SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Bioinformatika

Preklapanje (overlap) nizova u sastavljanju genoma koristeći semi-globalno poravnanje

Andrea Faltis 0036454009
Matej Filković 0165045504
Davor Obilinović 0036426857

Zagreb, siječanj 2014.

Sadržaj

1. Uvod

Sastavljanje genoma je postupak rekonstrukcije genoma s ciljem dobivanja što više informacija koji genom sadrži. Današnje metode sekvenciranja genoma ne mogu pročitati cijeli genom nekog organizma odjednom, već ga čitaju u dijelovima raznih duljina uz mogućnosti pojave različitih pogrešaka. Uobičajeno je da povećanjem duljine očitanja raste i broj grešaka. Iz ovako dobivenih očitanja pokušava se računalnim algoritmima rekonstruirati originalni genom. Danas postoji više različitih algoritama i pristupa koji se koriste prilikom rekonstrukcije genoma. Pronalazak preklapanja sekvenci jedan je od neizostavnih koraka algoritama sastavljanja genoma. U ovome radu razmatrana je metoda semi-globalnog poravnanja za pronalazak odgovarajućih preklapanja dijelova pojedenih sekvenci. Iako nije proveden postupak sastavljana genoma iz niza očitanja, promatranjem određenih parametara poravnanja provedena je analiza uspješnosti algoritma semi-globalnog poravnavanja.

U poglavlju Sastavljanje genoma opisan je današnji pristup sastavljanju genoma i neki problemi karakteristični problemi koji se pojavljuju. U poglavlju Poravnavanje sekvenci dan je kratak uvod u problem poravnanja dviju sekvenci. Prikazani su neki od osnovnih mehanizama i algoritama korišteni prilikom poravnanja dviju sekvenci. Poglavlje Formati podataka sadrži opis dva najpoznatija formata za spremanje nizova nukleotida. U poglavlju simulacija prikazani su parametri koji služe za evaluaciju rezultata danih u poglavlju Rezultati.

2. Sastavljanje genoma

Cilj sastavljanja genoma je doći do što više informacija koje genom sadrži. U ovu svrhu razvijeno je više različitih postupaka očitavanja strukture genoma te sastavljanja iste računalnim metodama. Suvremeni postupak sastavljanja genoma živog organizma koristeći računalne metode može se opisati u nekoliko koraka.

Prvi korak u sastavljanju genoma jest rastavljanje genoma na poprilično velik broj fragmenata. Veličine fragmenata su slučajno odabrane. Na ovako dobivenim fragmentima provodi se očitanje nukleotida. Očitanje je ništa drugo nego sam poredak nukleotida koji se sprema u računalno pogodan oblik. Duljina fragmenata je uvijek veća od najvećeg broja nukleotida koji je današnjom tehnologijom moguće pročitati. Ovaj korak ponavlja se više puta sve dok se ne osigura da je svaki nukleotid pročitan barem jedanput. Koeficijent kojim je izražen broj uzrokovanja cijelog genoma na ovaj način naziva se pokrivenost (engl. *coverage*). Pokrivenost je jednaka omjeru umnoška broja očitanja i njihove prosječne duljine te ukupne duljine genoma.

$$C = \frac{N \cdot L}{G} \tag{2-1}$$

U relaciji 2-1 s *N* je označen broj očitanja, *L* predstavlja prosječnu duljinu očitanja dok *G* duljinu genoma koji se čita. Iz dobivenih očitanja potrebno je sastaviti genom. U ovu svrhu koriste se računalni programi zvani asembleri. Rad asemblera ovisi o tehnici sastavljanja u kojoj se primjenjuju. U današnje vrijeme postoje dva glavna pristupa sastavljanju genoma. To su: komparativno i *de novo* sastavljanje genoma.

Komparativno sastavljanje je pristup mapiranja dobivenih očitanja na referentni, već postojeći rekonstruirani genom. Ovaj pristup se uglavnom koristi kod sastavljanja genoma koji je sličan nekom postojećem otprije sastavljenom genomu.

Nasuprot komparativnom sastavljanju genoma de novo pristup sastavlja genom bez pomoći bilo kakvog referentnog genoma. Pristup se temelji na traženju najvećeg zajedničkog niza znakova pojedinih očitanja. Moguće je zaključiti kako je ovakav pristup znatno složeniji.

3. Poravnavanje sekvenci

U bioinformatici se često postavlja pitanje kako za dva određena niza, bez poznavanja evolucijskih događaja, odrediti imaju li oni zajedničko podrijetlo ili pak odrediti pripada li jedan niz drugome. Pretpostavimo da samo izdvojili nizove ATAAGC i AAAAAC. Želimo li te nizove usporediti i utvrditi njihovu sličnost, napisat ćemo ih jedan ispod drugog, tako da je svaki znak drugog niza potpisan točno ispod jednog znaka prvog niza, kao što je vidljivo na primjeru:

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

Ovaj se postupak naziva poravnanje nizova, a za odgovarajuće parove znakova kažemo da su poravnati. Jednakost dvaju poravnatih slova zovemo podudaranjem. Poravnanje nizova zapravo slikovito prikazuje moguću evoluciju nizova, stoga bi velik broj podudarajućih znakova mogao sugerirati evolucijsku vezu. U slučaju nizova različitih duljina, kao što su npr. CGGGGC i CGCACTAGC na odgovarajuća mjesta u nizovima umetnut ćemo znakove '-' i time dobiti nizove jednakih duljina koje možemo poravnati. Time zapravo pretpostavljamo da je u evoluciji došlo do umetanja (engl. *insertion*) ili brisanja (engl. *delition*), pa za poravnanje tog znaka s bilo kojim drugim znakom upotrebljavamo prazninu (engl. *indel*). Gornje nizove primjerice možemo poravnati na 2 načina:

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

To naravno nisu jedine mogućnosti. Broj mogućih poravnanja jako velik i cilj je pronaći poravnanja koja izražavaju najveću sličnost među nizovima. Kako bi se poravnanja mogla uspoređivati uvedena je ocjena poravnanja. Optimalno poravnanje bit će ono s najvećom ocjenom. Zbog velike duljine nizova, velikog broja mogućih poravnanja, optimalno poravnanje nije lako naći.

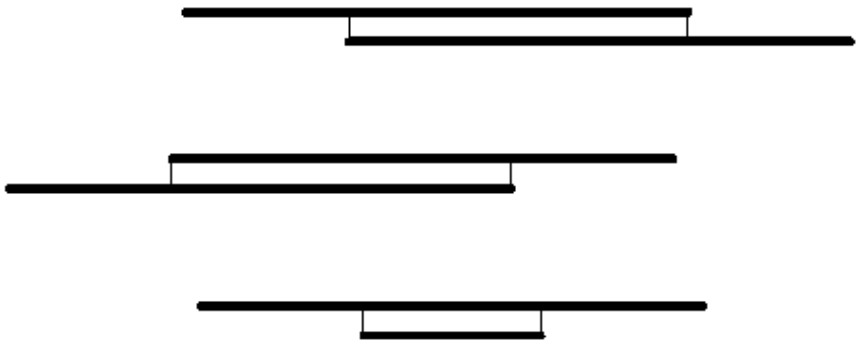
Postoje razni algoritmi vezani uz ovaj problem, a najpoznatiji su Needleman-Wunsch algoritam za globalno i Smith-Waterman algoritam za lokalno poravnanje. Globalno poravnanje je poravnanje prilikom kojeg su iskorištena sva slova u oba niza. Kod lokalnog poravnanja poravnavamo samo dijelove niza, pa se to poravnanje svodi na traženje najduljeg podudarajućeg segmenta. Također, prilikom lokalnog poravnanja možemo ili ne moramo dopustiti praznine. Postoje i tzv. višestruka poravnanja kada, kako sama riječ kaže, poravnavamo više nizova odjednom.

