UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

Zaključna naloga

Rekonstrukcija evolucij mešanih tumorskih vzorcev z uporabo celoštevilskega linearnega programiranja

(Reconstructing the evolution of mixed tumor samples using integer linear programming)

Ime in priimek: Špela Čučko
Študijski program: Bioinformatika
Mentor: prof. dr. Martin Milanič

Koper, avgust 2020

Ključna dokumentacijska informacija

Ime in PRIIMEK: Špela ČUČKO

Naslov zaključne naloge: Rekonstrukcija evolucij mešanih tumorskih vzorcev z uporabo celoštevilskega linearnega programiranja

Kraj: Koper

Leto: 2020

Število listov: 50 Število slik: 18 Število tabel: 4

Število referenc: 19

Mentor: prof. dr. Martin Milanič

Ključne besede: popolna filogenija, razvoj mutacij v tumorju, celoštevilsko linearno programiranje, CPLEX

Izvleček:
S problemom rekonstrukcije filogenetskega drevesa, ki predstavlja grafično ponazoritev evolucijske zgodovine skupine organizmov, se srečujejo mnoga področja biologije. V zaključni nalogi je prikazana razlaga, implementacija in ovrednotenje programa za problem rekonstrukcije razvoja različnih vrst celic v tumorjih. Model je razširitev obstoječega modela MIPUP (Husić idr., 2019) na primer, ko so vhodni podatki manjkajoči ali premalo natančno določeni. Vhodni podatki so predstavljeni z matriko vrednosti VAF, ki jih program pretvori v celoštevilske vrednosti 0, 1 ali 2, glede na dve mejni vrednosti t_0 in t_1. Uporabili smo model v obliki celoštevilskega linearnega programa (CLP), ki podaja matematičen opis najpreprostejše možne razlage razvoja tumorja. CLP celoštevilsko $\{0, 1, 2\}$-matriko spremeni v pomožno binarno matriko, ki problem opiše s pomočjo digrafa, in prek posebnih podgrafov, imenovanih vejitve, izračuna najmanjše število vrstic v razcepu pomožne matrike, ki predstavlja najpreprostejšo možno razlago razvoja tumorja. Kvaliteto rekonstruiranih rešitev smo ovrednotili s simulacijo podatkov tako, da smo preverili, do kolikšne mere je izračun uspel rekonstruirati dejansko evolucijo. Pokazali smo, da razširitev modela MIPUP rekonstruira prvotno drevo večinoma bolje kot orodje LICHeE (Popic idr., 2015) in, za razliko od orodja LICHeE, vselej izračuna drevo. Preučili smo tudi vpliv izbranega razmerja za določitev mejnih vrednosti t_0 in t_1 na dobljene rezultate.

Key document information

Name and SURNAME: Špela ČUČKO

Title of final project paper: Reconstructing the evolution of mixed tumor samples using integer linear programming

Place: Koper

Year: 2020

Number of pages: 50 Number of figures: 18 Number of tables: 4

Number of references: 19

Mentor: Prof. Martin Milanič, PhD

Keywords: perfect phylogeny, evolution of tumor mutations, integer linear programming, CPLEX

Abstract: The phylogenetic tree reconstruction problem aims at computing a graphical representation of an evolutionary history of a set of organisms and is used in many fields of Biology. In this final project paper we explain, implement, and evaluate a program for reconstructing the evolution of tumor cells from a collection of mixed samples. The model is extension of an existing model called MIPUP (Husić et al., 2019) to the case when some of the input data is either missing or not known precisely. The data is represented by a matrix of VAF values, which are than converted to values 0, 1, or 2, defined by two thresholds t_0 and t_1. Next, the problem is modeled using an integer linear program (ILP), which gives a mathematical description of a simplest possible explanation of tumor evolution. The ILP converts the $\{0, 1, 2\}$-matrix into an auxiliary binary matrix, which describes the problem using a digraph and special subgraphs of it called branchings, to compute the smallest number of rows in a row split of the auxiliary matrix, which represents the simplest possible explanation of tumor evolution. We evaluated the quality of solutions using simulated data, by verifying to what extent the obtained solution was able to reconstruct the actual evolution. Results show that the MIPUP extension reconstructs the original tree mostly better than LICHeE (Popic et al., 2015) and, in contrast with LICHeE, always computes a tree. Furthermore, we analyzed how the thresholds t_0 and t_1 for converting VAF values to integer values affect the results.

Zahvala

Zahvaljujem se mentorju prof. dr. Martinu Milaniču za vso strokovno pomoč pri izdelavi zaključne naloge, potrpežljivost in hitro odzivnost. Prav tako se zahvaljujem dr. Alexandru I. Tomescuju za vse dodatne razlage. Še posebej bi se zahvalila družini, fantu Petru in prijateljem za vso podporo tekom celotnega študija.

Kazalo vsebine

1 Uvod **1**

2 Problem popolne filogenije **3**
2.1 Problem najmanjšega brezkonfliktnega vrstičnega razcepa 4
2.2 Problem vejitve najmanjšega nepokritja 5

3 Metode **8**
3.1 Celoštevilski linearen program za problem najmanjše binarne razširitve 9
3.1.1 Spremenljivke 11
3.1.2 Omejitve 13
3.1.3 Kriterijska funkcija 15
3.2 Konkreten primer izračuna MIPUP+ 16
3.3 Izris filogenetskega drevesa 18

4 Rezultati **20**
4.1 Simulacija podatkov 20
4.2 Realni podatki 27
4.2.1 Svetlocelični karcinom ledvic (ccRCC) 27
4.2.2 Serozni karcinom visokega gradusa (HGSC) 32

5 Diskusija **37**

6 Zaključek **39**

7 Literatura in viri **40**

Kazalo tabel

1 $N_A^-(v)$ 11

2 Delež AD parov za 10 vozlišč in 100 mutacij 23
3 Delež AD parov za 20 vozlišč in 200 mutacij 24
4 Delež AD parov za 30 vozlišč in 300 mutacij 26

Kazalo slik

1 Digraf $D_{M'}$ 17
2 Vejitev B v digrafu $D_{M'}$ in nepokriti pari glede na B. 17
3 Filogenetsko drevo 19
4 Primer drevesa za EV003 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 28
5 Primer drevesa za EV005 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 28
6 Primer drevesa za EV006 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 29
7 Primer drevesa za EV007 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 30
8 Primer drevesa za RK26 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 30
9 Primer drevesa za RMH002 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 31
10 Primer drevesa za RMH004 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 31
11 Primer drevesa za RMH008 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj) 32
12 Primer drevesa za case 1 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno) 33
13 Primer drevesa za case 2 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno) 33
14 Primer drevesa za case 3 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno) 34
15 Primer drevesa za case 6 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno) 34
16 Primer drevesa za case 4 za MIPUP+ (desno) 35
17 Primer matrik za case 5 35
18 Primer drevesa za case 5 za MIPUP+ (spodaj) in LICHeE (zgoraj) . . 36

Seznam kratic

AD par	prikaz odnosa prednika in potomcev
ccRCC	svetlocelični karcinom ledvic
CLP	celoštevilski linearni program
HGSC	serozni karcinom visokega gradusa
MCRS	problem najmanjšega brezkonfliktnega vrstičnega razcepa
MUB	problem vejitve najmanjšega nepokritja
SSNV	somatska različica posameznega nukleotida
tj.	to je
VAF	frekvenca alelne različice

1 Uvod

Mutacije, ki se pojavijo v kateri koli celici telesa, razen v zarodnih celicah, imenujemo somatske mutacije [19]. Kopičenje le-teh v posameznikovi življenjski dobi lahko privede do nenadzorovane rasti skupka celic v tumor in lahko povzroči raka ali druge bolezni. Zaradi kopičenja bo v tumorju več kot ena vrsta celic. Skupine celic z izrazitimi skupinami mutacij imenujemo kloni ali celična populacija tumorja [14].

Klonska teorija raka predvideva, da vse celice v tumorju izhajajo iz ene same ustanoviteljske celice in da se bodo poznejše klonske ekspanzije pojavile zaradi dodatnih mutacij, ki povzročajo širjenje ustanoviteljske celice [14]. Posledično se lahko celice v tumorju med seboj razlikujejo po somatskih mutacijah, pri čemer je vsaka celica potomec klona iz klonske ekspanzije [3].

Sekvenciranje tumorskih genomov z visoko pokritostjo omogoča preučevanje te intratumorske heterogenosti z merjenjem frekvenc mutacij znotraj tumorja. Karakterizacija heterogenosti znotraj tumorja in sklepanje o klonski evolucijski zgodovini somatskih mutacij znotraj tumorja dajeta koristen vpogled v razvoj tumorja in lahko pomagata pri poteku zdravljenja [3].

Rekonstrukcijo razvoja heterogenosti v tumorjih lahko opišemo z neposredno uporabo filogenetskega drevesa, saj so si kloni v tumorju filogenetsko sorodni. Njihova medsebojna razmerja lahko predstavimo s filogenetskim drevesom, ki ponazarja evolucijski odnos med kloni in vrstni red mutacij. Za preučevanje heterogenosti tumorja je bilo razvitih veliko učinkovitih računskih orodij za analizo informacij o klonskih tumorjih in njihovih evolucijskih zgodovinah. Vhodni podatki za orodja so podatki o genetski variabilnosti pridobljeni s sekvenciranjem tumorskih genomov z visoko pokritostjo. Orodja vrnejo klonsko sestavo tumorja in razmerje prednikov med kloni (glej npr. [3] in tam navedene reference).

Z analizo več vzorcev tumorja lahko s tehnologijo sekvenciranja v vsakem vzorcu zaznamo somatsko različico posameznega nukleotida (SSNV, ang. somatic single nucleotide variant). Delež celic v vzorcu, ki vsebuje SSNV, izmed vseh celic se v tem vzorcu imenuje frekvenca alelne različice (VAF, ang. variant allele frequency) SSNV-ja. V vsakem vzorcu tumorja lahko izračunamo vrednosti VAF za vsak SSNV [15].

Namen naloge je razlaga, implementacija in ovrednotenje programa za problem rekonstrukcije razvoja različnih vrst celic v tumorjih. Model je razširitev obstoječega

modela za ta problem iz članka Husića idr. [11] na primer, ko so vhodni podatki manjkajoči ali premalo natančno določeni.

Zaključna naloga je razdeljena na tri večja poglavja. V poglavju 2 opišemo problem popolne filogenije ter njegovi posplošitvi problem najmanjšega brezkonfliktnega vrstičnega razcepa (v razdelku 2.1) in problem vejitve najmanjšega nepokritja (v razdelku 2.2). V poglavju 3 definiramo problem najmanjše binarne razširitve ter opišemo implementirani program za reševanje tega problema. Nato v poglavju 4 analiziramo dobljene rezultate na simuliranih podatkih in realnih setih.

2 Problem popolne filogenije

S problemom rekonstrukcije filogenetskega drevesa, ki predstavlja grafično ponazoritev evolucijske zgodovine skupine organizmov, se srečujejo mnoga področja biologije. Eden izmed najosnovnejših filogenetskih modelov je t.i. model popolne filogenije. *Popolna filogenija* je drevo s korenom, ki predstavlja evolucijsko zgodovino množice objektov in množico njihovih lastnosti. Ustreza $m \times n$ binarni matriki M, ki ima objekte zapisane v vrsticah in lastnosti v stolpcih. Popolno filogenetsko drevo T je drevo s korenom, za katerega predpostavimo, da koren drevesa nima nobene izmed n lastnosti. Vsakemu izmed m danih objektov pripada natanko en list v T (in obratno) in vsaki izmed n lastnosti pripada natanko ena povezava v T. Za katerikoli list w v drevesu T pa lastnosti po povezavah na poti od korena proti listu natančno opišejo vektor lastnosti, ki veljajo za objekt na listu w. Po zgornjih predpostavkah v korenu dobimo vektor samih ničel, saj koren nima nobene od n lastnosti, na poti od korena do lista w pa lastnosti, ki so na povezavah te poti označene, dobijo vrednost 1, ostale pa vrednost 0. Filogenija je popolna, če v njej ne pride do reverznih mutacij in vzporedne evolucije [6].

V praksi bi želeli za dano binarno matriko M, ki ima po eno vrstico za vsak vzorec, ugotoviti, ali jo je mogoče predstaviti z neko popolno filogenijo. Ta problem se imenuje problem popolne filogenije. Iz matrike M bi torej radi sklepali posamezne podklone tumorja, ki sestavljajo vsak vzorec r_i, tj. celoštevilski vzorec SSNV-jev v vsakem takem podklonu, in evolucijsko razmerje med temi neznanimi podkloni.

Problem popolne filogenije lahko rešimo z *Gusfieldovim algoritmom* [6]. Ta iz podane binarne matrike M ugotovi, če za to matriko obstaja filogenetsko drevo in če tako drevo obstaja, potem ga tudi izračuna. V nadaljevanju opišemo algoritem. Če vsak stolpec u matrike M upoštevamo kot dvojiško številko z najpomembnejšim bitom v prvi vrstici, lahko razvrstimo stolpce od največje številke do najmanjše. Tako uredimo stolpce po velikosti od največjega do najmanjšega. Nato ustvarimo prazno matriko L, ki je enakih dimenzij kot matrika M. Če je vrednost matrike M v r-ti vrstici in u-tem stolpcu enaka 0, potem vrednosti na tem mestu nove matrike L ni, mesto je prazno. Če je vrednost matrike M v r-ti vrstici in u-tem stolpcu enaka 1, potem v L na tem polju zapišemo največjo vrednost indeksa, manjšega od u, na katerem je v trenutni vrstici matrike M vrednost 1. Če tak indeks ne obstaja, potem v matriko L na tem mestu zapišemo vrednost 0. V zadnjem koraku preverimo, ali so si vse vrednosti v matriki L,

v stolpcu u, med sabo enake. To vrednost stolpca u zapišemo kot vrednost $L(u)$. Če so te vrednosti med seboj enake, potem filogenetsko drevo obstaja, sicer pa ne.

Algoritem preveri, če filogenetsko drevo obstaja, v času $\mathcal{O}(nm)$.

Ko smo se prepričali, da popolno filogenetsko drevo matrike M obstaja, ga lahko zgradimo po naslednjem postopku. Za vsak stolpec u matrike M ustvarimo točko n_u. Nato dodamo korensko točko s. V naslednjem koraku pogledamo vrednosti v matriki L.

- Če je vrednost para (r, u) matrike L enaka 0, potem ustvarimo povezavo (s, n_u) in jo označimo z u.
- Če je vrednost para (r, u) matrike L večja od 0, potem ustvarimo povezavo $(n_{L(u)}, n_u)$ in jo označimo z u.

V drugem delu algoritma gledamo vrstice matrike M. Za vsako vrstico r_i označimo z m_{r_i} največji indeks, za katerega velja, da par (r_i, m_{r_i}) v matriki M zavzame vrednost 1. V drevesu najdemo povezavo, označeno z m_{r_i}. Če povezava vodi do lista, na list pripnemo r_i, sicer pa iz točke ustvarimo povezavo do novega lista in jo označimo z r_i. Primer izrisanega drevesa je prikazan na Sliki 3 na str. 19.

Definicija 2.1. Različna stolpca c_i in c_j matrike M sta v *konfliktu*, če obstajajo take tri različne vrstice (r, r', r'') v M, katerih polja (i, j) v stolpcih c_i in c_j zavzamejo vrednosti (1, 1), (1, 0) in (0, 1).

