Esercizi sugli aspetti computazionali della Trasformata Discreta di Fourier

Al fine di invogliare gli studenti a esercitarsi con il materiale visto a lezione, i seguenti esercizi sono proposti senza soluzioni.

Exercise 1.1 Descrivere e analizzare un algoritmo per moltiplicare due polinomi di grado limitato da m e n (con $m \ll n$) in tempo $O(n \log m)$. (*Suggerimento:* Si determini una scomposizione del polinomio di limite di grado maggiore in polinomi di grado limitato da m moltiplicati per una potenza dell'indeterminata.)

The correctness of UNEVEN_LIN_CONV immediately follows from the above discussion. As for its running time, we have that the n/m calls to LIN_CONV account for a total of $\Theta((n/m)m \log m) = \Theta(n \log m)$ arithmetic operations between complex numbers, while the additions needed to obtain the final coefficients add a total of $(n/m)(2m-1) = \Theta(n)$ scalar sums, for a total time $T(m,n) = \Theta(n \log m)$.

Exercise 1.2 Si consideri l'operazione di convoluzione lineare $\boldsymbol{u} \star \boldsymbol{x}$, con entrambe le sequenze di lunghezza n e con $\boldsymbol{u} = (1, 1, \ldots, 1)$. Si sviluppi un algoritmo iterativo per eseguire tale operazione in tempo $O(n)$. (*Suggerimento:* Si determini una ricorrenza sulle componenti della convoluzione.)

Exercise 1.3 Siano k e n potenze di due, con $1 \leq k \leq n$. Un vettore $\boldsymbol{x} = (x_0, x_1, \ldots, x_{n-1})$ si dice (n,k)-*padded* se $x_i = 0$ per $k \leq i \leq n-1$.

Punto 1. Si fornisca lo pseudocodice e si provi la correttezza di un algoritmo PADDED_FFT$(\boldsymbol{x}, k)$, ottenuto modificando l'algoritmo ricorsivo FFT visto in classe, per il caso di vettori (n,k)-padded. (*Suggerimento*: la ricorsione deve essere in funzione del parametro k, con caso di base $k = 1$.)

Punto 2 Si imposti e si risolva la ricorrenza sul numero di operazioni aritmetiche tra scalari complessi $T_{\mathrm{PF}}(n,k)$ effettuate dall'algoritmo sviluppato al Punto 1.

Exercise 1.4 Siano k, n potenze di due, con $1 \leq k < n$. Dato un vettore $\boldsymbol{A} \in \boldsymbol{C}^k$, si consideri il vettore a n componenti $\boldsymbol{B} \in \boldsymbol{C}^n$ ottenuto concatenando $\boldsymbol{A}$ con se stesso per n/k volte:

$$\boldsymbol{B} = (\underbrace{\boldsymbol{A}|\boldsymbol{A}|\ldots|\boldsymbol{A}}_{n/k \text{ volte}}).$$

Si discuta di come ottenere $DFT_n(\boldsymbol{B})$ in tempo $\Theta((n/k)\log(n/k))$. Nell'analisi di complessità, si considerino di costo unitario e non nullo solo le operazioni aritmetiche tra numeri complessi. (*Suggerimento:* Si ricordi che $\boldsymbol{B}$ ricalca la struttura della trasformata di un vettore (n,k)-sparso (vista a lezione) e si utilizzi la relazione tra DFT_n^{-1} e DFT_n^R.)

Exercise 1.5

Punto 1. Si fornisca un algoritmo divide-and-conquer per ottenere la potenza k-sima z^k di un dato numero complesso z eseguendo $\Theta(\log k)$ moltiplicazioni complesse (*Suggerimento:* Si ottenga z^k in funzione di $z^{\lfloor k/2 \rfloor}$.)

Punto 2. Siano ora $k, n > 0$. Dato il vettore dei coefficienti $\boldsymbol{a} \in \boldsymbol{C}^n$ di un polinomio $A(x)$ di grado limitato da n , si fornisca un algoritmo che restituisca il vettore dei coefficienti del polinomio $p(x) = (A(x))^k$.

Exercise 1.6 Sia n una potenza di due. Usando la FFT, progettare e analizzare un algoritmo che, dato in ingresso n, produca in uscita il vettore $C[k] = \binom{n-1}{k}$, con $0 \leq k \leq n-1$ eseguendo $O(n \log n)$ operazioni tra scalari complessi. (*Suggerimento:* Si consideri l'espansione di Newton del polinomio $p(x) = (x+1)^{n-1} \ldots$.)

Exercise 1.7 Sia $n > 0$. Per $0 \leq j \leq n-1$, sia $\boldsymbol{F}_n^j$ la j-sima colonna della matrice di Fourier di ordine n. Si determinino le componenti di $DFT_n(\boldsymbol{F}_n^j)$ e si fornisca lo pseudocodice di un algoritmo per calcolarle che non esegue alcuna operazione aritmetica tra scalari complessi.

Exercise 1.8 Sia n una potenza di due e siano dati due insiemi $A, B \subseteq \{0, 1, 2, \ldots n-1\}$ rappresentati in ingresso per mezzo dei loro vettori caratteristici (cioè vettori binari $\boldsymbol{a}$ e $\boldsymbol{b} \in \{0,1\}^n$ con $a_i = 1 \Leftrightarrow i \in A$ e $b_j = 1 \Leftrightarrow j \in B$). Si voglia calcolare, per ogni intero i, $0 \leq i \leq 2n-2$, il numero z_i di coppie distinte $(a, b) \in A \times B$ tali che $a + b = i$.

Punto 1. Si riconduca il problema al calcolo di una opportuna convoluzione lineare.

Punto 2. Si fornisca lo pseudocodice dell'algoritmo CARTESIAN_SUM($\boldsymbol{a}, \boldsymbol{b}$) che risolve il problema in tempo $O(n \log n)$.

Exercise 1.9

Punto 1. Sia $n > 0$. Si dimostri rigorosamente che per ogni vettore $\boldsymbol{x} = (x_0, x_1, \ldots, x_{n-1}) \in \boldsymbol{C}^n$ si ha:

$$(F_n)^2 \boldsymbol{x} = n \cdot \boldsymbol{x}^R,$$

dove $(F_n)^2 = F_n \times F_n$ denota il quadrato della matrice di Fourier di ordine n e $\boldsymbol{x}^R = (x_0, x_{n-1}, x_{n-2} \ldots, x_1)$ denota il *reverse* del vettore $\boldsymbol{x}$.

Punto 2. Utilizzando la relazione provata al Punto 1, si fornisca lo pseudocodice e si analizzi un algoritmo *divide-and-conquer* KFT($\boldsymbol{x}, k$) che, dati in ingresso un vettore complesso $\boldsymbol{x} \in \boldsymbol{C}^n$, con n potenza di due e k un generico intero positivo o nullo, restituisca $\boldsymbol{y} = (F_n)^k \boldsymbol{x}$ eseguendo $T(n, k) = O(n(k + \log n))$ operazioni aritmetiche tra scalari complessi.
(*Suggerimento:* si osservi che per $k \geq 2$ vale $(F_n)^k \boldsymbol{x} = (F_n)^2((F_n)^{k-2}\boldsymbol{x}) \ldots$)