3.1 Semi-globalno poravnavanje sekvenci

Semi-globalno poravnavanje je hibridna metoda, kojom se pokušava pronaći poravnanje koje uključuje početak ili kraj jedne ili druge sekvence. Ovakva metoda je osobito korisna kada početni dio jedne sekvence poklapa s djelom kraja druge sekvence. Drugi slučaj za koji je semi-globalna metoda korisna je poravnanje kraće sekvence s dužom. U ovom slučaju, kraća sekvenca bi trebala biti globalno poravnana, a za dužu sekvencu je poželjno lokalno poravnanje.
Algoritam semi-globalnog poravnanja je modifikacija Needleman-Wunsch algoritma globalnog poravnanja. Needleman-Wunsch algoritam s početnim uvjetima ne može pronaći značajno preklapanje sekvenci stoga je potrebno unijeti određene modifikacije u algoritam. Razlika poravnanja kraće sekvence s duljom korištenjem algoritama semi-globalnog i globalnog poravnanja prikazana je slijedećim primjerom:

Semi-globalno poravnanje:	Globalno poravnanje:
C A G C A - C T T G G A T T C T C G G	C A G C A C T T G G A T T C T C G G
- - - C A G C GT G G - - - - - - - -	C A G C - - - - - G - T - - - - G G

Vidimo da prvo poravnanje ima više praznina od drugog, ali s obzirom da su praznine na početku i na kraju dopuštene, kazna je puno manja u prvom slučaju nego u drugom. Ovakvo poravnanje je izuzetno korisno kod traženja preklapanja nizova i mapiranja kraćeg niza na dulji. Primjeri preklapanja nizova prikazani su na slici 3.1.



Slika 3.1: Prikaz preklapanja nizova

Prva dva primjera na slici 3.1, prikazuju preklapanje nizova gdje se kraj jednog niza preklapa s početkom drugoga. Nakon što su pronađena ovakva preklapanja među svima nizovima moguće je rekonstruirati originalni niz iz većeg broja manjih nizova. Treći primjer prikazuje poravnanje kraćeg niza s duljim nizom.

Ako koristimo Needleman-Wunsch algoritam s početnim uvjetima nije moguće pronaći značajno preklapanje. Kako bi se to izbjeglo u tablici dinamičkog programiranja početne praznine se ne kažnjavaju. Postupak je prikazan slijedećim tablicama:

	-	y_1	y_2	…	y_n
-	0	g	$-g$	$-2g$	$-ng$
x_1	$-g$				
x_2	$-2g$				
…	…				
x_n	$-mg$				

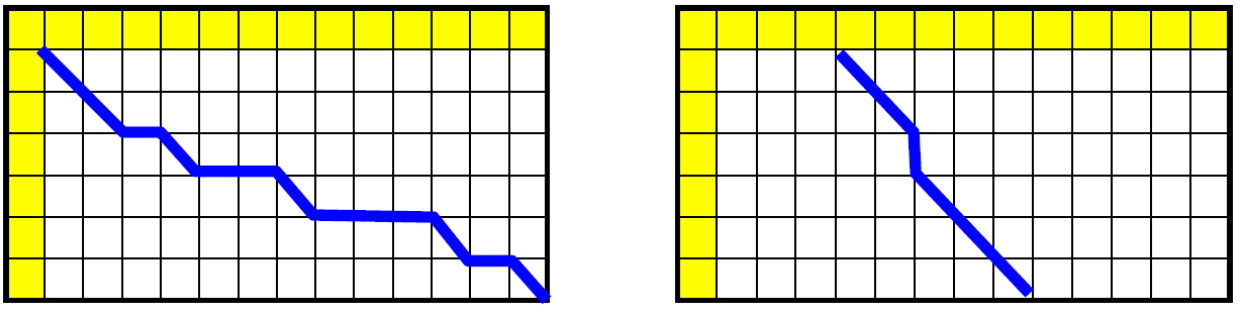
=>

	-	y_1	y_2	…	y_n
-	0	0	0	0	0
x_1	0				
x_2	0				
…	…				
x_n	0				

Razlika između globalnog i semi-globalnog poravnanja je u tome što je rezultat poravnanja maksimalna vrijednost u zadnjem retku (*x* je poravnan s prefiksom od y) ili zadnjem stupcu (*y* je poravnan s prefiksom od *x*). stoga, rezultat poravnanja *A(i,j)* je takav da je $A(i,j) = max_{k,l}(A(k,n),A(m,l))$, dok se poravnanje može dobiti vraćanjem iz *A(i,j*) u *A(0,0)*.