Matrika je brezkonfliktna, če ne vsebuje nobenega para konfliktnih stolpcev. Vrstice matrike M so listi popolnega filogenetskega drevesa, če in samo če je M brezkonfliktna [4,6]. Če je matrika M brezkonfliktna, potem iz te matrike vedno najdemo filogenetsko drevo.

Problem popolne filogenije je bil v literaturi posplošen na različne načine. V razdelkih 2.1 in 2.2 si bomo ogledali dve posplošitvi, pomembni za problem rekonstrukcije evolucij mešanih tumorskih vzorcev. Za dodatno branje v slovenskem jeziku o problemih popolne filogenije bralca napotimo na [12].

2.1 Problem najmanjšega brezkonfliktnega vrstičnega razcepa

V tem razdelku predpostavimo evolucijski model popolne filogenije in formalno opredelimo problem najmanjšega brezkonfliktnega vrstičnega razcepa (ang. Minimum Conflict-free Row Split problem, MCRS).

Vrstice eksperimentalno pridobljene binarne matrike M, ki predstavljajo mešanice subklonov tumorja, običajno sicer ne ustrezajo predpostavkam modela popolne filogenije. Vsako vrstico r_i matrike M želimo torej razcepiti na množico vrstic R_i, tako da

dobljena matrika M' ustreza neki popolni filogeniji. Izvedba razcepa teži k minimalnemu številu razcepljenih vrstic. V smislu popolnih filogenetskih dreves to pomeni, da želimo vsak vzorec razdeliti v zbirko podklonov, ki tvorijo popolno filogenijo, skupno število podklonov iz vseh vzorcev pa je minimalno. Hajirasouliha in Raphael sta v [7] ta problem poimenovala Najmanjši brezkonfliktni razcep vrstic. Model popolne filogenije predpostavlja, da so vse mutacije v matičnih celicah posredovane potomcem. Prav tako predpostavi, da ko se mutacija na določenem mestu pojavi, se na tem mestu ne zgodi več. V ta namen sta uvedla naslednjo definicijo:

Definicija 2.2. Naj bo $M \in \{0,1\}^{m\times n}$. Vrstice M označimo kot $r_1, \ldots, r_m$. Binarna matrika $M' \in \{0,1\}^{m'\times n}$ je *vrstični razcep* matrike M, če obstaja taka particija množice vrstic M' na m množic $R_1, \ldots, R_m$, da za vse $i \in \{1, \ldots, m\}$ in vse $j \in \{1, \ldots, n\}$ velja: $r_{i,j} = 1$ natanko tedaj, ko obstaja neka vrstica $\tilde{r} \in R_i$, za katero velja $\tilde{r}_j = 1$. Za množico R_i vrstic matrike M' rečemo, da je množica razcepljenih vrstic vrstice r_i matrike M (glej primer 3.4).

Najmanjše možno število vrstic v brezkonfliktnem vrstičnem razcepu M' matrike M označimo z $\gamma(M)$ (glej [8, 10]). Ustrezni optimizacijski problem se formalno glasi takole:

Problem 1: Najmanjši brezkonfliktni vrstični razcep (MCRS)
Vhod: Binarna matrika M.
Naloga: Izračunaj $\gamma(M)$.

Hujdurović idr. [10] so pokazali, da je problem MCRS NP-težek.

2.2 Problem vejitve najmanjšega nepokritja

Hujdurović idr. [8] so pokazali, da lahko problem MCRS ekvivalentno formuliramo s tako imenovanim problemom vejitve najmanjšega nepokritja (ang. Minimum Uncovering Branching, MUB). Ta je opisan s pomočjo vejitev v acikličnih digrafih. Za lažje razumevanje vpeljimo naslednje definicije. Kadar je vsaka povezava v grafu urejen par točk, govorimo o *digrafu*. Digraf je *acikličen*, če ne vsebuje nobenega cikla.

Naslednje oznake in predpostavke za matrike veljajo v okviru celotne zaključne naloge. Za poljubno matriko M bomo z $R_M = \{r_i \mid 1 \leq i \leq m\}$ in $C_M = \{c_j \mid 1 \leq j \leq n\}$ označili družini vrstic in stolpcev matrike M. Z M_{r_i,c_j} (oziroma kar z $M_{i,j}$) označimo vrednost matrike v i-ti vrstici in j-tem stolpcu. Za vse obravnavane matrike predpostavimo, da ne vsebujejo vrstice, v kateri so vse vrednosti enake 0. Matrika $M \in \mathbb{R}^{m\times n}$ je *binarna*, če je $M_{i,j} \in \{0,1\}$ za vse $i \in \{1, \ldots, m\}$ in $j \in \{1, \ldots, n\}$. Množico vseh $m \times n$ binarnih matrik označimo z $\{0,1\}^{m\times n}$.

Definicija 2.3. Za dano binarno matriko M in stolpec $c_j \in C_M$ je *nosilec* stolpca c_j definiran kot množica $\{r_i \in R_M \mid M_{i,j} = 1\}$. Označimo ga s $\text{supp}_M(c_j)$.

Definicija 2.4. *Digraf vsebovanosti* binarne matrike $M \in \{0,1\}$ velikosti $m \times n$ je acikličen digraf D_M z množico točk $V = \{\text{supp}_M(c) \mid c \in C_M\}$ in množico povezav $A = \{(v, v') \mid v, v' \in V \wedge v \subset v'\}$.

Primer digrafa D_M je prikazan v primeru 3.4 na str. 17.

Definicija 2.5. *Vejitev* v digrafu $D = (V, A)$ je taka podmnožica B množice A, da je (V, B) digraf, v katerem iz vsake točke v obstaja največ ena izhodna povezava.

Primer vejitve lahko bralec najde v primeru 3.4 na str. 17.

Definicija 2.6. Naj bo $M \in \{0,1\}^{m \times n}$ binarna matrika, naj bo $D_M = (V, A)$ digraf vsebovanosti matrike M in naj bo B vejitev v D_M. Za točko $v \in V$ označimo z $N_B^-(v)$ množico vseh takih točk $v' \in V$, da je $(v', v) \in B$. Za vozlišče $v \in V$ in vrstico $r \in v$ iz matrike M' je par (r, v) glede na B *pokrit*, če je $r \in \cup N_B^-(v)$, sicer je *nepokrit*.

Primer pokritih in nepokritih parov je podan v primeru 3.4 na str. 17.

Najmanjše možno število nepokritih parov glede na vejitev B v digrafu D_M označimo z $\beta(M)$. Problem vejitve najmanjšega nepokritja je naslednji:

Problem 2: Vejitev najmanjšega nepokritja (MUB)
Vhod: Binarna matrika M.
Naloga: Izračunaj $\beta(M)$.

Za poljubno vejitev B v digrafu vsebovanosti D_M lahko s pomočjo nepokritih parov glede na vejitev dobimo brezkonfliktni vrstični razcep M' matrike M. Vsaka vrstica predstavlja en nepokrit par oblike (r_i, v_j), kjer je v_j neka točka digrafa vsebovanosti, v kateri se nahaja r_i (glej primer 3.4 na str. 17).

Naj bo $M \in \{0,1\}^{m \times n}$ binarna matrika in $R_M = \{r_1, \ldots, r_m\}$, $C_M = \{c_1, \ldots, c_n\}$. Z D_M označimo ustrezni digraf vsebovanosti ter z B poljubno vejitev D_M. Naj bo $U(B) = \{u_1, \ldots, u_k\}$ množica vseh B-nepokritih parov. Naj bo $v_j = \text{supp}_M(c_j)$ za vse $j \in \{1, \ldots, n\}$ in $V = V(D_M)$. Za $v \in V$ z $B^+(v)$ označimo množico vseh točk v V, ki so iz točke v dosegljive prek neke usmerjene poti v digrafu (V, B).

Definicija 2.7. *B-razcep* matrike M je matrika M^B, katere vrstice so indeksirane z $u_1, \ldots, u_k$, stolpci z $1, \ldots, n$, za vse pare $(r, v) \in U(B)$ in vse $j \in \{1, \ldots, n\}$ pa velja:

$$M^B_{(r,v),j} = \begin{cases} 1, & \text{če je } v_j \in B^+(v); \\ 0, & \text{sicer.} \end{cases}$$

Naslednja lema iz [8] pokaže, da je B-razcep matrike M dejansko brezkonfliktni vrstični razcep te matrike. Z $U_B(r)$ označimo množico vseh z B-nepokritih parov s prvo koordinato enako r.

Lema 2.8. *Naj bo M binarna matrika in D_M njen digraf vsebovanosti. Naj bo B poljubna vejitev v digrafu D_M. Tedaj je M^B brezkonfliktni vrstični razcep matrike M z $|U(B)|$ vrsticami, dobljenimi z razcepom vsake vrstice r_i matrike M v vrstice iz M^B, indeksiranih z elementi iz $U_B(r_i)$.*

Problem najmanjšega brezkonfliktnega vrstičnega razcepa (MCRS) in problem vejitve najmanjšega nepokritja (MUB) sta ekvivalentna problema. Hujdurović idr. [8] so ugotovili, da za vsako binarno matriko $M \in \{0,1\}^{m\times n}$ velja $\gamma(M) = \beta(M)$ in podali učinkovit algoritem, ki poljubno optimalno rešitev za enega izmed obeh problemov pretvori v optimalno rešitev drugega problema. S pomočjo zgornje ekvivalence so Husić idr. [11] problem MCRS modelirali s celoštevilskim linearnim programom. Nedavne raziskave o problemu MUB, ki vključujejo novo spodnjo mejo za optimalno vrednost problema in dve družini učinkovito rešljivih primerov, lahko bralec najde v magistrskem delu [1].

3 Metode

V okviru zaključne naloge smo implementirali algoritem za problem rekonstrukcije razvoja različnih vrst celic v tumorjih. Recimo, da imamo tumorske vzorce, ki jih označimo z $r_1, \ldots, r_m$. Z različnimi metodami lahko s pomočjo točkovnih mutacij zaznamo somatske različice posameznega nukleotida (SSNV), ki so prisotne v vsakem vzorcu, in izpeljemo njihove vrednosti VAF iz prebranih poravnav nad njihovimi položaji. Te SSNV označimo s $c_1, \ldots, c_n$. Vhodni podatki algoritma so predstavljeni z matriko vrednosti VAF, ki se nahajajo v intervalu $[0, 1]$. Algoritem sestavljajo štiri osnovne funkcije. Ena izmed njih sloni na naslednjem optimizacijskem problemu, ki je posplošitev problema MUB. Definicijo $\beta(M)$ razširimo z binarnih matrik na matrike z vrednostmi $\{0, 1, 2\}$. Naj bo matrika $M \in \{0, 1, 2\}^{m \times n}$. Pravimo, da je binarna matrika $M' \in \{0, 1\}^{m \times n}$ *(binarna) razširitev* matrike M, če velja: $M'_{r,v} = M_{r,v}$ za vsak par (r, v) matrike M, za katerega je vrednost $M_{r,v}$ različna od 2. Vrednost $\beta(M)$ definiramo s predpisom

$$\beta(M) = \min_{M' \colon M' \text{ je razširitev } M} \beta(M') \,.$$

Če je vhodna matrika M binarna, potem je $M' = M$ edina razširitev matrike M, kar pomeni, da se v tem primeru definicija $\beta(M)$ ujema s prejšnjo definicijo. Ustrezni optimizacijski problem se imenuje problem najmanjše binarne razširitve in je formalno podan na naslednji način:

Problem 3: Najmanjša binarna razširitev
Vhod: Matrika $M \in \{0, 1, 2\}^{m \times n}$.
Naloga: Izračunaj $\beta(M)$.

Problem je NP-težek že za binarne matrike, saj je v tem primeru ekvivalenten NP-težkemu problemu MCRS.

Sedaj imamo pripravljeno vse potrebno za opis štirih osnovnih funkcij algoritma. Delovanje algoritma je odvisno od izbire dveh parametrov t_0 in t_1, ki sta števili z intervala $[0, 1]$, za kateri velja $t_0 \leq t_1$. Glede na to je algoritem razširitev algoritma MIPUP iz članka Husića idr. [11], poimenovali ga bomo **MIPUP+**.

Algoritem 1: MIPUP+

Vhod: Matrika z VAF vrednostmi.

Izhod: Filogenetsko drevo.

1. Algoritem pretvori vrednosti VAF v celoštevilske vrednosti 0, 1 ali 2, glede na dve mejni vrednosti t_0 in t_1. Vrednosti, manjše od mejne vrednosti t_0, pretvori v 0, vrednosti, večje od t_1, postanejo 1, vrednosti VAF, ki se nahajajo v intervalu med mejnima vrednostma, pa postanejo 2. Vrednost 2 se dodeli tudi poljubni celici matrike, kjer vrednost VAF ni znana. Te vrednosti algoritem zapiše v matriko M velikosti $m \times n$. Vrstice matrike M, označimo z $r_1, \ldots, r_m$, stolpce pa s $c_1, \ldots, c_n$. Vrednosti matrike M so torej definirane na naslednji način:

$$M_{i,j} = \begin{cases} 0, & \text{če je } 0 \leq VAF_{ij} < t_0; \\ 1, & \text{če je } t_1 < VAF_{ij} \leq 1; \\ 2, & \text{sicer.} \end{cases}$$

   Vsaka vrstica matrike M predstavlja en vzorec tumorja, sestavljen iz različnih vrst celic, in torej ustreza množici listov neznanega filogenetskega drevesa. Stolpci matrike pa predstavljajo mutacije.

2. Drugi del algoritma sprejme matriko M in jo uporabi kot vhod za problem najmanjše binarne razširitve. Vrednost $\beta(M)$ je izračunana s pomočjo celoštevilskega linearnega programa za problem najmanjše binarne razširitve, ki je opisan v razdelku 3.1. Optimalno rešitev celoštevilskega linearnega programa izračunamo z uporabo reševalnika CPLEX. Optimalna rešitev je predstavljena s parom (M', B), kjer je M' taka binarna razširitev matrike M in B taka vejitev v digrafu vsebovanosti $D_{M'}$ matrike M', da velja $\beta(M) = |U(B)|$.

3. S pomočjo vejitve B algoritem nato izračuna optimalen brezkonfliktni vrstični razcep M'' matrike M' z metodo, podano v razdelku 2.2 (definicija 2.7).

4. Z uporabo Gusfieldovega algoritma (opisanega v razdelku 2) algoritem nazadnje iz brezkonfliktne matrike M'' izračuna filogenetsko drevo. Z uporabo razcepa M'' obenem opišemo, kateri listi filogenetskega drevesa tvorijo mešane vzorce.

3.1 Celoštevilski linearen program za problem najmanjše binarne razširitve

Problem najmanjše binarne razširitve je mogoče modelirati in rešiti s celoštevilskim linearnim programom (CLP). CLP temelji na celoštevilski matriki z vrednostmi 0, 1

in 2, izračunani v prvem koraku, in podaja matematičen opis najpreprostejše možne razlage razvoja tumorja [9]. Prek različnih vrst binarnih spremenljivk CLP spremeni celoštevilsko matriko M z vrednostmi 0, 1 in 2 v pomožno binarno matriko M', ki problem opiše s pomočjo digrafa, in prek vejitev izračuna najmanjše število vrstic v brezkonfliktnem razcepu matrike M', ki predstavlja najpreprostejšo možno razlago razvoja tumorja. Za reševanje izpeljanega celoštevilskega linearnega programa bomo uporabili reševalnik CPLEX, ki ga je razvil IBM. V implementaciji programa smo reševalnik uporabili v programskem jeziku Java.

