Part II: Event detection in dynamic graphs

Part II: Outline

- Overview: Events in **point sequences**
  - **Change** detection in time series
  - Learning under **concept drift**

- Events in **graph sequences**
  - Change by graph **distance**
  - Change by graph **connectivity**

Event detection

- Anomaly detection in time series of multi-dimensional data points
  - Exponentially Weighted Moving Average
  - CUmulative SUM Statistics
  - Regression-based
  - Box-Jenkins models eg. ARMA, ARIMA
  - Wavelets
  - Hidden Markov Models
  - Model-based hypothesis testing
  - …
- This tutorial: time series of graphs

Part II: References (data series)

- Montgomery, D. C. Introduction to Statistical Quality Control. John Wiley and Sons, Inc., 2001.
- Box, George and Jenkins, Gwilym. Time series analysis: Forecasting and control, San Francisco: Holden-Day, 1970.
- Gama J., Medas P., Castillo G., Rodrigues P.P.: Learning with Drift Detection. SBIA 2004: 286-295.
- Grigg et al.; Farewell, VT; Spiegelhalter, DJ. The Use of Risk-Adjusted CUSUM and RSPRT Charts for Monitoring in Medical Contexts. Statistical Methods in Medical Research 12 (2): 147–170.
- Bay, S. D., and Pazzani, M. J., Detecting change in categorical data: Mining contrast sets. KDD, pages 302–306, 1999.
- M. Van Leeuwen, A. Siebes. StreamKrimp: Detecting Change in Data Streams. ECML PKDD, 2008.
- Wong, W.-K., Moore, A., Cooper, G. and Wagner, M. WSARE: An Algorithm for the Early Detection of Disease Outbreaks. JML, 2005.
- Tutorial: D. B. Neill and W.-K. Wong. A tutorial on event detection. KDD, 2009.

Part II: Outline

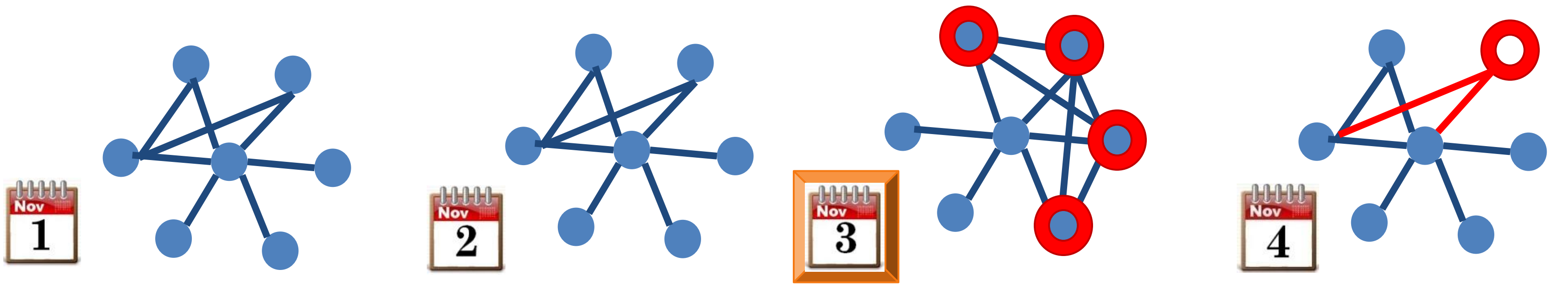
- Overview: Events in **point sequences**
  - **Change** detection in time series
  - Learning under **concept drift**

- Events in **graph sequences**
  - Change by graph **distance**
    - feature-based
    - structure-based
  - Change by graph **connectivity**

Events in time-evolving graphs

- Problem: **Given** a sequence of graphs,

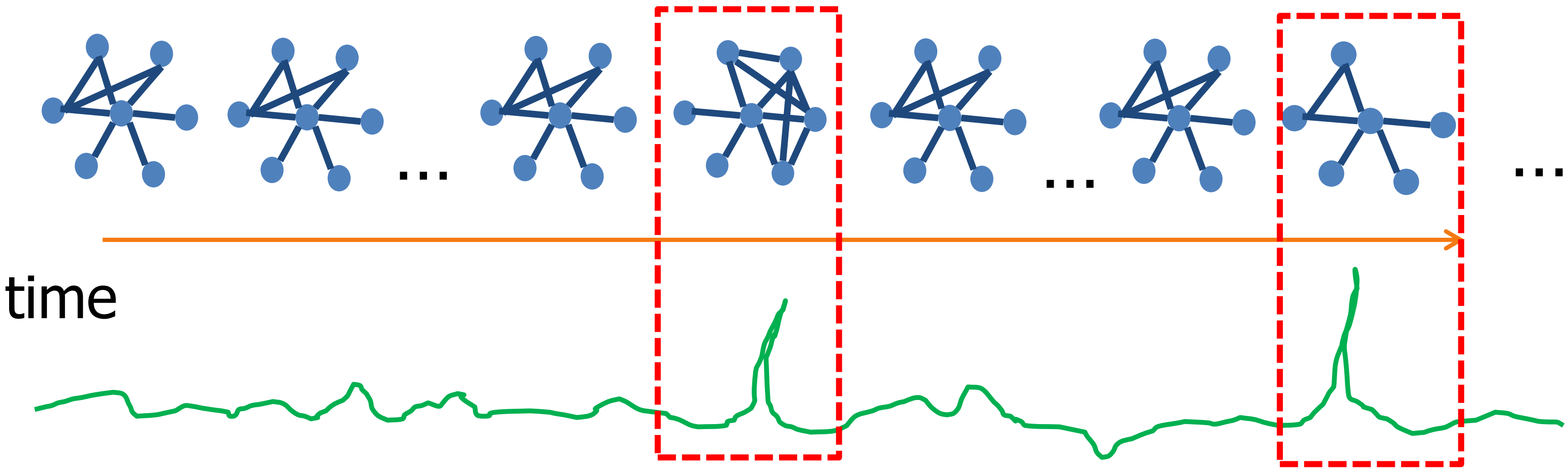
Q1. change detection: **find** time points at which graph changes significantly



Q2. attribution: **find** (top k) nodes / edges / regions that change the most

Events in time-evolving graphs

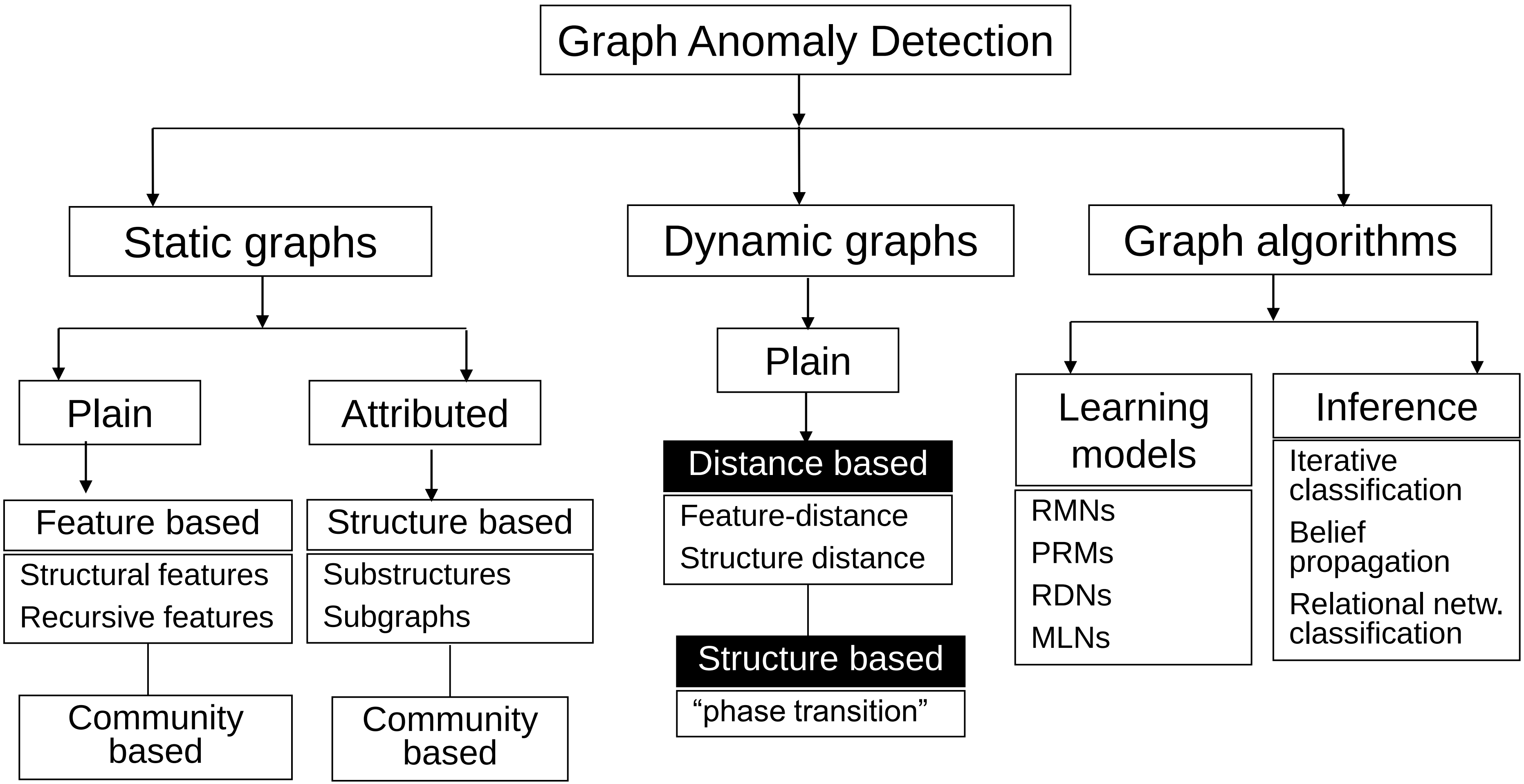
- Main framework
  - Compute graph similarity/distance scores



  - Find unusual occurrences in time series

- *Note: scalability is a desired property

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

Graph distance – 10 metrics

Shoubridge et al. '02
Dickinson et al. '04

- (1) Weight distance

$$d(G,H) = |E_G \cup E_H|^{-1} \sum_{u,v \in V} \frac{|w_E^G(u,v) - w_E^H(u,v)|}{\max\{w_E^G(u,v), w_E^H(u,v)\}}$$

- (2) Maximumum Common Subgraph (MCS) Weight distance

$$d(G,H) = |E_G \cap E_H|^{-1} \sum_{u,v \in V} \frac{|w_E^G(u,v) - w_E^H(u,v)|}{\max\{w_E^G(u,v), w_E^H(u,v)\}}$$

- (3) MCS Edge distance

$$d(G,H) = 1 - \frac{|\mathrm{mcs}(E_G, E_H)|}{\max\{|E_G|, |E_H|\}}$$

Graph distance – 10 metrics

- (4) MCS Node distance

$$d(G, H) = 1 - \frac{|\mathrm{mcs}(V_G, V_H)|}{\max\{|V_G|, |V_H|\}}$$

- (5) Graph Edit distance **Gao et al. '10 (survey)**

$$d(G, H) = |V_G| + |V_H| - 2|V_G \cap V_H| + |E_G| + |E_H| - 2|E_G \cap E_H|$$

  - Total cost of sequence of edit operations, to make two graphs isomorphic (costs may vary)
  - Unique labeling of nodes reduces computation
    - otherwise an NP-complete problem
  - Alternatives for **weighted** graphs

