

# Legrövidebb utak minden csúcspárra \*

Vadász Péter

## 1 Elméleti összefoglaló [2]

Adott egy élsúlyozott gráf, keressük minden csúcspárra a két csúcs között lévő legrövidebb utat. Hasonló feladatot lehet elképzelni egy autóstérkép készítésekor is. Irányított gráf esetén megengedünk negatív éleket, negatív összsúlyú köröket azonban nem, irányítatlan esetben negatív éleket sem engedünk meg.

A feladatot megoldhatjuk a Dijkstra vagy a Bellman-Ford algoritmussal is, vizsgáljuk meg, hogy ez milyen futási idő költséggel járna:

- **Dijkstra algoritmus:** futtassuk le a Dijkstra algoritmust, a gráf összes csúcsára mint start csúcsra. Ha az algoritmus által használt prioritásos sort bináris kupac segítségével reprezentáljuk, akkor a futásidő ritka gráfokra  $O(n(n + m) \log n) = O(n^2 \log n)$ , sűrű gráfokra  $O(n^3 \log n)$
- **Bellman-Ford algoritmus:** futtassuk a Bellman-Ford algoritmust a gráf összes csúcsára, mint start csúcsra. Ritka gráfok esetén a futási idő:  $O(n * n * m) = O(n^3)$ , sűrű gráfokra  $O(n^4)$

Láthatjuk, hogy sűrű gráfok esetén a fenti módszerek egyike sem túl hatékony, ezért egy olyan módszert is használhatunk, ami minden gráf esetén  $\Theta(n^3)$  költséggel működik.

---

\*Dr Ásványi Tibor jegyzete alapján

## 2 Floyd-Warshall algoritmus

Tegyük fel, hogy  $G$  gráf csúcsmátrixos reprezentációval adott. Az egyes csúcsok közötti távolságok számításához egy  $D$ -vel jelölt mátrixot fogunk használni,  $D_{ij}$  az  $i$  és  $j$  csúcs közti optimális út hosszát tartalmazza. Egy  $\pi$  mátrixban a közvetlen megelőző csúcsokat tároljuk,  $\pi_{ij}$  az  $i$  és  $j$  csúcs között vezető optimális úton  $j$  csúcs közvetlen megelőzőjét tartalmazza. Az optimális utakat úgynevezett **belső csúcsok** mentén keressük.

**Definíció 2.1.** Belső csúcs [3] Egy  $p = \langle v_1, v_2, \dots, v_k \rangle$  út belső csúcsa minden  $v_1$ -től és  $v_k$ -től különböző csúcs. Tehát  $\{v_2, \dots, v_{k-1}\}$

Az algoritmus  $n$  lépésben határozza meg a csúcspárok közti legrövidebb utakat. A  $k$ -adik lépése után a  $D_{ij}^{(k)}$  az  $i$  és  $j$  csúcsok között lévő optimális út hosszát tartalmazza, úgy, hogy, a belső csúcsok címkéje legfeljebb  $k$ , a  $k$  címkéjű csúcs vagy szerepel az úton vagy nem, attól függően, hogy az út ettől optimálisabb lesz-e vagy sem.

A  $D^{(k)}$  és  $\pi^{(k)}$  mátrixok kiszámítása [2]:

$$D_{ij}^{(0)} = \begin{cases} 0, & \text{ha } i = j \\ w(i, j), & \text{ha } (i, j) \in G.E \wedge i \neq j \\ \infty, & \text{ha } i \neq j \wedge (i, j) \notin G.E \end{cases}$$

$$\pi_{ij}^{(0)} = \begin{cases} 0, & \text{ha } i = j \\ i, & \text{ha } (i, j) \in G.E \wedge i \neq j \\ 0, & \text{ha } i \neq j \wedge (i, j) \notin G.E \end{cases}$$

