Mining Frequent Subgraphs

CS 145

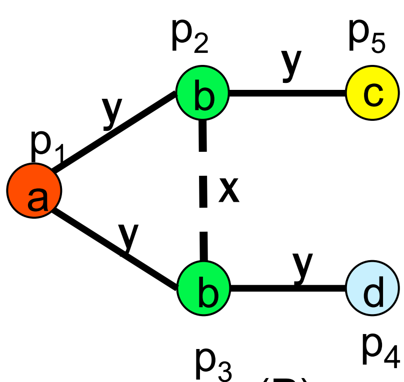
Fall 2015

FFSM: Fast Frequent Subgraph Mining -- An Overview:

- How to solve graph isomorphism problem?
  - A Novel Graph Canonical Form: CAM
- How to tackle subgraph isomorphism problem (NP-complete)?
  - Incrementally maintained embeddings
- How to enumerate subgraphs:
  - An Efficient Data Structure: CAM Tree
  - Two Operations: CAM-join, CAM-extension.

Adjacency Matrix

- <u>Every diagonal entry of adjacency matrix *M* corresponds to a distinct vertex in *G* and is filled with the label of this vertex.</u>
- Every off-diagonal entry in the lower triangle part of *M*[1] corresponds to a pair of vertices in *G* and is filled with the label of the edge between the two vertices and zero if there is no edge.



(P)

a				
y	b			
y	x	b		
0	y	0	c	
0	0	y	0	d

M_1

a				
y	b			
y	x	b		
0	0	y	d	
0	y	0	0	c

M_2

b				
x	b			
y	0	d		
0	y	0	c	
y	y	0	0	a

M_3

[1]for an undirected graph, the upper triangle is always a mirror of the lower triangle

Code

- A *Code* of $n \times n$ adjacency matrix M is defined as sequence of lower triangular entries (including the diagonal entries) in the order: $M_{1,1}\ M_{2,1}\ M_{2,2}\ \ldots\ M_{n,1}\ M_{n,2}\ \ldots M_{n,n-1}\ M_{n,n}$

a				
y	b			
y	x	b		
0	y	0	c	
0	0	y	0	d

M_1

a				
y	b			
y	x	b		
0	0	y	d	
0	y	0	0	c

M_2

b				
x	b			
y	0	d		
0	y	0	c	
y	y	0	0	a

M_3

Code(M_1): aybyxb0y0c00y0d >
Code(M_2): aybyxb00yd0y00c >
Code(M_3): bxby0d0y0cyy00a

assuming a>b>c> ... >0

- The *Canonical Adjacency Matrix* is the one produces the maximal code, using lexicographic order.

MP Submatrix

- For an *m* × *m* matrix *A*, an *n* × *n matrix B* is *A*'s *maximal proper submatrix* (MP Submatrix), iff *B* is obtained by removing the last none-zero entry from *A*.

$$\begin{array}{|c|}\hline a \\ \hline \end{array}$$

M_1

$$\begin{array}{|c|c|}\hline a & \\ \hline y & b \\ \hline \end{array}$$

M_2

$$\begin{array}{|c|c|c|}\hline a & & \\ \hline y & b & \\ \hline y & 0 & b \\ \hline \end{array}$$

M_3

$$\begin{array}{|c|c|c|}\hline a & & \\ \hline y & b & \\ \hline y & x & b \\ \hline \end{array}$$

M_4

$$\begin{array}{|c|c|c|c|}\hline a & & & \\ \hline y & b & & \\ \hline y & x & b & \\ \hline 0 & y & 0 & c \\ \hline \end{array}$$

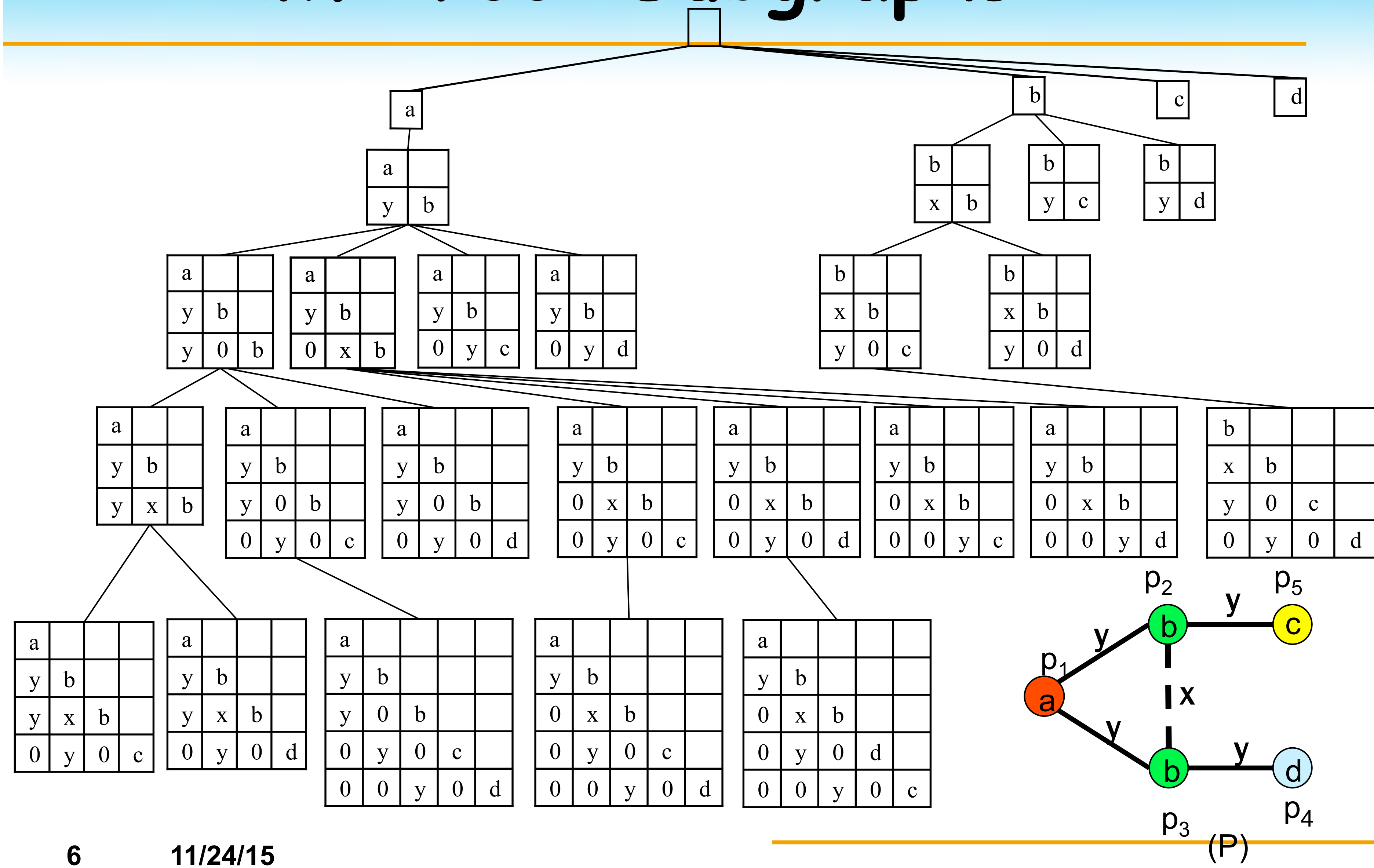
M_5

$$\begin{array}{|c|c|c|c|c|}\hline a & & & & \\ \hline y & b & & & \\ \hline y & x & b & & \\ \hline 0 & y & 0 & c & \\ \hline 0 & 0 & y & 0 & d \\ \hline \end{array}$$

