Traženje najkraćeg puta

Filip Nikšić

Problem traženja najkraćeg puta u grafu[1] vrlo je čest praktičan problem. U ovom članku pokušat ću pokazati dva jednostavna algoritma (da, ovaj put na stvari gledamo s informatičke strane) za rješavanje tog problema. Naravno, da bi se uopće moglo raspravljati o najkraćem putu u grafu, potrebno je znati što je graf. Stoga ću pružiti kratak uvod u teoriju grafova.

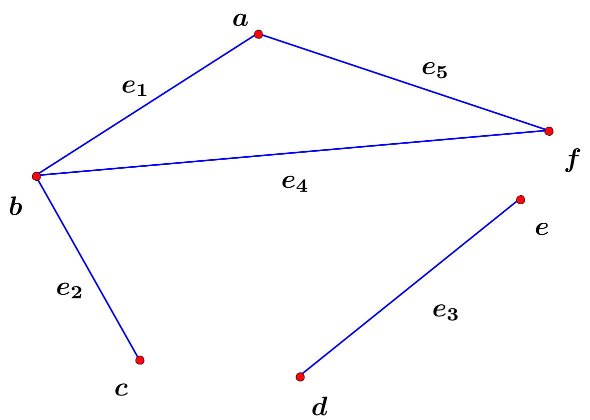
Graf? Nikad čuo.

Postoji nebrojeno mnogo skupova s konačnim brojem elemenata među kojima postoje neke relacije. Primjerice, skup bi se mogao sastojati od nekoliko ljudi, kompanija, sportskih timova ili gradova; relacije između dva elementa A i B takvog skupa bi mogle biti takve da osoba A pozna osobu B, država A graniči s državom B, kompanija A surađuje s kompanijom B, sportski tim A igrao je protiv tima B, grad A ima direktnu željezničku (ili cestovnu) liniju do grada B. Primijetimo da su gornje relacije simetrične ako država A graniči s državom B, onda vrijedi i obrnuto.

Velik broj takvih skupova može se modelirati koristeći *grafove*. Pa što su onda grafovi?

Definicija 1. Graf G je uređeni par $G = (V, E)$, gdje je V neprazan skup **vrhova**, a E je skup **bridova**. Te vrhove nazivamo krajevima brida. Broj vrhova u grafu označavamo s $v(G)$, a broj bridova s $e(G)$. Vrhove grafova označavamo malim slovima, a ponekad ih indeksiramo prirodnim brojevima: $v_1, v_2, v_3, ...$

Primjer 1. U ovom primjeru graf sa slike 1. definira se na slijedeći način:



Slika 1.

$G = (V, E)$
$V = \{a, b, c, d, e, f\}$
$E = \{e_1, e_2, e_3, e_4, e_5\}$
$e_1 = \{a, b\}$
$e_2 = \{b, c\}$
$e_3 = \{d, e\}$
$e_4 = \{b, f\}$
$e_5 = \{f, a\}$

[1]Mreže cesta mogu se predstaviti kao grafovi, pa se ovaj problem može i tu pokazati korisnim.

Put (path) u grafu je niz vrhova i bridova:

$$(v_0 e_1 v_2 e_2 ... v_{k-1} e_k v_k)$$

$$v_0, v_1, v_2, ..., v_k \in V$$

$$e_1, e_2, e_3, ..., e_k \in E$$

pri čemu je $e_i = \{v_{i-1}, v_i\}$. Npr. u primjeru 1. imamo sljedeći put: $(be_4fe_5ae_1be_2c)$. *Jednostavan put* je put u kojem se ne ponavljaju vrhovi, dakle $(be_4fe_5ae_1be_2c)$ *nije* jednostavan put.

Reprezentacija grafova

Svakom grafu s $v(G)$ vrhova možemo pridružiti kvadratnu matricu M dimenzija $v(G) \times v(G)$ u čijem se i-tom retku i j-tom stupcu nalazi 1[2] ako postoji brid koji spaja i-ti i j-ti vrh, odnosno 0 ako takav brid ne postoji. Takva matrica naziva se *matrica susjedstva*. Graf iz primjera 1. ima sljedeću matricu susjedstva:

	a	b	c	d	e	f
a	0	1	0	0	0	1
b	1	0	1	0	0	1
c	0	1	0	0	0	0
d	0	0	0	0	1	0
e	0	0	0	1	0	0
f	1	1	0	0	0	0

$$M = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Uočimo da je graf jednoznačno određen svojom matricom susjedstva, što nam omogućuje vrlo jednostavnu implementaciju grafa u računalu. Na taj način računalo u kombinaciji s grafovima postaje moćan alat kojim možemo rješavati probleme za koje bi nam "ručno" bilo potrebno mnogo vremena.

Težinski grafovi

U praksi se često javlja potreba da svakom bridu e grafa G pridružimo realan broj $w(e)$ koji zovemo *težina brida e*. Takav graf zovemo *težinski graf.* Tako npr. vrhovi grafa mogu predstavljati gradove, a težine bridova udaljenosti među tim gradovima ili troškove cestarine. Pojam matrice susjedstva možemo jednostavno proširiti i na slučaj težinskog grafa. Tada će elementi matrice biti težine odgovarajućih bridova, dakle realni brojevi $w(e)$, a ukoliko ne postoji brid koji spaja neki par vrhova, tada na odgovarajuće mjesto u težinskoj matrici susjedstva stavljamo *beskonačno.*

Najkraći put u grafu

Vratimo se sada na početak priče. Rječnikom obogaćenim novom terminologijom možemo reći da je problem najkraćeg puta u grafu problem traženja puta u težinskom grafu koji spaja dva zadana vrha x i y s najmanjim mogućim zbrojem težina bridova na tom putu. Ako su sve težine 1 (odnosno, ako graf nije težinski) problem je i dalje zanimljiv - traži se put koji povezuje x i y s najmanjim mogućim brojem bridova.

[2]Kod grafova koji sadrže veći broj bridova koji spajaju dva ista vrha piše se broj tih bridova. Takav graf se zove *multigraf.*

Pretraživanje u širinu

Pretraživanje u širinu (Breadth-First Search) jedan je od osnovnih algoritama za prolazak grafom i može se primijeniti za traženje najkraćeg puta u netežinskom grafu. Ideja je da se kroz graf prolazi počevši od x tako da se u prvom koraku posjete svi vrhovi s kojima je x spojen, u drugom koraku svi vrhovi s kojima su spojeni vrhovi iz prvog koraka i tako redom dok ne naiđemo na y. Put kojim smo došli do y bit će najkraći put od x do y.

U ovom algoritmu koristi se struktura podataka *red (queue)* uz koju vežemo dvije operacije: možemo *staviti* nešto na kraj reda i *uzeti* nešto s početka reda. Mi ćemo stavljati i uzimati vrhove grafa. M je matrica susjedstva zadanog grafa, x početni vrh, y vrh do kojeg tražimo najkraći put, a vektor *duljina* koristi se za pohranjivanje duljine puta od x do ostalih vrhova.

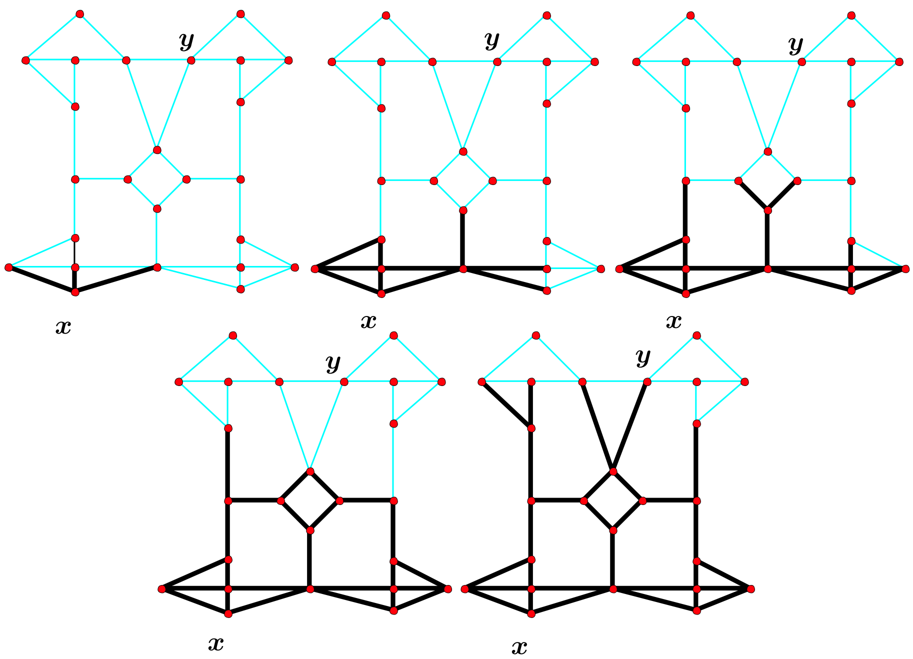
```
za svaki v iz V radi | duljina(v):= 0

stavi u red x

dok nije prazan red radi | v:= uzmi iz reda | ako je v = y onda
prekini petlju | za svaki u iz V\{v} radi | | ako je M(v,u) = 1 i
duljina(u) = 0 onda | | | stavi u red u | | | duljina(u):=
duljina(v) + 1

rjesenje:= duljina(y)
```

Pogledajmo na slici kako izgleda pretraživanje u širinu:



Slika 2.

Dijkstrin algoritam

Kod traženja najkraćeg puta u težinskom grafu stvari se malo kompliciraju. Kako put od x do y može proći svim bridovima, obično se problem pretvara u traženje najkraćih puteva od x do *svih* ostalih vrhova. U tome nam pomaže algoritam koji se pripisuje E. Dijkstri.

Duljine najkraćih pronađenih puteva spremaju se u vektor *duljina*. Na početku su duljine tih puteva *beskonačne*, tj. imaju neku veliku vrijednost *BESKONAČNO*. S je skup vrhova za koje je pronađena konačna vrijednost *duljina(v)*, a za preostale vrhove izvan tog skupa ta vrijednost se stalno *poboljšava*. Matrica susjedstva zadanog grafa je M. U njoj su, kao što je već navedeno, za težinski graf navedene težine bridova između vrhova. Ako između vrhova i i j ne postoji brid, $M(i,j) = M(j,i) = BESKONANO$.

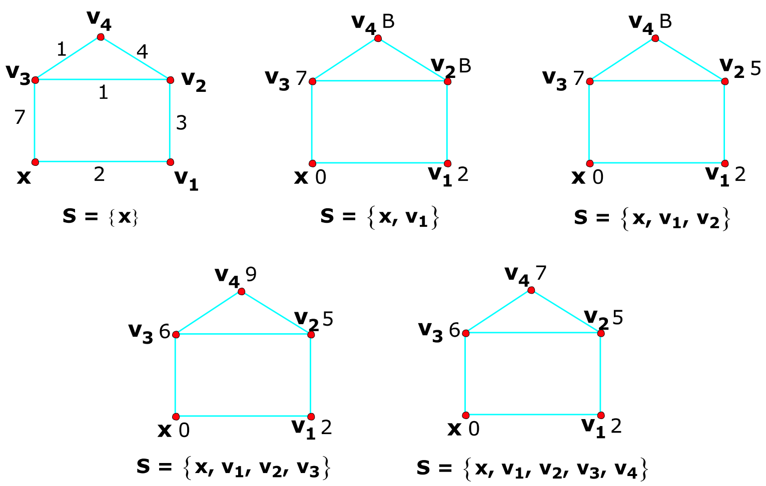
```
za svaki v iz V radi | duljina(v):= BESKONACNO duljina(x):= 0 S:=
{x}

u:= x dok nije prazan V\S radi | za svaki v iz V\S radi | |
duljina(v):= min{duljina(v),duljina(v) + M(u,v)} | izaberi u iz
V\S tako da je duljina(u) minimalna | S:= S + {u}
```

Pogledajmo što se događa na slici 3. Na prvoj sličici vidimo zadani graf s težinama napisanim uz bridove. Njezina matrica (ako uzmemo da je x vrh v_5) je

$$M := \begin{bmatrix} 0 & 3 & B & B & 2 \\ 3 & 0 & 1 & 4 & B \\ B & 1 & 0 & 1 & 3 \\ B & 4 & 1 & 0 & B \\ 2 & B & 3 & B & 0 \end{bmatrix}.$$

Na ostalim sličicama uz vrhove je napisana vrijednost iz vektora *duljina*, dakle najkraći pronađeni put od x (B je $BESKONACNO$).



Slika 3.

Što dalje?

Uz navedena dva algoritma postoji još mnogo drugih algoritama i postupaka za manipulaciju grafovima koji, ovisno o situaciji, mogu biti brži i efikasniji, no mogu biti i puno sporiji. Nadam se da sam vas ovim relativno kratkim člankom uspio zainteresirati za teoriju grafova, a na vama je da je pokušate još više istražiti. U tome vam mogu pomoći knjige poput *Algorithms in C* Roberta Sedgewicka, nezaobilazne *The Art of Computer Programming* Donalda E. Knutha, gotovo beskrajnih resursa koje pruža Internet, i naravno, budući brojevi *PlayMath*-a.