

7. predavanje  
**Samostojeći indeksi.**  
**Poravnavanje očitavanja na referentni genom**

**Mirjana Domazet-Lošo**

FER-ZPR



# **Samostojeći indeksi**

# Što je to indeks i čemu služi?

---

- ▶ količina digitalno dostupnih podataka rasla je eksponencijalno u posljednjem desetljeću (Gantz & Reinsel, 2012)
  - ▶ procjena za 2020.godinu: 40 EB (exabyte) = 40 bilijuna GB
  - ▶ nakon toga: očekuje se udvostručavanje svake 2 godine
- ▶ kako omogućiti učinkovito pretraživanje teksta (ali i drugih digitalnih podataka)?
  - ▶ izgradnja indeksa
  - ▶ indeks = struktura podataka koja omogućuje učinkovit dohvat podataka

# Ideja: indeks u knjizi

## Subject Index

J. Pevsner, 2009. Bioinformatics and Functional Genomics, 2nd ed.

- AAT program, 666  
*ABCD1* gene, 734, 857  
Aberrations, chromosomal, 683, 863  
Ab initio approaches:  
  bacterial and archaeal genomes, 618  
  gene-finding software, 666, 755  
  genome analysis, 782  
  prediction of protein structure,  
    450–451, 455  
Absolute value, 346  
Accepted point mutations, 58–63, 217.  
  *See also* PAM matrices  
Accession numbers, significance of,  
  27–28, 33, 37, 51–52, 90, 106,  
  128, 196, 198, 248  
Accuracy, in microarray data analysis,  
  344–345  
Acetylcholine, 768  
Acetyltransferases, 39  
ACOR, 883  
Acrylamide gel, 382–383, 544  
Actin, 224, 817  
Actinobacteria, 600  
Actinomycetes, 603  
Acyl-CoA dehydrogenases, 718  
*Acyrthosiphon pisum*, 609  
*ADAM20*, 653  
Adaptation, 216  
*ADE2*, 468  
Adenine, 64, 110, 242, 286, 545  
Adenoma, 874  
Adenosine:  
  cyclic (cAMP), 398, 756  
  monophosphate (AMP), 127  
  triphosphate (ATP), 408, 545–546,  
    732, 858  
Adenovirus, 570  
Adenylate kinase, 223  
Adhesin, 579  
Adhesion molecules, 397, 579, 758  
ADP, 224  
Adrenoleukodystrophy, 718, 848, 850  
Advanced database searching:  
  characteristics of, 141–142  
  hidden Markov models (HMM),  
    152, 156–161, 174  
  gene discovery using BLAST,  
    169–173  
  pitfalls of, 174  
  position-specific iterated BLAST  
    (PSI-BLAST), 145–156, 174  
  PSSM, 146–153, 174  
  rapid search of genomic DNA,  
    161–169  
  SAM, 174  
  specialized BLAST sites, 142–145  
  web resources, 175  
*Aedes* spp., 762, 764  
*Aeropyrum pernix*, 533, 601, 608  
Affinity chromatography, 499–500  
Affymetrix, 315–317, 332, 333–335,  
  337, 343, 345–346  
Affymetrix GeneChip, 343  
Agarose gel, 293  
Agglomerative hierarchical clustering,  
  355–357  
Agilent, 317, 335  
Agricultural issues, 541  
*Agrobacterium tumefaciens*, 483  
AIDS, 541, 579, 583, 585, 876  
*Ajiellomyces capsulatus*, 716  
Akaike information criterion (AIC),  
  253  
Alanine, 50–61, 63, 65, 68, 92, 94,  
  148, 154, 382, 428, 544  
Albinism, 843, 847  
Albumins, 305, 378, 380  
Alcohol dehydrogenase, 702  
Algae, 530, 746. *See also* Brown algae;  
  Green algae  
Algorithms, *see specific algorithms*  
  advanced database searches, 141,  
    144  
  applications, 5, 55, 161–162  
  BLAST search, 115, 174  
  defined, 55  
AliBaba2, 670  
ALIGN, 93  
Align-m, 195  
Alignment, significance of:  
  advanced database searches,  
    141–142  
  gapped, 120–121, 123  
  pairwise, *see* Pairwise sequence  
    alignment  
  phylogenetic analysis, 60–61  
  protein, 47–49  
  score, 110  
  ungapped, 119–120, 123  
Alkaline phosphatase, 397  
Alkaptonuria, 842–843  
Alleles/allelic:  
  functional genomics, 475, 478, 492,  
    508  
  human disease, 863, 877  
  human genome, 826  
  single nucleotide polymorphisms  
    (SNPs), 684–686  
  variants, 27  
AllGenes, 754  
*Allomyces macrogynus*, 717  
Allopolyploids, 753  
Alpha crystallin A chain, 223  
Alpha globin, 33

*Bioinformatics and Functional Genomics, Second Edition.* By Jonathan Pevsner  
Copyright © 2009 John Wiley & Sons, Inc.

913

# Tipovi indeksnih struktura

---

- ▶ potpuni indeks (eng. *full-text index*) – indeks koji omogućuje dohvat cijelog teksta ili bilo kojeg njegovog dijela
  - ▶ sufiksno stablo
  - ▶ sufiksno polje
    - memorijsko zauzeće teksta:  $O(n \log |\Sigma|)$  bita
    - memorijsko zauzeće indeksa:  $\Theta(n \log n)$  bita
- ▶ potpuni indeks  $\neq$  samostojeći indeks (eng. *self-index*)
  - ▶ samostojeći indeks - proporcionalan veličini komprimiranog teksta

# Samostojeći indeks

---

- ▶ samostojeći indeks (eng. *self-index* ili *compressed self-index*)
  - ▶ indeksira tekst
  - ▶ zamjenjuje tekst, tj. indeks sam omogućuje pristup tekstu ili dijelovima teksta (podnizovima) nad kojima je indeks izgrađen
  - ▶ memorijsko zauzeće: proporcionalno veličini komprimiranog teksta (eng. *compressed text*), tj. sublinearno u odnosu na originalni tekst
  - ▶ problem: memorijsko zauzeće za izgradnju takvog indeksa može biti  $5n-9n$  (Ferragina et al. 2008)
  - ▶ prvi takav indeks: FM-indeks (eng. *FM-index*; Ferragina i Manzini 2000; 2005)

# Podjela samostojećih indeksa

---

Podjela u 3 osnovne skupine (Ferragina et al., 2008):

1. FM-skupina indeksa
2. Indeksi temeljeni na *CSA* (*Compressed Suffix Arrays*)
3. LZ-skupina indeksa (Lempel-Ziv sažimanje)

- ▶ teorijsko memorijsko zauzeće za niz  $S$  duljine  $n$ :  
 $O(nH_k(S)) + o(n)$  bita
- ▶ za brojanje pojavljivanja podniza  $P$  u  $S$ :  
 $O(|P| \log|\Sigma|)$
- ▶ pronalazak svakog pojavljivanja  $P$  u  $S$ :  
 $O(\log^{1+\varepsilon}|S|)$

# Entropija

---

- ▶ Primjer: bacanje kocke (6 mogućih događaja)
  - ▶ vrijedi:  $P(X=1) + P(X=2) + P(X=3) + P(X=4) + P(X=5) + P(X=6) = 1$
  - ▶ promatramo 2 slučaja:
    - (i) nepristrana kocka:  
 $P(X=1) = P(X=2) = P(X=3) = P(X=4) = P(X=5) = P(X=6) = 1/6$
    - (ii) pristrana kocka – npr. uvijek dobijemo 6:  $P(X=6) = 1, P(X \neq 6) = 0$
- ▶ statistička entropija (Shannon, 1948):  $H(S) = - \sum_i p_i \ln(p_i)$ 
  - ▶  $H$  je maksimalna, ako su svi mogući ishodi jednako vjerojatni ( $H = \ln 6$  za slučaj (i))
  - ▶  $H = 0$ , ako je moguć samo jedan ishod (slučaj (ii))



# Definicija nulte entropije za niz

---

- ▶ Neka je zadan niz  $S$  duljine  $n$ , i neka je  $n_i$  broj pojavljivanja znaka  $i \in \Sigma$  u  $S$ .

