\delta}\log\frac{1}{\delta}}$ then with probability $1 - \delta$:

$$\|AB - CR\|_F^2 \leq \frac{\eta^2}{\beta c}\|A\|_F^2\|B\|_F^2$$

- Using matrix sparsification technique won't allow for approximating odd matrix powers
  - If $A = B$ is $n \times n$ then C is $n \times c$, and R is $c \times n$
  - $C \cdot R$ is $n \times n$, but multiplying again by C yields an $n \times c$ matrix

Matrix Sparsification – version 2

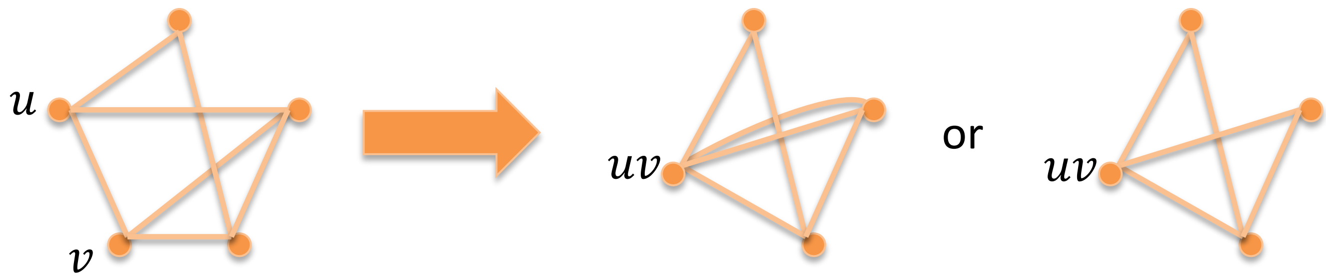
- Input
  - $A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times p}$, constant $1 \leq c \leq n$, probability distributions $\{p_{ij}\}_{i,j=1}^{m,n}$ and $\{q_{ij}\}_{i,j=1}^{n,p}$
- Output
  - $S \in \mathbb{R}^{m \times n}$, $R \in \mathbb{R}^{n \times p}$
- Procedure
  - Select elements from A using probability distribution p (elements of S are either A_{ij}/p_{ij} or 0)
  - Select elements from B using probability distribution q (elements of R are either B_{ij}/q_{ij} or 0)
- This is equivalent to throwing away edges of a graph G whose adjacency matrix is $A = B$ and then reweighting those edges that remain.
- Removes some paths of interest

Outline

- Problem statement
  - “Pass the hash”
  - Network model
  - Our questions and goals
- Matrix sparsification
- Graph Minors
- Performance

Graph Minors

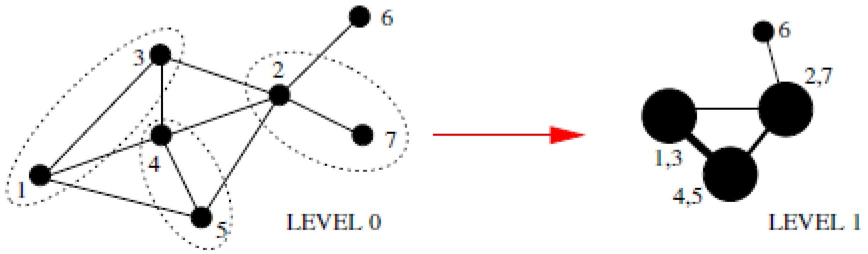
- Given a graph, G, and a pair of adjacent vertices, $u, v \in V(G)$, we form the minor, $G^{(u,v)}$, by
  - Removing u, v from the vertex set
  - Adding new vertex uv
  - Replacing all edges (x, u) and (y, v) where $x, y \neq u, v$ with $(x, uv), (y, uv)$



  - Do not create loop, i.e., $(uv, uv) \notin E(G^{(u,v)})$
- In undirected graph, **paths are preserved** under minor operation
- Lose information about two vertices after each minor operation

Relationship to coarsening

- Similar to the *strict aggregation (SAG)* scheme for multilevel graph partitioning
  - Vertices partitioned into disjoint groups based on edge weights within and between partitions
  - All vertices in partition contracted into a single vertex



Representation of SAG scheme from Chevallier, Safro 2009

- We contract one edge at a time

Chevallier, Cédric; and Safro, Ilya. Comparison of coarsening schemes for multilevel graph partitioning. In Thomas Stützle, editor, Learning and Intelligent Optimization, volume 5851 of Lecture Notes in Computer Science, pages 191–205. Springer-Verlag, 2009.

