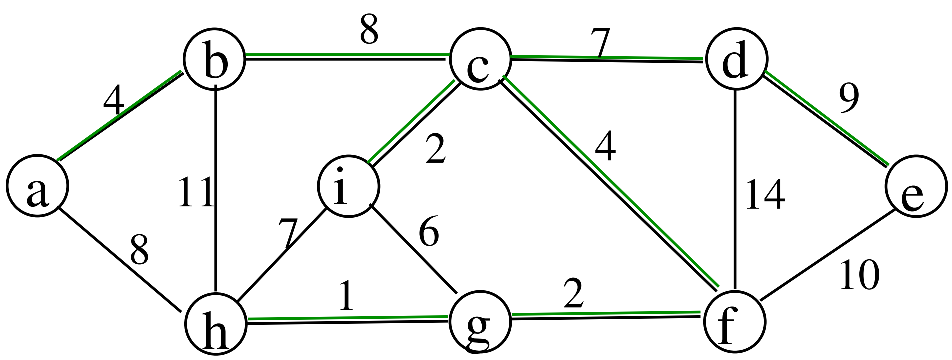
12. Minimális feszítőfák

Legyen $G=(V,E,c)$, $c:E\to\mathbb{R}$ súlyozott irányítatlan gráf.
Terjesszük ki a c súlyfüggvényt a $T\subseteq E$ élhalmazokra:

$$C(T)=\sum_{(u,v)\in T} c(u,v)$$

Az $F=(V,T)$ gráf minimális feszítőfája G-nek, ha
1. F feszítőfája G-nek, és
2. $C(T)\to$ minimális



1. ábra. Példa minimális feszítőfára

Input: $G=(V,E,c);c:E\to\mathbb{R}$ irányítatlan gráf
Output: $MFF=(V,F)$; minimális feszítőfája G-nek

Legyen $A\subseteq F$ valamely $MFF=(V,F)$ M.F.F.-nak és $(u,v)\in E$.
Def. (u,v) *biztonságos él* A-ra nézve, ha $A\bigcup\{(u,v)\}$ is része valamely MFF' M.F.F.-nak.
Elvi algoritmus

$A:=\emptyset$
While A nem M.F.F. Do Begin
    (u,v) biztonságos él keresése;
    $A:=A\bigcup\{(u,v)\}$
End

Def. A $G=(V,E)$ gráf *vágása*: $(S,V-S)$; $S\subseteq V$;
Def. $(u,v)\in E$ *keresztél* az $(S,V-S)$ vágásra, ha $u\in S$ és $v\in V-S$, vagy $u\in V-S$ és $v\in S$
Def. Az $(S,V-S)$ vágás *elkerüli* az $A\subseteq E$ élhalmazt, ha A-ban nincs keresztél.
Def. (u,v) *könnyű* él az $(S,V-S)$ vágásra, ha a legkisebb c-értékű (súlyú) keresztél.

12.1. tétel. *Ha A része a $G=(V,E,c)$ v.m. M.F.F.-jának és elkerüli az $(S,V-S)$ vágást, továbbá (u,v) könnyű él az $(S,V-S)$ vágásra, akkor (u,v) biztonságos él A-ra nézve.*

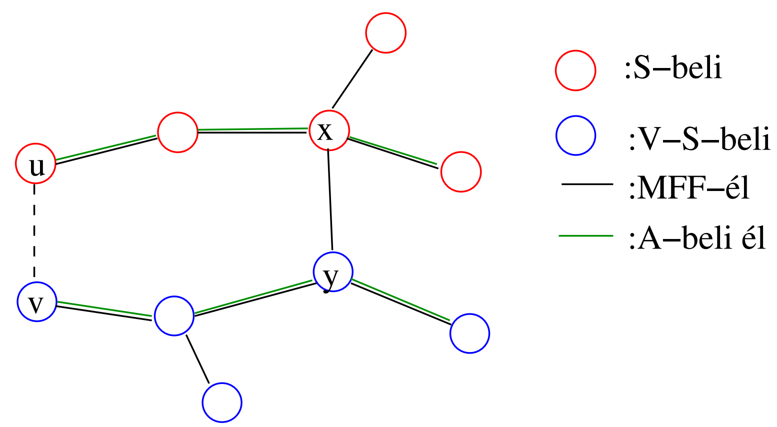
Bizonyítás. Legyen $T=(V,F)$ egy olyan M.F.F., hogy $A\subseteq F$. Ha $(u,v)\in F$, akkor készenvagyunk. Tfh. $(u,v)\notin F$, tehát (u,v) hozzávétele az F faélekhez kört képez.Mivel u és v az S vágás különböző oldalán van, ezért van a körben (x,y) keresztél. Az

