Partíció probléma

Az n természetes szám egy partíciója olyan $\pi = (a_1, \ldots, a_k)$ sorozat, amelyre teljesül:
- $a_1 \geq a_2 \geq \cdots \geq a_k > 0$
- $\sum_{i=1}^{k} a_i = n$

Jelölje $P(n)$ az n szám összes partíciójának számát.

- P(1)=1 (1=1)
- P(2)=2 (2=2, 2=1+1)
- P(3)=3 (3=3, 3=2+1, 3=1+1+1)
- P(4)=5 (4=4, 4=3+1, 4=2+2, 4=2+1+1, 4=1+1+1+1)
- P(5)=7 (5=5, 5=4+1, 5=3+2, 5=3+1+1, 5=2+2+1, 5=2+1+1+1, 5=1+1+1+1+1)

Partíciószám számító algoritmus

Jelölje $P2(n,k)$ n azon partícióinak számát, amelyben minden elem legfeljebb k. Például $P2(4,3) = 4$, $P2(4,2) = 3$, $P2(5,3) = 5$.

Ekkor a következő összefüggések teljesülnek:

- $P2(1,k) = 1, P2(n,1) = 1$
- $P2(n,n) = 1 + P2(n,n-1)$ ($n = n$ és a többiek)
- $P2(n,k) = P2(n,n)$ ha $n < k$
- $P2(n,k) = P2(n,k-1) + P2(n-k,k)$ ha $k < n$ (k benne van vagy nincs)
- A megoldás: $P(n) = P2(n,n)$

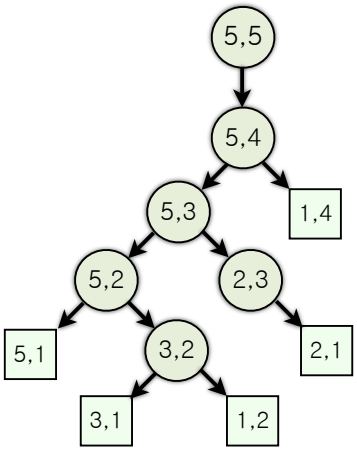
```
PARTÍCIÓ(n)
  Return P2(n,n)
P2(n,k)
  If (k=1) Or (n=1)  return 1
  If k>=n  return  P2(n,n-1)+1
  return P2(n, k-1)+P2(n-k, k)
```

Rekurziós fa: Olyan fa, amelynek minden pontja egy eljáráshívást jelent adott aktuális paraméterekre, úgy, hogy a pont fiai megfelelnek azoknak az eljáráshívásoknak, amelyek végrehajtódnak az aktuális paraméterek esetén.

Futási idő elemzése

Legyen $E(n,k)$ a $P2(n,k)$ eljáráshívás hatására végrehajtott eljárásutasítások száma.
Ekkor $P2(n,k) \leq E(n,k) \leq 2P2(n,k) - 1$.

Lemma $P(n) = \Omega(2^{\sqrt{n}})$



1. ábra.

Bizonyítás Tekintsük az összes olyan $[a_1,\ldots,a_k]$ sorozatot, ahol $\lfloor\sqrt{n}\rfloor \geq a_1$ és $\sqrt{n} \geq a_1 > a_2 > \cdots > a_k > 1$. Ezek száma pontosan $2^{\lfloor\sqrt{n}\rfloor-1}$, mivel minden ilyen sorozat egyértelműen megadható egy $b_{\lfloor\sqrt{n}\rfloor-1},\ldots,b_1$ bitvektorral, ahol $b_x = 1$ akkor és csak akkor, ha $x+1$ eleme a sorozatnak.

Mivel $k \leq \sqrt{n}$, ezért $\sum_{i=1}^{k} a_i \leq n$. Minden ilyen sorozathoz adjunk hozzá a végén annyi 1-est, hogy a számok összege n legyen, tehát n egy partícióját kapjuk. Ha két kiindulási sorozat különböző volt, akkor a kiegészítéssel különböző partíciót kapunk.

Következmény. A P algoritmus futási ideje $\Omega(2^{\sqrt{n}})$.

Rekurzív algoritmus helyességének bizonyítása

I. Terminálás bizonyítása

Bebizonyítandó, hogy minden eljáráshívás végrehajtása véges lépésben befejeződik. (Az eljárás terminál.)

Megállási feltétel: Legyen $P(x_1,\ldots,x_n)$ egy n-paraméteres rekurzív eljárás. Az $M(x_1,\ldots,x_n)$ egész értékeket felvevő kifejezés megállási feltétele a P rekurzív eljárásnak, ha

- 1. $M(a_1,\ldots,a_n) \geq 0$ minden megengedett $a_1,\ldots,a_n$ aktuális paraméterre.
- 2. Ha $M(a_1,\ldots,a_n) = 0$ akkor nincs rekurzív hívás $P(a_1,\ldots,a_n)$ végrehajtásakor.
- 3. Ha van rekurzív hívás $P(a_1,\ldots,a_n)$ végrehajtásakor valamely $b_1,\ldots,b_n$ paraméterekre, akkor $M(b_1,\ldots,b_n) < M(a_1,\ldots,a_n)$.

Példa: $M(n,k) = (n-1)\times(k-1)$ megállási feltétel P2-re.

II. Helyesség bizonyítása

1. Alaplépés. Annak bizonyítása, hogy az eljárás helyes eredményt számít ki, ha az aktuális paraméterek esetén nincs rekurzív hívás.

2. Rekurzív lépés. Feltéve, hogy minden rekurzív hívás helyes eredményt ad, annak bizonyítása, hogy a rekurzív hívások által adott értékekből az eljárás helyes eredményt számít ki.

Összefésülő Rendezés

Oszd meg: A rendezendő sorozatot felosztjuk két $n/2$ méretű részsorozatra.

Uralkodj: A két részsorozatot összefésülő rendezéssel rekurzív módon rendezzük.

Egyesít: Összefésüljük a két rendezett részsorozatot, így kapjuk a rendezett sorozatot.

```
Összefésülő-rendezés(A,p,r)
 If p<r
      Then {q:=[(p+r)/2]
           Összefésülő-rendezés(A,p,q)
           Összefésülő-rendezés(A,q+1,r)
           Összefésül(A,p,q,r)}
```

A teljes tömb rendezését a Összefésülő-rendezés$(A,1,hossz(A))$ kezdőhívás adja meg.

Futási idő elemzés

Legyen $T(n)$ az n darab szám Összefésülő-rendezéssel történő rendezésének legrosszabb eset korlátja.
Ekkor $T(1) = \Theta(1)$ és $T(n) = 2T(n/2) + \Theta(n)$, ha $n > 1$

Állítás $T(n) = O(n \log n)$

Bizonyítás: Teljes indukcióval. Az indukciós lépés:

$$T(n) \leq 2(c \cdot n/2 \log(n/2)) + d \cdot n \leq c \cdot n(\log n - \log 2) + d \cdot n \leq c \cdot n \log n,$$

ha $c \log 2 > d$.

Gyorsrendezés

Oszd meg: Az $A[p,\ldots,r]$ tömböt két nem üres $A[p,\ldots,q]$ és $A[q+1,\ldots,r]$ résztömbre osztjuk úgy, hogy $A[p,\ldots,q]$ minden eleme kisebb vagy egyenlő $A[q+1,\ldots,r]$ minden eleménél.

Uralkodj: Az $A[p,\ldots,q]$ és $A[q+1,\ldots,r]$ résztömböket a gyorsrendezés rekurzív hívásával rendezzük.

Egyesít: A rendezés helyben rendez, így nincs szükség egyesítésre.

```
GYORSRENDEZÉS(A,p,r)
 If p<r
      Then {q=FELOSZT(A,p,r)
            GYORSRENDEZÉS(A,p,q)
            GYORSRENDEZÉS(A,q+1,r)}
```

A teljes tömb rendezését a GYORSRENDEZÉS$(A,1,hossz(A)$ kezdőhívás adja meg.

Feloszt eljárás (Hoare féle)

```
Feloszt(A,p,r)
  x:=A[p]
  i:=p-1
  j:=r+1
  while(Igaz)
       {repeat j:=j-1
              until A[j]<=x
        repeat i:=i+1
              until A[i]>=x
         if i<j
             then csere(A[i],A[j])
             else return j}
```

Példa:

$i[5,3,2,6,1,4,3,7]j$ x=5

$[5^i,3,2,6,1,4,3^j,7]$

$[3^i,3,2,6,1,4,5^j,7]$

$[3,3,2,6^i,1,4^j,5,7]$

$[3,3,2,4^i,1,6^j,5,7]$

$[3,3,2,4,1^j,6^i,5,7]$

Return 5

Feloszt eljárás (Lomuto féle)

```
Feloszt(A,p,r)
  x:=A[r]
  i:=p-1
  for j=p to r-1
        {if A[j]<=x
         then {i:=i+1
               csere(A[i],A[j])}}
  csere(A[i+1],A[r])
  return i+1
```

Példa:

$i[2^j,8,7,1,3,5,6,|4]$ x=4

$[2^i|,8^j,7,1,3,5,6,|4]$

$[2^i|,8,7^j,1,3,5,6,|4]$

$[2,1^i|,7,8,3^j,5,6,|4]$

$[2,1,3^i|,8,7,5,6,|4]$

$[2,1,3^i|,4,7,5,6,8]$

Megjegyzés: A Lomuto féle felosztásnál a gyorsrendezést a (A,p,q-1) (A,q+1,r) paraméterekre hívjuk meg.

Futási idő elemzés

Legrosszabb eset

A felosztás mindig egy egyelemű tömbre és a többi elemre oszt. Ekkor

$$T_{lr}(n) = T_{lr}(n-1) + \Theta(n)$$

$$T(n) = \sum_{k=1}^{n} \Theta(k) = \Theta(\sum_{k=1}^{n} k) = \Theta(n^2).$$

Legjobb eset

A felosztás mindig felezi a tömböt. Ekkor

$$T_{lj}(n) = 2T(n/2) + \Theta(n)$$

$$T_{lj}(n) = \Theta(n \log n)$$

Átlagos eset

- Legyen X a eljárás végrehajtása során a FELOSZT által végrehajtott összehasonlítások száma n elemű bemenetre. Ekkor a teljes futási idő $O(n+X)$.
- Legyen z_i az i-edik legkisebb eleme a bemenetnek, és $Z_{ij} = \{z_i, z_{i+1}, \ldots, z_j\}$.
- Legyen p_{ij} annak a valószínűsége, hogy z_i-t és z_j-t valamelyik feloszt eljárás összehasonlítja.
- Ekkor $p_{ij} = 2/(j-i+1)$, mivel akkor lesznek összehasonlítva, ha a Z_{ij} halmazból vagy z_i vagy z_j az első felosztó elem.
- Az összehasonlítások várható száma $\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} 2/(j-i+1) = O(n \log n)$. Itt kihasználjuk, hogy a várható érték additív.

Partíció probléma rekurzíómemorizálással

A partíciószám rekurzív algoritmusa $\Omega(2^{\sqrt{n}})$ műveletet végez, pedig a megoldandó részfeladatatok száma sokkal kisebb $O(n^2)$.

A probléma, hogy bizonyos már megoldott részfeladatokat az algoritmus nagyon sokszor újra kiszámol.

Megoldás: jegyezzük fel a kiszámolt értéket, és ha már megvan nincs szükség rekurzív hívásra.

```
PARTÍCIÓRM(n)
  Return P2RM(n,n)
P2RM(n,k)
  if T[n,k]>0 then return T[n,k]
  if (k=1) Or (n=1) then  {T[n,k]:=1
                           return 1}
  else if k>=n  then   {T[n,k]:=P2RM(n,n-1)+1
                       return  T[n,k]}
  else  {T[n,k]:=P2RM(n,k-1)+P2RM(n-k,k)
        return  T[n,k]}
```

A futási idő és tárigény $O(n^2)$.

Kisdolgozatok

- Beszúró rendezés
- Partíciószámot kiszámító algoritmus
- Partíciószámot kiszámító algoritmus rekurziós fája egy megadott értékre
- Gyorsrendezés+Feloszt Hoare
- Gyorsrendezés+Feloszt Lomuto