LIMOS
UMR 6158 CNRS

UNIVERSITE BLAISE PASCAL CLERMONT-FERRAND

Graph Mining

Frequent subgraph selection by means of substitution matrix

Presented by:

Wajdi DHIFLI

PhD superviser :
Prof. Engelbert MEPHU NGUIFO

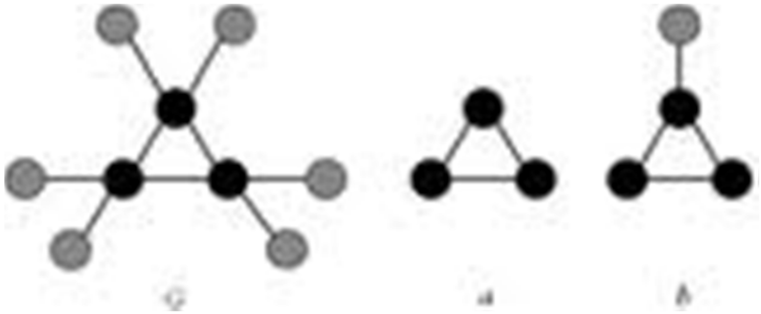
Frequent subgraphs

Problem description

- Given a collection of graphs or a single massive graph find all frequent subgraphs.

- A subgraph is frequent if:
  - Graph set D = {G1, G2, …,Gn}
  - Support(g) = number of Gi in D where g appears
  - g is frequent if : its support >= minimum support threshold

- Frequent subgraphs are useful at : characterizing graph sets, classifying and clustering graphs, graph compression, outliers discovery, …



Frequent subgraphs

Problems to resolve

- **Subgraph Isomorphism:** *For two labeled graphs* g *and* g'*, a subgraph isomorphism is an injective function* $f : V(g) \rightarrow V(g')$*, i.e.,* $\forall v \in V(g), l(v) = l'(f(v))$*; and,* $\forall (u, v) \in E(g), (f(u), f(v)) \in E(g')$ *and* $l(u, v) = l'(f(u), f(v))$*, where* l *and* l' *are the labeling functions of* g *and* g'*, respectively.* f *is called an embedding of* g *in* g'*.*

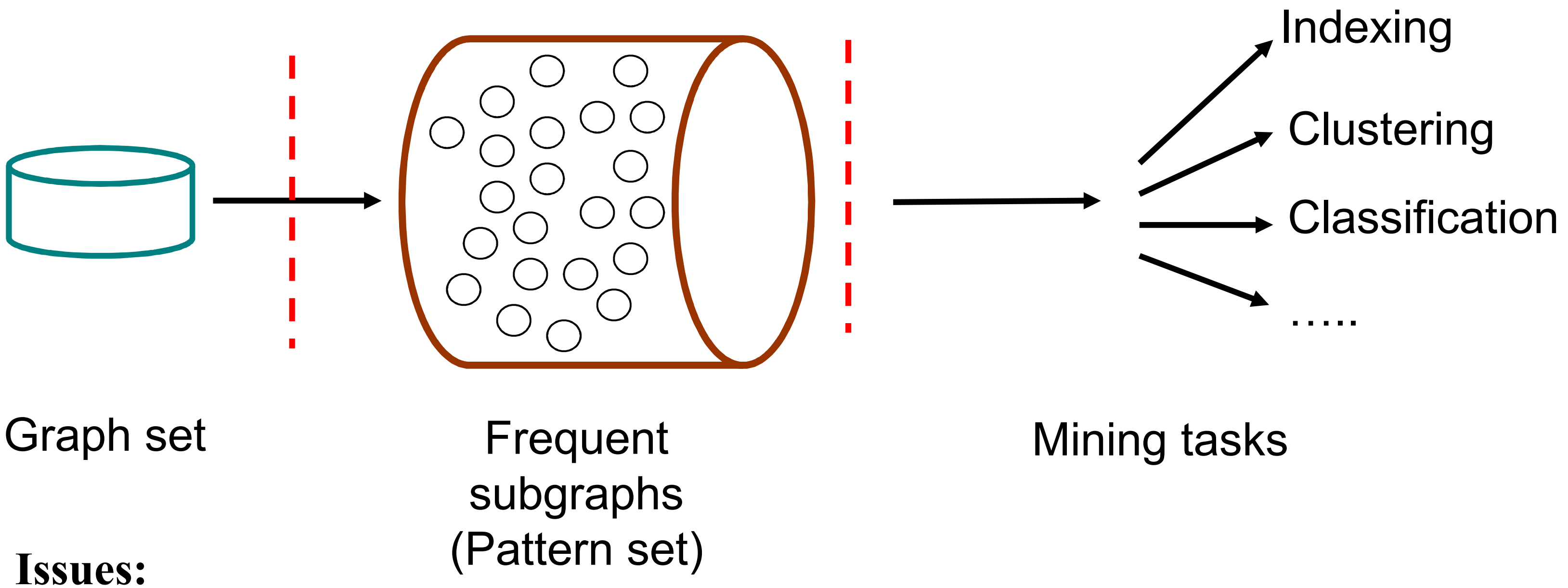
- **Frequent subgraph:** *Given a labeled graph dataset* $D = \{G1, G2, \ldots, Gn\}$ *and a subgraph* g*, the supporting graph set of* g *is* $Dg = \{Gi | g \subseteq Gi, Gi \in D\}$*. The support of* g *is* $\text{support}(g) = |Dg|/|D|$*. A frequent graph is a graph whose support is no less than a minimum support threshold, min sup.*