M_6

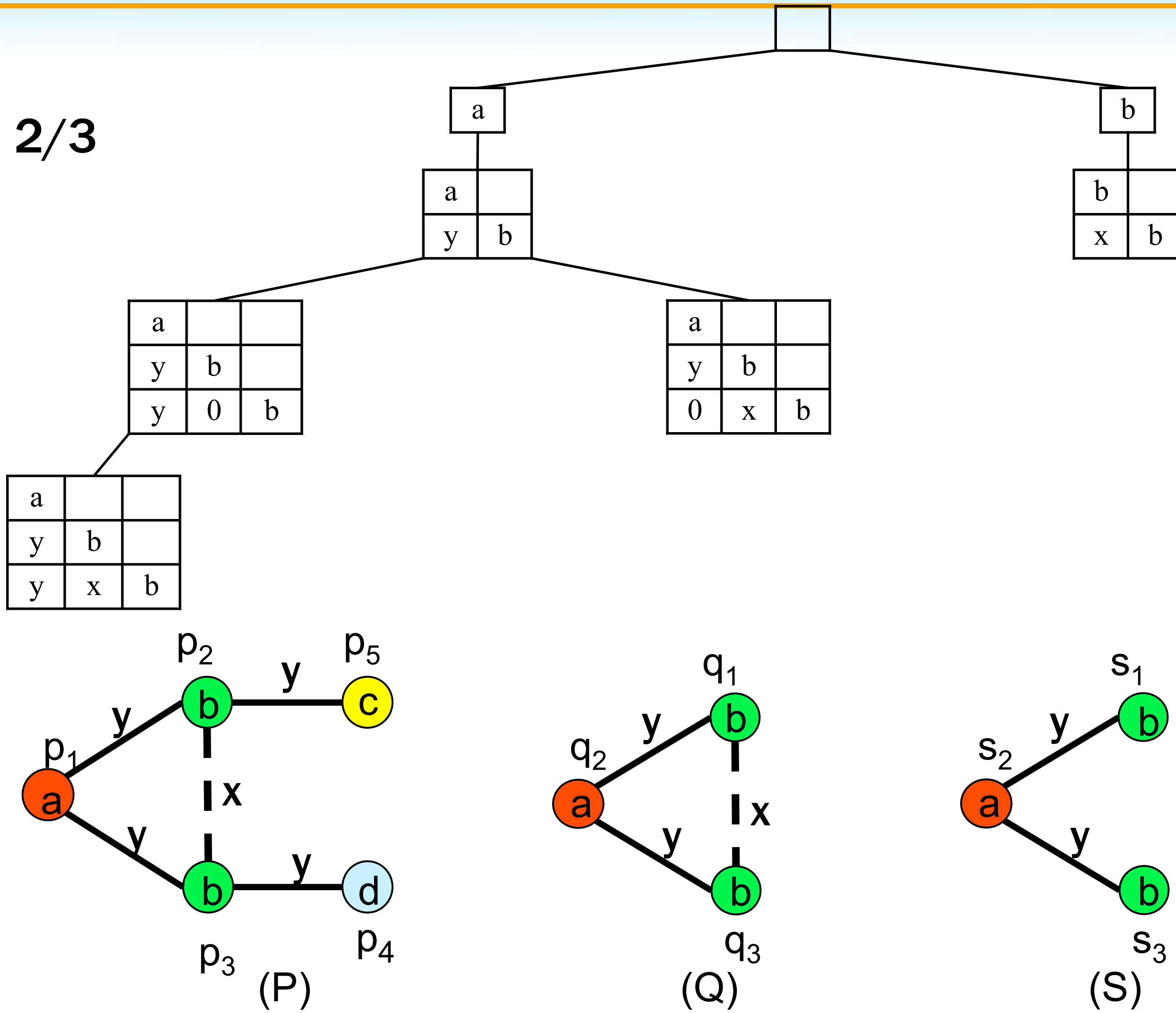
- We define a CAM is *connected* iff the corresponding graph is connected.
- Theorem I: A CAM's MP submatrix is CAM
- Theorem II: A connected CAM's MP submatrix is connected

CAM Tree: Subgraphs



CAM Tree: Frequent Subgraphs

$\sigma = 2/3$



How to Enumerate Nodes in a CAM Tree?

- Two operations to explore CAM tree:
  - CAM-Join
  - CAM-Extension
- Augmenting CAM tree with Suboptimal CAMs
- Objectives:
  - no false dismissal
  - no redundancy
- Plus: We want to this efficiently!

CAM-Join

- Superimpose two adjacency matrices if they share the same MP submatrix.

