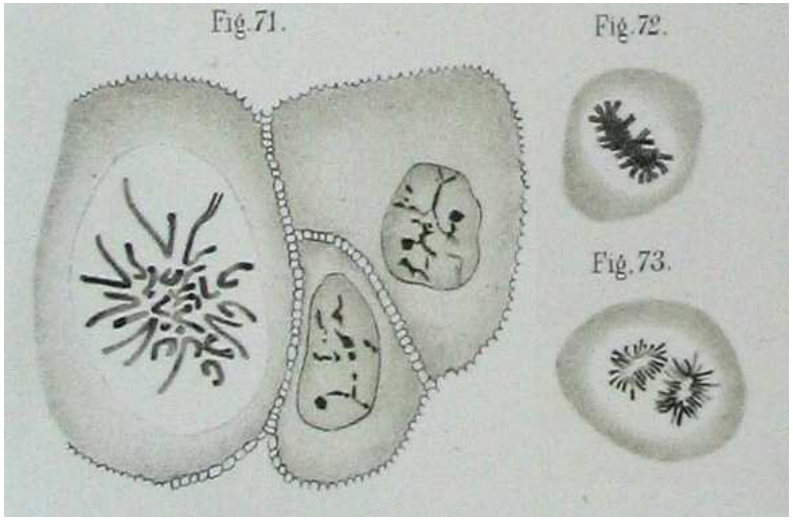
Brief Introduction to Biology

Broňa Brejová

Sept. 23, 2021



Walther Flemming, 1881

Principal players

Deoxyribonucleic acid (DNA)

carrier of genetic information passed from one generation to the next.
Long string of nucleotides from $\{A, C, G, T\}$
(adenine, cytosine, guanine, thymine).
Information stored in symbolic, digital form.

Ribonucleic acid (RNA)

Similar to DNA, thymine T replaced with uracil U

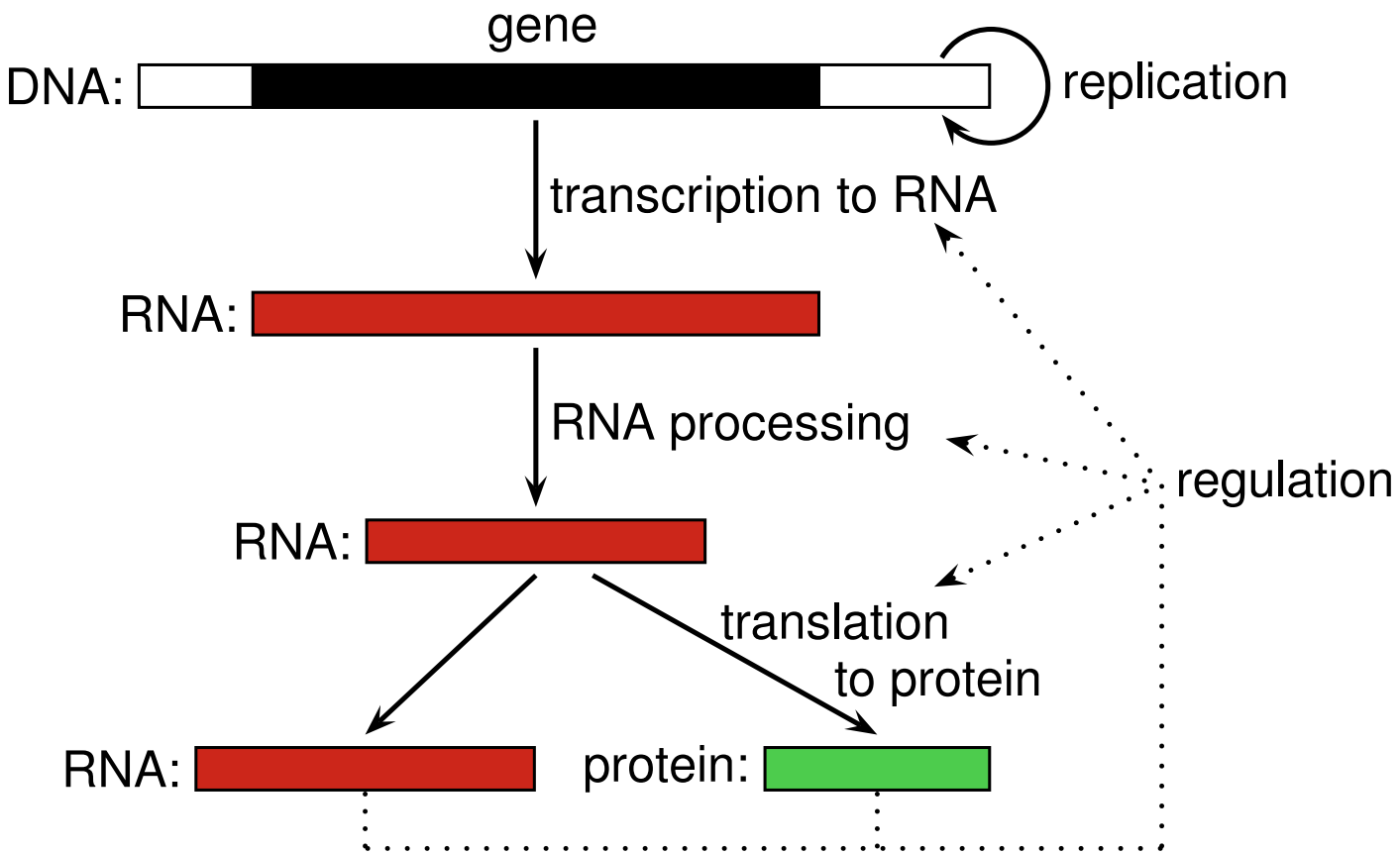
Proteins

catalyze biochemical reactions in the cell (enzymes),
carry signals within/between cells,
also important for cell structure and movement, etc.
String of amino acids (20 different amino acids).

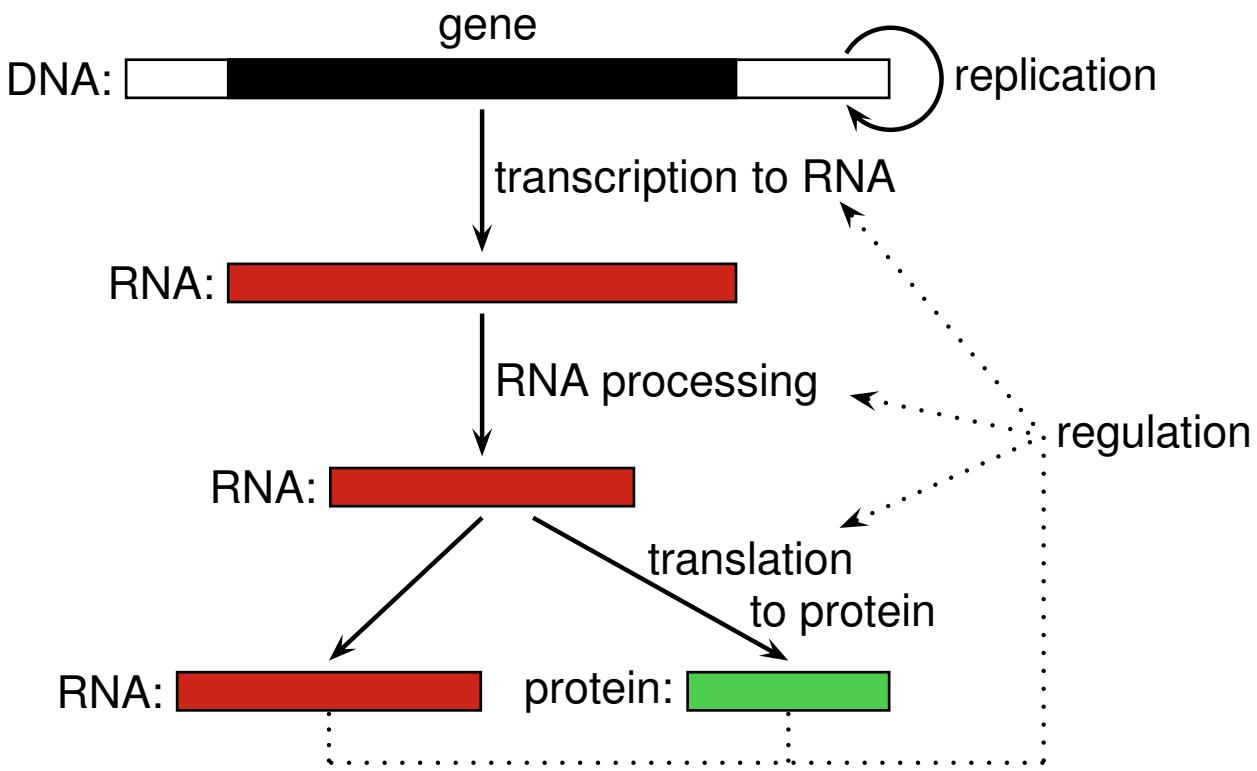
What information is stored in DNA?

Genes: Recipes for synthesis of proteins and functional RNAs.

Regulation of their expression: when and how many molecules to synthesize



Central dogma (Francis Crick 1958,1970)

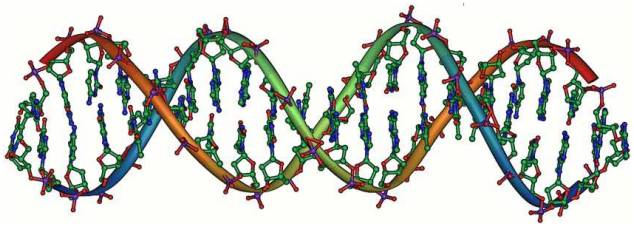


"The central dogma of molecular biology deals with the detailed residue-by-residue transfer of sequential information. It states that such information cannot be transferred back from protein to either protein or nucleic acid."

DNA, chromosomes

DNA: two complementary strands (pairs A-T, C-G),
in opposite orientation (ends are called 5' and 3').
For example ACCATG is complementary with CATGGT.

Shape of double helix



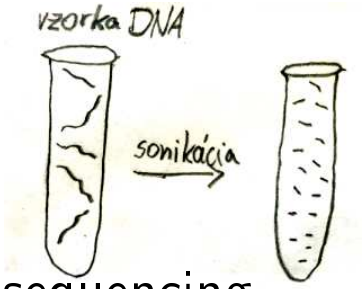
Double stranded structure allows redundancy,
repair after damage in one strand.
During cell division double stranded DNA unwinds and second strand
is synthesized to each original strand (DNA replication).

Chromosome: Complex of double-stranded DNA molecule and
support proteins

The human genome has 22 pairs chromosomes plus two sex chrom.,
together 3GB.

Technology: DNA sequencing

- Technology for determining sequence of nucleotides in chromosomes
- Complex process:
  chromosomes are cut to short pieces,
  each piece is duplicated many times,
  each piece is sequenced separately e.g. by Sanger sequencing
  – uses natural enzymes, e.g. DNA polymerase



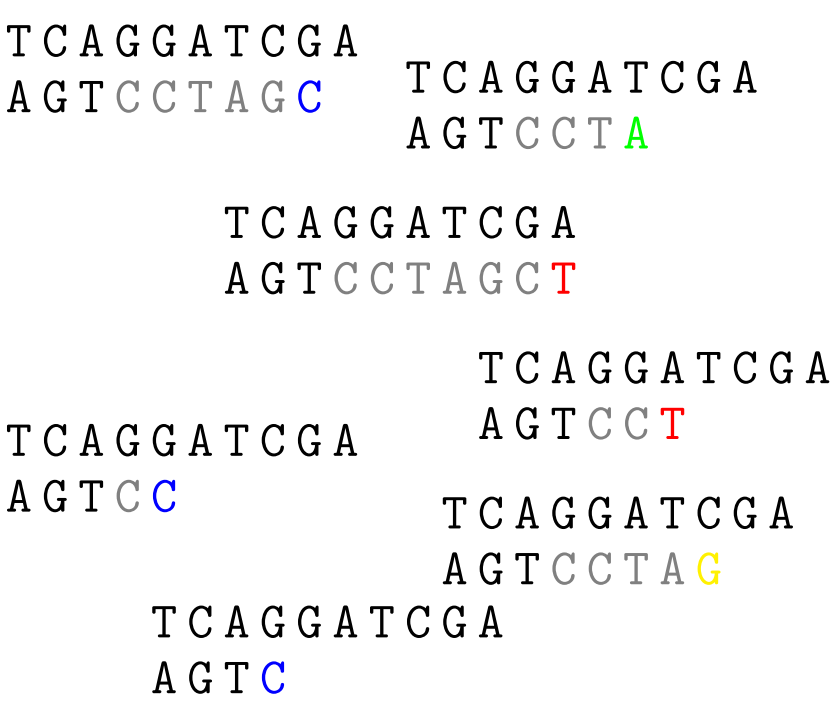
Sanger sequencing

Example: sequencing AGCTAGGACT (below drawn right-to-left)

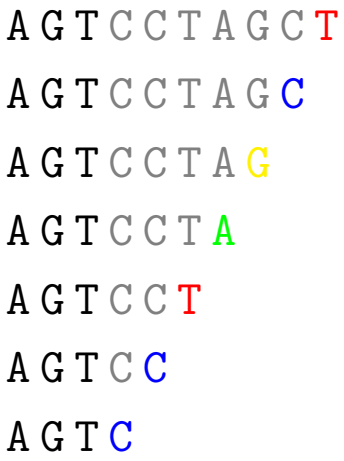
Primer AGT + enzymes + nucleotides

+ modified color-labeled nucleotides

Results of sequencing reaction:



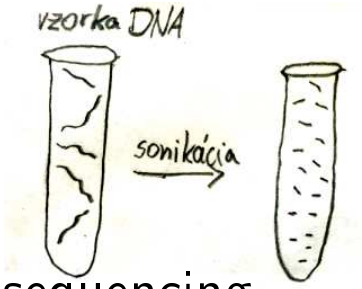
Order by length on a gel:



Read color labels to obtain complementary strand: AGTCCTAGCT

Technology: DNA sequencing

- Technology for determining sequence of nucleotides in chromosomes
- Complex process:
  chromosomes are cut to short pieces,
  each piece is duplicated many times,
  each piece is sequenced separately e.g. by Sanger sequencing
  – uses natural enzymes, e.g. DNA polymerase



- **Computational problem:** genome assembly from short pieces.
- Genome availability allows
  annotate genes and other functional regions,
  seek similarities and differences between species and organisms.

