SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DOKUMENTACIJA

Pronalaženje varijanti gena iz podataka dobivenih sekvenciranjem

Lovre Antonio Budimir, Dino Ehman, Luka Justić

Voditelj: *Doc. dr. sc. Krešimir Križanović*

Zagreb, siječanj 2020 .

Sadržaj

1. Uvod

U prirodi se geni javljaju u više varijacija koje nose genetsku informaciju za istu osobinu. Te varijacije gena nazivamo aleli. Aleli nastaju kroz mutacije, a jedinka od svakog roditelja nasljeđuje po jedan alel. Cilj ovog seminara bio je istražiti, implementirati i testirati različite metode za grupiranje gena kako bi se pronašle varijante istog gena prisutne u uzorku. Dio algoritama implementiranih u ovome radu kao i pristup problemu inspirirani su završnim radom Sanje Kosier [1] te je cilj bio pokušati postići točnije rezultate referencirajući se pritom na njene konačne rezultate. Za ulazne podatke dobiveni su sekvencirani uzorci nekoliko varijanti istih gena. Sekvenciranje je naziv za proces kojim se određuje redoslijed nukleinskih baza u DNA lancu. Postoje različite metode sekvenciranja, ali niti jedna metoda ne proizvodi u potpunosti točne rezultate. Većina metoda se zasniva na lomljenju dugih lanaca DNA i provođenjem sekvenciranja nad manjim fragmentima te naknadnim spajanjem rezultata. Zbog takvog pristupa česte su pogreške prilikom očitanja i metode većinom ne proizvode u potpunosti točne rezultate.

U radu je predstavljeno 5 algoritama grupiranja gena koji na različite načine određuju sličnost između očitanja. Objašnjeni su koraci algoritama i njihov rad na jednostavnom primjeru. Za svaki algoritam su provedena ispitivanja točnosti na ispitnim podacima te su uspoređene njihove točnosti i vremena trajanja. Na kraju je dan zaključak rada i ideje za daljnje unapređenje algoritama.

2. Podatci

Ulazni podaci su sekvencirani uzorci zapisani u FASTQ formatu i predstavljaju očitanja za uzorak divokoza i jelena. Uzorci su dobiveni metodom Ion Torrent. FASTQ format je format baziran na tekstu koji služi za pohranu sekvence i njezine kvalitete očitanja. FASTQ datoteka uobičajeno koristi četiri linije po uzorku:

- Linija 1 počinje sa znakom '@' i zatim slijedi identifikator sekvence te opis koji je opcionalan
- Linija 2 je sekvenca nukleotidnih baza
- Linija 3 počinje sa znakom '+' i zatim opcionalno slijedi ponovo identifikator sekvence i opis
- Linija 4 kodira ocjenu kvalitete za sekvencu u liniji 2 i mora sadržavati jednaki broj simbola kao sekvenca

Primjer FASTQ datoteke je sljedeći:

```
@16WBS:00025:00006
GATCCTCTCTCTGCAGCACATTTTCCTCT
+
--;;76CCDDCCCCCC@@@....(/+.--
```

Uz ulazne podatke imamo na raspolaganju dvije datoteke već pronađenih varijanti gena koje nam služe za testiranje. Izlazni podatci su skup otkrivenih varijanti gena u FASTA formatu te popis očitanja koja pripadaju pojedinoj varijanti gena.

3. Metode za grupiranje

Prilikom razvoja metoda za grupiranje počeli smo od metoda koje su opisane u radu Sanje Kosier [1]. To su metode velikog kažnjavanja i gruba metoda. Ideja je bila implementirati te metode, upoznati se s radom biblioteke SPOA [9] i reproducirati rezultate te sa prikupljenim znanjem početi graditi složenije algoritme koji nose nazive metoda A i metoda B.

3.1 Metoda velikog kažnjavanja

Koraci metode velikog kažnjavanja [1]:

1. Filtriranje očitanja po duljini.
2. Generiranje višestrukog poravnanja sekvenci za sva očitanja.
3. Grupiranje sličnih sekvenci u ovisnosti o parametru k u grupe.
4. Filtriranje grupa po veličini.
5. Određivanje konsenzusnih sekvenci za najveće grupe.

Prvi korak metode je filtriranje očitanja po duljini gdje smo pronašli koja je najčešća duljina sekvenci te odbacili sve ostale sekvence.

Sljedeći korak je izgradnja POA grafa koristeći biblioteku SPOA (https://github.com/rvaser/spoa). Metode unutar biblioteke SPOA nam omogućuju izgradnju poravnatih sekvenci te u konačnici konsenzus za svaku grupu koji nam predstavlja pronađeni alel. Cilj metode velikog kažnjavanja je oštro kazniti ubacivanje ili brisanje nukleotidnih baza. Parametri korišteni za izgradnju grafa i poravnatih sekvenci su sljedeći, 0 za podudaranje, -1 za zamjenu, -100 za umetanje ili brisanje te globalno poravnanje.

Treći korak bilo je grupiranje sličnih sekvenci u grupe. Ako je određena sekvenca različita od svih sekvenci neke grupe za manje od parametra k, ona se ubacuje u tu grupu. U slučaju da sekvenca ne zadovoljava uvjete niti jedne grupe, ona formira novi vlastiti grupu.

Zadnji korak je određivanje centroida/konsenzusa za grupe koje imaju određeni broj sekvenci. Za određivanje konsenzusa koristimo funkciju iz biblioteke SPOA.

3.1.1 Rezultati

Za datoteku J29_B_CE_IonXpress_005, metoda velikog kažnjavanja pronalazi dva alela u potpunosti, ali ne uspijeva pronaći treći očekivani alel. Korišteni parametri su k = 10 za udaljenosti sekvenca za grupe i s = 5 kao minimalni broj sekvenci potrebnih u grupi.

Što se tiče datoteke J30_B_CE_IonXpress_006, metoda velikog kažnjavanja pronalazi prvi alel u potpunosti, drugi alel uz potrebnu jednu zamjenu, te ne pronalazi treći očekivani alel. Korišteni parametri su k = 10 za udaljenosti sekvenca za grupe te s = 10 kao minimalni broj sekvenci potrebnih u grupi.

Izlaz za uzorak J29_B_CE_IonXpress_005:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATACTACGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTTCTTGACAGTTTCGCTGTGCAGCGGCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGGTGGCCGAGTACCTGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACACGTACTGCAGACACAACTACGGCGGCGTTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

Izlaz za uzorak J30_B_CE_IonXpress_006:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCTTAAGGCCGAGTGTCATTTCTTCAACGGGACGGAGCGGATG
CAGTTCCTGGCGAGATACTTCTATAACGGAGAAGAGTACGCGCGCTTCGACAGCGACGTGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCAAGTACTGGAACAGCCAGAAGGAGATCCTGGAGCAGCACGGGGCAGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTCGGTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
CGGTTCCTGGACAGATACTTCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGATTTCATGGAGCAGAAGCGGGCCGAG
GTGGACACGGTGTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

3.2 Algoritam grube metode

Koraci metode velikog kažnjavanja [1]:

1. Filtriranje očitanja po duljini.
2. Generiranje konsenzusne sekvence i višestrukog poravnanja sekvenci za sva očitanja.
3. Grupiranje sličnih sekvenci u ovisnosti o parametru k u grupe.
4. Određivanje konsenzusnih sekvenci za najveće grupe.

Algoritam grube metode sličan je algoritmu velikog kažnjavanja, ali za razliku od njega dopušta rad s nizovima različitih duljina. Zbog toga smo prilikom filtriranja nizova po duljini ovoga puta dopuštali i nizove koji odstupaju za do +/- *n* od duljine najučestalijeg niza. Nakon filtriranja slijedi korak izgradnje poravnatih nizova i oni se kao i u velikom kažnjavanju izgrađuju korištenjem SPOA knjižnice. U gruboj metodi parametri korišteni za izgradnju grafa i poravnatih sekvenci su sljedeći, 0 za podudaranje, -1 za zamjenu, -1 za umetanje ili brisanje te globalno poravnanje.

Ključni dio algoritma, grupiranje, ostvareno je također na sličan način. Prolazi se kroz filtrirana poravnata očitanja i za svaku se grupu uspoređuje to očitanje sa svim očitanjima u toj grupi te ako je očitanje udaljeno za manje od vrijednosti praga *k* od svih elemenata u grupi, onda se to očitanje dodaje u grupu. Razlika u odnosu na metodu velikog kažnjavanja je u tome što se u grupu ne dodaju poravnate sekvence već se dodaju originalne.

Nakon grupiranja sekvenci, grupe koje imaju manje od s elemenata odbacuju se. Za preostale grupe se od originalni sekvenci uz pomoć SPOA-e grade konsenzusi grupa.

3.2.1 Rezultati

Kao ni metoda velikog kažnjavanja, gruba metoda ne uspijeva pronaći treći alel za uzorak J29_B_CE_IonXpress_005. Uz parametre *n* = 5 , *k* = 15 i *s* = 80 Preostala dva alela pronalazi u potpunosti. Na uzorku J30_B_CE_IonXpress_006 gruba metoda pokazala se uspješnijom. Prvi alel pronašla je u potpunosti, a za drugi

alel joj je potrebna jedna zamjena. Najveći uspjeh je pronalazak trećeg alela u J30 skupu sa samo 1 zamjenom, 2 brisanja, i 1 umetanjem.

Izlaz za uzorak J29_B_CE_IonXpress_005:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGGTGGCCGAGTACCTGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACACGTACTGCAGACACAACTACGGCGGCGTTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (298)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATACTACGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTTCTTGACAGTTTCGCTGGTGCAGCGGTCGAGGTGACGCGAA
```

Izlaz za uzorak J30_B_CE_IonXpress_006:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCTTAAGGCCGAGTGTCATTTCTTCAACGGGACGGAGCGGATG
CAGTTCCTGGCGAGATACTTCTATAACGGAGAAGAGTACGCGCGCTTCGACAGCGACGTGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCAAGTACTGGAACAGCCAGAAGGAGATCCTGGAGCAGCACGGGGCAGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTCGGTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (297)
GGATCCTCTCTCTGCAGCACATTTCCTGGAGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGT
GCGGTTCCTGGACAGATACTTCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGC
GGTGACCGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGATTTCATGGAGCAGAAGCGGGCCGA
GGTGGACACGGTGTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (295)
GATCCTCTCTCTGCAGCACATTTCCTGATGTATACTAAGAAAGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGCTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCGAGGCTGGAACAGACAGAAGGAGCTCCTGGAGCAGAGGCGGGCCGCGG
TGGACACGTACTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

3.3 Algoritam metode A

Glavna ideja prve originalne metode u ovom radu je pronalazak poravnatih sekvenci koje nisu ni blizu slične i postaviti ih za centroide grupa te na temelju tih centroida kreirati grupe.

Koraci metode A:

1. Filtriranje očitanja po duljini.
2. Generiranje višestrukog poravnanja sekvenci.
3. Odabir poravnatih sekvenci koje će predstavljati početne centroide u grupama.
4. Stvaranje grupa na temelju udaljenosti poravnatih sekvenci od centroida.
5. Filtriraj grupe po broju poravnatih sekvenci koji se nalaze u grupi.
6. Poravnate sekvence u grupama pretvori nazad u očitanja.
7. Generiraj POA graf za svaku grupu i odredi konsenzusne sekvence za svaku grupu.

Metoda A također dopušta rad s različitim duljinama očitanja i zbog toga prva dva koraka ovog algoritma rade na isti način kao u algoritmu grube metode.

Nakon što dobijemo višestruko poravnate sekvence koristeći biblioteku SPOA tražimo elemente koji će predstavljati početne centroide u grupama. Prvu poravnatu sekvencu postavljamo za prvi centroid. Zatim za centroid postavljamo svaku sljedeću poravnatu sekvencu čija je Hammingova udaljenost veća od zadane vrijednosti parametra *c* od svih ostalih već određenih centroida.

Nakon što smo odredili početne centroide krećemo sa stvaranjem grupa. Svaka poravnatu sekvenca čija je Hammingova udaljenost manja od zadane vrijednosti parametra *g* dodaje se u grupu tog centroida. Bitan podatak je to što jedna poravnata sekvenca može pripadati više grupa odjednom.

Peti korak je filtriranje grupa na temelju njihovih veličina. Na kraju sve poravnate sekvence koje se nalaze u grupama pretvaramo u početna očitanja i iz tih očitanja za svaku grupu stvaramo konsenzusne sekvence koji predstavljaju pronađene alele.

U metodi A parametri korišteni za izgradnju grafa i poravnatih sekvenci su sljedeći, 0 za podudaranje, -1 za zamjenu, -1 za umetanje ili brisanje te globalno poravnanje.

3.3.1 Rezultati

Metoda A za očitanja J29_B_CE_IonXpress_005 pronalazi prva dva alela u potpunosti dok treći ne uspijeva pronaći. Za J30_B_CE_IonXpress_006 prvi alel pronalazi u potpunosti, drugi uz 1 zamjenu, a treći uz 1 umetanje, 1 zamjenu i 2 brisanja. Ovi rezultati dobiveni su koristeći parametre n = 5, c = 30, g = 15 i s = 5.

Izlaz za uzorak J29_B_CE_IonXpress_005:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGGTGGCCGAGTACCTGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACACGTACTGCAGACACAACTACGGCGGCGTTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATACTACGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTTCTTGACAGTTTCGCTGTGCAGCGGCGAGGTGACGCGAA
```

