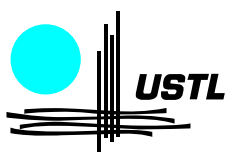
Large Scale Matching for Position Weight Matrices

Aude Liefooghe, Hélène Touzet and Jean-Stéphane Varré

LIFL UMR CNRS 8022 — Université Lille 1

July 07, 2006



Plan

Context

- Position Weigth Matrices (PWM)
- PWM Matching Problem

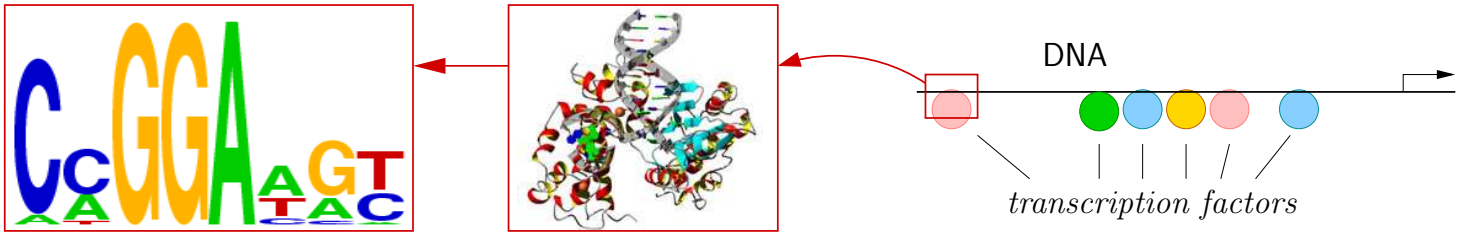
Multiple PWM Matching Problem

- Optimization 1 : Pruning PWMs
- Optimization 2 : Preprocessing PWMs

PWM Matching Problem for Similar Matrices

- General analysis
- Speed-up PWM matching problem with clusters

Position Weigth Matrices (PWM)



	A	C	G	T
1	-1.8	1.9	-2.3	-2.3
2	0.4	1.26	-2.3	-1.8
3	-2.3	-2.3	2	-2.3
4	-2.3	-2.3	2	-2.3
5	2	-2.3	-2.3	-2.3
6	1.1	-0.94	-2.3	0.4
7	0.11	0.07	1.42	-2.3
8	-1.8	0.4	0	1.1

- Transcription Factors : Control transcriptional regulation of gene expression
- DNA-protein interaction : Approximate nucleic motif
- Binding sites modelled by Position Weigth Matrices

PWM Matching Problem

A	C	G	T
-1.8	1.9	-2.3	-2.3
0.4	1.26	-2.3	-1.8
-2.3	-2.3	2	-2.3
-2.3	-2.3	2	-2.3
2	-2.3	-2.3	-2.3
1.1	-0.94	-2.3	0.4
0.11	0.07	1.42	-2.3
-1.8	0.4	0	1.1

- M a PWM of length n
- Score of a word u = $Score(u, M) = \sum_{i=1}^{n} M(i, u_i)$

PWM Matching Problem

A	C	G	T	
-1.8	1.9	-2.3	-2.3	A
0.4	1.26	-2.3	-1.8	C
-2.3	-2.3	2	-2.3	G
-2.3	-2.3	2	-2.3	G
2	-2.3	-2.3	-2.3	A
1.1	-0.94	-2.3	0.4	T
0.11	0.07	1.42	-2.3	A
-1.8	0.4	0	1.1	C

$$\text{score} = 6.37$$

- M a PWM of length n
- Score of a word u = $Score(u, M) = \sum_{i=1}^{n} M(i, u_i)$

PWM Matching Problem

A	C	G	T	
-1.8	1.9	-2.3	-2.3	A
0.4	1.26	-2.3	-1.8	C
-2.3	-2.3	2	-2.3	G
-2.3	-2.3	2	-2.3	G
2	-2.3	-2.3	-2.3	A
1.1	-0.94	-2.3	0.4	T
0.11	0.07	1.42	-2.3	A
-1.8	0.4	0	1.1	C

$$\text{score} = 6.37$$

- M a PWM of length n
- Score of a word u = $Score(u, M) = \sum_{i=1}^{n} M(i, u_i)$
- PWM matching problem
  - Input : a sequence S, a PWM M, a score threshold α
  - Output : Finding all positions i of S such that

$$Score(S(i...i+n-1), M) \geq \alpha$$

Multiple PWM Matching Problem

Input

- A (long) DNA sequence
- A large set of PWMs
  Examples of databases of PWMs : Jaspar, Transfac, etc.
- A score threshold for each PWM
  or a P-value corresponding to a uniform score threshold

Output

- All occurrences of all PWMs

How to speed up the computation of scores ?