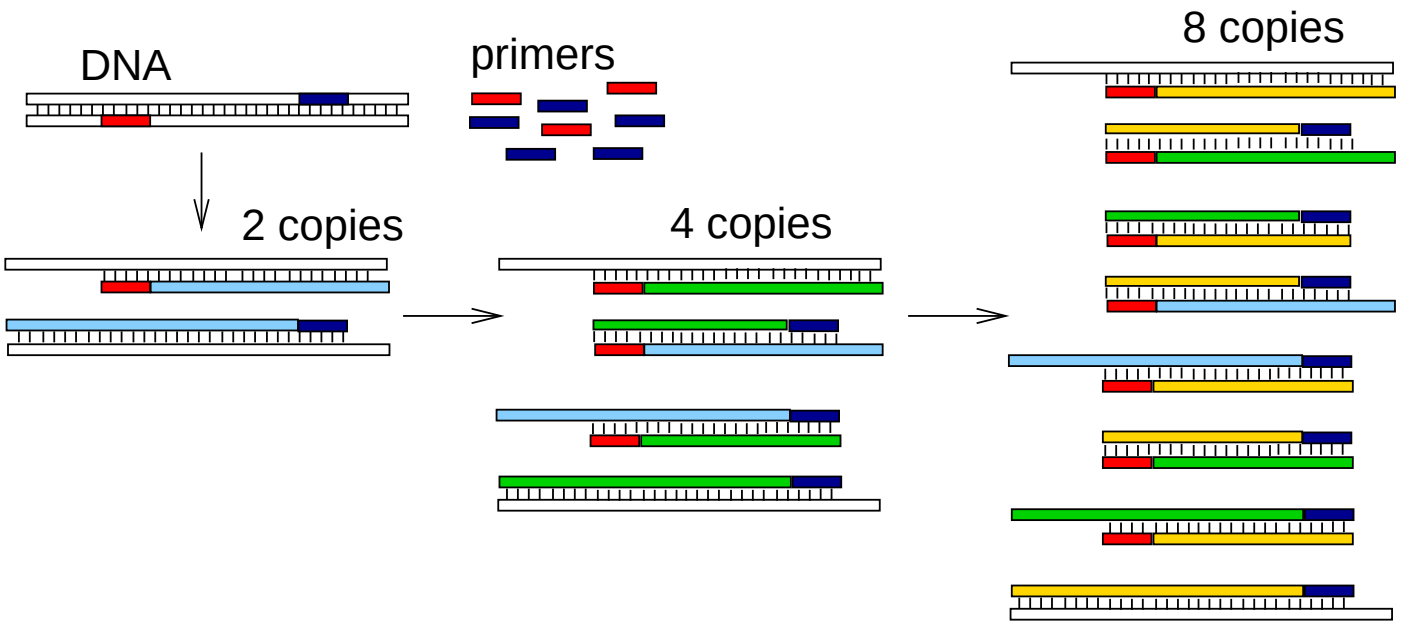
PCR (polymerase chain reaction)

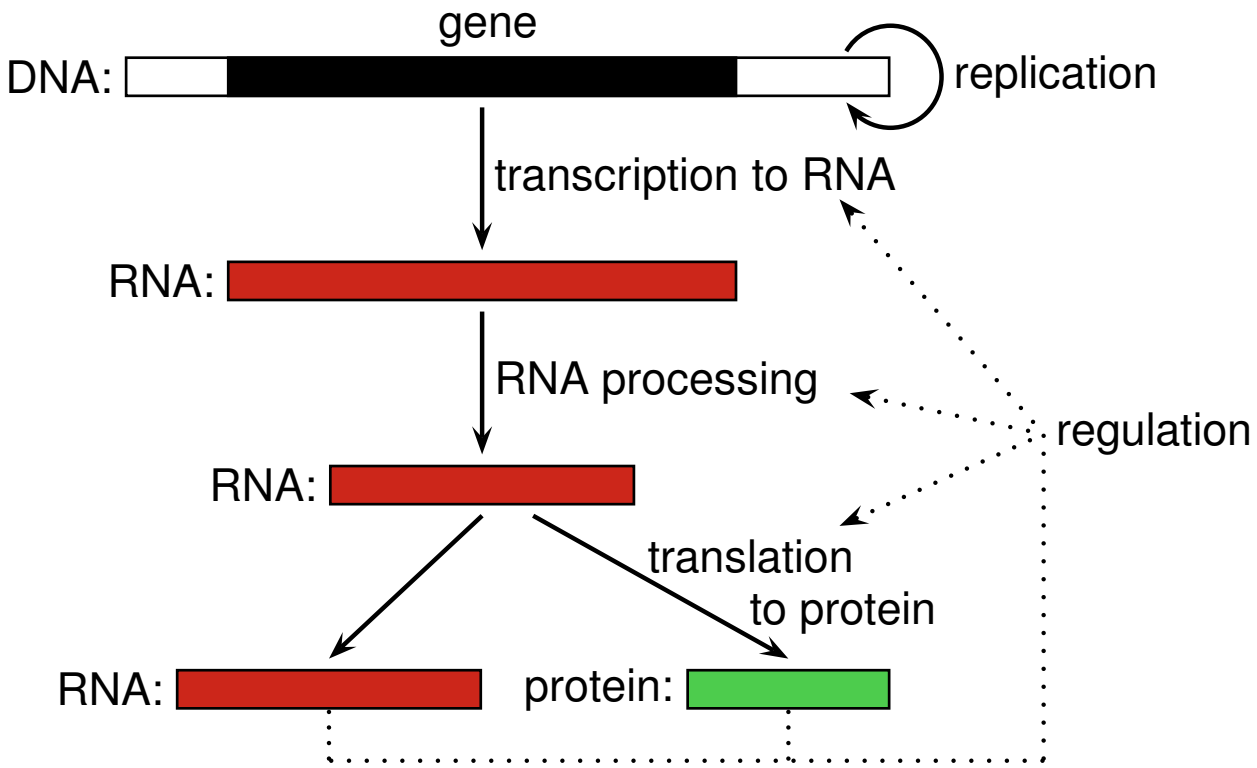
We select two short pieces of DNA (primers)

PCR tests if they are close together in DNA sample

(hundreds/thousands of bases)

If yes, it will make many copies of the region between them

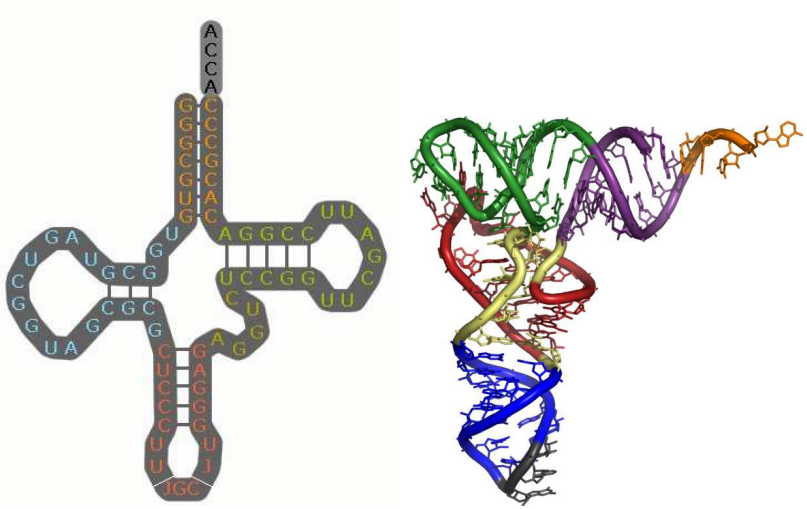



gene
DNA:
replication
transcription to RNA
RNA:
RNA processing
regulation
RNA:
translation
to protein
RNA:
protein:

RNA

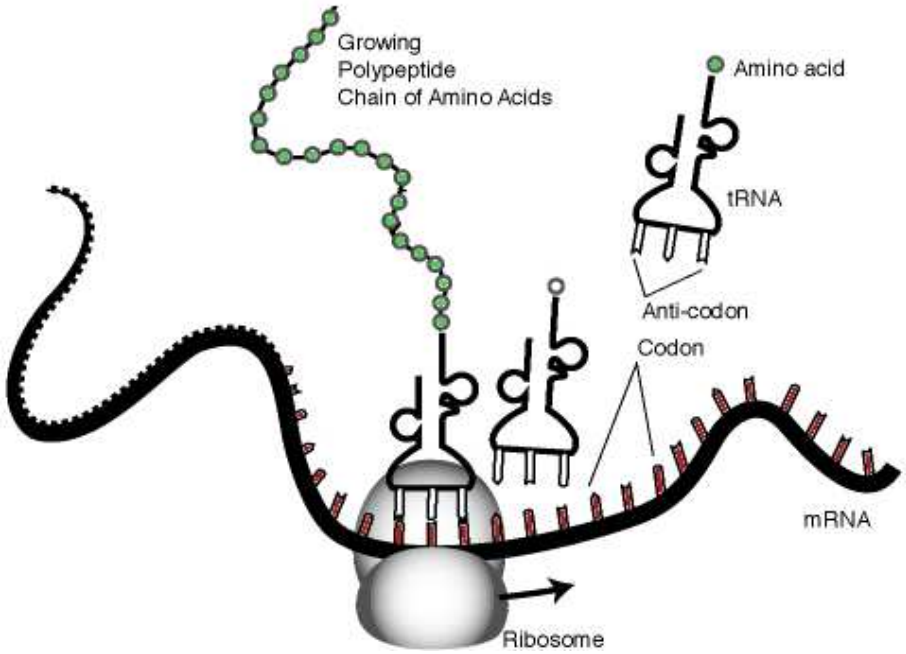
Differences from DNA

- contains ribose instead of deoxyribose
- contains uracil instead of thymine (bases A,C,G,U)
- single-stranded sequences, usually shorter
- complex secondary structure: paired complementary regions

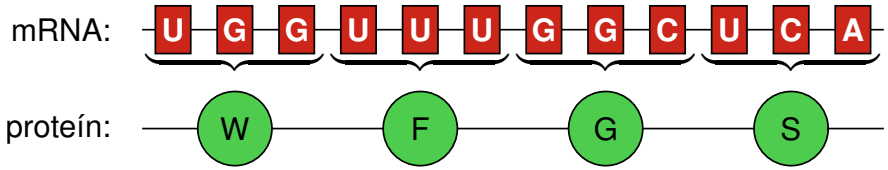


transfer RNA (tRNA), figures from Wikipedia

Translation



Codon (triple of nucleotides) determines 1 amino acid



Genetic code

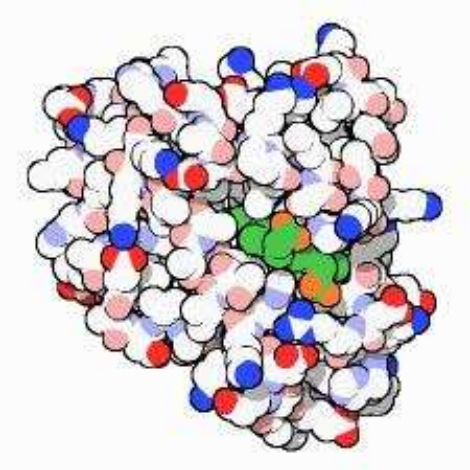
Ala / A	GCT, GCC, GCA, GCG	Leu / L	TTA, TTG, CTT, CTC, CTA, CTG
Arg / R	CGT, CGC, CGA, CGG, AGA, AGG	Lys / K	AAA, AAG
Asn / N	AAT, AAC	Met / M	ATG
Asp / D	GAT, GAC	Phe / F	TTT, TTC
Cys / C	TGT, TGC	Pro / P	CCT, CCC, CCA, CCG
Gln / Q	CAA, CAG	Ser / S	TCT, TCC, TCA, TCG, AGT, AGC
Glu / E	GAA, GAG	Thr / T	ACT, ACC, ACA, ACG
Gly / G	GGT, GGC, GGA, GGG	Trp / W	TGG
His / H	CAT, CAC	Tyr / Y	TAT, TAC
Ile / I	ATT, ATC, ATA	Val / V	GTT, GTC, GTA, GTG
START	ATG	STOP	TAA, TGA, TAG

Proteins

Strings of 20 different amino acids with different chemical properties:

Amino Acid	Side chain	Its properties
Alanine (A)	-CH3	hydrophobic
Arginine (R)	-(CH2)3NH-C(NH)NH2	basic
Asparagine (N)	-CH2CONH2	hydrophilic
Aspartic acid (D)	-CH2COOH	acidic
Cysteine (C)	-CH2SH	hydrophobic
Glutamic acid (E)	-CH2CH2COOH	acidic
Glutamine (Q)	-CH2CH2CONH2	hydrophilic
Glycine (G)	-H	hydrophilic
Histidine (H)	-CH2-C3H3N2	basic
Isoleucine (I)	-CH(CH3)CH2CH3	hydrophobic
Leucine (L)	-CH2CH(CH3)2	hydrophobic
Lysine (K)	-(CH2)4NH2	basic
Methionine (M)	-CH2CH2SCH3	hydrophobic
Phenylalanine (F)	-CH2C6H5	hydrophobic
Proline (P)	-CH2CH2CH2-	hydrophobic
Serine (S)	-CH2OH	hydrophilic
Threonine (T)	-CH(OH)CH3	hydrophilic
Tryptophan (W)	-CH2C8H6N	hydrophobic
Tyrosine (Y)	-CH2-C6H4OH	hydrophobic
Valine (V)	-CH(CH3)2	hydrophobic

Protein structure

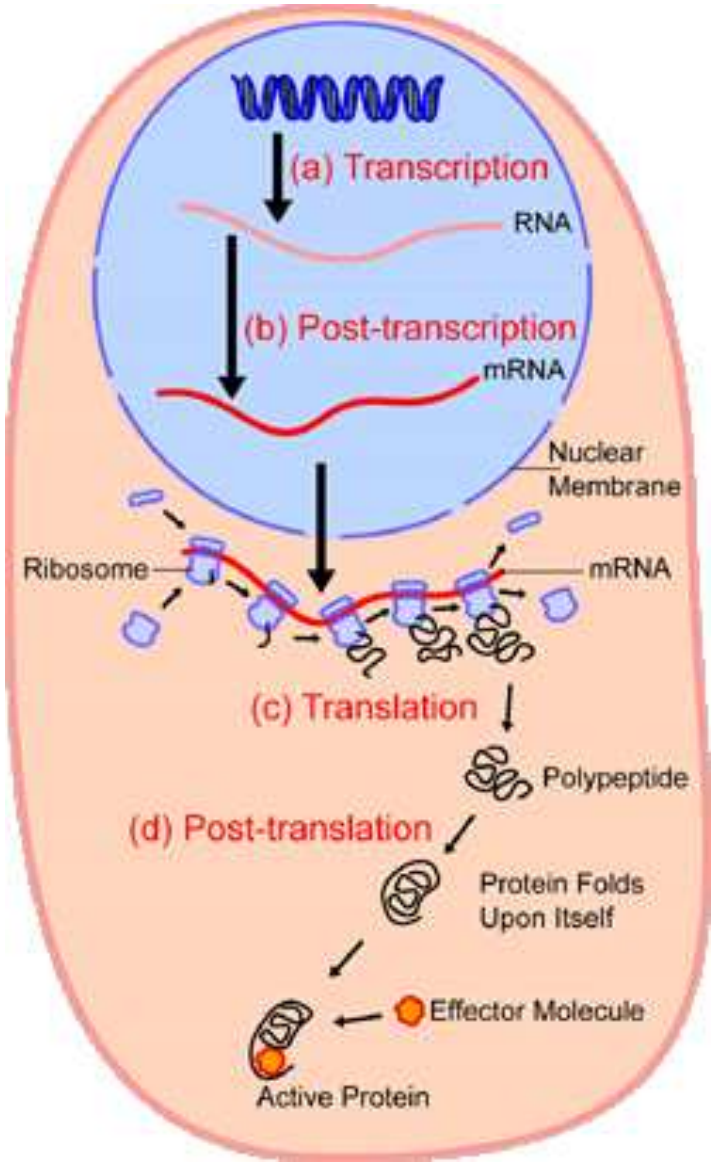


Myoglobin, the first protein with a known structure.

Proteins occur folded in a stable structure,
or move between several conformations.

Hydrophobic amino acids do not interact with water,
usually located inside the structure.

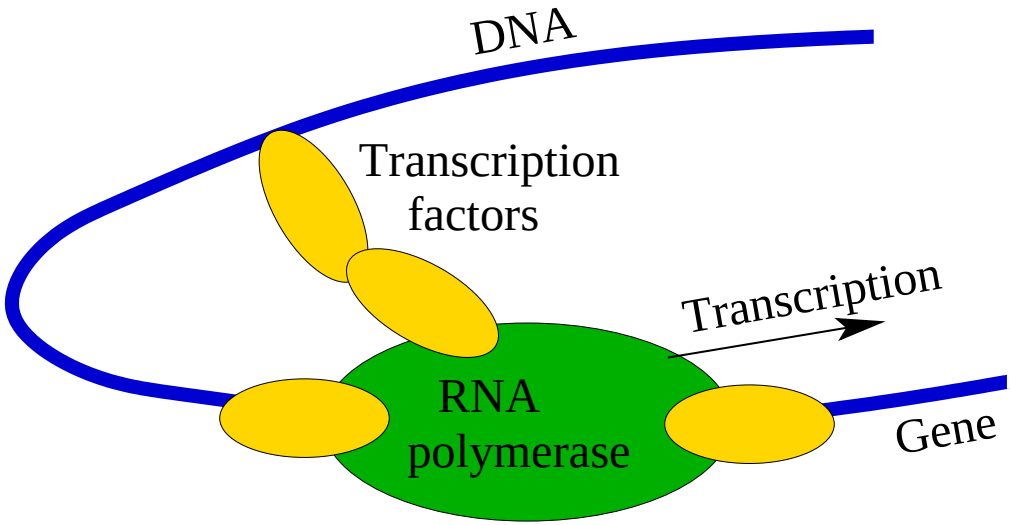
Structure of a protein determines its function.


(a) Transcription
RNA
(b) Post-transcription
mRNA
Nuclear Membrane
Ribosome
mRNA
(c) Translation
Polypeptide
(d) Post-translation
Protein Folds Upon Itself
Effector Molecule
Active Protein

Regulation of gene expression

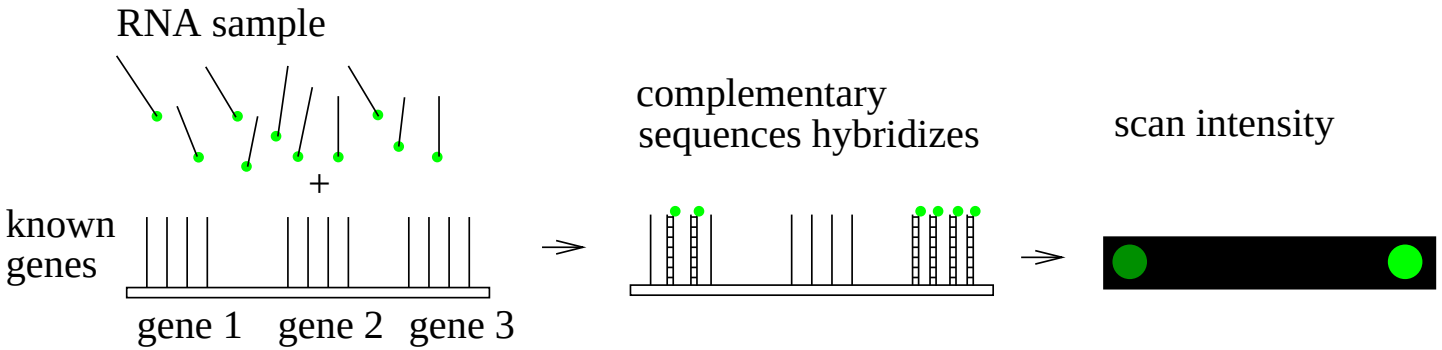
Cells in different tissues of the same organisms share the same genome, yet look and function very differently.
Some proteins are produced only under special circumstances or in variable amounts.

Regulation of transcription initiation by transcription factors:



Computational problem: determine, which factors influence a given gene and where they bind DNA.

Technology: microarray

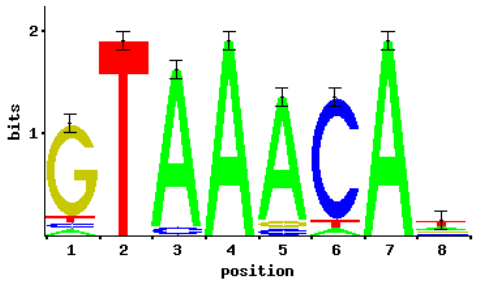


Measurement of the amount of mRNA in the sample for **many genes** simultaneously.

Repeat for different samples, study correlations among genes.

These could be caused by a common regulator (transcription factor).

Computational problem: for several co-regulated genes, find a motif where the common transcription factor could bind **(motif finding)**



Example of microarray data

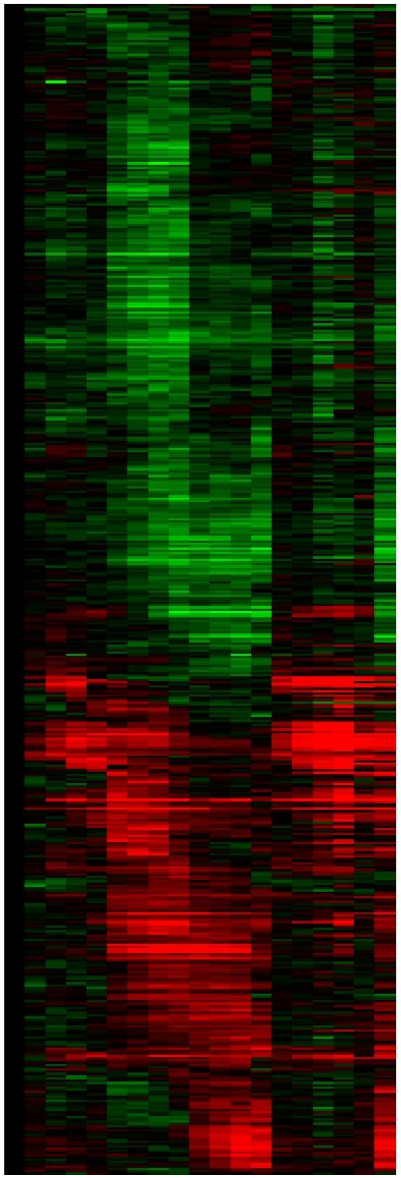
Ratio of gene expression in treated and control sample fg/bg

Red: fg>bg

Green: fg<bg

517 genes

19 experiments



Mutations in DNA

Occasionally DNA is changed, mutated
(for example due to environmental factors, replication errors).

Mutation types:

substitution (one nucleotide changes to another),
insertion (inserts several new nucleotides),
deletion (delets several nucleotides),
large scale changes (e.g. translocations).

Computational problems:

Which sequences evolved from a common ancestor?
(homology search)
Which nucleotides in two related sequences correspond to each other?
(sequence alignment)

Population genetics

Mutations are propagated in a population from parents to offspring. Dangerous mutations are quickly eliminated, advantageous mutations are more likely to spread (natural selection).

Polymorphisms: genetic differences between organisms within species. They cause differences in phenotype, e.g. appearance, genetic diseases. Sequencing several individuals within a species helps to map common polymorphisms.

Computational problem:

Determine polymorphisms linked to a certain disease or other character

Evolution

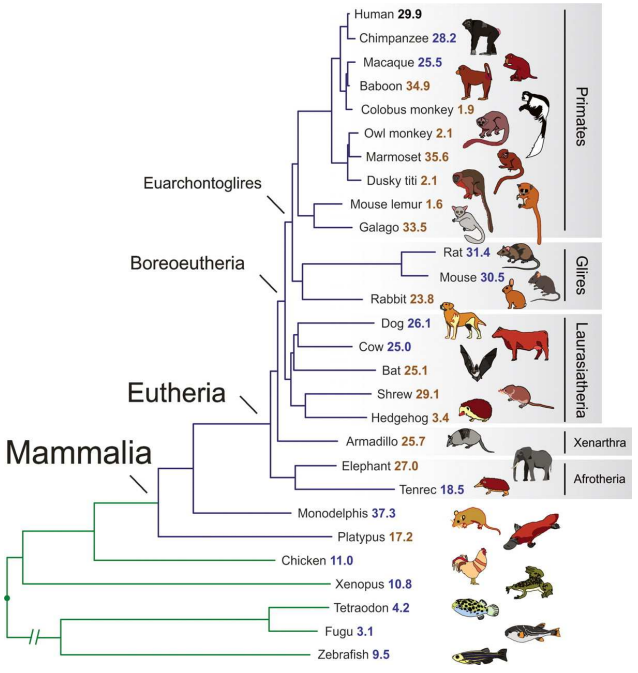
Speciation:

If a population is divided into several isolated subpopulations, genetic material is not exchanged.

Mutations accumulate independently, until mating no longer possible – new separate species.

Computational problem:

Using sequences of current species, reconstruct a **phylogenetic tree** representing their evolutionary history



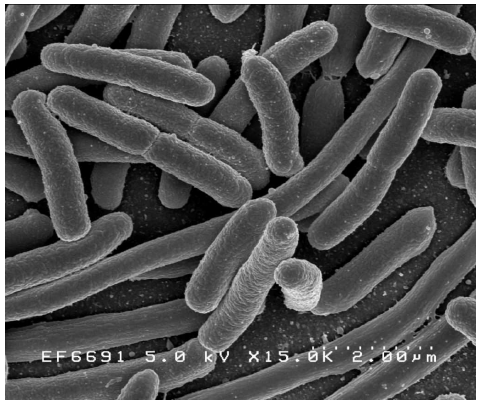
Prokaryotes vs. eukaryotes

Prokaryotes: bacteria and archea, simple unicellular organisms.
DNA directly in the cytoplasm.
Genome in one circular chromosome (plus shorter plasmids),
simple gene structure

Eukaryotes: animals, fungi, plants, some unicellular organisms.
DNA in a nucleus, cell contains also other organelles.
Mitochondria and chloroplasts are engulfed prokaryotes which became part of the eukaryotic cell.
Longer genome in several linear chromosomes.

Model organisms

Species important for biology research, explored more than other related species. General principles applicable to other species as well.



Escherichia coli: bacterium living in digestive tract. Simple to manipulate, cell division every 20 min. Study of basic biological processes: DNA replication, gene expression, etc. Genome 4.6MB, 4000 genes.

Saccharomyces cerevisiae: baker's yeast. Simple eukaryotic organism. Genome with 6000 genes, 13MB. Cell division every 2 hours. Study of processes specific for eukaryotes.

Model organisms

Arabidopsis thaliana: small flowering plant, 6-week reproduction cycle. Model organism for plant research.

Caenorhabditis elegans: small worm, nematode, living in soil. Study of development, cell differentiation.

Drosophila melanogaster: fruit fly. Study of genetics, development genes.

Vertebrates: frog Xenopus laevis (large eggs, easy to manipulates), aquarium fish Danio rerio (translucent embryos), mouse Mus musculus (many laboratory breeds with different properties).

Available data

- DNA sequences: whole genomes or their parts
- Genome annotation: location of genes and other functional elements
- RNA sequences and structures
- Protein sequences, their function and structure
- Measurements of cell state (amount of RNA, protein, etc.)
- ...

Data obtained by experiments or by computational methods
Often unreliable, noisy (in both cases)

More information

- Zvelebil, Baum: Understanding Bioinformatics, chapter 1
- University textbooks of molecular biology
- English Wikipedia
- Tutorials linked on the course website

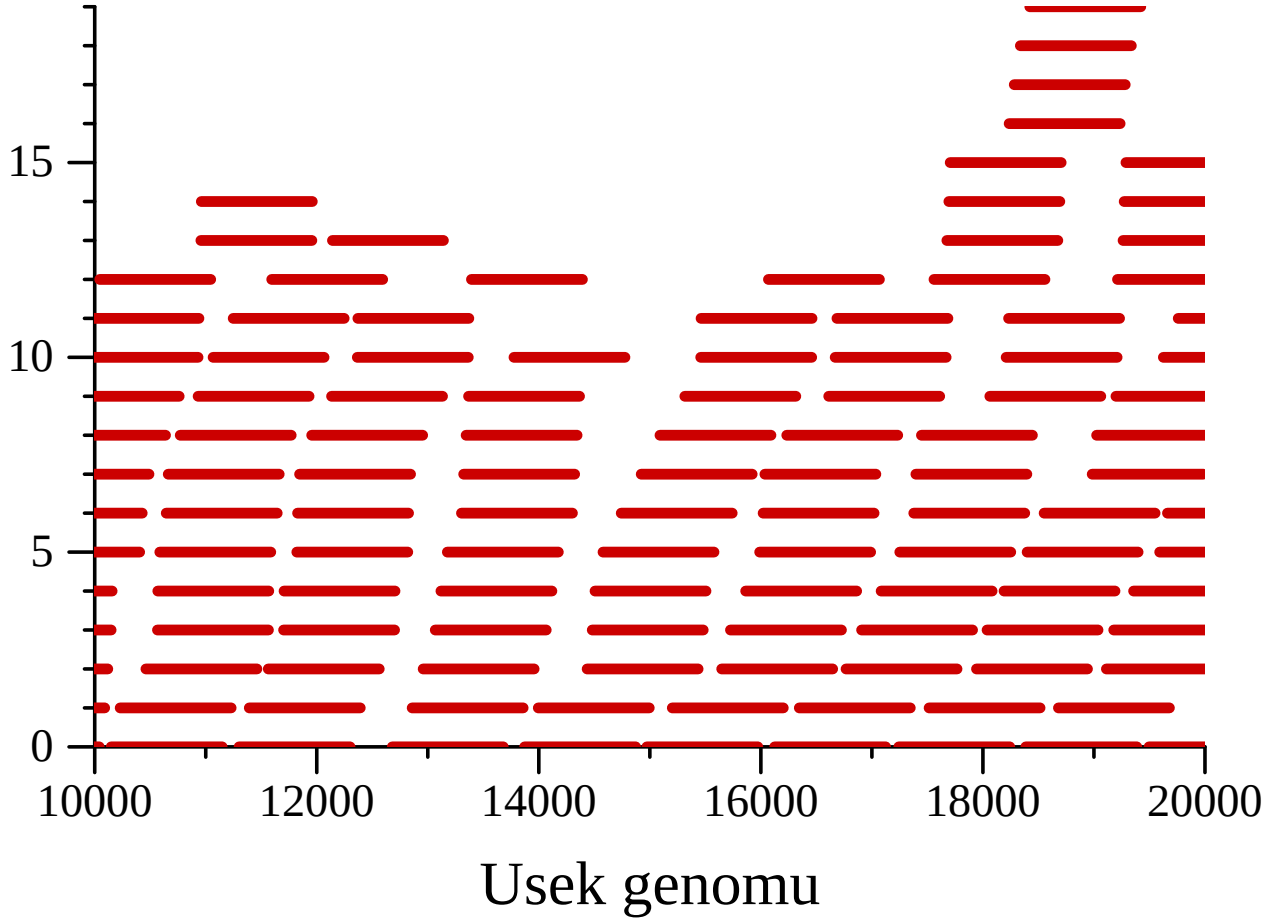
Úvod do pravdepodobnosti, sekvenovanie genómov (cvičenie)