Izlaz za uzorak J30_B_CE_IonXpress_006:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCTTAAGGCCGAGTGTCATTTCTTCAACGGGACGGAGCGGATG
CAGTTCCTGGCGAGATACTTCTATAACGGAGAAGAGTACGCGCGCTTCGACAGCGACGTGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCAAGTACTGGAACAGCCAGAAGGAGATCCTGGAGCAGCACGGGGCAGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTCGGTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (297)
GGATCCTCTCTCTGCAGCACATTTCCTGGAGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGT
GCGGTTCCTGGACAGATACTTCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGC
GGTGACCGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGATTTCATGGAGCAGAAGCGGGCCGA
GGTGGACACGGTGTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (295)
GATCCTCTCTCTGCAGCACATTTCCTGATGTATACTAAGAAAGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGCTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCGAGGCTGGAACAGACAGAAGGAGCTCCTGGAGCAGAGGCGGGCCGCGG
TGGACACGTACTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

3.4 Algoritam metode B

Implementacija metode B zamišljena je kao nadogradnja metode A, ali uzima u obzir činjenicu da pronađeni konsenzusi mogu pokazivati na isti alel. Zbog toga se provodi dodatni korak grupiranja svih grupa čiji su konsenzusi slični. Ovaj algoritam možemo smatrati najuspješnijim jer uspijeva pronaći u potpunosti treći alel iz uzorka J29_B_CE_IonXpress_005.

Koraci metode B:

1. Filtriranje očitanja po duljini.
2. Generiranje višestrukog poravnanja sekvenci za sva očitanja.
3. Odabir poravnatih sekvenci koje će predstavljati početne centroide u grupama.
4. Stvaranje grupa na temelju udaljenosti poravnatih sekvenci od centroida.
5. Poravnate sekvence u grupama pretvori nazad u očitanja.
6. Generiraj POA graf za svaku grupu i odredi konsenzusne sekvence za svaku grupu
7. Sve grupe čiji su konsenzusne sekvence (centroidi) dovoljno blizu spoji u jedan veliku grupu.
8. Ponovi korake 5. i 6.

Prvih 5. koraka identični su koracima koji su korišteni u metodi A, ali nakon što dobijemo konsenzusne sekvence ovaj put ih ne zapisujemo kao rezultate, nego provjeravamo koliko su ti međusobni konsenzusni slični. Sve grupe čiji konsenzusi imaju manju Hammingovu udaljenost od postavljene vrijednosti *m* spajaju se u jednu veliku grupu. Ovaj postupak se pokazao izuzetno korisnim za pronalaženje trećeg alela iz uzorka J29_B_CE_IonXpress_005. Razlog tome je što veliki broj konsenzusnih sekvenci dobivenih nakon prvih 6. koraka je jako slično, to jest rade predikciju istog alela iz uzorka.

U metodi B najuspješniji parametri korišteni za izgradnju grafa i poravnatih sekvenci su sljedeći, 1 za podudaranje, -1 za zamjenu, -1 za umetanje ili brisanje te poluglobalno poravnanje. Uspješni rezultati se mogu dobiti i globalnim poravnanjem.

3.4.1 Rezultati

Metoda B za očitanja J29_B_CE_IonXpress_005 pronalazi sva tri alela u potpunosti. Za J30_B_CE_IonXpress_006 prvi i drugi alel pronalazi u potpunosti , a treći kao i kod ostalih metoda uz 1 umetanje, 1 zamjenu i 2 brisanja. Ovi rezultati dobiveni su koristeći parametre *n* = 5, *c* = 30, *g* = 29, *m*=17 i *s* = 6. Rezultati koja je ostvarila metoda pokazali su se najboljima, pogotovo ako uzmemo u obzir da je u potpunosti našla 3 alel iz uzorka J29.

