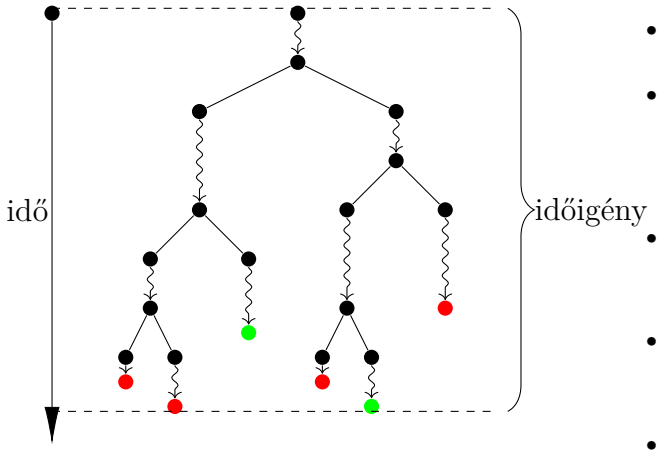
# Bonyolultságelmélet gyakorlat – 05
# Nemdeterminizmus, NP-teljesség

**Recap: nemdeterminisztikus algoritmus**

Egy **nemdeterminisztikus** algoritmusban megengedett elemi lépés az `x := nd()` **nemdeterminisztikus értékadás**: ezután az $x$ változó értéke vagy 0-ra, vagy 1-re áll be.

Első közelítésben felfoghatjuk úgy, mint egy randomizált értékadást, mondjuk $\frac{1}{2} - \frac{1}{2}$ eséllyel állítja be az $x$ változó értékét 0-ra vagy 1-re.

Így persze többféle futása is lehet az algoritmusnak ugyanazon az inputon; ezeket egy **számítási fával** lehet pl. ábrázolni, mint ez:




- az idő fentről lefelé telik
- minden nemdet bit generálásakor kettéágazik a számítás, egyik irány a 0, másik irány az 1 lesz
- a fa alján a zöld/piros levelek az accept/reject válaszok
- az algoritmus akkor fogadja el az inputot, ha **létezik** zöld levél
- időigénye pedig akkor $f(n)$, ha minden $n$ bites inputon minden egyes számítási szál hossza (a fa mélysége) legfeljebb $f(n)$

**Recap: NP**

Az **NP** osztályba azok az **eldöntési** problémák tartoznak, melyek eldönthetők **polinom időkorlátos nemdeterminisztikus algoritmussal**.

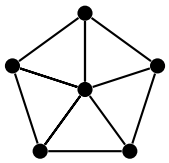
**1. feladat.**

Mutassuk meg, hogy SAT $\in$ **NP**!

**Recap: 3-Színezés**

- **Input:** egy $G$ gráf
- **Output:** kiszínezhetőek-e $G$ csúcsai 3 színnel **helyesen**, azaz úgy, hogy minden élnek a két végpontja különböző színt kapjon?

Pl. ez a lenti gráf a problémának egy „nem” példánya, bárhogy is próbáljuk kiszínezni a csúcsait 3 színnel, mindig lesz olyan él, melynek a két vége azonos színűre sikerül:



**2. feladat.**

Mutassuk meg, hogy 3-Színezés $\in$ **NP**!

**Recap: NP-nehézség, NP-teljesség**

Egy eldöntési probléma...

- **NP**-nehéz, ha **minden** **NP**-beli probléma (polinomidőben) visszavezethető rá,
- ha ezen felül még ő maga is **NP**-beli, akkor ő **NP**-teljes.

Néhány ismerten **NP**-teljes probléma:

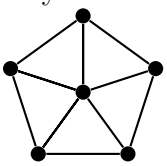
- SAT
- 3-Színezés
- Hamilton-út (input: egy $G$ gráf, output: van-e $G$-ben olyan út, mely $G$ minden csúcsát pontosan egyszer érinti?)

Ahhoz, hogy megmutassuk, hogy egy probléma **NP**-nehéz, elég visszavezetni rá egy ismerten **NP**-nehéz problémát (mert a visszavezetés tranzitív, ezért ettől egyszerre minden **NP**-beli probéma rá is visszavezethető lesz, egymás után alkalmazva a **két** inputkonverziót)

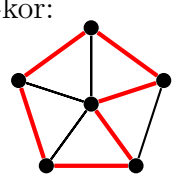
**Recap:** Hamilton-kör

- **Input:** egy $G$ gráf
- **Output:** van-e $G$-ben minden csúcson pontosan egyszer átmenő kör?

Pl. ez a gráf a problémának egy „igen” példánya...



...mert a piros kör benne egy Hamilton-kör:



**3. feladat.**

Mutassuk meg, hogy a Hamilton-kör probléma **NP**-teljes!

**4. feladat.**

Mutassuk meg, hogy a 4-Színezés probléma **NP**-nehéz!

