Algoritmuselmélet

Gráfok megadása, szélességi bejárás, összefüggőség, párosítás

Katona Gyula Y.

Számítástudományi és Információelméleti Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem

2. előadás

Gráfalgoritmusok

- irányított gráfok: $G = (V, E)$

Gráfalgoritmusok

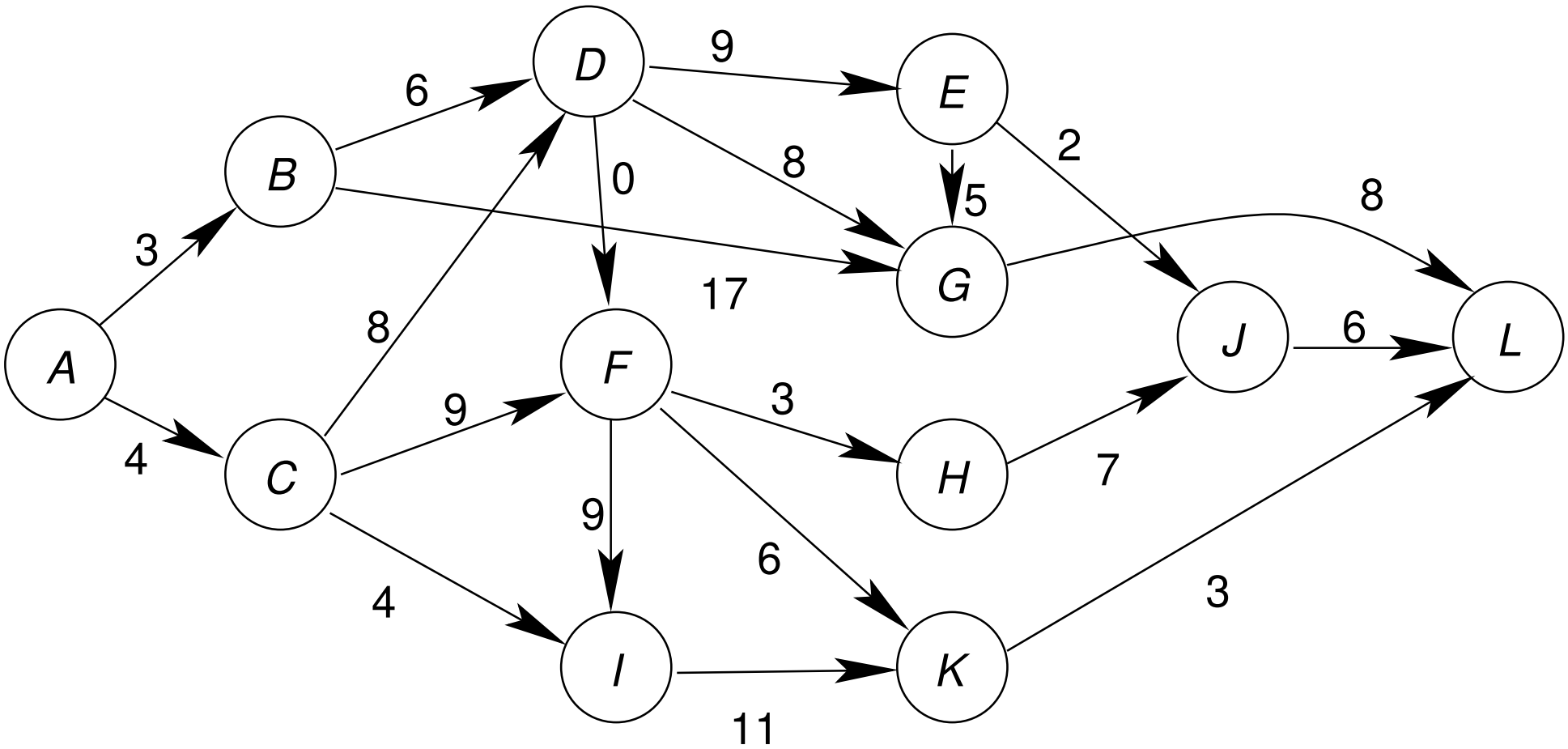
- irányított gráfok: $G = (V, E)$
- irányított él, irányított út, irányított kör

Gráfalgoritmusok

- irányított gráfok: $G = (V, E)$
- irányított él, irányított út, irányított kör
- élsúlyok: $c(e)$ — lehetnek negatívak is

Gráfalgoritmusok

- irányított gráfok: $G = (V, E)$
- irányított él, irányított út, irányított kör
- élsúlyok: $c(e)$ — lehetnek negatívak is



Adjacencia-mátrix

Definíció

A $G = (V, E)$ gráf adjacencia-mátrixa (vagy szomszédossági mátrixa) a következő – a V elemeivel indexelt – n-szer n-es mátrix:

$$A[i,j] = \begin{cases} 0 & ha\ (i,j) \notin E, \\ 1 & ha\ (i,j) \in E. \end{cases}$$

Adjacencia-mátrix

Definíció

A $G = (V, E)$ gráf adjacencia-mátrixa (vagy szomszédossági mátrixa) a következő – a V elemeivel indexelt – n-szer n-es mátrix:

$$A[i,j] = \begin{cases} 0 & \text{ha } (i,j) \notin E, \\ 1 & \text{ha } (i,j) \in E. \end{cases}$$

Irányítatlan gráfok esetén a szomszédossági mátrix szimmetrikus lesz (azaz $A[i,j] = A[j,i]$ teljesül minden i, j csúcspárra).

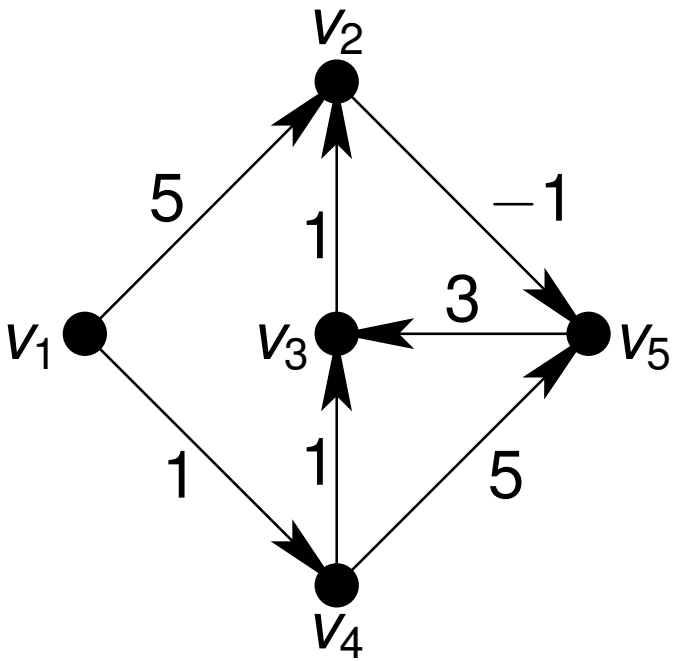
Adjacencia-mátrix

Definíció

A $G = (V, E)$ gráf adjacencia-mátrixa (vagy szomszédossági mátrixa) a következő – a V elemeivel indexelt – n-szer n-es mátrix:

$$A[i,j] = \begin{cases} 0 & ha\ (i,j) \notin E, \\ 1 & ha\ (i,j) \in E. \end{cases}$$

Irányítatlan gráfok esetén a szomszédossági mátrix szimmetrikus lesz (azaz $A[i,j] = A[j,i]$ teljesül minden i, j csúcspárra).



$$A = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Súlyozott adjacencia-mátrix

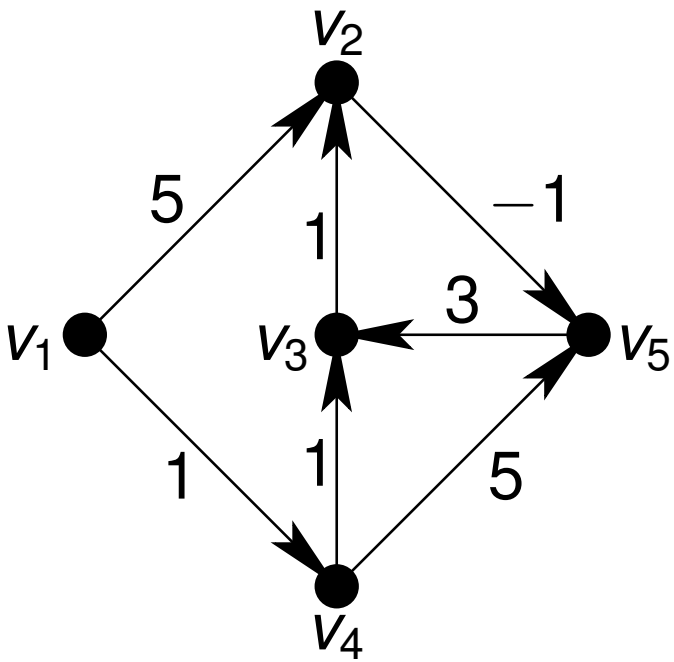
Ha költségek is vannak $\Longrightarrow$

$$C[i,j] = \begin{cases} 0 & \text{ha } i = j, \\ c(i,j) & \text{ha } i \neq j \text{ és } (i,j) \text{ éle } G\text{-nek}, \\ * & \text{különben.} \end{cases}$$

Súlyozott adjacencia-mátrix

Ha költségek is vannak $\Longrightarrow$

$$C[i,j] = \begin{cases} 0 & \text{ha } i = j, \\ c(i,j) & \text{ha } i \neq j \text{ és } (i,j) \text{ éle } G\text{-nek}, \\ * & \text{különben.} \end{cases}$$

