Bioinformatika 1
SCCG - visoko-učinkoviti referencijski algoritam za kompresiju genoma

Jan Celin, Mate Papak

Lipanj 2025.

1 Uvod

Naglim razvojem tehnologija za sekvenciranje genoma, kao i rastućim interesom za znanstveni rad u području genetike, znatno raste i količina podataka koje je potrebno pohraniti. Standardni kompresijski alati, kao što je, primjerice, 7-Zip, nisu optimizirani za sekvence genoma, pošto ne iskorištavaju njihova svojstva, poput malene abecede (A, T, C, G) i velikih sličnosti između genoma istih vrsta [1].

Neki od najčešćih tipova kompresije genoma su referencijski, koji koriste već poznati sekvencionirani genom (referencu) za kompresiju novog genoma. Na taj način nije potrebno pamtiti sve nukleotide, nego samo mjesta na kojima se ta dva genoma razlikuju.

Cilj našeg projekta bio je implementirati referencijski kompresijski algoritam opisan u radu *Shi et al. 2019. High efficiency referential genome compression algorithm.* Alat za kompresiju i dekompresiju razvijen je u programskom jeziku C++, slijedeći algoritam opisan u radu.

2 Opis algoritma

Algoritam kombinira lokalno i globalno pretraživanje podudaranja referentnog i ciljnog genoma, čime autori postižu visok postotak kompresije uz smanjenje brzine izvršavanja algoritma [**?**].

Kompresija se bazira na pronalasku identičnih podnizova između referentne (R) i ciljne (T) sekvence. U slučaju pronalaska takvog preklapanja, ono se pamti parom brojeva *(p, l)*, gdje je p početna pozicija podniza u referenci, a l njegova duljina. Dijelovi niza za koje preklapanje nije pronađeno zapisuju se kao izvorni niz znakova.

Posebnost algoritma implementiranog u radu je kombiniranje između dviju strategija pretraživanja:

1. Lokalno pretraživanje: Ova se strategija primjenjuje u slučaju visoke sličnosti genoma. U ovom se slučaju obje sekvence dijele na segmente jednake duljine te se pretraživanje podudaranja obavlja samo između odgovarajućih segmenata.

2. Globalno pretraživanje: Ako lokalno pretraživanje ne daje dobre rezultate, odnosno ako je postotak nepodudaranja znakova unutar segmenta veći od predodređenog praga *T1* kroz nekoliko uzastopnih segmenata *T2*, algoritam se prebacuje na globalno pretraživanje. Njime se za svaki podniz u ciljnoj sekvenci traži podudaranje unutar cijele referentne sekvence. Ovim se postupkom dobiva bolja kompresija, ali je resursno znatno zahtjevniji [1].

2.1 Koraci kompresije

Algoritam je podijeljen u nekoliko faza izvršavanja:

1. Pretprocesiranje: U memoriju se učitavaju referentna i cilja sekvenca. Sva se mala slova pretvaraju u velika te se u datoteci pamte njihove pozicije. U slučaju globalnog pretraživanja brišu se i svi nepoznati nukleotidi (znakovi "N"), također uz bilježenje njihovih pozicija.

2. Pretraživanje podudaranja: Ciljna sekvenca se uspoređuje s referentnom pomoću algoritma temeljenog na hashiranju k-mera. Svi k-meri (podnizovi duljine k) iz referentne sekvence pohranjuju se u hash tablicu radi brzog pretraživanja. Zatim se sekvenca cilja pretražuje pomičnim prozorom. Za svaki k-mer iz ciljne sekvence provjerava se postoji li podudaranje u referenci. Ako se ne pronađe podudaranje, trenutni znak se označava kao nepodudaranje i prelazi se na sljedeći. U suprotnom, pokušava se proširiti podudaranje s odgovarajućim dijelom reference dok se ne naiđe na razliku. U globalnom načinu rada dodatno se vodi računa o udaljenosti između uzastopnih podudaranja da budu što bliže.

3. Generiranje međudatoteke: nakon pronalaženja podudarajućih segmenata, rezultati se pišu u međudatoteku koja sadrži lokacije podudaranja u referentnom genomu (parove *(p, l)*) i znakove koji se nisu podudarali.

4. Delta kodiranje: S ciljem dodatnog smanjenja veličine datoteke, pozicije p se delta-kodiraju, čime se umjesto apsolutne pozicije u referentoj sekvenci zapisuje razlika u odnosu na poziciju prethodnog podudaranja.

5. Kompresija datoteke: Na kraju se cijela međudatoteka komprimira PPMd algoritmom alata 7-Zip.

Rezultat Kompresije je 7-Zip arhiva originalne sekvence.

2.1.1 Koraci dekodiranja

Dekodiranje datoteke je znatno brži i jednostavniji proces:

1. Dekompresija arhive: Konačna zip arhiva dobivena zadnjim korakom kompresije dekomprimira se alatom 7-Zip.

2. Rekonstrukcija: Dobivena se međudatoteka čita znak po znak; Kad se naiđe na par brojeva *p, l*, odgovarajući se podniz kopira iz referentne sekvence; Kad se naiđe na znak nukleotida, on se izravno dodaje u rekonstruiranu sekvencu.

3. Vraćanje obrisanih i izmijenjenih znakova: Znakovi koji su pretvoreni u velika slova te nepoznati znakovi ("N"), čije su pozicije zapisane u datoteku, vraćaju se u konačni rekonstruirani niz.

Rezultat dekodiranja jest rekonstruirana originalna sekvenca, bez gubitaka.

3 Rezultati

U originalnom radu [1] su omjeri dekompresije koristeći hg18 genom bili oko 0.395%.

Table 1: Vrijeme izvođenja kompresije i dekompresije (u sekundama) te omjer kompresije (u %) za različite ciljne 19. kromosome, pri korištenju referentnog 19. **hg18** kromosoma

Ciljni kromosom	**Vrijeme (kompresija + dekompresija)**	**Omjer kompresije (%)**
hg17	36.96 + 3.97	0.425
hg19	79.00 + 4.35	0.466
Mus musculus	129.75 + 1.89	21.279

Table 2: Vrijeme izvođenja kompresije i dekompresije (u sekundama) te omjer kompresije (u %) za ciljni 19. **hg18** kromosom, pri korištenju referentnog 19. **mišjeg** (Mus musculus) kromosoma

Ciljni kromosom	**Vrijeme (kompresija + dekompresija) (s)**	**Omjer kompresije (%)**
hg18	136.06 + 2.50	17.618

References

[1] W. Shi, J. Chen, M. Luo, and M. Chen. High efficiency referential genome compression algorithm. *Bioinformatics*, 35(12):2058–2065, 11 2018.