- ▶  $H_0(S) = -\sum_i \frac{n_i}{n} \log \frac{n_i}{n}, \quad n = \sum_i n_i \quad (\log \rightarrow \log_2)$

- ▶ Primjer:

- ▶  $S_1 = \text{ACCA}, n = 4, n_A = n_C = 2$

- $$H_0(S_1) = -\left(\frac{n_A}{n} \log \frac{n_A}{n} + \frac{n_C}{n} \log \frac{n_C}{n}\right) = -\left(\frac{2}{4} \log \frac{2}{4} + \frac{2}{4} \log \frac{2}{4}\right) = 1$$

- ▶  $S_2 = \text{ACCC}, n = 4, n_A = 1, n_C = 3$

- $$H_0(S_2) = -\left(\frac{n_A}{n} \log \frac{n_A}{n} + \frac{n_C}{n} \log \frac{n_C}{n}\right) = -\left(\frac{1}{4} \log \frac{1}{4} + \frac{3}{4} \log \frac{3}{4}\right) = 0.811$$

# Definicija $k$ -te entropije niza (1)

---

- ▶ Neka je zadan niz  $S$  duljine  $n$ , i neka je  $n_i$  broj pojavljivanja znaka  $i \in \Sigma$  u  $S$ .
- ▶ Neka je  $con$  podniz od  $S$  duljine  $k$  (kontekst, eng. *context*). Neka je  $S^{con}$  niz koji sačinjavaju znakovi koji se pojavljuju u  $S$  iza con gledano s lijeva na desno.
- ▶  $\Sigma^k$  je skup svih nizova duljine  $k$  čiji su znakovi iz  $\Sigma$ .

## Definicija $k$ -te entropije niza (2)

---

- ▶  $H_k(S)$  je  $k$ -ti red entropije od  $S$  (eng. *the  $k$ -th order entropy of  $S$* ),  $k \geq 0$
- ▶  $H_0(S) = -\sum_i \frac{n_i}{n} \log \frac{n_i}{n}, \quad n = \sum_i n_i$
- ▶  $H_k(S) = \frac{1}{n} \sum_{con \in \Sigma^k} |S^{con}| H_0(S^{con})$
- ▶ Vrijedi:  
$$0 \leq H_k(S) \leq H_{k-1}(S) \leq \dots \leq H_1(S) \leq H_0(S) \leq \log|\Sigma|$$
- ▶ Primjer:  
 $S = \text{ACCA\$}, con = C$   
 $S^{con} = CA$

# FM-skupina indeksa

---

- ▶ prvi samostojeći indeks uopće: FM-indeks (Ferragina i Manzini 2000; 2005)
- ▶ temelji se na Burrows-Wheeler transformaciji teksta (Burrows i Wheeler, 1994)
- ▶ memorijsko zauzeće: proporcionalno  $k$ -tom redu entropije teksta
- ▶ još neki indeksi iz ove skupine:
  - ▶ *Succinct Suffix Array* (SSA) (Mäkinen and Navarro, 2005)
  - ▶ *Alphabet-Friendly FM-index* (AF) (Ferragina et al. 2004)

# Samostojeći indeksi temeljeni na CSA

---

- ▶ temelje se na komprimiranim/sažetim sufiksnim poljima (eng. *Compressed Suffix Arrays*)
  - ▶ originalni CSA (Grossi i Vitter, 2000) nije bio samostojeći indeks (memorijsko zauzeće:  $O(n \log|\Sigma|)$ )
  - ▶ Sadakane je unaprijedio inicijalno rješenje kako bi dobio samostojeći indeks (Sadakane, 2003)
  - ▶ za sažimanje SA koristi se  $\psi$  funkcija, koja koristi uočene pravilnosti teksta

# Samostojeći indeksi temeljeni na LZ-sažimanju

---

- ▶ temeljeni na LZ78-sažimanju (Ziv i Lempel, 1978)
- ▶ primjer indeksa iz ove skupine:
  - ▶ Navarro (2004)

Table II. Ideal compressibility of our indexed texts. For every  $k$ -th order model, with  $0 \leq k \leq 4$ , we report the number of distinct contexts of length  $k$ , and the empirical entropy  $H_k$ , measured as number of bits per input symbol.

Text	$\log \sigma$	$H_0$	1st order		2nd order		3rd order		4th order	
			$H_1$	#	$H_2$	#	$H_3$	#	$H_4$	#
dna	4.000	1.974	1.930	16	1.920	152	1.916	683	1.910	2222
english	7.814	4.525	3.620	225	2.948	10829	2.422	102666	2.063	589230
pitches	7.055	5.633	4.734	133	4.139	10946	3.457	345078	2.334	3845792
proteins	4.644	4.201	4.178	25	4.156	607	4.066	11607	3.826	224132
sources	7.845	5.465	4.077	230	3.102	9525	2.337	253831	1.852	1719387
xml	6.585	5.257	3.480	96	2.170	7049	1.434	141736	1.045	907678

Table III. Real compressibility of our indexed texts, as achieved by the best-known compressors: gzip (option -9), bzip2 (option -9), and PPMDi (option -l 9).

Tekst (200 MB)	Veličina abecede	Text	$H_4$	gzip	bzip2	PPMDi
		dna	1.910	2.162	2.076	1.943
		english	2.063	3.011	2.246	1.957
		pitches	2.334	2.448	2.890	2.439
dna	16	proteins	3.826	3.721	3.584	3.276
		sources	1.852	1.790	1.493	1.016
english	225	xml	1.045	1.369	0.908	0.745
pitches	133					
proteins	25					
sources	230					
xml	96					

Ferragina et al. 2008. *Compressed Text Indexes: From Theory to Practice*

$$0 \leq H_k(T) \leq H_{k-1}(T) \leq \dots \leq H_1(T) \leq H_0(T) \leq \log \sigma$$

# Usporedba samostojećih indeksa (1)

---

- ▶ analiza koji su proveli Ferragina, González, Navarro i Venturini (2008):
  - ▶ usporedba implementacija sufiksnog polja i predstavnika svake od skupina samostojećih indeksa
  - ▶ promatrane su dvije tipične operacije:
    1. pronalaženje broja pojavljivanja uzorka  $P$  u tekstu  $S$
    2. dohvaćanje svih pojavljivanja uzorka  $P$  u tekstu  $S$



## Usporedba samostojećih indeksa (2)

---

- ▶ analiza koji su proveli Ferragina, González, Navarro i Venturini (2008):
  - ▶ brzina: samostojeći indeksi su sporiji 1-3 reda veličine
  - ▶ memorijsko zauzeće samostojećih indeksa: može biti i do red veličine manje od SA
    - ▶ usporedba: SA + tekst  $\rightarrow 5n$  okteta,  $n = |S|$
  - ▶ FM- i CSA-temeljeni indeksi zahtijevaju izgradnju SA polja kako bi se izgradio sam indeks, a LZ-indeks zahtijeva još neke pomoćne strukture
  - ▶ za izgradnju samostojećih indeksa potrebno  $5n - 9n$  okteta

Table V. Time and peak of main memory usage required to build the various indexes over the 200 MB file `english`. The indexes are built using the default value for the locate tradeoff (that is,  $s_A = 64$  for AF-index and SSA;  $s_A = 64$  and  $s_\Psi = 128$  for CSA; and  $\epsilon = \frac{1}{4}$  for the LZ-index).

Index	Build Time (sec)	Main Memory Usage (MB)
AF-index	772	1,751
CSA	233	1,801
LZ-index	198	1,037
SSA	217	1,251

Table VI. Experiments on the counting of pattern occurrences. Time is measured in microseconds per pattern symbol. The space usage is expressed as a fraction of the original text size. We put in boldface those results that lie within 10% of the best space/time tradeoffs.

