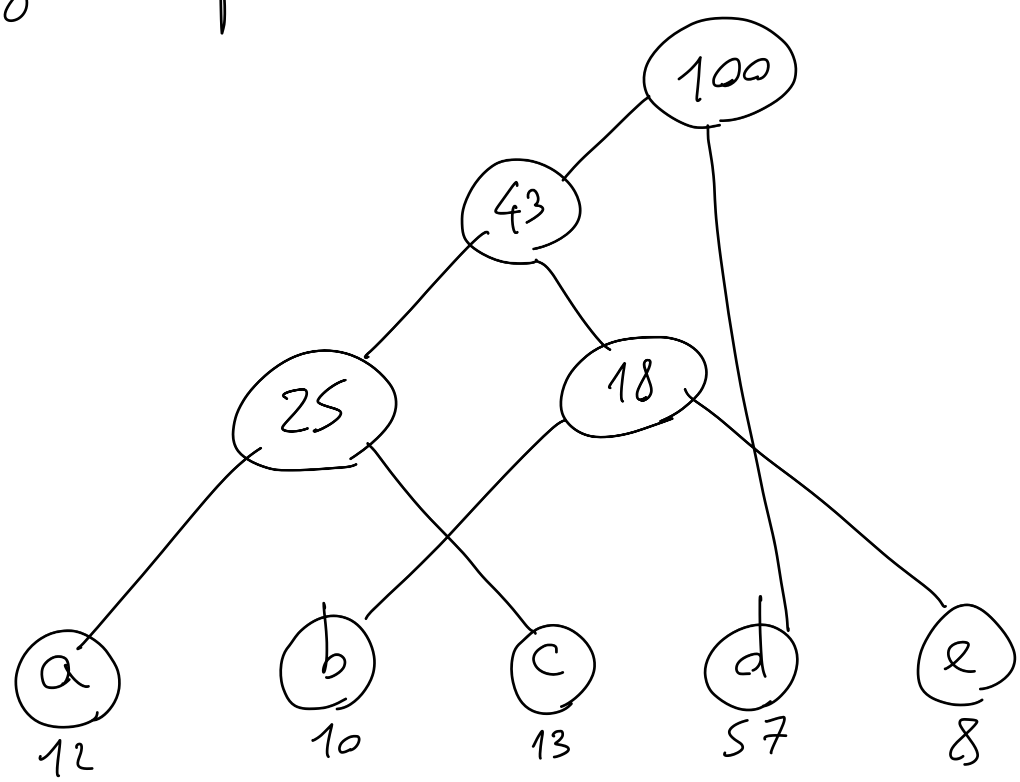
Esercitazioni

Domanda (5 punti): Indicare, in forma di albero binario, il codice prefisso ottenuto tramite l'algoritmo di Huffman per l'alfabeto {a, b, c, d, e} supponendo che ogni carattere appaia con le seguenti frequenze

a	b	c	d	e
12	10	13	57	8

Spiegare il processo di costruzione del codice



Esercizio (9 punti): Si consideri un file definito sull'alfabeto $\{a, b, c\}$, con frequenze $f(a), f(b), f(c)$. Per ognuna delle seguenti codifiche determinare, se esiste, un opportuno assegnamento di valori alle 3 frequenze per cui l'algoritmo di Huffman restituisce tale codifica, oppure argomentare che tale codifica non è mai ottenibile

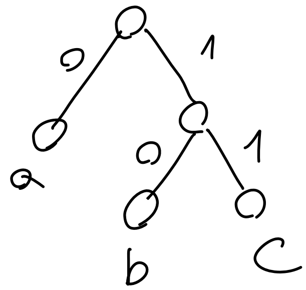
1) $e(a) = 0, \quad e(b) = 10, \quad e(c) = 11$
2) $e(a) = 1, \quad e(b) = 0, \quad e(c) = 11$
3) $e(a) = 10, \quad e(b) = 01, \quad e(c) = 00$

2) $e(a)$ è prefisso di $e(c)$ $\Rightarrow$
codifica non libera da prefissi $\Rightarrow$
mai output di Huffman

1)

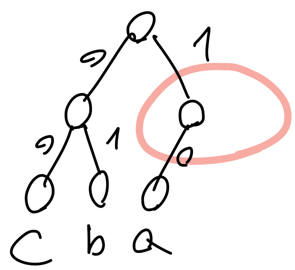
per esempio

$f(a) = 50$
$f(b) = 25$
$f(c) = 25$



$f(b), f(c) < f(a)$

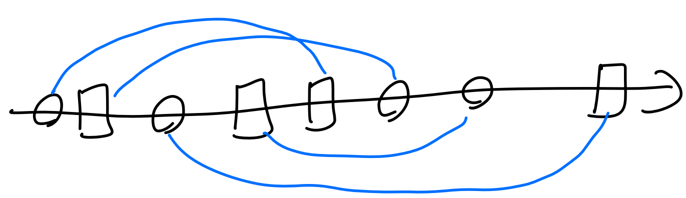
3)



non è pieno
$\Longrightarrow$ non è ottimo
$\Longrightarrow$ mai output di Huffman

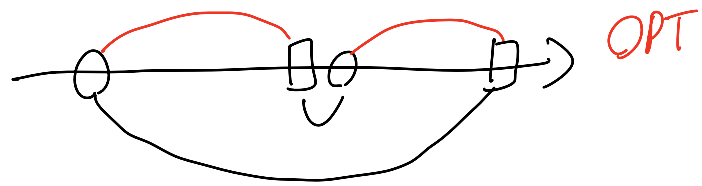
Esercizio: matching sulla linea

Sia $S = \{s_1, s_2, \dots, s_n\}$ un insieme di punti ordinati sulla retta reale, rappresentanti dei server. Sia $C = \{c_1, c_2, \dots, c_n\}$ un insieme di punti ordinati sulla retta reale, rappresentanti dei client. Il costo di assegnare un client c_i ad un server s_j è $|c_i - s_j|$. Fornire un algoritmo greedy che assegna ogni client ad un server distinto e che minimizzi il costo totale (equiv., medio) dell'assegnamento.

esempio:  $n=4$

~~1)~~ client al server + vicino, partendo dalla coppia client-server con distanza minore

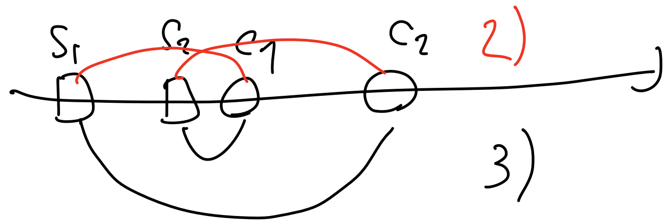
NON funziona:



√ 2) $c_1 - s_1$, $c_2 - s_2$, ...

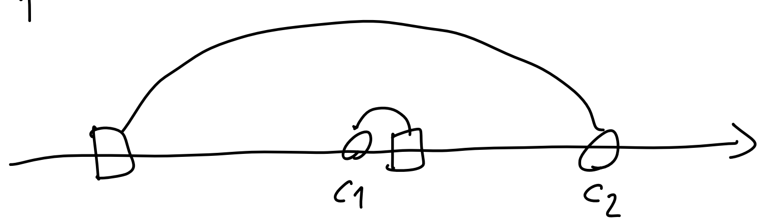
(equiv. $c_n - s_n$, $c_{n-1} - s_{n-1}$, ...)

~~3)~~ c_1 al server + vicino; c_2 al server + vicino



$\Rightarrow$ sono algoritmi $\neq$

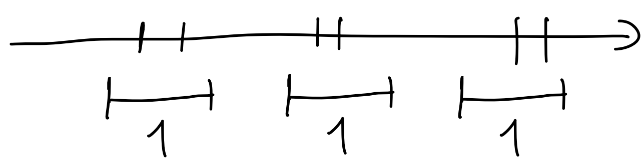
NON funziona:



Per casa: dimostrare ottimalità dell'algoritmo 2)

Esercizio: Sia $X = \{x_1, x_2, \dots, x_n\}$ un insieme di punti ordinati sulla retta reale.
Fornire un algoritmo greedy che determini un insieme I di cardinalità minima di intervalli chiusi di ampiezza unitaria $\left([a,b] \in I \Rightarrow b-a=1\right)$ tale che $\forall x_i \in X \ \exists j \in I$ tale che $x_i \in j$.

esempio:



✓ 1) da sx a dx, inizia un nuovo intervallo nel primo punto non coperto

```
MIN_COVER(X)
    n = length(X)
    C = {[x_1, x_1 + 1]}
    last = 1
```

```
for i=2 to n do
    if x_i > x_last + 1 then
        C = C ∪ { [x_i, x_i + 1] }
        last = i
return C
```

Per casa: dimostrare ottimalità di MIN_COVER