

Bioinformatika

Ak. god. 2019./2020.

Usporedba programa za mapiranje očitavanja na genom

Dokumentacija, Rev. 4

Zdravko Čičin-Šain

Fran Penić

Toma Jurčić

Mateo Žaja

Datum predaje: 14.01.2020.

Sadržaj

1.	Uvod	3
2.	SNAP	4
2.1.	Prednosti i nedostaci	4
2.2.	Princip rada	4
2.3.	Primjer rada	6
2.4.	Mogućnost poboljšanja SNAP-a	11
3.	BBmap	12
3.1.	Prednosti i nedostaci	12
3.2.	Princip rada	12
3.3.	Primjer rada	12
4.	GraphMap	18
4.1.	Prednosti i nedostaci	18
4.2.	Princip rada	18
4.3.	Primjer rada	18
5.	Bowtie2	20
5.1.	Prednosti i nedostaci	20
5.2.	Princip rada	20
5.3.	Primjer rada	21
6.	Usporedni prikaz rezultata	27
7.	Zaključak	32
	Literatura	33

1. Uvod

Razvojem programiranja i računalne moći, počelo se razmišljati gdje sve postoji potreba za računalima. Iako mnogima zvuči nespojivo, računala i biologija su ubrzo postala usko povezana područja.

Nakon početnog vremena prilagodne računala u biologiji, ubrzo je počela utrka u sekvenciranju. Prije par godina se činilo nedostižno, ali danas se već naširoko govori o ideji sekvenciranja za samo 1000\$. Sekvenciranjem možemo otkriti pregršt informacija o nama kao vrsti, ali kao i o pojedincu, stoga je stavljen velik naglasak na dostupnost jeftinog i pouzdanog sekvenciranja širokoj publici.

Ljudski genom je relativno kompliciran, stoga se u mnogim istraživačkim projektima koriste neki jednostavniji organizmi (npr. *E. Coli*). Razvojem softvera otvorenog koda (engl. *open-source*) mnogi alati za mapiranje očitavanja su postali dostupni široj zajednici.

Glavna ideja ovog projekta je testiranje četiri različita alata te njihova međusobna usporedba. Svaki projekt zahtjeva specifičan alat koji će najbolje služiti ostvarenju cilja, stoga je svrha ovog projekta dati uvid u prednosti i mane takvih alata kako bi njihov izbor bio jednostavniji široj publici.

2. SNAP

SNAP (engl. *Scalable Nucleotide Alignment Program*) jedan je od najbržih dostupnih programa za mapiranje očitavanja genoma. Razvijen je zajedničkim naporom timova s UC Berkeley AMP Lab, Microsoft i UCSF, a napisan je u programskom jeziku C++. Dostupan je za sve operacijske sustave, a njegovo korištenje je potpuno besplatno.

2.1. Prednosti i nedostaci

Glavna prednost SNAP-a je njegova brzina. SNAP je u prosjeku 3 do 20 puta brži od alata poput BWA-mem, Bowtie2 i Novoalign, a u isto vrijeme zadržava kvalitetu očitavanja. SNAP je prilagođen za arhitekturu x86 procesora što mu omogućava da se izvodi na velikom broju današnjih računala. Podržan je širok spektar formata datoteka (BAM, FASTQ, gzipped FASTQ, FASTA, SAM), stoga je prikladan alat za velik broj istraživanja koja zahtijevaju barem jedan format od gore navedenih podataka.

Iako SNAP ima velik broj pozitivnih strana, on dolazi s par nedostataka. Glavni problem je izrazito velika memorijska potrošnja. SNAP prilikom svog rada koristi Hash tablice zvane *Index* tablice (objašnjeno kasnije u tekstu) kako bi postigao veliku brzinu rada. Prije svakog mapiranja genoma, potrebno je izgraditi takvu tablicu za ulaznu FASTA datoteku. Veličina same *Index* tablice ovisi o velikom broju parametra, ali najviše ovisi o samom broju ulaznih sekvenci baza. Za kompletni ljudski genom (oko 3 bilijuna baza) gradi se *Index* tablica veličine 40~57 GB. Računalo koje pokreće SNAP nad takvom *Index* tablicom trebalo bi imati barem 64 GB RAM-a [1]. Osim ogromne količine RAM-a, računalo bi također trebalo imati što veći broj jezgara kako bi se izračun maksimalno ubrzao. Za izvođenje SNAP-a na složenijim podacima, potrebno je imati pristup izrazito jakom računalu zbog memorijskih zahtjeva. Za jednostavnije programe situacija nije toliko strašna, stoga se manje zahtjevni zadaci mogu izvoditi na manje naprednim računalima.

Drugi nedostatak zasniva se na formatu zapisa podataka. SNAP zbog načina zapisa podataka nikad neće moći izgraditi tablicu za genome koje sadrže više od 2^{32} baza, a u praksi sve iznad 3 bilijuna baza izaziva kolizije u *Index* tablici, stoga SNAP nije primjenjiv za izrazito dugačke genome (npr. ameba).

Zadnji nedostatak nije toliko nedostatak SNAP-a koliko nekonzistentnosti FASTQ formata. FASTQ format može sadržavati sekvence koje sadrže niz znakova koji se prostire u više redova. SNAP ne može učitati FASTQ datoteku u takvome formatu, stoga je potrebno takve datoteke pretprocesirati prije korištenja što oduzima dodatnu količinu vremena.

2.2. Princip rada

Kao što je već ranije spomenuto u tekstu, SNAP se bazira na *Index* tablicama. Prilikom pokretanja programa, potrebno je predati ulaznu FASTA datoteku uz pomoć koje će se stvoriti *Index* tablica. U komandnoj ljuscici potrebno je pozvati naredbu

➤ `.\snap index ulaz.fa index-dir`

Prethodna naredba gradi *Index* tablicu u direktoriju naziva *index-dir* (stvori se ako prethodno ne postoji).

Name	Date modified	Type	Size
Genome	6.1.2020. 18:44	File	5.013 KB
GenomeIndex	6.1.2020. 18:44	File	1 KB
GenomeIndexHash	6.1.2020. 18:44	File	50.297 KB
OverflowTable	6.1.2020. 18:44	File	1.173 KB

Slika 1 – Sadržaj *index-dir* direktorija

Slika 1 prikazuje sadržaj *index-dir* direktorija. Glavna datoteka koju SNAP koristi za prilikom svog rada je *GenomeIndexHash*. Veličina ulazne datoteke je ~5 MB, a veličina te tablice je 10 puta veća (~50 MB). Za male ulazne podatke nema problema prilikom pokretanja programa, problem dolazi ako želimo raditi kompleksnije izračune jer veličina *Index* tablice drastično raste.

Postavlja se pitanje zašto su nam uopće potrebne *Index* tablice i na koji način one služe programu prilikom izračuna.

```
0123456789012345678901234
AAGCTTCTCACCCTGTTCTGCATA
```

Glavna prednost hash tablice je upravo brzina kojom možemo pristupiti svakom pojedinom podatku, a upravo to je glavna ideja za njihovo korištenje. Početno se izgradi velik set podataka, ali taj set podataka koristimo za mapiranje očitavanja i značajno skraćujemo vrijeme izvođenja programa. Npr. ako imamo ulaznu sekvencu nekog kromosoma X

i želimo izgraditi *Index* s parametrom *seed-size* = 20, onda će *Index* tablica imati podatke

```
0123456789012345678901234
AAGCTTCTCACCCTGTTCTCT    - kromosom X, lokacija 0
AGCTTCTCACCCTGTTCTCTG    - kromosom X, lokacija 1
GCTTCTCACCCTGTTCTCTGCA   - kromosom X, lokacija 2
CTTCTCACCCTGTTCTCTGCA    - kromosom X, lokacija 3
TTCTCACCCTGTTCTCTGCAT    - kromosom X, lokacija 4
```

Ako iz FASTQ datoteke pročitamo slijed CTTCTCACCCTGTTCTCTGCA, SNAP će odmah prepoznati da se radi o kromosomu X, lokacija 3.

Ako iz FASTQ pročitamo slijed CTTCTA**T**CCCTGTTCTCTGCA, SNAP neće pronaći odgovarajući zapis u *Index* tablici, odnosno javit će da nije pronašao odgovarajuću sekvencu. SNAP očekuje da slijed koji pročita savršeno odgovara nekome od zapisa iz tablice. Ako povećamo parametar *seed-size*, onda će postojati manja vjerojatnost da će ulazni slijed biti identičan nekom od zapisa iz *Index* tablice. Manji *seed-size* uzrokuje da SNAP pronađe više zapisa iz *Index* tablice kojima ulazni slijed odgovara. Takva situacija za posljedicu ima povećanje vremena izvođenja jer SNAP mora prvo odlučiti koji će zapis iz tablice odabrati, stoga je taj parametar najbolje ostaviti na predefiniranoj vrijednosti 20 osim ako korisnik zbilja zna kako treba podesiti taj parametar da dobije najbolje rezultate. SNAP podržava *seed-size* do veličine 32.

Nakon što smo izgradili *Index* tablicu, napokon možemo krenuti s mapiranjem očitavanja.

