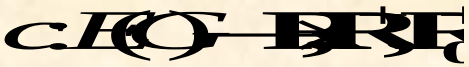
Najkrajše poti in Bellman-Fordov algoritem

Urša Remžgar

5.1.2010

Graf

- G je usmerjen, utežen graf. Utež je



- Pot je sprehod, v katerem se nobeno vozlišče ne ponovi.

$$P = \langle v_o, v_1, \ldots, v_k \rangle$$

- Dolžina ali teža poti je $c(P) = \sum_{e \in E(P)} c(e)$

- Iščemo najkrajše poti v grafu (vsota uteži je najmanjša).

- Algoritmi:
  - Dijkstrov algoritem (le za nenegativne uteži)
  - Bellman-Fordov algoritem (tudi za negativne uteži)
  - Floyd-Warshallov algoritem (najkrajša pot med dvema točkama)

Dijkstrov algoritem

- Dan je utežen graf *G* z nenegativnimi utežmi.
- Iščemo najkrajše poti z začetkom v *s*.
- Gradimo drevo - dodamo tisto točko, ki je trenutno najbližja s.

dijkstra(G, s)

za vse $u \in V(G)$
  $d[u] = \infty$ //d je tabela z razdaljami od točke s do drugih točk
  prednik[u] = **null** //prednik točke v drevesu najkrajših poti
$d[s] = 0$

$Q = V(G)$ //Q je struktura, v kateri hranimo točke iz grafa G, ki še niso v drevesu
dokler $Q \neq \emptyset$;
  u = minQ //vrne točko, ki ima pripadajoči d najmanjši, in jo odstrani iz Q
  za vse *uv* $\in \delta^+(u)$ //$\delta^+(u)$ je množica povezav, ki imajo začetek v u
    če *d[v] > d[u] + c(uv)* **potem**
      d[v] = d[u] + c(uv)
      prednik[v] = u

Primer Dijkstrovega algoritma

- http://optlab-server.sce.carleton.ca/POAnimations2007/DijkstrasAlgo.html

Bellman-Fordov algoritem

- V danem uteženem usmerjenem grafu *G* z utežmi *c* (lahko negativne) najde vse najkrajše poti iz točke *s*.
- Izbira primerne povezave (take, ki zmanjša težo najkrajše poti) do vseh vozlišč (|V(G)-1|)-krat
- Problem je, če graf vsebuje negativne cikle (cikle, ki imajo vsoto uteži negativno), dosegljive iz s (obstaja pot od s do katere koli točke negativnega cikla)
- Kot rezultat v primeru, ko v grafu ni negativnega cikla, dosegljivega iz *s*, vrne logično vrednost **true** (pri tem izračuna razdalje *d* in drevo najkrajših poti, podano prek tabele prednikov),
- sicer vrne **false**.
- Je najhitrejši znan algoritem za splošne grafe (reši v polinomskem času).
- Časovna zahtevnost: O(|V||E|)

Algoritem

bellman_ford(G, s)

za vse $u \in V(G)$
    $d[u] = \infty$
    $prednik[u]$ = **null**
$d[s] = 0$

//Sprostitev (S)
za $i = 1$ **do** $|V(G)| - 1$
    za vse $uv \in E(G)$ //uv je povezava iz u v v
        če $d[v] > d[u] + c(uv)$ **potem**
            $d[v] = d[u] + c(uv)$
            $prednik[v] = u$

za vse $uv \in E(G)$
    če d[v] > d[u] + c(uv) **potem vrni false** //obstaja negativen cikel, dosegljiv iz *s*
vrni true //iz s ni dosegljiv noben negativen cikel

Sprostitev

- **Sprostitev** nad povezavo *uv* preveri, če lahko zmanjšamo težo najkrajše poti *d[v]* od *s* do *v* tako, da najkrajši povezavi *d[u]* od *s* do *u*, dodamo povezavo *uv.*

- **Lastnost sprostitve:**
  - Dana je najkrajša pot $P = \langle v_0, v_1, \ldots, v_k \rangle$ od *s* do v_k.
  - Izvedemo sprostitev v vrstnem redu

$$\langle v_0, v_1, v_2, \ldots, v_{k+1} \rangle$$

  - Dobimo $d[v_k] = \delta(s, v_k)$, kjer je $\delta(s, v_k)$ dolžina najkrajše poti od *s* do v_k.

Dokaz pravilnosti algoritma

Naj bo $G = (V,E)$ utežen, $c : E(G) \rightarrow \boldsymbol{R}$, usmerjen graf z izvorom s.

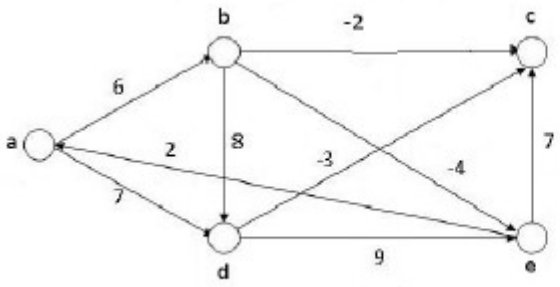
- **1. del:**
  Predpostavimo, da G ne vsebuje negativnega cikla, dosegljivega iz s. Trdimo, da po $|V(G)| - 1$ ponovitvah zanke (S), za vsako točko $v \in V$ velja $d[v] = \delta(s, v)$ in algoritem vrne **true**.

- **2. del:**
  Trdimo, da če graf G vsebuje negativen cikel, dosegljiv iz s, potem algoritem vrne **false**.

Primeri

- **Uporaba Bellman Fordovega algoritma**

http://links.math.rpi.edu/applets/appindex/graphtheory.html



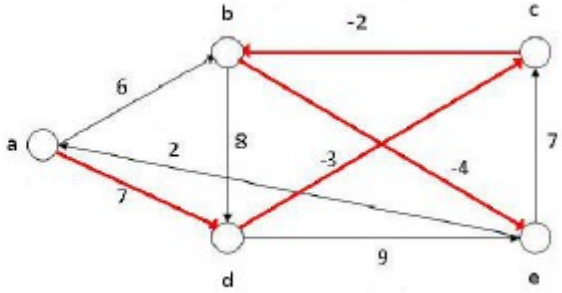


tabela d

a	b	c	d	e
0	∞	∞	∞	∞
	6	4	7	2
	2			
				-2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
0	2	4	7	-2

tabela *prednik*

a	b	c	d	e
/	/	/	/	/
	a	d	a	b
	c			
				b
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
/	c	d	a	b

seznam povezav

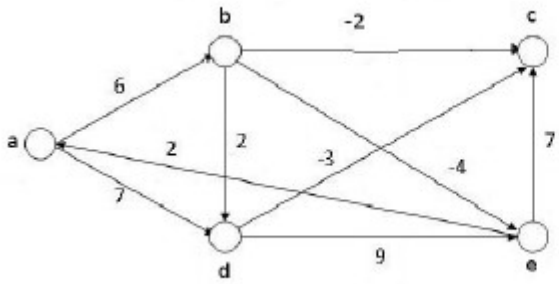
ab	6	$*$	–	–
ad	7	$*$	–	–
bd	8	–	–	–
be	-4	$*$	$*$	–
dc	-3	$*$	–	–
de	9	–	–	$\vdots$
cb	-2	$*$	–	
ea	2	–	–	
ec	7	–	–	

Algoritem vrne true, saj za vsaki točki u in v velja d[v] ≤ d[u] + c(uv).

Primeri

- **Uporaba Bellman Fordovega algoritma na grafu z negativnim ciklom, dosegljivim iz izvorne točke.**

http://links.math.rpi.edu/applets/appindex/graphtheory.html



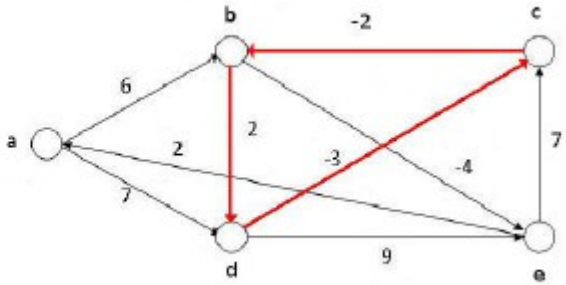


tabela d

a	b	c	d	e
0	∞	∞	∞	∞
	6	4	7	2
	2			
	-1	1	4	-2
-3	-4	-2	1	-5
-6	-7	-5	-2	-8

tabela *prednik*

a	b	c	d	e
/	/	/	/	/
	a	d	a	b
	c			
	c	d	b	b
e	c	d	b	b
e	c	d	b	b

seznam povezav

ab	6	$*$	—	—	—
ad	7	$*$	—	—	—
bd	2	—	$*$	$*$	$*$
be	-4	$*$	$*$	$*$	$*$
dc	-3	$*$	$*$	$*$	$*$
de	9	—	—	—	—
cb	-2	$*$	$*$	$*$	$*$
ea	2	—	—	$*$	$*$
ec	7	—	—	—	—

Algoritem vrne vrednost false, saj je npr. -2 = d[d] > d[b] + c(bd) = -7 + 2 = -5.

Floyd-Warshallov algoritem

- $d_{ij}^{(k)}$:= dolžina najkrajše poti od i do j, katere notranje točke so vsebovane v {1, 2, . . . , k}.
- Točke grafa smo označili z 1, 2, . . . , |V (G)|. Iščemo $d_{ij}^{|V|}$.
- Velja rekurzivna zveza:

$$d_{ij}^{(k)} = \begin{cases} c_{ij} = c(i,j) & \text{za } k = 0. \\ \min\{d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\} & \text{za } k \geq 0, \end{cases}$$

kjer je $c_{ii}=0$, $c_{ij}=\infty$ za $ij \notin E(G)$.

- Algoritem v uteženem grafu *G* s poljubnimi utežmi *c* najde vse najkrajše poti (med vsemi pari točk). Uteži so podane v matriki C; če med točkama ni povezave, utež (element matrike) postavimo na ∞.
- Uporabimo ga lahko za odkrivanje negativnih ciklov.

Algoritem

floyd_warshall(G)

$D^{(0)} = C$

za $k = 1$ **do** $|V(G)|$

za $i = 1$ **do** $|V(G)|$ //po vrsticah

za $j = 1$ **do** $|V(G)|$ //po stolpcih

$$d_{ij}^{(k)} = \min\{d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\}$$