Primjer poravnja kraćeg niza s duljim korištenjem semi-globalnog poravnanja prikazan je na slici 3.2.

```
AGTCAGTGCGTGC        AGTCAG_TGCGTGC
AG_C__T___T_C        ____AGCTTC____
```



Slika 3.2: Prikaz razlike globalnog i semi-globalnog poravnanja

Varijacije semi-globalnog poravnanja mogu odgovarati različitim situacijama kada se praznine zanemare samo za prvu i/ili drugu sekvencu. Također, praznine mogu biti ignorirane samo na početku i/ili kraju sekvence. Mogućnosti su prikazane slijedećom tablicom:

Mjesta gdje se praznine ne kažnjavaju	**Postupak**
Početak prve sekvence	Postavi sve elemente prvog retka tablice na nulu
Kraj prve sekvence	Rezultat poravnanja je maksimum zadnjeg retka
Početak druge sekvence	Postavi sve elemente prvog stupca na nulu
Kraj druge sekvence	Rezultat poravnanja je maksimum zadnjeg stupca

Tablica 3.1: Prikaz postupaka prilikom korištenja semi-globalnog poravnanja

4. Formati podataka

Kod genoma važno je zapisati redoslijed nukleotida, kvalitetu očitanja, početne i konačne pozicije očitanja, taksonomiju živog bića čija su to očitanja. Razvijeno je mnoštvo različitih načina spremanja podataka u raznim formatima. Neki od formata su PHYLIP, REF, FASTQ, FASTA. FASTA format je trenutno najpopularniji format spremanja informacija o sekvencama danas. Za potrebe simulacije očitanja i provedbe poravnanja korišteni su FASTA i FASTQ formati čiji su formati opisani u nastavku .

4.1. FASTA format

FASTA format datoteke sastoje se od dva dijela. Prvi dio datoteke obavezno počinje znakom „>“ nakon kojega obavezno slijedi identifikator sekvence. Dodatno u ovaj dio datoteke mogu biti zapisani i dodatni podatci kao što su položaj niza u genomu ili nekom od proteinskih lanaca itd. Identifikator sekvence slijedi neka važna pravila ovisno o standardu. Tako npr. NCBI (engl. *Nacionalni Center for Biotechnology Information*) koristi vlastiti standard koji od identifikatora sekvence zahtijeva da bude sastavljena od barem tri neizostavna dijela. Prvi dio je referentni broj, drugi je jedinstveni broj svake sekvence koji omogućava lakše praćenje većeg broja varijanti iste sekvence, dok treći broj predstavlja lokus ili položaj slijeda u nizu iz kojeg je dobiven. Drugi dio datoteke je sama sekvenca. Prazni redovi nisu dopušteni u zapisu. Primjer dvodijelnog zapisa u FASTA formatu datoteke:

```
>gi|5524211|gb|AAD44166.1| cytochrome b
LCLYTHIGRNIYYGSYLYSETWNTGIMLLLITMATAFMGYVLPWGQMSFWGATVITNLFSAIPYIGTNLV
EWIWGGFSVDKATLNRFFAFHFILPFTMVALAGVHLTFLHETGSNNPLGLTSDSDKIPFHPYYTIKDFLG
LLILILLLLLLALLSPDMLGDPDNHMPADPLNTPLHIKPEWYFLFAYAILRSVPNKLGGVLALFLSIVIL
GLMPFLHTSKHRSMMLRPLSQALFWTLTMDLLTLTWIGSQPVEYPYTIIGQMASILYFSIILAFLPIAGX
IENY
```

Jedna datoteka može sadržavati više dvodijelnih zapisa, ali je onda u multiFASTA formatu.