SNAP nudi dvije mogućnosti pokretanja programa:

- Pokretanje s jednom ulaznom datotekom (engl. *single unpaired read*)
- Pokretanje s dvije ulazne datoteke (engl. *paired end read*)

Ako program pokrećemo s jednom ulaznom datotekom u FASTQ formatu, potrebno je pozvati naredbu iz komandne ljuške

```
➤ .\ snap single index-dir ulaz.fq -o output.sam
```

Izlazna datoteka može biti u *.sam* ili *.bam* formatu. SNAP će generirati izlaznu datoteku i u komandnoj ljuški ispisati izvještaj.

Slično je pokretanje s dvije ulazne datoteke. Potrebno je pozvati naredbu iz komandne ljuške

```
➤ .\ snap paired index-dir ulaz1.fq ulaz2.fq -o output.sam
```

Postoji velik broj parametara koji se može koristiti za poboljšanje performansi, ali to je područje preopsežno, stoga se čitatelja upućuje na čitanje službene dokumentacije iz izvora [1] ako ga to zanima.

2.3. Primjer rada

Za demonstraciju rada programa, koristit ćemo šest ulaznih slučajeva:

- Kratak dio sekvence genoma E. Coli (~2500 baza, FASTQ format)
- Kompletne sekvence genoma E. Coli (~5 196 800 baza, FASTQ format)
- Slučajno generirana sekvenca genoma (~5 196 800 baza, FASTQ format)
- Kompletne sekvence genoma E. Coli i slučajno generirana sekvenca genoma (~5 196 800 baza, FASTQ format)
- Kompletne sekvence genome E. Coli generirane putem alata *wgsim* [6] (FASTQ format)
- Pet različitih datoteka generiranih pomoću *wgsim* alata (FASTQ format)

Određene FASTQ datoteke nisu u formatu koju SNAP prihvaća, stoga je potrebno pretprocesirati ulaznu datoteku.

Slučajno generirane sekvence napravljene su pomoću Python skripte.

Sekvence iz točaka 3. i 4. napravljene su parsiranjem ulazne FASTA datoteke te izvlačenjem sekvence iz njih. Znakovi kvalitete stavljeni su kao slučajni znakovi iz dozvoljenog opsega.

Za točku 5. generirana je FASTQ datoteka od ~187 MB pomoću alata *wgsim*, kao ulazni parametar smo stavili FASTA datoteku genome E. Coli i sve defaultne postavke.

Zadnji slučaj sastoji se od 5 generiranih FASTQ datoteka, gdje pogreška iznosi 1%, a duljina očitavanja 250.

Za parametar potrošnje memorije koristio sam ugrađeni alat *Resource Monitor* od operacijskog sustava Windows 10. Alat nema mogućnost praćenja zauzeća memorije u *ms* niti prikazuje povijest utroška memorije, stoga rezultate potrošnje memorije treba uzeti s dozom opreza.

```
import random

if __name__ == "__main__":
    input = open("ulaz.txt", "r")
    output = open("razlicitFO.fq", "w")
    znakovi = ["", ".", ")", " ", "(", "%", "+", "#", "-", "&", "!", "-", ":", "/"]
    duljina = len(znakovi)
    i = 0
    broj = 0
    #Prva linija je visak
    input.readline()
    while True:
        linija = input.readline()
        if not linija:
            break
        output.write("@SEQ_" + str(broj) + "\n")
        broj = broj + 1
        linija = linija[:-1]
        output.write(linija[:-1] + " \n")
        output.write("+" + "\n")
        j = 0
        while j < len(linija) - 1:
            random_znak = random.randrange(0, duljina, 1)
            output.write(str(znakovi[random_znak]))
            j = j + 1
        output.write("\n")
    input.close()
```

Slika 2 – Python skripta za generiranje sekvence genoma

Skripta uzima ulaznu datoteku koja sadrži sekvencu genoma, svakoj sekvenci pridaje vrijednost, odnosno kvalitetu te stavlja pripadajuće oznake. Takvim postupkom generiramo izlaznu datoteku u FASTO formatu.

Index tablica se koristi prilikom svakog mapiranja, stoga ju je potrebno napraviti na početku.

```
PS D:\Documents\FER\3.godina\9.semestar\Bioinformatika\Projekt> .\snap_index ulaz.fna index-dir
Welcome to SNAP version 1.0beta.18.

Hash table slack 0.300000
Loading FASTA file 'ulaz.fna' into memory...0s
Saving genome...0s
Computing bias table.
Computed bias table in 0s
Allocating memory for hash tables...0s
Building hash tables.
55589(1%) seeds occur more than once, total of 244656(4%) genome locations are not unique, 498(0%) bad seeds, 0 both complements used 500 no string
Hash table build took 0s
Building overflow table
Overflow table build and hash table save took 0s
Saving overflow table...0s
Index build and save took 0s (5133068 bases/s)
```

Slika 3 – Rezultat izgradnje Index tablice

Index tablica zbog relativno kratkog genoma E. Coli izgradi se za manje od jedne sekunde, a zauzima ~56 MB prostora.

1) Kratak dio sekvence genoma E. Coli

Nakon generiranja *Index* tablice pokrećemo SNAP i izvršavamo naredbu

```
➤ .\snap single index-dir ulazSlicni.fq -o output.sam
```

```

Welcome to SNAP version 1.0beta.18.
Loading index from directory... 0s. 5133068 bases, seed size 20
Aligning.
Total Reads      Aligned, MAPQ >= 10    Aligned, MAPQ < 10    Unaligned              Too Short/Too Many Ns    %Pairs    Reads/s    Time in Aligner (s)
8                8 (100.00%)                0 (0.00%)              0 (0.00%)              0 (0.00%)                0.00%     533        0

```

Slika 4 – Rezultat pokretanja programa (prvi slučaj)

Slika 4 prikazuje rezultat pokretanja SNAP-a na kratkoj sekvenci genome E. Coli. Rezultati izvođenja prikazani su u tablici 1.

Tablica 1 – Tablični rezultati pokretanja programa (prvi slučaj)

Ukupan broj čitanja	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
8	100.00 %	0.00 %	0.00 %	56	< 1

MAPQ (engl. *MAPping Quality*) nam govori o kvaliteti mapiranja. Zapisi koji imaju MAPQ ≥ 10 se mogu smatrati pouzdano mapiranim, a oni koji imaju MAPQ < 10 nisu u potpunosti pouzdani. Vidimo da gornji primjer ima sva mapiranja svrstana u MAPQ ≥ 10 kategoriju, stoga možemo sa sigurnošću zaključiti da je ulazna sekvenca savršeno mapirana s referentnim genomom.

Vrijeme izvođenja kraće je od jedne sekunde, a potrošnja memorije je ~56 MB jer se zbog veće brzine izvođenja *Index* tablica direktno učitava u RAM (ne radi se čitanje s diska). Postoji opcija da se *Index* tablica ne učitava direktno u RAM, ali onda se povećava vrijeme izvođenja programa.

Kao izlaz generirana je *output.sam* datoteka koja može koristiti za detaljnije analize.

2) Kompletna sekvenca genoma E. Coli

Izvršavamo naredbu

```
➤ .\snap single index-dir slicniFQ.fq -o output.sam
```

```
Welcome to SNAP version 1.0beta.18.
Loading index from directory... 0s. 5133068 bases, seed size 20
Aligning.
Total Reads    Aligned, MAPQ >= 10    Aligned, MAPQ < 10    Unaligned    Too Short/Too Many Ns    %Pairs    Reads/s    Time in Aligner (s)
64,151         60,928 (94.98%)         3,223 (5.02%)         0 (0.00%)    0 (0.00%)                0.00%    152,016    0
```

Slika 5 – Rezultat pokretanja programa (drugi slučaj)

Slika 5 prikazuje rezultat pokretanja SNAP-a na kompletnoj sekvenci genome E. Coli. Rezultati izvođenja prikazani su u tablici 2.

Tablica 2 – Tablični rezultati pokretanja programa (drugi slučaj)

Ukupan broj čitanja	Poravnato (MAPQ ≥ 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	94.98 %	5.02 %	0.00 %	68	< 1

Slika 5 nam govori da je 94.98% zapisa poravnato s MAPQ ≥ 10 , ali 5.02% zapisa nije pouzdano poravnato (MAPQ < 10). Vidimo da ulazna datoteka gotovo u potpunosti odgovara referentnom genomu, ali zbog određenih razlika ipak nije mapirana u potpunosti.

Vrijeme izvođenja je ostalo gotovo identično u usporedbi s prvim slučajem, ali su se memorijski zahtjevi malo povećali zbog veličine ulazne datoteke.