**1. feladat megoldása.**

Megadunk SATra egy nemdeterminisztikus, polinom időkorlátos algoritmust:

- első lépésben gyűjtsük ki az input CNF változóit – ez $O(n)$ lépés (egyszer kell végigolvasnunk az inputot)
- ezután mindegyik változónak **nemdeterminisztikusan** generálunk egy-egy 0/1 (hamis/igaz) értéket (ez újabb $O(n)$ lépés)
- végül determinisztikusan **ellenőrizzük**, hogy a generált értékadás kielégíti-e a formulát (ez újabb $O(n)$ lépés)
- ha kielégíti, Accept, ha nem, Reject választ adunk (ezen a szálon).

Így végeredményben van egy $O(n) + O(n) + O(n) = O(n)$ időkorlátos nemdeterminisztikus algoritmusunk, ami tényleg eldönti a SAT problémát, hiszen:

- ha az input CNF kielégíthető, akkor legalább az egyik szálon tényleg generálunk egy kielégítő értékadást, és az a szál el fogja fogadni az inputot $\Rightarrow$ az egész algoritmus is,
- ha az input CNF kielégíthetetlen, akkor mindegyik szál rejectelni fog $\Rightarrow$ az egész algoritmus is.

**Megjegyzés.**

Minden NP-algoritmus ilyen alakra hozható, mint a fenti:

- először generálunk valami **rövid** (polinom méretű) „potenciális tanút”, ami igazolhatja, hogy az input a problémának egy „igen” példánya,
- eztán determinisztikusan **gyorsan ellenőrizzük** (gyorsan = polinom időben), hogy a generált „potenciális tanú” tényleg tanú-e, ha az, akkor acceptelünk, ha nem, rejectelünk.

Sok esetben nagyon adja magát a tanúsítvány-rendszer, sok olyan **NP**-beli problémát fogunk látni, ami „igaz-e, hogy az inputban van ilyen-meg-ilyen valami” alakú, ilyenkor ezt a „valami”t kell csak megpróbálnunk nemdeterminisztikusan kigenerálni. (Nem mindig ennyire egyértelmű a történet azért, de gyakon sokszor lesz az.)

---

**2. feladat megoldása.**

Az előző feladathoz hasonló „tanú generálás” módszer itt kb. ilyen:

- **generáljunk nemdeterminisztikusan** $G$ minden csúcsához **két** nemdeterminisztikus bitet (terv: 00 jelentse az egyik, 01 a másik, 10 a harmadik színt, az 11 invalid)
- (innentől már végig determinisztikus lesz az algoritmus)
- ha van olyan csúcs, amihez az 11 biteket generáltuk, akkor Reject (ez biztos nem tanú)
- különben, iteráljunk végig az éleken és nézzük meg, hogy egyezik-e az él két végpontján mindkét generált bit – ha igen, akkor Reject
- ha minden él két végpontján legalább az egyik generált bit eltér az él másik végén lévő párjától, akkor Accept.

Ez ismét egy $O(n)$ időigényű nemdeterminisztikus algoritmus, és tényleg eldönti a 3-Színezést.

---

**3. feladat megoldása.**

Ahhoz, hogy belássuk, hogy egy probléma **NP**-teljes, meg kell mutatnunk, hogy **NP**-beli és azt is, hogy **NP**-nehéz.

**NP**-beli: ismét tanú generálással mehet, mint minden **NP**-beli probléma esetén:

- minden egyes élt **nemdeterminisztikusan** vagy válasszunk ki, vagy nem (ezzel egy részgráfot generálunk)
- (innentől determinisztikusan ellenőrizzük, hogy a generált részgráf egy Hamilton-kör-e vagy sem)
- minden csúcsra nézzük meg, hogy pontosan két rá illeszkedő élt sikerült-e generáljunk, ha nem, akkor Reject
- nézzük meg, hogy a generált részgráf összefüggő-e (és nem pl néhány diszjunkt kör uniója), ha nem, akkor Reject
- ha idáig eljutunk, akkor Accept

Ez ismét egy polinom idejű nemdeterminisztikus algoritmus lesz, mely tényleg eldönti a Hamilton-kör problémát.

**NP**-nehéz: ehhez elég választanunk egy szimpatikus **NP**-nehéz problémát és azt visszavezetni erre az aktuális problémánkra.

Talán célszerű lehet ilyenkor olyat keresni, ami „hasonlít” az aktuális problémánkra, pl. most a HAMILTON-ÚT ilyen. (Persze ilyenkor **minden** **NP**-teljes problémáról van erre visszavezetés, csak van, amelyiket könnyebb és van, amelyiket nehezebb megkonstruálni.)

Vezessük tehát vissza a HAMILTON-ÚT problémát a HAMILTON-KÖR problémára!

Ehhez kell egy polinomidejű („gyors”) inputkonverzió, mely

- egy $G$ gráfból egy $G'$ gráfot (a HAMILTON-ÚT inputjából a HAMILTON-KÖR inputját) készít
- választartó módon, azaz
  - ha $G$-ben van Hamilton-út, akkor $G'$-ben lesz Hamilton-kör,
  - ha pedig $G$-ben nincs Hamilton-út, akkor $G'$-ben sem lesz Hamilton-kör,
  - ezt az utóbbit általában úgy egyszerűbb ellenőrizni („kontrapozícióval”), hogy „ha $G'$-ben lett Hamilton-kör, akkor $G$-ben is volt Hamilton-út”.