Text	SSA		AF-index		CSA		LZ-index		plain SA	
	Time	Space	Time	Space	Time	Space	Time	Space	Time	Space
dna	<b>0.956</b>	<b>0.29</b>	1.914	<b>0.28</b>	5.220	0.46	43.896	0.93	0.542	5
english	<b>2.147</b>	0.60	2.694	<b>0.42</b>	4.758	<b>0.44</b>	68.774	1.27	0.512	5
pitches	<b>2.195</b>	0.74	2.921	<b>0.66</b>	3.423	<b>0.63</b>	55.314	1.95	0.363	5
proteins	<b>1.905</b>	<b>0.56</b>	3.082	<b>0.56</b>	6.477	0.67	47.030	1.81	0.479	5
sources	<b>2.635</b>	0.72	2.946	0.49	4.345	<b>0.38</b>	162.444	1.27	0.499	5
xml	2.764	0.69	<b>2.256</b>	0.34	4.321	<b>0.29</b>	306.711	0.71	0.605	5

Ferragina et al. 2008. *Compressed Text Indexes: From Theory to Practice*

# FM-indeks

---

- ▶ temelji se na podatkovnim strukturama SA i BWT (eng. *Burrows-Wheeler Transform* ili *block-sorting compression*)
  - ▶ BWT - koristi se kod sažimanja podataka, npr. bzip2 (Seward, 2007)
  - ▶ sublinearno memorijsko zauzeće:  
 $O(nH_k(S)) + o(n)$  bita za niz  $S$  duljine  $n$
  - ▶  $H_k(S)$  je  $k$ -ti red entropije od  $S$  (eng. *the  $k$ -th order entropy*)

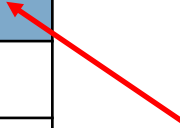
# BWT (1)

BWT(*Burrows-Wheeler Transform*):

- ▶ određuje se u  $O(n)$  vremenu iz  $SA$
- ▶ memorijsko zauzeće:  $n$  okteta

Konstrukcija niza  $BWT$  rotacijom niza  $S = ACCA\$$ .  $\$$  je abecedno najveći znak.

$i$	$SA[i]$	Abecedno poredani sufiksi od $S$	$BWT[i]$
1	1	ACCA\$	\$
2	4	A\$	C
3	3	CA\$	C
4	2	CCA\$	A
5	5	\$	A



$B[i] = \$$  za  $SA[i] = 1$   
 $B[i] = S[SA[i] - 1]$  inače

## BWT (2)

---

Izgradnja niza *BWT* rotacijom niza  $S = \text{ACCA\$}$ .  $\$$  je abecedno najveći znak.

$i$	Permutacije niza $S$	Abecedno poredane permutacije niza $S$	$BWT[i]$
1	ACCA\$	<b>A</b> CCA <b>\$</b>	\$
2	CCA\$A	<b>A</b> \$AC <b>C</b>	C
3	CA\$AC	<b>C</b> A\$A <b>C</b>	C
4	A\$ACC	<b>C</b> CA\$ <b>A</b>	A
5	\$ACCA	<b>\$</b> ACC <b>A</b>	A

# Reverzibilnost BWT-a

## LF-mapiranje (eng. *LF-mapping*; *Last-to-Front mapping*; *Last-to-First mapping*)

$i$ -ta pojava znaka  $X$  u zadnjem stupcu ( $L$ ) odgovara  $i$ -toj pojavi znaka  $X$  u prvom stupcu ( $F$ ), tj.  $BWT[i]$  se nalazi u stupcu  $F$  na mjestu  $LF[i]$ .

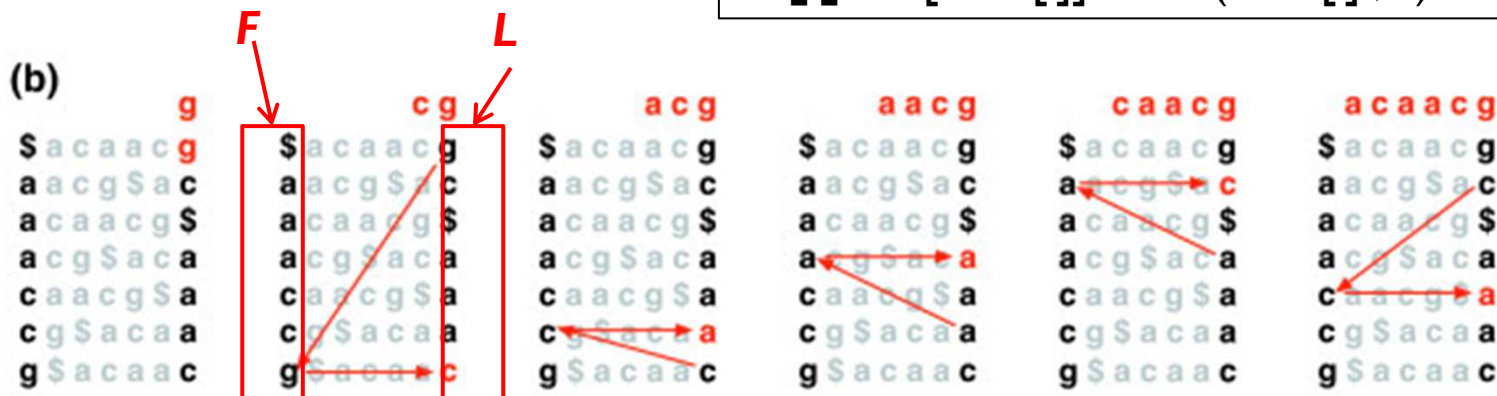
Primjer: 1. znak **g** u stupcu L se mapira na 1. znak **g** u stupcu F (vidjeti sliku).



**C(c)** je broj znakova u  $S[l, |S|-l]$  koji su abecedno prije  $c$  (uključujući ponavljanja znakova)

**Occ(c, i)** je broj pojavljivanja znaka  $c$  u  $BWT[l, i]$

$$LF[i] = C[BWT[i]] + Occ(BWT[i], i)$$



Langmead et al. 2009. *Ultrafast and memory-efficient alignment of short DNA sequences to the human genome.*

# Interval sufiksnog polja

- ▶ Neka je zadan niz  $S$ .
- ▶ Sva pojavljivanja podniza  $P$  u  $S$  određena su intervalom sufiksnog polja  $[L_p, R_p]$ :

$$L_p = \min \{k: P \text{ je prefiks sufiksa } S_{SA[k]}\}$$

$$R_p = \max \{k: P \text{ je prefiks sufiksa } S_{SA[k]}\}$$

- ▶ Poseban slučaj:  
Ako je  $P = \emptyset$ , onda je  $[L_p, R_p] = [1, |S|]$ .

Primjer:  $S = \text{ACCA\$}$ ,  $P = A \rightarrow [L_p, R_p] = [1, 2]$

$i$	$SA[i]$	$S[SA[i], n]$
1	1	<b>A</b> CCA\$
2	4	<b>A</b> \$
3	3	CA\$
4	2	CCA\$
5	5	\$

# Pretraživanje unatrag (1)

---

Pretraživanje unatrag (eng. *backward search*)

- ▶ Neka je  $B = \text{BWT}(S)$ , a  $c \in \Sigma$ .
- ▶ Tada su definirane funkcije  $C$  i  $\text{Occ}$  na sljedeći način:
  - ▶  $C(c)$  je broj znakova u  $S[1, |S|-1]$  koji su abecedno prije  $c$  (uključujući i ponavljanja znakova)
  - ▶  $\text{Occ}(c, i)$  je broj pojavljivanja znaka  $c$  u  $B[1, i]$ , gdje  $i = 1, \dots, |S|$



## Pretraživanje unatrag (2)

---

- ▶ Neka je  $P$  neki podniz koji tražimo u  $S$  i  $c \in \Sigma$ . Tada je  $P$  sufiks podniza  $cP$  kojeg tražimo u  $S$  nakon što smo pronašli  $P$ .

- ▶ Vrijedi sljedeće (Ferragina i Manzini, 2000):

$$L_{cP} = C(c) + \text{Occ}(c, L_P - 1) + 1$$

$$R_{cP} = C(c) + \text{Occ}(c, R_P)$$

$\text{Occ}(c, i) = \text{broj pojavljivanja } c \text{ u } B[1, i]; i = 1, \dots, |S|$

- ▶ BW\_Count algoritam (Ferragina i Manzini, 2000)
  - ▶ kreće s pretraživanjem počevši od zadnjeg znaka od  $P$
  - ▶ ako  $P$  postoji u  $S$ , algoritam vraća broj pojavljivanja od  $P$ , a 0 inače

# Algoritam *BW\_Count*

**Ulaz:**  $C$ ,  $Occ$ ,  $P$

**Izlaz:** broj pojavljivanja  $P$  u  $S$

$i = |P|$  /\* kreće od zadnjeg mjesta u  $P$  \*/

$c = P[|P|]$  /\* zadnji znak u  $P$  \*/

$L_P = C[c] + 1$

$R_P = C[csljed]$  /\*  $csljed$  je znak nakon  $c$  u  $\Sigma$  \*/

**dok**  $((L_P < R_P) \text{ i } (i \geq 2))$  **ponavljaj**

$c = P[i - 1]$ ; /\* uzmi sljedeći znak iz  $P$  \*/

$L_P = C[c] + Occ(c, L_P - 1) + 1$

$R_P = C[c] + Occ(c, R_P)$

$i = i - 1$

**ako**  $(R_P < L_P)$  **onda vrati** 0

**inače vrati**  $(R_P - L_P + 1)$

$C(c)$  je broj znakova u  $S[1, |S|-1]$   
koji su abecedno prije  $c$

$Occ(c, i) =$  broj pojavljivanja  $c$  u  $B[1, i]$

# Algoritam *BW\_Count* – primjer (1)

- ▶ Neka je zadan  $S = \text{ACCA\$}$  (\$ je abecedno najveći).  
Postoji li podniz  $P = \text{CA}$  u  $S$ ?
- ▶ Rješenje:  
 $P$  je  $S[3, 4]$  (sufiks  $s_3$  sadrži prefiks koji je jednak  $P$ -u).

- ▶ 1. korak: određujemo  $C$

$$C('A') = 0$$

$$C('C') = 2$$

$C(c)$  je broj znakova u  $S[1, |S|-1]$  koji su abecedno prije  $c$

- ▶ 2. korak: određujemo  $Occ$

$$Occ('A', 1) = 0; Occ('A', 2) = 0; \text{ itd.}$$

$$Occ('C', 1) = 0; Occ('C', 2) = 1; \text{ itd.}$$

$Occ(c, i) = \text{broj pojavljivanja } c \text{ u } B[1, i]$

	1	2	3	4	5
<b>B</b>	\$	C	C	A	A

# Algoritam *BW\_Count* – primjer (2)

- ▶ Neka je zadan  $S = \text{ACCA\$}$  ( $\rightarrow B = \$\text{CCAA}$ ). Postoji li podniz  $P = \text{CA}$  u  $S$ ?
- ▶ Traženje  $P$  u  $S$  počinjemo sa zadnjim znakom, tj. s 'A'.

$$C('A') = 0$$

$$L_p = C['A'] + 1 = 1$$

$$R_p = C(\text{csljed}) = C('C') = 2$$

$$(i) L_p = C[c] + 1; R_p = C[\text{csljed}]$$

$$(ii) L_p = C['C'] + \text{Occ}('C', L_p - 1) + 1$$

$$R_p = C['C'] + \text{Occ}('C', R_p)$$

- ▶ Nastavlja pretraživanje s 'C':

$$L_p = C['C'] + \text{Occ}('C', L_p - 1) + 1 =$$

$$2 + 0 + 1 = 3$$

$$R_p = C['C'] + \text{Occ}('C', R_p) = 2 + 1 = 3$$

$i$	$SA[i]$	Abec. poredane permutacije niza $S$	$B[i]$
1	1	<b>A</b> CCA\$	\$
2	4	A\$AC <b>C</b>	C
3	3	CA\$A <b>C</b>	C
4	2	CCA\$ <b>A</b>	A
5	5	\$ACCA <b>A</b>	A

- ▶ Funkcija vraća vrijednost:

$$R_p - L_p + 1 = 3 - 3 + 1 = 1, \text{ tj. } P \text{ se pojavljuje samo jedanput u } S.$$

$$\text{Occ}(c, i) = \text{broj pojavljivanja } c \text{ u } B[1, i]$$

# Algoritam *BW\_Count* – primjer (3)

$i$ -ta pojava znaka  $X$  u zadnjem stupcu odgovara  $i$ -toj pojavi znaka  $X$  u prvom stupcu, tj.  $BWT[i]$  se nalazi u prvom stupcu (F) na mjestu  $LF[i]$ ;  $LF[i] = C[BWT[i]] + Occ(BWT[i], i)$

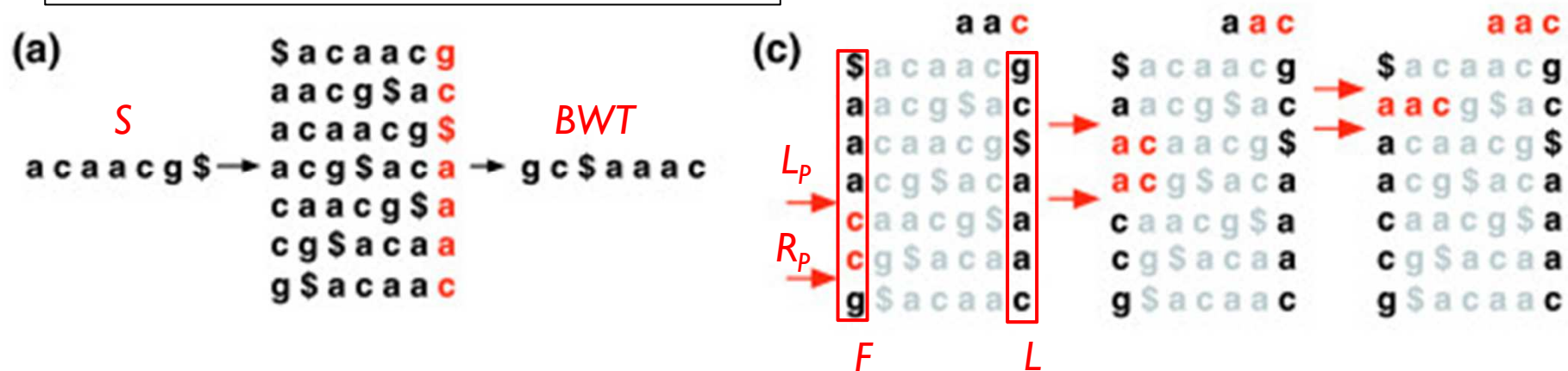
$C(c)$  je broj znakova u  $S[1, |S|-1]$  koji su abecedno prije  $c$  (uključujući i ponavljanja znakova)

