Anomaly Detection
Graph data applications

Paul Irofti
Cristian Rusu
Andrei Pătraşcu

Computer Science Department
University of Bucharest

Outline

- Graphs. Example of applications and anomalies.
- Anomalies in static graphs. Detection methods.
- Anomalies in dynamic graphs. Detection methods.

Graphs $G = (V, E)$ have the ability to model relations in:

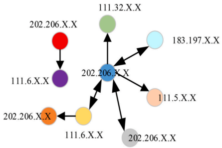
- Computer Networks
- Financial Networks
- Social Networks
- etc.

Problem

Given the graph topology (and attributes), detect unusual connectivity or other abnormal structures.

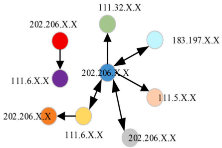
V is (possibly) the set of IP adresses. E is the set of connections (flows) between IPs. Anomalies are intrusions or attacks such as:

- port-scan: a particular node has a very high out-degree.



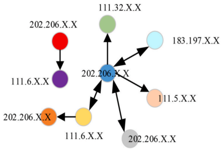
V is (possibly) the set of IP adresses. E is the set of connections (flows) between IPs. Anomalies are intrusions or attacks such as:

- port-scan: a particular node has a very high out-degree.
- DDoS: many nodes have connections towards the same destination.



V is (possibly) the set of IP adresses. E is the set of connections (flows) between IPs. Anomalies are intrusions or attacks such as:

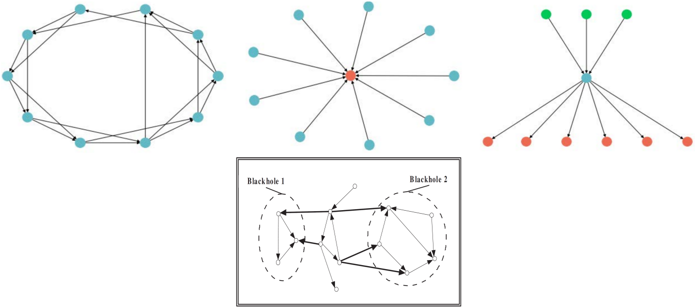
- port-scan: a particular node has a very high out-degree.
- DDoS: many nodes have connections towards the same destination.



The edges might support multiple attributes: flow duration, nr. packets, etc.

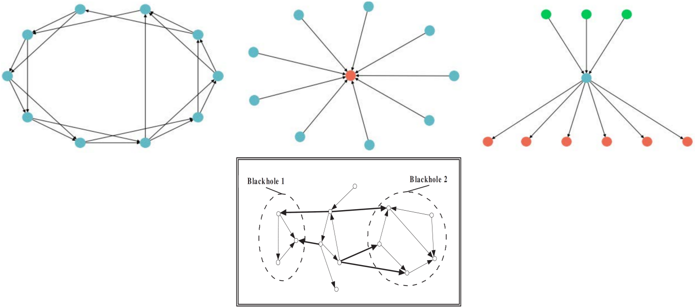
V is the set of bank clients. E is the set of weighted transactions between two clients. Anomalies as frauds reflected as abnormal patterns in the financial transactions graph:

- ring: the amount that is transfered over a cycle is nearly constant.



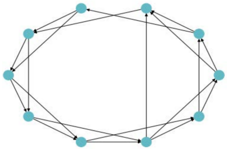
V is the set of bank clients. E is the set of weighted transactions between two clients. Anomalies as frauds reflected as abnormal patterns in the financial transactions graph:

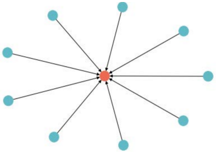
- ring: the amount that is transfered over a cycle is nearly constant.
- star: suspiciously many clients make transactions toward a common single destination.

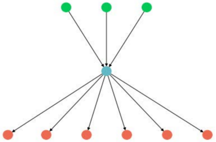


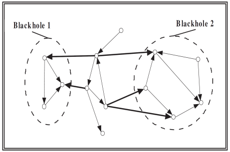
V is the set of bank clients. E is the set of weighted transactions between two clients. Anomalies as frauds reflected as abnormal patterns in the financial transactions graph:

- ring: the amount that is transfered over a cycle is nearly constant.
- star: suspiciously many clients make transactions toward a common single destination.
- blackholes: nodes with high in-traffic (directed graphs).