Izlaz za uzorak J29_B_CE_IonXpress_005:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGGTGGCCGAGTACCTGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACACGTACTGCAGACACAACTACGGCGGCGTTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (298)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATACTACGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTTCTTGACAGTTTCGCTGGTGCAGCGGTCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCATAAGTGCGAGTGTCATTTCTCCAACGGGACGGAGCGGGTG
CAGTTCCTGCAGAGATACATCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGTCCGCCAAGTACTATAACAGCCAGAAGGAGCTCCTGGAGCAGAAGCGGGCCGCG
GTGGACAGGTACTGCAGACACAACTACGGGGTCGTTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

Izlaz za uzorak J30_B_CE_IonXpress_006:

```
Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCTTAAGGCCGAGTGTCATTTCTTCAACGGGACGGAGCGGATG
CAGTTCCTGGCGAGATACTTCTATAACGGAGAAGAGTACGCGCGCTTCGACAGCGACGTGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCAAGTACTGGAACAGCCAGAAGGAGATCCTGGAGCAGCACCGGGCAGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTCGGTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
CGGTTCCTGGACAGATACTTCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGATTTCATGGAGCAGAAGCGGGCCGAG
GTGGACACGGTGTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (295)
GATCCTCTCTCTGCAGCACATTTCCTGATGTATACTAAGAAAGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGCTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCGAGGCTGGAACAGACAGAAGGAGCTCCTGGAGCAGAGGCGGGCCGCGG
TGGACACGTACTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

4. Alternativne metode

Sljedeće dvije opisane metode koriste korake pretprocesiranja podataka, kao i konačni izračun centroida pojedine grupe identične prethodno opisanim metodama. Ipak sami algoritmi grupiranja kao i izračunavanje udaljenosti specifični su za svaki algoritam.

4.1 Grupiranje algoritmom K-sredina

Algoritam K-sredina je poznati i često korišteni algoritam za grupiranje iz područja strojnog učenja. K-sredina grupira vektore u K grupa pritom uspoređujući pojedini vektor s srednjim vrijednostima do sada pronađenih grupa. Nakon izračuna udaljenosti, vektor se dodjeljuje grupi koja mu je po udaljenosti najbliža te se iznova računa vektor srednjih vrijednosti te grupe, ovog puta uzimajući u izračun i novo pridodani vektor.

Prvi korak algoritma bilo je učitavanje podataka iz FASTQ datoteke te pronalazak sekvenci jednakih duljina, a čije su duljine pritom najzastupljenije u očitanjima. Učitane sekvence se zatim poravnavaju uporabom SPOA knjižnice. Pomoću poravnatih sekvenci se određuju početni centroidi grupa koristeći već spomenutu metodu koja inicijalizira početne te se na taj način određuje broj grupa. Poravnate sekvence i broj grupa zapisuju se novu datoteku koja se koristi u nastavku algoritma.

Glavni dio algoritma, grupiranje, ostvareno je pomoću pythona i scikit-learn knjižnice koja pruža implementaciju algoritma k-sredina. Algoritam k-sredina koristi euklidsku udaljenost kao funkciju udaljenosti i zbog toga je bilo potrebno alel koji je zapisan kao niz slova kodirati kao vektor brojeva. To je postignuto tako da je slovo A kodirano kao broj 1, T kao broj 2, C kao broj 3 i G kao broj 4, a tu je također i znak ubacivanja/brisanja “-” koji je kodiran s nulom. Na tako kodirane vektore primjenjen je algoritam k-sredina koji je grupirao vektore u grupe. Tako grupirani vektori ponovno su pretvoreni u nizove slova i zapisani u novi dokument, ovoga puta zajedno s oznakom grupe.

Posljednji korak algoritma je izračun konsenzusa za svaku od grupa. Grupirani geni učitani su u C++ program i za svaku grupu se ponovno pomoću uporabe SPOA knjižnice izračunao konsenzus te grupe.

4.1.1 Rezultati

Algoritam k-sredina uspješno pronalazi po dvije grupe alela za svako od očitanja, J29 i J30, ali niti kod jednog uzorka ne pronalazi treći alel. Kod J29 nalazi prvi uz 1 umetanje i drugi uz 1 umetanje. Za J30 prvi pronalazi u potpunosti,a za drugi mu je potrebna 1 zamjena.

J29_B_CE_IonXpress_005

