

## 8. Algoritmi di Ordinamento Riepilogo e Minimo Numero di Confronti

Il *problema dell'ordinamento* può essere definito come segue:

**Input:**  $n$  elementi  $x_1, x_2, \dots, x_n$ , appartenenti a un dominio  $D$  su cui è definita una relazione  $\leq$  di *ordine totale*.

**Output:** Sequenza  $x_{j_1}, x_{j_2}, \dots, x_{j_n}$  dove  $(j_1, j_2, \dots, j_n)$  è una permutazione di  $1, 2, \dots, n$  tale che  $x_{j_1} \leq x_{j_2} \leq \dots x_{j_n}$ .

Per *ordinamento stabile* si richiede inoltre che se  $x_{j_{k'}} = x_{j_{k''}}$  e  $j_{k'} \leq j_{k''}$ , allora  $k' \leq k''$ .

La seguente tabella riassume alcune caratteristiche degli algoritmi di ordinamento che abbiamo considerato:

Algoritmo	Numero confronti	Spazio	Note	Stabile
selectionSort	$\Theta(n^2)$ sempre	$\Theta(1)$	in loco	sì
insertionSort	$\Theta(n^2)$ nel caso peggiore $n - 1$ su array già ordinato	$\Theta(1)$	in loco	sì
bubbleSort	$\Theta(n^2)$ nel caso peggiore $n - 1$ su array già ordinato	$\Theta(1)$	in loco	sì
mergeSort	$\Theta(n \log n)$	$\Theta(n)$	spazio $\Theta(n)$ per array ausiliario più $\Theta(\log n)$ per stack ricorsione	sì
quickSort	$\Theta(n^2)$ nel caso peggiore $\Theta(n \log n)$ caso migliore $\approx 1.39n \log_2 n$ in media	$\Theta(n)$ $\Theta(\log n)$	in loco spazio $\Theta(1)$ più stack ricorsione: $\Theta(n)$ algoritmo base $\Theta(\log n)$ algoritmo migliorato	no
heapSort	$\Theta(n \log n)$	$\Theta(1)$	in loco	no

Il presente materiale integra *ma non sostituisce* il libro di testo consigliato.

Data pubblicazione: 20 novembre 2019

© 2019 Giovanni Pighizzini

Il contenuto di queste pagine è protetto dalle leggi sul copyright e dalle disposizioni dei trattati internazionali. Il titolo ed i copyright relativi alle pagine sono di proprietà dell'autore. Le pagine possono essere riprodotte ed utilizzate liberamente dagli studenti, dagli istituti di ricerca, scolastici e universitari afferenti al Ministero dell'Istruzione, dell'Università e della Ricerca, per scopi istituzionali, non a fine di lucro. Ogni altro utilizzo o riproduzione (ivi incluse, ma non limitatamente a, le riproduzioni a mezzo stampa, su supporti magnetici o su reti di calcolatori) in toto o in parte è vietata, se non esplicitamente autorizzata per iscritto, a priori, da parte dell'autore.

L'informazione contenuta in queste pagine è ritenuta essere accurata alla data della pubblicazione. Essa è fornita per scopi meramente didattici e non per essere utilizzata in progetti di impianti, prodotti, ecc.

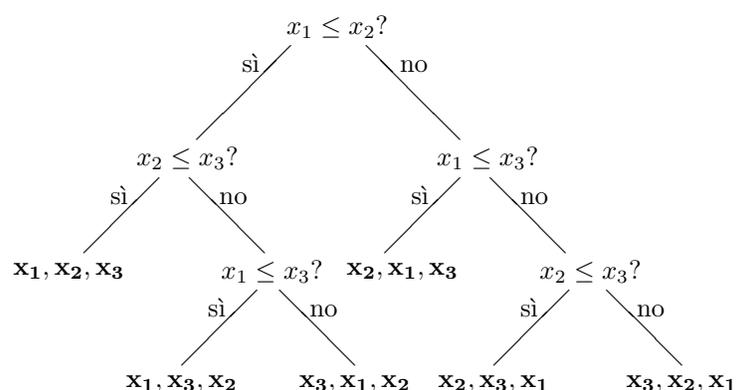
L'informazione contenuta in queste pagine è soggetta a cambiamenti senza preavviso. L'autore non si assume alcuna responsabilità per il contenuto di queste pagine (ivi incluse, ma non limitatamente a, la correttezza, completezza, applicabilità ed aggiornamento dell'informazione). In ogni caso non può essere dichiarata conformità all'informazione contenuta in queste pagine. In ogni caso questa nota di copyright non deve mai essere rimossa e deve essere riportata anche in utilizzi parziali.

## 8.1 Numero minimo di confronti

Dimostreremo ora che ogni algoritmo di ordinamento basato su confronti richiede, nel caso peggiore, un numero di confronti tra chiavi dell'ordine di  $n \log n$ , dove  $n$  è il numero degli elementi da ordinare.

A tale scopo, osserviamo che un confronto tra due chiavi  $x_i$  e  $x_j$  è una domanda del tipo  $x_i \leq x_j?$  con risposta binaria *sì/no*.

Le possibili computazioni di un algoritmo di ordinamento su sequenze di  $n$  elementi possono essere rappresentate mediante un *albero di decisione*, cioè alberi binari in cui ciascun nodo interno rappresenta un'operazione di confronto, con associati due sottoalberi, che dipendono dall'esito di tale operazione, mentre ogni foglia rappresenta una risposta dell'algoritmo, cioè un possibile ordine tra le chiavi. Un esempio di albero di decisione per  $n = 3$  è rappresentato nella seguente figura.



Indipendentemente dalla strategia utilizzata per effettuare i confronti, l'albero dovrà un numero di foglie pari almeno al numero dei possibili ordini tra le chiavi, cioè al numero di possibili permutazioni di  $n$  elementi, che è  $n!$ . Il numero massimo di confronti, utilizzato da una strategia è pari alla profondità dell'albero (3 nel caso della figura).

Si può verificare che la profondità di un albero binario con  $K$  foglie è almeno logaritmica in  $K$  (si pensi a quanto abbiamo visto per gli alberi bilanciati).

Per trovare il numero di confronti necessari nel caso peggiore stimiamo quindi la profondità minima che deve un albero con  $n!$  foglie, calcolando il logaritmo di  $n!$ . Utilizzando l'approssimazione di Stirling  $n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$ , si ottiene:

$$\begin{aligned}
 \log(n!) &\approx \log\left(\sqrt{2\pi n} \left(\frac{n}{e}\right)^n\right) \\
 &= \log\sqrt{2\pi} + \frac{1}{2}\log n + n\log n - n\log e \\
 &= \Theta(n\log n)
 \end{aligned}$$

Possiamo concludere dunque che *ogni* algoritmo di ordinamento basato su confronti richiede nel caso peggiore un numero di confronti tra chiavi dell'ordine di  $n \log n$  per ordinare  $n$  elementi.