Askar Gafurov

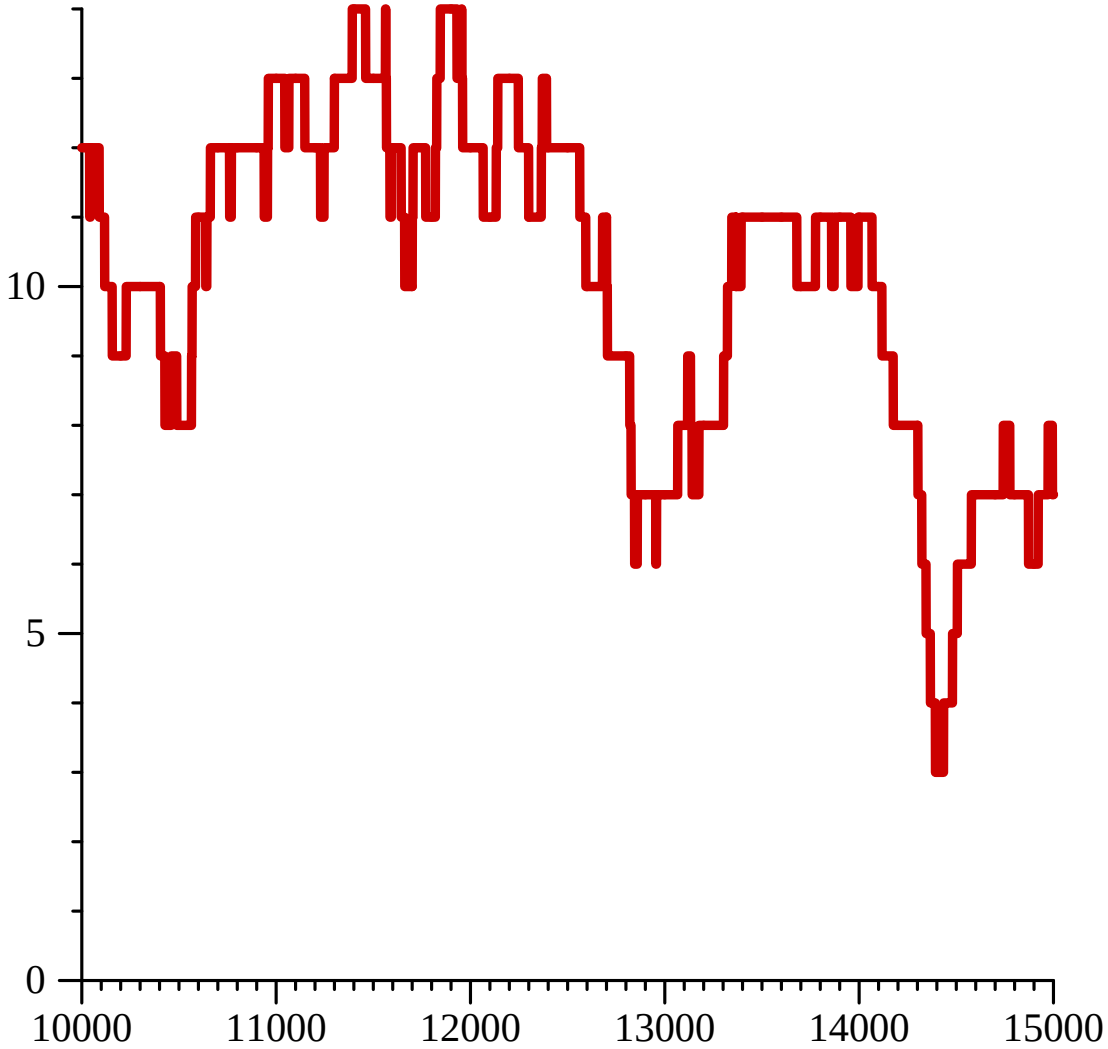
3.10.2019

- $G =$ dĺžka genómu, napr. 1 000 000
- $N =$ počet čítaní (readov), napr. 10 000
- $L =$ dĺžka čítania, napr. 1000
- $T =$ potrebná dĺžka prekryvu, napr. 50

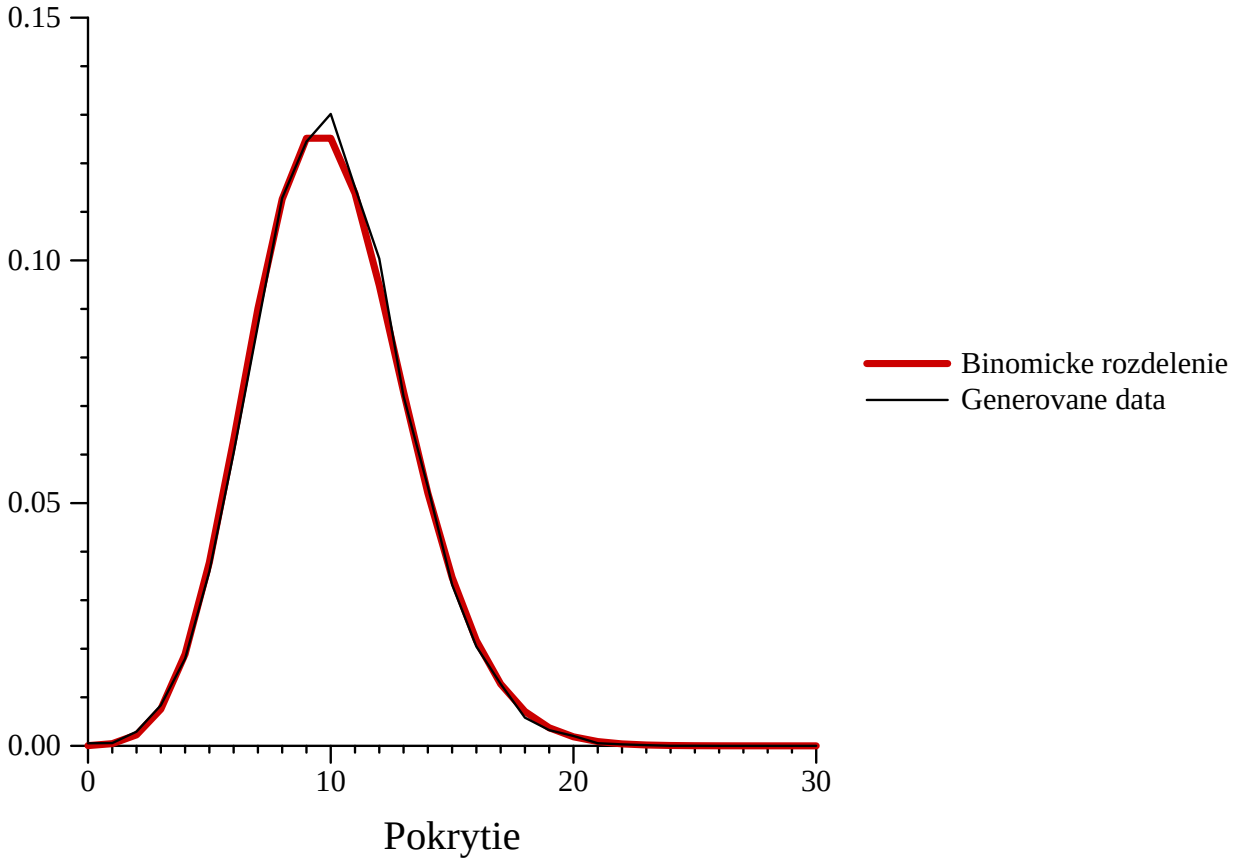
Náhodne generované čítania



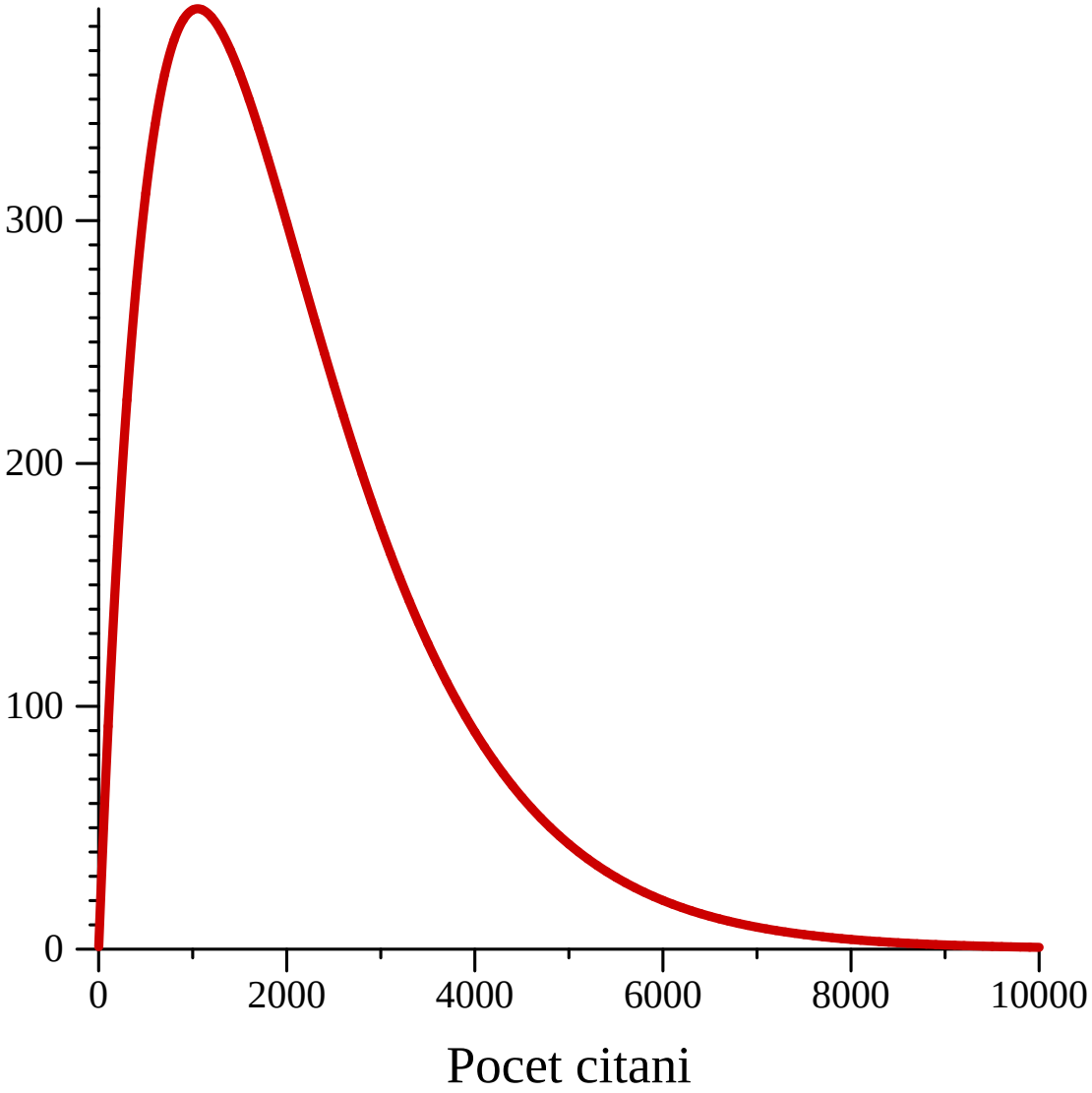
Pokrytie jednotlivých báz



Počet báz s určitým pokrytím



Predpokladaný počet kontigov od počtu čítaní



```
nepokr: 0 koncov: 0
nepokr: 274 koncov: 2
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 122 koncov: 1
nepokr: 13 koncov: 1
nepokr: 265 koncov: 1
nepokr: 0 koncov: 0
nepokr: 217 koncov: 1
nepokr: 0 koncov: 0
nepokr: 139 koncov: 2
nepokr: 76 koncov: 1
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 103 koncov: 2
nepokr: 69 koncov: 1
nepokr: 0 koncov: 0
nepokr: 282 koncov: 1
nepokr: 0 koncov: 0
nepokr: 12 koncov: 1
nepokr: 135 koncov: 1
nepokr: 1 koncov: 1
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 3 koncov: 1
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 221 koncov: 1
nepokr: 1 koncov: 1
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 8 koncov: 1
nepokr: 0 koncov: 0
nepokr: 111 koncov: 1
nepokr: 56 koncov: 1
nepokr: 10 koncov: 1
nepokr: 130 koncov: 1
nepokr: 0 koncov: 0
nepokr: 86 koncov: 1
nepokr: 0 koncov: 0
nepokr: 26 koncov: 1
nepokr: 0 koncov: 0
nepokr: 0 koncov: 0
nepokr: 12 koncov: 1
nepokr: 71 koncov: 1
```

Úvod do dynamického programovania, proteomika

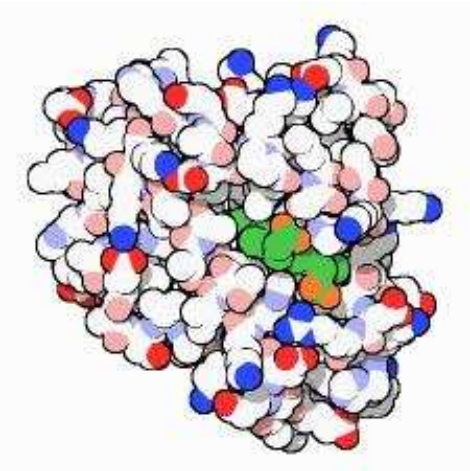
Askar Gafurov

7.10.2021

Proteomika

Proteín: sekvencia pozostáva z 20 rôznych aminokyselín

```
MGLSDGEWQLVLNVWGKVEADIPGHGQEVLIRLFKGHPETLEKFDKFKHLKSEDEMKASE
DLKKHGATVLTALGGILKKKGHHEAEIKPLAQSHATKHKIPVKYLEFISECIIQVLQSKH
PGDFGADAQGAMNKALELFRKDMASNYKELGFQG
```



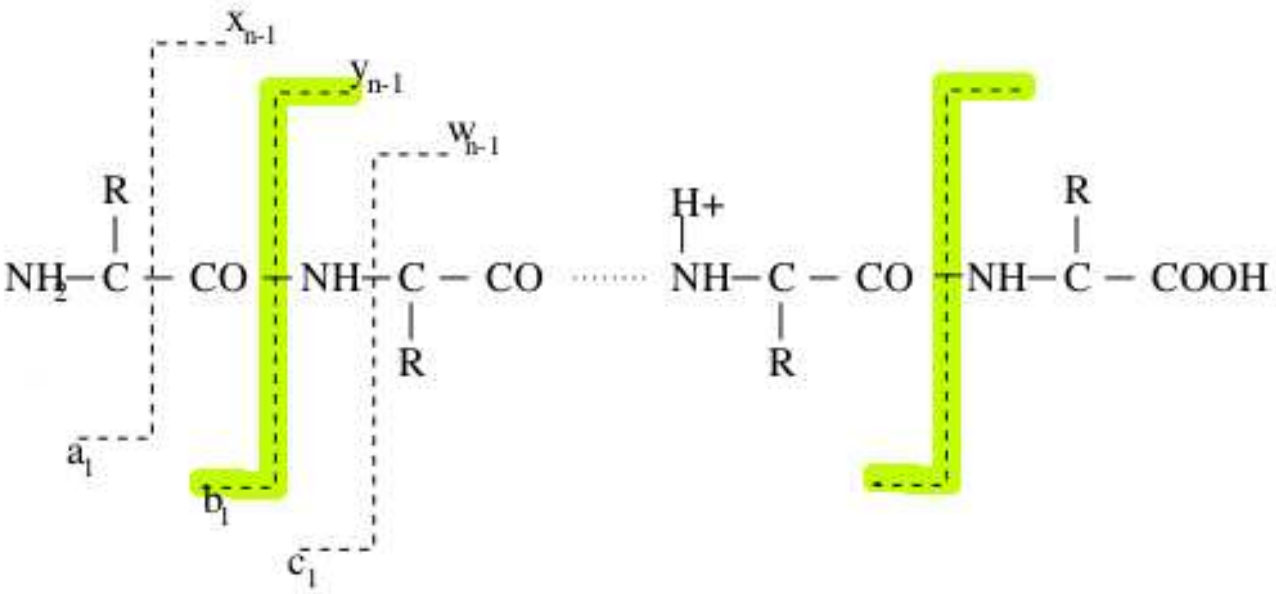
Z bunky sme izolovali určitý proteín, chceme zistiť jeho sekvenciu.

Hmotnostná spektrometria (mass spectometry)

- Meria pomer hmostnosť/náboj molekúl vo vzorke
- Používa sa na identifikáciu proteínov
- Proteín nasekáme enzýmom trypsín (seká na `[KR]{P}`) na peptidy
- Meriame hmostnosť kúskov, porovnáme s databázou proteínov.
- Tandemová hmotnostná spektrometria (MS/MS) ďalej fragmentuje každý kúsok a dosiahne podrobnejšie spektrum, ktoré obsahuje viac informácie
- V niektorých prípadoch tak vieme sekvenciu proteínu určiť priamo z MS/MS, bez databázy proteínov

Tandemová hmotnostná spektrometria MS/MS

Štiepenie peptidu na prefixy a sufixy



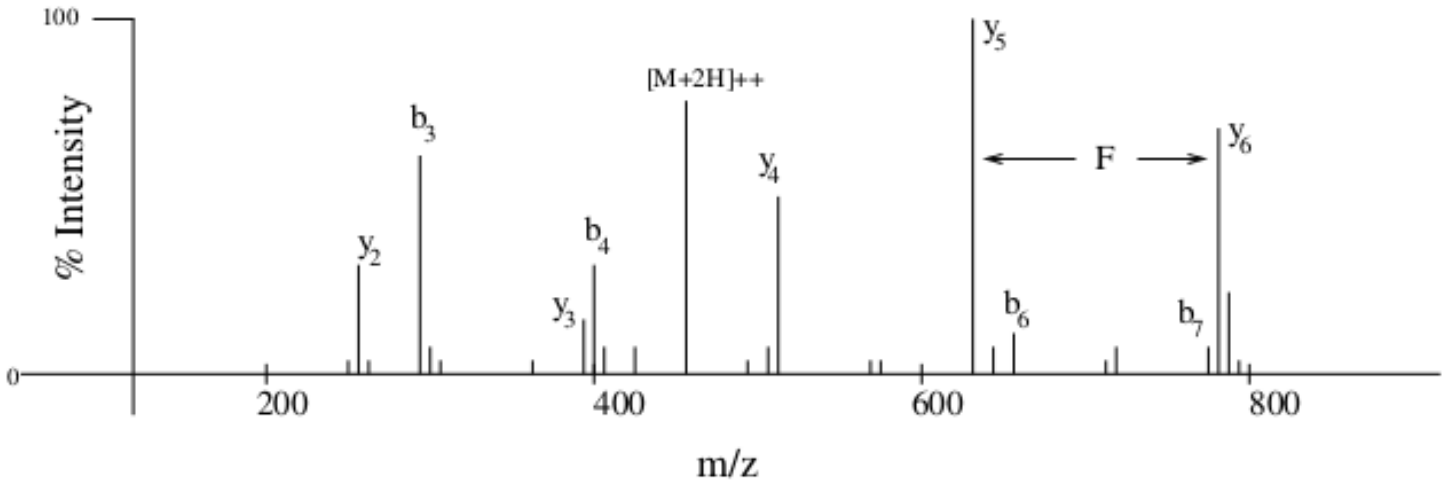
zdroj: Bafna and Reinert

b-ióny: prefixy

y-ióny: sufixy

Tandemová hmotnostná spektrometria MS/MS

88	145	292	405	534	663	778	924	b-ions
S	G	F	L	E	E	D	K	
924	837	780	633	520	391	262	141	y-ions



zdroj: Bafna and Reinert

Sekvenovanie peptidov pomocou MS/MS

Vstup: celková hmotnosť peptidu M,
hmotnosti aminokyselín $a[1], \ldots, a[20]$ (celé čísla),
spektrum ako tabuľka $f[0], \ldots, f[M]$, ktorá hmotnosti určí skóre
podľa signálu v okolí príslušného bodu grafu

Označenie:
Nech $x = x_1 \ldots x_k$ je postupnosť aminokyselín
Nech $m(x) = \sum_{j=1}^{k} a[x_j]$ je hmotnosť x
Nech $\mathcal{M}_P(x) = \{m(x_1 \ldots x_j) \mid j = 1, \ldots, k\}$ sú hmotnosti prefixov x
Nech $\mathcal{M}_S(x) = \{m(x_j \ldots x_k) \mid j = 1, \ldots, k\}$ sú hmotnosti sufixov x

Problém 1: uvažujeme iba b-ióny (prefixy)

Výstup: postupnosť aminokyselín x taká, že $m(x) = M$ a
$\sum_{m \in \mathcal{M}_P(x)} f[m]$ je maximálna možná

Príklad

Uvažujme len 3 aminokyseliny X,Y,Z

$M = 23$, $a[X] = 4$, $a[Y] = 6$, $a[Z] = 7$

m	4	6	7	11	12	17	18	19
$f[m]$	1	1	1	1	1	1	1	1

Hmotnosti prefixov $\mathcal{M}_P(XZYY) =$
$\{m(), m(X), m(XZ), m(XZYY), m(XZYY)\} = \{0, 4, 11, 17, 23\}$

Hmotnosti sufixov $\mathcal{M}_S(XZYY) =$
$\{m(), m(Y), m(YY), m(ZYY), m(XZYY)\} = \{0, 6, 12, 19, 23\}$

Skóre XZYY: $\sum_{m \in \mathcal{M}_P(ZYXX)} f[m] = 0 + 1 + 1 + 1 + 0 = 3$

Skóre XZXXX: $\sum_{m \in \mathcal{M}_P(ZYZZZ)} f[m] =$
$f[0] + f[4] + f[11] + f[15] + f[19] + f[23] = 0 + 1 + 1 + 0 + 1 + 0 = 3$

Sekvenovanie peptidov pomocou MS/MS

Problém 2: uvažujeme prefixy aj sufixy, sčítame ich skóre

Výstup: postupnosť aminokyselín x taká, že $m(x) = M$ a $\sum_{m \in \mathcal{M}_P(x)} f[m] + \sum_{m \in \mathcal{M}_S(x)} f[m]$ je maximálna možná

Problém 3: uvažujeme prefixy aj sufixy, sčítame ich skóre, ale každú hmotnosť započítame najviac raz

Výstup: postupnosť aminokyselín x taká, že $m(x) = M$ a $\sum_{m \in \mathcal{M}_P(x) \cup \mathcal{M}_S(x)} f[m]$ je maximálna možná

Príklad

$M = 23,\ a[X] = 4,\ a[Y] = 6,\ a[Z] = 7$

m	4	6	7	11	12	17	18	19
$f[m]$	1	1	1	1	1	1	1	1

$\mathcal{M}_P(XZYY) = \{0, 4, 11, 17, 23\} \quad \mathcal{M}_S(XZYY) = \{0, 6, 12, 19, 23\}$

$\mathcal{M}_P(XZXXX) = \{0, 4, 11, 15, 19, 23\}$
$\mathcal{M}_S(XZXXX) = \{0, 4, 8, 12, 19, 23\}$

Problém 2: $\sum_{m \in \mathcal{M}_P(x)} f[m] + \sum_{m \in \mathcal{M}_S(x)} f[m]$

Skóre XZYY: $0 + 1 + 1 + 1 + 0 + 0 + 1 + 1 + 1 + 0 = 6$

Skóre XZXXX: $0 + 1 + 1 + 0 + 1 + 0 + 0 + 1 + 0 + 1 + 1 + 0 = 6$

Problém 3: $\sum_{m \in \mathcal{M}_P(x) \cup \mathcal{M}_S(x)} f[m]$

XZYY: $\{0, 4, 6, 11, 12, 17, 19, 23\},\ 1 + 1 + 1 + 1 + 1 + 1 + 0 = 6$

XZXXX: $\{0, 4, 8, 11, 12, 15, 19, 23\},\ 1 + 0 + 1 + 1 + 0 + 1 + 0 = 4$

Ekvivalencia problémov

Problém 2: maximalizujeme $\sum_{m \in \mathcal{M}_P(x)} f[m] + \sum_{m \in \mathcal{M}_S(x)} f[m]$

Iná formulácia: maximalizujeme $\sum_{m \in \mathcal{M}_p(x)} g[m]$
kde $g[m] = f[m] + f[M - m]$

Ekvivalencia problémov

Problém 3: maximalizujeme $\sum_{m \in \mathcal{M}_P(x) \cup \mathcal{M}_S(x)} f[m]$

Iná formulácia: maximalizujeme $\sum_{m \in \mathcal{M}_p(x) \cup \mathcal{M}_S(x), m \leq M/2} h[m]$

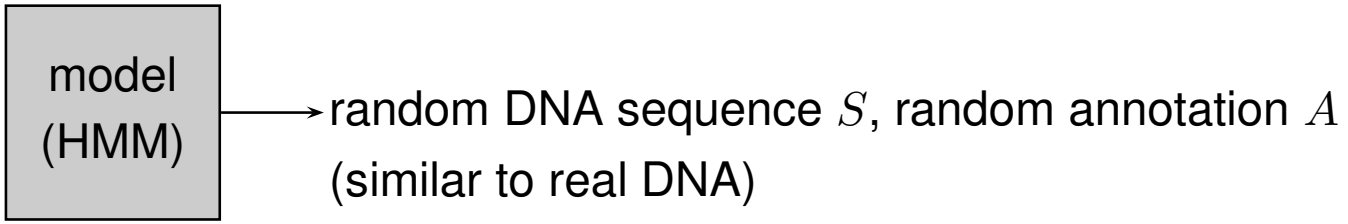
kde $h[m] = \begin{cases} f[m] + f[M-m] & \text{ak } m < M/2 \\ f[m] & \text{ak } m = M/2 \end{cases}$

Algorithms for HMMs

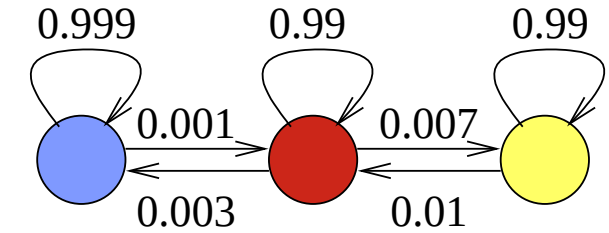
Broňa Brejová

October 28, 2021

Recall: HMM (hidden Markov model, skrytý Markovov model)



$\Pr(S, A)$ – probability that the model generates pair (S, A).



Assume the model starts in the blue state

$$\Pr(\text{[redacted]g}) = 0.27 \cdot 0.001 \cdot 0.27 \cdot 0.99 \cdot 0.24 \cdot 0.99 \cdot 0.28 = 4.8 \cdot 10^{-6}$$

$$\Pr(\text{[redacted]g}) = 0.27 \cdot 0.999 \cdot 0.23 \cdot 0.999 \cdot 0.27 \cdot 0.999 \cdot 0.23 = 0.0038$$

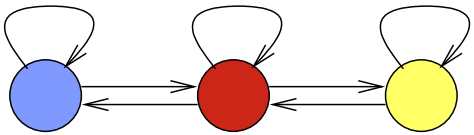
Another toy example: weather

- Period of low atmospheric pressure: mostly raining
- Period of high atmospheirc pressure: mostly sunny

Each period typically lasts several days

Exercise: Represent by an HMM

Recall: Parameters of HMMs (notation)



Sequence $S = S_1, \ldots, S_n$

Annotation $A = A_1, \ldots, A_n$

Model parameters:

Transition probability $a(u, v) = \Pr(A_{i+1} = v | A_i = u)$,

Emission probability $e(u, x) = \Pr(S_i = x | A_i = u)$,

Starting probability $\pi(u) = \Pr(A_1 = u)$.

a			
	0.99	0.007	0.003
	0.01	0.99	0
	0.001	0	0.999

e	a	c	g	t
	0.24	0.27	0.28	0.21
	0.26	0.22	0.22	0.30
	0.27	0.23	0.23	0.27

The resulting probability:

$\Pr(A, S) = \pi(A_1)e(A_1, S_1) \prod_{i=2}^{n} a(A_{i-1}, A_i)e(A_i, S_i)$

Viterbi algorithm

For a given HMM and sequence S,
find the most probable annotation (state path)
$A = \arg\max_A \Pr(A, S) = \arg\max_A \Pr(A \mid S)$

Any ideas?