Optimization 1

- Prune the score computation of the PWM using intermediate thresholds
- Occurrence probability is low

Optimization 2

- Preprocess the set of PWMs with an index structure
- Additivity property of score

Optimization 1 : Pruning PWMs

Score threshold $= 1$

A	C	G	T	
−1.8	1.9	−2.3	−2.3	*G*
0.4	1.26	−2.3	−1.8	*G*
−2.3	−2.3	2	−2.3	*T*
−2.3	−2.3	2	−2.3	*A*
2	−2.3	−2.3	−2.3	*T*
1.1	−0.94	−2.3	0.4	*G*
0.11	0.07	1.42	−2.3	*C*
−1.8	0.4	0	1.1	*C*

Optimization 1 : Pruning PWMs

Score threshold $= 1$

A	C	G	T	
−1.8	1.9	−2.3	−2.3	*G*
0.4	1.26	−2.3	−1.8	*G*
−2.3	−2.3	2	−2.3	*T*
−2.3	−2.3	2	−2.3	*A*
2	−2.3	−2.3	−2.3	*T*
1.1	−0.94	−2.3	0.4	*G*
0.11	0.07	1.42	−2.3	*C*
−1.8	0.4	0	1.1	*C*

Score of the prefix = -6.9

Maximum score of any suffix = 7.62

- -6.9 + 7.62 can never be greater than the score threshold 1

Optimization 1 : Pruning PWMs

Score threshold $= 1$

A	C	G	T	
−1.8	1.9	−2.3	−2.3	G
0.4	1.26	−2.3	−1.8	G
−2.3	−2.3	2	−2.3	T
−2.3	−2.3	2	−2.3	A
2	−2.3	−2.3	−2.3	T
1.1	−0.94	−2.3	0.4	G
0.11	0.07	1.42	−2.3	C
−1.8	0.4	0	1.1	C

Score of the prefix = -6.9

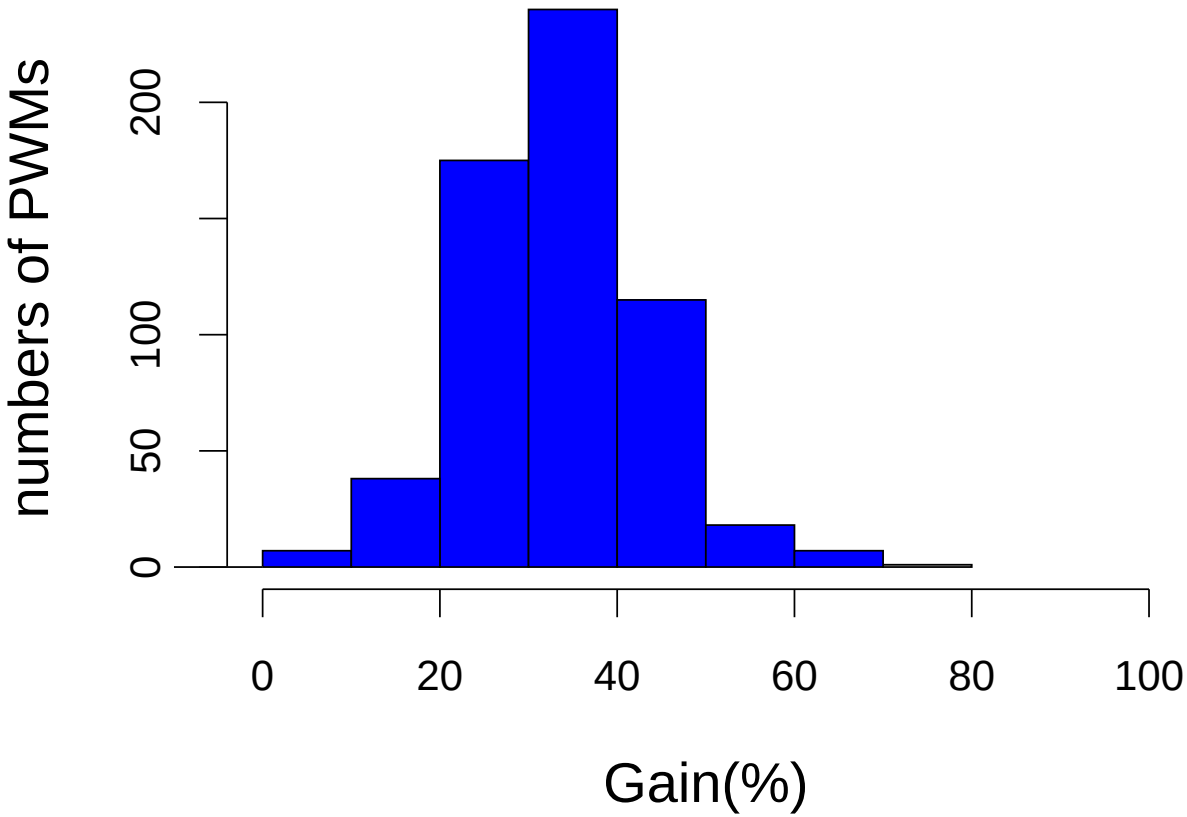
Maximum score of any suffix = 7.62

- -6.9 + 7.62 can never be greater than the score threshold 1
- Greater Lower Bound : the minimal score that should be reached to be able to have an occurrence of the PWM
- M a PWM of length n, α a score threshold, i a position in $\{1, ..., n\}$

$$GLB(M, i, \alpha) = \alpha - \sum_{j=i+1}^{n} \max_{x \in \Sigma} M(j, x)$$

Optimization 1 : Pruning PWMs

Evaluation of the pruning strategy



602 PWMs of Jaspar and Transfac vertebrate databases

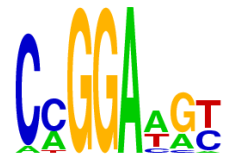
Optimization 2 : Preprocessing PWMs

- Databases of PWMs are permanent objects
- Store all possible scores in an index table

  a word ▷ index hashtable ▷ every score for every word for each PWM

  Index size does not fit in memory !
- Index slices
  cut PWMs into slices and compute subtable slices,
  using additivity property of score

Optimization 2 : Preprocessing PWMs

initial PWM  = submatrix 1  + submatrix 2 

A	C	G	T
-1.8	1.9	-2.3	-2.3
0.4	1.26	-2.3	-1.8
-2.3	-2.3	2	-2.3
-2.3	-2.3	2	-2.3
2	-2.3	-2.3	-2.3

0	...
...	...
...	...
...	...
104	5.46
...	...
$4^5 - 1$	...

ACGGA : 5.46

+

A	C	G	T
1.1	-0.94	-2.3	0.4
0.11	0.07	1.42	-2.3
-1.8	0.4	0	1.1

0	...
49	0.91
...	...
$4^3 - 1$	...

