

Problemi di Algoritmica 2

Jacopo Notarstefano

8 dicembre 2012

If you're having girl problems
I feel bad for you son
I got 99 problems
but a bitch ain't one.

Jay-Z

Problema 1

Ordinamento in memoria esterna

Problema. *Nel modello EMM (external memory model), mostrate come implementare il k -way merge, ossia la fusione di n sequenze individualmente ordinate e di lunghezza totale N , con costo I/O di $O(\frac{N}{B})$ dove B è la dimensione del blocco. Minimizzare e valutare il costo di CPU. Analizzare il costo del merge (I/O complexity, CPU complexity) che utilizza tale k -way merge.*

Problema 2

Limite inferiore per la permutazione

Problema. *Estendere l'argomentazione usata per il limite inferiore del problema dell'ordinamento in memoria esterna a quello della permutazione: dati N elementi e_1, e_2, \dots, e_N e un array π contenente una permutazione degli interi in $[1, 2, \dots, N]$, disporre gli elementi secondo la permutazione in π . Dopo tale operazione, la memoria esterna deve contenerli nell'ordine $e_{\pi[1]}, e_{\pi[2]}, \dots, e_{\pi[N]}$.*

Problema 3

Permutazione in memoria esterna

Problema. *Dati due array A e C di N elementi, dove A è l'input e C una permutazione di $\{0, 1, \dots, n-1\}$, descrivere e analizzare nel modello EMM un algoritmo ottimo per costruire $A[C[i]]$ per $0 \leq i \leq n-1$.*

Problema 4

Multi-selezione in memoria esterna

Problema. *Scrivere tutti i passaggi dell'analisi del costo e della correttezza dell'algoritmo di multi-selezione visto a lezione.*

Vogliamo esibire un algoritmo che selezioni un certo numero di pivot da un insieme S di cardinalità N in modo tale che la distanza fra pivot consecutivi sia piccola. Il nostro scopo sarà usare questo algoritmo per costruire un analogo del QUICKSORT in memoria esterna, così come la k -way merge ci ha permesso di costruire l'algoritmo di MERGE SORT in memoria esterna.

Ci potremmo aspettare di dover trovare m pivot, in analogia a quanto facciamo per la Merge. In realtà è sufficiente determinarne \sqrt{m} . Diamo di seguito l'algoritmo e due lemmi. Nel primo dimostreremo il costo lineare, nel secondo la correttezza dell'algoritmo.

Algoritmo 1 Multi-selezione in memoria esterna

- 1: Carico e ordino in memoria principale $\frac{N}{M}$ run di M elementi ciascuno.
 - 2: Da ogni run seleziono un elemento ogni $\frac{\sqrt{m}}{4}$ e chiamo G (elementi verdi) l'insieme degli elementi selezionati.
 - 3: Uso l'algoritmo dei cinque autori \sqrt{m} volte per selezionare in G un elemento ogni $\frac{4N}{m}$ e chiamo R (elementi rossi) l'insieme degli elementi selezionati.
 - 4: Ritorno R .
-

Lemma 1 (Costo). *L'algoritmo compie $O(n)$ I/O.*

Dimostrazione. *La prima riga dell'algoritmo comporta soltanto di scandire tutti gli elementi: l'ordinamento di ogni run viene infatti svolto in memoria principale, e non comporta ulteriori I/O. Anche la seconda riga consiste in una scansione di tutti gli elementi. Per stimare il numero di I/O della terza riga sfruttiamo invece il fatto che ogni esecuzione dell'algoritmo dei cinque autori comporta una scansione di tutti gli elementi. Abbiamo dunque \sqrt{m} scansioni di $|G|$ elementi, perciò:*

$$\sqrt{m} \cdot O\left(\frac{|G|}{B}\right) = \sqrt{m} \cdot O\left(\frac{4N}{B\sqrt{m}}\right) = O\left(\frac{4N}{B}\right) = O(n),$$

dove la prima eguaglianza discende dal fatto che, avendo selezionato un elemento ogni $\frac{\sqrt{m}}{4}$, la cardinalità di G è $\frac{4N}{\sqrt{m}}$. Ogni riga contribuisce quindi $O(n)$ I/O, da cui la tesi.



Figura 4.1: Ogni riga orizzontale rappresenta un run ordinato, e i cerchietti gli elementi di ogni run. Cerchietti rossi e verdi rappresentano rispettivamente gli elementi di R e G , cerchietti neri i restanti elementi senza colore. Sono inoltre raffigurati in giallo i bordi definiti dalla posizione degli elementi rossi nei quali possiamo avere elementi senza colore compresi fra un rosso e un verde.

Lemma 2 (Correttezza). *Il numero di elementi di S compresi fra due elementi di R è minore di $\frac{3}{2} \frac{N}{\sqrt{m}}$.*

Dimostrazione. *Vogliamo dunque stimare il numero di elementi di S compresi fra due generici elementi rossi r_1 e r_2 . Possiamo dividerli in tre categorie:*

- *Gli elementi verdi compresi fra i due elementi rossi r_1 e r_2 .*
- *Gli elementi senza colore compresi fra due elementi verdi.*
- *Gli elementi senza colore compresi fra un rosso e un verde.*

I primi sono facilmente maggiorati da $X = \frac{4N}{m}$: nella terza riga dell'algoritmo abbiamo infatti scelto un rosso ogni $\frac{4N}{m}$ elementi verdi.

I secondi sono invece maggiorati da $Y = \frac{N}{\sqrt{m}} - \frac{4N}{m}$. Per la seconda riga dell'algoritmo abbiamo infatti $\frac{\sqrt{m}}{4} - 1$ elementi senza colore fra due verdi consecutivi appartenenti allo stesso run, avendo scelto un verde ogni $\frac{\sqrt{m}}{4}$ elementi. Per la terza riga abbiamo al più $\frac{4N}{m}$ elementi verdi fra r_1 e r_2 , dunque al più $\frac{4N}{m}$ coppie di elementi verdi consecutivi nello stesso run. Ma allora il numero cercato è stimato dal prodotto, e quindi da:

$$\frac{4N}{m} \left(\frac{\sqrt{m}}{4} - 1 \right) = \frac{N}{\sqrt{m}} - \frac{4N}{m}.$$

I terzi sono invece maggiorati da $Z = \frac{n}{2\sqrt{m}} - \frac{2n}{m}$. Per vedere questo abbiamo bisogno della figura. Osserviamo infatti che la posizione dei due elementi rossi definisce un paio di "bordi" (raffigurati in giallo in figura) in cui è possibile trovare gli elementi del terzo tipo. Ognuno di questi bordi può contenere al più $\frac{\sqrt{m}}{4} - 1$ elementi, perché se ne contenesse di più conterrebbe sicuramente due verdi, quindi elementi compresi fra due verdi. Per ognuno degli $\frac{N}{M}$ run abbiamo insomma 2 bordi contenenti al più $\frac{\sqrt{m}}{4} - 1$ elementi, perciò possiamo stimare il numero di elementi del terzo tipo con il loro prodotto, cioè:

$$2 \cdot \frac{n}{m} \cdot \left(\frac{\sqrt{m}}{4} - 1 \right) = \frac{n}{2\sqrt{m}} - \frac{2n}{m},$$

dove abbiamo usato che $\frac{N}{M} = \frac{n}{m}$ essendo $n = \frac{N}{B}$ e $m = \frac{M}{B}$.

Di conseguenza il numero totale degli elementi compresi fra r_1 e r_2 è maggiorato da:

$$\begin{aligned} X + Y + Z &= \frac{4N}{m} + \frac{N}{\sqrt{m}} - \frac{4N}{m} + \frac{n}{2\sqrt{m}} - \frac{2n}{m} \\ &\leq \frac{N}{\sqrt{m}} + \frac{n}{2\sqrt{m}} \end{aligned}$$

nella quale abbiamo cancellato i termini uguali di segno opposto e un termine negativo, ottenendo un'ulteriore maggiorazione. Abbiamo inoltre:

$$\begin{aligned} \frac{N}{\sqrt{m}} + \frac{n}{2\sqrt{m}} &\leq \frac{N}{\sqrt{m}} + \frac{N}{2\sqrt{m}} \\ &= \frac{3}{2} \frac{N}{\sqrt{m}} \end{aligned}$$

essendo $n = \frac{N}{B}$, da cui la tesi.