ÚJ SZÉCHENYI TERV

Ég és Föld vonzásában – a természet titkai

Informatikai tehetséggondozás:

Mohó stratégia 1.

TÁMOP-4.2.3.-12/1/KONV

MAGYARORSZÁG MEGÚJUL

A projektek az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósulnak meg.

Többféle feladat megoldási stratégia létezik. Közülük az egyik legegyszerűbb a mohó stratégia, melynek lényege röviden megfogalmazva: minden döntési helyzetben válasszuk azt a döntést, ami pillanatnyilag a legkedvezőbbnek tűnik.

Ez a stratégia persze nem mindig szerencsés, de a feladatok egy viszonylag széles körére alkalmazható.

A stratégia alapján egyértelmű, hogy olyan feladatok esetén merülhet fel egyáltalán az alkalmazása, amikor több döntést kell hozni egymás után (lépésenként), és a döntés jóságát is meg kell fogalmazni valahogy (azaz minimum vagy maximum feltételt fogalmazunk meg).

Filmek

Egy kábelhálózat különböző csatornáin N filmet játszanak. Ismerjük mindegyik film kezdési és végidejét. Egyszerre csak 1 filmet tudunk nézni. Add meg, hogy maximum hány filmet nézhetünk végig és melyeket!

Megoldás-1:

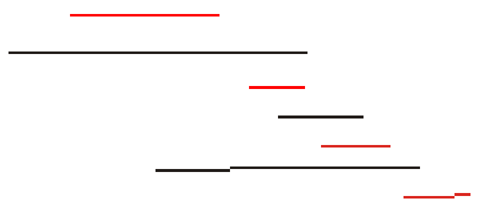
A megoldás egy N elemű halmaz legnagyobb, adott tulajdonsággal rendelkező részhalmazának kiválasztása (a legtöbb film, amelyek egyike sem fedi át a másikat).

Probléma: egy N elemű halmaznak 2^N részhalmaza van.

Futási idő: $O(2^N)$

Ötlet:

Rendezzük sorba a filmeket befejezési idejük szerint növekvő sorrendbe! Ekkor persze elvesznének a korábbi sorszámok, azaz nem tudnánk, melyik filmeket választottuk ki. Emiatt a rendezésben megőrizzük a filmek eredeti sorszámát is.



Ha a leghamarabb befejeződőt választjuk, akkor lesz a legtöbb lehetőségünk a többi közül választani.

Filmek (események) száma: N. Kezdőidők: K_i. Végidők: V_i. Eredeti (rendezés előtti) sorszám: S_i.

Megjegyzés: A sorszámokat tartalmazó tömbre nem lenne szükségünk, ha csak a megnézhető filmek számára lennénk kíváncsiak.

```
Kiválogatás(N,K,V,Db,X):
  Rendezés(N,K,V,S)
  Db:=1; X(Db):=S(1); j:=1
  Ciklus i=2-től N-ig
    Ha K(i)≥V(j) akkor Db:=Db+1; X(Db):=S(i); j:=i
  Ciklus vége
Eljárás vége.
```

Futási idő: O(N*log(N))

Megjegyzés: A továbbiakban feltesszük, hogy van olyan rendezésünk, ami N elemet N*log(N) idő alatt sorba rendez. A fenti algoritmusból látszik, hogy a teljes futási idő nagy részét a rendezési idő teszi ki.

Megoldás-2:

Filmek (események) száma: N. Kezdőidők: K_i. Végidők: V_i. $Kezd_j$: a j-ben végződő, legkésőbb kezdődő film kezdete, S_j a sorszáma (S_j=0, ha j-ben nem végződik egyetlen film sem). $1 \le K_i, V_i \le M$.

```
Kiválogatás(N,K,V,Db,X):
  Db:=0; idő:=0; Kezdetek(N,K,V,Kezd,S)
  Ciklus i=1-től M-ig
    Ha S(i)≠0 és idő≤Kezd(i)
       akkor Db:=Db+1; X(Db):=S(i); idő:=i
  Ciklus vége
Eljárás vége.
```

Megjegyzés: az azonos időpontban végződő filmek közül elég a legkésőbb kezdődőt vizsgálni.

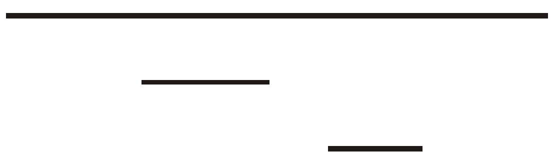
$Kezd_j$ előállítása:

```
Kezdetek(N,K,V,Kezd,S):
  Kezd:=(0,…,0)
  Ciklus i=1-től N-ig
    Ha K(i)>Kezd(V(i)) akkor Kezd(V(i)):=K(i); S(V(i)):=i
  Ciklus vége
Eljárás vége.
```

Futási idő: O(N+M)

Más ötletek

Mi lenne, ha először a leghamarabb kezdődőt választanánk? Nem jó, mert:



Mi lenne, ha először a legrövidebbet választanánk? Nem jó mert:

Munkák-1

Egy vállalkozó 1 napos munkákat vállal. Ismerjük mindegyik munka határidejét. N napot dolgozik, N igényt kapott. Egy nap csak 1 munkát végezhet. Add meg, hogy maximum hány munkát vállalhat el és melyek ezek!

Példa:

```
N=6, határidők: 3 2 7 4 2 1.
```

A megoldás:

```
Elvégezhető munkák száma: 5
Melyik munka Melyik napon
   5             1
   1             3
   2             2
   4             4
   3             7
```

Megoldás:

A megoldás egy N elemű halmaz (a lehetséges munkák halmaza) legnagyobb, adott tulajdonsággal rendelkező részhalmazának kiválasztása (mindegyik eleme – a kiválasztott munka – határidőre befejezhető és nem ütközik más munkával).

Probléma: egy N elemű halmaznak 2^N részhalmaza van.

Futási idő: $O(2^N)$

Ötlet:

Tegyünk minden munkát a legutolsó napra, amikor még elvégezhető!

Ezzel a lehető legkevesebb másik munka elvállalását akadályozzuk meg.

Megoldás-1:

Vegyük sorra a munkákat és mindegyiknek keressük meg a határideje előtt utolsó szabad napot!

```
Kiválogatás(N,H,Db,Nap):
  DB:=0; Nap():=(0,…,0)
  Ciklus i=1-től N-ig
    Ciklus amíg H(i)>0 és Nap(H(i))>0
      H(i):=H(i)-1
    Ciklus vége
    Ha H(i)>0 akkor Db:=Db+1; Nap(H(i)):=i
  Ciklus vége
Eljárás vége.
```

Futási idő: O(N^2)

Megoldás-2:

Rendezzük sorba a munkákat H(i) szerint! Egy munka elvégezhető a határidejére, ha kevesebbet választottunk ki előtte, mint a határideje. Tegyük a munkát az első szabad napra!

```
Kiválogatás(N,H,Db,Nap):
  Rendezés(N,H,S)
  DB:=1; Nap(Db):=S(1)
  Ciklus i=2-től N-ig
    Ha Db<H(i) akkor Db:=Db+1; Nap(Db):=S(i)
  Ciklus vége
Eljárás vége.
```

Futási idő: O(N*log(N))

Más ötletek

Mi lenne, ha a munkát a legelső szabad napra tennénk határidő rendezés nélkül? Nem jó, mert:

Munkák-2

Egy vállalkozó 1 napos munkákat vállal. Ismerjük mindegyik munka határidejét. Egy nap csak 1 munkát végezhet. Az egyes munkákért különböző bért kaphat. Add meg, hogy maximum mennyit kereshet, és mely munkákat kell ehhez elvégeznie!

Megoldás:

A megoldás egy N elemű halmaz legnagyobb értékű, adott tulajdonsággal rendelkező részhalmazának kiválasztása.

Megjegyzés: Mivel minden munka egy napos, ezért ez egyben a legnagyobb elemszámú részhalmaz is, de az összes ilyen elemszámú közül nem mindegy, hogy melyik.

Probléma: egy N elemű halmaznak 2^N részhalmaza van.

Futási idő: $O(2^N)$

Ötlet:

Rendezzük sorba a munkákat az összeg szerint csökkenő sorrendbe! Tegyünk minden munkát a legutolsó napra, amikor még elvégezhető!

Ezzel a lehető legkevesebb másik munka elvállalását akadályozzuk meg és csak nála olcsóbbakat.

Megoldás:

Vegyük sorra a munkákat és mindegyiknek keressük meg a határideje előtt utolsó szabad napot!