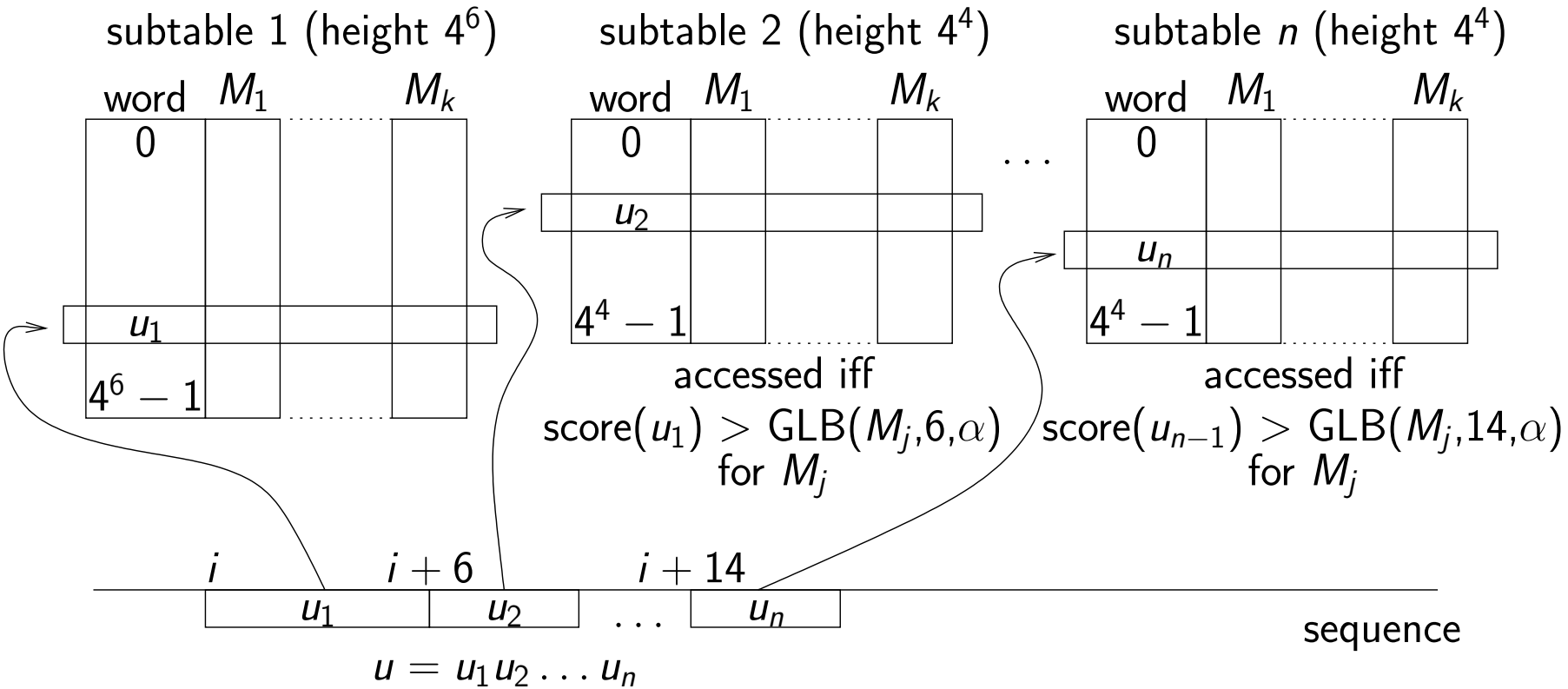
TAC : 0.91

=

score : 6.37

Optimization 2 : Subtables for multiple matrices

computing the score of u with n subtables for k matrices



Optimization 2 : Optimal index table construction

Input : a set of PWMs $\mathcal{M}$, the maximal memory space capacity e

Objective : Minimize the average number of accesses to the index $\phi(j, e)$ taking into account

- the pruning strategy
- the distribution of size of PWMs t_n
- the cardinality of the alphabet σ (Bernoulli model)

Output : Number and sizes of subtables

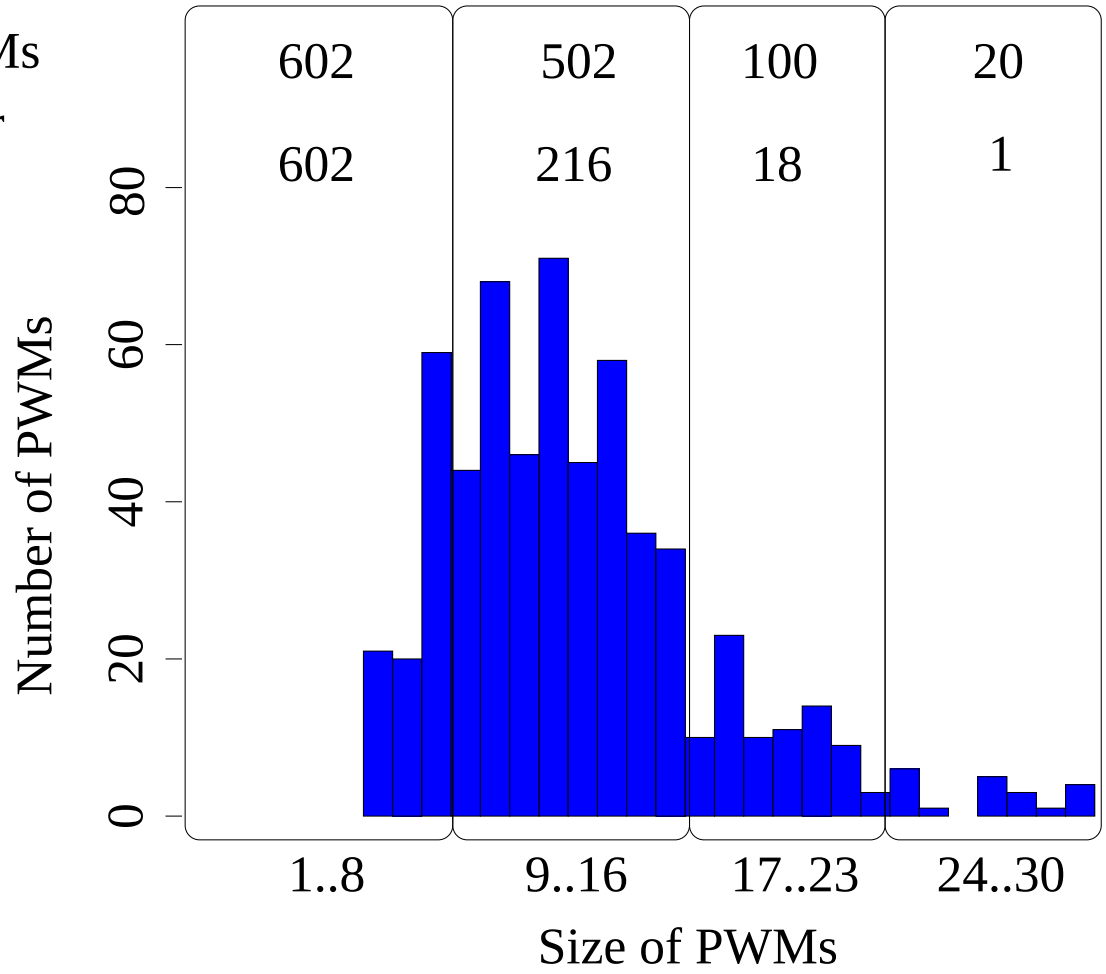
$$\phi(0, e) = 0 \text{ if } e \geq 0, +\infty \text{ else}$$

$$\phi(j, e) = \min_{0 \leq i < j} \{\phi(i, e - \sigma^{j-i} \sum_{y=i+1}^{j} t_y) + \sum_{M \in \mathcal{M}} Pvalue(M[1..i], GLB(M, i, \alpha))\}$$

ϕ can be computed by dynamic programming

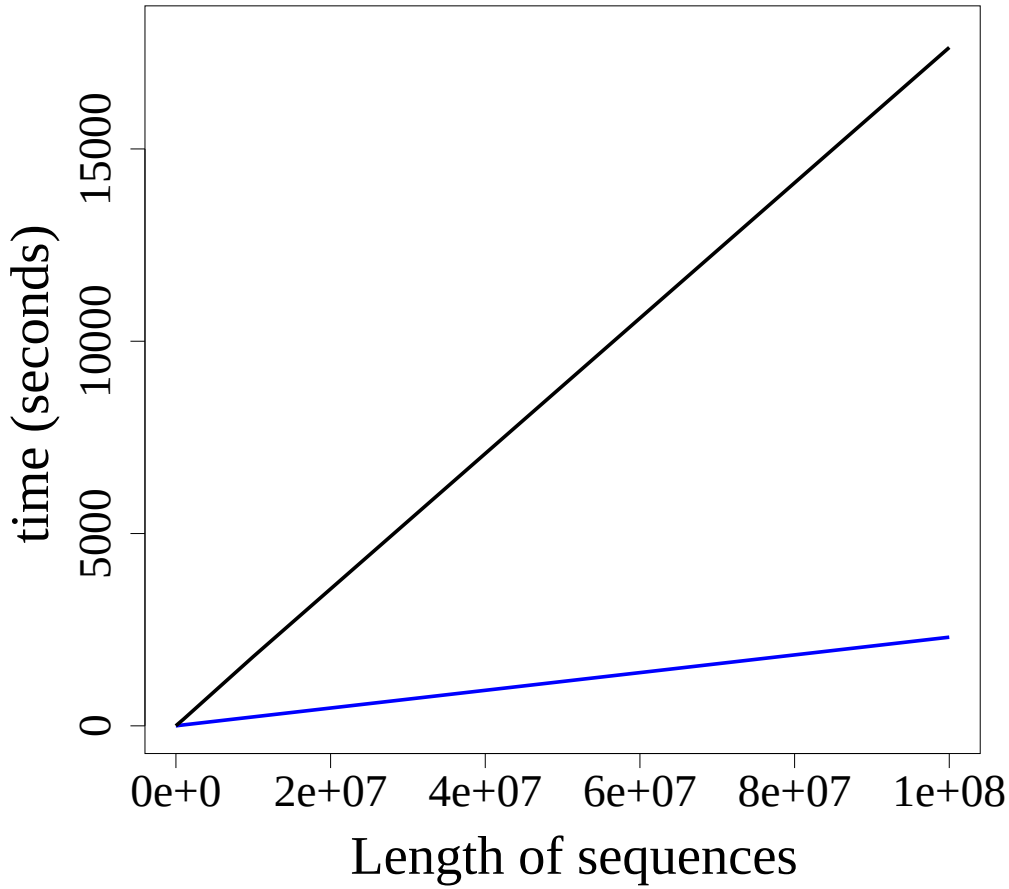
Optimization 2 : Results

	1..8	9..16	17..23	24..30
Number of PWMs	602	502	100	20
Average number of accesses	602	216	18	1



602 PWMs of Jaspar and Transfac vertebrate databases
Capacity memory $<$ 256 MB

Optimization 2 : Results



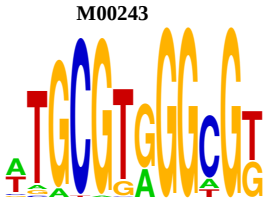
602 PWMs
Jaspar and Transfac

Brute force algorithm
TFMscan algorithm

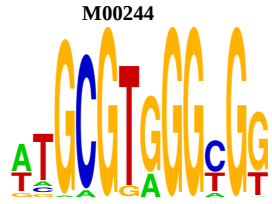
Human genome

- TFMscan : 4h
- Brute force : 32h

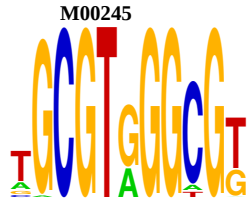
PWM Matching Problem for Similar Matrices



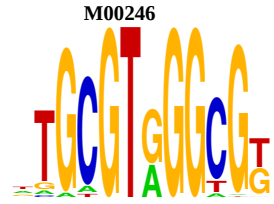
Egr-1



NGFI-C



Egr-3



Egr-2

PWMs naturally gather in families

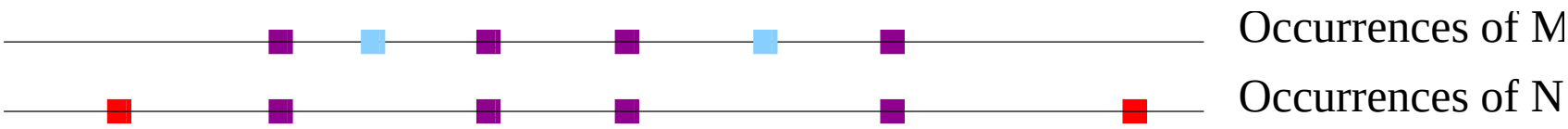
- Homology between transcription factors
- Redundancy between databases

How can we take advantage of this similarity ?