3) Slučajno generirana sekvenca genoma

Python skriptom prikazanom na slici 2 generiramo FASTQ datoteku koju pokrećemo pomoću SNAP-a naredbom

```
➤ .\snap single index-dir razlicitFQ.fq -o output.sam
```

```
Welcome to SNAP version 1.0beta.18.
Loading index from directory... 0s. 5133068 bases, seed size 20
Aligning.
Total Reads    Aligned, MAPQ >= 10    Aligned, MAPQ < 10    Unaligned    Too Short/Too Many Ns    %Pairs    Reads/s    Time in Aligner (s)
64,151         0 (0.00%)              0 (0.00%)             64,151 (100.00%)    0 (0.00%)                0.00%    341,228    0
```


Slika 6 – Rezultat pokretanja programa (treći slučaj)

Slika 6 prikazuje rezultat pokretanja SNAP-a na slučajno generiranoj sekvenci genoma. Rezultati izvođenja prikazani su u tablici 3.

Tablica 3 – Tablični rezultati pokretanja programa (treći slučaj)

Ukupan broj čitanja	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	0.00 %	0.00 %	100.00 %	68	< 1

Kao što je i bilo očekivano, slučajno generirana sekvenca nema poklapanja s originalnom sekvencom, stoga vidimo da SNAP nije pronašao poravnanja. Ostali rezultati su identični kao iz prethodnog pokretanja programa.

4) Kompletna sekvenca genoma E. Coli i slučajno generirana sekvenca genoma

Izvršavamo naredbu

```
➤ .\snap paired index-dir razlicitFQ.fq slicniFq.fq -o output.sam
```

```
welcome to SNAP version 1.0beta.18.
Loading index from directory... 0s. 5133068 bases, seed size 20
Aligning.
Total Reads    Aligned, MAPQ >= 10    Aligned, MAPQ < 10    Unaligned    Too Short/Too Many Ns    %Pairs    Reads/s    Time in Aligner (s)
128,302        60,928 (47.49%)          3,223 (2.51%)         64,151 (50.00%)    0 (0.00%)                0.00%     513,208    0
```

Slika 7 – Rezultat pokretanja programa (četvrti slučaj)

Slika 7 prikazuje rezultat pokretanja SNAP-a na dvije sekvence genoma. Rezultati izvođenja prikazani su u tablici 4.

Tablica 4 – Tablični rezultati pokretanja programa (četvrti slučaj)

Ukupan broj čitanja	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
128 302	47.49 %	2.51 %	50.00 %	80	< 1

Vidimo da se i u ovome slučaju vrijeme izvođenja nije promijenilo. Korištenje memorije blago je poraslo, ali ništa drastično. Zapis nije poravnat u 50% slučajeva, što je bilo i za očekivati jer u primjeru 3 poravnanje je bilo 0%, stoga će sad poravnanje iznositi duplo manje jer imamo kao ulaz još i datoteku iz primjera 2.

5) Kompletna sekvenca genome E. Coli generirana putem alata *wgsim*

Izvršavamo naredbu

```
➤ .\snap single index-dir esch_read_1.fq -o output.sam
```

```
PS D:\Documents\FER\5.godina\9.semestar\Bioinformatika\Projekt> .\snap.exe single index-dir esch_read_1.fq -o output.sam
Welcome to SNAP version 1.0beta.18.
Loading index from directory... 0s. 5133068 bases, seed size 20
Aligning.
Total Reads      Aligned, MAPQ >= 10  Aligned, MAPQ < 10  Unaligned      Too Short/Too Many Ns  %Pairs  Reads/s  Time in Aligner (s)
1,000,000         942,579 (94.26%)        47,366 (4.74%)      10,055 (1.01%)    0 (0.00%)              0.00%   546,746   2
```

Slika 8 – Rezultat pokretanja programa (peti slučaj)

Slika 8 prikazuje rezultat pokretanja SNAP-a na ulaznoj sekvenci genome E. Coli generirane pomoću alata *wgsim*. Rezultati izvođenja prikazani su u tablici 5.

Tablica 5 – Tablični rezultati pokretanja programa (peti slučaj)

Ukupan broj čitanja	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
1 000 000	94.26 %	4.74 %	1.01 %	243	3

Rezultati poravnanja su gotovo identični kao u primjeru 2. Glavna razlika je što smo u ovom slučaju imali ulaznu datoteku od ~187 MB, stoga je vrijeme izvođenja nešto veće nego u prethodnim primjerima. Utrošak memorije također se povećao jer je sama ulazna datoteka veća nego u prethodnim slučajevima.

6) Pet različitih datoteka generiranih pomoću *wgsim* alata (FASTQ format)

Pomoću *wgsim* alata smo generirali pet različitih datoteka od 50, 100, 200, 300 i 400 MB. Sve datoteke imaju duljinu očitavanja od 250 baza i grešku od 1%. Tablica 6 prikazuje rezultate pokretanja SNAP-a na ulaznim podacima.

Tablica 6 – Tablični rezultati pokretanja programa (šesti slučaj)

Naziv datoteke	Ukupan broj čitanja	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
read_50MB.fq	100 000	95.80 %	3.83 %	0.38 %	78	< 1
read_100MB.fq	200 000	95.69 %	3.87 %	0.44 %	198	1
read_200MB.fq	400 000	95.71 %	3.85 %	0.44 %	421	1
read_300MB.fq	600 000	95.68 %	3.87 %	0.45 %	542	2
read_400MB.fq	800 000	95.72 %	3.83 %	0.45 %	717	2

Vidimo da su svi rezultati jako slični. Poravnanje je gotovo identično u svakom od prethodnih pet slučajeva. Vrijeme izvođenja i memorijska potrošnja su jedini parametri koji se više razlikuju zbog same veličine ulaznih datoteka.

Na prethodnih šest primjera demonstrirali smo glavnu prednost SNAP-a, a to je upravo njegova brzina. Ako se žele raditi kompleksnija mjerenja, onda je potrebno imati jače računalo, ali za uporabu na manje kompleksnim projektima SNAP je izvrstan izbor jer nudi izrazito brza i točna očitavanja.

2.4. Mogućnost poboljšanja SNAP-a

Glavni nedostatak SNAP-a u smislu statističkih podataka je to što ne pokazuje sve bitne podatke. Vrijeme bi moglo biti mjereno u *ms* radi veće preciznosti i bilo bi poželjno imati ispis greške u postocima. Takav format izlaznih rezultata bi olakšao analizu performansi i povećao točnost rezultata.

3. BBmap

BBmap je program za očitavanje poravnanja DNA i RNA sekvenci genoma. Ističe se velikom brzinom i preciznošću osobito kod očitavanja velikih genoma. Nema ograničenja veličine očitavanja te je faza indeksiranja vrlo brza u usporedbi s ostalim algoritmima poravnanja. Algoritam je napisao Brian Bushnell u Java programskom jeziku.

3.1. Prednosti i nedostaci

Cilj razvoja BBmap algoritma bila je težnja za softverom koji će nadići sve dosadašnje alate u brzini i točnosti sekvenciranja.

Algoritam je besplatan za korištenje (engl. *open-source*), te zbog toga što je pisan u Javi kompatibilan je sa svim platformama. Jednostavne je instalacije (već je kompajliran – *unzip and run*) te omogućuje jednostavno korištenje uz sve podržane formate datoteka sekvenciranih genoma (fastq, fastq, fq, fasta, fa, fas, fna, ffn, frn, seq, fsa, faa...). Optimiziran je korištenjem automatskog *multithreadinga*, jako je efikasan te se smatra da je najosjetljiviji alat za kratka očitavanja sekvenci. [7]

Ima veliku toleranciju na greške te ga koriste svi veliki instituti i fakulteti koji se bave proučavanjem genetike i razvojem bioinformatike.

Glavni nedostatak BBmap algoritma je veliki utrošak memorije. Svaka referentna baza koristi otprilike 6 bajtova memorije, ali postoji i *low-memory mode* koji minimalno žrtvuje kvalitetu očitavanja s utroškom od otprilike 3 bajta po referentnoj bazi. Velike institucije koje se koriste BBmap algoritmom za sekvenciranje imaju na raspolaganju super računala tako da ovaj glavni nedostatak BBmap algoritma i nije toliko relevantan. [8]

3.2. Princip rada

BBmap pruža mnoštvo opcija za rad s RNA i DNA sekvencama. Algoritam je podijeljen u *shell* skripte koje olakšavaju rad na Linux operacijskom sustavu. Algoritam je specifičan u svom načinu rada; koristi kratke sekvence koje direktno poravnava s genomom. Osnovna ideja iza ove metode je korištenje kratkih sekvenci kako bi označili očitavanja referentnog genoma. Prije nego što su očitavanja obrađena, referentni genom je posebno označen tako da se pozicija određenih sekvenci lako može pronaći. Također, prije obrade, svako očitavanje je podijeljeno na određen broj malih sekvenci čije su pozicije u genomu locirane uz pomoć indeksa genoma. Te pozicije se koriste u određivanju najboljeg poretka očitavanja, koristeći dva posebna vektora: vektor pozicije i vektor praznine. [9]

3.3. Primjer rada

Za demonstraciju rada programa koristit ćemo slične slučajeve kao i u prethodnim primjerima rada drugih algoritama

Skripta uzima ulaznu datoteku koja sadrži sekvencu genoma, svakoj sekvenci pridaje vrijednost, odnosno kvalitetu te stavlja pripadajuće oznake. Takvim postupkom generiramo izlaznu datoteku u *.sam* formatu.