Recall our example:

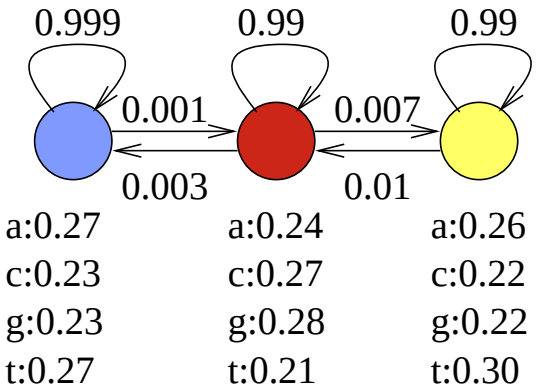
$$\Pr(\text{[redacted]g}) = 0.27 \cdot 0.001 \cdot 0.27 \cdot 0.99 \cdot 0.24 \cdot 0.99 \cdot 0.28 = 4.8 \cdot 10^{-6}$$

$$\Pr(\text{[redacted]g}) = 0.27 \cdot 0.999 \cdot 0.23 \cdot 0.999 \cdot 0.27 \cdot 0.999 \cdot 0.23 = 0.0038$$

Viterbi algorithm

Find the most probable state path $A = \arg\max_A \Pr(A, S)$

Subproblem $V[u, i]$: probability of the most probable state path generating $S_1 S_2 \ldots S_i$ and ending in state u



$V[u,i]$	a	c	a	g

Viterbi algorithm

Subproblem $V[u, i]$: probability of the most probable state path generating $S_1 S_2 \ldots S_i$ and ending in state u

Recurrence?

$V[u, 1] =$

$V[u, i] =$

Recall notation:

Sequence $S = S_1, \ldots, S_n$, annotation $A = A_1, \ldots, A_n$
Transition probability $a(u, v) = \Pr(A_{i+1} = v | A_i = u)$,
Emission probability $e(u, x) = \Pr(S_i = x | A_i = u)$,
Starting probability $\pi(u) = \Pr(A_1 = u)$.
$\Pr(A, S) = \pi(A_1) e(A_1, S_1) \prod_{i=2}^{n} a(A_{i-1}, A_i) e(A_i, S_i)$

Viterbi algorithm

Subproblem $V[u, i]$: probability of the most probable state path generating $S_1 S_2 \dots S_i$ and ending in state u

Recurrence:

$V[u, 1] = \pi_u \cdot e_{u,S_1}$

$V[u, i] = \max_w V[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Algorithm, final answer, running time?

Recall notation:

Sequence $S = S_1, \dots, S_n$, annotation $A = A_1, \dots, A_n$

Transition probability $a(u, v) = \Pr(A_{i+1} = v | A_i = u)$,

Emission probability $e(u, x) = \Pr(S_i = x | A_i = u)$,

Starting probability $\pi(u) = \Pr(A_1 = u)$.

$\Pr(A, S) = \pi(A_1) e(A_1, S_1) \prod_{i=2}^{n} a(A_{i-1}, A_i) e(A_i, S_i)$

Viterbi algorithm (overview)

Goal: Find the most probable state path $A = \arg\max_A \Pr(A, S)$

Subproblem $V[u, i]$: probability of the most probable state path generating $S_1 S_2 \ldots S_i$ and ending in state u

Recurrence:

$V[u, 1] = \pi_u \cdot e_{u,S_1}$

$V[u, i] = \max_w V[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Algorithm:

Initialize $V[*, 1]$

for $i = 2 \ldots n$ (n=length of S)

    for $u = 1 \ldots m$ (m =number of states)

        compute $V[u, i]$, keep best w in $B[u, i]$

Maximum $V[u, n]$ over all u is $\max_A \Pr(A, S)$

Retrieve the full path using matrix B

Dynamic programming in $O(nm^2)$ time

Second problem: overall probability of S

Viterbi computes $\arg\max_A \Pr(A, S)$

Now we want $\Pr(S) = \sum_A \Pr(A, S)$

Usefull e.g. to compare different models, which is more likely to produce S

Any ideas?

Recall our example:

$$\Pr(\text{[redacted]g}) = 0.27 \cdot 0.001 \cdot 0.27 \cdot 0.99 \cdot 0.24 \cdot 0.99 \cdot 0.28 = 4.8 \cdot 10^{-6}$$

$$\Pr(\text{[redacted]g}) = 0.27 \cdot 0.999 \cdot 0.23 \cdot 0.999 \cdot 0.27 \cdot 0.999 \cdot 0.23 = 0.0038$$

Forward algorithm (dopredný algoritmus)

Computes overall probability that the model emits S

$\Pr(S) = \sum_A \Pr(A, S)$

Subproblem $F[u, i]$: probability that in i steps we generate $S_1, S_2, \ldots S_i$ and end in state u.

$F[u, i] = \Pr(A_i = u \wedge S_1, S_2, \ldots, S_i) =$

$\sum_{A_1,\ldots,A_{i-1},A_i=u} \Pr(A_1, A_2, ..., A_i \wedge S_1, S_2, ..., S_i)$

Recurrence?

$F[u, 1] =$

$F[u, i] =$

Recall Viterbi recurrence:

$V[u, 1] = \pi_u \cdot e_{u,S_1}$

$V[u, i] = \max_w V[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Forward algorithm

Computes overall probability that the model emits S

$\Pr(S) = \sum_A \Pr(A, S)$

Subproblem $F[u, i]$: probability that in i steps we generate $S_1, S_2, \ldots S_i$ and end in state u.

Recurrence:

$F[u, 1] = \pi_u \cdot e_{u,S_1}$

$F[u, i] = \sum_w F[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Recall Viterbi recurrence:

$V[u, 1] = \pi_u \cdot e_{u,S_1}$

$V[u, i] = \max_w V[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Forward algorithm

Computes overall probability that the model emits S

$\Pr(S) = \sum_A \Pr(A, S)$

Subproblem $F[u, i]$: probability that in i steps we generate $S_1, S_2, \ldots S_i$ and end in state u.

Recurrence:

$F[u, 1] = \pi_u \cdot e_{u,S_1}$

$F[u, i] = \sum_w F[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Result? $\Pr(S) =$

Running time?

Forward algorithm

Computes overall probability that the model emits S
$\Pr(S) = \sum_A \Pr(A, S)$

Subproblem $F[u, i]$**:** probability that in i steps we generate $S_1, S_2, \ldots S_i$ and end in state u.

Recurrence:

$F[u, 1] = \pi_u \cdot e_{u,S_1}$
$F[u, i] = \sum_w F[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Result $\Pr(S) = \sum_u F[u, n]$

Running time $O(nm^2)$

Third problem: probability that S_i was generated in state u

$\Pr(A_i = u \mid S) = \frac{\Pr(A_i=u,S)}{\Pr(S)}$

$\Pr(A_i = u, S) = \sum_{A:A_i=u} \Pr(A, S)$

Compute this by a combination of forward and backward algorithms

$F[u, i]$: probability that in i steps we generate $S_1, S_2, \ldots S_i$ and end in state u.

$B[u, i]$: probability that if we start at u at position i, we will generate $S_{i+1} \ldots, S_n$ in the next steps

$\Pr(A_i = u, S) = F[u, i] \cdot B[u, i]$

Backward algorithm (spätný algoritmus)

Forward algorithm $F[u, i]$: probability that in i steps we generate $S_1, S_2, \ldots S_i$ and end in state u.

$F[u, 1] = \pi_u \cdot e_{u,S_1}$
$F[u, i] = \sum_w F[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Backward algorithm $B[u, i]$: probability that if we start at u at position i, we will generate $S_{i+1} \ldots, S_n$ in the next steps

How to compute $B[u, i]$**?**

Backward algorithm (spätný algoritmus)

Forward algorithm $F[u, i]$: probability that in i steps we generate $S_1, S_2, \ldots S_i$ and end in state u.

$F[u, 1] = \pi_u \cdot e_{u,S_1}$
$F[u, i] = \sum_w F[w, i-1] \cdot a_{w,u} \cdot e_{u,S_i}$

Backward algorithm $B[u, i]$: probability that if we start at u at position i, we will generate $S_{i+1} \ldots, S_n$ in the next steps

$B[u, n] = 1$
$B[u, i] = \sum_w F[w, i+1] \cdot a_{u,w} \cdot e_{w,S_{i+1}}$

Exercise: How to use matrix B to compute $\Pr(S)$?

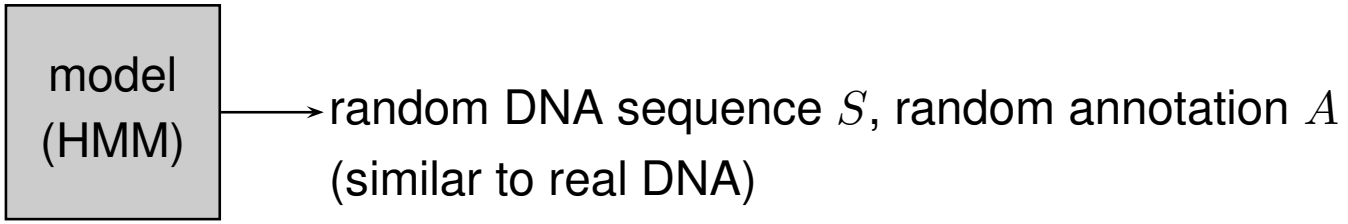
Posterior decoding

Using forward/backward we can compute
$\Pr(A_i = u \mid S)$ for each u and i (posterior probabilities of states)
in $O(nm^2)$ overall time

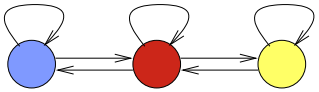
We can also select A such that $A_i = \max_i \Pr(A_i = u \mid S)$
Advantage: This takes into account suboptimal state paths
Disadvantage: $\Pr(A \mid S)$ can be zero or very low

Another option: use posterior probabilities to assign confidence to parts of prediction from Viterbi

Recall: Finding genes with HMMs



$\Pr(S, A)$ – probability that the model generates pair (S, A).

- **Determine states and transitions of the model:** by hand based on your knowledge about the gene structure 
- **Parameter training:** emission and transition probabilities are determined based on the real sequences with known genes (**training set**)
- **Use:** for a new sequence S, find the most probable annotation $A = \arg\max_A \Pr(A|S)$
  Viterbi algorithm in $O(nm^2)$ (dynamic programming)

Parameter training

- States and allowed transitions typically manually
- Probabilities of transition, emission, starting usually automatically from training adata
- More complex models with more parameters need more training data
  Otherwise **overfitting**: model fits training data very well but behaves poorly on unseen examples
- To test acurracy of the model use a separate testing set not used for training.

HMM parameter training from annotated sequences

Input: state diagram of the model and a training set of sequences and state paths $(S^{(1)}, A^{(1)}), (S^{(2)}, A^{(2)}), \ldots$

Goal: choose parameters maximalizing their likelihood in the model $\arg\max_{a,e,\pi} \prod_i \Pr(S^{(i)}, A^{(i)} | a, e, \pi)$

This is achieved by using observed frequencies

Fir example $a_{u,v}$: find all occurrences of state u and find out how often is it followed by v

HMM parameter training from unannotated sequences

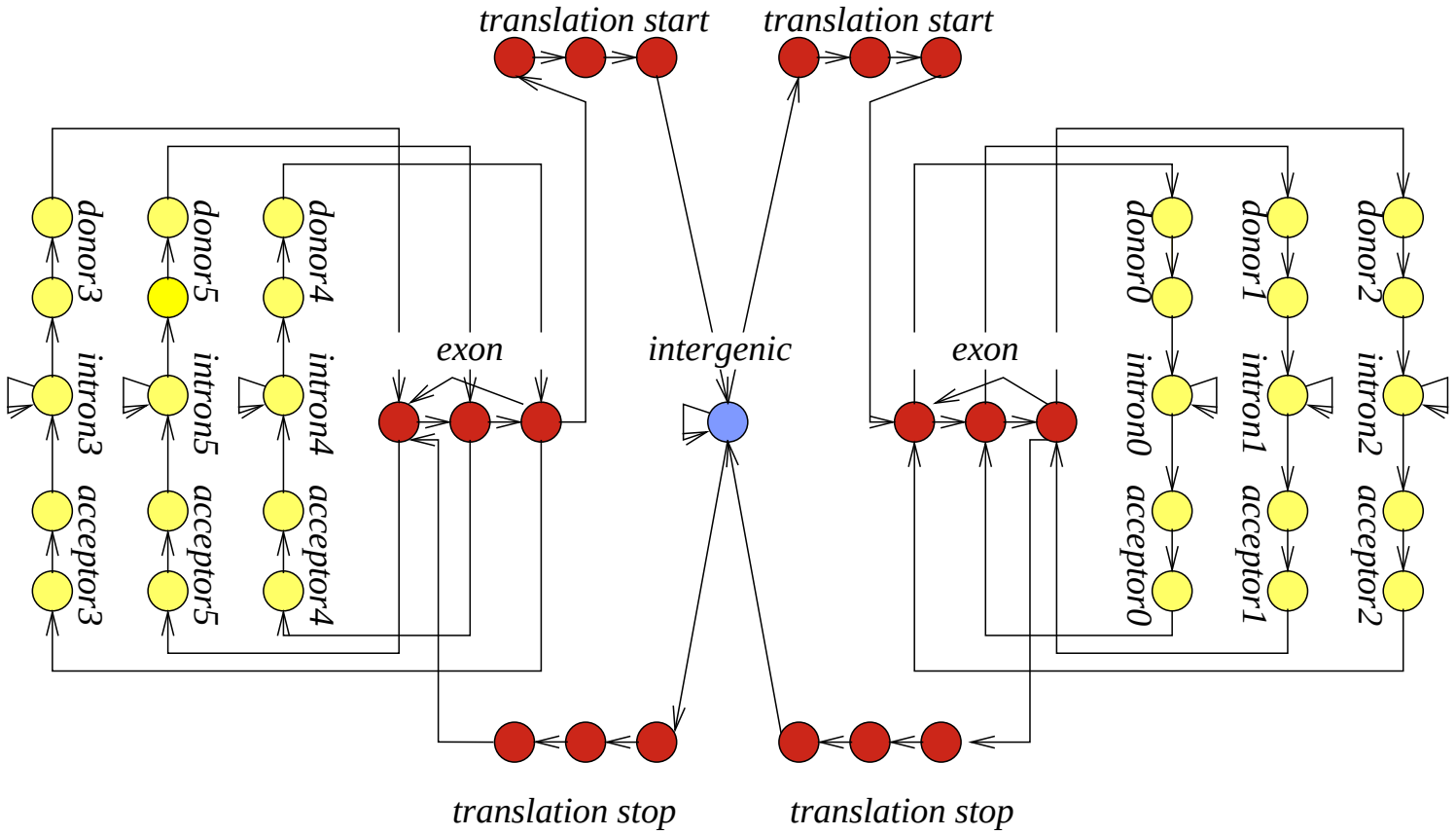
Input: state diagram of the model and a training set of sequences $S^{(1)}, S^{(2)}, \ldots$, state paths $A^{(1)}$ unknown

Goal: choose parameters maximalizing their likelihood in the model $\arg\max_{a,e,\pi} \prod_i \Pr(S^{(i)}|a, e, \pi)$

Baum-Welch algorithm (version of expectation maximization, EM).
Iterative heuristic algorithm improving parameters until convergence.
Each iteration forward and backward algorithms

Designing state diagram of HMM

We have seen example of gene finding



Two examples

- How would you modify gene finding HMM so that intergenic regions have length at least 10?
  What about lengths of introns and exons?

