5.1 Ordinamento randomizzato per distribuzione

Nel Capitolo 3 abbiamo osservato che l'algoritmo di ordinamento per distribuzione o *quicksort* descritto nel Codice 3.5 ha una complessità che dipende dall'ordine iniziale degli elementi: se la distribuzione iniziale degli elementi è sbilanciata si ha un costo quadratico contro un costo $O(n \log n)$ nel caso di distribuzioni bilanciate.

In questo paragrafo mostreremo un'analisi al caso medio più robusta che risulta essere *indipendente* dall'ordine iniziale degli elementi nell'array e si basa sull'uso della **casualità** per far sì che la distribuzione sbilanciata occorra con una probabilità trascurabile. Concretamente, se gli `n` elementi sono già ordinati, *quicksort* richiede sempre tempo $\Theta(n^2)$ anche se in media il tempo è $O(n \log n)$ se uno considera tutti i possibili array di ingresso. Con la sua versione randomizzata, facciamo in modo che ogni volta che *quicksort* viene eseguito su uno *stesso* array in ingresso, il comportamento non sia deterministico, ma sia piuttosto dettato da scelte casuali (che comunque calcolano correttamente l'ordinamento finale). Ne deriva che nell'analisi di complessità il numero medio di passi non si calcola più su *tutti* i possibili array di ingresso, ma su *tutte* le scelte casuali per un array d'ingresso: è una nozione più forte perché uno stesso array non può essere sempre sfavorevole a *quicksort*, in quanto anche le scelte casuali di quest'ultimo adesso entrano in gioco.

A tal fine, l'unica modifica che occorre apportare all'algoritmo *quicksort* (mostrato nel Codice 5.1) è nella riga 4 e riguarda la scelta del `pivot` che deve avvenire in modo aleatorio, equiprobabile e uniforme nell'intervallo `[sinistra...destra]`: a questo scopo viene utilizzata la primitiva `random()` per generare un valore reale `r` pseudocasuale compreso tra `0` e `1` inclusi, in modo uniforme ed equiprobabile (tale generatore è disponibile in molte librerie per la programmazione e non è semplice ottenerne uno statisticamente significativo, in quanto il programma che lo genera è deterministico). Un tale algoritmo si chiama **casuale** o **randomizzato** perché impiega la casualità per sfuggire a situazioni sfavorevoli, risultando più robusto rispetto a tali eventi (come nel nostro caso, in presenza di un array già in ordine crescente).

ALVIE **Codice 5.1** Ordinamento randomizzato per distribuzione di un array a.

```
1  QuickSort( a, sinistra, destra ):
2                              ⟨pre: 0 ≤ sinistra, destra ≤ n − 1⟩
3    IF (sinistra < destra) {
4      pivot = sinistra + (destra − sinistra) × random();
5      rango = Distribuzione( a, sinistra, pivot, destra );
```

```
6      QuickSort( a, sinistra, rango-1 );
7      QuickSort( a, rango+1, destra );
8    }
```

Teorema 5.1 *L'algoritmo* quicksort *randomizzato impiega tempo ottimo* $O(n \log n)$ *nel caso medio per ordinare* n *elementi.*

Dimostrazione Il risultato della scelta casuale e uniforme del `pivot` è che il valore di `rango` restituito da `Distribuzione` nella riga 5 è anch'esso uniformemente distribuito tra le (equiprobabili) posizioni in `[sinistra ... destra]`. Supponiamo pertanto di dividere tale segmento in quattro parti uguali, chiamate **zone**. In base a quale zona contiene la posizione `rango` restituita nella riga 5, otteniamo i seguenti *due* eventi *equiprobabili*:

- la posizione `rango` ricade nella prima o nell'ultima zona: in tal caso, `rango` è detto essere *esterno*;
- la posizione `rango` ricade nella seconda o nella terza zona: in tal caso, `rango` è detto essere *interno*.

Indichiamo con $T(n)$ il *costo medio* dell'algoritmo *quicksort* eseguito su un array di n elementi. Osserviamo che la media $\frac{x+y}{2}$ di due valori x e y può essere vista come la loro somma pesata con la rispettiva probabilità $\frac{1}{2}$, ovvero $\frac{1}{2}x + \frac{1}{2}y$, considerando i due valori come equiprobabili. Nella nostra analisi, x e y sono sostituiti da opportuni valori di $T(n)$ corrispondenti ai due eventi equiprobabili sopra introdotti. Più precisamente, quando `rango` è esterno (con probabilità $\frac{1}{2}$), la distribuzione può essere estremamente sbilanciata nella ricorsione e, come abbiamo visto, quest'ultima può richiedere fino a $x = T(n-1) + O(n) \leq T(n) + c'n$ tempo, dove il termine $O(n)$ si riferisce al costo della distribuzione effettuata nel Codice 3.6 e $c' > 0$ è una costante sufficientemente grande. Quando invece `rango` è interno (con probabilità $\frac{1}{2}$), la distribuzione più sbilanciata possibile nella ricorsione avviene se `rango` corrisponde al minimo della seconda zona oppure al massimo della terza. Ne deriva una distribuzione dei dati che porta alla ricorsione su circa $\frac{n}{4}$ elementi in una chiamata di `QuickSort` e $\frac{3}{4}n$ elementi nell'altra (le altre distribuzioni in questo caso non possono andare peggio perché sono meno sbilanciate). In tal caso, la ricorsione richiede al più $y = T\left(\frac{n}{4}\right) + T\left(\frac{3}{4}n\right) + O(n) \leq T(n) + c'n$ tempo, dove la costante c' è scelta sufficientemente grande da coprire entrambi gli eventi. Facendo la media pesata di x e y, otteniamo

$$T(n) \leq \frac{1}{2}x + \frac{1}{2}y = \frac{1}{2}\left[T(n) + T\left(\frac{n}{4}\right) + T\left(\frac{3}{4}n\right)\right] + c'n \qquad (5.1)$$

Moltiplicando entrambi i termini nella (5.1) per 2, risolvendo rispetto a $T(n)$ e ponendo $c = 2c'$, otteniamo la relazione di ricorrenza

$$T(n) \leq T\left(\frac{n}{4}\right) + T\left(\frac{3}{4}n\right) + cn \tag{5.2}$$

la cui soluzione dimostriamo essere $T(n) = O(n \log n)$ nel Paragrafo 5.1.1. Il tempo medio è ottimo se contiamo il numero di confronti tra elementi. ❑

5.1.1 Alternativa al teorema fondamentale delle ricorrenze

La relazione di ricorrenza (5.2) non è risolvibile con il teorema fondamentale delle ricorrenze (Teorema 3.1), in quanto non è un'istanza della (4.2). In generale, quando una relazione di ricorrenza non ricade nei casi del teorema fondamentale delle ricorrenze, occorre determinare tecniche di risoluzione alternative come più specificatamente vedremo nella dimostrazione del risultato che segue.

Teorema 5.2 *La soluzione della relazione di ricorrenza (5.2) è* $T(n) = O(n \log n)$.

Dimostrazione Notiamo che il valore $T(n)$ nella (5.2) (livello 0 della ricorsione) è ottenuto sommando a cn i valori restituiti dalle due chiamate ricorsive: quest'ultime, che costituiscono il livello 1 della ricorsione, sono invocate l'una con input $\frac{n}{4}$ e l'altra con input $\frac{3}{4}n$ e, in corrispondenza di tale livello, contribuiscono al valore $T(n)$ per un totale di $c\frac{n}{4} + c\frac{3}{4}n = cn$.

Passando al livello 2 della ricorsione, ciascuna delle chiamate del livello 1 ne genera altre due, per un totale di quattro chiamate, rispettivamente con input $\frac{n}{4^2}$, $\frac{3}{4^2}n$, $\frac{3}{4^2}n$ e $\frac{3^2}{4^2}n$, che contribuiscono al valore $T(n)$ per un totale di $c\frac{n}{4^2} + c\frac{3}{4^2}n + c\frac{3}{4^2}n + c\frac{3^2}{4^2}n = cn$ in corrispondenza del livello 2. Non ci dovrebbe sorprendere, a questo punto, che il contributo del livello 3 della ricorsione sia al più cn (in generale qualche chiamata ricorsiva può raggiungere il caso base e terminare).

Per calcolare il valore finale di $T(n)$ in forma chiusa, occorre sommare i contributi di tutti i livelli. Il livello s più profondo si presenta quando seguiamo ripetutamente il "ramo $\frac{3}{4}$", ovvero viene soddisfatta la relazione $\left(\frac{3}{4}\right)^s n = 1$ da cui deriva che $s = \log_{4/3} n = O(\log n)$. Possiamo quindi limitare superiormente $T(n)$ osservando che ciascuno degli $O(\log n)$ livelli di ricorsione contribuisce al suo valore per al più $cn = O(n)$ e, pertanto, $T(n) = O(n \log n)$. ❑

Intuitivamente, dividere n elementi in proporzione a $\frac{1}{4}$ e $\frac{3}{4}$, invece che a $\frac{1}{2}$ e $\frac{1}{2}$ (come accade nel caso dell'algoritmo di ordinamento per fusione), fornisce comunque una partizione bilanciata perché la dimensione di ciascuna parte differisce dall'altra soltanto per un fattore costante. La proprietà che $T(n) = O(n \log n)$ può essere estesa a una partizione di n in proporzione a $\frac{1}{3}$ e $\frac{2}{3}$ e, in generale, in proporzione a δ e $1 - \delta$ per una qualunque costante $0 < \delta < 1$.

Da quanto discusso finora, possiamo dedurre una linea guida per lo sviluppo di una forma chiusa della soluzione $T(n)$ di una relazione di ricorrenza del tipo

$$T(n) \leq \begin{cases} c_0 & \text{se } n \leq n_0 \\ T(\delta n) + T((1-\delta)n) + cf(n) & \text{altrimenti} \end{cases} \tag{5.3}$$

una qualunque costante $0 < \delta < 1$. Non potendo applicare il teorema fondamentale delle ricorrenze, procediamo per passaggi intermedi con le corrispondenti chiamate ricorsive. La chiamata ricorsiva iniziale (livello 0) con input n contribuisce al valore di $T(n)$ per un totale di $cf(n)$ e dà luogo a due chiamate ricorsive di livello 1, una con input $n' = \delta n$ e l'altra con input $n'' = (1-\delta)n$, dove $n' + n'' = n$. Queste ultime due chiamate contribuiscono per un totale di $cf(n') + cf(n'')$ e inoltre invocano ulteriori due chiamate ricorsive a testa, le quali costituiscono il livello 2 della ricorsione e ricevono in input m_0, m_1, m_2 e m_3 tali che $m_0 + m_1 + m_2 + m_3 = n$. Dovrebbe essere chiaro a questo punto che queste chiamate contribuiscono per un totale di $cf(m_0) + cf(m_1) + cf(m_2) + cf(m_3)$ e inoltre invocano ulteriori due chiamate ricorsive a testa. La forma chiusa per un limite superiore della (5.3) è data dalla somma dei termini noti

$$T(n) \leq cf(n) + [cf(n') + cf(n'')] + [cf(m_0) + cf(m_1) + cf(m_2) + cf(m_3)] + \cdots$$

la cui valutazione dipende dal tipo della funzione $f(n)$: per esempio, abbiamo visto che se $f(n) = n$, allora otteniamo $T(n) = O(n \log n)$. Siccome alcune chiamate potrebbero terminare prima, abbiamo che la somma dei termini noti rappresenta un limite superiore.

Esercizio svolto 5.1 Consideriamo la funzione `QuickSelect` riportata nel Codice 3.7 del Capitolo 3: rendiamola randomizzata operando la scelta del `pivot` come avviene nella riga 4 del Codice 5.1. Mostrare che la media del costo nel caso della `QuickSelect` è $O(n)$.

Soluzione L'equazione di ricorrenza per il costo al caso medio è costruita in modo simile all'equazione (5.1), con la differenza che conteggiamo una sola chiamata ricorsiva (la più sbilanciata) ottenendo $T(n) \leq \frac{1}{2}\left[T(n) + T\left(\frac{3}{4}n\right)\right] + c'n$.