Scritta da Gabriel

Un algoritmo greedy per il problema può essere simile a “Metric Matching”, quindi:

$TIME\_COMPLETION(C, n)$
$A = C[1]$
$last = 1$
$for\ j = 2\ to\ n$
    $if\ C_j > C_{last} + 1$
        $last\ =\ j$
        $A = C_j \cup A$
$return\ A$

Dovendo minimizzare la somma del tempo di completamento dei programmi sapendo che ciascuno ha tempo di completamento C_j, allora il modo per poterlo fare è considerare di volta in volta il programma con tempo di completamento maggiore più lontano rispetto a quello attuale (sapendo che i tempi di completamento sono ordinati, dovendo minimizzare la somma dei tempi di completamento, cerco di “prendere meno programmi”, quindi “il più vicino e il più lontano per valore” da quello e, in tal modo, trovo tutti i programmi che hanno tempo <= al tempo di completamento che devo considerare.

Quindi: prende il primo programma che trova che abbia lunghezza maggiore all’ultima scelta greedy e così via linearmente fino alla fine.

- Caso base, il programma è minimo e abbiamo concluso.
- Caso induttivo, abbiamo un insieme di k programmi e, considerando da subito il programma minimo, realizzando come spiegato la scelta ottima, riduco il tempo rimanente e compio di volta in volta la scelta minima, cercando di assegnare i programmi minimizzando la somma dei tempi di completamento