1) Kratak dio sekvence genoma E. Coli

Pokrećemo BBmap u konzoli naredbom:

```
➤ ./bbmap.sh in=ulazSlicni.fq ref=EColi.fna out=out1.sam
```

Results				
Genome:	1			
Key Length:	13			
Max Indel:	16000			
Minimum Score Ratio:	0.56			
Mapping Mode:	normal			
Reads Used:	8	(3192 bases)		
Mapping:	0.095 seconds.			
Reads/sec:	84.65			
kBases/sec:	33.78			
Read 1 data:	pct reads	num reads	pct bases	num bases
mapped:	100.0000%	8	100.0000%	3192
unambiguous:	100.0000%	8	100.0000%	3192
ambiguous:	0.0000%	0	0.0000%	0
low-Q discards:	0.0000%	0	0.0000%	0
perfect best site:	100.0000%	8	100.0000%	3192
semiperfect site:	100.0000%	8	100.0000%	3192
Match Rate:	NA	NA	100.0000%	3192
Error Rate:	0.0000%	0	0.0000%	0
Sub Rate:	0.0000%	0	0.0000%	0
Del Rate:	0.0000%	0	0.0000%	0
Ins Rate:	0.0000%	0	0.0000%	0
N Rate:	0.0000%	0	0.0000%	0
Total time:	4.185 seconds.			

Slika 9 - Rezultat pokretanja programa (1. slučaj)

Slika 9 prikazuje rezultat pokretanja SNAP-a na kratkoj sekvenci genome E. Coli. Rezultati izvođenja prikazani su u tablici 7.

Tablica 7 - Tablični rezultati pokretanja programa (1. slučaj)

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
8	100.000 %	0.000 %	~2	4.185

Ukupno izvršavanje naredbe trajalo je malo više od 4 sekunde, od čega je na mapiranje potrošeno 0.1 sekunda a ostatak na očitavanja. Program je 100% poravnao dani gen s referentnim genom E. Coli uz minimalni utrošak memorije.

2) Kompletna sekvenca genoma E. Coli

Pokrećemo BBmap u konzoli naredbom:

```
➤ ./bbmap.sh in=SlicniFQ.fq ref=EColi.fna out=out2.sam
```

```

----- Results -----
Genome:          1
Key Length:      13
Max Indel:       16000
Minimum Score Ratio: 0.56
Mapping Mode:    normal
Reads Used:      64151 (5132068 bases)

Mapping:         0.433 seconds.
Reads/sec:       148013.28
kBases/sec:      11841.04

Read 1 data:      pct reads      num reads      pct bases      num bases
mapped:           5.2314%         3356           5.2314%         268480
unambiguous:      5.0506%         3240           5.0506%         259200
ambiguous:        0.1808%          116           0.1808%          9280
low-Q discards:   94.6922%         60746         94.6922%        4859668

perfect best site: 5.2314%         3356           5.2314%         268480
semiperfect site: 5.2314%         3356           5.2314%         268480

Match Rate:       NA             NA             100.0000%       268480
Error Rate:       0.0000%         0              0.0000%         0
Sub Rate:         0.0000%         0              0.0000%         0
Del Rate:         0.0000%         0              0.0000%         0
Ins Rate:         0.0000%         0              0.0000%         0
N Rate:           0.0000%         0              0.0000%         0

Total time:       5.095 seconds.

```

Slika 10 - Rezultat pokretanja programa (2. slučaj)

Slika 10 prikazuje rezultat pokretanja SNAP-a na kratkoj sekvenci genome E. Coli. Rezultati izvođenja prikazani su u tablici 8.

Tablica 8 - Tablični rezultati pokretanja programa (2. slučaj)

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	100.000 %	0.000 %	~12	5.095

Naredba je izvršena u nešto više od 5 sekunda od čega je na mapiranje potrošeno oko pola sekunde ostatak na očitavanju. Ponovo imamo 100 postotno poravnanje uz nešto nižu kvalitetu, što je i očekivano.

3) Slučajno generirana sekvenca genoma

Pokrećemo BBmap u konzoli naredbom:

```
➤ ./bbmap.sh in=razlicitFQ.fq ref=EColi.fna out=out3.sam
```

Results				
Genome:	1			
Key Length:	13			
Max Indel:	16000			
Minimum Score Ratio:	0.56			
Mapping Mode:	normal			
Reads Used:	64151 (5132068 bases)			
Mapping:	0.975 seconds.			
Reads/sec:	65784.09			
kBases/sec:	5262.71			
Read 1 data:	pct reads	num reads	pct bases	num bases
mapped:	0.0000%	0	0.0000%	0
unambiguous:	0.0000%	0	0.0000%	0
ambiguous:	0.0000%	0	0.0000%	0
low-Q discards:	0.0000%	0	0.0000%	0
perfect best site:	0.0000%	0	0.0000%	0
semiperfect site:	0.0000%	0	0.0000%	0
Match Rate:	NA	NA	NaN%	0
Error Rate:	0.0000%	0	NaN%	0
Sub Rate:	0.0000%	0	NaN%	0
Del Rate:	0.0000%	0	NaN%	0
Ins Rate:	0.0000%	0	NaN%	0
N Rate:	0.0000%	0	NaN%	0
Total time:	5.065 seconds.			

Slika 11 - Rezultat pokretanja programa (3. slučaj)

Slika 11 prikazuje rezultat pokretanja SNAP-a na kratkoj sekvenci genome E. Coli. Rezultati izvođenja prikazani su u tablici 9.

Tablica 9 - Tablični rezultati pokretanja programa (3. slučaj)

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	0.000 %	100.000 %	~12	5.065

Isti broj očitavanja znači i isti utrošak memorije kao i u prošlom primjeru, uz nešto kraće vrijeme izvođenja od 5 sekundi od čega je skoro cijela sekunda utrošena na mapiranje, ostatak na očitavanja. Imamo 0% poravnanja, tj. 100% različitosti slučajno generiranog genoma s referentnim genom E. Coli.

4) Kompletna sekvenca genome E. Coli generirana putem alata *wgsim*

Pokrećemo BBmap u konzoli s naredbom:

```
➤ ./bbmap.sh ref=EColi.fna in=esch_read_1.fq out=out4.sam
```

Results				
Genome:	1			
Key Length:	13			
Max Indel:	16000			
Minimum Score Ratio:	0.56			
Mapping Mode:	normal			
Reads Used:	1000000 (70000000 bases)			
Mapping:	31.868 seconds.			
Reads/sec:	31379.40			
kBases/sec:	2196.56			
Read 1 data:	pct reads	num reads	pct bases	num bases
mapped:	99.1161%	991161	99.1161%	69381270
unambiguous:	95.1127%	951127	95.1127%	66578890
ambiguous:	4.0034%	40034	4.0034%	2802380
low-Q discards:	0.0000%	0	0.0000%	0
perfect best site:	22.9352%	229352	22.9352%	16054640
semiperfect site:	22.9352%	229352	22.9352%	16054640
Match Rate:	NA	NA	97.9314%	67950509
Error Rate:	76.8602%	761809	2.0686%	1435317
Sub Rate:	76.7023%	760244	2.0557%	1426350
Del Rate:	0.3077%	3050	0.0066%	4556
Ins Rate:	0.3283%	3254	0.0064%	4411
N Rate:	0.0000%	0	0.0000%	0
Total time:	36.326 seconds.			

Slika 12 prikazuje rezultat pokretanja SNAP-a na kratkoj sekvenci genome E. Coli. Rezultati izvođenja prikazani su u tablici 10.

Tablica 10 - Tablični rezultati pokretanja programa (4. slučaj)

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
1 000 000	97.900 %	2.068 %	289	36.326

Ukupno izvršavanje u ovom slučaju trajalo je 36 sekundi od čega je 32 sekunde trajalo mapiranje, a ostatak poravnavanje. Imamo poravnanje od skoro 98% uz nešto veći *error rate*.

5) Pet različitih datoteka generiranih pomoću *wgsim* alata

Tablica 11 prikazuje rezultate pokretanja programa na pet različitih datoteka generiranih pomoću *wgsim* alata. Učestalost pogrešnih očitavanja iznosi 1%, a duljina očitavanja iznosi 250.

Tablica 11 - Tablični rezultati pokretanja programa (5. slučaj)

Naziv datoteke	Broj očitavanja	Vrijeme (s)	Maksimalna korištena memorija (MB)	Postotak mapiranih očitavanja
read_50MB.fq	99 789	74	1225	99.79 %
read_100MB.fq	199 547	141	1275	99.77 %
read_200MB.fq	399 112	277	1485	99.77 %
read_300MB.fq	598 747	415	1333	99.79 %
read_400MB.fq	798 296	546	1516	99.78 %

4. GraphMap

GraphMap je algoritam za mapiranje očitavanja na genom razvijen prvenstveno za rad s očitanjima nanopore sekvenciranja. Nanopore sekvenciranje je jeftino, dobivena očitavanja imaju veliku duljinu (do 50kbp), ali je točnost očitavanja niska (60-70%). U usporedbi s drugim algoritmima, pokazalo se da GraphMap povećava osjetljivost za 10 – 80% i mapira više od 95% očitavanja dobivenih Nanopore sekvenciranjem [10]. GraphMap je algoritam otvorenog koda i dostupan je na github-u.

