Plán semestra

Skončili sme: editačná vzdialenosť, približné výskyty vzorky

Dnes: *zostavovanie DNA sekvencií, najkratšie spoločné nadslovo*

Streda 1.5.: sviatok

Štvrtok 2.5.: *viacnásobné zarovnanie, opakujúce sa sekvenčné motívy*

Streda 8.5.: sviatok

Štvrtok 9.5.: prezentácie

Streda 15.5.: prezentácie

Štvrtok 16.5.: prezentácie

Prezentácie

- Štvrtok 9.5., streda 15.5., štvrtok 16.5.
- 15-20 minút + diskusia
- Bude k dispozícii notebook, prineste si pdf na USB kľúči, príp. vlastný počítač
- Vyberte z článku niečo zaujímavé, nemusíte pokryť všetko
- Na slajdy nedávajte priveľa textu, použite dosť veľké písmo a kontrastné farby
- Zavádzajte čo najmenej pojmov a označenia. Pojmy, algoritmy a dôkazy radšej ilustrujte na obrázku alebo príklade, než zložitým textom.

Prezentácie

Typická osnova:

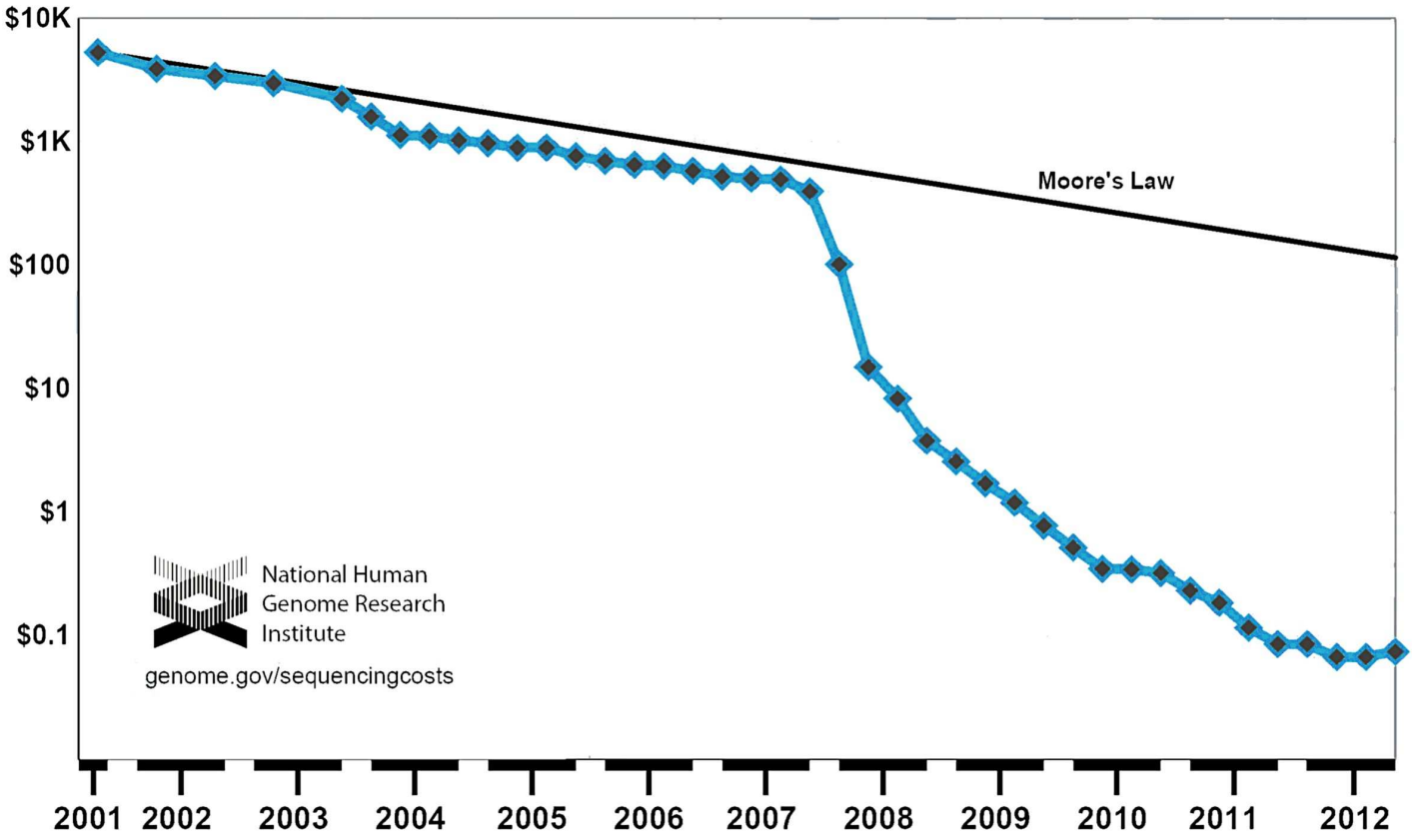
- Úvod: aký problém autori študujú, prečo je dôležitý alebo zaujímavý, má nejaký vzťah k niečomu z prednášok?
- Prehľad výsledkov: V čom je prínos autorov oproti predchádzajúcim prácam?
- Jadro: Vysvetlite časť algoritmu alebo dôkazu, niektoré výsledky z testovania na dátach a pod.
- Záver: zhrnutie prezentácie, váš názor (čo sa Vám na článku páčilo alebo nepáčilo)

DNA sequencing

- Human genome: string of length $3 \cdot 10^9$ over alphabet $\{A, C, G, T\}$
- Sequencing of the human genome: 1988-2004, $3 \cdot 10^9$ dollars
- Today the cost is 20,000 USD or less
- Next generation sequencing: cca from 2004, still rapid development
- Sequencing costs drop faster than Moore's law

Cost per million letters (raw data)

Sequencing costs drop faster than Moore's law

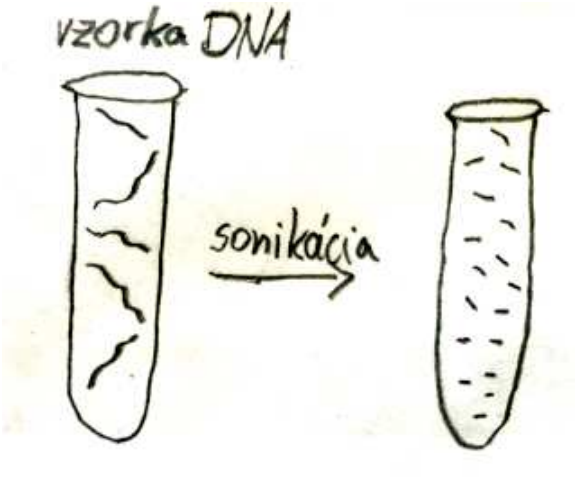


Example of sequencing technology: SOLiD

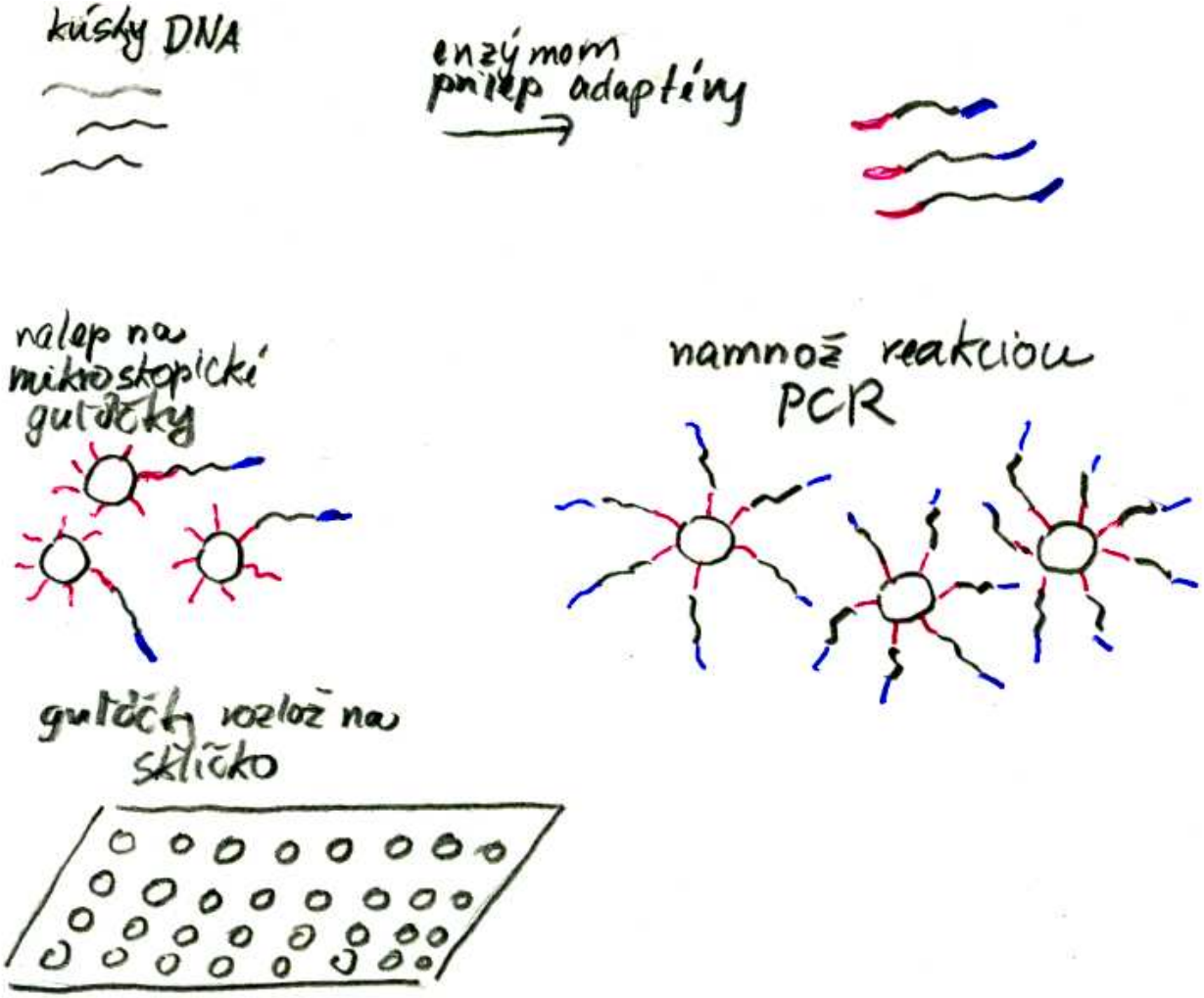
- Massively parallel
- Use of natural enzymes:

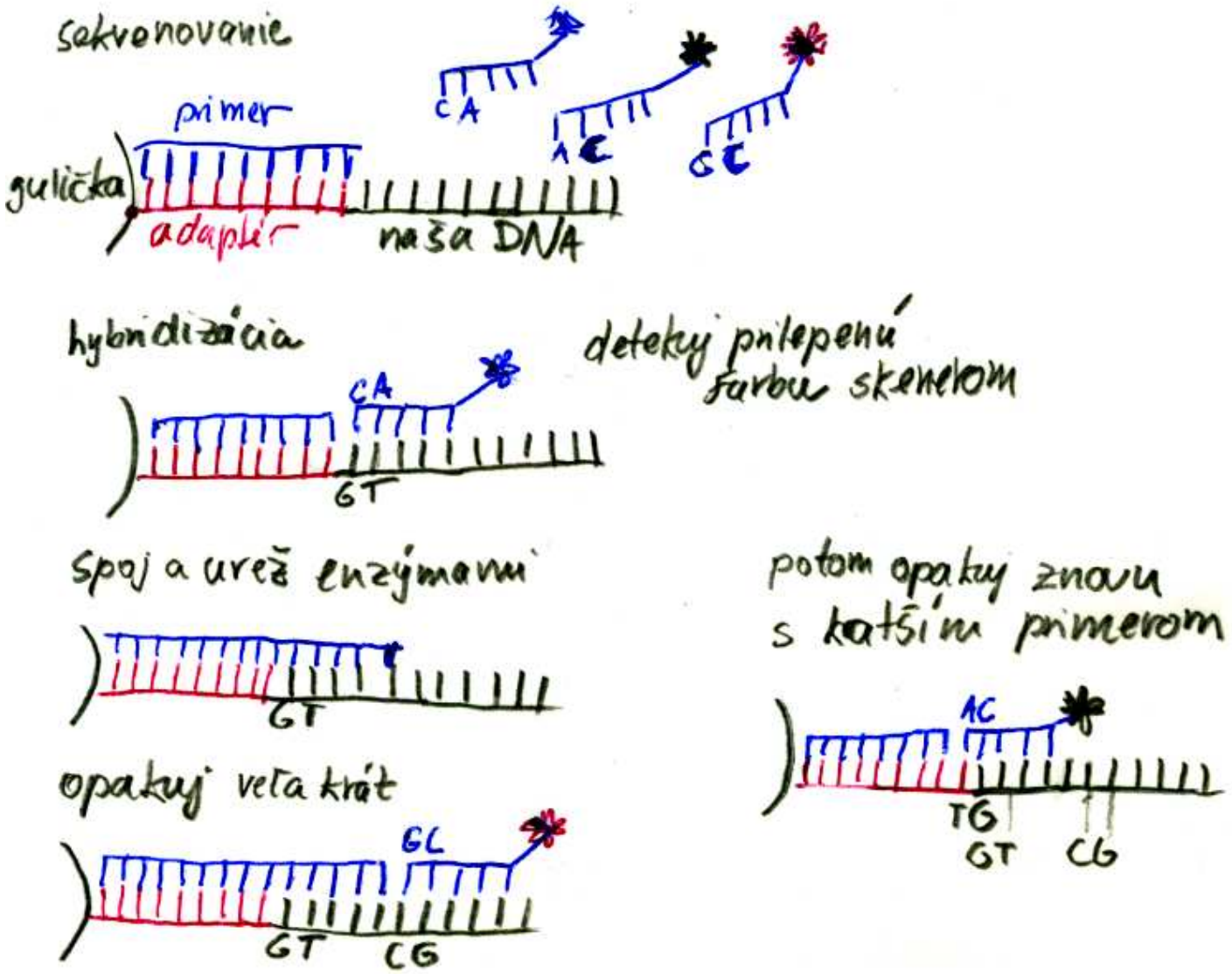
  DNA polymerase can synthetize a complementary strand

Sample preparation for sequencing:



SOLiD: sample preparation




sekvenovanie
primer
guličky
adaptér
naša DNA
CA
AC
GC
hybridizácia
CA
GT
spoj a urež enzýmami
GT
opakuj veľa krát
GC
GT
CG
deteky pripojení farbou skenerom
potom opakuj znova s kratším primerom
AC
TG
GT
CG

SOLiD: color coding

SOLiD uses only four colors, each XOR of two adjacent letters

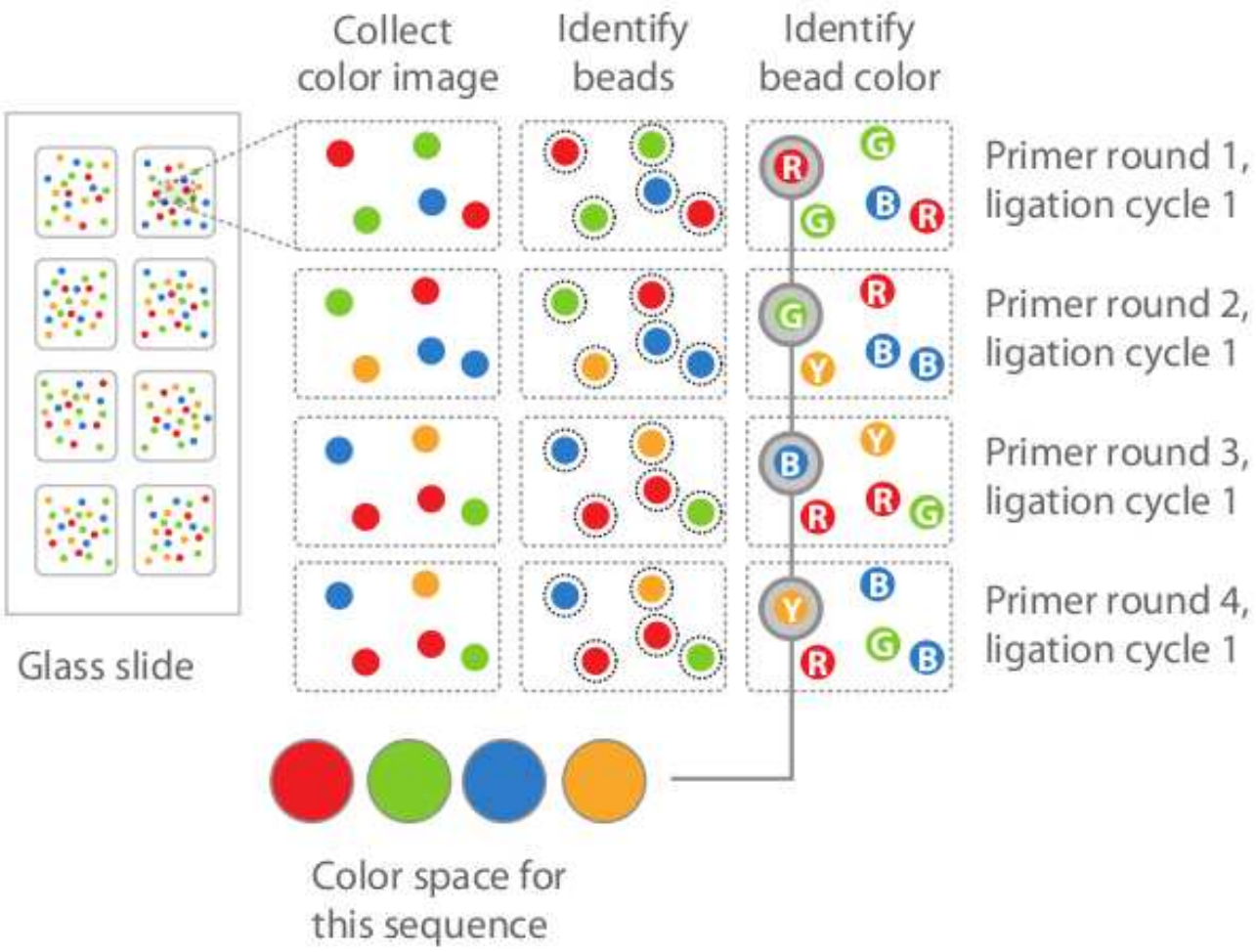
Assign numbers to letters

A	00
C	01
G	10
T	11

Assign numbers to colors, assign letter pairs

blue	00	AA	CC	GG	TT
green	01	AC	CA	GT	TG
yellow	10	AG	CT	GA	TC
red	11	AT	CG	GC	TA

SOLiD: overall strategy



SOLiD: color encoding

blue	AA	CC	GG	TT
green	AC	CA	GT	TG
yellow	AG	CT	GA	TC
red	AT	CG	GC	TA

One color sequence encodes 4 DNA sequences,

depending on the first letter:

ATGGA, CGTTC, GCAAG, TACCT

Computer science problems

One SOLiD sequencing machine generates cca 10 Gb of data per day

(after processing images to sequences)

. . . however we get only short pieces of the sequence

We need fast algorithms for:

- Assembling DNA from short pieces (finding overlaps between pieces)

  . . . more on this today

- Mapping pieces to a reference genome

  approximate string matching

Genome assembly

- Get many short sequences (reads) from different (unknown) positions in the DNA sequence
- Recover the original DNA sequence from which these come
- Not unique answer:

  E.g. reads AR, BR, CR, RB, RC, RD

  (for strings A,B,C,D,R$\in \Sigma^*$)

  Answer ARBRCRD or ARCRBRD or ARBRBRCRCRD etc.?

  Problem if DNA has long repeating strings

  In human genome (inexact) repeats cca 50% of length

Shortest common superstring (najkratšie spoločné nadslovo)

Simplified formalization of genome assembly

Input: $S_1, S_2, \ldots, S_k \in \Sigma^*$

Output: Shortest string S such that each S_i is a substring of S.

Example: $\{$ ema, ma, mamu, mama, emu $\}$, $S =$emamamuemu

Assume: No S_i is a substring of S_j for $i \neq j$

(such S_i could be eliminated from input).

Genome assembly vs. shortest common superstring

- In biology sequencing errors
- Two DNA strands (each S_i substring of S or S^R)
- Repeats (shortest answer not always correct)
- Sequencing gives us paired reads with known distance

Sequence assembly in practice

- Heuristics
- deBruijn graps
- Lot of data, efficient structures, suffix arrays

Shortest common superstring is NP-hard

1980 Galant, Maier, Storer

Decision version:

Input: $S_1, S_2, \ldots, S_k \in \Sigma^*$, integer t

Output: Is there a string S of length $\leq t$ such that each S_i is a substring of S?

Reduction from a modified Hamiltonian path problem:

Input: Directed graph with vertices $1 \ldots n$, each vertex except n has at least 2 outgoing edges

Output: Is there a path from 1 to n passing through each vertex exactly once?

This problem is known to be NP-hard.

Greedy algorithm for shortest common superstring

1. remove words which are substring of another word
2. find words S_i and S_j with longest overlap,
   i.e. longest α such that $S_i = \beta\alpha$, $S_j = \alpha\gamma$ for some $i \neq j$
3. connect S_i and S_j to $\beta\alpha\gamma$
4. repeat steps 2 and 3 until only one word left

Efficient implementation

Problems:

- Given S_i and its suffix tree and given S_j,
  find longest prefix of S_j which is a sufix of S_i
- Given a suffix tree for $S_1, \ldots, S_k$, and S_j
  find longest prefix of S_j which is a suffix of some S_i, $i \neq j$
- Find largest overlap
- How to implement the whole algorithm?

Greedy algorithm rewritten for analysis

1. List all pairs of input strings sorted by their overlap
2. Choose the pair (x, y) with the highest overlap d, connect them together
3. Eliminate pairs disabled by this choice, i.e. $(x, ?)$, $(?, y)$, and the pair creating a cycle
4. Repeat steps (2) and (3) until no pairs remain