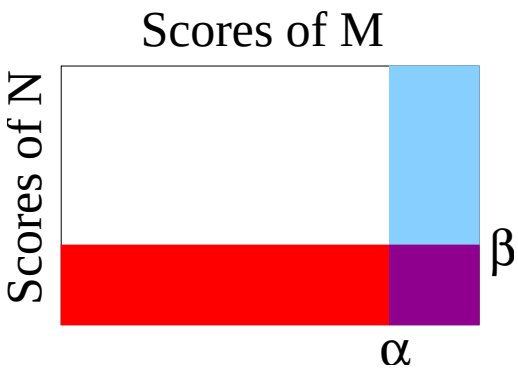
How to quantify similarity between PWMs ?

How to quantify similarity between PWMs ?

Measure the proportion of overlapping occurrences for 2 PWMs



- 2 PWMs M and N , 2 score thresholds α and β
- Probability for a word to be a common occurrence for both PWMs



$$P(M, N, \alpha, \beta) = I(length(M), \alpha, \beta)$$

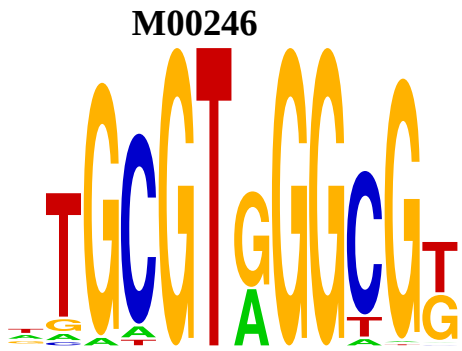
$$I(0, s1, s2) = \begin{cases} 1 & \text{if } s1 \geq 0 \text{ and } s2 < 0 \\ 0 & \text{otherwise} \end{cases}$$

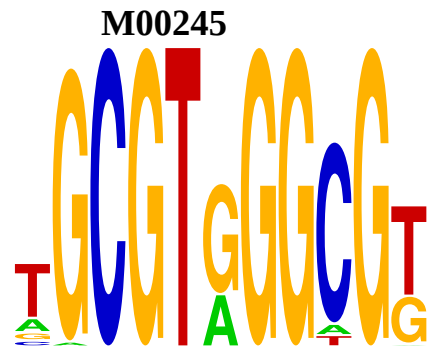
$$I(0, s1, s2) = \sum_{x \in \Sigma} I(i-1, s2-M(x, i), s2-N(x, i))$$

- Implementation by dynamic programming

Example of common occurrences between 2 PWMs

M00246 and M00245 (Transfac database)



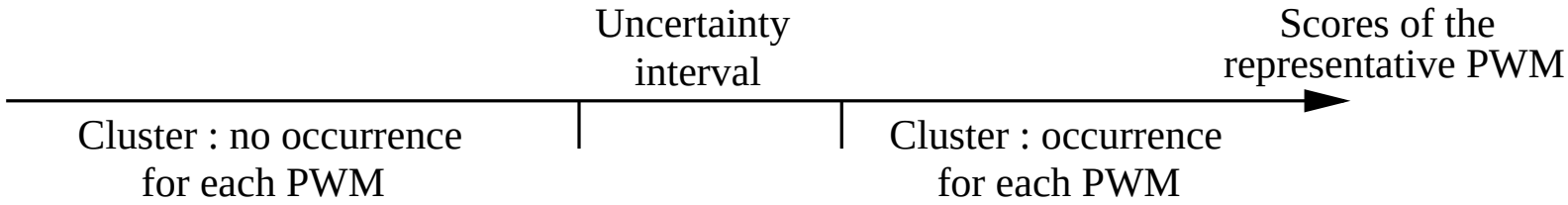


Proportion of overlapping occurrences between M00246 and M00245 with a p-value threshold of e^{-5}

	Occurrence	**No occurrence**
Common	83%	99%
Not common	17%	1%

Speed-up PWM matching problem with clusters

- Grouping PWMs (hierarchical clustering)
- Choice of a representative PWM (Arithmetical mean, minimum, ...)
- Lossless filtering



  - Uncertainty interval : individual computation for each representative PWM
  - Effective for very similar PWMs
- Approximate algorithm
  - Score of the representative PWM
  - Control prediction error rate

Conclusion

- Efficient algorithm for the PWM matching problem of a large set of PWMs
- Exact relastionship between occurrences of PWMs
- Analysis combining pattern matching and discrete probabilities (distribution of scores)