$$F':=F-\{(x,y)\}\cup\{(u,v)\}$$

élhalmaz is feszítófája lesz G-nek, és $c(u,v)\leq c(x,y)$ miatt

$$C(F'):=C(F)-c(x,y)+c(u,v)\leq C(F)$$

De F minimális volt, így $C(F')=C(F)$. ∎



2. ábra.

12.2. következmény. *Ha A része a $G = (V,E,c)$ v.m. M.F.F.-nak és az $(u,v) \in E$ élre u és v az A erdő két különböző fájában van és c-értéke (súlya) minimális, akkor az (u,v) él biztonságos A-ra.*

12.1. A Kruskal algoritmus

A biztonságos él mohó választása: mindig a legkisebb súlyú olyan élet válaszjuk, amely nem képez kört.

Hogyan dönthető el, hogy adott (u,v) él nem képez kört? Azaz ugyanazon fához tartoznak-e?

Az UnioHolvan absztrakt adattípus.

Értékhalmaz: $UnioHolvan = \{\{H_1,\dots,H_k\} : H_i \subseteq E, i = 1,\dots,k, i \neq j \Rightarrow H_i \cap H_j = \emptyset\}$

Műveletek:

$$S : UnioHolvan, X : Elemtip, N, N1, N2 : NevTip$$

$$\begin{array}{rcl}
\{Igaz\} & Letesit(S) & \{S = \emptyset\} \\
\{S = S\} & Megszuntet(S) & \{Igaz\} \\
\{S = S\} & Uresit(S) & \{S = \emptyset\} \\
\{S = \{H_1,\dots,H_k\} \wedge X \in \cup H_i\} & Holvan(S,X,N) & \}\{N = Y \wedge (X \in H_i \wedge X \in H_i)\} \\
\{S = \{H_1,\dots,H_k\} \wedge X \notin \cup H_i\} & Holvan(S,X,N) & \{N = X \wedge S = Pre(S) \cup \{\{X\}\} \\
\{S = \{H_1,\dots,H_k\} \wedge N1 \in H_i \wedge N2 \in H_j\} & Unio(S,N1,N2) & \{S = Pre(S) - H_i - H_j \cup \{H_i \cup H_j\}\}
\end{array}$$

```
Procedure Kruskal (    G       : GrafC.Tipus;
                   Var Fa      : Graf.Tipus ;
                   Var Minsuly: Real       );
Var
  H : UnioHolvan.Tipus;
  N1,N2 : UnioHolvan.Nevtip;
  Q: Prisor.Tipus;
Begin{Kruskal}
  Graf.Letesit(Fa); UnioHolvan.Letesit(H);
  Prisor.Letesit(Q);
  Minsuly:=0.0;
  For (u,v) In E Do Sorba(Q,u,v);
  While Elemszam(Q)>0 Do Begin
    Sorbol(Q,u,v);
    Holvan(H,u,N1);
    Holvan(H,v,N2);
    If N1<>N2 Then Begin
```

```
      Minsuly:=Minsuly + C(u,v);
      Elbovit(Fa,u,v);
      Unio(H,N1,N2);
    End
   End{While};
   Megszuntet(H);
End{Kruskal};
```

12.3. tétel. *Ha a G gráf összefüggő, akkor a* KRUSKAL *algoritmus minimális feszítőfát állít elő.*

Bizonyítás. Legyen $S := \{x : u$ és x ugyanazon fában van. $\}$ Ekkor az (u,v) élre és az $(S,V-S)$ vágásra teljesül a 12.1. tétel. ■

A KRUSKAL algoritmus futási ideje $O(E \lg E)$.

12.2. A Prim algoritmus

A biztonságos él választása: Ha $T = (S,F)$ részfája $G = (V,E,c)$ egy minimális feszítőfájának, és (u,v) egy legkisebb súlyú olyan él, hogy $u \in S$, $v \notin S$ és $c(u,v)$, akkor (u,v) biztonságos él.

```
Procedure Prim(G:Graf; r:PontTip; Var Apa:Fa);
Var
 D:Fuggyveny; S:PriSor; Kesz:Halmaz;
Begin{Prim}
  Fuggveny.Letesit(D); PriSor.Letesit(S); Halmaz.Letesit(Kesz);
  For v<>r In V Do Begin
    D[v]:=C(r,v);
    Apa[v]:=Nul;
    Sorba(S,v);
  End;
  D[r]:=0;
  Sorba(S,r);
  While Elemszam(S) > 0 Do Begin
    Sorbol(S,u);
    Bovit(Kesz,u);
    For v In Ki(G,u) Do
      If Not Eleme(Kesz,v) And (C[u,v]<D[v]) Then Begin
        D[v]:=C[u,v];
        Apa[v]:=u;
        Modosit(S, v, D[v]);
      End;
  End{while};
End{Prim};
```

A PRIM algoritmus futási ideje $O(E \lg V)$, feltéve
1. ELEME, BOVIT halmaz műveletek lr. esetben $O(\lg V)$ idejűek,
2. SORBA, MODOSIT $O(\lg V)$ idejű.

12.3. Egy pontból iduló legrövidebb utak

Def. Legrövidebb Utak Feszítőfája (LUF).
$G = (V,E,c); c : E \to \mathbb{R}$ irányított gráf, és $s \in V$
Az $F = (V,F)$ fa a G gráf s-gyökerű LUF-ja, ha F s-gyökerű feszítőfája G-nek, és $\forall v \in V$ az $s \rightsquigarrow v$ egyetlen út F-ben egy legrövidebb út G-ben.

Probléma:
Input: $G = (V,E,c); c : E \to \mathbb{R}$ irányított gráf, $s \in V$

Output:
1. $D[v] := \delta(s,v)(\forall v \in V)$;
2. $\pi : V \to V$;
$F = (V, \{(\pi[v], v) : v \in V, \pi[v] \neq Nul\})$ s-gyökerű LUF-ja G-nek
($s = v_0 \to v_1 \to \cdots \to v_k = v$, $\pi[v_i] = v_{i-1} (i = 1, \cdots, k)$ egy legrövidebb $s \rightsquigarrow v$ út.)

12.3.1. Fokozatos közelítés.

Legyen Q a $D[v]$-szerinti minimumos módosítható prioritási sor.

```
Procedure Kezd(s:PontTip); Var v:PontTip;
Begin{Kezd}
  For v<>s In V Do Begin
    D[v]:=Inf;
    Apa[v]:=Null;
    PriSor.Sorba(Q,v);
  End;
  D[s]:=0;
  PriSor.Sorba(Q,s);
End{Kezd};

Procedure Kozelit(u,v:PontTip);
Var Dv:Real;
Begin{Kozelit}
  Dv:=D[u]+C[u,v];
  If Dv<D[v] Then Begin
    D[v]:=Dv;
    Apa[v]:=u;
    PriSor.Modosit(Q,v,Dv);
  End;
End{Kozelit};
```

A legrövidebb utak tulajdonságai

Tegyük fel, hogy a $G = (V, E, c)$ írányított, súlyozott gráfra és s kezdőpontra vegrejatottul a KEZD eljárást, majd valahány KOZELIT(u,v) műveletet. Ekkor az alábbi összefüggásek teljesülnek.

Háromszög egyenlőtlenség. $(\forall (u,v) \in E)$ $(\delta(s,v) \leq \delta(s,u) + c(u,v))$.

Felső korlát tulajdonság. $D[v] \geq \delta(s,v)$ és ha egyszer $D[v] = \delta(s,v)$, akkor ezután mindig teljesül a $D[v] = \delta(s,v)$ egyenlőség.

Nincs-út tulajdonság. Ha nincs $s \rightsquigarrow v$ út, akkor mindvégig $D[v] = \delta(s,v) = \infty$ (=Inf) teljesül.

Konvergencia tulajdonság. Ha $s \rightsquigarrow u \to v$ egy legrövidebb út és $D[u] = \delta(s,u)$ *Kozelit*(u, v) végrehajtása előtt, akkor *Kozelit*(u, v) után $D[v] = \delta(s,v)$.

Út-közelítés tulajdonság. Ha $p = \langle v_0, v_1, \ldots, v_k \rangle$ egy $s = v_0 \rightsquigarrow v_k$ legrövidebb út, akkor a $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$ élekre ebben a sorrendben végrehajtott KOZELIT után $D[v_k] = \delta(s, v_k)$.

LUF tulajdonság. Tegyük fel, hogy G nem tartalmaz s-ből elérhető negatív összsúlyú köröket és minden $v \in V$ pontra $D[v] = \delta(s,v)$. Ekkor az $F = \{(Apa[v], v) : v \in V, Apa[v] \neq Null\}$ élhalmaz G-nek s gyökerű LUF-ja lesz.

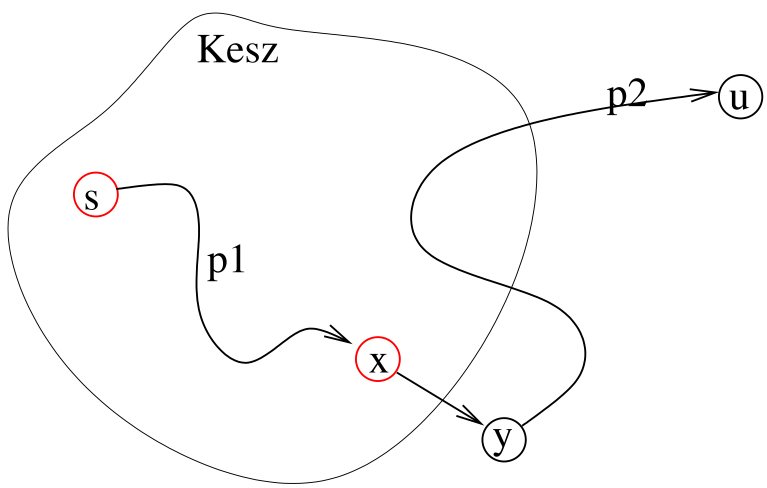
```
Procedure Dijkstra(Const G:Graf; s:PontTip; Var D:Fuggveny; Var Apa:Fa);
Var
  Kesz:Halmaz;
```

```
  Q:PriSor;
  u,v:PontTip;
Begin{Dijkstra}
  Fuggveny.Letesit(D);
  Fuggveny.Letesit(Apa);
  PriSor.Letesit(Q);
  Halmaz.Letesit(Kesz);
  Kezd;

  While Elemszam(Q) > 0 Do Begin
    Sorbol(Q,u);
    Bovit(Kesz,u);
    For v In Ki(G,u) Do
      If Not Eleme(Kesz,v) Then
        Kozelit(u,v);
  End{while};
End{Dijkstra};
```

12.4. tétel. *A* DIJKSTRA *algoritmust nemnegatív élsúlyozott irányított $G = (V, E, c)$ gráfra és s kezdőpontra végrehajtva, minden $v \in V$ pontra tejesül a $D[v] = \delta(s, v)$.*

Bizonyítás. Bizonyítás a `Kesz` halmazba kerülés szerinti indukcióval.
Az első pont, amely kikerül a Q prioritási sorbol és bekerül `Kesz`-be az s pont, amelyre ekkor $D[s] = 0 = \delta(s, s)$. Tfh. u az első



3. ábra.

olyan pont, amelynek a `Kesz` halmazba kerülésekor $D[u] \neq \delta(s, u)$. Mivel $s \neq u$, ezért közvetlenűl u-nak `Kesz`-be való berakásakor $Kesz \neq \emptyset$. Létezik $s \rightsquigarrow u$ út, mert különben a nincs-út tulajdonság miatt $D[u] = \delta(s, u) = \infty$ lenne, ami ellentmond a $D[u] \neq \delta(s, u)$ feltételezésnek. Tehát van $p : s \rightsquigarrow u$ legrövidebb út is. Az u pontnak `Kesz`-be kerülése előtt a p út összeköt egy `Kesz`-beli (s) és egy nem `Kesz`-beli (u) pontot. Legyen y a p úton az első olyan pont, amely nincs `Kesz`-ben, és legyen x a p úton y-t megelőző pont. A 3. ábra mutatja a helyzetet. Tehát a $p : s \rightsquigarrow u$ út felbontható $p : s \overset{p_1}{\rightsquigarrow} x \to y \overset{p_2}{\rightsquigarrow} u$ -ra. Mivel u az első olyan pont, amelynek `Kesz`-be kerülésekor $D[u] \neq \delta(s, u)$, így $D[x] = \delta(s, x)$ amikor x-t `Kesz`-be rakjuk. Ez után végrehajtódik a `Kozelit(u,y)`, így a konvergencia tulajdonság miatt $D[y] = \delta(s, y)$. Mivel minden él súlya nemnegatív, és y előbb van a p útban, mint u, ezért $\delta(s, y) \leq \delta(s, u)$. Tehát

$$\begin{aligned} D[y] &= \delta(s, y) \\ &\leq \delta(s, u) \\ &\leq D[u] \end{aligned}$$

Mivel u is és y is a Q prioritási sorban van, és u előbb került ki a sorból, mint y, így $D[u] \leq D[y]$. Tehát az előző egyenlőtlenségeknek egyenlőségeknek kell lenni.

$$D[y] = \delta(s,y) = \delta(s,u) = D[u]$$

Ez ellentmond a feltevésünknek. ■

A DIJKSTRA algoritmus futási ideje: $O(E \lg V)$, feltéve, hogy a Q prioritási sor `Sorba`,`Sorbol` és `Modosit` műveletek futási ideje $O(\lg n)$ n elemű sorra.

A legrövidebb utak tulajdonságainak bizonyítása

12.5. lemma. *Háromszög egyenlőtlenség.* $(\forall (u,v) \in E)\ (\delta(s,v) \leq \delta(s,u) + c(u,v))$

Bizonyítás. Mivel $s \rightsquigarrow u \rightarrow v$ is $s \rightsquigarrow v$ út, így $\delta(s,v) \leq \delta(s,u) + c(u,v)$. ■

12.6. lemma. *Felső korlát tulajdonság. A $G = (V,E,c)$ irányított, súlyozott gráfra és s kezdőpontra a* KEZD *eljárás után akármilyen* KOZELIT *műveletsor után $D[v] \geq \delta(s,v)$ és ha egyszer $D[v] = \delta(s,v)$, akkor ezután mindig teljesül a $D[v] = \delta(s,v)$ egyenlőség.*

Bizonyítás. Az egyenlőtlenség teljesülését a végrehajtott KOZELIT lépések száma szerint indukcióval bizonyítjuk. Kezdetben $D[s] = 0 \geq \delta(s,s)$ és minden más v pontra $D[v] = \infty \geq \delta(s,v)$.
Az indukciós lépésnél tekintsünk egy (u,v) éllel való KOZELIT(U,V) közelítést. Csak $D[v]$ értéke változhat:

$$\begin{aligned} D[v] &= D[u] + c(u,v) \\ &\geq \delta(s,u) + c(u,v) \\ &\geq \delta(s,v) \end{aligned}$$

Továbbá, $D[v]$ az alsó $\delta(s,v)$ korlátot elérve tovább nem csökkenhet, mert $D[v[\geq \delta(s,v)$, de nem is nőhet. ■

12.7. lemma. *Nincs-út tulajdonság. A $G = (V,E,c)$ irányított, súlyozott gráfra és s kezdőpontra a* KEZD *eljárás után akármilyen* KOZELIT *műveletsor után, ha nincs s $\rightsquigarrow$ v út, akkor mindvégig $D[v] = \delta(s,v) = \infty$ (=Inf) teljesül.*

Bizonyítás. A felsőkorlát tulajdonság miatt $\infty = \delta(s,v) \leq D[v]$, így $D[v] = \infty = \delta(s,v)$. ■

12.8. lemma. *A $G = (V,E,c)$ irányított, súlyozott gráfra és s kezdőpontra a* KEZD *eljárás után akármilyen* KOZELIT(U,V) *végrehajtás után $D[v] \leq D[u] + c(u,v)$ teljesül.*

Bizonyítás. Ha közvetlenül KOZELIT(U,V) előtt $D[v] > D[u] + c(u,v)$, akkor utána $D[v] = D[u] + c(u,v)$ lesz. Ellenben, ha $D[v] \leq D[u] + c(u,v)$, akkor sem $D[v]$ sem $D[u]$ nem változik. ■

12.9. lemma. *Konvergencia tulajdonság. A $G = (V,E,c)$ irányított, súlyozott gráfra és s kezdőpontra a* KEZD *eljárás után akármilyen* KOZELIT *műveletsor után, ha s $\rightsquigarrow$ u $\rightarrow$ v egy legrövidebb út és $D[u] = \delta(s,u)$ Kozelit(u, v) végrehajtása előtt, akkor Kozelit(u, v) után $D[v] = \delta(s,v)$*

Bizonyítás. A felsőkorlát tulajdonság szerint ha $D[u] = \delta(s,u)$ fenáll KOZELIT(U,V) előtt, akkor utána is fenáll. Az (u,v) élű közeklítés után:

$$\begin{aligned} D[v] &\leq D[u] + c(u,v) \\ &= \delta(s,u) + c(u,v) \\ &= \delta(s,v) \end{aligned}$$

■

12.10. lemma. *Út-közelítés tulajdonság. Ha $p = \langle v_0, v_1, \ldots, v_k \rangle$ egy $s = v_0 \rightsquigarrow v_k$ legrövidebb út, akkor a $(v_0,v_1),(v_1,v_2),\ldots,(v_{k-1},v_k)$ élekre ebben a sorrendben végrehajtott* KOZELIT *után $D[v_k] = \delta(s,v_k)$.*

Bizonyítás. Indukcióval mutatjuk meg, hogy a p út i-edik élével való közelítés után $D[v_i] = \delta(s,v_i)$ teljesül. A kezdeti értékadás miatt $D[v_0] = D[s] = 0 = \delta(s,s)$.
Tfh. $D[v_{i-1}] = \delta(s,v_{i-1})$, és tekintsük a (v_{i-1},v_i) éllel való közelítést. Ezen közelítés utána a konvergencia tulajdonság miatt fenáll $D[v_i] = \delta(s,v_i)$. ■

12.11. lemma. *Legyen $G = (V, E, c)$ irányított, súlyozott gráf és $s \in V$ kezdőpont. és tegyük fel, hogy G nem tartalmaz s-ből elérhető negatív összsúlyú köröket. Ekkor a* Kezd *eljárás után akármilyen* Kozelit *műveletsort végrehajtva az $\{(Apa[v], v) : v \in V, Apa[v] \neq Null\}$ élhalmaz G egy részfája lesz.*

Bizonyítás. Indirekt bizonyítunk, tfh. kialakul egy $r = \langle v_0, v_1, \ldots, v_k \rangle$, $v_0 = v_k$ kör, ahol $Apa[v_i] = v_{i-1}$, $i = 1, \ldots, k$. Feltehetjük, hogy a Kozelit(v_{k-1}, v_k) végrehajtása hozta létre a kört. Minden körbeli v_i pontra $Apa[v_i] \neq Null$, így $D[v_i] \neq \infty$. Tehát a felsőkorlát tulajdonság miatt a körbeli pontok elérheőek s-ből. Tekintsük a $D[v_i]$ értékeket közvetlenűl a Kozelit(v_{k-1}, v_k) végrehajtása előtt. $D[v_i]$ utolsó módosítása a $[v_i] := D[v_{i-1}] + c(v_{i-1}, v_i)$ értékadás volt. Ha eztán $D[v_i]$ megváltozott, csak csökkenhetett. Ezért Kozelit(v_{k-1}, v_k) végrehajtása előtt minden $i = 1, \ldots, k$-ra

$$D[v_i] \geq D[v_{i-1}] + c(v_{i-1} v_i).$$

$Apa[v_k]$ azért kapott értéket, mert teljesült

$$D[v_k] > D[v_{k-1}] + c(v_{k-1} v_k)$$

$$\begin{aligned} \sum_{i=1}^{k} D[v_i] &> \sum_{i=1}^{k} D[v_{i-1}] + c(v_{i-1}, v_i) \\ &= \sum_{i=1}^{k} D[v_{i-1}] + \sum_{i=1}^{k} c(v_{i-1}, v_i) \end{aligned}$$

De

$$\sum_{i=1}^{k} D[v_i] = \sum_{i=1}^{k} D[v_{i-1}]$$

így

$$0 > \sum_{i=1}^{k} c(v_{i-1}, v_i)$$

ami ellentmond azon feltevésünknek, hogy G-ben nincs negatív kör. ∎

12.12. lemma. *LUF tulajdonság. Legyen $G = (V, E, c)$ irányított, súlyozott gráf és $s \in V$ kezdőpont, és tegyük fel, hogy G nem tartalmaz s-ből elérhető negatív összsúlyú köröket. A* Dijkstra *algoritmust végrehajtva a $G = (V, E, c)$ gráfra és s kezdőpontra, az $F = \{(Apa[v], v) : v \in V, Apa[v] \neq Null\}$ élhalmaz G-nek s gyökerű LUF-ja lesz.*

Bizonyítás. Az előző lemma szerint F s-gyökerű részfája G-nek.
Először lássuk be, hogy F pontjai éppen az s-ből elérhető pontok. $\delta(s, v) < \infty$ akkor és csak akkor, ha van $s \rightsquigarrow v$ út. De $D[v]$ akkor és csak akkor kap nem ∞ értéket, ha $Apa[v] \neq Null$, azaz F pontjai éppen az s-ből elérhető pontok G-ben.
Azt kell még bizonyítani, hogy minden $v \in V$ pontra, amelyre $Apa[v] \neq Null$, az egyetelen $s \overset{p}{\rightsquigarrow} v$ út F-ben egy legrövidebb út G-ben. Legyen $p = \langle v_0, v_1, \ldots, v_k \rangle$, ahol $v_0 = s$, $v_k = v$. Minden $i = 1, \ldots, k$-ra $D[v_i] = \delta(s, v_i)$ és $D[v_i] \geq D[v_{i-1}] + c(v_{i-1}, v_i)$, amiből következik, hogy $c(v_{i-1}, v_i) \leq \delta(s, v_i) - \delta(s, v_{i-1})$.

$$\begin{aligned} C(p) &= \sum_{i=1}^{k} c(v_{i-1}, v_i) \\ &\leq \sum_{i=1}^{k} \delta(s, v_i) - \delta(s, v_{i-1}) \\ &= \delta(s, v_k) - \delta(s, v_0) \\ &= \delta(s, v_k) \end{aligned}$$

Tehát $C(p) \leq \delta(s, v_k)$, ugyanakkor $C(p) \geq \delta(s, v_k)$, így $C(p) = \delta(s, v_k)$. ∎

12.4. Legrövidebb utak minden pontpárra

Input: $G = (V, E, c)$ irányított súlyozott gráf; $c : E \to \mathbb{R}^+$
Output:Minden u, v pontpárra $\delta(u, v)$ és egy $u \rightsquigarrow v$ legrövidebb út.

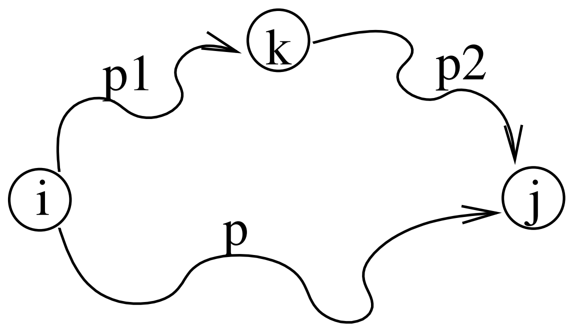
Megoldás dinamikus programozás módszerével.
Tfh. $V = 1..n$ és $c(i, j) = \infty$, ha $(i, j) \notin E$, azaz $G[i, j] = c(i, j)$ és $G[i, i] = 0, i = 1, \cdots, n$

Részproblémákra bontás:
$\forall i, j \in \{1, \ldots, n\}$-ra és $\forall k \in \{0, 1, \cdots, n\}$-re
$D^k(i, j) =$ az i-ből j-be vezető olyan utak hosszának minimuma, amelyek legfeljebb az $\{1, \cdots, k\}$ (belső) pontokon mennek keresztül.
$D^n(i, j) = \delta(i, j)$

Rekurzív összefüggés a részproblémák között.

$$\begin{aligned} D^0(i, j) &= G[i, j] \\ D^k(i, j) &= Min\{D^{k-1}(i, j), D^{k-1}(i, k) + D^{k-1}(k, j)\} \quad k > 0 \end{aligned}$$

$D[i, j]$ helyben számolása; egy D tömbben.



4. ábra. $p : i \rightsquigarrow j$ az {1..k-1} pontokon át haladó út, $p_1 : i \rightsquigarrow k$, $p_2 : k \rightsquigarrow j$ {1..k-1} pontokon át haladó legrövidebb út.

A k-adik iterációban felülírt elemek: $D^{k-1}(i, k)$ és $D^{k-1}(k, j)$.
De

$$D^k(i, k) = Min\{D^{k-1}(i, k), D^{k-1}(i, k) + D^{k-1}(k, k)\} = D^{k-1}(i, k)$$

```
{Globális programelemek a Floyd_Warshall algoritmushoz}
Const
  MaxN = ???  ;{a gráf pontjainak max. száma}
  Inf  = 10E12;{a végtelen reprezentánsa}
Type
  Graf = Array[1..MaxN,1..MaxN] Of Real;
  Utak = Array[1..MaxN,1..MaxN] Of Record
                                     tav : Real;
                                     elod : 0..MaxN
                                   End;

Procedure Floyd_Warshall(Const G : Graf;
                         Var   U : Utak );
  Var
    i,j,k : Integer;
    Ujtav : Real;
  Begin{Floyd_Warshall}
    For i := 1 To N Do         {inicializálás}
      For j := 1 To N Do Begin
```

```
        U[i,j].tav:=G[i,j];
        If (i<>j)And(G[i,j]<Inf) Then
          U[i,j].elod:=i
        Else
          U[i,j].elod:=0
      End;
    For k := 1 To N Do
      For i := 1 To N Do
        For j := 1 To N Do Begin
          Ujtav := U[i,k].tav + U[k,j].tav;
          If Ujtav < U[i,j].tav Then Begin
            U[i,j].tav := Ujtav;
            U[i,j].elod:= U[k,j].elod
          End
        End{for j}
  End{Floyd_Warshall};
```

```
Procedure MinUtIro(Const U : Utak; i,j : Integer);
Var
  S:Array[1..MaxN] of 0..MaxN;
  h,k:Word;
  Begin{MinUtIro}
    If U[i,j]=0 Then Begin
      WriteLn('Nincs 'i,'->',j,'út!');
      Exit;
    End;
    h:=0;
    Repeat
      Inc(h);
      S[h]:=j;
      j:=U[i,j];
    Until j=i;
    Write(i);
    For k:=h DownTo 1 Do
      Write('->',S[k]);
    writeLn;
  End{MinUtIro};
```

A Floyd-Warshall algoritmus futási ideje $\Theta(n^3)$.

12.5. Irányított gráf tranzitív lezártja

A $G=(V,E)$ gráf tranzitív lezártja aza $G^*=(V,E^*)$ gráf, ahol

$$E^* = \{(u,v) : u \overset{G}{\rightsquigarrow} v\}$$

Egy lehetséges algoritmus G^* kiszámítására: vegyük azt a gráfot, amelyben minden létező él súlya 1, és alkalmazzuk a Floyd-Warshall algoritmust. Az i és j pont között akkor és csak akkor van út G-ben, ha távolságuk nem ∞.
Azonban a Floyd-Warshall algoritmus egyszerű módosításával hatékonyabb megoldást kapunk.

```
{ Globális programelemek a Warshall eljáráshoz :
Const
  MaxN = ???   ;{A gráf pontjainak max. száma}
Type
  Graf = Array [1..MaxN,1..MaxN] Of Boolean;
} Procedure Warshall(Const G:Graf; N:Word; Var T:Graf);
```

```
Var
  i,j,k:integer;
Begin
  T:=G;
  For k:=1 to N do
    For i:=1 To N Do
        For j:=1 To N Do
           If Not T[i,j] Then
             T[i,j]:=T[i,k] And T[k,j]
End{Warshall};
```

A WARSHALL algoritmus futási ideje $\Theta(n^3)$.