- Create a model of prokaryotic genes without introns
  which are grouped into operons,
  each operon starting with a promoter containing sequence
  TATAAT 10bp before transcription start.
  Genes in an operon are separated by short untranslated regions.
  Operons are separated by some untranscribed regions.

Substitution Models

Tomáš Vinař

November 4, 2021

Substitution models, notation

$P(b|a,t)$: probability that if we start with symbol a, after time t we will see symbol b

Transition probability matrix:

$$S(t) = \begin{pmatrix} P(A|A,t) & P(C|A,t) & P(G|A,t) & P(T|A,t) \\ P(A|C,t) & P(C|C,t) & P(G|C,t) & P(T|C,t) \\ P(A|G,t) & P(C|G,t) & P(G|G,t) & P(T|G,t) \\ P(A|T,t) & P(C|T,t) & P(G|T,t) & P(T|T,t) \end{pmatrix}$$

Substitution models, basic properties

- $S(0) = I$
- $\lim_{t\to\infty} S(t) = \begin{pmatrix} \pi_A & \pi_C & \pi_G & \pi_T \\ \pi_A & \pi_C & \pi_G & \pi_T \\ \pi_A & \pi_C & \pi_G & \pi_T \\ \pi_A & \pi_C & \pi_G & \pi_T \end{pmatrix}$

  Distribution π is called stationary (equilibrium)
- $S(t_1 + t_2) = S(t_1)S(t_2)$ (multiplicativity)
- Jukes-Cantor model should also satisfy

$$S(t) = \begin{pmatrix} 1-3s(t) & s(t) & s(t) & s(t) \\ s(t) & 1-3s(t) & s(t) & s(t) \\ s(t) & s(t) & 1-3s(t) & s(t) \\ s(t) & s(t) & s(t) & 1-3s(t) \end{pmatrix}$$

$$S(t) = \begin{pmatrix} 1-3s(t) & s(t) & s(t) & s(t) \\ s(t) & 1-3s(t) & s(t) & s(t) \\ s(t) & s(t) & 1-3s(t) & s(t) \\ s(t) & s(t) & s(t) & 1-3s(t) \end{pmatrix}$$

$$S(2t) = S(t)^2 =$$

$$= \begin{pmatrix} 1-6s(t)+12s(t)^2 & 2s(t)-4s(t)^2 & 2s(t)-4s(t)^2 & 2s(t)-4s(t)^2 \\ 2s(t)-4s(t)^2 & 1-6s(t)+12s(t)^2 & 2s(t)-4s(t)^2 & 2s(t)-4s(t)^2 \\ 2s(t)-4s(t)^2 & 2s(t)-4s(t)^2 & 1-6s(t)+12s(t)^2 & 2s(t)-4s(t)^2 \\ 2s(t)-4s(t)^2 & 2s(t)-4s(t)^2 & 2s(t)-4s(t)^2 & 1-6s(t)+12s(t)^2 \end{pmatrix}$$

$$\approx \begin{pmatrix} 1-6s(t) & 2s(t) & 2s(t) & 2s(t) \\ 2s(t) & 1-6s(t) & 2s(t) & 2s(t) \\ 2s(t) & 2s(t) & 1-6s(t) & 2s(t) \\ 2s(t) & 2s(t) & 2s(t) & 1-6s(t) \end{pmatrix}$$

for $t \to 0$

Substitution rate matrix (matica rýchlostí, matica intenzít)

- Substitution rate matrix for Jukes-Cantor model:

$$R = \begin{pmatrix} -3\alpha & \alpha & \alpha & \alpha \\ \alpha & -3\alpha & \alpha & \alpha \\ \alpha & \alpha & -3\alpha & \alpha \\ \alpha & \alpha & \alpha & -3\alpha \end{pmatrix}$$

- For very small t we have $S(t) \approx I + Rt$
- Rate α is the probablity of a change per unit of time for very small t, or derivative of $s(t)$ with respect to t at $t = 0$
- Solving the differential equation for the Jukes-Cantor model we get $s(t) = (1 - e^{-4\alpha t})/4$

Jukes-Cantor model

$$S(t) = \begin{pmatrix} (1+3e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 \\ (1-e^{-4\alpha t})/4 & (1+3e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 \\ (1-e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 & (1+3e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 \\ (1-e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 & (1-e^{-4\alpha t})/4 & (1+3e^{-4\alpha t})/4 \end{pmatrix}$$

The rate matrix is typically normalized so that there is on average one substitution per unit of time, here $\alpha = 1/3$

Jukes-Cantor model, summary

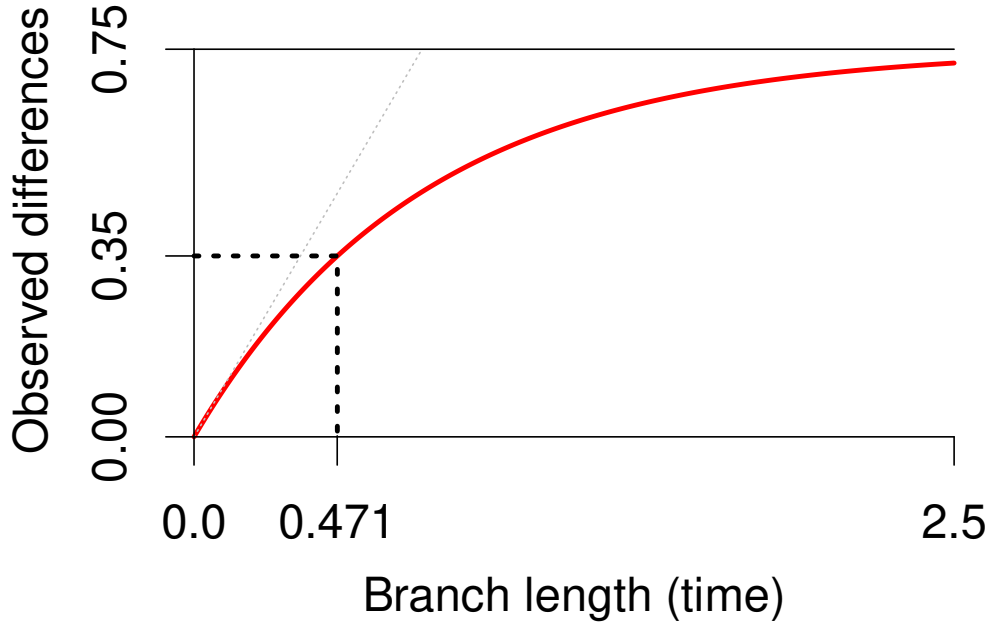
- $S(t)$: matrix 4×4, where $S(t)_{a,b} = P(b|a,t)$ is the probability that if we start with base a, after time t we have base b.
- Jukes-Cantor model assumes that $P(b|a,t)$ is the same for all $a \neq b$
- For a given time t, off-diagonal elements are $s(t)$, diagonal $1 - 3s(t)$
- Rate matrix R: for J-C off-diagonal α, diagonal -3α
- For very small t we have $S(t) \approx I - Rt$
- Rate α is the probablity of a change per unit of time for very small t, or derivative of $s(t)$ with respect to t for $t = 0$
- Solving the differential equation for the Jukes-Cantor model, we get $s(t) = (1 - e^{-4\alpha t})/4$
- The rate matrix is typically normalized so that there is on average one substitution per unit of time, that is, $\alpha = 1/3$

Correction of evolutionary distances

$$\Pr(X_{t_0+t} = C \,|\, X_{t_0} = A) = \tfrac{1}{4}(1 - e^{-\frac{4}{3}t})$$

The expected number of observed changes per base in time t:

$$D(t) = \Pr(X_{t_0+t} \neq X_{t_0}) = \tfrac{3}{4}(1 - e^{-\frac{4}{3}t})$$



Correction of observed distances

$$D = \frac{3}{4}\left(1 - e^{-\frac{4}{3}t}\right) \qquad \Rightarrow \qquad t = -\frac{3}{4}\ln\left(1 - \frac{4}{3}D\right)$$

More complex models

- General rate matrix R

$$R = \begin{pmatrix} \cdot & \mu_{AC} & \mu_{AG} & \mu_{AT} \\ \mu_{CA} & \cdot & \mu_{CG} & \mu_{CT} \\ \mu_{GA} & \mu_{GC} & \cdot & \mu_{GT} \\ \mu_{TA} & \mu_{TC} & \mu_{TG} & \cdot \end{pmatrix}$$

- μ_{xy} is the rate at which base x changes to a different base y
- Namely, $\mu_{xy} = \lim_{t \to 0} \frac{\Pr(y \mid x,t)}{t}$
- The diagonal is added so that the sum of each row is 0
- There are models with a smaller number of parameters (compromise between J-C and an arbitrary matrix)

Kimura model

- A and G are purines, C and T pyrimidines
- Purines more often change to other purines
  and pyrimidines to pyrimidines
- Transition: change within group $A \Leftrightarrow G$, $C \Leftrightarrow T$,
  Transversion: change to a different group $\{A, G\} \Leftrightarrow \{C, T\}$
- Two parameters: rate of transitions α, rate of transversions β
- $$R = \begin{pmatrix} -2\beta - \alpha & \beta & \alpha & \beta \\ \beta & -2\beta - \alpha & \beta & \alpha \\ \alpha & \beta & -2\beta - \alpha & \beta \\ \beta & \alpha & \beta & -2\beta - \alpha \end{pmatrix}$$

HKY model (Hasegawa, Kishino, Yano)

- Extension of Kimura model, which allows different probabilities of A, C, G, T in the equilibrium
- If we set time to infinity, original base is not important, base frequencies stabilize in an equilibrium.
- Jukes-Cantor has probability of each base in the equilibrium 1/4.
- In HKY the equilibrium frequencies $\pi_A, \pi_C, \pi_G, \pi_T$ are parameters (summing to 1)
- Parameter κ: transition / transversion ratio (α/β)
- Rate matrix:

$$\mu_{x,y} = \begin{cases} \kappa\pi_y & \text{if mutation from } x \text{ to } y \text{ is transition} \\ \pi_y & \text{if mutation from } x \text{ to } y \text{ is transversion} \end{cases}$$

From rate matrix R to transition probabilities $S(t)$

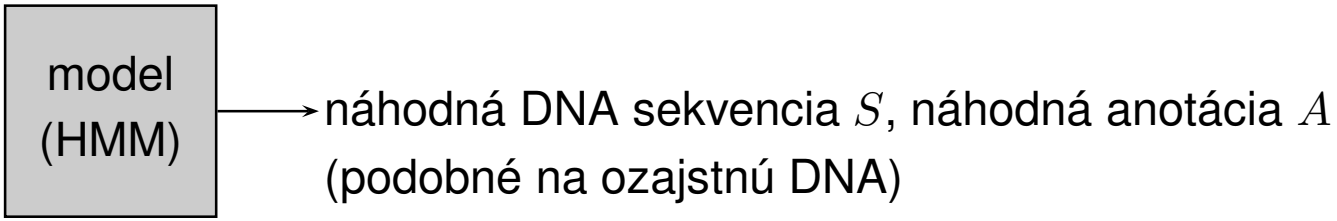
- J-C and some other models have explicit formulas for $S(t)$
- For more complex models, such formulas are not available
- In general, $S(t) = e^{Rt}$
- Exponential of a matrix A is defined as $e^A = \sum_{k=0}^{\infty} \frac{1}{k!} A^k$
- If R is diagonalized $R = UDU^{-1}$, where D is a diagonal matrix, then $e^{Rt} = Ue^{Dt}U^{-1}$ and the exponential function is applied to the diagnal elements of D
- Diagonalization always exists for symmetric matrices R (the diagonal contains eigenvalues)

Algoritmy pre HMM

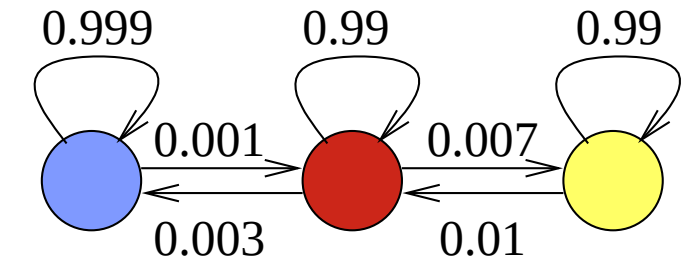
Askar Gafurov

7.11.2019

Opakovanie: HMM (skrytý Markovov model)



$\Pr(S, A)$ – pravdepodobnosť, že model vygeneruje pár (S, A).

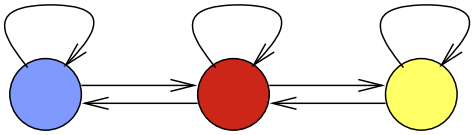


Predpokladajme, že model vždy začína v modrom stave.