Case 1: both A and B have at least two edge entries in the last row

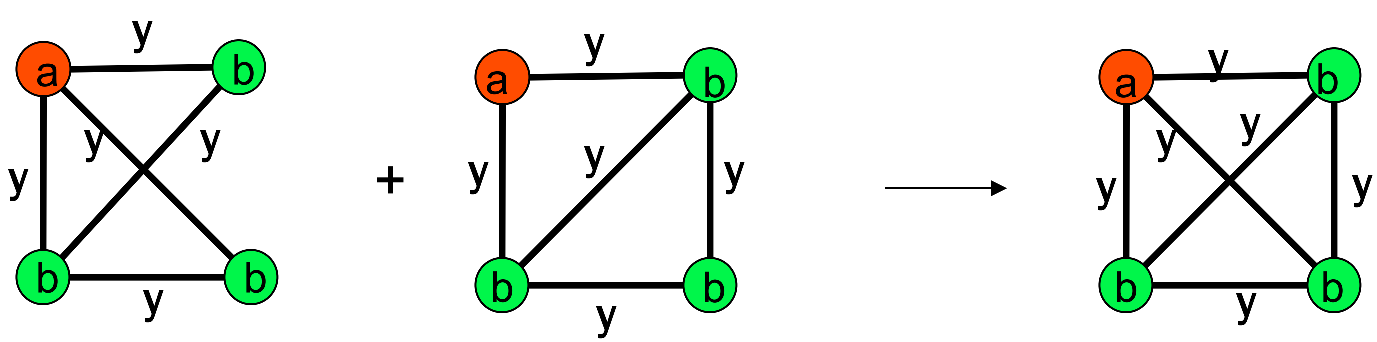
a			
y	b		
y	y	b	
y	y	0	b

+

a			
y	b		
y	y	b	
y	0	y	b

→

a			
y	b		
y	y	b	
y	y	y	b



Join Case 1

if $f \neq k$ **then**

$join(A, B) = \{C\}$ where C is a $m \times m$ matrix such that

$$c_{i,j} = \begin{cases} b_{i,j} & i = n, j = k \\ a_{i,j} & \text{otherwise} \end{cases}$$

else

$join(A, B) = \emptyset$

end if

Join Case 2

A has at least two edge entries in last row but B has only one

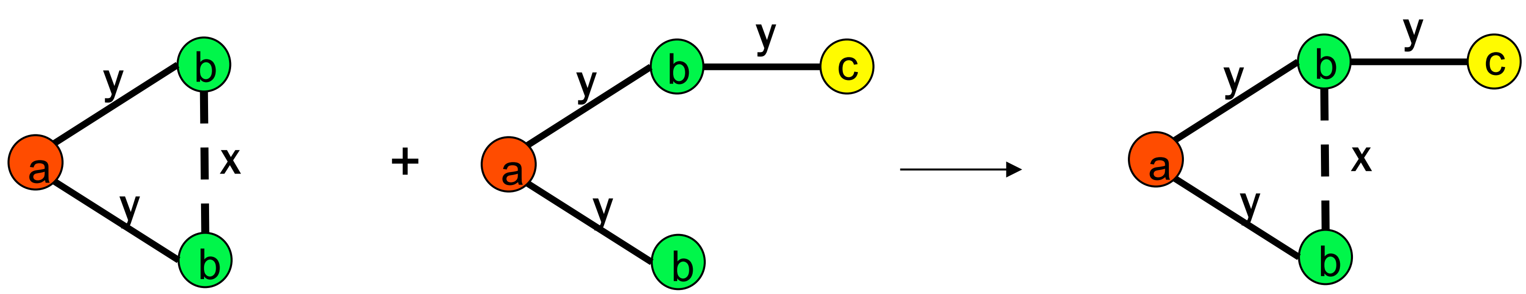
a		
y	b	
y	x	b

+

a			
y	b		
y	0	b	
0	y	0	c

→

a			
y	b		
y	x	b	
0	y	0	c



Join Case 2

1: $join(A, B) = \{C\}$ where C is a $n \times n$ matrix and

2:

$$c_{i,j} = \begin{cases} a_{i,j} & 0 < i, j \leq m \\ b_{i,j} & \text{otherwise} \end{cases}$$