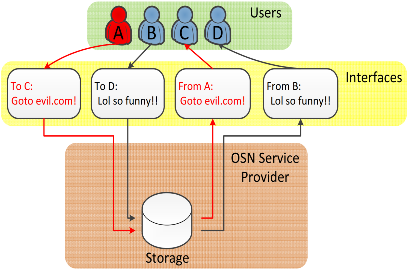




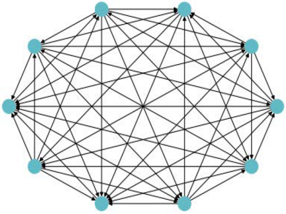


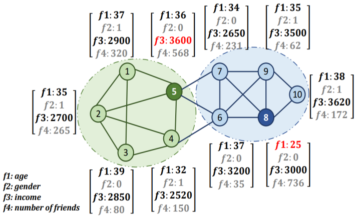
V is the set of clients. E is the set of connections/messages between clients. Anomalies arise as malware/spam messages sent toward correct users, that directs the user toward a malitious sites.

- Simple solution: filtering an aggregated set of messages over a finite time period.



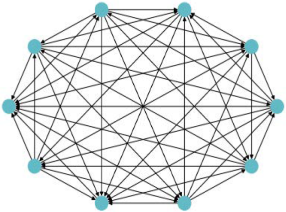
1. Choose the best grah model that meets the data is modeled by:
   - **Static (attributed) graphs**
   - Dynamic (attributed) graphs (edge streams)
   - Multi-graphs

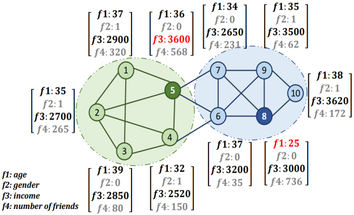




2. Define a specific ”normality” principle:

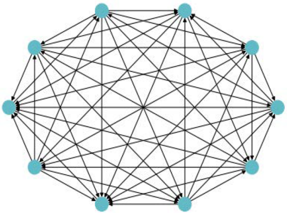
- Local structure: the neighborhood of normal vertices is almost uniform over the entire graph.

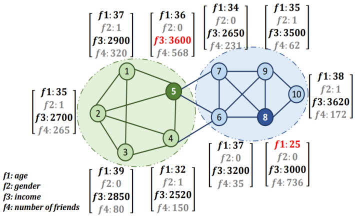




Modelling principles

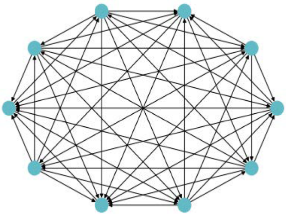
2. Define a specific "normality" principle:
   - Local structure: the neighborhood of normal vertices is almost uniform over the entire graph.
   - Similar weights: the nodes have similar weights on out-edges.

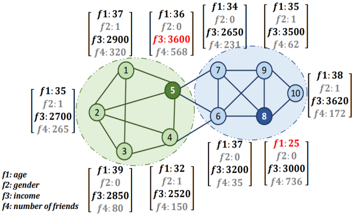




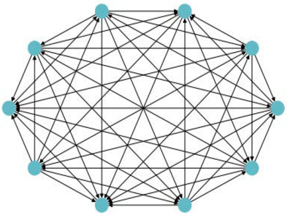
Modelling principles

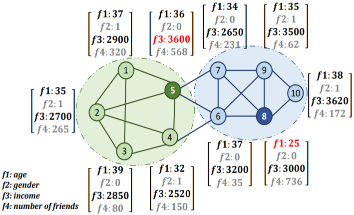
2. Define a specific "normality" principle:
   - Local structure: the neighborhood of normal vertices is almost uniform over the entire graph.
   - Similar weights: the nodes have similar weights on out-edges.
   - Uniformity: subgraphs of certain dimension are similar.





2. Define a specific ”normality” principle:
   - Local structure: the neighborhood of normal vertices is almost uniform over the entire graph.
   - Similar weights: the nodes have similar weights on out-edges.
   - Uniformity: subgraphs of certain dimension are similar.
   - Proximity: normal objects stay close on the graph.



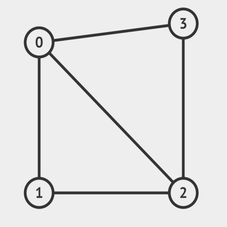


3. Design the algorithm based on the assumed ”normality”:
   - Feature extraction + Outlier Detection Method: (*i*) various features are extracted from data in order to convert the graph topology into node attributes; (*ii*) and then usual outlier detection methods are applied.