Graph distance – 10 metrics

- (5.5) Weighted Graph Edit distance

Kapsabelis et al. '07

$$d_2(G,H) = c\left[|V_G| + |V_H| - 2|V_G \cap V_H|\right] + \sum_{e \in E_G \cap E_H} |\beta_G(e) - \beta_H(e)| + \sum_{e \in E_G \setminus (E_G \cap E_H)} \beta_G(e) + \sum_{e \in E_H \setminus (E_G \cap E_H)} \beta_H(e)$$

edge weights

Non-linear cost functions

$\epsilon = 1$

$$d_3(G,H) = c\left[|V_G| + |V_H| - 2|V_G \cap V_H|\right] + \sum_{e \in E_G \cap E_H} \frac{|(\beta_G(e)+\epsilon) - (\beta_H(e)+\epsilon)|^2}{(\min(\beta_G(e), \beta_H(e))+\epsilon)^2} + \sum_{e \in E_G \setminus (E_G \cap E_H)} (\beta_G(e)+\epsilon)^2 + \sum_{e \in E_H \setminus (E_G \cap E_H)} (\beta_H(e)+\epsilon)^2$$

Graph distance – 10 metrics

- (6) Median Graph distance **Dickinson et al. '04**
  - Median graph of sequence $(G_{n-L+1}, \ldots, G_n)$

$$\tilde{G}_n = \arg\min_{G \in S} \sum_{i=n-L+1}^{n} d(G, G_i)$$

  - $d(\tilde{G}_n, G_{n+1})$ for each graph G_{n+1} in sequence
  - free to choose any distance function d

- (7) Modality distance **Kraetzl et al. '06**

$$d(G,H) = \|\pi(G) - \pi(H)\|$$

$$A\pi = \rho\pi, \quad \pi > 0$$

Perron vector

Graph distance – 10 metrics

- (8) Diameter distance

Gaston et al. '06

$$d(G,H) = \left| \sum_{v \in V_H} \mathrm{maxd}(H,v) - \sum_{v \in V_G} \mathrm{maxd}(G,v) \right|$$

shortest distance

- (9) Entropy distance

$$d(G,H) = -\sum_{e \in E_H} \left(\tilde{w}_e^H - \ln \tilde{w}_e^H\right) + \sum_{e \in E_G} \left(\tilde{w}_e^G - \ln \tilde{w}_e^G\right)$$

$$\tilde{w}_e^* = w_e^* / \sum_{e \in E_*} w_e^*$$

- (10) Spectral distance

$$d(G,H) = \sqrt{\frac{\sum_{i=1}^{k} (\lambda_i - \mu_i)^2}{\min\left\{\sum_{i=1}^{k} \lambda_i^2, \sum_{i=1}^{k} \mu_i^2\right\}}}$$

Largest pos. eigenvalues of Laplacian

Graph distance – 10 metrics

Metric	Vertices used?	Edges used?	Vertex weights used?	Edge weights used?	Range	Value if graphs identical
Weight	No	Yes	No	Yes	[0,1]	0
MCS Weight	No	Yes	No	Yes	[0,1]	0
MCS Edge	No	Yes	No	No	[0,1]	0
MCS Vertex	Yes	No	No	No	[0,1]	0
Graph Edit	Yes	Yes	No	No	$[0,\infty)$	0
Median Edit	Yes	Yes	No	No	$[0,\infty)$	0
Modality	No	Yes	No	Yes	[0,1]	0
Diameter	Yes	Yes	No	No	$[0,\infty)$	0
Entropy	No	Yes	No	Yes	(-1,1)	0
Spectral	No	Yes	No	Yes	[0,1]	0

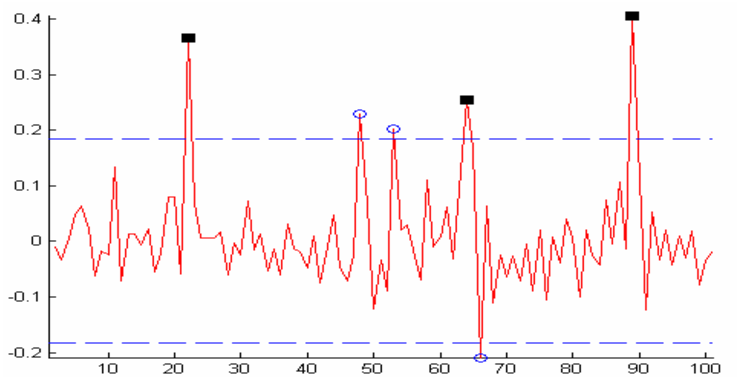
Graph distance to time series

- Each graph as a feature vector
  - graph distance metrics
- Time series of graph distances per feature
- $\mathrm{ARMA}(p, q)$ model for each time series

$$X_t = \alpha_1 X_{t-1} + \cdots + \alpha_p X_{t-p} + \epsilon_t + \beta_1 \epsilon_{t-1} + \cdots + \beta_q \epsilon_{t-q}$$

  - assumes stationary series, due to construction
- Anomalous time points: where residuals exceed a threshold

residuals



day

Graph distance to time series

- Minimum mean squared error $S = X_1, X_2, \ldots, X_M$

L R m

$$\mathrm{MSE}(m) = \sum_{i=1}^{m} (X_i - \bar{X}_L)^2 + \sum_{i=m+1}^{M} (X_i - \bar{X}_R)^2$$

  - change point: m with minimum MSE(m)
  - randomized bootsrapping for confidence

- CUmulative SUMmation

series mean

$$C = (s_0, s_1, \ldots, s_M) \qquad \begin{aligned} s_0 &= 0 \\ s_k &= s_{k-1} + X_k - \bar{X} \end{aligned}$$

  - bootstrap $\triangle C = \max_{i=1,\ldots,M} C - \min_{i=1,\ldots,M} C$

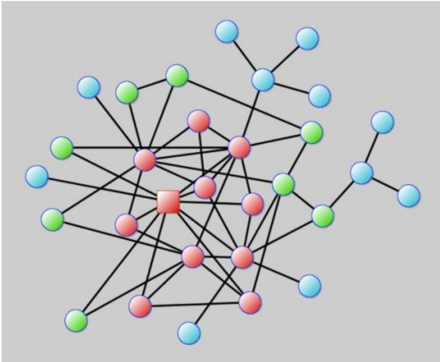
Note: single feature to represent whole graph

Priebe et al. '05

Scan statistics on graphs

- For each "scan region"
  - compute **locality statistic**
- (11) Scan statistic = max of locality statistics

Scan statistics framework



- For graph data
  - k-th order neighborhood
  - scan region: induced k-th order subgraph
  - locality stat.: e.g., #edges, density, domination #, …
  - scale (k)-specific scan stat. $M_k(D) = \max_{v \in V(D)} \Psi_k(v)$

Scan statistics on graphs

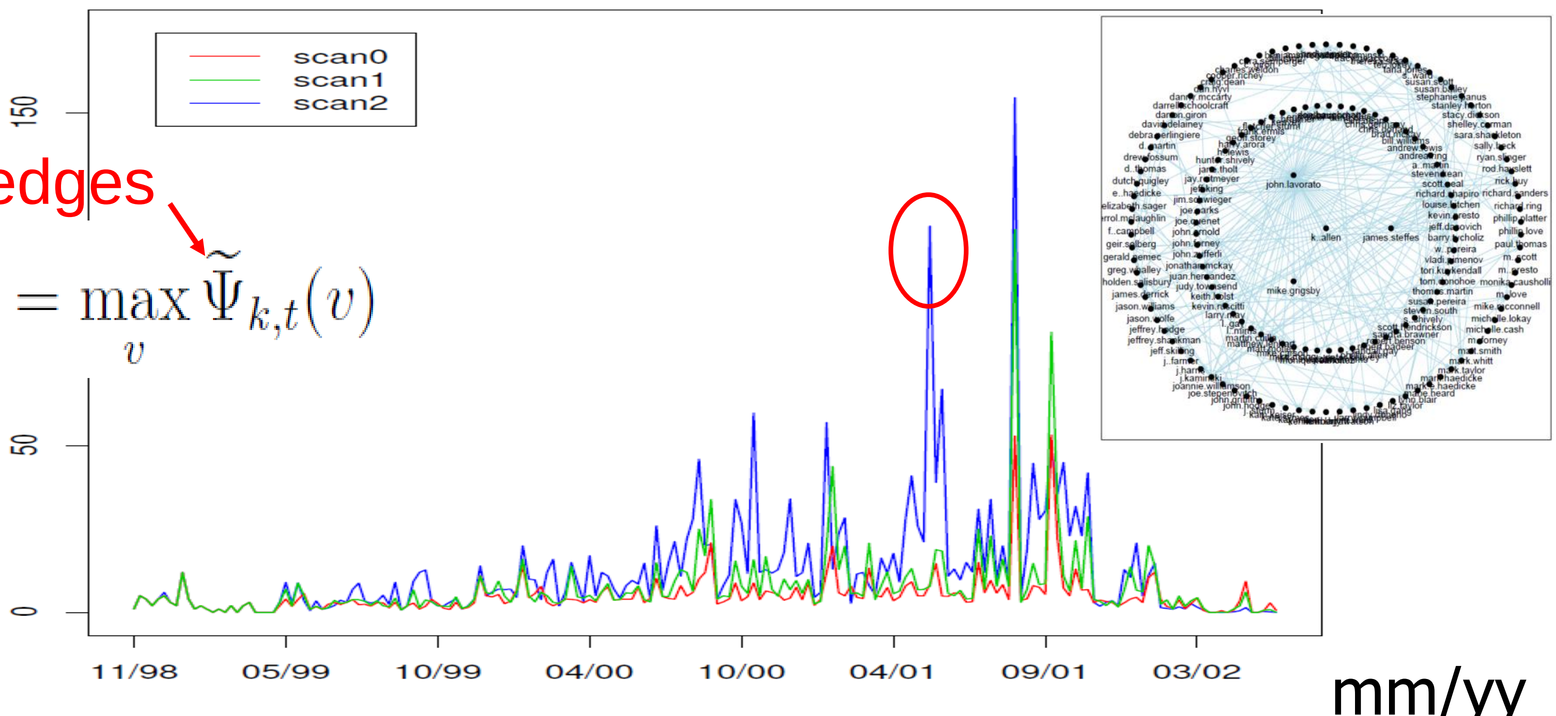
- Vertex-dependent normalized locality statistic

$$\widetilde{\Psi}_{k,t}(v) = (\Psi_{k,t}(v) - \widehat{\mu}_{k,t,\tau}(v)) \,/\, \max(\widehat{\sigma}_{k,t,\tau}(v), 1)$$

mean and std in (t–tau) window

#edges

$$\widetilde{M}_{k,t} = \max_v \widetilde{\Psi}_{k,t}(v)$$



Graph similarity

- *Note: sensitivity is a desired property
  - e.g. "high/low-quality" pages in Web graph
  - quality/importance: e.g., pagerank