4.2. FASTQ format

FASTQ format najpopularniji je oblik pohrane očitanja sekvenci danas. FASTQ format je kreiran iz razloga što kod FASTA formata ne postoji predviđeno mjesto za pohranu ocjene kvalitete očitanja pojedeni baze što je često važan parametar. Za razliku od FASTA formata datoteke svaki zapis u FASTQ datoteci podijeljen je na četiri dijela. Prvi redak zapisa obavezno počinje znakom „@“ nakon kojeg slijedni niz proizvoljnih znakova. Nakon prvog retka slijedi jedan ili više redova niza znakova koji predstavljaju aminokiseline. Nakon niza aminokiselina slijedi redak koji započinje s znakom „+“. Ovaj redak sadrži ocjene kvalitete. U četvrtome retku se nalaze kodovi pridijeljeni pojedinim ocjenama kvalitete. Primjer zapisa u FASTQ formatu:

```
@SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
GGGTGATGGCCGCTGCCGATGGCGTCAAATCCCACC
+SRR001666.1 071112_SLXA-EAS1_s_7:5:1:817:345 length=36
IIIIIIIIIIIIIIIIIIIIIIIIIIII9IG9IC
```

Trenutno postoje tri varijante FASTQ formata datoteke. Originalni FASTQ format naziva se Sanger Fastq. Ovaj format kvalitetu pojedinog očitanja kodira ASCII znakovima između 33 i 126. Pošto druge varijante FASTQ formata datoteke koriste druge znakove prilikom kodiranja dolazi do nekompatibilnosti istih. Također vrijednosti kvalitete sekvence su različite za svaku varijantu formata. Prilikom provedbe simulacije nisu promatrane vrijednosti kvalitete sekvenci, već je FASTQ datoteka korištena samo za pohranu simuliranih očitanja genoma.

5. Simulacija

Očitanja genoma dobivena određenim alatima sadrže nekoliko vrsta pogrešaka. Količina pogrešaka općenito ovisi o duljini očitanja. Neke od pogrešaka u očitanju mogu biti posljedica umetanja (engl. *insertion*), to jest alat za očitanje je ubacio nepostojeći nukleotid na neku poziciju u očitanju. Nasuprot toga, može se dogoditi izostavljanje postojećeg nukleotida (engl. *deletion*). Također uvijek postoji i mogućnost krivog očitanja nukleotida na nekoj poziciji što se naziva mutacija (engl. *mutation*). Za ovaj projekt simulacija očitanja izvedena je alatom *wgsim*[1]. *Wgsim* producira očitanje u FASTQ formatu.

Analiza podataka temelji se na dva pojma iz raspoznavanja uzoraka. To su odziv (engl. *recall*) i preciznost (engl. *precision*). Ove dvije veličine pokazuju koliko je poravnavanje sekvenci korištenjem algoritma semi-globalnog poravnanja uspješno. Visoki odziv predstavlja uspješno detektiranje većine ispravnih pozitiva (engl. *true positives*) dok visoka preciznost predstavlja visok detekciju većeg broja ispravnih od neispravnih pozitiva (engl. *false positive*) . Ispravni pozitiv je svako preklapanje koje je razvijeni program točno detektirao. Nasuprot tome, lažni pozitiv je svako preklapanje koje je razvijeni program pogrešno detektirao, to jest ono poravnanje koje je dao na izlaz, a koje ne bi trebalo biti prisutno. Za ocjenu ispravnih i lažnih poravnanja korišteno je svojstvo alata *wgsim*. Naime, *wgsim* prilikom svakog očitanja daje i podatak o položaju očitanja u referentnom genomu na kojemu se nalazi referentno očitanje. Korištenjem tih podataka moguće je odrediti broj točnih poravnanja. Za dva očitanja možemo reći da su preklapanju ukoliko vrijede slijedeće dvije relacije:

$$x_2 \geq x_1 \tag{5-1}$$

i

$$x_2 \leq e_1 \tag{5-2}$$

U prethodnim relacijama x_2 i x_1 predstavljaju indekse početka prvog i drugog očitanja, dok e_1 predstavlja kraj prvog očitanja. Prema relaciji 5-1 početak drugog očitanja mora

[1] Vise o alatu moguće je pronaći na slijedećoj poveznici: https://github.com/lh3/wgsim

biti nakon početka prvog očitanja ili početi na istom mjestu kao i prvo očitanje. Relacijom 5-2 početak drugog očitanja ograničen je na interval početka i kraja prvog.

Preciznost je sada moguće definirati kao:

$$P = \frac{N_P}{N} \tag{5-3}$$

gdje je s N je označen ukupan broj preklapanja koje je razvijeni program detektirao, a s N_P broj ispravnih pozitiva koje je razvijeni program detektirao. Odziv je definiran slijedećom relacijom:

$$R = \frac{N_P}{N_t} \tag{5-4}$$

gdje je s N_t označen ukupni broj ispravnih preklapanja.

6. Rezultati

U svrhu provedbe poravnanja sekvenci algoritmom semi-globalnog poravnavanja, korišten je dio genoma bakterije *Acaryochloris marina*. Korišten je samo početni dio genoma, otprilike 10^4 nukleotida. Očitanje je simulirano alatom *wgsim* slijedećom naredbom:

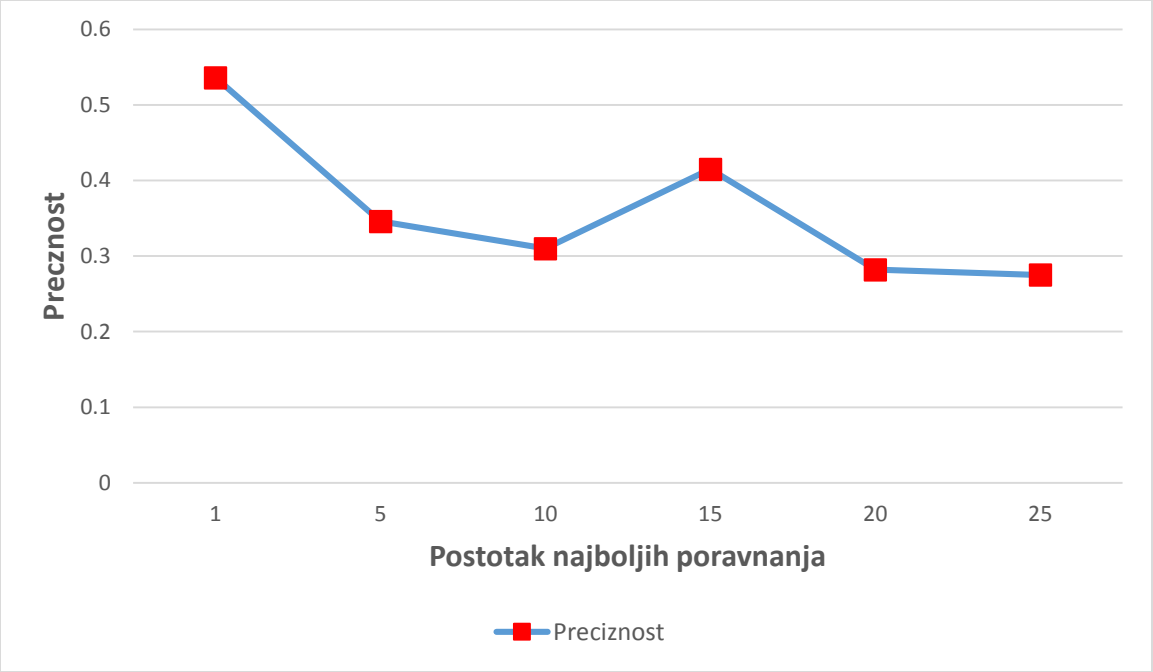
```
./wgsim –N1000 –1100 dioGenomaBakterije.fa readSim.fq /dev/null
```

