23/11

[PDF]

SLIDE (REVEAL.JS)

↓

MERCURIAL (GIT)

HG CLONE (HTML)

...

APPUNTI

(ESERCIZIO VALUTATO)

3 PUNTI

Esercizio 2 (9 punti) Data una stringa $X = x_1, x_2, \ldots, x_n$, si consideri la seguente quantità $\ell(i, j)$, definita per $1 \le i \le j \le n$:

$$\ell(i,j) = \begin{cases} 1 & \text{se } i = j \\ 2 & \text{se } i = j-1 \\ 2 + \ell(i+1, j-1) & \text{se } (i < j-1) \text{ e } (x_i = x_j) \\ \sum_{k=i}^{j-1} (\ell(i,k) + \ell(k+1,j)) & \text{se } (i < j-1) \text{ e } (x_i \neq x_j). \end{cases}$$

1. Scrivere una coppia di algoritmi INIT_L(X) e REC_L(X, i, j) per il calcolo memoizzato di $\ell(1, n)$.
2. Si determini la complessità *al caso migliore* $T_{\text{best}}(n)$, supponendo che le uniche operazioni di costo unitario e non nullo siano i confronti tra caratteri.

INIT_L / REC_L

↓

N = LENGTH (X)

LUNGH. ARR.

$X \rightarrow i \leq J \leq m$

IF N = 1 RETURN 1 → I = J

IF N = 2 RETURN 2 → I = J - 1

FOR I = i TO N - 1 DO

L (i , 0) = 1

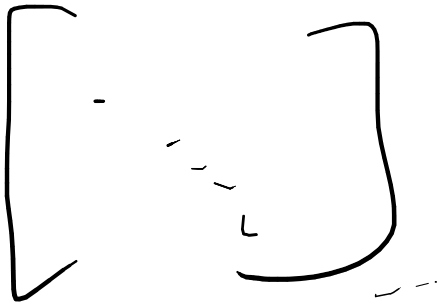
→ UN INDICE SOLO

FOR I = 1 TO N - 1 DO

L[I, I] = 1

L[I, I + 1] = 2

INIZIAL.



→ INIT (INIZIALIZZA LA MATRICE)

RSC

↓

[CALCOLO SOLUZIONE]

GIÀ RISOLTO

↓

FOR I TO N − 2 (N − 1)

FOR J = I + 2 TO N

L[I, J] = 0 (RISOLVO PER RSC)

REC

$$\ell(i,j) = \begin{cases} 1 & \text{se } i = j \\ 2 & \text{se } i = j-1 \\ 2 + \ell(i+1, j-1) & \text{se } (i < j-1) \text{ e } (x_i = x_j) \\ \sum_{k=i}^{j-1} (\ell(i,k) + \ell(k+1,j)) & \text{se } (i < j-1) \text{ e } (x_i \neq x_j). \end{cases}$$

1. Scrivere una coppia di algoritmi INIT_L(X) e REC_L(X, i, j) per il calcolo memoizz
2. Si determini la complessità *al caso migliore* $T_{\text{best}}(n)$, supponendo che le uniche oper unitario e non nullo siano i confronti tra caratteri.

Soluzione:

1. Pseudocodice:

```
INIT_L(X)
 n <- length(X)
 if n = 1 then return 1
 if n = 2 then return 2
 for i=1 to n-1 do
   L[i,i] <- 1
   L[i,i+1] <- 2
 L[n,n] <- 1
 for i=1 to n-2 do
   for j=i+2 to n do
     L[i,j] <- 0
 return REC_L(X,1,n)
```

GREEDY

SOL. ATTIVITÀ

HUFFMAN

GREEDY → FACCIO LA SCELTA GIUSTA

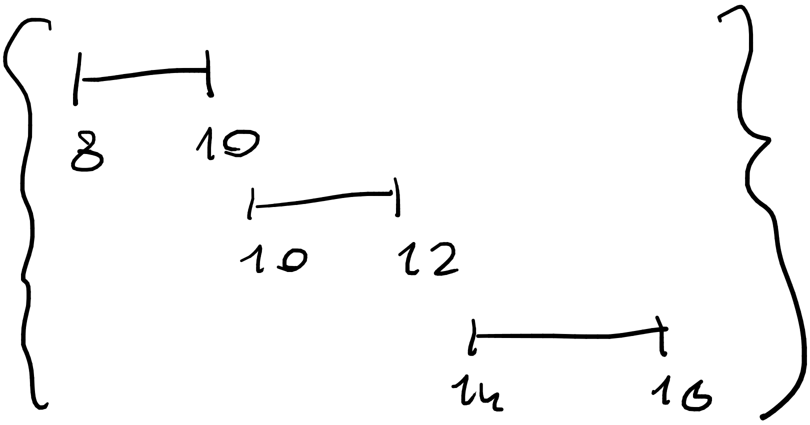
PROG. DINAMICA [ ] SOLUZIONE MIGLIORE

MAX/MIN

SELEZIONE ATTIVITÀ COMPATIBILI

→ ORARIO LEZIONI

A = ATTIVITÀ



PIÙ ATTIVITÀ POSSIBILI SENZA SOVRAPPOSIZIONI

COMPATIBILI

GREEDY-SEL(S, f)

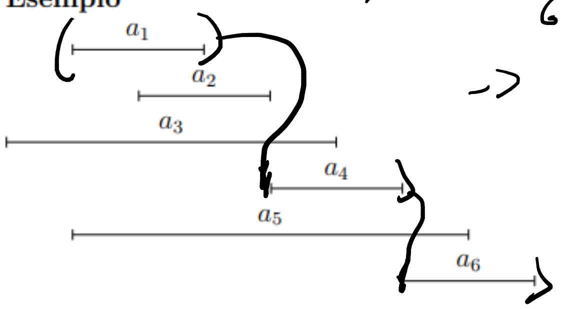
```
n = S.length
A = {a_1}                 // ULTIMO
last = 1 // indice dell'ultima attività selezionata
for m = 2 to n
    if s_m ≥ f_last
        A = A ∪ {a_m}
        last = m
return A
```

→ RICORDATI IL GREEDY (INDICE)

IL TEMPO DI FINE È > RISPETTO A T. DI INIZIO

← GREEDY = "ATTACCA" LA SCELTA

Esempio



$A = a_1$	$A = a_1, a_4$	$A = a_1, a_4, a_6$
$last = 1$	$last = 4$	$last = 6$

GREEDY = → PRENDI LA PRIMA (O UNA ALL'INIZIO) E "ATTACCA" L'ATTIVITÀ CHE INIZIA PER PRIMA SUBITO DOPO

a_1 (CUT AND PASTE)

+ a_{opt} $]_x$

SCELTA GREEDY

GREEDY ↕ TIPO DI PROBLEMA

[2-1 BOXING]

$\ldots = \sum 1$

(PARTIZIONI CHE SOMMANO AD 1)

[0.1 0.3 0.02 ...]

(1)

2 PARTIZIONI LA CUI SOMMA È 1

0.8 | 0.3 | 0.2 | 0.4 → 1

→ 0.1 | 0.2 | 0.3 | 0.8

→ "IMPRINTING" CODICE GREEDY

GREEDY-SEL(S, f)

$n = S.length$
$A = \{a_1\}$ → PRIMA
last $= 1$ // indice dell'ultima attività selezionata
for $m = 2$ **to** n
  if $s_m \geq f_{last}$
    $A = A \cup \{a_m\}$
    last $= m$
return A

IF = SCELTA GREEDY

2-1 BOXING → FIRST LAST

Esercizio 2 (10 punti) Dato un insieme di n numeri reali positivi e distinti $S = \{a_1, a_2, \ldots, a_n\}$, con $0 < a_i < a_j < 1$ per $1 \leq i < j \leq n$, un *(2,1)-boxing* di S è una partizione $P = \{S_1, S_2, \ldots, S_k\}$ di S in k sottoinsiemi (cioè, $\bigcup_{j=1}^{k} S_j = S$ e $S_r \cap S_t = \emptyset, 1 \leq r \neq t \leq k$) che soddisfa inoltre i seguenti vincoli:

$$|S_j| \leq 2 \quad \text{e} \quad \sum_{a \in S_j} a \leq 1, \quad 1 \leq j \leq k.$$

In altre parole, ogni sottoinsieme contiene al più due valori la cui somma è al più uno. Dato S, si vuole determinare un (2,1)-boxing che minimizza il numero di sottoinsiemi della partizione.

1. Scrivere il codice di un algoritmo greedy che restituisce un (2,1)-boxing ottimo in tempo lineare. (Suggerimento: si creino i sottoinsiemi in modo opportuno basandosi sulla sequenza ordinata.)
2. Si enunci la proprietà di scelta greedy per l'algoritmo sviluppato al punto precedente e la si dimostri, cioè si dimostri che esiste sempre una soluzione ottima che contiene la scelta greedy.

A = [0.7 / 0.3 / 0.8 / 0.2]

GREEDY → SORT

0.2 + 0.8 = 1

[0.2 / 0.3 / 0.7 / 0.8]

(2 MOSSE)

NON OTTIMA

SCELTE GREEDY NON OTTIMALE] → CONTRO ESEMPIO

{ 0.2 0.3 0.4 }

NO!

LA SOMMA È 1 SE SCELGO SUBITO DALLA PRIMA?

1. L'idea è provare ad accoppiare il numero più piccolo (a_1) con quello più grande (a_n). Se la loro somma è al massimo 1, allora $S_1 = \{a_1, a_n\}$, altrimenti $S_1 = \{a_n\}$. Poi si procede analogamente sul sottoproblema $S \setminus S_1$.

Esempio:
1 2 3 4 5 6 7

P = (7, 6, 5, 4, 3, 2, 1)

```
(2,1)-BOXING(S)
n <- |S|
P <- empty_set
first <- 1
last <- n
while (first <= last)
   if (first < last) and a_first + a_last <= 1 then
      P <- P U {{a_first, a_last}}
      first <- first + 1
   else
      P <- P U {{a_last}}
   last <- last - 1
return P
```

Inizializziamo l'insieme, la partizione e gli estremi.

Si considera un ciclo per cui "first" è <= "last" (perché scansioniamo gli estremi, come de

Se l'estremo inf. è < dell'estremo sup.
non abbiamo ancora salvato nulla in P (non abbiamo estr. inf)
allora salvo in P sia l'estremo inf che l'estremo sup
e incremento first. Salvandolo una volta sola, so che la somma è :

Altrimenti,
salvo solo l'estremo superiore migliore, la cui somma
è sempre >= 1

[0.2 ... 0.3 0.8

1

[HUFFMAN → GREEDY]

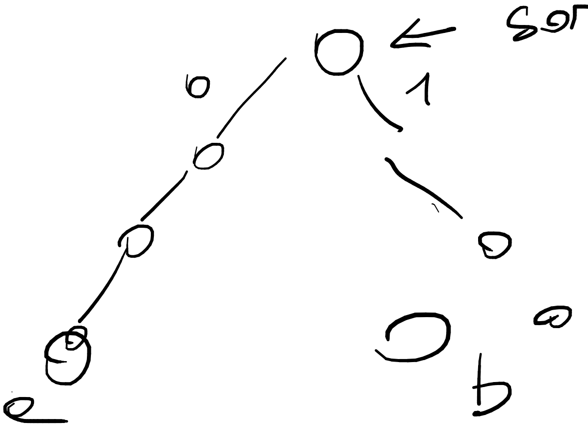
COMPRESSIONE

→ aaaa bbccdd

CARATTERE (FREQUENZA)

ALBERO

a	b	c	d	e	f	g
3	8	7	12	6	23	21



GREEDY

→ ORDINO I CAR.
PER FREQUENZA

→ PARTO DAI MENO
FREQUENTI
E COSTRUISCO L'ALBERO

$\underbrace{aaa}_{0}b\underbrace{bbbbbbbbb}_{0}$ → RAPPR. IN ZIP

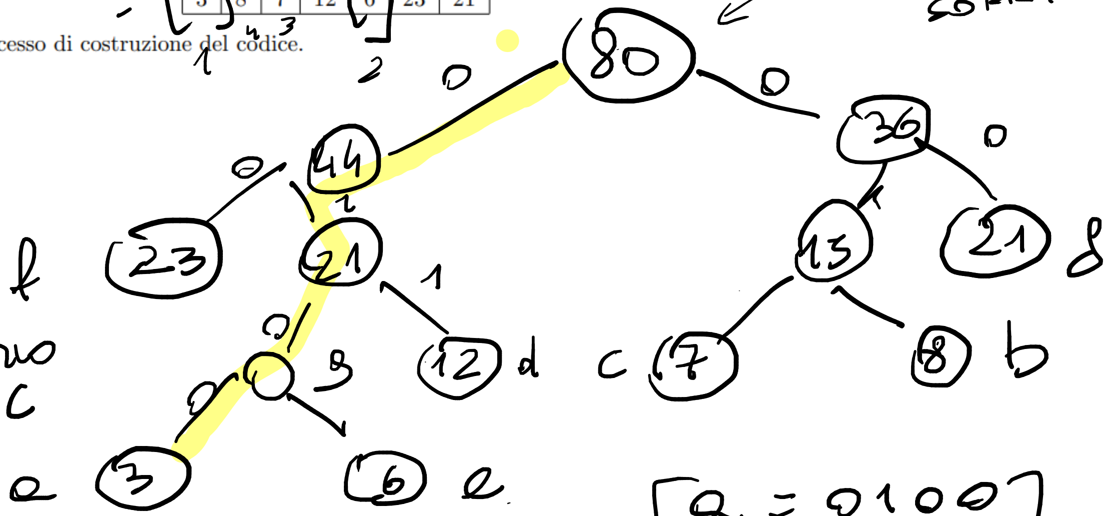
0 = LEFT
1 = RIGHT

Domanda 44 Indicare il codice prefisso ottenuto utilizzando l'algoritmo di Huffmann per l'alfabeto $\{a, b, c, d, e, f, g\}$, supponendo che ogni simbolo appaia con le seguenti frequenze.

a	b	c	d	e	f	g
3	8	7	12	6	23	21

Spiegare il processo di costruzione del codice.

SOLO NODO SOPRA

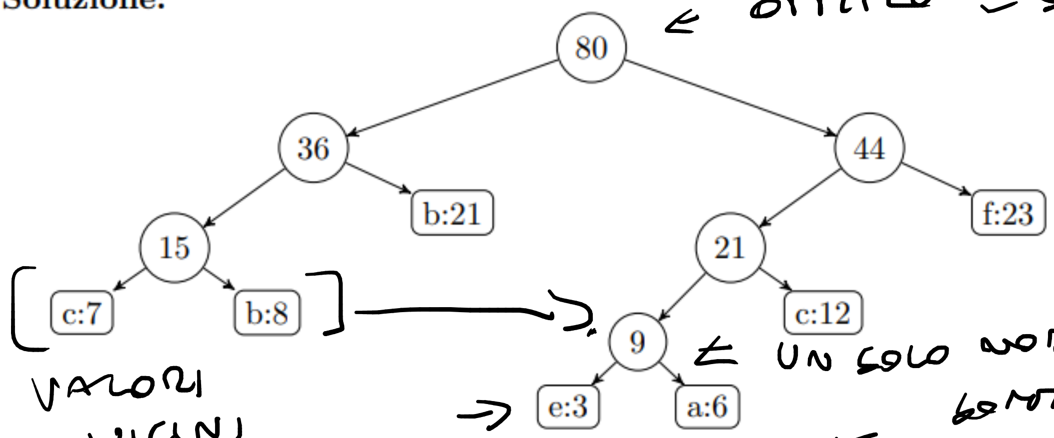


C STESSO LIVELLO PER b E C PERCHÉ HANNO VAL. SIMILI

[a = 0100
g = 00] ...

Soluzione:

OTTIMO = SOPRA AD UN SOLO NODO SOPRA CON LO STESSO VALORE



UN SOLO NODO SOPRA

VALORI VICINI PER EVITARE CODIFICA CON PREFISSI

MENO FREQUENTI