Algoritmi v bioinformatiki
Najkrajše in najdaljše poti v acikličnih digrafih

Martin Milanič
martin.milanic@upr.si
UP FAMNIT

2. april 2025

Aciklični digrafi

usmerjeni grafi (digrafi):

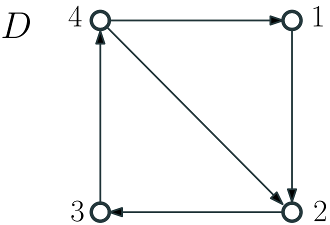
$D = (V, E)$

V: množica točk, E: množica usmerjenih povezav

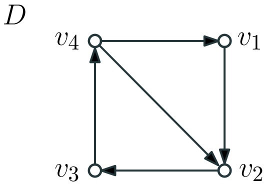
usmerjena povezava = urejen par točk, $E \subseteq V \times V$

Zgled:

$V = \{1, 2, 3, 4\}$, $E = \{(1, 2), (2, 3), (3, 4), (4, 1), (4, 2)\}$



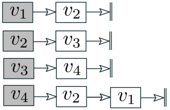
Zgled:



Matrika sosednosti:

$$\begin{array}{cc} & \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

Seznami sosedov:



Pravimo, da je problem na digrafih **rešljiv v linearnem času**, če ga je mogoče rešiti z algoritmom časovne zahtevnosti $\mathcal{O}(n+m)$, kjer je $n = |V|$ in $m = |E|$.

Predpostavimo, da je digraf podan s seznami sosedov.

Poti in **cikli** v digrafih so definirani podobno kot v (neusmerjenih) grafih, pri čemer upoštevamo tudi

usmerjenost povezav.

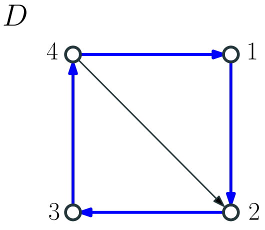
Na primer: **cikel** v digrafu $D = (V, E)$ tako zaporedje

$$x_1, \dots, x_k$$

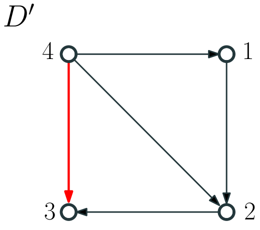
paroma različnih točk, da velja $(x_i, x_{i+1}) \in E$ za vse $i \in \{1, \dots, k-1\}$ in $(x_k, x_1) \in E$.

Acikličen digraf (ang. DAG, directed acyclic graph) = digraf brez ciklov

Zgled:



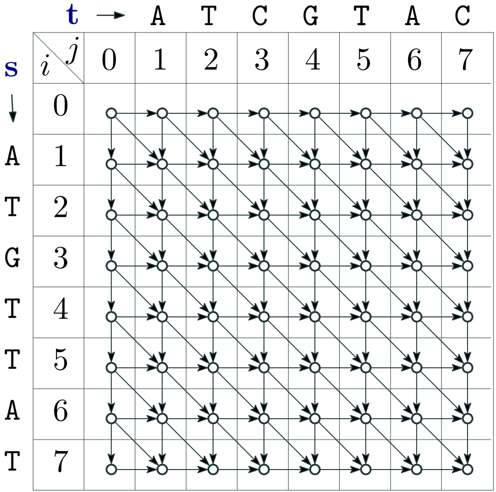
$1, 2, 3, 4$ je cikel v D



D' je acikličen

Zgled:

Graf poravnav je acikličen:



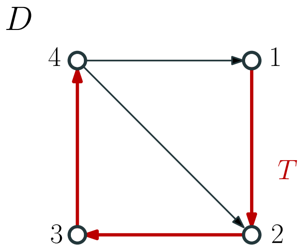
Iskanje v globino

Iskanje v globino (ali: pregled v globino, *ang.: depth-first search, DFS*) je eden izmed najosnovnejših načinov, kako pregledati graf. Uporabno je tako za neusmerjene kot za usmerjene grafe.

Za dan digraf D, iskanje v globino iz dane začetne točke s v linearnem času izračuna vse točke, ki so dosegljive iz točke s.

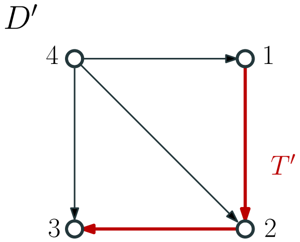
Pravimo, da je točka t **dosegljiva** *iz točke s, če v digrafu D obstaja (usmerjena) pot od s do t.*

Zgled:



T = DFS drevo v grafu D iz začetne točke $s = 1$

Zgled:



$T' =$ DFS drevo v grafu D' iz začetne točke $s = 1$
(točka 4 ni dosegljiva iz točke 1)

Topološko urejanje

Topološka ureditev acikličnega digrafa $D = (V, E)$ je taka linearna razvrstitev $(v_1, \ldots, v_n)$ množice V, za katero velja:

$$(v_i, v_j) \in E \Rightarrow i < j\,.$$

Vsak acikličen digraf ima vsaj eno topološko ureditev.

