Curs 5.

Tehnici algoritmice utilizate in bioinformatica
Problema identificarii sabloanelor

Biblio: cap. 4 din “An introduction to Bioinformatics algorithms”, N.Jones, P. Pevzner

Identificarea sabloanelor (motif discovery)

Contextul problemei:

- Genele sunt activate (exprimate) sau inactivate de către anumite proteine reglatoare; genele sunt considerate activate (exprimate) când produc un efect (de exemplu încep să sintetizeze proteine care distrug agentul patogen prezent în cazul unei infecții)
- Proteinele reglatoare se “leagă” în regiuni specifice ale genelor determinând stimularea sau inhibarea ARN polimerazei (controlând astfel procesul de transcripție)
- Aceste regiuni specifice sunt de regulă secvențe ADN scurte, numite motiv sau șablon (de exemplu la musculița de oțet genele implicate în reacțiile imunologice au la început o secvența de forma: TCGGGGATTTCC)
- Identificarea aceluiași șablon în mai multe gene cu caracter reglator sugerează prezența unei relații între acestea

Identificarea sabloanelor (motif discovery)

- Enunțul problemei:

  - Să se identifice șabloane frecvente (motive) într-un set de secvențe ADN
  - Exemplu: Se consideră cele 5 secvențe de mai jos.
    Exista vreun șablon comun în aceste secvențe ?

```
atgaccgggatactgataaaaaaaaggggggggcgtacacattagataa
acccctatttttgagcagatttagtgacctggaaaaaaaggggggggg
tgagtatccctgggatgacttaaaaaaaggggggtgctctcccgattt
gctgagaattggatgaaaaaaaggggggtccacgcaatcgcgaaccaa
tcaaaaaaaggggggggcgggaggctggttacgtagggaagccctaac
```

Identificarea sabloanelor (motif discovery)

- Enuntul problemei:

  - Să se identifice șabloane frecvente (motive) într-un set de secvențe ADN
  - Exemplu: Se consideră cele 5 secvențe de mai jos.
    Exista vreun șablon comun în aceste secvențe ?

atgaccgggatactgataaaaaaaaggggggggcgtacacattagataa
acccctatttttgagcagatttagtgacctggaaaaaaaaggggggggg
tgagtatccctgggatgacttaaaaaaaaggggggtgctctcccgattt
gctgagaattggatgaaaaaaaaggggggtccacgcaatcgcgaaccaa
tcaaaaaaaaggggggggcgggaggctggttacgtagggaagccctaac

Identificarea sabloanelor (motif discovery)

Dificultatea problemei constă în:

- Sablonul căutat nu este cunoscut (dacă ar fi cunoscut s-ar identifica ușor locația folosind algoritmi de căutare sau “pattern matching”)
- Nu se cunoaște locația (dacă s-ar cunoaște locația șablonului în cadrul fiecărei gene atunci ar fi relativ ușor de identificat)
- Sabloanele pot sa varieze ușor de la o genă la alta (pot fi alterate de mutații)

Apare întrebarea:

- Cum pot fi discriminate șabloanele sistematice de cele pur aleatoare ?

Identificarea sabloanelor (motif discovery)

- Exemple de șabloane care diferă între ele din cauza mutațiilor (mutația constă în acest caz în înlocuirea unei nucleotide cu o altă nucleotidă)

atgaccgggatactgataataaacagggggggggcgtacacattagataa
acccctatttttgagcagatttagtgacctggaaataaaagcgggggggg
tgagtatccctgggatgacttagaaaaaaagggtggtgctctcccgattt
gctgagaattggatgaaaaaaagggaggggtccacgcaatcgcgaaccaa
tcacaaaaaaaggggtggggcgggaggctggttacgtagggaagccctaac

Identificarea sabloanelor (motif discovery)

Problema e oarecum similară cu cea a descifrării codurilor secrete

Abordare simplistă:

- Se determină frecvența fiecărui simbol și se realizează identificarea cu tabelul de frecvențe cunoscut pentru limba din care provine textul inițial
- Nu furnizează rezultate bune

O abordare mai bună:

- Se determină frecvențe ale succesiunilor de litere (ex: 2, 3 litere)
- Identificarea secvențelor frecvente (de exemplu “the” în engleză) permite efectuarea unor inferențe asupra simbolurilor necunoscute

Identificarea sabloanelor (motif discovery)

- In baza acestei analogii rezultă că, la fel cum cunoașterea frecvenței cuvintelor/simbolurilor dintr-o limbă permite decodificarea unui text codificat, cunoașterea "distribuției" asociate unei secvențe poate fi utilă in "descifrarea ADN-ului"

- Aceasta motivează analiza frecvențelor unor șabloane în secvențele de nucleotide

Identificarea sabloanelor (motif discovery)

- Identificarea șabloanelor în secvențele biologice este mai dificilă decât decodificarea unui text bazat pe o limbă cunoscută întrucât:
  - Nu se cunoaște un dicționar complet de șabloane (așa cum există dicționare de cuvinte)
  - Limbajul “genetic” nu se bazează pe o gramatică bine definită
  - Doar o fracțiune mică din ADN corespunde unor șabloane reglatoare; aceasta fracțiune este “ascunsă” intr-un volum mare de date

Identificarea sabloanelor (motif discovery)

- Exemplu: 5 secvențe aleatoare în care este “implantat” câte un șablon

```
cctgatagacgctatctggctatccacgtacgtaggtcctctgtgcgaatctatgcgtttccaaccat

agtactggtgtacatttgatacgtacgtacaccggcaacctgaaacaaacgctcagaaccagaagtgc

aaacgtacgtgcaccctctttcttcgtggctctggccaacgagggctgatgtataagacgaaaatttt

agcctccgatgtaagtcatagctgtaactattacctgccacccctattacatcttacgtacgtataca

ctgttatacaacgcgtcatggcggggtatgcgttttggtcgtcgtacgctcgatcgttaacgtacgtc
```

- Informații suplimentare utile pentru identificare:
  - Sablonul are lungimea 8 și nu este alterat de mutații

Identificarea sabloanelor (motif discovery)

- Exemplu: 5 secvențe aleatoare în care este “implantat câte un șablon”

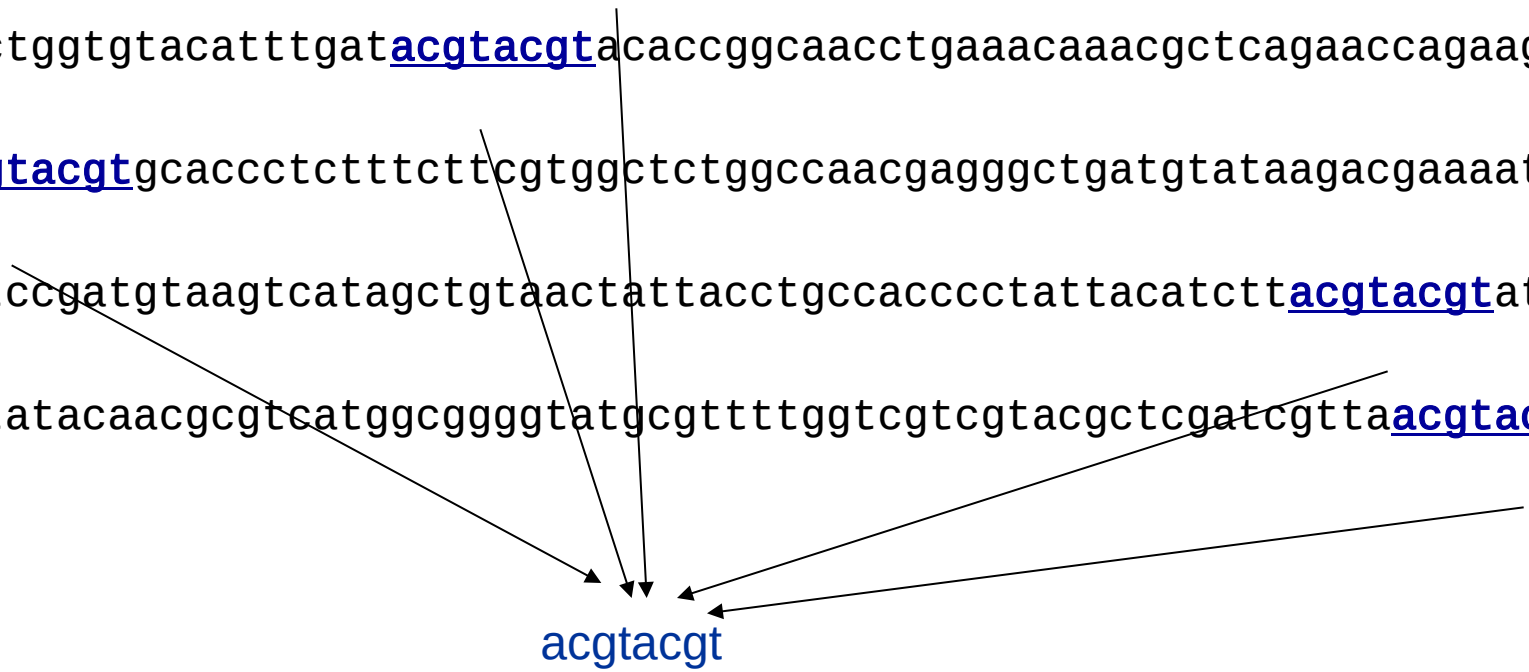
cctgatagacgctatctggctatcc**acgtacgt**aggtcctctgtgcgaatctatgcgtttccaaccat

