4. Legrövidebb utakat kereső algoritmusok

BFS

* Breadth fisrt search szóból ered, szélességi keresés másnéven
* lehet irányított vagy irányítatlan, de általában irányított gráfról beszélünk
* Input: G(V,E), sV
* Output: megadja a legrövidebb utat s-ből a többi pontba
* Működése:
  + 0. lépés: s=0
  + 1. lépés: s szomszédos csúcsi (v0,v1,..vk)->1, tehát i=1
  + 2. lépés: a legutóbb címkézett csúcsok még bejáratlan szomszédos csúcsait is felcímkézzük i=i+1, ezt addig ismételjük, amíg van bejáratlan csúcs, így kaphatjuk meg a legrövidebb utak fáját
* Helyessége

Dijkstra

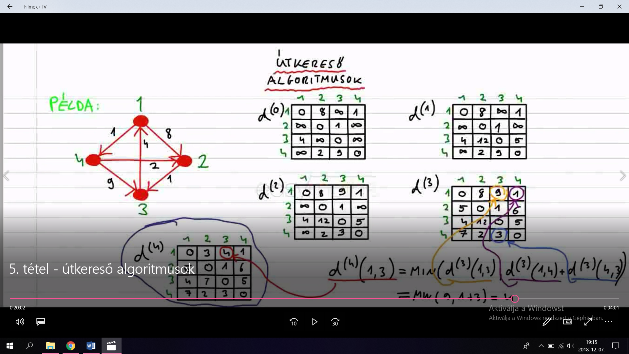
* hasonló, mint a BFS, de itt élsúlyozottak az élek
* Input: G(V,E), l:E->R+, sV gyökérpont
* Output: megkapjuk a legrövidebb utat s és a többi pont között/ dist(s,v) minden vV csúcsra meghatározunk egy legrövidebb sv utat
* Működése:
  + Kezdetben: d(s)=0, d(v)=∞ minden további élre, ha v≠s
  + Először elvégezzük a javítást s-ből induló élekre
  + Utána tovább lépünk u1-be a minimumot meghatározó élen (s és u1 között már biztos nem lesz rövidebb út, ezért u1-el már nem foglalkozunk) és megvizsgáljuk u1 szomszédjait is, majd itt is kiválasztjuk a minimumot meghatározó élt, ezt addig ismételjük, míg mindegyik élen meghatározzuk a minimum élt.
* Helyessége

Ford

* Input: G=(V,E) gráf, l:E->R konzervatív hosszfüggvény és uϵV csúcs (negatív éleknél is).
* Output: a dist(u,.) hosszfüggvény, azaz G minden v csúcsára dist(u,v) (egy kitüntetett pont és a többi csúcs távolságát tőle adja meg)
* Működése: legyen E={e1,e2….} és legyen d(u)=0 (a kiindulási pont mint a dijkstránál) éd d(v)=, ha u. Az algoritmus fázisokból áll és egy fázis abból áll, hogy sorra megpróbálunk az e1, e2,…. élek mentén javítani. Ha egy fázisban nem történik javítás, akkor az algoritmus véget ért és a dist(u,v)=d(v) áll minden csúcsra.
* Lépésszám: Mivel egy útnak legfeljebb n-1 éle lehet, ezért legkésőbb az (n-1)-dik fázis végén az algoritmus véget ér. A Ford algoritmus minden fázisban nagyjából élszámnyi lépést végzünk, ezért az egész algoritmus lépésszáma n\*m konstansszorzóval becsülhető, ahol m a G élszáma.
* Helyessége

Floyd

* Input: G=(V,E) irányított gráf, l:E->R konzervatív hosszfüggvény
* Output: dist(u,v) minden u,v –re. (megadja a gráf összes pontpárjai közötti távolságot)
* Működése:
  + Legyenek a gráf pontjai :𝑣1, 𝑣2 … 𝑣𝑛 a G csúcsai. Jelölje 𝑑 𝑘(𝑖,𝑗) a 𝑣𝑖 -ből 𝑣𝑗 -be vezető legrövidebb út hossza, úgy, hogy csúcsai a 𝑣𝑖, 𝑣𝑗, 𝑣1, 𝑣2 … 𝑣𝑛 halmazból kerülnek ki.
  + Az algoritmus szomszédossági mátrixhoz hasonló táblázat segítségével működik, ahol a táblázat cellái a sorhoz és oszlophoz tartozó pontok távolságát tartalmazzák.
  + 0.lépés: k=0, (i,j), akkor {0,ha i=j} {∞, ha ij között nincs él) {l(i,j), ha ij közt van él)
  + 1. lépés: (i,j)=min((i,j), (i,k+1)+(k+1,j)), addig amíg k=n, utána vége
  + Lépésszám: konstans \*



* Helyessége

Legrövidebb utak fája bejárásokkal kapcsolatos fogalmak:

Bejárási fa:? A gráfból kirajzolt BFS fa. ?

**Faél**: ami mentén egy új csúcsot elérek

**Előreél**: olyan uvϵE éle a gráfnak, melyre létezik a bejárási fában uv út. (Ha BFS bejárása után a fában van előre él akkor elrontottam valamit)

**Visszaél:** olyan uvϵE éle a gráfnak, melyre létezik a bejárási fában vu út.

**Keresztél:** Minden más éle a G-nek, ha olyan él van a gráfban, aminek a végpontjai között nincs út a BFS fában

**Legszélesebb utak keresése irányítatlan gráfban:**

* út szélessége: az út mentén a legkisebb súlyú él, mivel itt a súlyok nem költséget, hanem szélességet jelentenek (egy út szélessége a legszűkebb útszakasz)/Ha a P a G egy útja, akkor w szélessége a P legkeskenyebb élének szélessége: w(P)=min{W(e)}
* ahhoz h ezt megtaláljuk a módosított kruskal alg. kell használni.

Módosított Kruskal:

* Input: G (V,E) összefüggő gráf, és w: 𝐸 → 𝑅+ költségfüggvény
* Output: w (p), a gráf legszélesebb útja
* Módosítás: minden lépésben a legszélesebb olyan élet választjuk, ami még nem alkot kört, tehát a kruskal algoritmust csökkenő szélesség szerint futtatjuk, akkor a kapott maximális összszélességű feszítőfa olyan lesz, hogy bármely 2 csúcsa között G egy legszélesebb útját tartalmazza.
* Helyessége: