Перекрытия
в
множествах слов

.

План занятия

- 1. Вхождения паттерна в текст. Вероятность встретить не менее s вхождений в случайном тексте длины не менее N.

  [Бернулли. НММ - самостоятельно]
- 2. Решение с помощью автомата Ахо-Корасик. Что можно придумать?
- 3. Перекрытия. Ахо-Корасик. Кнут-Моррис-Пратт.
- 4. Граф перекрытий.
- 5. Рекурсия через дальние и близкие множества.
- 6. Оценка сложности – теория
- 7. Оценка сложности – таблицы.
- 8. Раскрытие секрета – вычитание вероятностей.

1. Введение

Pattern (motif) – set of words specifying functional fragments in biological sequences

***Examples*:**

• TATA-box **TATA(A/T)A(A/T)** – 4 words of length 7

•Consensus of transcription factor binding site Antp (Drosophila) **ANNNNCATTA –** 256 words of length 10

Example of occurrences of pattern **TATA(A/T)A(A/T)**

Постановка задачи

Дано: Алфавит **A**

Паттерн $\boldsymbol{H = \{h_1,\ldots,h_k\}}$, длины m

Длина текста L

Кол-во вхождений s

Вероятностное распределение на A^L

Найти: Вероятность того, что случайный текст длины L содержит не менее s вхождений паттерна H

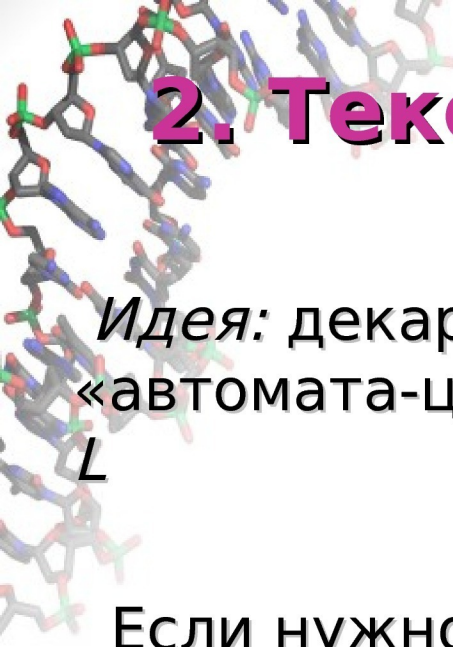
Термин: *P-значение [P-value]*

P-value

P-value *is the probability to find at least S occurrences of words from a pattern H in a random sequence of length N generated according to a given probability model.*

P-значение – мера перепредставленности паттерна в данном фрагменте текста.

Если P-значение *p* мало (= -log p велико),
то это неспроста*!*



2. Текст длины *L* содержит *w* € *H*, *H* – к.а.

Идея: декартово произведение автомата *Aut(H)* и «автомата-цепочки», который допускает все слова длины *L*

Если нужно не менее *s* вхождений – «помним» сколько вхождений уже было (к-во состояний умножается на *s).*

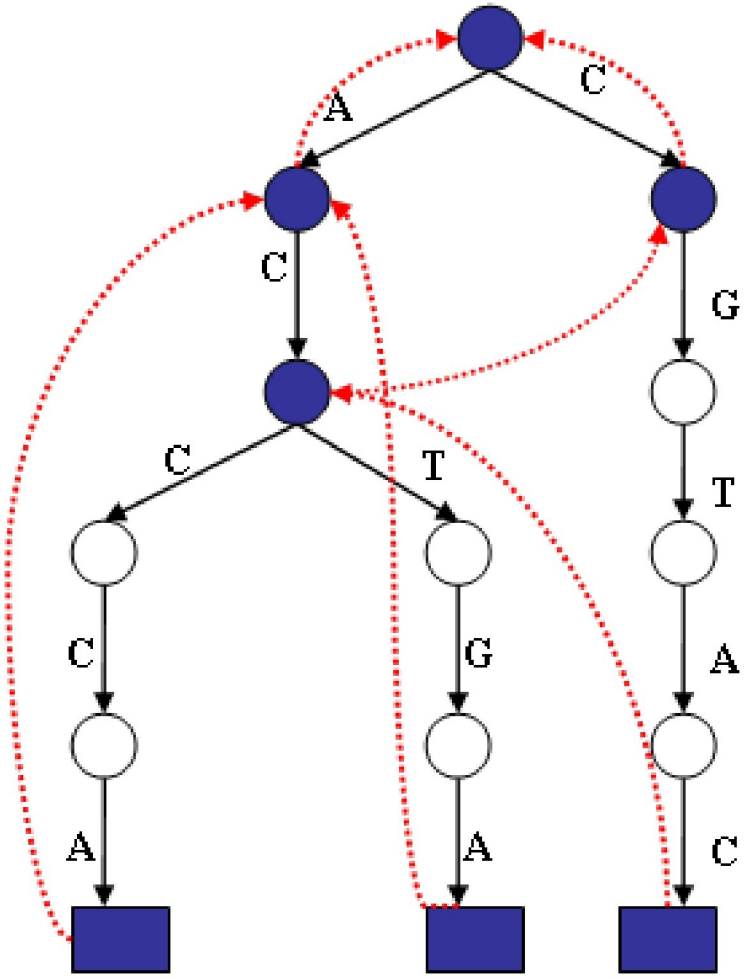
Если нужна вероятность – декартово умножаем на граф HMM, потом решаем задачу Больцмана.

Стоки графа соответствуют допускающим состояниям автомата *Aut(H) .*

Plan

I. Overlap graph
II. Text sets
III. Algorithm SufPref
IV. Complexities
V. Comparison with other programs

Aho-Corasick trie

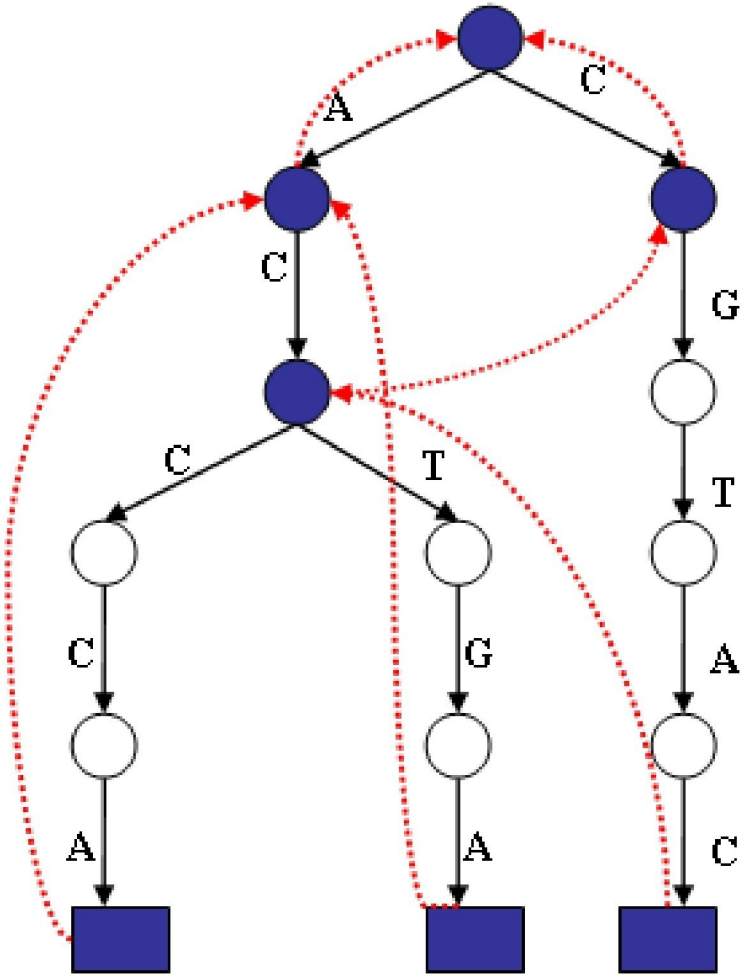


H:

CGTAC
ACCCA
ACTGA

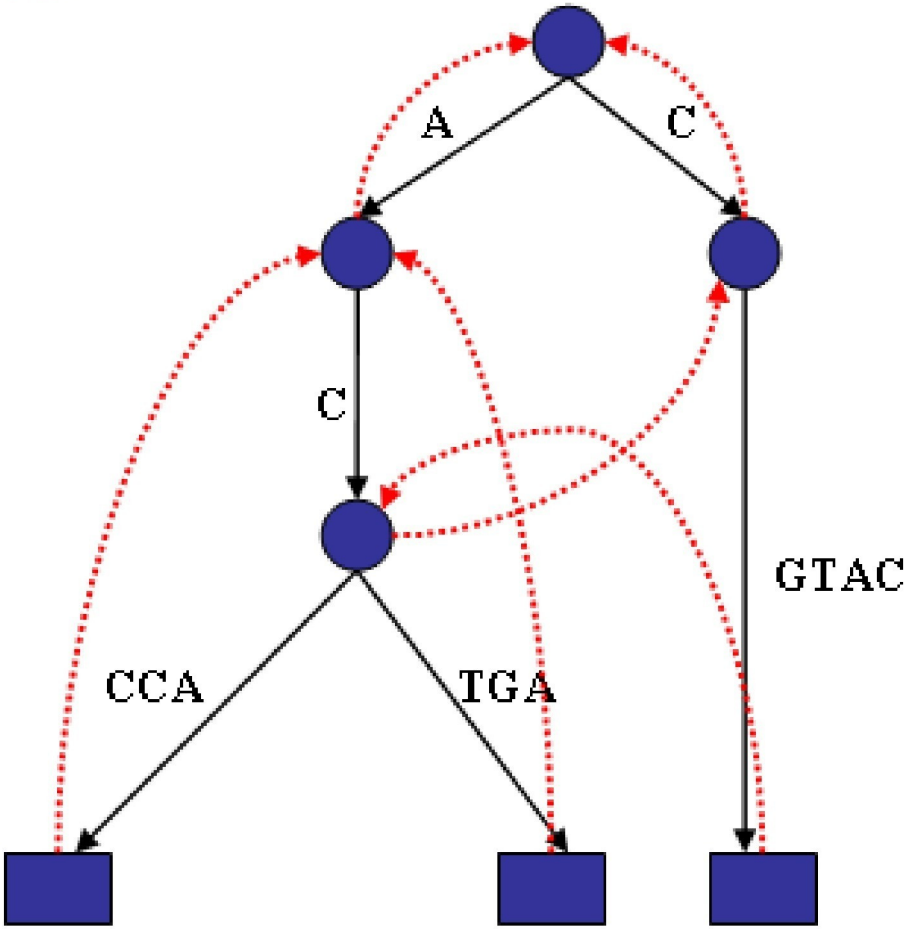
$OV(H) = \{\varepsilon, A, AC, C\}$

Алгоритм KMP



Time: ~L,
а не ~mL

Overlap graph
ACCCA, ACTGA, CGTAC

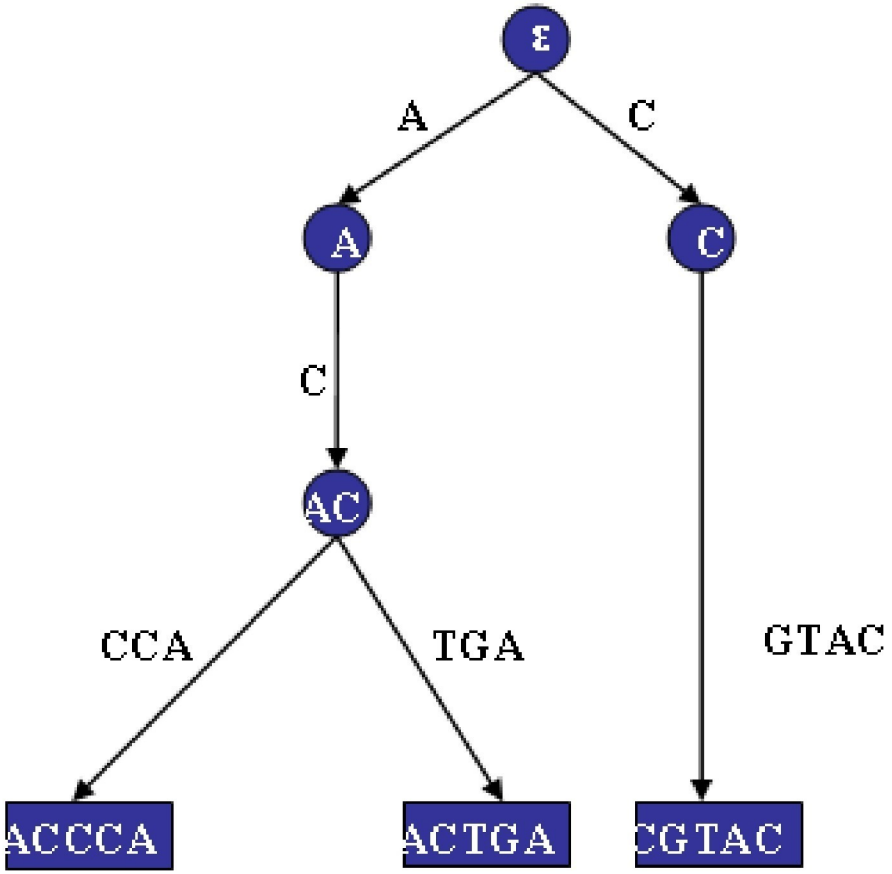


H:

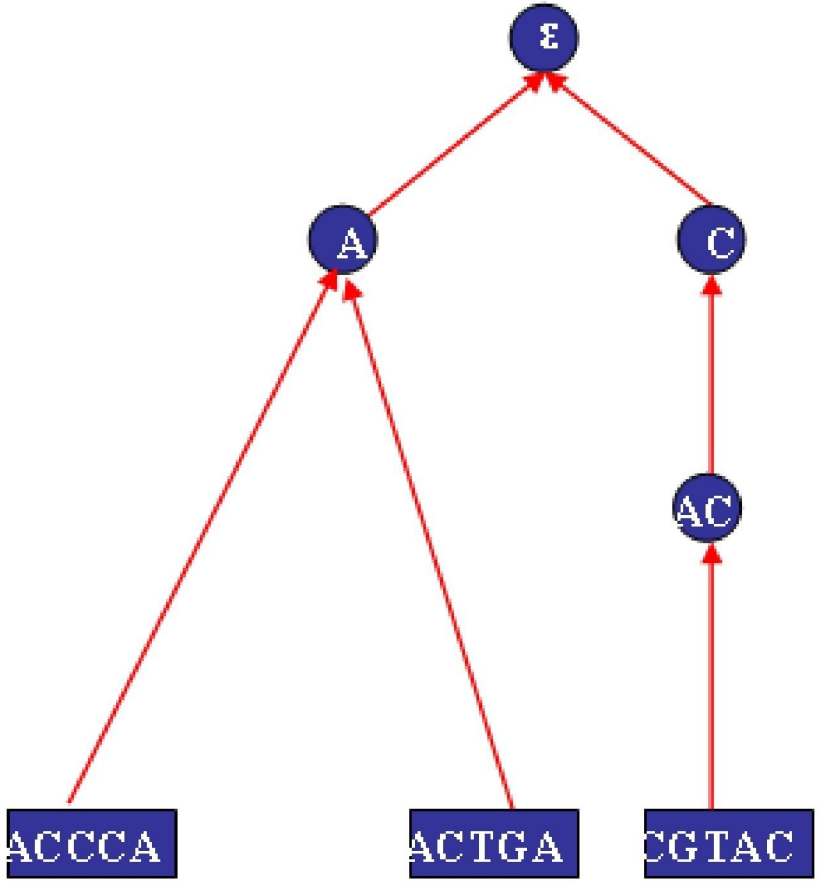
CGTAC
ACCCA
ACTGA

Regnier M., Kirakosian Z., Furletova E.I., Roytberg M.A. **A word counting gpaph.** // *London algorithmics 2008: theory and practice*. 2009. P. 10-43.

Overlap graphs
ACCCA, ACTGA, CGTAC



Left overlap graph (LOG)



Right overlap graph (ROG)

R and E-sets

$E(n,s,h) = \{T \in V_n \mid T$ contains at least s occurrences of H & & T ends with $h\}$

$$R(n,s,h) = E(n,s,h) \setminus E(n,s+1,h)$$

Computation of E-sets probabilities is the main part of the algorithm!

Текстовые множества

$B(n,s) = \{T \in V^n |\ T$ contains at least s occurrences $H\}$

P-value is $Prob(B(N,S))$

$R(n,s,h) = \{T \in V^n |\ T$ contains exactly s occurrences of H &
& T ends with $h\}$

Proposition Let $n > m$ then

$$B(n,s) = B(n\text{-}1,s)\cdot V \cup R(n,s,H)$$

Proof. Two cases are possible:

1) prefix of T of length n-1 contains at least s occurrences of H

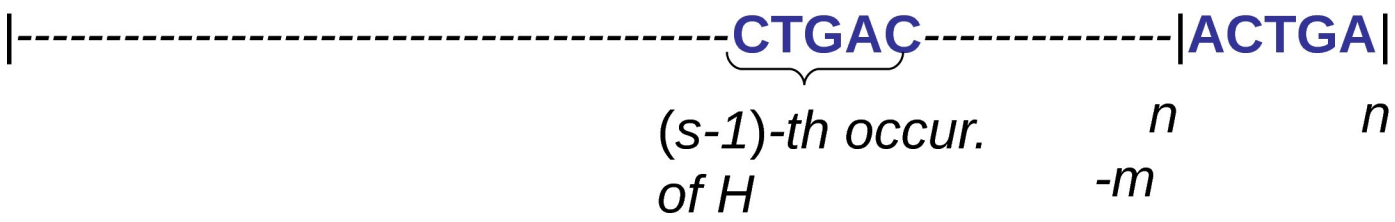
2) T ends with s-th occurrence of H

E- sets

$E(n,s,h) = \{T \in V^n \mid T$ contains at least s occurrences of H &
& T ends with $h\}$

Example. Let H={ACCCA, ACTGA, CGTAC}. Consider $E(n,s,$ ACTGA)