Join Cases 3

a		
y	b	
y	0	b

+

a		
y	b	
0	x	b

→

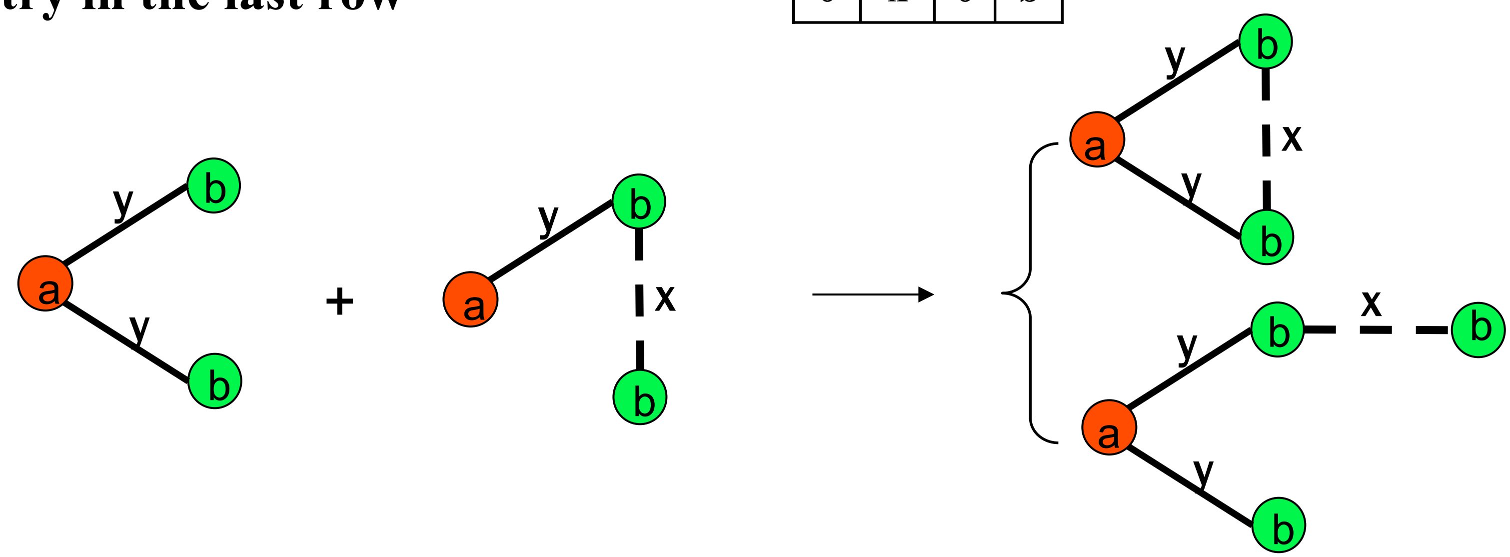
a		
y	b	
y	x	b

Join Case 3a

a			
y	b		
y	0	b	
0	x	0	b

Join Case 3b

both A and B have one edge entry in the last row



Join Case 3

let matrix D be a $(m+1) \times (m+1)$ matrix where (case 3b)

$$d_{i,j} = \begin{cases} a_{i,j} & 0 < i,j \leq m \\ b_{m,j} & i = m+1, 0 < j < m \\ 0 & i = m+1, j = m \\ b_{m,m} & i = m+1, j = m+1 \end{cases}$$

if $(f \neq k, a_{m,m} = b_{m,m})$ **then**

  C is $m \times m$ matrix where (case 3a)

$$c_{i,j} = \begin{cases} b_{i,j} & i = n, j = k \\ a_{i,j} & \text{otherwise} \end{cases}$$

  $join(A, B) = \{C, D\}$

else

  $join(A, B) = \{D\}$

end if

CAM-Extension

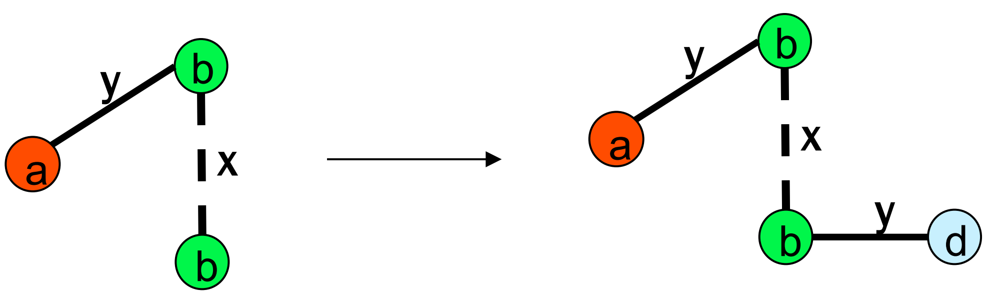
- only one edge entry in the last row
- Extend the current pattern by adding one more edge entry.

a		
y	b	
0	x	b

→

a			
y	b		
0	x	b	
0	0	y	d

Extension



Efficiency

- Comparing to FSG, the join efficiency is improved after “sorting” the CAMs.