4.1. Prednosti i nedostaci

Najveća prednost alata GraphMap je visoka osjetljivost kod mapiranja dugih očitavanja s čestim pogreškama. Nedostatak je spor rad u usporedbi s ostalim testiranim programima, kao i niži broj mapiranih očitavanja za kratka očitavanja.

4.2. Princip rada

GraphMap algoritam se sastoji od pet koraka, svaki od kojih smanjuje set mogućih lokacija poravnanja. U prvom koraku se korištenjem razmaknutih začetaka (engl. *spaced seeds*) smanjuje prostor pretrage i začetci se grupiraju u svrhu grubog poravnanja. U drugom koraku se konstruiraju ishodišne točke, u trećem se ulančavaju ishodišne točke, u četvrtom se odvija finije poravnanje i u petom se obavlja evaluacija ostalih kandidata u svrhu odabira optimalnih lokacija očitavanja.

4.3. Primjer rada

Za demonstraciju rada programa korišteno je 9 ulaznih datoteka:

- Kratak dio genoma E. Coli
- Kompletan genom E. Coli
- Nasumično generirana FASTQ datoteka
- 6 očitavanja generiranih *wgsim* alatom različitih duljina očitavanja i učestalosti pogrešaka

GraphMap algoritam je za svaku od datoteka pokrenut s istim parametrima :

```
➤ ./graphmap align --min-read-len 40 -r esch.fna -d read.fq -o align.sam
```

Ulazni parametri algoritma jednaki su zadanim, osim smanjenja minimalne duljine očitavanja na 40.

Program nakon izvršavanja ispisuje vrijeme izvršavanja i korištenu memoriju, a za određivanje postotka mapiranih očitavanja se koristi jednostavna Python skripta koja broji očitavanja sa MAPQ različitim od 255 u generiranoj *.sam* datoteci.

Primjena algoritma na kratki dio genoma E. Coli daje sljedeće rezultate :

Tablica 12 - Tablični rezultati primjene algoritma na kratki dio genoma

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
8	100.000 %	0.000%	376	0.02

Primjena algoritma na kompletni genom E. Coli daje sljedeće rezultate :

Tablica 13 - Tablični rezultati primjene algoritma na kompletni genom

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	96.882 %	3.118 %	376	199

Primjena algoritma na nasumično generiranu FASTQ datoteku daje :

Tablica 14 - Tablični rezultati primjene algoritma na nasumičnu FASTQ datoteku

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	0.000 %	100.000 %	376	199

Rezultati postignuti primjenom ovog algoritma na očitavanja simulirana *wgsim* alatom :

Tablica 15 - Tablični rezultati primjene algoritma na očitavanja simulirana wgsim alatom

Učestalost pogrešnih očitavanja (%)	Duljina očitavanja	Broj očitavanja	Vrijeme (min)	Maksimalna korištena memorija (MB)	Postotak mapiranih očitavanja
20	50	513 376	45.16	547	0.012 %
20	100	253 388	8.63	428	19.319 %
20	250	102 675	2.58	376	78.454 %
20	500	51 337	2.03	376	96.909 %
20	1000	25 669	1.87	376	97.935 %
35	10 000	20 000	19.29	673	97.350 %

Rezultati nam pokazuju da ovaj algoritam ima vrlo visok postotak mapiranih očitavanja čak i kada u njima postoji velika učestalost pogrešaka.

5. Bowtie2

Bowtie2 je brzi i memorijski učinkovit alat za poravnavanje očitavanja genoma. Razvijen od strane autora: Langmead B, Wilks C, Antonescu V, Charles R., Salzberg SL., čiji radovi su dostupni na linkovima priloženim u literaturi [11] i [12]. Bowtie2 se distribuira pod GPLv2 [15] licencom i koristi se preko komandne linije kod Windows, Mac OS X te Linux operativnog sustava.

5.1. Prednosti i nedostaci

Bowtie2 je posebno je dobar u poravnavanju očitavanja od oko 50 do stotinjak znakova pa sve do relativno dugačkih genoma (npr. genoma sisavaca). Program koristi indeksiranje pomoću FM Indexa (slično sufiksnim poljima) [13] (koji je baziran na Burrows-Wheeler transformaciji ili skraćeno BWT [14]) kako bi osigurao malo zauzeće memorije. Npr. za ljudski genom, Bowtie2 koristi oko 3.2 GB radne memorije računala. Program podržava modove poravnavanja: razmaknuto poravnavanje, lokalno poravnavanje i poravnavanje uparenih krajeva. Također podržava korištenje više procesora (te time i više jezgreno) za veće brzine poravnavanja.

Bowtie2 namijenjen je za usklađivanje relativno kratkih sekvenci očitavanja pa sve do dugačkih genoma. Može se koristiti za proizvoljno male referentne nizove (npr. Amplikone, engl. *Amplicons* [16]), a i vrlo velika očitavanja (npr. u veličinama desetaka do stotinjaka kilobaza) iako je spor kod takvih primjena. Optimiziran je za duljine očitavanja i pogreške dobivenih od tipičnih illumina sekvencera [17]. Bowtie2 ne podržava poravnanje očitavanja u prostoru boja (engl. *colorspace reads*).

5.2. Princip rada

Po default postavkama Bowtie2 provodi poravnavanje s kraja na kraj (engl. *End to end alignment*). Odnosno traži poravnavanje koje uključuje sve pročitane znakove, takvo poravnavanje se zove “nerezano” (engl. *untrimmed or unclipped*).

Kad je u naredbi navedena opcija lokalno (engl. *local*) Bowtie2 vrši lokalno poravnavanje. U ovom modu rada Bowtie2 će možda odrezati određeni broj pročitanih karaktera s jednog ili oba kraja pročitane sekvence ako će to rezultirati boljim poravnanjem.

Primjeri koji slijedi prikazuje *end-to-end* poravnavanje jer uključuje sve pročitane znakove (Slika 12) te “local” poravnavanje koje ne uključuje sve pročitane znakove (Slika 13). Horizontalne linije (“-”) predstavljaju praznine, a vertikalne linije (“|”) gdje se poravnati znakovi poklapaju. Poravnavanje se koristi kako bi napravili educirano nagađanje gdje se pročitani niz nalazi u odnosu na referentni genom. Često nije moguće točno odrediti mjesto poravnavanja, npr. ako referentni genom sadrži velike nizove jednakih znakova (pr. TTTTTTTTTTTTTTTTTT...) a pročitani niz je kratki niz ponavljajućih znakova (TTTTTT) ne možemo sa sigurnošću odrediti od kuda je pročitani niz došao.

```
Read:      GACTGGGCGATCTCGACTTCG
Reference: GACTGCGATCTCGACATCG

Alignment:
Read:      GACTGGGCGATCTCGACTTCG
          ||||| ||||| ||||| |||
Reference: GACTG--CGATCTCGACATCG
```

Slika 12 – End to end poravnavanje

```
Read:      ACGGTTGCGTTAATCCGCCACG
Reference: TAACTTGCGTTAAATCCGCCTGG

Alignment:
Read:      ACGGTTGCGTTAA-TCCGCCACG
          ||||| ||||| |||||
Reference: TAACTTGCGTTAAATCCGCCTGG
```

Slika 13 – Lokalno poravnavanje

Preko index tablica računa se ocjena poravnavanja. Ocjena poravnavanja kvantificira koliko je pročitani niz sličan referentnom nizu (tj. koliko dobro je poravnat). Što je ocjena veća to je bolje poravnavanje. Ocjena se računa oduzimanjem bodova za svaku razliku (engl. *mismatch*, *gap*) te kod lokalnog poravnavanja dodaje se bonus za svako poklapanje.

Algoritam poravnavanja ne može uvijek s velikim povjerenjem odrediti točno porijeklo očitano niza. Očitavanje koje je nastalo iz ponavljajućeg dijela može se jednako dobro poklapati s više mjesta u referentnom genomu. Algoritam zato izražava stupanj pouzdanosti da određeno očitavanje pripada tamo gdje je poravnato, ta pouzdanost se skraćeno označava MAPQ (engl. *Mapping quality*). MAPQ je povezana sa “jedinstvenošću”, tj. poravnavanje je jedinstveno ako ima znatno bolju ocjenu poravnavanja od ostalih mogućih poravnavanja. Što je veća razlika između najboljeg i drugog najboljeg poravnavanja to je najbolje poravnavanje više jedinstveno tj. ima veći MAPQ.

5.3. Primjer rada

Kod demonstracije rada Bowtie2 programa korišteni su podaci opisani na početku 2.3. paragrafa. Korištene su sve default postavke.

