① Corectitudine

Fie S_a mulțimea tuturor subintervalelor care îl acoperă pe a. Unul dintre subintervale trebuie să aparțină soluției optime. Dacă îl înlocuim cu un subinterval (a_{max}, b_{max}) din S_a pentru care b_{max} este maxim în S_a (adică ajunge cel mai în dreapta), restul intervalului neacoperit (b_{max}, b) va reprezenta o submulțime a intervalului rămas din soluția de acoperire optimă. Deci o soluție construită din (a_{max}, b_{max}) și o soluție optimă pentru intervalul rămas (b_{max}, b) va fi de asemenea optimă.

Complexitate:

Funcția std::sort() are complexitatea worst-case $O(n \log n)$.

② Corectitudine

Request -i, deadline d_i, timp necesar realizării - t_i

Timpul de început - s, interval $[s_i, f_i]$, $f_i = s_i + t_i$

Întârzierea pt requestul i dacă deadlineul este matat va fi $l_i = f_i - d_i$

Scopul problemei de optimizare: minimizarea întârzierii maxime $L = \max_i L_i$ prin planificarea tuturor requesturilor a.î acestea nu se suprapun.

b) Este corect un algoritm Greedy bazat pe planificarea activităților lor în ordine crescătoare în raport cu lungimea lor?

Nu, deoarece sunt ignorate deadlineurile.

Request 1 → $d_1 = 100$, $t_1 = 1$

Request 2 → $d_2 = 10$, $t_2 = 10$.

Pt. a obține o întârziere 0, trebuie să planificăm mai întâi requestul 2. Deci nu poate fi corectă o astfel de implementare.

c) Este corect un algoritm Greedy bazat pe planificarea activităților în ordine crescătoare în raport cu timpul maxim la care trebuie să înceapă activitatea i pentru a respecta termenul limită?

Nu. Considerăm următorul exemplu:

Request 1 → $d_1 = 2$, $t_1 = 1$

Request 2 → $d_2 = 10$, $t_2 = 10$.

Sortând ca în enunț, Request 2 ar fi planificat primul, iar Request 1 va avea o întârziere de 9. Dacă le planificăm invers, Request 1 e terminat la timp și Request 2 are o întârziere de 1.

d) Metoda corectă implică sortarea requesturilor (joburilor) în ordinea deadlineurilor și le planificăm în această ordine (earliest deadline first)

Prin redenumirea joburilor dacă este necesar, presupunem că

Vom planifica toate joburile în această ordine.

Fie S timpul de început pentru toate joburile.

Job 1 → timp de început $s = s(1)$

timp de sfârșit $f(1) = s(1) + t_1$

etc. Notăm cu f timpul de sfârșit pentru ultimul job planificat.

Ordonez joburile după deadline-uri

pp. că $d_1 \leq \ldots \leq d_n$

Inițial $f = s$

- Considerăm joburile $i = 1, \ldots n$ în această ordine.

  Atribui jobul i intervalului de la $s(i) = f$ la $f(i) = f + t_i$

  Fie $f = f + t_i$

End

Returnez setul de intervale $[s(i), f(i)]$ pentru $i = 1, \ldots n$