Gráfelméleti alapok:

- **Összefoglaló** áttekintés:
  - *Szükséges* **előzetes ismeretek**:
    - A mobilrobot munkaterületek felosztása, majd a gráftérképeinek (topológiai) elkészítése
    - Legfontosabb gráfútvonalak algoritmusai (**graph**, **visibility**, **tangent-visibility**, **MakLink**, **Voronoi. Véletlenszerű - PPL**)
- **Gráf-, fa- keresési algoritmusokkal** kapcsolatos alapdefiníciók.
- **Bejárási stratégiák**
  - **Szélességi** keresési stratégia
  - **Mélységi** keresési stratégia
- **Legrövidebb utak egy forrásból**
  - **Dijsktra** algoritmus
  - **Bellman-Ford** algoritmus
- **Legrövidebb utak minden csúcspárra**
  - **Floyd** algoritmus
  - **Tranzitív lezárt**
  - **Warshall** algoritmus

Külön tárgyalva

Útvonalkereső algoritmusok

- **Bevezető alapfogalmak:**
  - Az útvonalkereső algoritmusok célja, hogy egy térképen megtalálják a legrövidebb útvonalat két pont között. A feladat bonyolódik, ha a térképen akadályok is vannak. Legrövidebb útból több is lehet; az algoritmusok célja, hogy ezekből egyet megtaláljanak.
  - Az útvonalkeresőnek fel kell derítenie a térképet, hogy megtalálja a legrövidebb utat, ehhez viszont előbb meg kell értenie magát a térképet. Ehhez a felderítendő térképet le kell egyszerűsíteni egy, a számítógép számára is megérthető **gráfra**.
  - **A gráfok olyan absztrakt matematikai modellek, amik csomópontokból (node vagy vertex) és az azokat összekötő élekből (edge vagy link) állnak; a modell párokat kapcsol össze és meghatározza a köztük lévő függést. Megkülönböztetünk irányított és nem irányított / súlyozott és nem súlyozott élekkel,** illetve **csomópontokkal gráfokat.**
  - Útvonalkeresésnél egy térképen a csomópontok az érinthető helyek, az élek pedig az őket összekötő út. Minél kevesebb csomópont van egy térképen, annál gyorsabb lesz a keresőalgoritmus.
  - Keresés közben számon tartjuk a **peremet** (frontier), ami folyamatosan tágul, ahogy a keresőalgoritmus deríti fel a csomópontokat, a **nyílt halmazt**, amibe a **még fel nem derített** csomópontok vannak, és gráfok esetén a **zárt halmazt**, ami a már **felderített** csomópontokat tartalmazza.
  - Az útvonalkeresésben általában **súlyozott gráfokat** használunk, azaz a gráf csomópontjaihoz és/vagy éleihez súly (költség) értékeket rendelünk.

Gráf-kereső, Fa- kereső algoritmusok

- A **gráfbejáró algoritmusok** bejárják a csomópontokat, egy-egy pontot akár többször is érintve (redundancia), ezért szükség van arra, hogy a már érintett pontokat számon tartsuk (**zárt halmaz**); ha nem is azért, hogy ne érintsük őket többször, akkor azért, hogy minimalizáljuk a többszörös áthaladást, különben az algoritmus soha nem fejeződik be, végtelen ciklusba torkollhat.
- A gráfbejárás speciális esete a **fabejárás**, ahol bármelyik két csomópont között csak egy él van (minden csomópontba csak egy úton lehet eljutni), nincsenek rajta hurkok, és egy pontot csak egyszer érint a keresés közben. A **fabejárásnál nincs zárt halmaz**, és ezért végtelen ciklusba eshet, ha gráfon alkalmazzuk.

Megjegyzés: *Az útvonalkeresésnél úgy kapjuk meg a tényleges utat, ha a célból visszafele haladva a szülő csomópontokat végig járva rögzítjük a célhoz vezető lépéseket; egyébként az algoritmus csak annyit ad vissza, hogy talált egy utat a sok közül.*

Gráf típusok

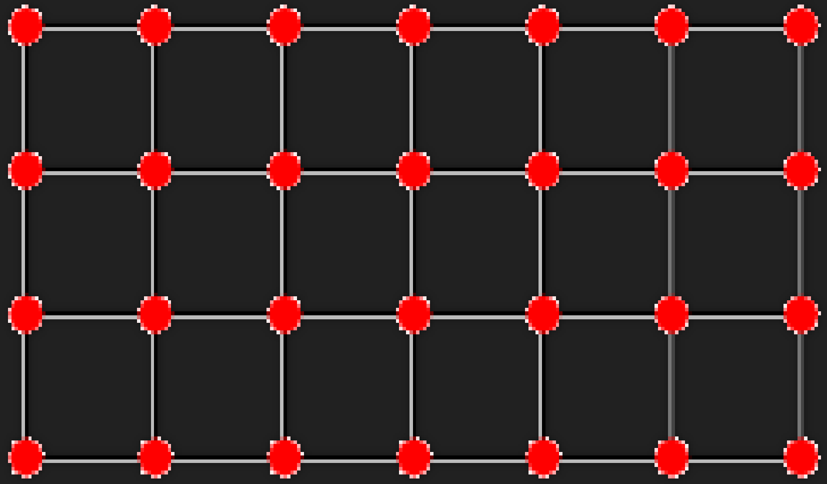
- Alapjában a gráfokat feloszthatjuk **irányított** (nyilak jelzik az irányokat, egy irányban járhatók) és **nem irányított** gráfokra. Mind ezek mellett a területfelosztás szempontjából lehetnek:
  - Rács
  - Poligon
  - Fa szerkezetű, hierarchikus

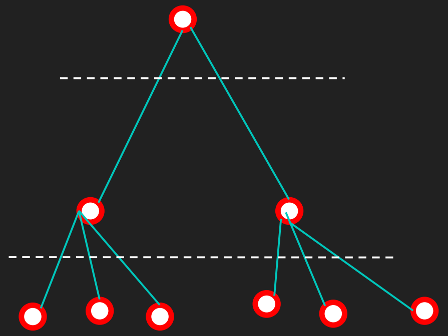
gráfok.

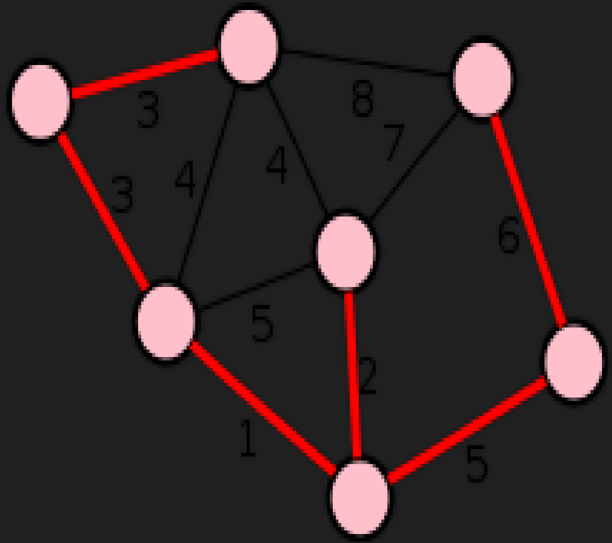
Rácsháló

Fa(hierarchikus) típusú gráfok

Poligon típusú







Informáltalan és informált (heurisztikus) típusú gráfok

- Informálatlan keresés esetén az algoritmusoknak **nincs** semmilyen extra **információja** a **problémáról** a probléma definiálásán kívül, nem tudja, hogy az aktuális csomópont milyen messze van a céltól; emiatt hatásfokuk sem megfelelő.

- Az informált keresőalgoritmusok rendelkeznek **probléma-specifikus információval**, intuícióval, becsléssel arról, hogy a probléma megoldását merre is kell keresni. Egyszerű példa városok közötti utak esetén heurisztikának használni a légvonalbeli távolságot.

Megjegyzés: Ügyelni kell arra, hogy a költségfüggvényhez használt valós költség és heurisztikus becslés értékei ugyanabban a mértékben legyen megadva, pl. távolság vagy idő.

Alapfogalmak, jelölések

Irányított gráf: $G=(V,E)$ pár, ahol V a csúcsok véges halmaza (általában $1,2,3,\ldots,n$), $E \in (V \times V)$ pedig az élek halmaza.

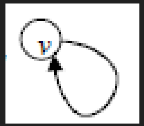
Egy él: $e=(u,v) \in E$, ahol u,v csúcspárok $\in V$ – *sorrend fontos az **irányított** gráfoknál!*

Nem irányított gráfokra érvényes: $G=(V,E)$, ahol érvényes: $(u,v) \in E = (v,u) \in E$

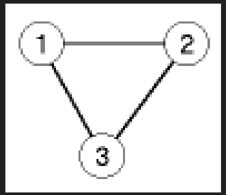
Szomszéd: v az u –nak szomszédja, rákövetkezője: $u \rightarrow v$ 

Út: $\langle v_0, v_1, \ldots, v_k \rangle$, k hosszúságú út, ahol $\forall_i \in [1..k]$: $(v_{i-1}, v_i) \in E$, jelölése: $v_0 \sim\triangleright v_k$

Kör: egy k hosszúságú *út* = *kör,* ha $v_0=v_k$



k=1, irányított hurokél



k=3, irányítás nélküli kör

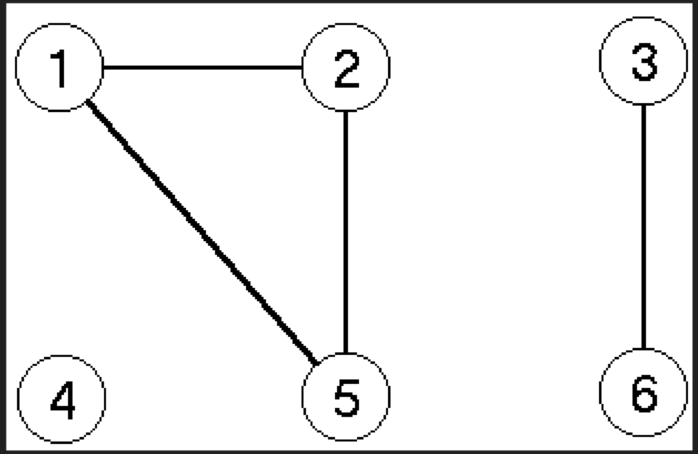
Alapfogalmak, jelölések

Fokszám:

- **Irányítás nélküli** gráf fokszáma a **csúcsból kiinduló élek száma**
- **Irányított gráfnál** megkülönböztetjük a **befokot** (bemenő élek száma) és a **kifokot** (kimenő élek száma).

Ekkor a **fokszámot ezek összege adja**.

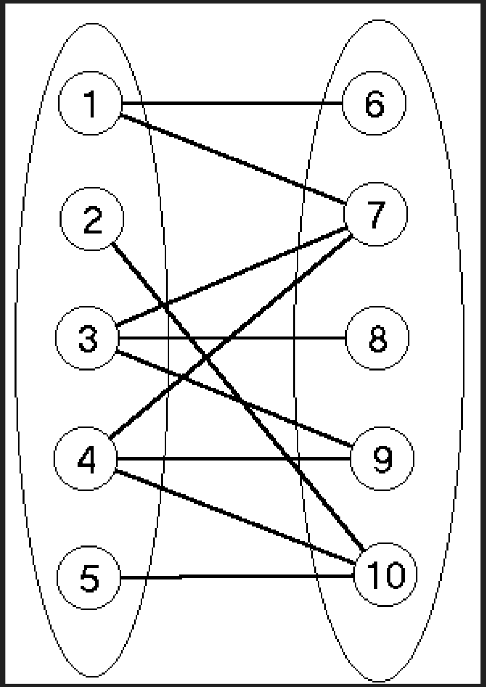
Összefüggőség: Az összekötött csúcsok.
A lenti gráf összefüggő komponensei: {1,2,5}; {3,6}; {4}



Alapfogalmak, jelölések

Teljes gráf: Olyan irányítás nélküli gráf, amelynek bármely két csúcsa szomszédos

Páros gráf: Olyan irányítás nélküli gráf, amelynek csúcsai két, diszjunkt halmazra bonthatók,
és él csak a két különböző halmaz csúcsai között mehet, azonos halmazban lévő csúcsok között nem.



Erdő: körmentes, irányítás nélküli gráf.

Fa: Összefüggő, körmentes, irányítás nélküli gráf.

Élsúlyok: $c : E \to R$ egy valós értékű függvény, amelynek értelmezési tartomány a gráf élhalmaza.

Gráfok ábrázolása

A gráfok ábrázolására **két**, a gyakorlatban elterjedt adatszerkezet ismert:

1. **aritmetikai ábrázolású** (szomszédsági-mátrix; adjacencia-mátrix, csúcsmátrix),
a másik vegyes,

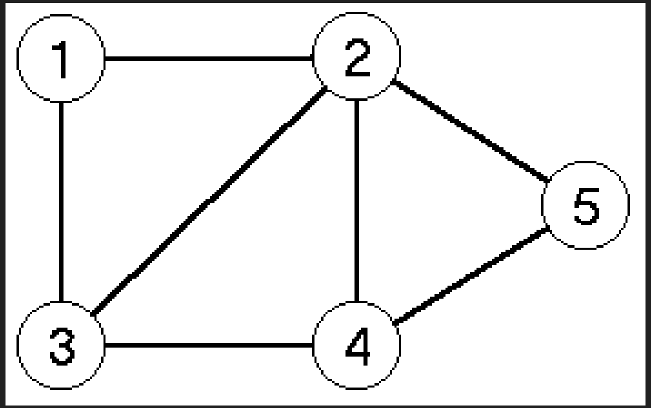
2. **aritmetikai és láncolt ábrázolású** (éllista).

Adjacencia mátrixok

Legyen *G=(V,E)* véges gráf, és *n* a csúcsok száma. Ekkor a gráfot egy *n x n* -es mátrixban ábrázoljuk, ahol az oszlopokat és a sorokat rendre a csúcsokkal indexeljük (ez leggyakrabban *1,..,n*). Egy mezőben akkor van *1*-es, ha a hozzá tartozó oszlop által meghatározott csúcs szomszédja a sor által meghatározott csúcsnak.

<u>Matematikailag</u>:

$$A[i,j] = \begin{cases} 0, ha\ (i,j)\ nem\ eleme\ E \\ 1, \quad\quad ha\ (i,j) \in E \end{cases}$$

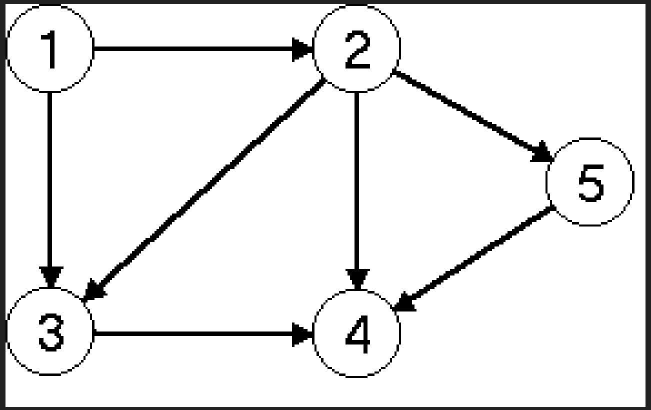


	1	2	3	4	5
1	0	1	1	0	0
2	1	0	1	1	1
3	1	1	0	1	0
4	0	1	1	0	1
5	0	1	0	1	0

Nem irányított gráf

Adjacencia mátrixok

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



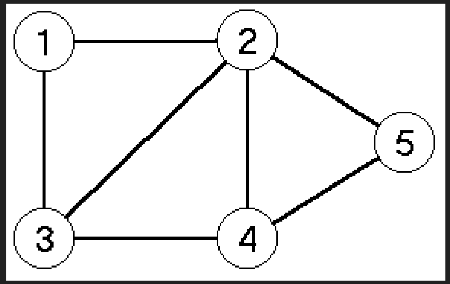
Irányított gráf

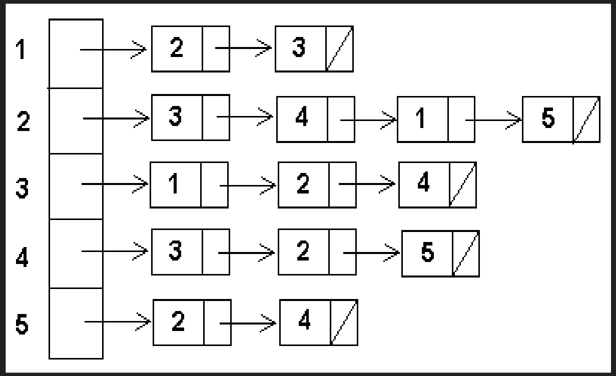
Ha a gráf súlyozott akkor az „1” helyett az élek súlyai vannak beírva

Éllisták (Szomszédsági lista)

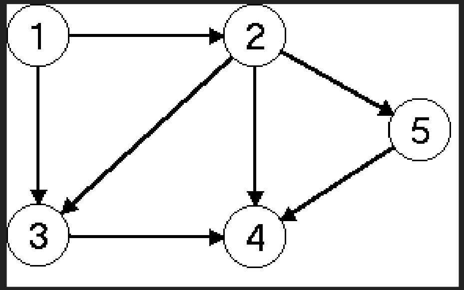
A Gráf minden csúcsához egy listát rendelünk. Ezen listában tartjuk nyilván az adott csúcsból kimenő éleket. Vegyünk fel, egy mutatókat tartalmazó *Adj[1..n]* tömböt (az „n” csúcsokkal indexeljük a tömböt). A tömbben lévő mutatók mutatnak az éllistákra.