Matrika $M \in \{0,1,2\}^{m\times n}$ ima m vrstic in n stolpcev. Vsaka vrstica matrike M predstavlja vektor v $\{0,1,2\}^n$; vsak stolpec pa vektor v $\{0,1,2\}^m$. Spomnimo, za vse obravnavane matrike predpostavimo, da ne vsebujejo vrstice, v kateri so vse vrednosti enake 0. Če se to zgodi, potem algoritem tako vrstico avtomatsko odstrani.

Za opis celoštevilskega linearnega programa potrebujemo še nekaj definicij in oznak. Z A označimo naslednjo množico parov stolpcev u in v matrike M:

$$A = \{(u,v) \mid u,v \in C_M, u \neq v, (\forall r \in R_M)(M_{r,u} = 1 \implies M_{r,v} \neq 0)\}.$$

Primer 3.1. Za primer vzemimo naslednjo matriko M velikosti 6×7, ki zavzema vrednosti 0, 1 in 2:

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 2 \\ 2 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 2 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 2 & 1 & 2 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 2 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Množica A matrike M vsebuje vse pare stolpcev, v katerih se ne pojavi vrstica, v kateri je vrednost stolpca u enaka 1 in vrednost stolpca v enaka 0. V našem primeru je množica $A = \{(c_1,c_2),(c_1,c_3),(c_1,c_4),(c_1,c_5),(c_1,c_6),(c_1,c_7),(c_3,c_2),(c_4,c_1),(c_4,c_2),(c_4,c_3),(c_4,c_5),(c_4,c_6),(c_4,c_7),(c_5,c_2),(c_7,c_1),(c_7,c_2),(c_7,c_3),(c_7,c_4),(c_7,c_5),(c_7,c_6)\}$.

Za vsako binarno razširitev M', ki jo dobimo iz matrike M, je množica povezav digrafa vsebovanosti $D_{M'}$ podmnožica množice A. Elementi množice A so pari (i,j), grafično predstavljeni kot $i \to j$ (glej Sliko 1 na str. 17).

Optimalna rešitev celoštevilskega linearnega programa bo predstavljena z neko binarno razširitvijo M' matrike M in vejitvijo B v digrafu $D_{M'}$. Dva stolpca binarne matrike smatramo za *identična*, če sta ustrezna binarna vektorja enaka oz. ekvivalentno, če sta nosilca stolpcev enaka. Množica V predstavlja različne stolpce matrike

M' in bijektivno ustreza vozliščem digrafa $D_{M'}$. Bralca naj ob tem opozorimo, da je množica V odvisna od optimalne rešitve celoštevilskega linearnega programa in jo je prav tako potrebno opisati s spremenljivkami CLP-ja.

Za poljuben $v \in C_M$ označimo z $N_A^-(v)$ množico vseh parov oblike $(u, v) \in A$.

Primer 3.2. V našem primeru (primer 3.1) množica $N_A^-(v)$ zavzema vrednosti:

Tabela 1: $N_A^-(v)$

v	$N_A^-(v)$
c_1	$\{c_3, c_6\}$
c_2	$\{c_1, c_3, c_4, c_5, c_7\}$
c_3	$\{c_1, c_4, c_7\}$
c_4	$\{c_1, c_7\}$
c_5	$\{c_1, c_4, c_7\}$
c_6	$\{c_1, c_4, c_7\}$
c_7	$\{c_1, c_4\}$

3.1.1 Spremenljivke

Za uporabo celoštevilskega linearnega programa vpeljemo različne vrste binarnih spremenljivk. To so spremenljivke, ki so celoštevilske in lahko zavzamejo le vrednosti 0 ali 1. Osnovna ideja modela CLP je v tem, da za vsak par (r, v) matrike M vpeljemo binarno spremenljivko $t_{r,v}$, pri čemer je $t_{r,v} = M'_{r,v}$ v binarni matriki M', ki predstavlja razširitev matrike M, ki minimizira vrednost $\beta(M')$. Iz vrednosti $t_{r,v}$ poznamo matriko M' in iz nje lahko, z ustrezno vejitvijo B, izračunamo vrednost $\beta(M')$.

V CLP-ju imamo pet osnovnih vrst binarnih spremenljivk:

- $t_{r,v}$
  spremenljivka je definirana za vse vrstice $r \in R_M$ in stolpce $v \in C_M$. S temi spremenljivkami opišemo binarno matriko M'; spremenljivka $t_{r,v}$ predstavlja vrednost $M'_{r,v}$ v optimalni razširitvi matrike M.
- p_v
  spremenljivka je definirana za vse stolpce $v \in C_M$. S temi spremenljivkami opišemo množico V. Spremenljivka p_v zavzame vrednost 1, če in samo če je istoležni stolpec v matrike M' element množice V.

- $q_{u,v}$
  spremenljivko vpeljemo za vse pare stolpcev (u, v) matrike M. S temi spremenljivkami opišemo, kateri pari stolpcev matrike M' so identični. Spremenljivka $q_{u,v}$ zavzame vrednost 1, če in samo če sta za par stolpcev (u, v) ustrezna binarna vektorja v matriki M' identična.
- $x_{u,v}$
  spremenljivka je definirana za vse pare stolpcev (u, v), ki pripadajo množici A. S temi spremenljivkami opišemo vejitev B v digrafu $D_{M'}$. Spremenljivka $x_{u,v}$ zavzame vrednost 1, če in samo če je par stolpcev (u, v) element množice B.
- $y_{r,v}$
  spremenljivko $y_{r,v}$ uvedemo za vse pare (r, v), kjer je r vrstica matrike M, v pa stolpec matrike M in velja $M_{r,v} \in \{1, 2\}$. Spremenljivka $y_{r,v}$ zavzame vrednost 1, če in samo če je $v \in V$, $M'_{r,v} = 1$ ter je par (r, v) nepokrit glede na B.

Število vseh $t_{r,v}$ spremenljivk je $m \cdot n$. Vseh p_v spremenljivk je enako številu vseh stolpcev matrike M, torej jih je n. Spremenljivk $q_{u,v}$ je toliko, kot je vseh parov stolpcev u in v, skupaj n^2. Število vseh $x_{u,v}$ spremenljivk je toliko, kot je vseh parov stolpcev $(u, v) \in A$, teh je največ n^2. Spremenljivk $y_{r,v}$ je največ $n \cdot m$. Število vseh spremenljivk je tako v splošnem $\mathcal{O}(mn + n^2)$.

V matriki M iz našega primera (primer 3.1) je $m = 6$ in $n = 7$. Matrika vsebuje 21 ničel, 11 enic in 10 dvojk. Če poračunamo število vseh spremenljivk, dobimo:

- $t_{r,v}$: 42 spremenljivk,
- p_v: 7 spremenljivk,
- $q_{u,v}$: 49 spremenljivk,
- $x_{u,v}$: 20 spremenljivk,
- $y_{r,v}$: $11 + 10 = 21$ spremenljivk.

V omejitvah se pojavijo še dodatne, t.i. *produktne* spremenljivke, ki so potrebne za linearizacijo nekaterih produktov osnovnih spremenljivk. Opisane so v razdelku 3.1.2. Preštejmo še produktne spremenljivke. Teh je:

- $z_{r,u,v}$: 294 spremenljivk,
- $\tilde{q}_{u,v}$: 49 spremenljivk
- $\tilde{t}_{r,u,v}$: 64 spremenljivk,
- $\tilde{p}_{r,v}$: $11 + 10 = 21$ spremenljivk.

Vseh skupaj je spremenljivk torej 567.

3.1.2 Omejitve

V celoštevilskem linearnem programu je potrebno določiti omejitve, ki različne vrste spremenljivk povežejo med seboj in zagotovijo, da bodo imele spremenljivke želen pomen.

Za vse vrstice $r \in R_M$ in stolpce $v \in C_M$, za katere velja $M_{r,v} \in \{0,1\}$, velja:

$$t_{r,v} = M_{r,v} \tag{3.1}$$

Spremenljivka $t_{r,v}$ bo tako zavzela vrednost $M_{r,v}$ natanko tedaj, ko je $M_{r,v} \in \{0,1\}$.

Za vse stolpce u in v matrike M velja:

$$q_{u,v} + \sum_{r \in R} (t_{r,u} + t_{r,v} - 2t_{r,u}t_{r,v}) \geq 1 \tag{3.2}$$

in

$$|R_M| \cdot q_{u,v} + \sum_{r \in R} (t_{r,u} + t_{r,v} - 2t_{r,u}t_{r,v}) \leq |R_M|\,, \tag{3.3}$$

kjer $|R_M|$ predstavlja število vseh vrstic v matriki M. Neenakosti (3.2)–(3.3) modelirajo pogoj, da spremenljivka $q_{u,v}$ zavzame vrednost 1, če in samo če sta za par stolpcev (u,v) ustrezna binarna vektorja v matriki M' identična. Pravilnost teh neenakosti sledi iz dejstva, da je za $t_{r,u}, t_{r,v} \in \{0,1\}$ vrednost izraza $t_{r,u} + t_{r,v} - 2t_{r,u}t_{r,v}$ enak vrednosti izraza $(t_{r,u} - t_{r,v})^2$, ta vrednost pa je enaka 0 natanko tedaj, ko je $t_{r,u} = t_{r,v}$, sicer pa je strogo pozitivna.

Za vse stolpce $u \in C_M$ velja:

$$\sum_{v \in C_M} q_{u,v} p_v = 1 \tag{3.4}$$

Enakosti (3.4) modelirajo pogoj, da je stolpec u matrike M' identičen natanko enemu od stolpcev matrike M', ki pripadajo množici V.

Za vse pare stolpcev $(u,v) \in A$ velja:

$$2x_{u,v} \leq p_u + p_v \tag{3.5}$$

Neenakosti (3.5) modelirajo pogoj, da če je (u,v) povezava vejitve B, potem sta obe krajišči u in v te povezave vsebovani v množici V, ki jo identificiramo z množico točk digrafa $D_{M'}$.

Za vse pare stolpcev $(u,v) \in A$ in vse vrstice $r \in R_M$, kjer sta vrednosti v matriki $M_{r,u} = 2$ ali $M_{r,v} = 2$, velja:

$$x_{u,v} + t_{r,u} - t_{r,v} \leq 1 \tag{3.6}$$

Neenakosti (3.6) modelirajo pogoj, da če je (u,v) povezava vejitve B, potem za vse vrstice $r \in R_M$ velja $t_{r,u} \leq t_{r,v}$, oz. ekvivalentno, da je $\mathrm{supp}_{M'}(u) \subset \mathrm{supp}_{M'}(v)$ oz. ekvivalentno, da je (u,v) povezava digrafa $D_{M'}$.

Za vse stolpce $u \in C_M$ velja:

$$\sum_{(u,v)\in A} x_{u,v} \leq 1 \tag{3.7}$$

Neenakosti (3.7) modelirajo pogoj iz definicije vejitev. Pogoj zagotavlja, da spremenljivke $x_{u,v}$ opisujejo vejitev B.

Za vse vrstice $r \in R_M$ in stolpce $v \in C_M$, kjer je $M_{r,v} \in \{1, 2\}$, velja:

$$y_{r,v} + \sum_{u\in N_A^-(v)} t_{r,u}x_{u,v} \geq p_v t_{r,v} \tag{3.8}$$

Neenakosti (3.8) modelirajo pogoj, da spremenljivka $y_{r,v}$ zavzame vrednost 1, če je $v \in V$, $M'_{r,v} = 1$ ter je (r, v) nepokrit glede na B.

Ker imamo v omejitvah produkte spremenljivk, je treba tudi njih linearizirati. To storimo tako, da vsak produkt oblike $c = a \cdot b$ predstavimo z novo binarno spremenljivko, za katero dodamo dodatne omejitve, in sicer $c \leq a$, $c \leq b$ in $c \geq a + b - 1$.

Produkt $t_{r,u} \cdot t_{r,v}$ opazimo v omejitvah vrste (3.2) in (3.3). Produkt nadomestimo z novo binarno spremenljivko $z_{r,u,v}$. Dodatne omejitve za to produktno spremenljivko izgledajo tako:

$$z_{r,u,v} \leq t_{r,u}$$

$$z_{r,u,v} \leq t_{r,v}$$

$$z_{r,u,v} \geq t_{r,u} + t_{r,v} - 1$$

Produkt $q_{u,v} \cdot p_v$ zapišimo kot $\tilde{q}_{u,v}$. Ta produkt najdemo v omejitvah vrste (3.4) in ga zapišemo na sledeči način:

$$\tilde{q}_{u,v} \leq q_{u,v}$$

$$\tilde{q}_{u,v} \leq p_v$$

$$\tilde{q}_{u,v} \geq q_{u,v} + p_v - 1$$

Produkta $t_{r,u} \cdot x_{u,v}$ in $p_v \cdot t_{r,v}$ najdemo v zadnjih omejitvah, vrste (3.8). Prvega zapišimo kot $\tilde{t}_{r,u,v}$, drugega pa kot $\tilde{p}_{r,v}$. Dodatne omejitve za prvi produkt, $\tilde{t}_{r,u,v}$, zapišemo kot:

$$\tilde{t}_{r,u,v} \leq t_{r,u}$$

$$\tilde{t}_{r,u,v} \leq x_{u,v}$$

$$\tilde{t}_{r,u,v} \geq t_{r,u} + x_{u,v} - 1$$

Dodatne omejitve za drugi produkt, $\tilde{p}_{r,v}$, izgledajo tako:

$$\tilde{p}_{r,v} \leq p_v$$

$$\tilde{p}_{r,v} \leq t_{r,v}$$

$$\tilde{p}_{r,v} \geq p_v + t_{r,v} - 1$$

Število vseh omejitev vrste (3.1) ali (3.8) je največ $m \cdot n$. Vseh omejitev vrste (3.2), (3.3) ali (3.5) je največ n^2. Omejitev vrste (3.4) in (3.7) je natanko n. Število vseh omejitev vrste (3.6) je največ mn^2. Število produktnih spremenljivk vrste $z_{r,u,v}$ je največ mn^2, z njimi povezanih omejitev je $\mathcal{O}(mn^2)$. Podobno se prepričamo, da je število omejitev, povezanih s produktnimi spremenljivkami vrste $\tilde{q}_{u,v}$, reda $\mathcal{O}(n^2)$, število omejitev, povezanih s produktnimi spremenljivkami vrste $\tilde{t}_{r,u,v}$, reda $\mathcal{O}(mn^2)$, število omejitev, povezanih s produktnimi spremenljivkami vrste $\tilde{p}_{r,v}$, pa je reda $\mathcal{O}(mn)$. Skupno število vseh omejitev je tako v splošnem enako $\mathcal{O}(mn^2)$.

Primer 3.3. Kot smo opazili že prej, v matriki M iz našega primera (primer 3.1) najdemo 21 ničel, 11 enic in 10 dvojk. Če poračunamo število vseh omejitev, dobimo:

- omejitev (3.1): $21 + 11 = 32$ omejitev,
- omejitev (3.2): 49 omejitev,
- omejitev (3.3): 49 omejitev,
- omejitev (3.4): 7 omejitev,
- omejitev (3.5): 20 omejitev,
- omejitev (3.6): 63 omejitev,
- omejitev (3.7): 7 omejitev,
- omejitev (3.8): $11 + 10 = 21$ omejitev,
- omejitev produktnih spremenljivk vrste $z_{r,u,v}$: 882 omejitev,
- omejitev produktnih spremenljivk vrste $\tilde{q}_{u,v}$: 49 omejitev,
- omejitev produktnih spremenljivk vrste $\tilde{t}_{r,u,v}$: 192 omejitev,
- omejitev produktnih spremenljivk vrste $\tilde{p}_{r,v}$: 42 omejitev.

Vseh omejitev je torej 1413.

3.1.3 Kriterijska funkcija

Želimo, da nam celoštevilski linearen program izračuna dopustno rešitev, ki minimizira vrednost kriterijske funkcije, ki prešteva nepokrite pare glede na vejitev B:

$$\min \sum_{v \in C_M} \sum_{\substack{r \in R_M \\ M_{r,v} \in \{1,2\}}} y_{r,v} \,.$$

Pravilnost celoštevilskega linearnega programa je podrobneje utemeljena v [9].

Po končanem izračunu nam reševalnik CPLEX vrne matriko M' in vejitev B, ki ju pridobi tako, da na podlagi spremenljivk in omejitev izračuna najbolj optimalno zamenjavo dvojk iz matrike M v ničle in enice ter vejitev v pripadajočem digrafu

vsebovanosti, ki minimizira število nepokritih parov. Matriko M' iz neke optimalne rešitve CLP razberemo s pomočjo vrednosti spremenljivk $t_{r,v}$, optimalno vejitev B v digrafu $D_{M'}$ pa kot množico vseh parov $(u, v) \in A$, za katere spremenljivka $x_{u,v}$ v optimalni rešitvi zavzame vrednost 1.

3.2 Konkreten primer izračuna MIPUP+

Primer 3.4. Naši matriki M iz primera 3.1 CLP priredi matriko M' na sledeči način:

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 2 \\ 2 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 2 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 2 & 1 & 2 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 2 & 1 & 1 & 0 \end{pmatrix} \end{array} \rightarrow \quad M' = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{array} \begin{array}{c} \begin{array}{ccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

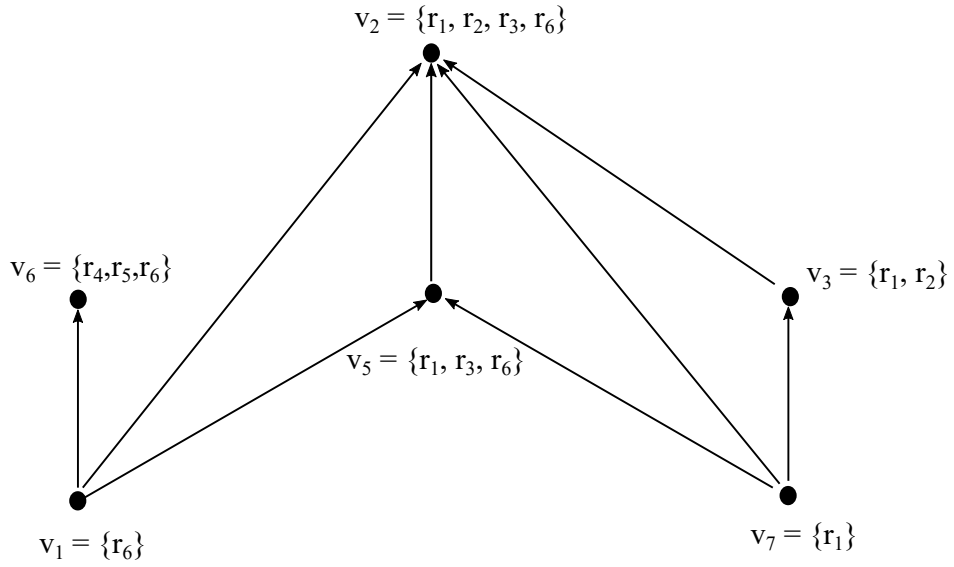
Ker imata matriki M in M' enaki dimenziji, bomo vrstici obeh matrik označevali preprosto z $r_1, \ldots, r_m$, stolpce pa s $c_1, \ldots, c_n$, pri čemer bo iz konteksta vselej razvidno, za katero od matrik gre. Opazimo, da je matrika M' res binarna razširitev matrike M. Opazimo tudi, da sta stolpca c_1 in c_4 matrike M' identična.

Iz matrike M' rekonstruiramo digraf $D_{M'}$. To naredimo tako, da vsaka točka v_i predstavlja en stolpec, pri čemer identični stolpci predstavljajo isto točko. Poleg vsake točke v zavite oklepaje zapišemo nosilec $\text{supp}_{M'}(c_i)$. Točke povežemo, če je nosilec $\text{supp}_{M'}(c_i)$ podmnožica nosilca $\text{supp}_{M'}(c_j)$. Digraf vsebovanosti $D_{M'}$ je prikazan na sliki 2. V vseh nadaljnjih zgledih in izračunih bomo stolpce c_i pogosto identificirali z ustreznimi nosilci v_i.

Kot smo omenili, nam celoštevilski linearen program poda tudi optimalno vejitev. Za naš primer celoštevilski linearen program izbere vejitev, določeno z naslednjo množico povezav:

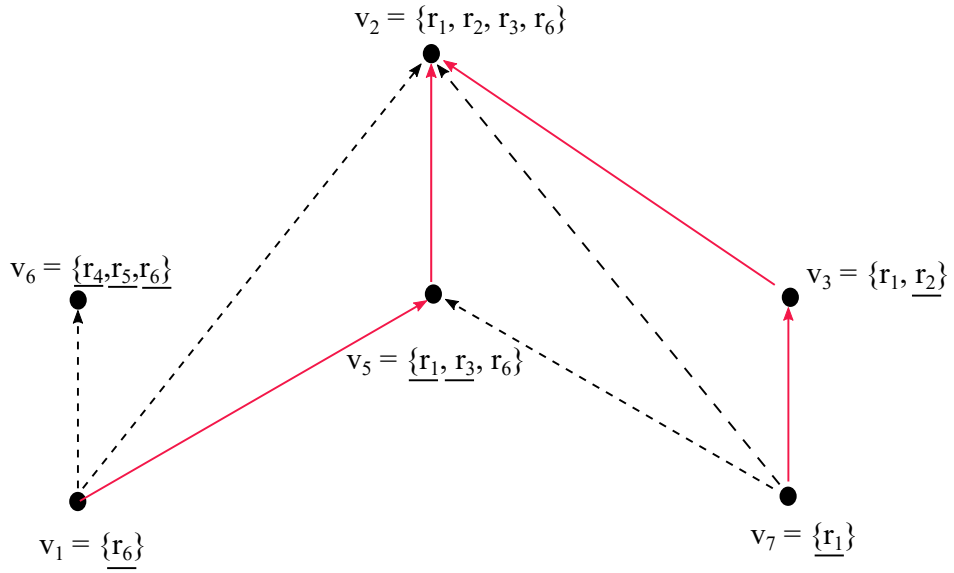
$$\{(v_1, v_5), (v_3, v_2), (v_5, v_2), (v_7, v_3)\}.$$

Te povezave so na Sliki 2 označene z rdečo barvo. Na podlagi teh povezav lahko določimo nepokrite elemente množice B. V točki, v katero ne prihaja nobena izbrana vhodna povezava, so vsi elementi iz nosilca ustreznega stolpca nepokriti. Nato pogledamo izbrane povezave in prek njih določimo nepokrite elemente. Na Sliki 2 so nepokriti elementi podčrtani. Če si na Sliki 2 ogledamo točko v_1, opazimo, da ne obstaja povezava, s katero lahko dosežemo to točko, zato so vsi elementi v nosilcu nepokriti (v našem primeru je nepokrit element r_6). Iz točke v_1 najdemo izbrano povezavo do točke



Slika 1: Digraf $D_{M'}$.

v_5. V tej točki nosilec $\text{supp}_{M'}(c_5)$ vsebuje elemente r_1, r_3 in r_6. Zaradi povezave, sta v tej točki nepokrita elementa samo r_1 in r_3.



Slika 2: Vejitev B v digrafu $D_{M'}$ in nepokriti pari glede na B.

Iz vejitve na Sliki 2 lahko naredimo razcep vrstic matrike M'. Razcep vrstic matrike M' dobimo iz nepokritih parov v vejitvi. Vsaka vrstica predstavlja nepokrit par (r_i, v_j), pri čemer se element r_i nahaja v točki v_j.

Vrstični razcep M'' matrike M', ki ustreza vejitvi B:

$$M'' = \begin{array}{c} \\ (r_1, v_5) \\ (r_1, v_7) \\ (r_2, v_3) \\ (r_3, v_5) \\ (r_4, v_6) \\ (r_5, v_6) \\ (r_6, v_1) \\ (r_6, v_6) \end{array} \begin{array}{c} \begin{array}{ccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 \end{array} \\ \left(\begin{array}{ccccccc} 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right) \end{array}$$

Razcep vrstic dobimo tako, da za vsako vrstico (r_i, v_j) zapišemo enice na tistih stolpcih c, katerih nosilec je v digrafu (V, B) dosegljiv iz točke v_j (glej definicijo 2.7). Za vse $i \in \{1, \ldots, 6\}$ razcep vrstice r_i tvori množica R_i vrstic, indeksiranih z (glede na vejitev B) nepokritimi pari oblike (r_i, v_j).

Če pogledamo Sliko 2, na kateri najdemo vejitev digraf $D_{M'}$, najdemo nepokrit element r_1 v točkah v_5 in v_7, zato se vrstica r_1 matrike M' razcepi na dve vrstici. Enice v matriki določimo na podlagi vejitve. Iz točke v_5 imamo izbrano povezavo (v_5, v_2), zato zapišemo enico pri c_5 kot tudi pri c_2. Iz točke v_7 imamo izbrano povezavo (v_7, v_3). Zato zapišemo enico pri c_7 kot tudi pri c_3. Ker pa imamo iz tega vozlišča (v_3) izbrano povezavo (v_3, v_2), zapišemo pri tem razcepu vrstice enico tudi pri c_2.

Če smo pravilno razcepili vrstice, potem moramo, če skupaj združimo vse vrstice iz množice R_i, dobiti vrstico r_i v enaki obliki, kot je zapisana v matriki M'.

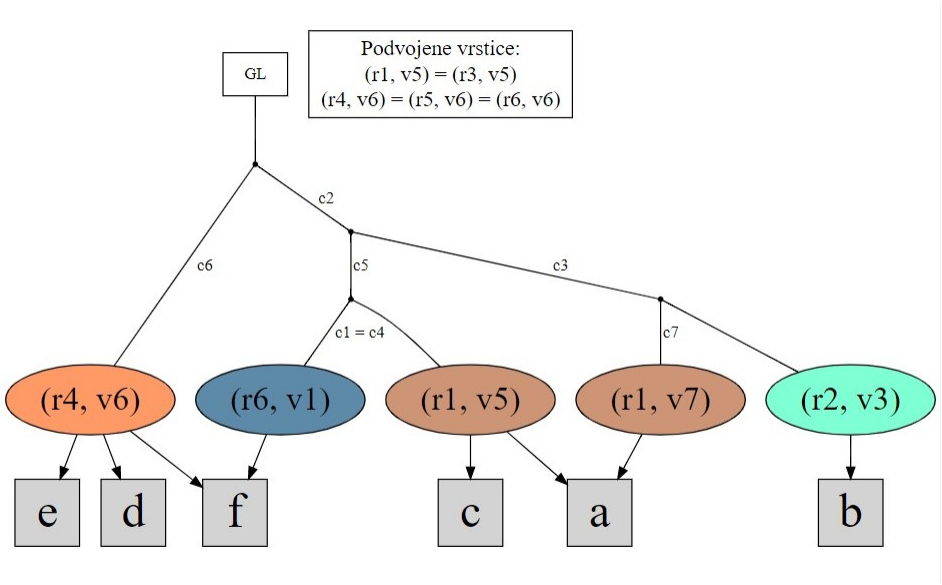
Spomnimo, minimalno število vrstic v razcepu matrike M' označimo z $\gamma(M')$. V zgornjem primeru je $\gamma(M') = 8$.

3.3 Izris filogenetskega drevesa

Za izris drevesa smo ustvarili datoteko .dot, kot so to storili v članku [11]. V tej datoteki so shranjena imena razcepljenih vrstic in mutacij ter podatki o drevesu. Za grafičen prikaz drevesa smo uporabili prosto dostopen program Graphviz [18].

Filogenetsko drevo pridobljeno iz vrstičnega razcepa matrike M' iz primera 3.4 je prikazano na Sliki 3.

Različne vrstice matrike M'' prikazujejo liste filogenetskega drevesa. Te so prikazane v ovalnih, barvnih vozliščih. Kvadratna, siva vozlišča pa predstavljajo originalne vzorce. Puščice označujejo sestavo izvirnih vzorcev v smislu razcepljenih vrstic. Oznake c_i na vejah označujejo mutacije v stolpcih. Vsak vzorec označuje eno vrstico matrike



Slika 3: Filogenetsko drevo, ki ustreza razcepu M'', podanemu na primeru 3.4.

M: vzorec a ustreza vrstici r_1, vzorec b vrstici r_2, c vrstici r_3, d vrstici r_4, e ustreza vrstici r_5 in vzorec f ustreza vrstici r_6. Iz drevesa razberemo, da imamo dva mešana vzorca, in sicer vzorec a in vzorec f.

4 Rezultati

4.1 Simulacija podatkov

Simulirane podatke smo ustvarili na enak način, kot so to storili v članku [11] (z upoštevanjem pristopa v [3,15]). S pomočjo Prüferjevega algoritma [16] smo generirali naključna drevesa z $n = 10$ vozlišči. Tem vozliščem smo nato naključno dodelili $p = 100$ mutacij. Pri tem smo zagotovili, da ima vsako vozlišče vsaj eno mutacijo. Nato smo naključno generirali še drevesa z $n = 20$ vozlišči ter s $p = 200$ mutacijami in z $n = 30$ vozlišči ter s $p = 300$ mutacijami. Potem smo vsakemu vozlišču naključno dodelili velikost populacije celic med 100 in 200. V naslednjem koraku smo iz generiranega drevesa ustvarili m vzorcev. Vsak vzorec je naključno sestavljen iz 2–4 vozlišč generiranega drevesa in vključuje vse celice in mutacije izbranih vozlišč. To smo naredili tako, da smo najprej naključno izbrali število vozlišč med števili 2, 3 in 4, nato pa samo naključno izbrali toliko vozlišč izmed vseh vozlišč generiranega drevesa.

V naslednjem koraku smo definirali matrike U, B in F. Stolpci matrike U prikazujejo vozlišča drevesa, vrstice pa vzorce. Vrednost matrike $U \in \mathbb{R}^{m\times n}$ v vrstici r_i in stolpcu n_j je po definiciji enaka deležu celic klona n_j izmed vseh celic v vzorcu r_i.

Matriko $B \in \{0,1\}^{c\times n}$ dobimo iz naključno generiranega drevesa. Vrednost matrike v vrstici n_i in stolpcu n_j je enaka 1, če in samo če je $n_i = n_j$ ali če je vozlišče n_i v drevesu potomec vozlišča n_j.

Matrika $F \in \mathbb{R}^{m\times n}$ predstavlja vrednosti VAF. To matriko izračunamo tako, da velja

$$F = \frac{1}{2}UB\,.$$

Matrika F vsebuje realne VAF vrednosti vseh mutacij vsakega klona.

Za boljšo simulacijo podatkov smo še dodatno simulirali odčitke, kot so jih naredili v raziskavi avtorja El-Kebir, 2015 [3]. Glede na parameter $a \in \{100, 1000, 10000\}$, ki določa pokritost odčitkov, določimo število odčitkov y_{r_i}, ki vsebujejo mutacijo m_j v vzorcu r_i kot $y_{r_i,m_j} \sim \text{Poiss}(a)$, torej kot nenegativno celoštevilsko slučajno spremenljivko, porazdeljeno glede na Poissonovo porazdelitev s parametrom a. Nato določimo število odčitkov, ki vsebujejo alel variante, in sicer kot $x_{r_i,m_j} \sim \text{Binomial}(y_{r_i,m_j}, F_{r_i,m_j})$, torej kot celoštevilsko nenegativno slučajno spremenljivko, porazdeljeno binomsko s parametroma y_{r_i,m_j} in F_{r_i,m_j}. Število odčitkov, ki vsebujejo referenčni alel, je $y_{r_i,m_j} -$

x_{r_i,m_j}.

Vrednosti, ki jih dobimo, tako da za vsak par (r_i, m_j) izračunamo $x_{r_i,m_j}/y_{r_i,m_j}$ so šumne vrednosti VAF, ki se uporabljajo za vhodni podatek algoritma.

Za vsako skupino generiranih dreves m in vsako izbrano pokritost odčitkov a smo naključno generirali 100 dreves.

Naključno generirana drevesa smo nato primerjali z drevesi, izračunanimi s pomočjo algoritma, predstavljenega v razdelku 3. Naj T_1 označuje simulirano drevo in T_2 drevo, ki nam ga izračuna naš algoritem na vhodni matriki vrednosti $x_{r_i,m_j}/y_{r_i,m_j}$. Kvaliteto izračunanih dreves smo ovrednotili s pomočjo t.i. *AD* parov. Če je prišlo do dveh mutacij v dveh različnih klonih in je bil en klon prednik drugega, potem imata ti dve mutaciji odnos prednik–potomec (ang. ancestor–descendant, *AD*). Natančneje: glede na drevo $T \in \{T_1, T_2\}$ in dve mutaciji m_i in m_j v klonih n_i in n_j pravimo, da je m_i *prednik* m_j (oz. ekvivalentno, da je m_j potomec m_i), če $n_i \neq n_j$ in je n_i prednik n_j v drevesu T. *AD par* je definiran kot urejen par (m_i, m_j) takšnih mutacij, da je m_i prednik m_j. Upoštevajte, da dve mutaciji v istem vozlišču nista *AD* par. Kvaliteto izračunanega drevesa smo ovrednotili tako, da smo število *AD* parov drevesa T_1, ki so tudi *AD* pari za drevo T_2, delili s številom *AD* parov drevesa T_1.

Rezultate našega programa smo primerjali z drugim orodjem, in sicer z orodjem LICHeE [15]. Za izračun *AD* parov iz drevesa, ki ga vrne LICHeE, smo uporabili orodje za vrednotenje podatkov iz članka [11, 13]. To orodje sprejme vrednosti VAF, podatke izračunanega drevesa in podatke iz simuliranega drevesa. Nato nam vrne delež *AD* parov iz izračunanega drevesa v simuliranem drevesu. LICHeE deluje tako, da mutacije najprej razvrsti v skupine, kjer vsaka skupina vsebuje mutacije, odkrite v isti podskupini vzorcev. Nato se mutacije v vsaki skupini nadalje združijo glede na njihovo podobnost vrednosti VAF. Če je v skupini samo ena taka mutacija, jo program odstrani. Potem LICHeE zgradi evolucijsko omrežje z omejitvami (ang. evolutionary constraint network). Omrežje je acikličen digraf, kjer vsako vozlišče ustreza skupku SSNV (razen korena, ki predstavlja germalno linijo (ang. germ line)), vsaka povezava med dvema vozliščema $u \to v$ pa pomeni, da bi lahko vozlišče u bilo evolucijski predhodnik vozlišča v. Povezava $u \to v$ se doda le, če vrednosti VAF vozlišč u in v izpolnjujeta določeno množico omejitev. Omejitve zagotavljajo, da bo omrežje aciklično. Na koncu LICHeE poišče vpeta drevesa v omrežju, ki bi lahko bila obstoječa filogenetska drevesa.

Za analizo simuliranih podatkov smo obravnavali naslednji način določitve vrednosti parametrov t_0 in t_1. Izberemo tri pozitivna števila a, b, c in interval $[0, 1]$ razdelimo na tri podintervale I_1, I_2, I_3 v razmerju dolžin $a : b : c$ in vrednosti t_0 in t_1 določimo kot desni krajišči intervalov I_1 in I_2. Na primer če je $(a, b, c) = (1, 5, 100)$ in imamo razmerje $1 : 5 : 100$, se za neničelne vrednosti VAF, ki so manjše od $1/106$, v matriko M zapiše vrednost 0, za vrednosti, ki so v intervalu med $1/106$ in $5/106$, se zapiše

vrednost 2, za vrednosti, ki so večje od 5/106, pa se v matriki M zapiše vrednost 2.

Razmerje vpliva na to, kolikšen delež ničel, enic in dvojk bo v začetni matriki M. Na obeh programih smo preizkusili naslednja razmerja: $1:1:1$, $1:3:9$, $1:3:20$, $1:5:100$, $1:7:100$, $1:8:100$, $1:9:100$, $1:10:95$, $1:10:100$, $1:10:191$, $3:50:950$, $3:70:530$ in $3:70:927$. Za podrobno obravnavo smo izbrali dve razmerji, in sicer $1:5:100$ in $1:7:100$. Ti dve razmerji smo izbrali zato, ker program LICHeE pri teh razmerjih izračuna relativno veliko dreves, pri čemer pri drugih razmerjih, ki smo jih preizkusili, veliko dreves ne izračuna, kljub temu da ta obstajajo.

Iz zgoraj opisane funkcijske zveze med določenim razmerjem $a:b:c$ in parametroma t_0 in t_1 sledi, da število dvojk v matriki M, izračunani glede na razmerje $1:5:100$, ne bo presegalo števila dvojk v matriki M, izračunani glede na razmerje $1:7:100$. Pričakujemo, da bo problem rekonstrukcije evolucij tumorskih vzorcev v povprečju računsko zahtevnejši za matrike z večjim deležem dvojk.

Rezultati rekonstrukcije simuliranih dreves so prikazani v spodnjih tabelah, v katerih so zapisane povprečne vrednosti za 100 dreves za vsako kombinacijo parametrov ter za vsako razmerje posebej. Za vsako fiksno število vozlišč $n \in \{10, 20, 30\}$ smo drevesa generirali enkrat in te podatke uporabili na dveh različnih razmerjih. Z m označujemo število vzorcev, ki so zajeti, z a označimo parameter, ki določa pokritost odčitkov, s povp. je označeno povprečje, s SD standardni odklon in s ŠND število primerov, v katerih metoda LICHeE ni izračunala drevesa.

V Tabeli 2 je prikazana primerjava med MIPUP+ in LICHeE za 10 vozlišč in 100 mutacij. Za razmerje $1:5:100$ je MIPUP+ v povprečju boljši za 10 kombinacij parametrov od 12, LICHeE pa ima boljše povprečje za kombinaciji parametrov $m = 5$ in $a \in \{100, 1000\}$. LICHeE v povprečju ni izračunal po 6,75 dreves (od 100), pri čemer MIPUP+ najde drevo za vsak primer. Za razmerje $1:7:100$ je MIPUP+ v povprečju sedemkrat boljši od LICHeE, LICHeE pa petkrat, pri čemer LICHeE v povprečju ni izračunal po 6,58 dreves.

V Tabeli 3 najdemo primerjavo med MIPUP+ in LICHeE za 20 vozlišč in 200 mutacij. Za razmerje $1:5:100$ je MIPUP+ v povprečju boljši za vse kombinacije parametrov. Poleg tega LICHeE v povprečju ni izračunal po 10,33 dreves, pri čemer MIPUP+ najde drevo za vsak primer. Za razmerje $1:7:100$ je MIPUP+ v osmih primerih (od 12) v povprečju boljši od LICHeE, pri čemer LICHeE v povprečju ni izračunal po 10,25 dreves.

V Tabeli 4 je zapisana primerjava med MIPUP+ in LICHeE za 30 vozlišč in 300 mutacij. Za razmerje $1:5:100$ ima MIPUP+ v primerjavi z LICHeE boljše povprečje za vse kombinacije parametrov, prav tako ima MIPUP+ boljše povprečje in standardni odklon. LICHeE v povprečju ni izračunal po 16,08 dreves, pri čemer MIPUP+ najde drevo za vsak primer. Za razmerje $1:7:100$ je MIPUP+ v devetih primerih (od 12)

Tabela 2: Delež AD parov za 10 vozlišč in 100 mutacij

1 : 5 : 100

		MIPUP+		LICHeE		
m	a	povp.	SD	povp.	SD	ŠND
5	100	0,651	0,222	**0,670**	0,181	3
	1000	0,726	0,165	**0,727**	0,184	7
	10000	**0,707**	0,193	0,706	0,198	3
10	100	**0,780**	0,213	0,761	0,131	11
	1000	**0,866**	0,120	0,835	0,155	3
	10000	**0,859**	0,150	0,858	0,138	8
15	100	**0,776**	0,221	0,728	0,126	8
	10000	**0,906**	0,114	0,890	0,108	6
	100000	**0,911**	0,109	0,896	0,117	7
20	100	**0,817**	0,204	0,731	0,132	12
	1000	**0,949**	0,068	0,928	0,073	7
	10000	**0,934**	0,083	0,911	0,097	6
		0,824	0,155	0,803	0,137	6,75

1 : 7 : 100

		MIPUP+		LICHeE		
m	a	povp	SD	povp	SD	ŠND
5	100	0,602	0,226	**0,665**	0,187	3
	1000	0,726	0,165	**0,727**	0,184	7
	10000	**0,707**	0,193	0,706	0,198	3
10	100	0,659	0,246	**0,760**	0,132	10
	1000	**0,863**	0,123	0,835	0,155	3
	10000	**0,859**	0,150	0,858	0,138	8
15	100	0,631	0,249	**0,726**	0,129	8
	1000	**0,898**	0,140	0,891	0,108	6
	10000	**0,911**	0,109	0,896	0,117	7
20	100	0,622	0,241	**0,729**	0,132	11
	1000	**0,938**	0,089	0,928	0,073	7
	10000	**0,934**	0,083	0,911	0,097	6
		0,779	0,168	0,803	0,137	6,58

Tabela 3: Delež AD parov za 20 vozlišč in 200 mutacij

1 : 5 : 100

		MIPUP+		LICHeE		
m	a	povp.	SD	povp.	SD	ŠND
5	100	**0,543**	0,161	0,518	0,155	7
	1000	**0,573**	0,144	0,561	0,167	5
	10000	**0,550**	0,148	0,547	0,155	0
10	100	**0,708**	0,160	0,675	0,139	11
	1000	**0,760**	0,132	0,740	0,126	10
	10000	**0,755**	0,130	0,747	0,128	8
15	100	**0,742**	0,188	0,737	0,105	24
	1000	**0,876**	0,099	0,858	0,100	8
	10000	**0,870**	0,085	0,855	0,092	11
20	100	**0,782**	0,156	0,751	0,118	19
	1000	**0,902**	0,080	0,886	0,084	10
	10000	**0,909**	0,076	0,896	0,085	11
		0,748	0,130	0,731	0,121	10,33

1 : 7 : 100

		MIPUP+		LICHeE		
m	a	povp.	SD	povp.	SD	ŠND
5	100	0,496	0,178	**0,515**	0,154	7
	1000	**0,572**	0,144	0,561	0,167	5
	1000	**0,550**	0,148	0,547	0,155	0
10	100	0,589	0,203	**0,676**	0,141	11
	1000	**0,756**	0,139	0,740	0,126	10
	10000	**0,755**	0,130	0,747	0,128	8
15	100	0,589	0,230	**0,733**	0,105	24
	1000	**0,870**	0,106	0,858	0,100	8
	10000	**0,871**	0,085	0,855	0,092	11
20	100	0,548	0,238	**0,745**	0,119	18
	1000	**0,887**	0,105	0,886	0,084	10
	10000	**0,909**	0,076	0,896	0,085	11
		0,699	0,149	0,730	0,121	10,25

v povprečju boljši od LICHeE, pri čemer LICHeE v povprečju ni izračunal po 16,08 dreves.

Povzemimo izsledke. Opazimo, da ima LICHeE povprečje vseh povprečij in standardni odklon pri razmerju $1 : 7 : 100$ boljši od MIPUP+, kljub temu da ima MIPUP+ večkrat boljše povprečje. Ta razlika je posledica dejstva, da je standardna deviacija v povprečju večja pri programu MIPUP+. V primerih kombinacij parametrov, kjer je LICHeE v povprečju boljši, ima MIPUP+ bistveno višjo vrednost standardne deviacije kot pri ostalih kombinacijah parametrov. Podrobnejša analiza podatkov je razkrila, da je za vsako kombinacijo parametrov kvaliteta drevesa, ki ga izračuna MIPUP+, za določeno število simuliranih dreves (okrog 5–10%) zelo slaba, in sicer v razponu 10–30%, pri programu LICHeE pa tega ni bilo opaziti. Predvidevamo, da je to povezano z dejstvom, da program LICHeE problem rekonstrukcije drevesa poenostavi tako, da odstrani stolpce matrike, ki ustrezajo skupinam z eno samo mutacijo, in na ta način zmanjša prostor možnih filogenetskih dreves. Program MIPUP+ takih poenostavitev nima, kar pomeni večjo verjetnost za ohranitev osamelcev v vhodnih podatkih.

Zanimala nas je tudi primerjava programov na podmnožici generiranih vhodnih podatkov, na kateri oba programa izračunata filogenetsko drevo. Morda bi sprva pričakovali, da program LICHeE drevesa ni uspel izračunati na vhodnih podatkih, pri katerih je bila kvaliteta rekonstrukcije za program MIPUP+ relativno slaba. Vendar temu ni tako. Vhodne podatke, ki so se izkazali za najzahtevnejše za program MIPUP+, je namreč LICHeE poenostavil in tako dosegel kvalitetno rekonstrukcijo. Po drugi strani pa je bila rekonstrukcija dreves za primere, v katerih LICHeE drevesa ni uspel izračunati, za program MIPUP+ zelo dobra (nad 85%). Izpostaviti velja tudi dejstvo, da je v določenih vhodnih primerih program MIPUP+ uspel doseči najvišjo možno kvaliteto rekonstrukcije drevesa (torej 100%), pri programu LICHeE pa tega nismo zasledili.

Skupna obema programoma je lastnost, da povprečna vrednost kvalitete izračunanega drevesa narašča z večjim številom vzorcev (m) in pada z večjim številom vozlišč drevesa (n). Pri obeh programih je zaznati tudi velik skok v kvaliteti rešitev, ko se vrednost parametra a, ki določa pokritost odčitkov, poveča iz 100 na 1000, medtem ko opaznih razlik v kvaliteti rekonstruiranih rešitev, če primerjamo vrednosti parametra $a = 1000$ in $a = 10000$, nismo zasledili.

Tabela 4: Delež AD parov za 30 vozlišč in 300 mutacij

1 : 5 : 100

		MIPUP+		LICHeE		
m	a	povp.	SD	povp.	SD	ŠND
5	100	**0,459**	0,147	0,450	0,140	8
	1000	**0,492**	0,138	0,472	0,136	6
	10000	**0,504**	0,119	0,482	0,131	5
10	100	**0,635**	0,157	0,623	0,123	16
	1000	**0,713**	0,106	0,691	0,116	11
	10000	**0,700**	0,121	0,667	0,127	9
15	100	**0,713**	0,135	0,673	0,100	23
	1000	**0,800**	0,100	0,768	0,102	22
	10000	**0,776**	0,128	0,751	0,132	20
20	100	**0,695**	0,187	0,683	0,109	29
	1000	**0,853**	0,087	0,834	0,088	15
	10000	**0,782**	0,072	0,853	0,081	29
		0,708	0,125	0,662	0,115	16,08

1 : 7 : 100

		MIPUP+		LICHeE		
m	a	povp	SD	povp	SD	ŠND
5	100	0,427	0,166	**0,449**	0,141	8
	1000	**0,493**	0,138	0,472	0,136	6
	1000	**0,504**	0,119	0,482	0,131	5
10	100	**0,636**	0,159	0,623	0,123	16
	1000	**0,713**	0,106	0,691	0,116	11
	10000	**0,700**	0,121	0,667	0,127	9
15	100	0,549	0,181	**0,673**	0,100	23
	1000	**0,793**	0,103	0,768	0,102	22
	10000	**0,776**	0,128	0,751	0,132	20
20	100	0,521	0,220	**0,683**	0,109	29
	1000	**0,840**	0,104	0,834	0,088	15
	10000	**0,882**	0,072	0,853	0,081	29
		0,653	0,135	0,662	0,115	16,08

4.2 Realni podatki

Za analizo na realnih podatkih smo uporabili dva javno dostopna podatkovna seta: svetlocelični karcinom ledvic (ccRCC) [5] in serozni karcinom visokega gradusa (HGSC) [2]. Pri vseh posameznikih v obeh setih smo določili praga $t_0 = 0{,}00943$ in $t_1 = 0{,}047$. Vrednosti t_0 in t_1 se ujemata z vrednostma t_0 in t_1, ki smo ju uporabili za simulirane podatke glede na razmerje $1 : 5 : 100$. Nato smo vse realne podatke zagnali v programih MIPUP+ in LICHeE. V drevesu, ki ga vrne MIPUP+, zadnja vrsta kvadratnih sivih vozlišč predstavlja originalne vzorce. Ovalna, barvna vozlišča predstavljajo vrstice v razcepu vrstic. Puščice označujejo sestavo izvirnih vzorcev v smislu razcepljenih vrstic.

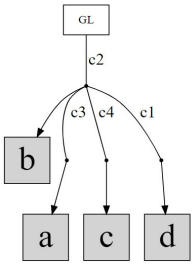
4.2.1 Svetlocelični karcinom ledvic (ccRCC)

V študiji ccRCC so Gerlinger s sodelavci [5] potrdili 602 nesinonimnih nukleotidnih substitucij in indelov iz več vzorcev osmih posameznikov (EV0003, EV0005, EV0006, EV0007, RK26, RMH002, RMH004 in RMH008). Za določitev podklonov, uporabljenih pri rekonstrukciji drevesa, so uporabili združevanje posameznega vzorca na podlagi vrednosti VAF. Na naslednjih slikah je prikazana primerjava izrisanih filogenetskih dreves za programa MIPUP+ in LICHeE. Na zgornji levi strani slike je zapisana matrika M, pridobljena iz vrednosti VAF glede na parametra t_0 in t_1. Spodaj levo pa je prikazana matrika M'', ki prikazuje vrstični razcep matrike M', ki ju izračuna MIPUP+ (glej algoritem 1). Zaradi večje preglednosti matrik M' na slikah ne prikazujemo. Po zamenjavi dvojk v matriki M v ničle in enice se lahko zgodi, da se par različnih $\{0, 1, 2\}$-stolpcev v matriki M preslika v par identičnih binarnih stolpcev. Ker so vsi stolpci matrike M'' paroma različni, je lahko število stolpcev matrike M'' strogo manjše od števila stolpcev matrike M. Če je stolpcev matrike M, ki ustrezajo stolpcu c_i matrike M'', več, so označeni zaporedoma s $c_i^1, \ldots, c_i^k$. kjer je k število teh stolpcev.

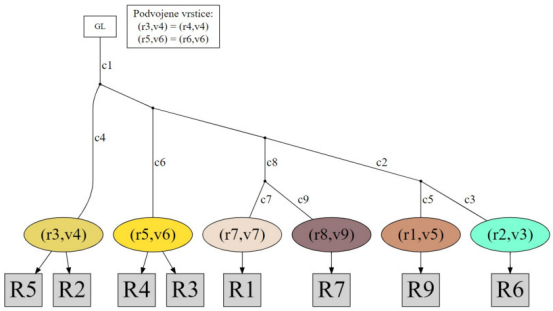
Na Sliki 4 sta poleg matrik M in M'' prikazani drevesi posameznika EV003. Opazimo, da primer ne vsebuje mešanih vzorcev, saj obstaja samo ena povezava iz končnega vozlišča v vsak list obeh dreves. Med drevesoma pride do rahlih odstopanj. V drevesu, ki ga vrne MIPUP+, vzorec R4 vsebuje mutaciji c_1 in c_6, medtem ko vzorec R4 v drevesu programa LICHeE-ja vsebuje samo mutacijo c_1. Zaradi odstranitve mutacije c_8 pride do razlike tudi v vzorcu R1 in R7, in sicer ta dva vzorca v drevesu programa LICHeE ne vsebujeta mutacije c_8.

Slika 5 prikazuje filogenetski drevesi posameznika EV005 ter matriki M in M''. Iz matrike M'' razberemo, zakaj se stolpec c_6 ne pojavi na nobeni povezavi obeh dreves. Ta stolpec ne vsebuje nobene mutacije. Med drevesi pride do razlik, saj se stolpec c_3 v matrikah obeh programov preslika v različna stolpca po odpravi dvojk v binarna števila. Prav tako LICHeE odstrani stolpec c_2. Do razlike pride pri vzorcih R1, R2, R3, R4 in

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \end{array} \begin{array}{c} \begin{array}{ccccccccccc} c_1 & c_2 & c_3^1 & c_4^1 & c_4^2 & c_5 & c_3^2 & c_6 & c_7 & c_8 & c_9 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

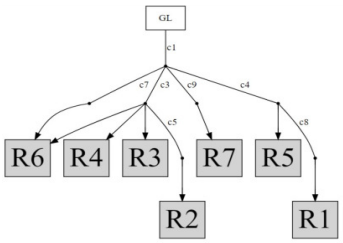


$$M'' = \begin{array}{c} \\ (r_1, v_5) \\ (r_2, v_3) \\ (r_3, v_4) \\ (r_4, v_4) \\ (r_5, v_6) \\ (r_6, v_6) \\ (r_7, v_7) \\ (r_8, v_9) \end{array} \begin{array}{c} \begin{array}{ccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

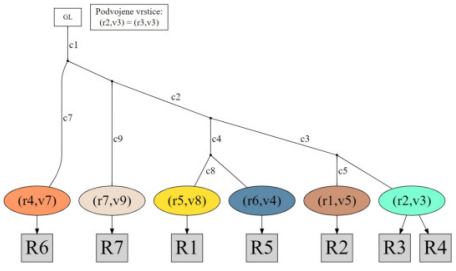


Slika 4: Primer drevesa za EV003 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \end{array} \begin{array}{c} \begin{array}{ccccccccccc} c_1^1 & c_1^2 & c_2 & c_3 & c_4 & c_5 & c_6^1 & c_7 & c_8 & c_6^2 & c_9 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$



$$M'' = \begin{array}{c} \\ (r_1, v_5) \\ (r_2, v_3) \\ (r_3, v_3) \\ (r_4, v_7) \\ (r_5, v_8) \\ (r_6, v_4) \\ (r_7, v_9) \end{array} \begin{array}{c} \begin{array}{ccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

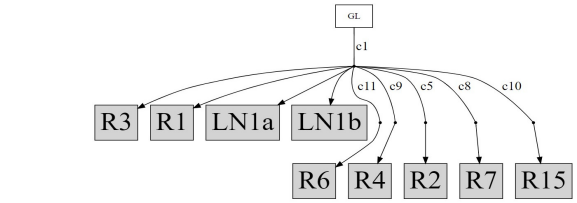


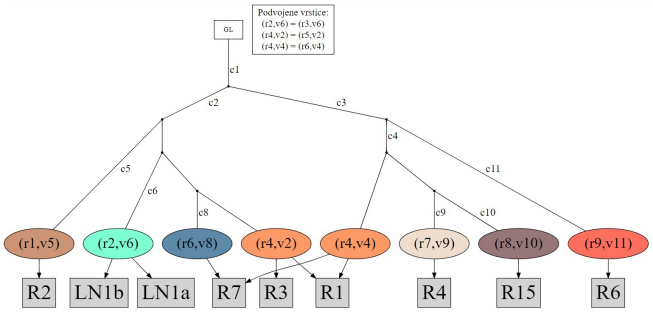
Slika 5: Primer drevesa za EV005 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

R5, saj ti vzorci v drevesu MIPUP+ vsebujejo mutacije iz stolpca c_2. Največja razlika se pojavi pri vzorcu R6, saj ta vzorec pri drevesu, ki ga vrne MIPUP+, ne predstavlja mešanega vzorca, pri čemer vzorec R6 v drevesu, ki ga vrne LICHeE, vsebuje dva različna vzorca.

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \end{array} \begin{array}{c} \begin{array}{ccccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} & c_{11} \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

$$M'' = \begin{array}{c} \\ (r_1, v_5) \\ (r_2, v_6) \\ (r_3, v_6) \\ (r_4, v_2) \\ (r_4, v_4) \\ (r_5, v_2) \\ (r_6, v_4) \\ (r_6, v_8) \\ (r_7, v_9) \\ (r_8, v_{10}) \\ (r_9, v_{11}) \end{array} \begin{array}{c} \begin{array}{ccccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} & c_{11} \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$





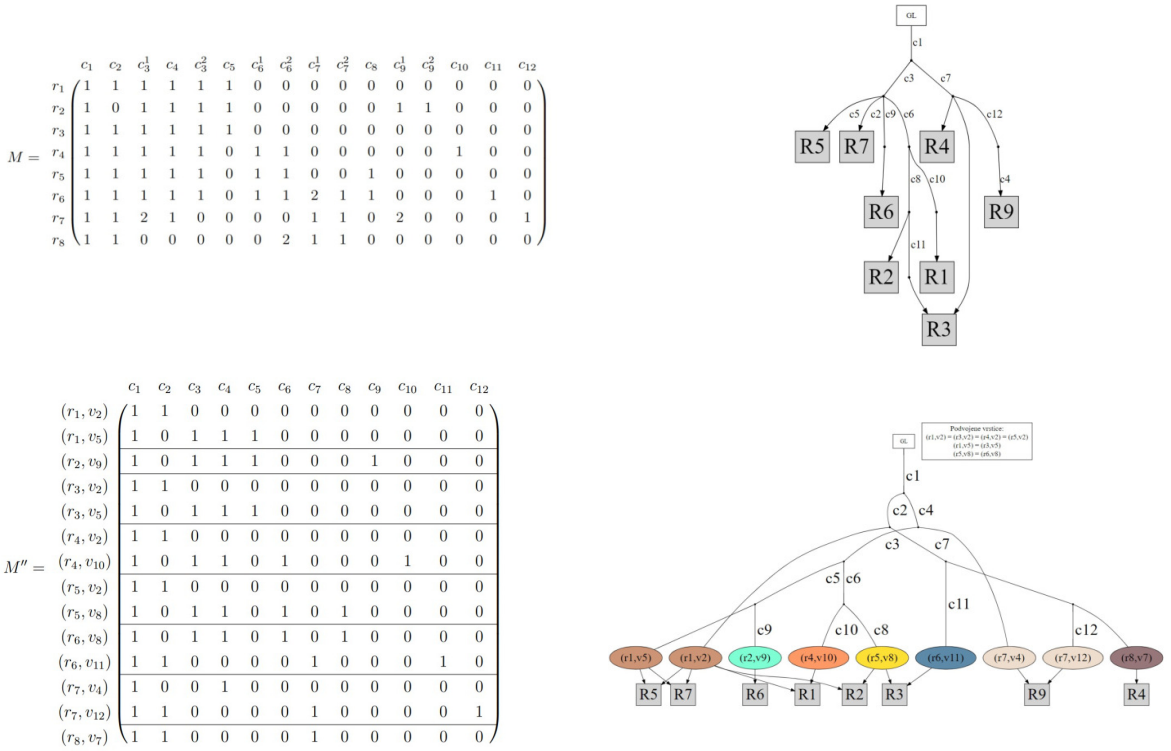
Slika 6: Primer drevesa za EV006 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

Drevesi posameznika EV006 sta prikazani na desni strani Slike 6. Iz slike je razvidno, da drevo, ki ga vrne MIPUP+, vsebuje mešane vzorce za lista R1 in R7, pri čemer drevo, ki ga vrne LICHeE, ne vsebuje mešanih vzorcev. Drevesi sta si precej različni, do tega pride najverjetneje zaradi različne zamenjave dvojk v binarna števila v stolpcih c_3 in c_4.

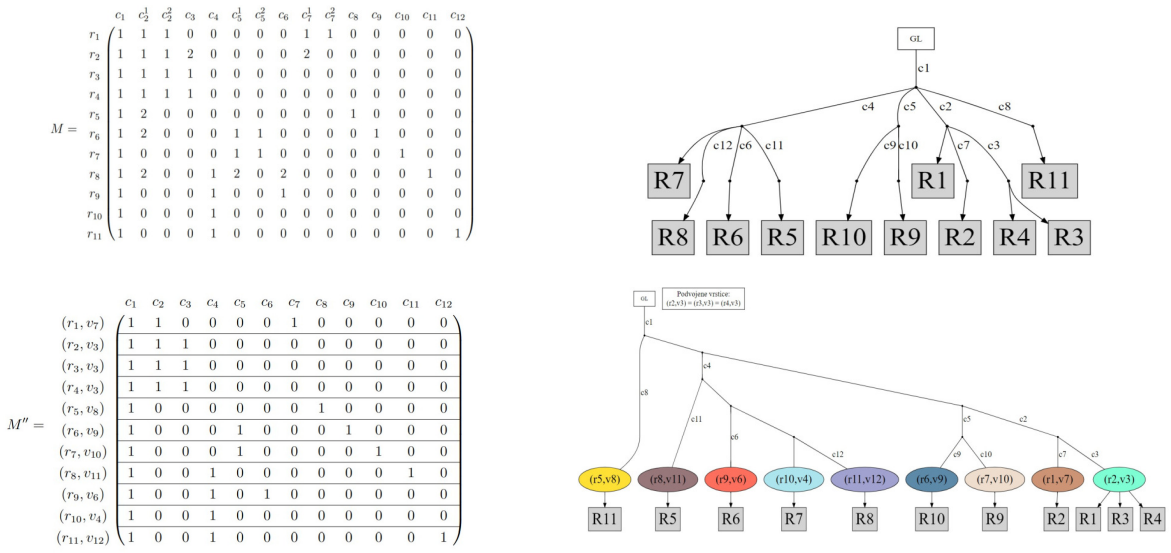
Slika 7 na desni strani prikazuje drevesi posameznika EV007. Opazimo, da drevo, ki ga vrne MIPUP+, vsebuje šest mešanih vzorcev za liste R1, R2, R3, R5, R7 in R9, pri čemer drevo, ki ga vrne LICHeE, vsebuje mešani vzorec samo za list R3. Do tega najverjetneje pride, ker LICHeE stolpce s podobnimi mutacijami združi.

Drevesi posameznika RK26 sta prikazani na desni strani Slike 8. Iz slike razberemo, da drevesi ne vsebujeta mešanih vzorcev. Med drevesi se pojavi majhna razlika, in sicer na drevesu, ki ga vrne LICHeE, vzorec R1 ne vsebuje mutacij iz stolpca c_3, pri čemer drevo, ki ga vrne MIPUP+, te mutacije vsebuje.

Na Sliki 9 sta na desni strani prikazani drevesi posameznika RMH002. Opazimo, da drevo, ki ga vrne MIPUP+, vsebuje mešane vzorce za liste R3, R6 in R7, pri čemer drevo, ki ga vrne LICHeE, ne vsebuje nobenega mešanega vzorca. Do razlik v drevesu pride, ker se stolpca c_2 po odpravi dvojk v enice oz. ničle razlikujeta med seboj, torej programa drugače zamenjata dvojko v binarno število. Omenimo še, da vrstica r_2 matrike M', kot jo izračuna MIPIP+, vsebuje same ničle, kar pomeni, da jo program odstrani, z drugimi besedami, v matriki M'' razcepi v prazno množico vrstic. Posledično je vzorec R2 v drevesu, ki ga vrne MIPUP+, sestavljen iz prazne množice



Slika 7: Primer drevesa za EV007 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

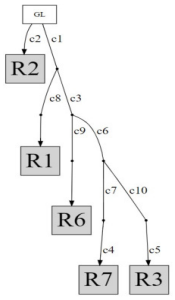


Slika 8: Primer drevesa za RK26 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

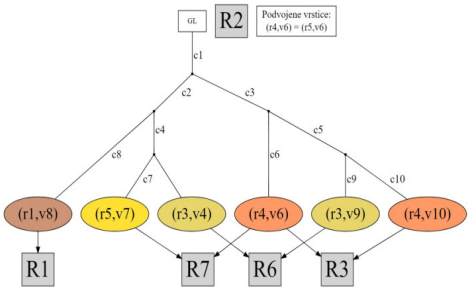
listov in se zato v izrisu pojavi ločeno od drevesa. MIPUP+ vzorec R2 klasificira kot benigen.

Slika 10 prikazuje matriki M in M'' z izračunanima filogenetskima drevesoma posa-

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{array} \begin{array}{c} \begin{array}{ccccccccccc} c_1 & c_2 & c_3^1 & c_3^2 & c_4 & c_5 & c_6 & c_7^1 & c_8 & c_9 & c_{10} & c_7^2 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 2 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

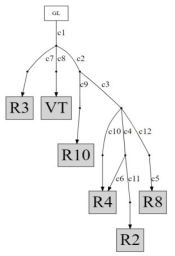


$$M'' = \begin{array}{c} \\ (r_1, v_8) \\ (r_3, v_4) \\ (r_3, v_9) \\ (r_4, v_6) \\ (r_4, v_{10}) \\ (r_5, v_6) \\ (r_5, v_7) \end{array} \begin{array}{c} \begin{array}{cccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

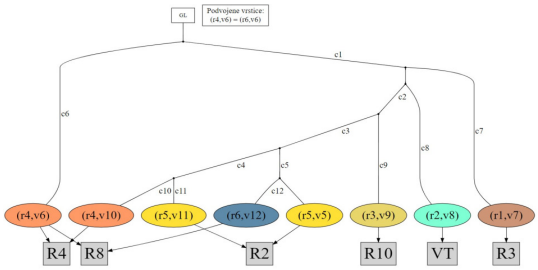


Slika 9: Primer drevesa za RMH002 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{array} \begin{array}{c} \begin{array}{cccccccccccccc} c_1^1 & c_1^2 & c_2^1 & c_2^2 & c_3 & c_4 & c_5 & c_6 & c_7^1 & c_7^2 & c_8 & c_9 & c_{10} & c_{11} & c_{12} \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 2 & 1 & 2 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$



$$M'' = \begin{array}{c} \\ (r_1, v_7) \\ (r_2, v_8) \\ (r_3, v_9) \\ (r_4, v_6) \\ (r_4, v_{10}) \\ (r_5, v_5) \\ (r_5, v_{11}) \\ (r_6, v_6) \\ (r_6, v_{12}) \end{array} \begin{array}{c} \begin{array}{cccccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} & c_{11} & c_{12} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

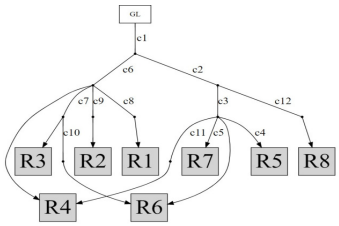


Slika 10: Primer drevesa za RMH004 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

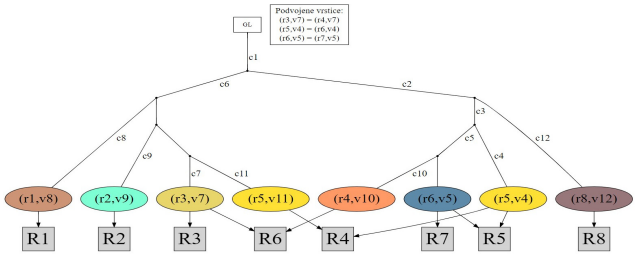
meznika RMH004. Iz slike je razvidno, da drevo, ki ga vrne MIPUP+, vsebuje mešane vzorce za liste R2, R4 in R8, pri čemer drevo, ki ga vrne LICHeE, vsebuje mešani vzorec samo za list R4, druga dva lista ne vsebujeta mešanega vzorca. Vzorec R2 ne vsebuje mutacij iz stolpca c_5 in vzorec R8 ne vsebuje mutacij iz stolpca c_6, kot prikazuje

izračunano drevo programa MIPUP+.

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \end{array} \begin{array}{c} \begin{array}{cccccccccccc} c_1^1 & c_1^2 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} & c_{11} & c_{12} \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$



$$M'' = \begin{array}{c} \\ (r_1, v_8) \\ (r_2, v_9) \\ (r_3, v_7) \\ (r_4, v_7) \\ (r_4, v_{10}) \\ (r_5, v_4) \\ (r_5, v_{11}) \\ (r_6, v_4) \\ (r_6, v_5) \\ (r_7, v_5) \\ (r_8, v_{12}) \end{array} \begin{array}{c} \begin{array}{cccccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} & c_{11} & c_{12} \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$



Slika 11: Primer drevesa za RMH008 za MIPUP+ (desno spodaj) in LICHeE (desno zgoraj).

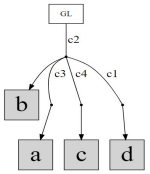
Drevesi posameznika RMH008 sta prikazani na desni strani Slike 11. Opazimo, da drevo, ki ga vrne MIPUP+, vsebuje mešane vzorce za liste R4, R5 in R6, pri čemer drevo, ki ga vrne LICHeE, vsebuje mešani vzorec le za lista R4 in R6. Vzorec R5 na drevesu, ki ga izračuna LICHeE, ne vsebuje mutacij iz stolpca c_5.

4.2.2 Serozni karcinom visokega gradusa (HGSC)

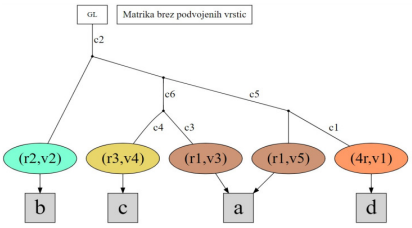
V študijo Bashashati idr. [2] je bilo vključenih šest žensk (case 1, case 2, case 3, case 4, case 5, case 6) s histološko diagnosticiranim seroznim rakom visokega gradusa. Tkivo so dobili iz 31 tumorskih mest v razponu 3–10 vzorcev/bolnika, vključno z dvema lezijama v jajčnikih z enakovredno normalno DNK za vsakega pacienta. S pomočjo tehnologije sekvenciranja naslednje generacije so natančno izmerili klonsko raznolikost HGSC, ki je prisotna ob diagnozi. Njihovi rezultati kažejo, kako vsak HGSC kaže edinstvene evolucijske usmeritve s široko regionalno raznolikostjo mutacij in regij izražanja genov pred posegom v zdravljenje [2]. Na naslednjih slikah sta prikazani matriki M in M'' ter primerjava izrisanih filogenetskih dreves v programih MIPUP+ in LICHeE. Tudi v tem razdelku zaradi večje preglednosti matrik M' na slikah ne prikazujemo. Prav tako, če je stolpcev matrike M, ki ustrezajo stolpcu c_i matrike M'', več, so ti označeni zaporedoma s $c_i^1, \ldots, c_i^k$. kjer je k število teh stolpcev.

Na Sliki 12 sta prikazani drevesi posameznice case 1. Opazimo, da drevo, ki ga vrne MIPUP+, vsebuje mešani vzorec za list a, medtem ko drevo, ki ga vrne LICHeE,

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \end{array} \begin{array}{c} \begin{array}{cccccccccc} c_1^1 & c_2^1 & c_3 & c_4 & c_1^2 & c_5 & c_1^3 & c_2^2 & c_6 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 2 & 1 & 2 & 1 & 1 \\ 0 & 1 & 0 & 0 & 2 & 2 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 2 & 0 \end{pmatrix} \end{array}$$



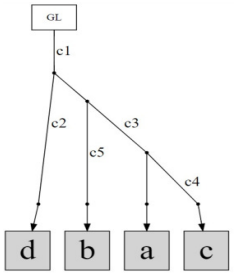
$$M'' = \begin{array}{c} \\ (r_1, v_3) \\ (r_1, v_5) \\ (r_2, v_2) \\ (r_3, v_4) \\ (r_4, v_1) \end{array} \begin{array}{c} \begin{array}{cccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$



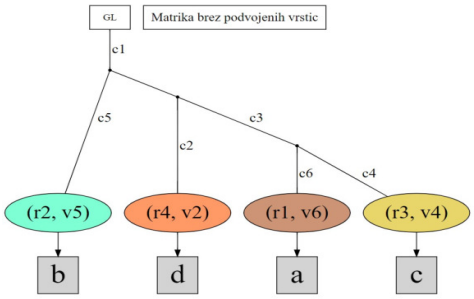
Slika 12: Primer drevesa za case 1 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno).

ne vsebuje mešanih vzorcev. Do odstopanj najverjetneje prihaja zaradi odstranitve stolpcev c_5 in c_6.

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \end{array} \begin{array}{c} \begin{array}{cccccccc} c_1 & c_2^1 & c_3 & c_4^1 & c_5 & c_2^2 & c_4^2 & c_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 2 & 2 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 2 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$



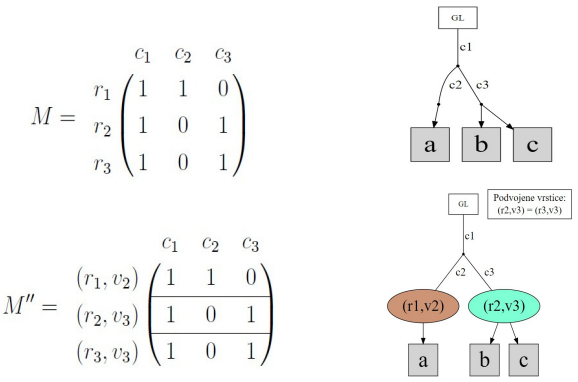
$$M'' = \begin{array}{c} \\ (r_1, v_6) \\ (r_2, v_5) \\ (r_4, v_4) \\ (r_4, v_2) \end{array} \begin{array}{c} \begin{array}{cccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$



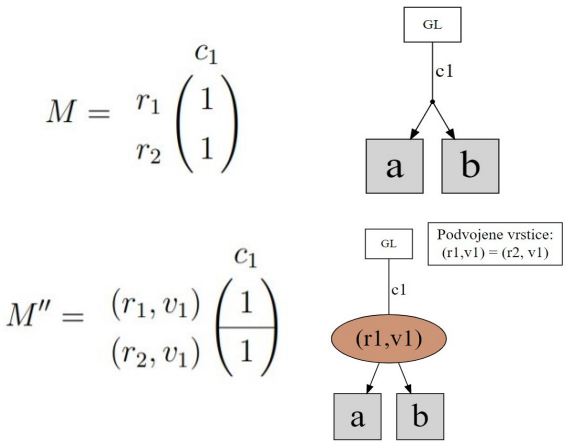
Slika 13: Primer drevesa za case 2 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno).

Drevesa posameznic case 2, case 3 in case 6 so prikazana na desni strani Slik 13, 14 in 15. Opazimo, da sta primera za posameznici case 3 in case 6 posebej preprosta, saj sta matriki M binarni in brezkonfliktni. Tako ni presenetljivo, da sta drevesi za posameznici case 3 in case 6 popolnoma enaki. Iz slik razberemo, da za vsako

posameznico drevesi, ki ju vrneta MIPUP+ in LICHeE, ne vsebujeta mešanih vzorcev. Do rahle razlike prihaja pri posameznici case 2, in sicer v LICHeE odstrani stolpec c_6, zato pride do razlike v vzorcu a.



Slika 14: Primer drevesa za case 3 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno).



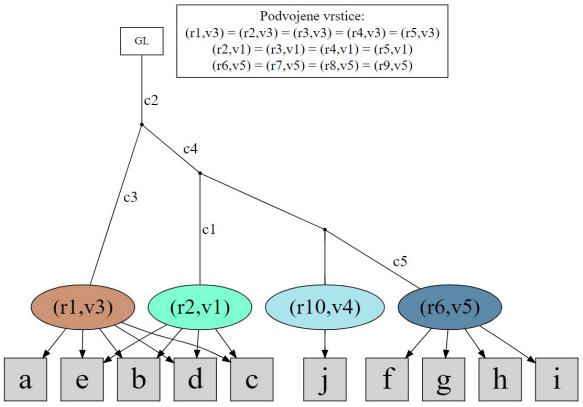
Slika 15: Primer drevesa za case 6 za MIPUP+ (spodaj desno) in LICHeE (zgoraj desno).

Slika 16 na desni strani prikazuje filogenetsko drevo posameznice case 4, ki ga vrne MIPUP+. Za ta primer, s temi parametri, LICHeE ne izračuna drevesa. Opazimo, da drevo vsebuje mešani vzorec za liste b, c, d in e.

Na Sliki 18 sta prikazani drevesi posameznice case 5. Iz slike razberemo, da ima drevo, ki ga vrne MIPUP+, mešane vzorce za vse liste, medtem ko LICHeE ne vsebuje mešanih vzorcev v listu d. Do razlike pride zaradi tega, ker sta stolpca c_4 v obeh programih med seboj različna. Do tega pride zaradi drugačne zamenjave dvojke v binarno število. Poleg tega LICHeE zbriše stolpce c_8, c_{14} in c_{15}.

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \end{array} \begin{array}{c} \begin{array}{cccccc} c_1 & c_2 & c_3^1 & c_4 & c_3^2 & c_5 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 2 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 2 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

$$M'' = \begin{array}{c} \\ (r_1, v_3) \\ (r_2, v_1) \\ (r_2, v_3) \\ (r_3, v_1) \\ (r_3, v_3) \\ (r_4, v_1) \\ (r_4, v_3) \\ (r_5, v_1) \\ (r_5, v_3) \\ (r_6, v_5) \\ (r_7, v_5) \\ (r_8, v_5) \\ (r_9, v_5) \\ (r_{10}, v_4) \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

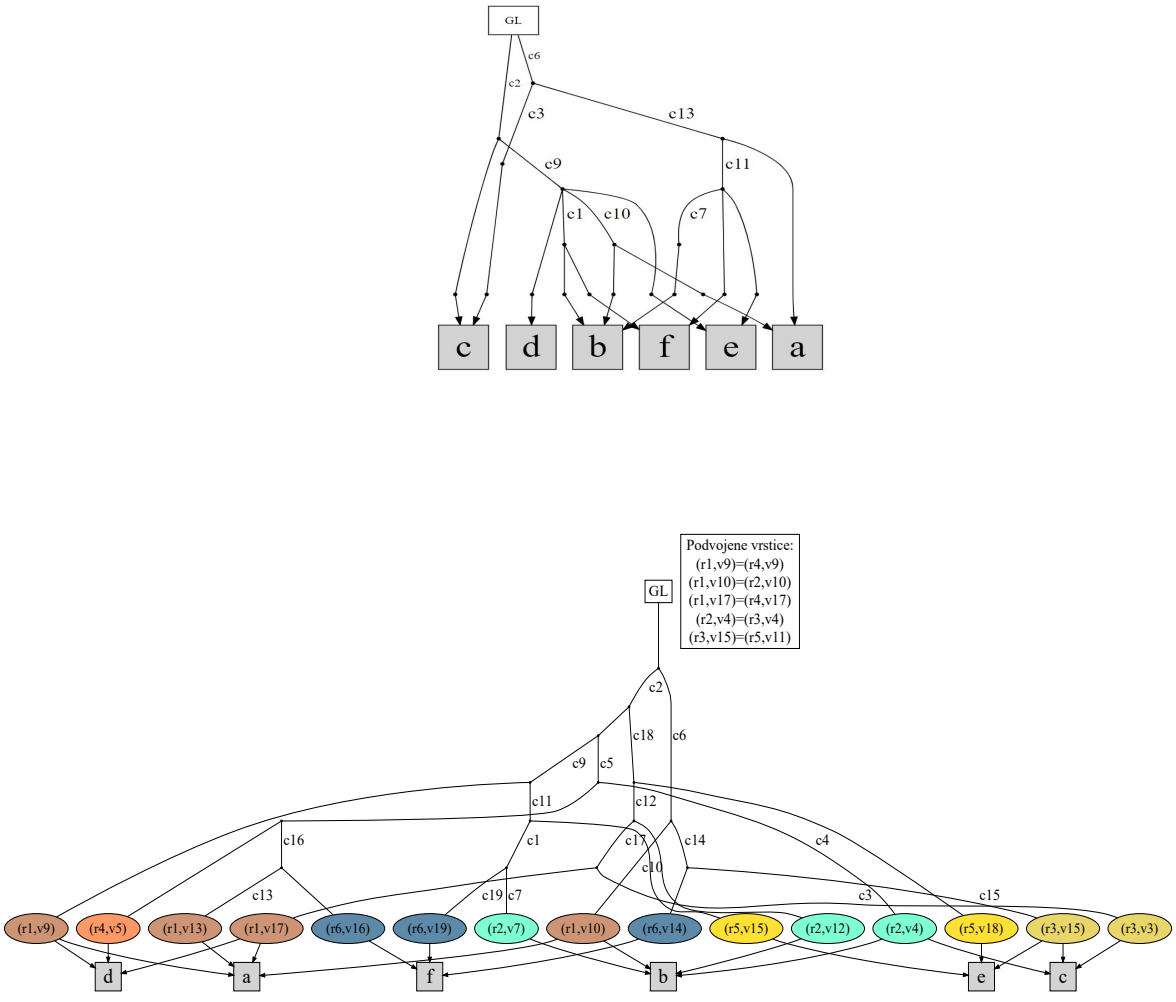


Slika 16: Primer drevesa za case 4 za MIPUP+ (desno).

$$M = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{array} \begin{array}{c} \begin{array}{cccccccccccccccccccccccccc} c_1^1 & c_2 & c_3^1 & c_4 & c_5 & c_6^1 & c_7 & c_8 & c_1^2 & c_9 & c_3^1 & c_1^4 & c_{10} & c_{11}^1 & c_{12} & c_{13}^1 & c_{14} & c_{15} & c_3^2 & c_{16}^1 & c_6^2 & c_1^5 & c_6^3 & c_{17} & c_{18} & c_{13}^2 & c_{19} & c_3^3 & c_{11}^2 & c_{16}^2 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 2 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 2 & 1 & 1 & 1 & 0 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 0 & 2 & 0 & 2 & 0 & 2 & 1 & 2 & 0 & 1 & 2 & 0 & 2 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 2 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 1 & 0 & 0 & 0 & 2 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 2 & 2 & 0 & 2 & 2 & 2 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 2 & 1 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & 2 & 1 & 2 \end{pmatrix} \end{array}$$

$$M'' = \begin{array}{c} \\ (r_1, v_9) \\ (r_1, v_{10}) \\ (r_1, v_{13}) \\ (r_1, v_{17}) \\ (r_2, v_4) \\ (r_2, v_7) \\ (r_2, v_{10}) \\ (r_2, v_{12}) \\ (r_3, v_3) \\ (r_3, v_4) \\ (r_3, v_{15}) \\ (r_4, v_5) \\ (r_4, v_9) \\ (r_4, v_{17}) \\ (r_5, v_{11}) \\ (r_5, v_{15}) \\ (r_5, v_{18}) \\ (r_6, v_{14}) \\ (r_6, v_{16}) \\ (r_6, v_{19}) \end{array} \begin{array}{c} \begin{array}{ccccccccccccccccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} & c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} & c_{17} & c_{18} & c_{19} \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Slika 17: Primer matrik za case 5.



Slika 18: Primer drevesa za case 5 za MIPUP+ (spodaj) in LICHeE (zgoraj).

5 Diskusija

Namen naloge je bil razširitev obstoječega modela MIPUP [11] za problem rekonstrukcije razvoja različnih vrst celic v tumorjih. MIPUP+ natančno in učinkovito rešuje problem rekonstrukcije razvoja različnih vrst celic v tumorjih. Pokazali smo, da MIPUP+ rekonstruira prvotno drevo večinoma bolje kot LICHeE. Prav tako naš program vedno izračuna drevo, medtem ko LICHeE ne najde vedno rešitve, kljub temu da ta obstaja. Program MIPUP+ je v določenih primerih simuliranih podatkov filogenetsko drevo uspel rekonstruirati popolnoma, program LICHeE pa ne. Po drugi strani pa se je LICHeE zaradi drugačnega pristopa (določitve in odstranitve osamelcev) bolje izkazal na določenih simuliranih podatkih, ki so bili za MIPUP+ težavni.

Pri primerjavi programov s simuliranimi podatki smo opazili, da izbrano razmerje ($1 : 5 : 100$ oziroma $1 : 7 : 100$) vpliva na dobljene rezultate iz programa MIPUP+. Če izberemo razmerje, ki ima velik interval za vrednosti VAF, ki se pretvorijo v 2, potem se natančnost AD parov zmanjšuje. Torej, več kot imamo dvojk v matriki M, slabša je natančnost AD parov glede na simulirano drevo. Do tega najverjetneje pride zaradi velikega števila dvojk. Ker imamo veliko število dvojk, je možnih veliko pomožnih binarnih matrik M' in za vsako od njih obstaja optimalen razcep vrstic. Prostor preiskovanja možnih dopustnih rešitev je bistveno večji in zato je tudi večja verjetnost, da optimalna rešitev ne bo enolična. Za izboljšanje tega problema bi morali izračunati vse optimalne rešitve in nato izbrati najboljšo izmed njih.

Opazili smo, da pri obeh programih povprečna vrednost kvalitete izračunanega drevesa narašča z večjim številom vzorcev in da na večjo kvaliteto rekonstrukcije lahko sklepamo pri dovolj veliki pokritosti odčitkov ($a = 1000$).

Čeprav se rezultati MIPUP+ in LICHeE na splošno skladajo, nastane med njima v mnogih primerih veliko majhnih razlik. Če primerjamo filogenije programov MIPUP+ in LICHeE s filogenijami, ki so jih konstruirali avtorji realnih podatkov v [2, 5], zaključimo, da se rezultati programa MIPUP+ na splošno bolj skladajo z izvirnimi filogenijami kot rezultati programa LICHeE.

Poleg tega mora uporabnik za natančen zagon LICHeE uganiti veliko vhodnih parametrov, medtem ko mora uporabnik v primeru MIPUP+ določiti samo praga za pretvorbo vrednosti VAF v matriko 0, 1 in 2. Čas, ki ga za izračun porabita oba programa, MIPUP+ in LICHeE, zelo variira in je odvisen od matrike. Da sta pro-

grama izračunala vse simulirane podatke in rekonstruirala vsa drevesa, za obe razmerji $1 : 5 : 100$ in $1 : 7 : 100$ in za vse možne kombinacije parametrov, je MIPUP+ potreboval 5 ur, medtem ko je LICHeE potreboval več kot 12 ur.

6 Zaključek

V zaključni nalogi smo spoznali problem rekonstrukcije razvoja različnih vrst celic v tumorjih. Prav tako smo uspešno implementirali in ovrednotili razširitev programa MIPUP [11]. Kvaliteta rekonstruiranih rešitev je bila ovrednotena glede na to, do kolikšne mere je izračun uspel rekonstruirati dejansko evolucijo, in preverjena s simulacijo podatkov. Pokazali smo, da v povprečju MIPUP+ deluje bolje v primerjavi s programom LICHeE. V nadaljnjem delu bi bilo smiselno program izboljšati tako, da ne obravnava vseh optimalnih rešitev enako, ampak jih razlikuje na nek način, ki bi vodil do izboljšane rekonstrukcije dejanskih evolucijskih razmerij. To je seveda zahtevno vprašanje. Drugi možni pristopi, ki bi lahko vodili do boljših filogenij, vključujejo:

(1) Izračun pragov t_0 in t_1 v odvisnosti od vhodnih podatkov.

(2) Določitev ter odstranitev osamelcev. Zdi se namreč, da je ravno ta pristop programu LICHeE omogočil uspešno rekonstrukcijo filogenije na nekaterih instancah, težavnih za MIPUP+.

MIPUP+ ima potencialno aplikativno uporabnost v zvezi s trenutno aktualno temo epidemije COVID-19, saj nam lahko pomaga pri razumevanju filogenij virusa SARS-CoV-2 [17].

Program je dostopen na e-mail naslovu: scucko1@gmail.com.

7 Literatura in viri

[1] N. Bahgirova, *Reconstructing perfect phylogenies via branchings in acyclic digraphs: a new lower bound and efficiently solvable cases.* Magistrsko delo. Univerza na Primorskem, Fakulteta za matematiko, naravoslovje in informacijske tehnologije, 2020. *(Citirano na strani 7.)*

[2] A. Bashashati, G. Ha, A. Tone, J. Ding, L. M. Prentice, A. Roth, J. Rosner, K. Shumansky, S. Kalloger, J. Senz, W. Yang, M. McConechy, N. Melnyk, M. Anglesio, M. T. Luk, K. Tse, T. Zeng, R. Moore, Y. Zhao, M. A. Marra, B. Gilks, S. Yip, D. G. Huntsman, J. N. McAlpine in S. P. Shah, Distinct evolutionary trajectories of primary high-grade serous ovarian cancers revealed through spatial mutational profiling. *Journal of Pathology* 231 (2013) 21–34. *(Citirano na straneh 27, 32 in 37.)*

[3] M. El-Kebir, L. Oesper, H. Acheson-Field in B. J. Raphael, Reconstruction of clonal trees and tumor composition from multi-sample sequencing data. *Bioinformatics* 13.12 (2015) 62–70. *(Citirano na straneh 1 in 20.)*

[4] G. F. Estabrook, C. S. Johnson in F. R. Mc Morris, An idealized concept of the true cladistic character. *Mathematical Biosciences* 23 (1975) 263–272. *(Citirano na strani 4.)*

[5] M. Gerlinger, S. Horswell, J. Larkin, A. J. Rowan, M. P. Salm, I. Varela, R. Fisher, N. Mcgranahan, N. Matthews, C. R. Santos, P. Martinez, B. Phillimore, S. Begum, A. Rabinowitz, B. Spencer-Dene, S. Gulati, P. A. Bates, G. Stamp, L. Pickering, M. Gore, D. L. Nicol, S. Hazell, P. A. Futreal, A. Stewart in C. Swanton, Genomic architecture and evolution of clear cell renal cell carcinomas defined by multiregion sequencing. *Nature Genetics* 46 (2014) 225–233. *(Citirano na straneh 27 in 37.)*

[6] D. Gusfield, Efficient algorithms for inferring evolutionary trees. *Networks* 21 (1991) 19–28. *(Citirano na straneh 3 in 4.)*

[7] I. Hajirasouliha, B. J. Raphael, Reconstructing mutational history in multiply sampled tumors using perfect phylogeny mixtures., v: Proceedings of the 14th

International Workshop on Algorithms in Bioinformatics (WABI'14), *Lecture Notes in Comput. Sci. 8701. Springer*, Heidelberg, (2014), 354–367. *(Citirano na strani 5.)*

[8] A. Hujdurović, E. Husić, M. Milanič, R. Rizzi in A. I. Tomescu, Perfect phylogenies via branchings in acyclic digraphs and a generalization of Dilworth's theorem. *ACM Transactions on Algorithms* 14 (2018) 1–26. *(Citirano na straneh 5 in 7.)*

[9] A. Hujdurović, E. Husić, M. Milanič in A. I. Tomescu, *The minimum conflict-free row split problem: extension of the model to the case of missing data.* Neobjavljeni zapiski, 2016. *(Citirano na straneh 10 in 15.)*

[10] A. Hujdurović, U. Kačar, M. Milanič, B. Ries in A. I. Tomescu, Complexity and Algorithms for Finding a Perfect Phylogeny from Mixed Tumor Samples. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 15 (2018) 96–108. *(Citirano na strani 5.)*

[11] E. Husić, X. Li, A. Hujdurović M. Mehine, R. Rizzi, V. Mäkinen, M. Milanič in A. I. Tomescu, MIPUP: Minimum perfect unmixed phylogenies for multi-sampled tumors via branchings and ILP. *Bioinformatics* 35 (2019) 769–777. *(Citirano na straneh 2, 7, 8, 18, 20, 21, 37 in 39.)*

[12] U. Kačar, *Problemi popolne filogenije.* Zaključna projektna naloga. Univerza na Primorskem, Fakulteta za matematiko, naravoslovje in informacijske tehnologije, 2015. *(Citirano na strani 4.)*

[13] X. Li, *Data simulation of tumor phylogenetic trees and evaluation of phylogenetic reconstructing tools.* Magistrsko delo. Aalto University School of Science, 2017. *(Citirano na strani 21.)*

[14] P. C. Nowell, The clonal evolution of tumor cell populations. *Science* 194 (1976) 23–28. *(Citirano na strani 1.)*

[15] V. Popic, R. Salari, I. Hajirasouliha, D. Kashef-Haghighi, R. B. West in S. Batzoglou, Fast and scalable inference of multi-sample cancer lineages. *Genome Biology* 16.1 (2015) 91. *(Citirano na straneh 1, 20 in 21.)*

[16] H. Prüfer, Neuer Beweis eines Satzes über Permutationen. *Arch. Math. Phys.* 27 (1918) 742–744. *(Citirano na strani 20.)*

[17] D. Ramazzotti, F. Angaroni, D. Maspero, C. Gambacorti-Passerini, M. Antoniotti, A. Graudenzi in R. Piazza, Characterization of intra-host SARS-CoV-2 variants improves phylogenomic reconstruction and may reveal functionally convergent mutations. bioRxiv 2020.04.22.044404, poslano v objavo. 2020 *(Citirano na strani 39.)*

[18] *GraphViz Online*, `https://dreampuf.github.io/GraphvizOnline/`. (Datum ogleda: 18. 6. 2020.) *(Citirano na strani 18.)*

[19] *NCI slovar izrazov o raku*, NCI. `https://www.cancer.gov/publications/dictionaries/cancer-terms/def/somatic-mutation?redirect=true`. (Datum ogleda: 18. 6. 2020.) *(Citirano na strani 1.)*