3. Design the algorithm based on the assumed ”normality”:
   - Feature extraction + Outlier Detection Method: (*i*) various features are extracted from data in order to convert the graph topology into node attributes; (*ii*) and then usual outlier detection methods are applied.
   - Matrix Factorization: the adjacency matrix is splited into low-rank factor and sparse residual factor. The structure of residuals reveals possible outliers.

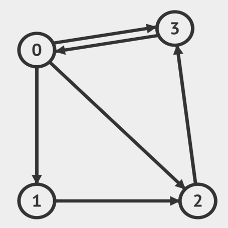
3. Design the algorithm based on the assumed ”normality”:
    - Feature extraction + Outlier Detection Method: (*i*) various features are extracted from data in order to convert the graph topology into node attributes; (*ii*) and then usual outlier detection methods are applied.
    - Matrix Factorization: the adjacency matrix is splited into low-rank factor and sparse residual factor. The structure of residuals reveals possible outliers.
    - Deep graph learning: the graph topology is aggregated and encoded through deep learning layers of neural networks.

The main matrices associated to a given graph $G = (V, E)$ are adjacency matrix, incidence matrix and Laplacian matrix.
Undirected case:



$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}, L = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}.$$

The main matrices associated to a given graph $G = (V, E)$ are adjacency matrix, incidence matrix and Laplacian matrix.
Directed case:

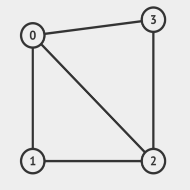


$$L_{out} = \begin{bmatrix} 3 & -1 & -1 & -1 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 1 \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -1 & 0 & -1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 1 & 1 \end{bmatrix}.$$

The main matrices associated to a given graph $G = (V, E)$ are adjacency matrix, incidence matrix and Laplacian matrix.

Spectrum of Laplacian $\Lambda(L) = \{\lambda_1(L), \cdots, \lambda_n(L)\}$:



$$\Lambda(L) = \begin{bmatrix} 4 \\ 4 \\ 2 \\ 0 \end{bmatrix} \text{ (undirected case)}$$

- $\lambda_n(L)$ always 0 (the sum on the lines is 0);
- the multiplicity of null eigenvalues = number of components;
- $\lambda_{n-1}(L)$ is the Fiedler value (algebraic connectivity); the further from 0, the more connected.

Feature extraction: take the graph $G = (V, E)$ and for each node v_i assign a feature vector based on the v_i's connectivity within G.

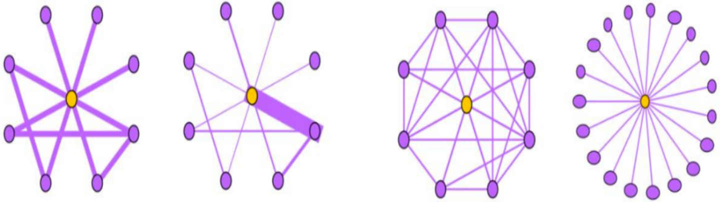
Examples fo features:

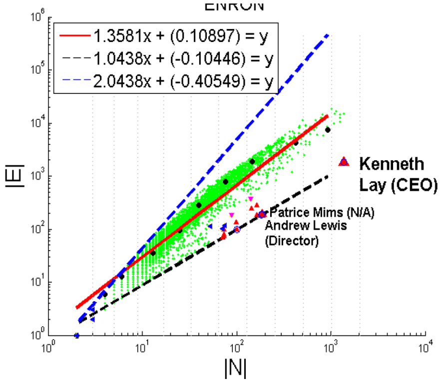
- N_i: in - degree / out - degree
- E_i : nr. of edges in egonet
- W_i : total weight of neighborhood
- λ_i : principal eigenvalues of egonet i.

Feature extraction: take the graph $G = (V, E)$ and for each node v_i assign a feature vector based on the v_i's connectivity within G.

Oddball: choose pairs of features and plot to find patterns of normal behaviour

- N *vs.* E: detect clique and stars
- W *vs.* E: detect heavy vicinity
- λ *vs.* W: detect single dominating edge





- red: LS to median values
- blue: clique
- black: stars

PCA formulation: Let $A \in \mathbb{R}^{n \times n}$ be the data matrix. Then

$$\min_{F,G} \|A - FG^T\|_F^2$$

reveal a low-rank approximation of data.

- FG^T is the best low-rank approximation (encodes maximum of information)
- The residual $R = A - FG^T$ is the error reconstruction of data points.

Non-negative Residual Matrix Factorization (NrMF)

Data: An adjacency matrix of a bipartite graph $A \in \mathbb{R}^{n \times l}$, rank size r;
Find: The low-rank approximation structure $F \cdot G$ and the residual matrix R such that:

(*i*) $A \approx F \cdot G$;

(*ii*) $R = A - F \cdot G$;

(*iii*) $R_{ij} \geq 0$ for all $A_{ij} > 0$.

- matrix R is an indicator for anomalies in graphs

Non-negative Residual Matrix Factorization (NrMF)

Data: An adjacency matrix of a bipartite graph $A \in \mathbb{R}^{n \times l}$, rank size r;
Find: The low-rank approximation structure $F \cdot G$ and the residual matrix R such that:

(*i*) $A \approx F \cdot G$;

(*ii*) $R = A - F \cdot G$;

(*iii*) $R_{ij} \geq 0$ for all $A_{ij} > 0$.

- matrix R is an indicator for anomalies in graphs
- R_{ij} is constrained to be positive when there is an edge between the nodes i and j

Non-negative Residual Matrix Factorization (NrMF)

Data: An adjacency matrix of a bipartite graph $A \in \mathbb{R}^{n \times l}$, rank size r;
Find: The low-rank approximation structure $F \cdot G$ and the residual matrix R such that:

(i) $A \approx F \cdot G$;

(ii) $R = A - F \cdot G$;

(iii) $R_{ij} \geq 0$ for all $A_{ij} > 0$.

- matrix R is an indicator for anomalies in graphs
- R_{ij} is constrained to be positive when there is an edge between the nodes i and j
- a large entry in R reflects strage interaction between two objects (e.g. DDoS attack)

Formulation 1:

$$\begin{aligned}\min_{F,G,R} \sum_{(i,j):A_{ij}>0} &\|R_{ij}\|^2 \\ \text{s.t.}\ \ & R_{ij} \geq 0 \quad \forall (i,j) : A_{ij} > 0 \\ & R = A - F \cdot G.\end{aligned}$$

Equivalently

$$\begin{aligned}\min_{F,G} \sum_{(i,j):A_{ij}>0} &(A_{ij} - F_{i:} \cdot G_{:j})^2 \\ \text{s.t.}\ \ & F_{i:} \cdot G_{:j} \leq A_{ij} \quad \forall (i,j) : A_{ij} > 0.\end{aligned}$$

If we reduce the rank to 1, then we have to solve

$$\begin{aligned}\min_{f,g} \sum_{(i,j):A_{ij}>0} &(A_{ij} - f_i g_j)^2 \\ \text{s.t.}\ \ & f_i g_j \leq A_{ij} \quad \forall (i,j) : A_{ij} > 0.\end{aligned}$$

$$\min_{F,G} \sum_{(i,j):A_{ij}>0} (A_{ij} - F_{i:} \cdot G_{:j})^2$$
$$\text{s.t. } F_{i:} \cdot G_{:j} \leq A_{ij} \quad \forall (i,j) : A_{ij} > 0.$$

Challenges:

- the problem formulation is not convex w.r.t. jointly F and G (we do not seek global solution)

$$\min_{F,G} \sum_{(i,j):A_{ij}>0} (A_{ij} - F_{i:} \cdot G_{:j})^2$$
$$\text{s.t. } F_{i:} \cdot G_{:j} \leq A_{ij} \quad \forall (i,j) : A_{ij} > 0.$$

Challenges:

- the problem formulation is not convex w.r.t. jointly F and G (we do not seek global solution)
- good news: the problem remains convex if we keep fixed one of the factor F, or G. Moreover, it is quadratic and it can be solved with standard solvers.

$$\min_{F,G} \sum_{(i,j):A_{ij}>0} (A_{ij} - F_{i:} \cdot G_{:j})^2$$
$$\text{s.t.} \;\; F_{i:} \cdot G_{:j} \leq A_{ij} \quad \forall (i,j) : A_{ij} > 0.$$

Challenges:

- the problem formulation is not convex w.r.t. jointly F and G (we do not seek global solution)
- good news: the problem remains convex if we keep fixed one of the factor F, or G. Moreover, it is quadratic and it can be solved with standard solvers.
- Alternating schemes arise naturally in this case.

Input: The original matrix A, rank size r

1. Initialize F and G
2. **While** Not convergent **do**
    1. Update: $G = UpdateFactor(A, F)$
    2. Update: $F = UpdateFactor(A', G')$
3. **endwhile**
4. **Output:** $R = A - FG$.

- Initialize the sequence $(F^k, G^k)_{k \geq 0}$ and update each factor.

Input: The original matrix A, rank size r

1. Initialize F and G
2. **While** Not convergent **do**
   1. Update: $G = \mathit{UpdateFactor}(A, F)$
   2. Update: $F = \mathit{UpdateFactor}(A', G')$
3. **endwhile**
4. **Output:** $R = A - FG$.

- Initialize the sequence $(F^k, G^k)_{k \geq 0}$ and update each factor.
- The main effort in each iteration is the call *UpdateFactor*.

Denote $\Omega(j) = \{i : A_{ij} > 0\}$

$$\min_{G} \sum_{j=1}^{n} \sum_{i\in\Omega(j)} (A_{ij} - F_{i:} \cdot G_{:j})^2 = \sum_{j=1}^{n} \|F_{\Omega(j):}G(:j) - A_{\Omega(j)j}\|_2^2$$
$$\text{s.t.} \;\; F_{\Omega(j):}G(:j) - A_{\Omega(j)j} \leq 0, \quad \forall j.$$

- the problem is separable over j:

$$G^*(:,j) = \arg\min_{G(:,j)} \|F_{\Omega(j):}G(:j) - A_{\Omega(j)j}\|_2^2$$
$$\text{s.t.} \;\; F_{\Omega(j):}G(:j) - A_{\Omega(j)j} \leq 0.$$

Denote $\Omega(j) = \{i : A_{ij} > 0\}$

$$\min_{G} \sum_{j=1}^{n} \sum_{i \in \Omega(j)} (A_{ij} - F_{i:} \cdot G_{:j})^2 = \sum_{j=1}^{n} \|F_{\Omega(j):}G(:j) - A_{\Omega(j)j}\|_2^2$$
$$\text{s.t.} \;\; F_{\Omega(j):}G(:j) - A_{\Omega(j)j} \leq 0, \quad \forall j.$$

- the problem is separable over j:

$$G^*(:,j) = \arg\min_{G(:,j)} \|F_{\Omega(j):}G(:j) - A_{\Omega(j)j}\|_2^2$$
$$\text{s.t.} \;\; F_{\Omega(j):}G(:j) - A_{\Omega(j)j} \leq 0.$$

- quadratic LS-type objective

Denote $\Omega(j) = \{i : A_{ij} > 0\}$

$$\begin{aligned} \min_{G} \sum_{j=1}^{n} \sum_{i \in \Omega(j)} (A_{ij} - F_{i:} \cdot G_{:j})^2 &= \sum_{j=1}^{n} \| F_{\Omega(j):} G(:j) - A_{\Omega(j)j} \|_2^2 \\ \text{s.t. } F_{\Omega(j):} G(:j) - A_{\Omega(j)j} &\leq 0, \quad \forall j. \end{aligned}$$

- the problem is separable over j:

$$\begin{aligned} G^*(:,j) = \arg \min_{G(:,j)} & \| F_{\Omega(j):} G(:j) - A_{\Omega(j)j} \|_2^2 \\ \text{s.t. } & F_{\Omega(j):} G(:j) - A_{\Omega(j)j} \leq 0. \end{aligned}$$

- quadratic LS-type objective
- linear inequality constraints

Input: The original matrix A, factor F

1. **For** $j = 1 : n$ **do**
   1. Compute: $a = A_{:j}, B = F(\Omega(j), :)$
   2. **For** $i = 1 : m$ **do**
      1. **if** $A_{ij} > 0$ **then** $v_i = A_{ij}$
      2. **else** $v_i = inf$
      3. **endif**
   3. **endfor**
   4. **Set:** $H = B^T B, q = -2B^T a, S = F$
   5. **Solve:** $G_{:j} = QPsolver(H, q, S, v)$.
2. **endfor**
3. **Output:** $R = A - FG$.

- The QP step is very costly and it is repeated n times

Input: The original matrix A, factor F

1. **For** $j = 1 : n$ **do**
    1. Compute: $a = A_{:j}, B = F(\Omega(j), :)$
    2. **For** $i = 1 : m$ **do**
        1. **if** $A_{ij} > 0$ **then** $v_i = A_{ij}$
        2. **else** $v_i = inf$
        3. **endif**
    3. **endfor**
    4. **Set:** $H = B^T B, q = -2B^T a, S = F$
    5. **Solve:** $G_{:j} = QPsolver(H, q, S, v)$.
2. **endfor**
3. **Output:** $R = A - FG$.

- The QP step is very costly and it is repeated n times
- Considering the complexity of a single *QPsolver* call to be $O(d^k)$ then the total cost of *UpdateFactor* is $O(mnr^2 + nr^k)$.