Bowtie2 se skida kao *.zip* datoteka te je spreman za korištenje kod Linux i MAC OS X operacijskih sustava, kod Windows operacijskog sustava potrebno je prethodno kompajliranje korištenjem MinGW-a. Kod svih operativnih sustava potrebno je dodati Bowtie2 datoteke u ‘put’ (engl. *path*, *environment variable*) kako bi se mogao program koristiti bilo gdje u komandnoj liniji (Slika 14).

```
GNU nano 2.0.6      File: .bash_profile

# Setting PATH for Python 3.6
# The original version is saved in .bash_profile.pysave
PATH="/Library/Frameworks/Python.framework/Versions/3.6/bin:${PATH}"
export PATH

# Setting PATH for Python 3.7
# The original version is saved in .bash_profile.pysave
PATH="/Library/Frameworks/Python.framework/Versions/3.7/bin:${PATH}"
export PATH

# Setting PATH for bowtie2
export PATH=$PATH:/Users/zachary/Desktop/bowtie2
```

Slika 14 – Dodavanje Bowtie u 'path' operativnog sustava

Za korištenje Bowtie2 prvo je potrebno indeksirati referentni genom E. Coli što generira 6 datoteka koje se koriste kod daljnjih primjera za poravnavanje. Pozicioniramo se u bilo koji folder koji želimo te pokrenemo sljedeću naredbu u komandnoj liniji (potrebno je osigurati da se datoteka Ecoli.fna nalazi u direktoriju koji je naveden u naredbi):

➤ BT2_HOME/bowtie2-build \$BT2_HOME/example/myReference/Ecoli.fna
e_coli



Slika 15 – Generirane index datoteke E.Coli

Index datoteke zajedno zauzimaju 14.9 MB te je potrebno oko 10 sekundi da se generiraju.

1) Kratak dio sekvence genoma E. Coli

Izvršavanjem sljedeće naredbe pokrećemo Bowtie2 nad datotekom ulazSlicni.fq, u komandnoj liniji se ispisuju podaci o poravnavanju te se generira datoteka ulazSlicni.sam (veličine 7 kB) koja sadrži dodatne informacije za daljnje analize:

```
$BT2_HOME/bowtie2 --local e_coli -U
$BT2_HOME/example/myReads/ulazSlicni.fq -S ulazSlicni.sam
```

```
8 reads; of these:
 8 (100.00%) were unpaired; of these:
 0 (0.00%) aligned 0 times
 8 (100.00%) aligned exactly 1 time
 0 (0.00%) aligned >1 times
100.00% overall alignment rate
```

Slika 16 – Rezultat poravnavanja nad ulazSlicni.fq

Slika 16 prikazuje rezultat pokretanja Bowtie2-a na kratkoj sekvenci genoma E. Coli. Rezultati izvođenja prikazani su u tablici 16.

Tablica 16 – Tablični rezultati na kratkoj sekvenci genoma E.Coli

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
8	100.00 %	0.00 %	7	~1

2) Kompletna sekvenca genoma E. Coli

Izvršavanjem sljedeće naredbe pokrećemo Bowtie2 nad datotekom slicniFQ.fq, u komandnoj liniji se ispisuju podaci o poravnavanju te se generira datoteka slicniFQ.sam (veličine 17.1 MB) koja sadrži dodatne informacije za daljnje analize:

```
$BT2_HOME/bowtie2 --local e_coli -U
$BT2_HOME/example/myReads/slicniFQ.fq -S slicniFQ.sam
```

```
64151 reads; of these:
64151 (100.00%) were unpaired; of these:
 0 (0.00%) aligned 0 times
59578 (92.87%) aligned exactly 1 time
 4573 (7.13%) aligned >1 times
100.00% overall alignment rate
```

Slika 17 – Rezultat poravnavanja nad slicniFQ.fq

Slika 17 prikazuje rezultat pokretanja Bowtie2-a na kompletnoj sekvenci genoma E. Coli. Rezultati izvođenja prikazani su u tablici 17.

Tablica 17 – Tablični rezultati na kompletnoj sekvenci genoma E.Coli

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	92.87 %	0.00 %	11	~1

Slika 17 pokazuje da je 92.87 % poravnato pouzdano dok 7.13% nije pouzdano poravnato, nepouzdan poravnavanje nam govori da postoji vjerojatnost od 10% da očitavanje pripada nekom drugom mjestu, a ne tamo gdje ga je Bowtie2 poravnao. Memorijski zahtjevi su se nešto povećali zbog povećane veličine ulazne datoteke naspram prvom primjeru, a vrijeme izvođenja je i dalje oko 1 sekunde.

3) Slučajno generirana sekvenca genoma E. Coli

Pokrećemo naredbu nad `razlicitFQ.fq` datotekom te slijede podaci o poravnavanju i generira se datoteka `razlicitFQ.sam` (veličine 12.6 MB):

```
$BT2_HOME/bowtie2 --local e_coli -U
$BT2_HOME/example/myReads/razlicitFQ.fq -S razlicitFQ.sam
```

```
64151 reads; of these:
  64151 (100.00%) were unpaired; of these:
    64151 (100.00%) aligned 0 times
    0 (0.00%) aligned exactly 1 time
    0 (0.00%) aligned >1 times
0.00% overall alignment rate
```

Slika 18 – Rezultat poravnavanja nad razlicitFQ.fq

Slika 18 prikazuje rezultat pokretanja Bowtie2-a na slučajno generiranoj sekvenci genoma E. Coli. Rezultati izvođenja prikazani su u tablici 18.

Tablica 18 – Tablični rezultati na kompletnoj sekvenci genoma E.Coli

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	0.00 %	100.00 %	10	~1

Slučajno generirana sekvenca nema poklapanja s originalnom sekvencom, stoga vidimo da Bowtie2 nije pronašao poravnanja.

4) Kompletan sekvenca genoma E.Coli i slučajno generirana sekvenca genoma

Pokrećemo naredbu nad datotekama `razlicitFQ.fq` i `slicniFQ.fq` te slijede podaci o poravnavanju i generira se datoteka `samTwo.sam` (veličine 31.5 MB):

```
$BT2_HOME/bowtie2 -x e_coli -1
$BT2_HOME/example/myReads/razlicitFQ.fq -2
$BT2_HOME/example/myReads/slicniFQ.fq -S samTwo.sam
```



```

64151 reads; of these:
  64151 (100.00%) were paired; of these:
    64151 (100.00%) aligned concordantly 0 times
    0 (0.00%) aligned concordantly exactly 1 time
    0 (0.00%) aligned concordantly >1 times
  ----
  64151 pairs aligned concordantly 0 times; of these:
    0 (0.00%) aligned discordantly 1 time
  ----
  64151 pairs aligned 0 times concordantly or discordantly; of these:
    128302 mates make up the pairs; of these:
      64151 (50.00%) aligned 0 times
      60865 (47.44%) aligned exactly 1 time
      3286 (2.56%) aligned >1 times
50.00% overall alignment rate

```

Slika 19 – Rezultat poravnavanja nad različitFQ.fq i slicniFQ.fq (Paired-end example)

Slika 19 prikazuje rezultat pokretanja Bowtie2-a na dvije sekvence genoma E. Coli. Rezultati izvođenja prikazani su u tablici 19.

Tablica 19 – Tablični rezultati na dvije sekvence genoma E.Coli

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
64 151	47.44 %	50.00 %	9.2	6

Slika 18 pokazuje da je 47.44 % poravnato pouzdano dok 2.56% nije pouzdano poravnato, nepouzdanost poravnavanja nam govori da postoji vjerojatnost od 10% da očitavanje pripada nekom drugom mjestu, a ne tamo gdje ga je Bowtie2 poravnao. Ukupno poravnanje je 50%.

5) Kompletan sekvenca genoma E.Coli generirana alatom *wgsim*

Pokrećemo naredbu nad `esch_read_1.fq` datotekom te slijede podaci o poravnavanju i generira se datoteka `esch_read_1.sam` (veličine 288.2 MB):

```

$BT2_HOME/bowtie2 --local -x e_coli -U
$BT2_HOME/example/myReads/esch_read_1.fq -S esch_read_1.sam

```

```

1000000 reads; of these:
  1000000 (100.00%) were unpaired; of these:
    10523 (1.05%) aligned 0 times
    923157 (92.32%) aligned exactly 1 time
    66320 (6.63%) aligned >1 times
98.95% overall alignment rate

```

Slika 20 – Rezultat poravnavanja nad esch_read_1.fq

Slika 20 prikazuje rezultat pokretanja Bowtie2-a na kompletnoj sekvenci genoma E. Coli. Generiranoj *wgsim* alatom. Rezultati izvođenja prikazani su u tablici 20.

Tablica 20 – Tablični rezultati na kompletnoj sekvenci generiranoj wgsim alatom

Ukupan broj čitanja	Poravnato	Različito	Memorija (MB)	Vrijeme izvođenja (s)
1 000 000	92.32 %	1.05 %	14.2	57

Slika 20 pokazuje da je 92.32 % poravnato pouzdano dok 6.63% nije pouzdano poravnato, nepouzdanost poravnanja nam govori da postoji vjerojatnost od 10% da očitavanje pripada nekom drugom mjestu, a ne tamo gdje ga je Bowtie2 poravnao. Ukupno poravnanje je 98.85%. Memorijske potrebe relativno malo povećale (~50%), ali se vrijeme izvođenja znatno povećalo (9-57 puta više od ostalih primjera).

