Zatitu eta irabazi teknika

(Divide y vencerás - divide & conquer)

R. Arruabarrena
LSI - UPV/EHU

Sarrera

- Goitik beherako diseinua
  - Problema azpiproblematan banatu
  - Azpiproblema bakoitza errekurtsiboki ebatzi
  - Azpiproblemen emaitzak konbinatuz jatorrizko soluzioa eraiki
  - Kostua: 3 atalen kostuen batuketa
  - Adibideak:
    - Bilaketa dikotomikoa, MergeSort, QuickSort, k.hautaketa,…
  - Abantailak:
    - Diseinu teknika naturala
  - Desabantailak
    - Azpiproblemak independienteki ebazten dira, eta azpiproblema bera behin eta berriro ebaztea gerta liteke, alferrikako T(n) gastatuz

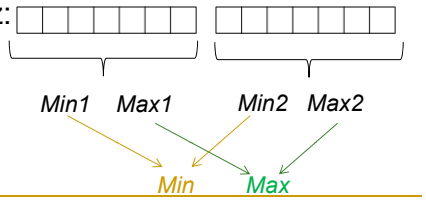
Sekuentziako balio Max eta Min

Bektore bateko balio handiena eta balio txikiena mugatzen dituen algoritmoa egizu. Soluzioak (3/2) n – 2 alderaketa gehienez eta zatitu eta irabazi teknika erabili behar du.

- Soluzio nabariak:
  - maximoa aurkitzeko: (n-1) alderaketa
  - Minimoa aurkitzeko: (n-1) alderaketa

  } *2n-2*

- Zatitu eta irabazi teknika bidez:




```
procedure MIN_MAX  ( B: in BEKTOREA; MIN, MAX: out INTEGER) is
   MIN1,MIN2,MAX1,MAX2: INTEGER;
begin
    if B'LENGTH=2 then
        if B(B'FIRST)> B(B'LAST)
         then MAX:= B(B'FIRST); MIN:= B(B'LAST);
        else MAX:= B(B'LAST); MIN:= B(B'FIRST);
        end if;

   esle MIN_MAX(B(B'FIRST..(B'FIRST+B'LAST)/2),  MIN1, MAX1);
        MIN_MAX(B((B'FIRST+B'LAST)/2)+1)..B'LAST),  MIN2, MAX2);

        if MAX1 > MAX2  then  MAX:= MAX1;
        else MAX:= MAX2;
        end if;

        if MIN1 < MIN2  then  MIN:= MIN1;
        else MIN:= MIN2;
        end if;
   end if;
```

MergeSort ordenazio algoritmoa

```
procedure  MERGESORT (S: in out OS_SEK)  is
   S_LAG: OS_SEK(S'RANGE);
begin
   if  S'LENGTH>1  then
       MERGESORT (S(S'FIRST..((S'FIRST+S'LAST)/2)-1));
                  1              (n/2)-1
       MERGESORT (S(S'FIRST+S'LAST)/2).. S'LAST));
                 (n/2)            n
       MERGE (S(S'FIRST..((S'FIRST+S'LAST)/2)-1),
              S(S'FIRST+S'LAST)/2).. S'LAST)),
              S_LAG);
       S:=S_LAG;
   end if;
end MERGESORT;
```


- Analisia:

  T(n)= #alderaketak
  - T(2)=1
  - T(n)= 2 T(n/2) + 2

  Hedapen metodoa erabiliz:

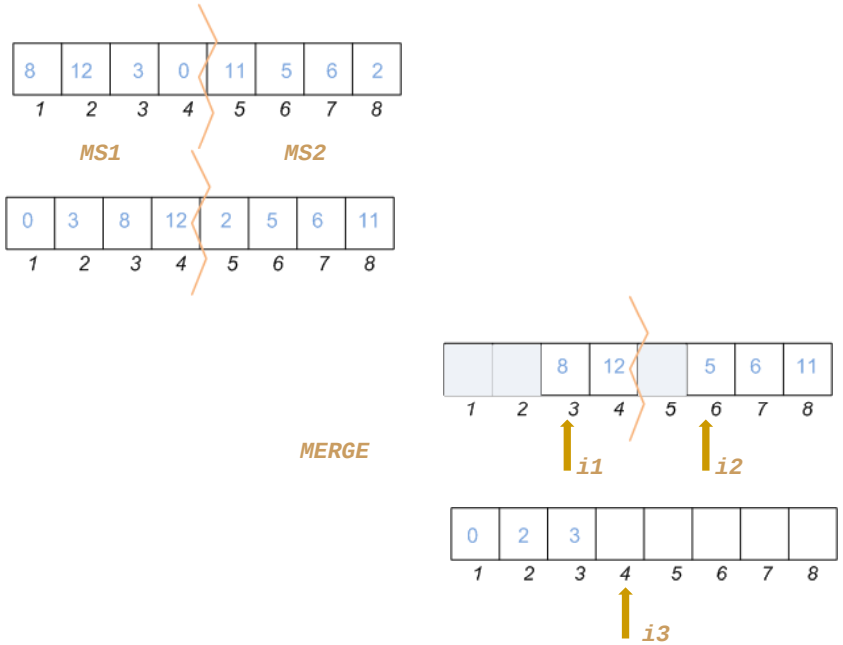
  T(n) = 2 T(n/2) + 2

  = 2 (2T(n/2²) +2)+ 2 = $2^2T(n/2^2)+2^2+2$

  = $2^iT(n/2^i)+2^i+\ldots+2$ i pausoan

  = $2^iT(2)+2^{i+1}- 2$ $2=n/2^i$; $n/2= 2^i$

  = 2^i 1+ 2 2^i -2 = **(3/2) n - 2**




```
procedure  MERGE  (L1,L2: in OS_SEK; L3: out OS_SEK) is

   Ind_L1: INDIZEA:= L1'FIRST; Ind_L2: INDIZEA:= L2'FIRST;
   Ind_L3: INDIZEA;

begin
   for  Ind_L3  in 1..L1'LENGTH+L2'LENGTH  loop
        if      (Ind_L1<=L1'LAST) and (Ind_L2<=L2'LAST)
        then    if      L1(Ind_L1)<L2(Ind_L2)
                then    L3(Ind_L3):=L1(Ind_L1); Ind_L1:= Ind_L1+1;
                else    L3(Ind_L3):=L2(Ind_L2); Ind_L2:= Ind_L2+1;
                end if;
        elsif (Ind_L1>L1'LAST)
        then    L3(Ind_L3):=L2(Ind_L2); Ind_L2:= Ind_L2+1;
        else    L3(Ind_L3):=L1(Ind_L1); Ind_L1:= Ind_L1+1;
        end if;
   end loop;
end MERGE;
```

- Analisia:

$T(1)= \Theta(1)$

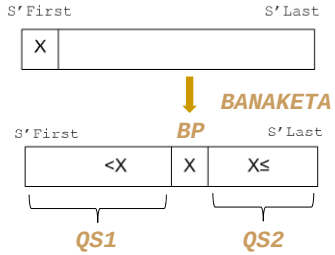
$T(n)=2\,T(n/2) + T_{merge}(n) = 2T(n/2) + \Theta(n)$

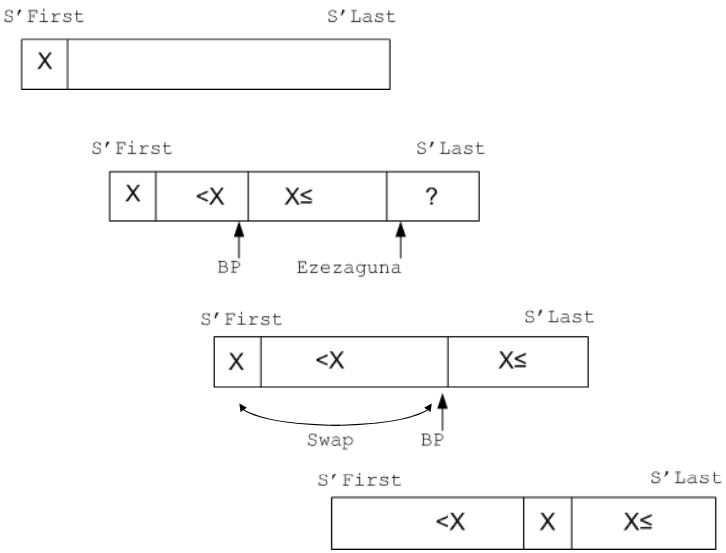
```
T(n) = 2T(n/2)+n
     = 2(2T((n/2)/2)+n/2)+n=2²T(n/2²)+2n
     = 2² (2T((n/2²)/2)+n/2²)+2n=2³T(n/2³)+3n

     = 2ⁱ T(n/2ⁱ)+i n      i pausoan; 1=n/2ⁱ; n=2ⁱ; lg n=i
     = n T(1)+n lg n= n + n lg n
```

$T(n) \in \Theta(n \lg n)$

QuickSort ordenazio algoritmoa

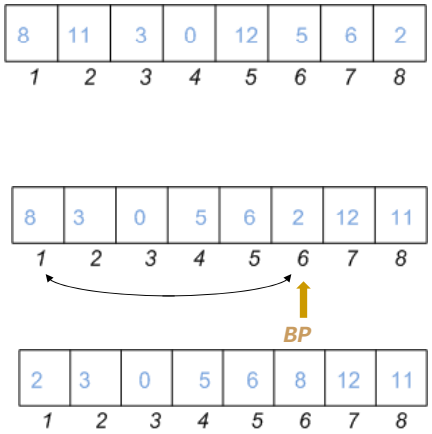
```
procedure QUICKSORT (S: in out OS_SEK) is
   BP:PUNTUA:INDIZEA;
begin
   if  S'LENGTH >1  then
       BANAKETA(S, BP);
       QUICKSORT (S(S'FIRST.. BP-1));
       QUICKSORT (S(BP+1.. S'LAST));
   end if;
end QUICKSORT;
```






```
procedure BANAKETA (S: in out OSOEN_SEK; BP: out INDIZEA) is
   X: INTEGER; BP_LAG: INDIZEA;

begin
   X := S(S'FIRST); BP_LAG:= S'FIRST;

   for  EZEZAGUNA in  S'FIRST+1 .. S'LAST  loop
        if     S(EZEZAGUNA) < X
        then   BP_LAG:= BP_LAG+1;
               TRULEA(S(BP_LAG), S(EZEZAGUNA));
        end if;
   end loop;

   TRUKEA(S(S'FIRST), S(BP_LAG));
   BP := BP_LAG;

end BANAKETA;
```

Banaketa:



- Analisia:
  - T(n)= #alderaketak
  - $T(n)= T_{banaketa}(n)+T_{qs1}(f(n))+T_{qs2}(n-f(n))$
  - Sentikorra da sarreraren antolaketarekiko
    - Kasu onena: banaketa puntua beti erdian:
      $T(n)= 2T(n/2) + n \in \Theta(n \lg n)$
    - Kasu txarrena: banaketa puntua beti ertz batean geratzen denean. Adibidez, ordenatua
      $T(n)= T(n-1) + n \in \Theta(n^2)$
    - Bataz-besteko kostua: kasu onena eta txarrenaren artekoa

- $T_b(1)=0$
- $T_b(n)= 1/n (n-1+T_{qs1}(0)+T_{qs2}(n-1)) +$ BP=1
  $1/n (n-1+T_{qs1}(1)+T_{qs2}(n-2)) +$ BP=2
  $1/n (n-1+T_{qs1}(2)+T_{qs2}(n-3)) +$ BP=3
  …
  $1/n (n-1+T_{qs1}(n-1)+T_{qs2}(0))$ BP=n

$$T_b(n) = (n-1) + \frac{1}{n} 2\sum_{i=2}^{n-1} T_b(i)$$

Batuketaren balioa indukzio bidez frogagarria: *Tb(n) ≤ k n ln n*

- n=1 denean: T(1)= 0 eta k 1 ln 1=0
- n>1 denean: indukzio hipotesi garatua $\forall i(1 \le i < n \rightarrow Tb(i) \le ki \ln i)$

$$T_b(n) \le (n-1) + \frac{2}{n}\sum_{i=2}^{n-1} ki \ln i$$

$$T_b(n) \le (n-1) + \frac{2k}{n}\int_2^n x \ln x dx = (n-1) + \frac{2k}{n}\left[\frac{x^2}{2}\ln x - \frac{x^2}{4}\right]_2^n$$

$$T_b(n) \le kn \ln n - 1 + \left(1 - k/2\right)n + 2k/n\,(1 - 2\ln 2)$$

$T_b(n) \in \Theta(n \lg n)$

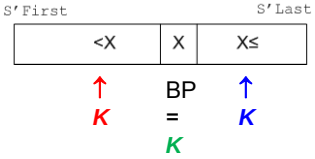
K.aren hautaketa

Osokoen taula bat emanik, bertako K osagai txikiena bilatuko duen algoritmoa idatzi eta ordena azter ezazu; hots:

m S bektoreko K. osagai txikiena da ⇔

$$\#i(1 \le i \le n \rightarrow S(i) < m) < K \quad \text{eta} \quad \#i(1 \le i \le n \rightarrow S(i) \le m) \ge K$$

- Kasu ezagunak
  - K=n/2 denean: medianaren problema; K=1: minimoa; K=n: maximoa
- Soluzio nabaria:
  - Bektorea ordenatu eta K posizioan dagoen itzuli. Kostua $\Theta(n \lg n + 1)$
- Zatitu eta irabazi teknika bidez, QuickSort-en oinarrituta
  - Bataz bestean: $T^K(n) \in O(n)$
  - Kasu txarrenean: $T^K(n) \in O(n^2)$




```
procedure KHautaketa (S:Bektorea; K: integer; BalK: Balioa)
--N=S'Last-Bek'First+1   1≤k≤N
begin
Banaketa(S,BP);
  if    BP=S'First+K-1        --BP=K
  then  BalK:= S(BP);

  elseif  BP>S'First+K-1      --BP>K
  then  KHautaketa(S(S'First..BP-1), K, BalK);

  else -- BP<S'First+K-1      --BP<K
       KHautaketa(S(BP+1.. Bek'Last), K-(BP-S'First+1), BalK);
  end if;
end.
```


- Analisia, batazbesteko kasua:

$$T_b(n,K) = \mathbf{T^K(n)} = \text{\#alderaketak}$$

  - Oharrak:
    - BP non kokaturik geratzen den garrantzia du
      - ondorengo dei errekurtsibo mugatzen baitu
    - BP n posizio desberdinetan eror liteke.
      - Denak ekiprobableak
    - 1/n da posizio konkretu batean erortzeko probabilitatea

$$
\begin{aligned}
&T^K(1) = 1\\
&T^K(n) = 1/n\ (n + T^{K-1}(n-1)) + && BP=1 \wedge BP<K\\
&\qquad 1/n\ (n + T^{K-2}(n-2)) + && BP=2 \wedge BP<K\\
&\qquad \ldots\\
&\qquad 1/n\ (n + T^{K-(k-1)}(n-(K-1))) + && BP=K-1 \wedge BP<K\\
&\qquad 1/n\ n && BP=K\\
&\qquad 1/n\ (n + T^{K}((K+1)-1)) + && BP=K+1\ (\wedge BP>K)\\
&\qquad \ldots\\
&\qquad 1/n\ (n + T^{K}(n-1)) && BP=n\quad (\wedge BP>K)
\end{aligned}
$$

$$T^k(n) = n + \frac{1}{n}\left(\sum_{bp=1}^{k-1} T^{k-bp}(n-bp) + \sum_{bp=k+1}^{n} T^k(bp-1)\right)$$

- Batuketa garatzeko, goi bornatu egiten da honakoa espresioa erabiliz

$$R(n) = \max_{1\le k\le n}\left\{T^k(n)\right\} \qquad \forall n\ \ R(n) \le R(n+1)$$

- T(n) definizioan bornaketa espresioa ordezkatuz:

$$T^k(n) \le n + \frac{1}{n}\max_{1\le k\le n}\left(\sum_{bp=1}^{k-1} R(n-bp) + \sum_{bp=k+1}^{n} R(bp-1)\right)$$

$$= n + \frac{1}{n}\max_{1\le k\le n}\left(\sum_{i=n-k+1}^{n-1} R(i) + \sum_{i=k}^{n-1} R(i)\right)$$

- R(n) goi-bornaketa lineala dela froga liteke indukzio bidez (ikus oinarrizko bibliografia):

$$R(n) = \max_{1\le k\le n}\left\{T^k(n)\right\} \le 4n$$

- $T^K(n)$ funtzioa R(n) bidez goi bornatu da, eta $R(n) \in O(n)$; ondorioz,

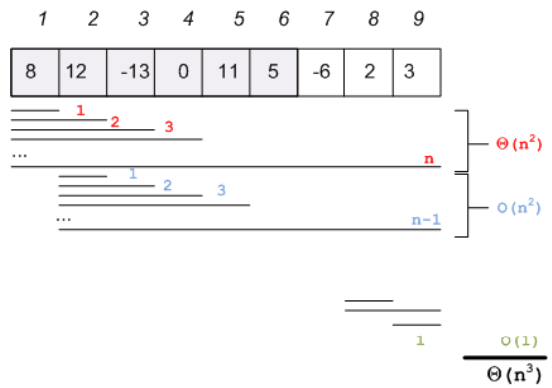
$$\mathbf{T^K(n) \in O(n)}$$

Batura maximoko segmentua

Osokoen taula bat emanik, batura maximoa duen osagaien segmentua identifikatu behar da.

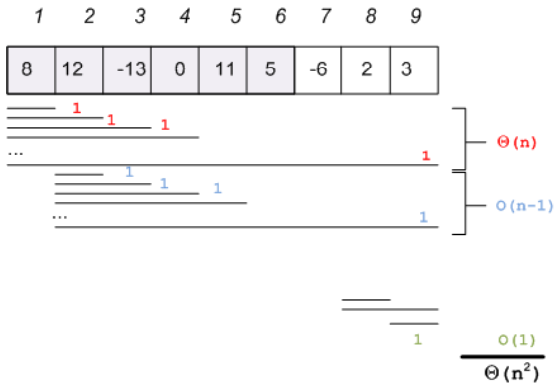
- Soluzioa nabaria:

Kubikoa



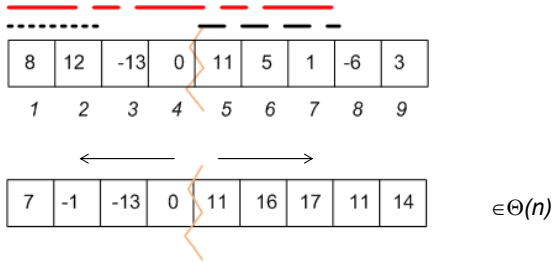
- Errazki hobe liteke:

soluzio koadratikoa



- Zatitu eta irabazi teknika bidez, magnitude hori hobetzeko, adibidez:
  - T(n)=2T(n/2)+O(n) kostuko kodea asmatu beharko genuke

- Algoritmoa:
  - 2 dei eta
  - aztertu ez diren sarrerak eskuz eraginkorki
    - Zehazki O(n)



$\in\Theta(n)$

- Eta aldagaia honela geldituko lirateke:
  - `B_Ez= 20` `Lehena_Ez=1` `Azkena_Ez=2`
  - `B_Erd_Ez+B_Erd_Es=7+17` `Posi_Erd_Ez=1` `Posi_Erd_Es=7`
  - `B_Es=17` `Lehena_Es=5` `Azkena_Es=7`


```
procedure   Batura_Max     (S: In Bektorea; B: Out Integer;
                            Lehena,Azkena: Out Indizea) Is
begin
   if  S'length=2
   then 3_artean_Max(S(S'first..S'first), S'first,S'first,
                     S(S'first..S'last), S'first,S'last,
                     S(S'last..S'last), S'last,S'last,
                     B,Lehena,Azkena);
   else Batura_Max   (S(S'first..(S'first+S'last)/2),
                     B_Ez,Lehena_Ez,Azkena_Ez);
        Batura_Max   (S(((S'first+S'last)/2)+1..S'last),
                     B_Es,Lehena_Es,Azkena_Es);
        Lag(S'first+S'last)/2):= S(S'first+S'last)/2);
        for I in reverse S'first..((S'first+S'last)/2)-1 loop
             Lag(I):= S(I)+Lag(I+1);
        end loop;
        Bektorean_Maximoa(Lag(S'first..(S'first+S'last)/2),
                          B_Erd_Ez, Posi_Erd_Ez);
```



```
        Lag((S'first+S'last)/2)+1):= S((S'first+S'last)/2)+1);
        for I in ((S'first+S'last)/2)+2..S'last  loop
              Lag(I):= Lag(I-1)+S(I);
        end loop;
        Bektorean_Maximoa(Lag((S'first+S'last)/2)+1..S'last),
                          B_Erd_Es, Posi_Erd_Es);

        3_Artean_Max(B_Ez,Lehena_Ez,Azkena_Ez,
              B_Erd_Ez+B_Erd_Es, Posi_Erd_Ez ,Posi_Erd_Es,
              B_Es,Lehena_Es,Azkena_Es,
              B,Lehena,Azkena);
end Batura_Max;
```