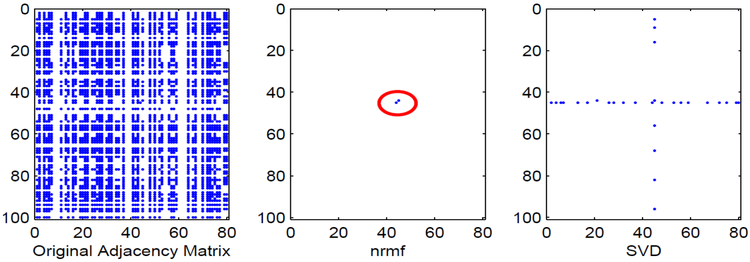
Idea to reduce the cost: instead of finding a rank-r approximation

$$\min_{F,G} \sum_{(i,j):A_{ij}>0} (A_{ij} - F_{i:}G_{:j})^2 \text{ s.t. } \textit{constraints}$$

find a rank-1 approximation at each iteration of the residual matrix

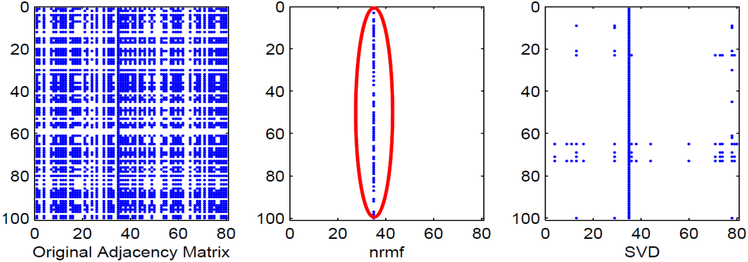
$$\min_{f,g} \sum_{(i,j):A_{ij}>0} (R_{ij} - f(i)g(j))^2 \text{ s.t. } f(i)g(j) \leq R_{ij} \quad \forall (i,j) : A_{ij} > 0.$$

Strange connection: It is a connection between two nodes which belong to two remotely connected communities, respectively.



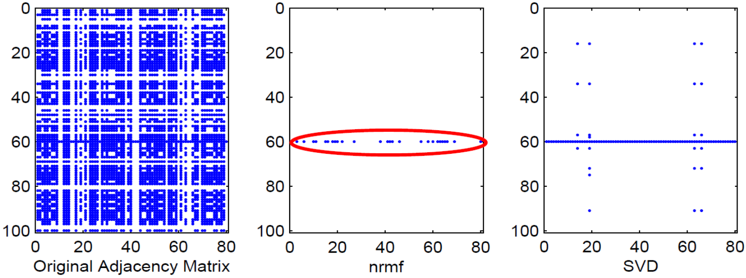
(a) strange connection

Port-scanning like behavior:It is a type-1 node that is connected to many different type-2 nodes in the bipartite graph. For example, in an IP traffic network, this could be an IP source which sends packages to many different IP destinations (therefore it might be a suspicious port scanner)



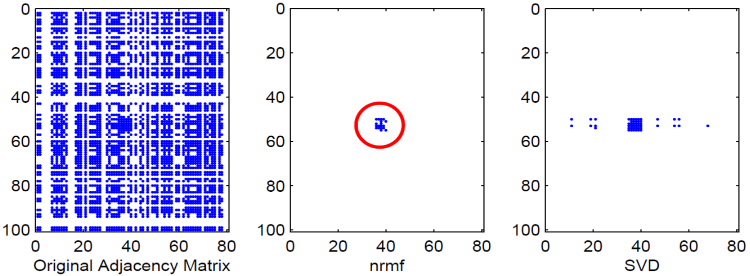
(b) port scanning

DDoS like behavior: It is a type- 2 node that is connected to many different type-1 nodes in the bipartite graph. For example, in an IP traffic network, this could be an IP destination which receives packages from many different IP sources (therefore it might be under DDoS, distributed denial-of-service, attack)



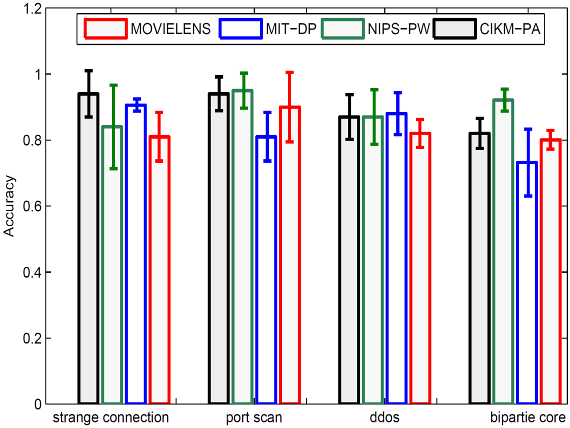
(c) ddos

Collusion type of fraud: It is a group of type-1 nodes and a group of type-2 nodes which are tightly connected with each other. For example, in financial transaction network, this could be a group of users who always give good ratings to another group of users in order to artificially boost the reputation of the target group

