Titolo nota 04/03/2016

$n > k$

$n \leq k$

MERGE_SORT
INSSORT
} MERGEINSSORT

$$T(n) = \underbrace{c_1 n \log(n/k)}_{MS} + \underbrace{c_2 n k}_{INSSORT}$$

$\log(n/k) =$

$\log n - \log k$

$$T(n) = \Theta(n \log n + nk)$$

finché $k = \Theta(\log n)$

Titolo nota 04/03/2016

ipotesi: $T(n) = 2n^2$ INSSORT

$T(n) = 64\, n \log n$ MERGESORT

trovare il max di n per cui è conveniente usare INSSORT

$$2n^2 \leq 64\, n \log n$$

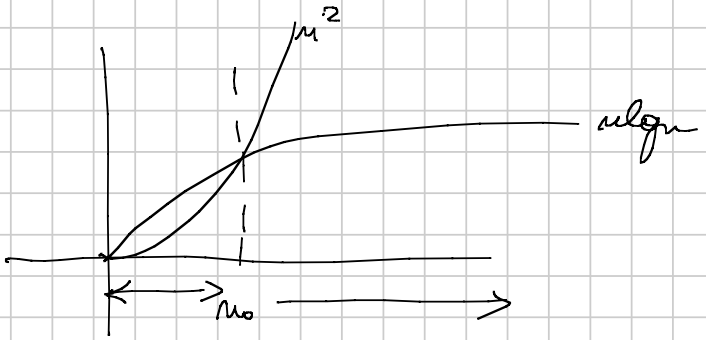
$$n \leq 32 \log n$$

$n = 2^7$

$$2^7 \leq 32 \log 2^7$$

$$128 \leq 32 \cdot 7 = 224$$

Conviene INSSORT



Titolo nota 04/03/2016

$n = 2^8$

$2^8 \leq 32 \cdot \log_2 2^8 = 32 \cdot 8 = 256$

$\downarrow$

256

$k = 2^8$ MERGEINSSORT (A, sx, cx, K)

Titolo nota 04/03/2016

Limiti inferiori

Problema $\Theta(n \log n)$

- dimensione dei dati (input, output) — Tempo — Confronti
- eventi contabili
- albero di decisione $k^i \geq |sol|$
- oracolo o avversario

Esiste un algoritmo che faccia un numero minore di confronti?

Stampare il contenuto di un array A di n elementi

$\Omega(n)$

stampare il numero di permutazioni di A di n el.

$\Omega(n!)$

Titolo nota 04/03/2016

Permutazioni $= n!$ $\quad n = 3$

a b c
b a c
c b a
c a b
b c a
a c b

$n! = 6$

$n! = (n-1)! \, n$

$n! = n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot 1$

$n! = O(n^n)$

↑

crescita esponenziale

Sorting . limite inf. con dimensione dei dati

$\Omega(n)$

Titolo nota 04/03/2016

SORTING che usa confronti di elementi

$n = 3$ a, b, c

5, 7, 3

c a b

3 5 7

$2^0 = 1$ radice

$2^1 = 2$

$2^2 = 4$

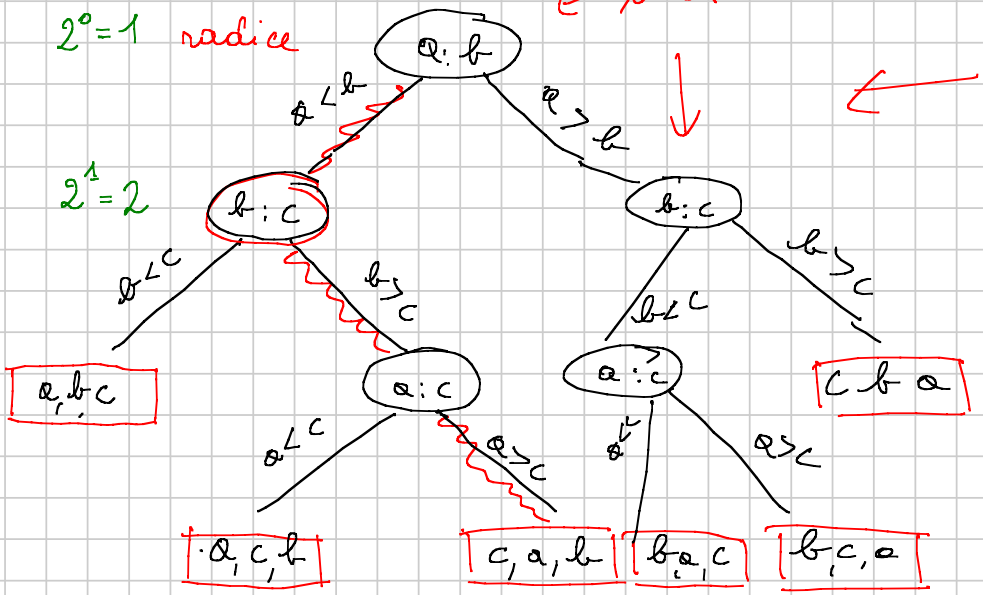
$2^3 = 8$

$2^i \geq 6$

$i \geq \lceil \log_2 6 \rceil$

$i \geq 3$

a livello i ci sono 2^i nodi



← nodi

albero di decisione

$3! = 6$ = numero di casi possibili

foglie

i = n° di confronti

2^i = numero di nodi a livello i

Titolo nota 04/03/2016

SORTING n elementi

schema gen. per limite inferiore

1) contare le soluzioni (tutti i casi possibili)

tutte le permutazioni di n elementi = $n!$

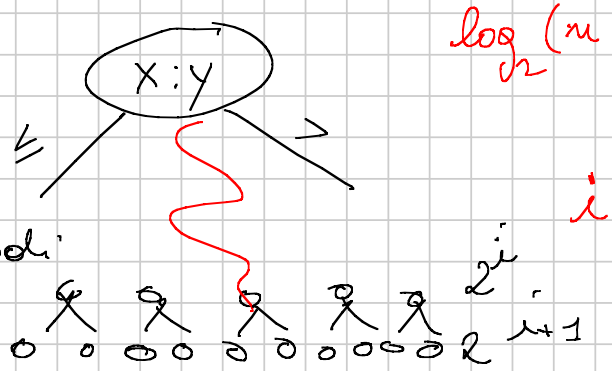
$2^i \geq |\text{soluzioni}| \geq n!$

$i \geq \log_2(n!)$

$\log_2(n!) = \Theta(n \log n)$

$i \geq \Theta(n \log n)$

2) Albero di decisione



livello i 2^i nodi

Titolo nota 04/03/2016

Problema dell'ordinamento

richiede $\Omega(n \log n)$ confronti nel caso pessimo

limiti superiori:
- $O(n^2)$ con INSSOR e $\Theta(n^2)$ con SelSort
- $\Theta(n \log n)$ con Merge Sort

limite inferiore:
- ~~$\Omega(n)$ confronti con dimensione dei dati~~
- $\Omega(n \log n)$ " con albero di decisione

Merge Sort è ottimo rispetto al numero di confronti

Titolo nota 04/03/2016

Ricerca di k in A di n elementi

stabilire limite inferiore $\Rightarrow$ albero di decisione

1) conta soluzioni : $n+1$

2)

3^0

3^1

$\vdots$

$\rightarrow$ ○ ○ ○ ○ 3^i

$3^i \geq n+1$

$i \geq \log_3(n+1) = \Omega(\log n)$

$T_{RICBIN}(n) \longleftrightarrow O(\log n)$

Ricerca Binaria è ottimo

Titolo nota 04/03/2016

Eventi contabili

1) individuare eventi che devono per forza succedere nella soluzione del problema
2) contare gli eventi

Calcolare il max di A di n elementi

$\Omega(n)$

evento: per poter dire che un elemento non è max deve essere risultato minore di almeno un confronto

$n-1$ è il n° confronti minimo per trovare max

Titolo nota 04/03/2016

Algoritmo per il max banale $n-1$ confronti OTTIMO

ALG. del Torneo

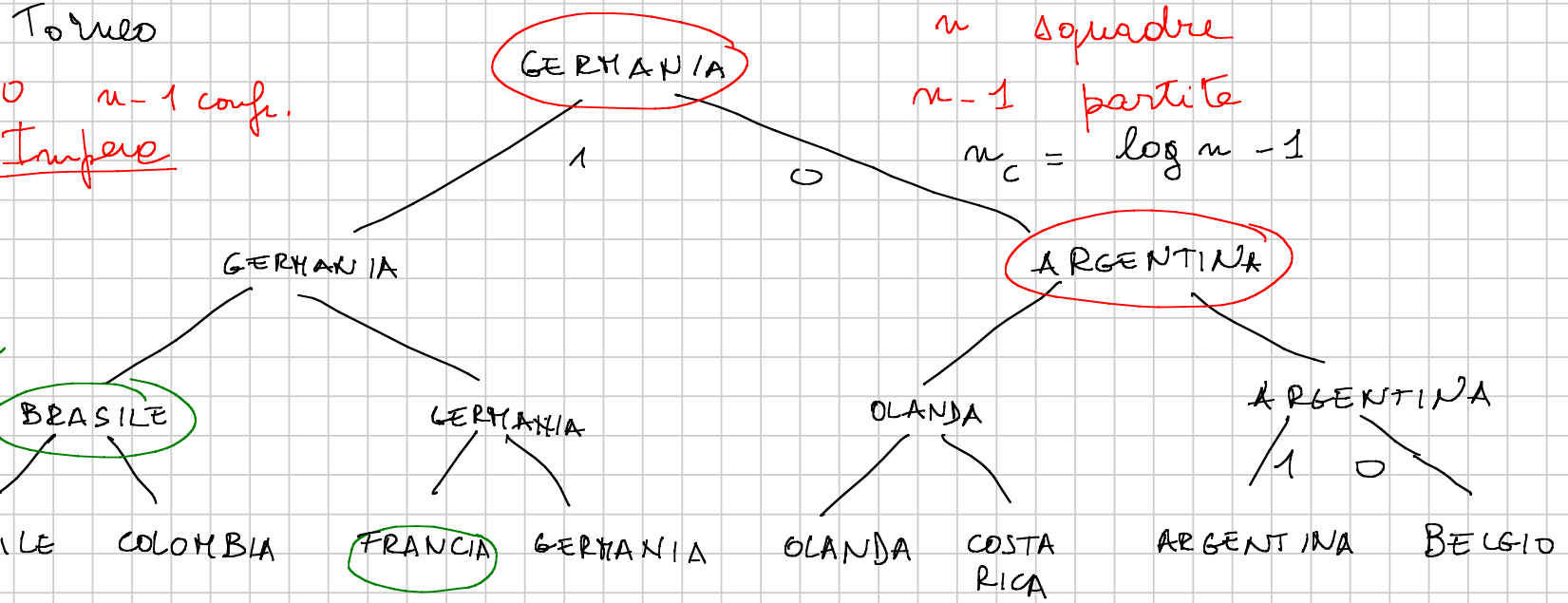
OTTIMO $n-1$ confr.

Divide et Impera

n squadre

$n-1$ partite

$n_c = \log n - 1$



Il secondo classificato è il vero secondo con $p \simeq \frac{1}{2}$

$\Sigma = n$

Titolo nota

04/03/2016

Titolo nota 04/03/2016

Fare un nuovo torneo tra tutte le squadre che hanno perso dal campione

Problema di determinare il primo e il secondo

quanti confronti?

$C \geq \underbrace{n-1}_{\text{per trovare il max}} + \underbrace{X-1}_{\text{per trovare il max tra i perdenti dal campione}}$

Con TORNEO

$X = \underbrace{\log n}_{\text{è il valore minimo}} - 1$

ORACOLO

ORACOLO (AVVERSARIO)

Ventualità peggiore che si può verificare