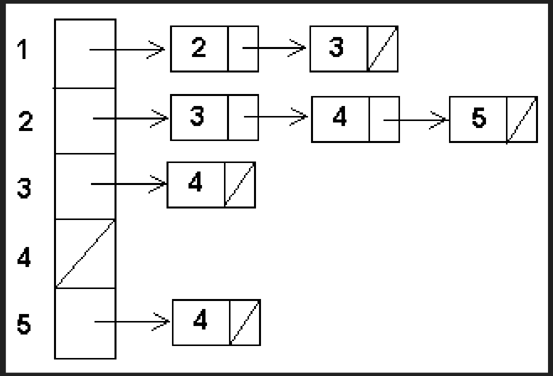
Nem irányított gráf esetén





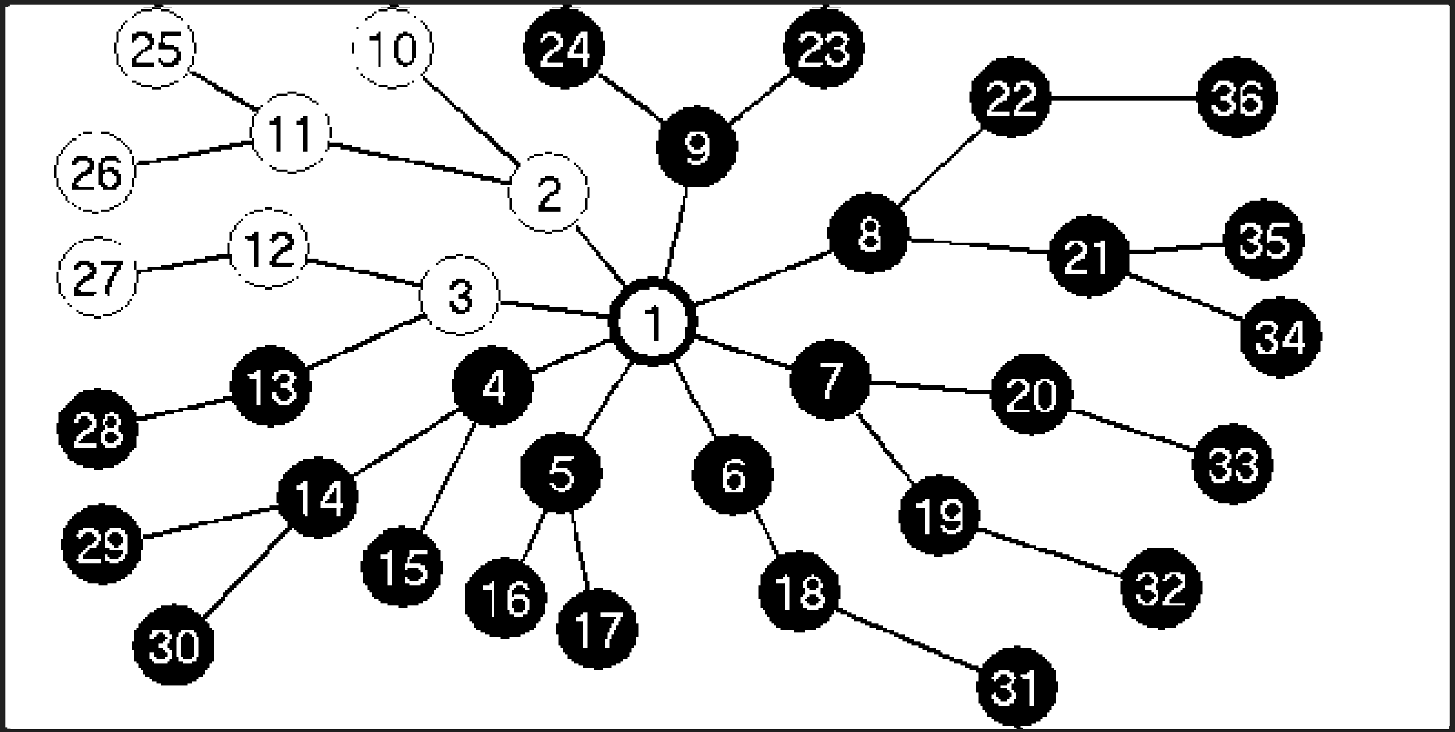
Irányított gráf esetén





Mélységi bejárás

Egy ágon halad végig, majd mikor a végére ért visszamegy a következő csomópontig és az abból leágazó ágon halad végig. Ha nincs már bejáratlan leágazás, akkor még halad visszafelé, addig a csomópontig, amíg bejáratlan ágat talál. (végül, a kezdőpontban végzi)



Szélességi bejárás

Kezdőpontból indul, egyszerre minden leágazásba, és a következő csomópontban megint szétoszlik és minden elágazásba (irányba) elindul,… egész a leágazás nélküli csomópontig.