HOKKAIDO UNIVERSITY

| | |
|---|---|
| Title | Mining Frequent Subgraphs from Linear Graphs |
| Author(s) | Tabei, Yasuo |
| Description | ERATO湊離散構造処理系プロジェクトシンポジウム(第2回) : 第73回情報処理学会全国大会イベント企画. 2011年3月2日(水). 東京工業大学 大岡山キャンパス. |
| Relation | 2010年度科学技術振興機構ERATO湊離散構造処理系プロジェクト講究録. p.510-517. |
| Issue Date | 2011-06 |
| Doc URL | https://hdl.handle.net/2115/48321 |
| Type | conference presentation |
| File Information | 10.tabei_LGM_20101129.pdf |

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

# Mining Frequent Subgraphs from Linear Graphs

Yasuo Tabei

Computational Biology Research Center, AIST

Joint work with
Daisuke Okanohara (Univ. of Tokyo),
Shuichi Hirose (AIST),
Koji Tsuda (AIST)

## Goal of this talk

- What is a linear graph?
- A property of linear graphs.
- Subgraph mining algorithm from linear graphs.

## Outline

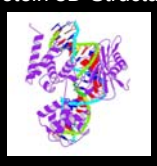
- Motivation
  - The needs for frequent subgraph mining algorithm
  - What is a linear graph?
- Method
  - Subgraph enumeration algorithm from a linear graph
  - Extension to frequent subgraph mining algorithm
- Experiments and Results
  - Motifs extraction from protein 3D-structures in molecular biology
  - Phrase extraction from predicate-argument structures in NLP
- Conclusion

## Outline

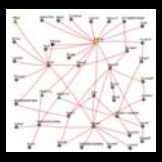
- Motivation
  - The needs for frequent subgraph mining algorithm
  - What is a linear graph?
- Method
  - Subgraph mining algorithm from a linear graph
  - Frequent subgraph mining algorithm from linear graphs
- Experiments and Results
  - Motifs extraction from protein 3D-structures in molecular biology
  - Phrase extraction from predicate-argument structures in NLP
- Conclusion

## Data Represented as Graphs

Protein 3D-Structure

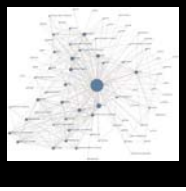


Gene co-expression Network



Chemical Compound

Social Network



## Frequent Subgraph Mining

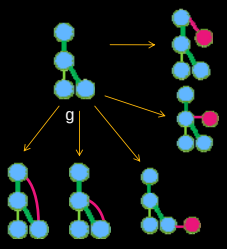
- Enumerate all frequent subgraphs in a graph database

Input: graph database $G=\{g_1,g_2,\ldots,g_N\}$

Output: frequent subgraphs appearing in at least m graphs

## gSpan algorithm (Yan et al., 2002)

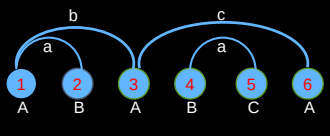
- Rightmost pattern extension
- Duplication can happen
- Minimum DFS code checking
  - Time exponential to pattern size




## Linear Graph (Davydov et al., 2004)

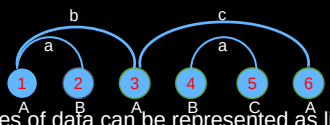
- Labeled graph whose vertices are totally ordered
- Linear graph g=(V,E,$L^V$,$L^E$)
  - V⊂N: ordered vertex set
  - E⊆V×V: edge set
  - $L^V$: V→$\Sigma^V$: vertex labeling
  - $L^E$:E→$\Sigma^E$: edge labeling

Ex) RNA, protein, alternative splicing forms, PAS




## Linear Graph (Davydov et al., 2004)

- Labeled graph whose vertices are totally ordered



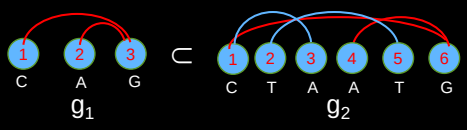

- Many types of data can be represented as linear graphs
  - Protein contact maps
  - Alternative splicing forms
  - RNA secondary structures
  - Predicate-argument structure

