Adatszerkezetek és algoritmusok

Horváth Géza

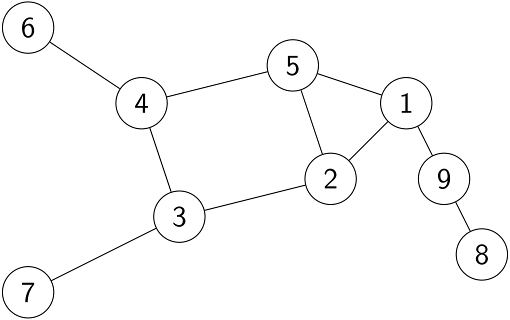
tizedik előadás

Előadások témái

1. Az algoritmusokkal kapcsolatos alapfogalmak bevezetése egyszerű példákon keresztül.
2. Az algoritmusok futási idejének aszimptotikus korlátai.
3. Az adatszerkezetekkel kapcsolatos alapfogalmak. A halmaz, a multihalmaz és a tömb adatszerkezet bemutatása.
4. Az adatszerkezetek folytonos és szétszórt reprezentációja. A verem, a sor és a lista.
5. Táblázatok, önátrendező táblázatok, hash függvények és hash táblák, ütközéskezelés.
6. Fák, bináris fák, bináris keresőfák, bejárás, keresés, beszúrás, törlés.
7. Kiegyensúlyozott bináris keresőfák: AVL fák.
8. Piros-fekete fák.
9. B-fák.
10. Gráfok, bejárás, legrövidebb út megkeresése.
11. Párhuzamos algoritmusok.
12. Eldönthetőség és bonyolultság, a P és az NP problémaosztályok.
13. Lineáris idejű rendezés. Összefoglalás.

Az irányítatlan gráf, mint absztrakt adattípus

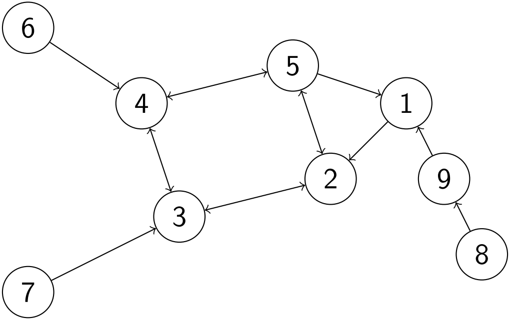
Az **irányítatlan gráf** egyrészt a csúcsok véges halmazából, másrészt az élek véges halmazából áll. A élek halmazát a csúcsok halmazának elemeiből alkotott rendezetlen párok alkotják.



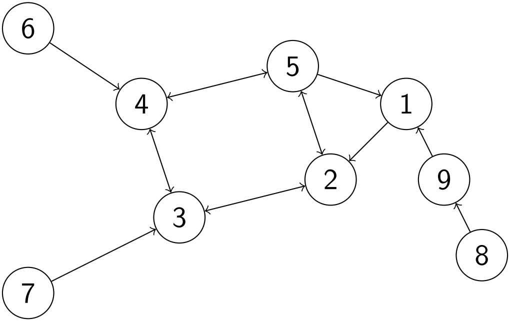
$G = (V, E)$, ahol $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, és
$E = \{(1, 2), (1, 5), (1, 9), (2, 3), (2, 5), (3, 4), (3, 7), (4, 6), (8, 9)\}$.

Az irányított gráf, mint absztrakt adattípus

Az **irányított gráf** esetén az élek halmazát alkotó csúcspárok rendezettek.



Az irányított gráf, mint absztrakt adattípus

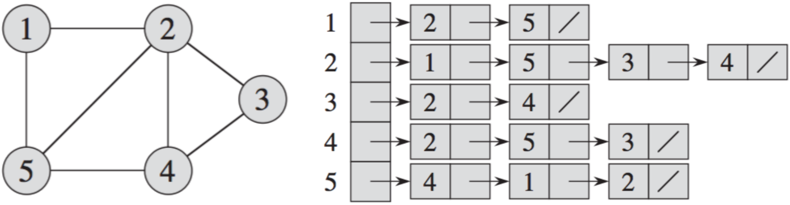


$G = (V, E)$, ahol
$V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, és
$E = \{(1, 2), (2, 3), (2, 5), (3, 2), (3, 4), (4, 3), (4, 5), (5, 1), (5, 2), (5, 4), (6, 4), (7, 3), (8, 9), (9, 1)\}$.

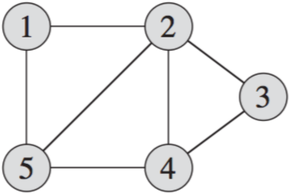
A gráfok főbb alkalmazási területei

- Autóba beépített GPS / Waze / Google Maps / Yahoo Maps esetén legrövidebb út keresése.
- Google search, homepagek megkeresése, ahol a homepagek alkotják a csúcsokat, az egyik homepageről a másikra mutató linkek pedig az éleket.
- Ismerősök hálózata a szociális médiában, ahol az egyének a csúcsok, az élek pedig a kapcsolatok.
- Számítógép-hálózatok esetén a számítógépek a csúcsok, a köztük lévő kapcsolatok pedig az élek.
- Hálós adatbázis-kezelő rendszerek.

Az irányítatlan gráf reprezentációja – szomszédsági lista



Az irányítatlan gráf reprezentációja – szomszédsági mátrix

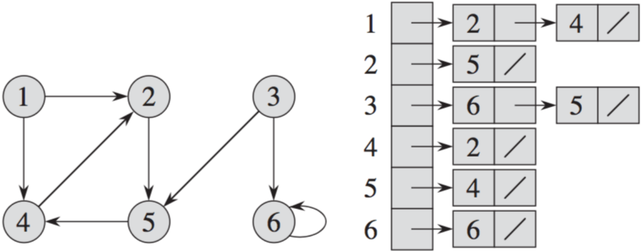


$$\begin{array}{c|ccccc|}
 & 1 & 2 & 3 & 4 & 5 \\ \hline
1 & 0 & 1 & 0 & 0 & 1 \\
2 & 1 & 0 & 1 & 1 & 1 \\
3 & 0 & 1 & 0 & 1 & 0 \\
4 & 0 & 1 & 1 & 0 & 1 \\
5 & 1 & 1 & 0 & 1 & 0 \\ \hline
\end{array}$$

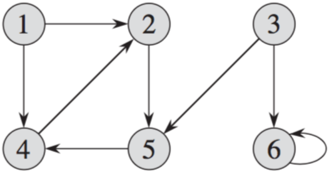
A G ábrázolásához használt $A = (a_{ij})$ csúcsmátrix mérete $|V| \times |V|$, és

$$a_{ij} = \begin{cases} 1, & \text{ha } (i, j) \in E, \\ 0, & \text{különben.} \end{cases}$$

Az irányított gráf reprezentációja – szomszédsági lista



Az irányított gráf reprezentációja – szomszédsági mátrix



	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1

A G ábrázolásához használt $A = (a_{ij})$ csúcsmátrix mérete $|V| \times |V|$, és

$$a_{ij} = \begin{cases} 1, & \text{ha } (i, j) \in E, \\ 0, & \text{különben.} \end{cases}$$

Melyik a jobb?

Melyik a jobb?

1. Hány eleme van?

Melyik a jobb?

1. Hány eleme van?
2. Meddig tart az élek megkeresése?

Melyik a jobb?

1. Hány eleme van?
   - szomszédsági lista: az élek számától függ
   - szomszédsági mátrix: $|V|^2$

Ritka gráfok esetén – amennyiben $|E|$ jóval kevesebb, mint $|V|^2$ – a **szomszédsági lista** használata ajánlott.

Sűrű gráfok esetén – amennyiben $|E|$ számossága közel van $|V|^2$-hez – a **szomszédsági mátrix** használata ajánlott.

2. Meddig tart az élek megkeresése?
   - szomszédsági mátrix: könnyű annak ellenőrzése, hogy két csúcs között vezet-e él
   - szomszédsági lista: könnyű végigmenni az egy adott csúcshoz tartozó összes élen

Szélességi bejárás és keresés

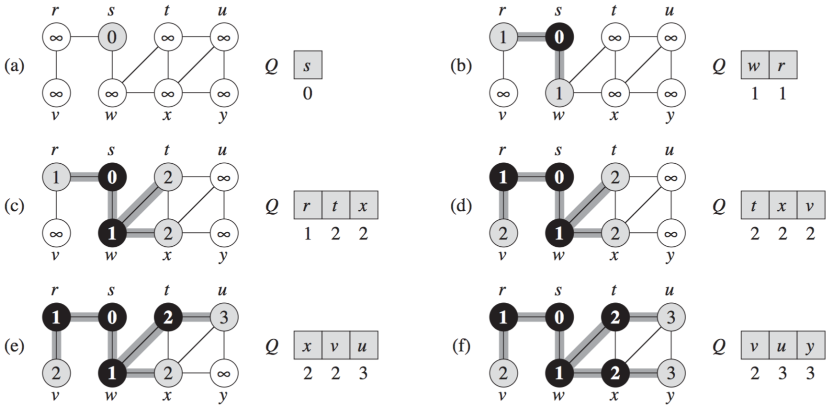
A szélességi bejárás és keresés használata esetén a csúcsok a következő árnyalatúak lehetnek:

- fehér,
- szürke,
- fekete.

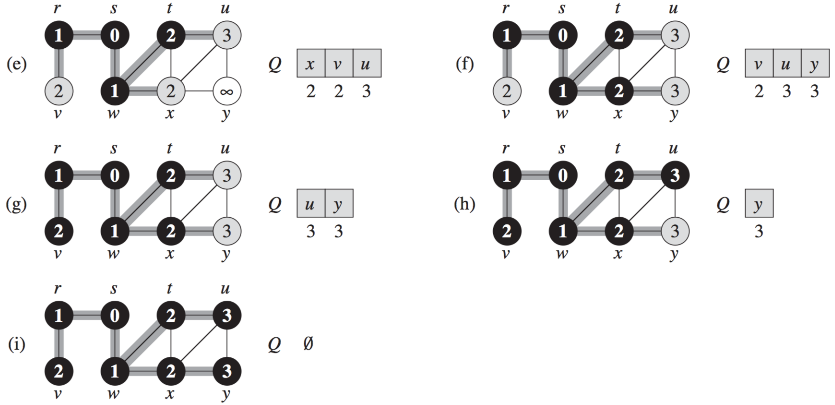
Az algoritmus használ továbbá egy **sor** adatszerkezetet (Q), mely a szürke csúcsokat tartalmazza.

Az $u.d$ érték az u csúcs és a kiinduló s csúcs távolságát jelöli.

Szélességi bejárás és keresés



Szélességi bejárás és keresés



Szélességi bejárás és keresés

```
BFS(G, s)
for each vertex u ∈ G.V − {s}
    u.color = WHITE
    u.d = ∞
    u.π = NIL
s.color = GRAY
s.d = 0
s.π = NIL
Q = ∅
ENQUEUE(Q, s)
while Q ≠ ∅
    u = DEQUEUE(Q)
    for each v ∈ G.Adj[u]
        if v.color == WHITE
            v.color = GRAY
            v.d = u.d + 1
            v.π = u
            ENQUEUE(Q, v)
    u.color = BLACK
```

A kezdeti beállításokhoz $\mathcal{O}(|V|)$ lépés szükséges, míg maga a keresés $\mathcal{O}(|E|)$ lépést tesz meg, így a teljes algoritmus futásához szükséges idő $\mathcal{O}(|V| + |E|)$. Ebből látható, hogy a szélességi bejárás és keresés futási ideje egy tetszőleges G gráf esetén a hozzá tartozó szomszédsági lista méretével egyenesen arányos, azaz a futási idő lineáris.

Mélységi bejárás és keresés

Mélységi kereséshez nem a könyvet használjuk!

A mélységi bejárás és keresés használata esetén a csúcsok a következő árnyalatúak lehetnek:

- fehér,
- szürke,
- fekete.

A mélységi bejáráó és kereső algoritmus használ továbbá egy **verem** adatszerkezetet (S), mely a szürke csúcsokat tartalmazza.

Mélységi bejárás és keresés

```
DFS(G, s)
for each vertex u ∈ G.V − {s}
    u.color = WHITE
    u.d = ∞
    u.π = NIL
s.color = GRAY
s.d = 0
s.π = NIL
S = Ø
PUSH(S,s)
while S ≠ Ø
    u = POP(S)
    for each v ∈ G.Adj[u]
        if v.color == WHITE
            v.color = GRAY
            v.d = u.d + 1
            v.π = u
            PUSH(S,v)
    u.color = BLACK
```

A mélységi bejárás és keresés teljes futási ideje szintén $\mathcal{O}(|V| + |E|)$.

Lássuk a példát a táblán!

Irodalomjegyzék