Case 1: (s-1)-th occurrence of H does not overlap h = ACTGA

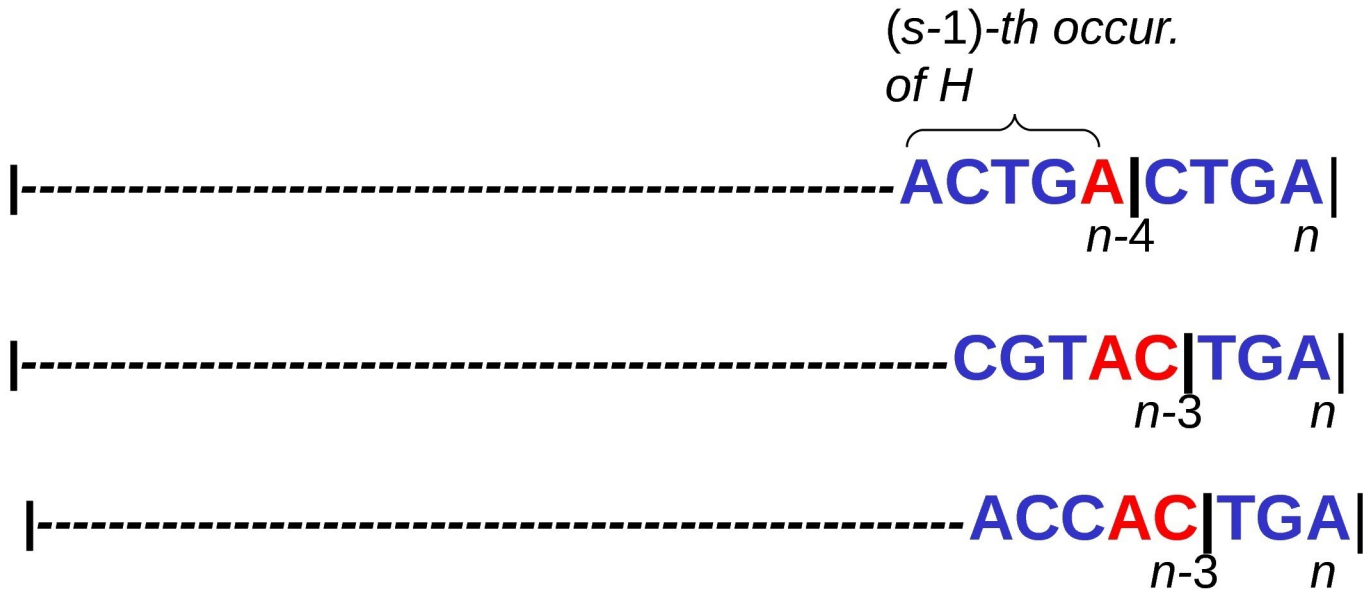


$B(n-m,s-1)\cdot$ACTGA

E- sets

H: CGTAC
ACCCA
ACTGA

Case 2: (*s*-1)-th occurrence overlaps *h*=ACTGA



$R(n\text{-}4,s\text{-}1,H(\text{A}))\cdot\text{CTGA} + R(n\text{-}3,s\text{-}1,H(\text{AC}))\cdot\text{TGA}$

H(A), *H*(AC) – sets of pattern words ending with *A* and *AC*

E-sets. Case 2

Case 2: (*s*-1)-th occurrence overlaps *h*=ACTGA

$$R(n-4,s-1,H(A))\cdot CTGA + R(n-3,s-1,H(AC))\cdot TGA =$$

$$= R(n-4,s-1,H(A))\cdot C\cdot TGA + R(n-3,s-1,H(AC))\cdot TGA =$$

$$= \underbrace{\left(R(n-4,s-1,H(A))\cdot C + R(n-3,s-1,H(AC))\right)}_{D(n-3,s-1,AC)}\cdot TGA$$

$\underbrace{R(n-4,s-1,H(A))}_{D(n-4,s-1,A)}$

R-sets induction

$R(n,s,H(w)) = \{T \in V^n \mid T$ contains exactly s occurrences of H & & T ends with h from $H(w)\}$

H: CGTAC, ACCCA, ACTGA

- $R(n,s,H(A)) = R(n,s,ACTGA) + R(n,s,ACCCA)$
- $R(n,s,H(AC)) = R(n,s,CGTAC)$
- $R(n,s,H(C)) = R(n,s,H(AC))$

$H(A)$, $H(AC)$ and $H(C)$ – sets of pattern words ending with A, AC and C

E-sets induction

$E(n,s,h) = \{T \in V^n \mid T$ contains at least s occurrences of H & & T ends with $h\}$

H: CGTAC
ACCCA
ACTGA

E-sets induction

- $D(n\text{-}4,s\text{-}1,A) = R(n\text{-}4,s\text{-}1,H(A))$
- $D(n\text{-}3,s\text{-}1,AC) = D(n\text{-}4,s\text{-}1,A)\cdot C + R(n\text{-}4,s\text{-}1,H(AC))$
- $E(n,s,ACTGA) = B(n\text{-}m,s\text{-}1)\cdot ACTGA + D(n\text{-}3,s\text{-}1,AC)\cdot TGA$

$H(A)$, $H(AC)$ and $H(C)$ – sets of pattern words ending with A, AC and C

Algorithm

Input: alphabet A, probabilities model, pattern H, text length N, minimal number of occurrences S

Output: $P(B(N,S))$

I. Pre-processing

Overlap graph creation, computation of constant values and probabilities of text sets for $n \leq m$

II. Main loop

For all $n = 1,\ldots,N$

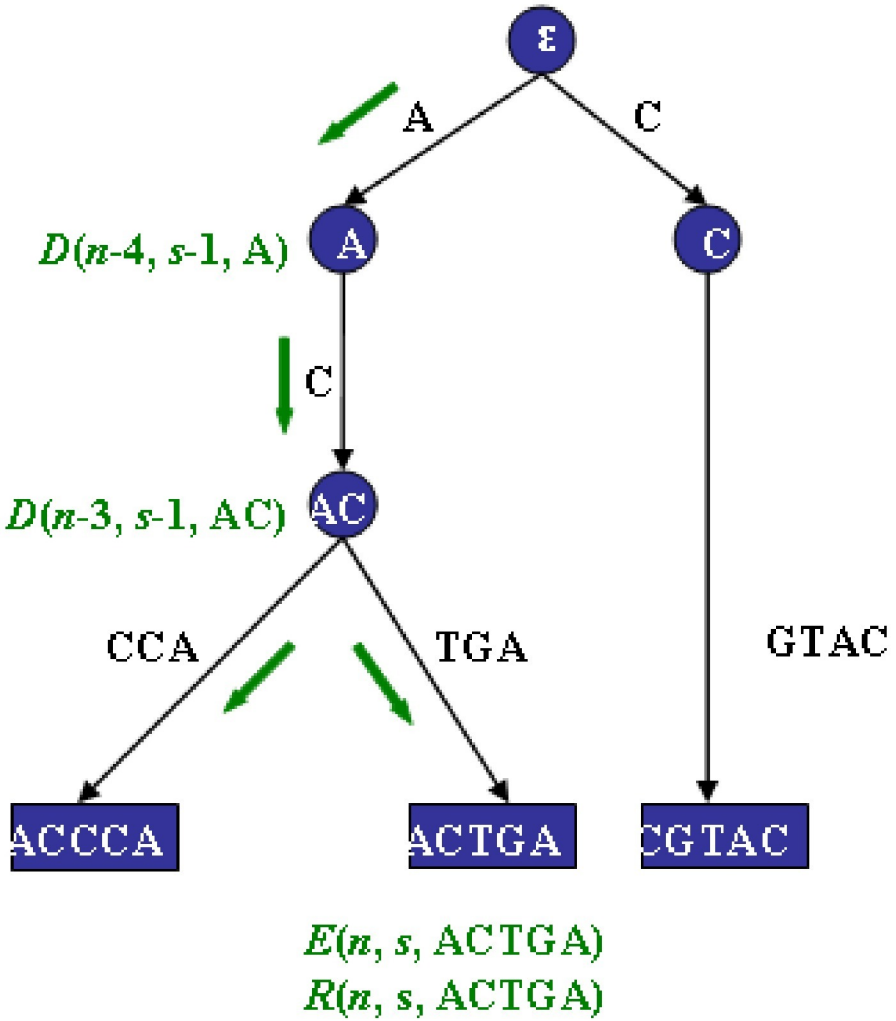
For all $s = 1, \ldots., S$

- Computation of $P(B(n\text{-}m,s\text{-}1))$
- Depth-first traversal of LOG. Computation of $P(E(n,s,h))$ and $P(R(s,n,h))$ (h in H)
- Bottom-up traversal of ROG. Computation of $P(R(s,n,H(w)))$ (w - overlap)

III. Post-processing

Computation of $P(B(N,S))$

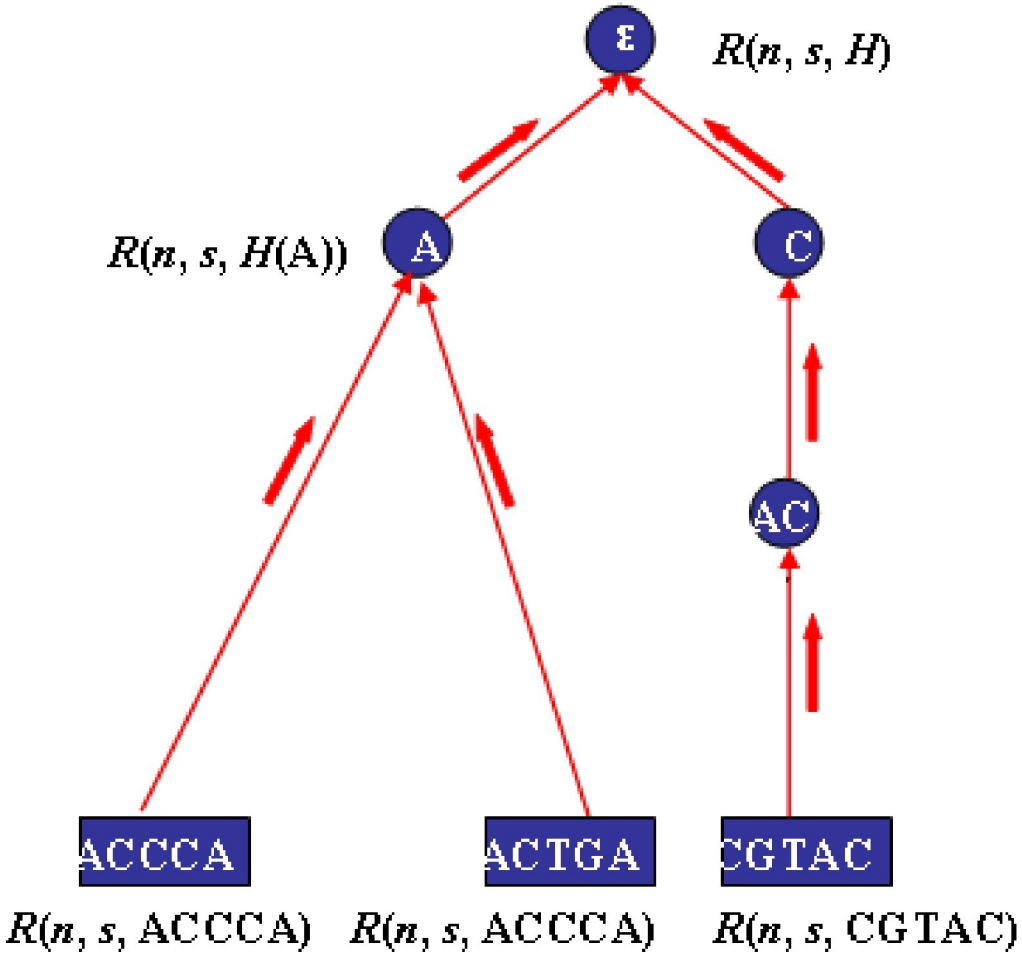
Depth-first traversal



E-sets induction

- D(n-4,s-1,A) = $R(n\text{-}4,s\text{-}1,H(\text{A}))$
- $D(n\text{-}3,s\text{-}1,\text{AC}) = D(n\text{-}4,s\text{-}1,\text{A})\cdot C +$
  $+ \; R(n\text{-}4,s\text{-}1,H(\text{AC}))$
- $E(n,s,\text{ACTGA}) = B(n\text{-}m,s\text{-}1)\cdot\text{ACTGA} +$
  $+ D(n\text{-}3,s\text{-}1,\text{AC})\cdot\text{TGA}$

Bottom-up traversal



R-sets induction

- $R(n,s,H(A)) = R(n,s,ACTGA) + R(n,s,ACCCA)$
- $R(n,s,H(AC)) = R(n,s,CGTAC)$
- $R(n,s,H(C)) = R(n,s,H(AC))$

Complexity
Bernoulli models

Memory needed for data stored in overlap nodes

Memory needed to store Aho-Corasick trie in preprocessing stage + data in leaves

space: $O(S \times m \times |OV(H)|) + m \times |H|)$

time of overlap graphs traversals

time: $O(N \times S \times (|H| + |OV(H)|)))$

|*OV*(*H*)| - number of overlaps, ***N*** – text length, ***S*** – minimal number of occurrences, ***m*** – length of words from *H*, **|*H*|** – number of words in *H*.