To je povezano z dejstvom, da so aciklični digrafi uporabni za prikazovanje **časovnih odvisnosti med dogodki**.

Digraf D:

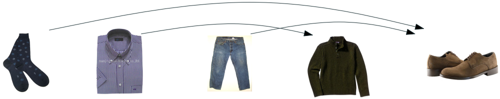
$$D = (V, E)$$

Digraf D in tri njegove topološke ureditve:

$D = (V, E)$

1. 

2. 

3. 

DEJSTVO: Z majhno spremembo iskanja v globino je mogoče izračunati topološko urejanje poljubnega acikličnega digrafa.

Algoritem uredi točke po tem, kdaj je DFS končal s tisto točko in njenim poddrevesom, in na koncu ta vrstni red obrne.

Najkrajše poti v acikličnih digrafih

Topološko ureditev acikličnega digrafa $D = (V, E)$ lahko
s pridom uporabimo za razvoj
algoritmov dinamičnega programiranja
za razne probleme na digrafu D.

Ta pristop bomo ponazorili na problemu najkrajše poti.

Problem najkrajše poti

Podatki: Digraf $D = (V, E)$, realne uteži na povezavah $w : E \to \mathbb{R}$, dve točki s, t.

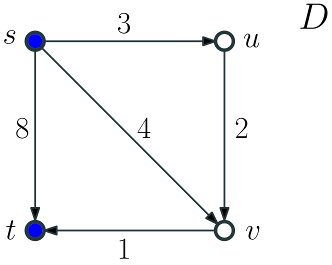
Naloga: Poišči najkrajšo pot od točke s do točke t.

V tem kontekstu je **dolžina** poti P definirana kot

$$w(P) = \sum_{e \in E(P)} w(e)\,,$$

kjer $E(P)$ označuje množico povezav na poti P.

Zgled:



D vsebuje tri poti od s do t:

- pot s, t dolžine 8,
- pot s, v, t dolžine $4 + 1 = 5$,
- pot s, u, v, t dolžine $3 + 2 + 1 = 6$.

Najkrajša s, t-pot: s, v, t

Naj bo digraf D acikličen, s topološko ureditvijo $(v_1, \ldots, v_n)$.

Pokazali bomo, kako lahko **najkrajše poti (glede na utežno funkcijo $w : E \to \mathbb{R}$) od točke s do vseh točk, dosegljivih iz točke s,**

izračunamo v **linearnem času** z uporabo topološke ureditve.

Dve očitni opazki:

1. Pot od $s = v_i$ do $t = v_j$ lahko obstaja, samo če je $i \leq j$. Predpostavimo lahko torej, da velja $s = v_1$.

   (Če je $s = v_i$ za $i > 1$, lahko točke $v_{i'}$, kjer je $i' < i$, odstranimo, preostale pa ustrezno preimenujemo.)

2. Predpostavimo lahko tudi, da so vse točke dosegljive iz točke $s = v_1$.

   (V nasprotnem primeru uporabimo DFS in odstranimo vse točke, ki niso dosegljive iz točke s.)

Oglejmo si naslednji algoritem:

for each $i = 1, \ldots, n$

Naj bo $d_i = \infty$, $\pi_i = 0$.

end for

$d_1 = 0$;

for each $i = 2, \ldots, n$

for each $(v_j, v_i) \in E$

if $d_j + w(v_j, v_i) < d_i$ **then**

$d_i \leftarrow d_j + w(v_j, v_i)$

$\pi_i \leftarrow j$

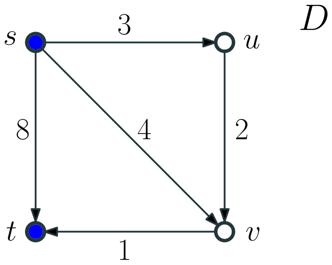
end if

end for

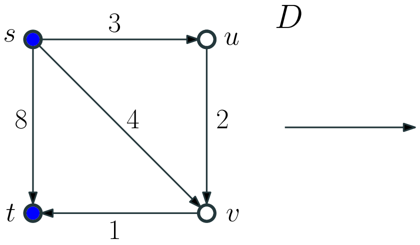
end for

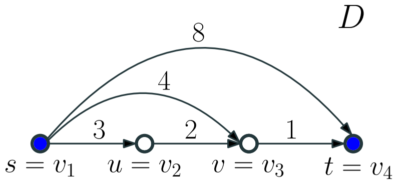
Zgled:

Recimo, da so dani naslednji vhodni podatki:



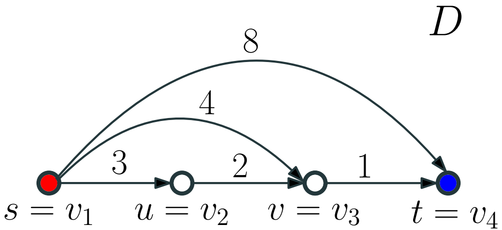
Zgled:





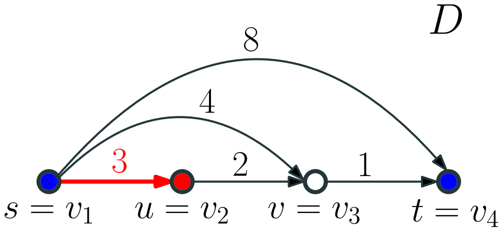
topološka ureditev digrafa D

Zgled:



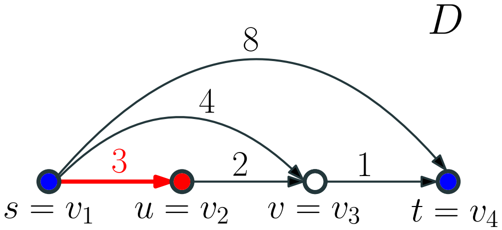
d_i	0	∞	∞	∞
π_i	0	0	0	0

Zgled:



d_i	0	∞ $0 + 3 < \infty$	∞	∞
π_i	0	0	0	0

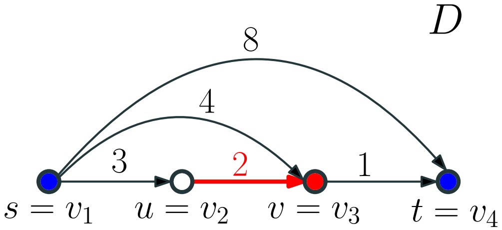
Zgled:



d_i	0	3	∞	∞
π_i	0	1	0	0

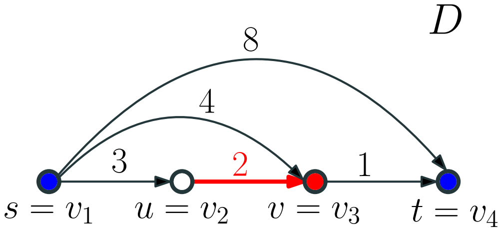
$\pi_2 = 1$

Zgled:



d_i	0	3	∞ $3+2<\infty$	∞
π_i	0	1	0	0

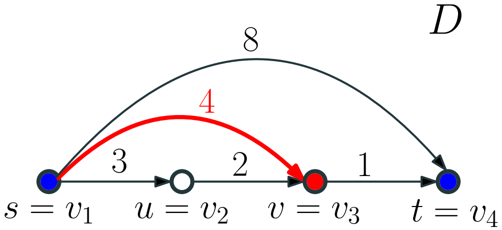
Zgled:



d_i	0	3	5	∞
π_i	0	1	2	0

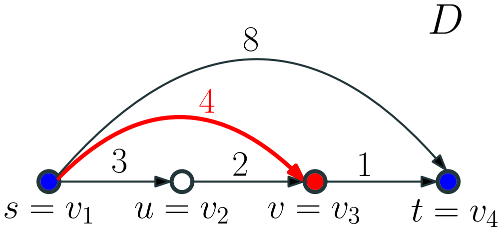
$\pi_3 = 2$

Zgled:



d_i	0	3	5 $0 + 4 < 5$	∞
π_i	0	1	2	0

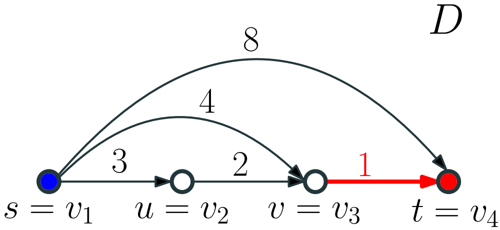
Zgled:



d_i	0	3	4	∞
π_i	0	1	1	0

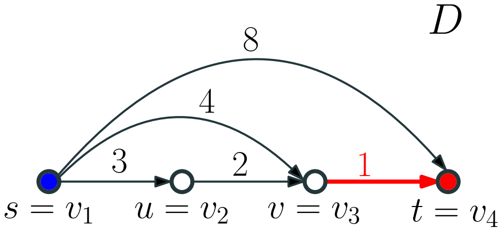
$\pi_3 = 1$

Zgled:



d_i	0	3	4	∞ $4+1<\infty$
π_i	0	1	1	0

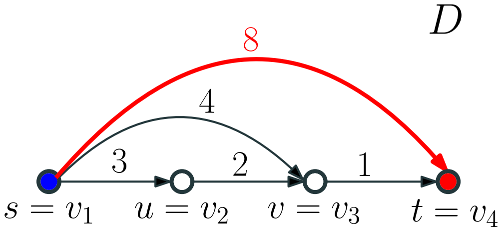
Zgled:



d_i	0	3	4	5
π_i	0	1	1	3

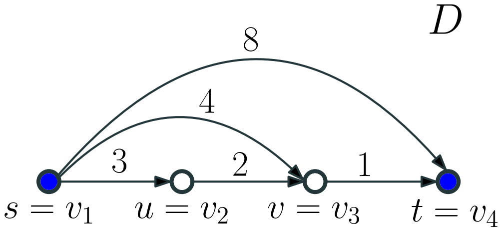
$\pi_4 = 3$

Zgled:



d_i	0	3	4	5 $0 + 8 \not< 5$
π_i	0	1	1	3

Zgled:



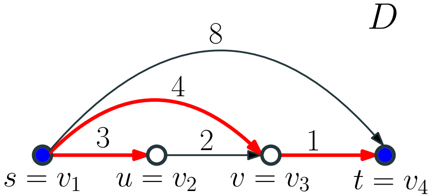
d_i	0	3	4	5
π_i	0	1	1	3

Na koncu algoritma tvori množica povezav (v_j, v_i), za katere je $\pi_i = j$,

usmerjeno drevo, ki vsebuje najkrajše poti iz točke s do vsake druge točke,

d_i pa označuje dolžino najkrajše s, v_i-poti.

Zgled:



d_i	0	3	4	5
π_i	0	1	1	3

Zakaj je algoritem **pravilen**?

Z indukcijo po i je mogoče dokazati, da zgornja trditev drži za točke $v_1, \ldots, v_i$ po $i - 1$ iteracijah glavne **for** zanke.

Zgornji algoritem je primer uporabe koncepta **dinamičnega programiranja**:

- Problem rešimo tako, da zaporedoma optimalno rešimo vrsto podproblemov.
- Optimalno rešitev konstruiramo **s pomočjo optimalne rešitvega enega ali več podproblemov, ki smo jih že rešili**.

V našem primeru:

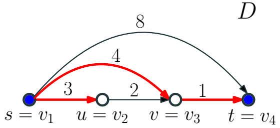
Za vsak $i = 1, \ldots, n$ rešimo podproblem, v katerem izračunamo **dolžine najkrajših poti od točke $s = v_1$ do vseh točk v množici $\{v_1, \ldots, v_i\}$.**

Rešitev podproblema, indeksiranega z i, dobimo tako, da rešitvi podproblema, indeksiranega z $i - 1$, dodamo dolžino najkrajše poti od s do v_i.

V algoritmu je ta dolžina označena z d_i.

Kolikšna je **časovna zahtevnost** algoritma?

for $i = 1, \ldots, n$
    Naj bo $d_i = \infty$, $\pi_i = 0$.
end for
$d_1 = 0$;
for $i = 2, \ldots, n$
    for $(v_j, v_i) \in E$
        if $d_j + w(v_j, v_i) < d_i$ **then**
            $d_i \leftarrow d_j + w(v_j, v_i)$
            $\pi_i \leftarrow j$
        end if
    end for
end for



d_i	0	3	4	5
π_i	0	1	1	3

Zahtevnost je **linearna**:

- $\mathcal{O}(n)$ za inicializacijo;
- vsako povezavo pregledamo kvečjemu enkrat, v času $\mathcal{O}(1)$.

Opisani pristop je uporaben tudi za iskanje

najdaljše u, v-poti v acikličnem digrafu:

dano utežno funkcijo $w : E \to \mathbb{R}$ zamenjamo s funkcijo $w' : E \to \mathbb{R}$, definirano s predpisom

$$w'(e) = -w(e) \qquad \text{za vse } e \in E$$

in izračunamo najkrajšo u,v-pot v (D, w').

Kot posebna primera tega algoritma dobimo rešitve za problema:

- globalne poravnave parov zaporedij **(Needleman-Wunsch)**;
- lokalne poravnave parov zaporedij **(Smith-Waterman)**.