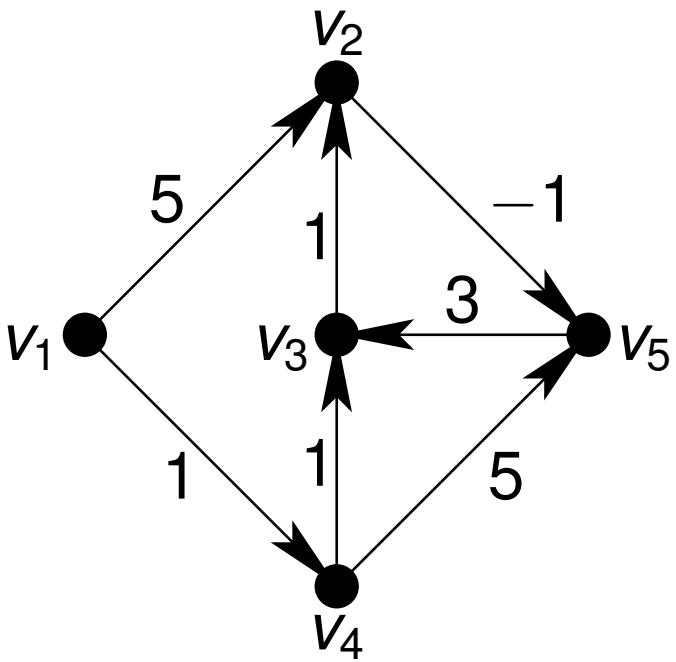


$$C = \begin{pmatrix} 0 & 5 & * & 1 & * \\ * & 0 & * & * & -1 \\ * & 1 & 0 & * & * \\ * & * & 1 & 0 & 5 \\ * & * & 3 & * & 0 \end{pmatrix}$$

Súlyozott adjacencia-mátrix

Ha költségek is vannak $\Longrightarrow$

$$C[i,j] = \begin{cases} 0 & \text{ha } i = j, \\ c(i,j) & \text{ha } i \neq j \text{ és } (i,j) \text{ éle } G\text{-nek}, \\ * & \text{különben.} \end{cases}$$



$$C = \begin{pmatrix} 0 & 5 & * & 1 & * \\ * & 0 & * & * & -1 \\ * & 1 & 0 & * & * \\ * & * & 1 & 0 & 5 \\ * & * & 3 & * & 0 \end{pmatrix}$$

Hátránya: a mérete (n^2 tömbelem) teljesen független az élek számától.

Éllistás megadás

$G = (V, E)$ gráf minden csúcsához egy lista tartozik.

Éllistás megadás

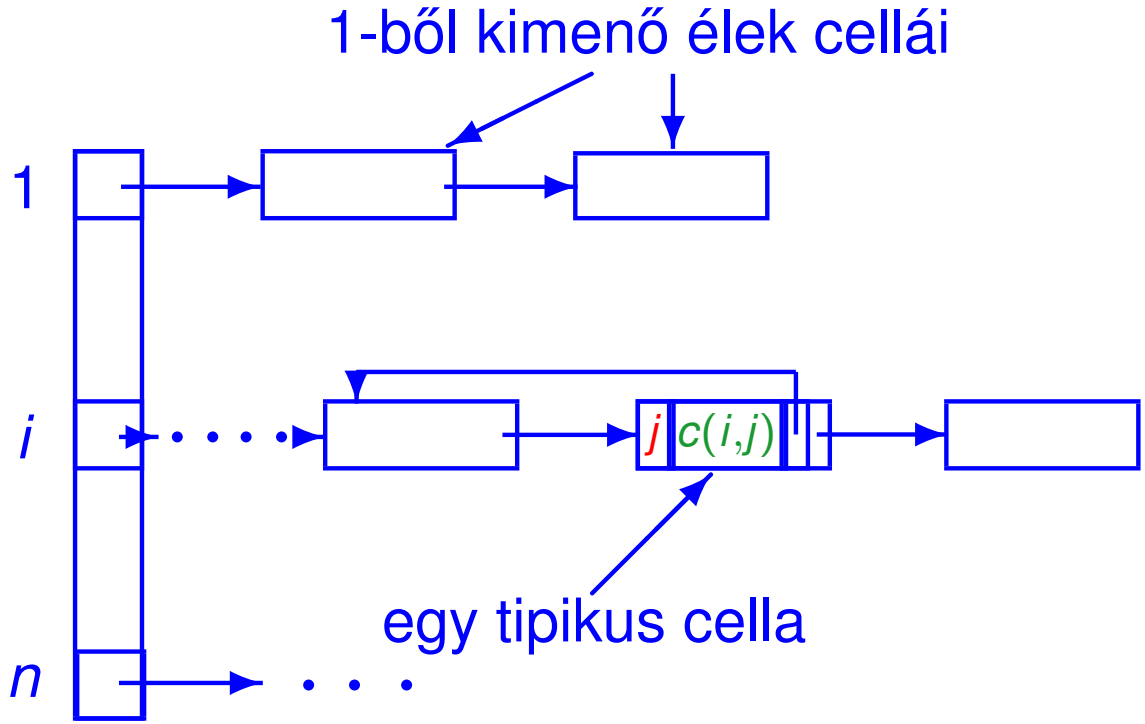
$G = (V, E)$ gráf minden csúcsához egy lista tartozik.
Az $i \in V$ csúcs listájában tároljuk az i-ből kimenő éleket, és ha kell, ezek súlyát is.

Éllistás megadás

$G = (V, E)$ gráf minden csúcsához egy lista tartozik.
Az $i \in V$ csúcs listájában tároljuk az i-ből kimenő éleket, és ha kell, ezek súlyát is.
Az i listáján egy élnek a lista egy eleme (cellája) felel meg.

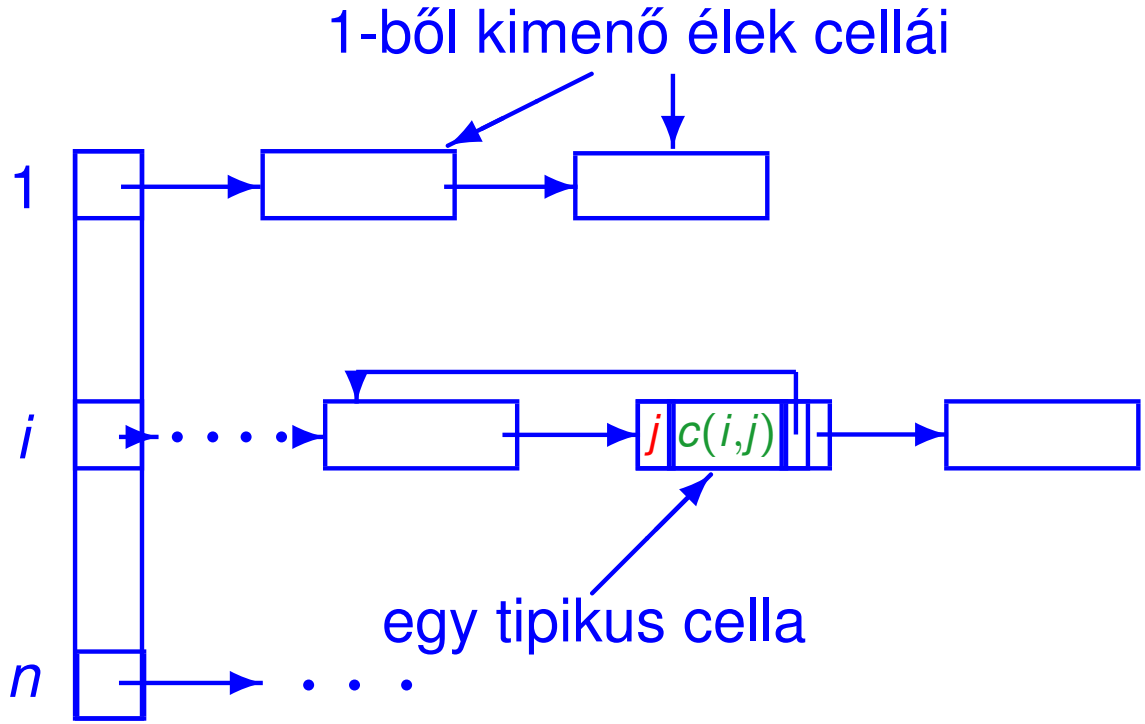
Éllistás megadás

$G = (V, E)$ gráf minden csúcsához egy lista tartozik.
Az $i \in V$ csúcs listájában tároljuk az i-ből kimenő éleket, és ha kell, ezek súlyát is.
Az i listáján egy élnek a lista egy eleme (cellája) felel meg.



Éllistás megadás

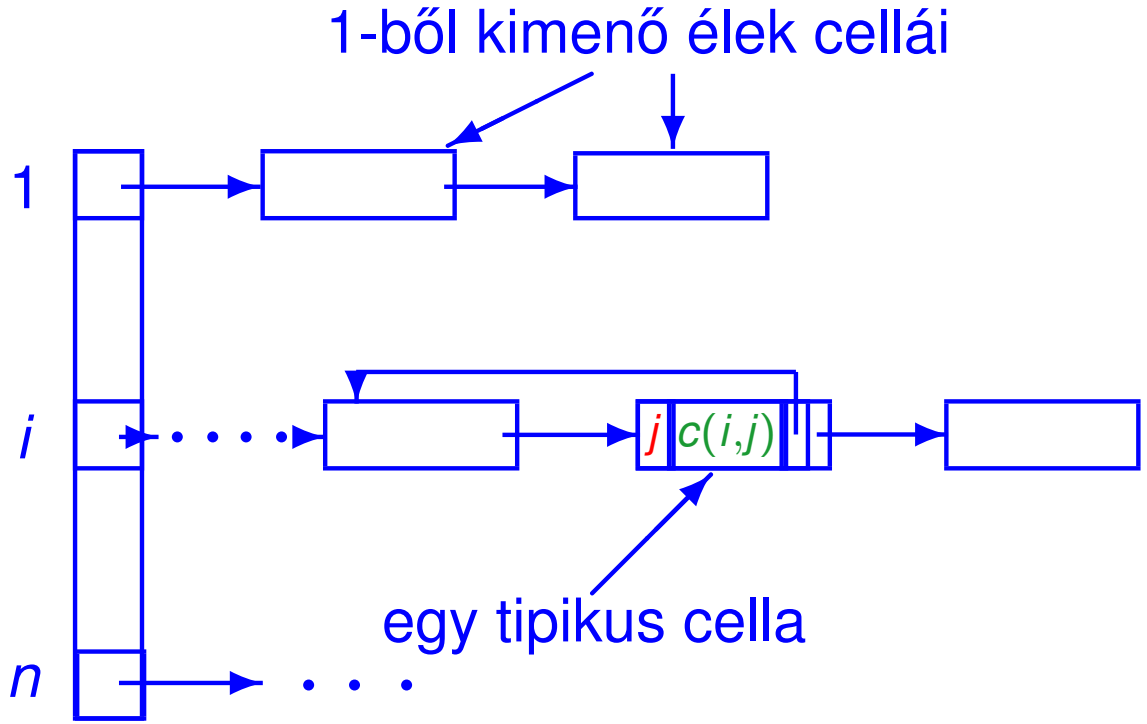
$G = (V, E)$ gráf minden csúcsához egy lista tartozik.
Az $i \in V$ csúcs listájában tároljuk az i-ből kimenő éleket, és ha kell, ezek súlyát is.
Az i listáján egy élnek a lista egy eleme (cellája) felel meg.



