What is Strange in Large Networks? Graph-based Irregularity and Fraud Detection

Leman Akoglu

Computer Science · Stony Brook University

Christos Faloutsos

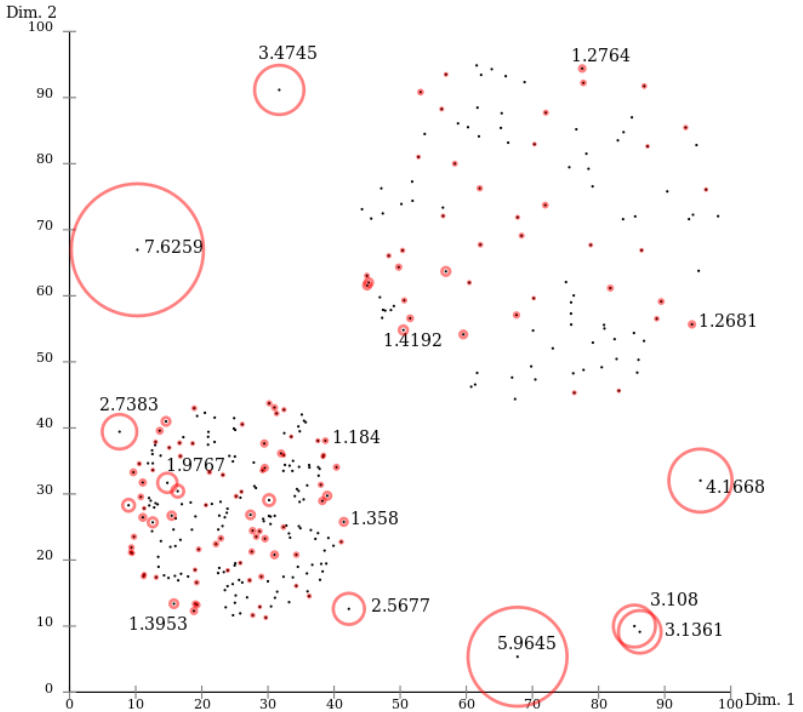
Carnegie Mellon University

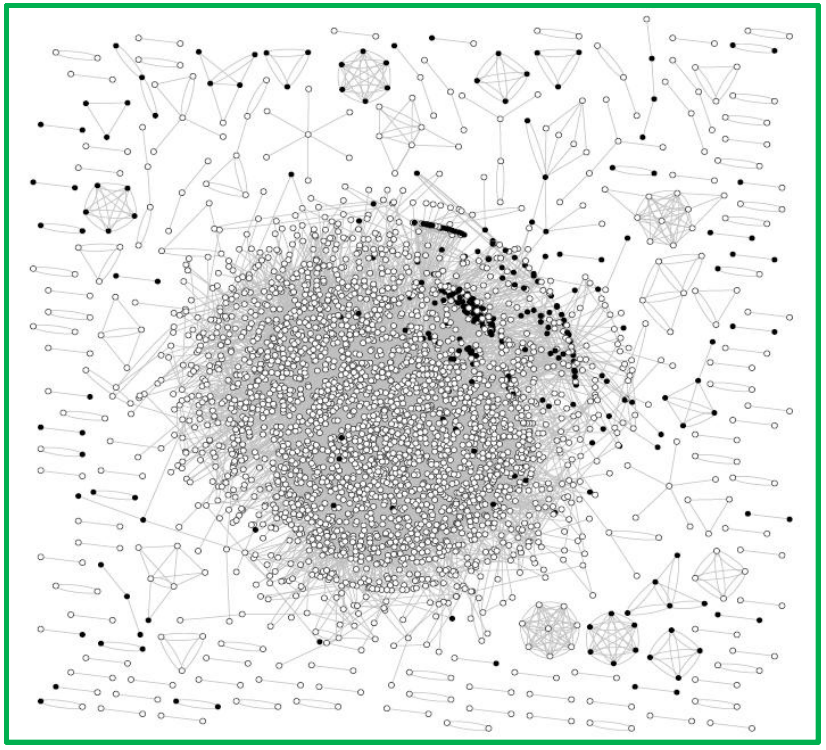
IEEE ICDM International Conference on Data Mining

Brussels 10-13 Dec 2012

Outliers vs. Graph anomalies

This tutorial





Clouds of points
(multi-dimensional)

Inter-linked objects
(network)

Roadmap

13:30	Introduction & motivation
	Part I: Anomaly detection in static data
15:30	**Coffee break**
16:00	**Part II:** Anomaly detection in dynamic data
	Part III: Graph-based algorithms & applications
18:00	The End

Disclaimers

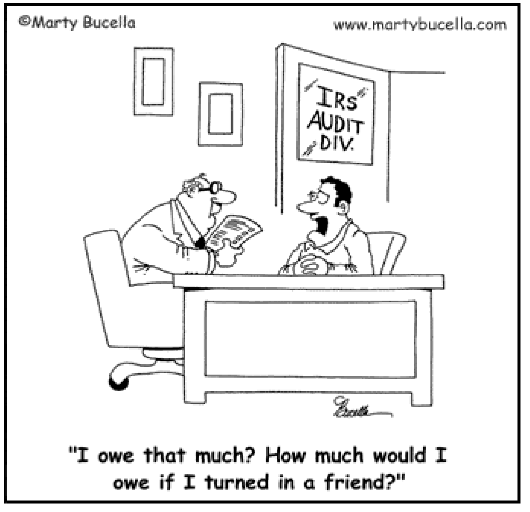
This tutorial does not necessarily cover all related work

References are not necessarily authoritative and complete

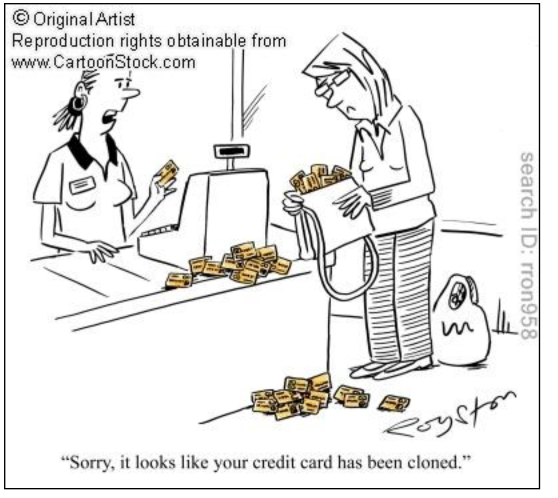
Several slides have been reused or modified by the permission of the original creators.

Anomaly detection: Applications

Tax evasion

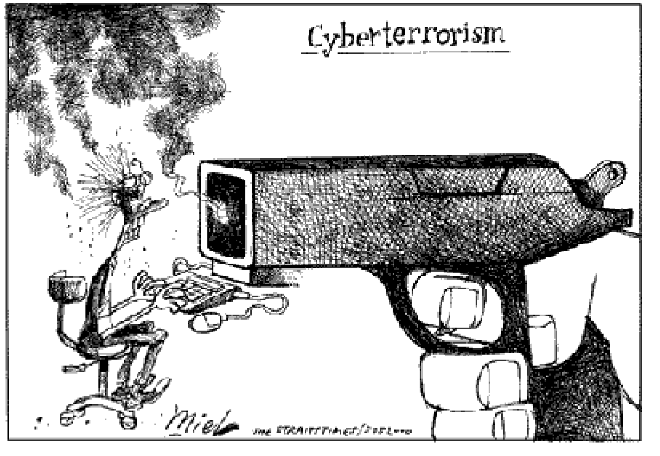


Credit card fraud



Healthcare fraud

Network intrusion



Applications

Malware

Investment fraud

Click fraud

Spyware

Insurance fraud

Malicious cargo

Auction fraud

Damage detection

Fake reviews

Medical diagnosis

Email spam

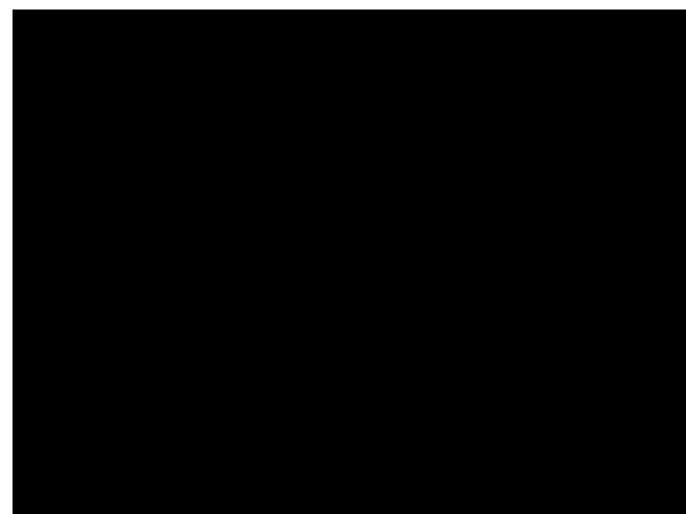
False advertising

Performance monitoring

Web spam

Insider threat

Image/video surveillance



Anomaly detection: definition

- (Hawkins' Definition of Outlier, 1980)

  "An outlier is an observation that differs so much from other observations as to arouse suspicion that it was generated by a different mechanism."

No unique definition → leads to → Many definitions in various contexts

outlier, anomaly, outbreak, event, fraud, …

Anomaly detection: definition

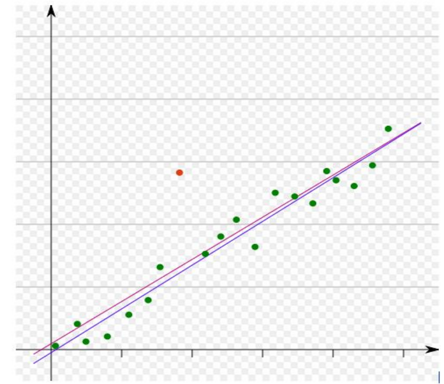
- for practical purposes,

  a record/point/graph-node/graph-edge

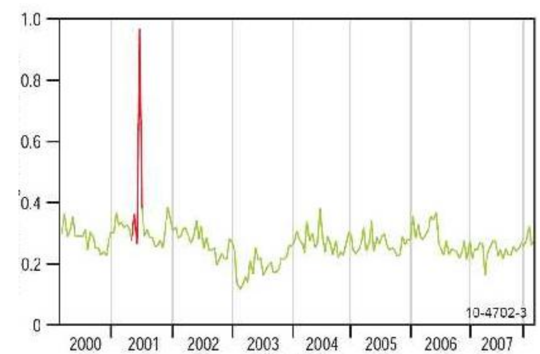
  is flagged as anomalous

  if a rarity/likelihood/outlierness score

  exceeds a user-defined threshold

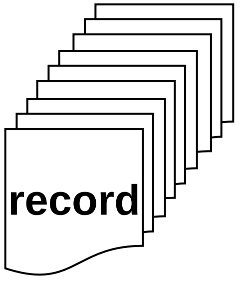


- anomalies:
  - → rare (e.g., rare combination of categorical attribute values)
  - → isolated points in n-d spaces
  - → surprising (don't fit well in our mental/statistical model == need too many bits under MDL)



Challenges

Unbalanced
(fraud is rare)

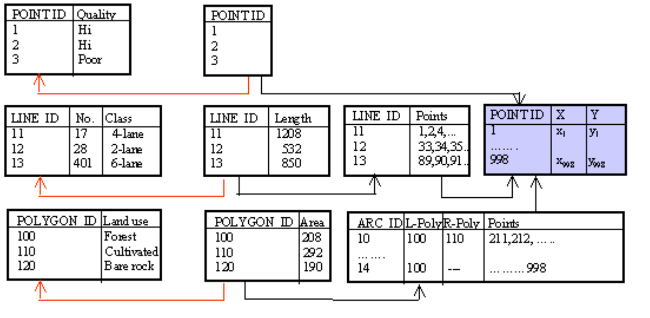


Temporal
(**V**elocity)

Data
(Volume)

Unlabeled
(no ground truth)

Categorical,
Numerical,
Relational, …
(**V**ariety)

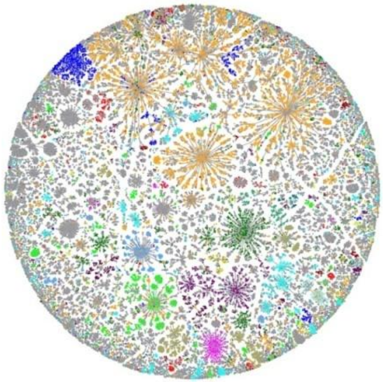


Why graph-based detection?

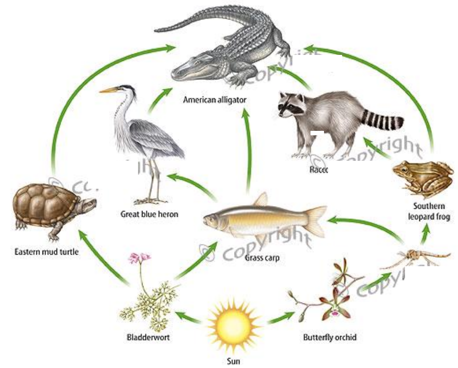
- Powerful representation
  - Interdependent instances
  - Long-range relations
  - Node/Edge attributes (data complexity)
  - Hard to fake/alter (adversarial robustness)

- Abundant relational data
  - Web, email, phone call, …

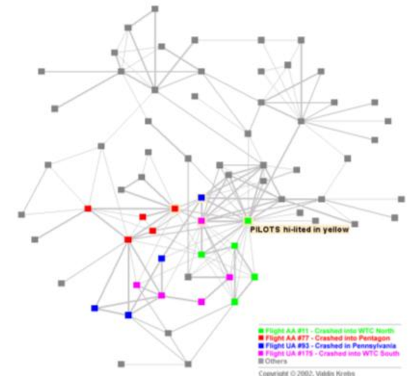
Real graphs (1)



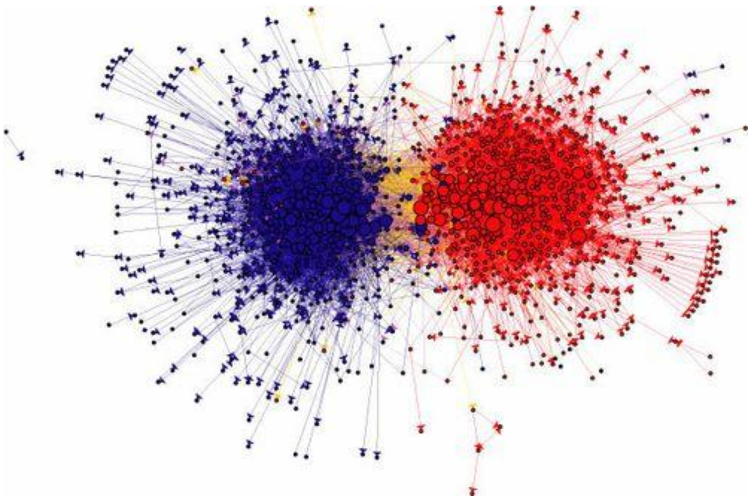
Internet Map



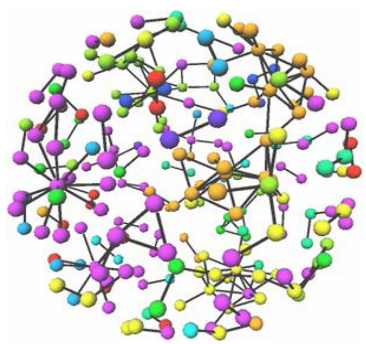
Food Web



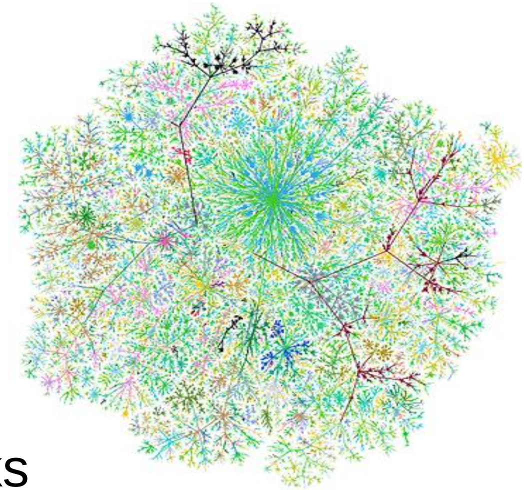
Terrorist Network



Blog networks

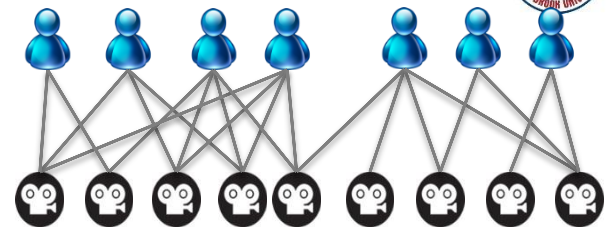


Biological networks

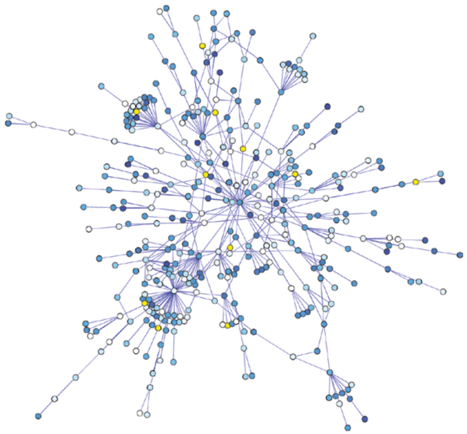


Web Graph

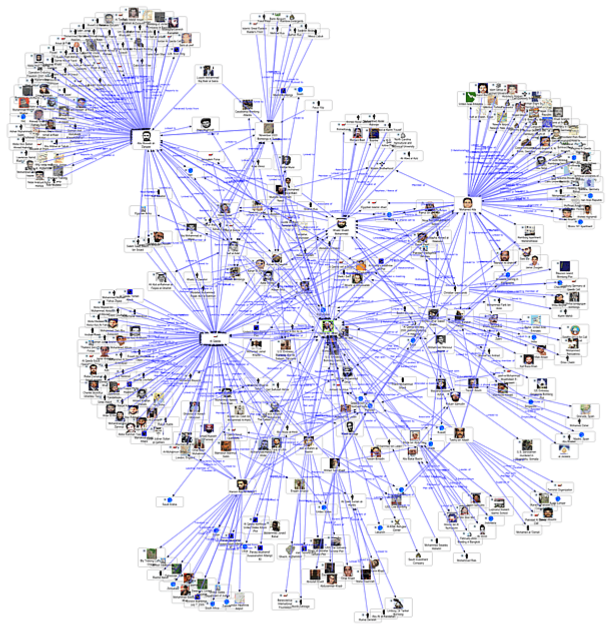
Real graphs (2)



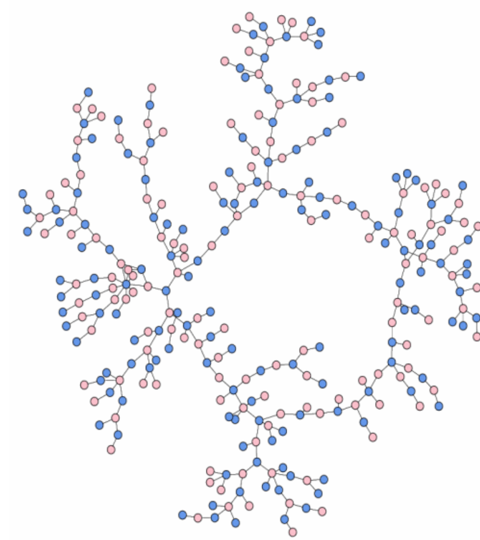
Retail networks



Protein-protein
Interaction

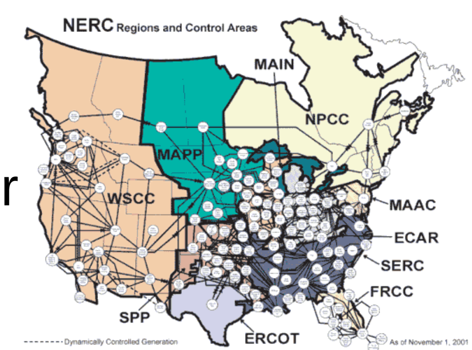


Social Network



Dating network

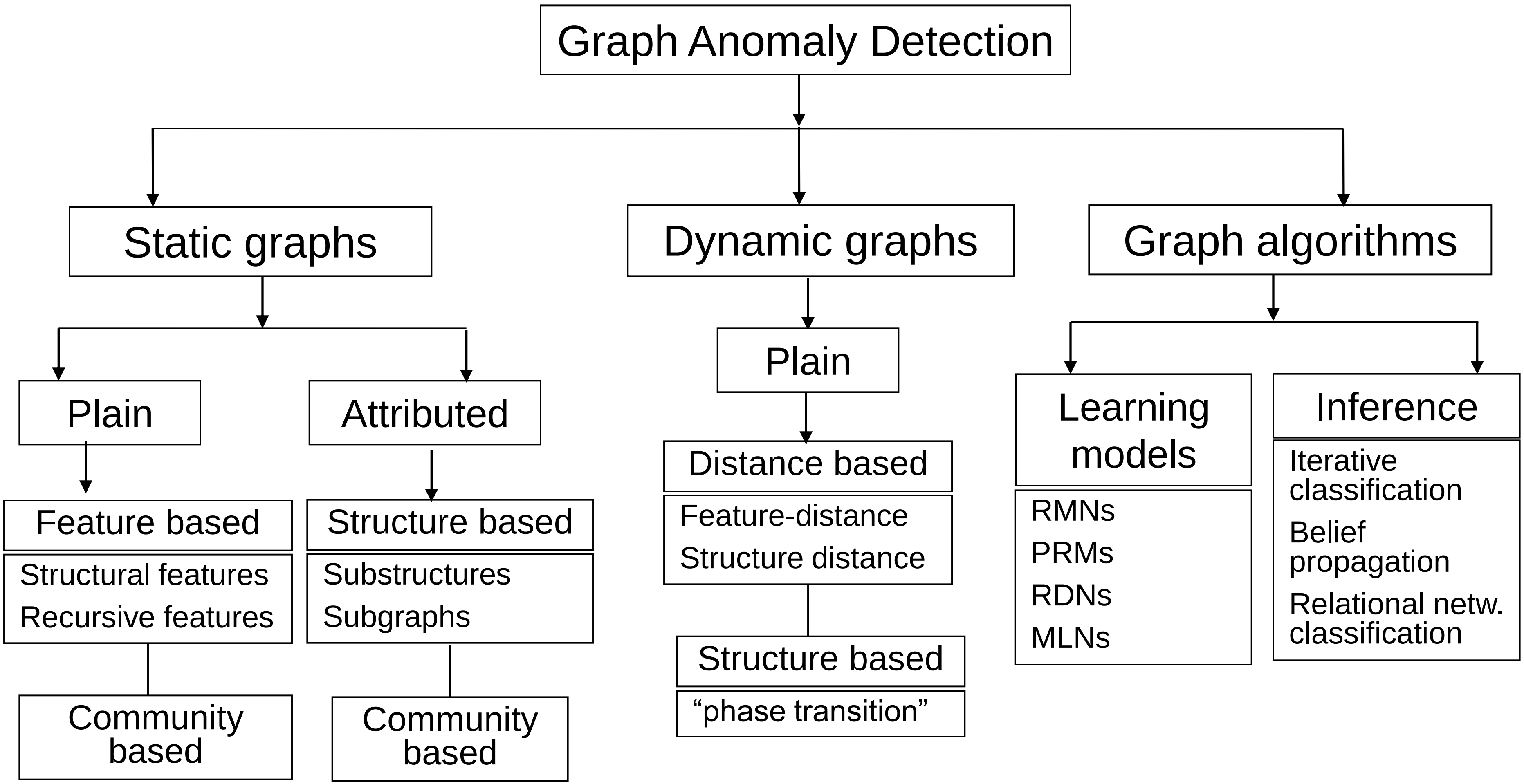
Power
Grid



Problem revisited for graphs

- Three different problem settings
  - Unlabeled/Labeled (Attributed) Graphs
  - Static/Dynamic Graphs
  - Un-/Semi-/- Supervised Graph Techniques

Taxonomy


Graph Anomaly Detection
Static graphs
Dynamic graphs
Graph algorithms
Plain
Attributed
Feature based
Structural features
Recursive features
Community based
Structure based
Substructures
Subgraphs
Community based
Plain
Distance based
Feature-distance
Structure distance
Structure based
"phase transition"
Learning models
RMNs
PRMs
RDNs
MLNs
Inference
Iterative classification
Belief propagation
Relational netw. classification

Goal of this tutorial

- Introduce various problem formulations
  - Definitions change by application/representation
- Applications of problem settings
  - Intrusion, fraud, spam
- Introduce existing techniques
  - Model fitting, factorization, relational inference
- Pros and Cons
  - Parameters, scalability, robustness

Tutorial Outline

- Motivation, applications, challenges
- **Part I:** Anomaly detection in **static** data
  - Overview: Outliers in **clouds of points**
  - Anomaly detection in **graph data**
- **Part II:** Event detection in **dynamic** data
  - Overview: Change detection in **time series**
  - Event detection in **graph sequences**
- **Part III:** Graph-based **algorithms and apps**
  - Algorithms: **relational learning**
  - Applications: **fraud and spam** detection

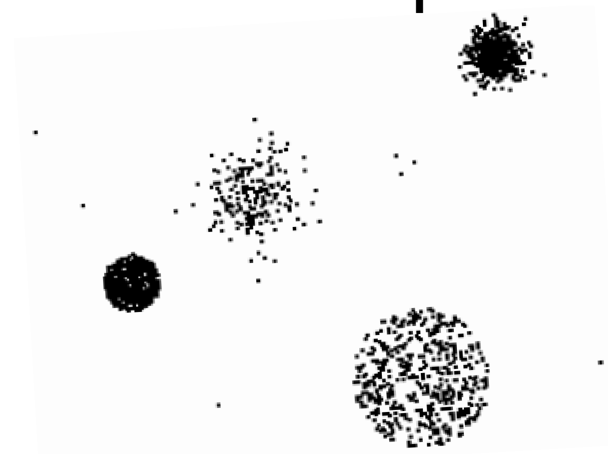
Part I: Anomaly detection in static graphs

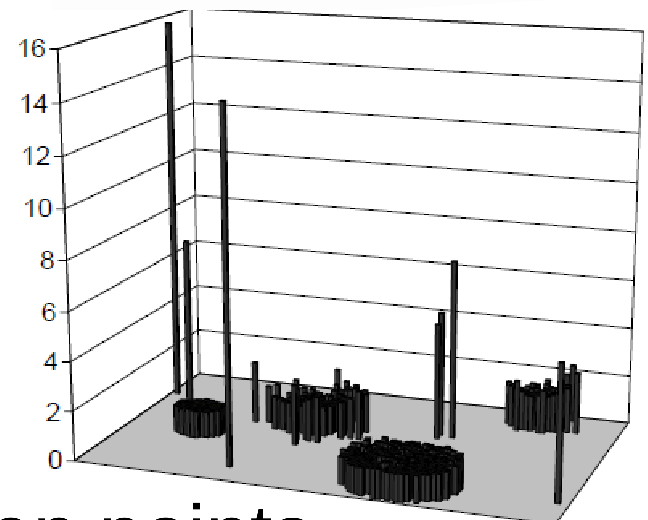
Part I: Outline

- Overview: Outliers in **clouds of points**
  - Outliers in **numerical** data points
    - distance-based, density-based, …
  - Outliers in **categorical** data points
    - model-based
- Anomaly detection in **graph data**
  - Anomalies in unlabeled, **plain** graphs
  - Anomalies in node-/edge-labeled, **attributed** graphs

Outlier detection

- Anomalies in multi-dimensional data points
  - Density-based
  - Distance-based
  - Depth-based
  - Distribution-based
  - Clustering-based
  - Classification-based
  - Information theory-based
  - Spectrum-based
  - …





- No relational links between points

Part I: References (outliers)

- M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. SIGMOD, 2000.
- S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos. LOCI: Fast outlier detection using the local correlation integral. ICDE, 2003.
- C. C. Aggarwal and P. S. Yu. Outlier detection for high dimensional data. SIGMOD, 2001.
- A. Ghoting, S. Parthasarathy and M. Otey, Fast Mining of Distance Based Outliers in High-Dimensional Datasets. DAMI, 2008.
- Y. Wang, S. Parthasarathy and S. Tatikonda, Locality Sensitive Outlier Detection. ICDE, 2011.
- Kaustav Das, Jeff Schneider. Detecting Anomalous Records in Categorical Datasets. KDD 2007.

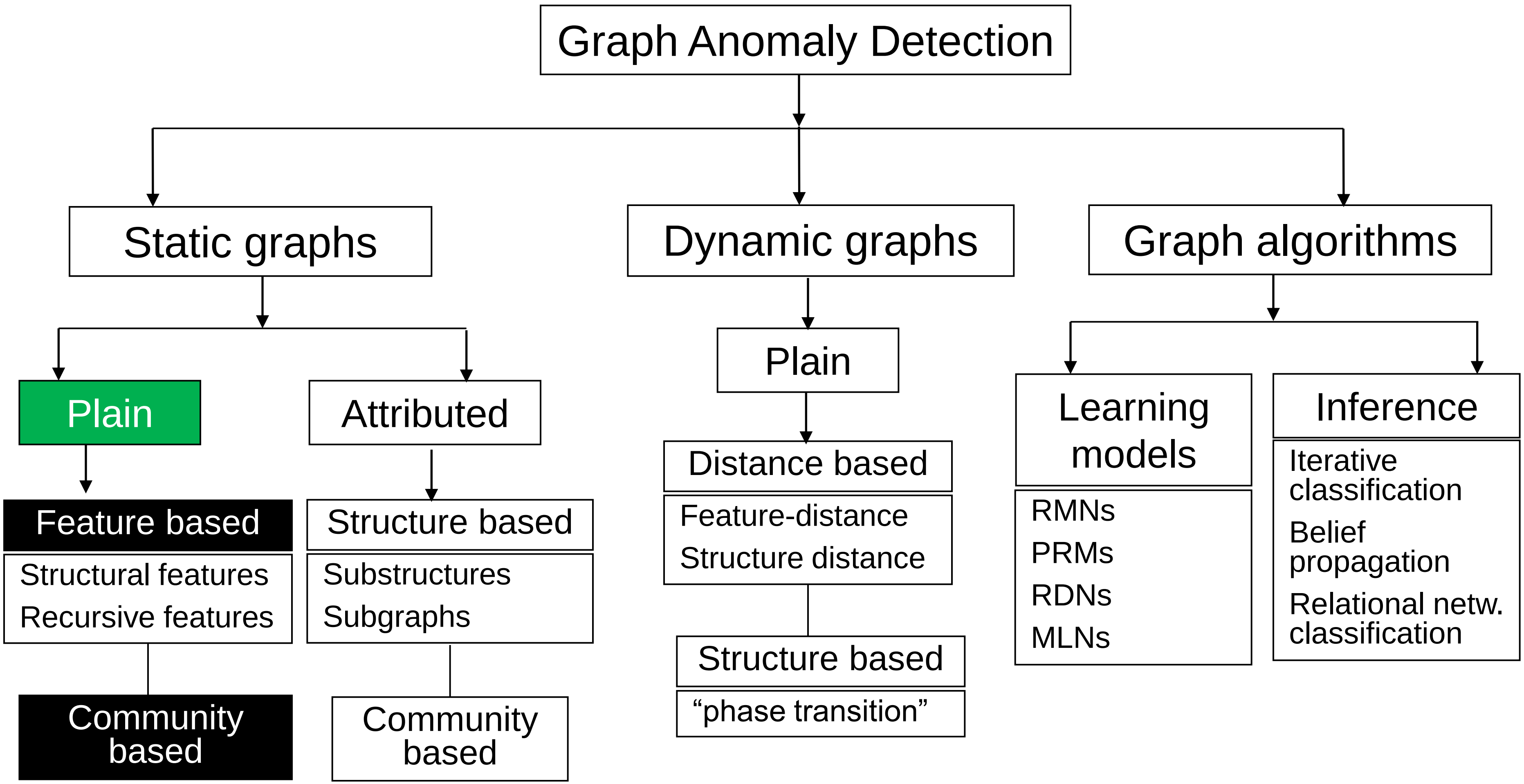
Part I: References (outliers)

- Müller E., Schiffer M., Seidl T. Adaptive Outlierness for Subspace Outlier Ranking. CIKM, 2010.
- Müller E., Assent I., Iglesias P., Mülle Y., Böhm K. Outlier Ranking via Subspace Analysis in Multiple Views of the Data. ICDM, 2012.
- L. Akoglu, H. Tong, J. Vreeken, and C. Faloutsos. Fast and Reliable Anomaly Detection in Categoric Data. CIKM, 2012.
- A. Chaudhary, A. S. Szalay, and A. W. Moore. Very fast outlier detection in large multidimensional data sets. DMKD, 2002.
- Survey: V. Chandola, A. Banerjee, V. Kumar: Anomaly Detection: A Survey. ACM Computing Surveys, Vol. 41(3), Article 15, July 2009.

Part I: Outline

- Overview: Outliers in **clouds of points**
  - Outliers in **numerical** data points
    - distance-based, density-based, …
  - Outliers in **categorical** data points
    - model-based
- Anomaly detection in **graph data**
  - Anomalies in unlabeled, **plain** graphs
  - Anomalies in node-/edge-labeled, **attributed** graphs

Taxonomy


Graph Anomaly Detection
Static graphs
Dynamic graphs
Graph algorithms
Plain
Attributed
Feature based
Structural features
Recursive features
Community based
Structure based
Substructures
Subgraphs
Community based
Plain
Distance based
Feature-distance
Structure distance
Structure based
"phase transition"
Learning models
RMNs
PRMs
RDNs
MLNs
Inference
Iterative classification
Belief propagation
Relational netw. classification

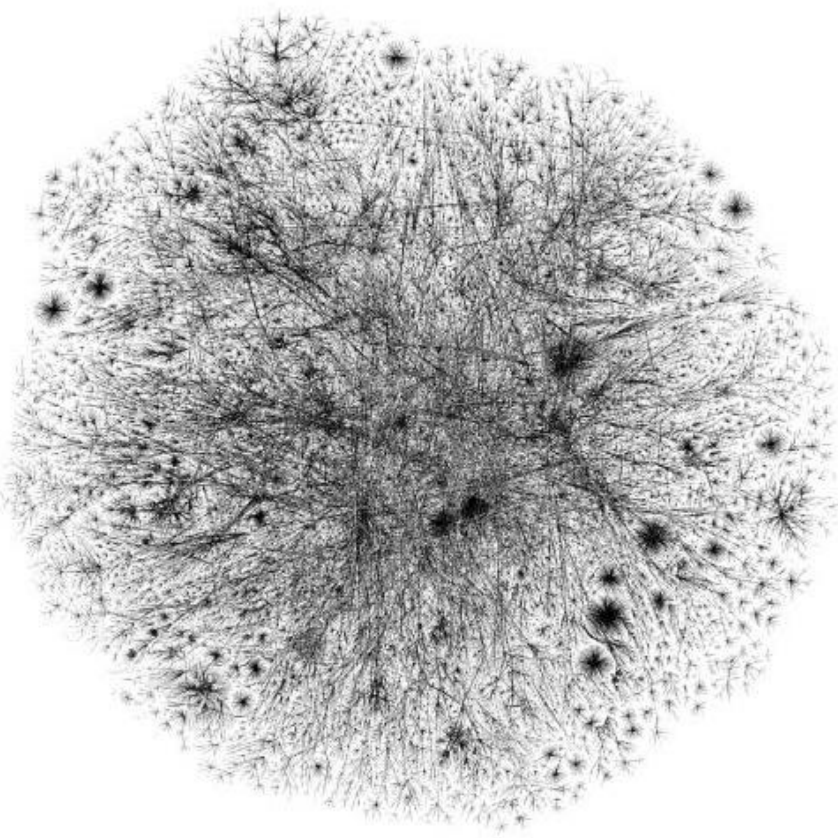
Anomalies in Weighted Graphs

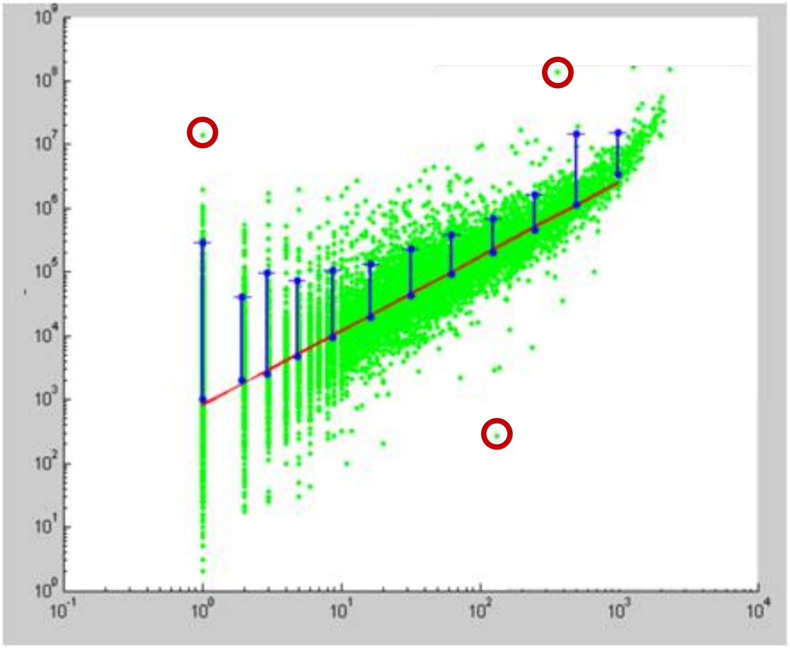
- Problem:

Q1. Given a **weighted** and unlabeled graph, how can we spot strange, abnormal, extreme nodes?

Q2. Can we explain why the spotted nodes are anomalous?

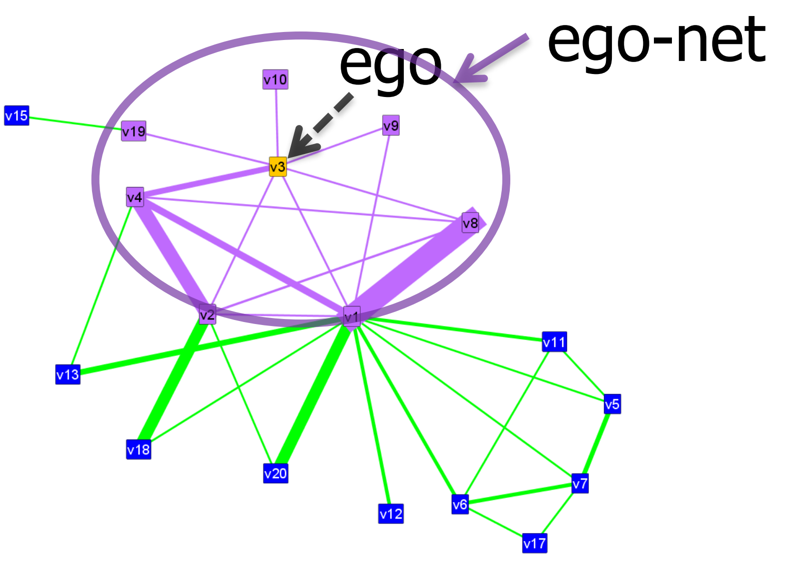
Problem sketch



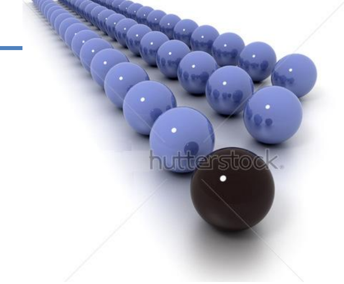


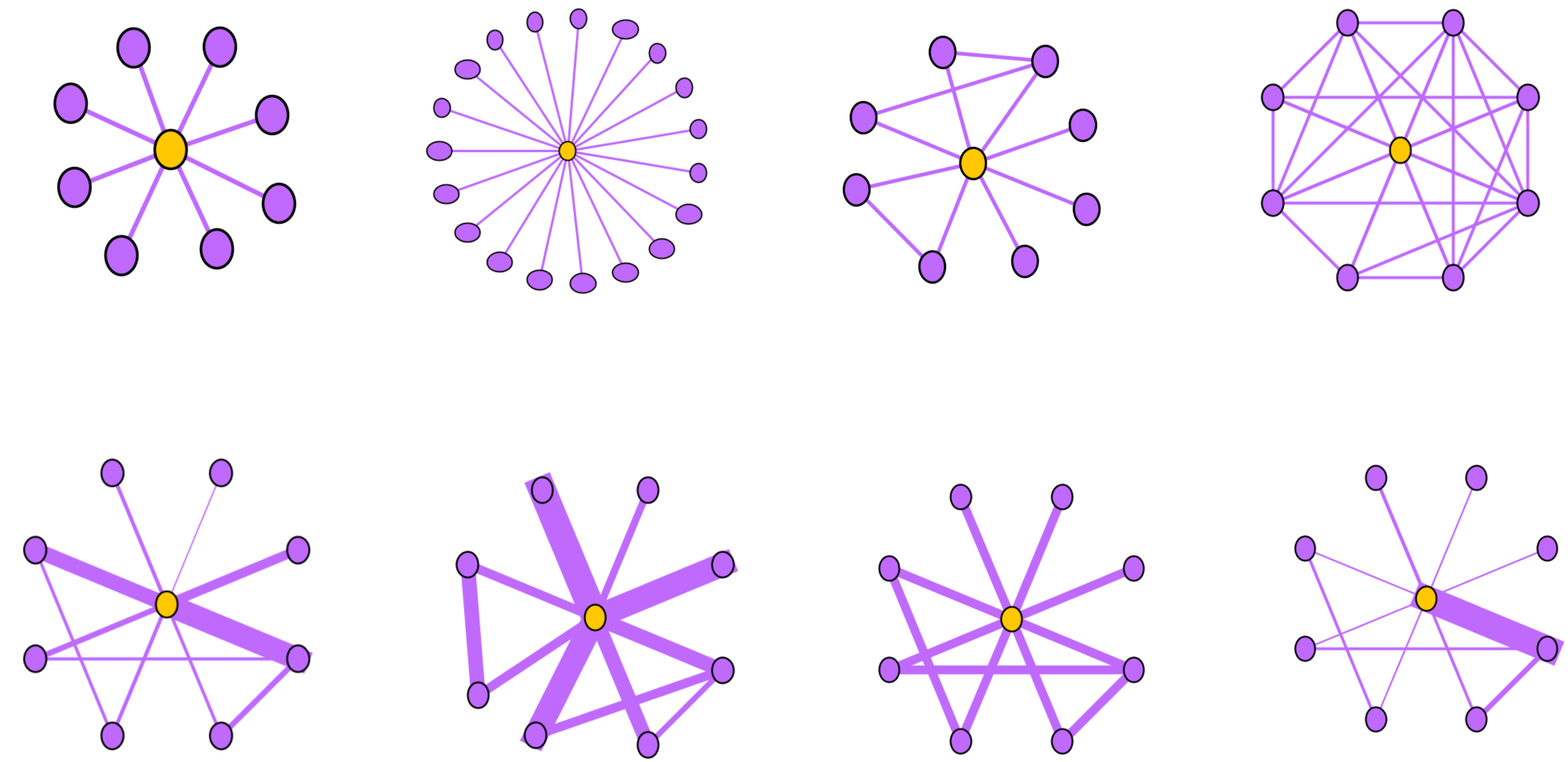
OddBall: approach

1) For each node,
   - 1.1) Extract "ego-net" (=1-step neighborhood)
   - 1.2) Extract features (#edges, total weight, etc.)
     - → features that could yield "laws"
     - → features fast to compute and interpret
2) Detect patterns:
   - → regularities
3) Detect anomalies:
   - →"distance" to patterns



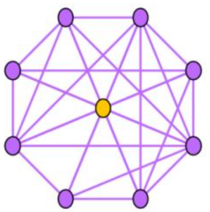
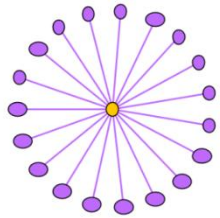
What is odd?



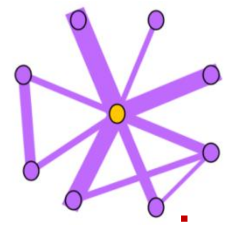


Which features to compute?

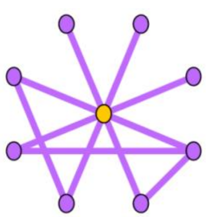
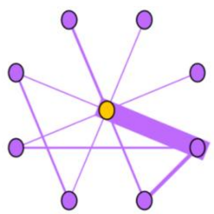
- N_i: number of neighbors (degree) of ego *i*
- E_i: number of edges in egonet *i*

- W_i: total weight of egonet *i*

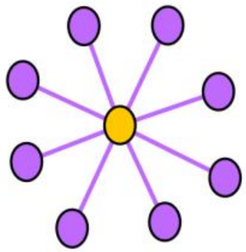


- $\lambda_{w,i}$: principal eigenvalue of the weighted adjacency matrix of egonet *i*

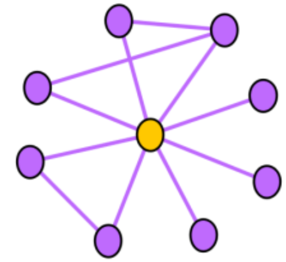



Weighted principal eigenvalue

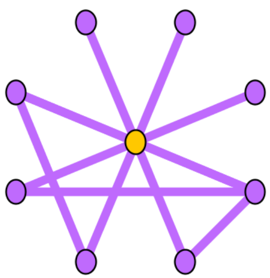
details



$$\lambda_{w,i} = \sqrt{N} = \sqrt{E} = \sqrt{W}$$

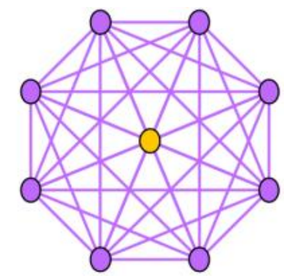


$$\lambda_{w,i} > \sqrt{N}$$
$$\propto \sqrt{E}, \sqrt{W}$$



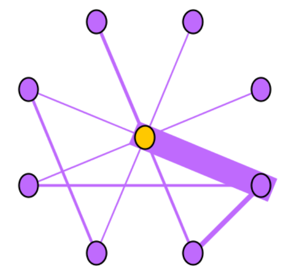
$$\lambda_{w,i} \propto \sqrt{W}$$

$$\lambda_{w,i} = N \approx \sqrt{W}$$



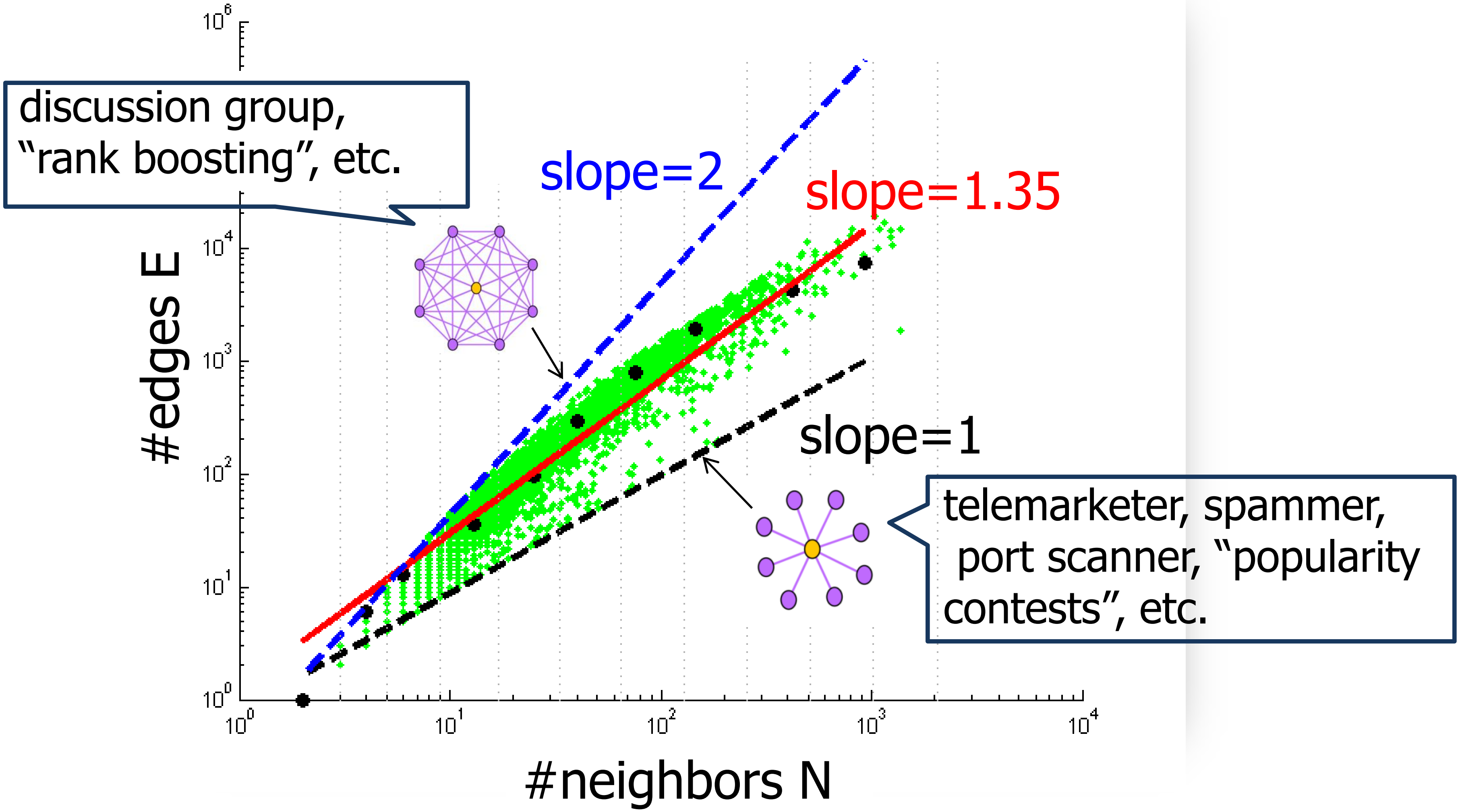
$$\lambda_{w,i} = W$$

$$\lambda_{w,i} \approx W$$

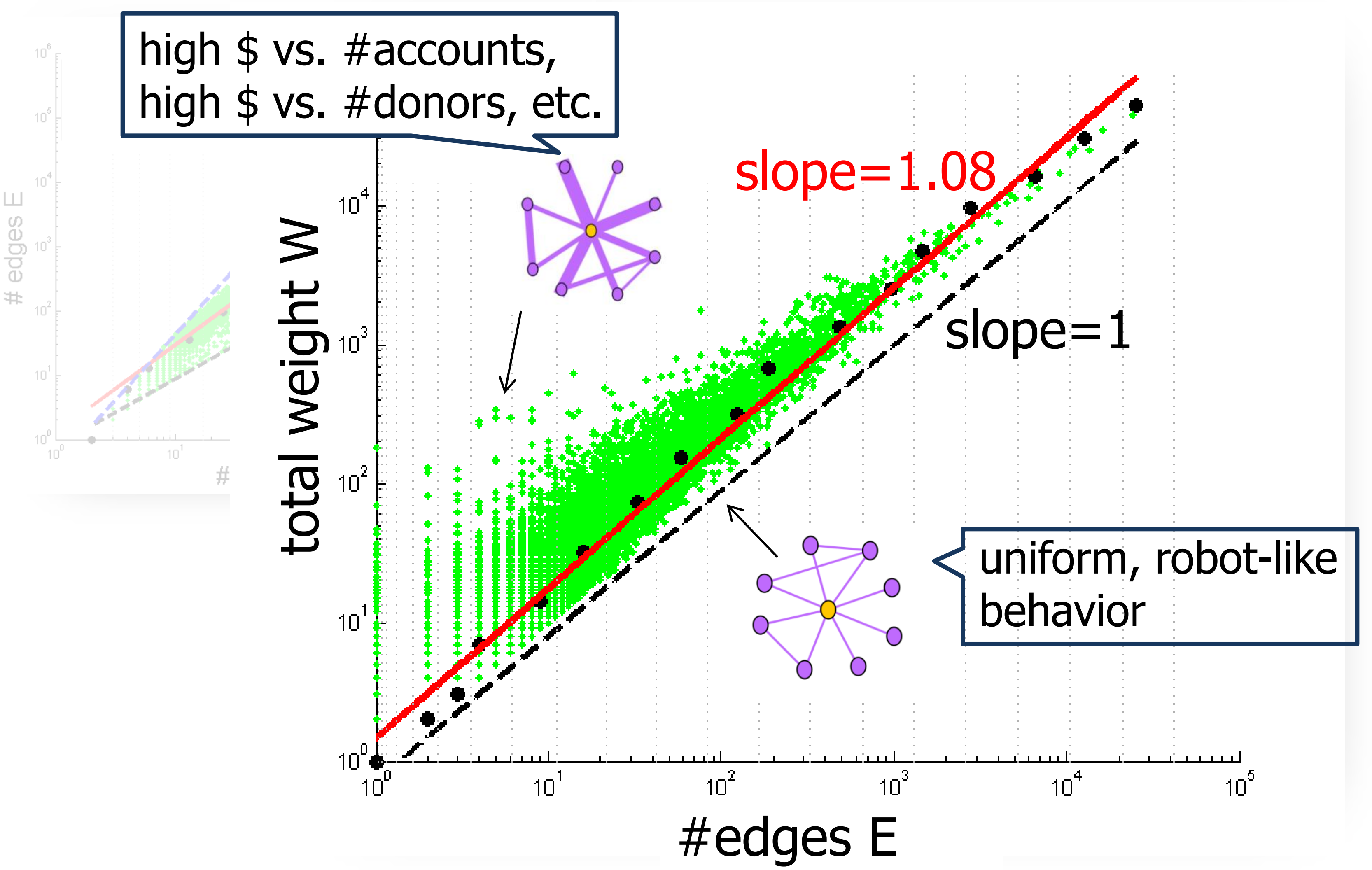


N: #neighbors, W: total weight

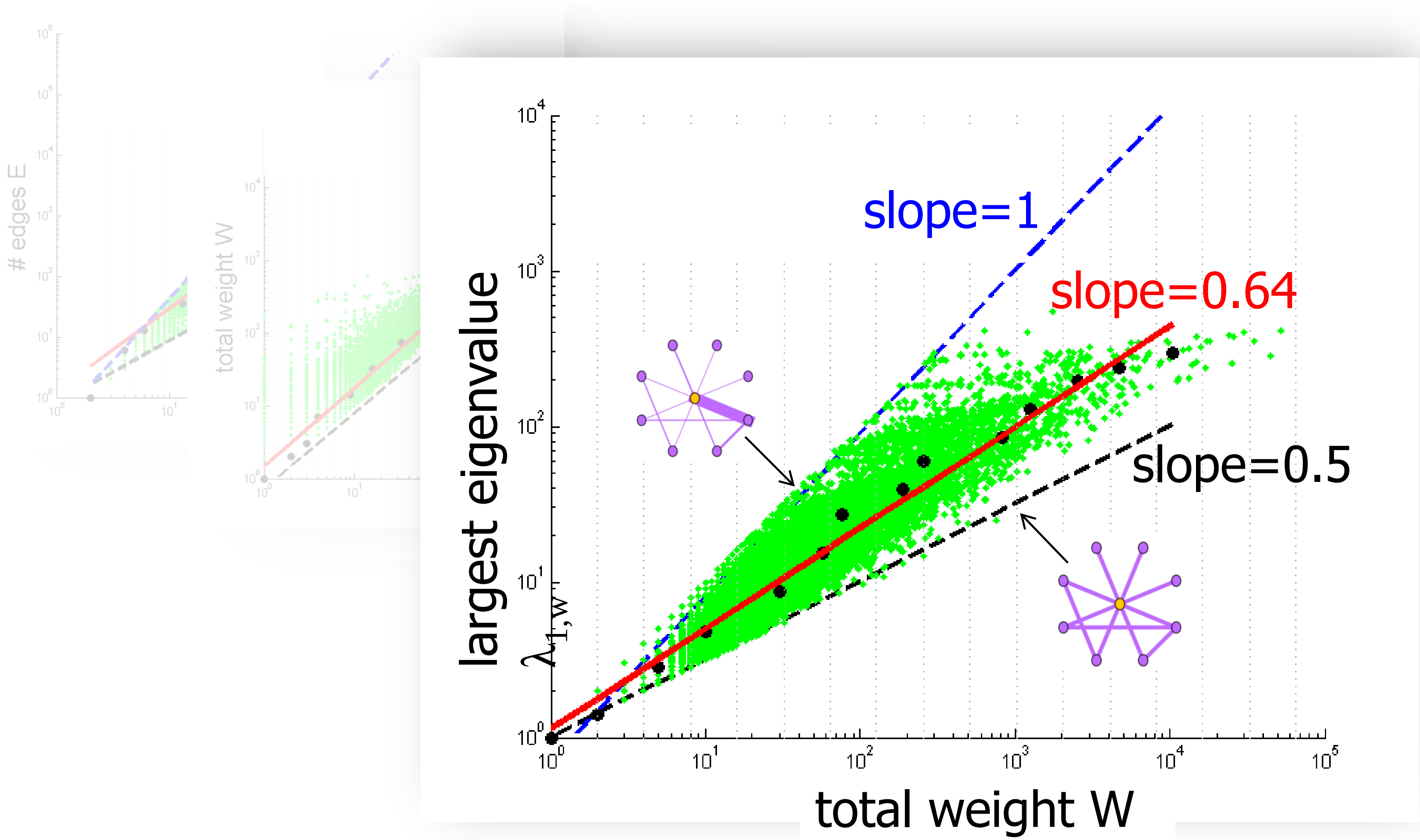
OddBall: pattern#1



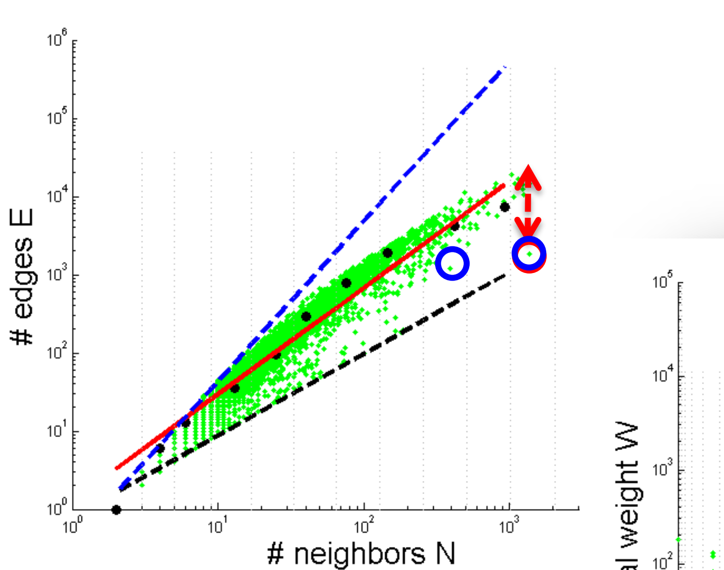
OddBall: pattern#2



OddBall: pattern#3



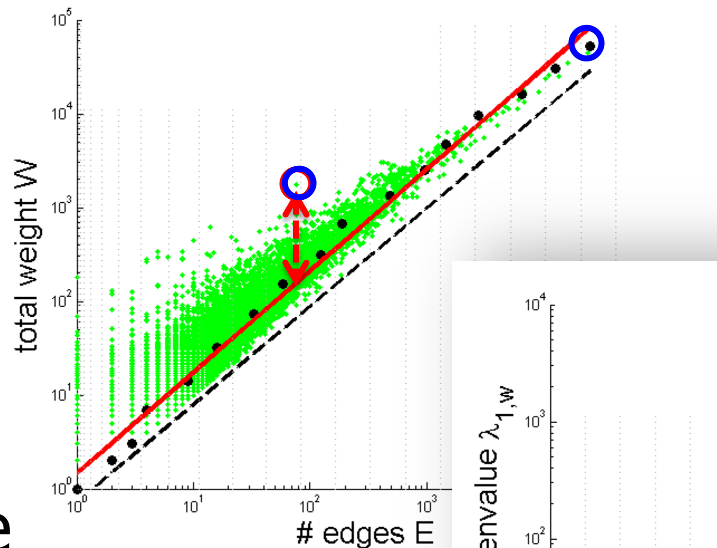
OddBall: anomaly detection

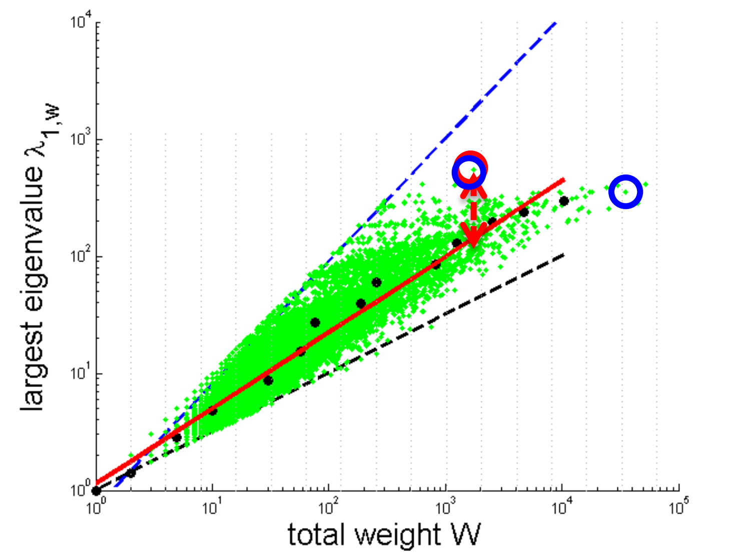


score_{dist} = distance to fitting line

score_{outl} = outlier-ness score

score = func (score_{dist} , score_{outl})



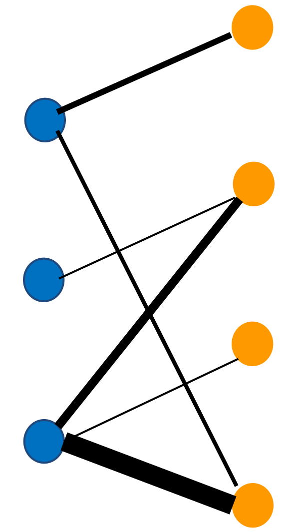


✓ can tell what type of anomaly a node belongs to

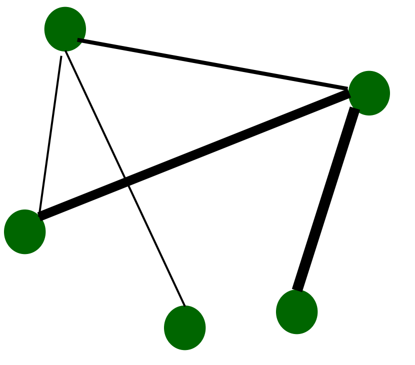
✓ can quantify "anomalous-ness" of nodes using score

OddBall: datasets

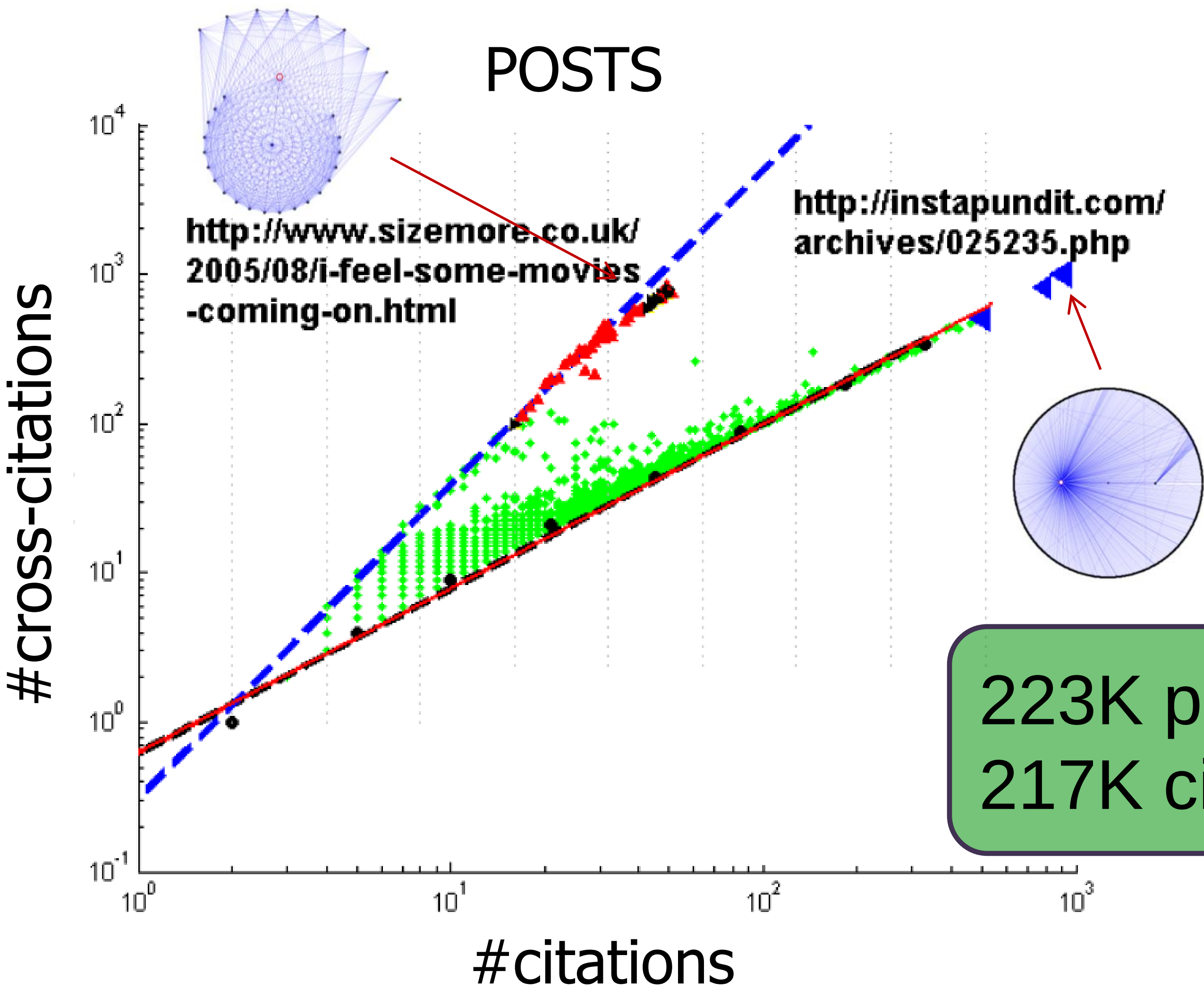
Bipartite graphs:	|V|	|E|
1. **FEC Don2Com**	1.6M	2M
2. **FEC Com2Cand**	6K	125K
3. **DBLP Auth2Conf**	21K	1M



Unipartite graphs:	|V|	|E|
4. **BlogNet**	27K	126K
5. **PostNet**	223K	217K
6. Enron	36K	183K
7. AS peering	11K	8K

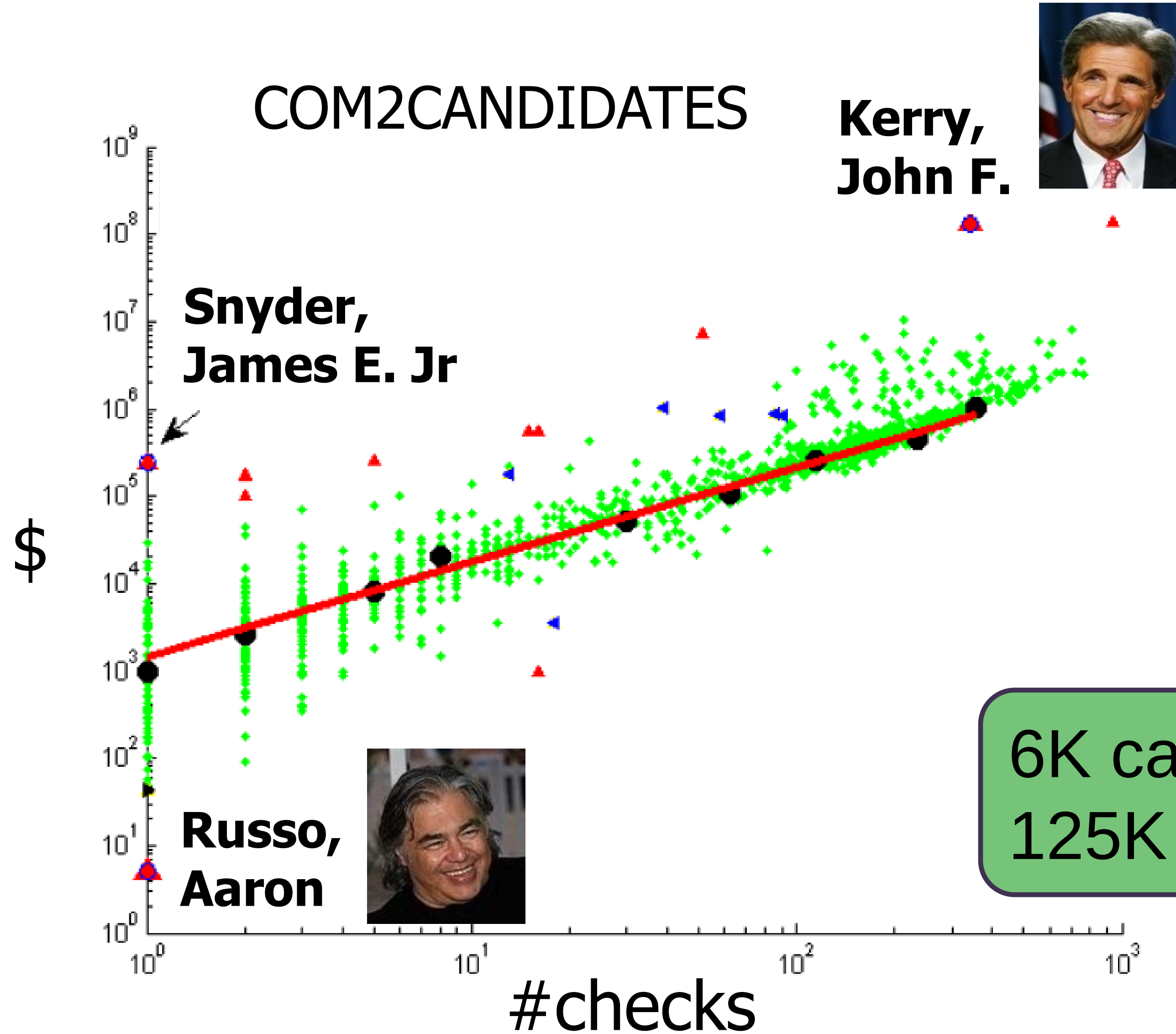


OddBall at work (Posts)



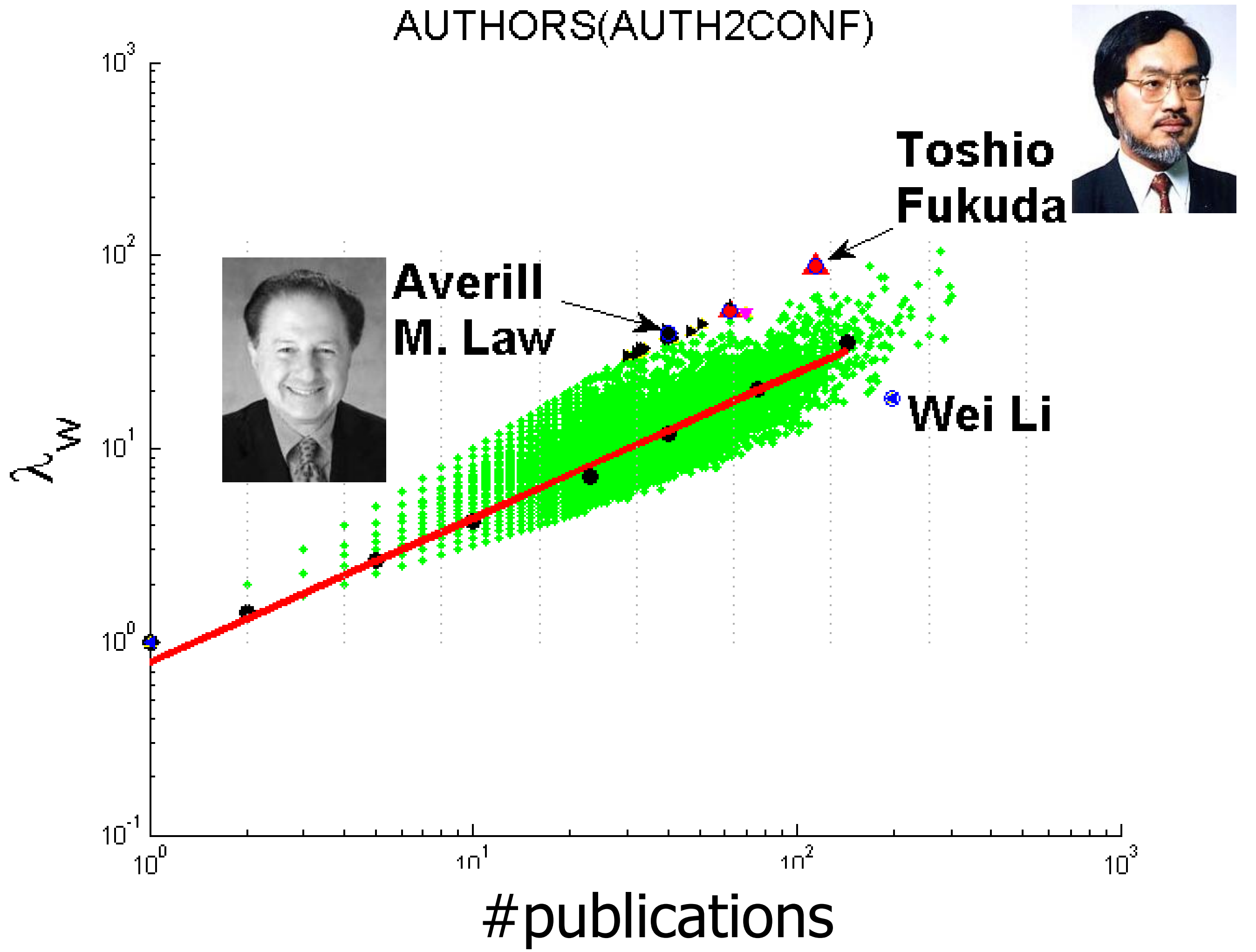
223K posts
217K citations

OddBall at work (FEC)



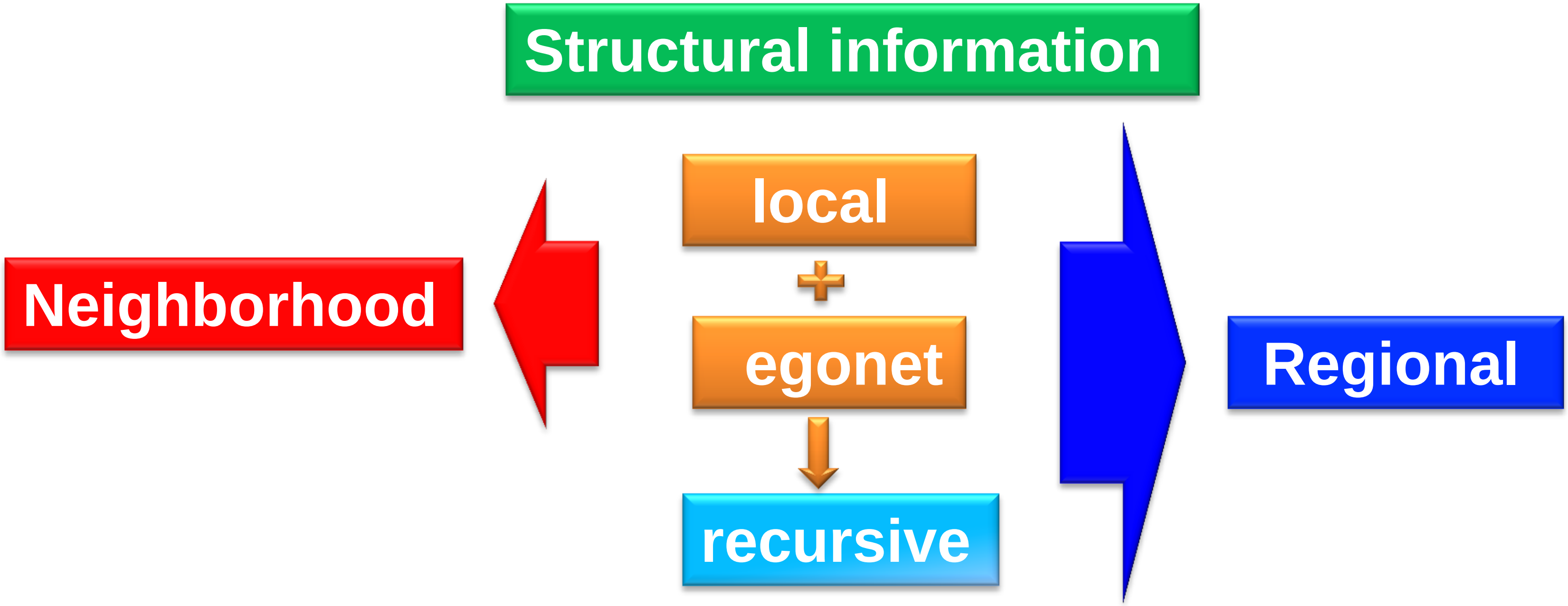
6K candidates
125K checks

OddBall at work (DBLP)



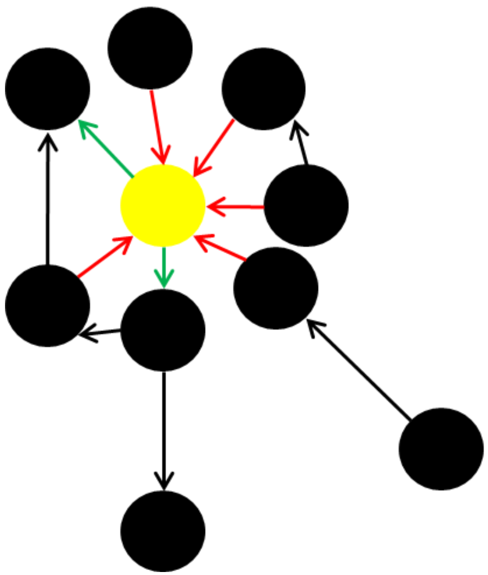
Recursive structural features

- Main idea: recursively combine "local" (node-based) and neighbor (egonet-based) features
  - Recursive feature: any aggregate computed over any feature (including recursive) value among a node's neighbors



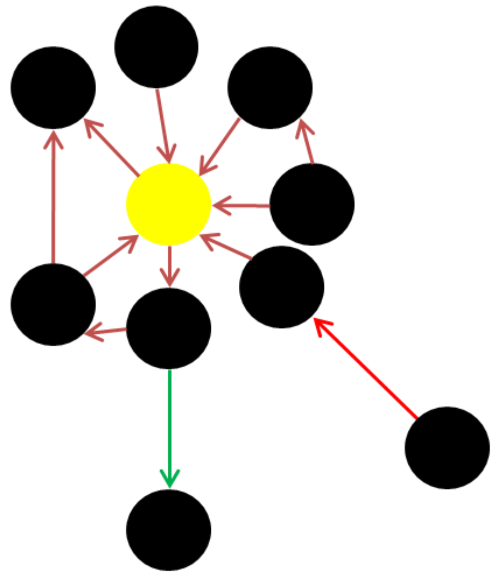
Recursive structural features

local



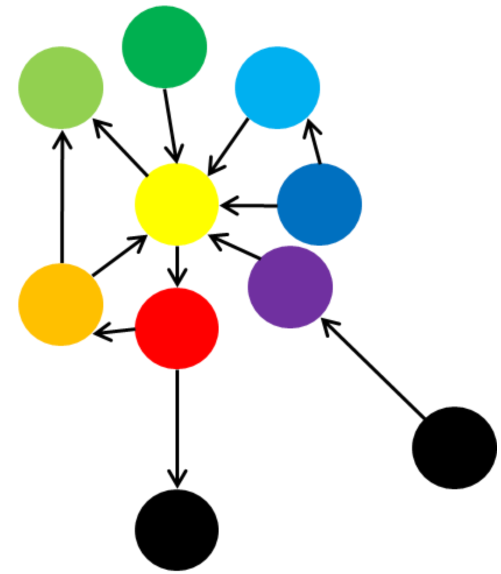
in- and out-degree, **weighted** versions

egonet



within-, incoming-, outgoing-*egonet* edges, **weighted** versions

recursive



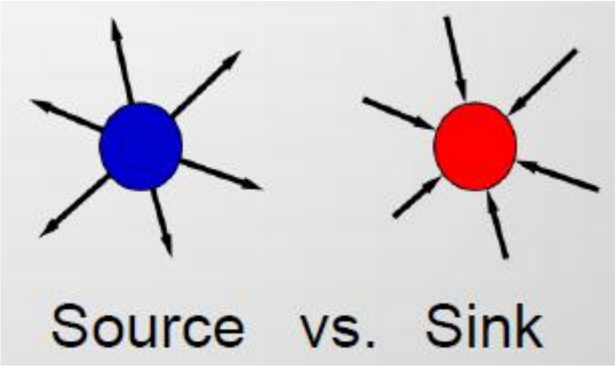
aggregate feature over neighbors

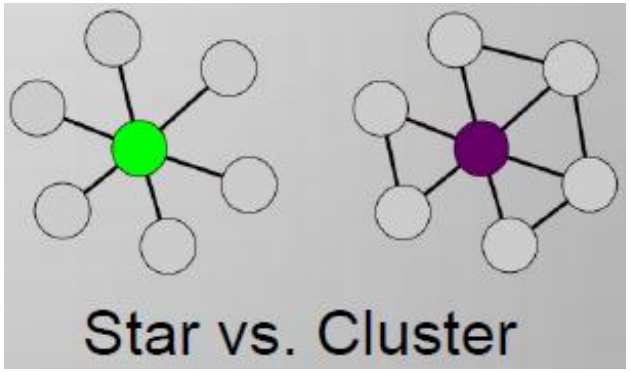
e.g. max/min/avg degree

(1 + 1 + 2 + 0 + 1 + 0 + 1)/7 = 0.86

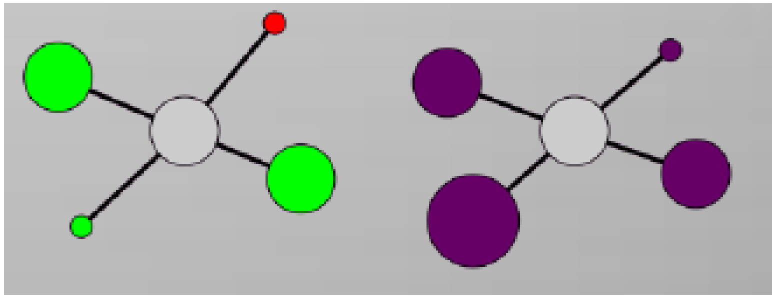
Recursive structural features

- Neigborhood features
  - captures node connectivity



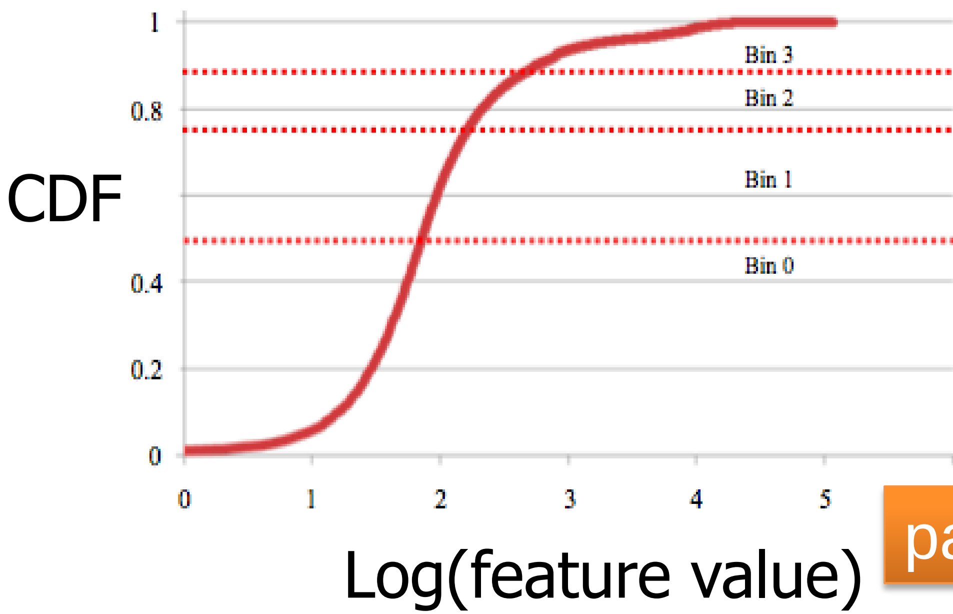


- Regional features
  - captures “kinds” of neighbors



Computing recursive features

details



recursive features

vertical logarithmic binning of size p

bin feature (integer)

not disagree at >s nodes

paired features (s-friend)

replace each CC in s-friend graph by single feature

retain simpler features

i.e. generated in fewer iterations

retained features from each iteration

repeat until no pruning

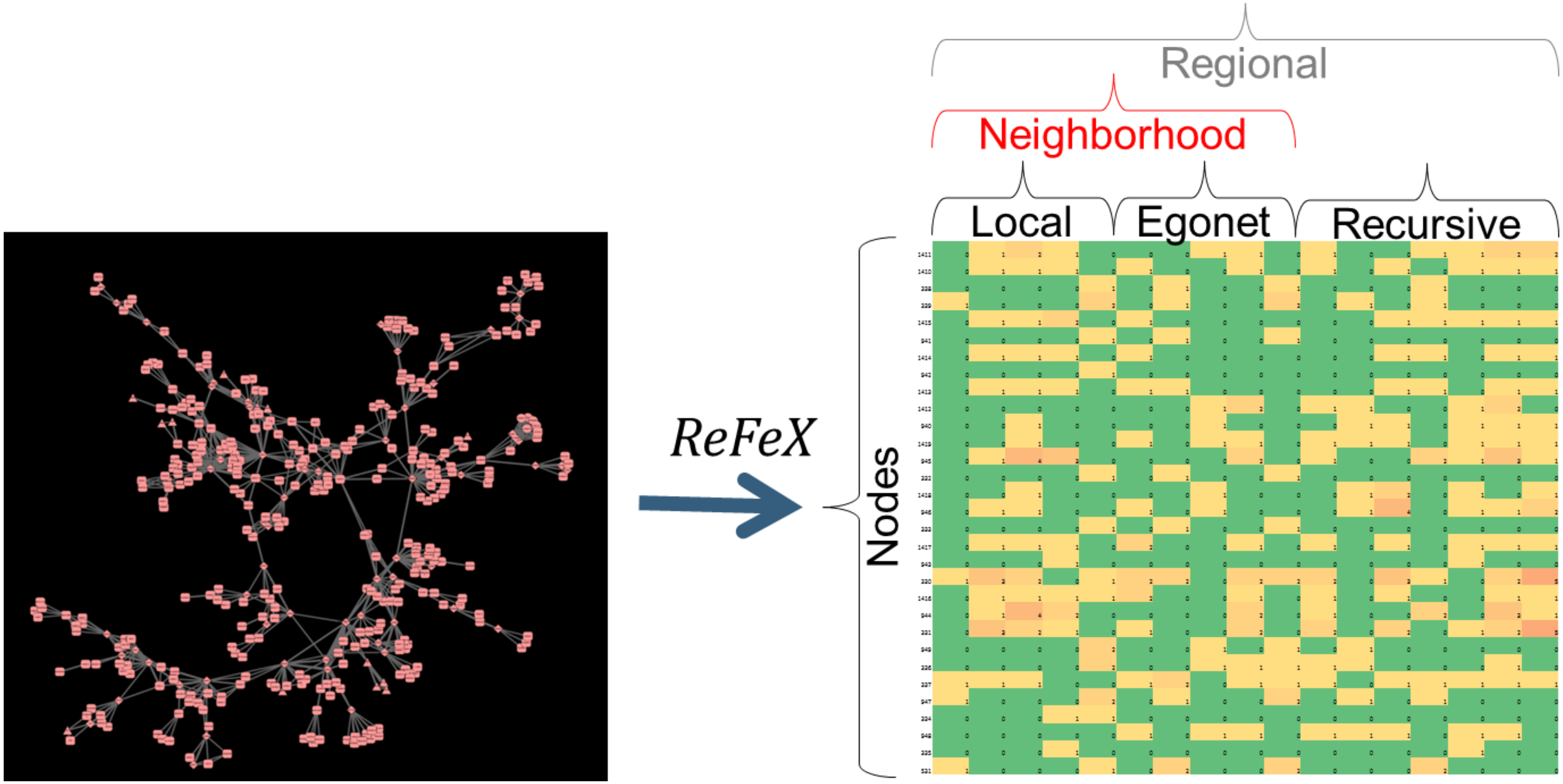
Prune highly correlated features

Recursive structural features

- Capturing regional (behavioral) information in large graphs
- Feature construction linear in graph size

- Aggregates only for numerical features
- Parameters p, s for binning and pruning

ReFeX: Recursive Feature eXtraction



- Recursive features proved effective in transfer learning, identity resolution
  (yet to be studied for anomaly detection)

Anomalies in Bipartite Graphs

Sun et al. '05

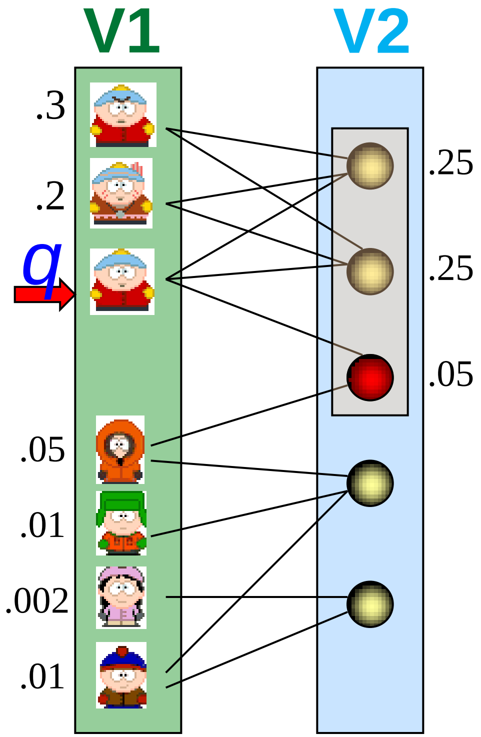
- Problem:

Q1. **Neighborhood formation (NF)**

- Given a query node *q* in V_1, what are the relevance scores of all the nodes in V_1 to *a* ?

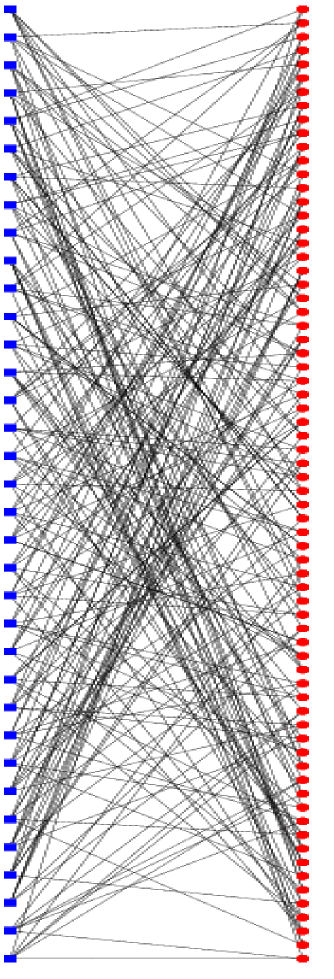
Q2. **Anomaly detection (AD)**

- Given a query node *q* in V_1, what are the normality scores for nodes in V_2 that link to *a* ?



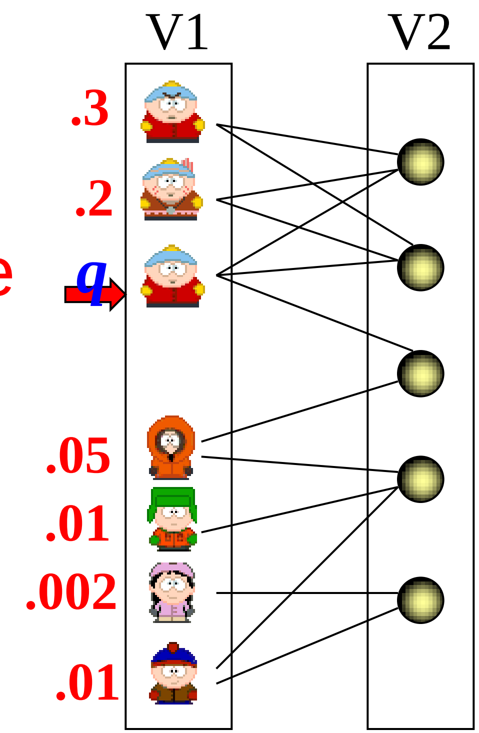
Applications of problem setting

- Publication network
  - (similar) authors vs. (unusual) papers
- P2P network
  - (similar) users vs. ("cross-border") files
- Financial trading network
  - (similar) stocks vs. (cross-sector) traders
- Collaborative filtering
  - (similar) users vs. ("cross-border") products



1) Neighborhood formation

- Main idea:
  - Random-Walk-with Restart from q
  - Steady-state V1 prob.s as relevance

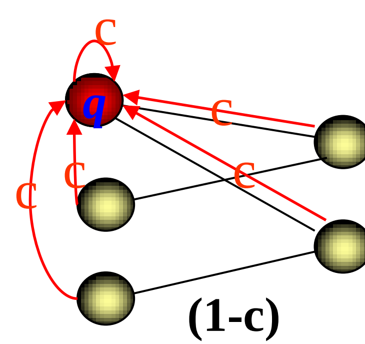


  - (1) Construct transition matrix P

$$P(a,b) = \begin{cases} \frac{1-c}{\text{outdeg}(a)} & \text{if } (a,b) \in E \\ 0 & \text{if } (a,b) \notin E \end{cases}$$

  - (2) Fly-back prob. c to q
  - (3) Solve for steady state

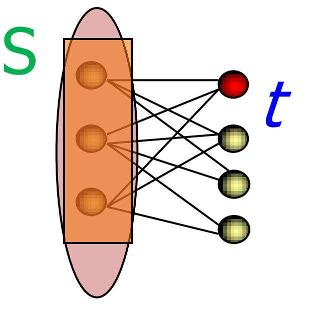
$$\vec{u}_a^{(t+1)} = P\, \vec{u}_a^{(t)} + c\vec{q}$$



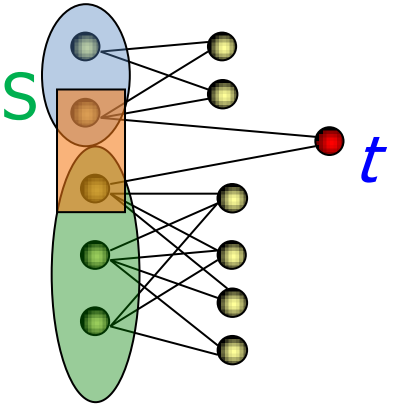
Approx: RWR on graph **partition** containing q

2) Anomaly detection

- Main idea:
  - Pairwise "normality" scores of neighbors(t)
  - Function of (e.g. avg) pair-wise scores



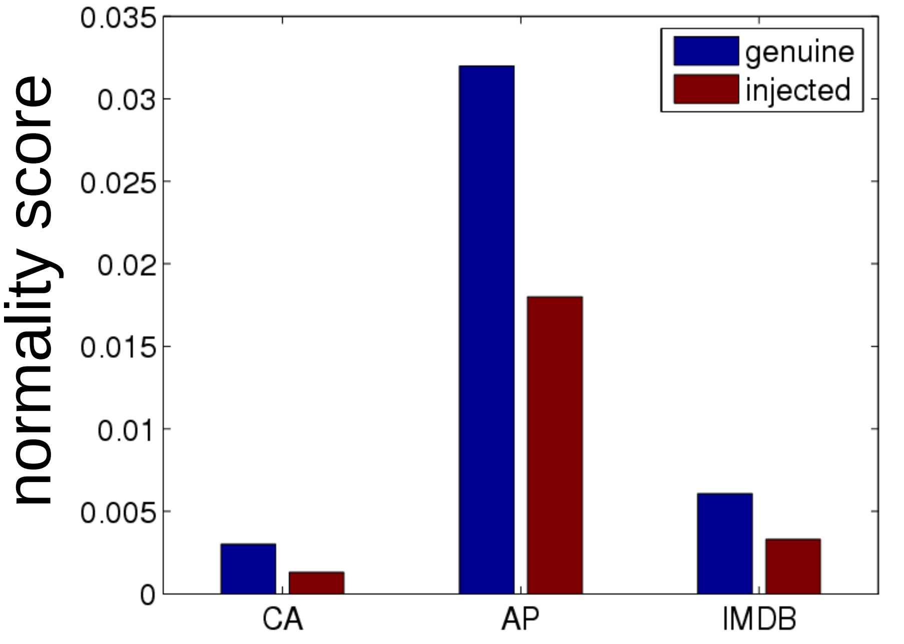
  - (1) Find set S of nodes connected to t
  - (2) Compute |S|x|S| normality matrix R
    - asymmetric, diagonal reset to 0
  - (3) Apply score function f(R)
    - e.g. f(R) = mean(R)



Sun+ICDM'05
modified with permission

Experiment

- 3 real datasets
  - DBLP conf-auth
  - DBLP auth-paper
  - IMDB movie-actor



- Randomly inject 100 nodes, each with k (avg. degree) edges (biased towards high-degree nodes)
- No qualitative results

Tong et al. '11

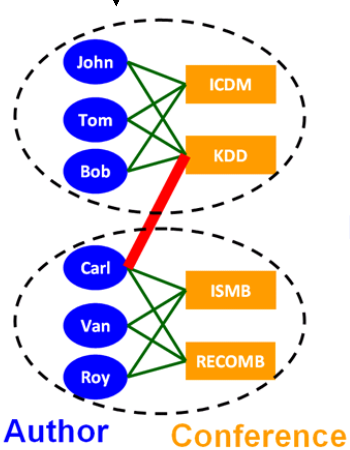
Graph Anomalies by NNrMF

- Low-rank adjacency matrix factorization of a (sparse) graph reveals communities and anomalies

Low-rank matrices   Residual matrix

Graph → Adj. Matrix A → $A = F \times G + R$

community   anomalies



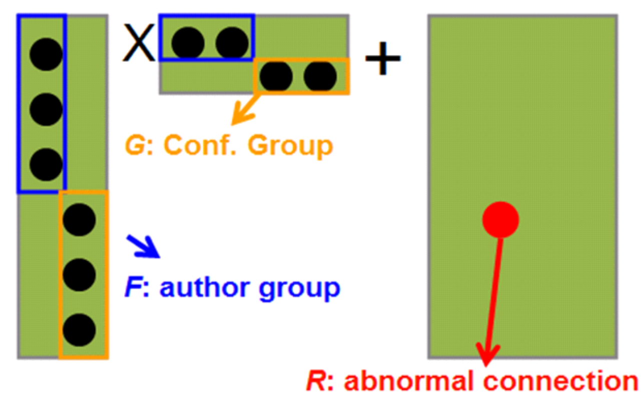
Conference

Author

1	1	0	0
1	1	0	0
1	1	0	0
0	1	1	1
0	0	1	1
0	0	1	1

Adjacency matrix: *A*

=



Non-negativity constraints

- For improved interpretability

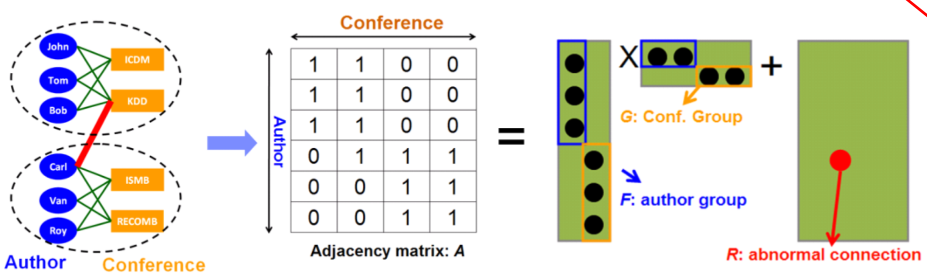
- A Typical Procedure:

community

Graph → Adjacency Matrix A → $A = F \times G + R$

anomalies

- An Example



1	1	0	0
1	1	0	0
1	1	0	0
0	1	1	1
0	0	1	1
0	0	1	1

Interpretation by Non-negativity

Non-negative Matrix Factorization

$F >= 0$; $G >= 0$

(for community detection)

Non-negative Residual Matrix Factorization

$R(i,j) >= 0$; for $A(i,j) > 0$

(for anomaly detection)

Tong+SDM'11
modified with permission

Optimization formulation

Common in Matrix Factorization

$$\operatorname{argmin}_{\mathbf{F},\mathbf{G}} \sum_{i,j,\ \mathbf{A}(i,j)>0} (\mathbf{A}(i,j) - \mathbf{F}(i,:)\mathbf{G}(:,j))^2$$

$$\text{s.t.} \quad \text{for all } \mathbf{A}(i,j) > 0 :$$

Non-negative residual →

$$\mathbf{F}(i,:)\mathbf{G}(:,j) \leq \mathbf{A}(i,j)$$

- Q: How to find 'optimal' **F** and **G**?
  - D1: Quality ←→ C1: objective non-convex
  - D2: Scalability ←→ C2: large graph size

Optimization: batch

details

- Basic Idea 1: Alternating

$$\mathrm{argmin}_{\mathbf{F},\mathbf{G}} \sum (\mathbf{A}(i,j) - \mathbf{F}(i,:)\mathbf{G}(:,j))^2$$

Not convex w.r.t. *F* and *G*, *jointly*
But convex if fixing either *F* or *G*

- Basic Idea 2: Separation

$$\mathrm{argmin}_G \sum_{i,j,\ \mathbf{A}(i,j)>0} (\mathbf{A}(i,j) - \mathbf{F}(i,:)\mathbf{G}(:,j))^2$$

s.t..
for all $\mathbf{A}(i,j) > 0$:
$\mathbf{F}(i,:)\mathbf{G}(:,j) \le \mathbf{A}(i,j)$

For each *j*

$$\mathrm{argmin}_G \sum_{j,\ \mathbf{A}(i,j)>0} (\mathbf{A}(i,j) - \mathbf{F}(i,:)\mathbf{G}(:,j))^2$$

s.t.. for all $\mathbf{A}(i,j) > 0$:
$\mathbf{F}(i,:)\mathbf{G}(:,j) \le \mathbf{A}(i,j)$

Standard Quadratic Programming

Overall Complexity: Polynomial

Optimization: incremental

details

- Basic Idea 0: Recursive
- Basic Idea 1: Alternating

$$\operatorname{argmin}_{\mathbf{f},\mathbf{g}} \sum_{i,j,\ \mathbf{A}(i,j)>0} (\mathbf{A}(i,j) - \mathbf{f}(i)\mathbf{g}(j))^2$$

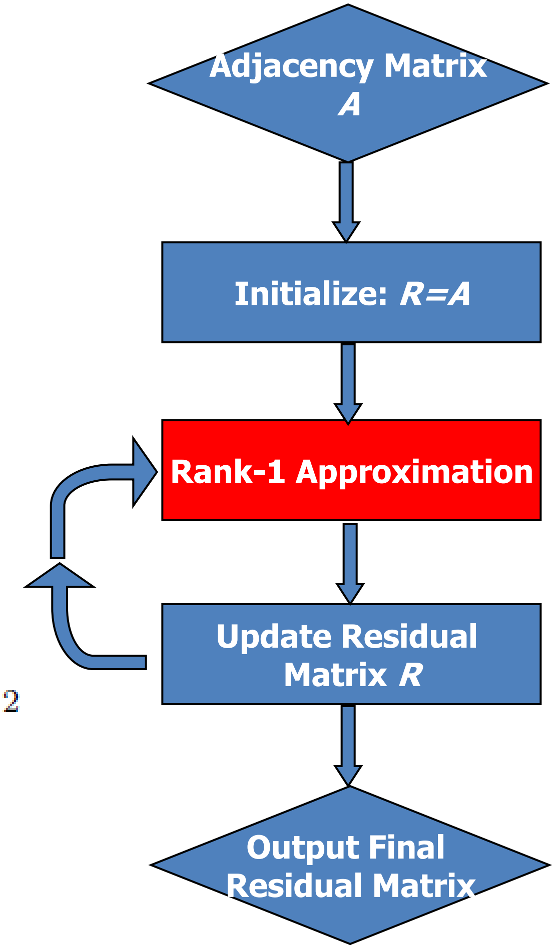
- Basic Idea 2: Separation

QP for a single variable
w/ boundary constrains

Solved in
constant time

$$\operatorname{argmin}_{\mathbf{g}_j} \sum_{i,\ \mathbf{A}(i,j)>0} (\mathbf{A}(i,j) - \mathbf{f}(i)\mathbf{g}(j))^2$$

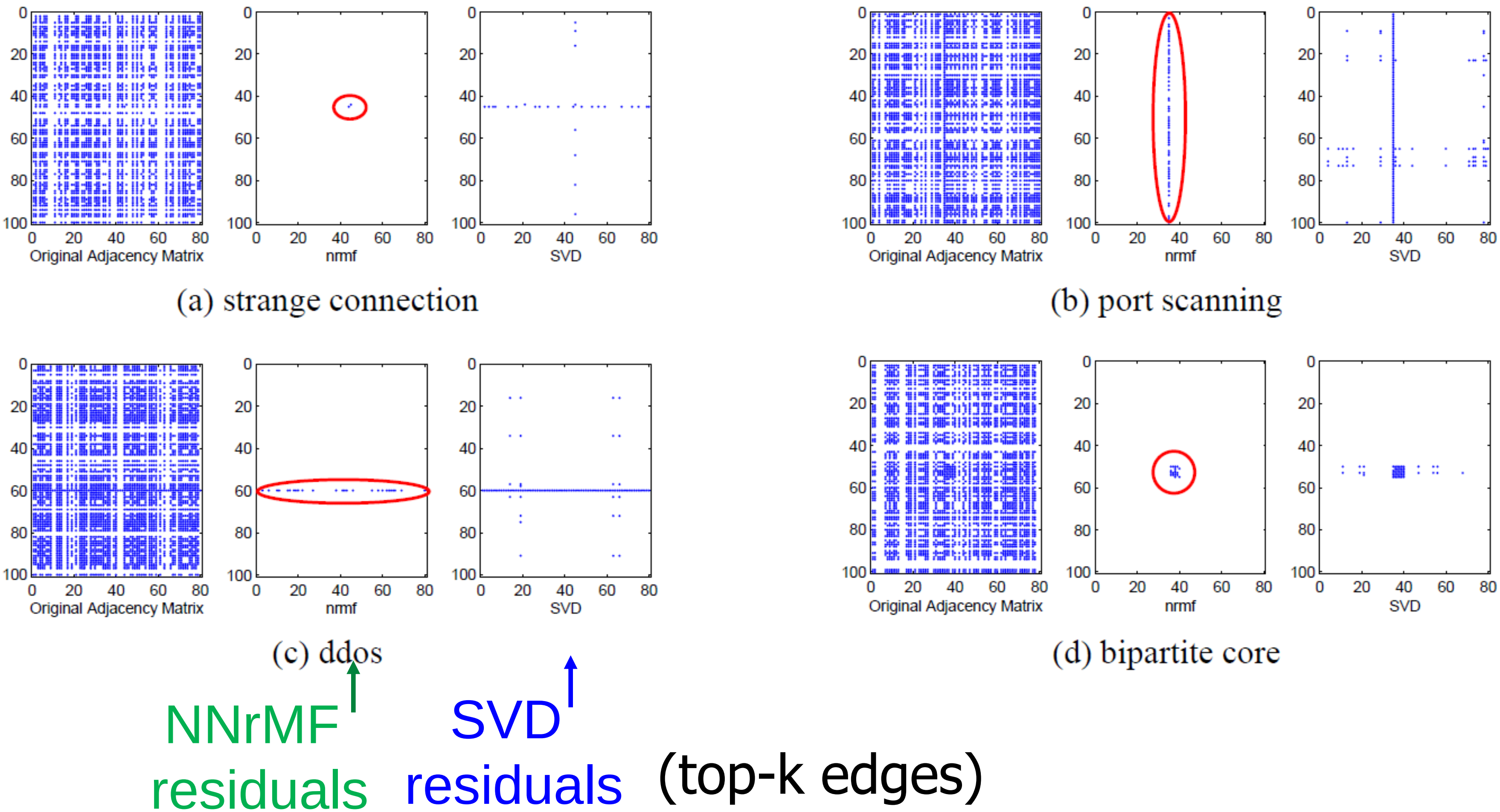
$$\text{s.t.} \quad \text{for all } \mathbf{A}(i,j) > 0: \quad \mathbf{f}(i)\mathbf{g}(j) \leq \mathbf{A}(i,j)$$



Overall Complexity: Linear wrt # of edges

Experiments

- NNrMF can spot 4 types of anomalies



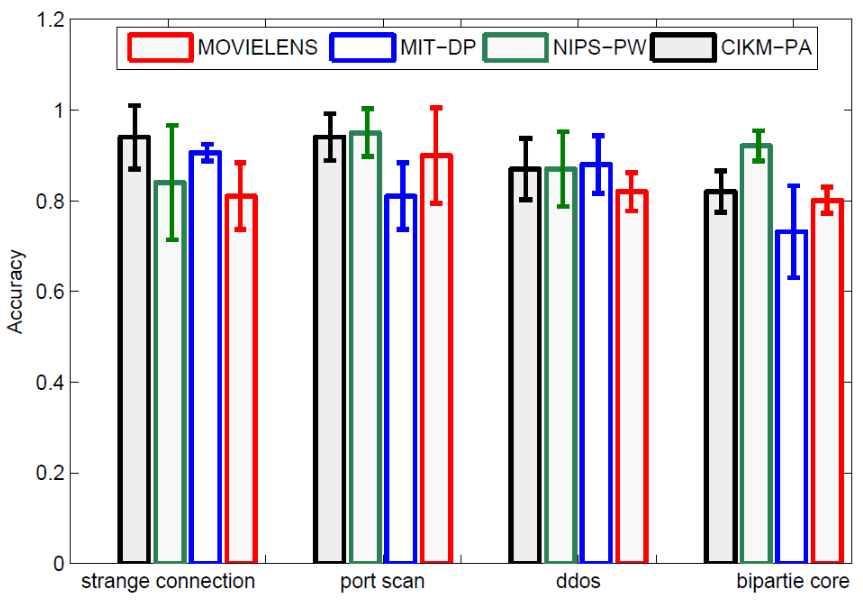
(top-k edges)

Experiments

- 4 real datasets, with injected anomalies

Effectiveness

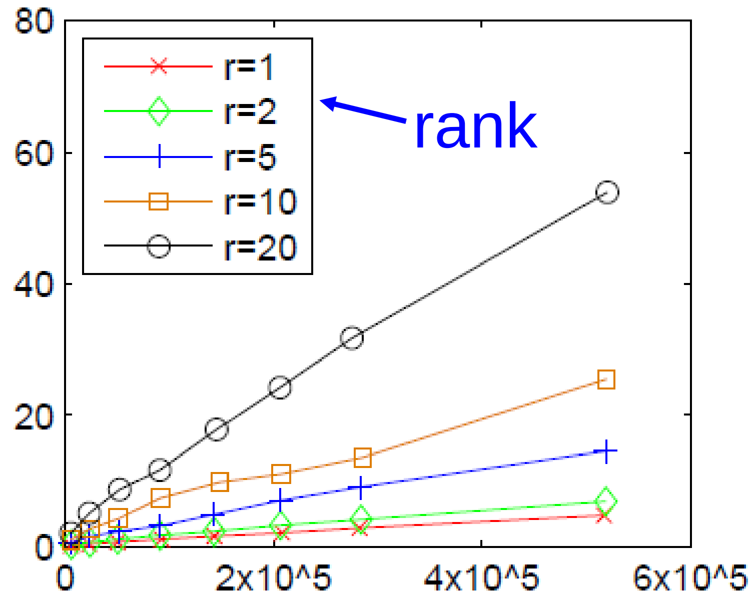
Accuracy



Anomaly Type

Efficiency

Wall-clock Time



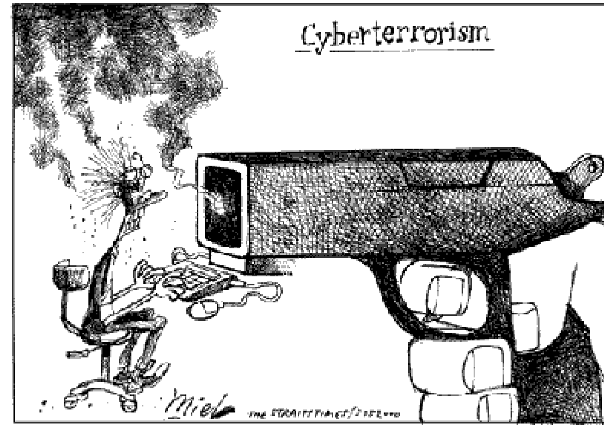
\# of edges

Ding et al. '12

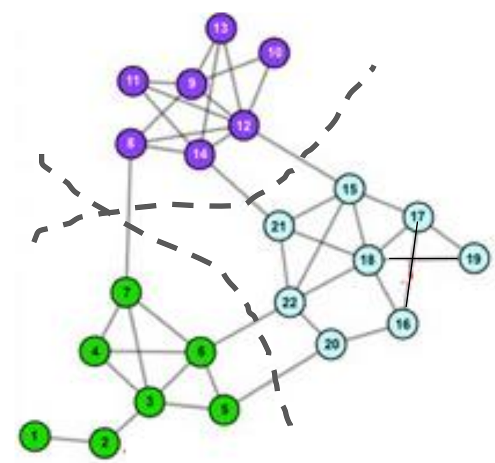
Intrusion as (Anti)social Communication

- Problem:

Q. How to detect malicious attacks in computer networks?



- Main insight for intrusion:
  - entering a community to which one doesn't belong
  - look for communication that does not respect **community boundaries**

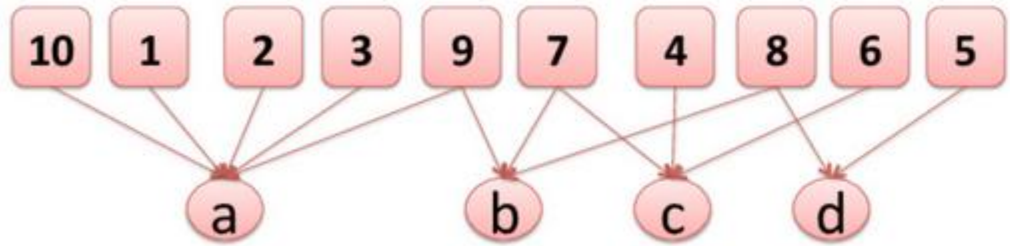


Problem formulation

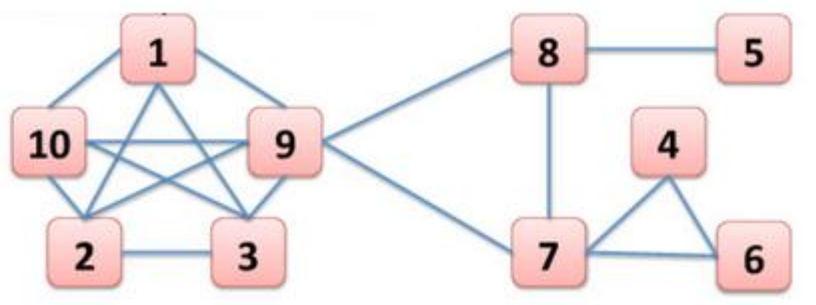
- Network representation as a bipartite graph

Source IPs ⟶

Dest. IPs ⟶



  - Source and destination IPs may overlap
- One mode projection G_P: connect two source IPs with at least 1 common neighbor
- Alternative G_W: weigh by correlation coefficient



Intrusion data with ground truth

- Data: netflow traffic
  - from a large European ISP
  - 2 weeks data in 2007: source IP, dest IP, start/end time, number of bytes/packets sent
  - Ground truth: traffic sources that attempted an intrusion as recorded by Dshield*
    - known IPs sending malicious or unwanted traffic

* http://www.dshield.org/

Detection methods

- **Community detection:** Standard community detection methods fail to distinguish known IPs from communities

Size of Cluster	# of Clusters	# of DShields
6784	1	158
986	1	1
8 to 243	10	0
≤ 7	56	2
Total	68	161

- **Cut-vertices:**

  Iteratively remove cut-vertices

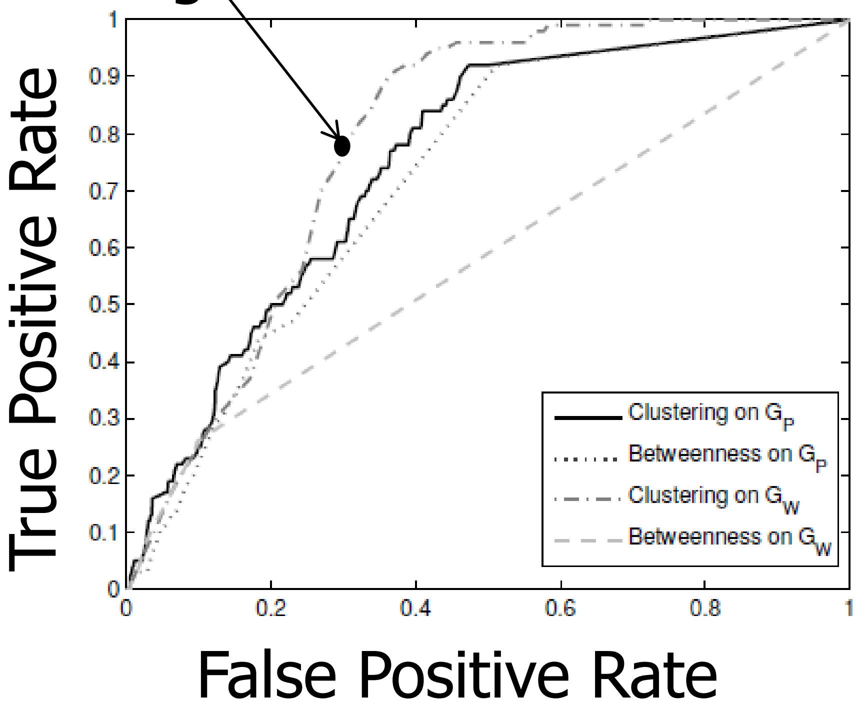
  - 6.6% of cut-vertices are Dshields (randomization yields significance; (1-2.2%) at 0.05)

→ **Clustering** and **betweenness** deemed discriminative

Experiments

- Malicious if clustering/betweenness below/above threshold

for a given threshold



	Mean(AUC)	SE(AUC)
Clustering on G_P	0.7440	0.0103
Betweenness on G_P	0.7180	0.0084
Clustering on G_W	0.7625	0.0080
Betweenness on G_W	0.5621	0.0034

- Clustering gives better discrimination
- GW does not provide much improvement over GP

Feature mining

Community mining

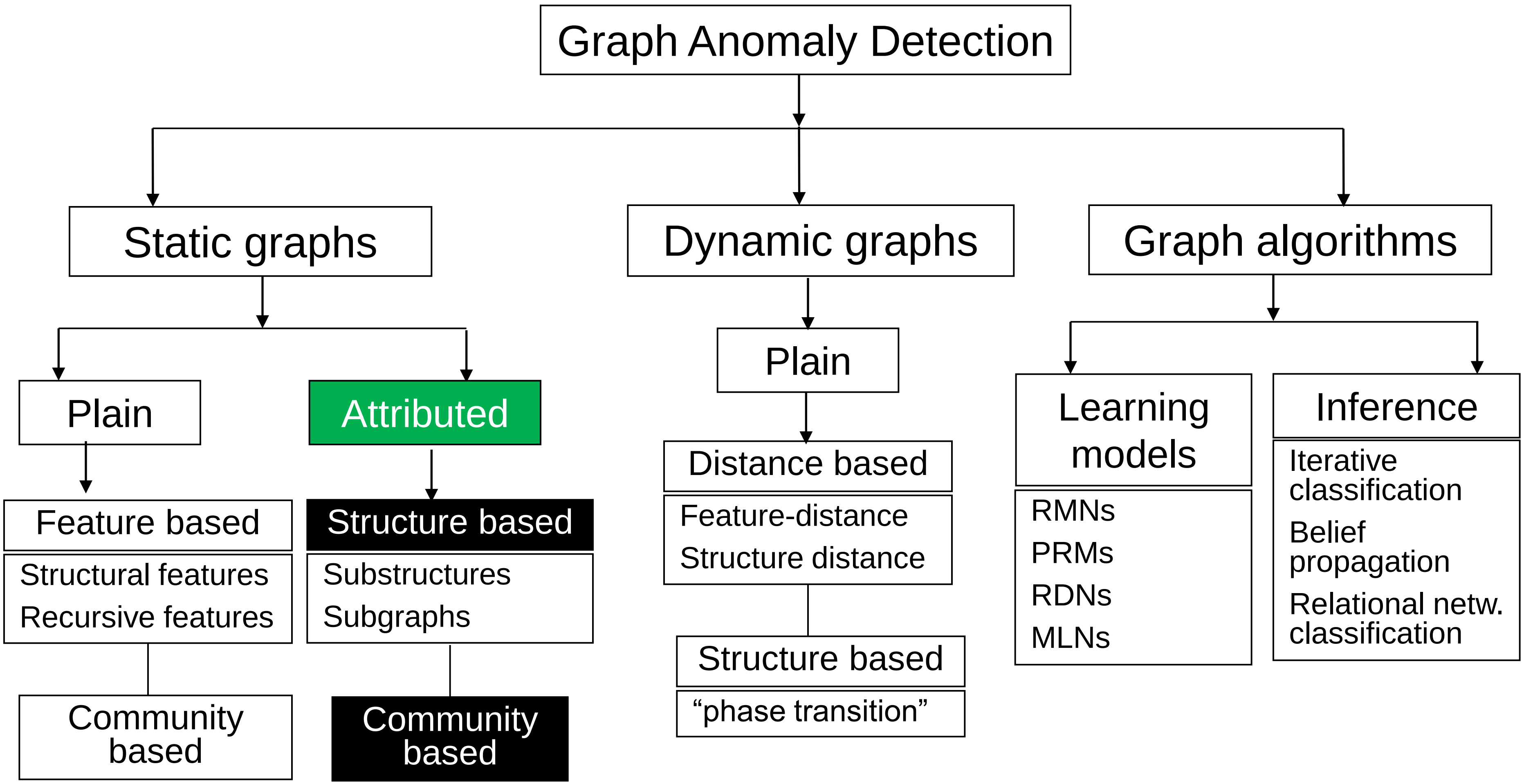
Part I: References (plain graphs)

- L. Akoglu, M. McGlohon, C. Faloutsos. OddBall: Spotting Anomalies in Weighted Graphs. PAKDD, 2010.
- K. Henderson, B. Gallagher, L. Li, L. Akoglu, T. Eliassi-Rad, H. Tong, C. Faloutsos. It's Who You Know: Graph Mining Using Recursive Structural Features. KDD, 2011.
- J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. ICDM, 2005.
- Hanghang Tong, Ching-Yung Lin: Non-Negative Residual Matrix Factorization with Application to Graph Anomaly Detection. SDM, pages 143-153, 2011.
- Q. Ding, N. Katenka, P. Barford, E. Kolaczyk, and M. Crovella. Intrusion as (Anti)social Communication: Characterization and Detection. KDD, 2012.

Part I: Outline

- Overview: Outliers in **clouds of points**
  - Outliers in **numerical** data points
    - distance-based, density-based, …
  - Outliers in **categorical** data points
    - model-based
- Anomaly detection in **graph data**
  - Anomalies in unlabeled, **plain** graphs
  - Anomalies in node-/edge-labeled, **attributed** graphs

Taxonomy

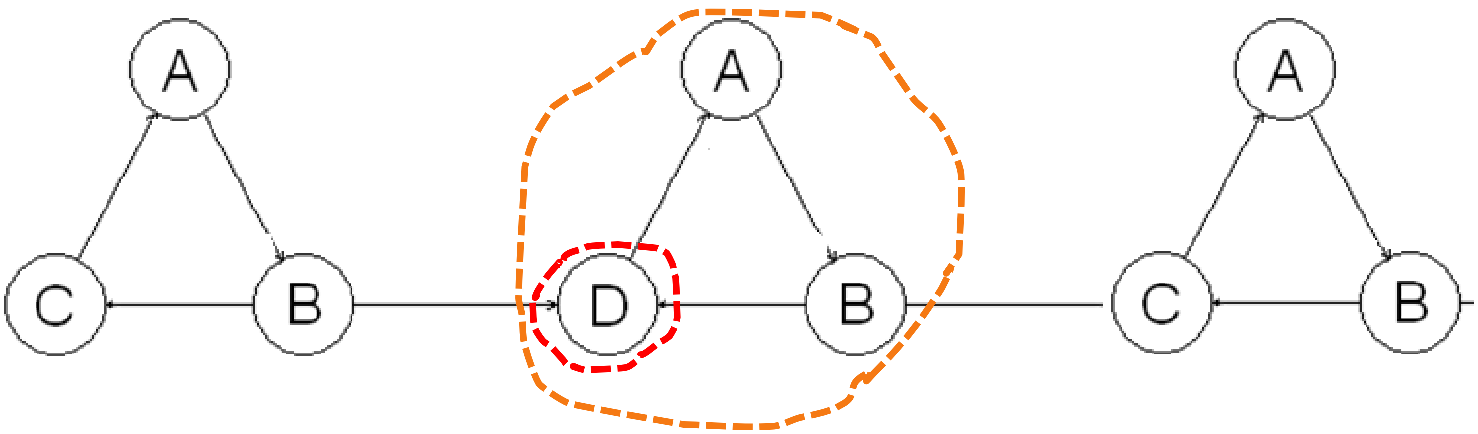

Graph Anomaly Detection
Static graphs
Dynamic graphs
Graph algorithms
Plain
Attributed
Feature based
Structural features
Recursive features
Structure based
Substructures
Subgraphs
Community based
Community based
Plain
Distance based
Feature-distance
Structure distance
Structure based
“phase transition”
Learning models
RMNs
PRMs
RDNs
MLNs
Inference
Iterative classification
Belief propagation
Relational netw. classification

Anomalies in labeled graphs

- Problem:

Q1. Given a graph in which nodes and edges contain (non-unique) labels, what are unusual substructures?

Q2. Given a set of subgraphs, what are the unusual subgraphs?



Note: assumption is anomalies are connected

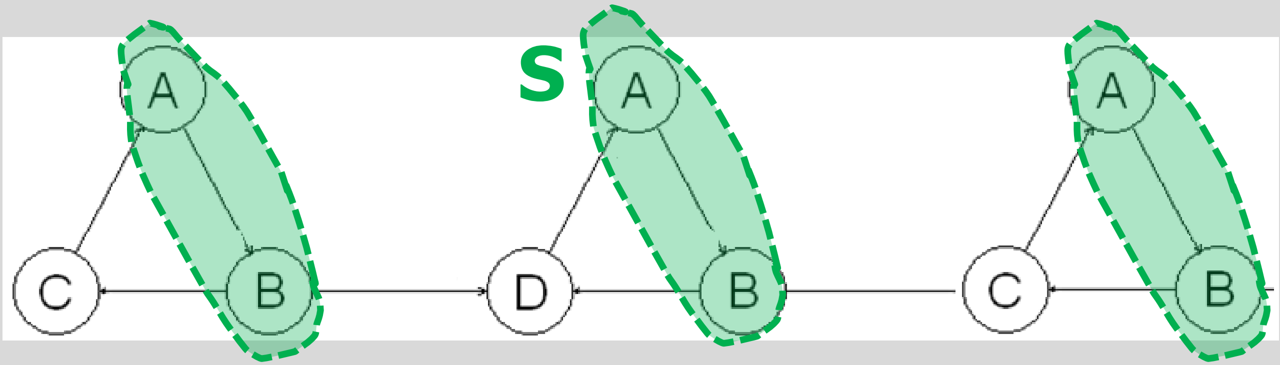
Background

- Subdue*: An algorithm for detecting repetitive patterns (substructures) within graphs.
- Substructure: A connected subgraph of the overall graph.
- Compressing a graph: Replacing each instance of the substructure with a new vertex representing that substructure.
- Description Length (DL): Number of bits needed to encode a piece of data

* http://ailab.wsu.edu/subdue/

Background

- Subdue uses the following heuristic:
  - The best substructure is the one that **minimizes**
    F1(S,G) = DL(G | S) + DL(S)
    - G: Entire graph, S: The substructure,
    - DL(G|S) is the DL of G after compressing it using S,
    - DL(S) is the description length of the substructure.



- Iterations after compressing at each step

Background

Given database D and set of models for D, **Minimum Description Length** selects model M that minimizes

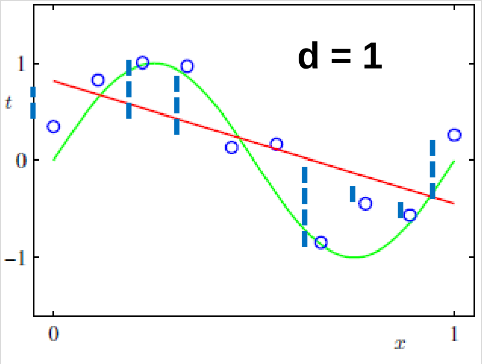
$$\underbrace{L\,(M)}_{\text{length in bits: description of model } M} + \underbrace{L\,(D|M)}_{\text{length in bits: data, encoded by } M}$$

length in bits: description of **model** M

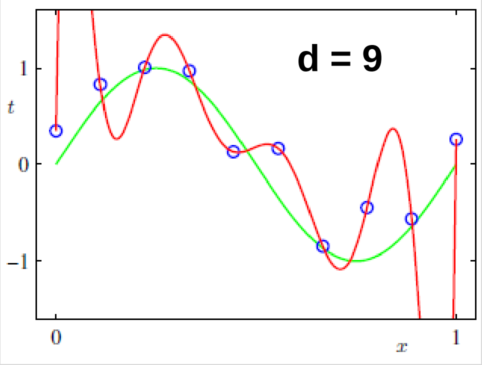
length in bits: **data**, encoded by M

⇩ a_1x+a_0 vs. $a_9x^9+\ldots+a_1x+a_0$

⇩ deltas vs. {}



vs.



Bishop: PR&ML

1) Anomalous Substructures

- Main idea: anomalies (by def.) occur infrequently, they are roughly opposite to "best substructures"
  - Find substructures **S** that maximize **F1(S,G)**?
    - Nope, it flags all single nodes as anomalies!
  - Instead, find those that **minimize**

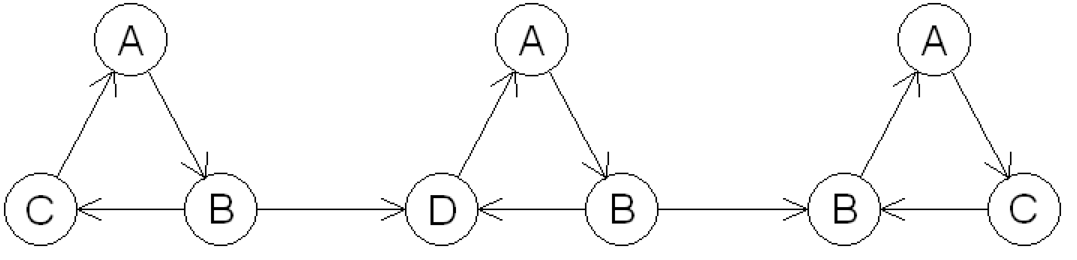
    F2(S, G) = Size(S) * Instances(S,G)

    - Approximate inverse of **F1(S,G)**

- Intuition: Larger substructures are <u>expected</u> to occur few times; the smaller the substructure, the less likely it is rare

Example

- F2(S, G) = Size(S) * Instances(S,G)
  - For node D, F2 = 1 * 1 = 1
  - For A→C and D→A, it is 2 * 1 = 2
  - For G (whole graph), it is 9 * 1 = 9
- Hence D is considered the most anomalous.



- Note: Usually a threshold for F2 is used and anomalies are ranked by their scores.

2) Anomalous Subgraphs

- Main idea: subgraphs containing few common substructures are generally more anomalous
  - Define compressibility score ***A*** in [0,1]

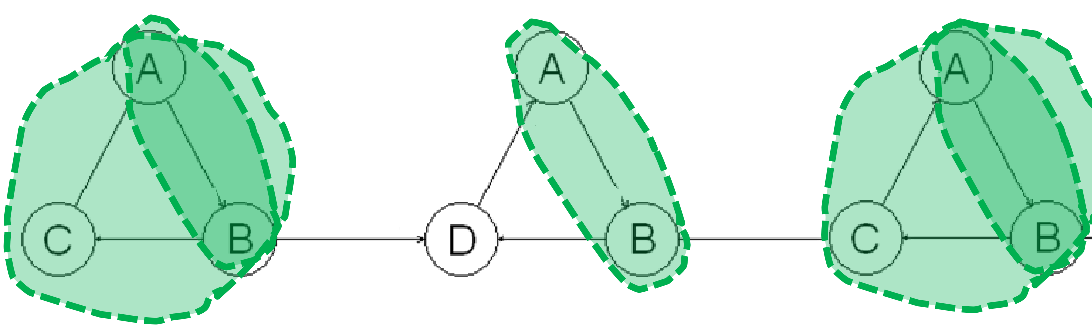
$$A = 1 - \frac{1}{n}\sum_{i=1}^{n}(n-i+1) * c_i$$

Subdue iterations

fast drop off in early iterations

fraction compressed at *ith* iteration

$$\boxed{\frac{DL_{i-1}(G) - DL_i(G)}{DL_0(G)}}$$



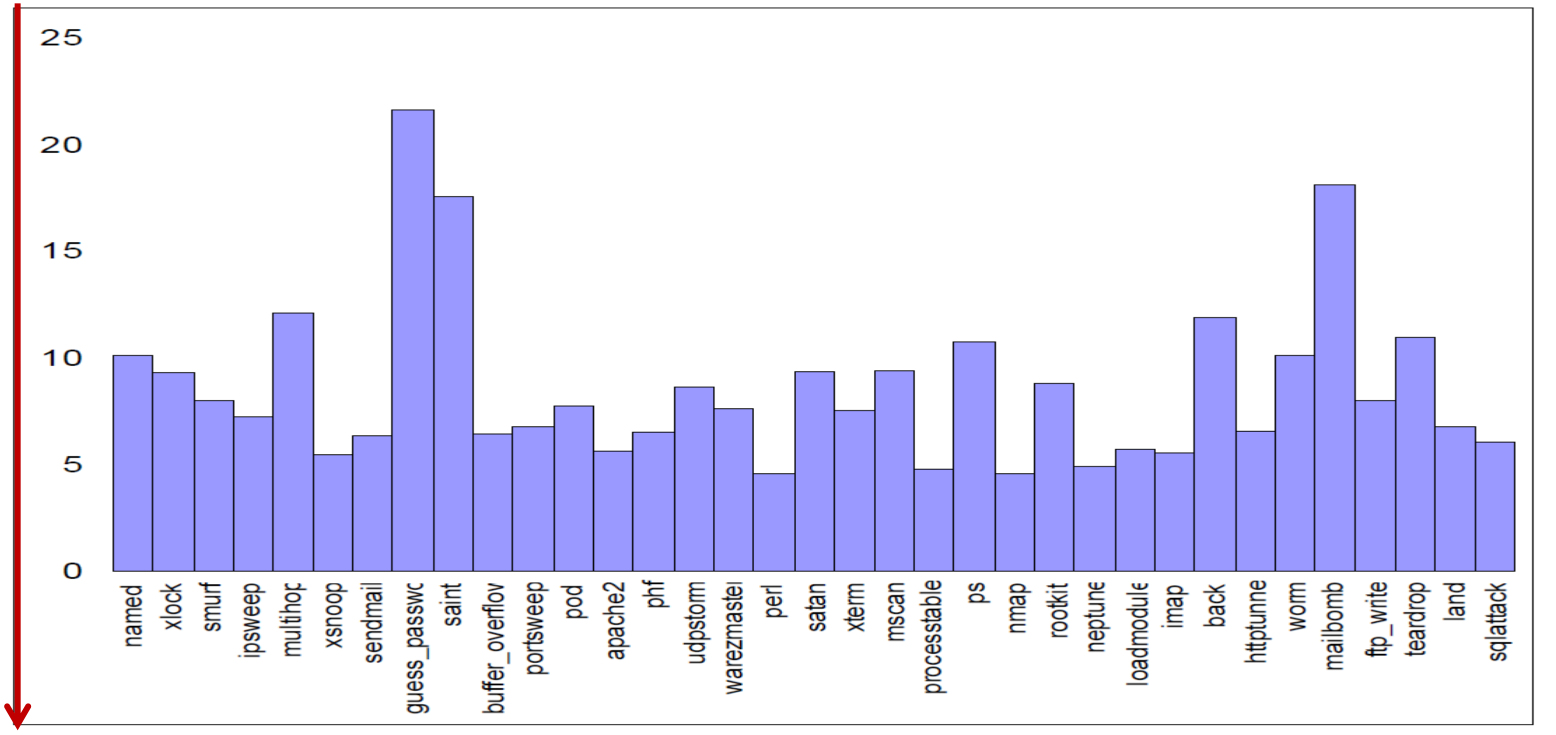
Experiments

- Data: 1999 KDD Cup Network Intrusion
  - Ground truth: connection records, “normal” or attack (37 types), 41 features of connection (duration, protocol type, number of bytes, etc.)
  - Each individual test involved 50 records of which only one is of a particular attack type.

- Use Subdue to find anomalous substructures
  - Prune all subgraphs with size>3, F2>6 (arbitrary)

Performance

Lower is better



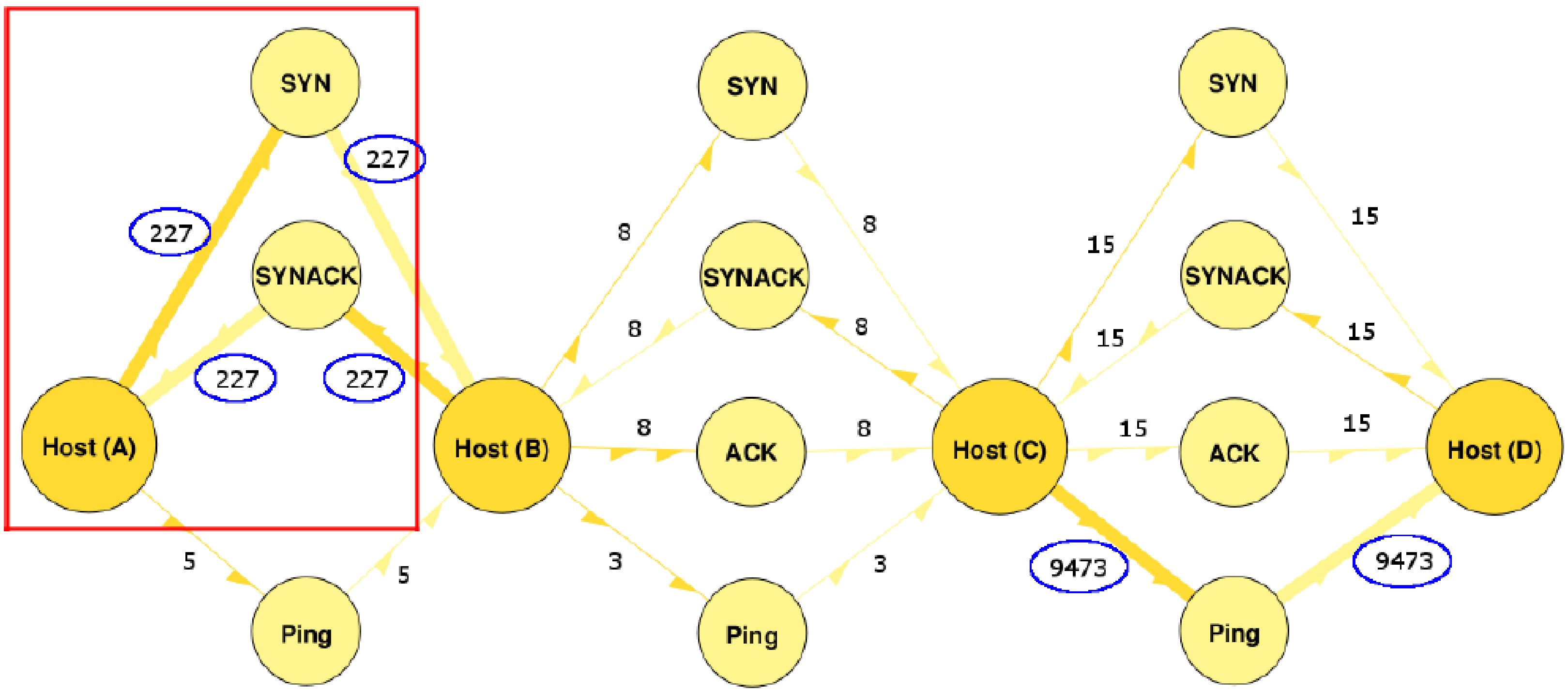
- Note: Degree of anomaly D(S): 1/F2
  - Attack accounts for D(S1) / (Sum [D(Si)]
  - e.g., if F2 = (2, 3, 4) for (S1, S2, S3) and S2 occurs in the attack, then attack accounts for (1/3) / (1/2 + 1/3 + 1/4) = 4/13 of discovered anomalies

Davis et al. '11

Anomalies with numeric labels

- How about **numeric** labels?
  - Noble & Cook work with **categorical** labels

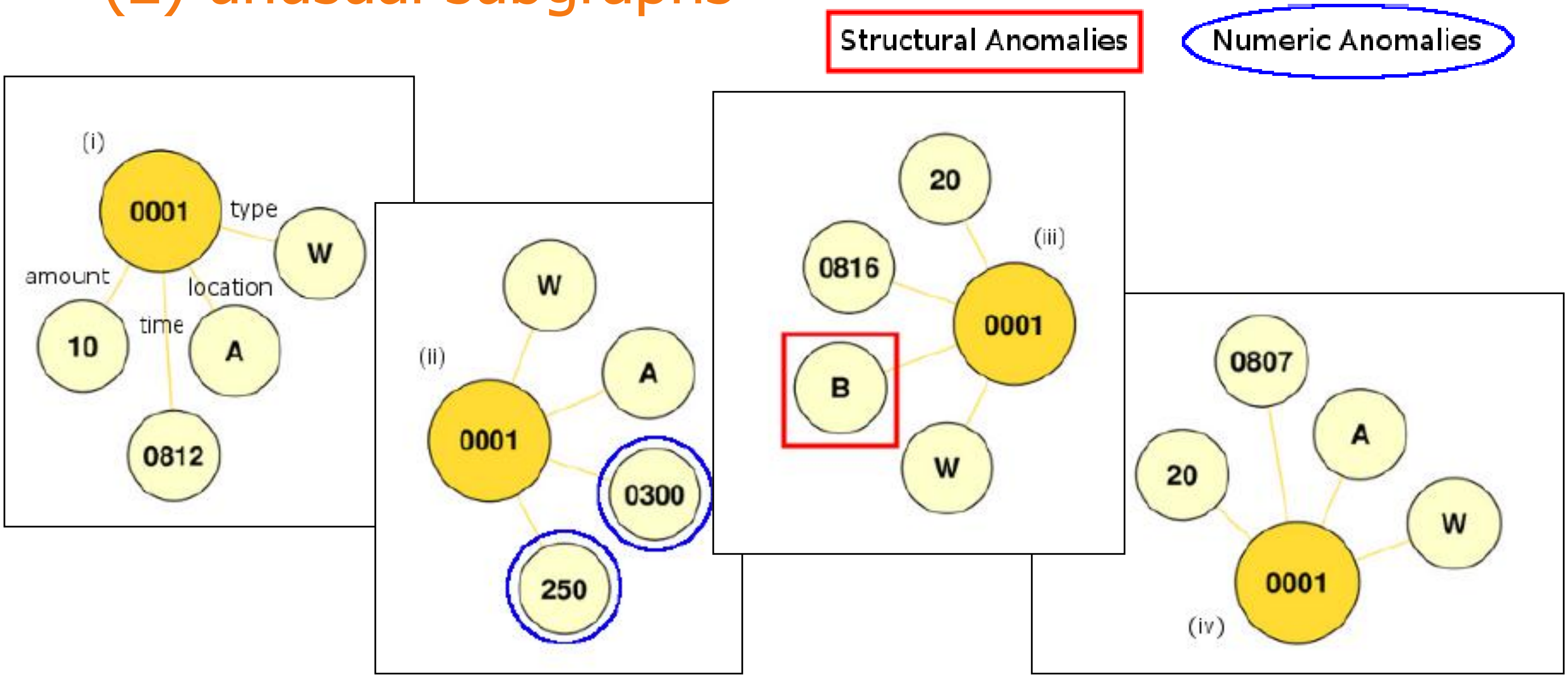
(1) unusual substructures



Anomalies with numeric labels

- How about **numeric** labels?
  - Noble & Cook work with **categorical** labels

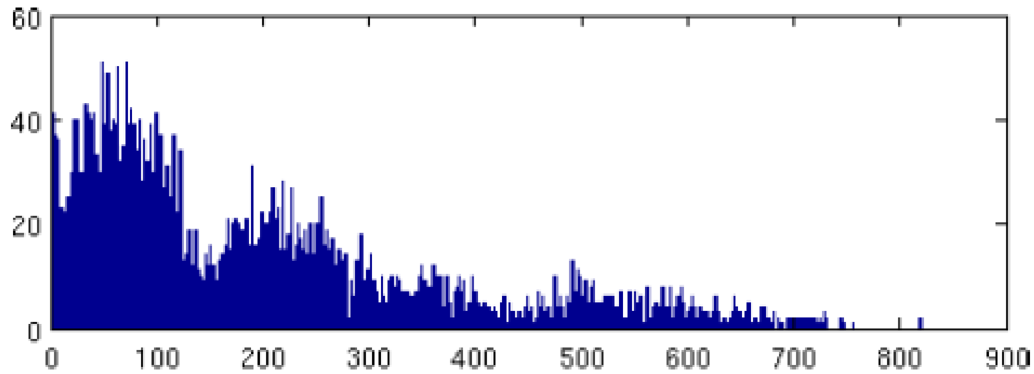
(2) unusual subgraphs



Anomalies with numeric labels

- Main idea (discretization):
  - assign categoric label q_0 to "normal" values, and
  - "outlierness" score q_i to all others i

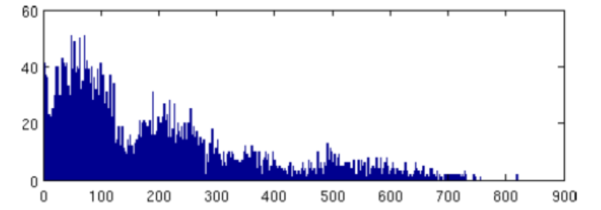
- Example: empirical distribution of a label

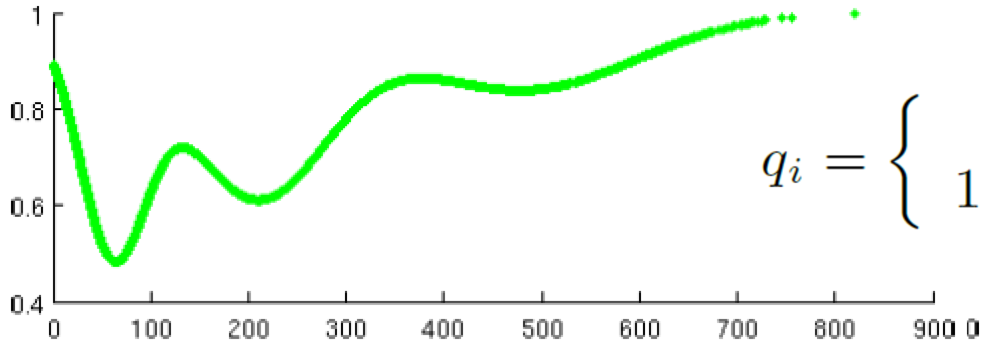


- Several "outlierness" scores (pdf-fitting, kNN, LOF, clustering-based)

Discretization

- Model fitting (GMM)

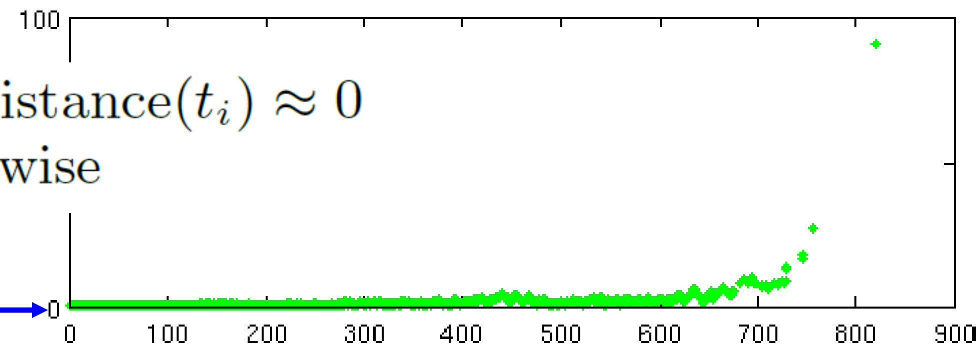




$$q_i = \begin{cases} q_0 & \text{if } 1 - P(t_i) < q_a \\ 1 - P(t_i) & \text{otherwise} \end{cases}$$

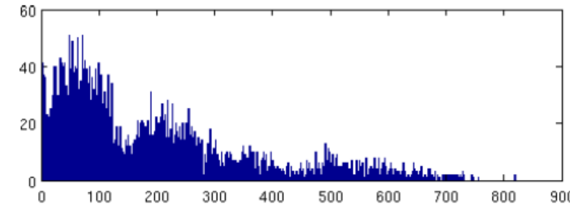
- kNN distance

$$q_i = \begin{cases} q_0 & \text{if k-distance}(t_i) \approx 0 \\ \text{k-distance}(t_i) & \text{otherwise} \end{cases}$$



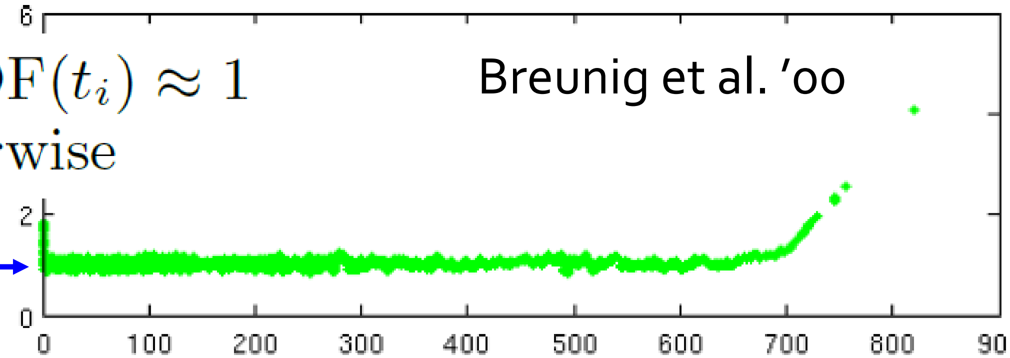
Discretization

- Density outlier score (LOF)

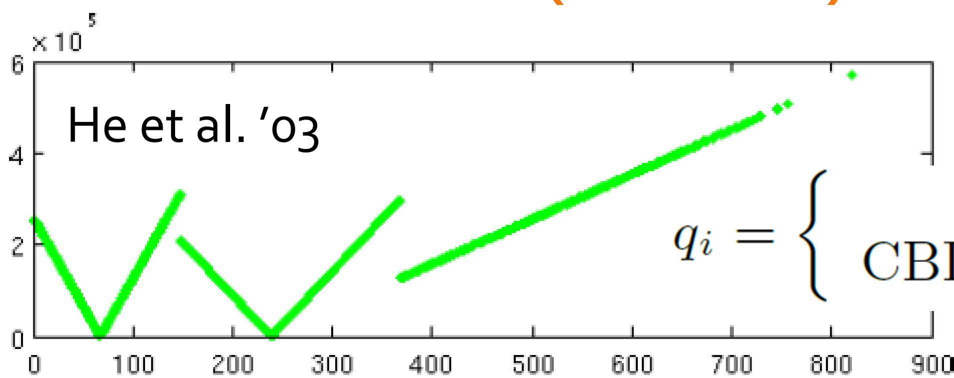


$$q_i = \begin{cases} q_0 & \text{if } \mathrm{LOF}(t_i) \approx 1 \\ \mathrm{LOF}(t_i) & \text{otherwise} \end{cases}$$

normal →



- Cluster-based (CbLOF)



$$q_i = \begin{cases} q_0 & \text{if } \mathrm{CBLOF}(t_i) < q_a \\ \mathrm{CBLOF}(t_i) & \text{otherwise} \end{cases}$$