$Occ(c, i)$  je broj pojavljivanja znaka  $c$  u  $BWT[1, i]$

$$(i) L_p = C[c] + 1; R_p = C[csljed]$$

$$(ii) L_p = C['C'] + Occ('C', L_p - 1) + 1$$

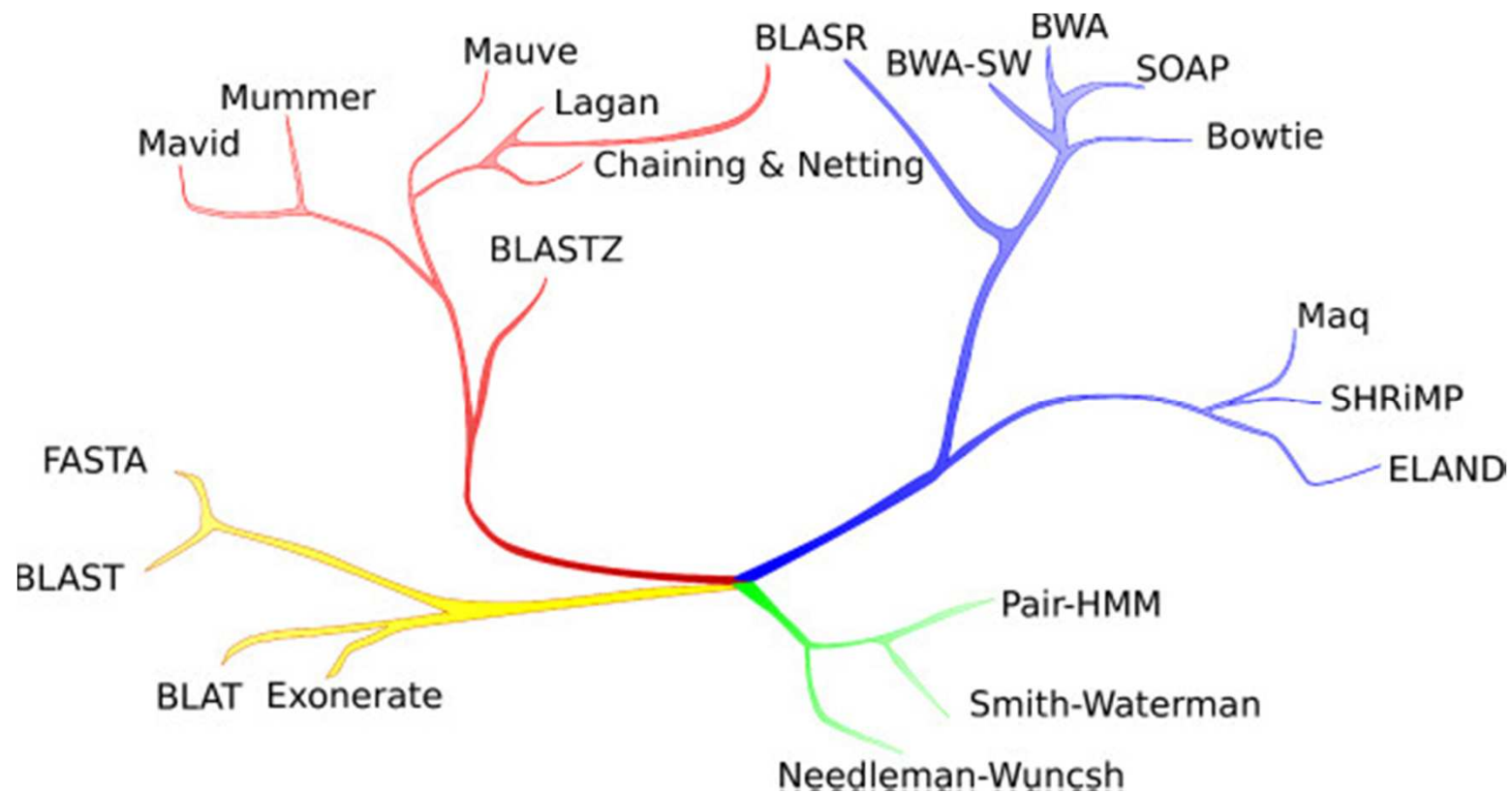
$$R_p = C['C'] + Occ('C', R_p)$$



Langmead et al. 2009. *Ultrafast and memory-efficient alignment of short DNA sequences to the human genome.*

# **Poravnavanje očitania na referentni genom**

## Pregled različnih metoda poravnavanja



Izvor: Chaisson and Tesler *BMC Bioinformatics* 2012, **13**:238

# Količina sekvenciranih baza parova

---

- ▶ Human Genome Sequencing Project, završen 14.4.2003.
  - ▶ počeo je 1990., planirano trajanje 15 godina
  - ▶ završeno 2 godine prije plana
  - ▶ troškovi: oko \$3 milijarde
- ▶ A danas?
  - ▶ Illumina GAII ili ABI SOLiD sekvencira  $10^9$  bp unutar nekoliko sati (Flicek & Birney, 2009)



**Table 1 Technical specifications of Next Generation Sequencing platforms utilised in t**

Platform	Illumina MiSeq	Ion Torrent PGM	PacBio RS
Instrument Cost*	\$128 K	\$80 K**	\$695 K
Sequence yield per run	1.5-2Gb	20-50 Mb on 314 chip, 100-200 Mb on 316 chip, 1Gb on 318 chip	100 Mb
Sequencing cost per Gb*	\$502	\$1000 (318 chip)	\$2000
Run Time	27 hours***	2 hours	2 hours
Reported Accuracy	Mostly > Q30	Mostly Q20	<Q10
Observed Raw Error Rate	0.80 %	1.71 %	12.86 %
Read length	up to 150 bases	~200 bases	Average 1500 bases**** (C1 chemistry)
Paired reads	Yes	Yes	No
Insert size	up to 700 bases	up to 250 bases	up to 10 kb
Typical DNA requirements	50-1000 ng	100-1000 ng	~1 µg

\* All cost calculations are based on list price quotations obtained from the manufacturer and assume expected sequencing volume.

\*\* System price including PGM, server, OneTouch and OneTouch ES.

\*\*\* Includes two hours of cluster generation.

\*\*\*\* Mean mapped read length includes adapter and reverse strand sequences. Subread lengths, i.e. the individual sequencing reads.

Quail et al. 2012. *A tale of three next generation sequencing platforms: comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq sequencers*

# Kako dobiti informaciju iz očitavanja?

---

- ▶ očitavanje (eng. *read*) = slijed nukleotida
  - ▶ tipična duljina očitavanja oko 500 bp (prije i manja 50 - 400 bp)
- ▶ Što s podacima/očitavanjima?  
*Sense from sequence reads: methods for alignment and assembly*  
(Flicek & Birney, 2009)
  - ▶ odrediti njihovu kvalitetu
  - ▶ složiti i analizirati podatke
    - slaganje i poravnanje/mapiranje (eng. *assembly and alignment/mapping*)

# Problem mapiranja očitavanja (poravnanja na referentni genom)

---

- ▶ Ulaz:

- ▶ skup očitavanja
- ▶ referentni genom

- ▶ Problem:

- ▶ pronaći mapiranje svakog očitavanja na genom, tj. pronaći
  - (i) sva poravnanja, ili
  - (ii) najbolje poravnanjesvakog ulaznog očitavanja s referentnim genomom uz dozvoljeno najviše  $k$  pogrešaka u poravnanju

# Poravnanje očitavanja - problemi

---

- ▶ mogući problemi:
  - ▶ pogreške u očitanjima
  - ▶ polimorfizam
  - ▶ razlike između sekvenciranog i referentnog genoma
    - ▶ dijelovi sekvenciranog genoma koji ne postoje u referentnom genomu
    - ▶ rekombinacija
    - ▶ preslagivanje (eng. *rearrangement*) dijelova genoma
  - ▶ ponavljanja

# Odabir načina poravnanja

---

Primjer: želimo poravnati 200 milijuna očitavanja duljine 200 nt na ljudski genom.

- ▶ dinamičko programiranje?
- ▶ korištenjem BLAST-a?
- ▶ poravnanje kratkih očitavanja (eng. *short-read alignment*):
  - ▶ poravnanje očitavanja na referentni genom
  - ▶ Problem:  
Kako postići visoku točnost poravnanja uz što manju pogrešku?