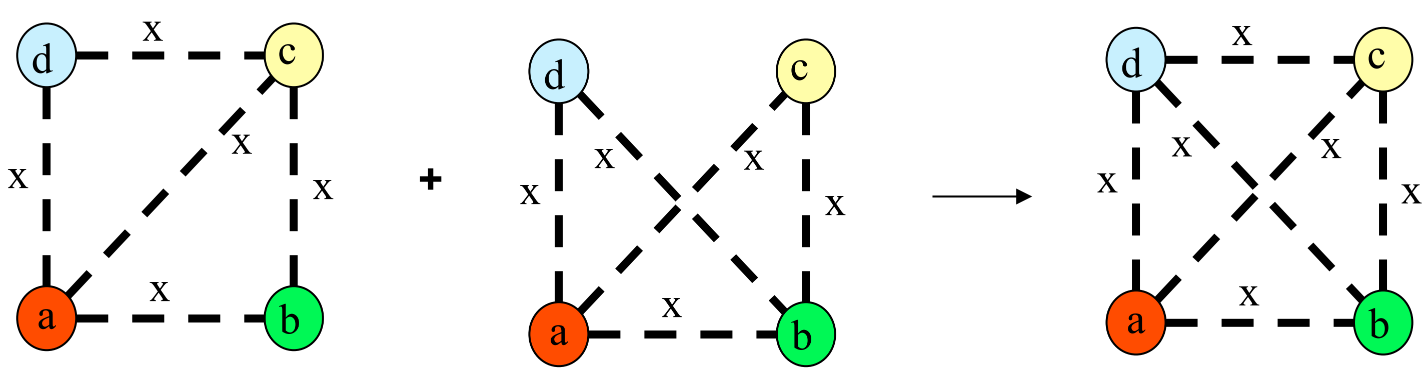
a			
x	b		
x	x	c	
x	0	x	d

+

a			
x	b		
x	x	c	
x	x	0	d

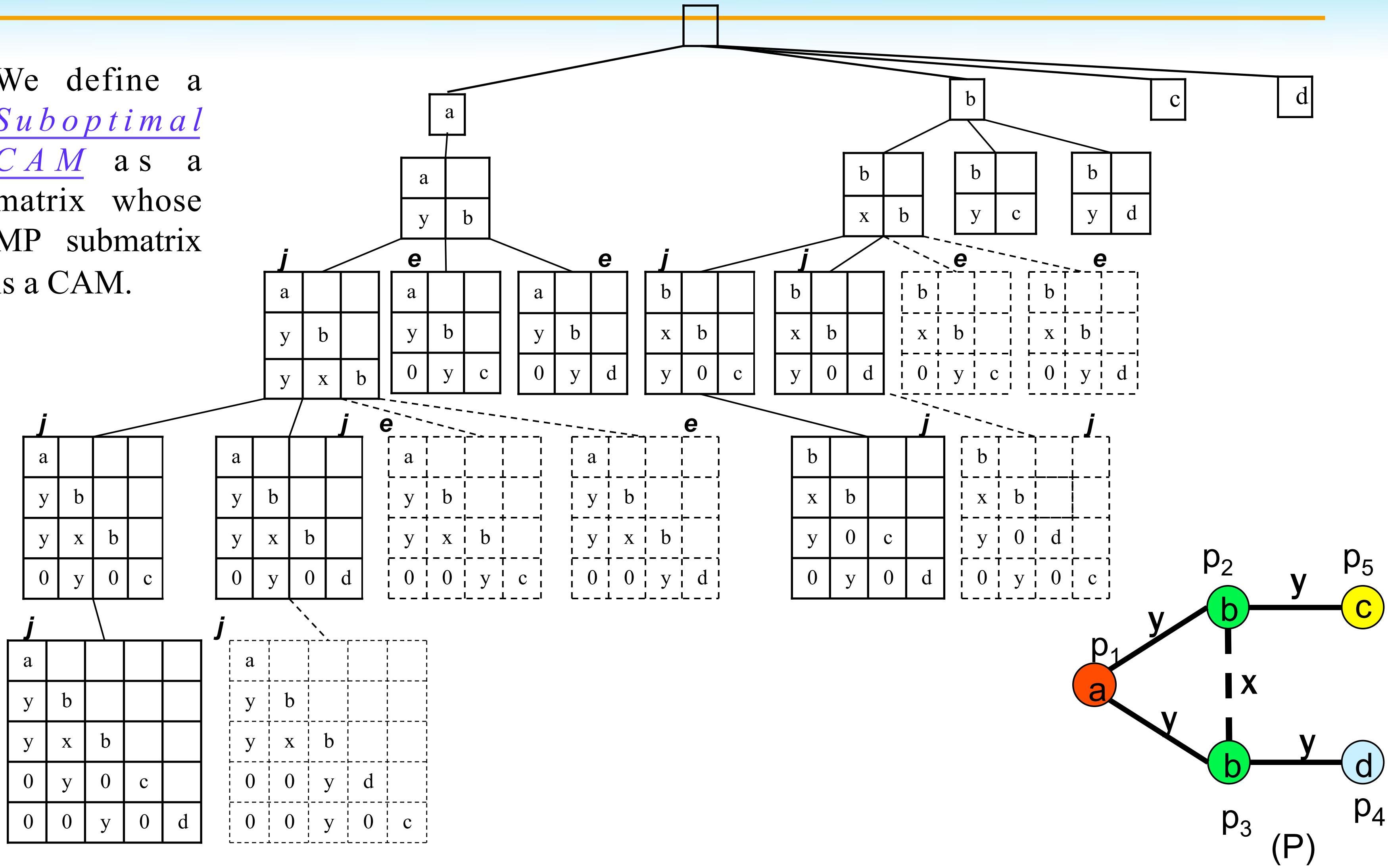
→

a			
x	b		
x	x	c	
x	x	x	d



Suboptimal Tree

We define a *Suboptimal CAM* as a matrix whose MP submatrix is a CAM.

