Network Science : Lecture VIII

Graph Pattern Mining

Computer Science Department
Data Mining Research

Nov 26, 2014

Announcement

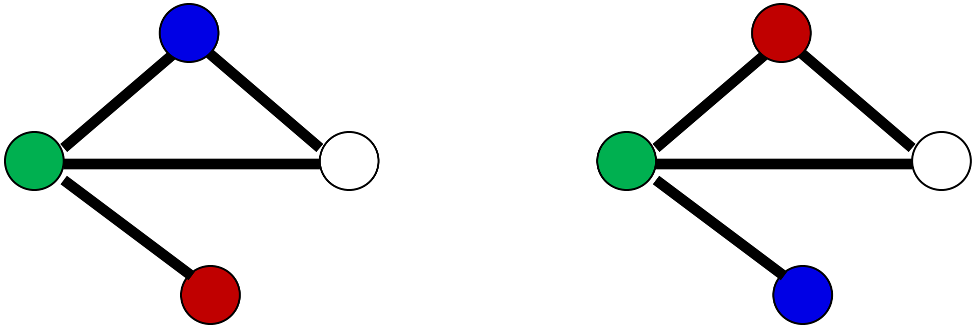
- No Homework
- Slides available at www.cs.ucsb.edu/~xyan/classes/NS201
- Two Quizzes (Dec 3, 10), mainly about concepts and ideas.

Graph Comparison

(Graph Comparison) Given two graphs G and G' from the space of graphs $\mathcal{G}$. The problem of graph comparison is to find a mapping

$$s : \mathcal{G} \times \mathcal{G} \rightarrow R$$

such that $s(G,G')$ quantifies the similarity (or dissimilarity) of *G and G'.*



Graph Isomorphism

(Graph Isomorphism) Find a mapping φ of the vertices of G to the vertices of G' such that G and G' are identical; i.e. (x,y) is an edge of G iff $(\varphi(x), \varphi(y))$ is an edge of G'. Then φ is an isomorphism, and G and G' are called isomorphic.

- No polynomial-time algorithm is known for graph isomorphism
- Neither is it known to be NP-complete

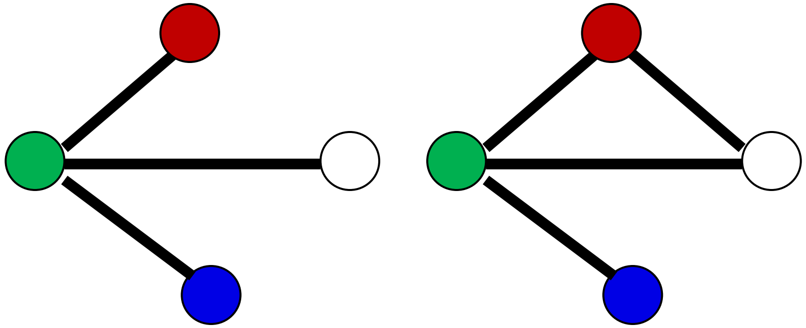
(Subgraph Isomorphism) Subgraph isomorphism asks if there is a subset of edges and vertices of G' that is isomorphic to a smaller graph G

- Subgraph isomorphism is NP-complete

Induced Subgraph Isomorphism

(Induced Subgraph Isomorphism) *G=(V,E)* is isomorphic to an induced subgraph of *G'=(V',E')* if there is an injective function φ which maps the vertices of *G* to vertices of *G'* such that for all pairs of vertices *x*, *y* in *V*, edge (*x*, *y*) is in *E* if and only if the edge (φ(*x*), φ(*y*)) is in *E'*.

- An **injective function** never maps distinct elements of its domain to the same element of its co-domain.
- Induced Subgraph isomorphism is NP-complete



Subgraph isomorphic,
Not induced subgraph isomorphic

Graph Edit Distance

- Edit Distance: Count the minimum operations needed to transform *G* into *G'*: edge/node insertion/deletion, modification of labels
- Variant: Assign costs to different types of operations
- Pros
  - Captures topological similarities between graphs
- Cons
  - Very expensive (NP-hard)
  - Choosing cost function for different operations is difficult

Maximum Common Subgraph

- Given two graphs *G* and *G'*, the maximum common subgraph is the largest subgraph of *G* isomorphic to a subgraph of *G'*.

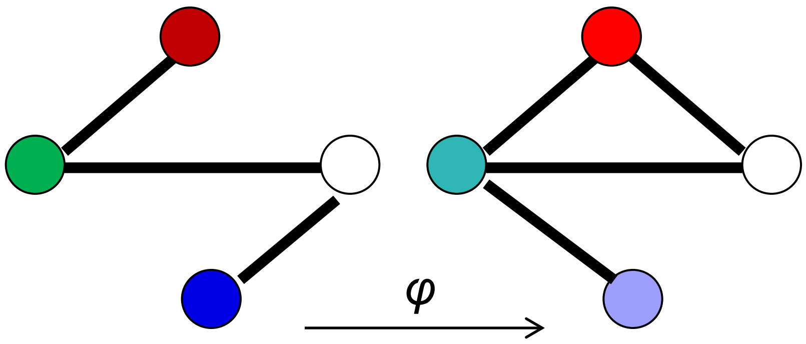
- The distance of *G* and *G'* and be defined as

$$\frac{2|M|}{|G|+|G'|}$$

where is *M* the maximum common subgraph of *G* and *G'*

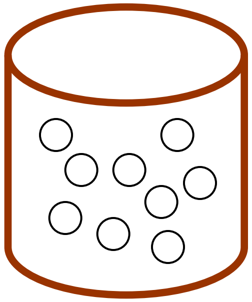
Attributed Graphs

- Node/Edge has labels
- Labels could be
  - Type of nodes/edges
  - Profiles, attribute/value lists
  - Messages between nodes
  - Time sequences
  - Any …,



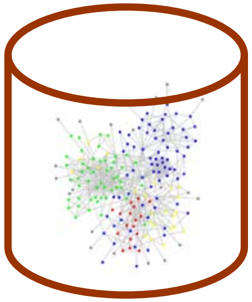
Graph Pattern Mining Scenarios

- Multiple Graphs Scenario



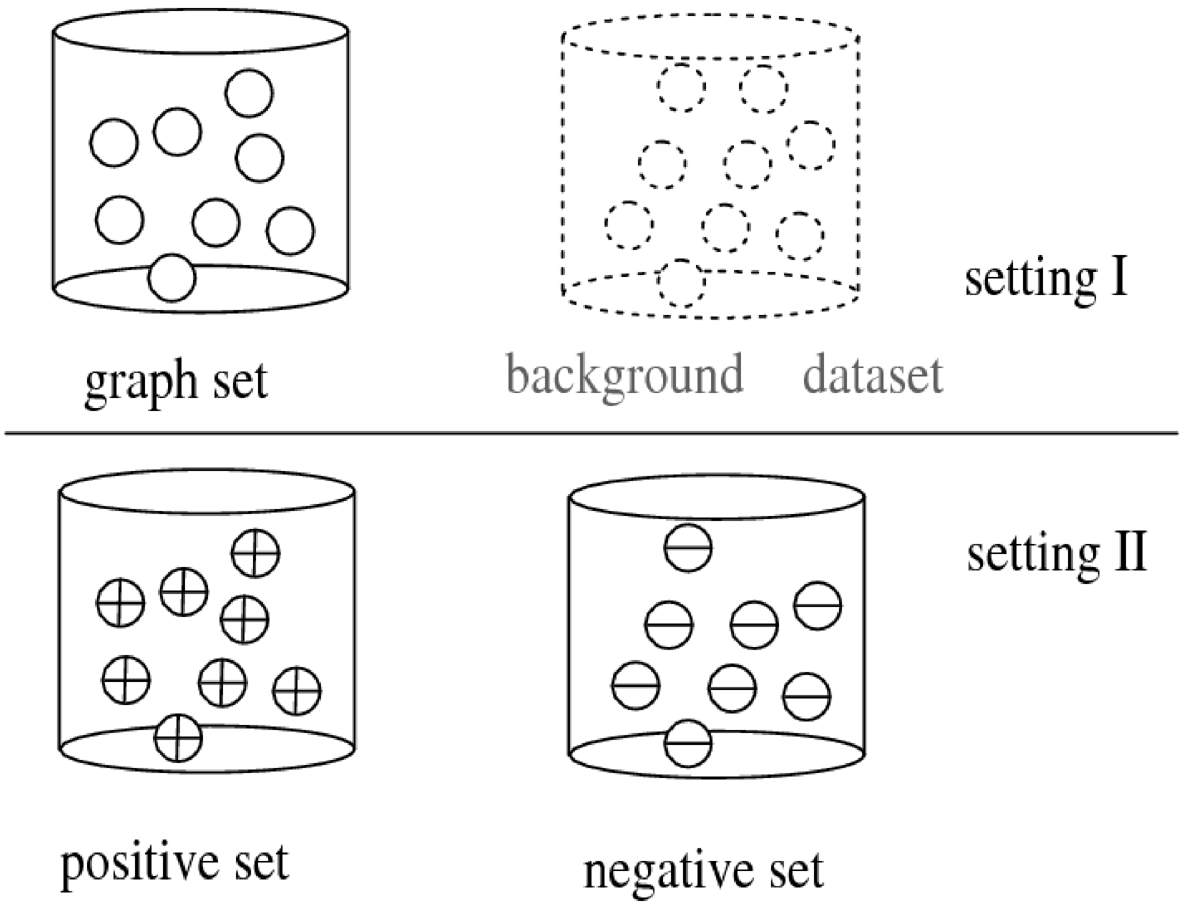
Multiple Graphs

- Single Graph Scenario



Single Graphs

Graph Pattern Mining



multiple graphs setting

Graph Pattern Mining

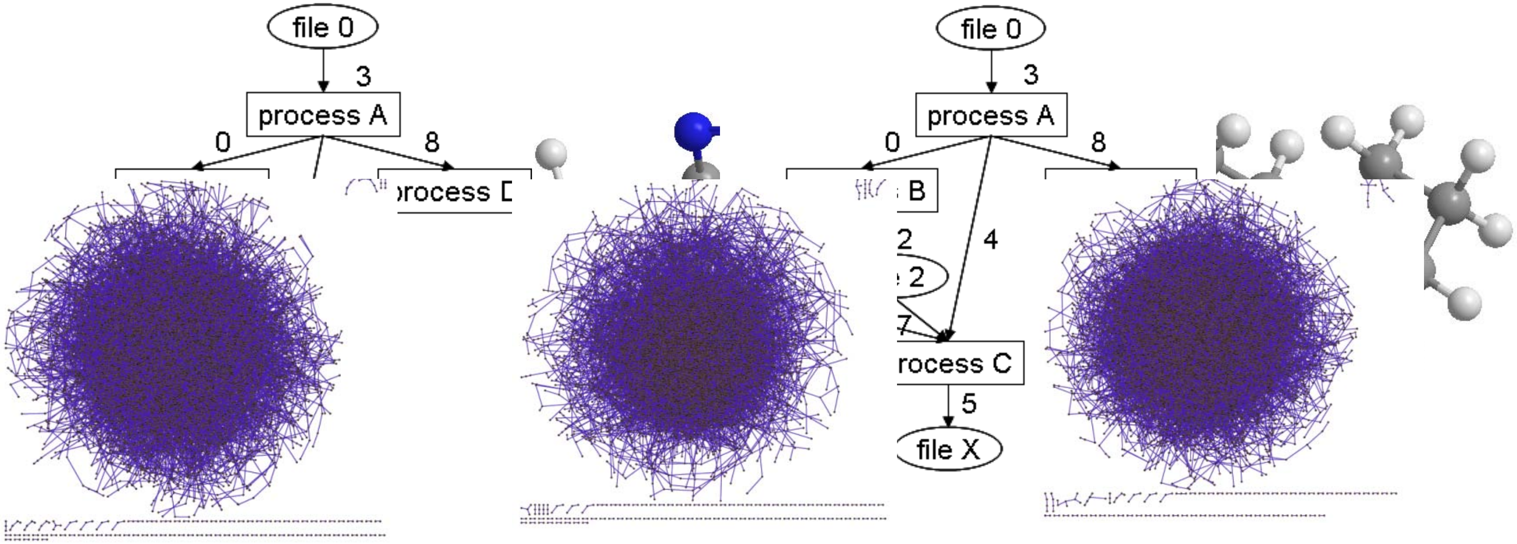
- Frequent graph patterns
- Optimal graph patterns
- Graph patterns with constraints
- Approximate graph patterns
- Pattern summarization

Applications of Graph Patterns

- Mining biochemical structures
- Finding biological conserved subnetworks
- Finding functional modules
- Program control flow analysis
- Intrusion network analysis
- Mining communication networks
- Anomaly detection
- Mining XML structures
- Building blocks for graph classification, clustering, compression, comparison, correlation analysis, and indexing

☐ …

Graph Patterns



Interestingness measures / Objective functions

- Frequency: frequent graph pattern
- Discriminative: information gain, Fisher score
- Significance: G-test
- …

Frequent Graph Pattern

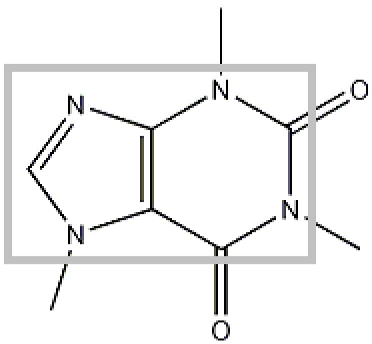
Given a graph dataset D, find subgraph g, s.t.

$$freq(g) \geq \theta$$

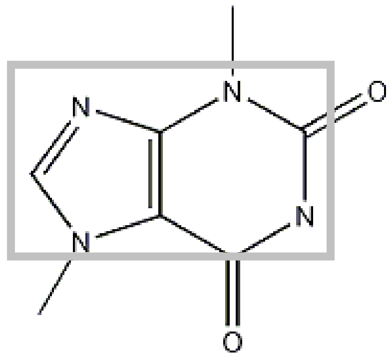
where $freq(g)$ is the percentage of graphs in D that contain g.

Example: Frequent Subgraphs

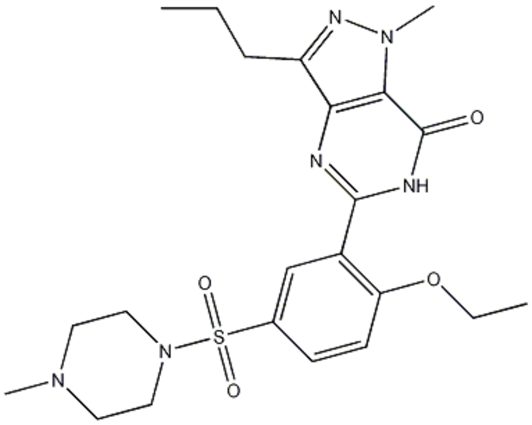
CHEMICAL COMPOUNDS



(a) caffeine



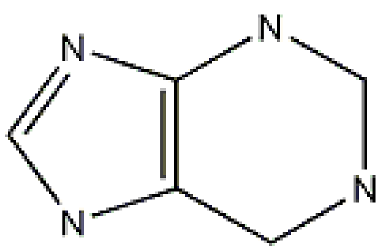
(b) diurobromine



(c) viagra

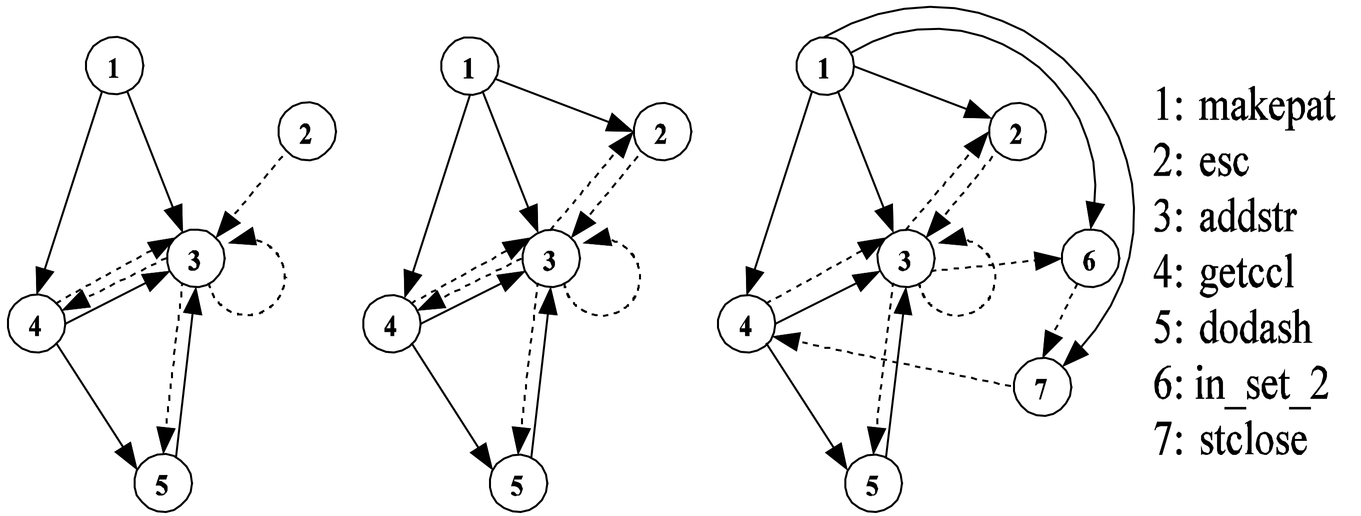
…

FREQUENT SUBGRAPH



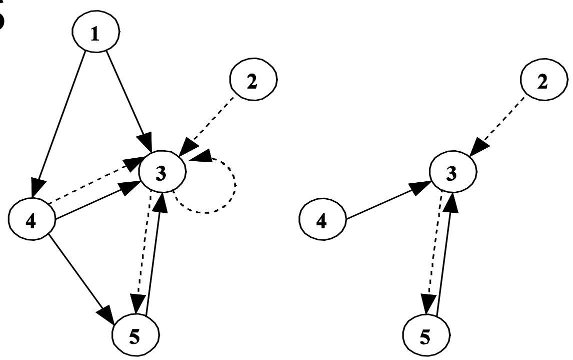
Example (cont.)

PROGRAM CALL GRAPHS



(1) (2) (3)

FREQUENT SUBGRAPHS
(MIN SUPPORT IS 2)



(1) (2)

Graph Mining Algorithms

Inductive Logic Programming (WARMR, King et al. 2001)

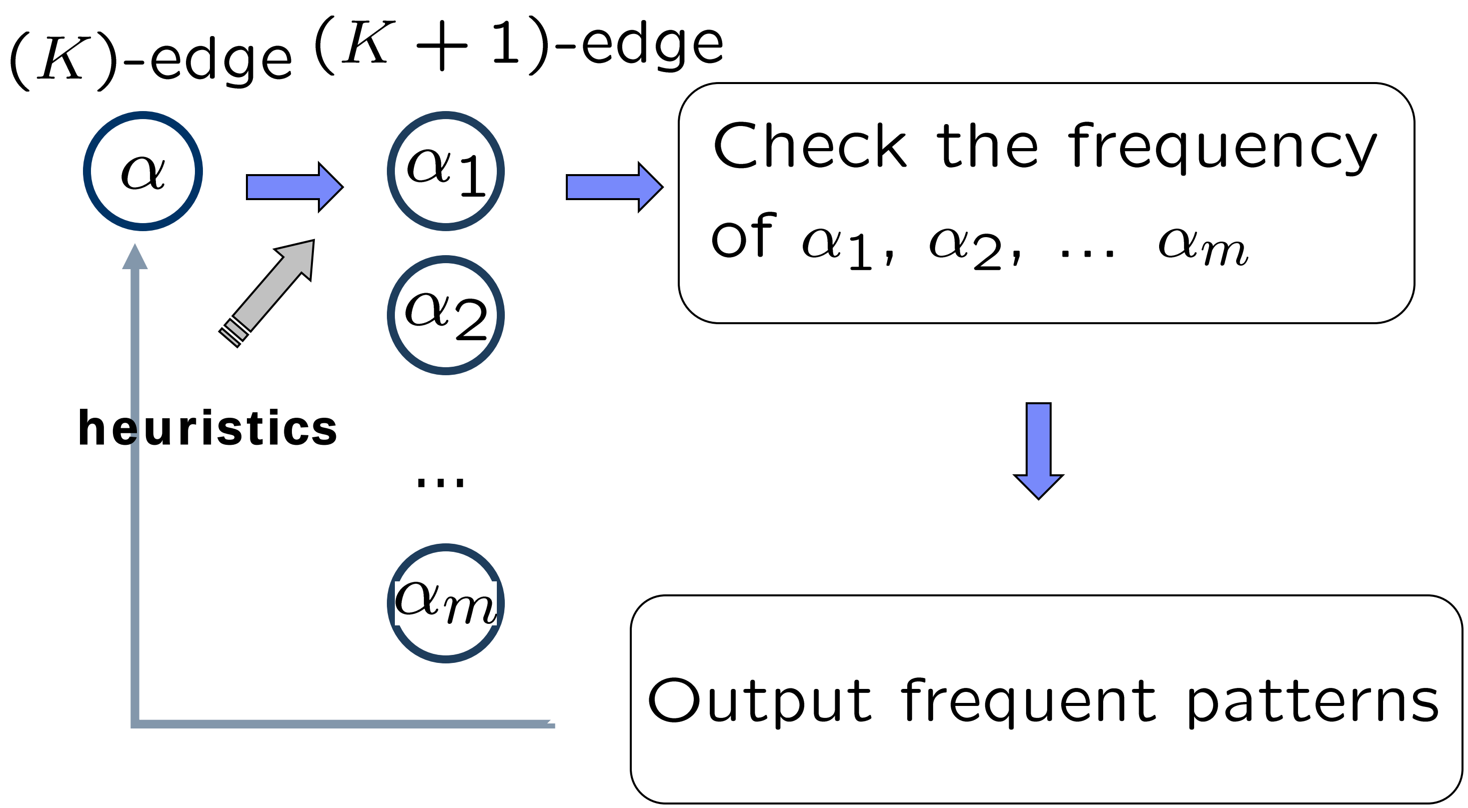
- Graphs are represented by Datalog facts

Graph Based Approaches

- Apriori-based approach
  - AGM/AcGM: Inokuchi, et al. (PKDD'00)
  - FSG: Kuramochi and Karypis (ICDM'01)
  - PATH#: Vanetik and Gudes (ICDM'02, ICDM'04)
  - FFSM: Huan, et al. (ICDM'03) and SPIN: Huan et al. (KDD'04)
  - FTOSM: Horvath et al. (KDD'06)
- Pattern growth approach
  - Subdue: Holder et al. (KDD'94)
  - MoFa: Borgelt and Berthold (ICDM'02)
  - gSpan: Yan and Han (ICDM'02)
  - Gaston: Nijssen and Kok (KDD'04)
  - CMTreeMiner: Chi et al. (TKDE'05)
  - LEAP: Yan et al. (SIGMOD'08)

Apriori Property

If a graph is frequent, all of its subgraphs are frequent.



Cost Analysis

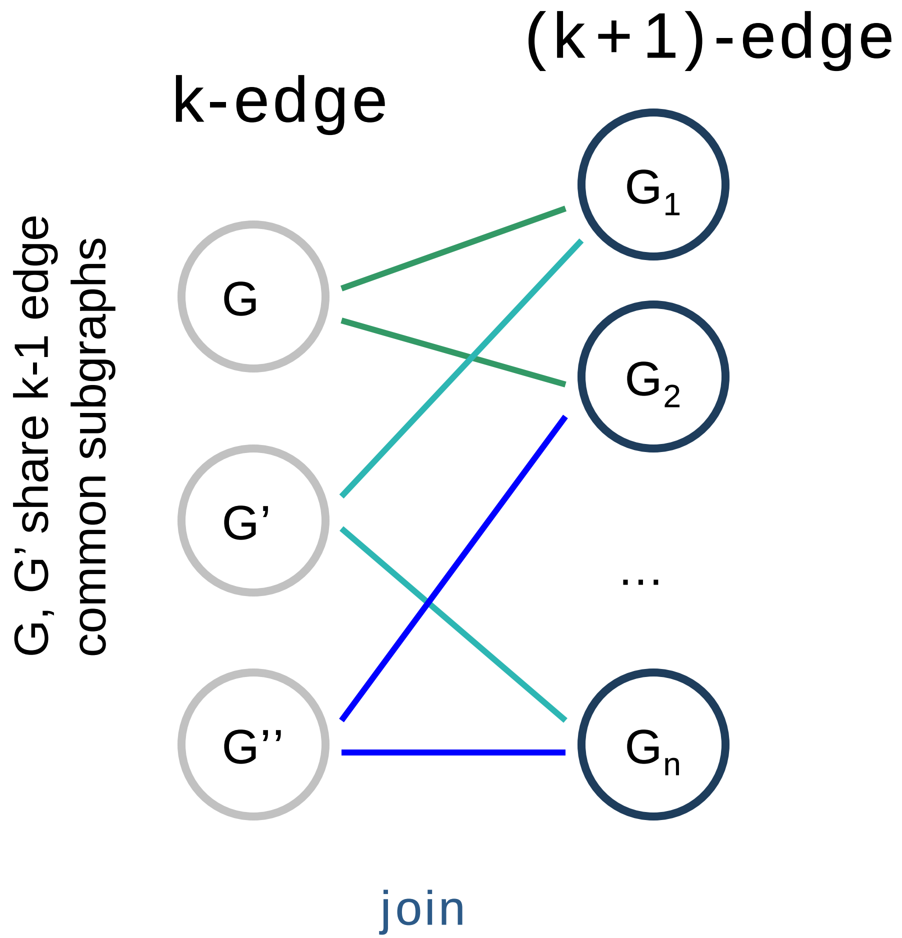
$$T_{total} \propto \sum_{\alpha} |D_{\alpha}| \times T_{\alpha}^{iso}$$

candidates
- frequent
- infrequent (X)
- duplicate (X)

data

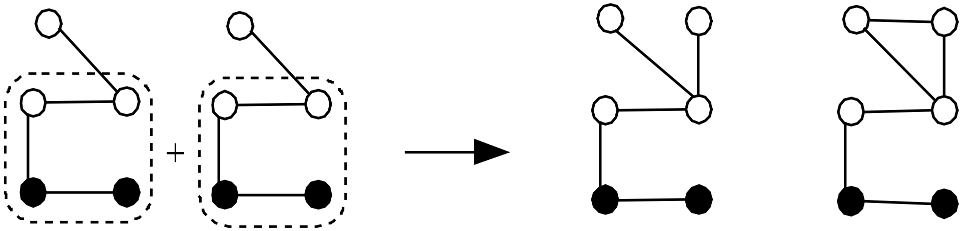
isomorphism checking

Apriori-Based Approach

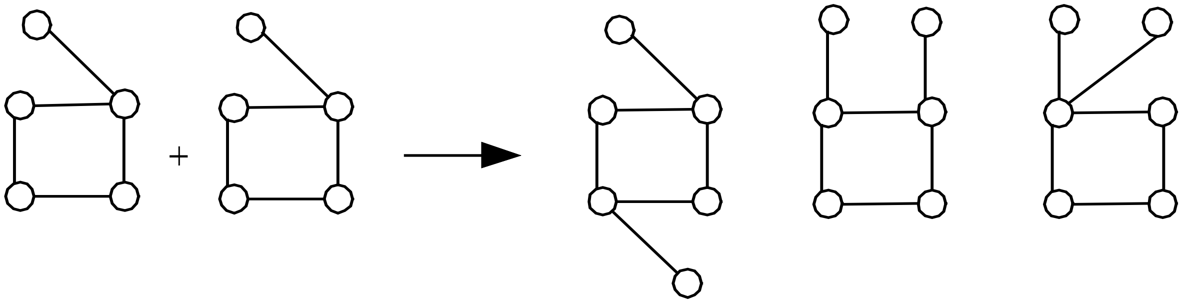


Apriori-Based, Breadth-First Search

- Methodology: breadth-search, joining two graphs

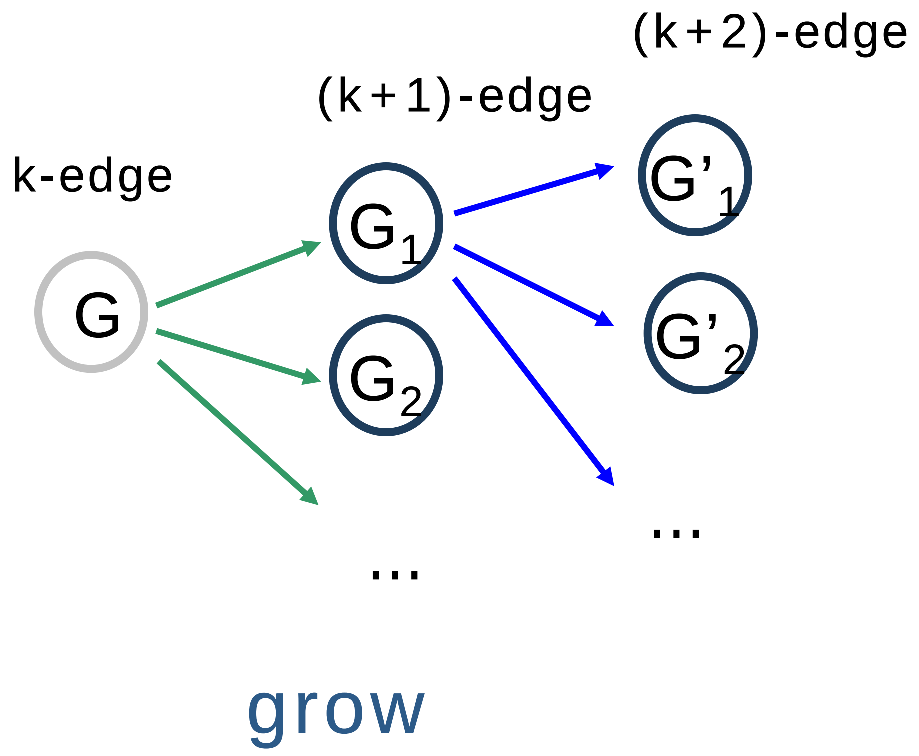


- AGM (Inokuchi, et al. PKDD'00)
  - generates new graphs with one more node

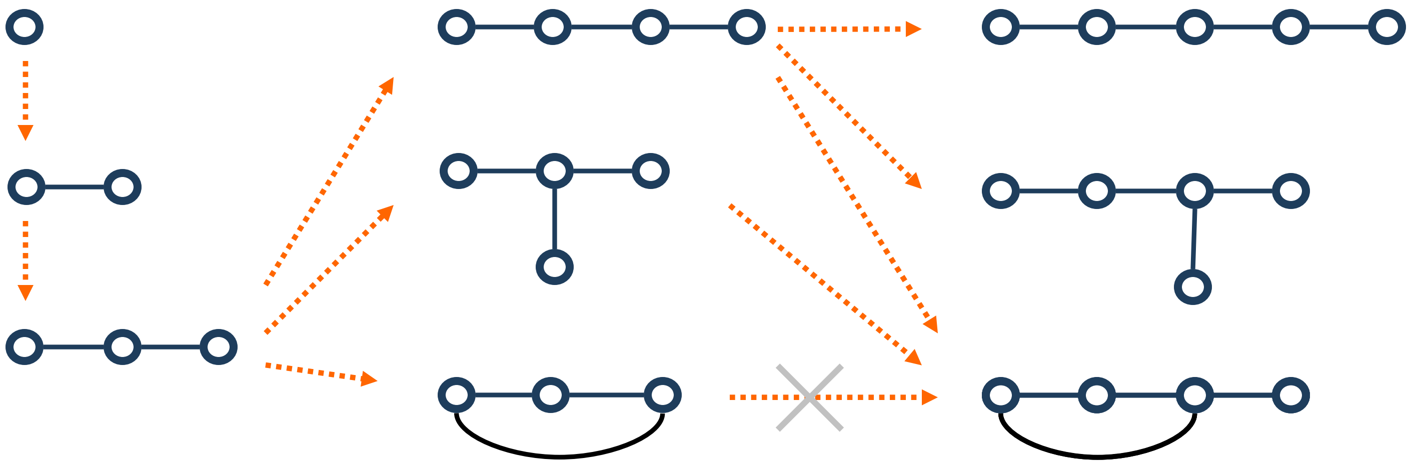


- FSG (Kuramochi and Karypis ICDM'01)
  - generates new graphs with one more edge

Pattern Growth Method

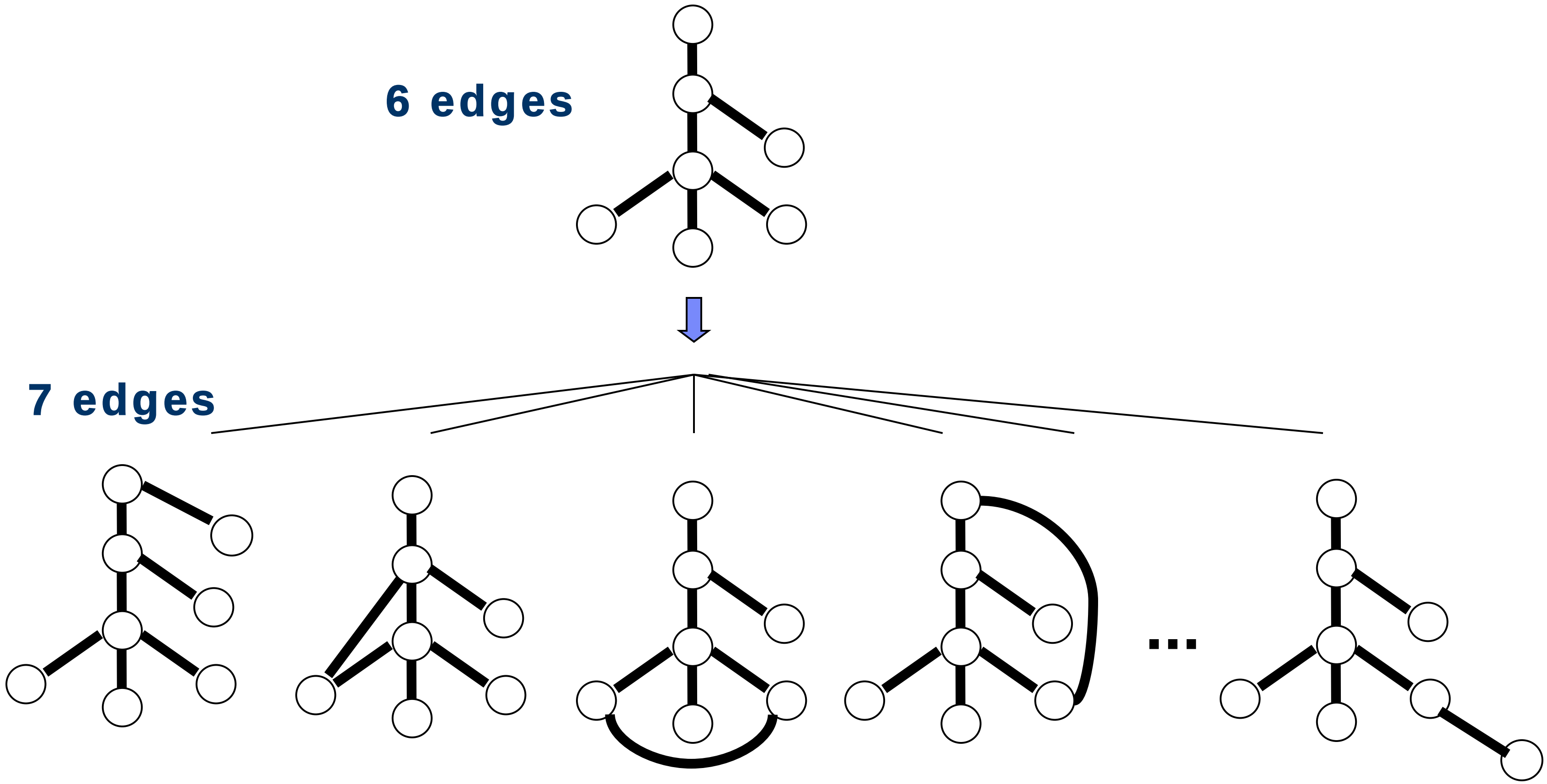


Pattern Growth Method



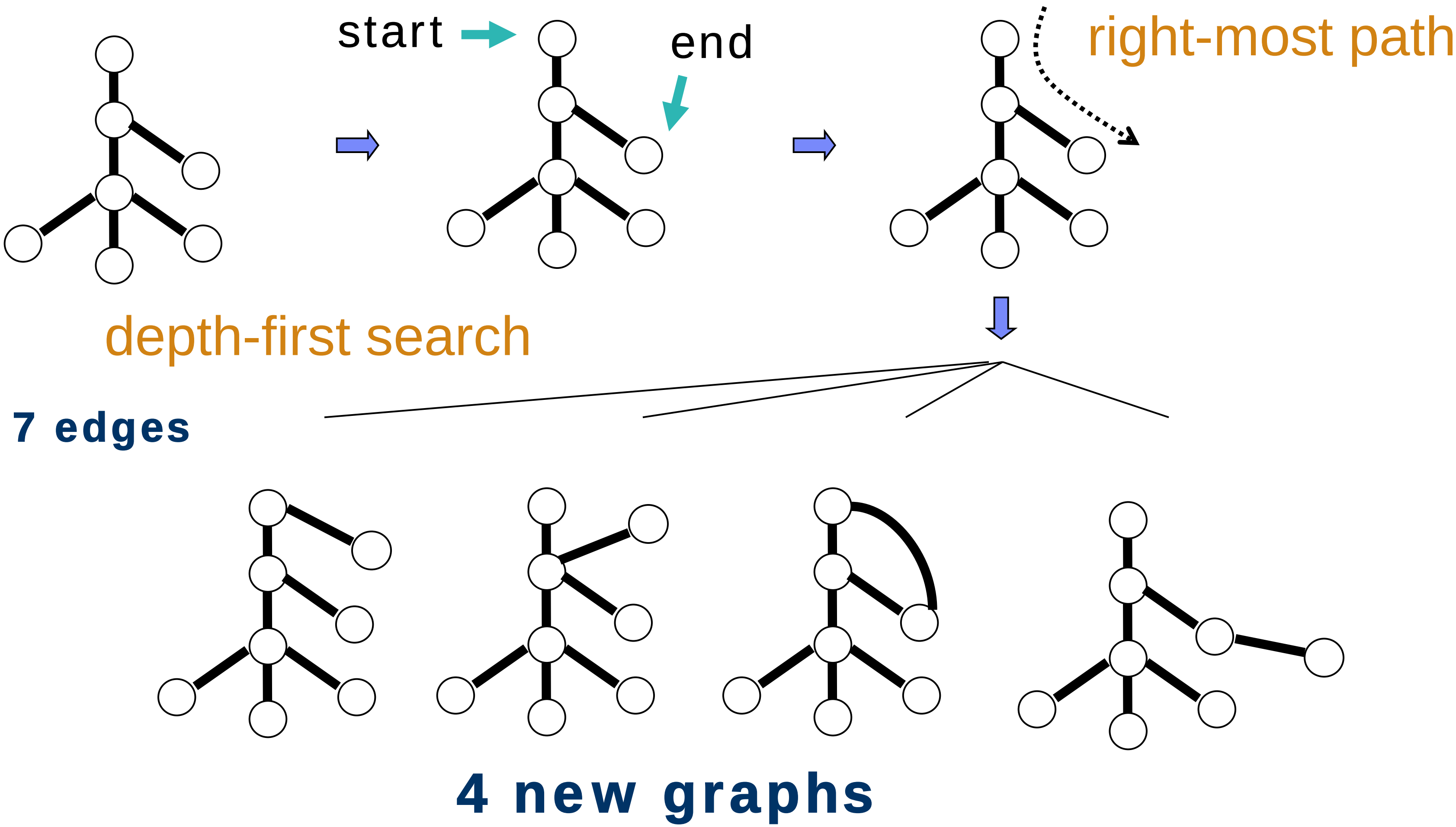
- detect duplicates
- avoid duplicates

Discovery Order: Free Extension



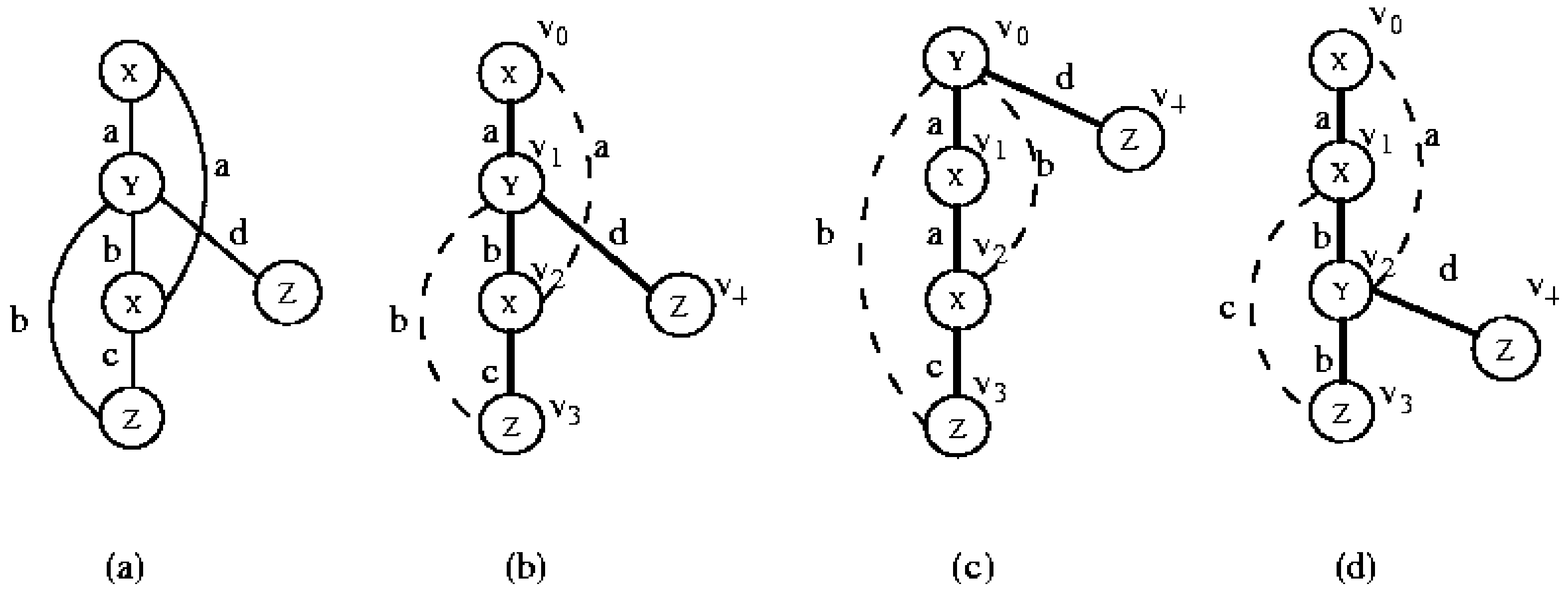
22 new graphs

Discovery Order: Right-Most Extension
(Yan and Han ICDM'02)



Depth First Search (DFS)

A depth-first search starting at one node in a graph, assuming the search remembers previously visited nodes and will not repeat them.

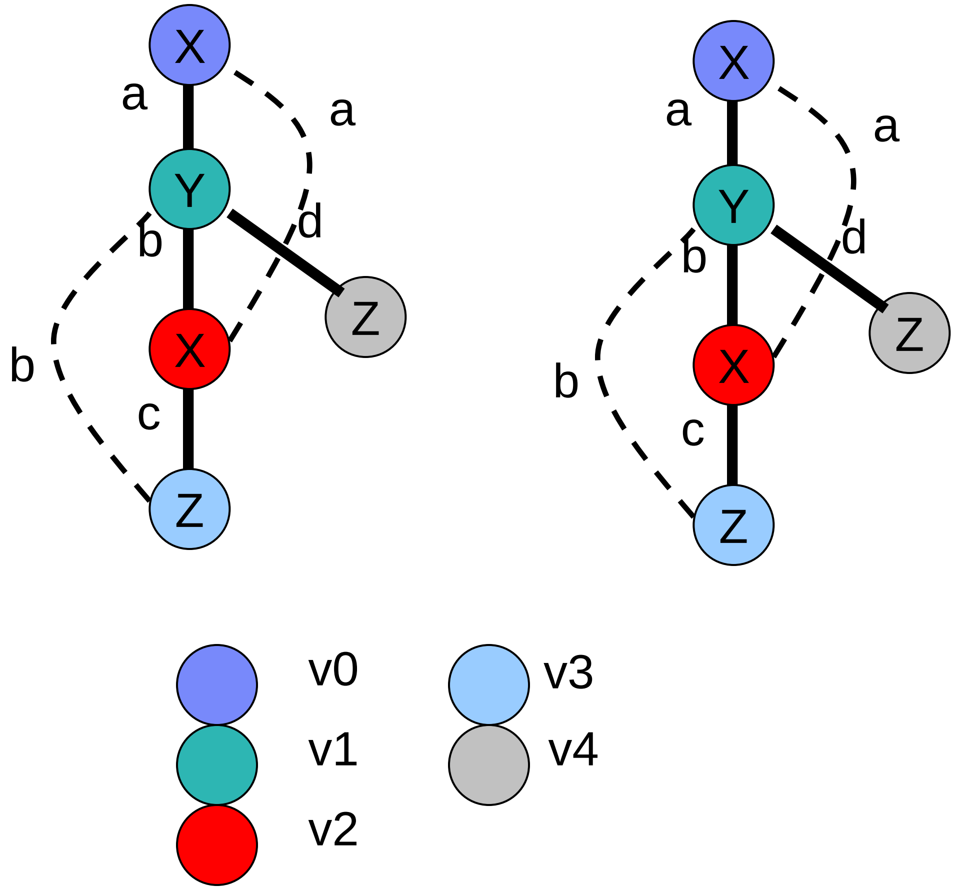


Forward Edge Set: Edges that are visited by a DFS
Backward Edge Set: Edges that are not visited by a DFS

DFS code and Minimum DFS code

- We use a 5-tuple $(v_i, v_j, l(v_i), l(v_j), l(v_i,v_j))$ to represent an edge.
- Turn a graph into a sequence whose basic element is 5-tuple. Form the sequence in such an order:
  - To extend one new node, add the forward edge that connect one node in the old graph with this new node.
  - Add all backward edge that connect this new node to other nodes in the old graph
  - repeat this procedure.

DFS code



e0: (0,1,x,y,a)

e1: (1,2,y,x,b)

e2: (2,0,x,x,a)

e3: (2,3,x,z,c)

e4: (3,1,x,y,b)

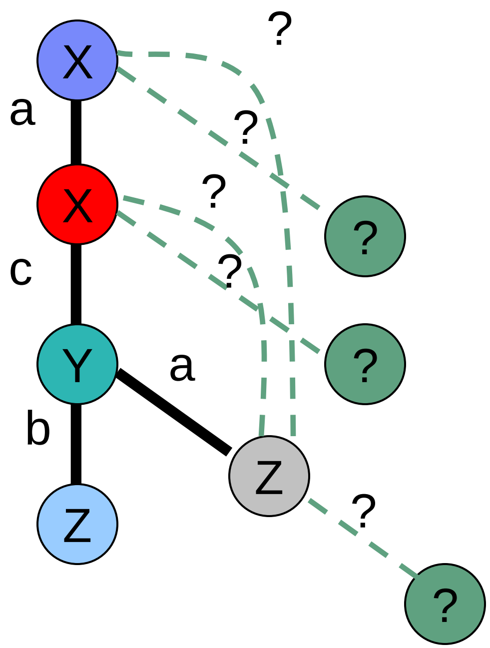
e5: (1,4,x,z,d)

Minimum DFS code

Each Graph may have lots of DFS code:
the smallest lexicographic one is its Minimum DFS Code

Edge no.	(B)	(C)	**(D)**
0	(0,1,x,y,a)	(0,1,y,x,a)	(0,1,x,x,a)
1	(1,2,y,x,b)	(1,2,x,x,a)	(1,2,x,y,b)
2	(2,0,x,x,a)	(2,0,x,y,b)	(0,1,y,x,a)
3	(2,3,x,z,c)	(2,3,x,z,c)	(2,3,y,z,a)
4	(3,1,z,y,b)	(3,0,z,y,b)	(3,1,z,x,c)
5	(1,4,x,z,d)	(0,4,y,z,d)	(2,4,y,z,d)

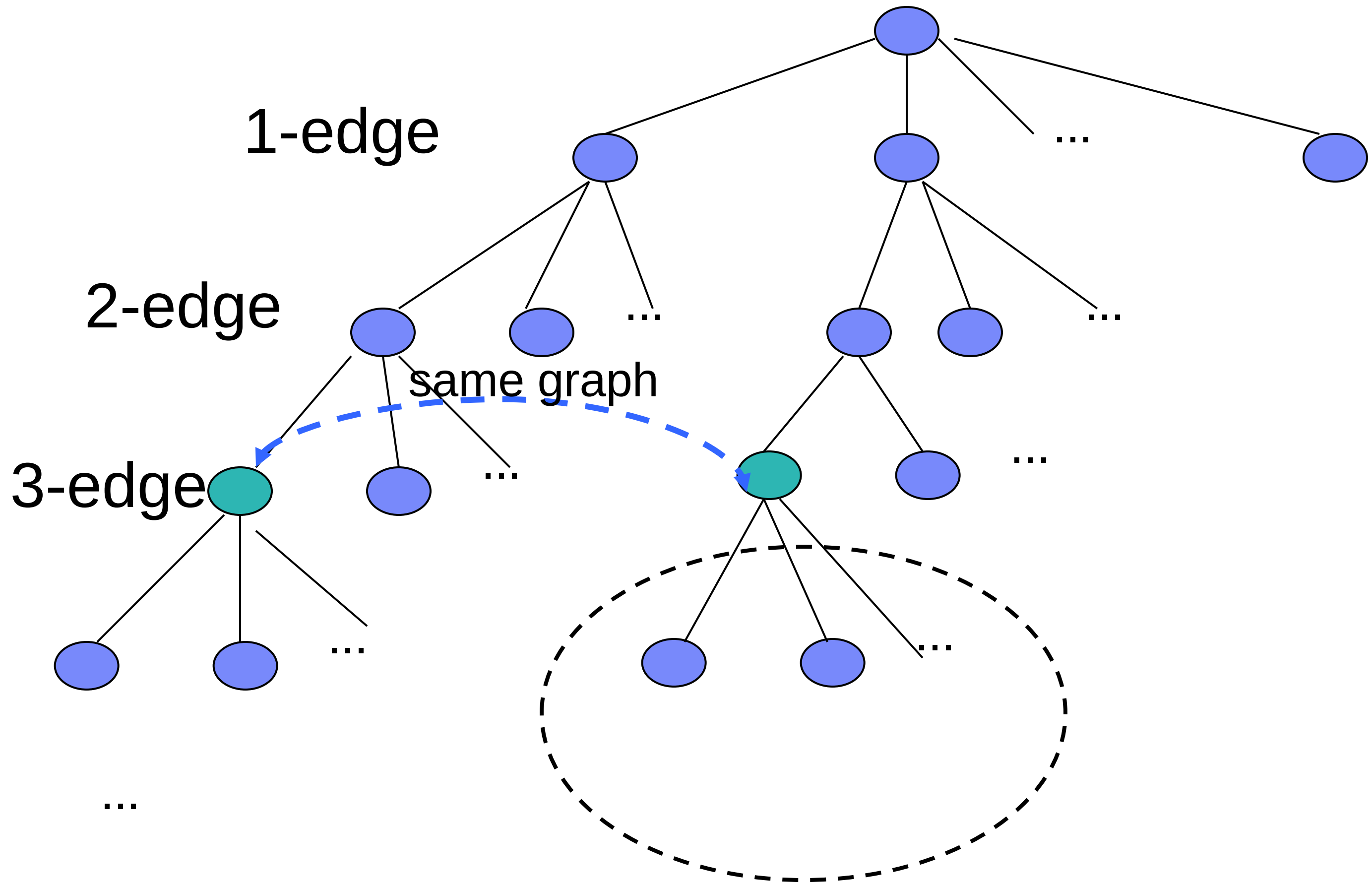
Parent and its Children



Given a minimum DFS code
$c_0=(e_0,e_1,\ldots,e_n)$
$c_1=(e_0,e_1,\ldots,e_n,e_x)$

c_0 is c_1's parent.
c_1 might not a minimum DFS code

DFS Code Tree



Theorems

- 1. Given two graphs G_0 and G_1, G_0 is isomorphic to G_1 iff min_dfs_code(G_0)=min_dfs_code(G_1).
- 2. DFS Code Tree covers all graphs although some tree nodes may represent the same graph. (Covering)
- 3. Given a node in DFS Code Tree, if its DFS code is not its minimum DFS code, prune this node and its all descendants won't change "Covering".

Duplicates Elimination

Existing patterns $g_1, g_2, \cdots, g_m$

Newly discovered pattern g

Option 1

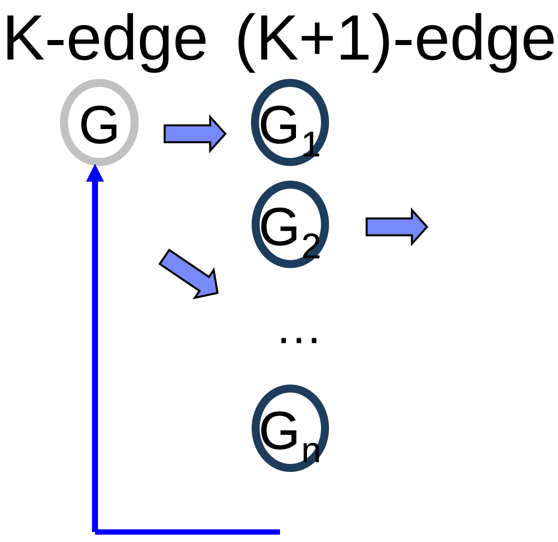
- Check graph isomorphism of g with each graph (slow)

- **Option 2**
  Transform each graph to a canonical label, create a hash value for this canonical label, and check if there is a match with g (faster)

- **Option 3**
  Build a canonical order and generate graph patterns in that order (fastest)

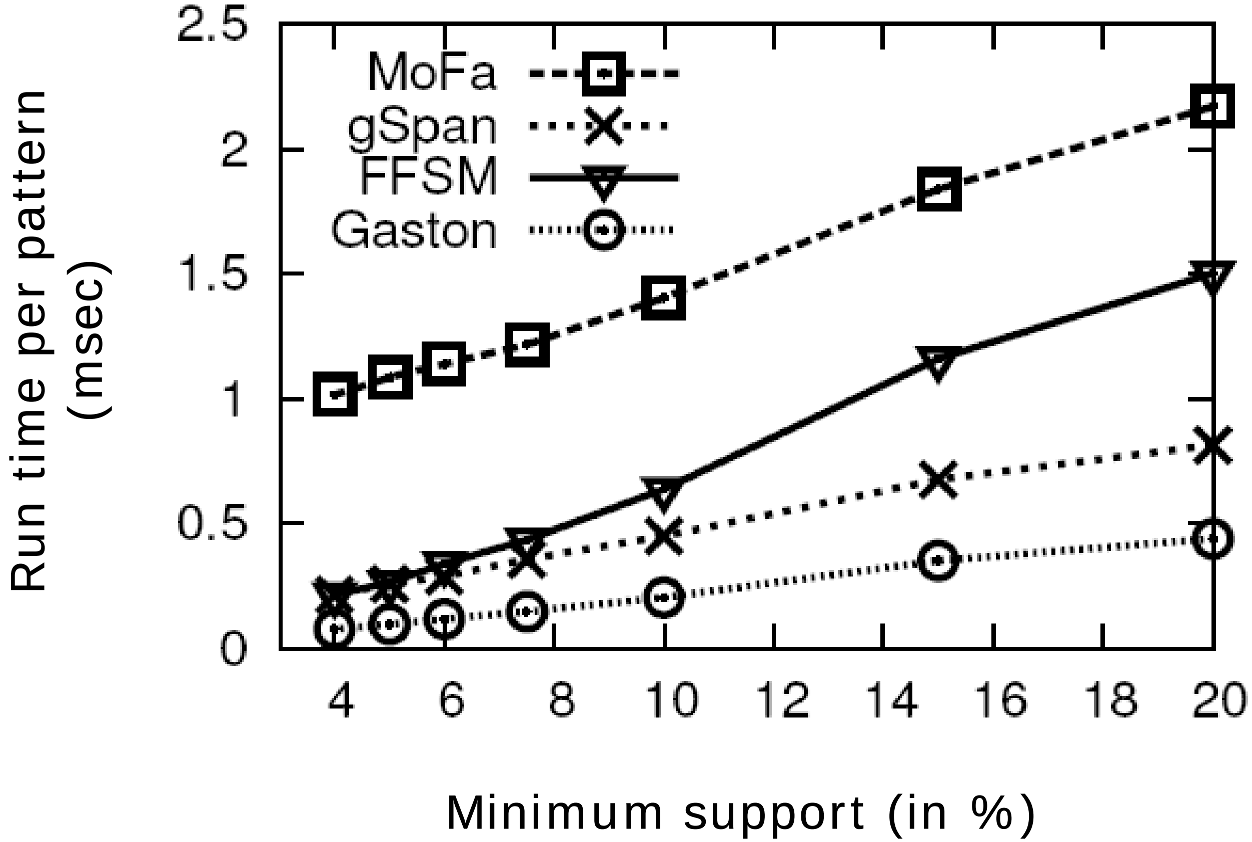
Properties of Graph Mining Algorithms

- Search order
  - breadth vs. depth
- Generation of candidate subgraphs
  - apriori vs. pattern growth
- Elimination of duplicate subgraphs
  - passive vs. active
- Support calculation
  - embedding store or not
- Discovery order of patterns
  - path → tree → graph

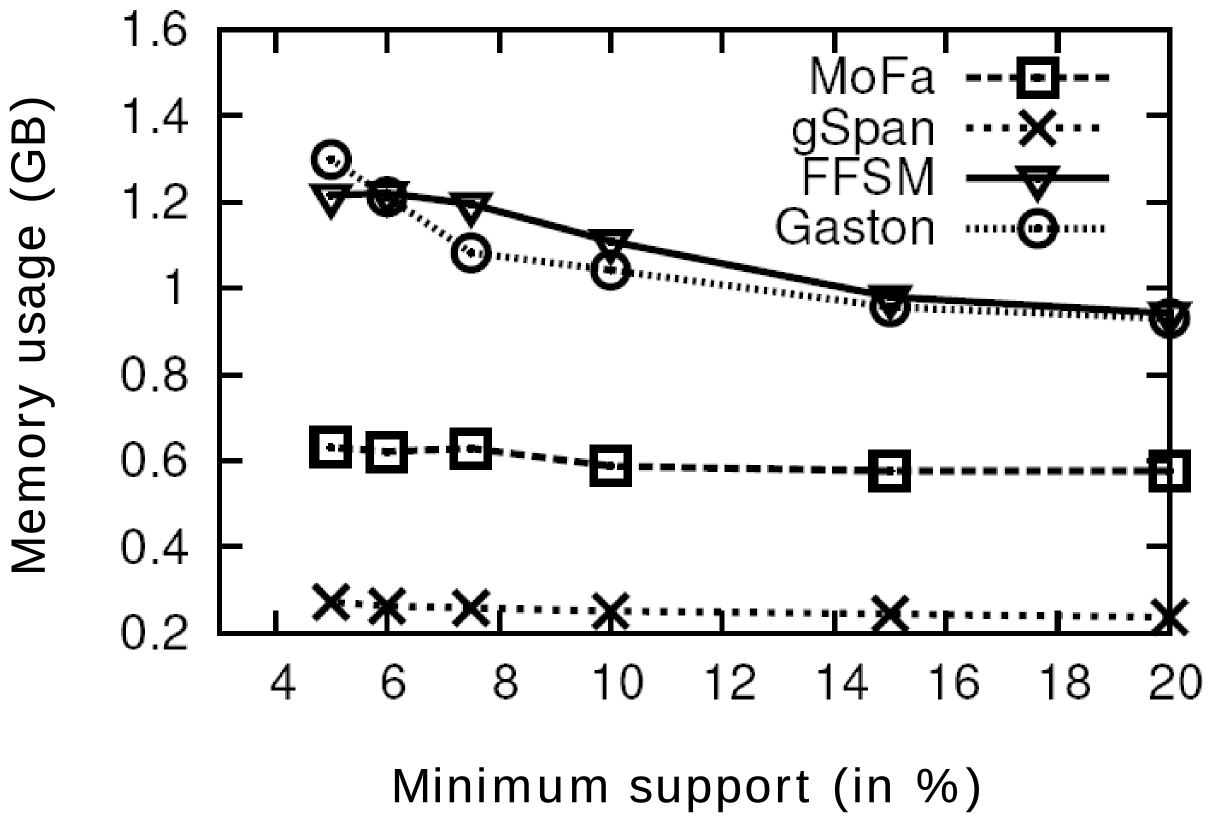


Performance: Run Time (Wörlein et al. PKDD'05)

The AIDS antiviral screen compound dataset from NCI/NIH



Performance: Memory Usage (Wörlein et al. PKDD'05)



Graph Pattern Explosion Problem

- If a graph is frequent, all of its subgraphs are frequent ─ **the Apriori property**
- An **n**-edge frequent graph may have 2^n subgraphs!
- In the AIDS antiviral screen dataset with **400+** compounds, at the support level 5%, there are > 1M frequent graph patterns

Conclusions: Many enumeration algorithms are available AGM, FSG, gSpan, Path-Join, MoFa, FFSM, SPIN, Gaston, and so on, but three significant problems exist.

Problem 1: Interpretation Poblem

Problem 2: Exponential Pattern Set

Problem 3: Threshold Setting

Closed and Maximal Graph Pattern

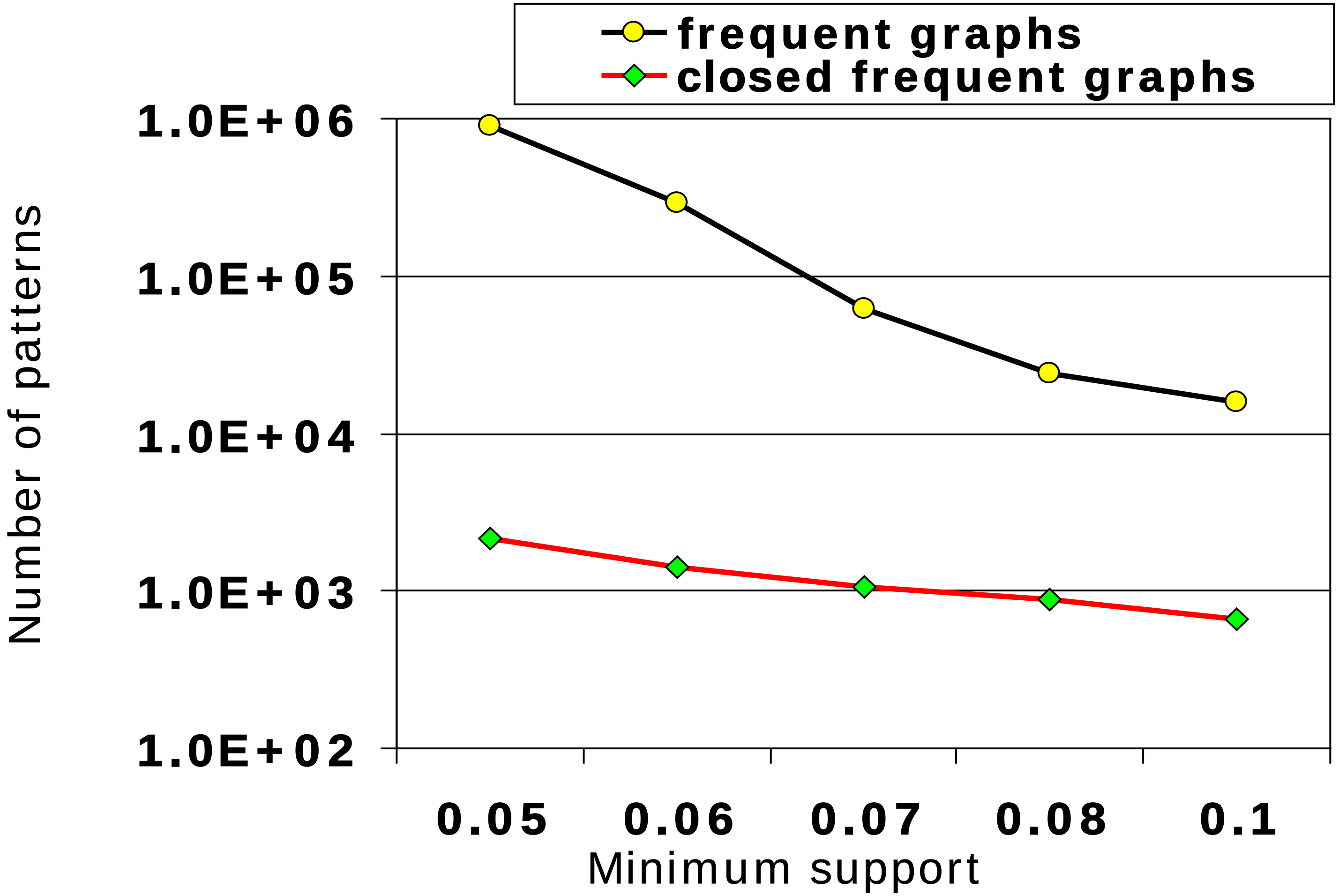
Closed Frequent Graph

- A frequent graph G is *closed* if there exists no supergraph of G that carries the same frequency as G
- If some of G's subgraphs have the same frequency, it is unnecessary to output these subgraphs (nonclosed graphs)
- *Lossless compression:* still ensures that the mining result is complete

Maximal Frequent Graph

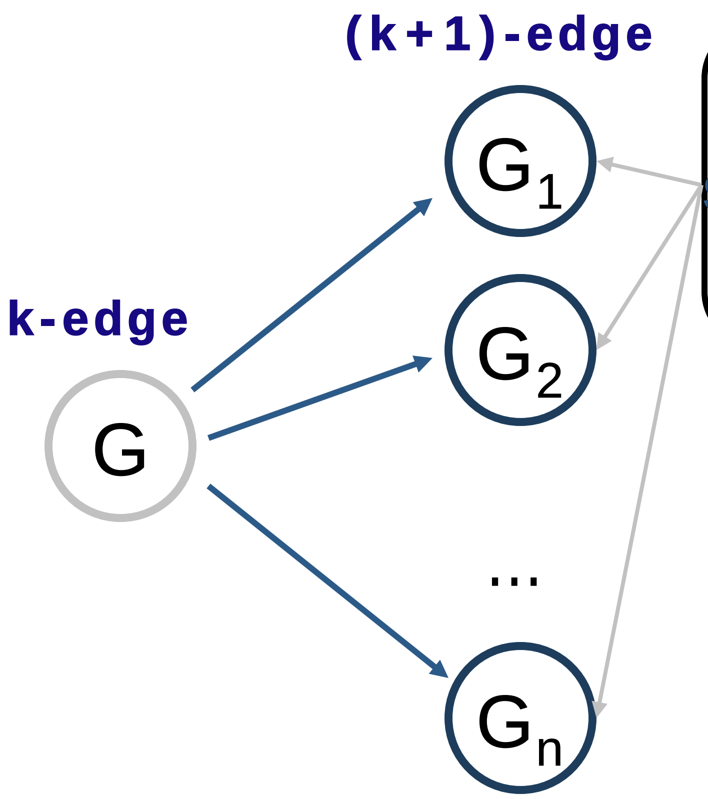
- A frequent graph G is *maximal* if there exists no supergraph of G that is frequent

Number of Patterns: Frequent vs. Closed



CLOSEGRAPH (Yan and Han, KDD'03)

A Pattern-Growth Approach

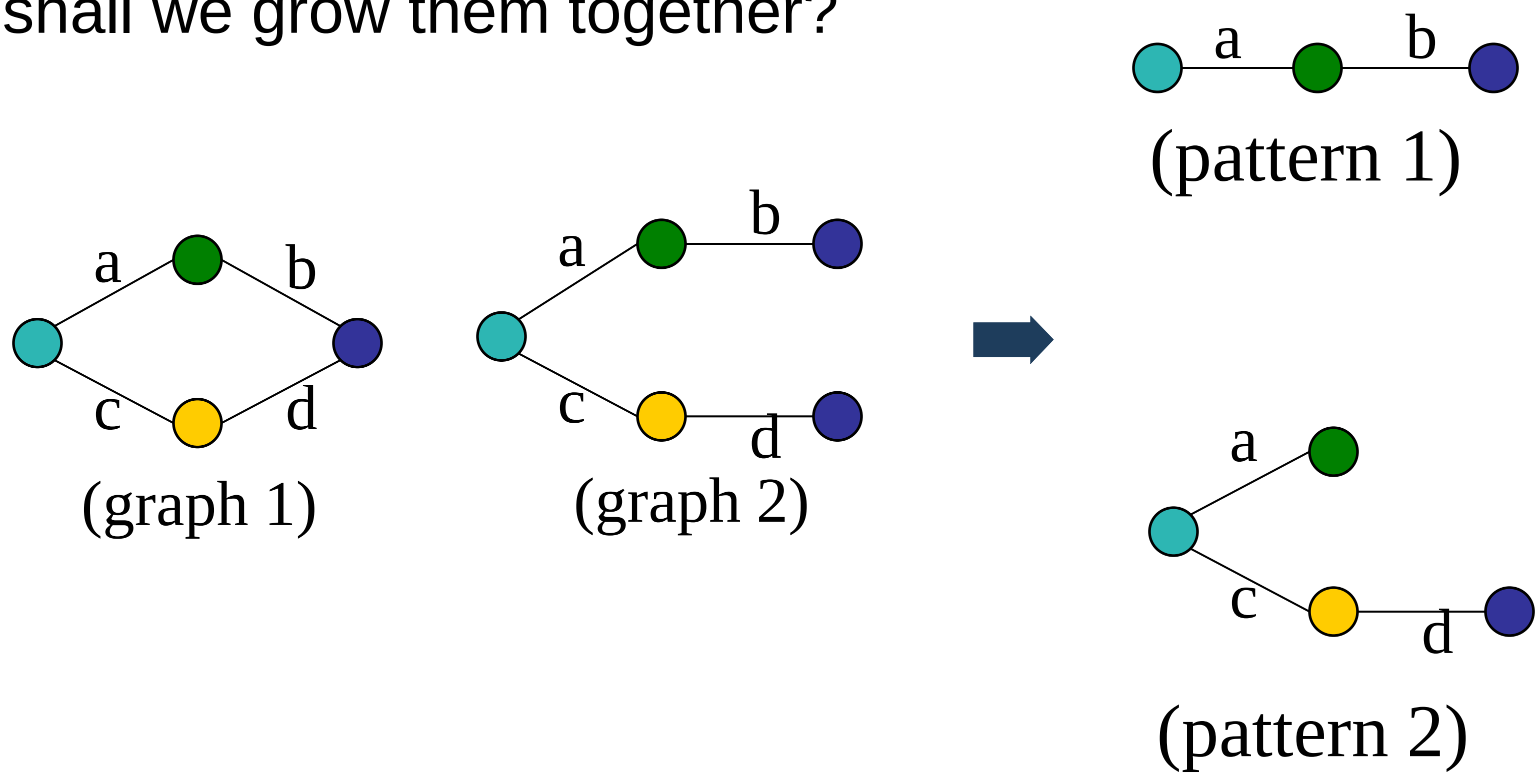


At what condition, can we stop searching their supergraph i.e., early termination?

If G and G' are frequent, G is a subgraph of G'. If **in any part of graphs in the dataset where G occurs, G' also occurs**, then we need not grow G, since none of G's supergraphs will be closed except those of G'.

Handling Tricky Cases

Edges a and b are always together, shall we grow them together?



Graph Pattern with Other Measures

Let p and q be the frequency of g in positive and negative graph datasets,

(1) Contrast: p/q,

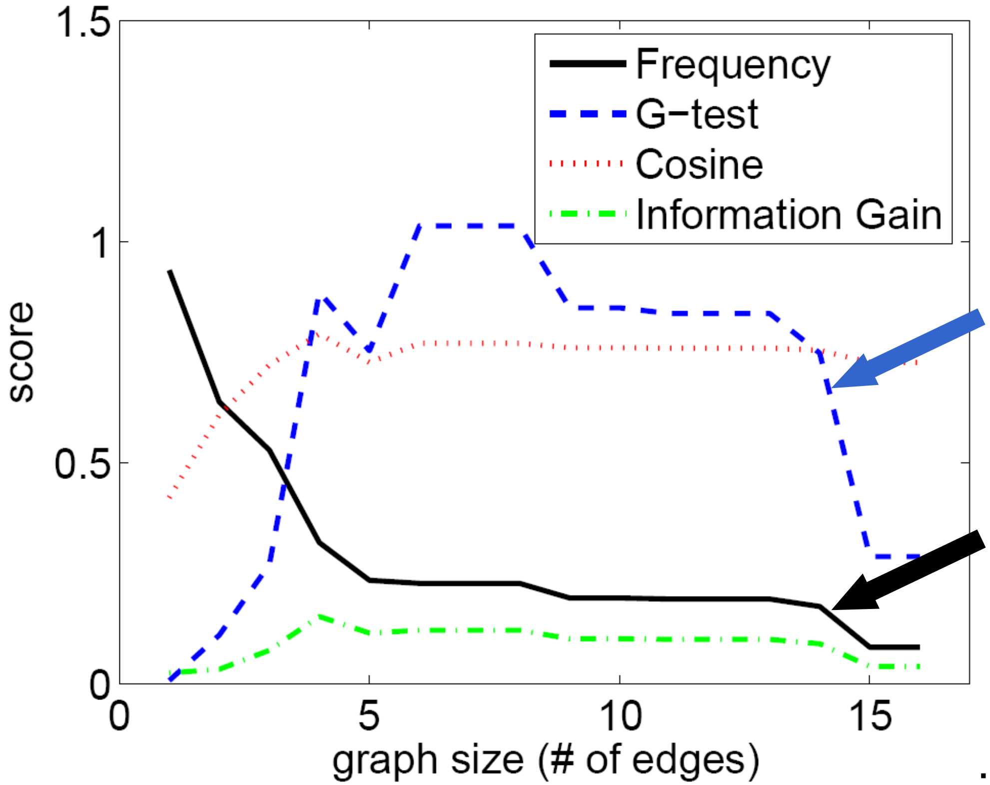
(2) G-test: $p \cdot ln\frac{p}{q} + (1-p) \cdot ln\frac{1-p}{1-q}$,

(3) Information Gain: $H(C) - H(C|X)$

(4) Cosine

(5) many others.

Challenge: Non Anti-Monotonic



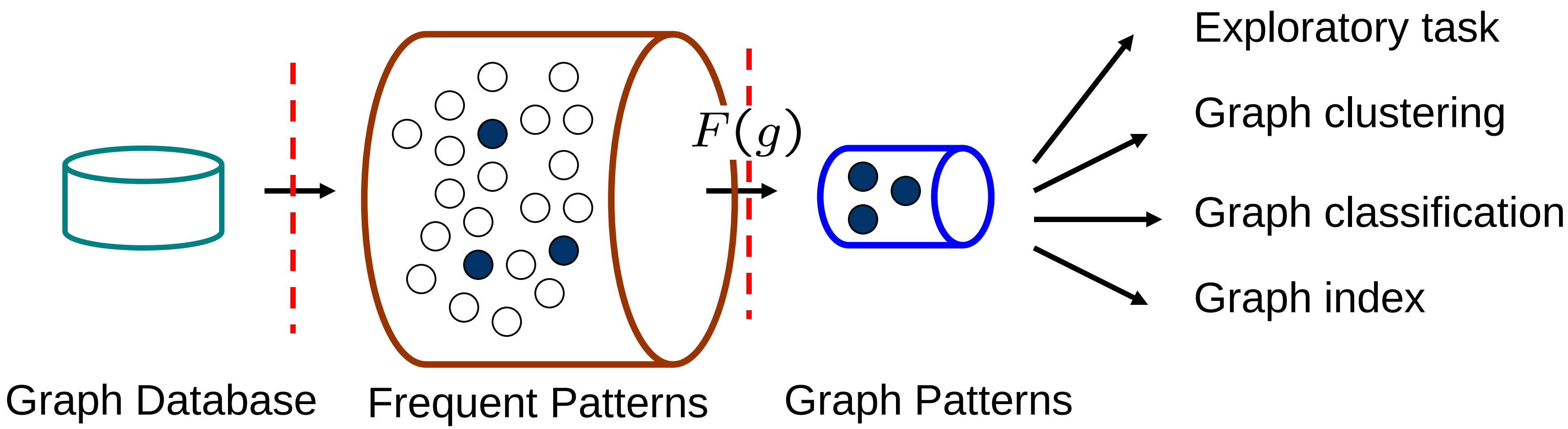
Non Monotonic

Anti-Monotonic

Enumerate subgraphs
: small-size to large-size

Non-Monotonic: Enumerate all subgraphs, then check their score?

Frequent Pattern Based Mining Framework



1. Bottleneck : millions, even billions of patterns

2. No guarantee of quality

Optimal Graph Pattern

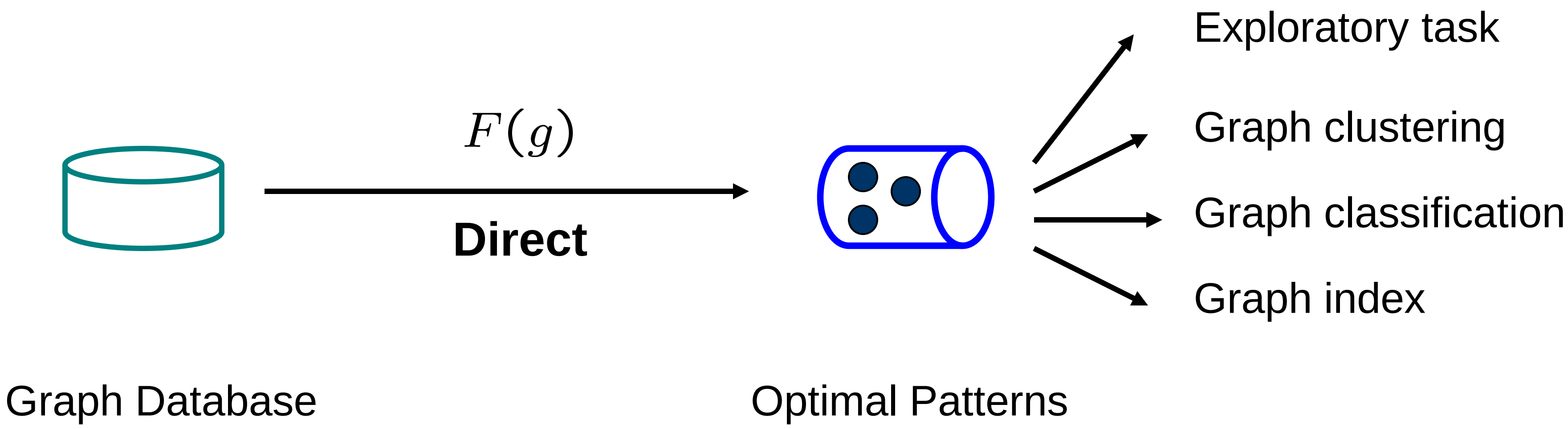
Given a graph dataset D and an objective function $F(g)$, find a graph pattern g^*, s.t.

$$g^* = arg\ max_g\ F(g).$$

Extension:

- Top-K Optimal Graph Patterns
- Redundancy-aware Graph Patterns
- Discriminative Patterns for Classification

Direct Pattern Mining Framework



Upper-Bound

Idea: derive an upper bound, $\widehat{F}(g)$, s.t., $\widehat{F}(g)$ is monotonic to freq(g).

$$G_t(p,q) = p \cdot ln\frac{p}{q} + (1-p) \cdot ln\frac{1-p}{1-q},$$

$$\begin{aligned} \frac{\partial G_t}{\partial q} &= \frac{q-p}{(1-q)q}, \\ \frac{\partial G_t}{\partial p} &= ln\frac{p(1-q)}{q(1-p)}. \end{aligned}$$

Since $\frac{p(1-q)}{q(1-p)} < 1$ when $p < q$, hence,

$$\text{if } p > q, \frac{\partial G_t}{\partial p} > 0, \frac{\partial G_t}{\partial q} < 0, \quad (1)$$

$$\text{if } p < q, \frac{\partial G_t}{\partial p} < 0, \frac{\partial G_t}{\partial q} > 0. \quad (2)$$

Upper-Bound: Anti-Monotonic (cont.)

$$\text{if } p > q, \frac{\partial G_t}{\partial p} > 0, \frac{\partial G_t}{\partial q} < 0, \tag{1}$$

$$\text{if } p < q, \frac{\partial G_t}{\partial p} < 0, \frac{\partial G_t}{\partial q} > 0. \tag{2}$$

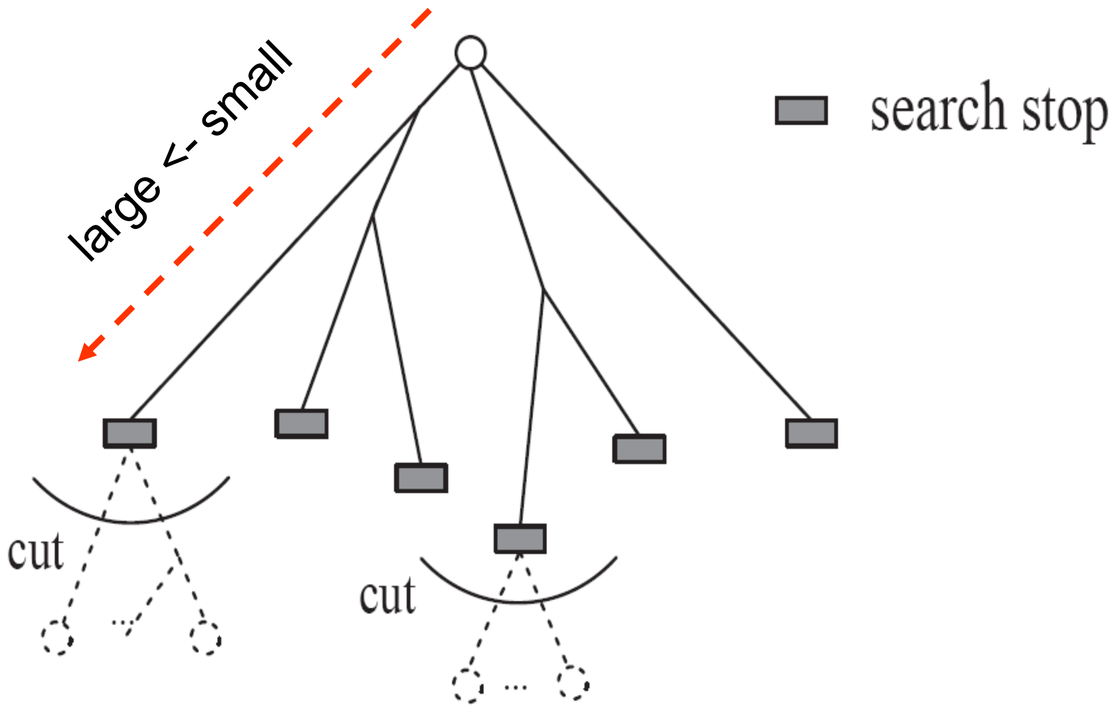
Rule of Thumb :
If the frequency difference of a graph pattern in the positive dataset and the negative dataset increases, the pattern becomes more interesting

small number

$$F(g) = F(p,q) < \max(\underbrace{F(p,\epsilon)}_{\text{Monotonic to p}}, \underbrace{F(\epsilon,q)}_{\text{Monotonic to q}}).$$

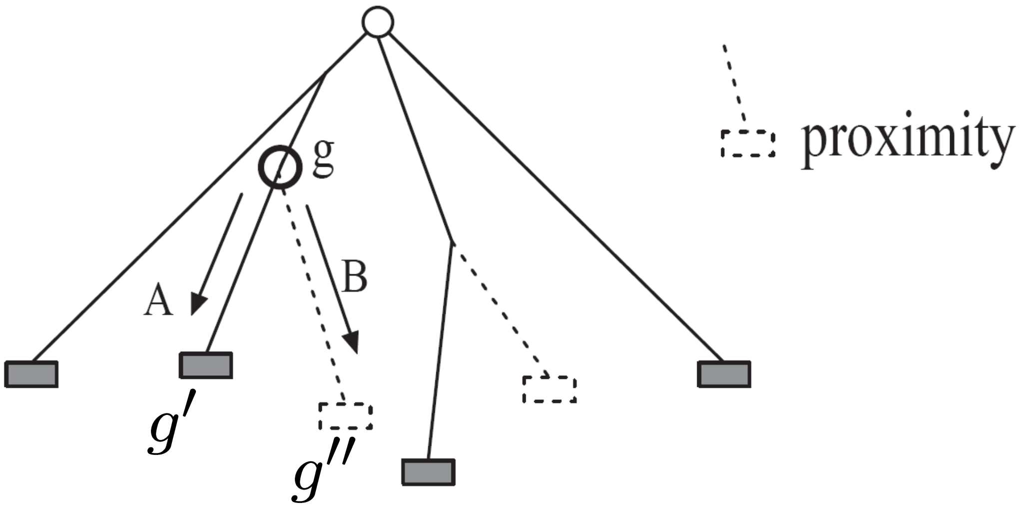
We can recycle the existing graph mining algorithms to accommodate non-monotonic functions.

Vertical Pruning



$$\max(F(p,\epsilon), F(\epsilon,q)) < F(g^*).$$

Horizontal Pruning: Structural Proximity



$$g' \sim g'' \Rightarrow F(g') \sim F(g'').$$

$$F(g') \ll F(g^*) \Rightarrow F(g'') \ll F(g^*).$$

Graph Pattern with Topological Constraints

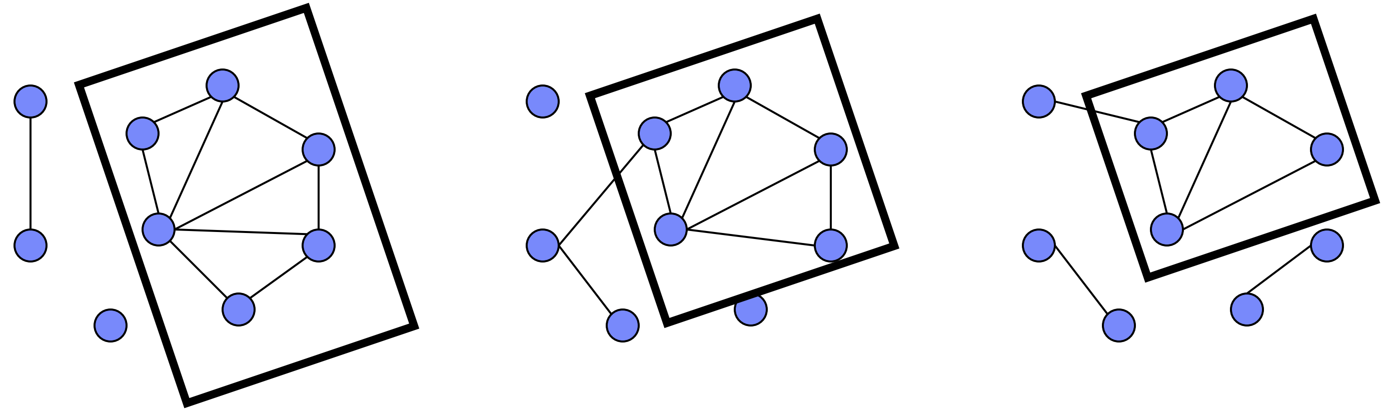
A constraint C is a boolean predicate, $C : P \rightarrow \{0, 1\}$, which maps a pattern α to a Boolean value. A pattern α satisfies constraint C if $C(\alpha) = 1$.

graph constraints

- Degree
- Size
- Density
- Density ratio
- Diameter
- Edge connectivity
- Vertex connectivity
- Aggregation (min, max, avg)

Constraint-Based Graph Pattern Mining

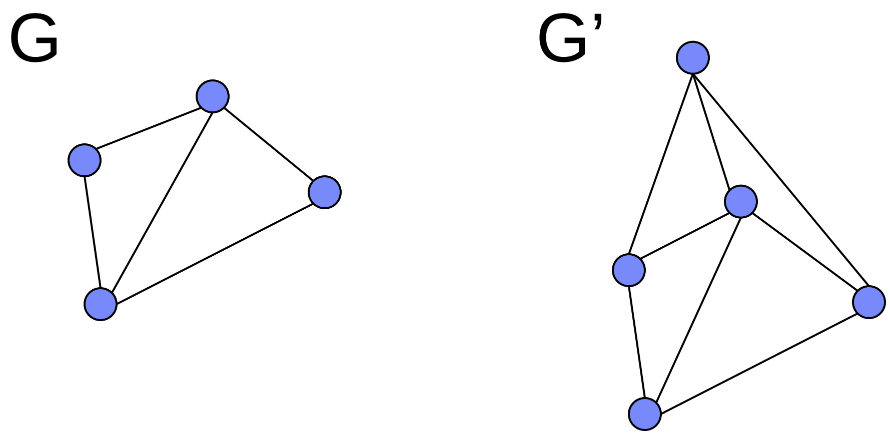
- Highly connected subgraphs in a large graph usually are not artifacts (group, functionality)



- Recurrent patterns discovered in multiple graphs are more robust than the patterns mined from a single graph

No Downward Closure Property

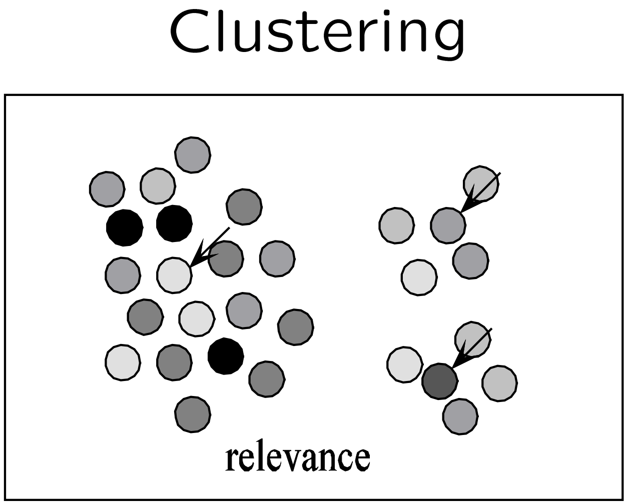
Given two graphs G and G', if G is a subgraph of G', it does not imply that the connectivity of G' is less than that of G, and vice versa.

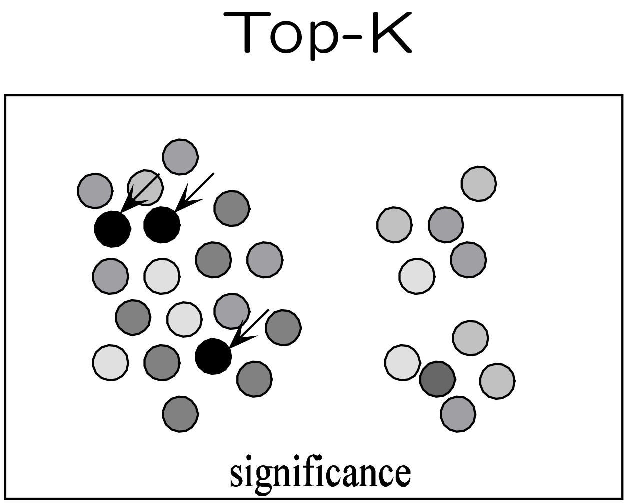


Pattern Summarization (Xin et al., KDD’06, Chen et al. CIKM’08)

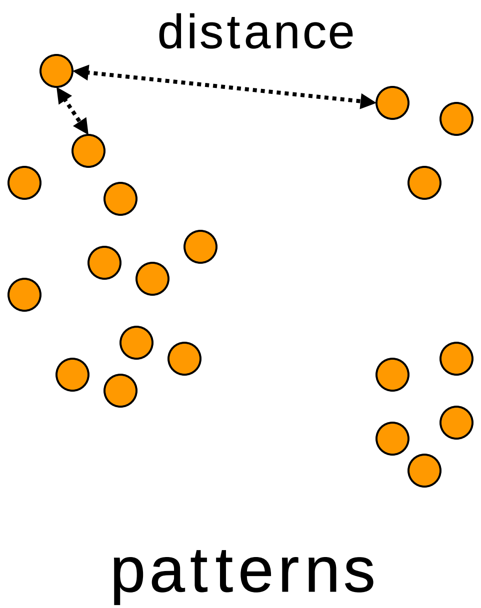
- Too many patterns may not lead to more explicit knowledge
- It can confuse users as well as further discovery (e.g., clustering, classification, indexing, etc.)
- A small set of “representative” patterns that preserve most of the information

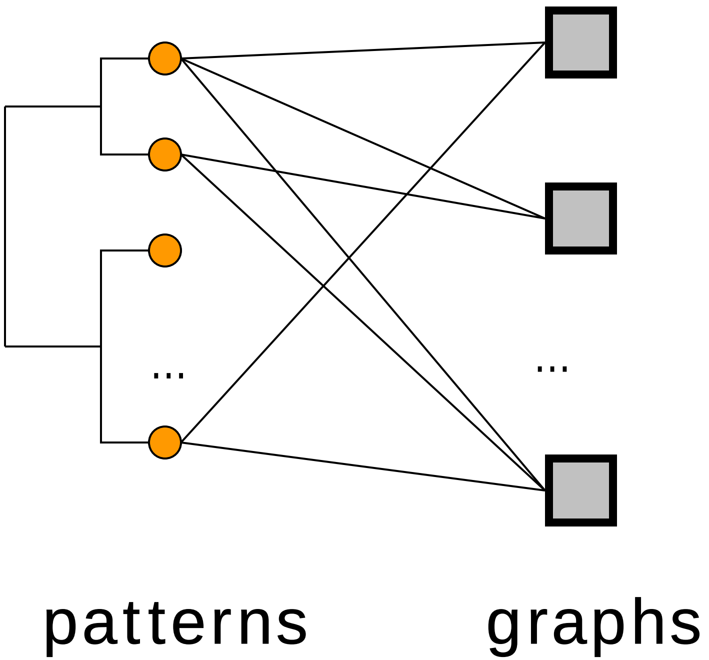
Pattern Summarization (Xin et al., KDD'06, Chen et al. CIKM'08)





Pattern Distance





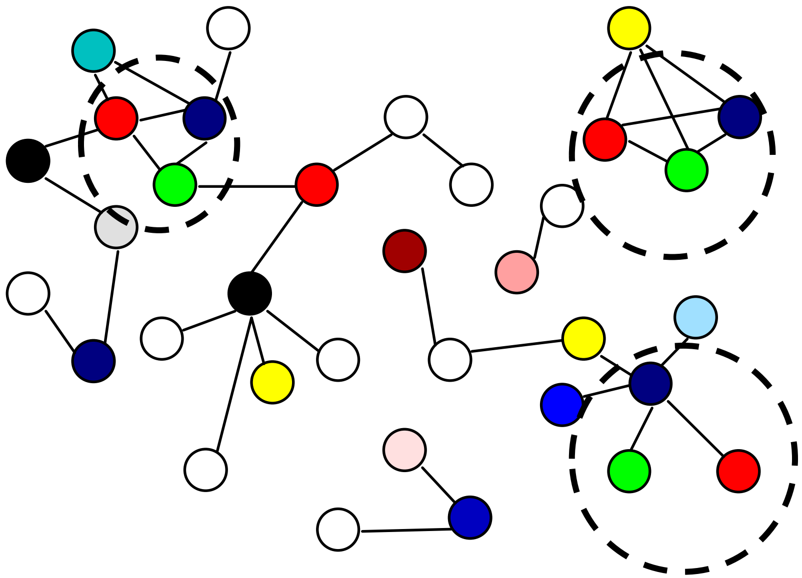
measure 1: pattern based

- pattern containment
- pattern similarity

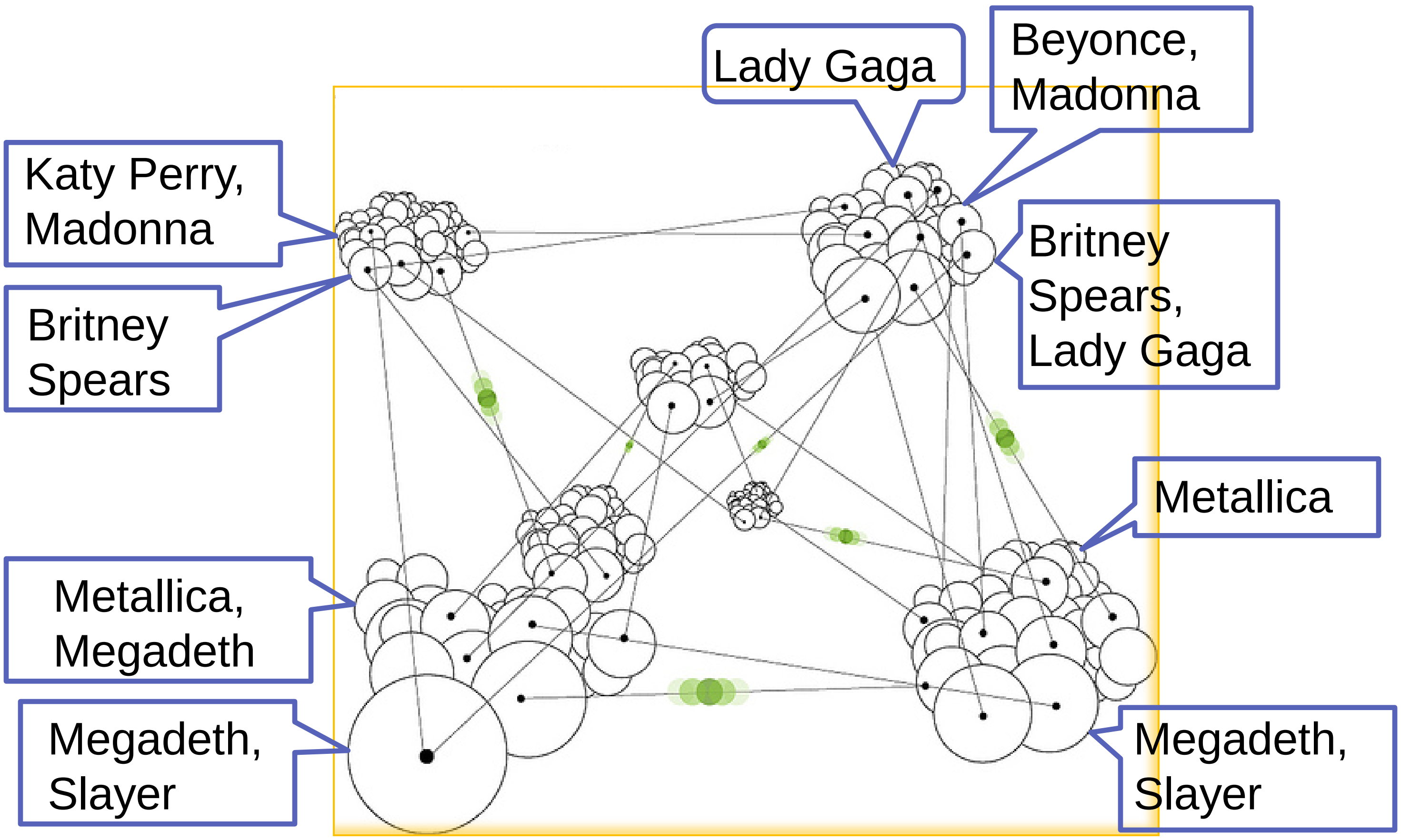
measure 2: data based

- data similarity

Graph Patterns in Social Network



What is the appropriate definition of graph patterns in social networks?



Homophily in Social Network

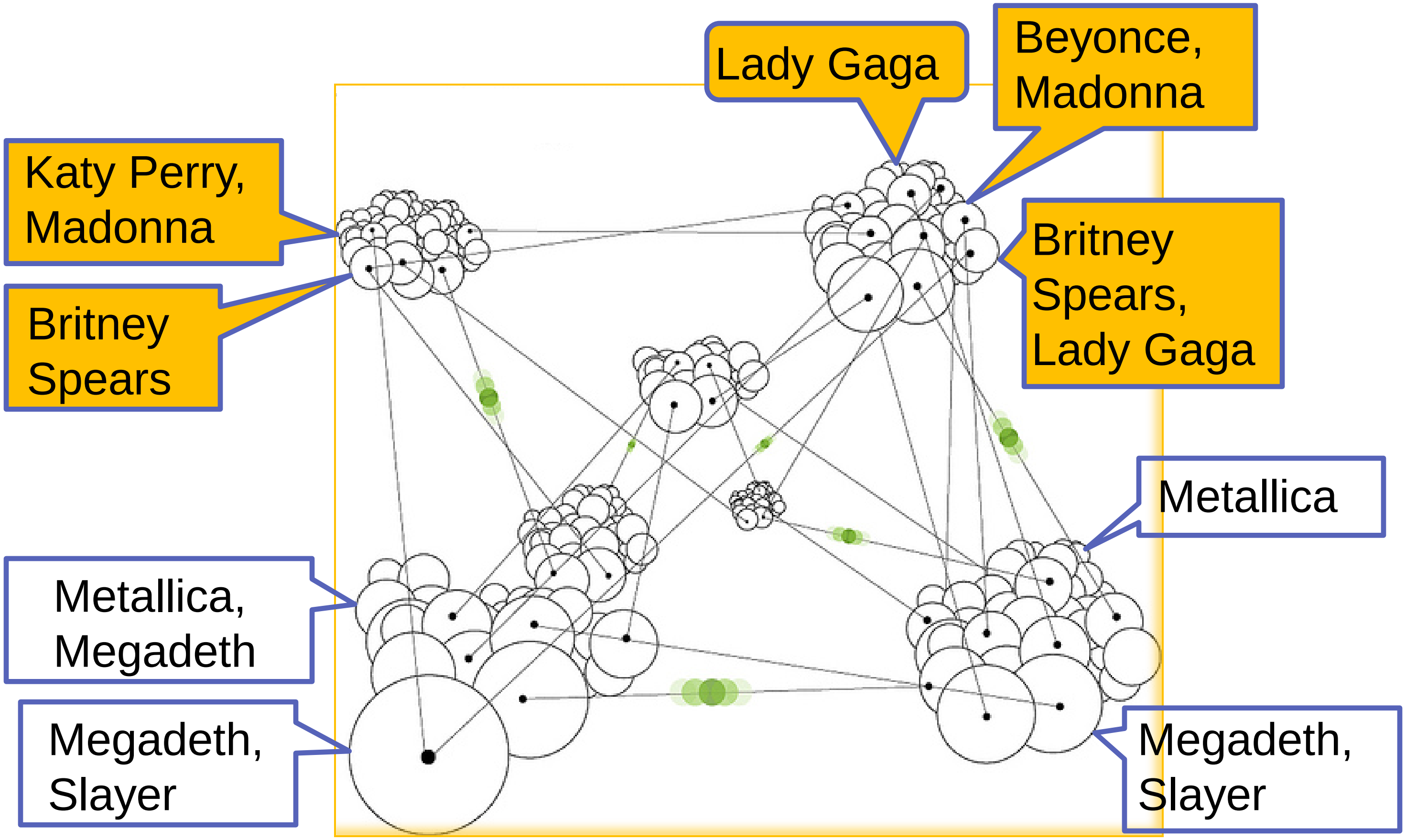
Last.FM

Nodes -> Users

Edges -> Links

List of Musical Bands/ Singers

What are the **related Musical Bands/ Singers** that **co-occur frequently in neighborhood?**



Homophily in Social Network

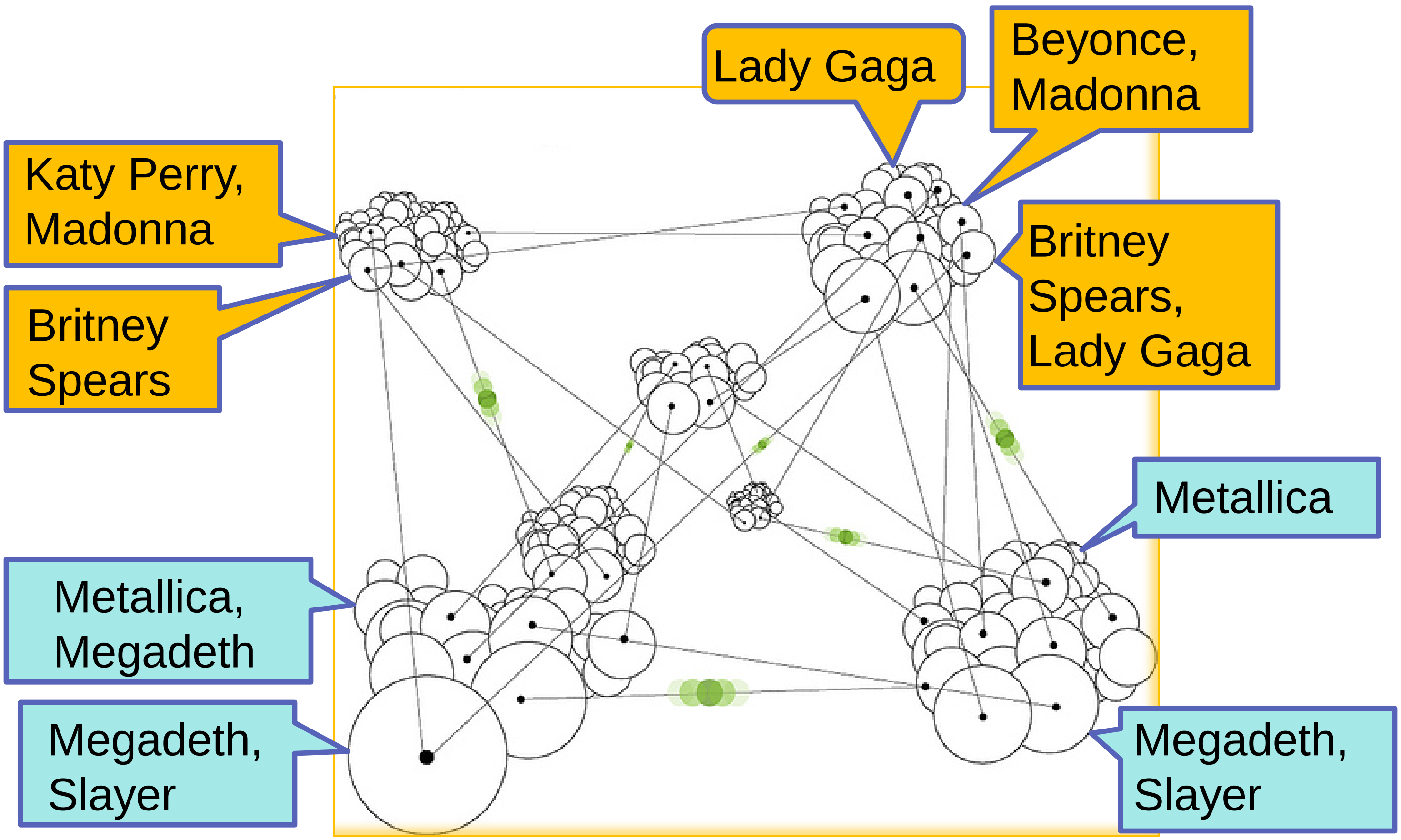
Last.FM

Nodes -> Users

Edges -> Links

List of Musical Bands/ Singers

What are the **related Musical Bands/ Singers** that **co-occur frequently in neighborhood**?



Homophily in Social Network

Last.FM

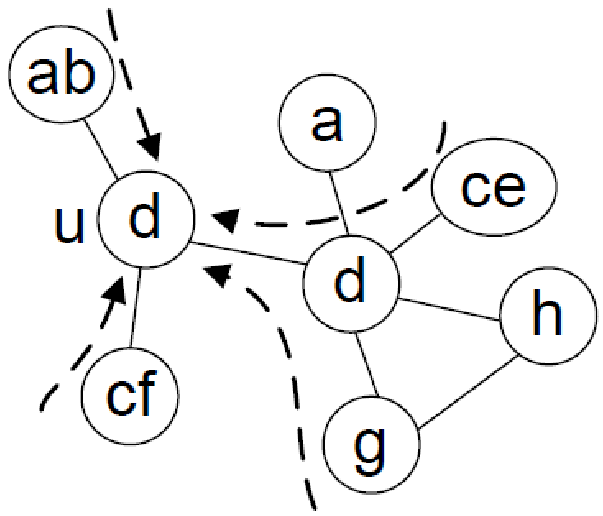
Nodes -> Users

Edges -> Links

List of Musical Bands/ Singers

What are the **related Musical Bands/ Singers** that **co-occur frequently in neighborhood?**

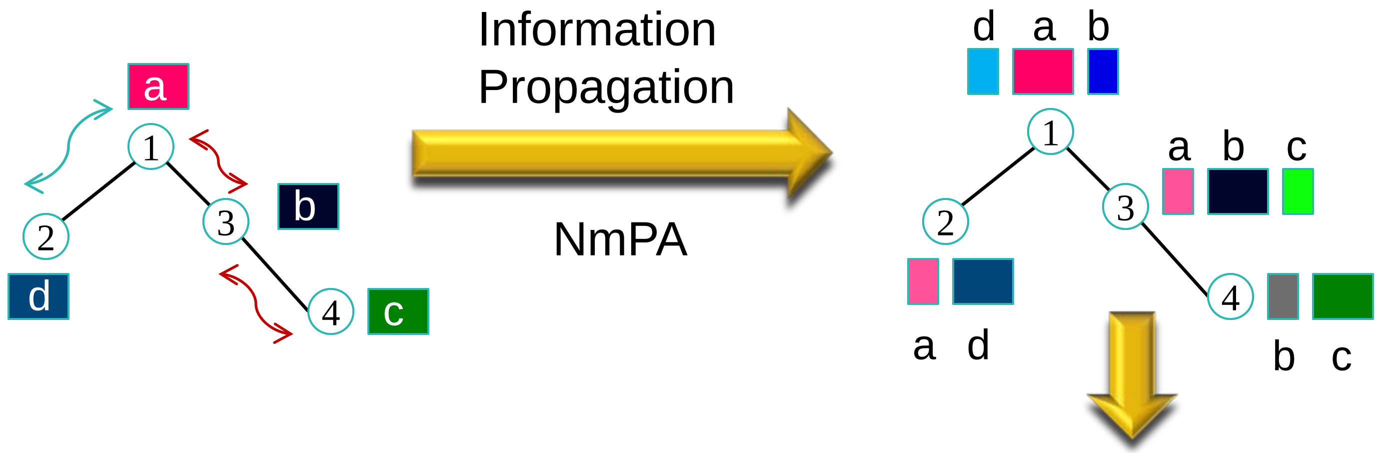
Information Propagation Model



node u

a	b	c	d	…
0.5	0.3	0.3	1.5	…

Probabilistic Itemset Mining



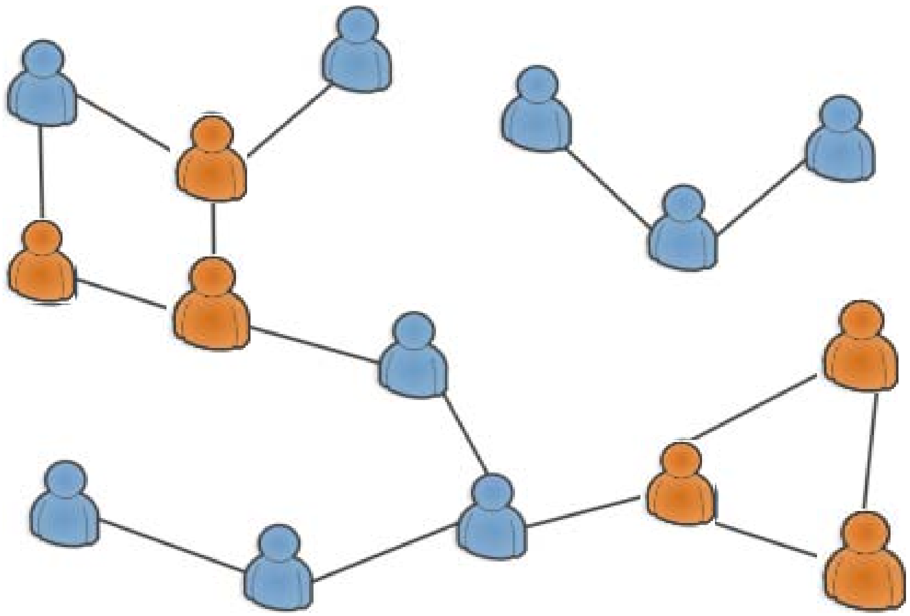
- **Frequent-Pattern (FP) Tree** cannot handle fractional association values because of the new definition of Support.
- Modify FP Tree Structure and Algorithm.
- *C. C. Aggarwal et. al (KDD ’09), Bernecker et. al (KDD ‘09).*

	a	b	c	d
1	1.00	0.12	0.00	0.12
2	0.19	0.00	0.00	1.00
3	0.12	1.00	0.12	0.00
4	0.00	0.19	1.00	0.00

Frequent Itemset Mining (Probabilistic)

Correlation and Anomaly in Graphs

Example of Correlations



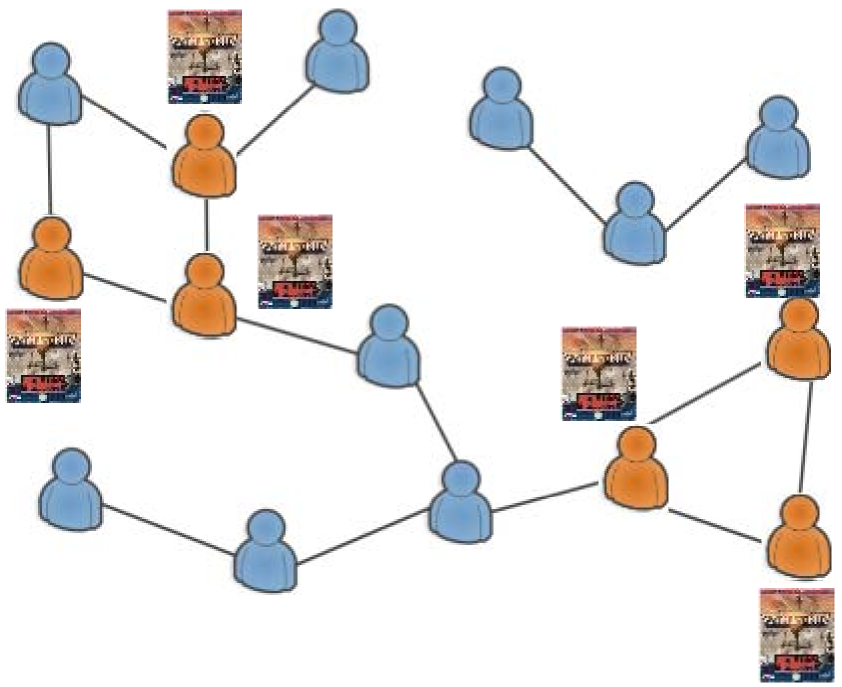
Correlation between the occurrence of an event and the network structure

Pattern Kaleidoscope

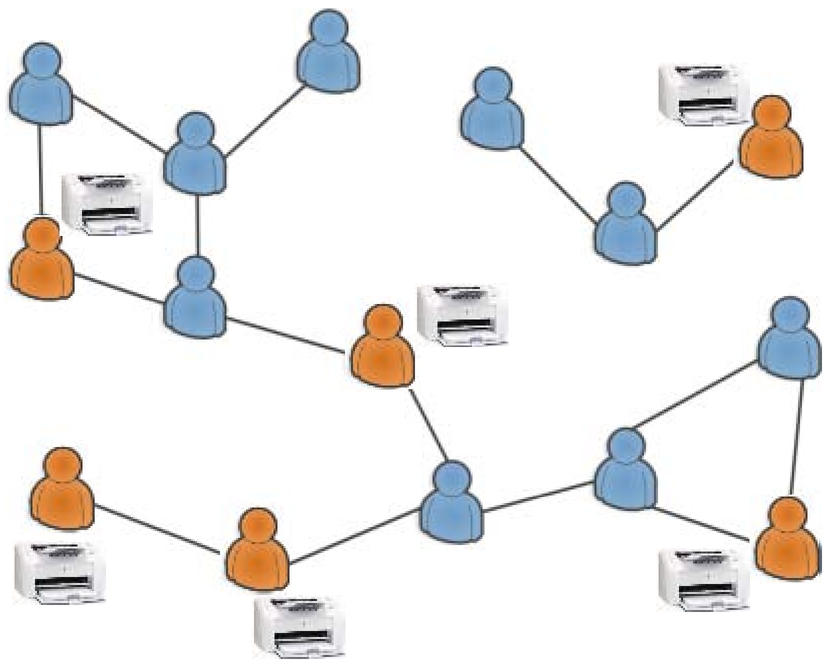
- Frequent Graph Pattern
- Proximity Pattern
- Attribute-Structure Correlations ✔
- Cohesive Pattern
- Itemset-sharing Pattern
- Graph Topological pattern
- Graph Iceberg ✔
- Graph Anomaly → **Akoglu et al., Tutorial at WSDM'13**

Structural Correlational Pattern [Guan et al.,SIGMOD'11]

Which product's sales is more correlated with the social network structure?



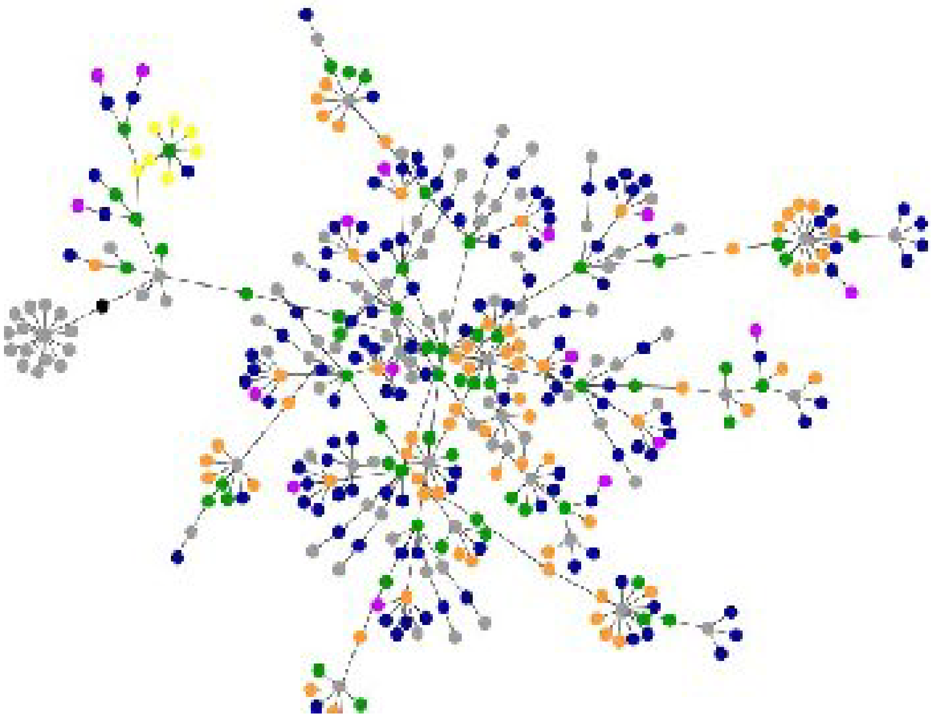
Waterworld game



HP printer

A General Situation

- Events taking place on nodes of a social graph
  - Online shopping
  - Blogging
  - Virus infection
- Social influence vs. Random occurrence

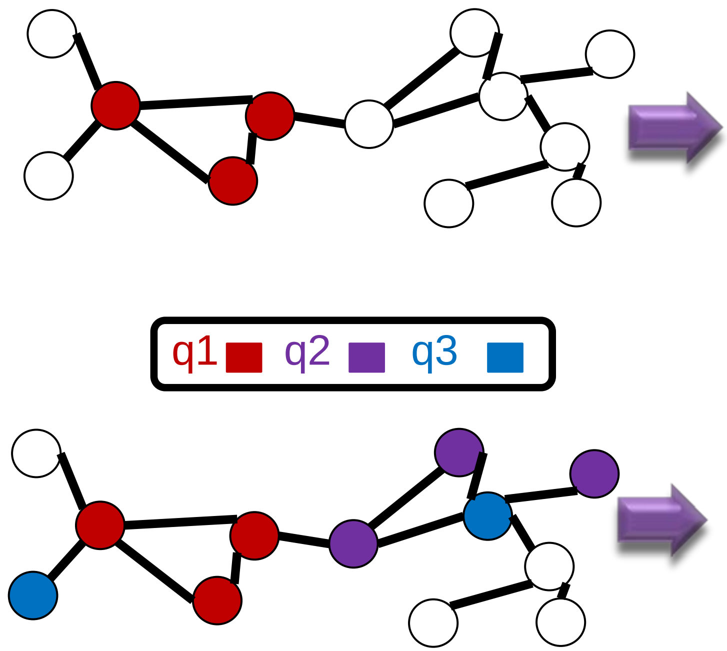


Problem Formulation

- A graph $G = (V, E)$ and an event set $Q = \{q_i\}$
- V_q--the set of nodes having event q. Let $|V_q| = m$, $|V| = n$

(1)

(2)

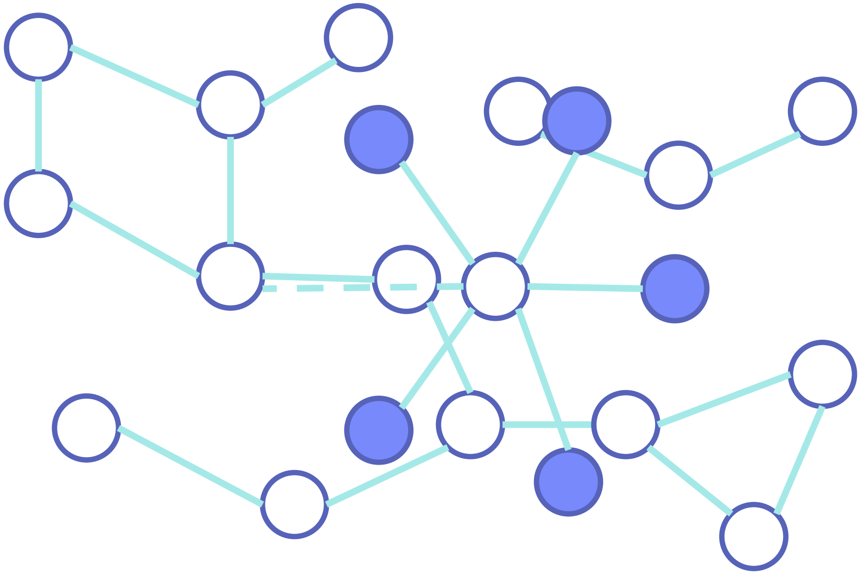


Is q1 correlated in G?

(1) q1
(2) q2 (ranking)
(3) q3

How to Characterize Correlation?

- If correlated, blue nodes tend to stick together.
- A naïve approach: only look at neighborhood
- General idea: compute the **aggregated proximity** among blue nodes



Measure Definition

- The measure is defined as

$$\rho(V_q) = \frac{\sum_{v \in V_q} s(v, V_q \setminus \{v\})}{|V_q|}$$

V_q: the set of nodes having event q; $s(\cdot)$ can be any graph proximity measure, e.g. hitting time.

