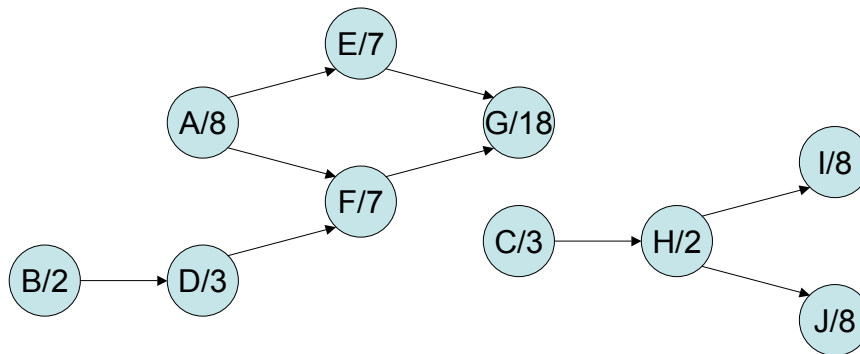


Homework1

(Due date to be announced)

- Adding computational resources to a task does not always speed things up. To illustrate this, schedule the following job set greedily to minimize maximum completion time; first two processors, then on three processors. Draw the two schedules.



In the job label of the form "X/Y" the X stands for the job identifier and the Y for the execution time.

- We are given a set of jobs J_1, J_2, \dots, J_n . For each Job J_i we have:

r_i arbitrary release time

c_i arbitrary execution time

d_i arbitrary deadline

w_i the relative importance of the job

In addition, each job J_i is decomposed into two subjobs: the mandatory subjob M_i and the optional subjob O_i . Let m_i and o_i be the processing times for M_i and O_i , respectively, with $m_i + o_i = c_i$.

Propose an algorithm to feasibly schedule the set of jobs on a single processor (pre-emptively) in order to minimize total error arising from the imprecision. Prove the correctness of the algorithm.

We define the error of Job J_i to be $e_i = o_i - \sigma_i$, where σ_i denotes the amount of processor time allocated to the optional subjob O_i . We want to minimize the total error $e = \sum_{i=1}^n w_i e_i$.

3. We are given a set of jobs J_1, J_2, \dots, J_n . For each Job J_i we have:

r_i arbitrary release time

c_i arbitrary execution time

d_i arbitrary deadline

Propose an algorithm to feasibly schedule the set of jobs on m processors, each of speed Δ_j . (The execution time of a job on a processor with speed Δ is $1/\Delta$ that of a processor of speed 1.) Preemption is allowed. Prove the correctness of the algorithm.