- (12) Vertex ranking

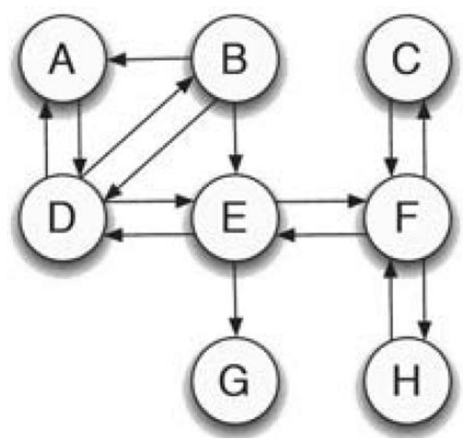
$$sim_{\mathrm{VR}}(G, G') = 1 - \frac{2\sum_{v \in V \cup V'} \underbrace{w_v}_{\text{quality(v)}} \times (\underbrace{\pi_v}_{\text{rank(v)}} - \pi'_v)^2}{D}$$

  - Rank: v in both G and G' → average
    v in only V → $\pi_v' = |V'|+1$
    v in only V' → $\pi_v = |V|+1$

Graph similarity

- (13) Sequence similarity
  - Depth-first-like sequencing with high-quality first

    Repeat
    - pick unvisited node with highest quality
    - visit highest quality unvisited neighbor, if any



A	0.56
B	0.43
C	0.51
D	1.01
E	0.93
F	1.29
G	0.41
H	0.51

  - Apply shingling
    - all k-length subsequences, i.e. shingles S(T)
    - $\text{sim}_{SS}(G,G') = \dfrac{S(T) \cap S(T')}{S(T) \cup S(T')}$

T= <F, E, D, A, C, H, B, G>

Graph similarity

- (14) Vector similarity
  - Compare **weighted** edge vectors
  - relative importance of an edge:

$$\gamma(u,v) = \frac{q_u \times \#outlinks(u,v)}{\sum_{\{v':(u,v')\in E\}} \#outlinks(u,v')}$$

  - Similarity over union of edges in G and G'

$$sim_{\mathrm{VS}}(G,G') = 1 - \frac{\sum_{(u,v)\in E\cup E'} \frac{|\gamma(u,v)-\gamma'(u,v)|}{max(\gamma(u,v),\gamma'(u,v))}}{|E\cup E'|}$$

note: for edges not in G' $\gamma'(u,v)=0$, and vice versa

Graph similarity

- (15) Signature similarity
  - Transfer graph G to a set L of weighted features

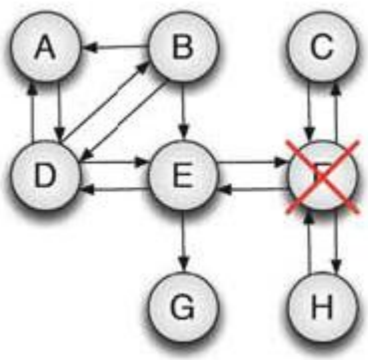
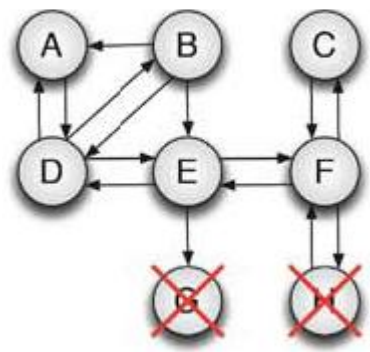
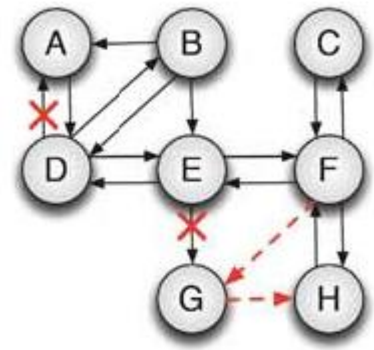
$$L = \{(t_i, w_i)\}$$

nodes/edges quality

e.g. $L(G) = \{(C, 0.51), (CF, 0.51), (F, 1.29), (FC, 1.29 \times 0.5), (FH, 1.29 \times 0.5), (H, 0.51), (HF, 0.51)\}.$

  - Construct b-bit signature for G
    - For each t_i
      - randomly choose b entries from $\{-w_i, +w_i\}$
    - Sum all b-dimensional vectors into h
    - Set '+' entries to 1 and '-'s to 0
  - $sim(L, L') = 1 - \frac{Hamming(h, h')}{b}$

Graph similarity

	Missing connected	Missing random	Connectivity change
			
Vertex ranking	**very good**	bad	bad
Sequence similarity	good	bad	**very good**
Vector similarity	**very good**	good	good
Signature similarity	**very good**	**very good**	**very good**

Part II: Outline

- Overview: Events in **point sequences**
  - **Change** detection in time series
  - Learning under **concept drift**

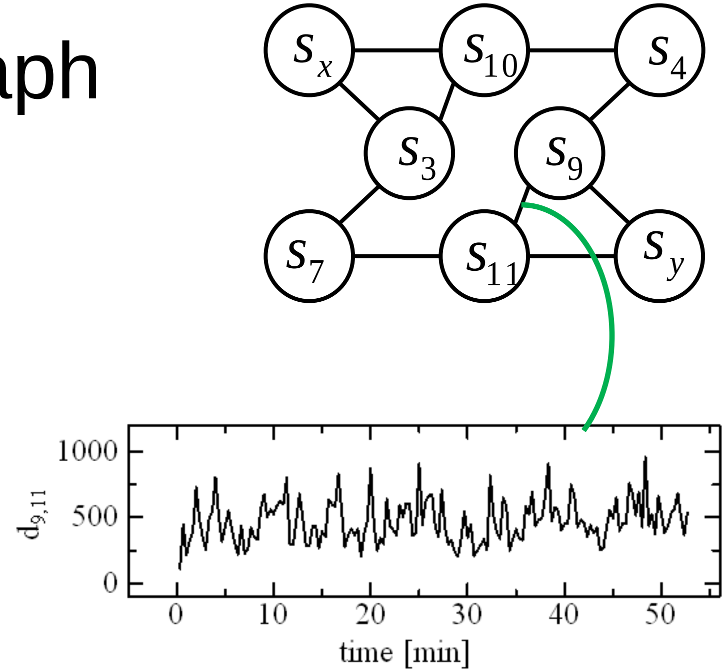
- Events in **graph sequences**
  - Change by graph **distance**
    - feature-based
    - structure-based
  - Change by graph **connectivity**
    - phase transition

Ide et al. '04

Eigen-space-based events

- **Given** a time-evolving graph

  Identify faulty vertices



- Challenges
  - Large number of nodes,
    impractical to monitor each
  - Edge weights are highly dynamic
  - Anomaly defined collectively (different than "others")

Event: a **"phase transition"** of the graph (in overall relation between the edge weights)

"Summary feature" extraction

- Definition of "activity" vector

$$\underbrace{\boldsymbol{u}(t)}_{\text{activity vector at } t} \equiv \arg\max_{\tilde{\boldsymbol{u}}} \left\{ \tilde{\boldsymbol{u}}^T \underbrace{\mathrm{D}(t)}_{\text{adjacency matrix at } t} \tilde{\boldsymbol{u}} \right\} \qquad \text{subject to } \tilde{\boldsymbol{u}}^T \tilde{\boldsymbol{u}} = 1$$

activity vector at *t*

adjacency matrix at *t*
(symmetric, non-negative)

- The above equation can be reduced to

$$\mathrm{D}(t)\tilde{\boldsymbol{u}} = \lambda \tilde{\boldsymbol{u}}. \qquad \text{subject to } \tilde{\boldsymbol{u}}^T \tilde{\boldsymbol{u}} = 1$$

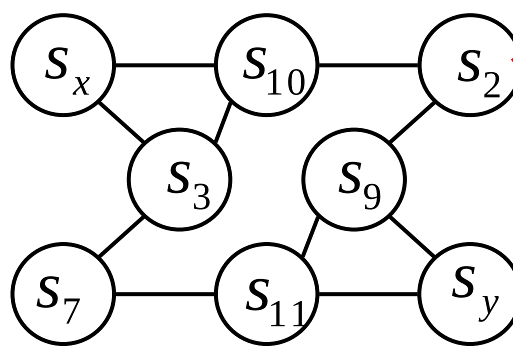
→ The principal eigenvector gives the summary of node "activity"

Activity feature

$$\boldsymbol{u}(t) \equiv \arg\max_{\tilde{\boldsymbol{u}}} \left\{ \tilde{\boldsymbol{u}}^T \mathrm{D}(t) \tilde{\boldsymbol{u}} \right\}$$

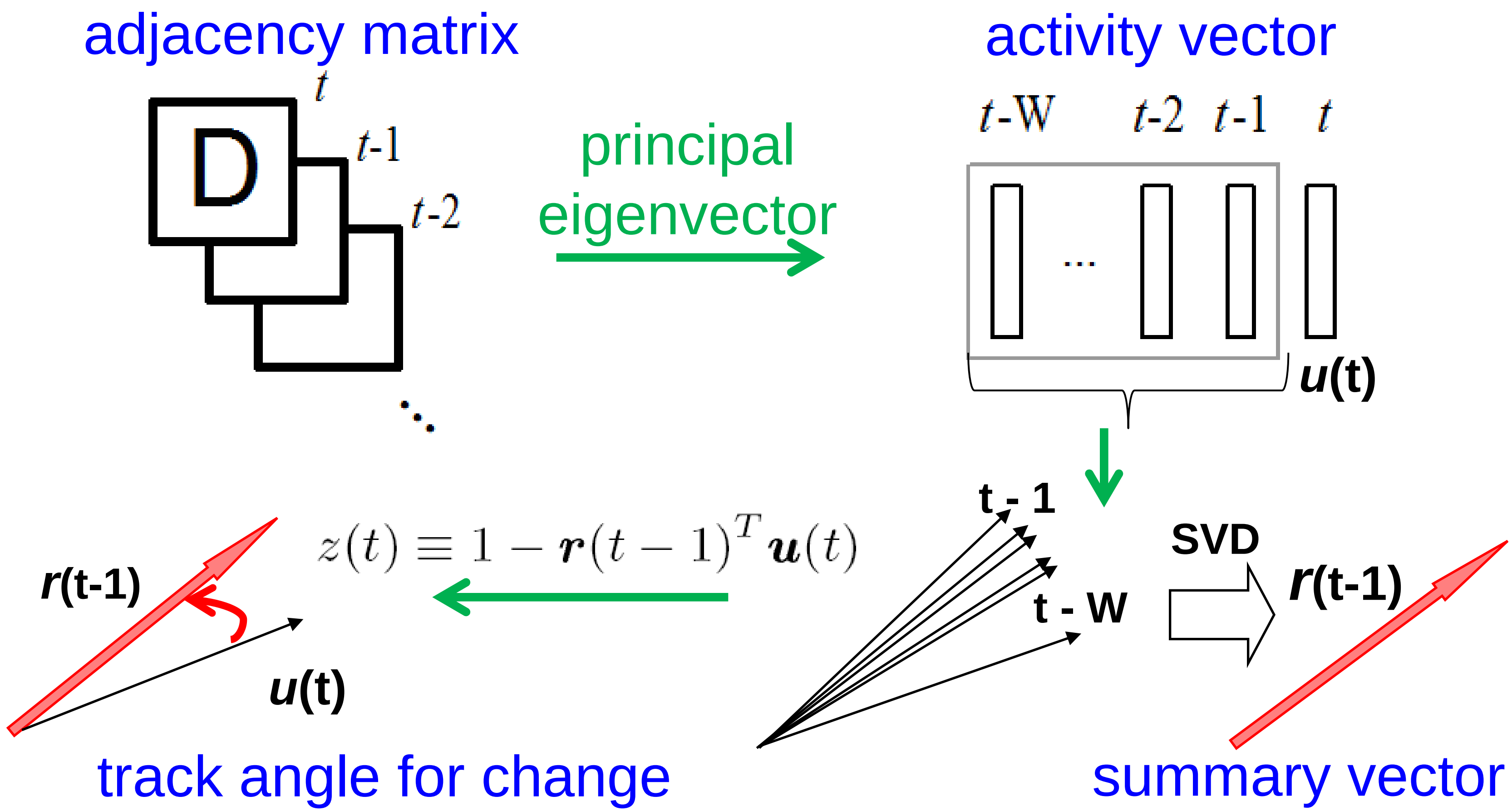
- Why “activity”? (intuition)
  - If D_{12} is large, then u_1 and u_2 should be large because of argmax (note: D is a positive matrix).
  - So, if s_1 actively links to other nodes at *t*, then the “activity” of s_1 should be large.
  - Also interpreted as **“stationary state”**: probability that a node is holding the “control token”