Ovom naredbom prosječna duljina očitanja postavljena je na 100, broj očitanja na 1000 što dovodi do zaključka da pokrivenost genoma iznosi 10. Broj točnih poravnanja očitanja dobivenih *wgsimom* iznosio je 80366. Nakon provedenog prebrojavanja pristupilo se pronalasku točnih poravnanja korištenjem razvijenog programa. Kako bi se poravnanje smatralo točnim potrebno je barem 30 poravnatih nukleotida. Za različite postotke najboljih poravnanja promatrana je preciznost i odziv. Svi pokusi su povedeni s verzijom programa napisanog u Java programskom jeziku. Potrebno je naglasiti da reversni komplementi očitanja nisu uzeti u obzir. Također, korišten je linearni model bodovanja podudaranja. Rezultati poravnanja prikazani su u tablici 6.1

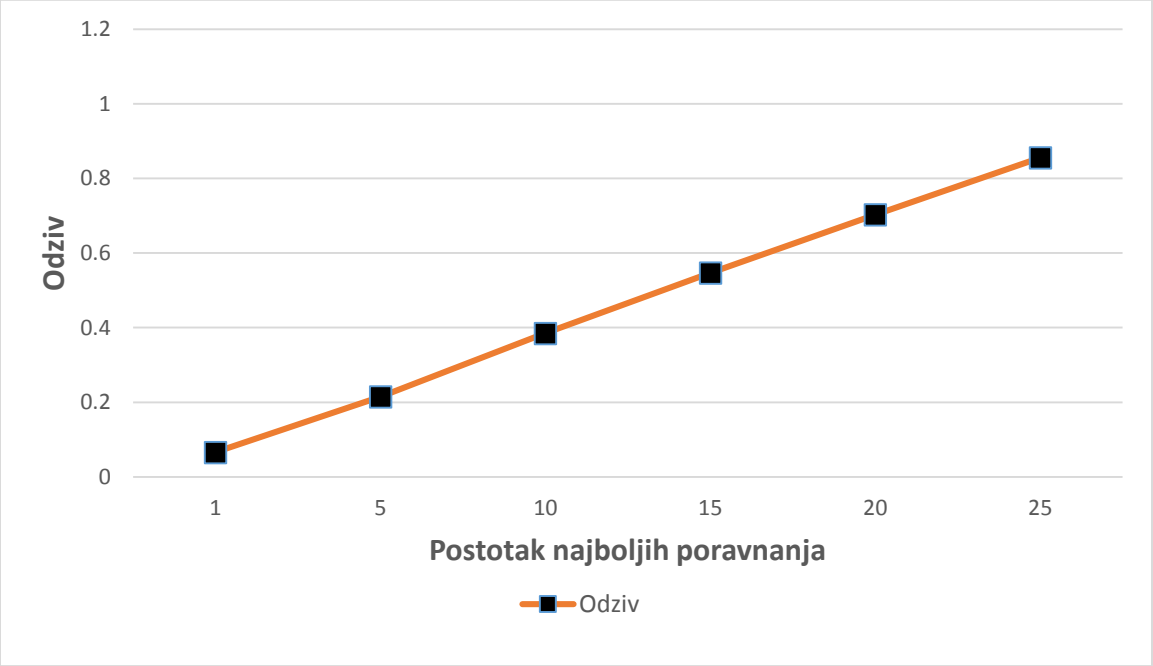
Postotak najboljih poravnanja	Broj ispravnih poravnanja	Broj neispravnih poravnanja	Preciznost	Odziv
1%	5355	4635	0.536	0.066
5%	17319	32631	0.346	0.215
10%	31001	68899	0.310	0.385
15%	43991	105859	0.415	0.547
20%	56540	143260	0.282	0.703
25%	68808	180942	0.275	0.856