- **Anti-Monotonicity:** Anti-monotonicity means that a size-k subgraph is frequent only if all of its subgraphs are frequent. This property is crucial to confine the search space of frequent subgraph mining.

Frequent subgraphs discovery approaches

- **ILP approaches**
  - WARMR : King R.D., Srinivasan A. and Dehaspe L. (J. of Computer-Aided Molecular Design 2001)
  - FARMER : Nijssen, S. and Kok, J. IJCAI 2001
  - …
- **Apriori based approaches**
  - AGM/AcGM : Inokuchi et al (PKDD 2000)
  - FFSM : Huan et al (ICDM 2003)
  - …
- **Pattern growth based approaches**
  - Gspan : Yan and Han (ICDM 2002)
  - Gaston : Nijssen and Kok (KDD 2004)
  - …
- **Closed subgraphs**
  - CloseGraph: Yan, X. and Han, J. (KDD 2003)
- **Maximal subgraphs**
  - SPIN : Huan et al (KDD 2004)
  - Margin
  - …

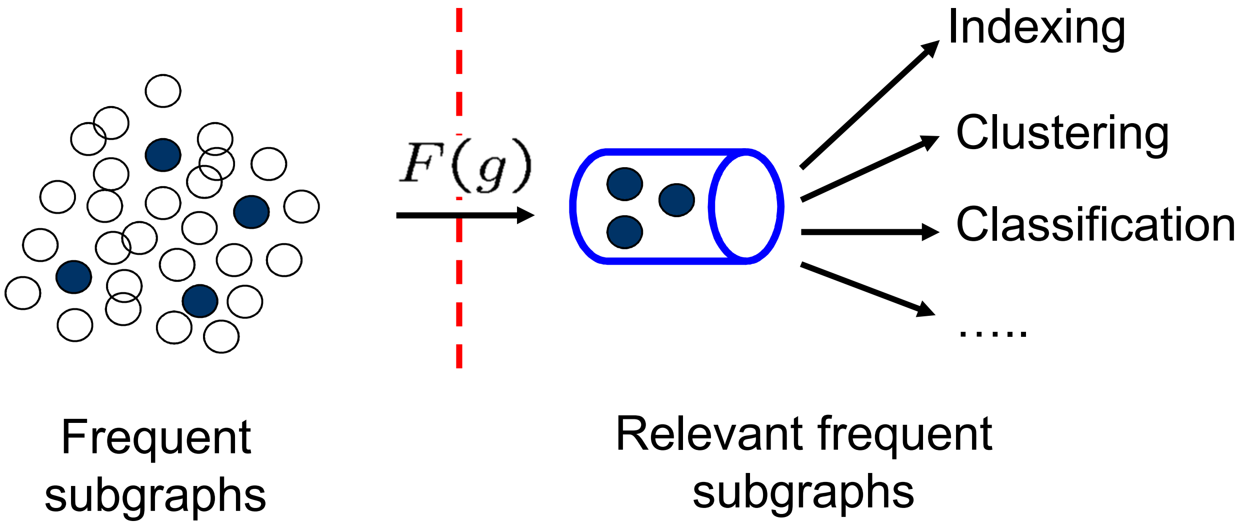
Frequent subgraph issues



Issues:

- Threshold setting
- Exponential Pattern Set
- Interpretation problem
- More information ≠ more knowledge
- No guarantee of the discovered subgraphs quality
- An **n**-edge frequent graph may have 2**n** subgraphs!

Patterns selection

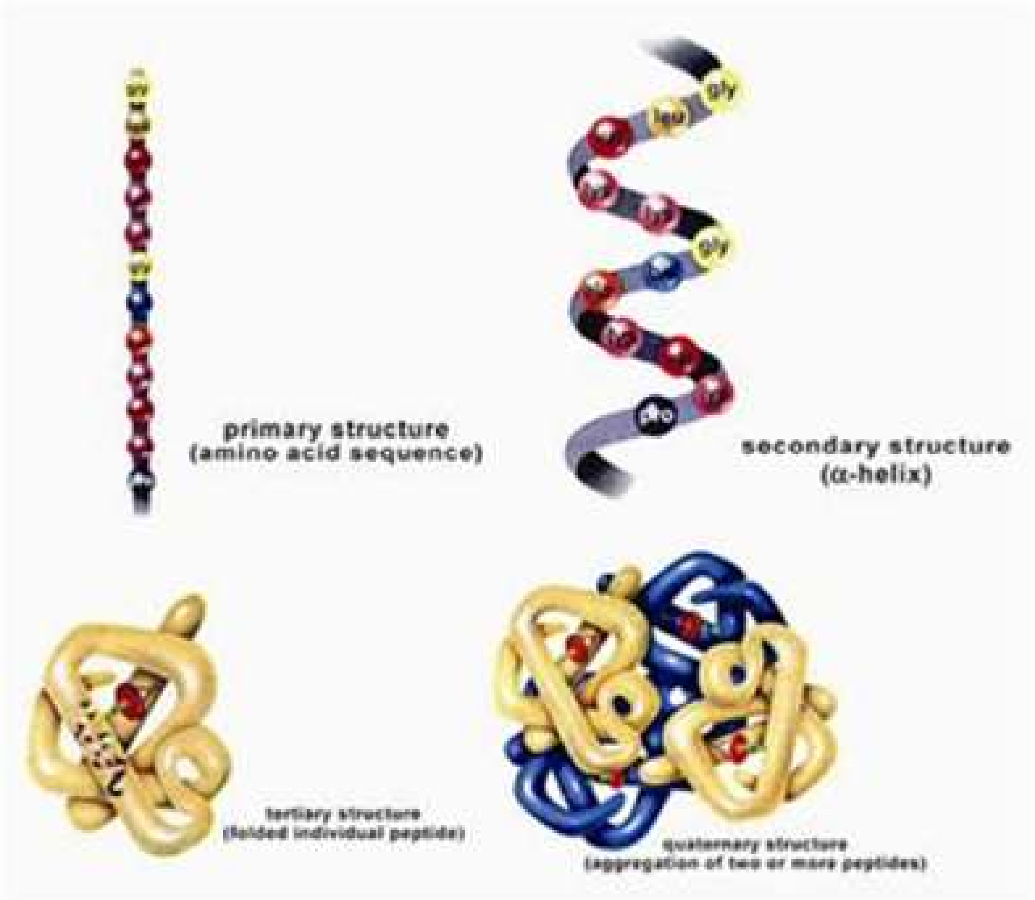


Aims:

- Decreasing the exponential number of discovered frequent subgraphs
- Enhancing (or at least maintaining) the quality of the pattern set
- Find relevant frequent subgraphs such that each frequent subgraph is close to one of the representative patterns

Frequent subgraphs selection by means of substitution matrix

Proteins



From sequence (string of characters) ➔ 3D structure (graph)

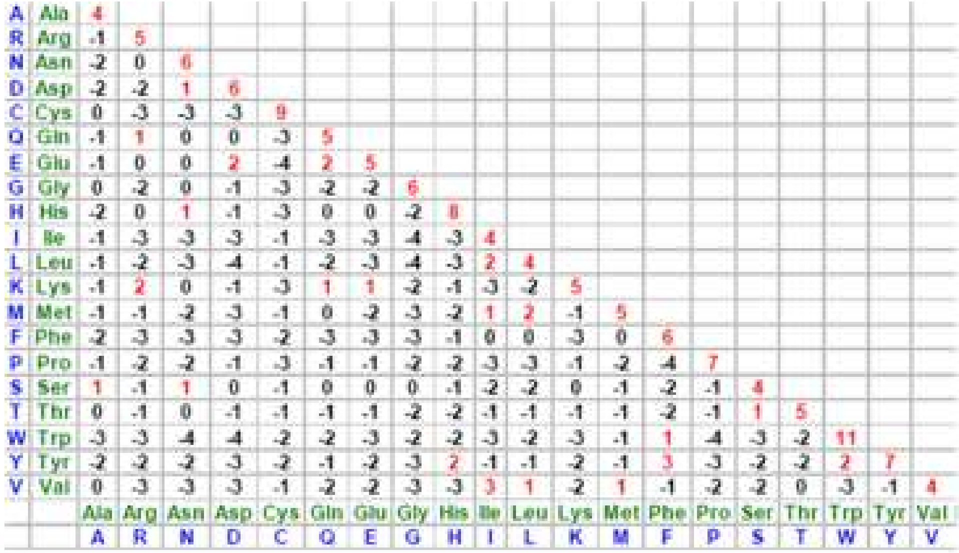
Frequent subgraphs selection by means of substitution matrix

- During the evolution, proteins go through changes, among them :
  - **Mutation** : is a substitution that exchanges one amino acid to another

CTGGAG
CTGGGG

In literature, there exist substitution matrices expressing scores of substitution between each possible pair of amino acids.

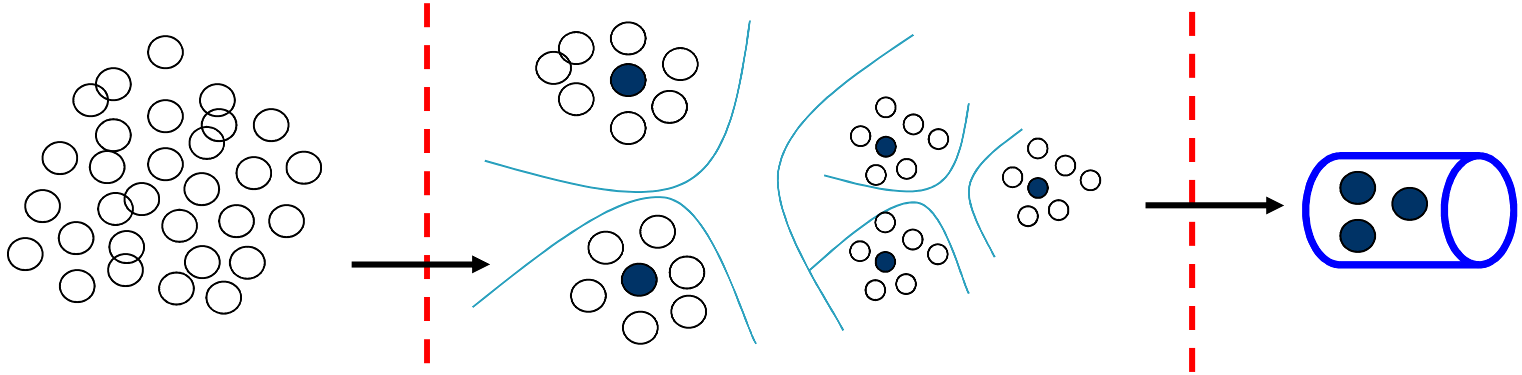
<u>Substitution matrix : Blosum 62</u>



A	Ala	4																			
R	Arg	-1	5																		
N	Asn	-2	0	6																	
D	Asp	-2	-2	1	6																
C	Cys	0	-3	-3	-3	9															
Q	Gln	-1	1	0	0	-3	5														
E	Glu	-1	0	0	2	-4	2	5													
G	Gly	0	-2	0	-1	-3	-2	-2	6												
H	His	-2	0	1	-1	-3	0	0	-2	8											
I	Ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	Leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K	Lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M	Met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	Phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	Pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	Ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	Thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	Trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	Tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	Val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4
		Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
		A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V

Motifs selection by means of substitution matrix

- **DDSM**: In [Saidi et al.: BMC bioinformatics, 2010] they proposed a new feature extraction method for protein sequences which explored the phenomenon of amino acids substitution to perform both a feature selection and dimension reduction.



- Keep only one motif for every set of substitutable motifs having the same length

Motifs selection by means of substitution matrix

- **Algorithm of DDSM**

$$P_m(M) = 1 - \prod_{i=1}^{k} P_i$$

$$P_i = S\,(M[i], M[i]) \,/\, \sum_{j=1}^{20} S^{+}\,(M[i], AA_j)$$

- *S(x, y)* is the substitution score of the amino acid y by the amino acid x as it appears in the substitution matrix.
- *S+(x, y)* : positive substitution score.
- *AAj* : amino acid of index *j* among the 20 amino acids.

```
Begin
 S :Set of motifs.
 Divide S into a set of groups of motifs having the same size;
 for each group M of S
   Sort M by descending order of $P_m$;
   for each motif M[i] (i from n to 1)
     if $P_m$(M[i])=0 then
       M[i] is a main motif;
     else
       x ← position of the first motif in M;
       for each M[j] (j from x to i)
         if M[j] substitute M[i] or j=i then
           M[j] is a main motif;
           break;
         end if
       end for
     end if
   end for
   for each M[i] in S
     if M[i] is not a main motif then
       delete M[i];
     end if
   end for
 end for
fin.
```

Motifs selection by means of substitution matrix

Main algorithm of DDSM

Motifs selection

P_m ranking

Substitution clustering

Shape verification

Pruning

```
Begin
 S :Set of motifs.
 Divide S into a set of groups of motifs having the same size;
 for each group M of S
   Sort M by descending order of $P_m$;
   for each motif M[$i$] (i *from* $n$ to 1)
     if $P_m$(M[$i$])=0 then
       M[$i$] is a main motif;
     else
       $x \leftarrow$ position of the first motif in M;
       for each M[j] ($j$ *from* $x$ to $i$)
         if M[$j$] substitute M[$i$] or $j$=$i$ then
           M[j] is a main motif;
           break;
         end if
       end for
     end if
   end for
   for each M[$i$] in S
     if M[$i$] is not a main motif then
       delete M[$i$];
     end if
   end for
 end for
fin.
```

Limits of DDSM and enhancements

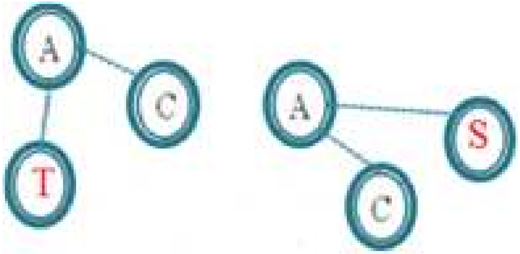
Shape isomorphism: we consider only the structure i.e. only nodes and edges, labels are ignored.

- **DDSM:**
  - Not universal: deals only with protein sequences i.e. strings of characters
  - Does not deal with more complex structures such as trees of graphs
  - Does not take into account spatial links between distant elements/nodes (amino acids)

NA**V**T ➔ NA**I**T

- **New Approach:**
  - Deals with more complex structures : dedicated to protein's 3D structure (graphs)
  - Takes into account spatial links
  - Deals also with protein sequences since a sequence can be also considered as graph i.e. paths

NA**V**T ➔ NA**I**T +

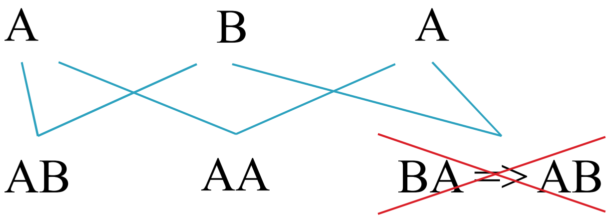


Limits of DDSM and enhancements

Shape isomorphism

- **DDSM**
  - We need only to verify motif size
- **With graphs:**
  - We need to verify both nodes and edges i.e.

- In order perform a **shape isomorphism** between two frequent subgraph, we benefit from the canonical order achieved during the candidate generation in the frequent subgraph generation process.



Limits of DDSM and enhancements

Motifs selection

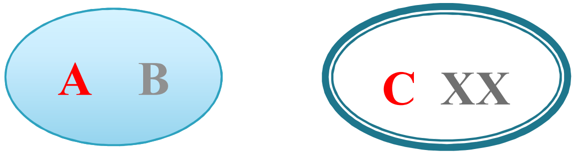
- ***P_m* ranking**
  - Formula : Same as DDSM
  - **DDSM :** (if Pm(M[i])=0 then {M[i] is a main motif;}) 🡐 This neglect the substitution between some motifs.

example: ***PPG*** ***PPN***

- **Substitution based clustering:**

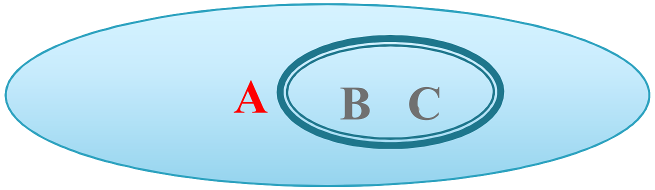
A, B and C three motifs having the same shape. P_m(A) > P_m(B) > P_m(C) :

- Independence



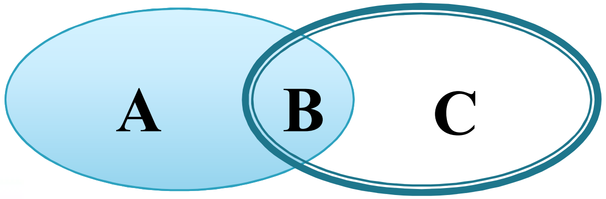
A substitute B
C substitute XX

- Inclusion



A substitute B
B substitute C
A substitute C

- **Intersection**

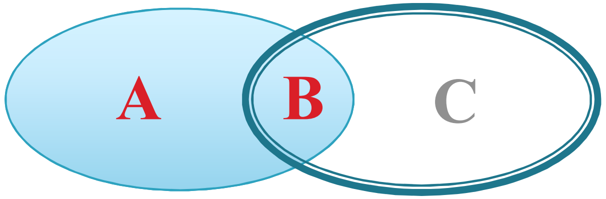


A substitute B
B substitute C

Limits of DDSM and enhancements

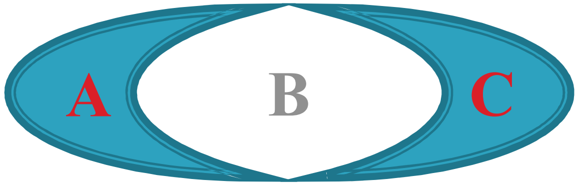
Intersection

- **DDSM**



  - Dependent motifs

- **New approach**



  - Considering more distinct motifs, hence better description : independent vectors

Which approach keeps less motifs !!!

Limits of DDSM and enhancements

Substitution Kernel function

Considering two motifs M and M’, having the same shape. M substitute M’ iff.:

- S(M[i], M’[i]) ≥ 0 , i = 1.. n
- SP(M,M’) ≥ Threshold

- **Old kernel function**
- $SP(M,M') = S_m(M, M') \,/\, S_m(M, M)$

$S_m(M, M')$ is the score of substitution of M’ by M, $S_m(X, Y) = \sum_{i=1}^{n} S(X[i], Y[i])$

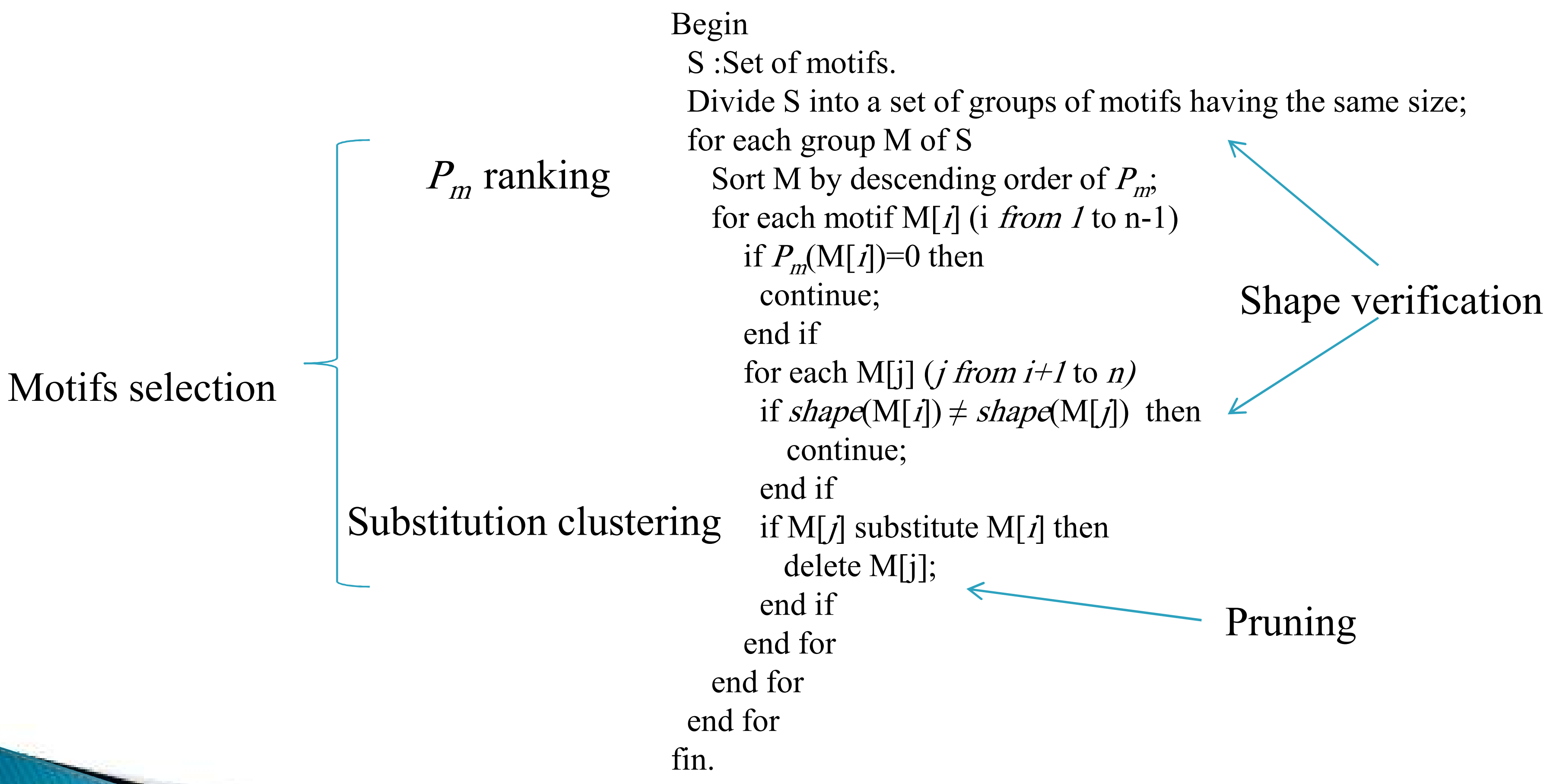
Express similarity between M and M’

- **New kernel function**
- $SP(M,M') = \prod_{i=1}^{k} P_i(M,M')$
  - $P_i(M,M')$ = S(M[i], M′[i]) / $\sum_{i=1}^{20} S^{+}$(M[i], AA_j)

Express the evolution probability of M to M’ among all the evolution possibilities

Frequent subgraphs selection by means of substitution matrix

Main algorithm



Motifs selection

P_m ranking

Substitution clustering

```
Begin
  S :Set of motifs.
  Divide S into a set of groups of motifs having the same size;
  for each group M of S
     Sort M by descending order of $P_m$;
     for each motif M[i] (i from 1 to n-1)
        if $P_m$(M[i])=0 then
          continue;
        end if
        for each M[j] (j from i+1 to n)
          if shape(M[i]) $\neq$ shape(M[j])  then
             continue;
          end if
          if M[j] substitute M[i] then
             delete M[j];
          end if
        end for
     end for
  end for
fin.
```

Shape verification

Pruning

Thanks