

## Esercizio 1

Poiché  $m$  è un valore costante, ovviamente anche  $T(m)$  lo è. E' facile notare che il numero di chiamate ricorsive sarà pari a  $\lceil n/m \rceil$ . Ognuna di queste chiamate costerà  $T(m) + 1$ , quindi ancora un valore costante. Possiamo quindi supporre un limite superiore e inferiore  $\Theta(n)$ .

$O(n)$  :

Come passo induttivo, supponiamo di aver provato per tutti i casi  $n' < m$  che  $T(n') \leq cn'$ ; vogliamo ora provare che  $\exists c > 0 : T(n) \leq cn$ .

$$\begin{aligned} T(n) &= T(m) + T(n-m) + 1 \\ &\leq 1 + cn - cm + 1 \\ &\leq cn - cm + 2 \\ &\stackrel{?}{\leq} cn \end{aligned}$$

La disequazione è vera per  $c \geq 2/m$ .

Come case base, consideriamo tutti i casi per cui  $1 \leq n \leq m$ :

$$T(i) = 1 \leq c \cdot i \Leftrightarrow c \geq \frac{1}{i}$$

Il valore massimo incontrato nelle disequazioni è 1; per soddisfarle tutte, basterà porre  $c = 1$ .

$\Omega(n)$  :

Come passo induttivo, supponiamo di aver provato per tutti i casi  $n' < m$  che  $T(n') \geq cn'$ ; vogliamo ora provare che  $\exists c > 0 : T(n) \geq cn$ .

$$\begin{aligned} T(n) &= T(m) + T(n-m) + 1 \\ &\geq 1 + cn - cm + 1 \\ &\geq cn \end{aligned}$$

L'ultima disequazione è banalmente vera per ogni  $c \leq 2/m$ .

Come caso base, consideriamo tutti i casi per cui  $1 \leq i \leq m-1$ :

$$T(i) = 1 \geq c \cdot i \Leftrightarrow c \leq \frac{1}{i}$$

Il valore  $\frac{1}{m}$  è il più piccolo fra tutte le disequazioni trovate; per soddisfarle tutte, compresa quella derivante dal passo induttivo, basterà porre  $c = 1/m$ .

## Esercizio 2

Tramite gli algoritmi che risolvono il problema della selezione visti a lezione, possiamo individuare la mediana in tempo (atteso)  $O(n)$ . Una volta individuata la mediana, possiamo costruire un vettore di appoggio che contenga la differenza fra gli elementi di  $A$  e la mediana. Utilizzando di nuovo l'algoritmo della selezione, possiamo individuare la  $k$ -esima differenza rispetto alla mediana. Utilizzando questo valore, possiamo individuare tutti i valori la cui distanza è inferiore.

---

**kmediana(integer[] A, integer n, integer k)**

---

```
integer m ← selezione(A, ⌊n/2⌋)
integer[] B ← new integer[1...n]
for i ← 1 to n do
  B[i] = |A[i] - m|
integer d ← selezione(B, k)
for i ← 1 to n do
  if |A[i] - m| ≤ d then
    print A[i]
```

---

### Esercizio 3

Un possibile grafo è quello rappresentato in Figura 1. L'albero di copertura è minimo perchè contiene tutti e soli gli archi con peso 2, mentre tutti gli altri archi hanno peso maggiore.

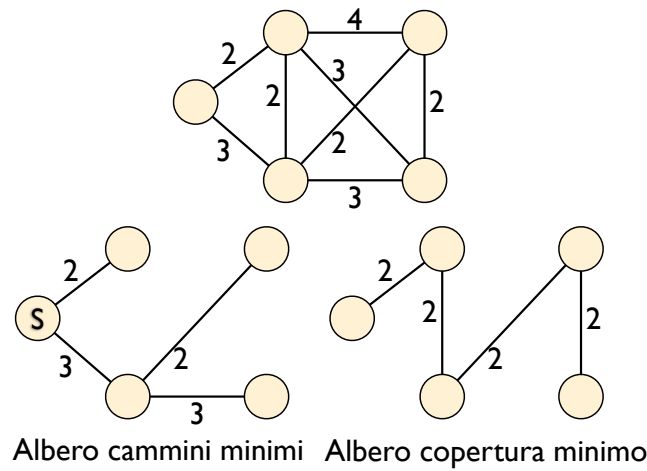


Figura 1: Un grafo che risolve l'esercizio 3

### Esercizio 4

Sia  $C[i]$  il numero di modi in cui è possibile comporre la stringa  $X[i \dots n]$  a partire da stringhe primitive.  $C[i]$  può essere calcolato semplicemente confrontando la stringa  $X$  a partire dalla posizione  $i$ -esima con ognuna delle stringhe primitive  $s \in S$ , e sommando i valori  $C[i + |s|]$  qualora le stringhe coincidano.  $C[n + 1]$  è uguale a 1, perchè esiste un solo modo per comporre una stringa vuota (la stringa a partire da  $n + 1$  è vuota).

$$C[i] = \begin{cases} \sum_{s \in S \wedge \text{check}(X, s, i)} C[i + |s|] & 1 \leq i \leq n \\ 1 & i = n + 1 \end{cases}$$

Tradotto con memoization, il codice diventa:

---

```
count(integer[] X, integer n, SET S, integer[] C, integer i)
```

---

```

if  $C[i] = n + 1$  then
   $\perp$  return 1
if  $C[i] = \perp$  then
   $C[i] \leftarrow 0$ 
  foreach  $s \in S$  do
    if  $\text{check}(X, s, i)$  then
       $C[i] \leftarrow \text{count}(X, n, S, C, i + |s|)$ 
  return  $C[i]$ 

```

---

La chiamata iniziale è  $\text{count}(X, n, S, C, 1)$ . Detto  $m = \sum_{s \in S} |s|$ , la complessità è  $O(mn)$ .