$\Pr(\blacksquare\blacksquare) = 0.27 \cdot 0.001 \cdot 0.27 \cdot 0.99 \cdot 0.24 = 0.000017$

$\Pr(\blacksquare\blacksquare\blacksquare) = 0.27 \cdot 0.999 \cdot 0.23 \cdot 0.999 \cdot 0.27 = 0.017$

Parametre HMM (označenie)



Sekvencia $S_1, \ldots, S_n$

Anotácia $A_1, \ldots, A_n$

Parametre modelu:

Prechodová pravdepodobnosť $a(u, v) = \Pr(A_{i+1} = v | A_i = u)$,

Emisná pravdepodobnosť $e(u, x) = \Pr(S_i = x | A_i = u)$,

Počiatočná pravdepodobnosť $\pi(u) = \Pr(A_1 = u)$.

a			
	0.99	0.007	0.003
	0.01	0.99	0
	0.001	0	0.999

e	a	c	g	t
	0.24	0.27	0.28	0.21
	0.26	0.22	0.22	0.30
	0.27	0.23	0.23	0.27

Výsledná pravdepodobnosť: $\Pr(A_1, \ldots, A_n, S_1, \ldots, S_n) = \pi(A_1)e(A_1, S_1) \prod_{i=2}^{n} a(A_{i-1}, A_i)e(A_i, S_i)$

Viterbiho algoritmus

Pre danú sekvenciu S nájde najpravdepodobnejšiu anotáciu
$A = \arg\max_A \Pr(A|S)$

Dynamické programovanie v čase $O(nm^2)$

Podproblém $V[i,u]$: pravdepodobnosť najpravdepodobnejšej cesty končiacej po i krokoch v stave u, pričom vygeneruje $S_1 S_2 \ldots S_i$

Rekurencia:

$V[1,u] = \pi_u \cdot e_{u,S_1}$
$V[i,u] = \max_w V[i-1,w] \cdot a_{w,u} \cdot e_{u,S_i}$

Algoritmus:

Inicializuj $V[1,*]$
for $i = 2 \ldots n$ (n=dĺžka reťazca)
    for $u = 1 \ldots m$ (m =počet stavov)
        vypočítaj $V[i,u]$
Maximálne $V[n,j]$ je pravdepodobnosť najpravdepodobnejšej cesty

Dopredný algoritmus

Počíta celkovú pravdepodobnosť, že vygenerujeme sekvenciu S
$\Pr(S) = \sum_A Pr(A, S)$

Podproblém $F[i, u]$**:** pravdepodobnosť, že po i krokoch vygenerujeme $S_1, S_2, \ldots S_i$ a dostaneme sa do stavu u.

$F[i, u] = \Pr(A_i = u \wedge S_1, S_2, \ldots, S_i) =$
$\sum_{A_1, A_2, \ldots, A_i = u} \Pr(A_1, A_2, ..., A_i \wedge S_1, S_2, ..., S_i)$

Rekurencia:

$F[1, u] = \pi_u \cdot e_{u,S_1}$
$F[i, u] = \sum_v F[i-1, v] \cdot a_{v,u} \cdot e_{u,S_i}$

Celková pravdepodobnosť $\Pr(S) = \sum_u F[n, u]$

Spätný algoritmus

Obdoba dopredného algoritmu

Dopredný algoritmus: $F[i,u] = \Pr(A_i = u \wedge S_1, \ldots, S_i)$

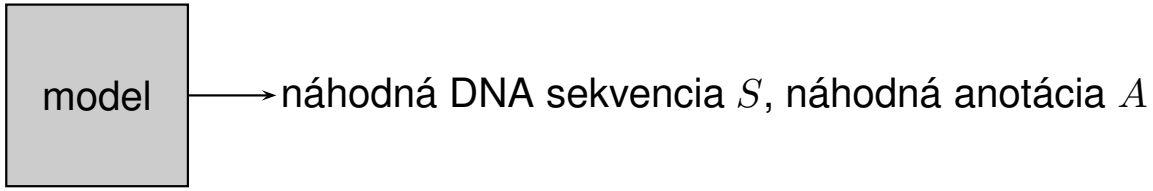
$F[1,u] = \pi_u \cdot e_{u,S_1}$

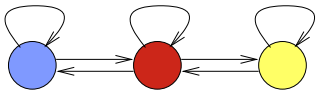
$F[i,u] = \sum_v F[i-1,v] \cdot a_{v,u} \cdot e_{u,S_i}$

$\Pr(S) = \sum_u F[n,u]$

Spätný algoritmus: $B[i,u] = \Pr(S_{i+1} \ldots, S_n | A_i = u)$

Hľadanie génov s HMM



- **Určenie stavov a prechodov v modeli:** ručne, na základe poznatkov o štruktúre génu. 
- **Trénovanie parametrov:** pravdepodobnosti určíme na základe sekvencií so známymi génmi (**trénovacia množina**).
  Model zostavíme tak, aby páry (S, A) s vlastnosťami podobnými skutočným génom mali veľkú pravdepodobnosť $\Pr(S, A)$
- **Použitie:** pre novú sekvenciu S nájdi najpravdepodobnejšiu anotáciu $A = \arg\max_A \Pr(A|S)$ Viterbiho algoritmom

Trénovanie HMM

- Stavový priestor + povolené prechody väčšinou ručne
- Parametre (pravdepodobnosti prechodu, emisie a počiatočné) automaticky z trénovacích sekvencií
- Čím zložitejší model a viac parametrov máme, tým potrebujeme viac trénovacích dát, aby nedošlo k preučeniu, t.j. k situácii, keď model dobre zodpovedá nejakým zvláštnostiam trénovacích dát, nie však ďalším dátam.
- Presnosť modelu testujeme na zvláštnych testovacích dátach, ktoré sme nepoužili na trénovanie.

Trénovanie HMM z anotovaných sekvencií

Vstup: topológia modelu a niekoľko trénovacích párov $S^{(i)}, A^{(i)}$

Cieľ: nastaviť π_u, $e_{u,x}$, $a_{u,v}$ tak, aby $\prod_i \Pr(S^{(i)}, A^{(i)})$ bola čo najväčšia

Dosiahneme jednoduchým počítaním frekvencií

Napr. $a_{u,v}$: nájdeme všetky výskyty stavu u a zistíme, ako často za nimi ide stav v

Trénovanie HMM z neanotovaných sekvencií

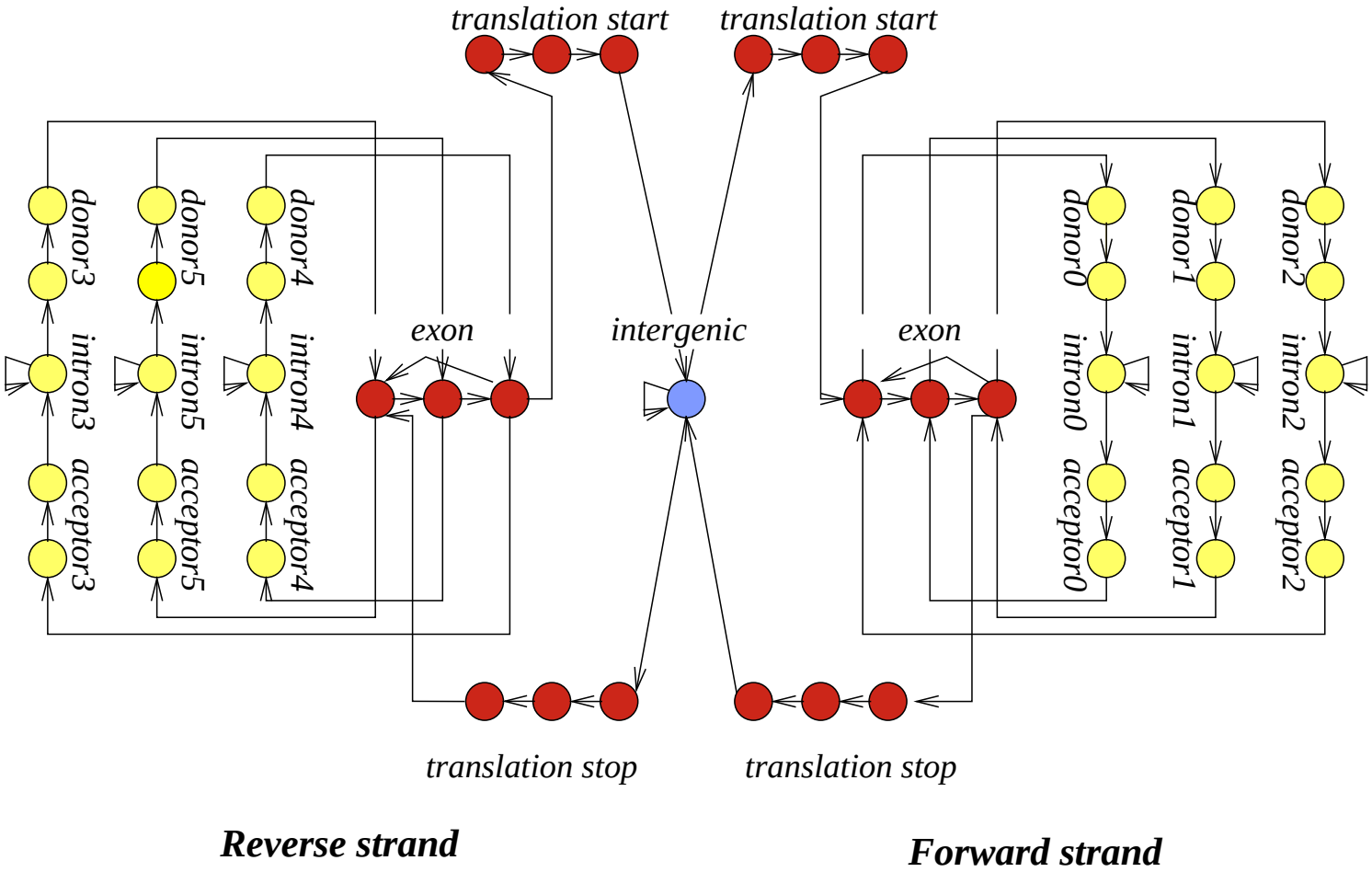
Vstup: topológia modelu a niekoľko trénovacích sekvencií $S^{(i)}$
anotácie $A^{(i)}$ nepoznáme

Cieľ: nastaviť π_u, $e_{u,x}$, $a_{u,v}$ tak, aby $\prod_i \Pr(S^{(i)})$ bola čo najväčšia

Používajú sa heuristické iteratívne algoritmy, napr. Baum-Welchov,
ktorý je verziou všeobecnejšieho algoritmu EM (expectation
maximization).

Tvorba stavového priestoru modelu

Príklad HMM na hľadanie génov



Course Summary

Broňa Brejová

December 16, 2021

Probabilistic models

- Hidden Markov models (gene finding, phylogenetic HMMs for conserved elements, profile HMMs for protein families)
- Phylogenetic trees and substitution models
- Stochastic context-free grammars
- Gibbs sampling
- Maximum likelihood method
- Expectation maximization (EM)

Statistical methods

- Statistical significance, E-value, P-value
- Positive selection test
- Linkage disequilibrium, association mapping

Practice in dynamic programming

- Sequence alignment
  (global, local, affine gaps, saving memory)
- Hidden Markov models (Viterbi and forward algorithms)
- Computation on trees
  (parsimony, Felsenstein algorithm for likelihood)
- Mass spectrometry (MS/MS)
- Secondary RNA structure

Other

- Integer linear programming
- deBruijn graphs
- Clustering and classification

How to model real-life problems

- Consider what data are avilable, what are relevant questions
- Formulate as a computer-science problem (e.g. score optimization)
- Probabilistic models often lead to a systematic choice of a scoring scheme
- The resulting problem often NP hard
  – Heuristics, approximation algorithms
  – ILP and other techniques for exact solutions
  – Can we change problem formulation?
- Testing: are computation results relevant in a given domain? (is our formulation sufficiently realistic?)

Ďalšie predmety

- **Strojové učenie** 2-INF-150, Vinař/Boža (ZS, 4P, 6kr)
- **Vybrané partie z dátových štruktúr** 2-INF-237, Kováč (ZS, 4P, 6kr)
- **Seminár z bioinformatiky (1)-(4)** 2-AIN-50[56],25[12] (oba semestre, 2S, 2kr)
- **Manažment dát** 1-DAV-202, Brejová, Vinař, Boža (LS, 1P/2C, 4kr)
- **Genomika** 2-INF-269, Nosek a kol. (LS, 2P/1C, 4kr)
- **Výzvy súčasnej bioinformatiky** 1-BIN-105, Brejová, Vinař (LS, 2S, 2kr)
- http://compbio.fmph.uniba.sk/vyuka/

Integer Linear Programming

Tomáš Vinař

December 16, 2021

Practical programs for NP-hard problems

They always find the optimal solution, often in reasonable time, but on some inputs very long runtimes

- ILP: CPLEX, Gurobi (commercial), SCIP (non-commercial)
- SAT: Minisat, Lingeling, glucose, CryptoMiniSat, painless
- TSP: Concorde

Other NP-complete problems can be transformed to one of these problems

ILP: Integer linear programming

Linear programming:

real-valued variables $x_1, \ldots, x_n$
minimize $\sum_i a_i x_i$ for given weights $a_1, \ldots, a_n$
under constraints of the form $\sum_i b_i x_i \leq c$

LP can be solved in polynomial time

Integer linear programming:

Add a constraint that some variables are integers or binary

NP-hard problem

Expressing known NP-hard problems as ILP

Knapsack

Given n items with weights $w_1 \dots w_n$ and costs $c_1 \dots c_n$.
Choose a subset so that overall weight is at most T and the overall cost is highest possible?

Expressing known NP-hard problems as ILP

Set cover

We have n subsets $S_1 \ldots, S_n$ of a set $U = \{1 \ldots m\}$.
Choose the smallest number of the input subsets so that their union is the whole set U.

Protein threading

Protein A has a known sequence and structure, protein B only sequence.

Align A and B so that if two amino acids are close in A, their equivalents in B should be "compatible".

Choose "cores" in A which should remain conserved without insertions, deletions and in the same order

Cores are separated by "loops", whose length can arbitrarily change and whose alignements will not be scored

Protein threading, problem formulation

Input: sequence $B = b_1 \dots b_n$,
lengths of m cores $c_1 \dots . c_m$,
scoring tables
– E_{ij}: how well $b_j \dots b_{j+c_i-1}$ agrees with sequence of core i,
– $E_{ijk\ell}$: how well would cores i and k interact, if they start at pos. j, ℓ.

Task: choose starts of cores $x_1, x_2, \dots, x_m$ so that
– they are in the correct order and without overlaps,
– they achieve maximum possible score

Note: we do not specify how to choose cores and scoring tables,
which is a modeling, not an algorithmic problem

Protein threading, ILP

Notation: sequence $B = b_1 \dots b_n$, lengths of m cores $c_1 \dots c_m$,
E_{ij}: how well $b_j \dots b_{j+c_i-1}$ agrees with sequence of core i,
$E_{ijk\ell}$: how well would cores i and k interact, if they start at pos. j, ℓ,
unknown starts of cores $x_1, \dots, x_m$.

ILP formulation:

Protein threading, ILP

Notation: sequence $B = b_1 \dots b_n$, lengths of m cores $c_1 \dots c_m$,
E_{ij}: how well $b_j \dots b_{j+c_i-1}$ agrees with sequence of core i,
$E_{ijk\ell}$: how well would cores i and k interact, if they start at pos. j, ℓ,
unknown starts of cores $x_1, \dots, x_m$.

ILP formulation: