1. Poiščite globalno poravnavo zaporedij s_1 =ATTCG, s_2 =GGAGT, s_3 =TGACG in s_4 =GAACT z uporabo

   (a) algoritma za progresivno poravnavo
   (b) 2-aproksimacijskega algoritma

Za razdaljo med dvema zaporedjema uporabite Needleman–Wunschev algoritem z nagrado 0 za ujemanje in kaznijo za zamenjavo, vstavljanje in brisanje enako 1.

a) Progresivna poravnava

1. Matrika razdalj
2. Drevo UPGMA
3. MSA

- $\binom{k}{2}$ parov zaporedij
- k zaporedij

1. Matrika razdalj (več možnih razdalj)

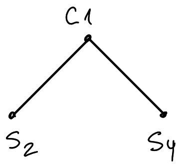
- Razdalja = ocena globalne poravnave (absolutne vrednosti)

	S_1	S_2	S_3	S_4
S_1	0	5	3	4
S_2		0	3	2
S_3			0	3
S_4				0

Poravnave so bile že ustvarjene, drugače moraš vse naredit sam.

2. Drevo UPGMA (več metod: UPGMA, neighbor joining, iz aditivnih matrik)

	S_1	S_2	S_3	S_4
S_1	0	5	3	4
S_2		0	3	2
S_3			0	3
S_4				0

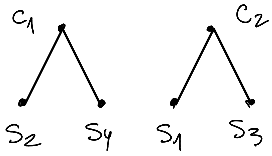


$$d(C_1, S_1) = \frac{d(S_2, S_1) + d(S_1, S_4)}{|C_1|} = \frac{5+4}{2} = 4.5$$

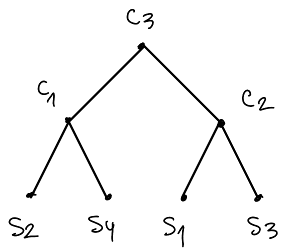
$d(C_1, S_3)$ = analogno

	S_1	C_1	S_3
S_1	0	4.5	3
C_1		0	3
S_3			0

izbereš enega



= očitno bo $C_3 = (C_2, C_1)$ $\Rightarrow$



3. MSA

- vsaki gruči določimo najboljšo poravnavo zaporedja ene skupine z zaporedjem iz druge skupine (najboljša = najmanjša):
  - C_1 ima poravnavo (S_2, S_4)
  - C_2 ima poravnavo (S_1, S_3)
  - C_3 ima poravnavo $\min\{C_1, C_2\} = \min\{d(S_2,S_1), d(S_2,S_3), d(S_4,S_1), d(S_4,S_3)\}$ $= \min\{5, 3, 4, 3\}$; izberemo (S_2, S_3), saj se S_4 boljše poravna z S_1.

- Začnemo na vrhu s $C_3 = (S_2, S_3)$ in progresivno dodajamo poravnave.
  - S_4 dodaš v (S_2, S_3) glede na to kako se poravna z S_2
  - S_1 dodaš v (S_4, S_2, S_3) glede na to kako se poravna z S_3

S_2	G	G	A	–	G	T
S_3	T	G	A	C	G	–

S_2	G	G	A	G	T
S_4	G	A	A	C	T

S_4	G	A	A	–	C	T
S_2	G	G	A	–	G	T
S_3	T	G	A	C	G	–

S_1	A	T	T	C	G
S_3	T	G	A	C	G

S_4	G	A	A	–	C	T
S_2	G	G	A	–	G	T
S_3	T	G	A	C	G	–
S_1	A	T	T	C	G	–

b) 2-aproksimacijski algoritem

- 1. korak je enak kot pri (a); 2. korak ni potreben – nadomestimo ga z naslednjim algoritmom:
  - Vrednost poravnave je enaka vsoti vrednosti poravnav vseh $\binom{k}{2}$ parov zaporedij, induciranih s poravnavo k zaporedij.
  - Predp., da iščemo čim manjše vrednosti
  - Vsota parov poravnav k-zaporedij bo kvečjemu za faktor $2-2/k$ večja od optimalne vsote parov.

	S_1	S_2	S_3	S_4	
S_1	0	5	3	4	= 12
S_2	5	0	3	2	= 10
S_3	3	3	0	3	= 9
S_4	4	2	3	0	= 9

seštejemo vrstice ...

Izberemo enega izmed teh dveh najmanjših. Izberimo npr. S_3.

- Poravnamo po vrsti (S_1, S_3), (S_1, S_2, S_3), (S_1, S_2, S_4, S_3). glede na (S_1, S_3), (S_2, S_3) in (S_4, S_3).

```
S1 | A T T C G        S2  G G A - G T
S3 | T G A C G        S3  T G A C G -
```

```
S1 | A T T C G -
S2 | G G A - G T
S3 | T G A C G -
```

```
S3 | T G A - C G
S4 | - G A A C T
```

```
MSA:  S1 | A T T - C G -
      S2 | G G A - - G T
      S3 | T G A - C G -
      S4 | - G A A C T -
```

Vrednost poravnave:

$$v(A) := \sum_{i<j} v_A(s_i, s_j)$$

Iz MSA dobimo Levenstheinovo razdaljo za vsak par:

- $V_A(S_1, S_2) = 5$
- $V_A(S_2, S_3) = 3$
- $V_A(S_2, S_4) = 5$
- $V_A(S_3, S_4) = 3$

```
S1 | A T T - C G -
S2 | G G A - - G T
```

- $V_A(S_1, S_3) = 3$
- $V_A(S_1, S_4) = 5$

$$v(A) = 5+3+5+3+5+3 = 24$$

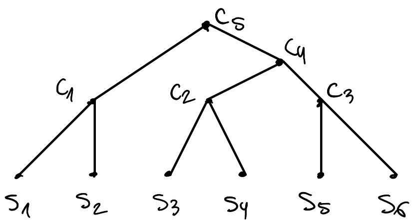
Za optimalno poravnavo A^* velja:

$$v(A_c) \leq (2-2/k)\, v(A^*)$$
$$v(A_c)/(2-2/k) \leq v(A^*)$$

Dobimo jo tako, da iz matrike razdalj odčitamo razdalje med zaporedji; vzamemo manjšo izmed te in one razdalje iz MSA:

$$v(A^*) = 5+3+4+3+2+3 = 20$$

***. Kako bi poravnal zaporedja glede na drevo:**



- C_1 ima (S_1, S_2)
- C_2 ima (S_3, S_4)
- C_3 ima (S_5, S_6)

- C_4 ima $\min\{C_2, C_3\} = \min\{d(S_3,S_5), d(S_3,S_6), d(S_4,S_5), d(S_4,S_6)\}$ npr. (S_3, S_5)
- C_5 ima $\min\{C_1, C_4\} = \min\{d(S_1,S_3), d(S_1,S_5), d(S_2,S_3), d(S_2,S_5)\}$ npr. (S_1, S_5)

par, ki smo ga izbrali za C_4, ne vsa zaporedja spodaj ...

$\boxed{\text{Za poravnavo potrebujemo poravnave zaporedij: } (S_1, S_2), (S_3, S_4), (S_5, S_6), (S_3, S_5) \text{ in } (S_1, S_5).}$

3. Izračunajte zaporedje s, ki vsebuje 4−terice podane v multimnožici

$$S = \{CGAA, ATAG, TAGT, GAAT, TAGA, AGTA, AATA, AGAT, GTAG, GATA, ATAG\}$$

tako, da bo $Spekter(s, 4) = S$, z uporabo

(a) Hamiltonske poti

(b) Eulerjeve sledi

Ali je rešitev enolično določena?

a) Hamiltonske poti

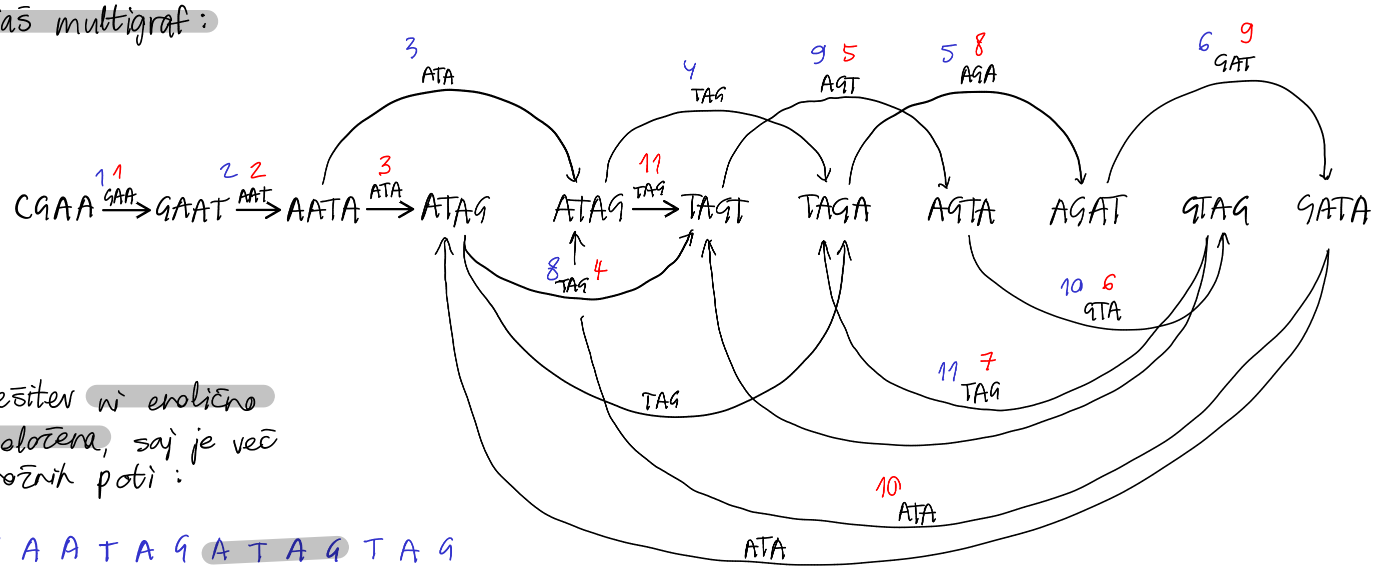
- Hamiltonska pot obišče vsako točko natanko enkrat.
- Hamiltonski cikel je cikel, ki gre skozi vse točke enkrat.
- Ne moremo podati preprostega pogoja potrebnega za obstoj hamiltonskega cikla, saj je problem NP-poln (enako za ham. poti).

- Metode sekvenciranja najprej določijo množico podzaporedij neznanega zaporedja, ki se med seboj prekrivajo. Kako ugotoviti celotno zaporedje?
  - Dano je zap. s nad abecedo Σ in $\ell \in \mathbb{N}$. ℓ-spekter zaporedja je multimnožica vseh podzap. zap. s dolžine ℓ. Označimo ga $Spekter(s, \ell)$.
  - Problem izračuna zaporedja s, za katerega velja $Spekter(s, \ell) = S$, je učinkovito rešljiv.

Naj bo $V = S$. $p, q \in E$ natanko tedaj, ko se p in q prekrivata $\Leftrightarrow$ ko je zadnjih $\ell - 1$ znakov p enakih prvih $\ell - 1$ znakov q.

p: A C C T G
　　C C T G G : q

Naš multigraf:

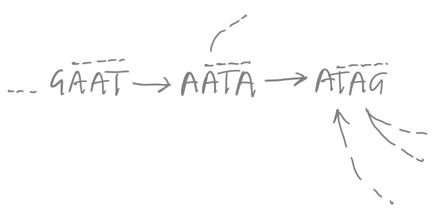


Rešitev ni enolično določena, saj je več možnih poti:

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

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

Namesto, da pišeš nad povezave ℓ-1 besede, smo naredi pikice na točkah:



3. Izračunajte zaporedje s, ki vsebuje 4−terice podane v multimnožici

$$S = \{CGAA, ATAG, TAGT, GAAT, TAGA, AGTA, AATA, AGAT, GTAG, GATA, ATAG\}$$

tako, da bo $Spekter(s, 4) = S$, z uporabo

(a) Hamiltonske poti
(b) Eulerjeve sledi

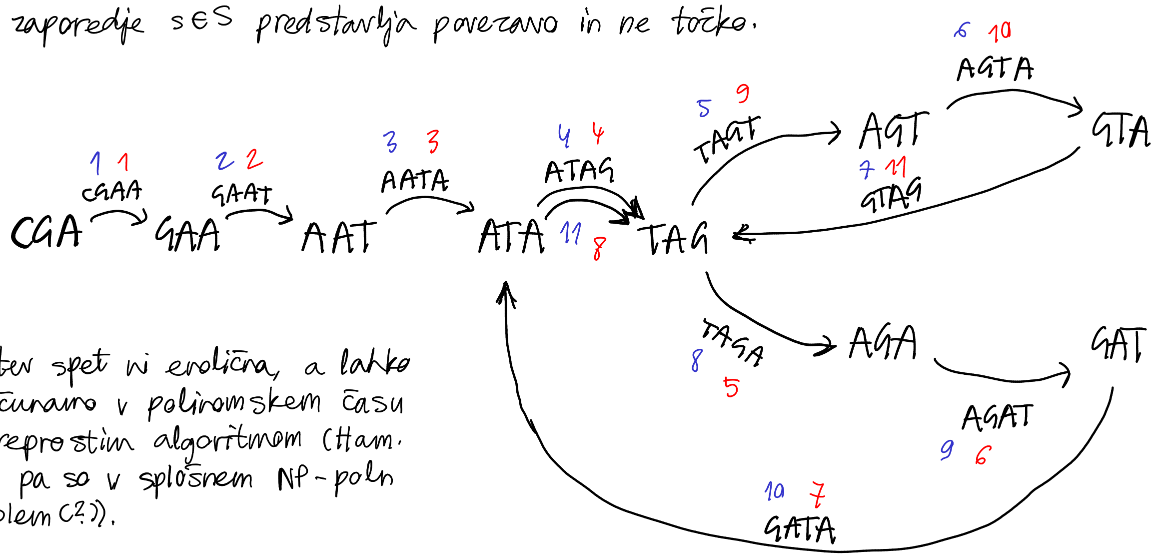
Ali je rešitev enolično določena?

b) Eulerjeve sledi

Za vhodne podatke (l, S) konstruiramo digraf $D = (V, E)$ (v katerem so možne večkratne povezave), kjer je:

- V množica vseh zaporedij dolžine $l-1$, ki nastopajo kot podzaporedja v S.
- $(p, q) \in E$ natanko tedaj, ko $\exists$ zaporedje $s \in S$, da je prvih $l-1$ znakov s enakih p, zadnjih $l-1$ s pa q. Povezavo označimo z l-terico s.

Vsako zaporedje $s \in S$ predstavlja povezavo in ne točko.



Rešitev spet ni enolična, a lahko izračunamo v polinomskem času s preprostim algoritmom (Ham. poti pa so v splošnem NP-poln problem (?)).

Rešitvi:

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

(enako kot prej :0)