Megoldás:

```
Kiválogatás(N,H,Ár,Db,Nap):
  Rendezés(N,H,Ár,S)
  DB:=0; Nap():=(0,…,0)
  Ciklus i=1-től N-ig
    Ciklus amíg H(i)>0 és Nap(H(i))>0
      H(i):=H(i)-1
    Ciklus vége
    Ha H(i)>0 akkor Db:=Db+1; Nap(H(i)):=S(i)
  Ciklus vége
Eljárás vége.
```

Futási idő: $O(N^2)$

Fényképezés-1

Egy rendezvényre N vendég érkezik. Ismerjük mindegyiknek az érkezési és a távozási idejét. A résztvevőket fényképeken szeretnénk megörökíteni. Add meg, hogy minimum hányszor kell fényképet készíteni!

Megoldás:

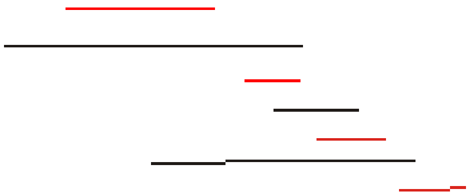
A megoldás a lehetséges időpontok halmaza legkisebb, adott tulajdonsággal rendelkező részhalmazának kiválasztása.

Probléma: egy N elemű halmaznak 2^N részhalmaza van.

Futási idő: $O(2^N)$

Megoldás:

Akkor fényképezzünk, amikor feltétlenül szükséges! Ez azt jelenti, hogy amikor elmenne az első ember, aki még nem volt rajta egy fényképen sem, akkor fényképeznünk kell.



Megoldás:

Emberek (események) száma: N. Érkezési idők: E_i. Távozási idők: T_i. Eredeti (rendezés előtti) sorszám: S_i.

```
Kiválogatás(N,E,T,Db,X):
  Rendezés(N,E,T,S)
  DB:=1; X(Db):=S(1); j:=1
  Ciklus i=2-től N-ig
    Ha E(i)≥T(j) akkor Db:=Db+1; X(Db):=S(i); j:=i
  Ciklus vége
Eljárás vége.
```

A megoldás szó szerint azonos a filmes feladat megoldásával!

Futási idő: O(N*log(N))

Más ötlet

Mi lenne, ha akkor fényképeznénk, amikor a legtöbben vannak jelen? Nem jó, mert:

Fényképezés-2

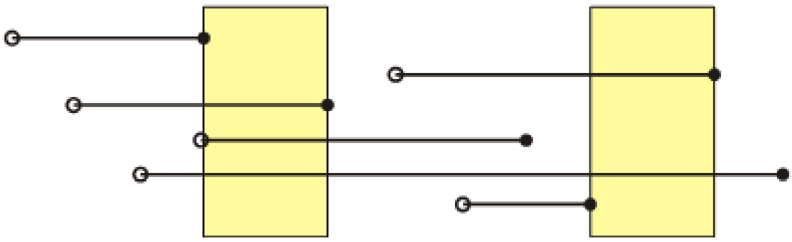
Egy rendezvényre N vendég érkezik. Ismerjük mindegyiknek az érkezési és a távozási idejét. A résztvevőket fényképeken szeretnénk megörökíteni. A fényképészt K perces időintervallumokra fizetjük. Add meg, hogy minimum hány intervallumra kell fizetni!

Megoldás:

A megoldás a lehetséges időpontok halmaza legkisebb, adott tulajdonsággal rendelkező részhalmazának kiválasztása.

Megoldás:

Akkor fényképezzünk, amikor feltétlenül szükséges! Ez azt jelenti, hogy amikor elmenne az első ember, aki még nem volt rajta egy fényképen sem, akkor kezdődik egy fényképezési intervallum.



Megoldás:

Emberek (események) száma: N. Érkezési idők: E_i. Távozási idők: T_i. Eredeti (rendezés előtti) sorszám: S_i.

```
Kiválogatás(N,E,T,K,Db,X):
  Rendezés(N,E,T,S)
  DB:=1; X(Db):=S(1); j:=1
  Ciklus i=2-től N-ig
    Ha E(i)≥T(j)+K akkor Db:=Db+1; X(Db):=S(i); j:=i
  Ciklus vége
Eljárás vége.
```

Futási idő: O(N*log(N))

Rendezvény

Egy rendezvényen N előadást szeretnének tartani. Minden előadó megadta, hogy az előadását mettől meddig tartaná.

Add meg, hogy minimum hány termet kell biztosítani az előadásoknak, hogy mindegyiket megtarthassák!

Megoldás:

Rendezzük sorba az előadásokat kezdési idő szerint! Vegyük sorra az előadásokat és tegyük be az első terembe, ahova betehetők! Ha mindegyik terem foglalt, akkor új termet kell kezdenünk!

