Gráfalgoritmusok

Gaskó Noémi

2023. március 16.

Tartalomjegyzék

Múlt órán

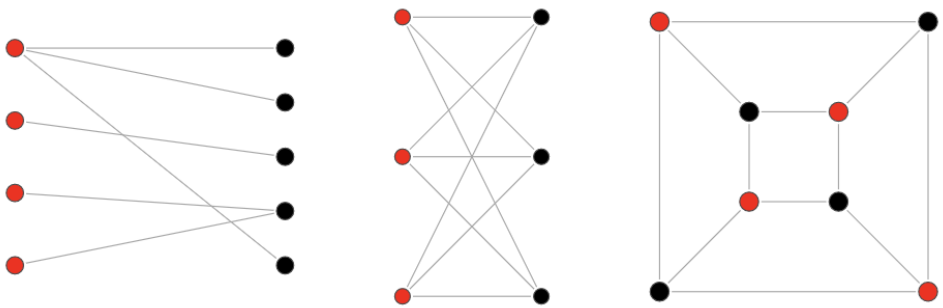
- mélységi bejárás
- szélességi bejárás
- alkalmazások

Páros gráf

eldönteni egy gráfról, hogy páros-e

Páros gráf

Egy gráf páros gráf, ha csomópontjainak a halmaza felosztható, két diszjunkt halmazra (A és B), úgy hogy az élek végpontjai különböző halmazokba kerüljenek.



Megoldás: lásd 3_jegyyet.pdf

Körök ellenőrzése

Hogyan ellenőrizzük irányított gráfokban, hogy van-e kör?

Megoldás: lásd 3_jegyzet.pdf

Elvágó pontok és hidak

Elvágó pont

Legyen $G = (V, E)$ egy nem irányított gráf és $u \in V$ egy tetszőleges csomópont a gráfban. Az u csomópont **elvágó pont** a G gráfban, ha létezik legalább két csomópont $x, y \in V, x \neq y, x \neq u,$ és $y \neq u$, úgy hogy bármely $x \rightsquigarrow y$ út átmegy u-n.

vagy

Elvágó pont

Egy irányítatlan gráfban elvágó pontnak vagy artikulációs pontnak nevezzük azokat a csúcsokat, melyeket ha eltávolítjuk a gráfból a hozzá illeszkedő élekkel együtt, növekszik az összefüggő komponensek száma.

Híd

Legyen $G = (V, E)$ egy nem irányított gráf, $(u, v) \in E$ egy él a gráfból. Az (u, v) egy **híd** a G gráfban ha létezik legalább két pont $x, y \in V, x \neq y$, úgy hogy bármely $x \rightsquigarrow y$ út a G-ben tartalmazza (u, v)-t.

vagy

Híd

Egy nem irányított gráfban hídnak nevezzük azokat az éleket, melyek eltávolításával növekszik az összefüggő komponensek száma.

Kétszeresen összefüggő komponensek

Kétszeresen összefüggő komponens

Legyen $G = (V, E)$ egy nem irányított gráf. Egy **kétszeresen összefüggő komponense** a G-nek egy maximális részgráf $G_b = (V_b, E_b)$, $V_b \subseteq V$ és $E_b \subseteq E$ amely nem tartalmaz elvágó pontot.

vagy

Kétszeresen összefüggő komponens

Legyen $G = (V, E)$ egy nem irányított gráf. Egy **kétszeresen összefüggő komponense** a G-nek egy maximális részgráf $G_b = (V_b, E_b)$, $V_b \subseteq V$ és $E_b \subseteq E$ úgyhogy bármely él esetén α és $\beta \in E_b$ létezik egy lánc mely tartalmazza az α és β éleket.

- elvágó pontok naiv megkeresése: O(n(n+m)) időben
- hidak megkeresése O(m(n+m)) időben
- Tarjan BICONNECT algoritmusa: egyetlen mélységi bejárással

Tarjan BICONNECT algoritmusa

- hasonlóan mint a Tarjan STRONGCONNECT algoritmusa
- tároljuk a belépési időt minden csúcsra (belep[v])
- a vermen éleket tárolunk
- minden v csomópont esetén tároljuk a low[v] értéket, a legkorábbi csúcs belépési ideje, amelyet v-ből elérhetünk faéleken vagy max 1 visszamutató élen

Tarjan BICONNECT algoritmusa (folyt.)

Híd

Egy (u,v) él akkor híd, ha (u,v) faél és $low[v] \geq belep[v]$

Elvágó pont

Egy v csomópont elvágó pont:

- Ha v nem a bejárási fa gyökere, és létezik olyan w gyereke a fában, melyre $low[w] \geq belep[v]$
- Ha v a bejárási fa gyökere és van legalább két gyereke

Tarjan BICONNECT algoritmusa[1] (folyt.)

```
BIC(v, parent)
time++
belep[v]=time
low[v]=time
gyerek=0
elvagocsucs=false;
FOR minden w szomszédjára v-nek
   IF (belep[w]=-1)
      verem.push(v,w)
      gyerek++
      BIC(w,v)
      low[v]=min(low[v],low[w])
      IF (low[w]≥belep[w])
         hidak.add(v,w)
```

[1] forrás: Patcas Csaba kurzus

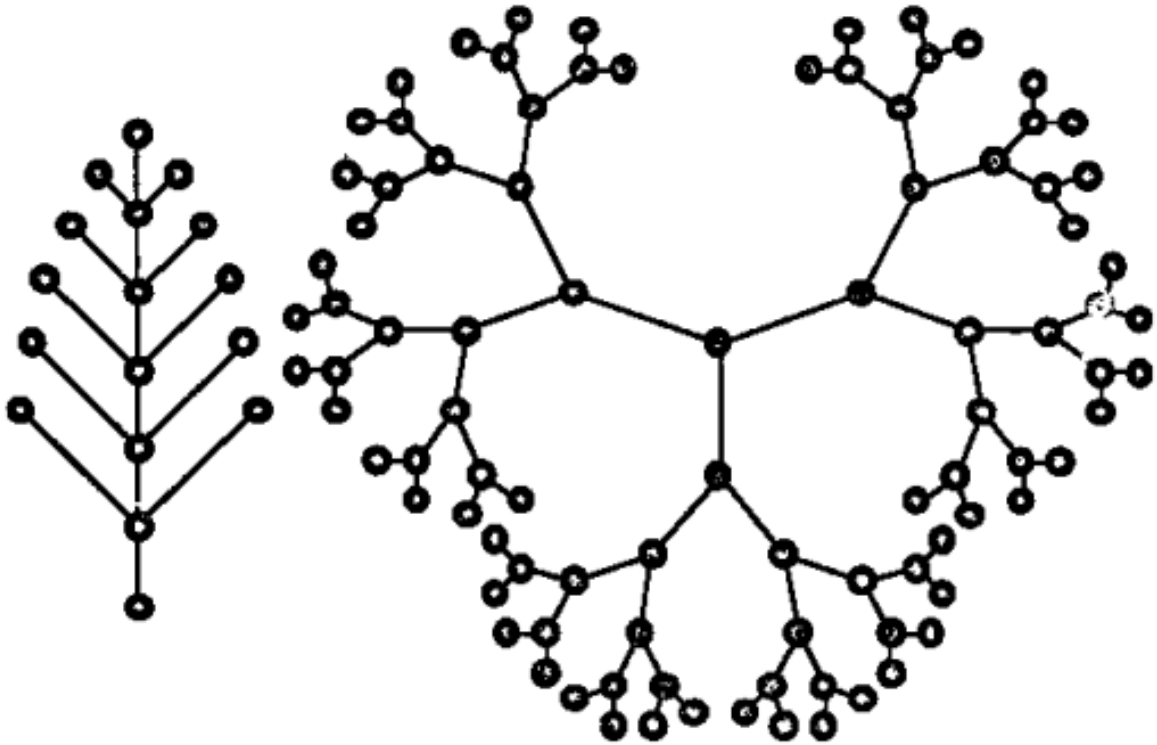
Tarjan BICONNECT algoritmusa (folyt.)

```
    IF (low[w]≥belep[v])
      elvagoCsucs=TRUE
      komponensek++
      WHILE (verem.empty==false) AND
(komponensek[komponensek].last≠ {v, w})
        komponensek[komponensek].add(verem.top())
        verem.pop()
  ELSE IF (w≠parent) AND (belep[w]<belep[v])
    verem.push(v,w)
    low[v]=min(low[v],belep[w])
IF ((parent≠-1) AND elvagoCsucs) OR ((parent=-1) AND (gyerek>1))
  elvagocsucsok.add(v)
```

Egy példa

lásd 3_jegyzet.pdf

Fák



Fák értelmezése

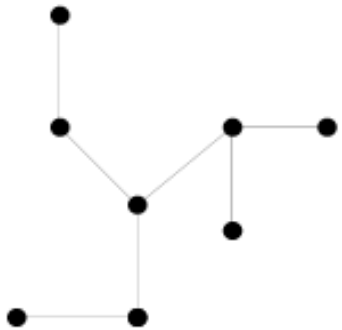
Legyen G egy n-csúcsú gráf. A következõ állítások egyenértékûek és a fákat jellemzik:

- G összefüggõ és körmentes
- G körmentes és $n-1$ éle van
- G összefüggõ és $n-1$ éle van
- G körmentes, de bármely két nem szomszédos csúcsának összekötésével kör keletkezik
- G összefüggõ, de bármely élének törlésével szétesik két komponensre

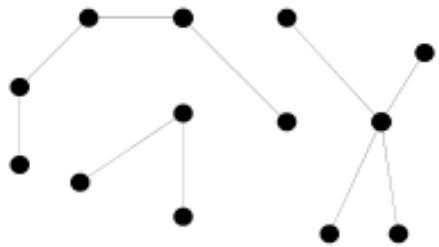
Bizonyítás: 2. szeminárium

Értelmezések

Egy körmentes gráfot ligetnek (erdõnek) nevezünk. Egy összefüggõ körmentes gráfot fának nevezünk. A liget több fából állhat. Az elsõfokú csúcsokat levélnek nevezzük.



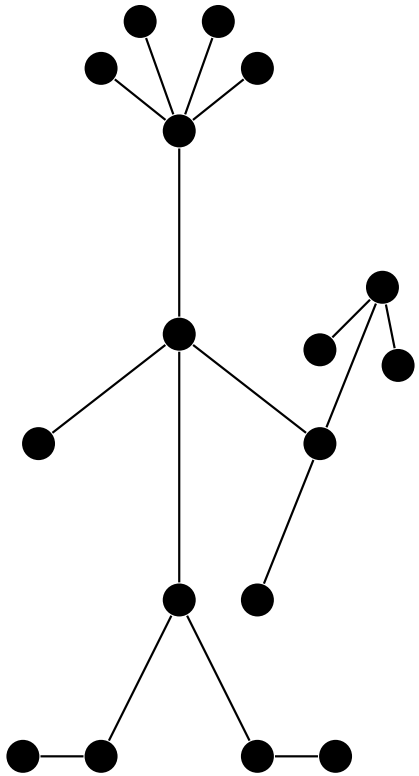
fa



liget

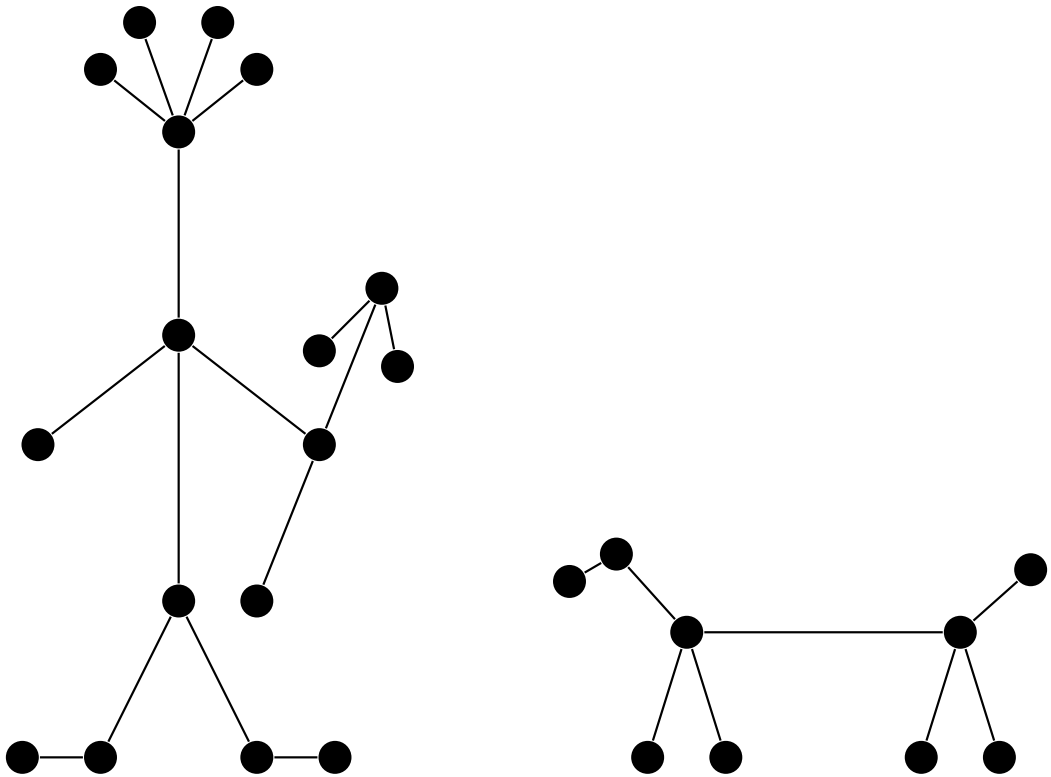
Fák és ligetek

- fa



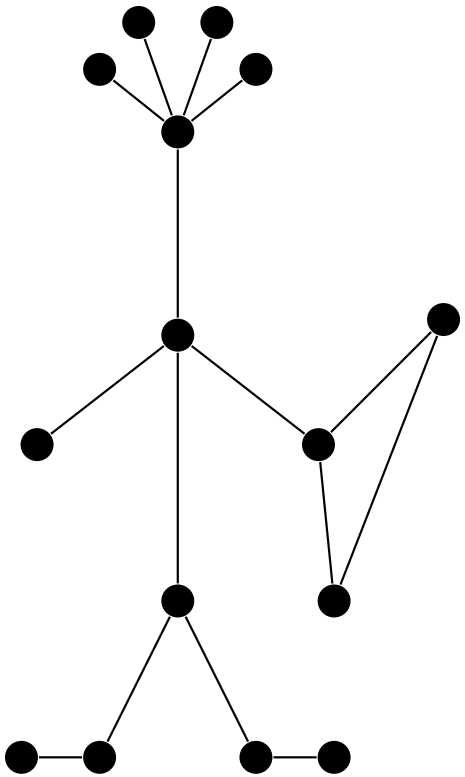
Fák és ligetek (II)

- liget



Fák és ligetek

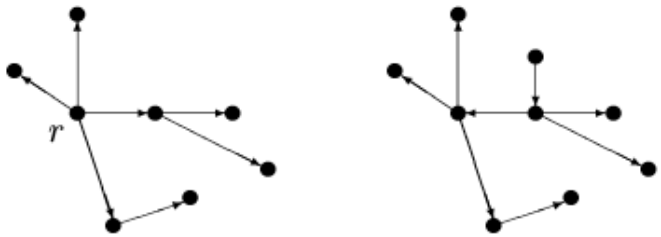
- ????????