6) Pet različitih datoteka generiranih pomoću wgsim alata (FASTQ format)

Pomoću wgsim alata smo generirali pet različitih datoteka od 50, 100, 200, 300 i 400 MB. Sve datoteke imaju duljinu očitavanja od 250 baza i grešku od 1%. Tablica 21 prikazuje rezultate pokretanja SNAP-a na ulaznim podacima.

Tablica 21 – Tablični rezultati pokretanja programa (šesti slučaj)

Naziv datoteke	Ukupan broj čitanja	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
read_50MB.fq	100 000	91.95 %	8.05 %	0.00 %	25.8	101
read_100MB.fq	200 000	91.90 %	8.10 %	0.00 %	25.9	206
read_200MB.fq	400 000	91.99 %	8.01 %	0.00 %	26.1	405
read_300MB.fq	600 000	91.95 %	8.05 %	0.00 %	27.3	616
read_400MB.fq	800 000	91.92 %	8.08 %	0.00 %	28.0	815

Poravnanje je skoro identično u svakom od pet slučajeva. Memorijska potrošnja se minimalno razlikuje. Vrijeme izvođenja je jedini parametar koji se drastično mijenja, što je veća datoteka duže traje proces.

6. Usporedni prikaz rezultata

Tablica 22 – Tablični rezultati na kratkoj sekvenci genoma E.Coli (ulazSlicni.fq)

	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	100.00 %	0.00 %	0.00 %	56.0	< 1
BBmap	100.00 %	0.00 %	0.00 %	~2.0	4.185
GraphMap	100.00 %	0.00 %	0.00 %	376.0	0
Bowtie2	100.00 %	0.00 %	0.00 %	7.0	~1

Tablica 23 – Tablični rezultati na kompletnoj sekvenci genoma E.Coli (slicniFq.fq)

	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	94.98 %	5.02 %	0.00 %	68.0	< 1
BBmap	94.69 %	5.23 %	0.00 %	12.0	5.095
GraphMap	94.95 %	1.92 %	3.12 %	376.0	199
Bowtie2	92.87 %	7.13 %	0.00 %	11.0	~1

Tablica 24 – Tablični rezultati na slučajno generiranoj sekvenci genoma E.Coli (razlicitFq.fq)

	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	0.00 %	0.00 %	100.00 %	68.0	< 1
BBmap	0.00 %	0.00 %	100.00 %	12.0	5.065
GraphMap	0.00 %	0.00 %	100.00 %	376.0	199
Bowtie2	0.00 %	0.00 %	100.00 %	11.0	~1

Tablica 25 – Tablični rezultati na Kompletna sekvenca genoma E.Coli i slučajno generirana sekvenca genoma (razlicitFq.fq i slicniFq.fq)

	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	47.49 %	2.51 %	50.00 %	80.0	< 1
BBmap	/	/	/	/	/
GraphMap	/	/	/	/	/
Bowtie2	47.44 %	2.56 %	50.00 %	9.2	6

Tablica 26 – Tablični rezultati na kompletnoj sekvenci geneiranoj wgsim alatom

	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	94.26 %	4.74 %	1.01 %	243.0	3
BBmap	97.90 %	4.00%	2.07 %	289.0	36
GraphMap	94.57 %	1.77 %	3.66 %	802.0	3827
Bowtie2	92.32 %	6.63 %	1.05 %	14.2	57

Tablica 27 – Tablični rezultati na ulaznoj datoteci read_50MB.fq

	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	95.80 %	3.83 %	0.38 %	78.0	< 1
BBmap	96.09 %	3.70 %	0.21 %	1225.0	74
GraphMap	95.86 %	1.55 %	2.59 %	376.0	242
Bowtie2	91.95 %	8.05 %	0.00 %	25.8	101

Tablica 28 – Tablični rezultati na ulaznoj datoteci read_100MB.fq

	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	95.69 %	3.87 %	0.44 %	198.0	1
BBmap	96.03 %	3.74 %	0.23 %	1275.0	141
GraphMap	95.85 %	1.49 %	2.66 %	446.0	464
Bowtie2	91.90 %	8.10 %	0.00 %	25.9	206

Tablica 29 – Tablični rezultati na ulaznoj datoteci read_200MB.fq

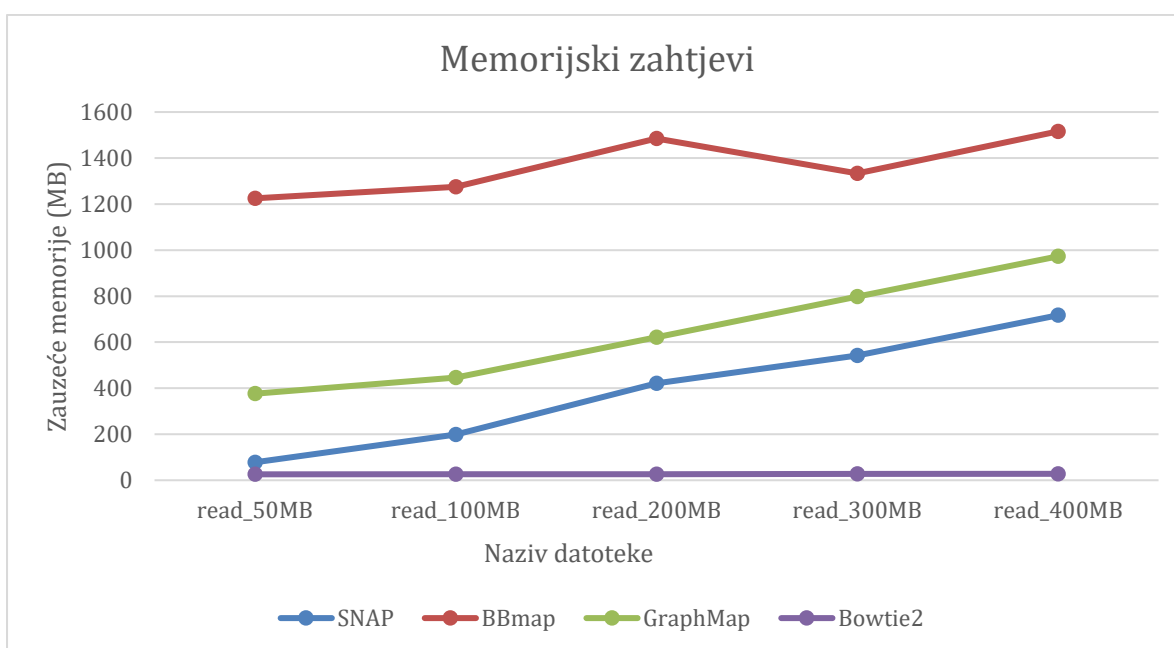
	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	95.71 %	3.85 %	0.44 %	421.0	1
BBmap	96.07 %	3.70 %	0.23 %	1485.0	277
GraphMap	95.85 %	1.52 %	2.63 %	622.0	928
Bowtie2	91.99 %	8.01 %	0.00 %	26.1	405

Tablica 30 – Tablični rezultati na ulaznoj datoteci read_300MB.fq

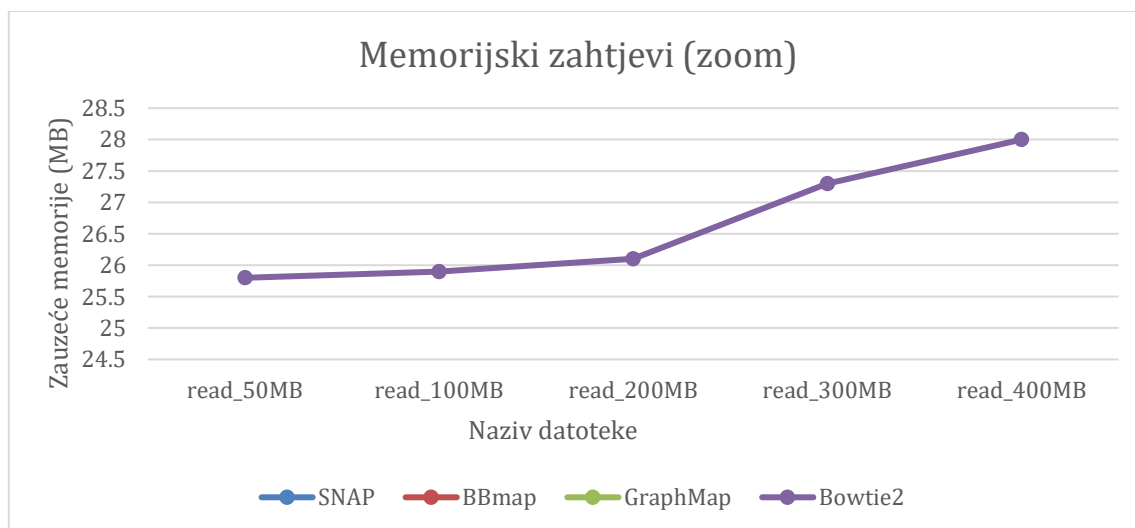
	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	95.68 %	3.87 %	0.45 %	542.0	2
BBmap	96.06 %	3.73 %	0.21 %	1333.0	415
GraphMap	95.84 %	1.52 %	2.64 %	798.0	1409
Bowtie2	91.95 %	8.05 %	0.00 %	27.3	616