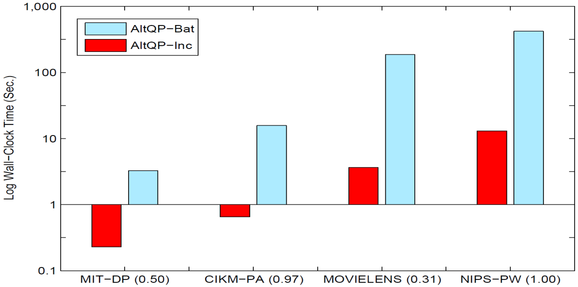


(d) bipartite core

Name	$n \times l$
MIT-DP	103×97
NIPS-PW	$2{,}037 \times 13{,}649$
CIKM-PA	$1{,}895 \times 952$
MovieLens	$6{,}040 \times 3{,}952$



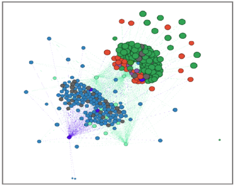
Experiments



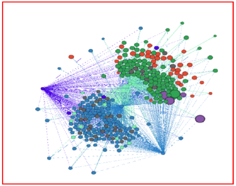
- Static (attributed) graphs
- **Dynamic (attributed) graphs (edge streams)**

Model:

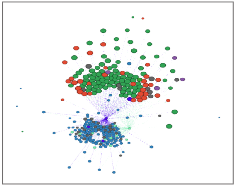
- let the sequence of weighted graphs $\{G_t\}_{t\geq 0}$, where $G_t = (V_t, E_t)$
- $e = (i, j, w) \in E_t$ if e occured at timestamp t
- edges can disappear and reappear at different moments
- $|V_t| = n$
- denote A_t and $L_t = D_t - A_t$ the adjacency and Laplacian matrices, respectively.



2012



2013



2014

Model:

- let the sequence of weighted graphs $\{G_t\}_{t\geq 0}$, where $G_t = (V_t, E_t)$
- $e = (i, j, w) \in E_t$ if e occured at timestamp t
- edges can disappear and reappear at different moments
- $|V_t| = n$
- denote A_t and $L_t = D_t - A_t$ the adjacency and Laplacian matrices, respectively.

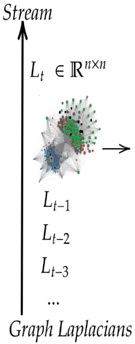
Problem

Given the sequence $\{G_t\}_{t\geq 0}$, detection unusual patterns with respect to a short/logn term window.

The goal is to find anomalous graphs G_t such that, given a scoring function $f : G_t \rightarrow \mathbb{R}$, one of the following conditions holds:

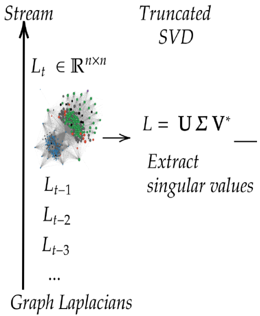
- $|f(G_t) - f(G_N)| \geq \delta$
- $|f(G_t) - f(G_W)| \geq \delta$

where G_N and G_W are normal behaviour of the graphs in global context and short-term context, respectively.

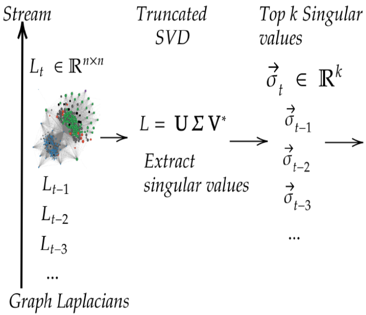


First step to define f: encode $\{G_t\}_{t\geq 0}$ into a sequence of Laplacians $\{L_t\}_{t\geq 0}$ and compute SVD of each L_t.

- actual result $f(G_t) = \sigma(L_t) \in \mathbb{R}^n$.
- node permutation invariant measure
- cost on a horizon of length T?



In order to reduce complexity we can modify: $f(G_t) = \sigma(L_t) \in \mathbb{R}^n \to f(G_t) = \sigma_k(L_t) \in \mathbb{R}^k$. Now the SVD is truncated to the first k singular values.



Second step: how to extract the normal behaviour over a fixed sliding window of size w?