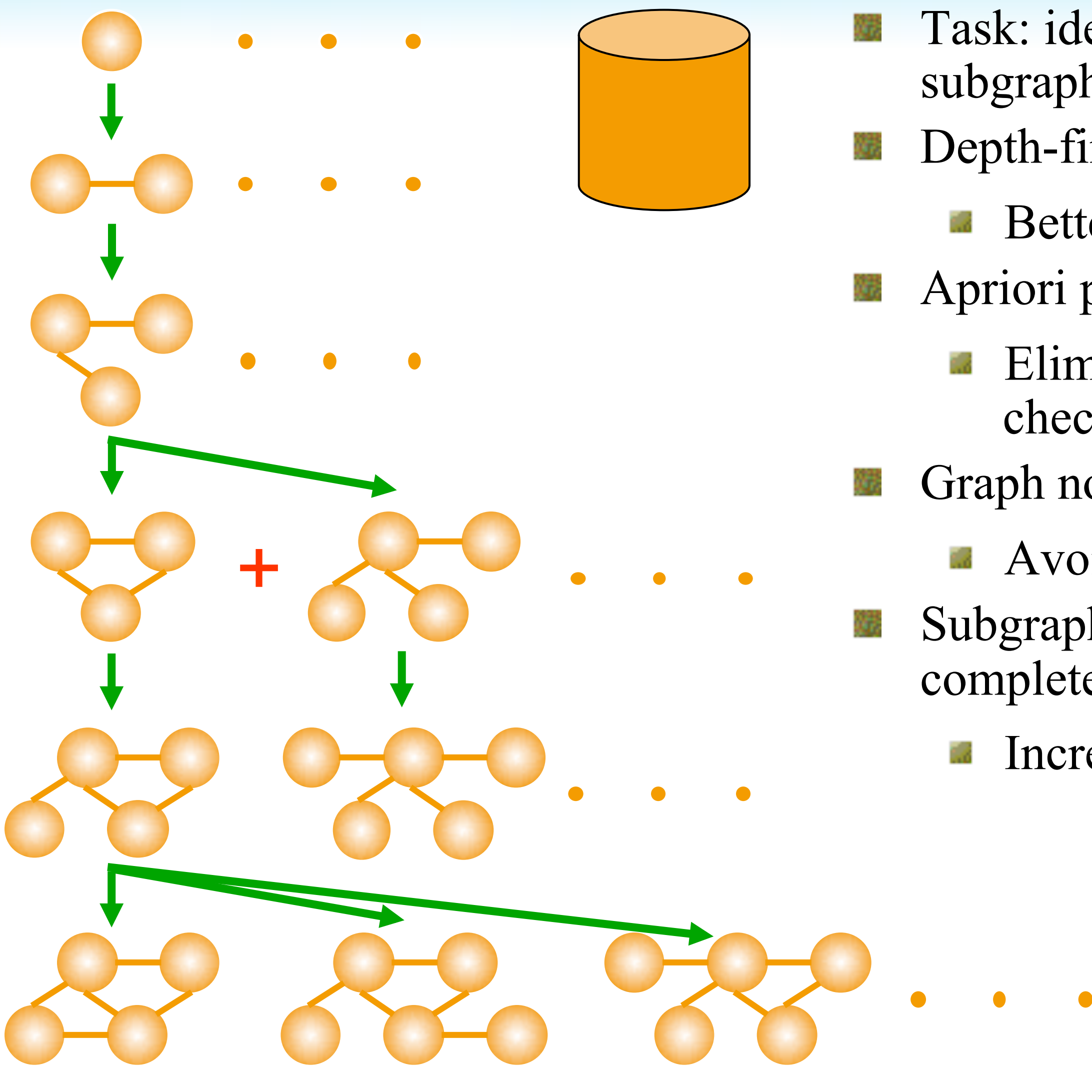


Summary

- Theorem:

  For a graph G, let C_{K-1} (C_k) be set of the suboptimal CAMs of all size-(K-1) (K) subgraphs of G ($K \geq 2$). Every member of set C_K can be enumerated unambiguously either by ***joining*** two members of set C_{K-1} or by ***extending*** a member in C_{K-1}.