## Linear Subgraph Relation

- $g_1$ is a linear subgraph of $g_2$
  ⇔ i)The ordinary subgraph condition
  - the vertex labels are matched
  - all edges of $g_1$ also exit in $g_2$ with the correct labels

  ii) The order of vertices are conserved

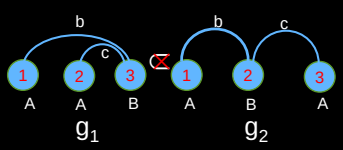
Ex)




## Example of Not Linear Subgraph

- $g_1$ is not a linear subgraph of $g_2$
  - vertex labels are matched
  - all edges of $g_1$ also exit in $g_2$ with the correct labels
  - the order of vertices is not conserved

Ex)



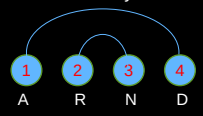

## Total order among edges in a linear graph

- Compare the left nodes first. If they are identical, look at the right nodes
- ∀$e_1$=(i,j),$e_2$=(k,l)∈$E_g$, $e_1$<$_e$$e_2$
  if and only if (i)i<k or (ii)i=k, j < l

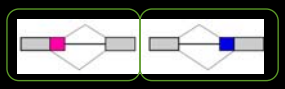
Ex)

## Disconnected Patterns

- Linear Graph: Sequence + Graph
- In sequence mining, gapped patterns are considered
- Need to mine disconnected patterns as well
- Data represented as disconnected patterns
  - Protein contact maps
  - RNA secondary structure




- Alternative splicing



## Outline

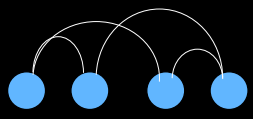
- Motivation
  - The needs for frequent subgraph mining algorithm
  - Linear Graph
- Method
  - Subgraph mining algorithm from a linear graph
  - Frequent subgraph mining algorithm from linear graphs
- Experiments and Results
  - Motifs extraction from protein 3D-structures in molecular biology
  - Phrase extraction from predicate-argument structures in NLP
- Conclusion

## Overview of LGM: Linear Graph Miner

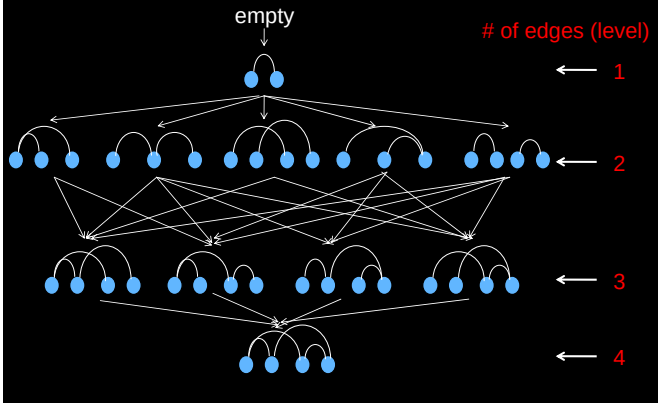
- Mining method of frequent linear subgraphs from a set of linear graphs
  - $|\{i = 1,...,|G| : g \subseteq g_i\}| \geq \sigma$ $\sigma$ : minimum support threshold
- Mining both connected subgraphs and disconnected subgraphs with a unified framework
- For the efficient enumeration, we use reverse search techniques (Avis and Fukuda 1996).
- The computational time is polynomial delay.

## Enumeration of All Linear Subgraphs of a Linear Graph

- Before considering a mining algorithm, we have to solve the problem of subgraph enumeration first
- How to enumerate all subgraphs of the following linear graph without duplication?

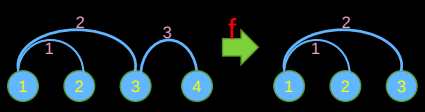


## Search Lattice of All Subgraphs



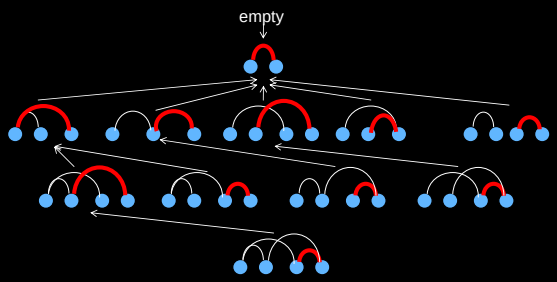

## Reverse Search (Avis and Fukuda, 1993)

- All subgraphs can be enumerated by traversing the search lattice
  - To prevent duplication is difficult
- Need to define a search tree in the search lattice
- Reduction map f
  - Mapping from a child to its parent
  - Remove the largest edge



## Search Tree induced by the reduction map

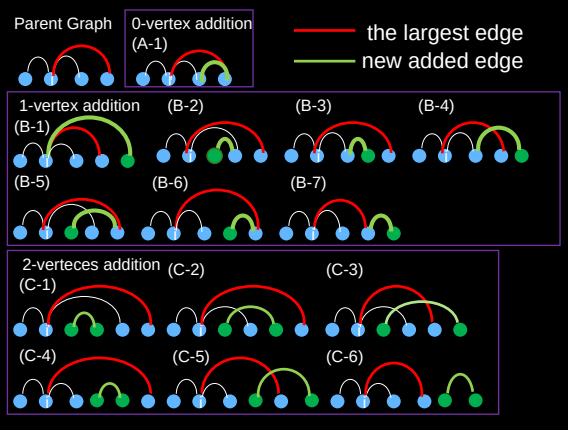
- By applying the reduction map to each element search tree can be induced




## Inverting the reduction map $f^{-1}$

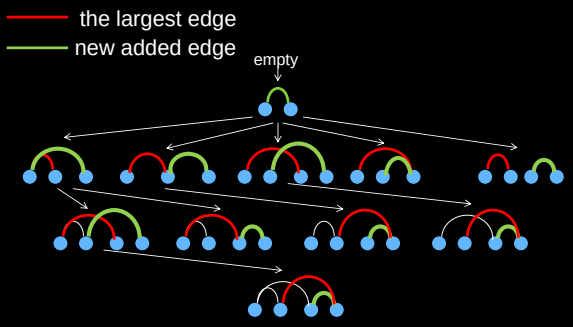
- In traversing the tree from root, children nodes are created on demand
  - Consider all children candidate
  - Take the ones that qualify the reduction map
  - Basically, to invert the reduction map is difficult
- However, in this particular case, the reduction map can be inverted explicitly
  - Can derive the pattern extension rule (from parent to children)

## Pattern Extension Rule



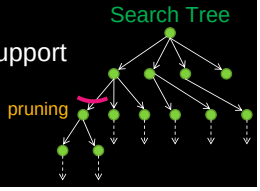

## Traversing search tree from root

- Depth first traversal for its memory efficiency




## Frequent Subgraph Mining

- Basic idea: find all possible extensions of a current pattern in the graph database, and extend the pattern.
- Occurrence list $L_G(g)$
  - Record every occurrence of a pattern g in the graph database G
  - Calculate the support of a pattern g by the occurrence list.
- Use anti-monotonicity of the support for pruning




## Outline

- Motivation
  - The need for frequent subgraph mining algorithm
  - Linear Graph
- Method
  - Subgraph mining algorithm from a linear graph
  - Frequent subgraph mining algorithm from linear graphs
- Experiments and Results
  - Motifs extraction from protein 3D-structures in molecular biology
  - Phrase extraction from predicate-argument structures in NLP
- Conclusion

## Experiments

- Motif extraction from protein 3D structures
  - Contact Maps
- Phrase discovery from movie evaluation texts
  - Predicate-Argument Structure

| Dataset name | Number of data | Average number of vertices | Average number of edges | Number of vertex labels |
|---|---|---|---|---|
| Protein3D | 742 | 371 | 498 | 6 |
| Sentiment | 10662 | 20 | 20 | 20326 |
| Subjectivity | 10000 | 23 | 24 | 22610 |

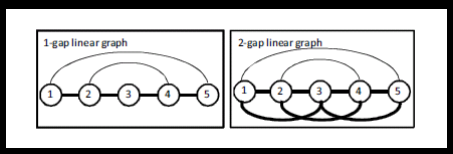
**Table 1. Datasets used in the experiments.**

## Motif extraction from protein 3D structures

- Pairs of homologous proteins in thermophilic organism and methophilic organism
- Construct a linear graph from a protein
  - Represent each amino acid as a vertex
  - Assign vertex labels from {1,…,6} according to its property (Mirny, 1999).
  - Draw an edge between the pair of amino acid residues whose distance is within 5 Å
  - No edge labels.
- Rank the patterns by statistical significance (p-values)
  - Association to thermophilic/methophilic label
  - Fisher exact test

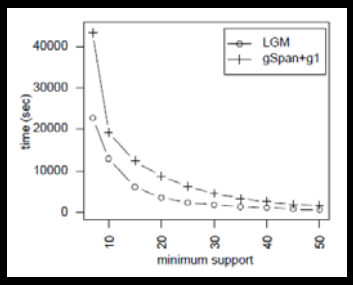
## Applying gSpan

- Want to compare the execution time of our algorithm with that of gSpan
- gSpan is not directly applicable
  - Contact maps are not always connected
  - Made 1-gap and 2-gap linear graphs



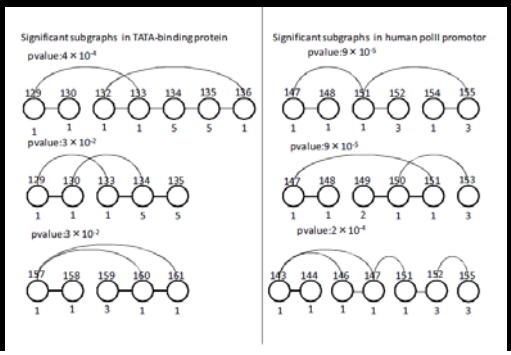

## Runtime comparison

- LGM is faster than gSpan
- Execution time of LGM is reasonable.

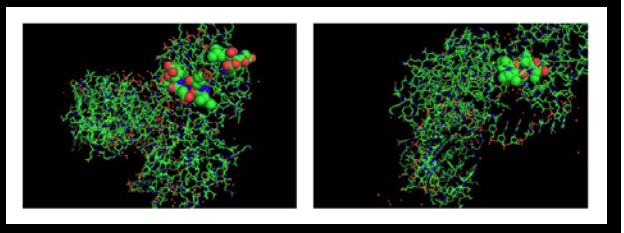



- gSpan does not work on the 2-gap linear graph dataset even if the minimum support threshold is 50.

- Minimum support = 10
- 103 patterns whose p-value < 0.001
- Thermophilic (TATA), Mesophilic (pol II)




## Mapping motifs in 3D structure

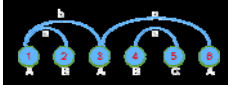


## Phrase extraction from predicate-argument structures

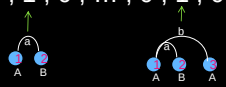
- Internet movie review dataset (Pang et al., 2004)
- Sentiment dataset
  - 5331 positive and 5331 negative opinions
- Subjectivity dataset
  - 5000 subjective and 5000 objective sentences
- PAS by ENJU syntactic parser (Miyao et al., 2008)
- Extract characteristic phrases (subgraph patterns)

## Select salient patterns by linear SVM

- Binary features from patterns
- Each pattern corresponds to an SVM's weight
  - interpretable
- Take the patterns with major weights



(0 , … , 0 , 1 , 0 , … , 0 , 1 , 0 , …)



## Methods in comparison

- PAS+gSpan
  - Predicate argument structure + gSpan
  - No edges added
- Dep+FREQT
  - Dependency tree (KSDep) + FREQT (Tree Miner)
- N-gram
  - Modified PrefixSpan (Sequence Miner)

## Classification Accuracy

- The accuracy of LGM is better than that of gSpan
- PAS representation is comparable to the other representations.

| Data | PAS+LGM | PAS+gSpan | dep | n-gram | bow |
|---|---|---|---|---|---|
| sentiment | 76.9 | 76.6 | 73.7 | 76.6 | 76.6 |
| subjectivity | 91.6 | 91.4 | 90.8 | 91.9 | 90.9 |

## Phrase structure extraction from predicate-argument structures

Only simple sequential patterns are extracted

| sentiment | PAS | | n-gram | | dep | |
|---|---|---|---|---|---|---|
| funny | -0.44 | unfunny | -0.44 | unfunny | 0.40 | funny |
| | 0.41 | funny | 0.40 | funny | -0.29 | unfunny |
| | -0.27 | occasionally funny | 0.21 | quite funny | -0.18 | occasionally funny |
| | -0.21 | it funny | 0.19 | quite funny . | -0.13 | less funny |
| | 0.20 | quite funny | -0.18 | really funny | -0.20 | funny but |
| | -0.20 | is n't funny | 0.15 | and funny | 0.13 | really funny |
| | -0.17 | not funny | -0.15 | funny . | 0.13 | funny film |
| | -0.14 | funny but | -0.14 | occasionally funny | 0.13 | charming and funny |
| | 0.14 | funny film | 0.14 | is funny | 0.13 | funny charming |
| | 0.14 | funny and | 0.13 | often-funny | -0.12 | , occasionally funny |
| boring | -0.57 | boring | -0.47 | boring | -0.54 | boring |
| | 0.15 | never boring | 0.16 | never boring | -0.13 | boring , |
| | -0.10 | It 's boring | -0.09 | boring . | 0.09 | it 's boring |
| | -0.06 | boring and | -0.08 | boring , | 0.09 | s boring |
| | -0.04 | so boring | -0.07 | is boring . | -0.04 | of boring |
| | -0.04 | movie boring | -0.07 | is boring | -0.03 | as boring |
| | -0.03 | of boring | -0.05 | boring and | -0.03 | is boring |
| | -0.03 | just boring | -0.35 | and boring | -0.02 | S boring . |
| | -0.03 | and boring | -0.03 | a boring | -0.02 | S boring |
| | -0.01 | just boring and obvious | -0.02 | S a boring . | -0.02 | boring . |

## Phrase structure extraction from predicate-argument structures

Phrase structures were extracted.

| sentiment | PAS | | n-gram | | dep | |
|---|---|---|---|---|---|---|
| funny | -0.44 | unfunny | -0.44 | unfunny | 0.40 | funny |
| | 0.41 | funny | 0.40 | funny | -0.29 | unfunny |
| | -0.27 | occasionally funny | 0.21 | quite funny | -0.18 | occasionally funny |
| | -0.21 | it funny | 0.19 | quite funny . | -0.13 | less funny |
| | 0.20 | quite funny | -0.18 | really funny | -0.20 | funny but |
| | -0.20 | is n't funny | 0.15 | and funny | 0.13 | really funny |
| | -0.17 | not funny | -0.15 | funny . | 0.13 | funny film |
| | -0.14 | funny but | -0.14 | occasionally funny | 0.13 | charming and funny |
| | 0.14 | funny film | 0.14 | is funny | 0.13 | funny charming |
| | 0.14 | funny and | 0.13 | often-funny | -0.12 | , occasionally funny |
| boring | -0.57 | boring | -0.47 | boring | -0.54 | boring |
| | 0.15 | never boring | 0.16 | never boring | -0.13 | boring , |
| | -0.10 | It 's boring | -0.09 | boring . | 0.09 | it 's boring |
| | -0.06 | boring and | -0.08 | boring , | 0.09 | s boring |
| | -0.04 | so boring | -0.07 | is boring . | -0.04 | of boring |
| | -0.04 | movie boring | -0.07 | is boring | -0.03 | as boring |
| | -0.03 | of boring | -0.05 | boring and | -0.03 | is boring |
| | -0.03 | just boring | -0.35 | and boring | -0.02 | S boring . |
| | -0.03 | and boring | -0.03 | a boring | -0.02 | S boring |
| | -0.01 | just boring and obvious | -0.02 | S a boring . | -0.02 | boring . |

## Conclusions

- Efficient subgraph mining algorithm from linear graphs
- Search tree is defined by reverse search principle
- Patterns include disconnected subgraphs
- Computational time is polynomial-delay
- Interesting patterns from proteins and sentences

## Another topics

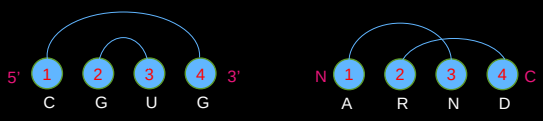
- Alignment algorithms for RNA sequences
  - Ph.D. study
- All pairs similarity search method
  - nearest neighbor graphs

## Q & A

## Data represented as linear graphs

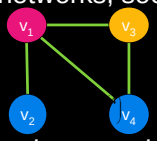
- DNA, RNA, protein-3D structure, predicate argument structure
  - reference point: 5-strand(DNA, RNA), N-terminal (protein)

Ex) RNA Protein (edge: 5 Å )

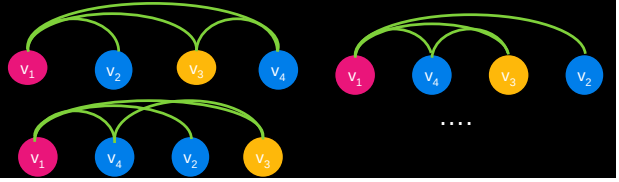



## Data NOT represented as linear graphs

- Chemical compounds, Gene co-expression networks, social networks etc

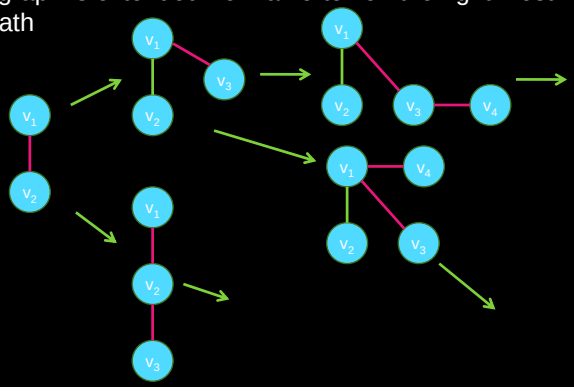



- Can NOT assign an unique vertex order
  - 4! manners




## Right most pattern extension

- — right most path
- A graph is extended from a vertex on the right most path

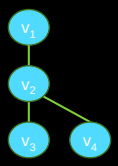


## What is a code for an edge

- A code assigned for an edge in a graph
  - a set of label ids, vertex labels, edge ids

Ex)

( vertex id1, vertex id2, vertex id1 label, vertex id2 label, edge label)




## What is the Minimum DFS Code?

DFS Code: a sequence of codes by depth-first traversal

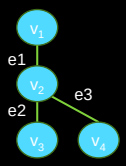
Ex)

- Start from v1

(v1,v2,l_v1,l_v2,l_e1)
(v2,v3,l_v1,l_v3,l_e2)
(v2,v4,l_v1,l_v3,l_e3)

-Start from v2

(v2,v1,l_v1,l_v2,l_e1)
(v2,v3,l_v2,l_v3,l_e2)
(v2,v4,l_v2,l_v4,l_e3)




- A graph is assinged several DFS Codes according to the starting vertex of DFS
- Define Total order among DFS codes
- Chose minimum DFS Code as a canonical code

## Motif extraction

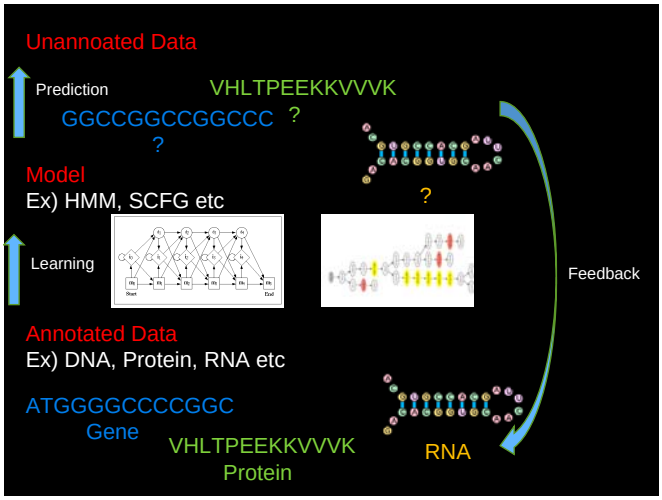
- To extract protein-3D motifs, we use the Fisher's exact test.

| | thermophilic | mesophilic | Total |
|---|---|---|---|
| With T | $n_{TP}$ | $n_{FP}$ | $n_T$ |
| Without T | $n_{FN}$ | $n_{TN}$ | $n_{T'}$ |
| Total | $n_P$ | $n_N$ | n |

$$\Pr = \frac{\binom{n_T}{n_{NT}}\binom{n_{T'}}{n_{FN}}}{\binom{n}{n_P}} = \frac{n_T!n_{T'}!n_P!n_N!}{n!n_{TP}!n_{FP}!n_{FN}!n_{TN}!}$$

Table1: 2×2 contingency table

- The P-value can be computed by the sum of all probabilities of tables that are more extreme than this table.
- Ranked the frequent subgraphs according to the P-values.
- Focused on a pair of proteins, TATA-binding protein and human polII promotor protein




- Algorithms for prediction and learning are based on Dynamic Programming (DP).
- Ordering in linear graphs is useful for designing DP algorithms