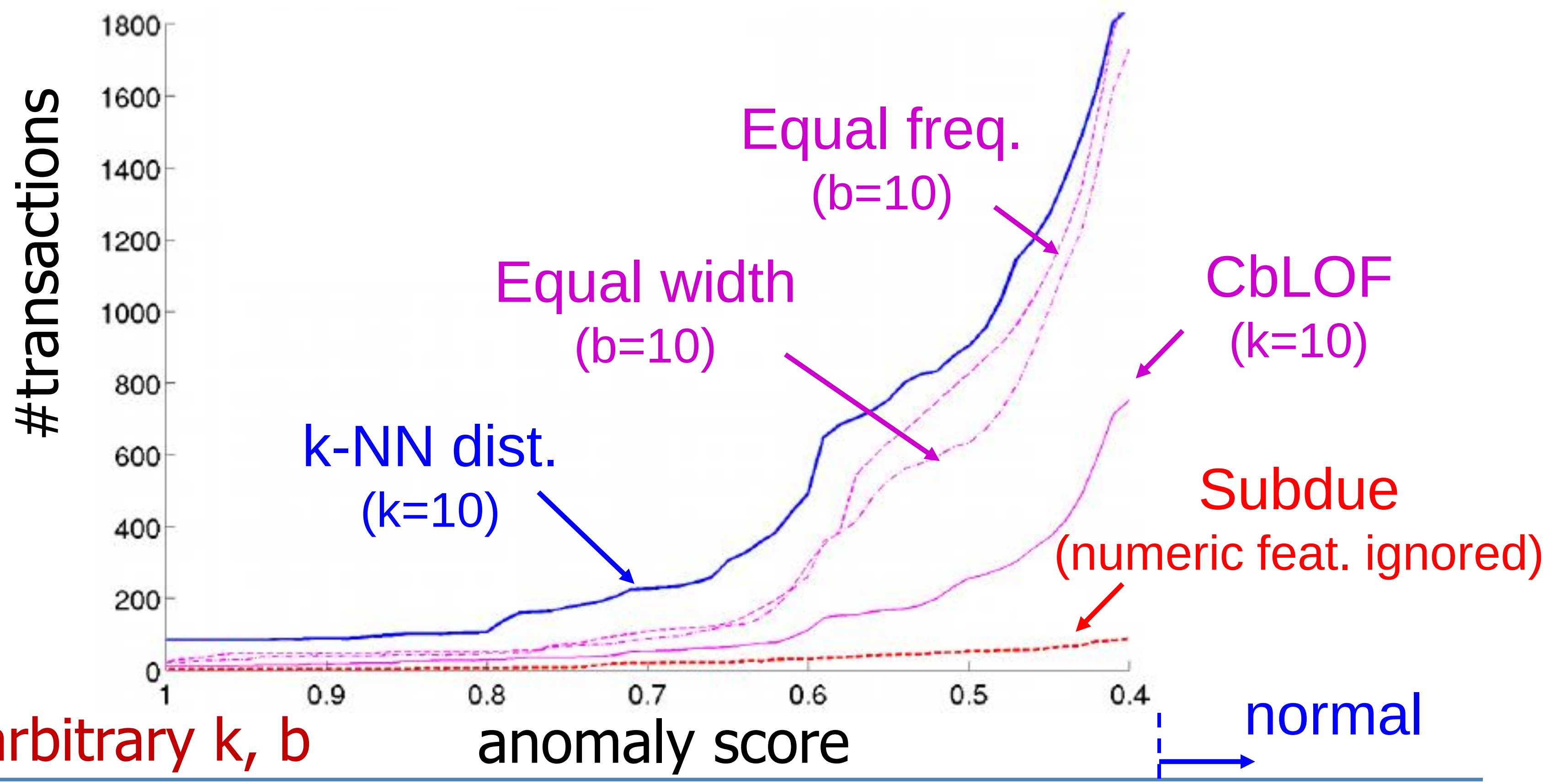
distance to closest “large” (k-means) cluster centroid

Discretization

- Other possible discretization techniques
  - SAX (**S**ymbolic **A**ggregate appro**X**imation)
    - http://www.cs.ucr.edu/~eamonn/SAX.htm
  - MDL-binning
    - P. Kontkanen and P. Myllymäki. *MDL histogram density estimation*. In AISTAT, 2007.
  - Minimum entropy discretization
    - U.M. Fayyad and K.B. Irani. *Multi-interval discretization of continuous-valued attributes for classification learning*. In Proc. IJCAI, pages 800–805, 1989.
  - Logarithmic binning
    - especially for skewed distributions

Experiment

- Data: Access card transaction graphs
  - node: door sensor, edge (u,w): movement u→w, weight(u,w): time u→w (only numeric attribute)



* arbitrary k, b

Anomalies in labeled graphs

- Problem:

Q1. **Given** a graph in which nodes and edges contain (non-unique) labels, how to **find** substructures that are very similar to, though not the same as, a normative substructure? ("best substructure" as for Subdue)*

- Intuition:

"The more successful money-laundering apparatus is in imitating the patterns and behavior of legitimate transactions, the less the likelihood of it being exposed."

– United Nations Office on Drugs and Crime

Formal definition

- Given graph *G* with a normative substructure *S*, a substructure *S'* is anomalous if difference *d* between *S* and *S'* satisfies $0 < d <= X$, where *X* is a (user-defined) threshold and *d* is a measure of the unexpected structural difference.
- Assumptions
  - Majority of *G* consists of a normative pattern, and no more than X% of it is altered in an anomaly.
  - Anomalies consist of one or more modifications, insertions or deletions.
  - Normative pattern is connected.

Three Types of Anomalies

1) GBAD-MDL (Minimum Descriptive Length):
   anomalous modifications

2) GBAD-P (Probability): anomalous insertions

3) GBAD-MPS (Maximum Partial Substructure):
   anomalous deletions

Note: prone to miss more than one type of anomaly

- e.g., a deletion followed by modification

1) Information Theoretic Approach

- Find normative substructure **S** that minimizes

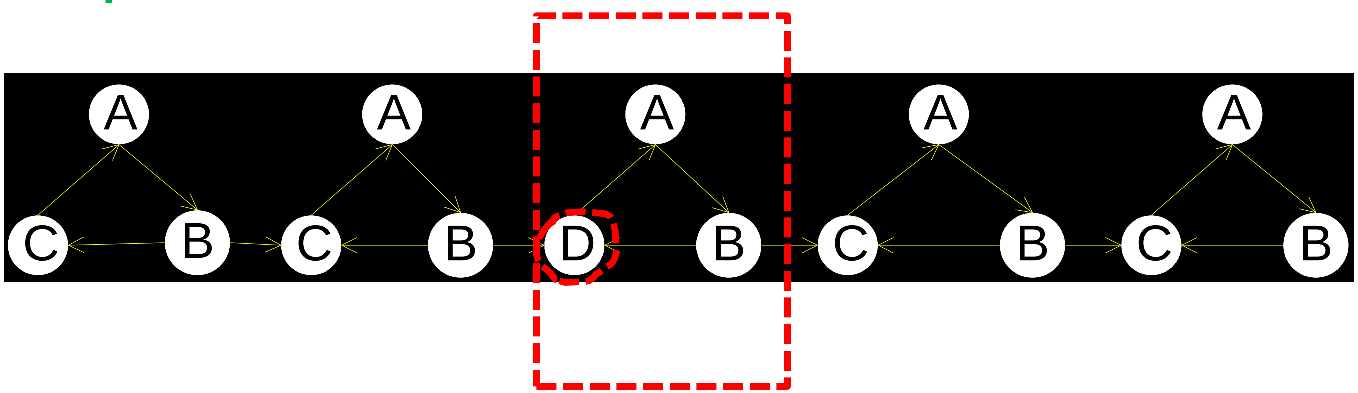
  F(S,G) = DL(G | S) + DL(S)

- For each instance I_k of **S**

$$\text{anomalyScore}(I_k) = \text{freq}(I_k) * \text{matchcost}(I_k, S)$$

cost to modify I_k into S

- Example:

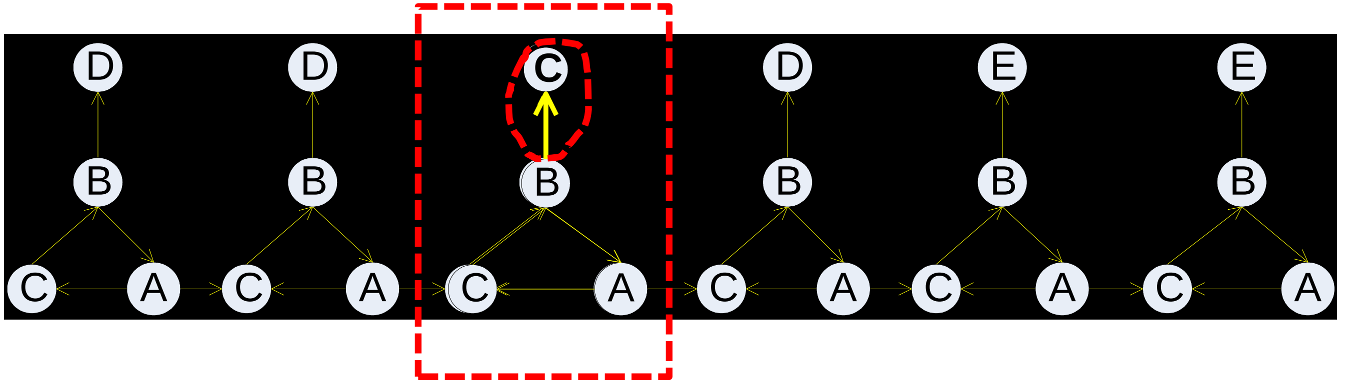


2) Probabilistic Approach

- Find normative substructure **S**
- Find extensions to **S** with lowest probability
- For each extension $\boldsymbol{I_k}$ of **S**

$$\text{anomalyScore}(I_k) = \frac{\text{number of instances of } I_k}{\text{all instances } I_n \text{ with a unique extension}}$$

- Example:



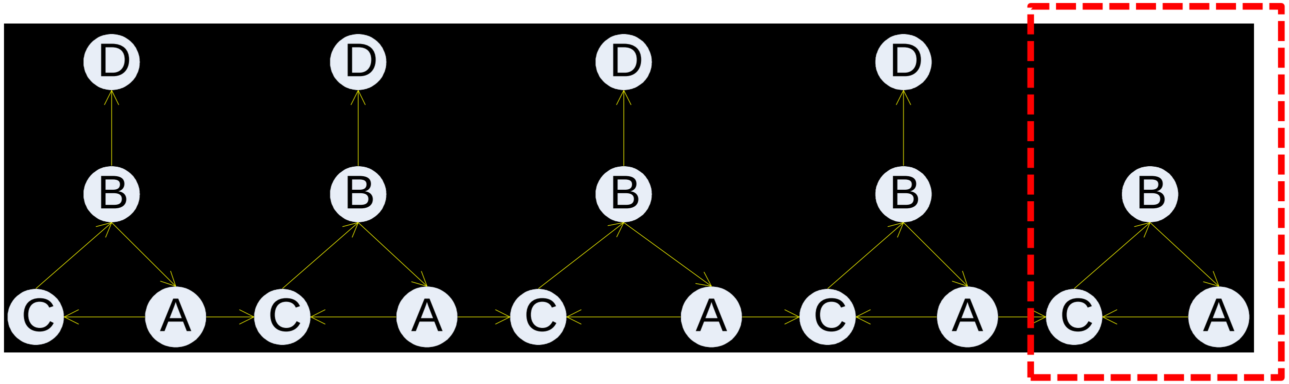
3) Maximum Partial Substructure Approach

- Find normative substructure **S**
- Find "ancestral" substructures $S_n \subseteq S$ that are missing various edges and vertices.
- For each instance I_k of $\mathbf{S}_n$

$$\text{anomalyScore}(I_k) = |\, I_n \,| * \text{matchcost}(I_k, S)$$

instances of I_k

- Example:



Experiments (Cargo shipments)

- Data: obtained from Customs and Borders Protection (CBP)
- Scenario:
  - Marijuana seized at Florida port [press release by U.S. Customs Service, 2000].
  - Smuggler did not disclose some financial information, and ship traversed extra port.
  - GBAD-P discovers the extra traversed port;
  - GBAD-MPS discovers the missing financial info.

Experiments (Network intrusion)

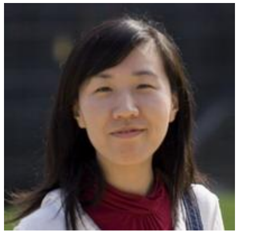
- Data: 1999 KDD Cup Network Intrusion
  - 100% of attacks were discovered with GBAD-MDL
  - 55.8% for GBAD-P and 47.8% for GBAD-MPS

  Note
  - Data consists of TCP packets that have fixed size
  - Thus, the inclusion of additional structure, or the removal of structure, is not relevant here.
  - Modification is the only relevant one, at which GBAD-MDL performs well

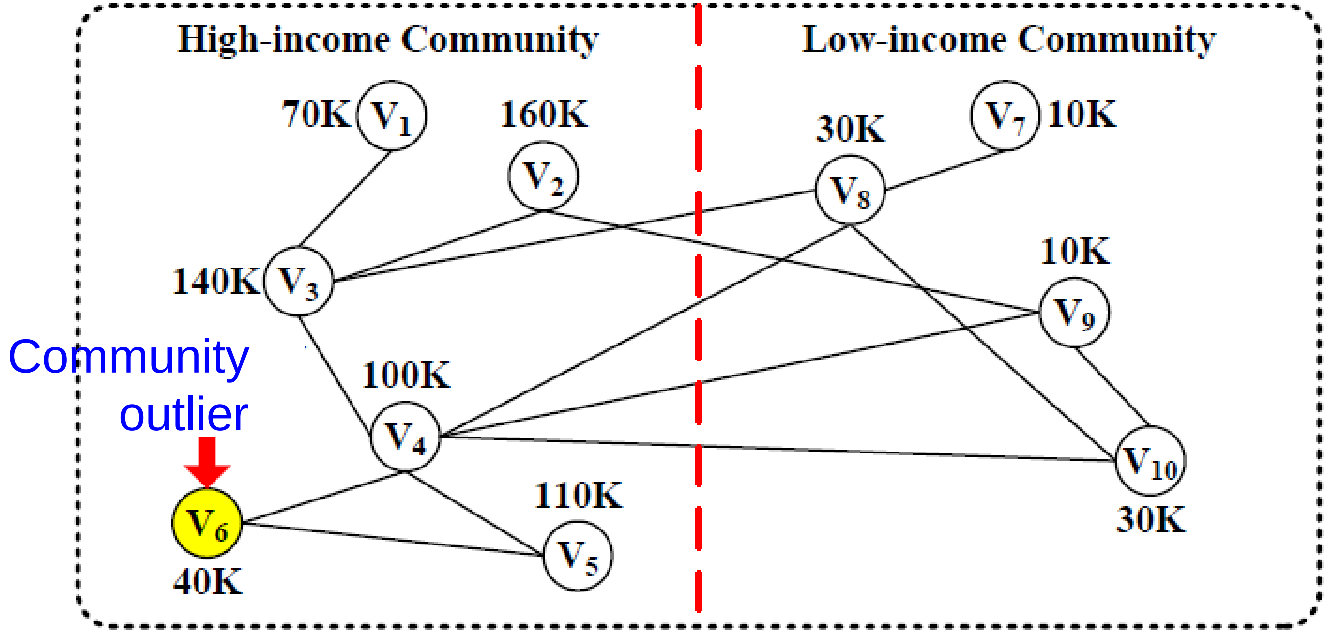
  - High (unreported) false positive rate!

Community Outliers

Gao et al. '10



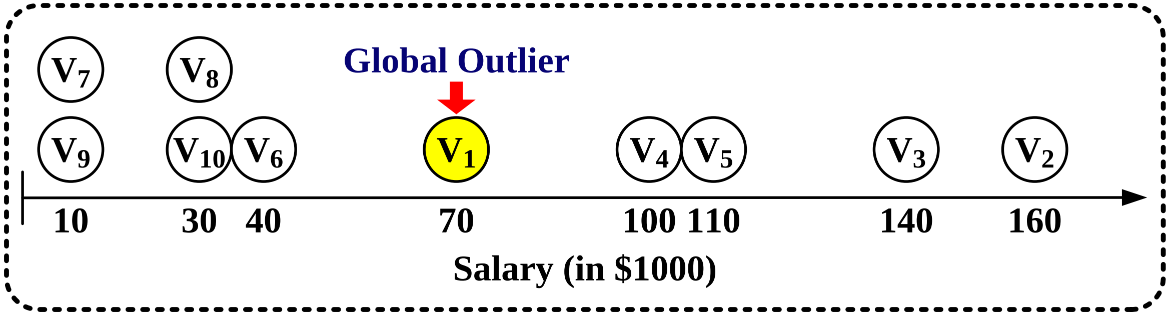
- Definition
  - Two information sources: links, node features
  - Communities based on **both** links and node features
  - Objects with features deviating from other community members defined as community outliers



Gao+KDD'10 modified with permission

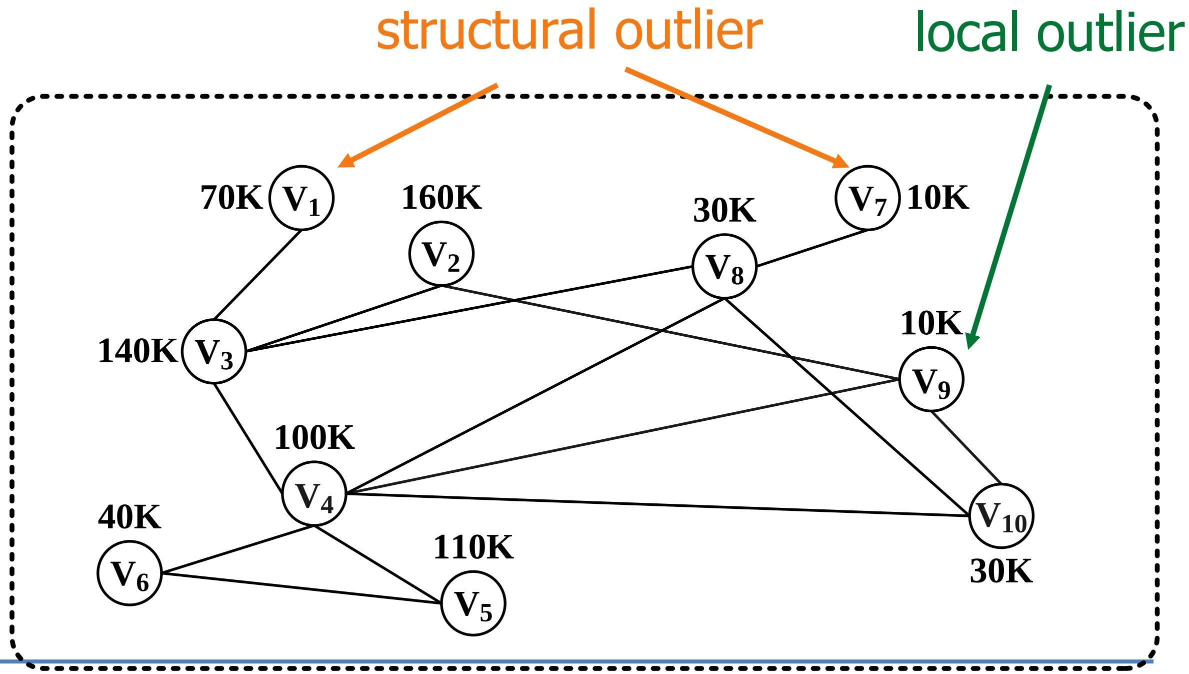
Other network outliers

1) Global outlier:
only considers node features



2) Structural outlier:
only consider links

3) Local outlier:
only consider the feature values of direct neighbors



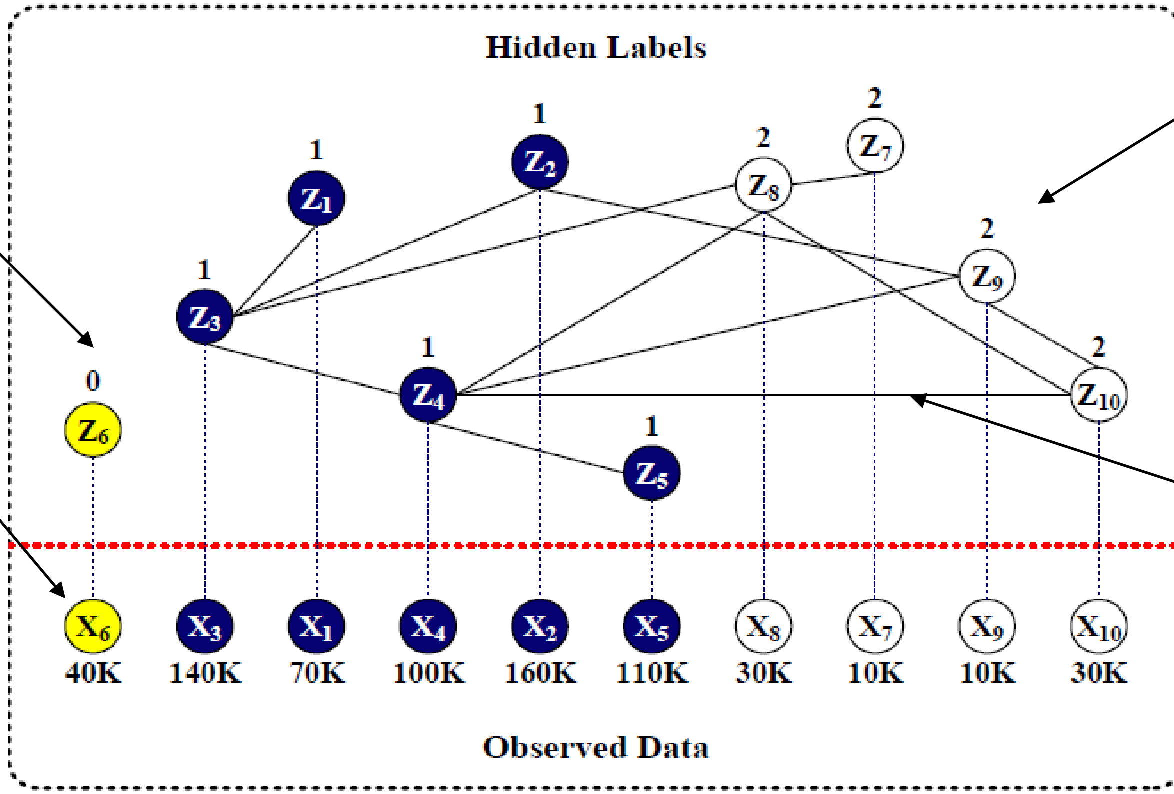
Gao+KDD'10
modified with permission

A unified probabilistic model

outlier

node features X

community label Z {0, 1, …, K}



link structure W

$$\Theta = (\theta_1, \ldots, \theta_K)$$

K: number of communities (user input)

model parameters X's are drawn from

Optimization formulation

details

- Maximize P(X) $\propto$ P(X|Z) P(Z)
  - P(X|Z) depends on community label and model param.s
    - e.g., salaries in the high or low-income communities follow Gaussian distributions defined by mean and std

$$P(x_i = s_i | z_i = k) = P(x_i = s_i | \theta_k)$$

Normal with $\{\mu_k, \sigma_k^2\}$

$$P(x_i = s_i | z_i = 0) = \rho_0$$

Uniform for outliers

  - P(Z) is higher if neighboring nodes from normal communities share the same community label
    - e.g., two linked nodes are likely to be in the same community
    - outliers are isolated—does not depend on the labels of neighbors

$$P(Z) \propto \sum_{w_{ij}>0, z_i \neq 0, z_j \neq 0} w_{ij} \delta(z_i - z_j)$$

Algorithm

Θ : model parameters

Z : community labels

Initialize Z

Fix Z, find Θ
that maximizes P(X|Z)

Parameter estimation

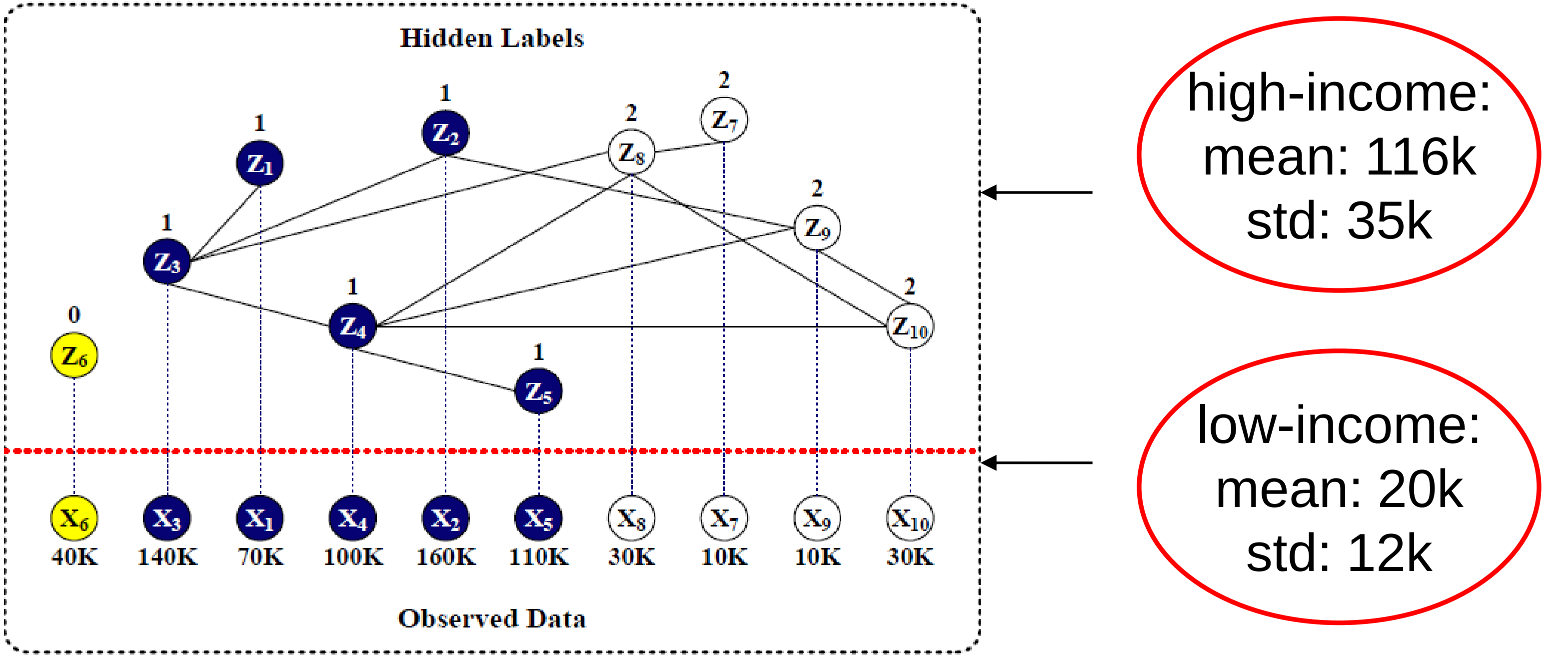
Fix Θ, find Z
that maximizes P(Z|X)

Inference

Gao+KDD'10
modified with permission

Algorithm: parameter estimation

- Calculate model parameters Θ
  - maximum likelihood estimation
- Continuous: $\{\mu_k, \sigma_k^2\}$
  - mean: sample mean of the community
  - std: square root of sample variance of community



Algorithm: Inference

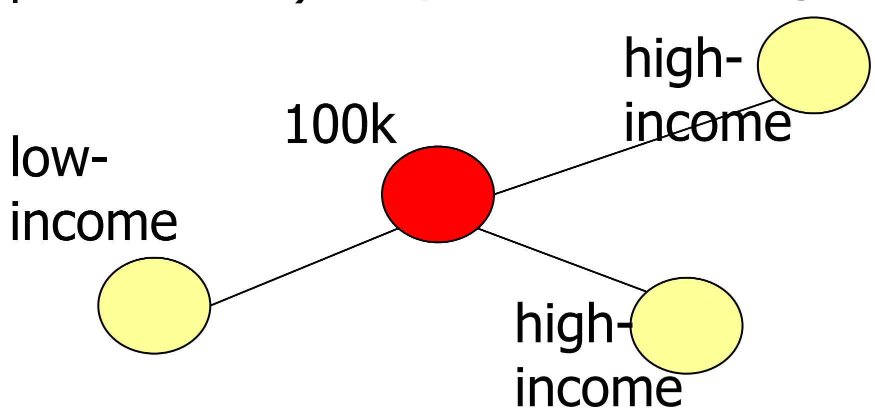
- Calculate label assignments **Z**
  - Model parameters are known
  - Iteratively update the community labels of nodes
  - For each node: select label that maximizes:

high-income:
mean: 116k
std: 35k

low-income:
mean: 20k
std: 12k

$$P(z_i | x_i = s_i, z_{I-\{i\}}) \propto P(x_i = s_i | z_i) \cdot \exp\left(\lambda \sum_{j \in N_i} w_{ij} \delta(z_i - z_j)\right)$$

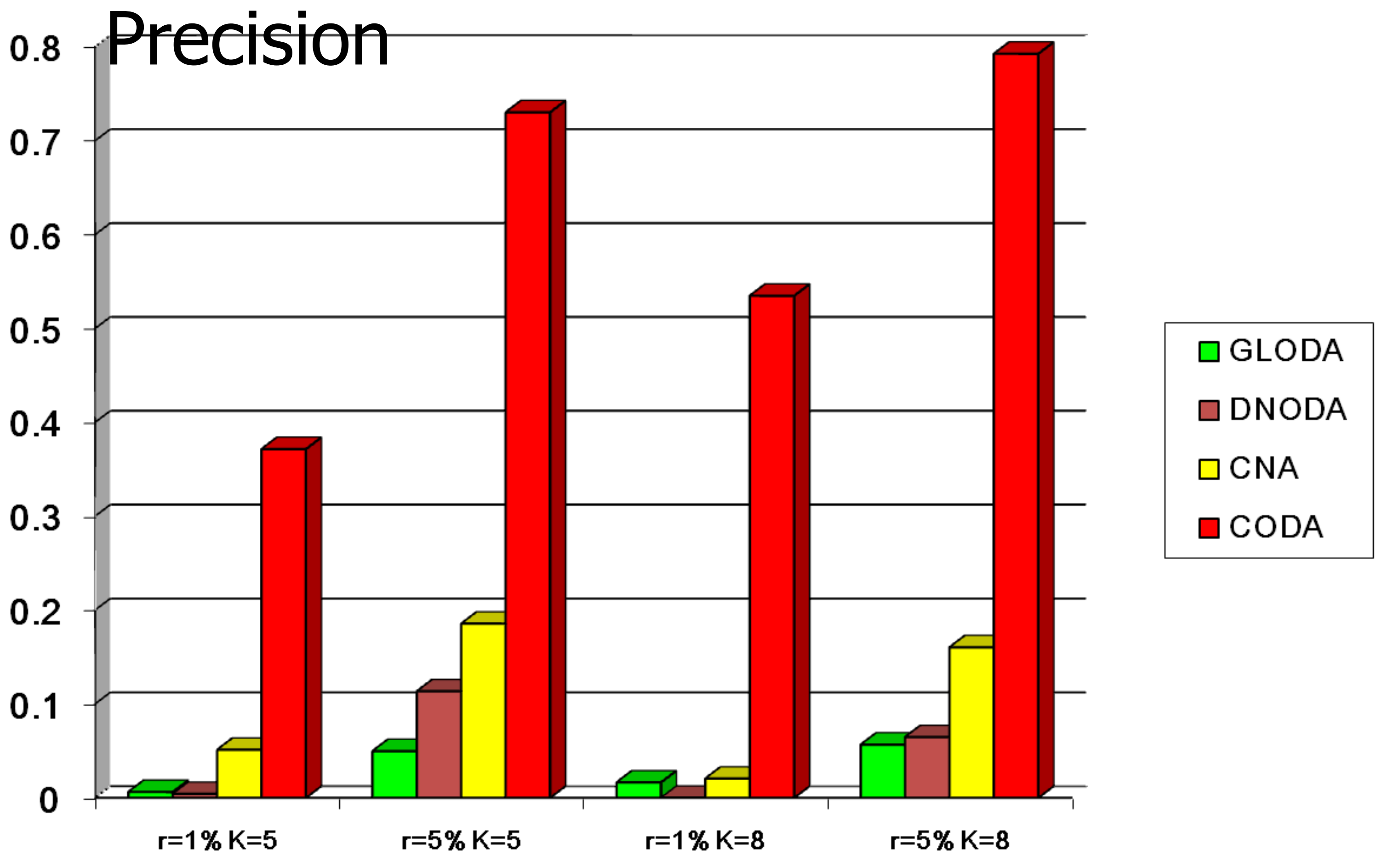
high-income:	P(salary=100k|high-income)	P(high-income|neighbors)
low-income:	P(salary=100k|low-income)	P(low-income|neighbors)
outlier:	constant	



Experiments: Simulations

- Data
  - Generate continuous data based on Gaussian distributions and generate labels according to the model
  - r: percentage of outliers, K: number of communities
- Baseline models
  - GLODA: global outlier detection (based on node features only)
  - DNODA: local outlier detection (check the feature values of direct neighbors)
  - CNA: partition data into communities based on links and then conduct outlier detection in each community

Experiments: Simulations



Case study on DBLP

- Conferences graph
  - Links: % common authors among two
  - Node features: publication titles in the conference
- Communities:
  - **Database:** ICDE, VLDB, SIGMOD, PODS, EDBT
  - **Artificial Intelligence:** IJCAI, AAAI, ICML, ECML
  - **Data Mining:** KDD, PAKDD, ICDM, PKDD, SDM
  - **Information Analysis:** SIGIR, WWW, ECIR, WSDM
- Community outliers: CVPR and CIKM

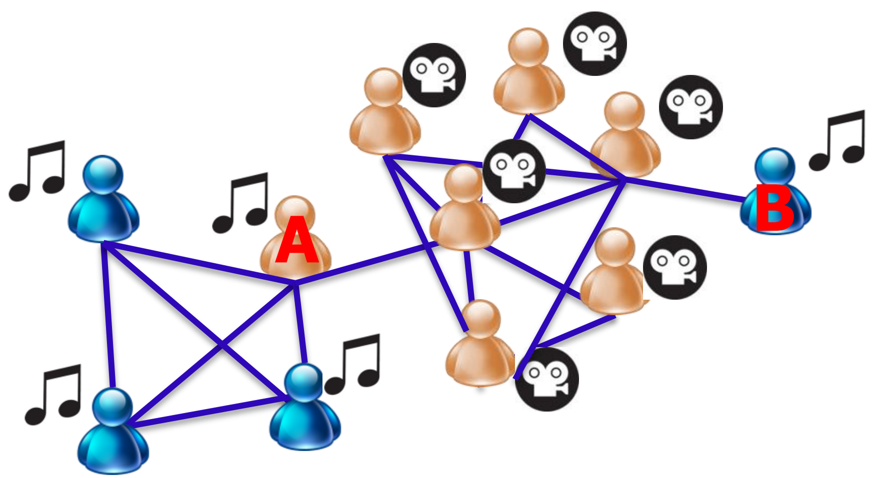
Cohesive groups in attributed graphs

- Problem:

Given a graph with node attributes (features)

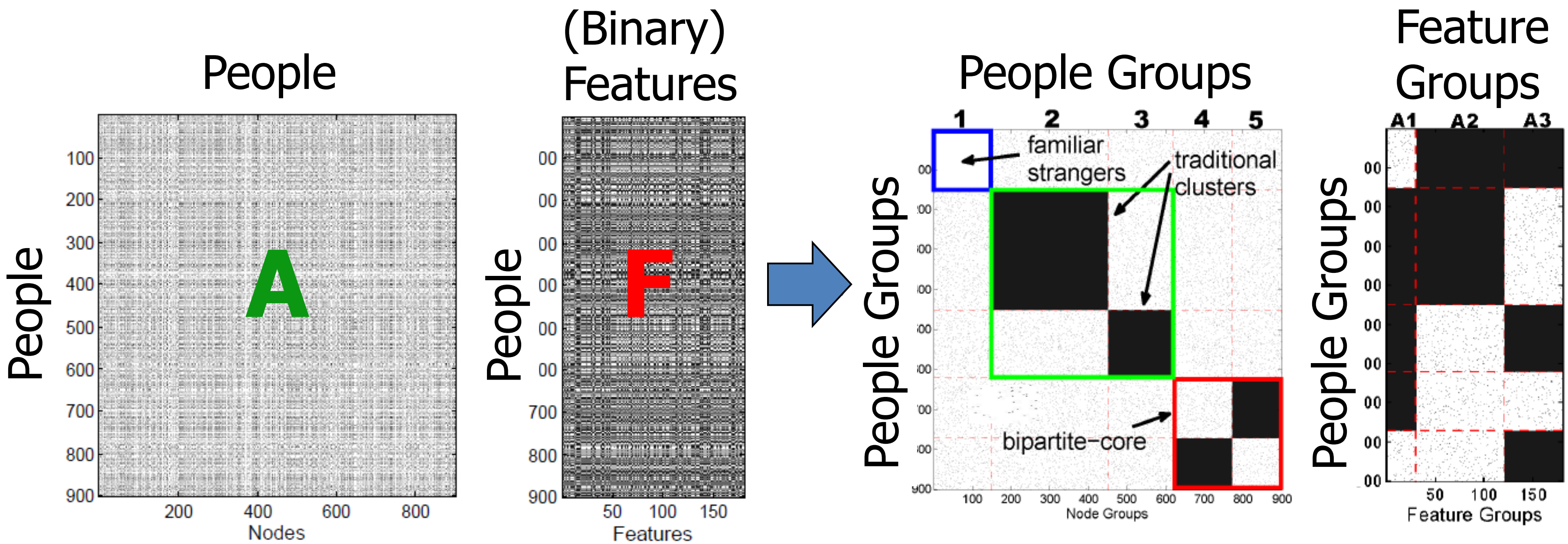
- social networks + user interests
- phone call networks + customer demographics
- gene interaction networks + gene expression info

Find cohesive clusters, bridges, anomalies



Note: cohesive cluster: similar connectivity & attributes

Problem sketch



Given adjacency matrix **A** and feature matrix **F**

Find homogeneous blocks (clusters) in **A** and **F**

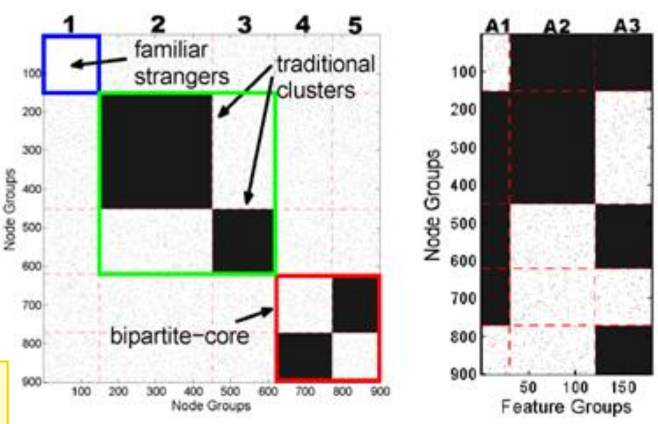
- * parameter-free
- * scalable

Problem formulation

1. How many node- & attribute-clusters?
2. How to assign nodes and attributes to clusters?

Main idea: employ Minimum Description Length

$$\underbrace{L\,(M)}_{\text{encoding length of clustering}} + \underbrace{L\,(D|M)}_{\text{encoding length of blocks}}$$



Good Clustering ⟺ **implies** ⟺ Good Compression

Problem formulation

details

- L (M) : Model description cost
  1. $\log^\star n + \log^\star f$ n: #nodes, f: #attributes
  2. $\log^\star k + \log^\star l$ k: #node-clusters, l: #attribute-clusters
  3. $nH(P) + fH(Q)$

$$p_i = \frac{r_i}{n}$$ size of node-cluster i

$$q_j = \frac{c_j}{f}$$ size of attribute-cluster j

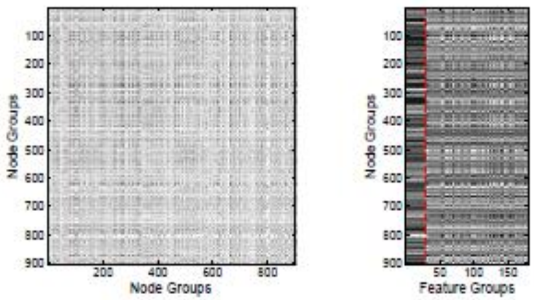
- L(D|M): Data description cost given Model
  1. For each block in A and F , #1s: $\log^\star n_1(B_{ij})$
  2. $E(B_{ij}) = -n_1(B_{ij}) \log_2(P_{ij}(1)) - n_0(B_{ij}) \log_2(P_{ij}(0))$

where $P_{ij}(1) = n_1(B_{ij})/n(B_{ij})$

$r_i c_j$ or $r_i r_j$

A similar problem (column re-ordering for minimum total run length) is shown to be NP-hard [Johnson+]. (reduction from Hamiltonian Path)

Algorithm sketch

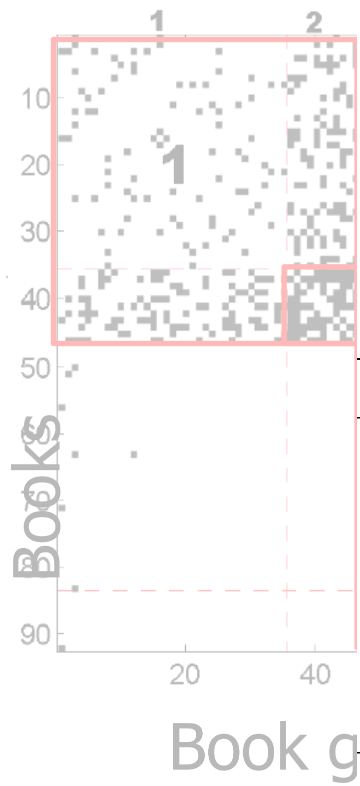


(a) k=1 l=2

`Split-FeatureGroup`

The algorithm is iterative and monotonic –will converge to local optimum

PICS at work (Political books)



“core and perip

Examples of “core” liberal and conservative books

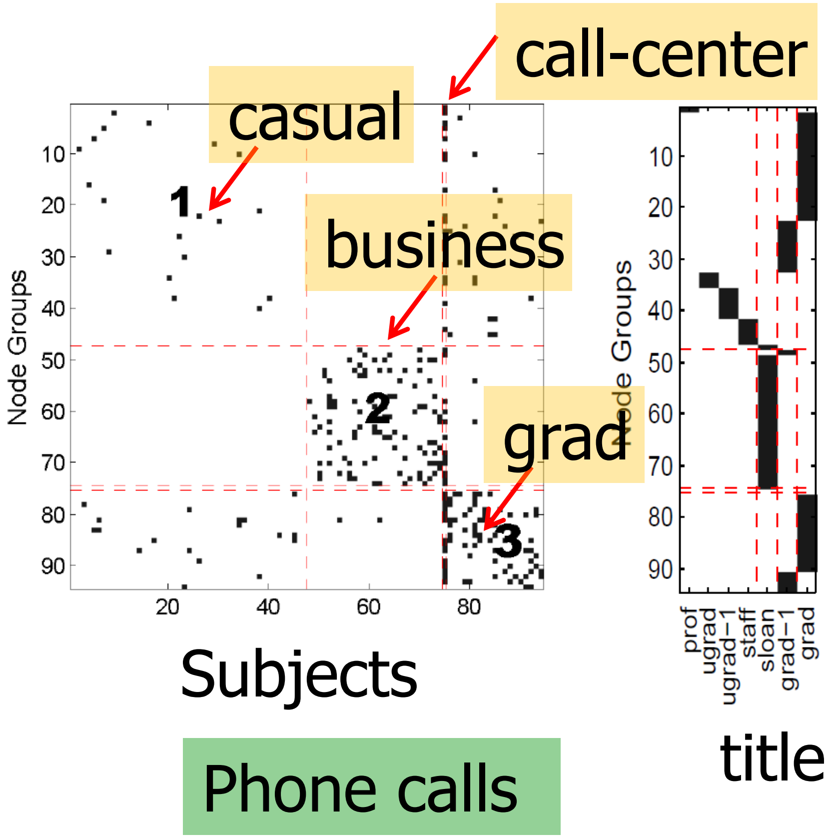
Liberal
–Lies and the Lying Liars Who Tell Them: *A Fair and Balanced Look at the Right*
–Big Lies: *The Right-Wing Propaganda Machine and How It Distorts the Truth*
–The Lies of George W. Bush
–Dude, Where's My Country?

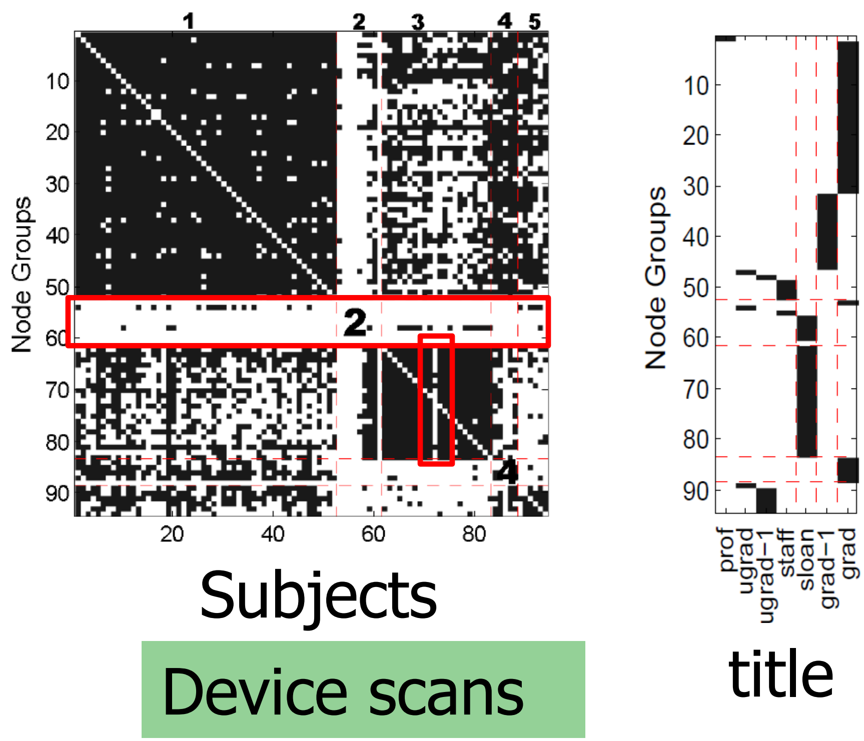
Conservative
–Persecution: *How Liberals Are Waging War Against Christianity*
–Deliver Us from Evil: *Defeating Terrorism, Despotism, and Liberalism*
–Tales from the Left Coast
–A National Party No More

Examples of bridging ‘conservative’ books

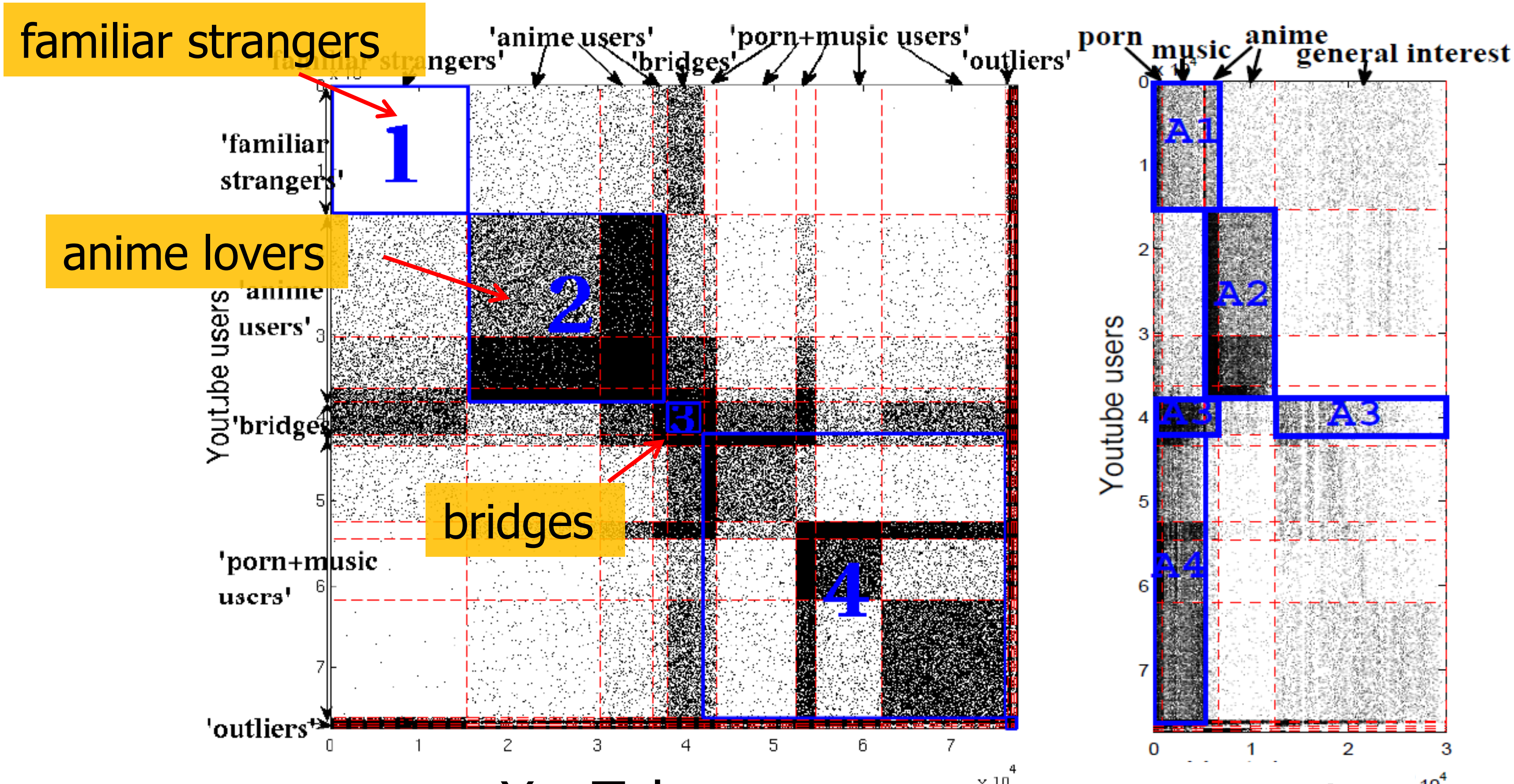
–*Bush at War*
–*The Bushes: Portrait of a Dynasty*
–*Rise of the Vulcans: The History of Bush's War Cabinet*

PICS at work (Reality mining)





PICS at work (YouTube)



YouTube users

YouTube groups

77K users
30K groups

Part I: References (attribute graphs)

Substructures

Community mining

- C. C. Noble and D. J. Cook. Graph-based anomaly detection. KDD, pages 631–636, 2003.
- W. Eberle and L. B. Holder. Discovering structural anomalies in graph-based data. ICDM Workshops, pages 393–398, 2007.
- Michael Davis, Weiru Liu, Paul Miller, George Redpath: Detecting anomalies in graphs with numeric labels. 1197-1202, CIKM 2011.
- Jing Gao, Feng Liang, Wei Fan, Chi Wang, Yizhou Sun, Jiawei Han: On community outliers and their efficient detection in information networks. KDD 2010: 813-822.
- Leman Akoglu, Hanghang Tong, Brendan Meeder, Christos Faloutsos. PICS: Parameter-free Identification of Cohesive Subgroups in large attributed graphs. SDM, 2012.

Tutorial Outline

- Motivation, applications, challenges
- **Part I:** Anomaly detection in **static** data
  - Overview: Outliers in **clouds of points**
  - Anomaly detection in **graph data**
- **Part II:** Event detection in **dynamic** data
  - Overview: Change detection in **time series**
  - Event detection in **graph sequences**
- **Part III:** Graph-based **algorithms and apps**
  - Algorithms: **relational learning**
  - Applications: **fraud and spam** detection

Coffee break…