5. Mélységi keresés/PERT

**Mélységi keresés** (DFS=Depth first search)

* Olyan bejárási algoritmus, amikor a soron következőnek elért csúcsot a lehető legkésőbb elért csúcsból érjük el.
* Input: G(V,E) s ∈ V
* Működése: Egy gyökércsúcsból indulok ki, és meglátogatom e csúcs egy szomszédját . Ha -nek van még bejáratlan szomszédja, azt is meglátogatom. Addig folytatom ezt, ameddig nem már nem tudok tovább menni bejáratlan csúcsra, ha ez megtörténik, visszamegyek az előző csúcsra és onnan próbálok tovább menni. Ha onnan sem tudok, újból visszalépek. Ha visszaérek -ba és nincs több bejáratlan csúcsom, akkor véget ért az algoritmus, ha van még bejáratlanom, akkor új gyökércsúcsot keresek és megismétlem ezen lépéseket.
* Alkalmazásai: PERT módszernél

Élek osztályozása:

**Mélységi számozás:** mélységi bejárás elérési sorrend szerinti számozás (a csúcsok bejárásának a sorrendje). Jele: m(V)

**Befejezési számozás:** A mélységi bejárás befejezési sorrend szerinti számozása (a csúcsokból való visszalépés sorrendje). Jele: b(V)

**Faél/Előreél:**

* az az uv él, ahol m(u)<m(v) és b(v)<b(u)
* szavakkal: Legyen uv a gráf éle. v mélységi száma nagyobb, mint u-é, akkor u eléréséig v-t nem értük el, így vissza sem léphettünk v-ből amikor az u-t elértük. Ezért amikor u-t befejezzük (visszalépünk), v-t már el kellett érnünk. v az u közvetlen, vagy közvetett leszármazottja, vagyis uv faél vagy előreél.

**Visszaél**:

* az az uv él, ahol m(v)<m(u) és b(u)<b(v)
* szavakkal: uv él olyan, hogy u mélységi száma nagyobb, mint v-énél. Ekkor persze u-ból v-be nem vezethet út (v nem lehet u leszármazottja), hiszen a faélek mentén a mélységi szám növekszik. Tehát u a v leszármazottja, amikor is uv él visszaél.

**Keresztél:**

* m(u)>m(v) és b(u)>b(v)
* szavakkal: ha u mélységi száma nagyobb, mint v-é, és u és v nem leszármazottjai egymásnak, de ehhez v-ből még u elérése előtt vissza kellett lépnünk.

**Irányított kör keletkezésének eldöntése DFS-el**

* Ha DFS után létezik visszaél, akkor G-ben van irányított kör.
* Ha DFS után nincs visszaél, akkor G-ben nincs irányított kör.
* bizonyítás:

**Alapkörrendszer:**

* alapkör: Ha a G gráf DFS fájába berajzolok egy visszaélet, akkor az egy kört fog alkotni
* alapkörrendszer, ha vesszük az összes ilyen visszaéllel kialakított kört, azok alkotják az alapkörrendszert

**Aciklikus (irányított kört nem tartalmazó) irányított gráfok (DAG)**

* G irányított gráf DAG, ha G-ben nincs irányított kör. (Aciklikus ugyanez)

**Jellemzésük a topologikus sorrenddel**

* V1,V2,…Vn a G csúcsainak topologikus sorrendje, ha minden ViVj ∈ E esetén Vi<Vj
* Tétel: G DAG-> minden topologikus sorrend (nyelőtörléssel)
* Ha G DAG->DFS befejezési sorrend fordítottja topologikus sorrend
* bizonyítás: G DAG->DFS után nincs visszaél, akkor ha uv∈E, akkor b(u)>b(v)

**Topologikus sorrend keresése:**

* topologikus sorrend: A G gráf csúcsinak sorba rendezése, úgy hogy G minden éle egy a sorrendben későbbi csúcsra mutasson (csak aciklikus gráfoknál lehetséges).
* előállítása: a gráfon végünk egy mélységi bejárást, majd az így kapott mélységi befejezési sorrendet fordítottan vesszük.

**PERT módszer:**

* Gráf pontjai munkafolyamatoknak felelnek meg, amiket el kell végezni, az élekre pedig az van ráírva, hogy a két feladat elvégzése között mennyi időnek kell eltelnie.
* A probléma lényege F feladat optimális ütemezése. Szabályok: v tevékenységet nem kezdődhet meg hamarabb u tevékenység után c(uv) időnél. P(F) az F irányított gráfja, aminek a csúcsai a tevékenységek. Minden szabálynak megfelel P(F) egy súlyozott, írányított éle. P(F) aciklikus. F feladat k ütemezése abból áll, hogy F feladat minden tevékenységéhez úgy jelölünk ki egy-egy kezdési időpontot, hogy minden szabályt betartunk. Az F feladat k ütemezés szerinti elvégzéséhet szükséges idő k(t). Optimális esetben k(t) minimális.
* Tétel: Az F feladat hossza megegyezik a P(F) gráfban az irányított st utak maximumával.
* Input: G(V,E) irányított aciklikus gráf, c: E->R+
* (V: tevékenység, E: sorrendiség, c (u,v): u és v kezdése között eltelt min. idő)
* Output: csúcsok legkorábbi kezdési ideje, kritikus út, kritikus tevékenység
* Működése:
  1. Topológikus sorrend megadása (pl.: DFS-el)-> aciklikus
  2. A topológikus sorrenden végighaladva, kiértékeljük a pontok leghamarabbi kezdési idejét (megnézzük az aktuális csúcsba befutó éleket és a legnagyobb élsúllyal rendelkező élt kiválasztjuk, hasonlít a Dijkstra algoritmushoz, csak itt nem a legrövidebbet, hanem a leghosszabb utat keressük a kezdő és az összes többi csúcs között)

**Kritikus út:** Azt az utat a gráfban, melynek a leghosszabb az élsúlyösszege kritikus útnak nevezzük.

**Kritikus tevékenység:** A kritikus úton lévő csúcsok a kritikus tevékenységek. Azért hívjuk őket kritikus tevékenységeknek, mert ha ezeket a munkákat valamilyen okból kifolyólag nem kezdjük el időben, az egész projekt csúszhat.