Itt pl. egy ötlet lehet a következő:

Kapjuk $G$-ből $G'$-t úgy, hogy felveszünk egy új $v$ csúcsot és azt hozzákötjük minden $G$-beli eredeti csúcshoz.

Maga a transzformáció nyilván polinomidejű ($G$ eredeti szomszédsági mátrixához hozzá kell vennünk egy új sort és oszlopot, azt kitöltve 1-esekkel mindenhol, kivéve a sarokban, ahol 0 lesz, hogy ne vegyük fel a hurokélet az új csúcsra, ez $O(n)$ időben megvan), és azért tartja a választ, mert

- Ha $G$-ben van Hamilton-út, mondjuk valami $x \rightsquigarrow y$ út bejárja $G$ minden csúcsát pont egyszer, $x$-ből indul és $y$-ban végez, akkor $G'$-ben ezt az új $v$ csúccsal egy $x \rightsquigarrow y \to v \to x$ körré tudjuk hosszabbítani;
- Ha pedig $G'$-ben lett Hamilton-kör, akkor egy ilyen át kell haladjon $v$-n is, kell jöjjön valahonnan és kell menjen tovább valahova – így ha ezt a kört „elvágjuk” $v$-nél, kivéve belőle $v$-t, akkor amit kapunk, egy út lesz, ami $G'$ összes nem-$v$ csúcsán pont egyszer megy át, a körben a $v$ után érintett csúcsból indul és a körben a $v$ előtti utolsó csúcsban fejeződik be – ami ezek szerint egy Hamilton-út lesz az eredeti $G$-ben.

Tehát ez tényleg egy hatékony visszavezetés a HAMILTON-ÚT problémáról a HAMILTON-KÖR problémára.

**Megjegyzés.**

Ha két **NP**-beli problémát vezetünk vissza egymásra, akkor sokszor így néz ki a visszavezetés:

- megadunk egy transzformációt
- a bal oldali probléma egy tényleges tanúját az eredményben megjelenítjük és megmutatjuk, hogy „be lehet fejezni" a jobb oldali probléma egy tanújává (itt pl. egy Hamilton-utat be tudtunk fejezni még két lépésben a generált gráfban egy körré $v$-n keresztül)
- és a jobb oldali probléma egy tanúját pedig megmutatjuk, hogy (annak egy része) miért is lesz tanú az eredeti probléma eredeti inputjában (itt pl. az új gráfban egy Hamilton-körből ha elhagyjuk az újonnan generált $v$ csúcsot, akkor pont az eredeti gráfban kaptunk egy Hamilton-utat, vagyis egy ottani tanút)

---

**4. feladat megoldása.**

(persze **NP**-teljes is, nem nehéz rá adni az eddigiek alapján egy **NP** algoritmust)

Első lépésben ismét egy „elég hasonló" **NP**-teljes problémát keresünk hozzá, ez lehet pl. most a 3-Színezés.

Adnunk kell tehát egy hatékony inputkonverziót, mely

- egy input $G$ gráfból (a 3-Színezés inputjából) elkészít egy $G'$ gráfot (a 4-Színezés inputját)
- úgy, hogy ha $G$ eredetileg kiszínezhető volt 3 színnel helyesen, akkor $G'$ kiszínezhető lesz 4 színnel helyesen,
- ha pedig $G'$ kiszínezhető lett 4 színnel helyesen, akkor $G$ is az volt 3 színnel.

Itt pl. egy ötlet lehet a következő:

Kapjuk $G$-ből $G'$-t úgy, hogy felveszünk egy új $v$ csúcsot és azt hozzákötjük minden $G$-beli eredeti csúcshoz.

A trafó még mindig persze polinomidejű, kérdés, hogy miért is választartó? Ismét a „tanút tanúvá és vissza alakító" módszerrel mutatjuk meg:

- ha $G$ 3-színezhető, akkor van helyes 3-színezése, ami mondjuk az $\{1, 2, 3\}$ színeket használja. Induljunk ki egy ilyenből és színezzük ki $G'$-t pont így, az új $v$ csúcsnak pedig adjuk a 4 színt. Ekkor $G'$-ben továbbra se lesz ütközés (az eredeti $G$-beli élek eleve helyesen voltak színezve, a $v$-t érintőeknek pedig az egyik vége a teljesen új 4 színt kapta), tehát $G'$ is 4-színezhető, ha $G$ 3-színezhető volt.
- a másik irány is fontos: ha $G'$ 4-színezhető, akkor vegyük neki egy helyes 4-színezését. Mivel az új $v$ csúcs mindenki mással szomszédos, neki egy teljesen egyedi színe van, így ha őt elhagyjuk, a kapott színezett gráfban már csak három színt használunk és a színezés továbbra is helyes, tehát $G$ tényleg 3-színezhető, hiszen ez neki lesz egy helyes 3-színezése.

**Megjegyzés.**

Az azért nem igaz, hogy minden gráfos visszavezetésnél jó megoldás lesz a „vegyünk fel egy új csúcsot és kössük össze mindenkivel” módszer :)