Possible answer: form the following matrix

$$C_w = \begin{bmatrix} | & | & | & | \\ \sigma_{t-w-1} & \sigma_{t-w-2} & \cdots & \sigma_{t-1} \\ | & | & | & | \end{bmatrix}$$

and compute the left (maximal) singular vector.
On short the normality vector is computed as:

$$f(G_W) = u_1(C_w).$$

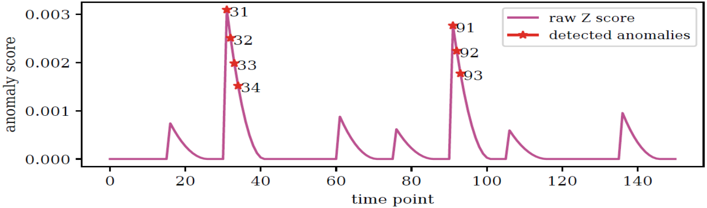
Third step: how to measure the abnormality of a given graph?

Possible answers:

- $Z_t = |f(G_t) - f(G_w)$ (distance)
- $Z_t = 1 - \frac{f(G_t)^T f(G_w)}{||f(G_t)|| \, ||f(G_w)||}$ (similarity)

Order	Time Point	Generative SBM Model
0	0	$N_c = 4,\ p_{in} = 0.25,\ p_{ex} = 0.05$
1	16	$N_c = 10,\ p_{in} = 0.25,\ p_{ex} = 0.05$
2	31	$N_c = 2,\ p_{in} = 0.5,\ p_{ex} = 0.05$
3	61	$N_c = 4,\ p_{in} = 0.25,\ p_{ex} = 0.05$
4	76	$N_c = 10,\ p_{in} = 0.25,\ p_{ex} = 0.05$
5	91	$N_c = 2,\ p_{in} = 0.5,\ p_{ex} = 0.05$
6	106	$N_c = 4,\ p_{in} = 0.25,\ p_{ex} = 0.05$
7	136	$N_c = 10,\ p_{in} = 0.25,\ p_{ex} = 0.05$

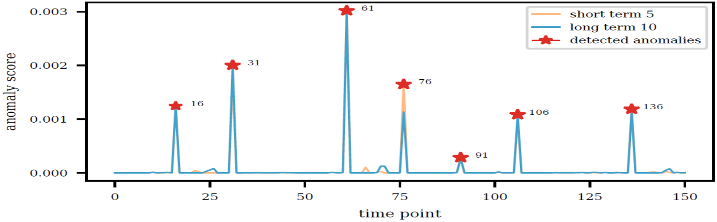
Results



0.003
0.002
0.001
0.000
anomaly score
0
20
40
60
80
100
120
140
time point
raw Z score
detected anomalies
31
32
33
34
91
92
93

Results

Time Point	Type	Generative SBM Model
0	start point	$N_c = 4,\ p_{in} = 0.25,\ p_{ex} = 0.05$
16	event	$N_c = 4,\ p_{in} = 0.25,\ \mathbf{p_{ex} = 0.15}$
31	change point	$\mathbf{N_c = 10},\ p_{in} = 0.25,\ p_{ex} = 0.05$
61	event	$N_c = 10,\ p_{in} = 0.25,\ \mathbf{p_{ex} = 0.15}$
76	change point	$\mathbf{N_c = 2},\ p_{in} = 0.5,\ p_{ex} = 0.05$
91	event	$N_c = 2,\ p_{in} = 0.5,\ \mathbf{p_{ex} = 0.15}$
106	change point	$\mathbf{N_c = 4},\ p_{in} = 0.25,\ p_{ex} = 0.05$
136	event	$N_c = 4,\ p_{in} = 0.25,\ \mathbf{p_{ex} = 0.15}$



short term 5
long term 10
detected anomalies
anomaly score
0.003
0.002
0.001
0.000
time point
0
25
50
75
100
125
150
16
31
61
76
91
106
136

References

1. Akoglu, L., McGlohon, M. and Faloutsos, C.. Oddball: Spotting anomalies in weighted graphs. PAKDD, 2010.
2. Tong, H. and Lin, C.Y. Non-Negative Residual Matrix Factorization with Application to Graph Anomaly Detection. In SDM, 2011.
3. Davide Mottin and Konstantina Lazaridou, Anomaly Detection, Hasso Plattner Institute, Graph Mining course Winter Semester 2016.
4. Huang, S., Hitti, Y., Rabusseau, G., & Rabbany, R. (2020, August). Laplacian change point detection for dynamic graphs. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 349-358).