Projektni zadaci

Prijava projekta se može obaviti slanjem e-maila na vladakova@gmail.com. Obavezno je navesti ime, prezime i broj indeksa. Rok za izradu projekta je **mesec dana** od dana prijave. U tom roku je potrebno poslati sve tražene rezultate na gore navedeni e-mail. Projektni zadaci su predviđeni za maksimalno dva studenta, osim ako je drukčije naznačeno. Moguće je prijaviti samo zadatke koji nemaju ime uz sebe (one koji nisu rezervisani).

Za sve što nije egzaktno definisano tekstom projekta uvesti logičnu pretpostavku. Slajdovi, prezentacija i komentari u programskom kodu poželjno je da budu na engleskom, ali je i srpski jezik u potpunosti prihvatljiv. **Poželjno je da se u toku razvoja projekta sav programski kod čuva na Github-u ili drugom sistemu za verzionisanje koda.** Budite kreativni!

	Studenti	Poeni
Zadatak 1	Ivan Kukrkić 2021/3133, Uroš Stoković 2021/3137	
Zadatak 1		
Zadatak 2	Pavle Divović 2021/3112	
Zadatak 2		
Zadatak 3	Ana Vanzo 2021/3355	
Zadatak 3	Sara Živković 2021/3285, Božidar Obradović2021/3081	
Zadatak 4	Aleksandar Malović 2021/3375	Under review
Zadatak 4	Rastko Zlatkovic 3272/2020, Milos Gvozdenovic 3082/2021	
Zadatak 5	Sandra Velimirović 2021/3364, Petar Lozo 2021/3437	
Zadatak 5	Василије Бецић 3201/2020, Јелена Дубак 3256/2020	
Zadatak 6		
Zadatak 7		
Zadatak 8		
Zadatak 9		

Zadatak 1

Napisati u programskom jeziku Python implementaciju algoritma za difuziju verovatnoća po čvorovima grafa opisanog u radu CTD: An information-theoretic algorithm to interpret sets of metabolomic and transcriptomic perturbations in the context of graphical models, Tabela 2. Smatrati da je matrica susednosti data na ulazu kao Padas dataframe koji se učitava iz csv fajla i da je u pitanju matrica susednosti neusmerenog težinskog grafa.

- Algoritam implementirati rekurzivno (kao što je opisano u radu) i iterativno i porediti performanse u pogledu vremena izvršavanja i memorijskog zauzeća. (5 poena)
- Trenutno se u algoritmu za difuziju ½ verovatnoće rasipa na susede, a ½ ostaje u čvoru (linije 8, 9, 10 i 11 pseudokoda u navedenom radu). Proširiti algoritam tako da implementirana funkcija prima parametar alfa koji određuje koji se procenat verovatnoće prenosi na susede, npr. ako je alfa = 0.7, prenosi se 70% verovatnoće, a 30% ostaje u čvoru. (5 poena)
- Definisati set testova koji će meriti performanse i porediti identičnost rezultata rekurzivne i iterativne implementacije na raznovrsnom skupu grafova. Testovi treba da pokriju četiri kategorije grafova u pogledu broja čvorova: male (5-15 čvorova), srednje (15-100) čvorova, veće (100 - 1000) i velike (preko 1000 čvorova), svaku kategoriju sa bar 5 testova koji se suštinski razlikuju u topologiji grafa. U testovima se pokreće implementirana funkcija za difuziju verovatnoće, počevši od slučajno izabranog čvora grafa, sa početnom verovatnoćom 0.5. (5 poena)
- Kreirati vizuelizaciju za testove iz kategorije malih grafova u kojim će čvorovi grafa biti gradaciono obojeni po količini verovatnoće u njima na kraju difuzije, recimo kao što je urađeno ovde. (5 poena)

Pripremiti slajdove i video prezentaciju projekta (3 - 5 minuta trajanja) koja će biti dostupna na YouTube ili drugom video servisu na kojem mu je moguće pristupiti on-line (20 poena).
Neophodno je da se u toku razvoja projekta sav programski kod čuva na Github-u ili drugom sistemu za verzionisanje koda.

Zadatak 2

Implementirati u programskom jeziku Python algoritam za rangiranje čvorova grafa (*node rank*) opisanog u radu CTD: An information-theoretic algorithm to interpret sets of metabolomic and transcriptomic perturbations in the context of graphical models, Tabela 3. Ovaj algoritam se koristi da se, koristeći difuziju verovatnoće, odredi bitsko enkodiranje čvorova iz odabranog podskupa čvorova S.

- Implementaciju sprovesti tako da se umesto implementiranja funkcije `DIFFUSE_PROBABILITY` napiše i koristi *mock* funkcija koja simulira njen rad slučajnim generisanjem izlaza. Prikazati rad implementirane funkcionalnosti na barem 5 grafova (datim preko matrice susednosti) različite veličine i topologije, sa različitom veličinom

podskupa S,. Varirati veličinu podskupa S i dozvoljeni broj promašaja (`num_misses`). (5 poena)

- Umesto da se koristi *mock* implementacija funkcije `DIFFUSE_PROBABILITY`, izvršiti obmotavanje i pozivanje R funkcije `graph.diffuseP1` date ovde direktno iz Python koda. Ovo se može postići korišćenjem *robjects* modula iz *rpi2* biblioteke. Iz funkcije `graph.diffuseP1` treba ukloniti pozive funkcije `graph.diffusionSnapShot` koja se koristi za crtanje grafa. (5 poena)
- Ponoviti demonstraciju rada sprovedenu u prvoj tački. Dodatno, za različite grafove vizuelno prikazati zavisnost dužine najkraćeg bitskog enkodiranja generisanog implementiranom funkcijom od dozvoljenog broja promašaja (`num_misses`). (5 poena)
- Izmeriti i vizualizovati vremena izvršavanja algoritma nad grafovima različite veličine korištenim u prvom delu zadatka. (5 poena)
- Pripremiti slajdove i video prezentaciju projekta (3 - 5 minuta trajanja) koja će biti dostupna na YouTube ili drugom video servisu na kojem mu je moguće pristupiti on-line (20 poena).

Neophodno je da se u toku razvoja projekta sav programski kod čuva na Github-u ili drugom sistemu za verzionisanje koda.

Zadatak 3

Napisati u programskom jeziku Python simulator sekvencera koji kreira veštačke Illumina paired-end readove u FASTQ formatu uzimajući nukleotide iz referentnog genoma koji je zadat na ulazu. Kao ulazne parametre simulator treba da prima prosečan kvalitet nukleotida i pokrivenost (coverage), na osnovu kojih će normalnom raspodelom formirati kvalitete nukleotida u svakom od readova i sam potreban ukupan broj readova. Duzina readova i insert size treba da budu parametri simulatora, kao i error rate - verovatnoća pojavljivanja pogrešnih nukleotida u okviru samih readova i to posebno za snipove (single nucleotide variations), za insercije i delecije. Ukoliko ima potrebe za dodatnim parametrima simulatora moguće ih je dodati, uz priložen precizan opis. Pored dva FASTQ fajla, kao izlaz simulatora, formira se i SAM fajl koji sadrži alajnovane sve readove iz FASTQ fajlova zajedno sa pozicijama iz referentnog genoma sa kojih su ekstrahovani nukleotidi. Na taj način izlazni fajlovi se mogu koristiti za testiranje kvaliteta alata koji rade poravnavanje (e.g. BWA-MEM). (15 poena)
Potrebno je izvršiti testiranje kvaliteta alajmenta BWA-MEM i Bowtie alata davajući mu na ulaz FASTQ fajlove koje je napravio simulator tako što će se porediti njegov izlaz, pozicije alajmenta u SAM fajlu, sa pozicijama alajmenta u SAM fajlu simulatora. Rezultate testiranja prikazati grafički na intuitivno razumljiv način koristeći klasifikacione metrike (na primer precision, recall, fscore, AUC). Kao test podatke koristiti 3 genoma po izboru iz NIH baze (10 poena).
Napisati test slučajeve za testiranje funkcionalnosti samog simulatora na početku razvoja simulatora (5 poena).
Pripremiti slajdove i video prezentaciju projekta (3 - 5 minuta trajanja) koja će biti dostupna na YouTube ili drugom video servisu na kojem mu je moguće pristupiti on-line (10 poena).

Poželjno je da se u toku razvoja projekta sav programski kod čuva na Github-u ili drugom sistemu za verzionisanje koda.

Zadatak 4

Implementirati na programskom jeziku Python algoritam za indeksirano pretraživanje stringova u zatadom tekstu koristeći Burrows-Wheeler transformaciju i FM index. Inicijalna verzija algoritma treba da bude realizovana na tradicionalan način opisan na predavanju, bez optimizicije memorije i vremena izvršavanja (10 poena).
Za svaku od funkcija u kodu, kao i za sam finalni algoritam napisati testove (5 poena).
Izvršiti optimizaciju koda iz aspekta zauzeća memorije i vremena izvršavanja. Pokrenuti prethodno definisane testove i proveriti da li i dalje svi prolaze (regresiono testiranje). Izmeriti unapređenje zauzeća memorije i vremena izvršavanja koristeći kao test podatke 3 seta (10 poena):

1. Tekst: Coffea arabica, Chromosome 1c i paterni: ATGCATG, TCTCTCTA, TTCACTACTCTCA
2. Tekst: Mus pahari chromosome X, i paterni: ATGATG, CTCTCTA, TCACTACTCTCA
3. Genom po slobodnom izboru iz NIH baze i proizvoljna 3 paterna različite dužine.

Pripremiti prezentaciju (Google slides ili power point) inicijalnog i optimizovanog algoritma, kao i samih rezultata (5 poena).
Pripremiti video prezentaciju projekta (3 - 5 minuta trajanja) koja će biti dostupna na YouTube ili drugom on-line video servisu (10 poena).

Zadatak 5

Implementirati na programskom jeziku Python:

(1) Algoritam za indeksirano pretraživanje stringova u zadatom tekstu koristeći Burrows-Wheeler transformaciju i FM index. Inicijalna verzija algoritma treba da bude realizovana na tradicionalan način opisan na predavanju, bez optimizacije memorije i vremena izvršavanja (**10 poena**).
(2) Algoritam za globalno poravnavanje stringova (global alignment - Needleman-Wunsch) koji koristi skoring tabelu (scoring matrix) kao ulazni parametar komandne linije programa (**5 poena**).
(3) Program prima kao ulaz referencu (FASTA fajl) i kolekciju ridova (FASTQ fajl) kao argumente komandne linije i za svaki od ridova vraća listu torki (pozicija gde se rid mapira na referencu, ocena optimalnog poravnanja na toj poziciji i edit transkript) - liste treba sortirati po oceni od najveće ka najmanjoj. Koristiti “Seed & Extend” metod gde se seed uzima sa početka rida (dužina seed-a je ulazni parametar), za lociranje pozicije seed-a u referenci koristiti algoritam iz tačke (1). Zatim, preostali deo rida treba poravnati na odgovarajući deo reference iza lociranog seed-a koristeći algoritam iz tačke (2). Deo

reference na koji se preostali deo rida poravnava treba biti duži od preostalog dela rida za 0 - 3 nukleotide (ulazni parametar “margin”). Za svaki od ridova izvrsiti pretraživanje i za njegov reverse komplement, jer je moguće da potiče sa backward strenda referentnog genoma. (**5 poena**).

(4) Uzeti za dužinu seed-a 10 nukleotida, “margin” parametar 2, vrednost poklapanja (match) [0, 1, 2], mutacije (mismatch) [-3, -2] i insercije ili delecije (gap) [-7, -5]. Dati dijagrame poređenja dobijenih najboljih alajnmenta za svaki rid u odnosu na rezultate dobijene BWA-MEM alatom. Pozicije najboljih alajnmenta BWA-MEM-a biće dostupne u njegovom izlaznom SAM fajlu. Testirati za svih 12 tabela dobijenih pomenutim kombinacijama parametara. Za testiranje koristiti dati referentni genom i kolekciju ridova. Prikazati rezultate tabelarno i grafički (**5 poena**).

Zadatak 6

Boyer-Moore algoritam koristi heuristike (Bad character and Good suffix rule) za preskakanje nepotrebnih poređenja karaktera pri pretraživanju. Definisati na sličan način dve heuristike (heuristika 1 i heuristika 2) koje mogu biti primenjivane za sračunavanje maksimalno dozvoljenog pomeraja. Za formiranje heuristika koristiti samo patern - algoritam treba da ostane on-line (10 poena).

Izvršiti merenje i poređenje vremena izvršavanja i memorijskog zauzeća seta on-line Exact Matching algoritama:

1. Boyer-Moore - Heuristika 1 + Heuristika 2
2. Boyer-Moore - Heuristika 1
3. Boyer-Moore - Heuristika 2
4. Boyer-Moore - Strong good suffix rule and bad character rule

Napisati wrapper funkciju ili klasu u programskom jeziku Python koja ima mogućnost izvršavanja zadate varijante algoritma na osnovu ulaznog parametra. Kao test podatke koristiti 3 seta podataka (5 poena):

1. Tekst: Coffea arabica, Chromosome 1c i paterni: ATGCATG, TCTCTCTA, TTCACTACTCTCA
2. Tekst: Mus pahari chromosome X, i paterni: ATGATG, CTCTCTA, TCACTACTCTCA
3. Genom po slobodnom izboru iz NIH baze i proizvoljna 3 paterna različite dužine.

Dobijene rezultate predstaviti tabelarno i grafički (Python matplotlib). Grafički način (dijagram) treba da bude dovoljno intuitivan da onaj koji ga čita može brzo izvući potrebne zaključke vezane za performanse zadatih varijanti algoritama (5 poena).

Za svaku od funkcija u kodu napisati testove (5 poena).

Pripremiti prezentaciju (Google slides ili power point) algoritama koji se testiraju, kao i samih rezultata (5 poena).

Pripremiti video prezentaciju projekta (3 - 5 minuta trajanja) koja će biti dostupna na YouTube ili drugom video servisu na kojem mu je moguće pristupiti (10 poena).

Zadatak 7 (1 student)

Analizirati razlike u ekspresiji gena između **dve sempl-grupe** (oko 10 semplova ukupno) koristeći (po izboru) programske jezike Python, R i gotova open source rešenja.

Ulazni podaci su ridovi u FASTQ formatu (simulirani podaci, generisani za svakog studenta). Potrebno je izvršiti pseudo-alignment **Salmon** alatom kako bi se dobila ekspresija na nivou gena za svaki sempl pojedinačno (**10 poena**), a zatim izvršiti adekvatnu normalizaciju (TMM, RLE ili neku drugu - **5 poena**).

Pre testova, proveriti da li se normalizovani semplovi klasteruju po grupama (npr. izvršiti PCA analizu, pa iscrtati scatterplot prve dve komponente) i eventualno odstraniti iz dalje analize semplove koji drastično odudaraju od očekivanog ponašanja (**5 poena**).

Testiraj razliku u ekspresiji i grafički uporedi setove diferencijalno ekspresovanih gena bez kontrole pri višestrukom testiranju i sa adekvatnim FDR ili FWER kontrolama po izboru. Venov dijagram ili nešto slično i kratak komentar (**20 poena**).

Uz izveštaj dostaviti i CSV fajl sa 2 kolone: gene_id (oznaka gena) i p-val.

Zadatak 8 (1 student)

Analizirati razlike u ekspresiji gena između **dve sempl-grupe** (oko 10 semplova ukupno) koristeći (po izboru) programske jezike Python, R ili gotova open source rešenja.

Ulazni podaci su ridovi u FASTQ formatu (simulirani podaci, generisani za svakog studenta). Potrebno je izvršiti poravnavanje (alignment, **5 poena**) i kvantifikovati ekspresiju na nivou gena za svaki sempl pojedinačno (**5 poena**), a zatim izvršiti adekvatnu normalizaciju (TMM, RLE ili neku drugu - **5 poena**).

Pre testova, proveriti da li se normalizovani semplovi klasteruju po grupama (npr. izvršiti PCA analizu, pa iscrtati scatterplot prve dve komponente) i eventualno odstraniti iz dalje analize semplove koji drastično odudaraju od očekivanog ponašanja (**5 poena**).

Testiraj razliku u ekspresiji i grafički uporedi setove diferencijalno ekspresovanih gena bez kontrole pri višestrukom testiranju i sa adekvatnim FDR ili FWER kontrolama po izboru. Venov dijagram ili nešto slično i kratak komentar (**20 poena**).

Uz izveštaj dostaviti i CSV fajl sa 2 kolone: gene_id (oznaka gena) i p-val.

Zadatak 9 (1 student)

RNA sequencing (RNA-seq) is used to measure gene expression levels across the transcriptome for a huge variety of samples. For example, RNA-seq has been applied to study gene expression in individuals with rare diseases, in hard-to-obtain tissues or for rare forms of cancer. Recently, enormous RNA-seq datasets have been produced in the GTEx (Genotype-Tissue Expression) study, which comprises 9,662 samples from 551 individuals and 54 body sites, and in the Cancer Genome Atlas (TCGA) study, which comprises 11,350 samples from 10,340 individuals and 33 cancer types. Public data repositories, such as the Sequence Read Archive (SRA), host >50,000 human RNA-seq samples. It is estimated that these repositories are likely to double in size every 18 months.

These resources offer a unique opportunity to gain better insight into complex human diseases. However, integrative analysis of data across studies poses great challenges, due to differences in sample handling and processing, such as sequencing platform and chemistry, personnel... A batch effect occurs when non-biological factors in an experiment cause changes in the data produced by the experiment. Such effects can lead to inaccurate conclusions when their causes are correlated with one or more outcomes of interest in an experiment.

Analizirati razlike u ekspresiji gena između RNA-Seq semplova tkiva prostate (ili bešike ili štitne žlezde) iz GTEx (Genotype-Tissue Expression) i TCGA (The Cancer Genome Atlas) data setova koristeći (po izboru) programske jezike Python, R ili gotova open source rešenja.

Ulazni podaci su tab-delimited tekstualni fajlovi sa gene-level kvantifikacijama (integer). Potrebno je izvrsiti batch effect korekciju (koristeci npr.) (**20 poena**). Potvrditi uspešnost crtanjem PCA plotova pre i posle (ilustracija: slike a i c).

Testiraj razliku u ekspresiji i grafički uporedi setove diferencijalno ekspresovanih gena bez kontrole pri višestrukom testiranju i sa adekvatnim FDR ili FWER kontrolama po izboru. Venov dijagram ili nešto slično i kratak komentar (**20 poena**).

Uz izveštaj dostaviti i CSV fajl sa 2 kolone: gene_id (oznaka gena) i p-val.