8. Algoritmi di Ordinamento Riepilogo e Minimo Numero di Confronti

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

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

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

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 riepiloga alcune caratteristiche degli algoritmi di ordinamento che abbiamo considerato:

Algoritmo	*Numero confronti*	*Spazio*	*Note*	*Stabile*
`selectionSort`	$\Theta(n^2)$ sempre	$\Theta(1)$	in loco	no
`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 $\approx 1.39 n \log_2 n$ 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

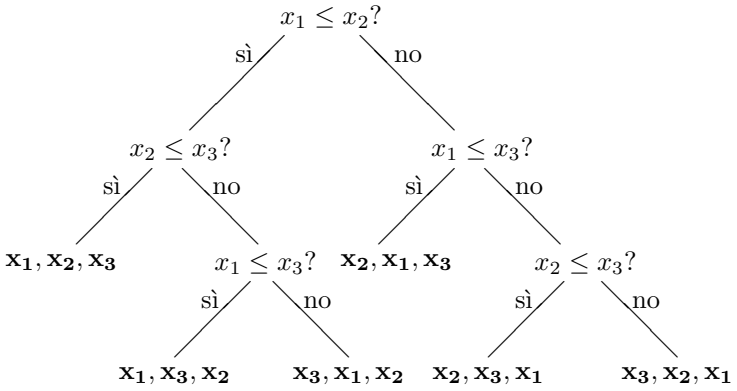
Il presente materiale integra *ma non sostituisce* il libro di testo consigliato.
Data pubblicazione: 7 novembre 2022

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 *si/no*.

Le possibili computazioni di un algoritmo di ordinamento su sequenze di n elementi possono essere rappresentate mediante un *albero di decisione*, cioè un albero binario in cui ciascun nodo interno rappresenta un'operazione di confronto, con associati due sottoalberi, corrispondenti ai successivi confronti che dovranno essere effettuati in base all'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.



Il numero di confronti che una data strategia utilizza nel caso peggiore è pari all'altezza dell'albero corrispondente (uno per ogni nodo interno incontrato partendo dalla radice fino ad arrivare alla foglia più lontana, nel caso della figura precedente 3). Indipendentemente dalla strategia utilizzata per effettuare i confronti, l'albero dovrà avere un numero di foglie pari almeno al numero dei possibile ordini tra le chiavi, cioè al numero di possibili permutazioni di n elementi, che è $n!$. Possiamo dunque affermare che *ogni strategia* deve utilizzare, nel caso peggiore, un numero di confronti pari almeno all'altezza minima degli alberi binari con $n!$ foglie.

Sappiamo che l'altezza di un albero binario con K foglie deve essere almeno logaritmica in K. Per trovare il numero di confronti necessari nel caso peggiore stimiamo dunque il valore minimo dell'altezza di un albero con $n!$ foglie, calcolando il logaritmo di $n!$. Utilizzando l'approssimazione di Stirling $n! \approx \sqrt{2\pi n}(\frac{n}{e})^n$, 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.