Legyen N az előadások száma, K_i az i. előadás kezdete, V_i az i. előadás vége, T_j pedig a j. terembe beosztott utolsó előadás vége.

```
Rendezvény(N,K,V,Db):
  Rendezés(N,K,V)
  Db:=0
  Ciklus i=1-től N-ig
    j:=1
    Ciklus amíg j≤Db és K(i)>T(j)
      j:=j+1
    Ciklus vége
    Ha j>db akkor Db:=Db+1; T(Db):=V(i) különben T(j):=V(i)
  Ciklus vége
Eljárás vége.
```

Futási idő: $O(N^2)$

Ha arra is szükségünk lenne, hogy melyik előadás melyik teremben lesz, akkor a rendezésnél meg kellene őrizni az egyes előadások eredeti sorszámát (S_i), majd az elágazást az alábbira módosítani:

```
    Ha j>db akkor Db:=Db+1; T(Db):=V(i); Hol(S(i)):=db
         különben T(j):=V(i); Hol(S(i)):=j
```

Más ötlet

Mi lenne, ha az első terembe beosztanánk a legtöbbet, amit lehet, utána a maradékot a második terembe, … és így tovább? Nem jó, mert:

Címletezés-1

N-féle pénzjegyünk van, P_1, P_2, …, P_n címletű ($P_i<P_{i+1}$). Add meg, hogy minimálisan melyek felhasználásával fizethető ki az F összeg! Feltehetjük, hogy minden pénzjegyből tetszőleges számú van.

Példa: N=6, P=(1,2,5,10,20,50)), F=196. A megoldás: F=3*50+2*20+5+1, azaz a lehetséges eredmény: Db=(1,0,1,0,2,3).

Megoldás:

Vegyünk a legnagyobb címletű pénzjegyből, amennyi szükséges, majd a maradék összeget fizessük ki a nála kisebb pénz-jegyekkel!

```
Pénzváltás(N,P,F,Db):
  i:=N
  Ciklus amíg F>0 és i>0
    db(i):=F div P(i); F:=F mod P(i)
    i:=i-1
  Ciklus vége
Eljárás vége.
```

Futási idő: O(N)

Probléma: P=(1,3,4), F=6 esetén a megoldás (2,0,1), azaz 3 pénzjeggyel fizetnénk ki a 6 forintot, pedig 6=3+3!

A helyes működés feltétele: 2*P(i)≤P(i+1).

Címletezés-2

N darab pénzjegyünk van, P_1, P_2, …, P_n címletű ($P_i \leq P_{i+1}$). Add meg, hogy minimálisan melyek felhasználásával fizethető ki az F összeg!

Példa: N=9, P=(1,1,5,10,10,10,20,20,50)), F=86. A megoldás: F=50+20+10+5+1, azaz egy lehetséges eredmény: Kell=(hamis, igaz, igaz, hamis, hamis, igaz, hamis, igaz, igaz).

Megoldás:

Vegyünk a legnagyobb címletű pénzjegyből egyet, ha szükséges, majd a maradék összeget fizessük ki a nála kisebb pénz-jegyekkel!

```
Pénzváltás(N,P,F,Kell):
  i:=N; Kell():=(hamis,…hamis)
  Ciklus amíg F>0 és i>0
    Ha F≥P(i) akkor Kell(i):=igaz; F:=F-P(i)
    i:=i-1
  Ciklus vége
Eljárás vége.
```

Futási idő: O(N)

Ha F>0 és i>0 feltétellel álltunk le, akkor maradt még pénz, az összeg nem fizethető ki a megadott pénzjegyekkel.

Probléma: P=(1,1,3,3,4), F=6 esetén a megoldás (igaz, igaz, hamis, hamis, igaz), azaz 3 pénzjeggyel fizetnénk ki a 6 forintot, pedig 6=3+3, azaz 2 pénzjegy is elég!

A helyes működés feltétele: P(i)=P(i+1) vagy 2*P(i)≤P(i+1).

A mohó stratégia elemei

1. Fogalmazzuk meg az optimalizációs feladatot úgy, hogy választások sorozatával építjük fel a megoldást!

2. Mohó választási tulajdonság: Mutassuk meg, hogy mindig van olyan megoldása az eredeti feladatnak, amely a mohó választással kezdődik!

3. Optimális részprobléma tulajdonság: Bizonyítsuk be, hogy a mohó választással olyan redukált problémát kapunk, amelynek optimális megoldásához hozzávéve a mohó választást, az eredeti probléma megoldását kapjuk!

4. Ha a bizonyítás nem megy, akkor keressünk ellenpéldát, ami megmutatja, hogy a mohó választás nem vezet optimumra!