# Poravnavanje očitavanja - ideja

---

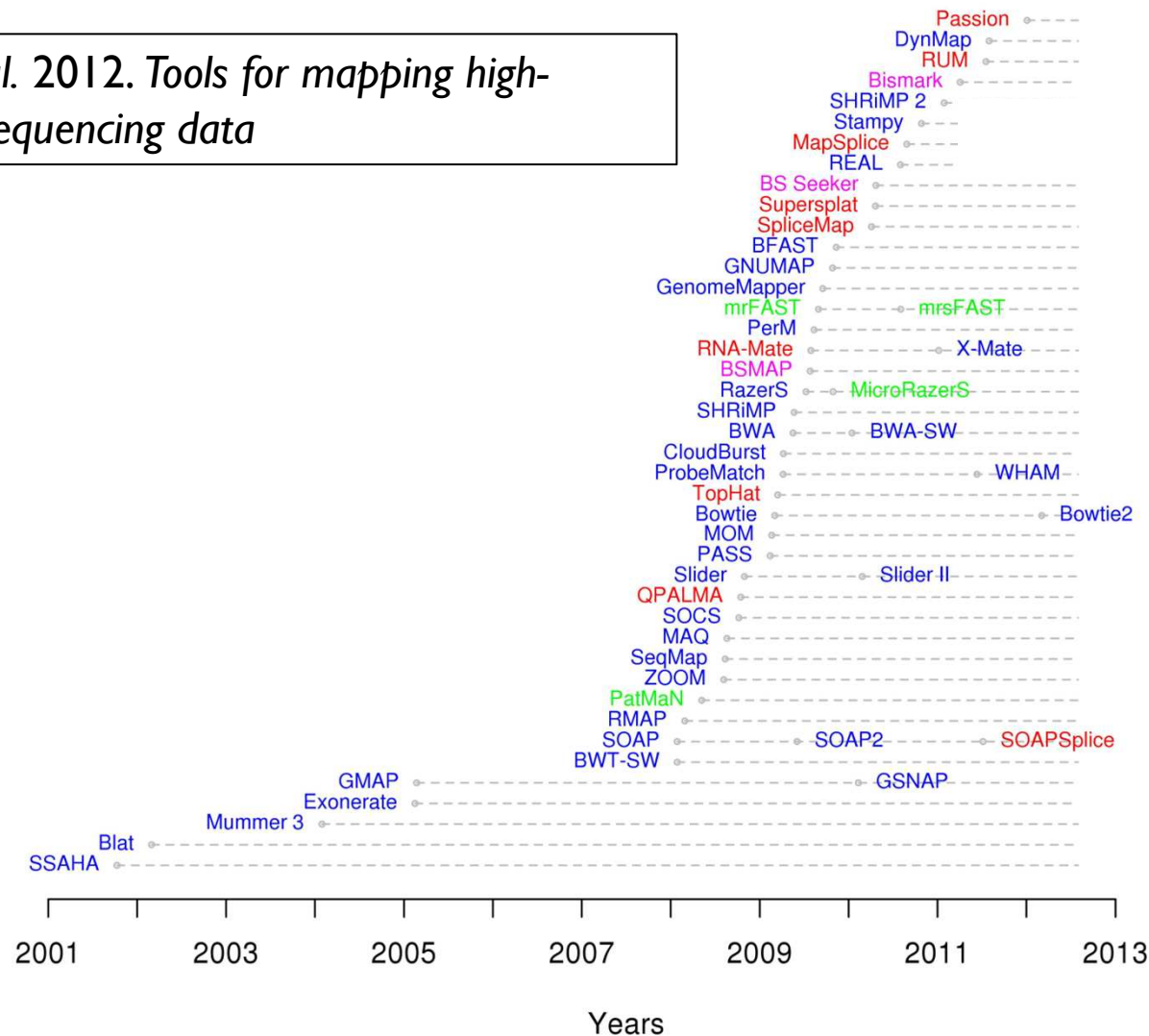
1. Brzo pronaći mjesta mogućih podudaranja (djelomična/potpuna)  
→ smanjiti prostor pretraživanja (eng. *search space*)
2. Odabrana podudaranja proširiti u lokalna poravnanja korištenjem sporijeg i preciznijeg postupka poravnanja

# Metode poravnavanja kratkih očitavanja

---

- ▶ metode temeljene na raspršenom adresiranju (eng. *hash-based*)
  - ▶ tablica raspršenog adresiranja se može izgraditi ili nad referentnim genomom ili nad skupom očitavanja
- ▶ metode temeljene na BWT (Burrows-Wheeler transformaciji)
  - ▶ izgradnja indeksa nad referentnim genomom koji omogućuje učinkovitu pretragu uz relativno male memorijske zahtjeve

Fonseca et al. 2012. Tools for mapping high-throughput sequencing data



Pregled alata za mapiranje (2001-2012). Alati za DNA mapiranje su prikazani plavom bojom, a alati za mapiranje RNA crvenom bojom (Fonseca et al. 2012).



# Metode temeljene na raspršenom adresiranju

---

- ▶ proširenje *seed-extend* algoritama na kratka očitavanja: pronaći kraća podudaranja i onda proširiti poravnanje
- ▶ tablica raspršenog adresiranja može biti izgrađena nad (Flicek & Birney, 2009):
  - ▶ očitanjima (npr. MAQ, SHRiMP)
    - ▶ referentni genom se uspoređuje s indeksiranim skupom očitavanja
    - ▶ dugotrajniji postupak pretraživanja referentnog genoma
  - ▶ referentnim genomom (npr. MOSAIK, SOAPv1, Novoalign, mrFAST, mrsFAST)
    - ▶ očitavanja se uspoređuju s indeksiranim referentnim genomom
    - ▶ veći memorijski zahtjevi

# Djelomično podudaranje

---

- ▶ ključ ne mora predstavljati potpuno podudaranje (eng. *spaced seed*)

1 → traži se podudaranje (eng. *match*); indeksirano mjesto

0 → ne traži se podudaranje; nije indeksirano mjesto

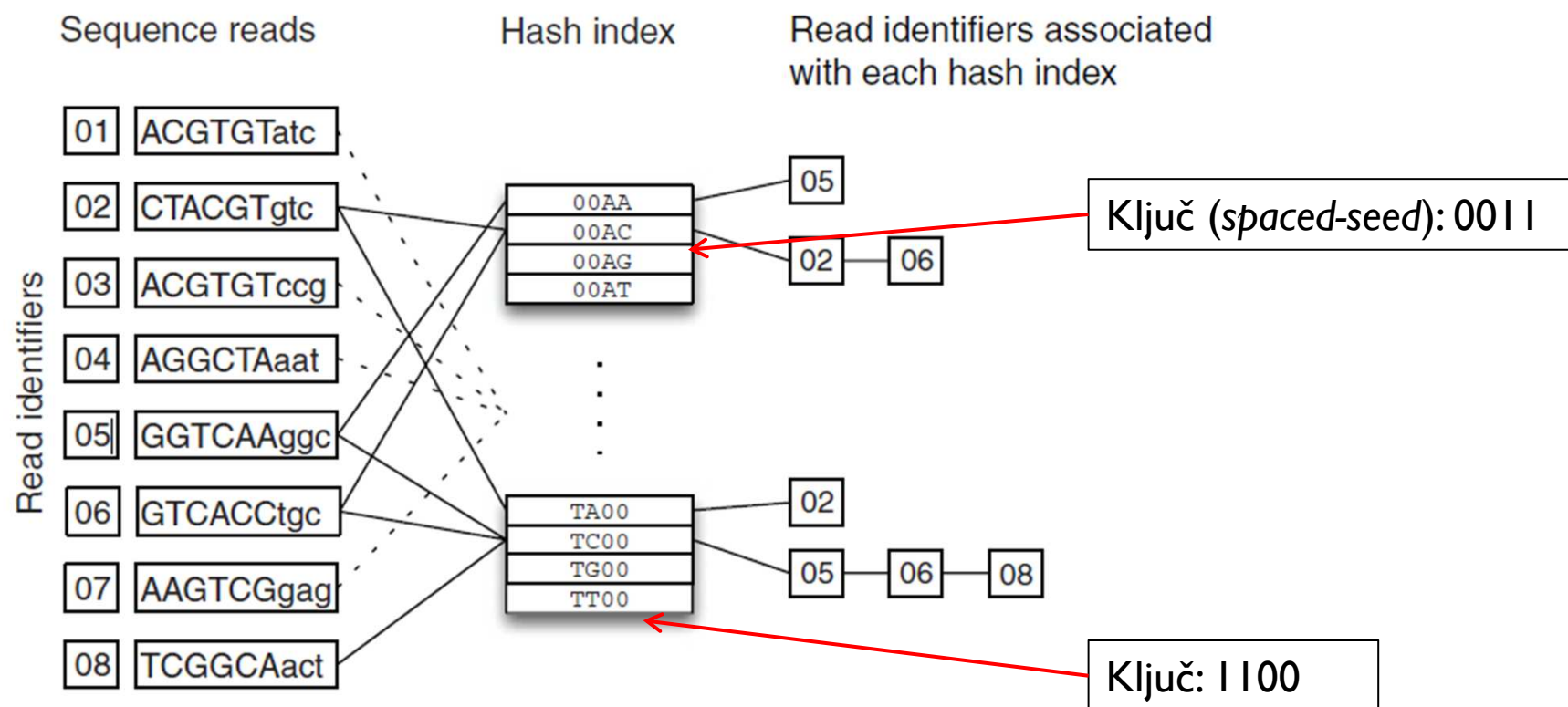
Primjer:

110011

→ prva dva i zadnja dva mjesta se trebaju podudarati, a treće i četvrto mjesto se ne trebaju podudarati

- ▶ broj jedinica: težina poravnanja (eng. *weight*)
- ▶ jedan zapis u tablici raspršenog adresiranja  
→ očitavanja koja sadrže iste znakove na pozicijama jedinica

# Metoda temeljena na raspršenom adresiranju - primjer



Flicek & Birney. 2009. *Sense from sequence reads: methods for alignment and assembly*

# Metode temeljene na *BWT* (1)

---

- ▶ pohraniti cijeli referentni genom
  - ▶ izgraditi indeks nad genomom
- ▶ pretraživanje indeksa tako da se uzima znak po znak ulaznog očitavanja počevši od zadnjeg znaka
- ▶ nakon što se pregledaju svi znakovi iz očitavanja, dojavljuju se sve pronađene pozicije u referentnom genomu
- ▶ ako podudaranje nije pronađeno, vraća se unatrag i pokreće se pretraga uz zamjenu znaka

# Metode temeljene na *BWT* (2)

---

- ▶ Bowtie – najbrži, nema umetanja/brisanja (eng. *indel*); umjerena preciznost
  - ▶ za pohranu ljudskog genoma: 1.3 GB memorije
- ▶ BWA – brz, moguća kratka umetanja/brisanja; dobra preciznost (nastavlja se na MAQ)

vs.