agtactggtgtacatttgat**acgtacgt**acaccggcaacctgaaacaaacgctcagaaccagaagtgc

aa**acgtacgt**gcaccctctttcttcgtggctctggccaacgagggctgatgtataagacgaaaatttt

agcctccgatgtaagtcatagctgtaactattacctgccacccctattacatctt**acgtacgt**ataca

ctgttatacaacgcgtcatggcggggtatgcgttttggtcgtcgtacgctcgatcgtta**acgtacgt**c



Identificarea sabloanelor (motif discovery)

- Exemplu: varianta cu mutații

```
cctgatagacgctatctggctatccaGgtacTtaggtcctctgtgcgaatctatgcgtttccaaccat

agtactggtgtacatttgatCcAtacgtacaccggcaacctgaaacaaacgctcagaaccagaagtgc

aaacgtTAgtgcaccctctttcttcgtggctctggccaacgagggctgatgtataagacgaaaatttt

agcctccgatgtaagtcatagctgtaactattacctgccacccctattacatcttacgtCcAtataca

ctgttatacaacgcgtcatggcggggtatgcgttttggtcgtcgtacgctcgatcgttaCcgtacgGc
```

- Tipuri de mutație :
  - **Inlocuirea unei nucleotide** Eliminarea/Inserarea unei nucleotide
- Doar primul tip de mutație e considerat in algoritmii prezentați în acest curs

- Intrebări:
  - Dacă intervin mutații care ar fi de fapt șablonul reprezentativ?
  - Cum poate fi măsurată calitatea unui astfel de șablon reprezentativ?

Aliniere/ matrice profil/ consens

Ipoteză simplificatoare: presupunem că știm unde este localizat șablonul (în varianta cu mutații) în fiecare secvență, adică se cunoaște (s_1,s_2,…,s_t) unde s_i este punctul de start al șablonului în secvența i

- Se aliniază succesiunile de 8 nucleotide ce încep în pozițiile specificate
- Se construiește **matricea profil** (conține frecvențele de apariție ale fiecărei nucleotide pe fiecare poziție)
- Pe baza matricii profil se construiește **șablonul consensual** (pentru fiecare poziție se alege simbolul (nucleotida) ce apare cel mai frecvent în porțiunile aliniate)

Aliniere

```
a G g t a c T t
C c A t a c g t
a c g t T A g t
a c g t C c A t
C c g t a c g G
```

Matrice profil

A	3	0	1	0	3	1	1	0
C	2	4	0	0	1	4	0	0
G	0	1	4	0	0	0	3	1
T	0	0	0	5	1	0	1	4

Șablon consensual

```
A C G T A C G T
```

Aliniere/ matrice profil/ consens

Observații:

- Sablonul consensual poate fi interpretat ca un motiv ancestral de la care s-au obținut configurațiile actuale prin mutații
- De regulă două șabloane diferă mai mult între ele decât cu șablonul consensual
- Pentru a evalua un șablon consensual este utilă introducerea unui scor care să permită compararea diferitelor sabloane

Aliniere

```
a G g t a c T t
C c A t a c g t
a c g t T A g t
a c g t C c A t
C c g t a c g G
```

Matrice profil

A	3	0	1	0	3	1	1	0
C	2	4	0	0	1	4	0	0
G	0	1	4	0	0	0	3	1
T	0	0	0	5	1	0	1	4

Sablon consensual

```
A C G T A C G T
```

Notatii

Pentru a stabili scoruri care să permită evaluarea calității unui șablon vom considera un set de secvențe ADN și următoarele notații:

- ***t*** - număr de secvențe ADN analizate
- ***n*** - lungimea fiecărei secvențe (nu este necesar să fie toate de aceeași lungime însă pentru simplitate în continuare vom presupune satisfacută această ipoteză)
- ***ADN*** - ansamblul tuturor secvențelor (matrice *t* x *n*)
- ***L*** - lungimea unui șablon (un șablon de lungime ***L*** este uneori numit ***L***-mer)

- s_i - poziția de start a unui șablon de lungime ***L*** în secvența i

  $1 <= s_i <= n-L+1$
- $s=(s_1, s_2, \ldots s_t)$ - vectorul cu pozițiile de start ale tuturor șabloanelor

Evaluare potrivire: scor

Gradul de potrivire a două sau mai multe secvențe aliniate poate fi măsurat printr-un scor de potrivire:

$$scor(s, ADN) = \sum_{i=1}^{L} \max_{k \in \{A,C,G,T\}} P(k,i)$$

$P(k,i) =$ frecventa de aparitie a nucleotidei k pe pozitia i

A G G T A C T T
C C A T A C G T
A C G T T A G T
A C G T C C A T
C C G T A C G G

A	3	0	1	0	3	1	1	0
C	2	4	0	0	1	4	0	0
G	0	1	4	0	0	0	3	1
T	0	0	0	5	1	0	1	4

Șablon consensual
A C G T A C G T

Scor: 3+4+4+5+3+4+3+4=30

Scorul poate fi folosit pentru a indica care dintre pozițiile de start este cea mai bună (cu cât scorul este mai mare cu atât e mai reprezentativ șablonul consensual)

Evaluare potrivire: continut informational

- O altă modalitate de estimare a gradului de potrivire a două sau mai multe secvențe aliniate se bazează pe așa numitul conținut informațional (information content) introdus în [Hertz, Stormo, Identifying DNA and proteins patterns with statistically significant alignments of multiple sequences, Bioinformatics, vol. 15, 1999]:

A 3 0 1 0 3 1 1 0
C 2 4 0 0 1 4 0 0
G 0 1 4 0 0 0 3 1
T 0 0 0 5 1 0 1 4

IC: 6.51

$$IC(s, ADN) = \sum_{i=1}^{L} \sum_{k \in \{A,C,G,T\}} f(k,i) \ln \frac{f(k,i)}{p(k)}$$

$f(k,i) = P(k,i)/t$ (frecventa relativa de aparitie a nucleotidei k pe pozitia i)

$p(k) =$ probabilitatea apriori a nucleotidei k (ex : $p(k) = 0.25$)

Obs: Cu cât conținutul informațional este mai mare cu atât e mai reprezentativ șablonul consensual)

Metode de identificare

In cazul general problema identificării șabloanelor poate fi enunțată astfel:

- Pornind de la un set de secvențe ADN să se găsească un set de subsecvențe de lungime dată L, câte una din fiecare secvență astfel încât scorul asociat să fie maxim

- Intrare: O matrice ADN(t , n) conținând cele t secvențe de lungime n și lungimea șablonului căutat, L

- Ieșire: Un vector cu t poziții de start:

  s = (s_1, s_2, … s_t) care maximizează scor(s,ADN)

  (sau IC(s,ADN))

Metode de identificare

- Există mai multe metode de identificare a șabloanelor.
- Acestea diferă între ele fie prin spațiul în care se face căutarea fie prin principiul care stă la baza căutarii
- In funcție de spațiul în care se face căutarea există:
  - Metode de căutare în spațiul pozițiilor posibile de start ale șabloanelor**: {1,2,…,n-L+1}x…x{1,2,…,n-L+1}**
    - Dimensiunea spațiului de căutare: $(n-L+1)^t$
  - Metode de căutare în spațiul configurațiilor ce pot fi șabloane (de lungime L): **{A,C,G,T}x … x{A,C,G,T}**
    - Dimensiunea spațiului de căutare: 4^L

Metode de identificare

- Există mai multe metode de identificare a șabloanelor.
- Acestea diferă între ele fie prin spațiul în care se face căutarea fie prin principiul care stă la baza căutarii
- In funcție de spațiul în care se face căutarea există:
  - Metode de căutare în spațiul pozițiilor posibile de start ale șabloanelor**: {1,2,…,n-L+1}x…x{1,2,…,n-L+1}**
  - Metode de căutare în spațiul configurațiilor ce pot fi șabloane: **{A,C,G,T}x … x{A,C,G,T}**
- Căutarea poate fi:
  - Sistematică (algoritmi de tip backtracking sau branch and bound)
  - Euristică (algoritmi de tip greedy, algoritmi aleatori)

Cautare sistematica

Rezolvare 1: Parcurgerea în întregime a spațiului pozițiilor (tehnica căutarii sistematice):

- maxScor:=0
- Pentru fiecare element $s=(s_1,\ldots,s_t)$ al produsului cartezian
  $\{1,2,\ldots,n-L+1\}$x $\{1,2,\ldots,n-L+1\}$x…x $\{1,2,\ldots,n-L+1\}$ (de t ori)

  - Se calculează scor(s,ADN)
  - IF scor(s,ADN) > maxScor THEN maxScor:=scor(s,ADN)
    maxScor:=s

    ENDIF

Observatie:

- Necesită generarea tuturor elementelor produsului cartezian
- Numărul de scoruri calculate este $\boldsymbol{(n-L+1)^t}$ iar pentru calculul fiecarui scor numărul de operații este de ordinul $\boldsymbol{tL}$
- Ordinul de complexitate este $\boldsymbol{O(tL\ n^t)}$
- Exemplu: pentru $\boldsymbol{t}$ = 10, $\boldsymbol{n}$ = 1000, $\boldsymbol{L}$ = 8 numărul de operații este $8*10^{31}$

Cautare sistematica: backtracking

Generarea produsului cartezian {1,2,…m}x…x{1,2,…,m} (m=n-L+1, t factori)

a) Tehnica numărarii: se pornește de la (1,1,…,1) si se ajunge la (m,m,…m)
b) Backtracking

```
Gen(k)
IF k==t+1 THEN "prelucreaza s[1..t] si
      actualizeaza maxScor"
ELSE
   FOR i=1, m DO
      s[k]=i
      Gen(k+1)
   ENDFOR
ENDIF
```

- Prelucrarea lui s[1..t] presupune calculul scorului și compararea cu valoarea curentă a lui maxScor (variabilă globală care se inițializează cu 0 înainte de apel)
- Numărul mare de prelucrări poate fi limitat aplicând o strategie de tip branch and bound: presupune estimarea șansei fiecărei soluții parțiale de a fi mai bună decât cea mai buna soluție descoperită până în momentul respectiv

Cautare sistematica: branch&bound

Rezolvare 2: Branch and Bound

- Pentru un vector parțial s(k)= (s_1,s_2,…,s_k,-,…,-) de poziții de start se determină scorul maxim ce s-ar putea obține dacă și celelalte componente ar fi completate.

- Se presupune că pentru celelalte componente s-ar obține scorul maxim: *L*(t-k)*

- Daca *scor(s(k)) + L*(t-k) < maxScor* (cel mai bun scor obținut până în momentul curent) atunci ramura este abandonată (întrucât chiar dacă scorul de potrivire corespunzător componentelor necompletate ar fi maxim nu e posibil să se obțină un scor global mai bun)

Cautare sistematica: branch&bound

Branch and Bound

Pentru a eficientiza implementarea se estimează scorurile maximale ale tuturor vectorilor parțiali având k elemente după care se explorează soluțiile viabile în ordinea descrescătoare a scorurilor maximale

In cel mai defavorabil caz algoritmul are încă complexitatea $O(tLn^t)$ dar în cazul mediu numărul de operații este mai mic decât cel de la backtracking

```
Gen(k)
IF k==t+1 THEN "prelucrează s[1..t] si
                actualizează maxScor"
ELSE
   initializare lista de candidati C
   FOR i=1, m DO
      s[k]=i
      IF scor(s[1..k])+L*(t-k)>maxScor
            THEN adaugă s[k] la lista C
      ENDIF
   ENDFOR
   sortează lista C descrescător dupa scor
   FOR fiecare element c din lista C DO
     s[k]=c
     Gen(k+1)
   ENDFOR
ENDIF
```

Median string problem

Rezolvare 3: parcurgerea spațiului configurațiilor ce pot fi șabloane

Obs. Problema devine echivalentă cu cea a găsirii subșirului “median” (“median string problem”)

Ideea de bază:

- Se parcurg toate sabloanele posibile de lungime L (în cazul alfabetului {A,C,G,T} sunt 4^L șabloane posibile
- Se calculează distanța dintre fiecare șablon posibil și setul de secvențe
- Se alege șablonul pentru care distanța este minimă (aceasta va fi șablonul “median” și este de fapt șablonul consensual căutat)

Median string problem

Rezolvare 3: Calculul distanței dintre un șablon și un set de t secvențe:

- considerăm că v este un șablon arbitrar de lungime L;
-
- Distanța de la șablonul v la setul de secvențe ADN se calculează ca fiind minimul distanțelor dintre v și fiecare fragment de lungime L din ADN

$$d(v, ADN) = \min_s d_H(v, s) = \min_s \sum_{i=1}^{t} d_H(v, s_i)$$

unde $s = (s_1, s_2, ..., s_t)$ e vectorul pozitiilor de start

$$d_H(v, s_i) = card\{ j \in \{1, ..., L\} \mid v_j \neq ADN(i, s_i + j - 1)\}$$

este distanta Hamming dintre v si subsirul ce incepe de la pozitia s_i in secventa i din setul de secvente ADN

Median string problem

Rezolvare 3: Calculul distanței dintre un șablon v și un set de t secvențe:

- Intrucât se caută minimul unei sume de termeni pozitivi este suficient să se determine valoarea minimă a fiecărui termen
- Aceasta înseamna că fiecare dintre cele t secvențe este parcursă folosind șablonul v și se identifică porțiunea de L nucleotide consecutive care se află la distanța cea mai mică de v. (Complexitate: $O(L(n-L+1))$)
- Se calculează suma celor t distanțe minime și aceasta reprezintă distanța căutată
- Complexitate calcul distanță: $O(Lt(n-L+1))=O(nLt)$

Median string problem

Rezolvare 3:

Structura generală a algoritmului:

- minDist:= valoare suficient de mare
- Pentru fiecare element $v=(v_1,\ldots,v_L)$ al produsului cartezian {A,C,G,T}x {A,C,G,T}x…x {A,C,G,T} (de L ori)

  - Se calculeaza dist(v,ADN)
  - IF dist(v,ADN) < minDist THEN minDist:=dist(v,ADN)
    sablon:=v

Complexitate: ***$O(4^L nLt)$*** (întrucât n e de regulă semnificativ mai mare decat 4, ordinul $O(Ln^t)$ este semnificativ mai mare decât $O(4^L nLt)$)

Ex. Pt L=8, n=1000, t=10 se obține $4^L nLt = 5*10^9 < 8*10^{31} = tL\ n^t$

Obs: Parcurgerea elementelor din produsul cartezian se poate baza pe backtracking (sau branch and bound)

Cautare greedy: varianta simpla

Rezolvare 4: Căutare euristică în spațiul pozițiilor de start (varianta greedy simplă)

- Vectorul de poziții se construiește incremental
- La fiecare etapă se completează cu poziția ce pare cea mai promițătoare (conduce la cel mai bun scor în raport cu ceea ce se cunoaște deja)

Structura generală a algoritmului:

- Se determina s_1 și s_2 prin căutare sistematică (***O(Ln²)***)
- FOR i:=3,t
  - Determina s_i ce maximizează scorul în raport cu matricea profil parțială constituită din primele i-1 șabloane selectate

Complexitate: ***O(Ln²+nLt)***

Dezavantaje:

- Nu este garantat faptul ca șablonul descoperit este cel mai reprezentativ
- Rezultatul depinde de ordinea de prelucrare a secvențelor

Cautare greedy: CONSENSUS

Rezolvare 5: Varianta greedy care nu depinde de ordinea de prelucrare a secvențelor (similară algoritmului CONSENSUS [Hertz, Stormo, 1999])

Structura generală a algoritmului:

- Se construiește lista cu toate subsecvențele de lungime L (L-merii) din cele t secvențe (sunt **t(n-L+1)** astfel de subsecvențe)
- Se calculează scorul tuturor perechilor de subsecvențe aparținând unor secvențe diferite și se alege perechea cu scorul maxim;

  sunt **$C(t,2)(n-L+1)^2$** astfel de perechi; subsecvențele incluse în secvențele luate în considerare la construirea perechii sunt eliminate din lista cu subsecvențe disponibile
- FOR i:=3,t
  - Setul curent constituit din (i-1) subsecvențe se combină cu toate subsecvențele disponibile în lista (sunt **(t-i+1)*(n-L+1)** astfel de subsecvențe) și se alege setul cu i subsecvențe care are scorul maxim

Complexitate: $O(t^2nL + C(t,2)(n-L+1)^2 L + t(n-L+1)L)$

Exemplu: pt n=1000, t=10, L=8 nr de operatii este cca $4*10^7$

Cautare greedy: CONSENSUS

Rezolvare 5: Varianta greedy care nu depinde de ordinea de prelucrare a secvențelor (similară algoritmului CONSENSUS [Hertz, Stormo, 1999])

Varianta descrisă se caracterizează prin faptul că generează un singur profil, iar acesta conține exact o subsecvență din fiecare secvență

Implementarea algoritmului CONSENSUS (http://bifrost.wustl.edu/consensus/index.html)

Conține și alte variante:

- O secvență poate să nu contribuie la construirea șablonului
- O secvență poate să contribuie cu mai multe subsecvențe la construirea șablonului

Cautare probabilista

Există două categorii importante de algoritmi care se bazează pe o abordare probabilistă:

Gibbs Motif Sampler (http://bayesweb.wadsworth.org/gibbs/gibbs.html)

- se bazează pe o euristică aleatoare

MEME (Multiple Expectation Maximization for Motif Elicitation) (http://meme.sdsc.edu/meme4_3_0/intro.html

- se bazează pe algoritmul Expectation Maximization (algoritm aleator de estimare a parametrilor unui model probabilist)
- MEME suite web server: http://meme.nbcr.net

Gibbs Motif Sampler

Date de intrare: t secvențe ADN de lungime N

Ieșire: pozițiile de start (s_1,s_2,..., s_t) ale unui șablon consensual ce maximizează un scor de potrivire

Idee: pozițiile de start din secvențe sunt inițializate aleator, și sunt modificate printr-un proces iterativ:

- la fiecare iterație se modifică poziția de start într-una dintre secvențe astfel încât să fie maximizat raportul dintre probabilitatea de potrivire a subsecvenței (ce începe la acea poziție) cu șablonul consensual și probabilitatea de potrivire cu distribuția de fond (calculată pe baza nucleotidelor ce nu sunt incluse în șablon)

Exemplu:

A 3 0 1 0 2 1 1 0
C 1 3 0 0 1 3 0 0
G 0 1 3 0 0 0 2 1
T 0 0 0 4 1 0 1 3

frecvențe de fond: f_A= 0.24, f_C=0.26, f_G=0.21, f_T=0.29

subsecvența analizată: **ACATACTG**

Probabilitate potrivire cu șablon: $3*3*1*4*2*3*1*1/4^8$ = 0.0033

Probabilitate potrivire cu distribuția de fond:

0.24*0.26*0.24*0.29*0.24*0.26*0.29*0.21 = 1.6504e-005

Gibbs Motif Sampler

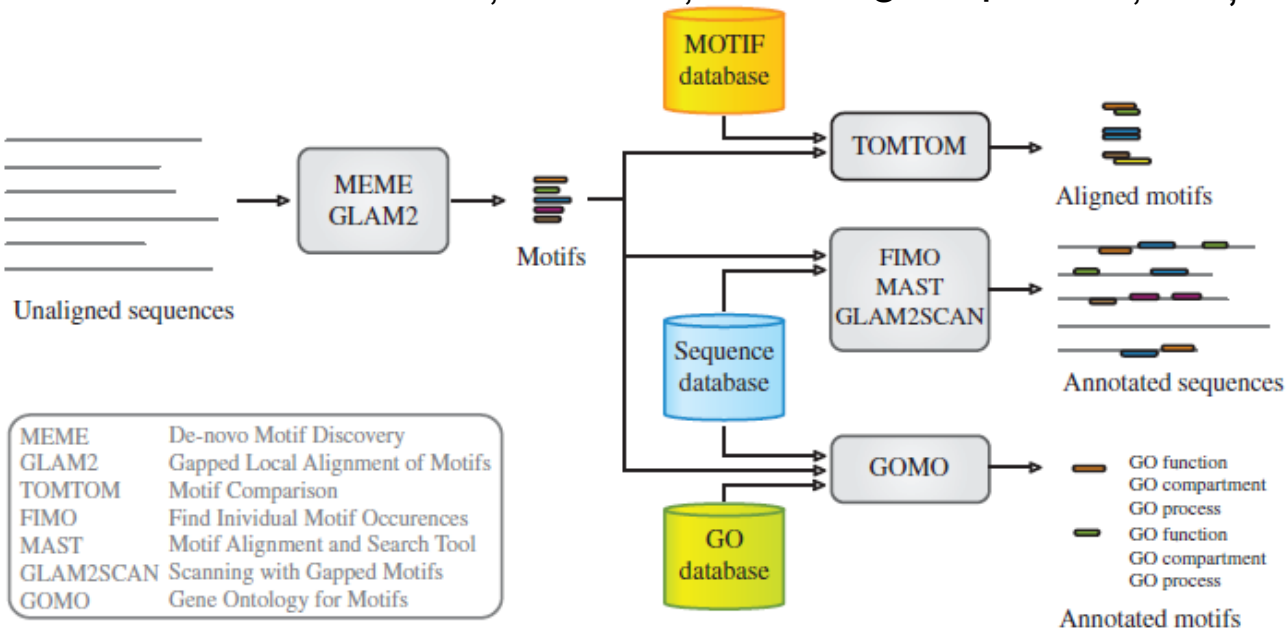
Structura generală:

- Inițializare aleatoare a listei cu pozițiile de start (s_1,s_2,...,s_t)
- WHILE nu este îndeplinită o condiție de oprire DO
  - Selectează una dintre cele t secvențe (sistematic sau aleator); secvența selectată este notată cu k
  - Calculează matricea profil și frecvențele de fond făcând abstracție de secvența k
  - FOR i:=1,N-L+1 DO
    - Calcul Q_{ki} = probabilitatea ca fragmentul ce începe cu poziția i în secvența k să fie generat pe baza probabilităților din matricea profil
    - Calcul P_{ki}= probabilitatea ca fragmentul ce începe cu poziția i în secvența k să fie generat pe baza frecvențelor de fond
    - Calcul $A_{ki}=Q_{ki}/P_{ki}$
  - Calculează probabilitatea de selecție pt fiecare poziție i $S_{ki}=A_{ki}/(A_{k1}+A_{k2}+....+A_{k,N-L+1})$
  - Setează s_k = i (i selectat aleator folosind distribuția de probabilitate de la pasul anterior)

MEME

Suita MEME (MEME suite web server: http://meme.nbcr.net):

- Permite identificarea mai multor șabloane în aceeași etapă
- Nu impune ca fiecare secvență sa conțină o singură prezență a șablonului



Biblio:

- T. Bailey et al. MEME SUITE: tools for motif discovery and searching, Nucleic Acids Research, 2009