FFSM Search

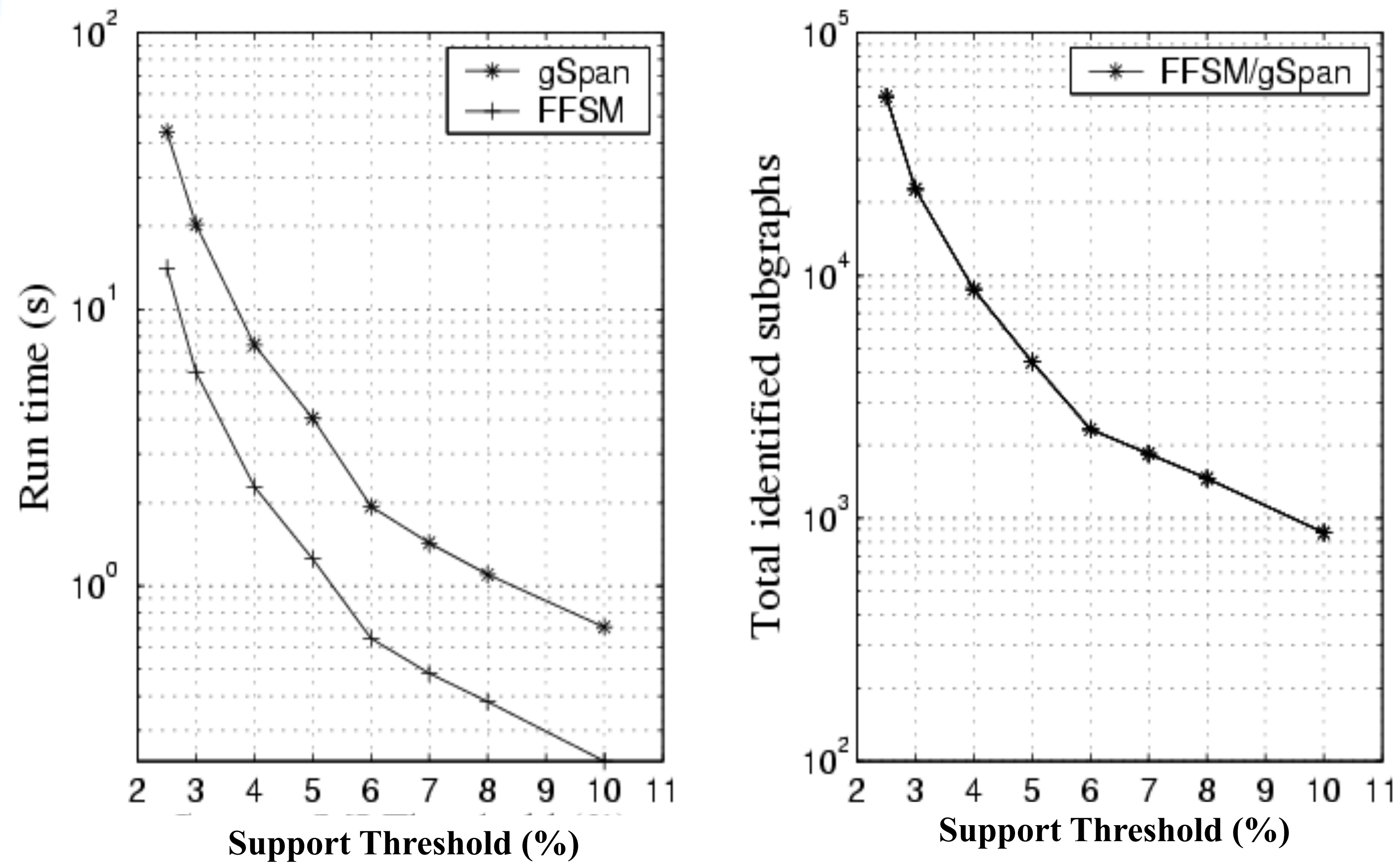


- Task: identify all frequently occurring subgraphs from a family of graphs
- Depth-first search
  - Better memory utilization
- Apriori property
  - Eliminate unnecessary isomorphism checks
- Graph normalization: CAM
  - Avoid redundant examination
- Subgraph isomorphism test is NP-complete
  - Incremental isomorphism check

Experimental Study

- Predictive Toxicology Evaluation Competition (PTE)
  - Contains: 337 compounds
  - Each graph contains 27 nodes and 27 edges on average
- NIH DTP Anti-Viral Screen Test (DTP CA/CM)
  - Chemicals are classified to be Confirmed Active (CA), Confirmed Moderate Active (CM) and Confirmed Inactive (CI).
  - We formed a dataset contains CA (423) and CM (1083).
  - Each graph contains 25 nodes and 27 edges on average

Performance (PTE)



Performance (DTP CACM)