Kombinatorikus optimalizálás 2. hét

Pusztai Pál
pusztai@sze.hu

Tartalom

- Fák és gráfok bejárása
  - Fák szélességi, mélységi és egyenletes bejárása
  - Gráfok szélességi, mélységi és egyenletes bejárása
  - Útkeresés gráfokban
    - Legrövidebb út adott pontból minden más pontba
    - Legrövidebb út minden pontból minden pontba

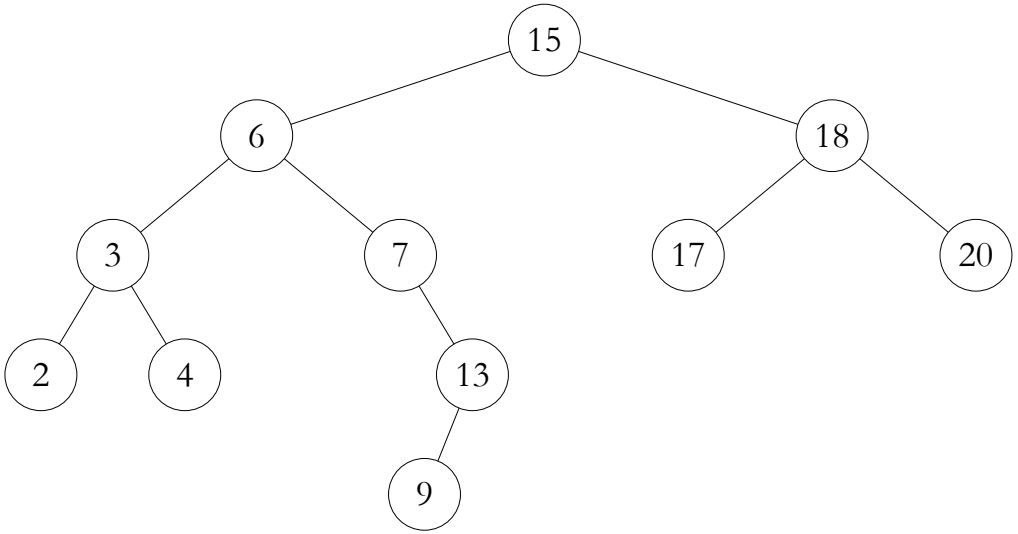
Fák bejárása

- Szélességi bejárás
  - Adatszerkezet
    - Input: a fa a gyökérpontjával azonosítva.
    - Output: a fa, minden pontján az elérésre jellemző távolságokkal.
    - Segédszerkezet: sor.
  - Algoritmus

1. Tegyük a sor végére a gyökérpontot, a távolsága legyen 0.
2. Ha a sor üres, vége.
3. Vegyük ki a sor elejéről a soron következő pontot.
4. A kivett pont minden gyerekét tegyük a sor végére, a távolságuk legyen a kivett pont távolsága +1.
5. Újra a 2. lépés következzen.

Feladatok

- Az alábbi bemenő fa esetén milyen sorrendben veszi ki a szélességi bejárás az egyes pontokat a sorból, ha egy pont gyerekeit balról-jobbra haladva teszi a sorba?
- Milyen lesz a sorból való kivételi sorrend, ha a bejárás egy pont gyerekeit jobbról-balra haladva teszi a sorba?
- Milyen távolság értékeket kapnak az egyes pontok?