Tablica 31 – Tablični rezultati na ulaznoj datoteci read_400MB.fq

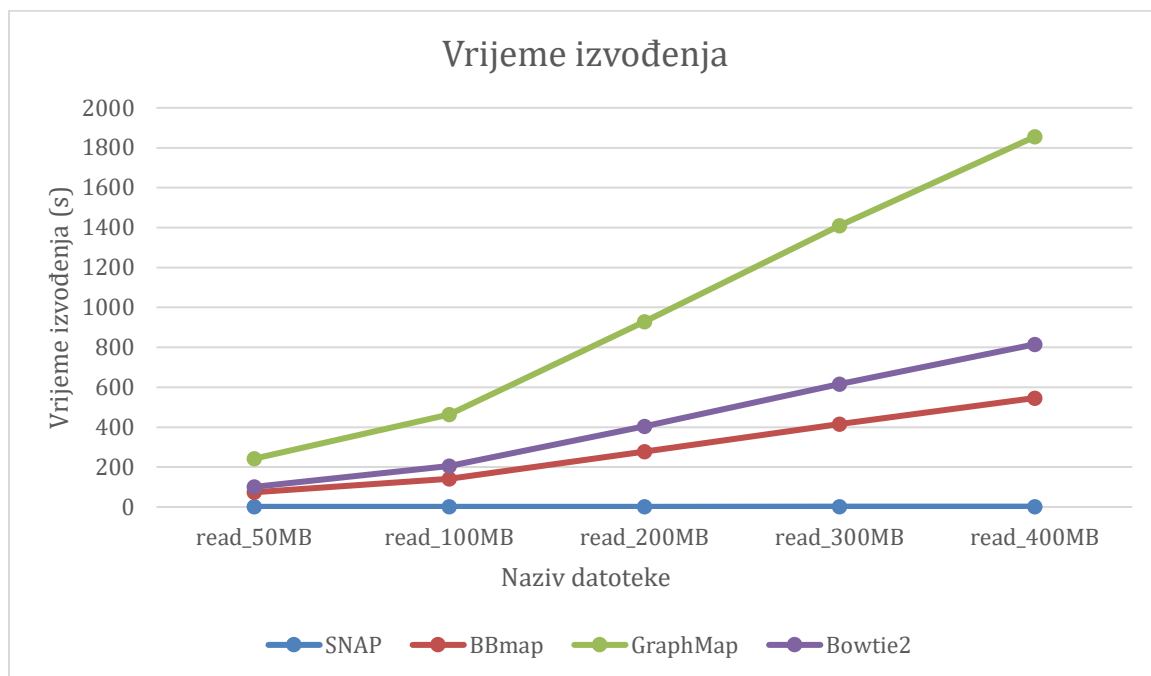
	Poravnato (MAPQ >= 10)	Poravnato (MAPQ < 10)	Različito	Memorija (MB)	Vrijeme izvođenja (s)
SNAP	95.72 %	3.83 %	0.45 %	717.0	2
BBmap	96.09 %	3.69 %	0.22 %	1516.0	546
GraphMap	97.40 %	1.50 %	2.60 %	973.0	1855
Bowtie2	91.92 %	8.08 %	0.00 %	28.0	815



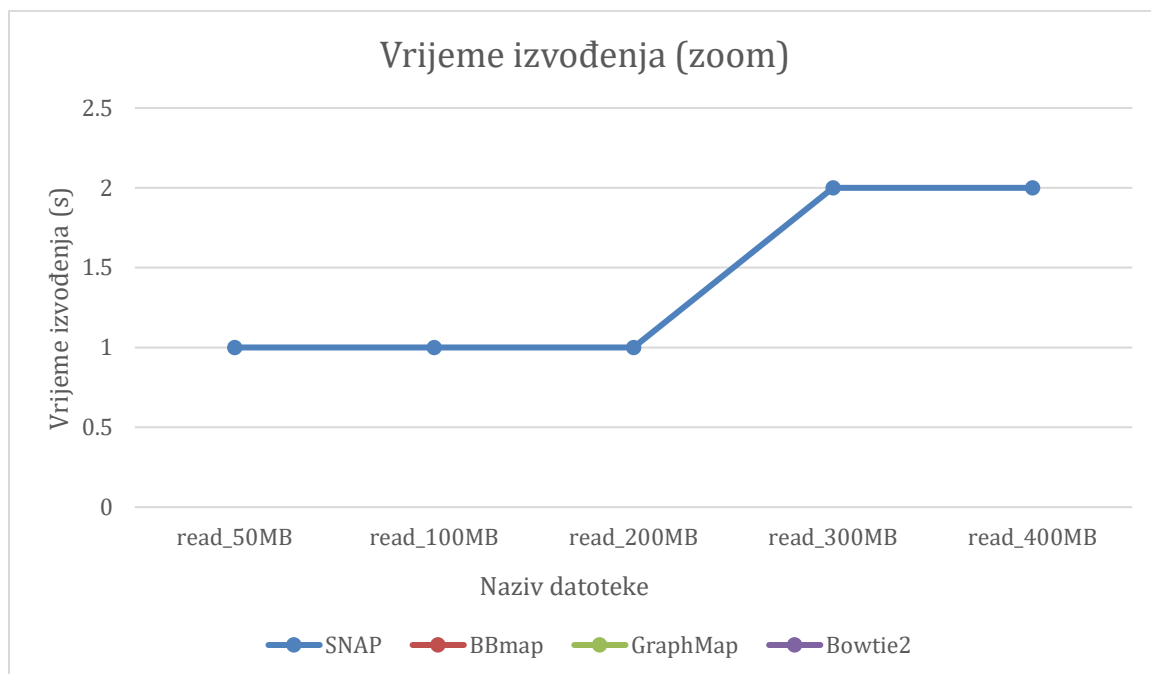
Slika 21 – Memorijski zahtjevi programa



Slika 22 – Memorijski zahtjevi programa (uvećano)



Slika 23 – Vrijeme izvođenja programa



Slika 24 – Vrijeme izvođenja programa (uvećano)

7. Zaključak

Glavno pitanje koje se uvijek postavlja prije početka projekta je: „Koju tehnologiju trebamo koristiti za rad na projektu?“. Odgovor na to pitanje, u većini slučajeva, nije jednoznačan. Svaki projekt može sadržavati veći broj tehnologija koje su pogodne za ostvarivanje cilja.

Ovaj projekt nastoji pomoć u donošenju takve odluke prilikom rada u području bioinformatike. Velik broj alata za mapiranje je danas dostupan svakom znanstveniku, ali odluka o korištenju i izboru alata ovisi isključivo o specifikacijama projekta.

Ako projekt zahtjeva prije svega izrazito brz alat, a dostupno je računalo s dovoljnom količinom memorije, onda je SNAP pravi izbor za takav projekt.

S druge strane, ako je potrebna preciznost, a vrijeme izvođenja nije ključan faktor, onda GraphMap dolazi kao savršen alat za takav projekt.

BBMap i Bowtie2 imaju najbolji omjer vremena izvođenja, preciznosti i potrošnje, stoga su oni idealan alat za korištenje u studentskim istraživačkim projektima.

Svaki alat posjeduje prednosti i mane, stoga ne postoji jedinstven pobjednik među njima. Na voditelju svakog projekta je da se dovoljno informira i donese pravu odluku o tehnologiji koju će koristiti.

Literatura

- [1] <http://snap.cs.berkeley.edu/downloads/snap-1.0beta-manual.pdf>
- [2] <https://www.ebi.ac.uk/ena/data/view/SRX5309760>
- [3] https://www.ncbi.nlm.nih.gov/genome/167?genome_assembly_id=161521
- [4] <http://snap.cs.berkeley.edu/downloads/snap-1.0beta-quickstart.pdf>
- [5] <https://github.com/PacificBiosciences/DevNet/wiki/E-coli-K12-MG1655-Resequencing>
- [6] <https://github.com/lh3/wgsim>
- [7] <https://sourceforge.net/projects/bbmap/>
- [8] <https://drive.google.com/file/d/0B3llHR93L14wbks0ZURFcFhFR1E/edit>
- [9] Josip Marić, „MASTER THESIS no. 1005 Long Read RNA-seq Mapper“, Zagreb, February 2015.
- [10] <https://www.nature.com/articles/ncomms11307.pdf>
- [11] <https://academic.oup.com/bioinformatics/article/35/3/421/5055585>
- [12] <https://www.nature.com/articles/nmeth.1923>
- [13] <https://en.wikipedia.org/wiki/FM-index>
- [14] https://en.wikipedia.org/wiki/Burrows%E2%80%93Wheeler_transform
- [15] <http://www.gnu.org/licenses/gpl-3.0.html>
- [16] <https://en.wikipedia.org/wiki/Amplicon>
- [17] <https://www.illumina.com/systems/sequencing-platforms.html>