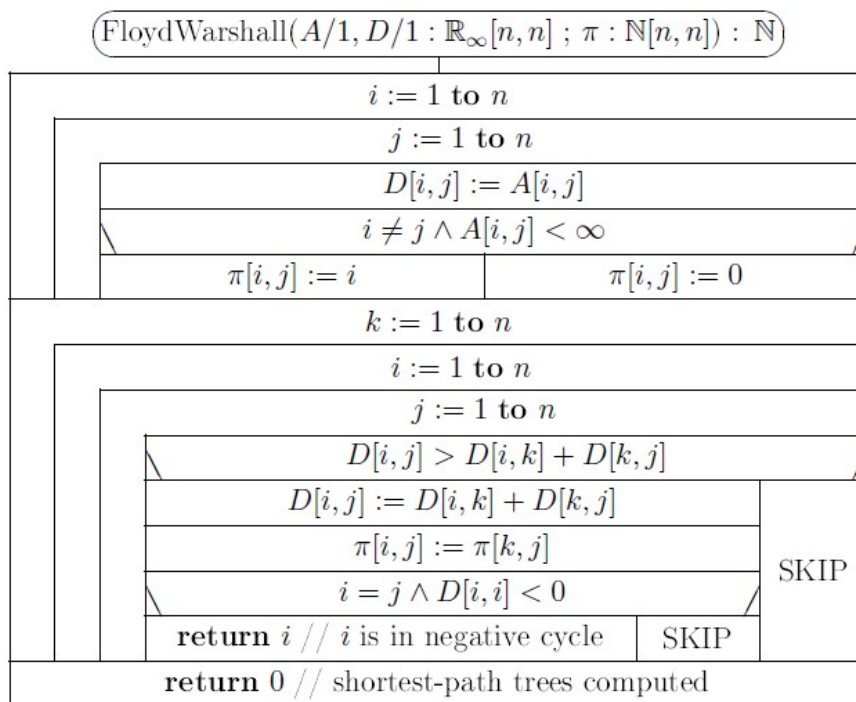
$D^{(0)}$  mátrix lényegében megegyezik a csúcsmátrixal.

$$D_{ij}^{(k)} = \begin{cases} D_{ik}^{(k-1)} + D_{kj}^{(k-1)}, & \text{ha } D_{ij}^{(k-1)} > D_{ik}^{(k-1)} + D_{kj}^{(k-1)} \\ D_{ij}^{(k-1)} & \text{különben} \end{cases}$$

$$\pi_{ij}^{(k)} = \begin{cases} \pi_{kj}^{(k-1)}, & \text{ha } D_{ij}^{(k-1)} > D_{ik}^{(k-1)} + D_{kj}^{(k-1)} \\ \pi_{ij}^{(k-1)} & \text{különben} \end{cases}$$

Az algoritmus mátrixpárok sorozatát állítja elő 0-tól  $n$ -ig  $n+1$  db mátrix. Az algoritmus működése során mégis elegendő egyetlen  $D$  és  $\pi$  mátrixot fenntartani, mivel a  $k$ -adik sor és  $k$ -adik oszlop a  $k-1$  és  $k$ -adik lépés között nem változik meg, hiszen a  $k$  címkéjű csúcsból

saját magán keresztül nem találhatunk rövidebb utat sehova, valamint a  $k$  címkéjű csúcsba sem találhatunk legrövidebb utat önmagán keresztül semelyik másik csúcsból sem.



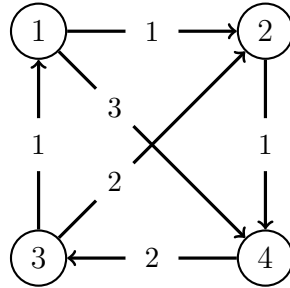
Ábra 1.: Floyd-Warshall algoritmus [1]

Floyd-Warshall algoritmus esetén, ha a gráf tartalmaz negatív összsúlyú kört, akkor a főátlóban negatív értékek jelennek meg, ezt az algoritmus is figyelembe veszi.

## 2.1 Példák

A feladatok megoldása során minden lépésben felrajzoljuk a mátrixokat, irányítatlan esetben a mátrixok szimmetrikusak. A  $k$ -adik lépésben végigmegyünk  $D$  mátrix celláin,  $k$ -adik sor és  $k$ -adik oszlop kivételével és megnézzük, hogy az  $(i, j)$  cella esetén  $(i, k)$  és  $(k, j)$  cellák összege, optimálisabb-e, mint  $(i, j)$ , ha igen, akkor módosítjuk az értéket ( $\pi$  mátrixban is).

**1. példa:** Szemléltessük a Floyd-Warshall algoritmus működését az alábbi gráfon! Adjuk meg  $D$  és  $\pi$  mátrixokat minden iterációban. Adjuk meg a csúcspárok közti optimális utakat is!



$D^{(0)}$	1	2	3	4
1	0	1	$\infty$	3
2	$\infty$	0	$\infty$	1
3	1	2	0	$\infty$
4	$\infty$	$\infty$	2	0

$\pi^{(0)}$	1	2	3	4
1	0	1	0	1
2	0	0	0	2
3	3	3	0	0
4	0	0	4	0

Láthatjuk, hogy  $D^{(0)}$  mátrix megegyezik a gráf csúcsmátrixos reprezentációjával.

$D^{(1)}$	1	2	3	4
1	0	1	$\infty$	3
2	$\infty$	0	$\infty$	1
3	1	2	0	4
4	$\infty$	$\infty$	2	0

$\pi^{(1)}$	1	2	3	4
1	0	1	0	1
2	0	0	0	2
3	3	3	0	1
4	0	0	4	0

$D^{(1)}$  és  $\pi^{(1)}$  mátrixokban dupla szegéllyel jelöltük azokat a sorokat, melyek nem változnak az első iterációban. Nézzük meg, hogyan határozhatjuk meg a kék színű cella értékét a piros színű cellák segítségével! A cella a mátrix 3. sorában és 4. oszlopában van, a fenti rekurzív képlet alapján, ha  $D_{34}^{(0)} > D_{31}^{(0)} + D_{14}^{(0)}$ , akkor  $D_{34}^{(1)} = D_{31}^{(0)} + D_{14}^{(0)}$ ,  $D_{31}^{(0)} + D_{14}^{(0)} = 4$ , ami határozottan kisebb, mint  $\infty$  ( $D_{34}^{(0)}$  értéke), tehát 4 lesz a cella új értéke. A  $\pi_{34}^{(1)}$  értékét  $\pi_{13}^{(0)}$  értéke határozza meg.

$D^{(2)}$	1	2	3	4
1	0	1	$\infty$	2
2	$\infty$	0	$\infty$	1
3	1	2	0	3
4	$\infty$	$\infty$	2	0

$\pi^{(2)}$	1	2	3	4
1	0	1	0	2
2	0	0	0	2
3	3	3	0	2
4	0	0	4	0

$D^{(3)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	1	$\infty$	2
<b>2</b>	$\infty$	0	$\infty$	1
<b>3</b>	1	2	0	3
<b>4</b>	3	4	2	0

$\pi^{(3)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	1	0	2
<b>2</b>	0	0	0	2
<b>3</b>	3	3	0	2
<b>4</b>	3	3	4	0

$D^{(4)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	1	4	2
<b>2</b>	4	0	3	1
<b>3</b>	1	2	0	3
<b>4</b>	3	4	2	0

$\pi^{(4)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	1	4	2
<b>2</b>	3	0	4	2
<b>3</b>	3	3	0	2
<b>4</b>	3	3	4	0

Az egyes csúcsok közötti legrövidebb utakat a  $\pi^{(4)}$  mátrix alapján tudjuk meghatározni:

- $1 \rightsquigarrow 2$ :  $1 \rightarrow 2$  [összsúly: 1],  $D_{12}^{(4)}$  érték alapján
- $1 \rightsquigarrow 3$ :  $1 \rightarrow 2 \rightarrow 4 \rightarrow 3$  [összsúly: 4] ( $\pi_{13}^{(4)}$  értéke 4, azaz az 1-es és 3-as csúcs között vezető optimális úton 3-as csúcs közvetlen megelőzője a 4-es csúcs. Most meg kell néznünk, hogy hogyan jutunk a 4-es csúcsba, azaz mi a 4-es megelőzője, ezt a  $\pi_{14}^{(4)}$  érték mutatja, ami 2, végül pedig a 2-es csúcs megelőzője kell, amit a  $\pi_{12}^{(4)}$  értéke ad meg, ez már 1, tehát megtaláltuk az  $1 \rightsquigarrow 3$  úton szereplő összes csúcsot.)
- $1 \rightsquigarrow 4$ :  $1 \rightarrow 2 \rightarrow 4$  [összsúly: 2]
- $2 \rightsquigarrow 1$ :  $2 \rightarrow 4 \rightarrow 3 \rightarrow 1$  [összsúly: 4]
- $2 \rightsquigarrow 3$ :  $2 \rightarrow 4 \rightarrow 3$  [összsúly: 3]
- $2 \rightsquigarrow 4$ :  $2 \rightarrow 4$  [összsúly: 1]
- $3 \rightsquigarrow 1$ :  $3 \rightarrow 1$  [összsúly: 1]
- $3 \rightsquigarrow 2$ :  $2 \rightarrow 3 \rightarrow 2$  [összsúly: 2]
- $3 \rightsquigarrow 4$ :  $3 \rightarrow 2 \rightarrow 4$  [összsúly: 3]
- $4 \rightsquigarrow 1$ :  $4 \rightarrow 3 \rightarrow 1$  [összsúly: 3]
- $4 \rightsquigarrow 2$ :  $4 \rightarrow 3 \rightarrow 2$  [összsúly: 4]

- $4 \rightsquigarrow 3: 4 \rightarrow 3$  [összsúly: 2]

### 3 Gráf tranzitív lezártja

**Definíció 3.1.** Tranzitív lezárt: Egy  $G(V, E)$  gráf tranzitív lezártja egy  $G'(V, E')$  gráf, ahol  $(u, v) \in G'.E'$  akkor és csak akkor, ha  $G$ -ben vezet út  $u$ -ból  $v$ -be.

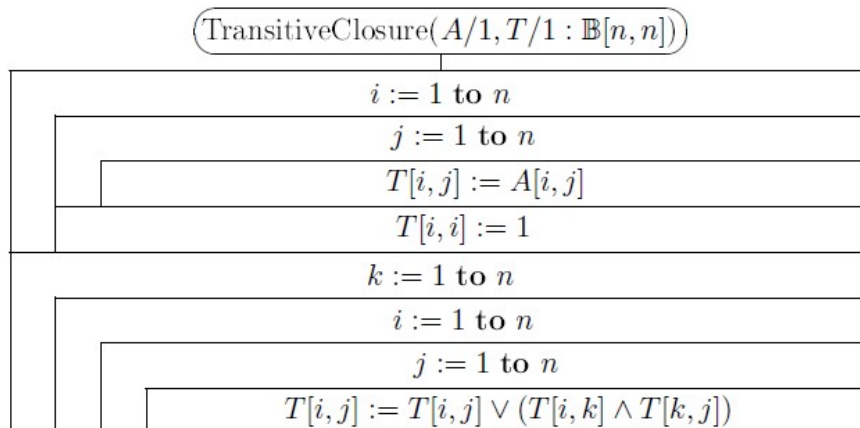
A tranzitív lezárt kiszámításához használhatjuk a Warshall algoritmust, amely a Floyd-Warshall algoritmus egy módosított változata. Egy  $T$   $n \times n$ -es logikai mátrixot használunk, az élsúlyok most nem számítanak, a gráf lehet irányított, irányítatlan.

A Floyd-Warshall algoritmushoz hasonlóan a Warshall algoritmus mátrixok sorozatát állítja elő ( $n$  iterációval), de a korábbi indoklás miatt itt is elég egy mátrixot használni. A  $k$ -edik iteráció után  $T_{ij}^{(k)}$  értéke igaz, ha az  $i$  és  $j$  címkéjű csúcsok között vezet olyan út, amelyben a belső csúcsok címkéje legfeljebb  $k$ .

A  $T^{(k)}$  mátrixok kiszámítása (1 jelentése "igaz", 0 jelentése "hamis"):

$$T_{ij}^{(0)} = \begin{cases} 1, & \text{ha } i = j \vee (i, j) \in G.E \\ 0, & \text{különben} \end{cases}$$

$$T_{ij}^{(k)} = T_{ij}^{(k-1)} \vee (T_{ik}^{(k-1)} \wedge T_{kj}^{(k-1)})$$

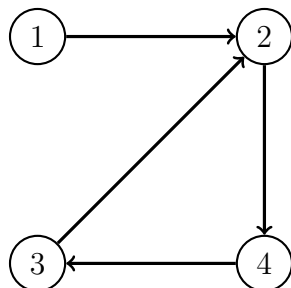


Ábra 2.: Warshall algoritmus [1]

Futási idő:  $\Theta(n^3)$

### 3.1 Példák

**2. példa:** Szemléltessük a Warshall algoritmus működését az alábbi gráfon! Adjuk meg  $T$  mátrixot az inicializálása és a fő ciklus mindegyik iterációja után is!



$T^{(0)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	1	1	0	0
<b>2</b>	0	1	0	1
<b>3</b>	0	1	1	0
<b>4</b>	0	0	1	1

$T^{(1)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	1	1	0	0
<b>2</b>	0	1	0	1
<b>3</b>	0	1	1	0
<b>4</b>	0	0	1	1

$T^{(2)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	1	1	0	1
<b>2</b>	0	1	0	1
<b>3</b>	0	1	1	1
<b>4</b>	0	0	1	1

$T^{(3)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	1	1	0	1
<b>2</b>	0	1	0	1
<b>3</b>	0	1	1	1
<b>4</b>	0	1	1	1

$T^{(4)}$	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	1	1	1	1
<b>2</b>	0	1	1	1
<b>3</b>	0	1	1	1
<b>4</b>	0	1	1	1

## Irodalomjegyzék

- [1] Dr Ásványi Tibor *Algoritmusok és adatszerkezetek II. előadásjegyzet - Élsúlyozott gráfok és algoritmusai*
- [2] Koruhely Gábor, Szalay Richárd *Algoritmusok és adatszerkezetek 2*  
(hallgatói jegyzet, lektorált és javított)  
<http://aszt.inf.elte.hu/~asvanyi/ad/>
- [3] Fekete István, Hunyadvári László *Algoritmusok és adatszerkezetek*  
<http://tamop412.elte.hu/tananyagok/algoritmusok/index.html>