2.3. El problema de la *partició interval* (*Interval Partitioning Problem*) és similar al problema de la selecció d'activitats vist a classe però, en lloc de tenir un únic recurs, tenim molts recursos (és a dir, diverses còpies del mateix recurs). Doneu un algorisme que permeti programar totes les activitats fent servir el menor número possible de recursos.

Sigui "r" el nº de recursos inicial, "r'" el nº de recursos final, "n" el nº de processos a tractar, "d" el màxim nº de processos que, simultàniament, se sol·lapen, i fem ús d'una cua de prioritat (minheap) per ordenar els processos per cada recurs (on el criteri d'ordenació ve marcat pel temps d'acabament dels processos continguts en ella):

1. Ordenar n (de manera creixent segons el temps d'inici [s_i,t_i]) **O(n*log(n))**
2. r' = 0
3. Mentre tinguem alguna n lliure: **O(n*log(d))**
    4. Agafem la primera n disponible -> l'anomenem n'
    5. Si el primer procés de la cua de prioritat, pel recurs r' actual, acaba abans del que comença n'
        6. Assignem n' a la r' actual **O(1)+O(log(d))**
    7. Sino, r'++ i assignem n' a la cua de prioritat pel nou recurs r'
8. Retornem r'

Cost total: **O(n*log(n)+n*log(d))**