- ▶ Novoalign (raspršeno adresiranje) – sporo, veliki memorijski zahtjevi, moguća dugačka umetanja/brisanja; visoka preciznost

# Uporedba alata za mapiranje

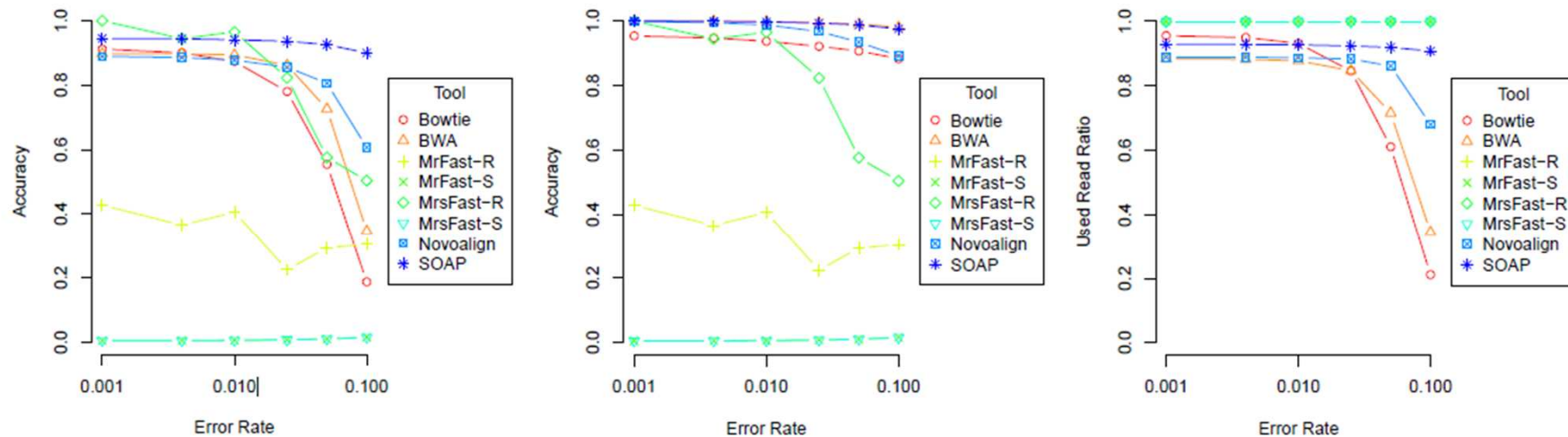


Fig. 1: Human genome: accuracy with varying error rate. (a) shows mapping quality threshold 0, (b) shows threshold 10 and (c) shows the proportion of reads that have mapping quality of at least 10. -R and -S suffixes denote relaxed and strict accuracy, respectively.

Ruffalo et al 2011. *Comparative analysis of algorithms for next-generation sequencing read alignment*

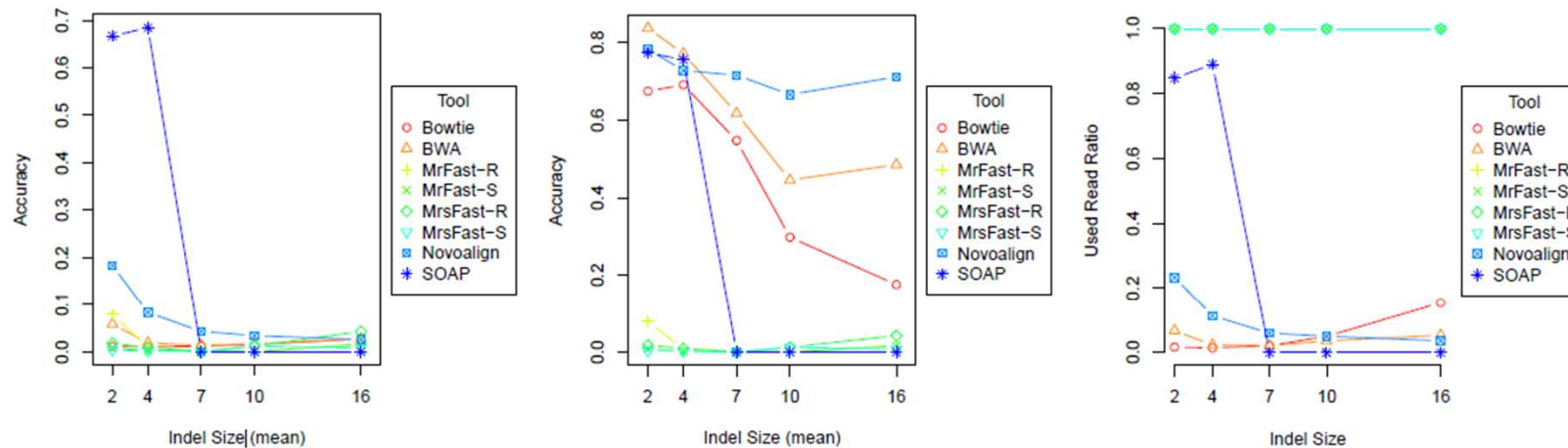


Fig. 3: Human genome: accuracy with varying indel sizes. (a) shows mapping quality threshold 0, (b) shows threshold 10 and (c) shows the proportion of reads that have mapping quality of at least 10. -R and -S suffixes denote relaxed and strict accuracy, respectively. At indel sizes 10 and 16, SOAP discards all reads, producing missing values in (c).

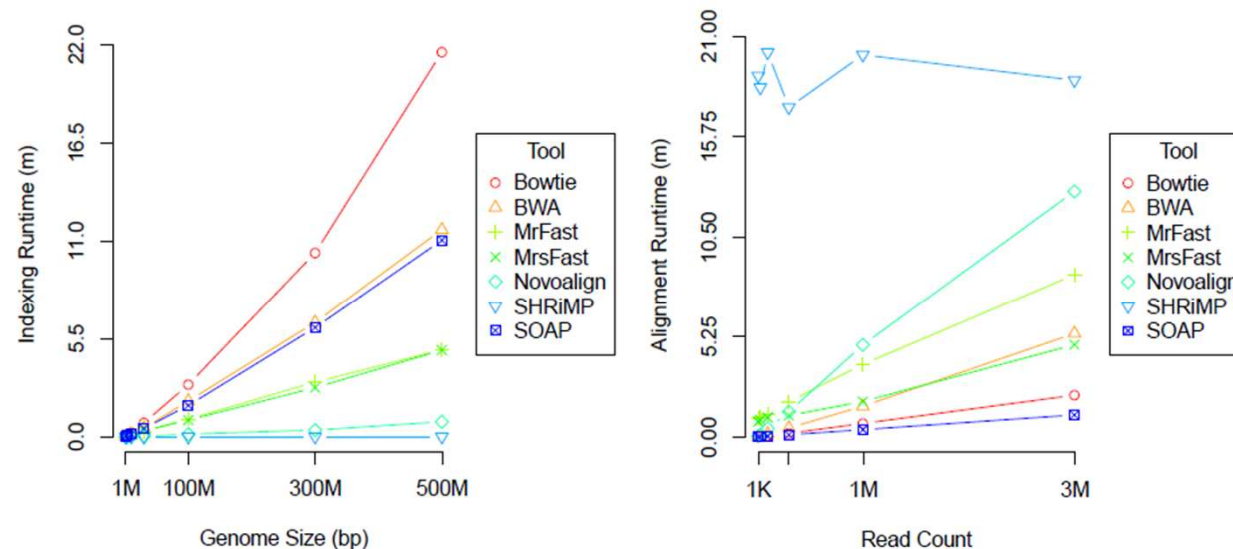


Fig. 5: Runtime measurements: (a) shows indexing time vs. genome size, (b) shows alignment time vs. read count on a 500Mbp genome.