Gyökeres fák

Gyökeres fa

A gyökeres fa olyan irányított élû fa, amelyben kijelölünk egy gyökérnek nevezett r csúcsot, azzal a tulajdonsággal, hogy bármely v csúcsára igaz legyen, hogy létezik r-v irányított út.



Bináris fák

Bináris fa

A bináris fa olyan sajátos gyökeres fa, amelynek élei nem irányítottak, ennek ellenére úgy tekinthetjük, mintha azok a gyökértõl a levelek felé lennének irányítva.

Bináris fák

Bináris fa

A bináris fa olyan sajátos gyökeres fa, amelynek élei nem irányítottak, ennek ellenére úgy tekinthetjük, mintha azok a gyökértõl a levelek felé lennének irányítva.

Értelmezés

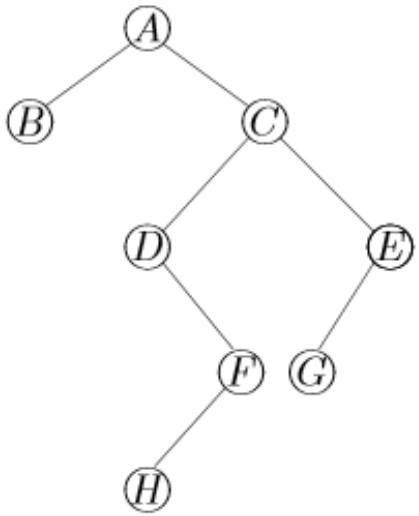
- Egy csúcs bináris fa és gyökér a neve
- Ha az A és B, a és b gyökerû bináris fák, akkor bináris fák a következõk is, amelyekben A bal oldali részfa, míg B jobb oldali részfa:
  – egy r gyökerû fa, amelyben r egy-egy éllel kapcsolódik a-hoz és b-hez,
  – egy r gyökerû fa, amelyben r egy éllel kapcsolódik a-hoz,
  – egy r gyökerû fa, amelyben r egy éllel kapcsolódik b-hez.

- A bináris fákat mindig úgy rajzoljuk le, hogy felül van a gyökére, alatta a többi csúcs.
- Amint az értelmezésbõl is látszik, a bináris fáknál megkülönböztetjük a bal és a jobb oldali részfákat. Ha ezeket felcseréljük, akkor más bináris fát kapunk, annak ellenére, hogy ezek mint gráfok, izomorfak.

Bináris fák bejárása

Bejárásuk:

- preorder bejárás: gyökér, bal, jobb
- inorder bejárás: bal, gyökér, jobb
- posztorder bejárás: bal, jobb, gyökér



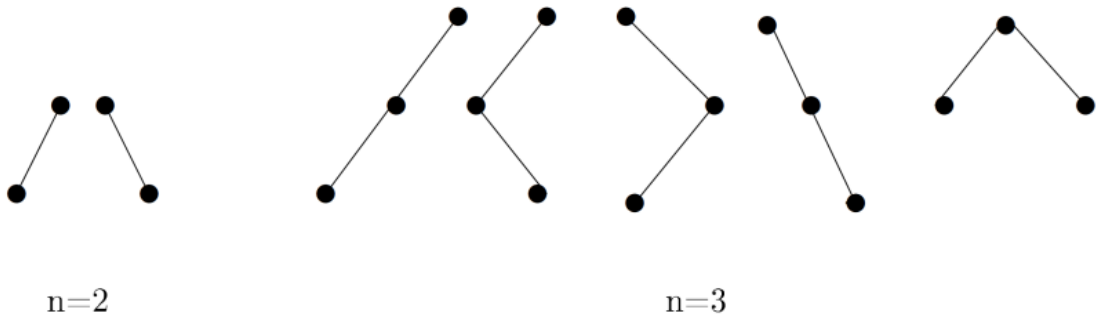
preorder: A, B, C, D, F, H, E, G

inorder: B, A, D, H, F, C, G, E

postorder: B, H, F, D, G, E, C, A

Bináris fák száma

Jelöljük b_n-el az n csomópontú bináris fák számát, $b_1 = 1, b_2 = 2, b_3 = 5$.



Ha $b_0 = 1$, rekurzivan megadva a bináris fák számát:
$b_n = \sum_{k=0}^{n-1} b_k \cdot b_{n-1-k}$

Bináris fák száma

Tétel

Az n csomópontú bináris fák száma $b_n = \frac{1}{n+1} C_{2n}^n = \frac{(2n)!}{n!(n+1)!}$. Ezeket az értékeket Catalan számoknak is nevezzük és C_n-el jelöljük.

Tétel

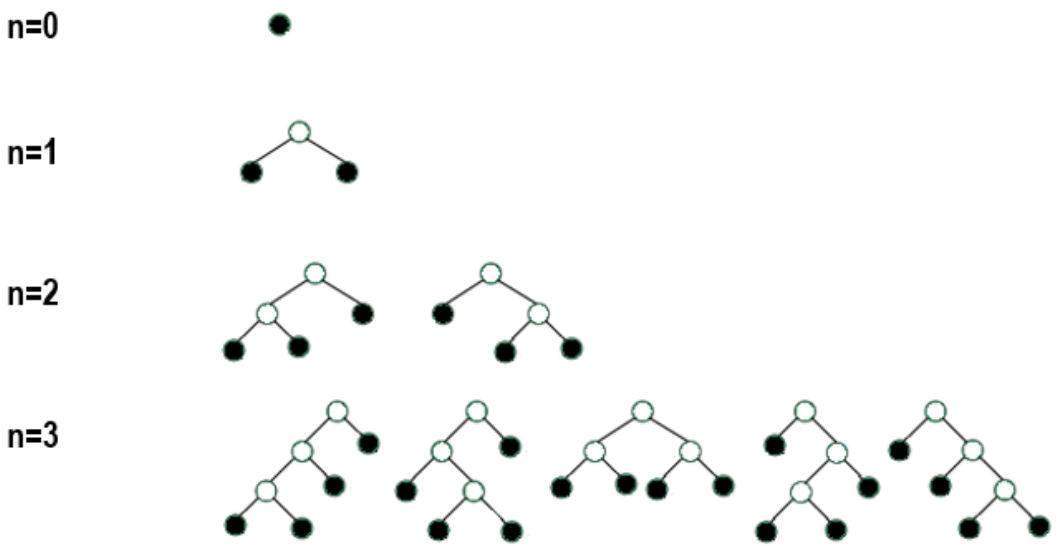
Az n csúcsú és k levelű bináris fák száma: $b_n^k = \frac{1}{n} C_{2k}^k C_n^{2k-1} \dot{2}^{n-2k}$

Catalan számok - Más feladatok

- a zárójelezés probléma - hányféleképpen lehet n pár zárójelet elrendezni (úgy hogy a zárójelezés jó legyen)

n=0	*	1
n=1	( ).	1
n=2	(( )), ( ) ( ).	2
n=3	((( ))), (( )) ( ), (( ) ( )), ( ) (( )), ( ) ( ) ( ).	5

- n+1 csomópontból hány gyökeres bináris fa építhető fel

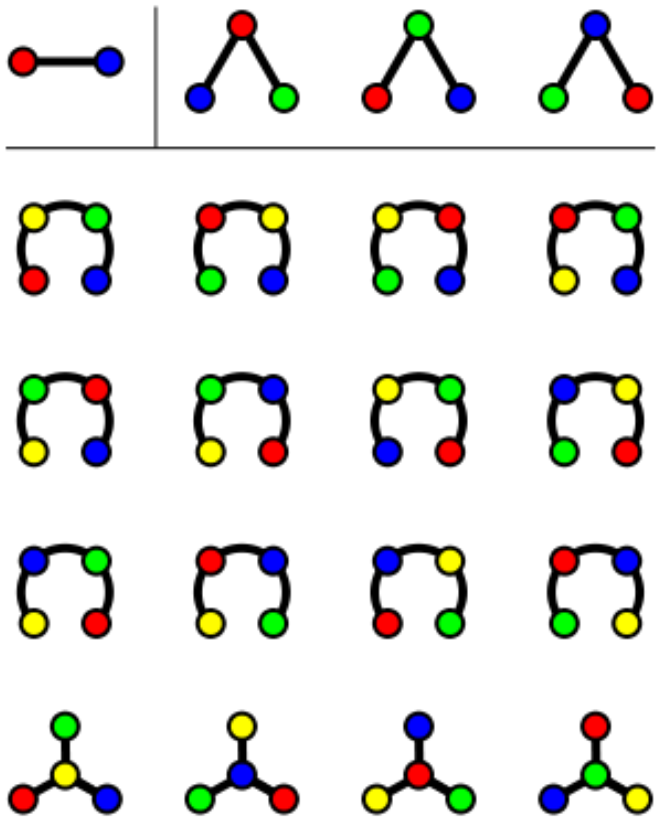


Feszítőfák száma

Cayley formula

Egy címkézett K_n teljes gráfban a feszítőfák száma n^{n-2}.

Bízonyítása Prüfer kóddal.

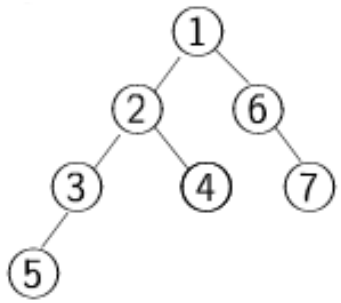


Prüfer kódolás

PrüferKódolás(F)

1. legyen K üres sorozat
2. **while** F nemcsak gyökérbõl áll **do**
3. legyen v a legkisebb címkéjû levél F-ben
4. írjuk be K-ba v õsét
5. töröljük v-t F-bõl
6. **return** K

Egy példa:



2,3,2,1,6,1

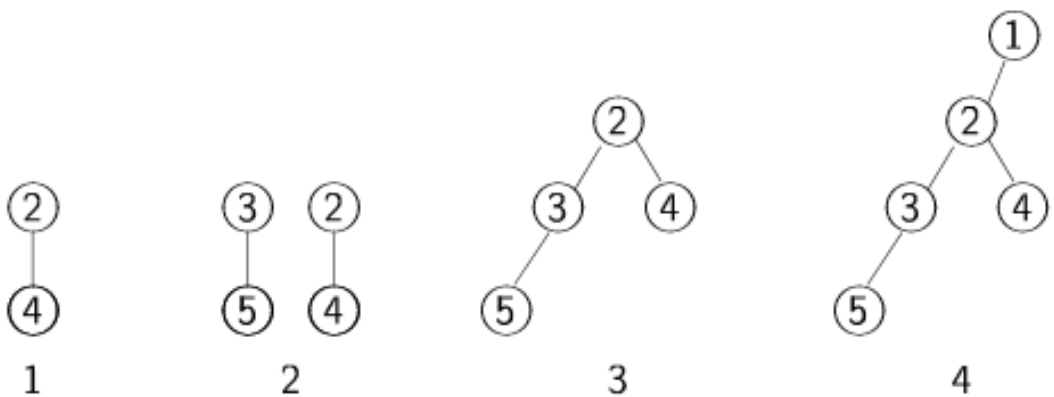
Prüfer dekódolás

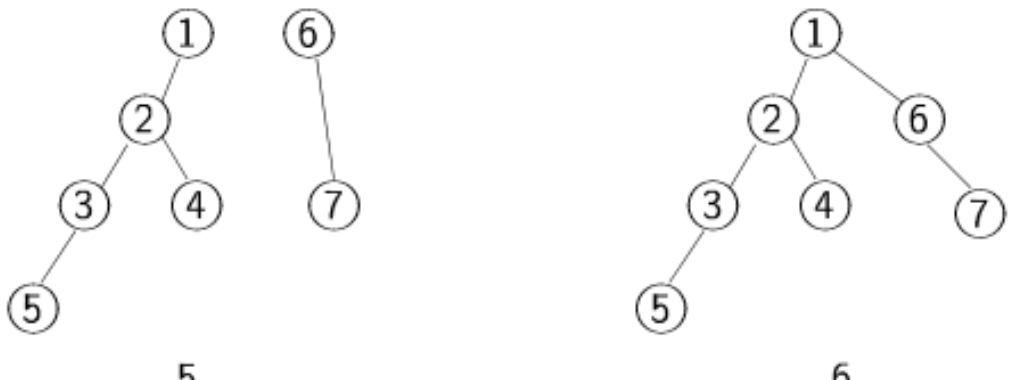
PrüferDekódolás(K, n)

1. legyen F egy üres gráf
2. **for** $i = 1, 2, \ldots, n-1$ **do**
3. legyen x a K sorozat elsõ eleme
4. legyen y a legkisebb természetes szám, amely nincs benne K-ban
5. rajzoljunk egy (x, y) élt F-be
6. töröljük x-et a K elejérõl, és adjuk hozzá a végére y-t
7. **return** F

Prüfer dekódolás

$2, 3, 2, 1, 6, 1 \parallel 4$
$3, 2, 1, 6, 1, 4 \parallel 5$
$2, 1, 6, 1, 4, 5 \parallel 3$
$1, 6, 1, 4, 5, 3 \parallel 2$
$6, 1, 4, 5, 3, 2 \parallel 7$
$1, 4, 5, 3, 2, 7 \parallel 6$
$4, 5, 3, 2, 7, 6$





Huffmann algoritmusa

Tekintsünk egy gyökeres fát, amelynek $v_1, v_2, \ldots, v_k$ levelei rendre a $w_1, w_2, \ldots, w_k$ súlyokkal rendelkeznek. Ha a gyökértől egy v_j levélig az út hosszát l_j-vel jelöljük, akkor értelmezzük a $\sum_{j=1}^{k} w_j l_j$ értéket, amelynek neve *súlyozott úthossz*.

Feladatunk, hogy adott véges számsorozathoz mint levelekhez rendelt súlyokhoz, keressünk minimális súlyozott úthosszú bináris fát.

Huffman algoritmusa

- Válasszuk ki a sorozatból a két legkisebbet, legyenek ezek w_i és w_j, töröljük ki õket a sorozatból, majd adjuk hozzá a sorozathoz a $w_i + w_j$ számot, aztán pedig adjuk hozzá a keresendõ fához a következõ részfát.
- Folytassuk az eljárást mindaddig, amíg a sorozat egyetlen számmá zsugorodik.
- Az így kapott bináris fa a keresett minimális súlyozott úthosszú bináris fa.

Példa.

Huffman kód - tömörítés

Huffman algoritmusát használhatjuk optimális kódok generálására.
Egy példa: kódoljuk a köv. szöveget:
alma a fa alatt
lásd 3. szeminárium

Feladatok

Hófehérkének egy szobát építettek a törpök (az építkezésről majd később :)), hogyan rendezzék el a vezetékeket, hogy a költség minimális legyen?

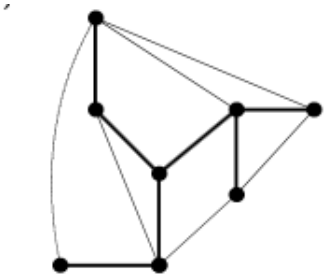
Feladatok

Hófehérkének egy szobát építettek a törpök (az építkezésről majd később :)), hogyan rendezzék el a vezetékeket, hogy a költség minimális legyen?

Más feladat: városokat szeretnénk autópályákkal összekötni, hogyan oldható meg, hogy a költség minimális legyen?

Feszítõfa

A G gráf feszítõfája vagy faváza a G olyan részgráfja, amely fa, és tartalmazza a G gráf minden csúcsát.



Ha egy gráf T részgráfja a következõ tulajdonságok közül bármely hárommal rendelkezik, akkor T feszítõfa:

- T összefüggõ
- T körmentes
- T-nek n csúcsa van
- T-nek n-1 éle van

Megjegyzés. A második és negyedik tulajdonság önmagában is elegendõ

Gazdaságos feszítõfák

Minimális feszítőfa

Egy irányítatlan súlyozott gráfban a gráf legkisebb költség? feszítőfáját nevezzük minimális feszítőfának (a költség az éleihez rendelt súlyok összege).

- Kruskal algoritmusa

Gazdaságos feszítõfák

Minimális feszítőfa

Egy irányítatlan súlyozott gráfban a gráf legkisebb költség? feszítőfáját nevezzük minimális feszítőfának (a költség az éleihez rendelt súlyok összege).

- Kruskal algoritmusa
- Prim algoritmusa

Gazdaságos feszítõfák

Minimális feszítőfa

Egy irányítatlan súlyozott gráfban a gráf legkisebb költség? feszítőfáját nevezzük minimális feszítőfának (a költség az éleihez rendelt súlyok összege).

- Kruskal algoritmusa
- Prim algoritmusa
- Boruvka algoritmusa

Gazdaságos feszítõfák

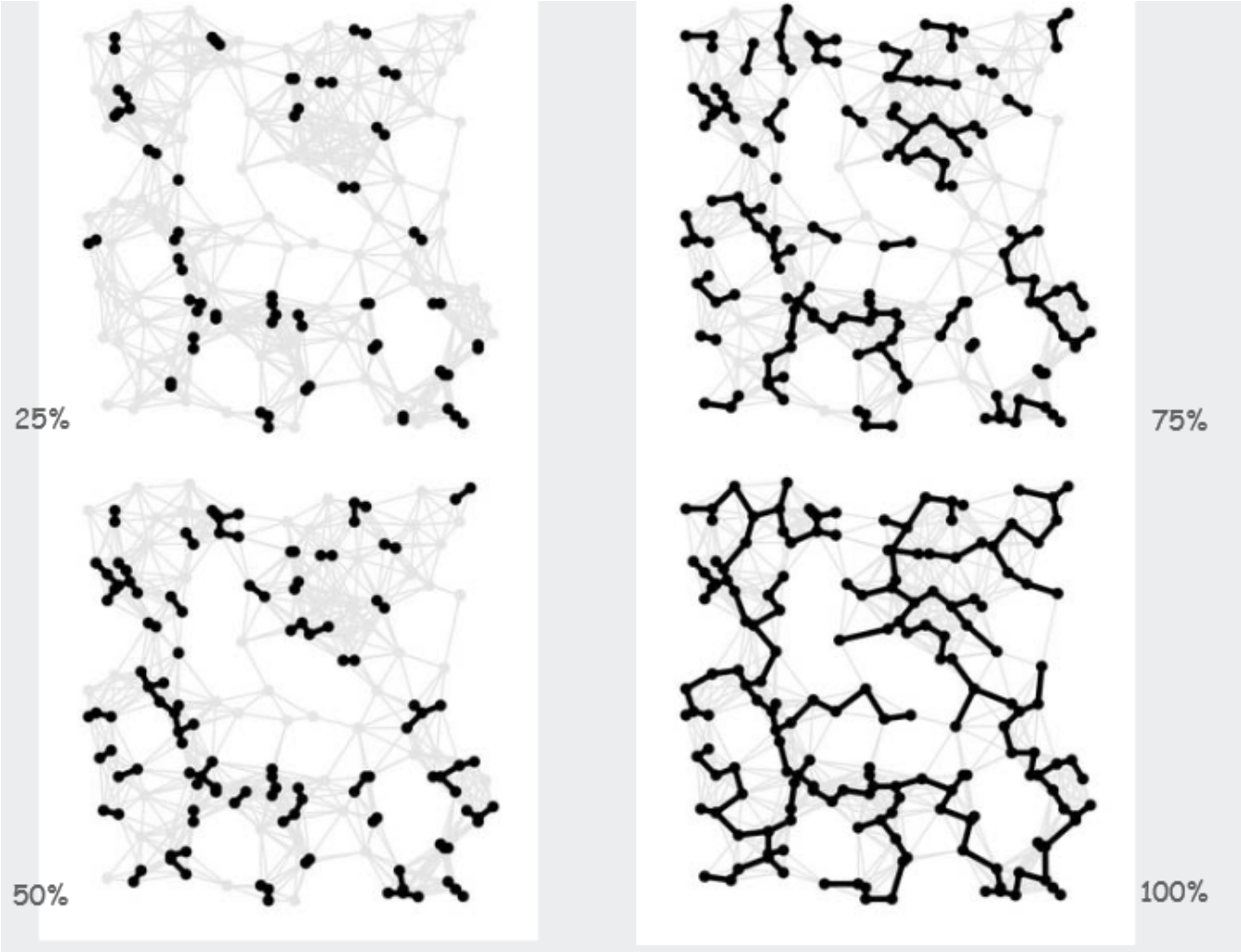
Minimális feszítőfa

Egy irányítatlan súlyozott gráfban a gráf legkisebb költség? feszítőfáját nevezzük minimális feszítőfának (a költség az éleihez rendelt súlyok összege).

- Kruskal algoritmusa
- Prim algoritmusa
- Boruvka algoritmusa
- fordított törlés algoritmusa

Kruskal algoritmusa

Kruskal, 1957



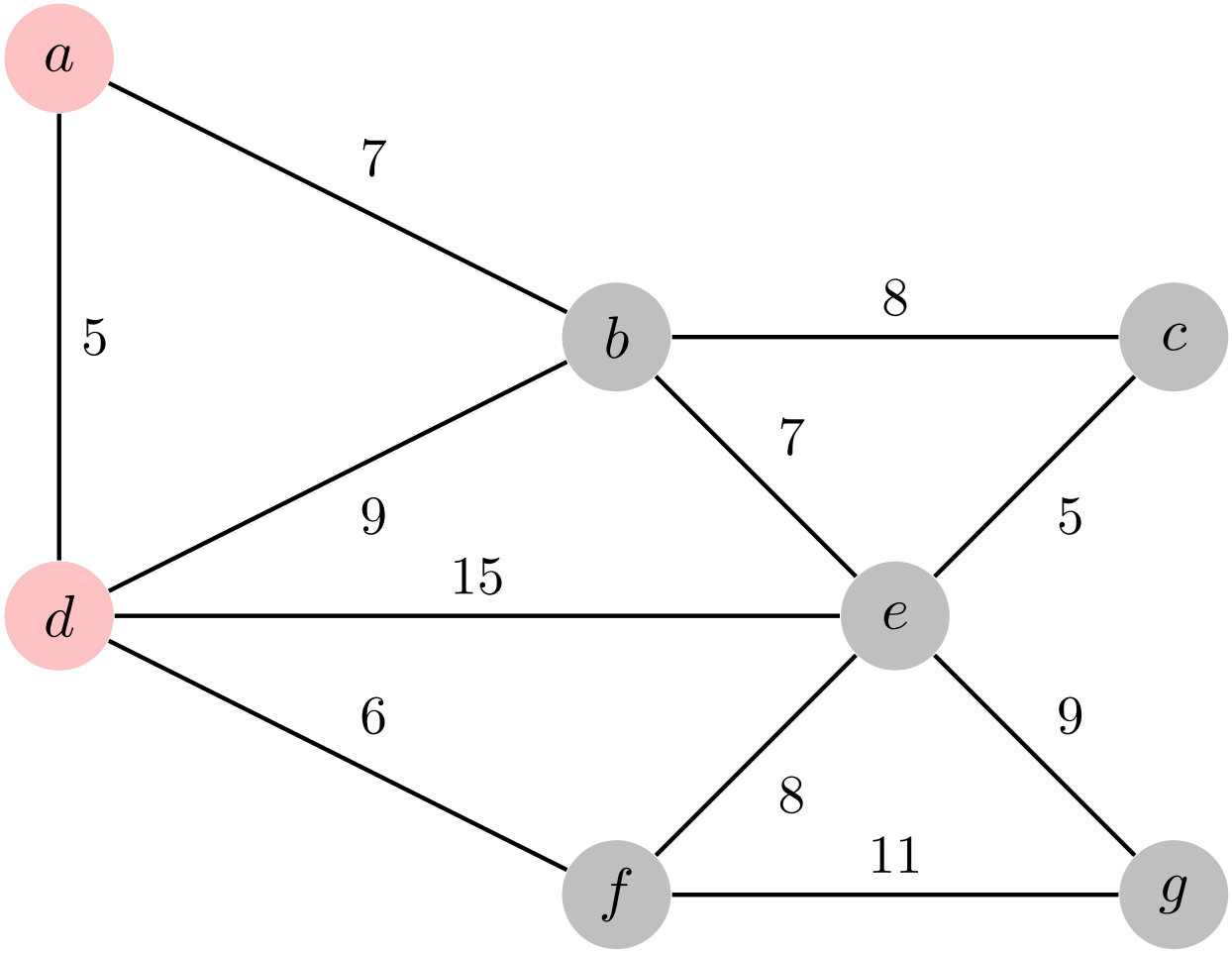
Kruskal algoritmusa

Az algoritmus:

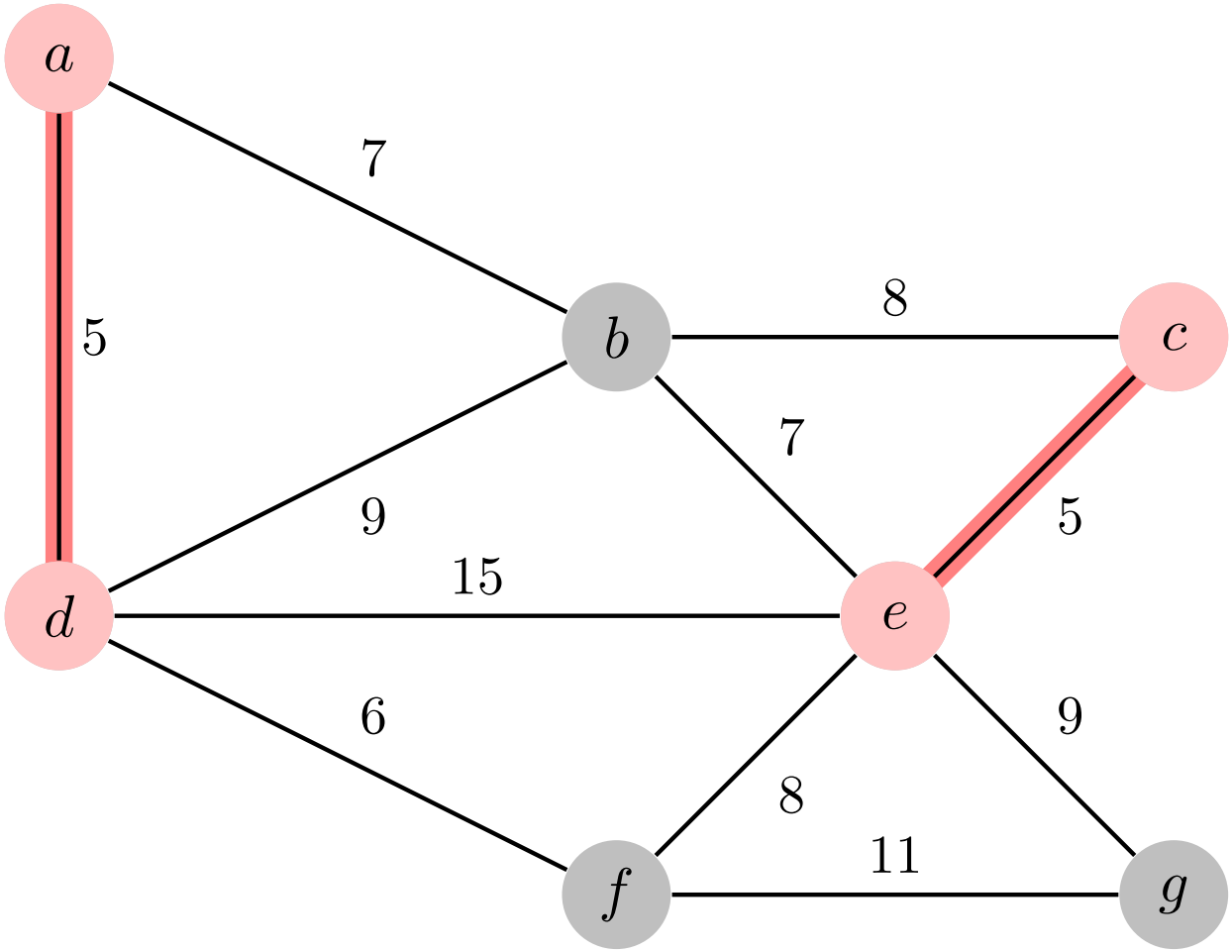
KRUSKAL(E)
  for j=1,2, . . . , n **do**
    $h_j := j$
  $i := 1$
  while h elemei különbözőek **do**
    if (e_i végpontjai v_k, v_l) és ($h_k \neq h_l$) **then**
      kiír e_i
      for j:=1, 2, . . . , n **do**
        if $h_j = h_l$ **then**
          $h_j := h_k$
    i:=i+1

Bonyolultság: legjobb esetben $O(mlogm)$, diszjunkt halmaz adatszerkezetet használva, lineáris rendezéssel

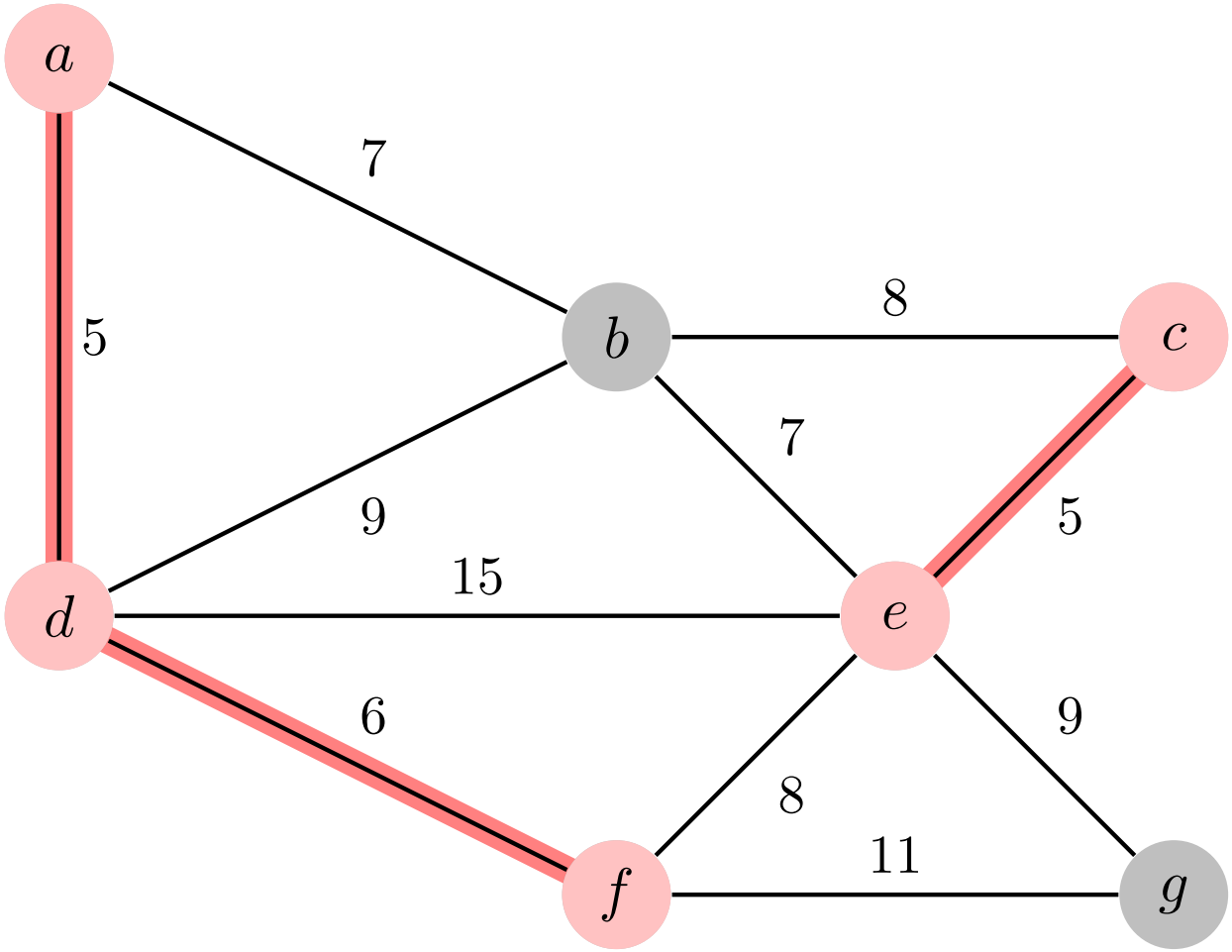
Kruskal - Példa



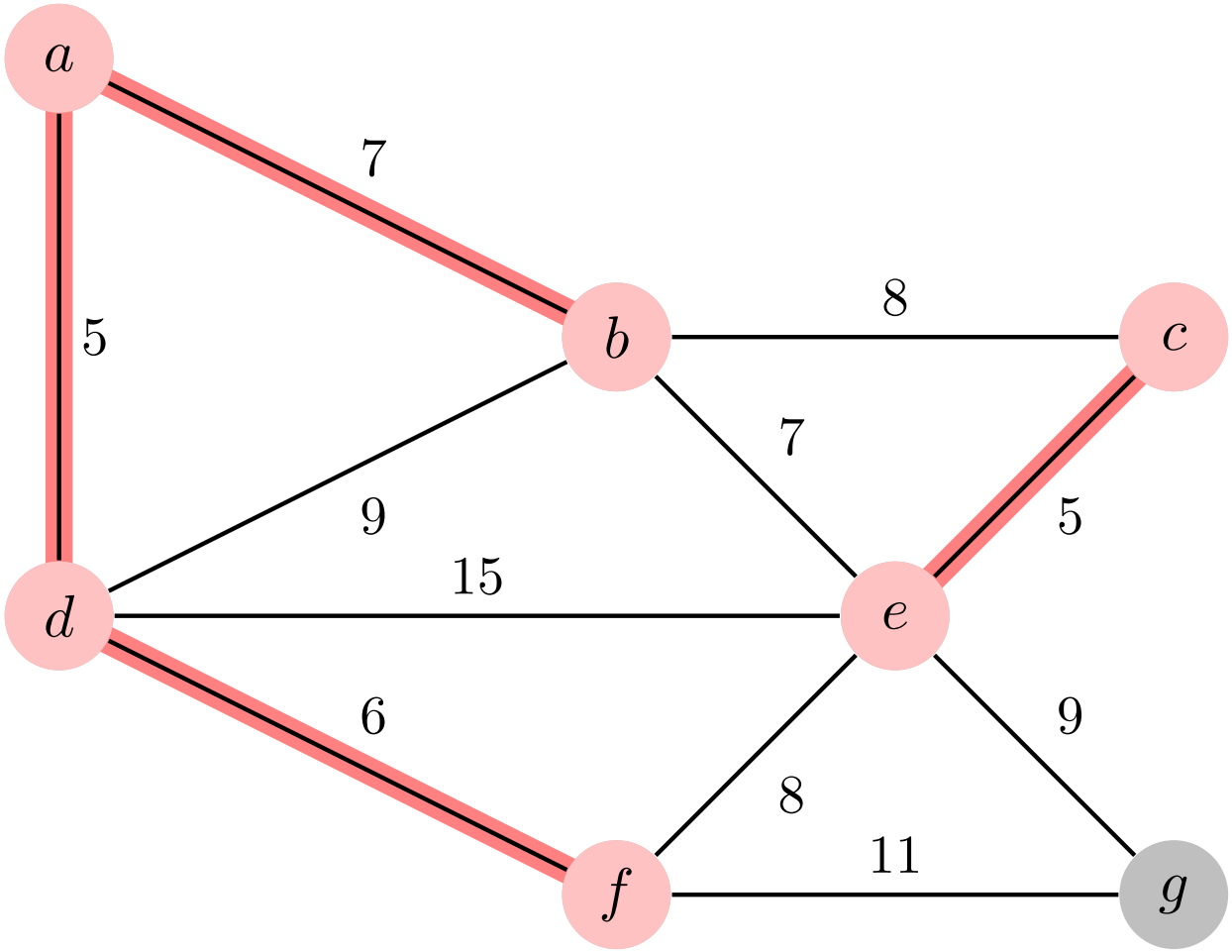
Kruskal - Példa



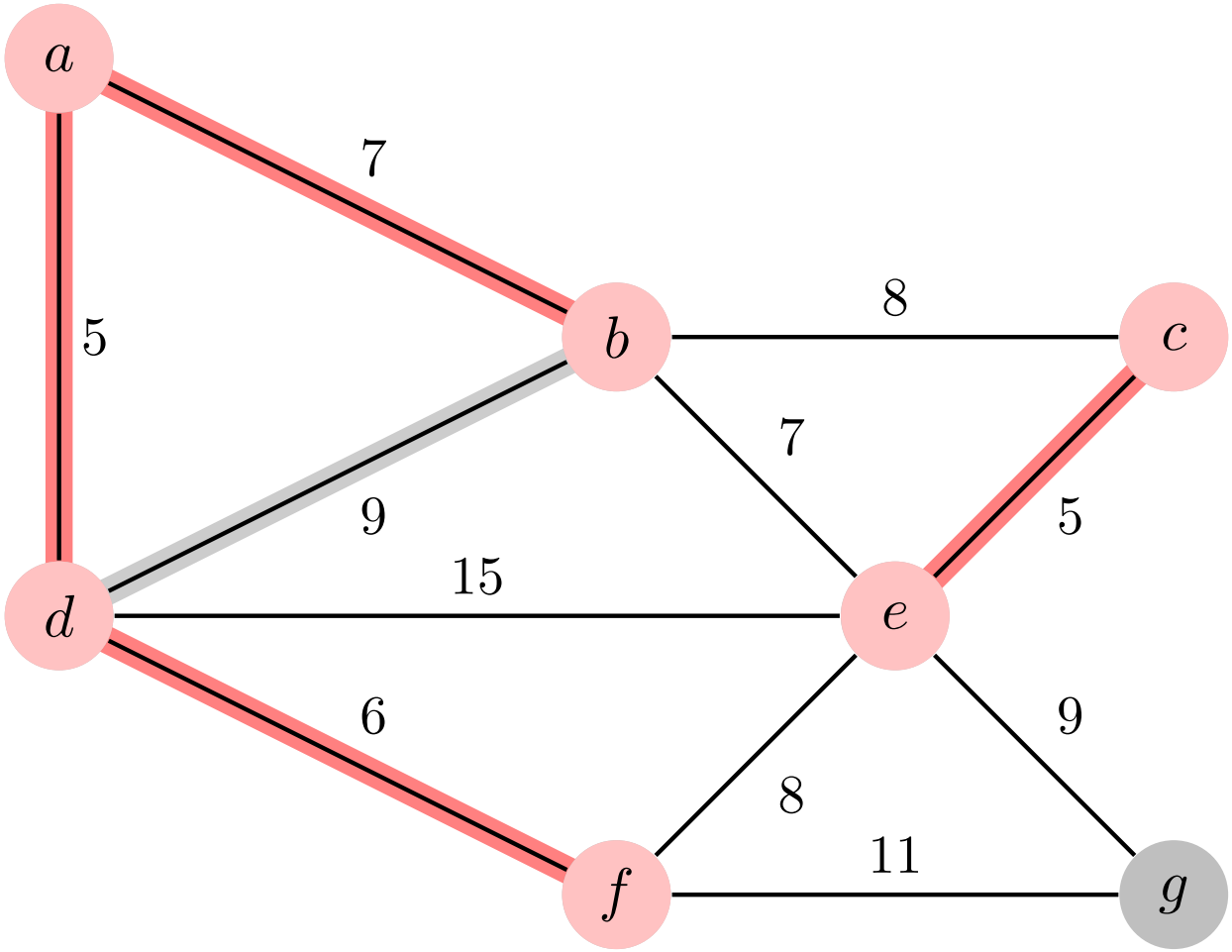
Kruskal - Példa



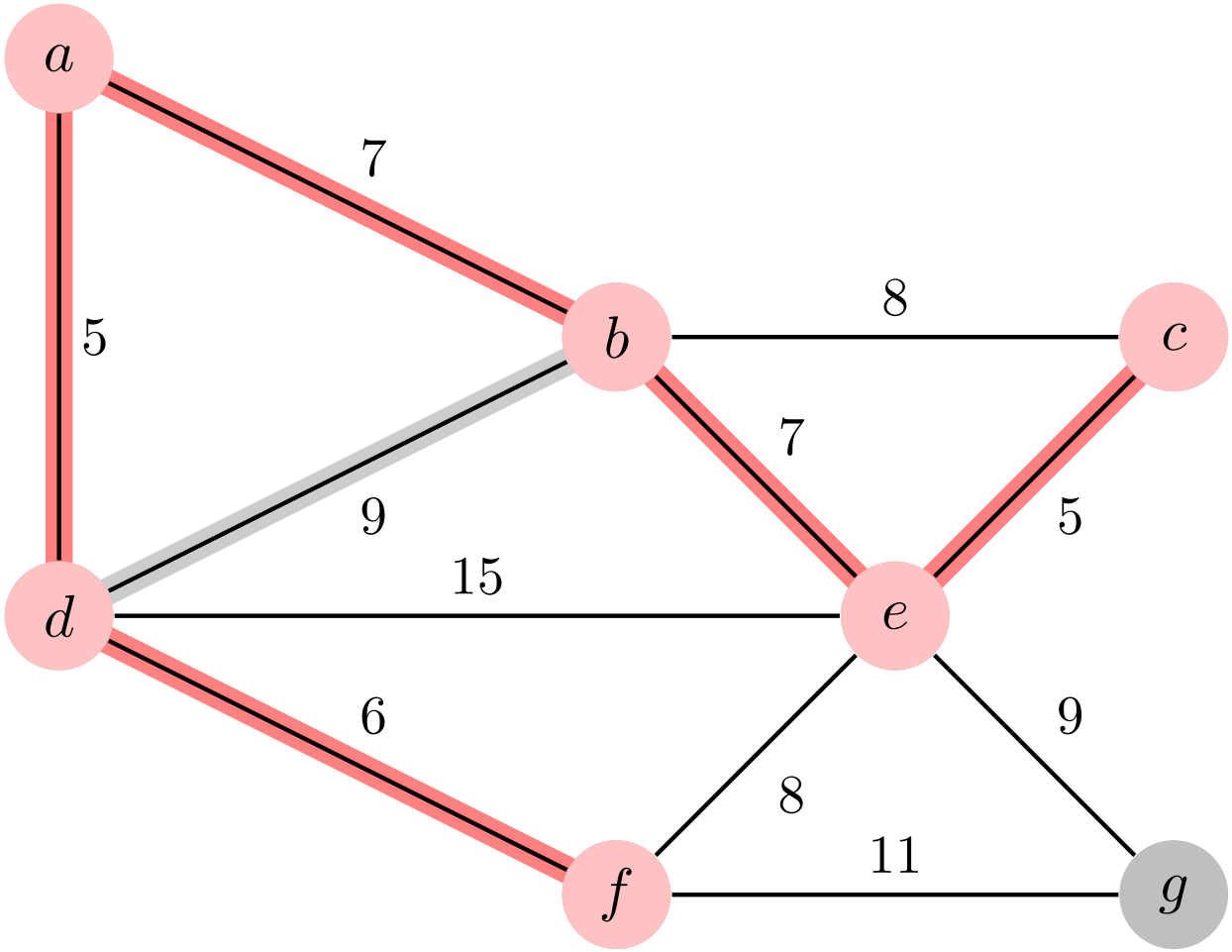
Kruskal - Példa



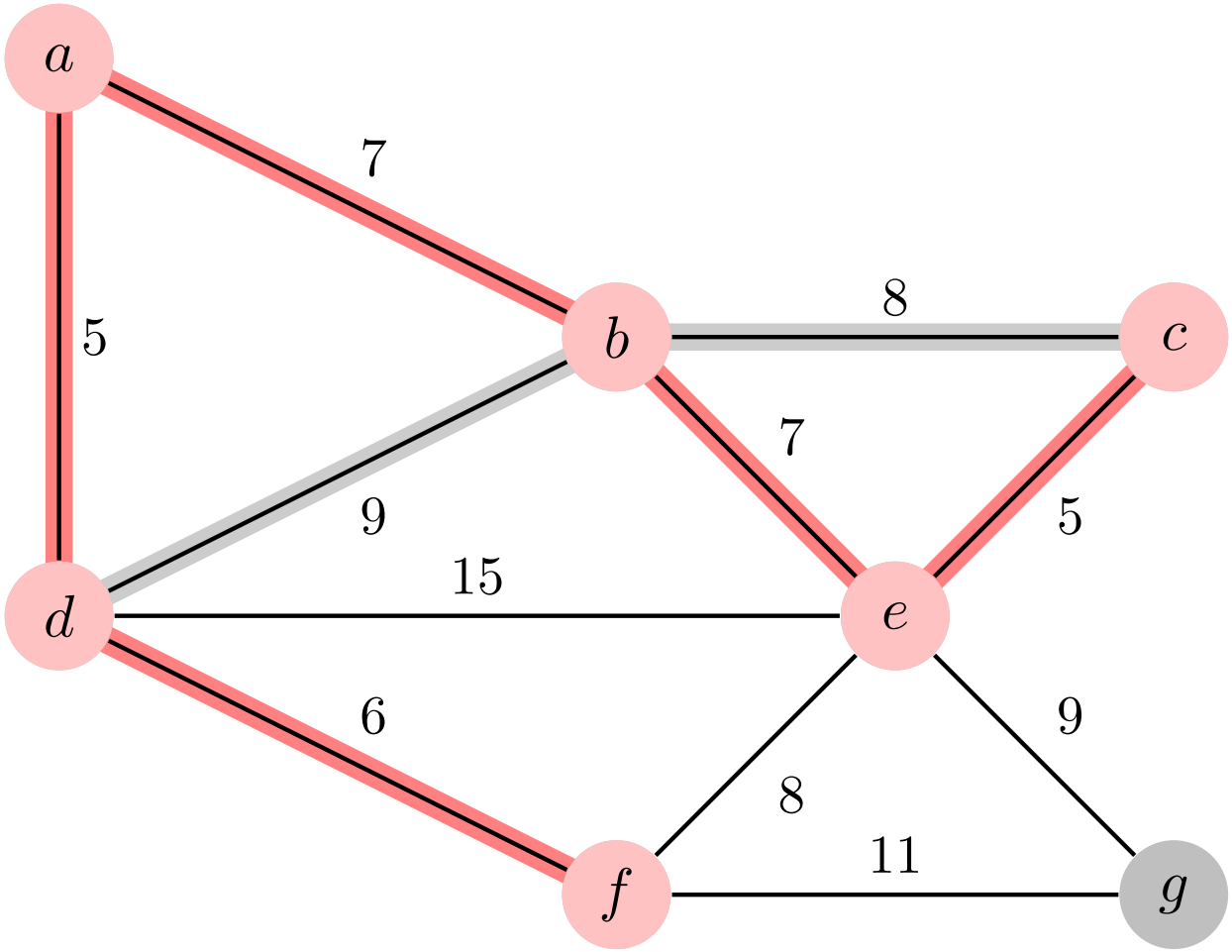
Kruskal - Példa



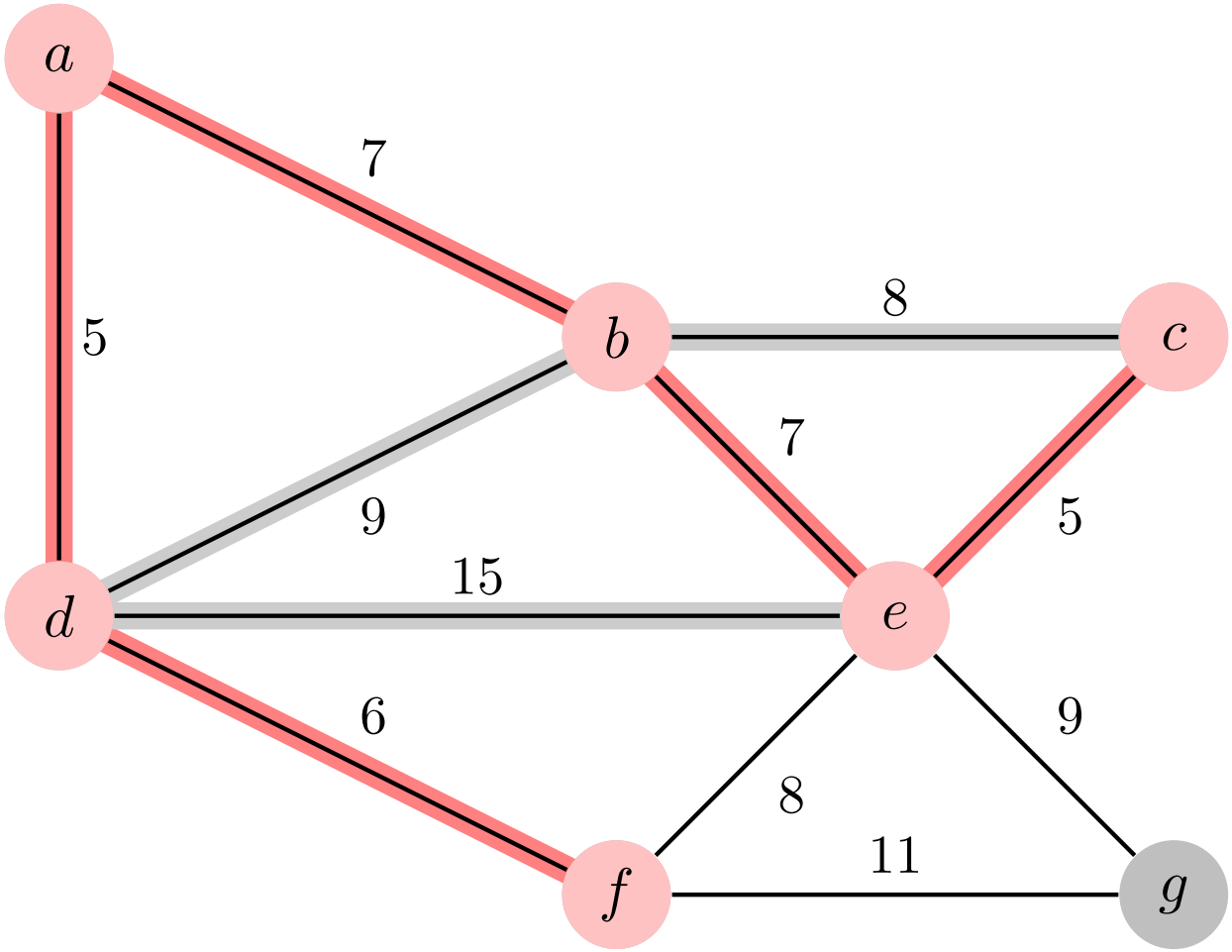
Kruskal - Példa



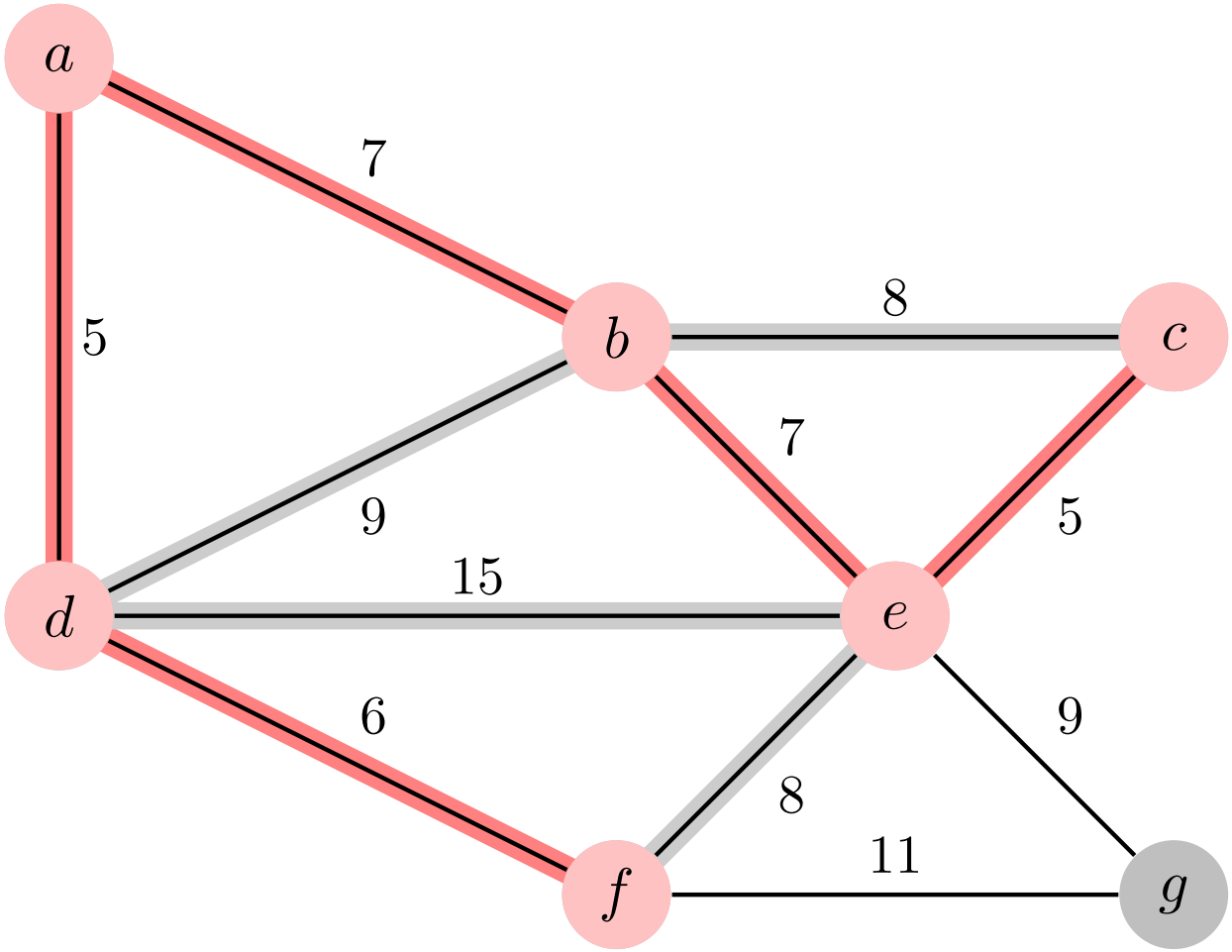
Kruskal - Példa



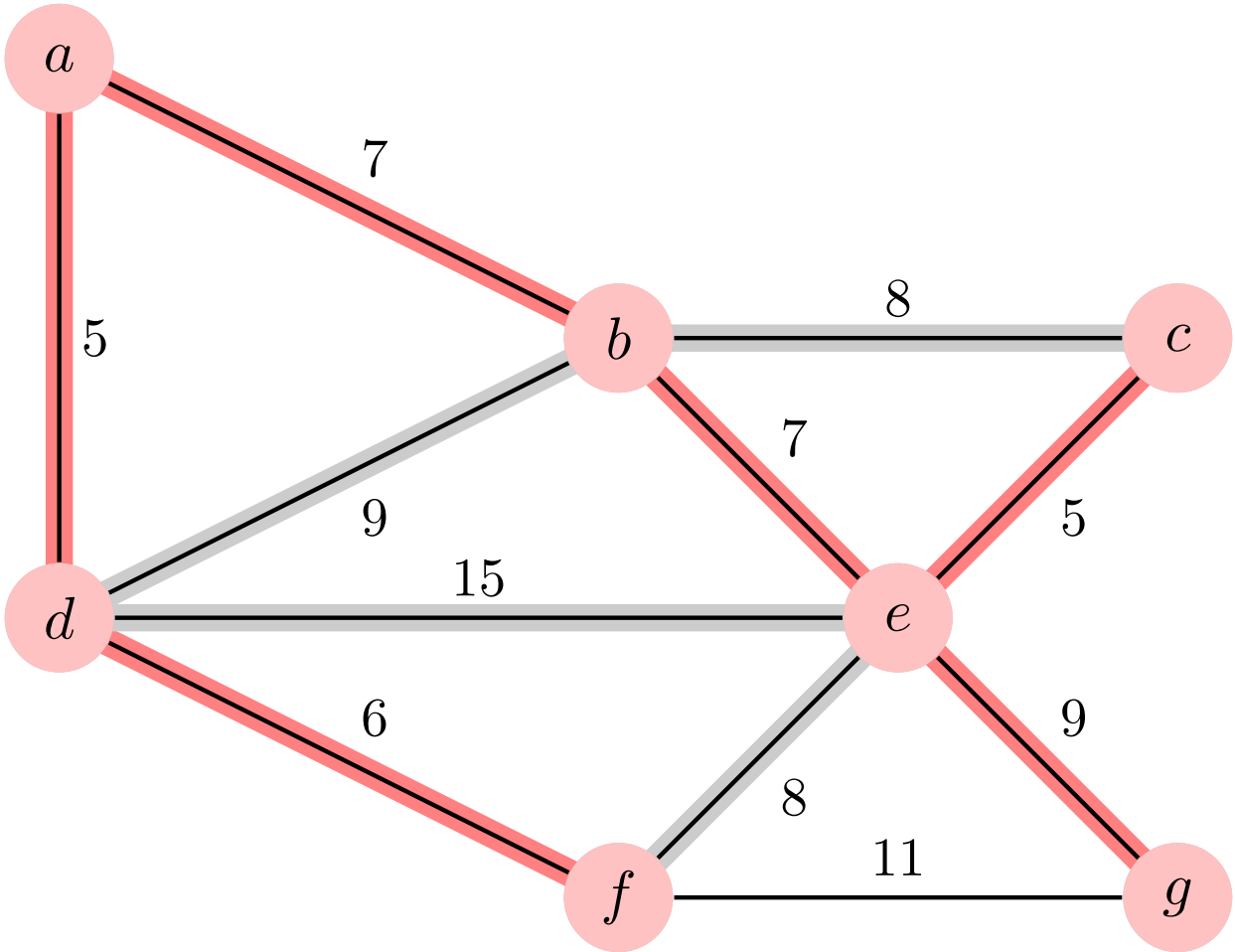
Kruskal - Példa



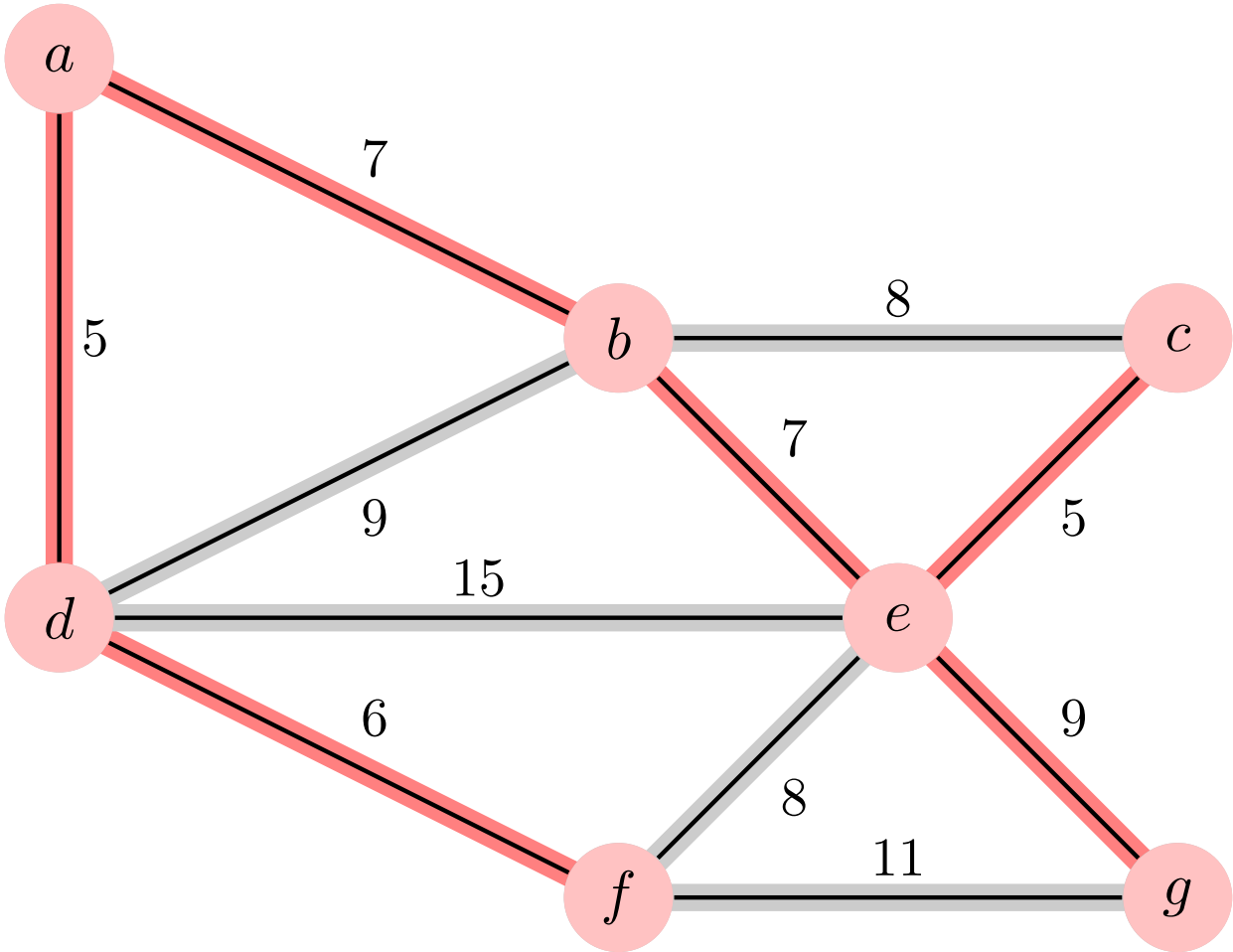
Kruskal - Példa



Kruskal - Példa

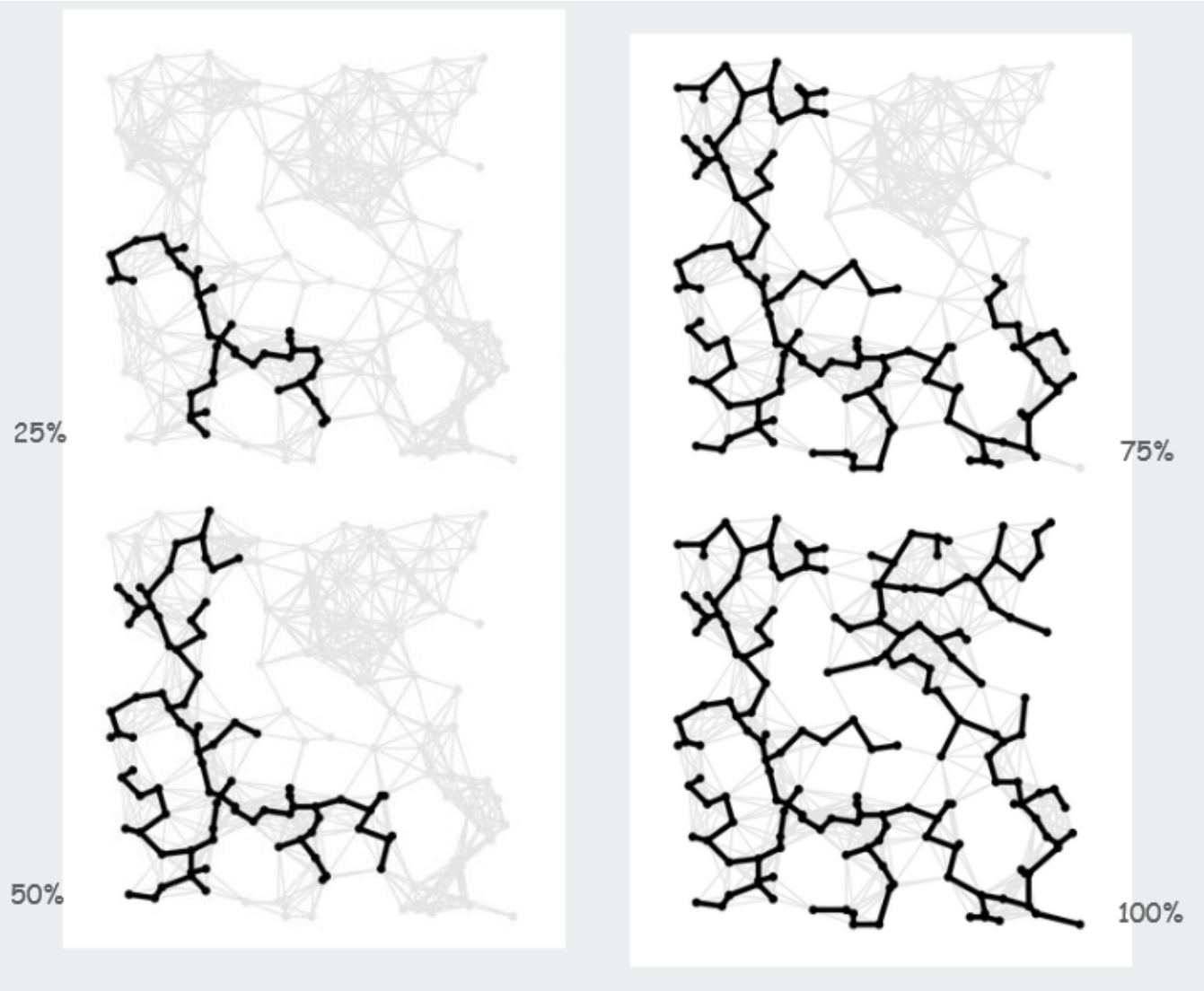


Kruskal - Példa



Prim algoritmusa

Jarník 1930, Dijkstra 1957, Prim 1959



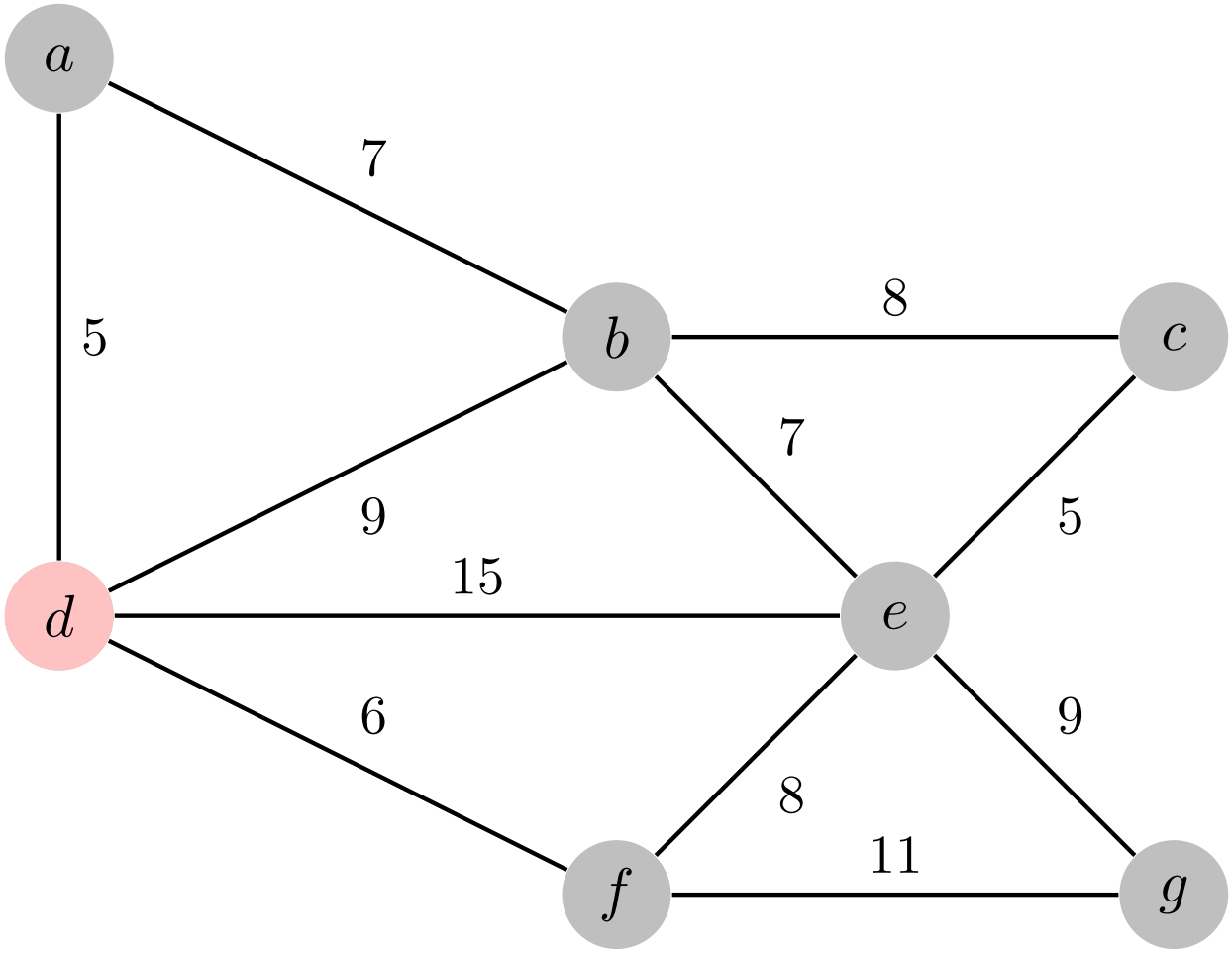
Prim algoritmusa

PRIM(G, x)

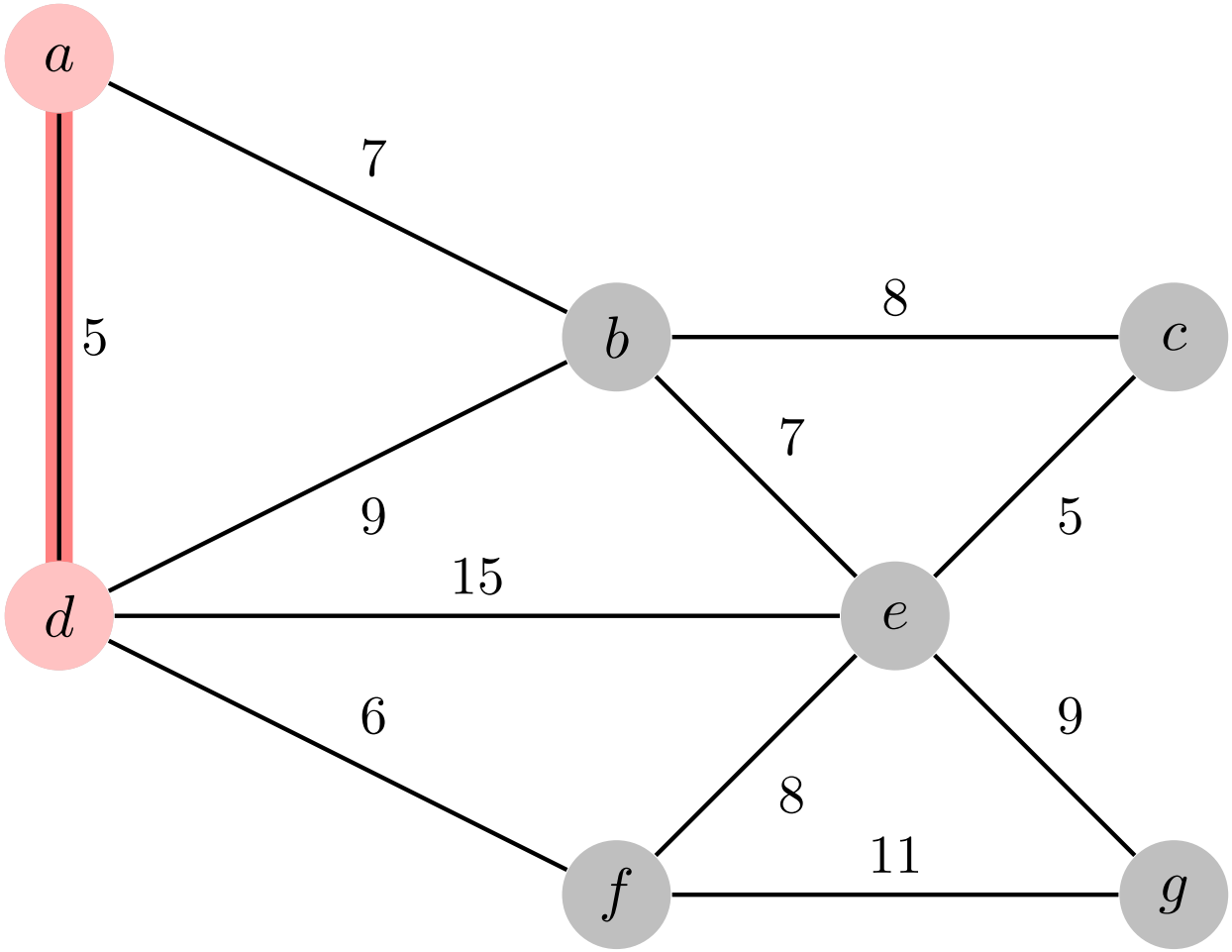
1. $A := \{x\}$
2. $B := V \setminus A$
3. **while** $A \neq V$ **do**
4. legyen $\{a, b\} \in E$, $a \in A$, $b \in B$ a legkisebb súlyú él az összes A és B közötti él közül
5. **kiír** $\{a, b\}$
6. $A := A \cup \{b\}$
7. $B := B \setminus \{b\}$

Bonyolultság: a használt adaszerkezettől függően $O(n^2)$, $O(mlogn)$ bináris kupaccal

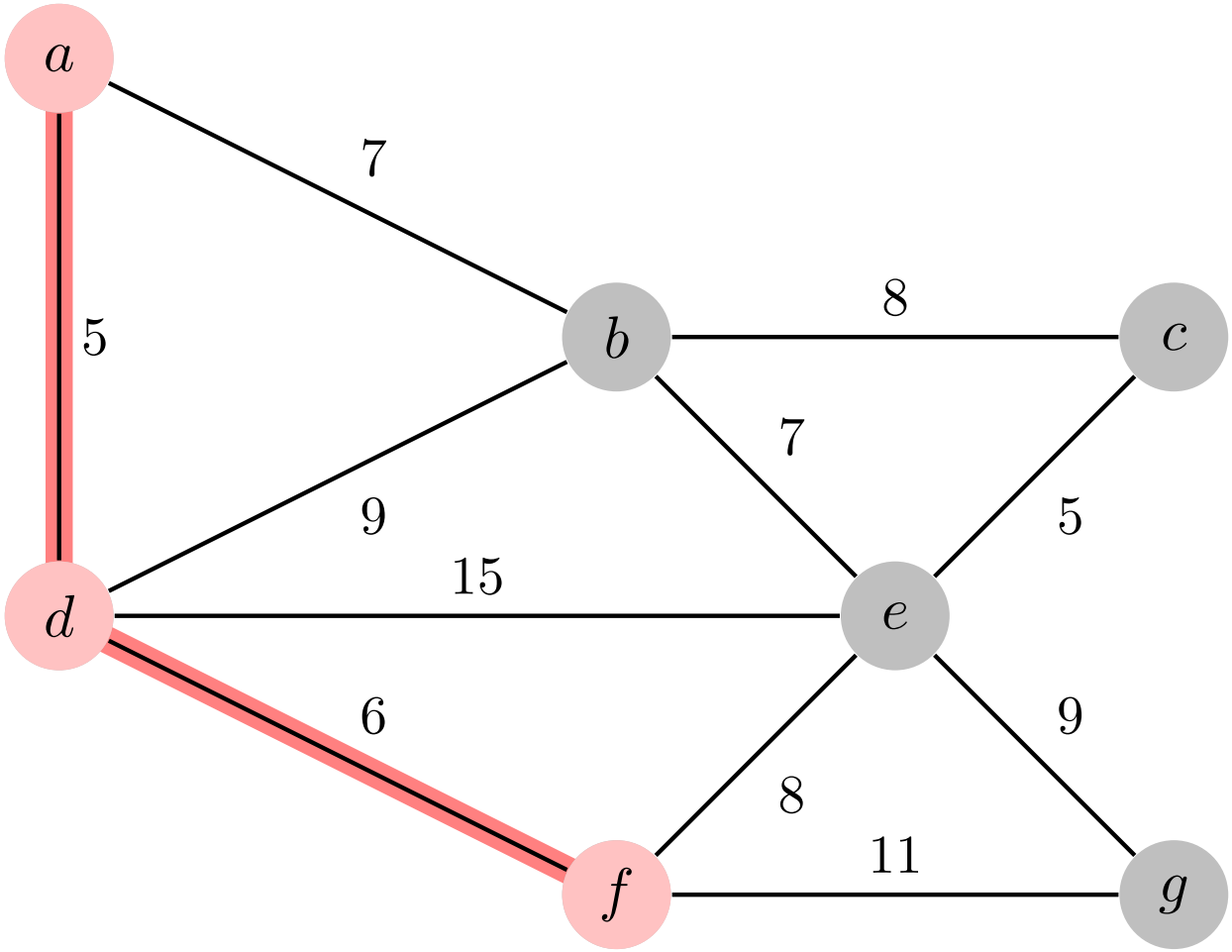
Prim - példa



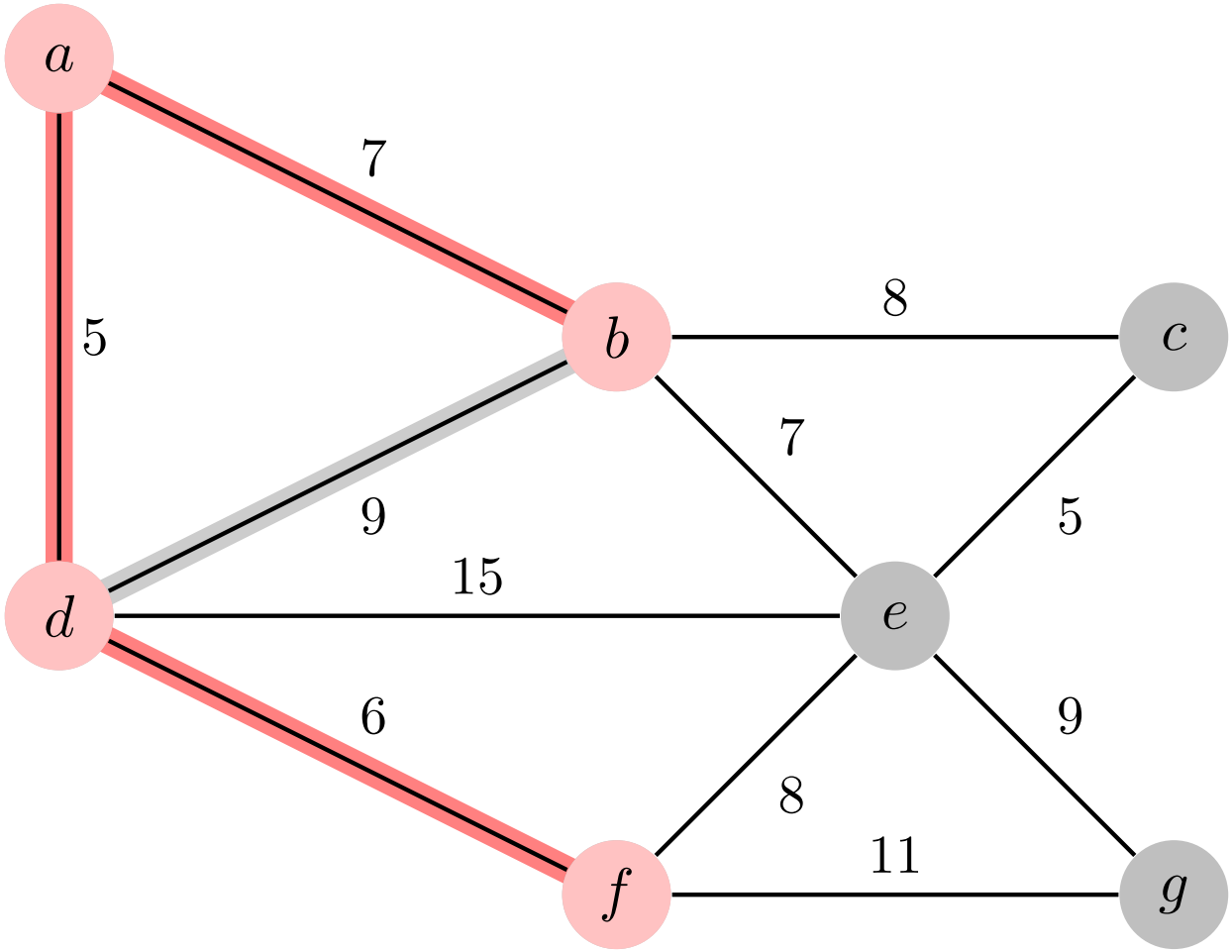
Prim - példa



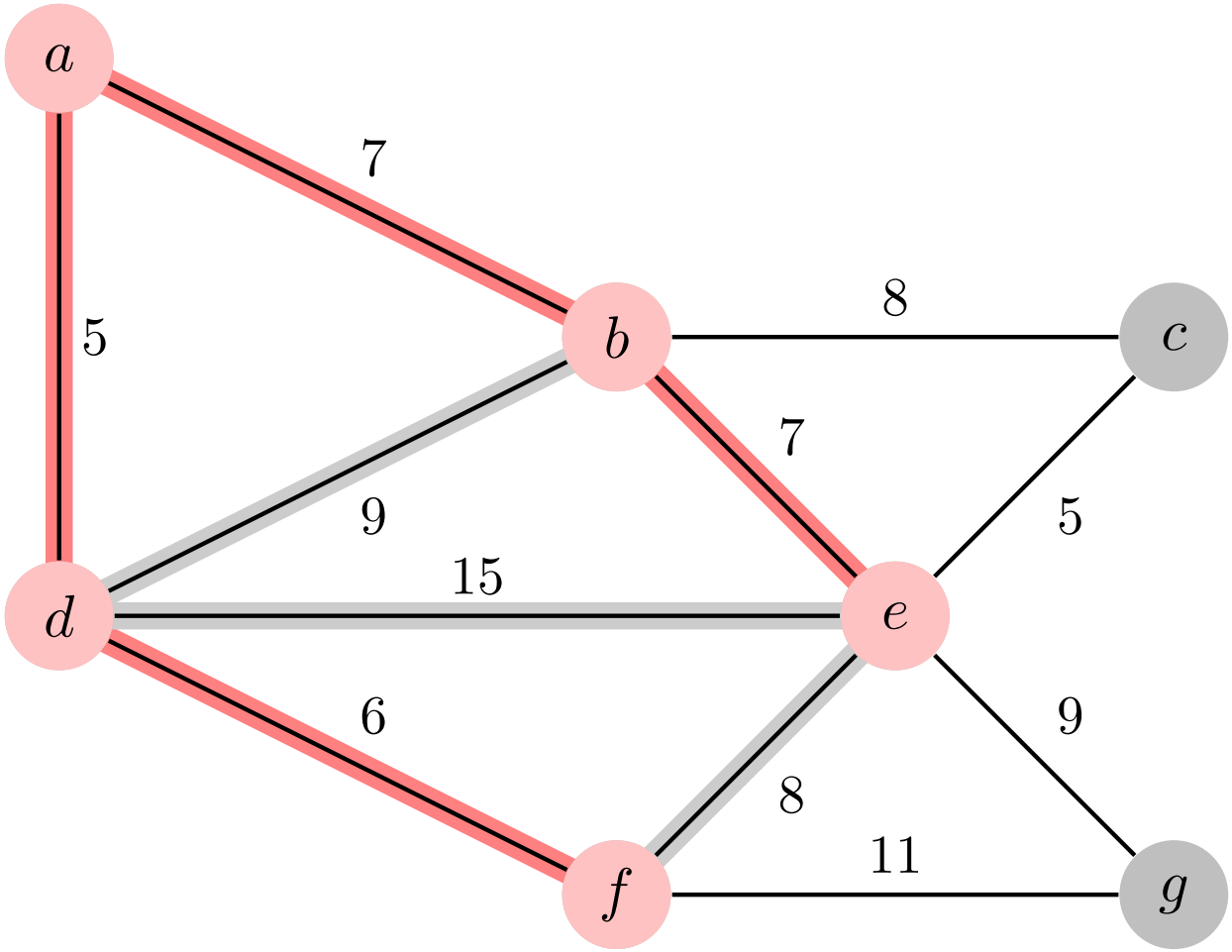
Prim - példa



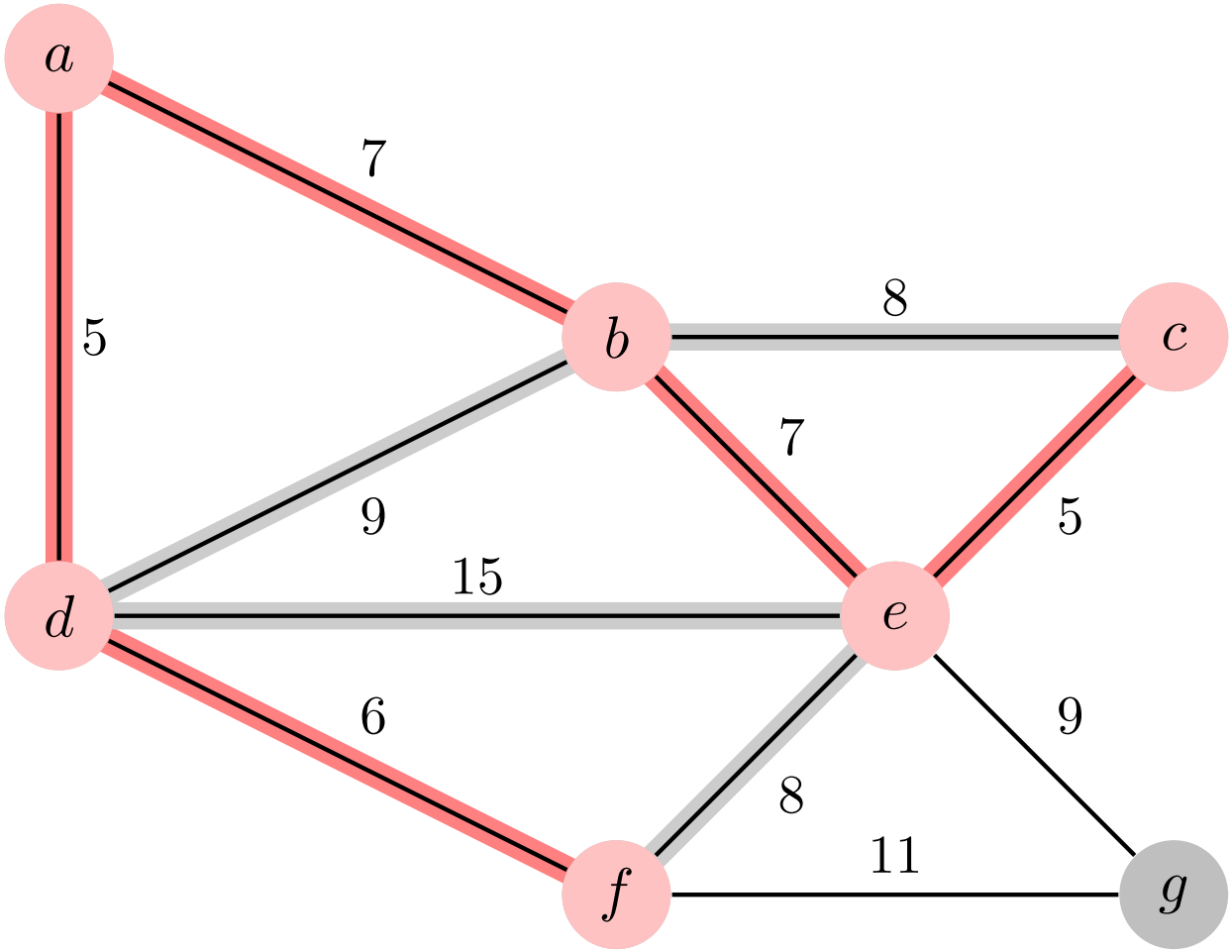
Prim - példa



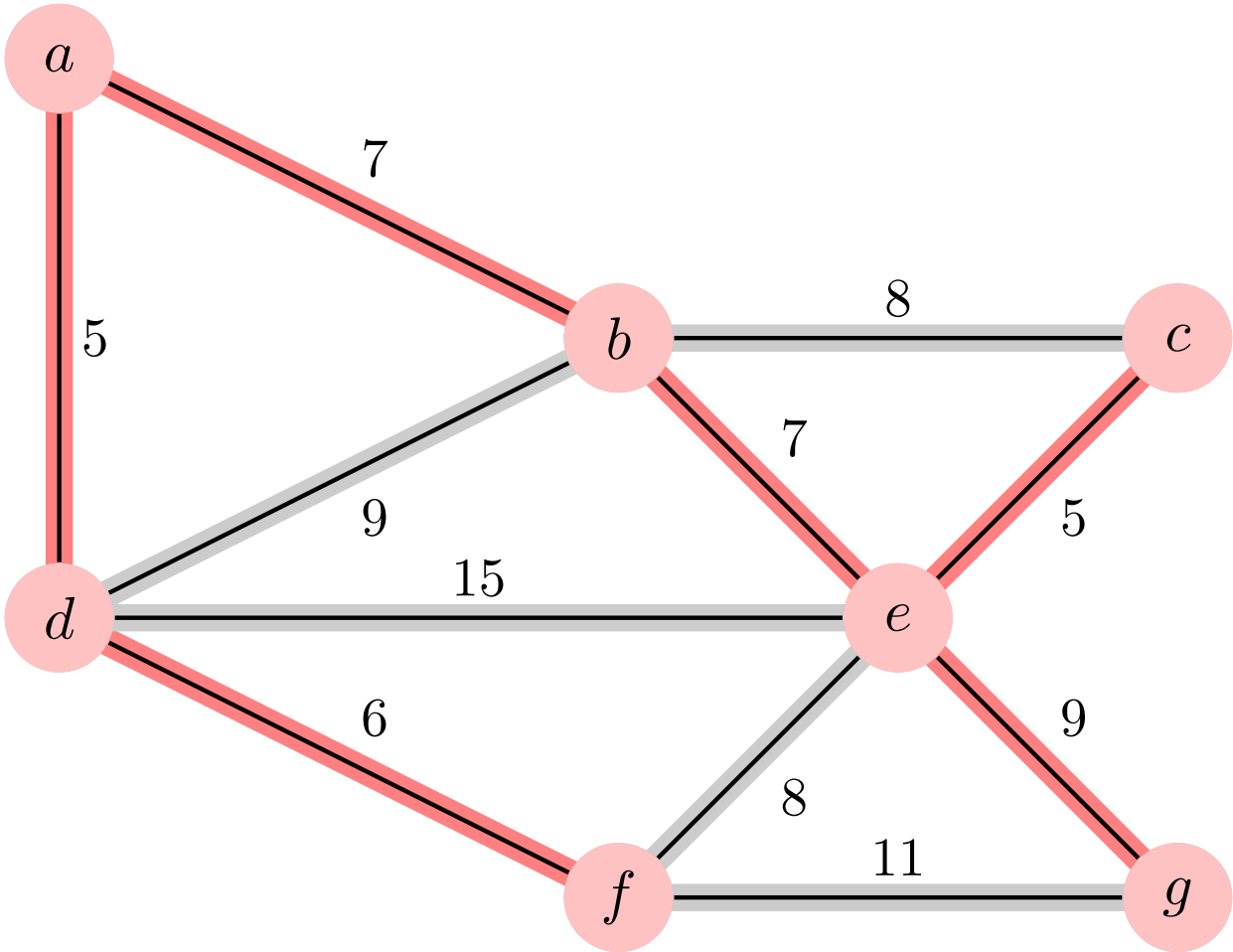
Prim - példa



Prim - példa



Prim - példa



Boruvka algoritmusa

Az algoritmus lépései:

- minden csomópont esetén válasszuk ki a minimális hosszúságú élt
- határozzuk meg az összefüggő komponenseket, az előző lépésben kiválasztott éleket tartalmazó gráfban
- ”egyesítjük” az összefüggő csomópontokat
- addig ismételjük ezeket a lépéseket, amig egy összefüggő gráfot kapunk

Példa: lásd 3_jegyzet.pdf

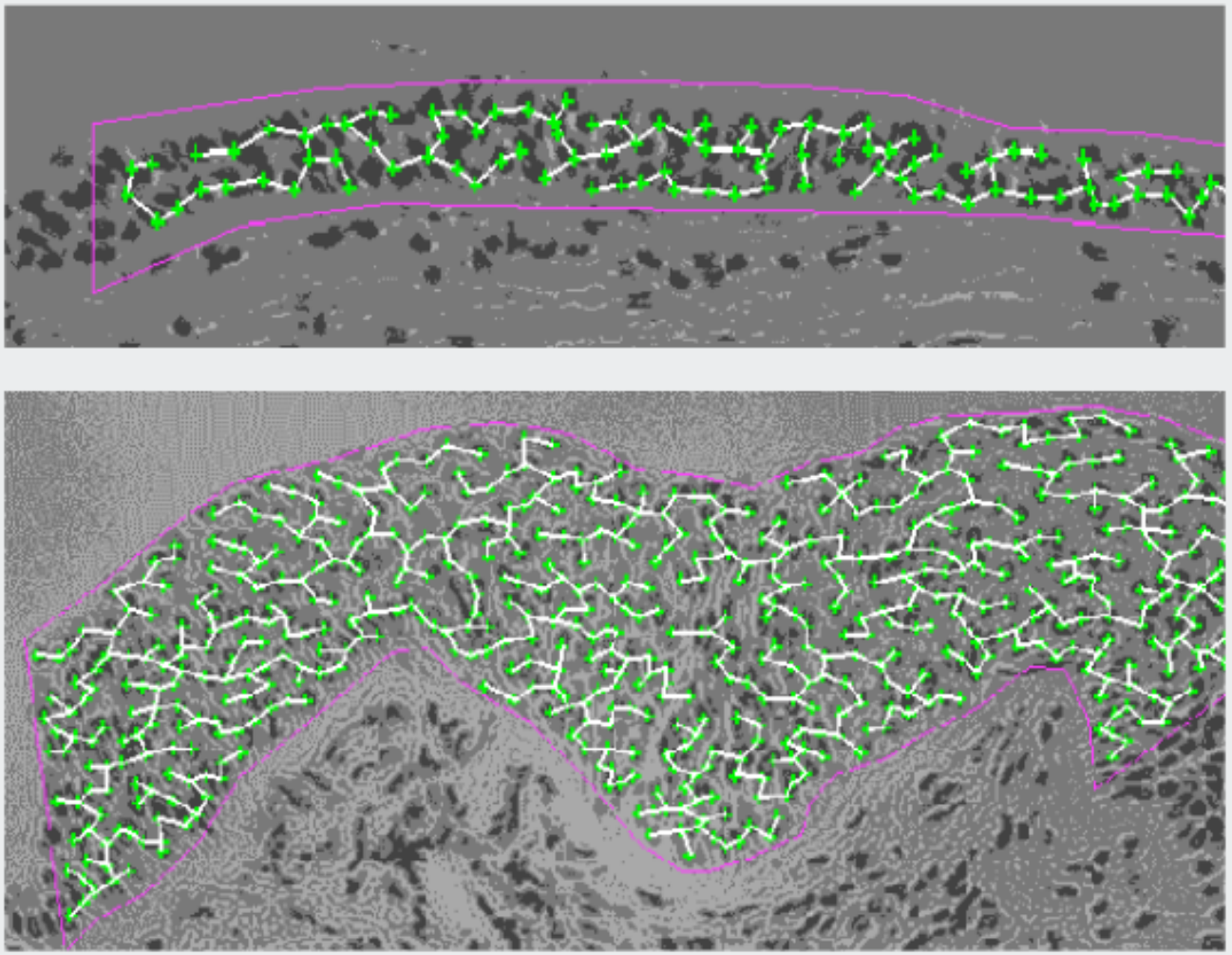
Fordított törlés algoritmusa

Alapötlet:

- az éleket a súlyúk szerinti csökkenő sorrendben járja be
- az eredeti gráfból kitörli a legnagyobb súlyú éleket, ha a törléssel a gráf nem esik szét két komponensre (ha nem nő az összefüggő komponensek száma)

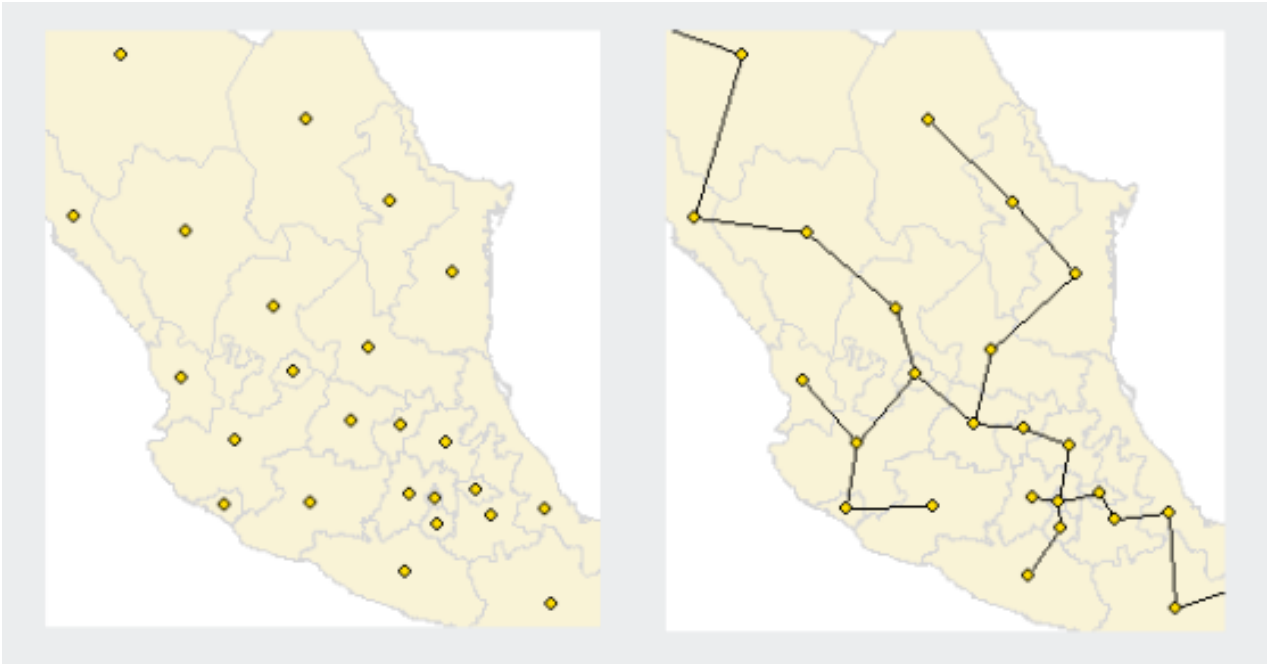
Példa: lásd 3_jegyzet.pdf

Alkalmazások



Alkalmazások 2

Euklideszi minimális feszítõfa:
Adott N pont a síkban, határozzuk meg a minimális feszítõfát.



Minimális feszítőfa irányított gráfokban

Irányított gyökeres fa

Egy irányított gráfot irányított gyökeres fának nevezünk, ha van egy kitüntetett s csomópont, és minden $u \neq s$ esetén pontosan egy irányított út létezik s-ből u-ba.

Gyökeres feszítőfa, minimális gyökeres feszítőfa

Egy irányított gráf gyökeres feszítőfája s pontra nézve egy s gyökerű irányított gyökeres fa, mely egy maximális részgráf. Ha az élek súlyának összege minimális, akkor minimális gyökeres feszítőfáról beszélünk.

Gabow-Tarjan algoritmus

A Prim algoritmus irányított gráfokra adaptálva

Gabow-Tarjan algoritmus

A Prim algoritmus irányított gráfokra adaptálva

Egy példa: lásd 4_jegyzet.pdf

Chiu-Liu/Edmonds algoritmus

Chiu-Liu 1965, Edmonds 1967
Az algoritmus lépései:

- minden csomópont esetén válasszuk ki a legkiseb bemenő élt, és vonjuk ki a többi bemenő élből
- ha kört találtunk húzuk össze egy csomópontba
- rekurzivan keressük meg a minimális szerteágazást az összehúzott gráfban
- "kibontjuk" az összehúzott gráfot

Példa: lásd 4_jegyzet.pdf