```
Alel (297)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGGTGGCCGAGTACCTGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACACGTACTGCAGACACAACTACGGCGGCGTTGAGAGTTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (298)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATACTACGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTTCTTGACAGTTTCGCTGGTGCAGCGGTCGAGGTGACGCGAA
```

J30_B_CE_IonXpress_006

```
Alel (297)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCTTAAGGCCGAGTGTCATTTCTTCAACGGGACGGAGCGGATG
CAGTTCCTGGCGAGATACTTCTATAACGGAGAAGAGTACGCGCGCTTCGACAGCGACGTGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGGCGGCCGGACGCCAAGTACTGGAACAGCCAGAAGGAGATCCTGGAGCAGCACGGGGCAGA
GGTGGACAGGTACTGCAGACACAACTACGGGGTCGGTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGGAGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
CGGTTCCTGGACAGATACTTCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGATTTCATGGAGCAGAAGCGGGCCGAG
GTGGACACGGTGTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

4.2 Grupiranje algoritmom DBSCAN

Sljedeća metoda grupiranja temeljena je na algoritmima koji grupiraju elemente na temelju gustoće. Jedan takav algoritam je DBSCAN - Density-based spatial clustering of applications with noise [2]. Ideja algoritma je grupirati elemente koji su blizu jedni drugima do određene granice. Za razliku od k-sredina koji za svaki element traži grupu u koju da ga smjesti, koristeći pritom udaljenost od te grupe, dbscan neće element stavljati u grupu ako se ne nalazi unutar određenog radiusa od središnje točke. Zbog toga se dbscan algoritam bolje nosi sa stršećim vrijednostima i za razliku od k-means algoritma nije potrebno unaprijed zadati broj grupa. Hiperparametri dbscan algoritma su radius i minimalni broj elemenata u susjedstvu. Algoritam radi na principu dodjeljivanja uloga elementima. Ako se oko elementa unutar definiranog radiusa nalazi određeni minimalni broj elemenata onda se taj element smatra središnjim elementom. Ako element nema minimalni broj, ali se nalazi u susjedstvu neke od središnjih točaka onda taj element pripada grupi u kojoj je i središnja točka. Elementi koji ne zadovoljavaju te uvjete smatraju se stršećim vrijednostima i ne pokušavaju dodjeliti niti jednoj grupi.

Algoritam grupiranja gena ponovno se sastoji od 3 djela. U prvome djelu učitavamo podatke iz FASTQ datoteka i filtriramo ih po duljini. Ovoga puta nije potrebno podatke poravnavati niti određivati broj grupa već se samo filtrirani geni zapisuju u novu datoteku.

U drugom i središnjem djelu algoritma, filtrirani podatci učitavaju se u python program koji pomoću dbscan algoritma implementiranog u scikit-learn knjižnici vrši grupiranje. Specifičnost ovog algoritma je i funkcija udaljenosti. Za razliku od euklidske udaljenosti koja se koristila u k-sredina algoritmu, dbscan-u je zadano da koristi Levenshtein-ovu udaljenost prilikom usporedbe elemenata. Levenshtein-ova udaljenost mjeri koliko je operacija ubacivanja, brisanja i zamjene potrebno kako bi se poravnala dva niza znakova. Rezultati dbscan grupiranja upisani su u novu datoteku zajedno s oznakama grupa i proslijeđeni na obradu.

U završnom djelu algoritma, grupirani podatci učitani su u c++ program gdje je za svaku grupu izračunat konsenzus koristeći SPOA knjižnicu.

4.2.1 Rezultati

Za izračun točnosti algoritma ponovno su korištena J29 i J30 očitanja. Za J29 očitanje i hiperparametre algoritma - radius 6 i minimalni broj elemenata u susjedstvu 3 algoritam pronalazi sva tri tražena alela. Prvi alel pronađen je u potpunosti dok je za drugi bila potrebna 1 zamjena kako bi se u potpunosti poravnali. Treći alel bilo je najteže pronaći, ali pronađen je nakon što je napravljeno 1 brisanje. To su znatno bolji rezultati od k-sredina algoritma.

Za J30 rezultati su bili nešto slabiji. Korišteni parametri su radius 6 i minimalni broj elemenata u susjedstvu 10. Prvi alel ponovno je pronađen u potpunosti, a za drugi alel bila je potrebna samo 1 zamjena. Treći je pronađen također, ali tek nakon 1 umetanja, 1 zamjene i 2 brisanja.