Ruffalo et al 2011. Comparative analysis of algorithms for next-generation sequencing read alignment



## Usporedba alata za mapiranje (mapiranje na genom čovjeka)

Mapper	Time (min)	Pre.Time (min)	Map.Time (min)	Mem. (GB)	R. Aligned (DNA 100 bp)
BFAST	39 $\pm$ 0	20 $\pm$ 0	20 $\pm$ 0	21.4	561,348
Blat	93 $\pm$ 7	2 $\pm$ 0	90 $\pm$ 7	3.8	950,220
Bowtie	169 $\pm$ 39	166 $\pm$ 38	3 $\pm$ 1	5	798,566
Bowtie2	176 $\pm$ 40	168 $\pm$ 39	8 $\pm$ 1	5.1	991,880
BSMAP	25 $\pm$ 5	0 $\pm$ 0	25 $\pm$ 5	8.3	802,430
BWA	97 $\pm$ 6	83 $\pm$ 5	13 $\pm$ 1	7.6	928,093
GEM	380 $\pm$ 65	373 $\pm$ 64	6 $\pm$ 1	5.5	855,313
GMAP	2,887 $\pm$ 95	17 $\pm$ 2	2,870 $\pm$ 94	7.6	998,454
GSNAP	40 $\pm$ 6	22 $\pm$ 6	18 $\pm$ 1	7.6	926,371
MicroRazerS	453 $\pm$ 22	0 $\pm$ 0	453 $\pm$ 22	1.8	989,089
MOSAİK	22 $\pm$ 2	4 $\pm$ 1	18 $\pm$ 1	15.6	267,173
Novoalign	48 $\pm$ 4	12 $\pm$ 1	36 $\pm$ 3	7.8	940,428
Soap2	82 $\pm$ 7	78 $\pm$ 7	4 $\pm$ 0	5.3	798,565
SSAHA2	207 $\pm$ 27	13 $\pm$ 2	194 $\pm$ 25	9.5	1e+06
Stampy	189 $\pm$ 19	33 $\pm$ 6	156 $\pm$ 14	4	986,593

Fonseca et al. 2012. Tools for mapping high-throughput sequencing data.



# Popis literature

---

- ▶ Ferragina *et al.* 2008. Compressed text indexes: from theory to practice! *ACM Journal of Experimental Algorithmics* 13
- ▶ A. Bowe, FM-Indexes and Backwards Search, <http://alexbowe.com/fm-index/>
- ▶ Li, H., and Durbin, R. 2009. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* 25, 1754-1760
- ▶ Li, H., and Durbin, R. 2010. Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics* 26, 589–595
- ▶ Flicek P & Birney E. 2009. Sense from sequence reads: methods for alignment and assembly. *Nature Methods* (2009) 6, S6-S12
- ▶ Ruffalo, LaFramboise, Koyutürk. 2011. Comparative analysis of algorithms for next-generation sequencing read alignment. *Bioinformatics* 27, 20, 2790-2796.
- ▶ Fonseca, Rung, Brazma, Marioni. 2012. Tools for mapping high-throughput sequencing data. *Bioinformatics* 28, 24, 3169-3177.
- ▶ Langmead, Trapnell, Pop, Salzberg. 2009. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology* 10:R25

# **Zadatci za vježbu**

# 1. zadatak

---

Zadani su sljedovi  $S_1 = \text{ACCA}$  i  $S_2 = \text{GCC}$ . Sljedove je potrebno optimalno poravnati korištenjem Needleman-Wunschovog algoritma.

Podudaranje nukleotida se boduje s +1, nepodudaranje s -1, a praznina s -2 boda. Prikazati postupak.

# 1. zadatak - rješenje

Zadani su sljedovi  $S_1 = \text{ACCA}$  i  $S_2 = \text{GCC}$ . Sljedove je potrebno optimalno poravnati korištenjem Needleman-Wunschovog algoritma.

Podudaranje nukleotida se boduje s +1, nepodudaranje s -1, a praznina s -2 boda. Prikazati postupak.

		A	C	C	A
	0	-2	-4	-6	-8
G	-2	-1	-3	-5	-7
C	-4	-3	0	-2	-4
C	-6	-5	-2	1	-1

Poravnanje:

ACCA

GCC–

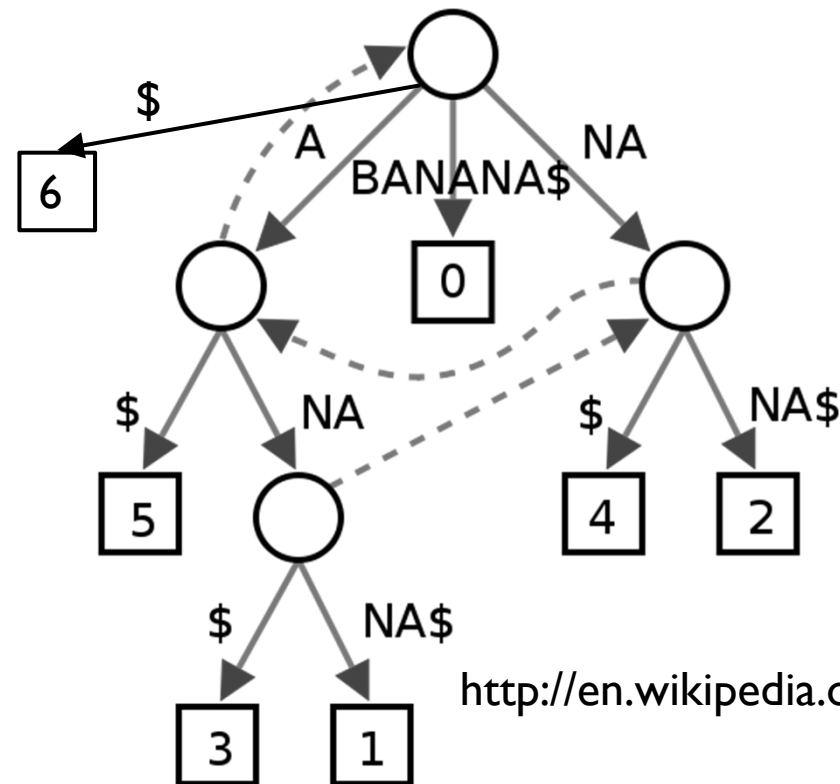
## 2. zadatak

---

Prikazati sufiksno stablo (*suffix tree*) za niz  $S = \text{BANANA}$  te označiti sufiksne veze na stablu. Pretpostaviti da je znak za kraj niza abecedno manji od svih znakova iz  $S$ .

## 2. zadatak - rješenje

Prikazati sufiksno stablo (*suffix tree*) za niz  $S = \text{BANANA}$  te označiti sufiksne veze na stablu. Pretpostaviti da je znak za kraj niza abecedno manji od svih znakova iz  $S$ .



[http://en.wikipedia.org/wiki/Suffix\\_tree](http://en.wikipedia.org/wiki/Suffix_tree)

### 3. zadatak

---

Odrediti sufiksno polje za niz  $S = \text{AGCGC}$ . Prikazati postupak izgradnje sufiksnog polja.

### 3. zadatak - rješenje

---

Odrediti sufiksno polje za niz  $S = \text{AGCGC}$ . Prikazati postupak izgradnje sufiksnog polja.

Abecedno poredani sufiksi od  $S$  (neka je  $\$$  abecedno najveći znak):

AGCGC\$

CGC\$

C\$

GCGC\$

GC\$

\$

Sufiksno polje:  $SA = [1, 3, 5, 2, 4, 6]$ .