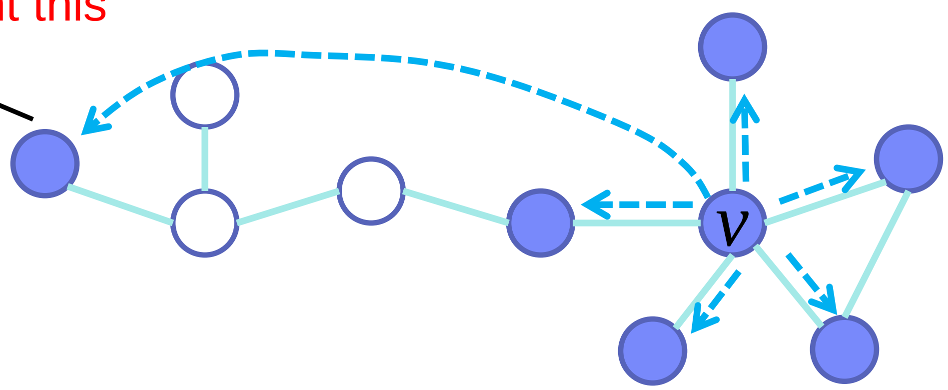
Measure Definition

- **Hitting time**: expected number of steps to reach a target node via random walk:

$$h(v_i, B) = \sum_{t=1}^{\infty} t \Pr(T_B = t \mid x_0 = v_i)$$

B: target node set; $\Pr(T_B=t|x_0=v_i)$: the probability that we start from v_i and reach B after t steps

Hitting time will not count this node



Hitting time & Decayed Hitting Time

- **Hitting time**: expected number of steps to reach a target node via random walk:

$$h(v_i, B) = \sum_{t=1}^{\infty} t \Pr(T_B = t \mid x_0 = v_i)$$

  - B: target node set; $\Pr(T_B=t|x_0=v_i)$: the probability that we start from v_i and reach B after t steps

Hitting time & Decayed Hitting Time

- **Hitting time**: expected number of steps to reach a target node via random walk:

$$h(v_i, B) = \sum_{t=1}^{\infty} t \Pr(T_B = t \mid x_0 = v_i)$$

  - B: target node set; $\Pr(T_B = t | x_0 = v_i)$: the probability that we start from v_i and reach B after t steps

- **Decayed Hitting Time (DHT)**:

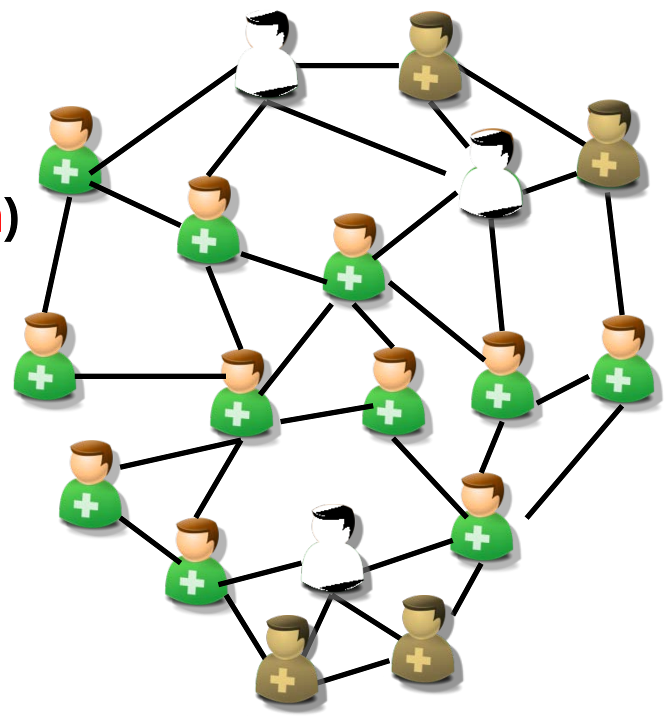
$$\tilde{h}(v_i, B) = \sum_{t=1}^{\infty} e^{-(t-1)} \Pr(T_B = t \mid x_0 = v_i)$$

  - Mapping [1,∞) to [0,1], high value means high proximity
  - Emphasizing the importance of local neighborhood and reducing the impact of long paths

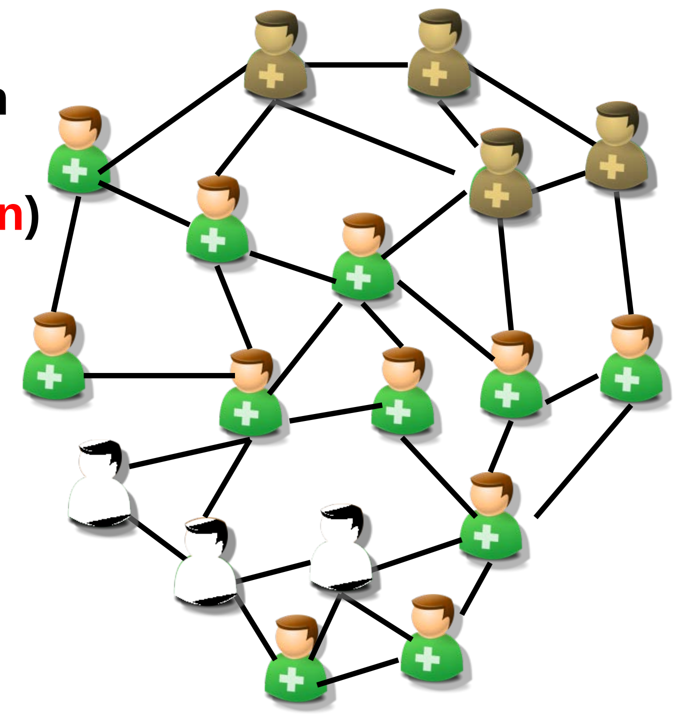
Two-event Structural Correlations

How is the relationship between the sales of two products in a social network?

Attraction (positive correlation)



Repulsion (negative correlation)



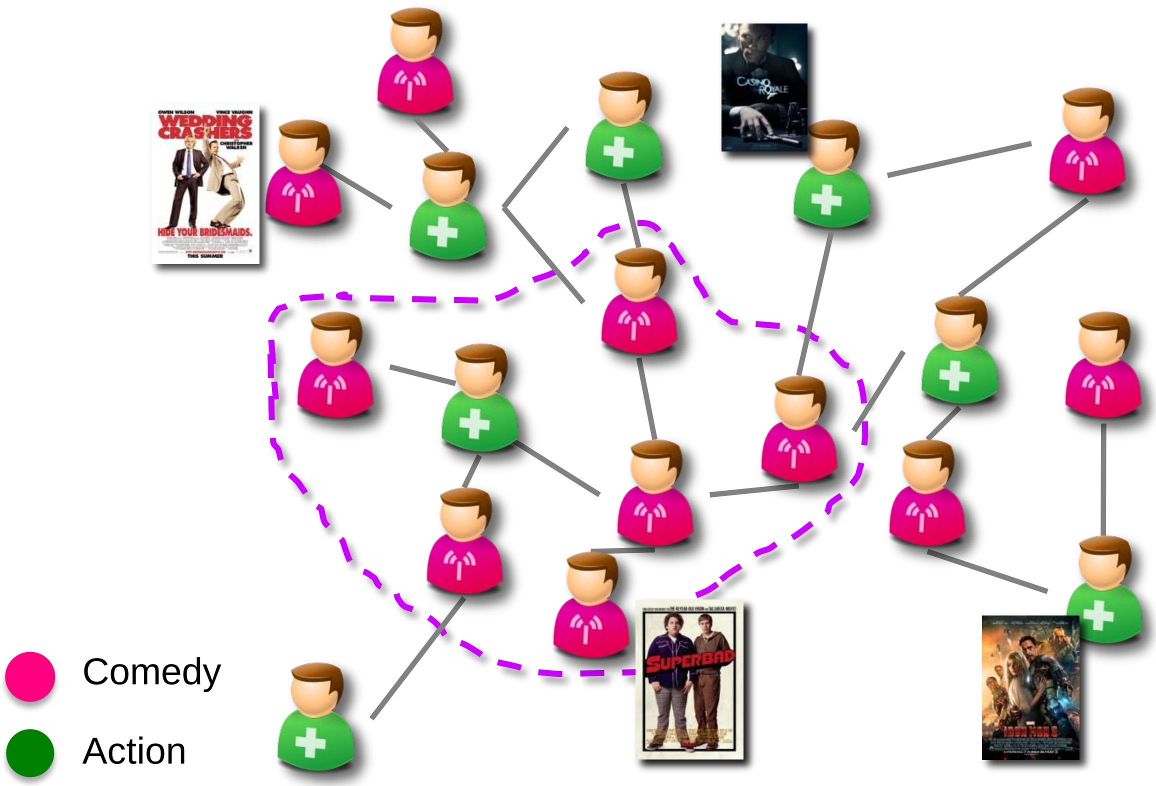
Video games

Computers

Anomaly Detection in Graphs

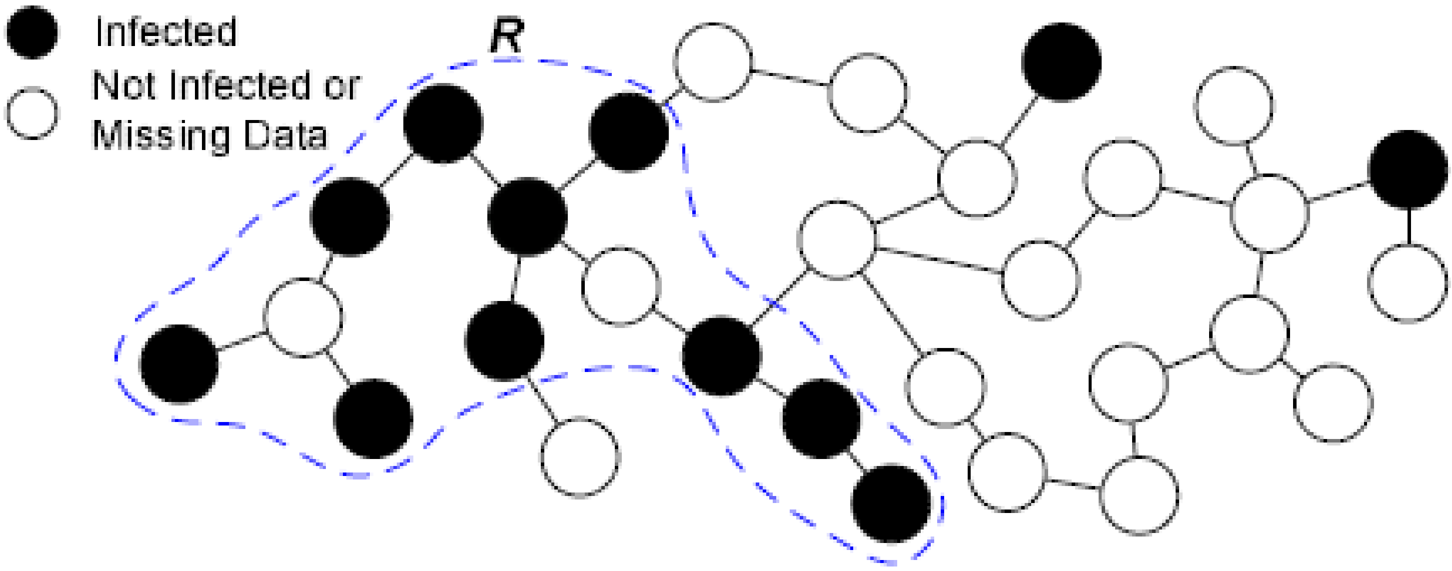
- Various Interesting-ness/Anomaly Criteria
  e.g.,
  - Bgp-lens: anomalies in internet routing updates.
    [Prakash et al., KDD'09]
  - Oddball: anomalies in weighted graphs.
    [Akoglu et al., PAKDD'10]
  - Heavy subgraphs in time-evolving networks.
    [Bogdanov et al., ICDM'11]
- Anomaly, Event, and Fraud Detection in Large Graph Datasets, Akoglu et al., http://www.cs.stonybrook.edu/~leman/wsdm13/

Anomaly Vertices/ Regions



1. Target marketing
2. Recommendation systems
3. Social influence analysis

Anomalous Regions (i.e., gAnomaly)



- Why does a disease occur more intensively in some portions of a network?
- Why do a subset of computers receive most of the attacks in the past day, and are they therefore targeted attacks?

References (1)

- T. Asai, et al. “Efficient substructure discovery from large semi-structured data”, SDM'02
- F. Afrati, A. Gionis,and H. Mannila, “Approximating a collection of frequent sets”, KDD’04
- C. Borgelt and M. R. Berthold, “Mining molecular fragments: Finding relevant substructures of molecules”, ICDM'02
- Y. Chi, Y. Xia, Y. Yang, R. Muntz, “Mining closed and maximal frequent subtrees from databases of labeled rooted trees,” TKDE 2005
- M. Deshpande, M. Kuramochi, and G. Karypis, “Frequent substructure based approaches for classifying chemical compounds”, ICDM’03
- M. Deshpande, M. Kuramochi, and G. Karypis. “Automated approaches for classifying structures”, BIOKDD'02
- L. Dehaspe, H. Toivonen, and R. King. “Finding frequent substructures in chemical compounds,” KDD'98
- C. Faloutsos, K. McCurley, and A. Tomkins, “Fast discovery of connection subgraphs”, KDD'04
- W. Fan, K. Zhang, H. Cheng, J. Gao, X. Yan, J. Han, P. S. Yu, O. Verscheure, “Direct mining of discriminative and essential graphical and itemset features via model-based search tree,” KDD'08
- H. Fröhlich, J. Wegner, F. Sieker, and A. Zell, “Optimal assignment kernels for attributed molecular graphs”, ICML’05
- T. Gärtner, P. Flach, and S. Wrobel, “On graph kernels: Hardness results and efficient alternatives”, COLT/Kernel’03

References (2)

- L. Holder, D. Cook, and S. Djoko, “Substructure discovery in the subdue system”, KDD'94
- T. Horváth, J. Ramon, and S. Wrobel, “Frequent subgraph mining in outerplanar graphs,” KDD’06
- J. Huan, W. Wang, D. Bandyopadhyay, J. Snoeyink, J. Prins, and A. Tropsha. “Mining spatial motifs from protein structure graphs”, RECOMB’04
- J. Huan, W. Wang, and J. Prins, “Efficient mining of frequent subgraph in the presence of isomorphism”, ICDM'03
- J. Huan, W. Wang, and J. Prins, and J. Yang, “SPIN: Mining maximal frequent subgraphs from graph databases”, KDD’04
- A. Inokuchi, T. Washio, and H. Motoda. “An apriori-based algorithm for mining frequent substructures from graph data”, PKDD'00
- H. Kashima, K. Tsuda, and A. Inokuchi, “Marginalized kernels between labeled graphs”, ICML’03
- B. Kelley, R. Sharan, R. Karp, E. Sittler, D. Root, B. Stockwell, and T. Ideker, “Conserved pathways within bacteria and yeast as revealed by global protein network alignment,” PNAS, 2003
- R. King, A Srinivasan, and L Dehaspe, "Warmr: a data mining tool for chemical data," J Comput Aided Mol Des 2001

References (3)

- M. Koyuturk, A. Grama, and W. Szpankowski. “An efficient algorithm for detecting frequent subgraphs in biological networks”, Bioinformatics, 20:I200--I207, 2004
- C. Liu, X. Yan, H. Yu, J. Han, and P. S. Yu, “Mining behavior graphs for ‘backtrace'' of noncrashing bugs,'' SDM'05
- T. Kudo, E. Maeda, and Y. Matsumoto, “An application of boosting to graph classification”, NIPS’04
- M. Kuramochi and G. Karypis. “Frequent subgraph discovery”, ICDM'01
- M. Kuramochi and G. Karypis, “GREW: A scalable frequent subgraph discovery algorithm”, ICDM’04
- P. Mahé, N. Ueda, T. Akutsu, J. Perret, and J. Vert, “Extensions of garginalized graph kernels”, ICML’04
- B. McKay. Practical graph isomorphism. Congressus Numerantium, 30:45--87, 1981.
- S. Nijssen and J. Kok, “A quickstart in frequent structure mining can make a difference,” KDD'04
- R. Sharan, S. Suthram, R. Kelley, T. Kuhn, S. McCuine, P. Uetz, T. Sittler, R. Karp, and T. Ideker, “Conserved patterns of protein interaction in multiple species,” PNAS, 2005
- J. R. Ullmann. “An algorithm for subgraph isomorphism”, J. ACM, 23:31--42, 1976.
- N. Vanetik, E. Gudes, and S. E. Shimony. “Computing frequent graph patterns from semistructured data”, ICDM'02
- K. Tsuda, “Entire regularization paths for graph data,” ICML’07

References (4)

- N. Wale and G. Karypis, “Acyclic subgraph based descriptor spaces for chemical compound retrieval and classification”, Univ. of Minnesota, Technical Report: #06–008
- C. Wang, W. Wang, J. Pei, Y. Zhu, and B. Shi. “Scalable mining of large disk-base graph databases”, KDD'04
- T. Washio and H. Motoda, “State of the art of graph-based data mining,” SIGKDD Explorations, 5:59-68, 2003
- M. Wörlein, T. Meinl, I. Fischer, M. Philippsen, “A quantitative comparison of the subgraph miners MoFa, gSpan, FFSM, and Gaston,” PKDD’05
- X. Yan, H. Cheng, J. Han, and P. S. Yu, “Mining significant graph patterns by leap search,” SIGMOD'08
- X. Yan and J. Han, “gSpan: Graph-based substructure pattern mining”, ICDM'02
- X. Yan and J. Han, “CloseGraph: Mining closed frequent graph patterns”, KDD'03
- X. Yan, X. Zhou, and J. Han, “Mining closed relational graphs with connectivity constraints”, KDD'05
- X. Yan et al. “A graph-based approach to systematically reconstruct human transcriptional regulatory modules,” ISMB’07
- M. Zaki. “Efficiently mining frequent trees in a forest”, KDD'02
- Z. Zeng, J. Wang, L. Zhou, G. Karypis, "Coherent closed quasi-clique discovery from large dense graph databases," KDD'06

References (5)

- Towards Proximity Pattern Mining in Large Graphs. [Khan et al., SIGMOD'10]
- Assessing and ranking structural correlations in graphs. [Guan et al., SIGMOD'11]
- Measuring Two-Event Structural Correlations on Graphs. [Guan et al., VLDB'11]
- Mining Attribute-structure Correlated Patterns in Large Attributed Graphs. [Silva et al., VLDB'12]
- Mining Cohesive Patterns from Graphs with Feature Vectors. [Moser et al., SDM'09]
- Finding Itemset-Sharing Patterns in a Large Itemset-Associated Graph. [Fukuzaki et al., PAKDD'10]
- Mining graph topological patterns: Finding covariations among vertex descriptors. [Prado et al., TKDE'13]
- Bgp-lens: anomalies in internet routing updates. [Prakash et al., KDD'09]
- Oddball: Anomalies in Weighted Graphs. [Akoglu et al., PAKDD'10]
- Heavy Subgraphs in Time-Evolving Networks. [Bogdanov et al., ICDM'11]
- Giceberg: Towards Iceberg Analysis in Large Graphs. [Li et al., ICDE'13]
- A Probabilistic Approach to Uncovering Attributed Graph Anomalies. [Li et al., SDM'14]