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}$$



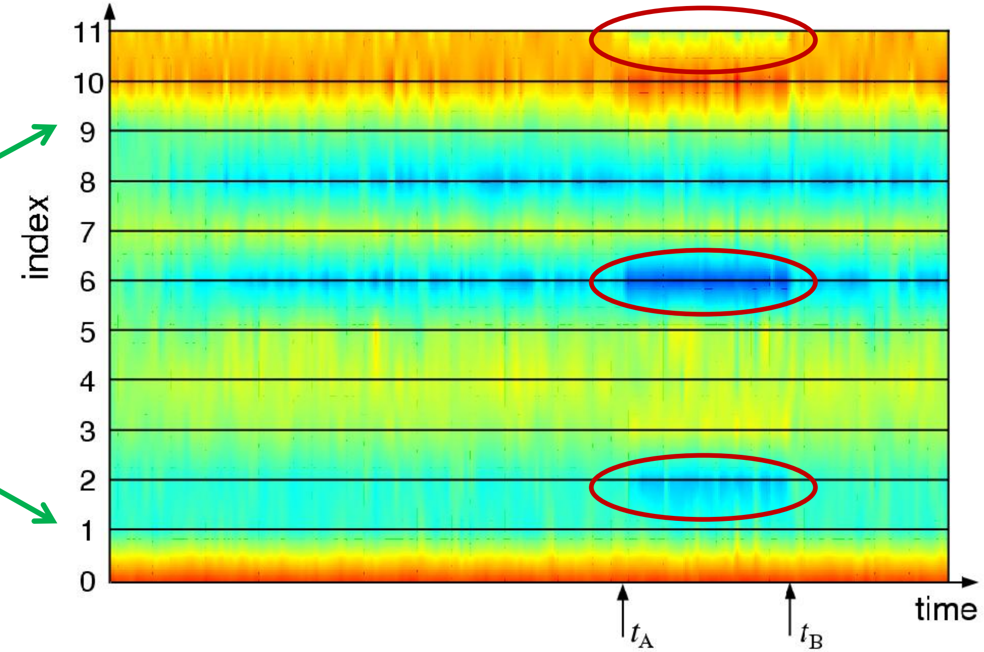
Anomaly detection

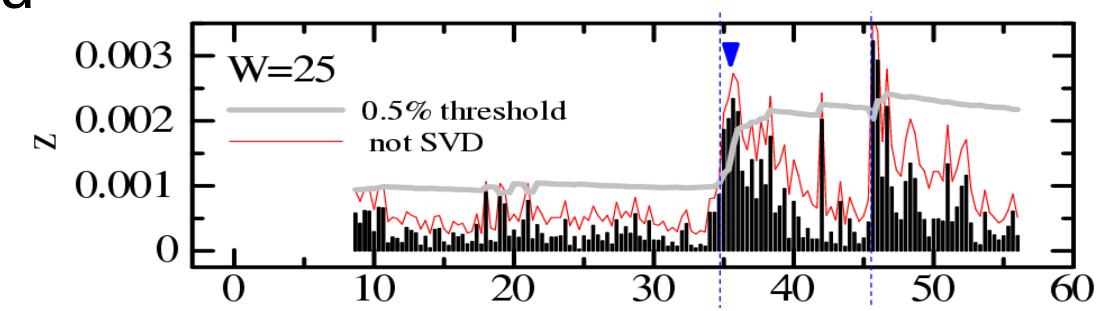
- Problem reduced from a sequence of graphs to a sequence of (activity) vectors

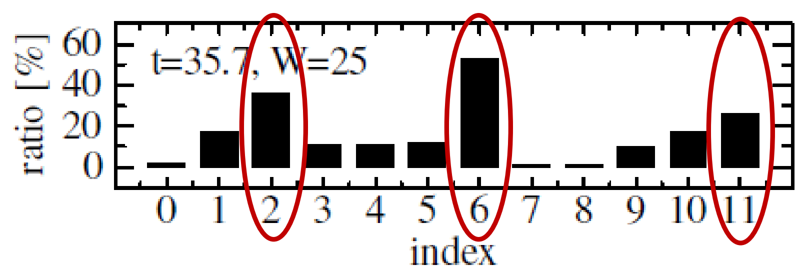


Experiment

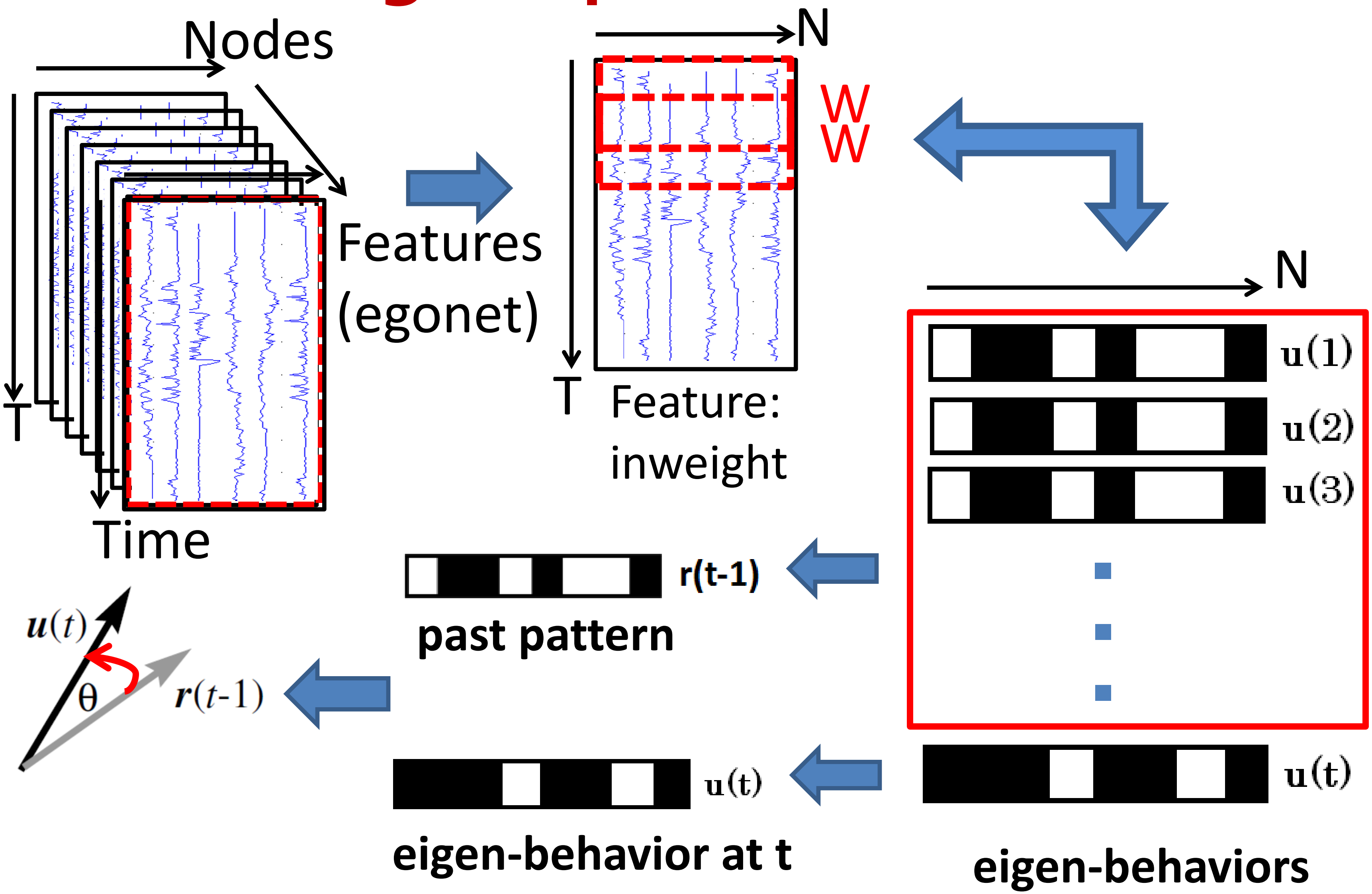
- Time evolution of activity scores effectively visualizes malfunction
- Anomaly measure and **online thresholding** dynamically capture activity change
- Nodes changing most can be attributed





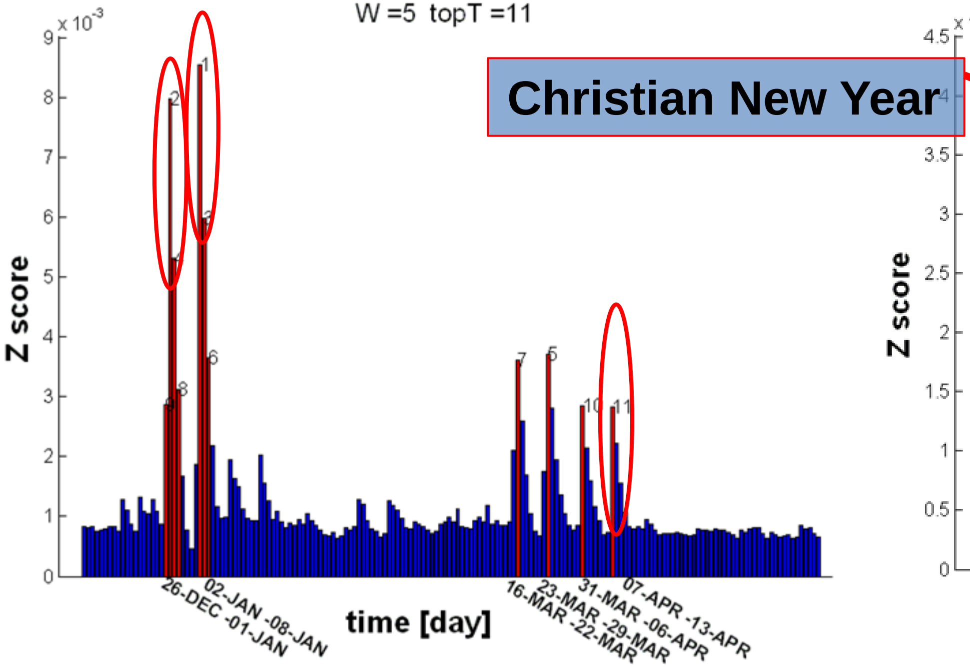


Feature/Eigen-space-based events

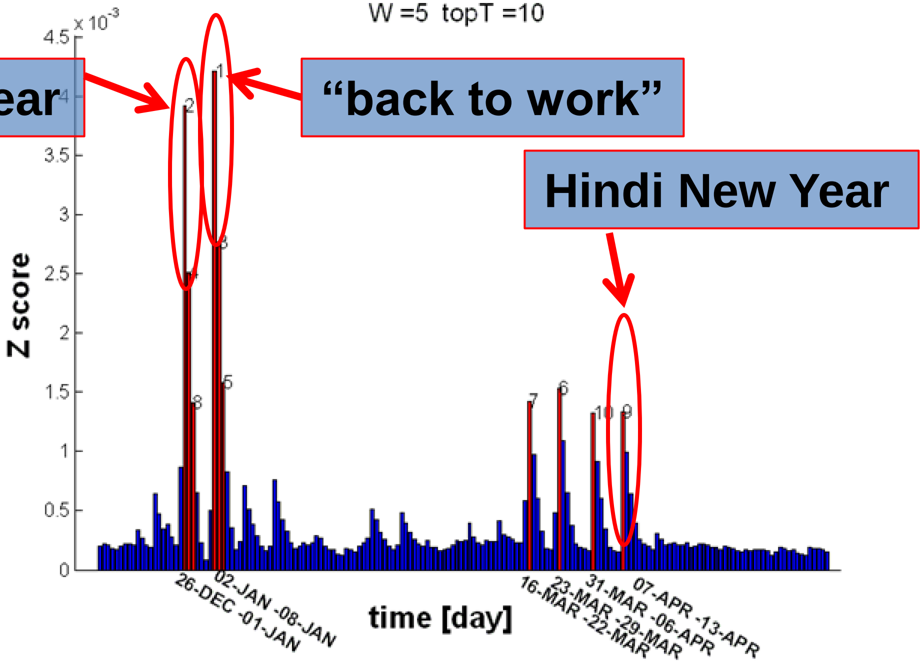


Change point detection

F: out-degree

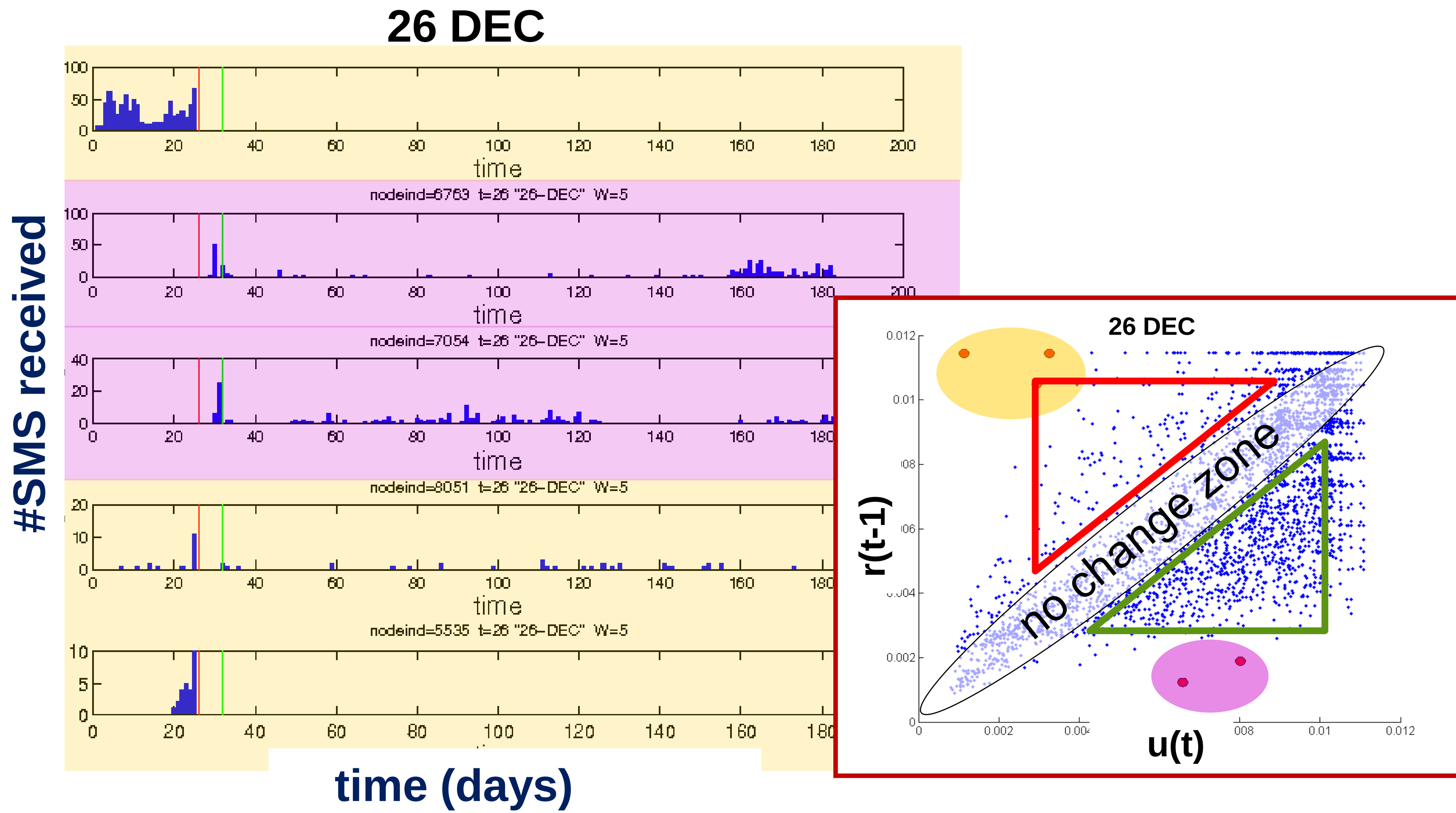


F: reciprocal degree



Event score Z over time

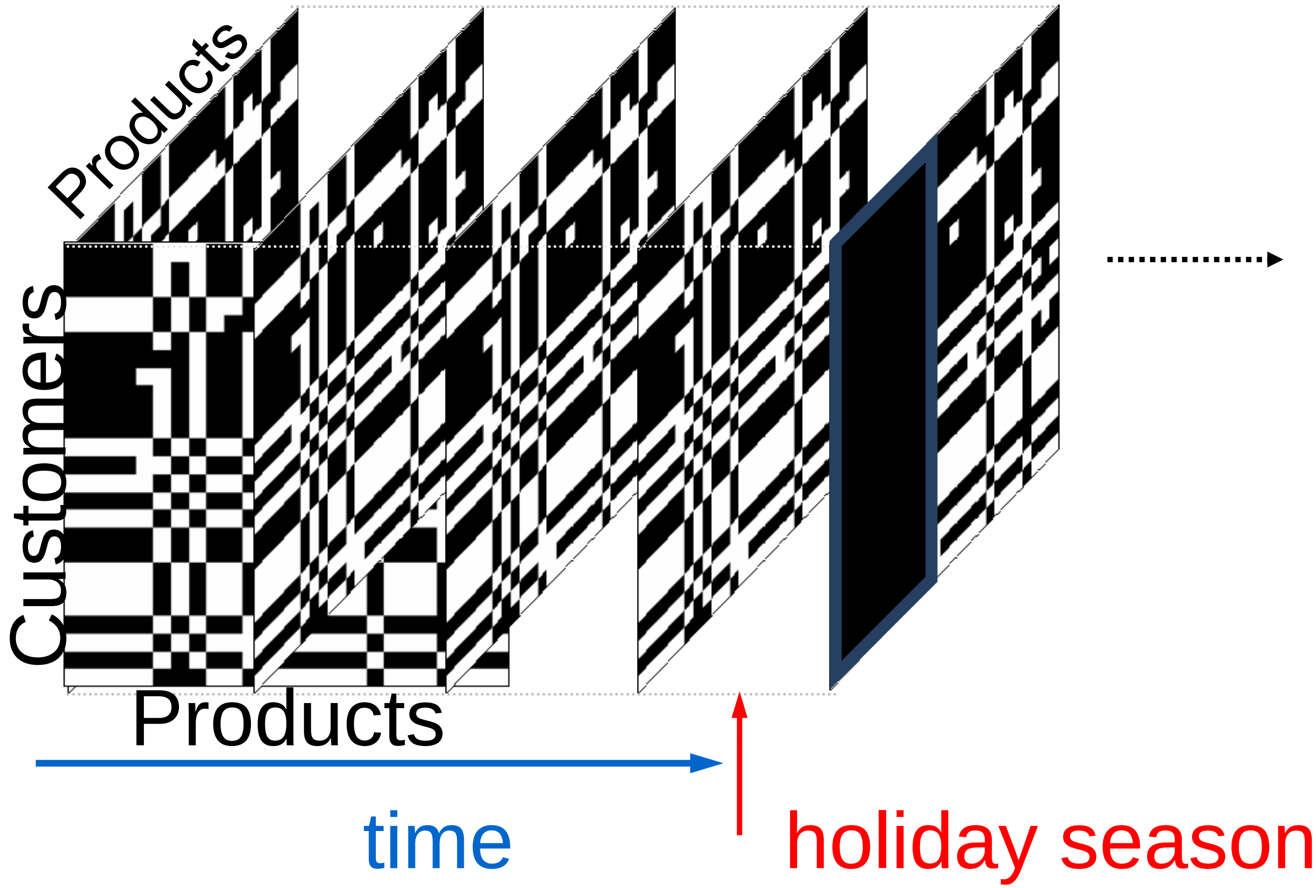
Change attribution



Time series of top 5 nodes with highest ratio index

Community-based events

- Main idea: monitor community structure and alert event when it changes



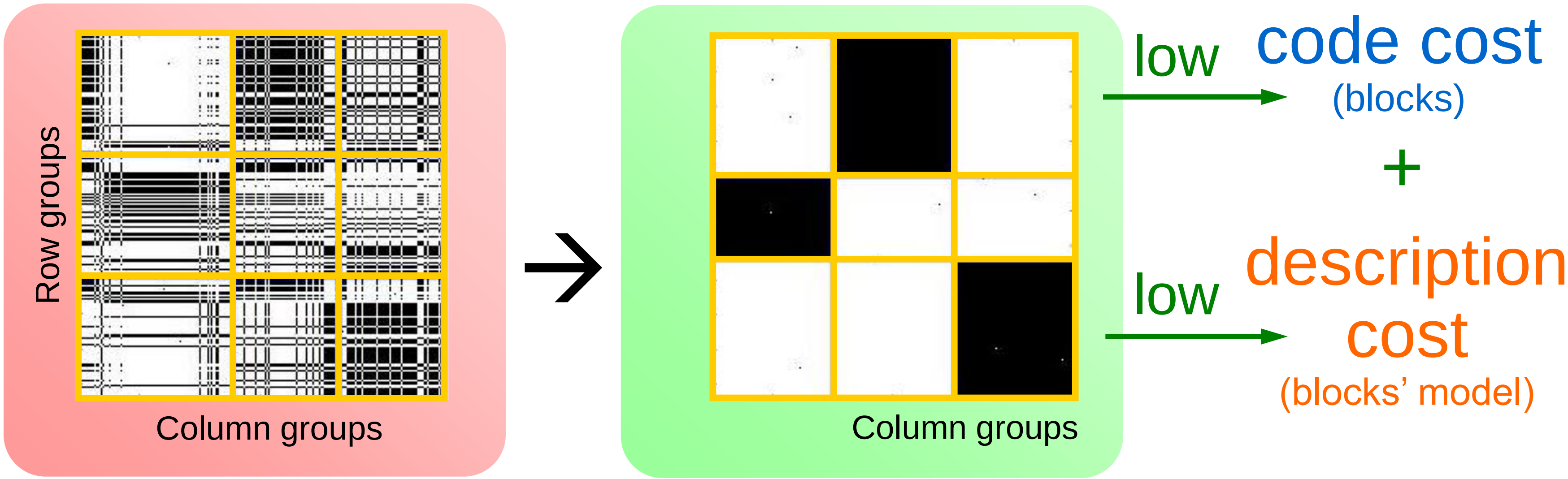
Community-based events

- Many graph clustering/partitioning algorithms
  - METIS **Karypis et al. '95**
  - Spectral Clustering **Shi & Malik '00** **Ng et al.'02**
  - Girvan-Newma **'03**
  - Co-clustering **Dhillon et al. '03** **Chakrabarti '04**
  - …
- Challenge
  - distance measure between clusterings

Chakrabarti '04

Community detection

- Clustering problem as compression problem



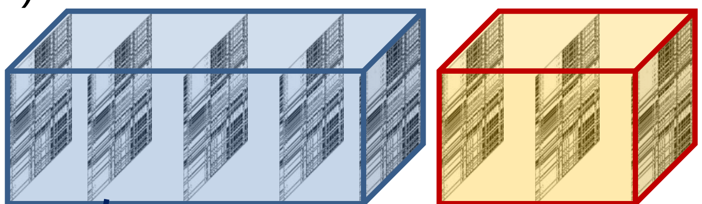
Good Clustering ⇔ 1. Similar nodes are grouped together 2. As few groups as necessary ⇔ A *few*, *homogeneous* blocks ⇔ Good Compression

implies

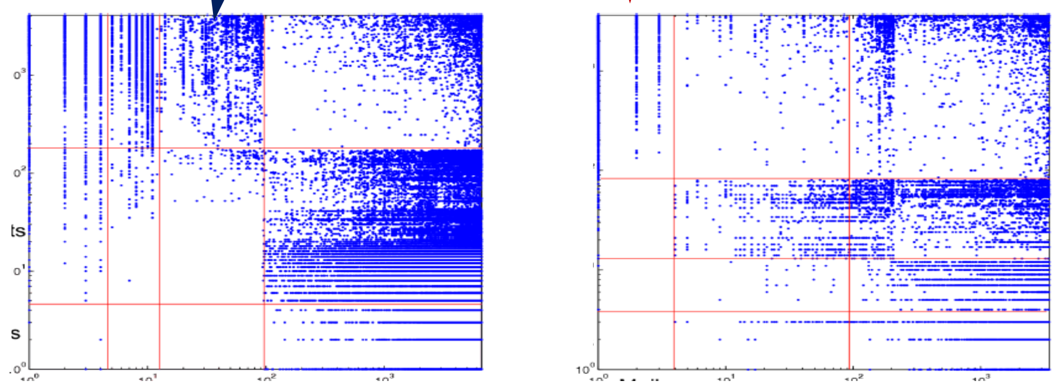
Community-based events

Sun et al. ’07

- Goal: partition the graph sequence into segments $G^{(1)}$ $G^{(2)}$ where



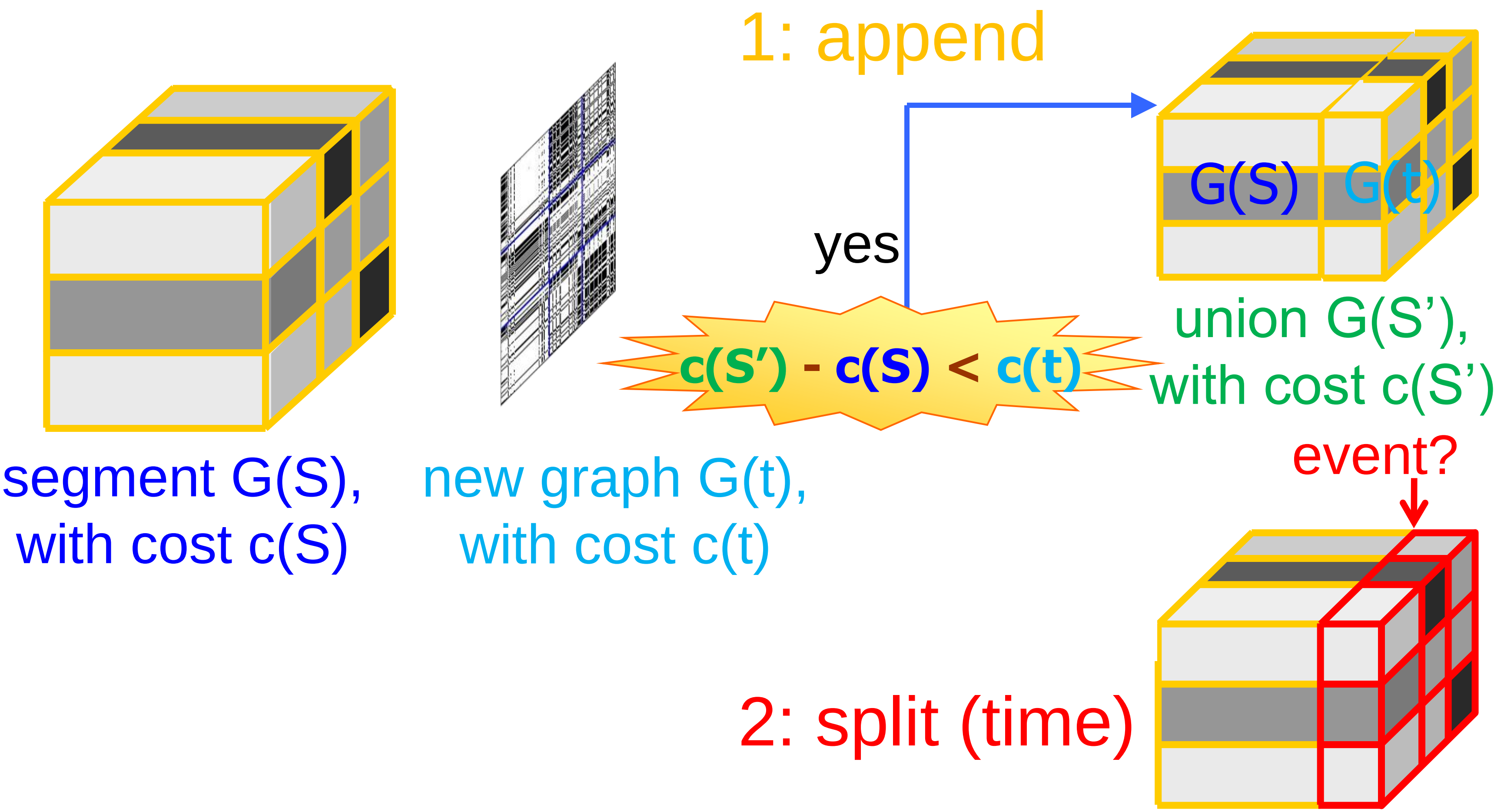
each segment exhibits a (different) clustering



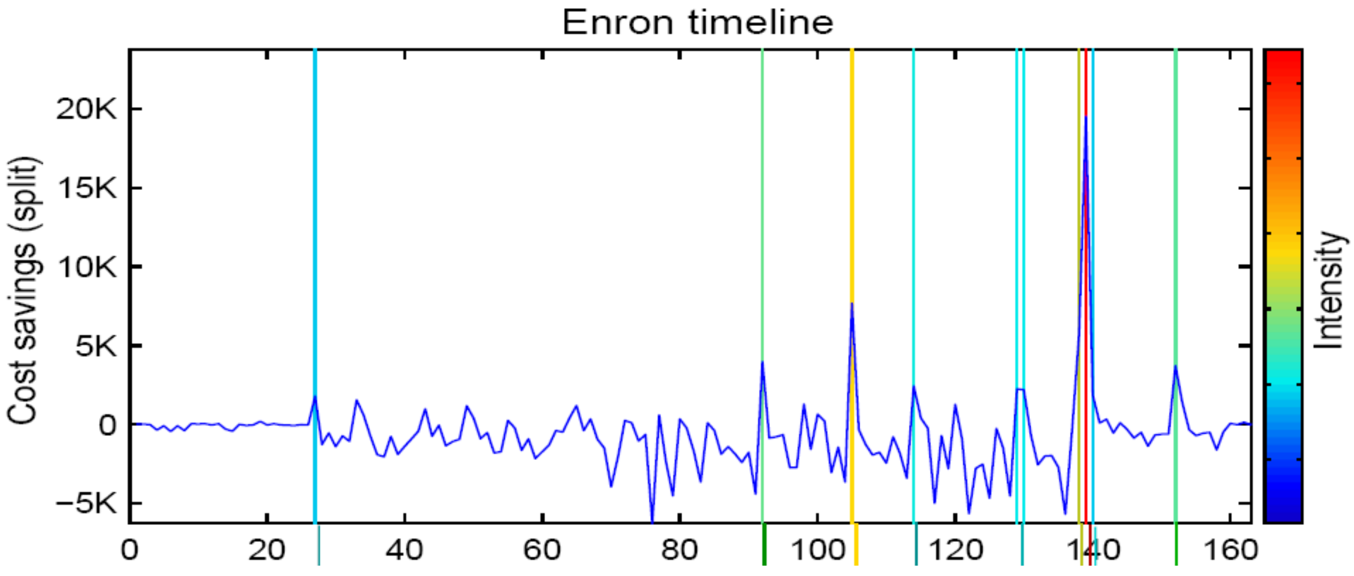
- Q: when does a new segment (=event) emerge?

Change detection

- Guiding principle: encoding cost benefit



Community changes in Enron

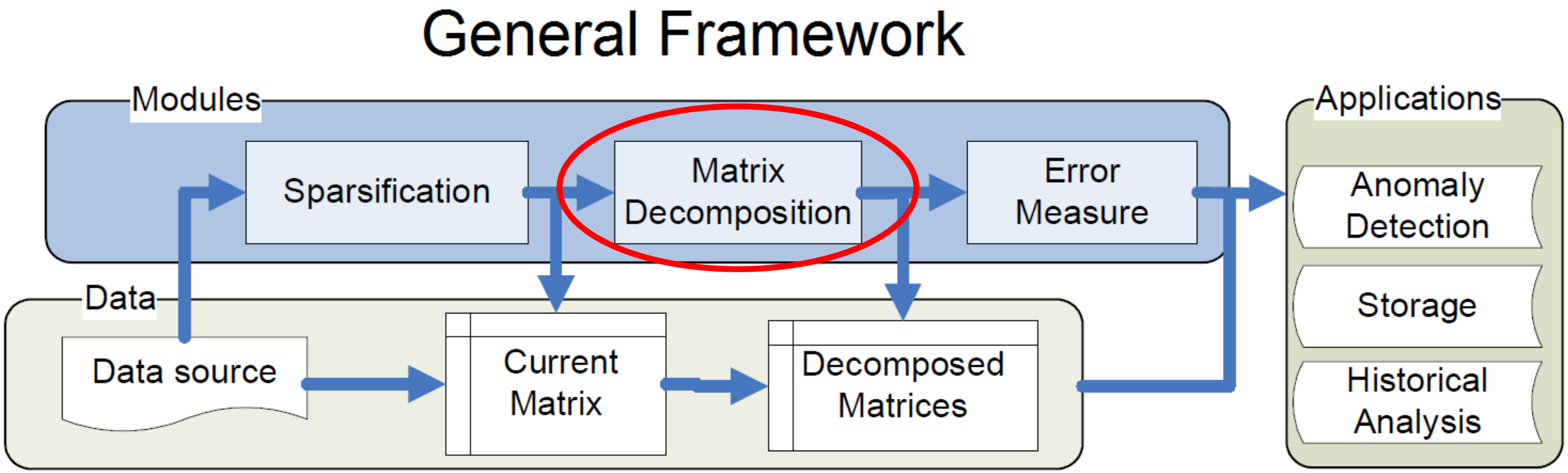


- 34K email addresses
- 165 weeks
- ~2M emails

Key change-points correspond to key events

Bit-cost can **quantify** event "intensity"

Reconstruction-based events



- Network forensics
  - Sparsification ➔ load shedding
  - Matrix decomposition ➔ summarization
  - Error Measure ➔ anomaly detection

Matrix decomposition

- Goal: summarize a given graph

↓

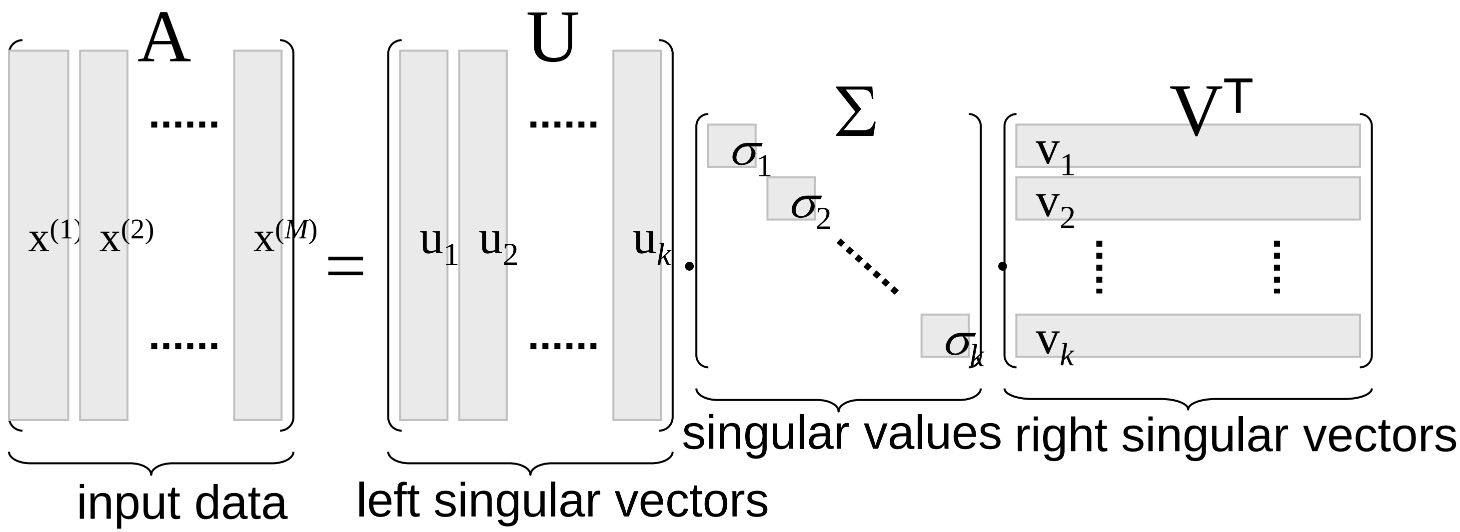
decompose adjacency matrix
into smaller components

↓

1. Singular Value Decomposition (SVD) **1800's, PCA, LSI, ...**
2. CUR decomposition **Drineas et al. '05**
3. Compact Matrix Decomposition (CMD) **Sun et al. '07**

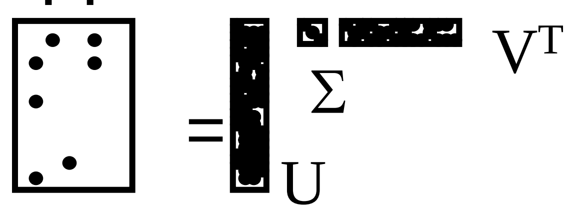
1. Singular Value Decomposition

$$A = U\Sigma V^T$$



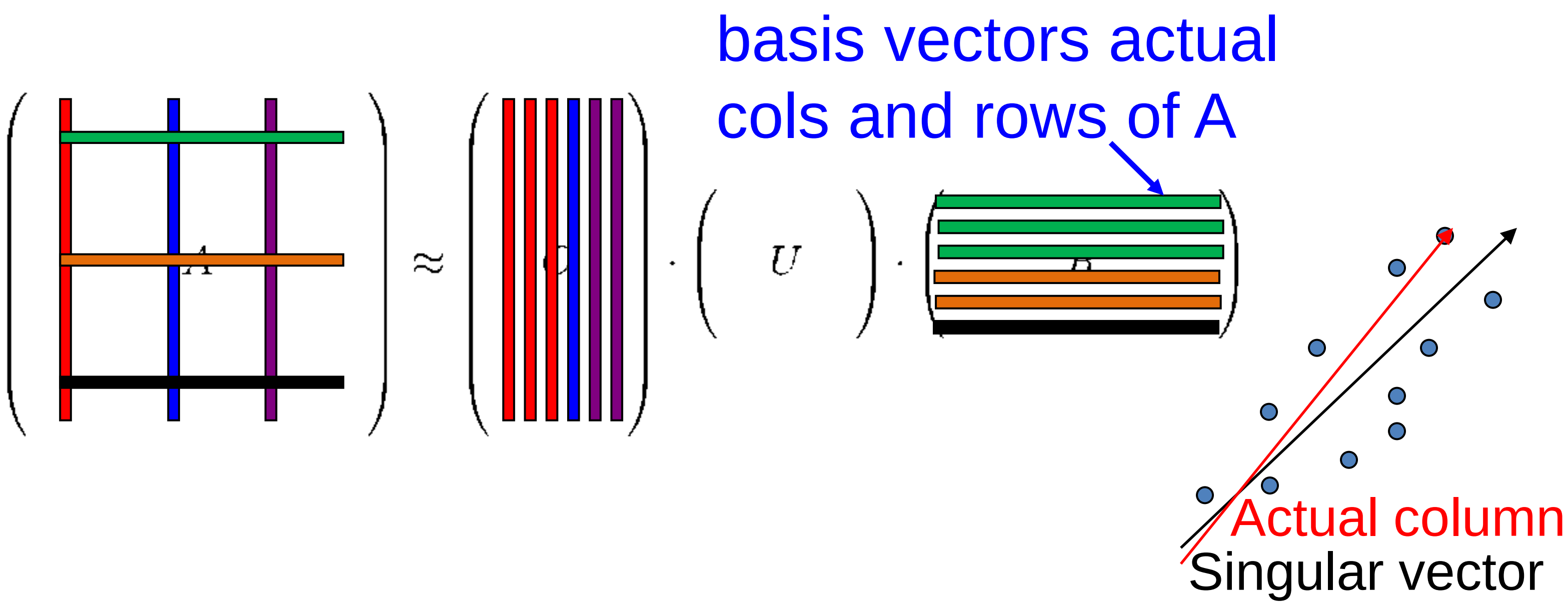
\+ Optimal low-rank approximation

\- Lack of Sparsity



2. CUR decomposition

C, U, R for small **||A-CUR||**



\+ Provably good approximation to SVD

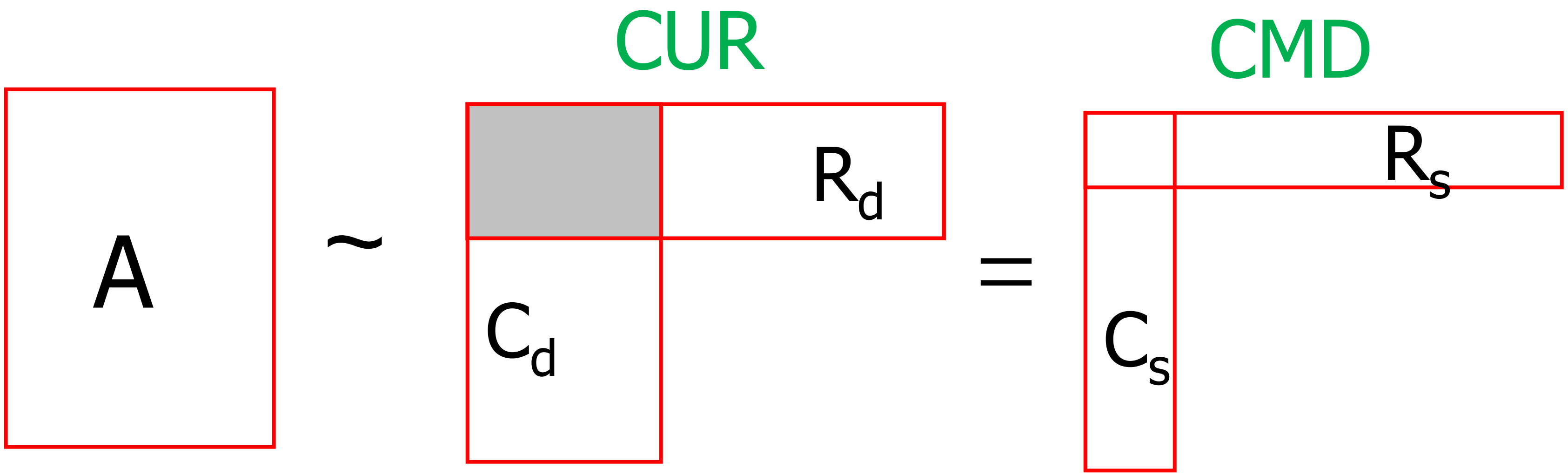
\+ Sparse basis (A is sparse)

\- Space overhead (duplicate bases)

3. Compact Matrix Decomposition

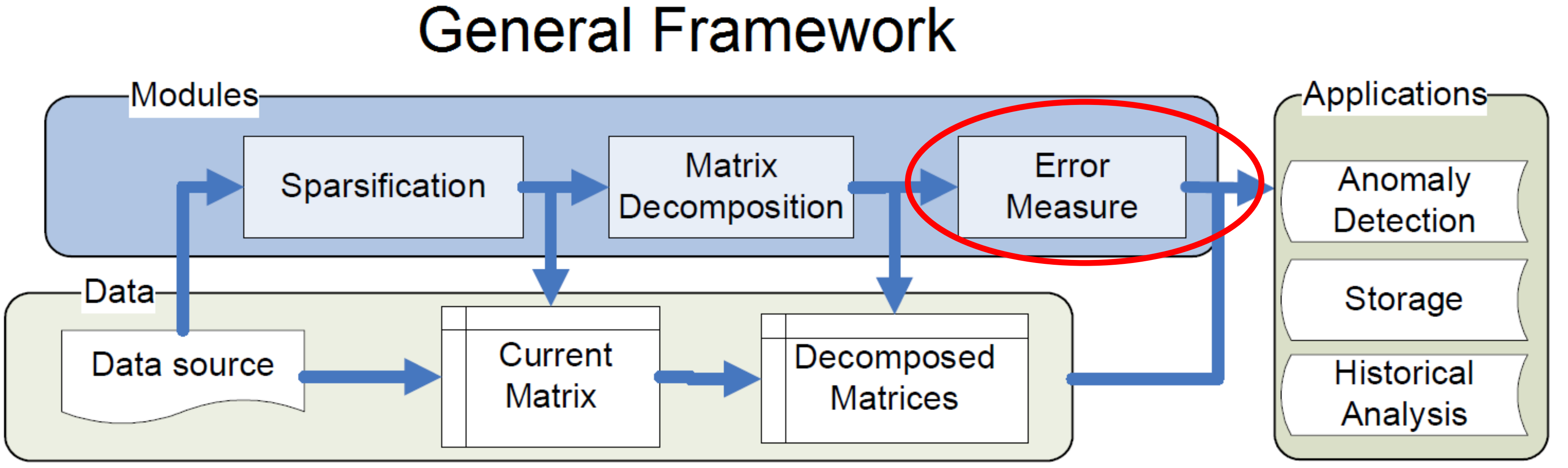
C, U, R for small ||A-CUR||, and

No duplicates in **C** and **R**



+ Sparse basis (A is sparse)
+ Efficiency in space and computation time

Reconstruction-based events



- Network forensics
  - Sparsification ➔ load shedding
  - Matrix decomposition ➔ summarization
  - Error Measure ➔ anomaly detection

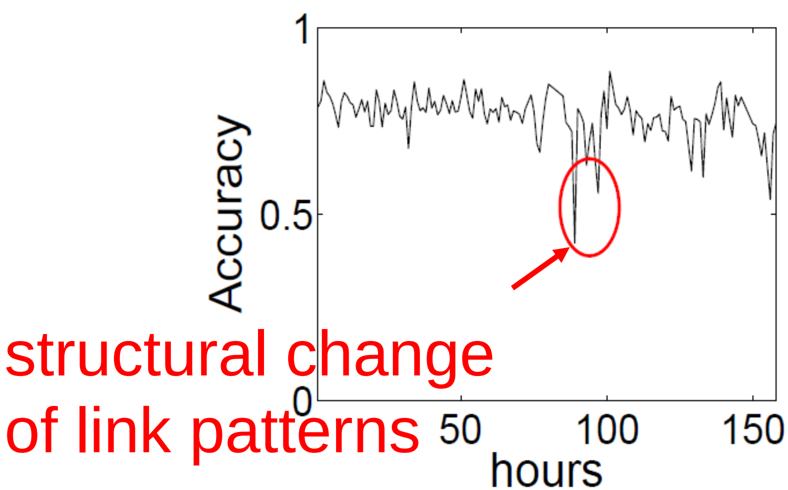
Error measure: reconstruction

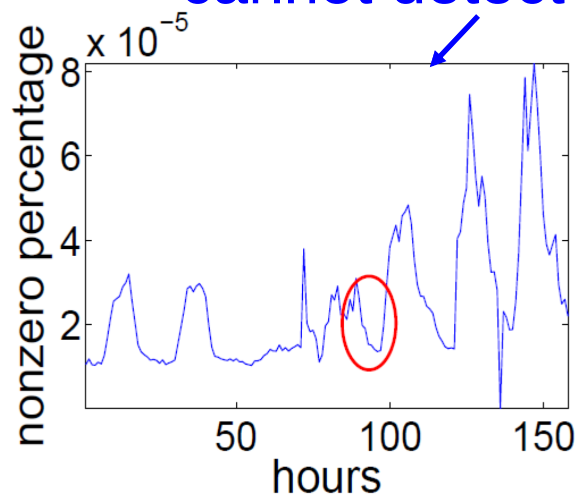
- accuracy = 1- Relative Sum-Square-Error

$$\text{RSSE} = \frac{\sum_{i,j}(\mathbf{A}(i,j) - \tilde{\mathbf{A}}(i,j))^2}{\sum_{i,j}(\mathbf{A}(i,j)^2)}$$

- Monitor accuracy over time

Volume monitoring cannot detect anomaly





- Also, high reconstruction error of rows/cols for static snapshot anomalies

Practical issue 1: non-linear scaling

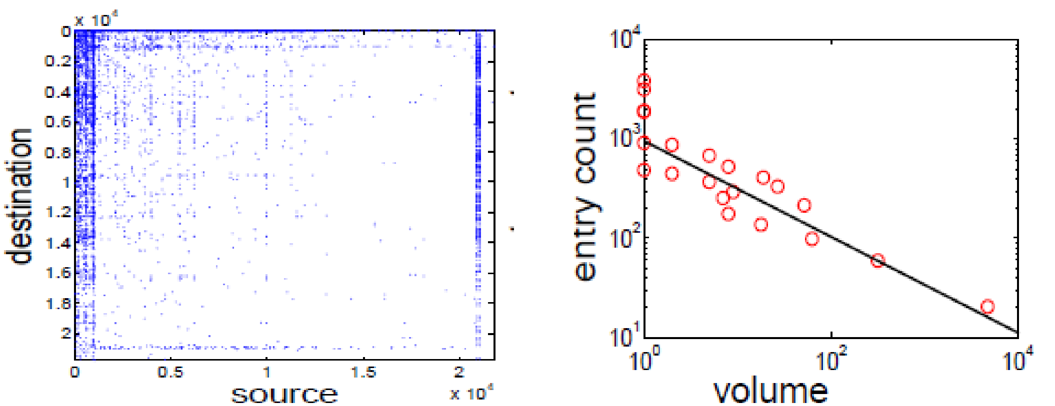
- Issue: skewed entries in A matrix

  ↓

  few "heavy" rows/cols dominate (CUR/CMD) decomposition

  ↓

  poor anomaly discovery



- Solution: rescale entries *x* by *log(x+1)*

Practical issue 2: fast approx. error

- Issue: Direct computation of SSE is costly; norm of two big matrices, $\mathbf{A}$ and $\mathbf{A}-\tilde{\mathbf{A}}$, are needed.
- Solution: approximated error

$$\tilde{e} = \frac{m \cdot n}{|S|} \sum_{(i,j)\in S} (\mathbf{A}(i,j) - \mathbf{C}_{(i)}\mathbf{UR}^{(j)})^2$$

$$i \begin{pmatrix} \quad A \quad \end{pmatrix} \approx i \begin{pmatrix} \quad C \quad \end{pmatrix} \cdot \begin{pmatrix} \quad U \quad \end{pmatrix} \cdot \begin{pmatrix} \quad R \quad \end{pmatrix}$$

Part II: References (graph series)

- Shoubridge P., Kraetzl M., Wallis W. D., Bunke H. Detection of Abnormal Change in a Time Series of Graphs. Journal of Interconnection Networks (JOIN) 3(1-2):85-101, 2002.
- Shoubridge, P. Kraetzl, M. Ray, D. Detection of abnormal change in dynamic networks. Information, Decision and Control, 1999.
- Kelly Marie Kapsabelis, Peter John Dickinson, Kutluyil Dogancay. Investigation of graph edit distance cost functions for detection of network anomalies. ANZIAM J. 48 (CTAC2006) pp.436–449, 2007.
- Panagiotis Papadimitriou, Ali Dasdan, Hector Garcia-Molina. Web graph similarity for anomaly detection. J. Internet Services and Applications (JISA) 1(1):19-30 (2010)
- B. Pincombe. Anomaly Detection in Time Series of Graphs using ARMA Processes. ASOR BULLETIN, 24(4):2, 2005.
- Gao, Xinbo and Xiao, Bing and Tao, Dacheng and Li, Xuelong. A survey of graph edit distance. Pattern Anal. and App.s 13 (1), pp. 113-129. 2010.
- Horst Bunke, Peter J. Dickinson, Andreas Humm, Christophe Irniger, Miro Kraetzl: Computer Network Monitoring and Abnormal Event Detection Using Graph Matching and Multidimensional Scaling. Industrial Conference on Data Mining 2006:576-590

Part II: References (graph series) (2)

- C.E. Priebe, J.M. Conroy, D.J. Marchette, and Y. Park. Scan Statistics on Enron Graphs. Computational & Mathematical Organization Theory, 11(3):229–247, 2005.
- Ide, T. and Kashima, H., Eigenspace-Based Anomaly Detection in Computer Systems. KDD, 2004.
- L. Akoglu, M. McGlohon, C. Faloutsos. Event Detection in Time Series of Mobile Communication Graphs. Army Science Conference, 2010.
- Sun, Jimeng and Xie, Yinglian and Zhang, Hui and Faloutsos, Christos. Less is more: Compact matrix representation of large sparse graphs. ICDM 2007.
- Sun, Jimeng and Tao, Dacheng and Faloutsos, Christos. Beyond streams and graphs: dynamic tensor analysis. KDD 2006: 374-383
- Sun J., Faloutsos C., Papadimitriou S., Yu P. S. GraphScope: parameter-free mining of large time-evolving graphs. KDD, 2007.
- R. Rossi, B. Gallagher, J. Neville, and K. Henderson. Role-Dynamics: Fast Mining of Large Dynamic Networks. 1st Workshop on Large Scale Network Analysis, WWW, 2012.
- Cemal Cagatay Bilgin , Bülent Yener . Dynamic Network Evolution: Models, Clustering, Anomaly Detection. Survey, 2008.

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