Az (i, j) élnek megfelelő cella tartalmazza a j sorszámot, a $c(i, j)$ súlyt (ha van), egy mutatót a következő cellára, és esetleg még egyet az előzőre is.

Éllistás megadás

$G = (V, E)$ gráf minden csúcsához egy lista tartozik.
Az $i \in V$ csúcs listájában tároljuk az i-ből kimenő éleket, és ha kell, ezek súlyát is.
Az i listáján egy élnek a lista egy eleme (cellája) felel meg.



Az (i, j) élnek megfelelő cella tartalmazza a j sorszámot, a $c(i, j)$ súlyt (ha van), egy mutatót a következő cellára, és esetleg még egyet az előzőre is.
Tárigény: $n + e$ cella,
Irányítatlan gráfoknál:
$n + 2e$ cella

Szélességi bejárás

BFS: Breadth First Search

Szélességi bejárás

BFS: Breadth First Search

Meglátogatjuk az első csúcsot, majd ennek a csúcsnak az összes szomszédját.

Szélességi bejárás

BFS: Breadth First Search

Meglátogatjuk az első csúcsot, majd ennek a csúcsnak az összes szomszédját. Aztán ezen szomszédok összes olyan szomszédját, ahol még nem jártunk, és így tovább.

Szélességi bejárás

BFS: Breadth First Search

Meglátogatjuk az első csúcsot, majd ennek a csúcsnak az összes szomszédját. Aztán ezen szomszédok összes olyan szomszédját, ahol még nem jártunk, és így tovább.
megvalósítás: *sor* (FIFO-lista)

Szélességi bejárás

BFS: Breadth First Search

Meglátogatjuk az első csúcsot, majd ennek a csúcsnak az összes szomszédját. Aztán ezen szomszédok összes olyan szomszédját, ahol még nem jártunk, és így tovább.
megvalósítás: *sor* (FIFO-lista)
Berakjuk az épp meglátogatott csúcsot, hogy majd a megfelelő időben a szomszédaira is sort keríthessünk.

Szélességi bejárás

BFS: Breadth First Search

Meglátogatjuk az első csúcsot, majd ennek a csúcsnak az összes szomszédját. Aztán ezen szomszédok összes olyan szomszédját, ahol még nem jártunk, és így tovább.
megvalósítás: *sor* (FIFO-lista)
Berakjuk az épp meglátogatott csúcsot, hogy majd a megfelelő időben a szomszédaira is sort keríthessünk.
Általános lépés: vesszük a sor elején levő x csúcsot, töröljük a sorból, meglátogatjuk azokat az y szomszédait, amelyeket eddig még nem láttunk, majd ezeket az y csúcsokat a sor végére tesszük.

Szélességi bejárás

```
procedure szb (v: csúcs) (* szélességi bejárás egy komponensre *)
    var
        Q: csúcsokból álló sor;
        x, y: csúcsok;
    begin
        bejárva[v] := igaz;
        sorba(v, Q);
        while Q nem üres do begin
            x := első(Q); (ez egyben törli is x-et a sorból)
            for minden x → y ∈ E élre do
                if bejárva[y] = hamis then begin
                    bejárva[y] := igaz;
(*)                 sorba(y, Q)
                end
        end
    end
```

Szélességi bejárás

```
procedure bejár (* szélességi bejárás minden komponensre *)
    begin
        for v := 1 to n do
            bejárva[v] := hamis;
        for v := 1 to n do
            if bejárva[v] = hamis then
              szb(v)
    end
```

Szélességi bejárás

```
procedure bejár (* szélességi bejárás minden komponensre *)
    begin
        for v := 1 to n do
            bejárva[v] := hamis;
        for v := 1 to n do
            if bejárva[v] = hamis then
              szb(v)
    end
```

Összköltség: $O(n + e)$

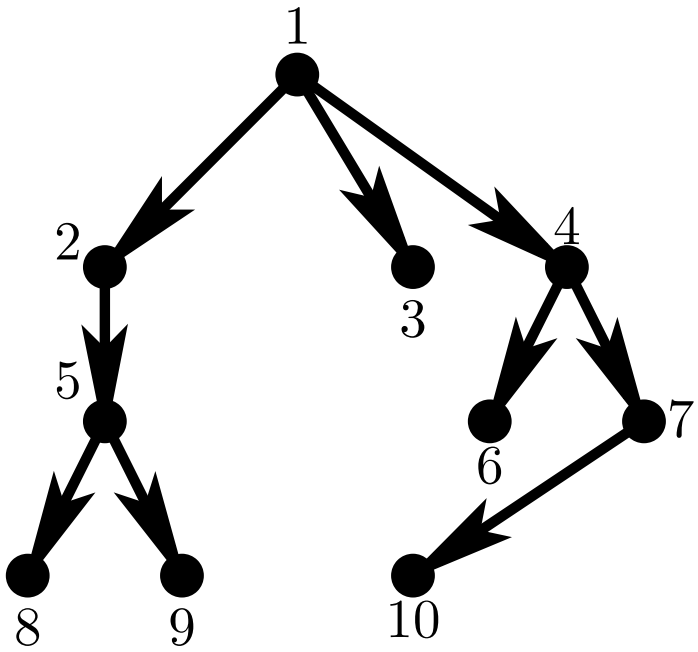
Szélességi bejárás

```
procedure bejár (* szélességi bejárás minden komponensre *)
    begin
        for v := 1 to n do
            bejárva[v] := hamis;
        for v := 1 to n do
            if bejárva[v] = hamis then
              szb(v)
    end
```

Összköltség: $O(n + e)$

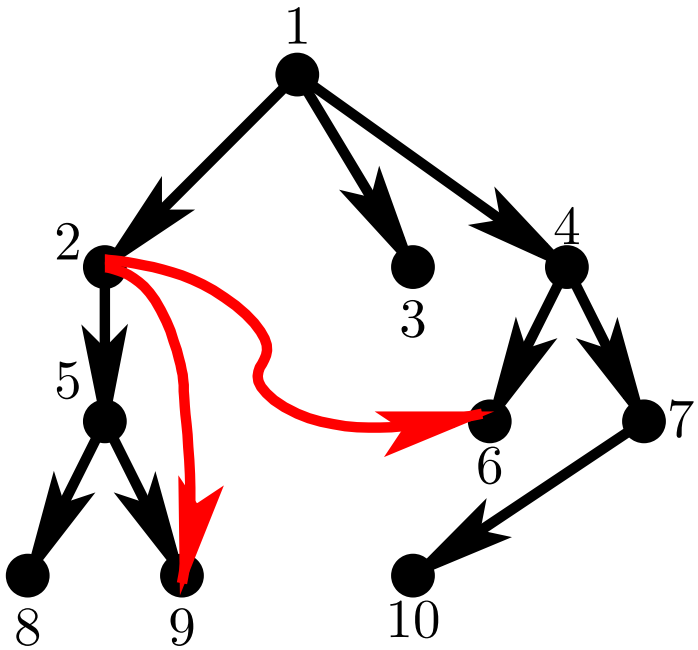
Faél: megvizsgálásukkor még be nem járt pontba mutatnak

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

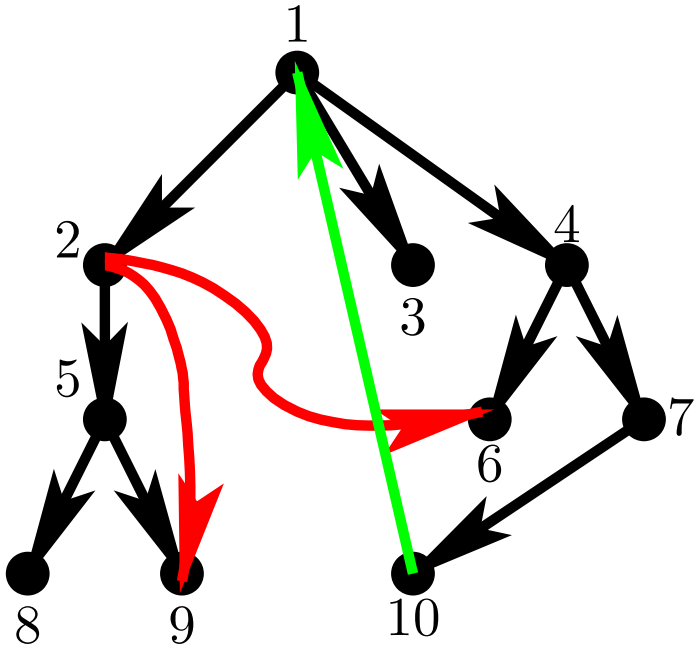
Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

ilyen nincs

Faél: megvizsgálásukkor még be nem járt pontba mutatnak

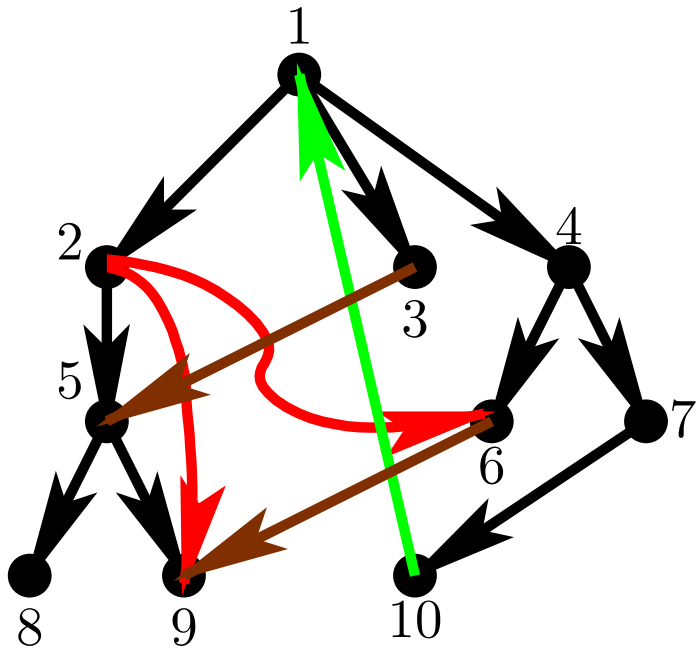


faél

ilyen nincs

visszaél

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



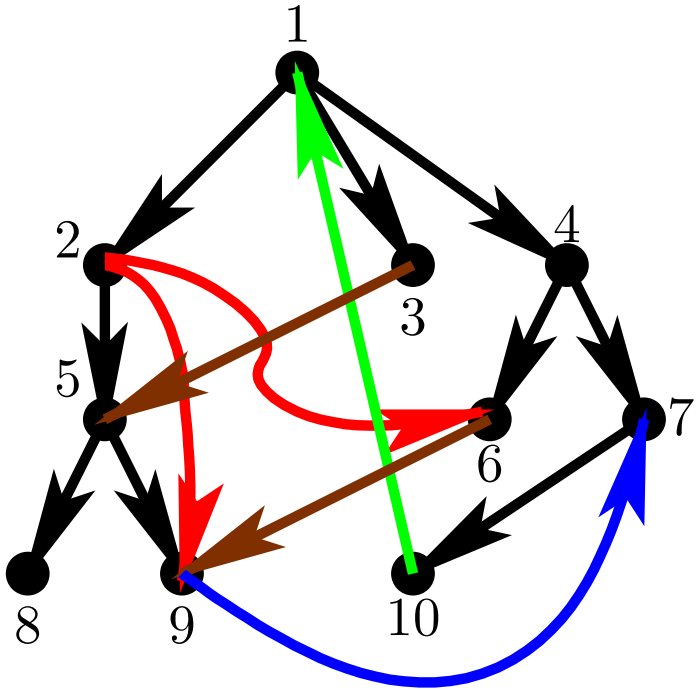
faél

ilyen nincs

visszaél

keresztél

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

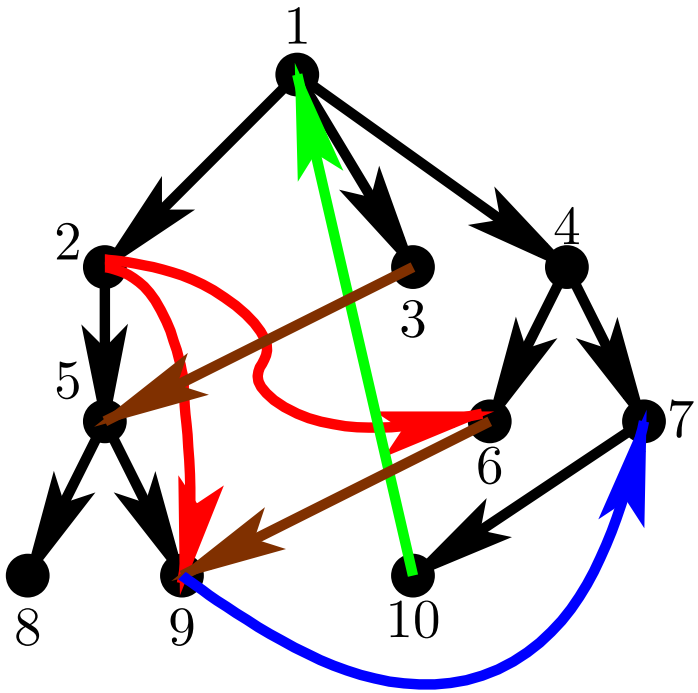
ilyen nincs

visszaél

keresztél

keresztél

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

ilyen nincs

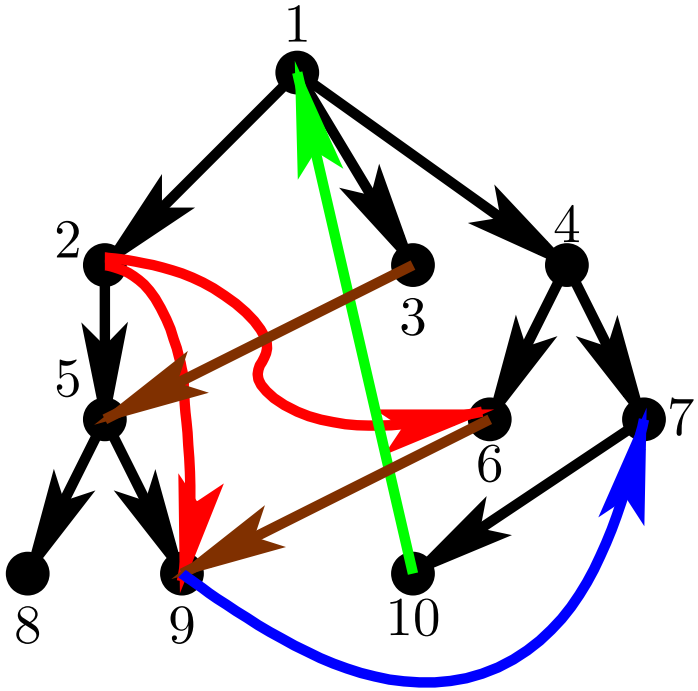
visszaél

keresztél

keresztél

Irányítatlan esetben csak faél és keresztél lehet.

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

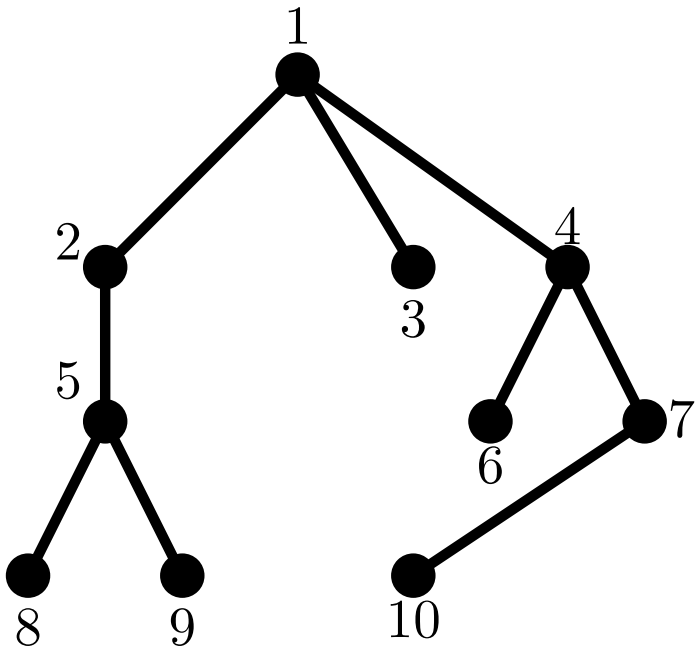
ilyen nincs

visszaél

keresztél

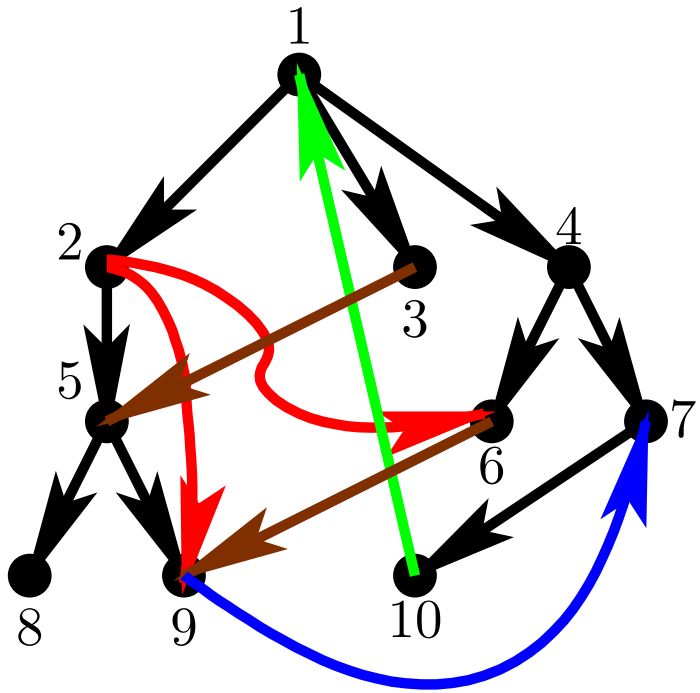
keresztél

Irányítatlan esetben csak faél és keresztél lehet.



faél

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

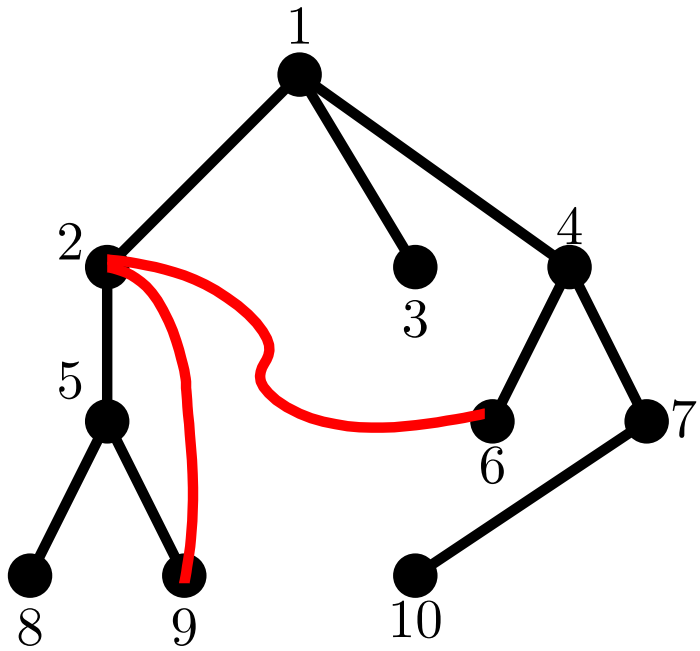
ilyen nincs

visszaél

keresztél

keresztél

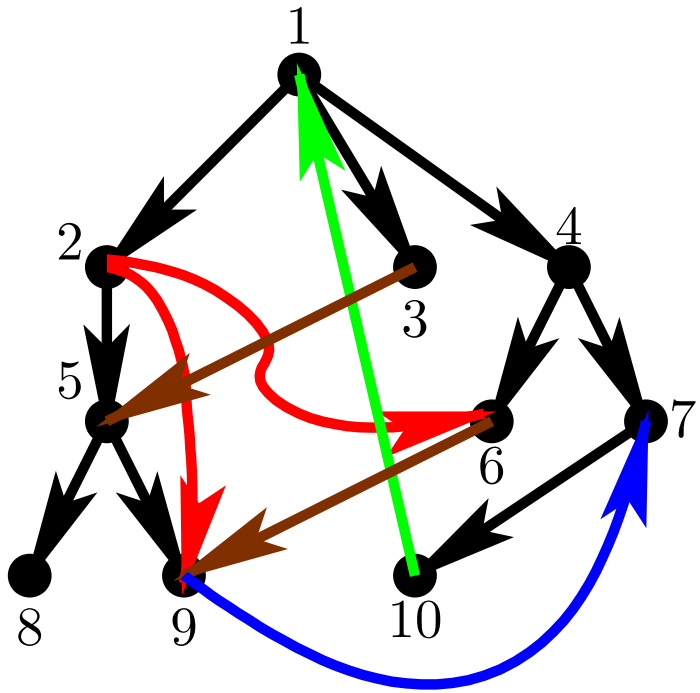
Irányítatlan esetben csak faél és keresztél lehet.



faél

ilyen nincs

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

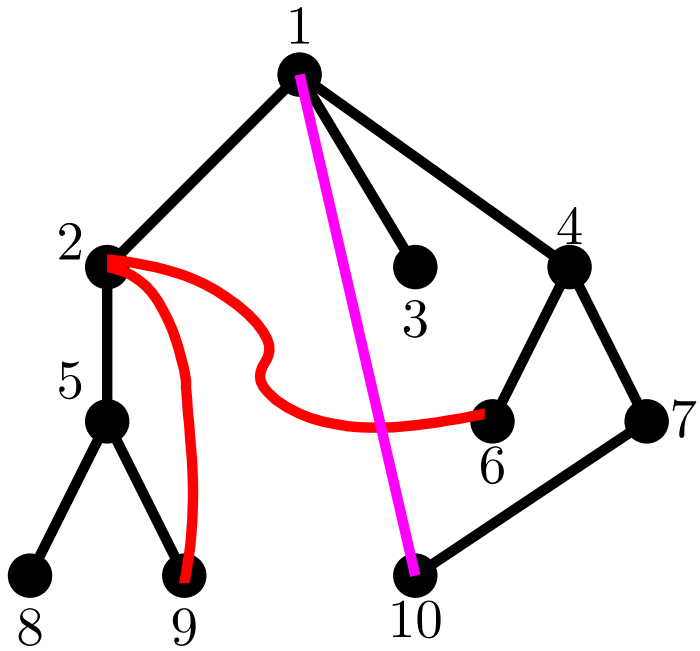
ilyen nincs

visszaél

keresztél

keresztél

Irányítatlan esetben csak faél és keresztél lehet.

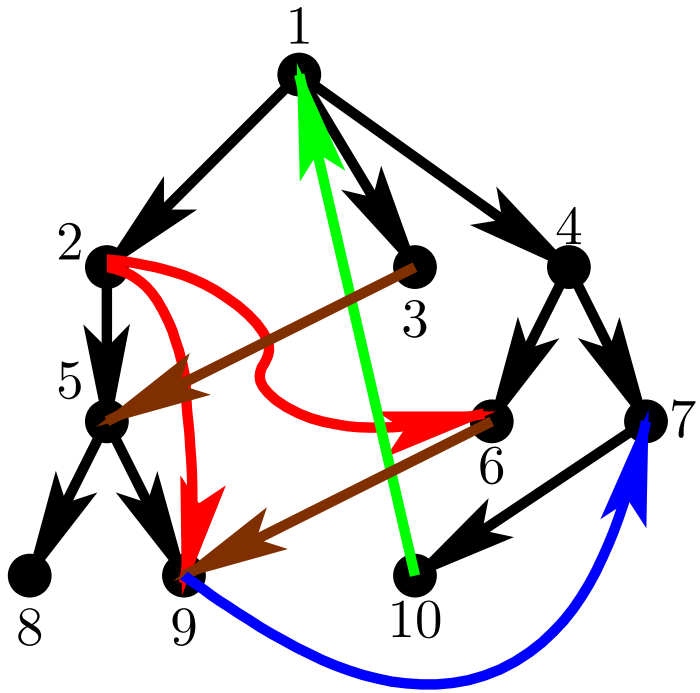


faél

ilyen nincs

ilyen nincs

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél

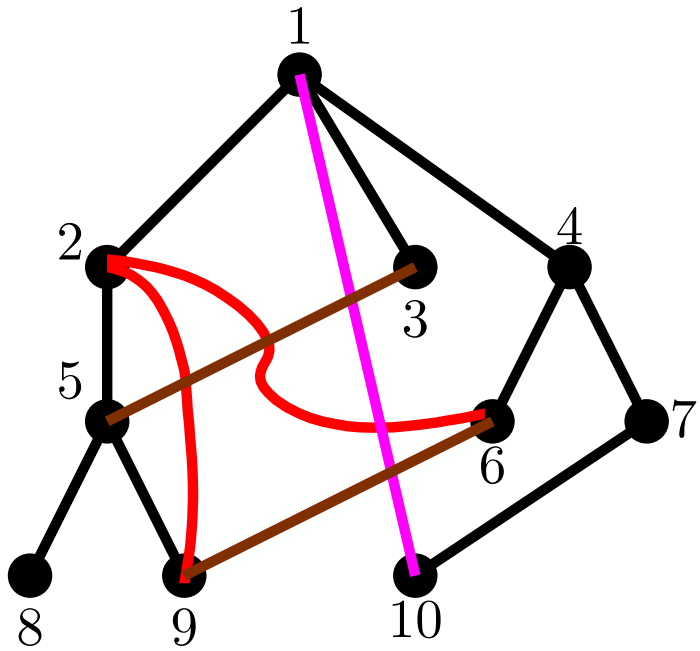
ilyen nincs

visszaél

keresztél

keresztél

Irányítatlan esetben csak faél és keresztél lehet.



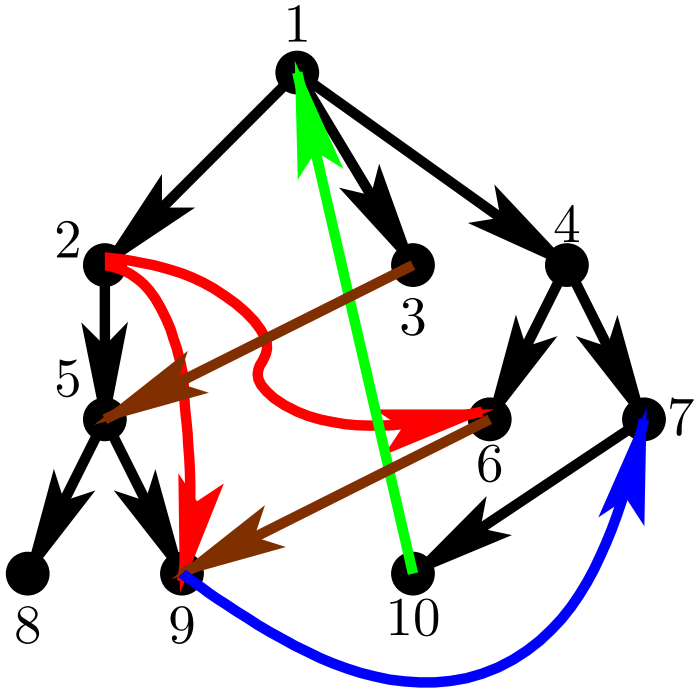
faél

ilyen nincs

ilyen nincs

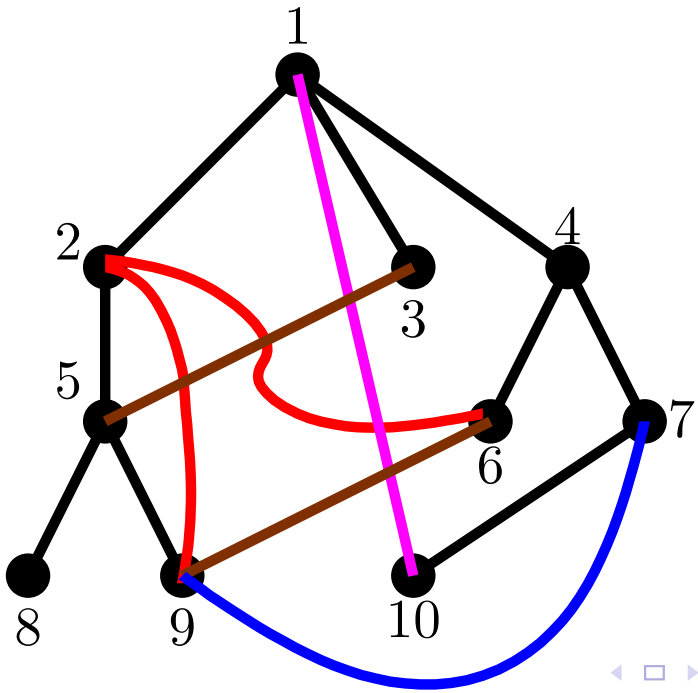
keresztél

Faél: megvizsgálásukkor még be nem járt pontba mutatnak



faél
ilyen nincs
visszaél
keresztél
keresztél

Irányítatlan esetben csak faél és keresztél lehet.



faél
ilyen nincs
ilyen nincs
keresztél
keresztél

Összefüggőség eldöntése

Kérdés

Adott G gráf összefüggő-e?

Összefüggőség eldöntése

Kérdés

Adott G gráf összefüggő-e?

```
procedure Összefüggő
    begin
        öf:=igaz;
        tetszőleges v-re szb(v);
        for u := 1 to n do
            if bejárva[u] = hamis then öf:=hamis;
        return(öf);
    end
```

Összefüggőség eldöntése

Kérdés

Adott G gráf összefüggő-e?

```
procedure Összefüggő
    begin
        öf:=igaz;
        tetszőleges v-re szb(v);
        for u := 1 to n do
            if bejárva[u] = hamis then öf:=hamis;
        return(öf);
    end
```

Összköltség: $O(n + e)$

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Kérdés

Hogyan döntjük el, hogy van-e javítóút?

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Kérdés

Hogyan döntjük el, hogy van-e javítóút?
Hogyan keresünk javítóutat?

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Kérdés

Hogyan döntjük el, hogy van-e javítóút?
Hogyan keresünk javítóutat?

Módosítjuk a szélességi keresést:

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Kérdés

Hogyan döntjük el, hogy van-e javítóút?
Hogyan keresünk javítóutat?

Módosítjuk a szélességi keresést:

- Egy párosítatlan pontból indulunk

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Kérdés

Hogyan döntjük el, hogy van-e javítóút?
Hogyan keresünk javítóutat?

Módosítjuk a szélességi keresést:

- Egy párosítatlan pontból indulunk
- Minden páratlan szinten a BFS lépéseit használjuk.

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Kérdés

Hogyan döntjük el, hogy van-e javítóút?
Hogyan keresünk javítóutat?

Módosítjuk a szélességi keresést:

- Egy párosítatlan pontból indulunk
- Minden páratlan szinten a BFS lépéseit használjuk.
- Minden páros szinten a párosítás éleit használjuk.

Maximális párosítás keresése páros gráfban

BSZ-ből tudjuk, hogy javító utakat kell keresnünk.

Tétel

Egy párosítás akkor és csak akkor maximális, ha nincs hozzá tartozó javítóút.

Kérdés

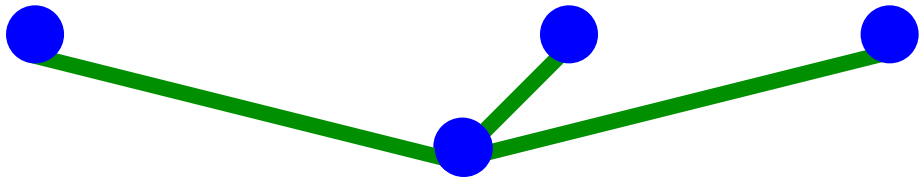
Hogyan döntjük el, hogy van-e javítóút?
Hogyan keresünk javítóutat?

Módosítjuk a szélességi keresést:

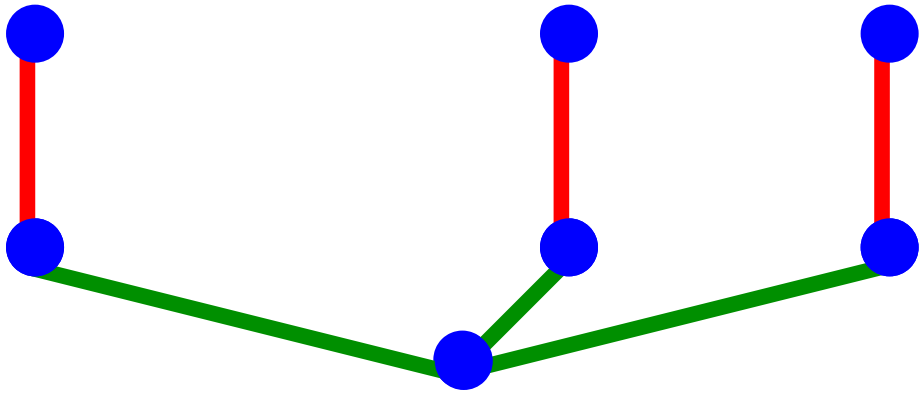
- Egy párosítatlan pontból indulunk
- Minden páratlan szinten a BFS lépéseit használjuk.
- Minden páros szinten a párosítás éleit használjuk.
- Ha olyan pontba érünk, aminek nincs párja, találtunk javítóutat

Maximális párosítás keresése páros gráfban

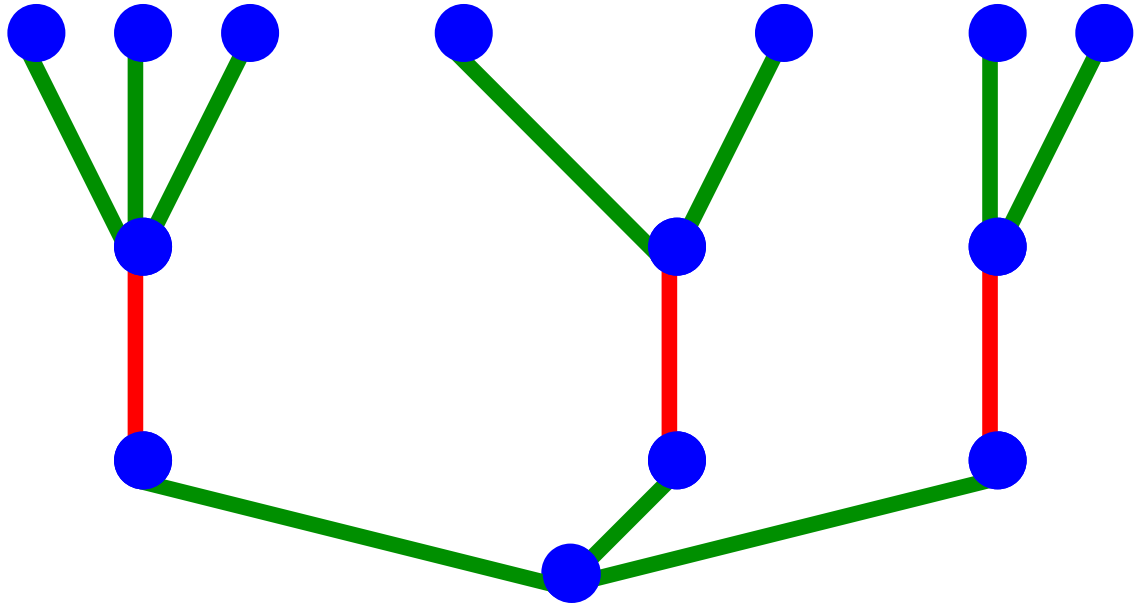
Maximális párosítás keresése páros gráfban



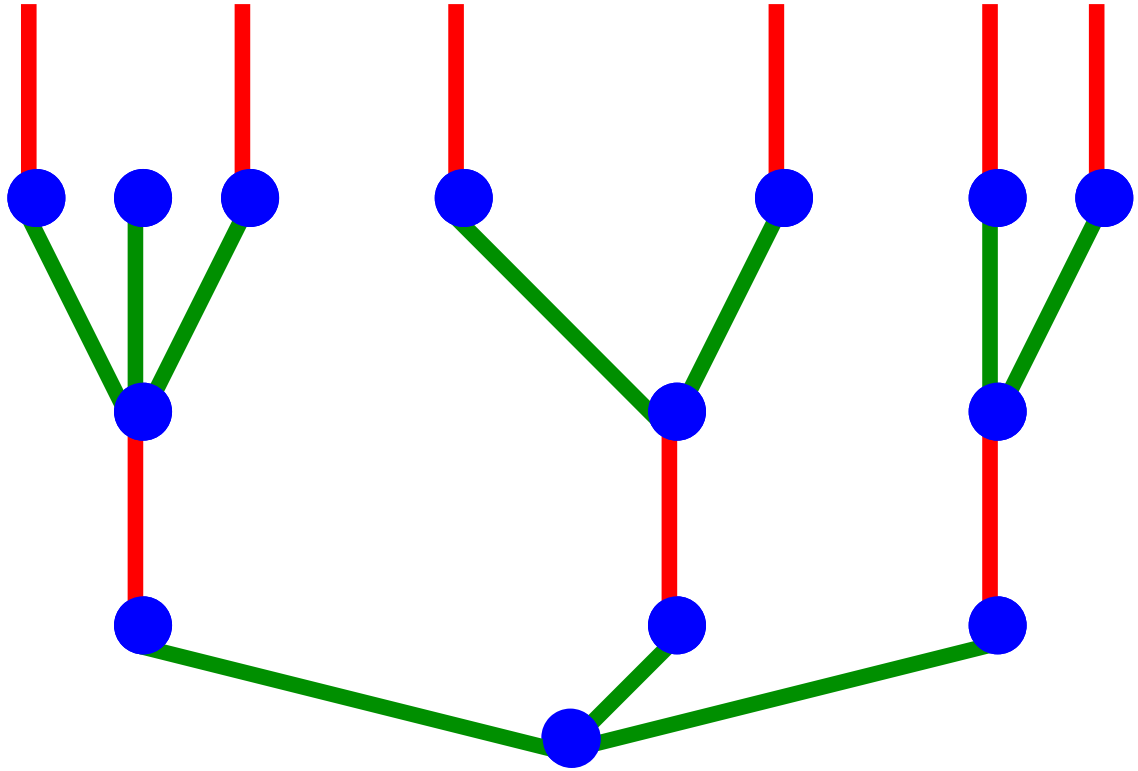
Maximális párosítás keresése páros gráfban



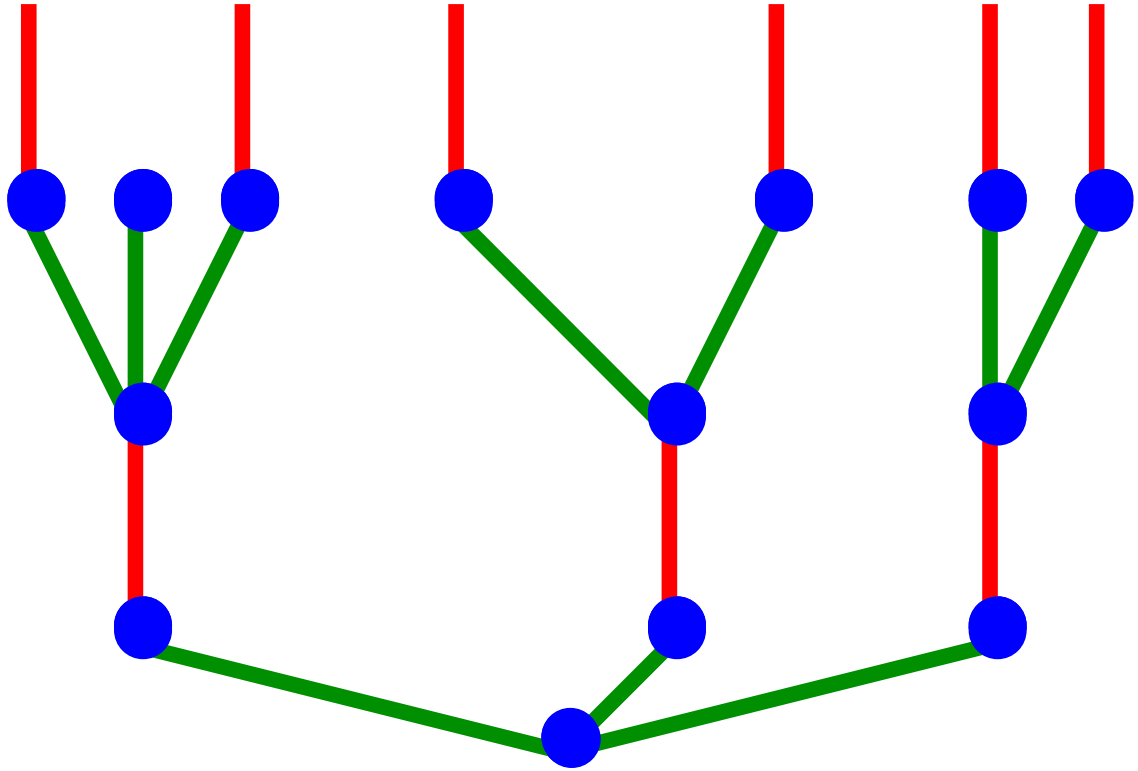
Maximális párosítás keresése páros gráfban



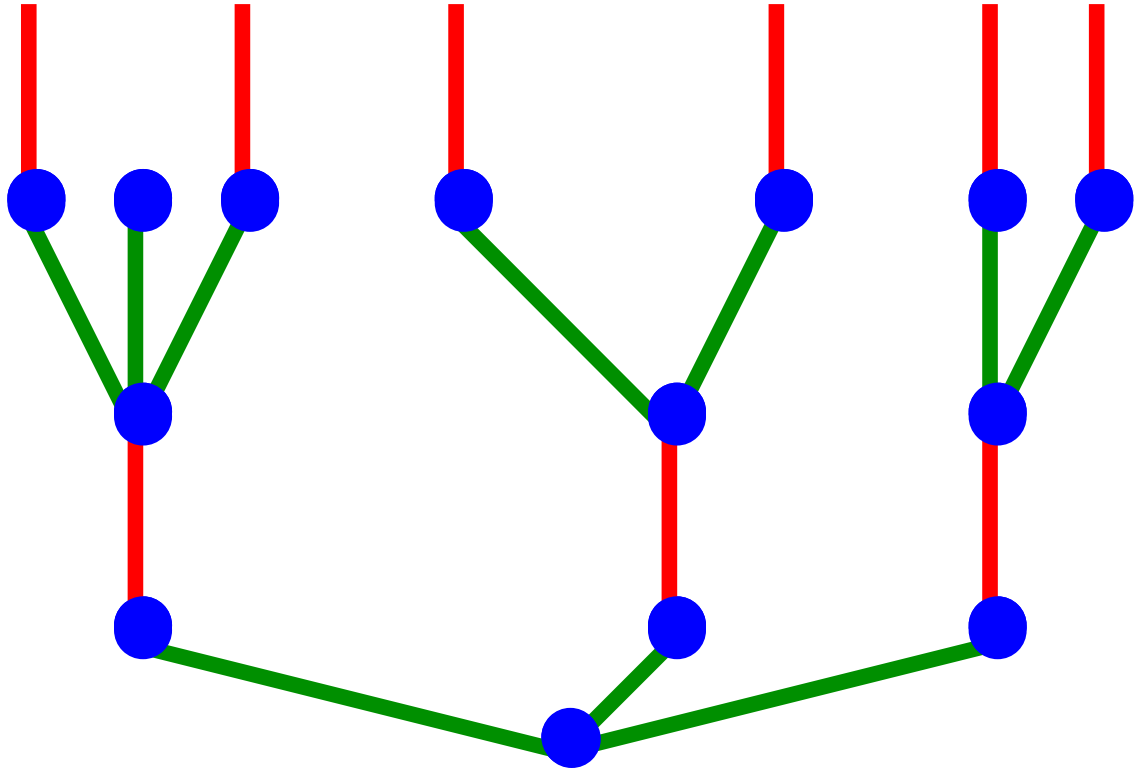
Maximális párosítás keresése páros gráfban



Maximális párosítás keresése páros gráfban



Maximális párosítás keresése páros gráfban

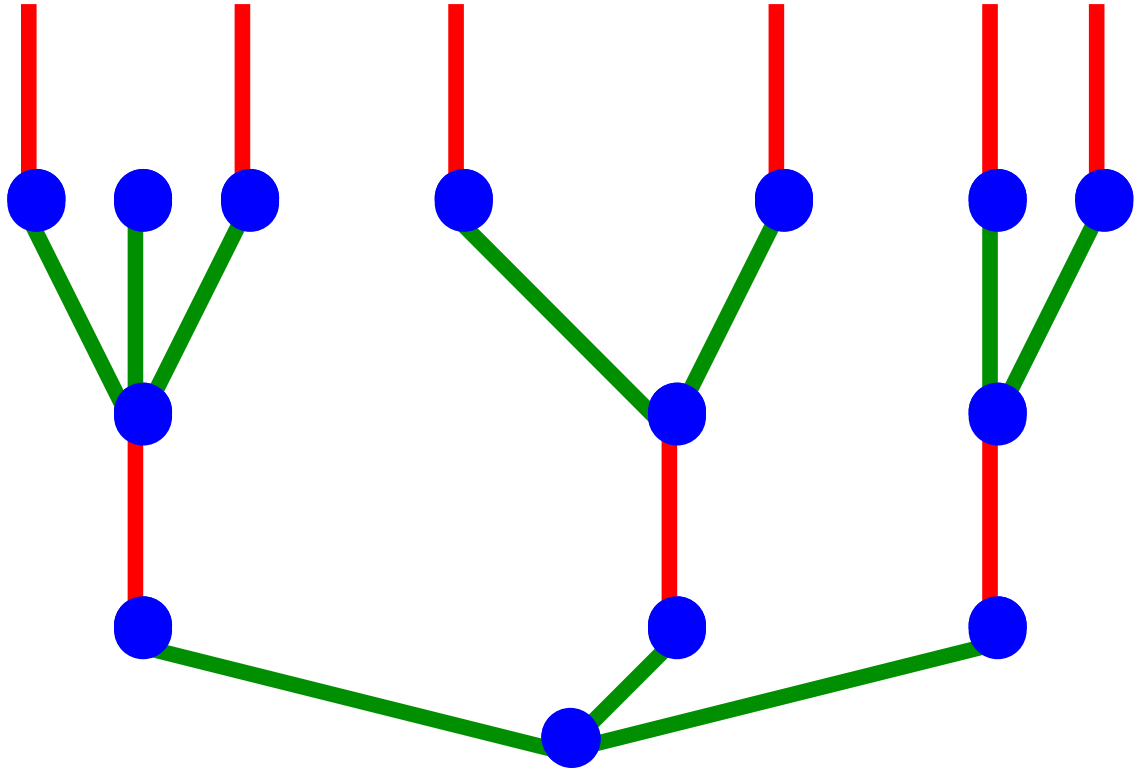


Költség:

Ha minden párosítatlan pontból építünk egy ilyen fát

$\Longrightarrow$ alternáló erdő

Maximális párosítás keresése páros gráfban



Költség:

Ha minden párosítatlan pontból építünk egy ilyen fát

$\Longrightarrow$ alternáló erdő $\Longrightarrow$ $O(e)$

Ezután eggyel növelhetjük a párok számát $\Longrightarrow$ $O(n)$-szer kell ismételni

Összköltség: $O(ne)$

Legrövidebb utak súlyozatlan gráfokban

Ha minden él hossza egy $\Longrightarrow$ út hossza = élek száma

Legrövidebb utak súlyozatlan gráfokban

Ha minden él hossza egy $\Longrightarrow$ út hossza = élek száma
Szélességi kereséssel: Jelentse $D[v]$ a v csúcsnak az s-től való távolságát az s-gyökerű szélességi fában.

Legrövidebb utak súlyozatlan gráfokban

Ha minden él hossza egy $\Longrightarrow$ út hossza = élek száma
Szélességi kereséssel: Jelentse $D[v]$ a v csúcsnak az s-től való távolságát az s-gyökerű szélességi fában.
Legyen kezdetben $D[s] := 0$

Legrövidebb utak súlyozatlan gráfokban

Ha minden él hossza egy $\Longrightarrow$ út hossza = élek száma
Szélességi kereséssel: Jelentse $D[v]$ a v csúcsnak az s-től való távolságát az s-gyökerű szélességi fában.
Legyen kezdetben $D[s] := 0$
az *szb* eljárásba tegyük be a $D[y] := D[x] + 1$; utasítást, miután elértük y-t.

Legrövidebb utak súlyozatlan gráfokban

Ha minden él hossza egy $\Longrightarrow$ út hossza = élek száma
Szélességi kereséssel: Jelentse $D[v]$ a v csúcsnak az s-től való távolságát az s-gyökerű szélességi fában.
Legyen kezdetben $D[s] := 0$
az *szb* eljárásba tegyük be a $D[y] := D[x] + 1$; utasítást, miután elértük y-t.
Lépésszám: $O(n + e)$

Tétel

Az előzőek szerint módosított szélességi bejárás végeztével teljesülnek a következők:
1. Legyen $s = x_1, x_2, \ldots, x_n$ a csúcsoknak a szélességi bejárás szerinti sorrendje. Ekkor $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Legrövidebb utak súlyozatlan gráfokban

Ha minden él hossza egy $\Longrightarrow$ út hossza = élek száma
Szélességi kereséssel: Jelentse $D[v]$ a v csúcsnak az s-től való távolságát az s-gyökerű szélességi fában.
Legyen kezdetben $D[s] := 0$
az *szb* eljárásba tegyük be a $D[y] := D[x] + 1$; utasítást, miután elértük y-t.
Lépésszám: $O(n + e)$

Tétel

Az előzőek szerint módosított szélességi bejárás végeztével teljesülnek a következők:
1. Legyen $s = x_1, x_2, \ldots, x_n$ a csúcsoknak a szélességi bejárás szerinti sorrendje. Ekkor $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.
2. Ha $x \to y$ éle G-nek, akkor $D[y] \leq D[x] + 1$.

Legrövidebb utak súlyozatlan gráfokban

Ha minden él hossza egy $\Longrightarrow$ út hossza = élek száma
Szélességi kereséssel: Jelentse $D[v]$ a v csúcsnak az s-től való távolságát az s-gyökerű szélességi fában.
Legyen kezdetben $D[s] := 0$
az *szb* eljárásba tegyük be a $D[y] := D[x] + 1$; utasítást, miután elértük y-t.
Lépésszám: $O(n + e)$

Tétel

Az előzőek szerint módosított szélességi bejárás végeztével teljesülnek a következők:
1. Legyen $s = x_1, x_2, \ldots, x_n$ a csúcsoknak a szélességi bejárás szerinti sorrendje. Ekkor $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.
2. Ha $x \to y$ éle G-nek, akkor $D[y] \leq D[x] + 1$.
3. $D[v] = d(s, v)$ teljesül minden $v \in V$ csúcsra.

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Bizonyítás.

A csúcsok az $s = x_1, x_2, \ldots, x_n$ sorrendben kerülnek bele a Q sorba $\Longrightarrow$ ebben a sorrendben is kerülnek ki a sorból.

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Bizonyítás.

A csúcsok az $s = x_1, x_2, \ldots, x_n$ sorrendben kerülnek bele a Q sorba $\Longrightarrow$ ebben a sorrendben is kerülnek ki a sorból.
Ha $x \neq s$ csúcs előbb van, mint $y \implies apa(x)$ nem lehet később, mint $apa(y)$, hiszen ha előbb lenne, y-hoz előbb eljutottunk volna.

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Bizonyítás.

A csúcsok az $s = x_1, x_2, \ldots, x_n$ sorrendben kerülnek bele a Q sorba $\Longrightarrow$ ebben a sorrendben is kerülnek ki a sorból.
Ha $x \neq s$ csúcs előbb van, mint $y \implies apa(x)$ nem lehet később, mint $apa(y)$, hiszen ha előbb lenne, y-hoz előbb eljutottunk volna.
Indukció $\Longrightarrow$ Gyökérre $D[s] = 0$, fiaira mind nagyobb. $\surd$

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Bizonyítás.

A csúcsok az $s = x_1, x_2, \ldots, x_n$ sorrendben kerülnek bele a Q sorba $\Longrightarrow$ ebben a sorrendben is kerülnek ki a sorból.
Ha $x \neq s$ csúcs előbb van, mint $y \implies apa(x)$ nem lehet később, mint $apa(y)$, hiszen ha előbb lenne, y-hoz előbb eljutottunk volna.
Indukció $\Longrightarrow$ Gyökérre $D[s] = 0$, fiaira mind nagyobb. $\surd$
$D[x_i] = D[apa(x_i)] + 1$ és $D[x_{i+1}] = D[apa(x_{i+1})] + 1 \implies$

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Bizonyítás.

A csúcsok az $s = x_1, x_2, \ldots, x_n$ sorrendben kerülnek bele a Q sorba $\Longrightarrow$ ebben a sorrendben is kerülnek ki a sorból.
Ha $x \neq s$ csúcs előbb van, mint $y \implies apa(x)$ nem lehet később, mint $apa(y)$, hiszen ha előbb lenne, y-hoz előbb eljutottunk volna.
Indukció $\implies$ Gyökérre $D[s] = 0$, fiaira mind nagyobb. $\surd$
$D[x_i] = D[apa(x_i)] + 1$ és $D[x_{i+1}] = D[apa(x_{i+1})] + 1 \implies$
Ha a két apa különböző
$\implies D[apa(x_i)] \leq D[apa(x_{i+1})] \implies D[x_i] \leq D[x_{i+1}]$.

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Bizonyítás.

A csúcsok az $s = x_1, x_2, \ldots, x_n$ sorrendben kerülnek bele a Q sorba $\Longrightarrow$ ebben a sorrendben is kerülnek ki a sorból.
Ha $x \neq s$ csúcs előbb van, mint $y \implies apa(x)$ nem lehet később, mint $apa(y)$, hiszen ha előbb lenne, y-hoz előbb eljutottunk volna.
Indukció $\implies$ Gyökérre $D[s] = 0$, fiaira mind nagyobb. $\surd$
$D[x_i] = D[apa(x_i)] + 1$ és $D[x_{i+1}] = D[apa(x_{i+1})] + 1 \implies$
Ha a két apa különböző
$\implies D[apa(x_i)] \leq D[apa(x_{i+1})] \implies D[x_i] \leq D[x_{i+1}]$.
Ha pedig az apák megegyeznek $\implies D[x_i] = D[x_{i+1}]$.

Tétel

1. $D[x_1] \leq D[x_2] \leq \ldots \leq D[x_n]$.

Bizonyítás.

A csúcsok az $s = x_1, x_2, \ldots, x_n$ sorrendben kerülnek bele a Q sorba $\Longrightarrow$ ebben a sorrendben is kerülnek ki a sorból.
Ha $x \neq s$ csúcs előbb van, mint $y \implies apa(x)$ nem lehet később, mint $apa(y)$, hiszen ha előbb lenne, y-hoz előbb eljutottunk volna.
Indukció $\implies$ Gyökérre $D[s] = 0$, fiaira mind nagyobb. $\surd$
$D[x_i] = D[apa(x_i)] + 1$ és $D[x_{i+1}] = D[apa(x_{i+1})] + 1 \implies$
Ha a két apa különböző
$\implies D[apa(x_i)] \leq D[apa(x_{i+1})] \implies D[x_i] \leq D[x_{i+1}]$.
Ha pedig az apák megegyeznek $\implies D[x_i] = D[x_{i+1}]$.

$\square$

Tétel

2. Ha $x \to y$ éle G-nek, akkor $D[y] \leq D[x] + 1$

Bizonyítás.

Nézzük, hogy mi történik, amikor x kikerül a Q sorból, és éppen az (x, y) élet vizsgáljuk.

Tétel

2. Ha $x \to y$ éle G-nek, akkor $D[y] \leq D[x] + 1$

Bizonyítás.

Nézzük, hogy mi történik, amikor x kikerül a Q sorból, és éppen az (x, y) élet vizsgáljuk.
Ha bejárva$[y] = \mathit{hamis} \implies y$ apja x, vagyis $D[y] = D[x] + 1$.

Tétel

2. Ha $x \to y$ éle G-nek, akkor $D[y] \leq D[x] + 1$

Bizonyítás.

Nézzük, hogy mi történik, amikor x kikerül a Q sorból, és éppen az (x, y) élet vizsgáljuk.
Ha bejárva$[y] = \mathit{hamis} \implies y$ apja x, vagyis $D[y] = D[x] + 1$.
Különben y-t már korábban láttuk $\implies y$ apja előbb van, mint x

Tétel

2. Ha $x \to y$ éle G-nek, akkor $D[y] \leq D[x] + 1$

Bizonyítás.

Nézzük, hogy mi történik, amikor x kikerül a Q sorból, és éppen az (x, y) élet vizsgáljuk.
Ha bejárva$[y] = \mathit{hamis} \implies y$ apja x, vagyis $D[y] = D[x] + 1$.
Különben y-t már korábban láttuk $\implies y$ apja előbb van, mint x
$\implies D[\mathit{apa}(y)] \leq D[x]$

Tétel

2. Ha $x \to y$ éle G-nek, akkor $D[y] \leq D[x] + 1$

Bizonyítás.

Nézzük, hogy mi történik, amikor x kikerül a Q sorból, és éppen az (x, y) élet vizsgáljuk.
Ha bejárva$[y] = \mathit{hamis} \implies y$ apja x, vagyis $D[y] = D[x] + 1$.
Különben y-t már korábban láttuk $\implies y$ apja előbb van, mint x
$\implies D[\mathit{apa}(y)] \leq D[x] \implies D[y] = D[\mathit{apa}(y)] + 1 \leq D[x] + 1$. □

Tétel

3. $D[v] = d(s, v)$ teljesül minden $v \in V$ csúcsra.

Bizonyítás.

$d(s, v) \leq D[v]$ $\surd$

Tétel

3. $D[v] = d(s, v)$ teljesül minden $v \in V$ csúcsra.

Bizonyítás.

$d(s, v) \leq D[v]$ $\surd$

Legyen $s = y_0, y_1, \ldots y_k = v$ egy minimális hosszúságú G-beli irányított út s-ből v-be.

Tétel

3. $D[v] = d(s, v)$ teljesül minden $v \in V$ csúcsra.

Bizonyítás.

$d(s, v) \leq D[v]$ $\surd$

Legyen $s = y_0, y_1, \ldots y_k = v$ egy minimális hosszúságú G-beli irányított út s-ből v-be.

$\Longrightarrow$ az út éleire: $D[y_1] \leq D[s] + 1 = 1$, majd $D[y_2] \leq D[y_1] + 1 \leq 2$

Tétel

3. $D[v] = d(s, v)$ teljesül minden $v \in V$ csúcsra.

Bizonyítás.

$d(s, v) \leq D[v]$ $\surd$

Legyen $s = y_0, y_1, \ldots y_k = v$ egy minimális hosszúságú G-beli irányított út s-ből v-be.

$\Longrightarrow$ az út éleire: $D[y_1] \leq D[s] + 1 = 1$, majd $D[y_2] \leq D[y_1] + 1 \leq 2 \ldots$

$D[v] = D[y_k] \leq k = d(s, v) \Longrightarrow$ $\surd$

Tétel

3. $D[v] = d(s, v)$ teljesül minden $v \in V$ csúcsra.

Bizonyítás.

$d(s, v) \leq D[v]$ $\surd$

Legyen $s = y_0, y_1, \ldots y_k = v$ egy minimális hosszúságú G-beli irányított út s-ből v-be.

$\Longrightarrow$ az út éleire: $D[y_1] \leq D[s] + 1 = 1$, majd $D[y_2] \leq D[y_1] + 1 \leq 2 \ldots$

$D[v] = D[y_k] \leq k = d(s, v) \Longrightarrow$ $\surd$ $\square$