"Sparse" minors

- Goal is to get smaller adjacency matrix and use well known dense matrix multiply algorithms
- Find "sparse pair" of adjacent vertices to contract
  - Vertices u, v such that $deg(u) + deg(v)$ is small and $(u, v) \in E(G)$

$$A = \left(\begin{array}{ccc|cc} & & & a_{x,u} & a_{x,v} \\ & A' & & \vdots & \vdots \\ & & & a_{y,u} & a_{y,v} \\ \hline a_{u,x} & \cdots & a_{u,y} & 0 & 1 \\ a_{v,x} & \cdots & a_{v,y} & 1 & 0 \end{array}\right)$$

$$A^{(u,v)} = \left(\begin{array}{ccc|c} & & & a_{x,u} + a_{x,v} \\ & A' & & \vdots \\ & & & a_{y,u} + a_{y,v} \\ \hline a_{u,x} + a_{v,x} & \cdots & a_{u,y} + a_{v,y} & 0 \end{array}\right)$$

Here can replace "+" with "max"

- Do M edge contractions to yield graph G_M with adjacency matrix A_M

Outline

- Problem statement
  - “Pass the hash”
  - Network model
  - Our questions and goals
- Matrix sparsification
- Graph Minors
- Performance

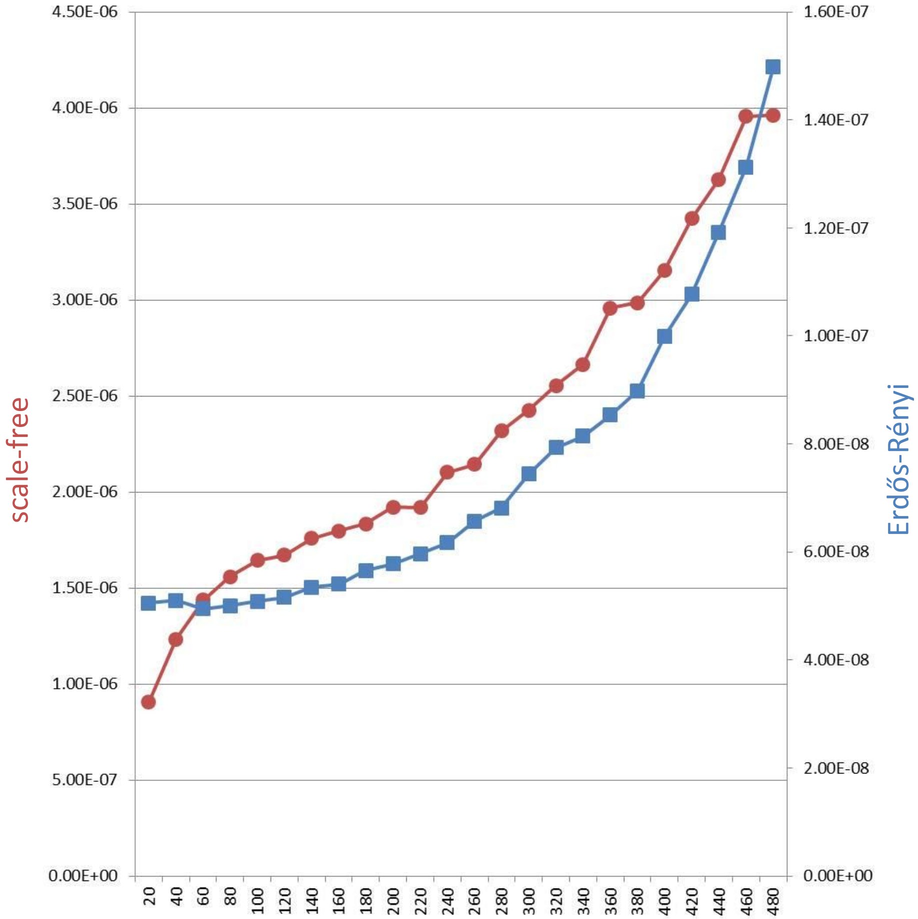
Measures of accuracy

- After taking minors, compute $W_k(G)$ and $W_k(G_M)$
- There is a set of vertices, V_M, that are common to both G and G_M
  - Vertices in G which were not removed by an edge contraction
  - Vertices in G_M which were not created as a result of an edge contraction
- Compare sub-matrices restricted only to the vertices in V_M
- Define

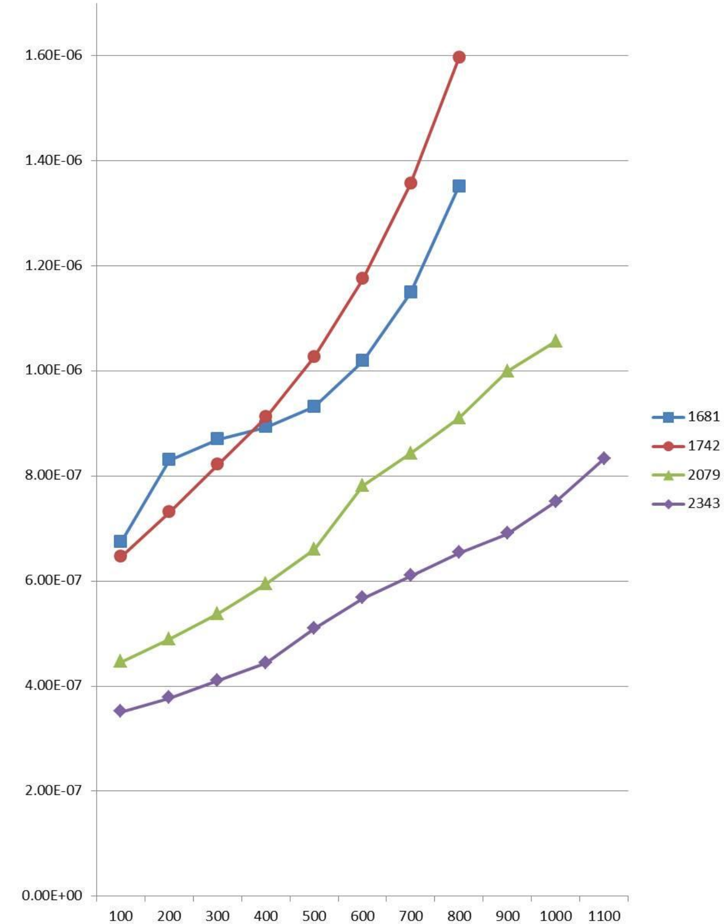
$$D_{k,M} := \left.\frac{W_k(G)}{\|W_k(G)\|_1}\right|_{V_M} - \left.\frac{W_k(G_M)}{\|W_k(G_M)\|_1}\right|_{V_M}$$

- The (i,j) entry of $D_{k,M}$ is the number of walks of length k from i to j in G as a percentage of the total number of walks of length k minus the same quantity for G_M.
  - $\|\cdot\|_1$ = the L_1 norm of the matrix (the sum of its entries)

Data



Average of the absolute values of the entries of $D_{20,\,M}$, with $m = 20, 40, \ldots, 480$, for an Erdős-Rényi ($p = 0.5$) and scale-free random graph with $|V| = 1000$.



Average of the absolute values of the entries of $D_{10,M}$, with $M = 100, 200, \ldots, |V|/2$, for randomly chosen induced subgraphs of our cybersecurity graphs with $|V|$ values as indicated.

Observations and Future work

- Performance of minors algorithm
  - Poor performance on full comparison – total number of walks
    - Fewer vertices means fewer walks
  - Appears to be good approximation for portion of total walks
- Future plans for pass-the-hash
  - How does the graph spectrum change when you take repeated minors?
  - Minors in directed graphs
  - Can we use minors to approximate all pairs shortest paths?
  - Make symbolic adjacency matrix less memory intensive
- General graph signature plans
  - Goal to generalize the process of finding graph-based signatures
  - Looking for more applications and we are on the lookout for data!

Acknowledgements

- GRADIENT team
  - Daniel Best, Satish Chikkagoudar, Sutanay Choudhury, Glenn Fink, Mahantesh Halappanavar, Peter Hui, John Johnson, Chaomei Lo, Bill Nickless, Bryan Olsen, and Elena Peterson
- Nathan Baker – SDI lead
- George Bonheyo – Dynamics and Detection Area lead