Complexity
Markov and hidden Markov models

Markov models of order *K*

space: $O(S \times m \times (K \times |V|^{K+1} + |OV(H)|) + m \times |H|)$

time: $O(N \times S \times (K \times |V|^{K+1} + |OV(H)|) + |H|))$

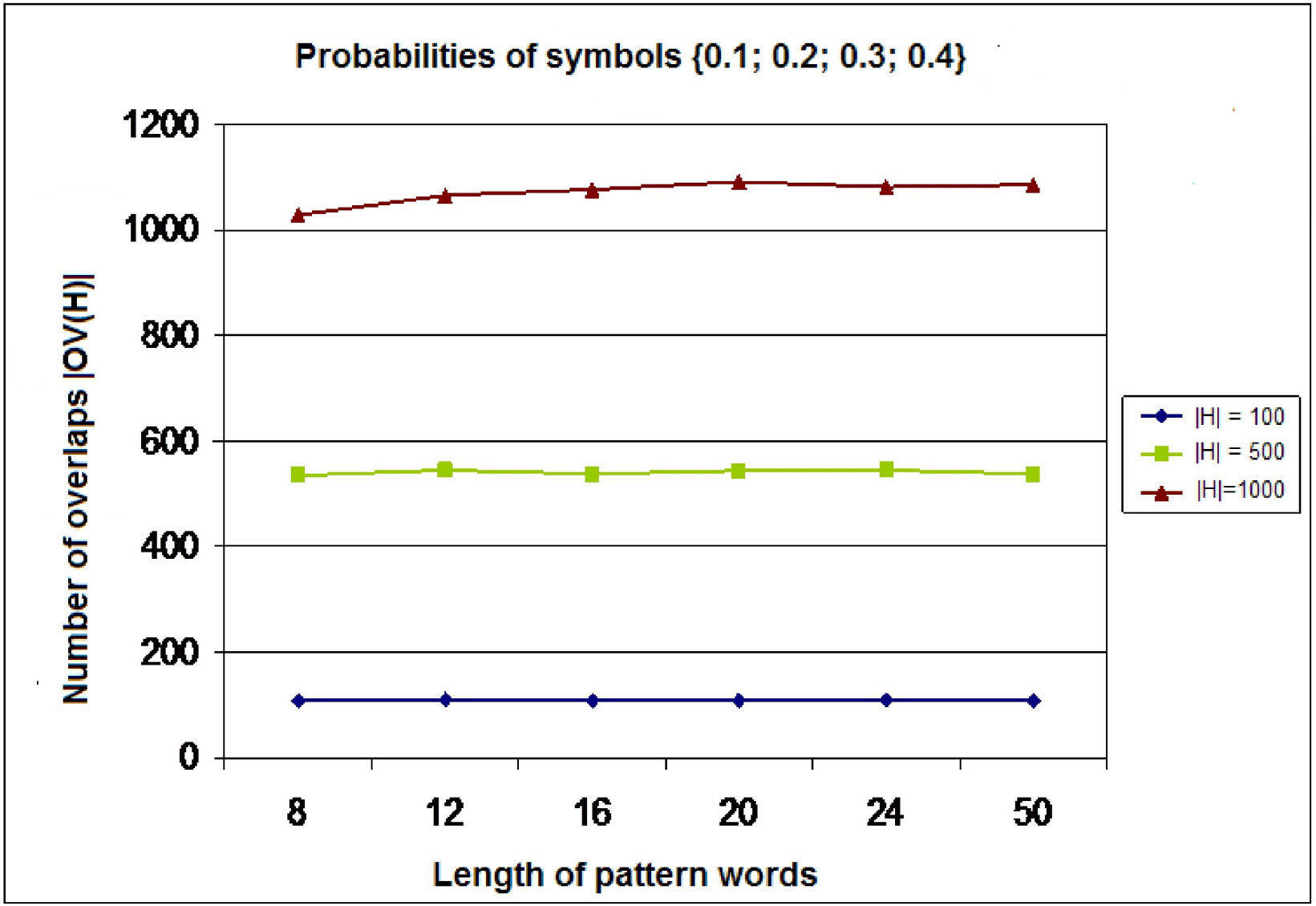
Hidden Markov models (HMM)

space: $O(|Q|^2 \times (|OV(H)|) + |H|) + |Q| \times S \times m \times |OV(H)|) + m \times |H|)$

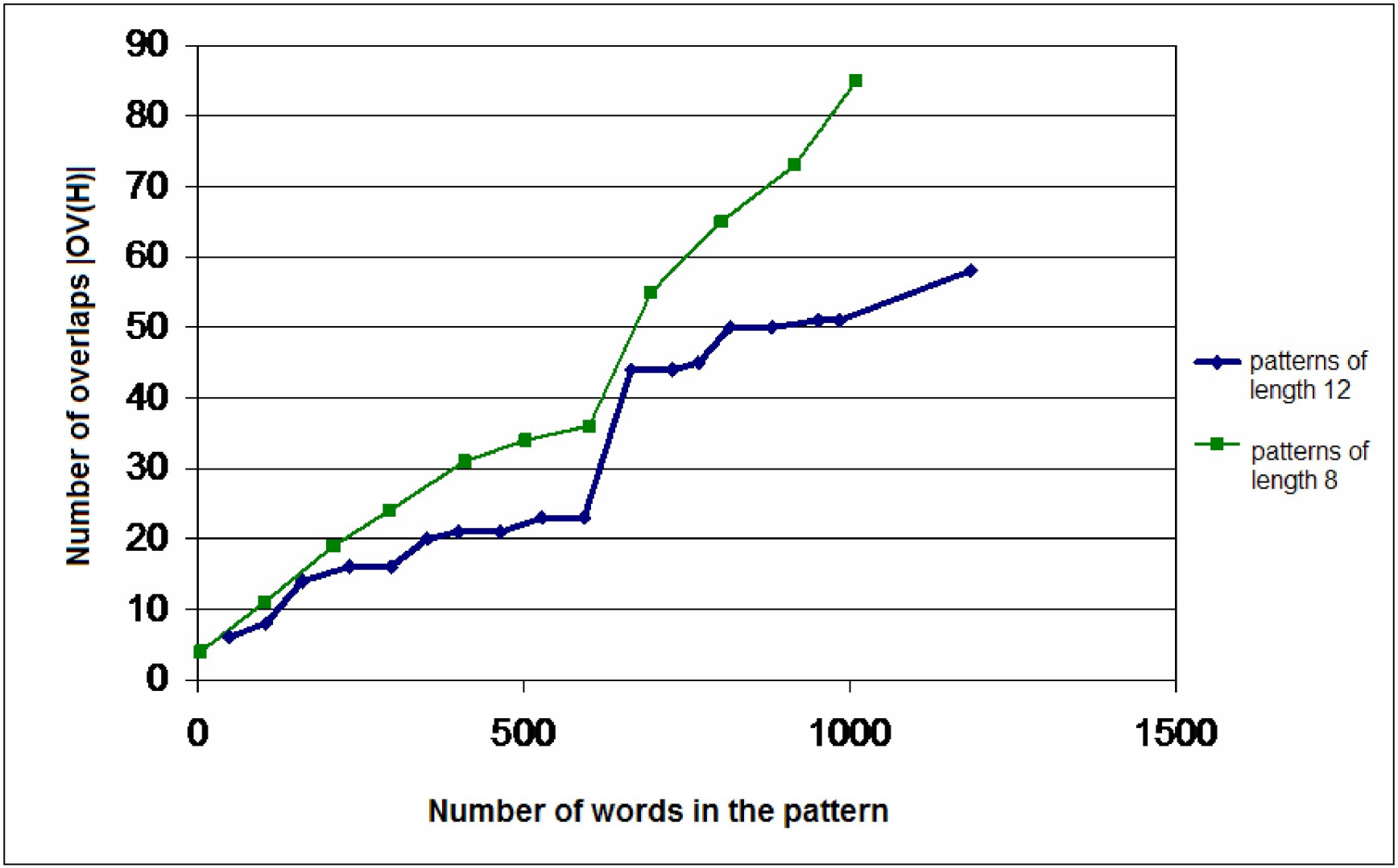
time: $O(N \times S \times |Q|^2 \times (|OV(H)| + |H|))$

|*V*| - alphabet size, **|*Q*|** - number of HMM states
|*OV*(*H*)| - number of overlaps, ***N*** – text length, ***S*** – minimal number of occurrences, ***m*** – length of words from *H*, **|*H*|** – number of words in *H*.

Number of overlaps
Random patterns



Number of overlaps
Patterns of lengths 8 and 12 given by PWM



Improvements of algorithm
classes of pattern words

Class $H^*(v,w)$ – set of words from H having the same maximal overlap prefix v and maximal overlap suffix w

Example:

$H^*(AC,A) = \{ACCCA, ACTGA\}$

$H^*(C,AC) = \{CGTAC\}$

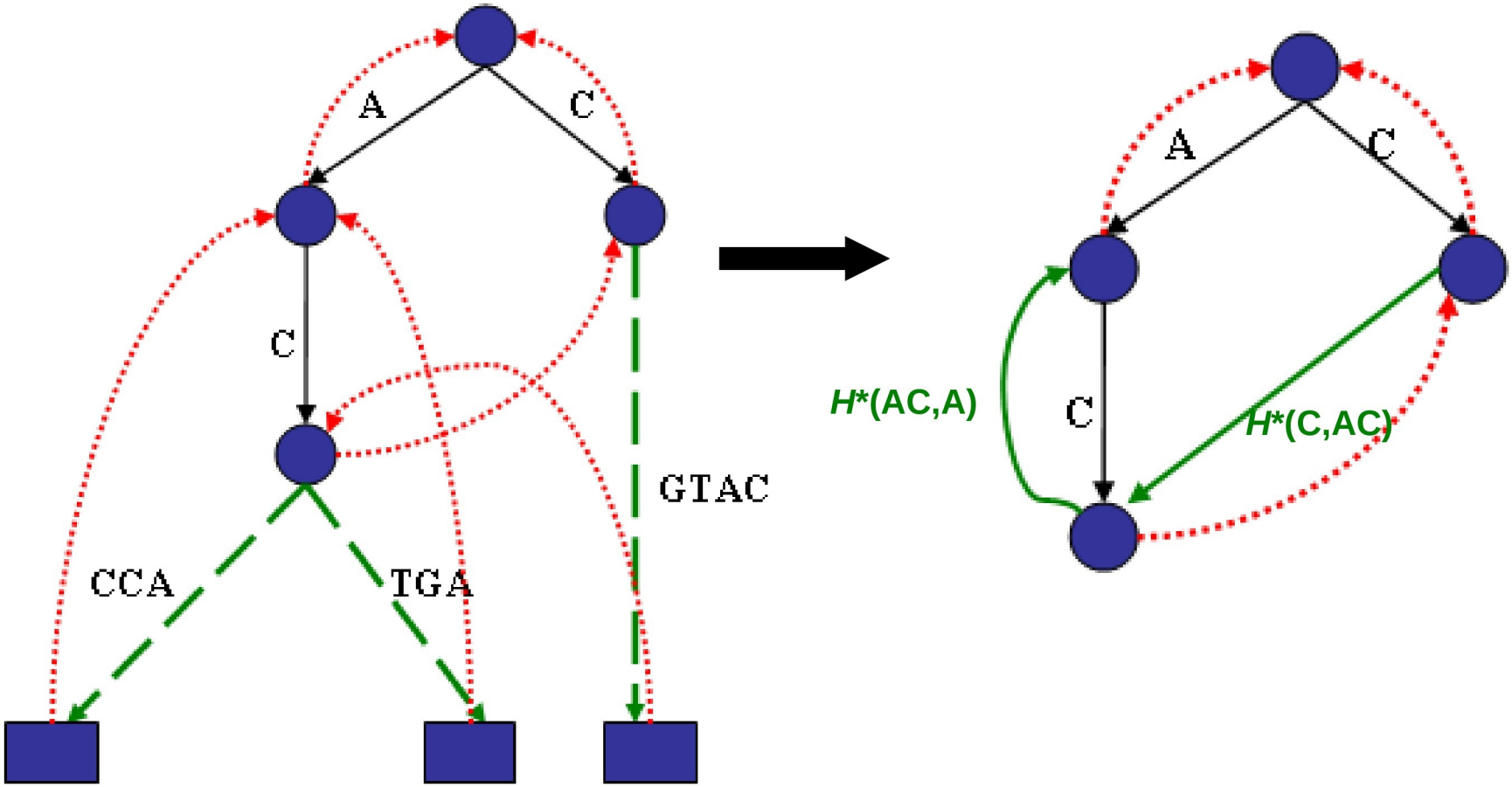
H:

CGTAC
ACCCA
ACTGA

$$R(n,s,h) \rightarrow R(n,s,H^*(v,w))$$

$$E(n,s,h) \rightarrow E(n,s,H^*(v,w))$$

Improvements of algorithm



Deep edge **(*v*,*w*)** corresponds to class ***H**(*v*,*w*)***

Implementation

The algorithm SufPref was implemented as a C++ program. It can be used both as Web-server and a stand alone program for Linux and Windows.

The program is available at

http://server2.lpm.org.ru/bio/online/sf/

Comparison of programs SufPref and AhoPro*

Parameters: Text length - 1000; Number of occurrences - 10; patterns are given by a matrix PWM described Drosophila genes of length 12; probabilities: Bernoulli model and Markov model of order 1 (the models described uniform distribution of letters)

Bernoulli model:

- SufPref is faster in 4-20 times;
- space of SufPref is smaller in 1-5 times.

Markov model of order 1:

- SufPref faster in 4-15 times;
- space of SufPref smaller in 1-5 times.

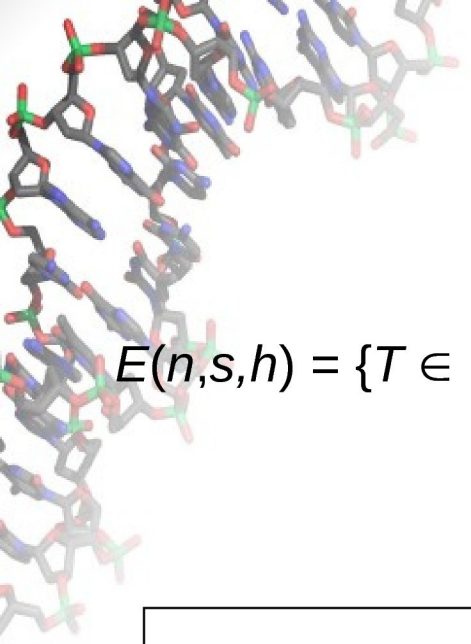
$\|H\|$	$\|OV(H)\|$	NAC	Prob Distrib	Time (seconds)			Space (megabytes)		
				SP	Aho	Aho/SP	SP	Aho	Aho/SP
5272	183	11325	Bern	0,46	9,47	20,58	2,38	6,75	2,83
47448	987	87341	Bern	5,00	83,52	16,71	9,15	45,68	4,99
91432	1663	157613	Bern	10,90	156,99	1,47	15,26	81,52	5,34
170032	3563	283237	Bern	22,85	294,66	12,89	26,20	145,71	5,56
467056	14428	766549	Bern	73,90	896,74	12,13	68,34	392,95	5,75
5272	183	11325	Mark	0,63	9,57	15,22	2,44	6,80	2,79
47448	987	87341	Mark	6,05	84,52	13,96	9,18	45,73	4,98
91432	1663	157613	Mark	12,89	157,18	12,20	15,34	81,56	5,32
170032	3563	283237	Mark	28,32	297,45	10,50	26,32	145,76	5,54
467056	14428	766549	Mark	101,54	904,54	8,91	68,76	392,74	5,71

***** Boeva V, et al. **Exact p-value calculation for heterotypic clusters of regulatory motifs and its application in computational annotation of cis-regulatory modules.** *Algorithms for molecular biology* 2007, 2(13):25 page

Publications

1. Regnier M., Kirakosian Z., Furletova E.I., Roytberg M.A. **A word counting gpaph.** *// London algorithmics 2008: theory and practice*. 2009. P. 10-43.

2. Regnier M., Furletova E.I., Yakovlev V.V., Roytberg M.A. **Pattern occurrences P-values, Hidden Markov Models and Overlap Graph** // The paper is submitted to the journal "*Algorithms for molecular biology*".



В чем секрет?
R and E-sets

$E(n,s,h) = \{T \in V^n \mid T$ contains at least s occurrences of H & & T ends with $h\}$

$$R(n,s,h) = E(n,s,h) \setminus E(n,s+1,h)$$

Computation of E-sets probabilities is the main part of the algorithm!

Вероятностные модели

•Модель Бернулли: вероятность встретить букву в некоторой позиции случайной последовательности не зависит ни от позиции, ни от предшествующих букв

•Марковская модель порядка K: вероятность встретить букву в некоторой позиции случайной последовательности зависит от букв, стоящих в K предшествующих позициях этой последовательности

•Скрытые Марковские модели (СММ): задается конечно-автоматным генератором вероятностей $G = <Q, Q^0, \pi, \delta>$, где

- Q – множество состояний;

- Q^0 – множество начальных состояний;

- π: $Q \times A \times Q \rightarrow [0,1]$ – функция генерации вероятностей, $\pi(q', a, q)$ – вероятность, находясь в состоянии q', сгенерировать символ a и перейти в состояние q.

- δ: $Q^0 \rightarrow [0,1]$ – распределение вероятностей начальных состояний