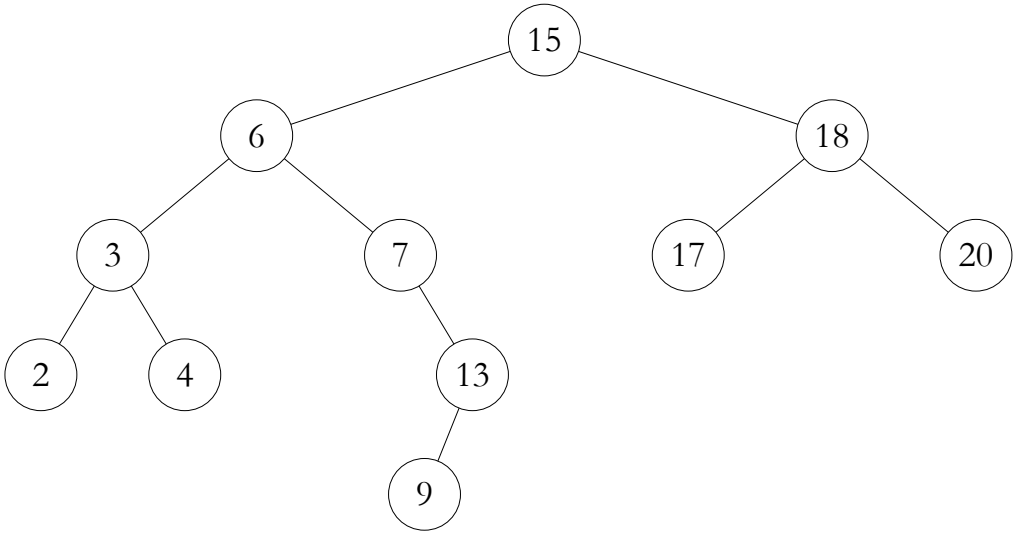
Fák bejárása

- Mélységi bejárás
  - Adatszerkezet
    - Input: a fa a gyökérpontjával azonosítva.
    - Output: a fa, minden pontján az elérésre jellemző távolságokkal.
    - Segédszerkezet: verem.
  - Algoritmus

1. Tegyük a verembe a gyökérpontot, a távolsága legyen 0.
2. Ha a verem üres, vége.
3. Vegyük ki a veremből a soron következő pontot.
4. A kivett pont minden gyerekét tegyük a verembe, a távolságuk legyen a kivett pont távolsága +1.
5. Újra a 2. lépés következzen.

Feladatok

- Az alábbi bemenő fa esetén milyen sorrendben veszi ki a mélységi bejárás az egyes pontokat a veremből, ha egy pont gyerekeit balról-jobbra haladva teszi a verembe?
- Milyen lesz a veremből való kivételi sorrend, ha a bejárás egy pont gyerekeit jobbról-balra haladva teszi a verembe?
- Milyen távolság értékeket kapnak az egyes pontok?



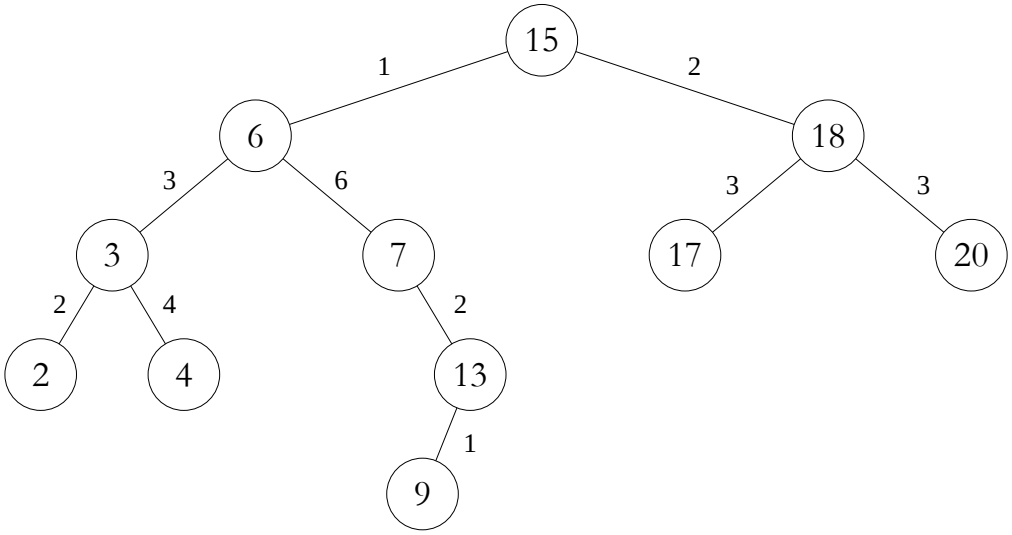
Fák bejárása

- Egyenletes bejárás
  - Adatszerkezet
    - Input: a fa a gyökérpontjával azonosítva.
    - Output: a fa, minden pontján az elérésre jellemző távolságokkal.
    - A fa súlyozott, a súlyok nemnegatívak.
  - Algoritmus

1. A gyökérpontot jelöljük meg, a távolsága legyen 0.
2. Ha nincs jelöletlen pont, vége.
3. Keressük meg azt a jelöletlen pontot, amely a legközelebb van a gyökérponthoz, azaz azt a pontot, amelyhez van jelölt pontból él, és a jelölt pont távolságának valamint az él súlyának az összege minimális.
4. A megtalált pontot jelöljük meg, a távolsága legyen a megtalált minimum.
5. Újra a 2. lépés következzen.

Feladatok

- Az alábbi bemenő fa esetén milyen sorrendben jelöli meg az egyenletes bejárás az egyes csúcsokat, ha az élek súlyait a melléjük írt számok adják?
  - Ha a 3. lépésben több (minimális elérhetőségű) pont létezne, akkor a kisebb (azonosító/kulcs) pontot válasszuk!
- Milyen távolság értékeket kapnak az egyes pontok?
- Végezzük el az előző feladatot úgy is, hogy a fa gyökérpontjának nem a 15-ös pontot, hanem a 7-es pontot tekintjük!



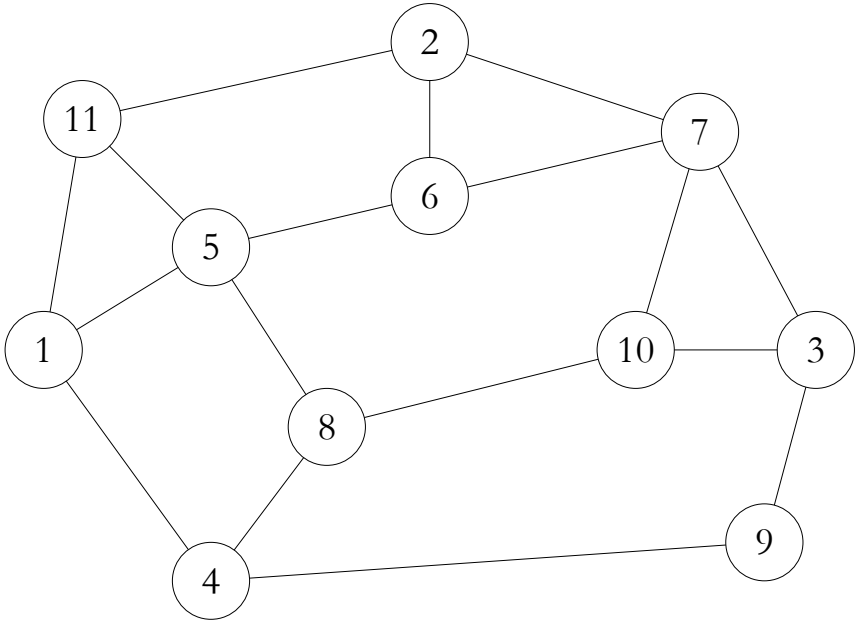
Gráfok bejárása

- Szélességi bejárás
  - Adatszerkezet
    - Input: a gráf, és a bejárás kezdőpontja.
    - Output: a gráf, minden pontján az elérésre jellemző távolságokkal.
    - Segédszerkezet: sor.
  - Algoritmus

1. Tegyük a sor végére a kezdőpontot, jelöljük meg, a távolsága legyen 0.
2. Ha a sor üres, vége.
3. Vegyük ki a sor elejéről a soron következő pontot.
4. A kivett pont minden jelöletlen szomszédját jelöljük meg, tegyük őket a sor végére, a távolságuk legyen a kivett pont távolsága +1.
5. Újra a 2. lépés következzen.

Feladatok

- Az alábbi bemenő gráf esetén milyen sorrendben veszi ki a szélességi bejárás az egyes pontokat a sorból, ha a kezdőpont a 6-os, és egy pont jelöletlen szomszédjait növekvő pontazonosító szerint haladva teszi a sorba?
- Milyen távolság értékeket kapnak az egyes pontok?



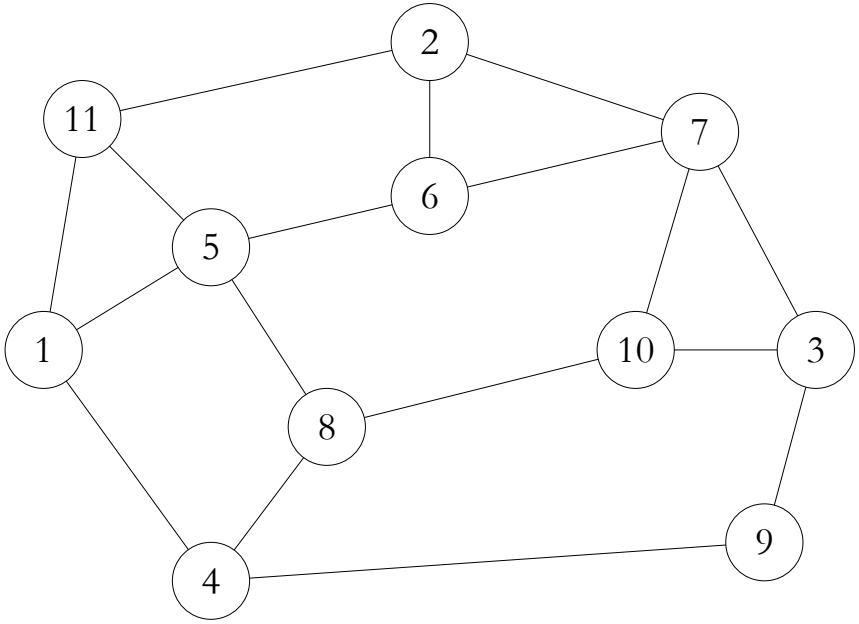
Gráfok bejárása

- Mélységi bejárás
  - Adatszerkezet
    - Input: a gráf, és a bejárás kezdőpontja.
    - Output: a gráf, minden pontján az elérésre jellemző távolságokkal.
    - Segédszerkezet: verem.
  - Algoritmus

1. Tegyük a verembe a kezdőpontot, jelöljük meg, a távolsága legyen 0.
2. Ha a verem üres, vége.
3. Vegyük ki a veremből a soron következő pontot.
4. A kivett pont minden jelöletlen szomszédját jelöljük meg, tegyük őket a verembe, a távolságuk legyen a kivett pont távolsága +1.
5. Újra a 2. lépés következzen.

Feladatok

- Az alábbi bemenő gráf esetén milyen sorrendben veszi ki a mélységi bejárás az egyes pontokat a veremből, ha a kezdőpont a 6-os, és egy pont jelöletlen szomszédjait növekvő pontazonosító szerint haladva teszi a verembe?
- Milyen távolság értékeket kapnak az egyes pontok?



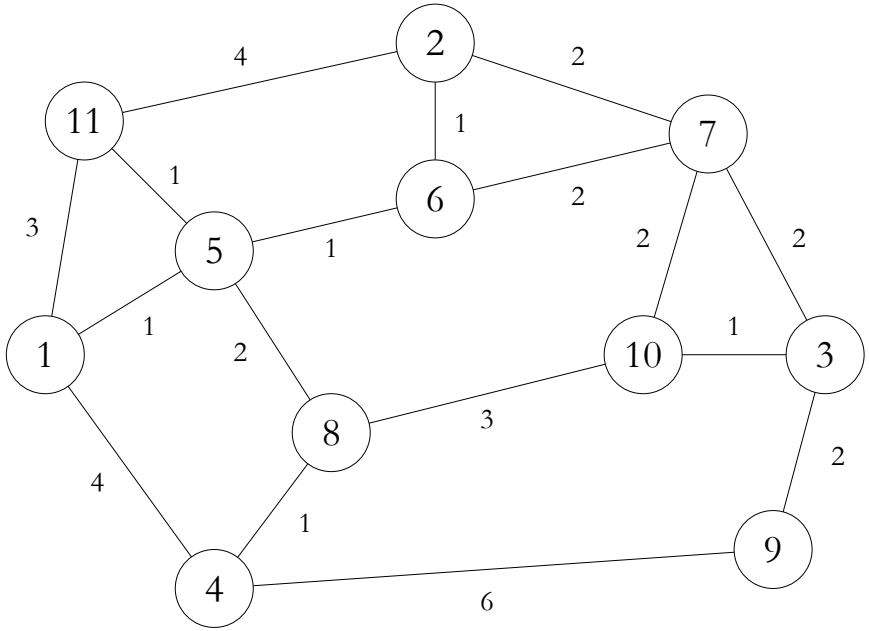
Gráfok bejárása

- Egyenletes bejárás
  - Adatszerkezet
    - Input: a gráf, és a bejárás kezdőpontja.
    - Output: a gráf, minden pontján az elérésre jellemző távolságokkal.
    - A gráf súlyozott, a súlyok nemnegatívak.
  - Algoritmus

1. A kezdőpontot jelöljük meg, a távolsága legyen 0.
2. Ha nincs jelöletlen pont, vége.
3. Keressük meg azt a jelöletlen pontot, amely a legközelebb van a kezdőponthoz, azaz azt a pontot, amelyhez van jelölt pontból él, és a jelölt pont távolságának valamint az él súlyának az összege minimális.
4. A megtalált pontot jelöljük meg, a távolsága legyen a megtalált minimum.
5. Újra a 2. lépés következzen.

Feladatok

- Az alábbi bemenő gráf esetén milyen sorrendben jelöli meg az egyenletes bejárás az egyes csúcsokat, ha a kezdőpont a 6-os, és az élek súlyait a melléjük írt számok adják?
  - Ha a 3. lépésben több (minimális elérhetőségű) pont létezne, akkor a kisebb pontazonosítóval rendelkező pontot válasszuk!
- Milyen távolság értékeket kapnak az egyes pontok?



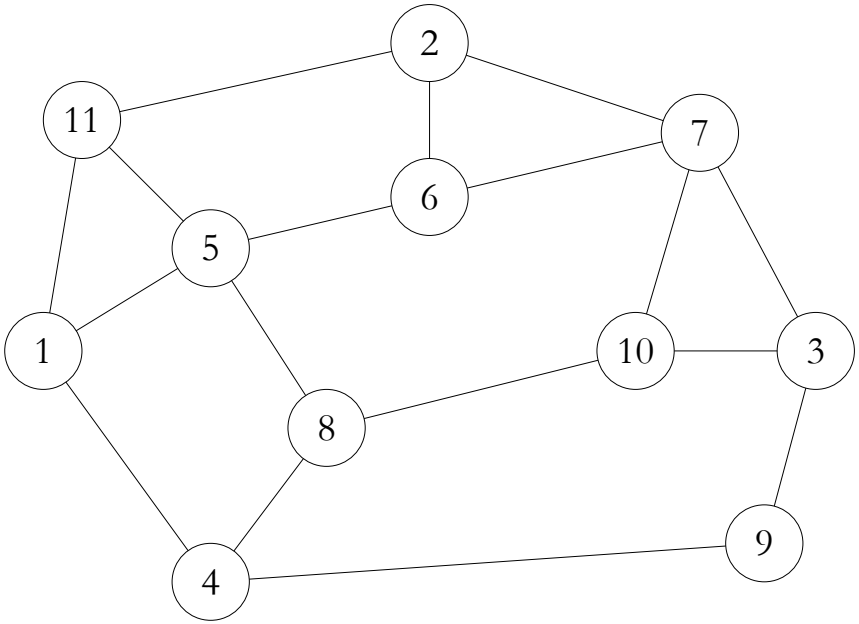
Útkeresés gráfokban

- Adott pontból minden más pontba
  - Adott pontból adott pontba való út megkeresése nem egyszerűbb.
  - A gráfbejáró algoritmusok alkalmasak útkeresésre, ha az elért pontokat megfelelően címkézzük:
    - A kezdőpontot címkézzük meg önmagával.
    - Minden más pont címkéje legyen annak a pontnak az azonosítója, ahonnan a pontot elértük.
  - A megtalált út előállításának algoritmusa:
    - Az algoritmus bemenő paramétere: a végpont azonosítója.
    - Az algoritmus a kezdőpontból a végpontba vezető út pontazonosítóinak listáját adja az érintés sorrendjében.

1. Ha a végpont címkéje nem egyezik meg a végpont azonosítójával, akkor írjuk fel az utat a végpont címkéjéig.
2. Írjuk fel a végpont azonosítóját.

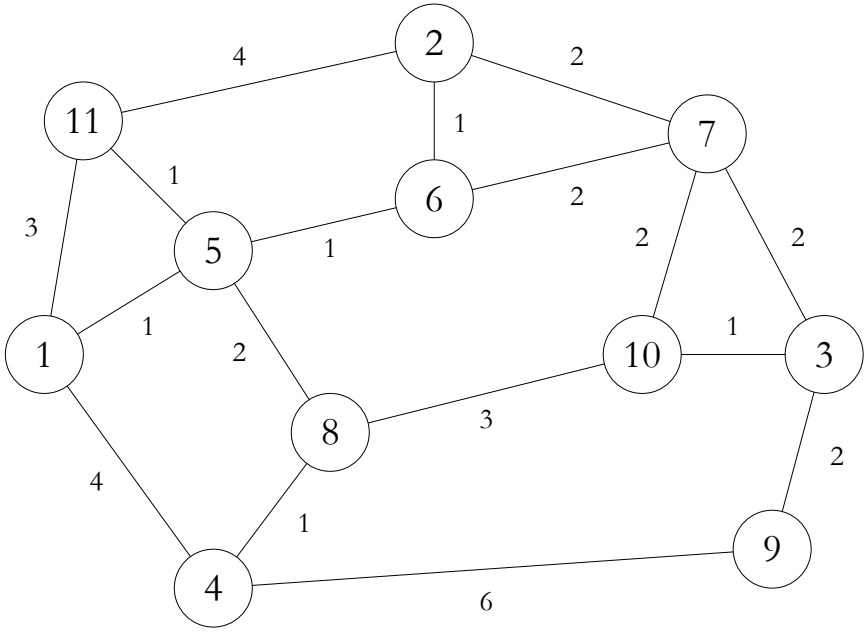
Feladatok

- Milyen címkéket és milyen távolság értékeket kapnak az egyes pontok, ha útkeresésre a gráf szélességi, ill. mélységi bejárását használjuk, és a kezdőpont a 6-os? Egy pont jelöletlen szomszédjait növekvő pontazonosító szerint haladva tegyük a sorba, ill. verembe!



Feladatok

- Milyen címkéket és milyen távolság értékeket kapnak az egyes pontok, ha útkeresésre a gráf egyenletes bejárását használjuk, a kezdőpont a 6-os, és az élek súlyait a melléjük írt számok adják?



Útkeresés gráfokban

- Legrövidebb út adott pontból minden más pontba
  - Adott pontból adott pontba való legrövidebb út megkeresése nem egyszerűbb.
- Kétféleképpen definiálható a legrövidebb út:
  - Legkevesebb élt tartalmazzon.
    - A gráf szélességi bejárása az említett címkézéssel egy lehetséges megoldás.
  - Ha ismerjük az élek hosszát, az út éleinek összhossza a lehető legkisebb legyen.
    - A gráf egyenletes bejárása az említett címkézéssel egy lehetséges megoldás.
    - Megjegyzés: A Dijkstra algoritmus is egyenletes gráfbejárást valósít meg.
- Legrövidebb út minden pontból minden pontba
  - Minden pontra, mint kezdőpontra elvégezzük az előző algoritmust.
  - Mátrixalgoritmus

Útkeresés gráfokban

- Mátrixalgoritmus (Floyd-Warshall)
  - Adatszerkezet
    - Input: a gráf élhosszait tartalmazó **C** $^{(n \times n)}$ mátrix ($c_{i,i}$=0).
    - Output: a legrövidebb utak hosszát tartalmazó **T** $^{(n \times n)}$ mátrix.
  - Algoritmus

1. **T** ← **C**, $l \leftarrow 1$
2. **U** legyen a következő:

   $u_{i,j} \leftarrow \min(t_{i,j}, t_{i,l} + t_{l,j})$ minden $1 <= i <= n$ és $1 <= j <= n$ értékekre.

   T ← **U**
3. $l \leftarrow l + 1$. Ha $l > n$, akkor vége, különben a 2. lépés következzen.

- Megjegyzés
  - Az algoritmus a megfelelő címkézéssel kiegészítve nemcsak a minimális utak hosszát adja meg, de a minimális utak felírását is lehetővé teszi.

Szélességi keresés gráfokban

```
SZK(G, s)
1   for minden u ∈ V[G] −{s} csúcsra
2         szín[u] ← FEHÉR
3         d[u] ← ∞
4         π[u] ← NIL
5   szín[s] ← SZÜRKE
6   d[s] ← 0
7   π[s] ← NIL
8   Q ← Ø
9   SORBA(Q, s)
10  while Q ≠ Ø
11        u ← SORBÓL(Q)
12        for minden v ∈ Adj[u] csúcsra          /* Az u-ból kimenő élek végpontjain */
13              if szín[v] = FEHÉR
14                    szín[v] ← SZÜRKE
15                    d[v] ← d[u] + 1
16                    π[v] ← u
17                    SORBA(Q, v)
18        szín[u] ← FEKETE
```

Hatékonyság: Egy $G=(V, E)$ bemeneti gráf esetén a futási idő $O(V+E)$.

Szélességi keresés gráfokban

```
UTAT-KIÍR(G, s, v)
1   if v = s
2         print s
3   else
4         if π[v] = NIL
5               print ”Nincs út”, s, ”-ből”, v, ”-be”
6         else
7               UTAT-KIÍR(G, s, π[v])
8               print v
```

Feltétel: A π értékeket az SZK eljárással már meghatároztuk és rendelkezésre állnak.

Hatékonyság: Az eljárás futási ideje az úton lévő csúcsok számával arányos, azaz $O(V)$.

Mélységi keresés gráfokban

```
MK(G)
1   for minden u ∈ V[G] csúcsra
2         szín[u] ← FEHÉR
3         π[u] ← NIL
4   idő ← 0
5   for minden u ∈ V[G] csúcsra
6         if szín[u] = FEHÉR
7               MK-BEJÁR(u)

MK-BEJÁR(u)
1   szín[u] ← SZÜRKE                    /* Most érjük el a fehér u csúcsot */
2   idő ← idő + 1
3   d[u] ← idő
4   for minden v ∈ Adj[u] csúcsra       /* Az u-ból kimenő élek végpontjain */
5         if szín[v] = FEHÉR
6               π[v] ← u
7               MK-BEJÁR(v)
8   szín[u] ← FEKETE                    /* u fekete, mert elhagyjuk */
9   f[u] ← idő ← idő + 1
```

Hatékonyság: Egy $G=(V, E)$ bemeneti gráf esetén a futási idő $\Theta(V+E)$.