J29_B_CE_IonXpress_005

```
Alel (298)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATACTACGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTTCTTGACAGTTTCGCTGGTGCAGCGGTCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGCTGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGTTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGGCCGGTGGCCGAGTACCTGAACAGCCAGAAGGAGTACATGGAGCAGACGCGGGCCGAG
GTGGACACGTACTGCAGACACAACTACGGCGGCGTTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (295)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCATAAGTGCGAGTGTCATTTCTCCAACGGGACGGAGCGGGTG
CAGTTCCTGCAGAGATACATCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTACCGGGCG
GTGACAGAGCTGGGGCGCCGTCCGCCAAGTACTATAACAGCCAGAAGGAGCTCCTGGAGCAGAAGCGGGCCGCGG
TGGACAGGTACTGCAGACACAACTACGGGGTCGTTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA
```

J30_B_CE_IonXpress_006

Alel (297)
GATCCTCTCTCTGCAGCACATTTCCTGGAGCATCTTAAGGCCGAGTGTCATTTCTTCAACGGGACGGAGCGGATG
CAGTTCCTGGCGAGATACTTCTATAACGGAGAAGAGTACGCGCGCTTCGACAGCGACGTGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCAAGTACTGGAACAGCCAGAAGGAGATCCTGGAGCAGCACGGGGCAGAG
GTGGACAGGTACTGCAGACACAACTACGGGGTCGGTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (296)
GATCCTCTCTCTGCAGCACATTTCCTGATGTATACTAAGAAAGAGTGTCATTTCTCCAACGGGACGCAGCGGGTG
GGGCTCCTGGACAGATACTTCTATAACGGAGAAGAGTTCGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGCG
GTGACCGAGCTGGGGCGGCCGGACGCCGAGGCTGGAACAGACAGAAGGAGCTCCTGGAGCAGAGGCGGGCCGCGG
TGGACACGTACTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

Alel (298)
GGATCCTCTCTCTGCAGCACATTTCCTGGAGTATGCTAAGAGCGAGTGTCATTTCTCCAACGGGACGCAGCGGGT
GCGGTTCCTGGACAGATACTTCTATAACCGGGAAGAGTACGTGCGCTTCGACAGCGACTGGGGCGAGTTCCGGGC
GGTGACCGAGCTGGGGCGGCCGTCCGCCAAGTACTGGAACAGCCAGAAGGATTTCATGGAGCAGAAGCGGGCCGA
GGTGGACACGGTGTGCAGACACAACTACGGGGTTATTGAGAGTTTCACTGTGCAGCGGCGAGGTGACGCGAA

5. Zaključak

U ovome radu komentirali smo problem grupiranja sekvenci. Predstavili smo više različitih algoritama za grupiranje i demonstrirali njihov rad. Pokazali smo rezultate rada algoritama na dva reprezentativna primjerka i usporedili njihove točnosti. Kako rad algoritama uvelike ovisi o ispravno odabranim hiper-parametrima veće točnosti mogle bi se postići uz dugotrajnije proučavanje i isprobavanje hiper-parametara. Također dostupne su još brojne druge metode grupiranja nizova tekstova iz područja prirodne obrade jezika koje u ovome radu nisu korištene. Ipak metode koje su prikazane uspjele su postići zadovoljavajuće rezultate, posebice metode DBSCAN i metoda B.

6. Literatura

[1] Kosier, Sanja; Prof. dr. sc. Šikić, Mile: "Završni rad: Pronalaženje varijanti gena iz podataka dobivenih sekvenciranjem", Fakultet elektrotehnike i računarstva, Zagreb, 2019.

[2] Wikipedia. DBSCAN. URL: https://en.wikipedia.org/wiki/DBSCAN (14.1.2020)

[3] Wikipedia. K-means clustering. URL: https://en.wikipedia.org/wiki/K-means_clustering (14.1.2020)

[4] Python scikit learn documentation. Sklearn.cluster.DBSCAN. URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

[5] Python scikit learn documentation. Sklearn.cluster.K-means. URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

[6] Wikipedia. FASTQ format. URL: https://en.wikipedia.org/wiki/FASTQ_format (14.1.2020)

[7] Wikipedia. FASTA format. URL: https://en.wikipedia.org/wiki/FASTA_format (14.1.2020)

[8] Šikić M., Domazet-Lošo M., Bioinformatika: Skripta, Zagreb, prosinac 2013.

[9] Biblioteka SPOA. URL : https://github.com/rvaser/spoa