# Algoritmusok és adatszerkezetek I. 6. Előadás

MST, általános algoritmus, biztonsá-gos élek, Kruskal algoritmus, unió-holvan adatszerkezet.

# Tartalom

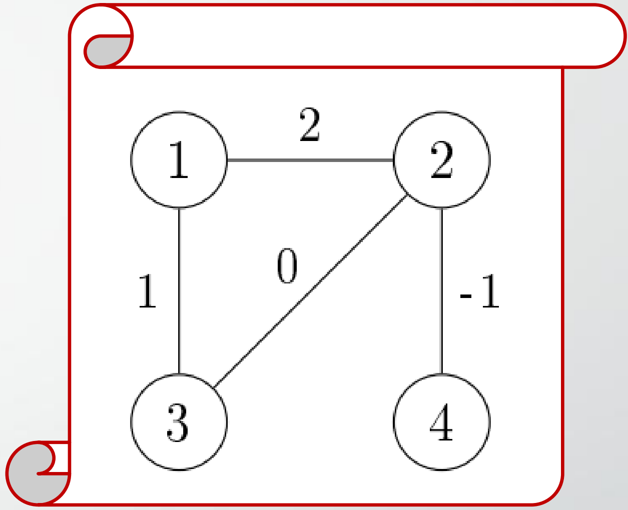
# Élsúlyozott gráfok és ábrázolásaik

1. **Definíció. Élsúlyozott gráf** alatt egy $G = (V, E)$ gráfot értünk a $w : E \longrightarrow \mathbb{R}$ súlyfüggvénnyel, ahol
   - $V$ a csúcsok (vertices) tetszőleges, véges halmaza,
   - $E \subseteq V \times V \setminus \{(u; v) : u \in V\}$ az élek (edges) halmaza.
   - A $w : E \longrightarrow \mathbb{R}$ minden egyes élhez hozzárendeli annak súlyát, más néven hosszát, illetve költségét.
     - (Az élsúly, élhossz és élköltség elnevezések szinonimák.)
2. **Definíció.** Élsúlyozott gráfban tetszőleges **út hossza** (más néven **költsége**, **súlya** ) az út mentén található élek összsúlya.
   Hasonlóképpen tetszőleges **gráf/fa súlya** az élei súlyainak összege.

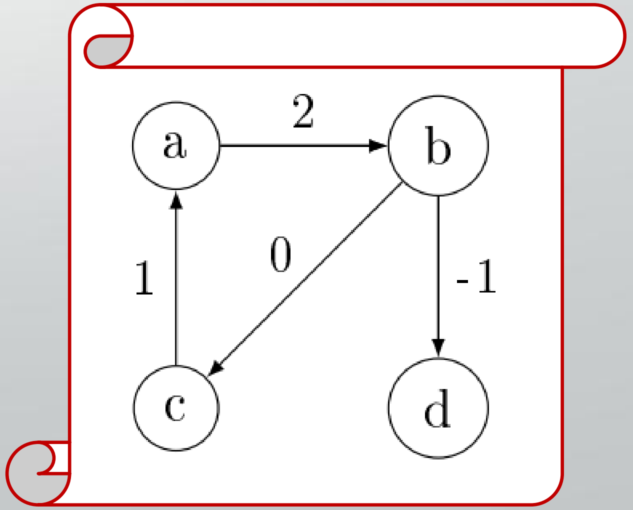
# Élsúlyozott gráfok ábrázolásai

1 – 2, 2 ; 3, 1.
2 – 3, 0 ; 4, -1.

- Grafikus ábrázolás:
  - Az éleket súlyukkal címkézzük
- Szöveges ábrázolás
  - irányítatlan gráfoknál: „$u$ - $v_{u1}$, $w_{u1}$ ; . . . $v_{uk}$ ,$w_{uk}$." jelentése:
    - ($u$, $v_{u1}$), . . .; ($u$, $v_{uk}$) élei a gráfnak
      $w(u, v_{u1})$ = $w_{u1}$ … , $w(u, v_{uk})$ = $w_{uk}$ súlyokkal.
  - irányított gráfoknál: „$u$ -> $v_{u1}$, $w_{u1}$ ; . . . $v_{uk}$ ,$w_{uk}$." jelentése:
    - a gráfban az $u$ csúcsból ($u$, $v_{u1}$), . . .; ($u$, $v_{uk}$) irányított élek indulnak ki, azaz az $u$ csúcs rákövetkezői (gyerekei): $v_{u1}$; . . . $v_{uk}$ csúcsok.



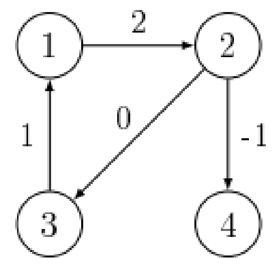

a → b, 2.
b → c, 0 ; d, -1.
c → a, 1.



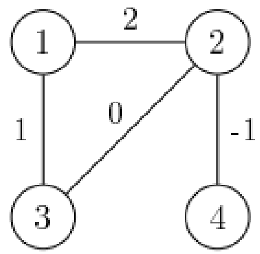
# Szomszédossági mátrixos (adjacency matrix), más néven csúcsmátrixos reprezentáció

- $G = (V, E)$ gráf $w : E \rightarrow \mathbb{R}$ súlyfüggvénnyel ($V = \{v_1, \cdots, v_n\}$) reprezentációja:
  - $A/1 : \mathbb{R}_\infty [n, n]$ mátrix, ahol
    - $n = |V|$ a csúcsok száma
    - $1..n$ a csúcsok sorszámai
    - tetszőleges $i, j \in 1..n$ csúcssorszámokra:
      - $A[i, j] = w(v_i, v_j) \Leftrightarrow (v_i, v_j) \in E$
      - $A[i, i] = 0$
      - $A[i, j] = \infty \Leftrightarrow (v_i, v_j) \notin E \wedge i \neq j$

| $A$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0 | 2 | $\infty$ | $\infty$ |
| 2 | $\infty$ | 0 | 0 | -1 |
| 3 | 1 | $\infty$ | 0 | $\infty$ |
| 4 | $\infty$ | $\infty$ | $\infty$ | 0 |




| $A$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0 | 2 | 1 | $\infty$ |
| 2 | 2 | 0 | 0 | -1 |
| 3 | 1 | 0 | 0 | $\infty$ |
| 4 | $\infty$ | -1 | $\infty$ | 0 |

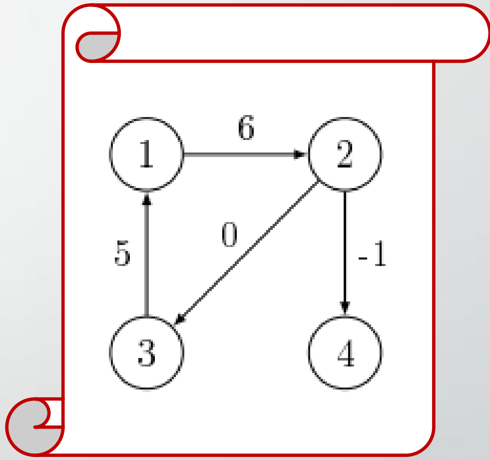


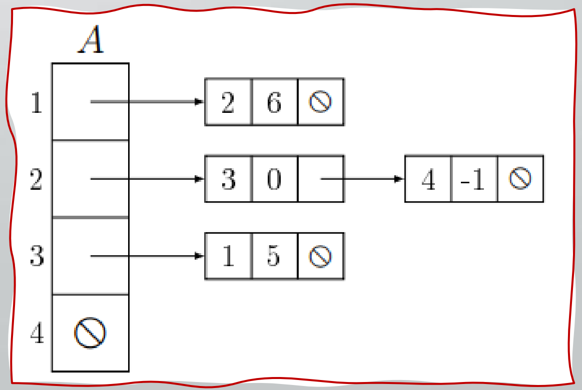

- *Megjegyzések:*
  - A főátlóban mindig nullák vannak
    - ➢ csak egyszerű gráfokkal foglalkozunk (amelyekben nincsenek hurokélek)
    - ➢ tetszőleges csúcsból önmaga közvetlenül, nulla költségű úton érhető el.
  - Irányítatlan esetben a szomszédossági mátrixos reprezentáció mindig szimmetrikus,
    - $(v_i, v_j) \in E$ esetén $(v_j, v_i) = (v_i, v_j) \in E$

# Szomszédossági listás (adjacency list) reprezentáció

- $G = (V, E)$ gráf $w : E \longrightarrow \mathbb{R}$ súlyfüggvénnyel ($V = \{v_1, \cdots, v_n\}$) reprezentációja:
  - $A/1 : \mathbb{R}_\infty[n, n]$, ahol
  - $A : Edge^*[n]$ pointertömb, ahol
    - az *Edge* típus a következő:
    - A *next* és a *v* attributumok szerepe ugyanaz, mint az élsúlyozatlan gráfoknál, *w* pedig a megfelelő él súlya.

| $Edge$ |
|---|
| $+v : \mathbb{N}$ |
| $+w : \mathbb{R}$ |
| $+next : Edge*$ |

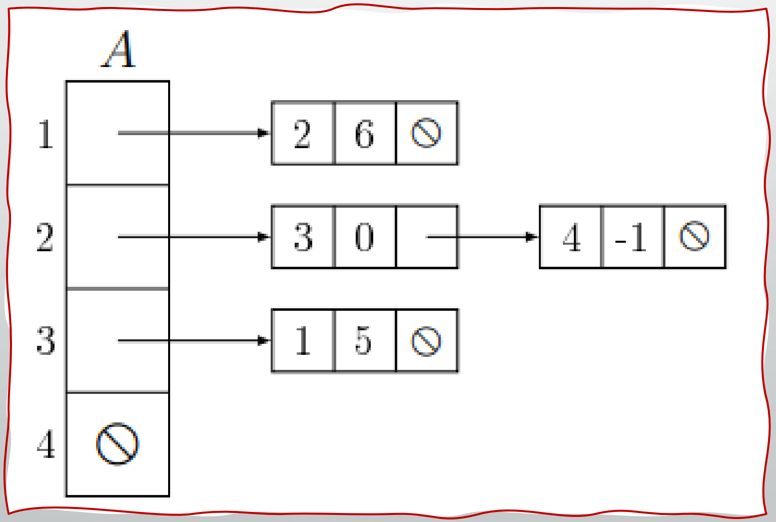


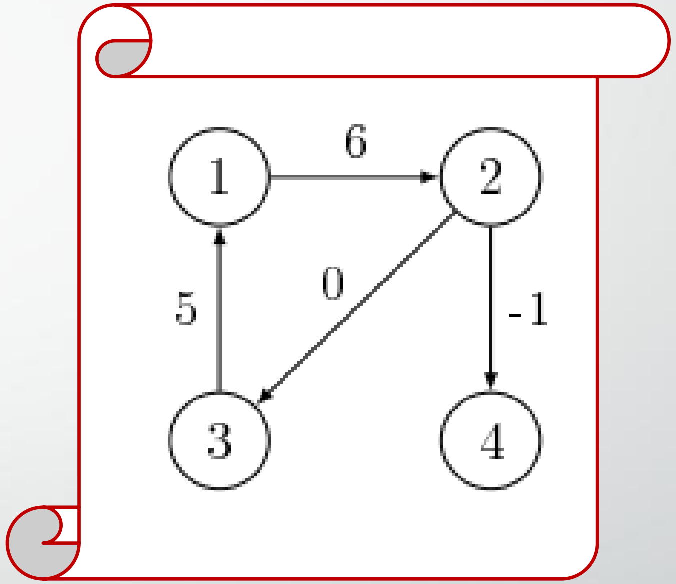




- Az élsúlyozatlan gráfokhoz hasonlóan az élsúlyozott gráfoknál is
  - Irányítatlan gráfok esetén minden élet kétszer ábrázolunk
  - Irányított gráfok esetén csak egyszer

# Szomszédossági listás reprezentáció szemléltetés

- A[1] → v = 2 ; A[1] → w = 6 ; A[1] → next = ⦸
- A[2] → v = 3 ; A[2] → w = 0 ; A[2] → next → v = 4;
- A[2] → next → w = −1 ; A[2] → next → next = ⦸
- A[3] → v = 1 ; A[3] → w = 5 ; A[3] → next = ⦸
- A[4] = ⦸



# Élsúlyozott gráfábrázolások tárigénye

- Szomszédossági mátrixok:
  - Tárigényük hasonlóan számolható, mint élsúlyozatlan esetben
  - Tfh. egy valós számot egy gépi szóban tárolunk
  - Alapesetben: $n^2$ szó
  - Irányítatlan gráfoknál:
    - Csak az alsóháromszög mátrixot tárolva
    - $n * (n-1) / 2$ szó

➢ $n * (n-1) / 2 \in \Theta(n^2)$ -> az aszimptotikus tárigény mindkét esetben: $\boldsymbol{\Theta(n^2)}$

# Élsúlyozott gráfábrázolások tárigénye

- Szomszédossági listák:
  - Mindegyik élben eggyel több mező van, mint az élsúlyozatlan esetben
  - Ez az aszimptotikus tárigényt nyilván nem befolyásolja: $\boldsymbol{\Theta(n + m)}$

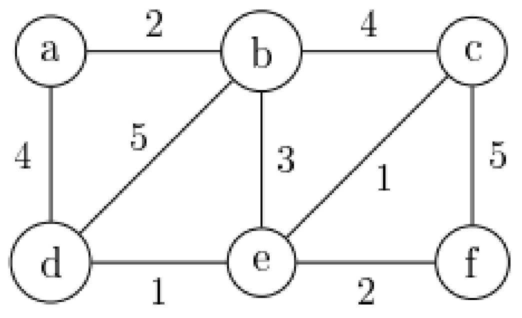
# Élsúlyozott gráfok absztrakt osztálya

| $\mathcal{G}_w$ |
|---|
| $+\ V : \mathcal{V}\{\}$<br>$+\ E : \mathcal{E}\{\}$ // $E \subseteq V \times V \setminus \{(u,u) : u \in V\}$<br>$+\ A : V \to 2^V$ // $A(u) = \{\, v \in V \mid (u,v) \in E \,\}$<br>// $A(u) =$ the adjacent vertices of vertex $u$.<br>$+\ w : E \to \mathbb{R}$ // weights of edges |

# Minimális feszítőfák
# (MST = Minimum Spanning Tree)

a – b, 2 ; d, 4.
b – c, 4 ; d, 5 ; e, 3.
c – e, 1 ; f, 5.
d – e, 1.
e – f, 2.

- Feladat minimális feszítőfa keresésére
  - A csúcsok pl. városok,
  - az élek lehetséges összeköttetések, a megépítésük költségeivel.
  - A lehető legkisebb építési költséggel szeretnénk elérni, hogy tetszőleges városból bármelyik másikba el lehessen jutni.
- Gráfok
  - összefüggő,
  - irányítatlan,
  - Élsúlyozott. (Az élsúlyok negatívak is lehetnek.)



# Minimális feszítőfák

**1.Definíció.** A $G = (V, E)$ irányítatlan gráf feszítő erdeje a $T = (V, F)$ gráf

- ha $F \subseteq E$
- valamint $T$ (irányítatlan) erdő (azaz $T$ olyan irányítatlan gráf, aminek mindegyik komponense irányítatlan fa, a fák páronként diszjunktak, és együtt éppen lefedik a G csúcshalmazát)

**2.Definíció.** A $G = (V, E)$ irányítatlan, összefüggő gráf feszítőfája a $T = (V, F)$ gráf

- ha $F \subseteq E$
- és $T$ (irányítatlan) fa.

# Minimális feszítőfák

**3. Definíció.** Amennyiben $G = (V, E)$ élsúlyozott gráf (fa, erdő stb.) a $w : E \to R$ súlyfüggvénnyel, akkor a $G$ súlya az élei súlyainak összege:

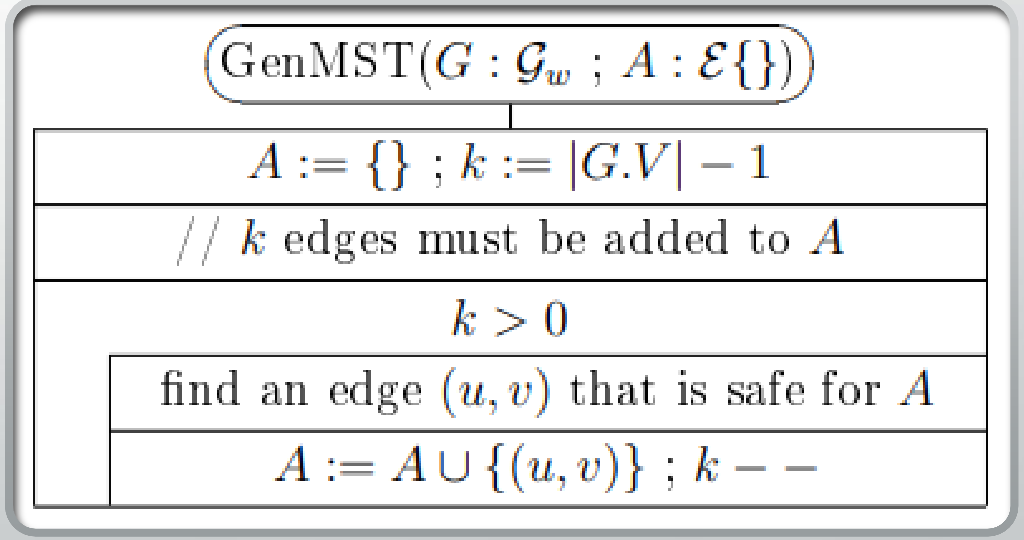
$$w(G) = \sum_{e \in E} w(e)$$

**4. Definíció.** A $G$ irányítatlan, összefüggő, élsúlyozott gráf minimális feszítőfája $T$

- ha $T$ a $G$ feszítőfája
- és $G$ bármely $T'$ feszítőfájára: $w(T) \leq w(T')$.

# Általános módszer

- Az $A$ = {} üres élhalmazból indul
- Ezt bővíti újabb és újabb élekkel
  - Úgy, hogy $A$ végig a $G$ összefüggő, irányítatlan, élsúlyozott gráf valamelyik minimális feszítőfája élhalmazának a részhalmaza marad
- Az így választott éleket nevezzük az ***A* élhalmazra nézve biztonságosnak (safe for *A*)**
- Amikor az élek száma eléri a $|G.V|-1$ értéket
  - az $A$ szükségszerűen feszítőfa
  - és így minimális feszítőfa is lesz



# Általános módszer

5. **Definíció.** Tegyük fel, hogy $G = (V, E)$ élsúlyozott, irányítatlan, összefüggő gráf, és $A \subseteq$ a $G$ valamelyik minimális feszítőfája élhalmazának!

- Ekkor az $(u, v) \in E$ él biztonságosan hozzávehető az A élhalmazhoz (safe for $A$)
  - ha $(u, v) \notin A$
  - és $A \cup \{(u, v)\} \subseteq$ a $G$ valamelyik (az előzővel nem okvetlenül egyező) minimális feszítőfája élhalmazának.

6. **Következmény.** Tegyük fel, hogy $G = (V, E)$ élsúlyozott, irányítatlan, összefüggő gráf!

- Ha egy kezdetben üres $A$ élhalmazt
- újabb és újabb biztonságosan hozzávehető éllel bővítünk,
- ➢ akkor $|G.V|-1$ bővítés után éppen a $G$ egyik minimális feszítőfáját kapjuk meg.

# Általános módszer

- Hogyan tudunk mindig biztonságos élet választani az A élhalmazhoz?
  - Ehhez lesz szükségünk az alábbi fogalmakra és tételre.

**7. Definíció.** Ha $G = (V, E)$ gráf és $\{\} \subseteq S \subseteq V$

➢ akkor a $G$ gráfon $(S, V \setminus S)$ egy **vágás**.

**8. Definíció.** $G = (V, E)$ gráfon az $(u, v) \in E$ él **keresztezi** az $(S, V \setminus S)$ **vágást**,

- ha $(u \in S \land v \in V \setminus S) \lor (u \in V \setminus S \land v \in S)$.

# Általános módszer

**8. Definíció.** $G = (V, E)$ élsúlyozott gráfon az $(u, v) \in E$ **könnyű él** az (S, V\S) vágásban,

- ha $(u, v)$ keresztezi a vágást,
- és $\forall(p, q)$ a vágást keresztező élre $w(u, v) \leq w(p, q)$.

**9. Definíció.** A $G = (V, E)$ gráfban az $A \subseteq E$ élhalmazt **elkerüli** az $(S, V \setminus S)$ **vágást**,

- ha az $A$ egyetlen éle sem keresztezi a vágást.

# Általános módszer

**11. Tétel.** Ha a $G = (V, E)$ irányítatlan, összefüggő, élsúlyozott gráfon

1. $A$ részhalmaza a $G$ valamelyik minimális feszítőfája élhalmazának,
2. Az $(S, V \setminus S)$ vágás elkerüli az $A$ élhalmazt, és
3. Az $(u, v) \in E$ könnyű él az $(S, V \setminus S)$ vágásban,

➢ az $(u, v)$ él biztonságosan hozzávehető az $A$ élhalmazhoz.

# Általános módszer

- **Bizonyítás.** $(u, v) \notin A$, ui. $(u, v)$ keresztezi az $A$-t elkerülő vágást.
  - Legyen $T = (V, T_E)$ olyan MST, amire $A \subseteq T_E$

a) $(u, v) \in T_E$

➢ készen vagyunk.

b) $(u, v) \notin T_E$ esetén:

- $T$ feszítőfa -> $T$-ben el lehet jutni $u$-ból $v$-be,
- a $T$-ben az $u$-ból a $v$-be vezető út egyértelmű.

➢ Ezen az úton van olyan él, ami keresztezi az $(S, V \setminus S)$ vágást.

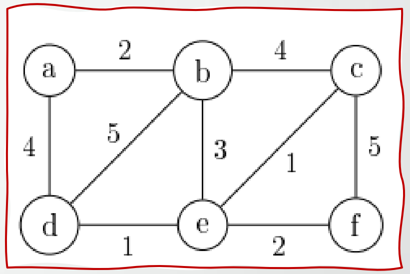
- Legyen $(p, q)$ egy ilyen él!
  - ➢ $w(p, q) \geq w(u, v) \wedge (p, q) \notin A$.

# Általános módszer

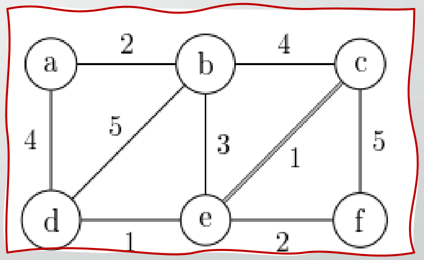
- A $(p, q)$ él törlésével
  - a $T$ MST szétesik (azaz két fából álló „feszítő erdő” lesz, az egyik fában az $u$, a másikban a $v$ csúccsal)
  - de ha az eredményhez az $(u, v)$ élet hozzávesszük
  - ➢ az így adódó $T'$ újra feszítőfa lesz
- $T' := T \setminus (p, q) \cup (u, v)$.
  - $w(T') = w(T) - w(p, q) + w(u, v) \leq w(T)$
  - $T$ MST volt -> $w(T') \geq w(T)$
  - ➢ $w(T') = w(T)$
  - ➢ $T'$ is MST kell, hogy legyen

# Általános módszer szemléltetése

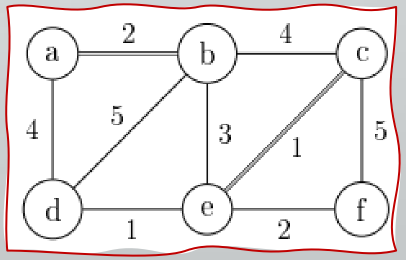
- Az előbbi tétel segítségével már módszeresen tudunk minimális feszítőfát építeni.
  - Jelölés: a gráfban dupla vonallal az *A* élhalmaz elemei
  - Minden lépésben választunk egy *A*-t elkerülő vágást
    - majd ebben egy könnyű élt, amit hozzáveszünk *A*-hoz.
  - A hozzávétel eredménye a következő lépés kiinduló pontja




1. *A*={}
   - L! pl. az ({*a*, *b*, *c*}, {*d*, *e*, *f*}) vágás
   - Egyik könnyű, tehát biztonságos él: (*c*, *e*) -> hozzávesszük *A*-hoz



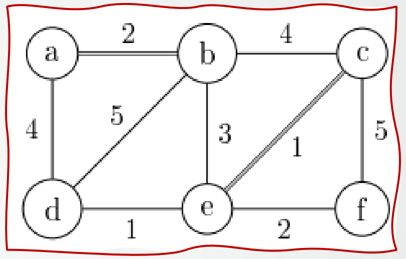

2. *A={(c, e)}*
   - L! pl. az ({*a*, *f*}, {*b*, *c*, *d*, *e*}) vágás
   - Egyik könnyű, tehát biztonságos él: (*a*, *b*) -> hozzávesszük *A*-hoz

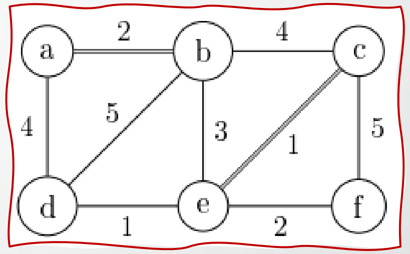


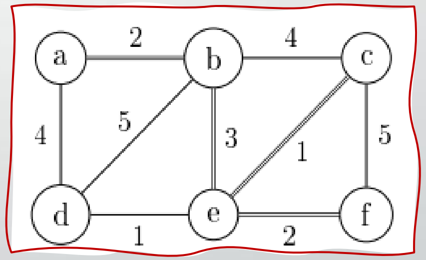
# Általános módszer szemléltetése

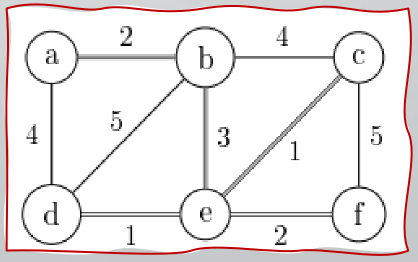
3. $A$={($a$, $b$), ($c$, $e$)}
   - L! pl. az ({$a$, $b$, $c$, $d$, $e$}, {$f$}) vágás
   - Egyik könnyű, tehát biztonságos él: ($e$, $f$) -> hozzávesszük $A$-hoz
4. $A$={($a$, $b$), ($c$, $e$), ($e$, $f$)}
   - L! pl. az ({$a$, $b$}, {$c$, $d$, $e$, $f$}) vágás
   - Egyik könnyű, tehát biztonságos él: ($b$, $e$) -> hozzávesszük $A$-hoz
5. $A$={($a$, $b$), ($b$, $e$), ($c$, $e$), ($e$, $f$)}
   - L! pl. az ({$a$, $b$, $c$, $e$, $f$}, {$d$}) vágás
   - Egyik könnyű, tehát biztonságos él: ($d$, $e$) -> hozzávesszük $A$-hoz

- $A$ = {($a$, $b$),($b$, $e$),($c$, $e$),($d$, $e$),($e$, $f$)}.
- $|A| = 5 = |G.V| - 1$ -> $A$ egy minimális feszítőfa (MST) élhalmaza



# Általános módszer

- A fenti módszer még nem tekinthető szigorú értelemben vett algoritmusnak
  - nem adtuk meg, hogyan válasszunk ki az *A* halmazt elkerülő vágást, és abban könnyű élt
- A **Kruskal** és **Prim** algoritmusok éppen ezt teszik
  - $O(m * \lg n)$ (nagyon jó) maximális műveletigénnyel
    - *n:* a gráf csúcsainak száma
    - *m:* az éleinek száma
  - Egyik sem adja meg az *A*-t elkerülő vágást expliciten, a vágásban (az egyik) könnyű élt viszont igen
  - Ilyen módon, mivel lokálisan optimalizálnak, mindkettő **mohó algoritmus**
    - A 11. tétel szerint viszont mindkét algoritmus minimális feszítőfát számol

# Kruskal algoritmusa

- $G = (V, E)$ gráf éleit a súlyuk (hosszuk) szerint monoton növekvően veszi sorba
  - Azokat az éleket eldobja, amelyek az A bizonyos éleivel együtt kört képeznének
  - A többit hozzáveszi $A$-hoz
- Az explicit körkeresés azonban nem hatékony
  - ➢ minden egyes $e$ élre bejárást kellene indítani a $(V, A \cup \{e\})$ gráfon.
  - Helyette a következő definíción és invariánson alapuló megoldáshoz folyamodunk:

# Kruskal algoritmusa

**13. Definíció.** A $G = (V, E)$ gráf **feszítő erdeje** a $(V, A)$ gráf

- ha egymástól diszjunkt fákból, mint komponensekből áll,
- és $A \subseteq E$
- (Két fa egymástól diszjunkt, ha nincs közös csúcsuk [és így közös élük sem].)

- A Kruskal algoritmus **invariánsa**
  - $(V, A)$ a $G = (V, E)$ összefüggő, irányítatlan, élsúlyozott gráf feszítő erdeje
  - és $A$ részhalmaza a $G$ valamelyik minimális feszítőfája élhalmazának

# Kruskal algoritmusa szövegesen

- $A = \{\}$-val indulunk
  - Jelentése: a kezdeti feszítő erdő fái a $G = (V, E)$ gráf egycsúcsú fái.
- A $G$ gráf éleit a súlyuk (hosszuk) szerint monoton növekvően vesszük sorba
- Tetszőleges élet pontosan akkor veszünk hozzá $A$-hoz, ha a $(V, A)$ erdő két fáját köti össze
  - Azaz nem egy fán belül fut, és így nem zár be egyetlen kört sem

➢ Minden egyes él hozzávételével eggyel csökken az erdő fáinak száma, de továbbra is feszítő erdőt alkotnak

# Kruskal algoritmusa szövegesen

- *G* összefüggő -> bármelyik két fa között van út -> előbb-utóbb összekapcsolódnak -> egy *T* fából áll az erdő, T feszítőfa
- A fenti invariáns miatt
  - a *T* élhalmaza részhalmaza valamelyik *M* minimális feszítőfa élhalmazának
  - A *G* minden feszítőfájának $|V| - 1$ éle van -> a *T* és *M* élhalmaza megegyezik
  - Mindkettő csúcshalmaza *V*
  - ➢ $T = M$, azaz *T* minimális feszítőfa.

# Invariáns igazságának belátása

- Kezdetben a (*V*, *A*) úgy feszítő erdő, hogy A = {}
  - *A* részhalmaza a *G* bármelyik minimális feszítőfája élhalmazának
  - az invariáns igaz
- Tekintsünk most az algoritmus futása során egy olyan pillanatot, amikor az invariáns még igaz, és egy újabb *e* élet készülünk megvizsgálni
  - Ha *e* a jelenlegi feszítő erdő valamelyik fáján belül fut
    - eldobjuk -> a feszítő erdő nem változik -> az invariáns igaz marad

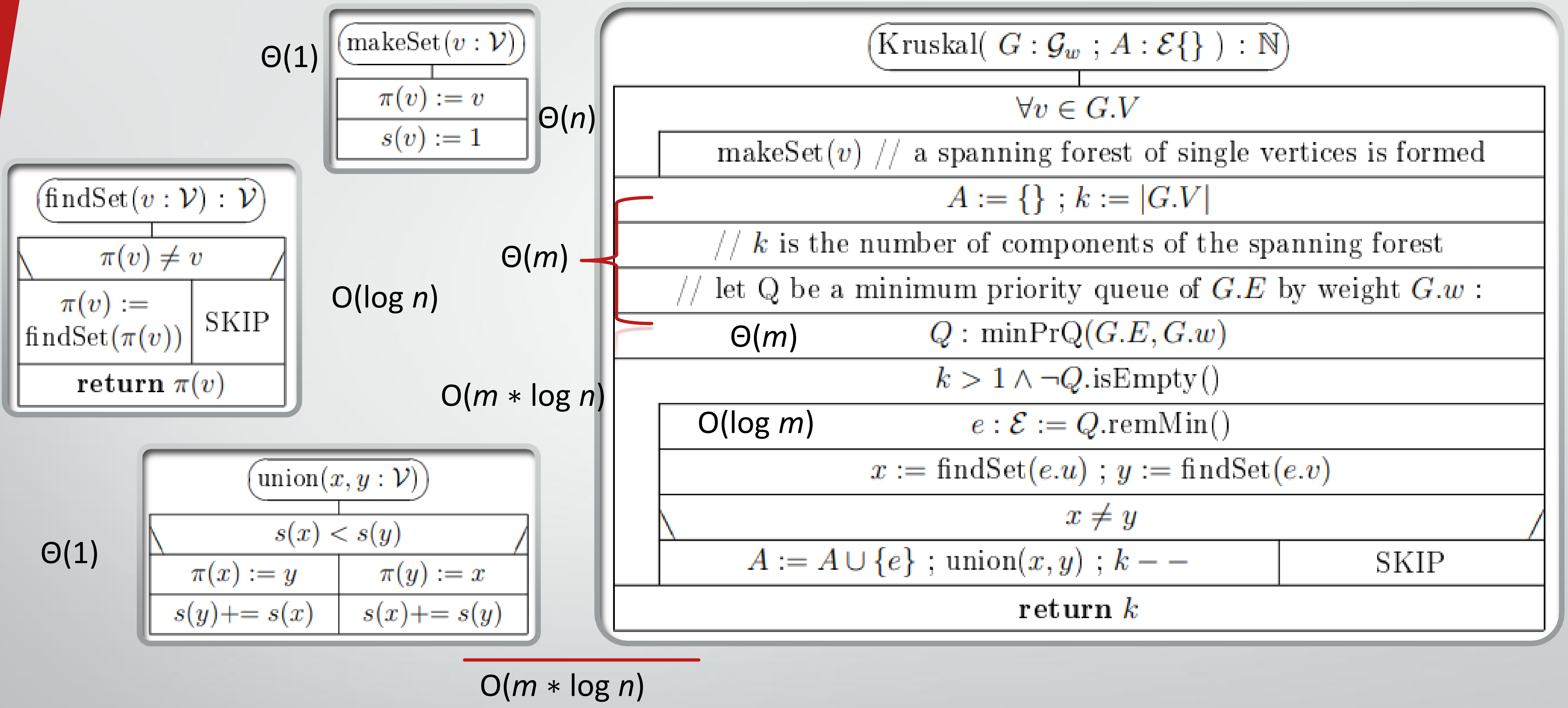
# Invariáns igazságának belátása

- Ha *e* a jelenlegi feszítő erdő két fáját köti össze
  - legyen *S* az egyik fa csúcshalmaza, és tekintsük az $(S, V\setminus S)$ vágást, ami nyilván elkerüli *A*-t
  - Ekkor *e* könnyű él a vágásban
    - [mert a G gráf éleit a súlyuk (hosszuk) szerint monoton növekvően vesszük sorba
    - az aktuális élnél kisebb súlyú éleket már korábban feldolgoztuk
    - ezek vagy benne vannak *A*-ban, vagy nincsenek benne *A*-ban, de a $(V, A)$ feszítő erdő egyik fájának két csúcsát kötik össze, ezért eldobtuk őket]
  - Teljesülnek tehát a 11. tétel feltételei -> *e* biztonságosan hozzávehető *A*-hoz, és hozzá is vesszük.
- A fentiek szerint tehát az invariáns az *e* él feldolgozása után is igaz marad

# Kruskal algoritmus műveletigénye

- Feltételezések (Biz. 29. dia)
  - makeSet($v$) $\in \Theta(1)$
  - union($x$, $y$) $\in \Theta(1)$
  - findSet(v) $\in O(\log n)$
- Tfh. $Q$ prioritásos sort bináris kupaccal valósítjuk meg
  - inicializásása $\in \Theta(m)$
    - a gráf éleit tároljuk benne
  - remMin() $\in O(\log m)$
- Az első ciklus műveletigénye: $\Theta(n)$
- A két ciklus közti részé: $\Theta(m)$

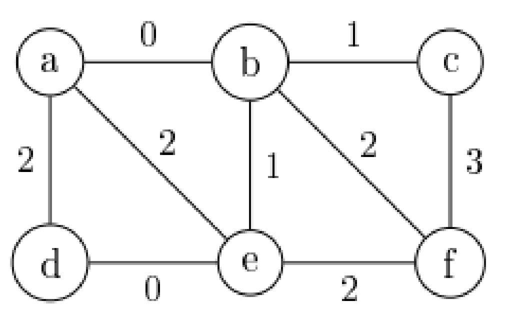
# Kruskal algoritmus stuktogramja és műveletidő




- az algoritmus ellenőrzi, hogy $G$ összefüggő-e
  - Ha összefüggő: k = 1
  - Ha nem összefüggő: k > 1

# Kruskal algoritmus szemléltetése

a – b, 0 ; d, 2 ; e, 2.
b – c, 1 ; e, 1 ; f, 2.
c – f, 3.
d – e, 0.
e – f, 2.

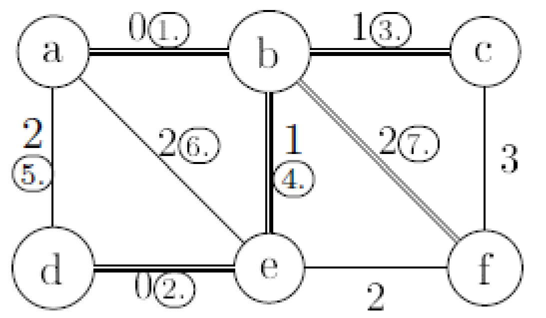



összefüggő

élsúlyozott

Irányítatlan gráf

| lépés | komponensek | él | biztonságos? |
|---|---|---|---|
| 1. | a, b, c, d, e, f | $a \overset{0}{-} b$ | + |
| 2. | ab, c, d, e, f | $d \overset{0}{-} e$ | + |
| 3. | ab, c, de, f | $b \overset{1}{-} c$ | + |
| 4. | abc, de, f | $b \overset{1}{-} e$ | + |
| 5. | abcde, f | $a \overset{2}{-} d$ | – |
| 6. | abcde, f | $a \overset{2}{-} e$ | – |
| 7. | abcde, f | $b \overset{2}{-} f$ | + |
| - | abcdef | - | - |

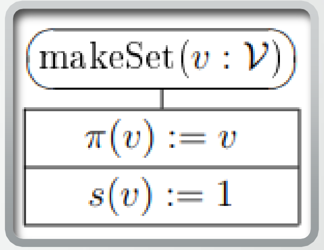


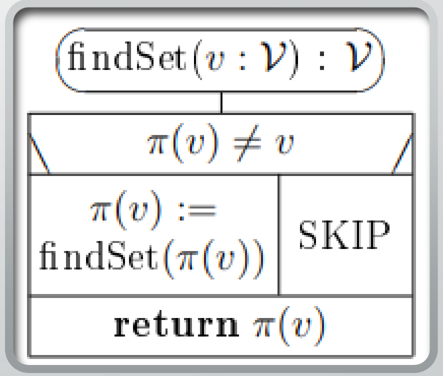
# Kruskal algoritmus műveletigénye

- *Fő ciklus*
  - *Egy iterációja* ∈ O O(log $n$)
    - $e$ := $Q$.remMin() ∈ O(log $m$) = O(log $n$)
    - G összefüggő és irányítatlan
    - így $n - 1 \leq m \leq n * (n - 1)/2 < n^2$
    - (log $n$) – 1 < log($n$ – 1) ≤ log $m$ < log $n^2$ = 2 ∗ log $n$
    - log $m$ ∈ Θ(log $n$) -> Θ(log $m$) = Θ(log $n$) -> O(log $m$) = O(log $n$)
  - legfeljebb m-szer iterál
  - a teljes műveletigénye O($m$ ∗ log $n$)
- Ezt aszimptotikus értelemben már nem módosítja a fő ciklust megelőző inicializálások nála nagyságrenddel kisebb Θ($n$ + $m$) futási ideje.

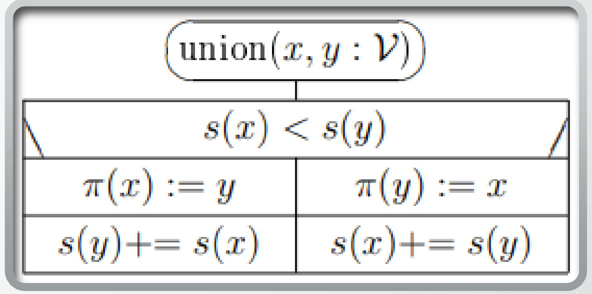
# A Kruskal algoritmus halmazműveletei

- makeSet(*v*)
  - a gráf mindegyik *v* csúcsából egyelemű irányított fát képez
- findSet(*v*)
  - megállapítja, hogy a csúcs melyik irányítatlan fában van
    - azaz megkeresi a csúcsot tartalmazó irányított fa gyökércsúcsát
    - [Közben a *v* csúcs és mindegyik őse *π* mutatóját a gyökércsúcsra állítja]
      - ➢ a későbbi findSet-hívások már majd hatékonyabbak legyenek
      - út-rövidítés tényleges hatékonyságnövekedést hoz magával
        - Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Új Algoritmusok, Scolar Kiadó, Budapest, 2003. ISBN 963 9193 90 9 könyvben részletes elemzés



# A Kruskal algoritmus halmazműveletei




- union($x$, $y$)
  - a megfelelő irányított fák gyökércsúcsait köti össze
    - $x$ := findSet($e.u$) ; $y$ := findSet($e.v$) utasítások az $e$ él két végpontjához tartozó gyökércsúcsokat határozzák meg
  - a kisebb méretű irányított fa gyökerét a nagyobb (vagy vele egyenlő méretű) irányított fa gyökere alá köti be
  - Belátjuk, hogy ezzel a módszerrel sosem lesz az egyes fák magassága > log (*fa mérete*)
    - Ebből pedig közvetlenül következik majd a findSet($v$) függvény műveletigényével kapcsolatos állításunk

# Unió-hol van adatszerkezet

- Az irányítatlan feszítőerdő nyilvántartásához egy másik, irányított erdőt is kezelünk
- Irányított erdő
  - Az irányítatlan feszítőerdő minden egyes (irányítatlan) fájának megfelel az irányított erdő egy (irányított) fája
    - Ugyanaz a csúcshalmaza
  - Az irányított fák a gyökerük felé irányítottak
  - Mindegyik csúcsnak van egy $\pi$ címkéje
    - értéke az ő szülője
    - kivéve az irányított fák gyökércsúcsait
      - tetszőleges $r$ gyökércsúcsra viszont $\pi(r) = r$
  - $s(r)$ : $r$-hez tartozó fa mérete

➢ Így mindegyik irányítatlan fát a neki megfelelő irányított fa gyökércsúcsával azonosítjuk.

# Unió-hol van asz. fáinak magassága

**17. Tulajdonság.** A makeSet($v$) és a union($x$, $y$) eljárások hatására létrejövő, gyökércsúcsuk felé irányított fák magassága sosem lesz nagyobb, mint a méretük kettes alapú logaritmusa, azaz, ha $r$ egy ilyen fa gyökércsúcsa, $s(r)$ a mérete és $h(r)$ a magassága, akkor $\log s(r) \geq h(r)$.

- **Bizonyítás.**
  - makeSet($v$) : egy csúcsú, nulla magasságú fákat hoz létre
    - $\log 1 = 0$
    - ➢ a bizonyítandó tulajdonság kezdetben igaz
  - Elegendő belátni, hogy a union($x$, $y$) eljárás is tartja a fenti egyenlőtlenséget
    - Legyen $r$ a union($x$, $y$) eljárás hatására létrejövő fa gyökere
    - Feltehető, hogy az eljáráshívás előtt $\log s(x) \geq h(x) \wedge \log s(y) \geq h(y)$
    - Azt kell belátnunk, hogy a union($x$, $y$) eljáráshívás után $\log s(r) \geq h(r)$

# Unió-hol van asz. fáinak magassága

- Feltehető még, hogy $s(x) \geq s(y)$
  - Ekkor az $x$ gyökércsúcs alá csatoljuk be az $y$ gyökércsúcsot, így $h(r) = \max(h(x), h(y) + 1)$
    - $h(x) > h(y)$
      - $h(r) = h(x) \land s(r) \geq s(x)$
      - $\log s(r) \geq \log s(x) \geq h(x) = h(r)$
      - $\log s(r) \geq h(r)$
    - $h(x) \leq h(y)$
      - $h(r) = h(y) + 1 \land s(r) = s(x) + s(y) \geq 2 * s(y)$
      - $\log s(r) \geq \log(2 * s(y)) = 1 + \log s(y) \geq 1 + h(y) = h(r)$
      - $\log s(r) \geq h(r)$

# Halmazműveletek műveletigénye

- Kruskal algoritmus műveletigény-számításával kapcsolatos feltételezések jogosságának belátása
  - Jelölések
    - *h*:= a *v* csúcsot tartalmazó irányított fa magassága
    - *s*:= a *v* csúcsot tartalmazó irányított fa mérete
  - findSet(v) ∈ O(log *n*)
    - findSet(v) fv műveletigénye O(*h*)
    - a 17. tulajdonság alapján $h \leq \log s$
    - $s \leq n$

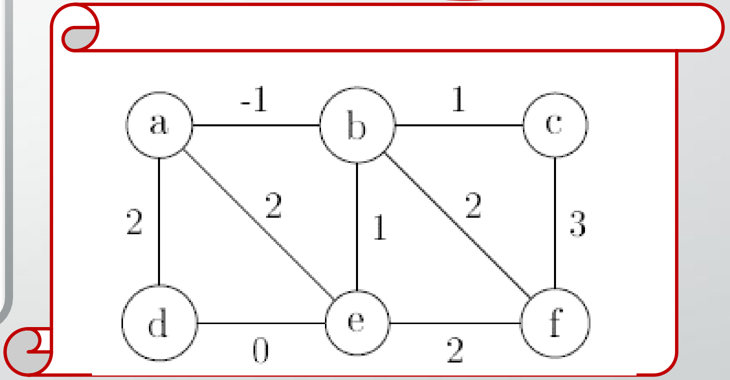
findSet(*v*) fv. futási ideje durva felső becsléssel O(log *n*)

  - makeSet(*v*) ∈ Θ(1) ∧ union(*x*, *y*) ∈ Θ(1)
    - Sem ciklust, sem eljáráshívást nem tartalmaznak

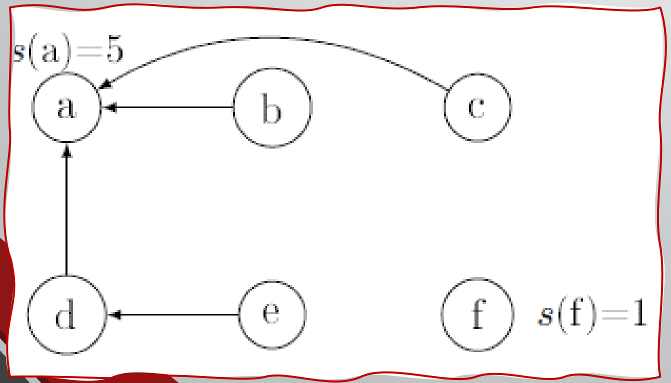
# A Kruskal algoritmus táblázatos szemléltetése a halmazműveletekkel együtt*

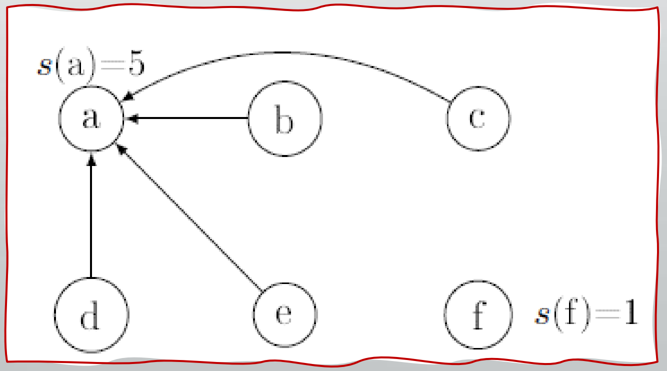
| komponensek | a | b | c | d | e | f | $u \overset{w}{\text{—}} v$ | $u\pi^*s$ | $v\pi^*s$ | $\pm$ |
|---|---|---|---|---|---|---|---|---|---|---|
| a, b, c, d, e, f | a1 | b1 | c1 | d1 | e1 | f1 | $a \overset{-1}{\text{—}} b$ | a1 | b1 | + |
| ab, c, d, e, f | b | b2 | | | | | $d \overset{0}{\text{—}} e$ | d1 | e1 | + |
| ab, c, de, f | | | | e | e2 | | $b \overset{1}{\text{—}} c$ | b2 | c1 | + |
| abc, de, f | | b3 | b | | | | $b \overset{1}{\text{—}} e$ | b3 | e2 | + |
| abcde, f | | b5 | | | b | | $a \overset{2}{\text{—}} d$ | ab5 | deb5 | − |
| abcde, f | | | | b | | | $a \overset{2}{\text{—}} e$ | ab5 | eb5 | − |
| abcde, f | | | | | | | $b \overset{2}{\text{—}} f$ | b5 | f1 | + |
| abcdef | | b6 | | | | b | - | | | |

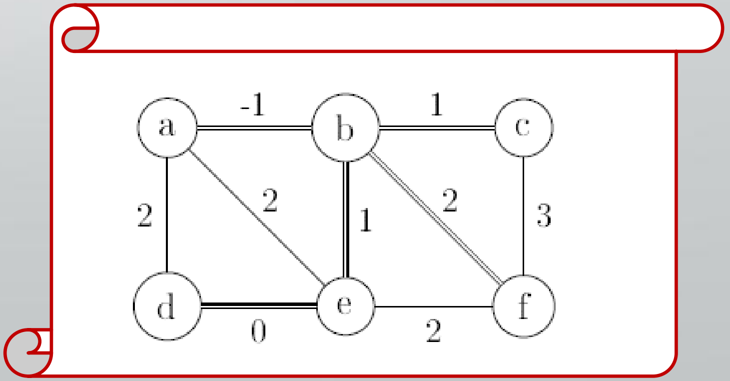
a – b, -1 ; d, 2 ; e, 2.
b – c, 1 ; e, 1 ; f, 2.
c – f, 3.
d – e, 0.
e – f, 2.




Útrövidítéssel:



# Ellenőrző kérdések

1. Mit számol ki a *Kruskal* algoritmus?
2. Szemléltesse a működését az alábbi gráfon!
    - a - b, 0 ; d, 1. b - c, 5 ; d, 2 ; e, 3. d - e, 4.
    - Mekkora az algoritmusnak az előadásról ismert műveletigénye, és milyen feltételekkel?
3. Mondja ki a *biztonságos élekről* és a *minimális feszítőfákról* szóló tételt!
    - Definiálja a tételben szereplő *vágás* és könnyű él fogalmakat!
    - Hogyan következik a *Kruskal* algoritmus helyessége ebből a tételből?

# Köszönöm a figyelmet!

**Pusztai Kinga**

A bemutató Ásványi Tibor: Algoritmusok és adatszerkezetek II. eladásjegyzet:Élsúlyozott gráfok és algoritmusaik alapján készült.