Tablica 6.1: Prikaz rezultata poravnanja

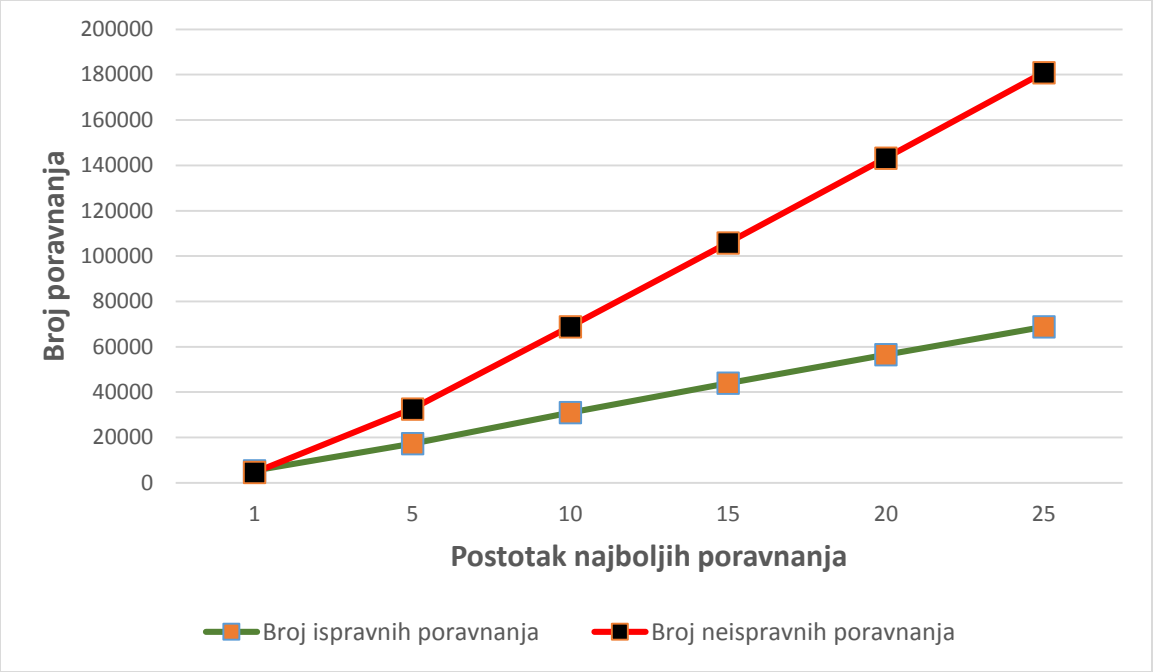
Grafom prikazanim na slici 6.2 dana je u uvid ovisnost preciznosti o postotku najboljih poravnanja. Prikaz odziva u ovisnosti o postotku najboljih poravnanja dan je na slici 6.3.



Slika 6.2: Prikaz preciznosti u ovisnosti o postotku najboljih poravnanja



Slika 6.3: Prikaz odziva u ovisnosti o postotku najboljih poravnanja



Slika 6.4: Prikaz ispravnih i neispravnih poravnanja u ovisnosti o postotku najboljih poravnanja

Sva testiranja provedena su s verzijom programa napisanog u Java programskom jeziku. Ukupno vrijeme izvođenja iznosilo je 181296. Zauzeće memorije za poravnanje svih nizova iznosilo je približno 742 megabajta.

7. Zaključak

Radom su obrađeni neki od osnovnih problema koji se pojavljuju prilikom sastavljanja genoma. Također razmatrana je mogućnost primjene algoritma semi-globalnog poravnanja prilikom sastavljanja genoma. Iz priloženih rezultata možemo zaključiti da algoritam daje zadovoljavajuće rezultate točnih poravnanja. Međutim, vrijeme izvođenja ovog algoritma za ovaj problem je poprilično veliko. Iako je originalni dio genom sadržavao otprilike 10^4 nukleotida, simuliranjem očitanja dobiveno je 1000 očitanja prosječne duljine 100 nukleotida. Za ovako simulirana očitanja prosječno vrijeme izvođenja iznosilo je 237134 ms što dovodi do zaključka da ovaj algoritam nije primjenjiv za veći broj očitanja. Također iz priloženih rezultata moguće je zaključiti da uspješnost ovog algoritma podosta ovisi i o modelu bodovanja podudaranja nukleotida. Tako je za 1% najboljih rezultata po linearnom modelu bodovanja dobiveno približno pola neispravnih poravnanja.

8. Literatura

[1] Mile Šikić, Mirjana Domazet–Lošo, Predavanja iz bioinformatike, 2013./2014., Zagreb

[2] Neil C. Jones, Pavel A. Pevzner, An introduction to bioinformatics algorithms, The MIT Press