Single Machine Problems: Branch and Bound Algorithm

Branch and Bound algorithm (B&B) is an exact method for finding an optimal solution to an NP-hard problem. It is an enumerative technique that can be applied to a wide class of combinatorial optimisation problems.

To describe the B&B algorithm, consider problem $1||\Sigma T_j$. This problem is NP-hard, i.e., it is unlikely that there can be developed a polynomial-time algorithm for finding an optimal schedule.

The input data for the instance with n=4 jobs is given by the following table:

Jobs	1	2	3	4
p_j	12	8	15	9
d_j	16	26	25	27

A schedule corresponds to an assignment of the jobs to the four positions in the job sequence.

Initialisation: To generate a schedule we start with no job sequenced and indicate this by (*,*,*,*). Here "*" in the job sequence indicates that no job has yet been assigned to that position.

Step 1: To construct a schedule starting from the first position, we move from node (*,*,*,*) to one of the four possible nodes (1,*,*,*); (2,*,*,*); (3,*,*,*); (4,*,*,*).

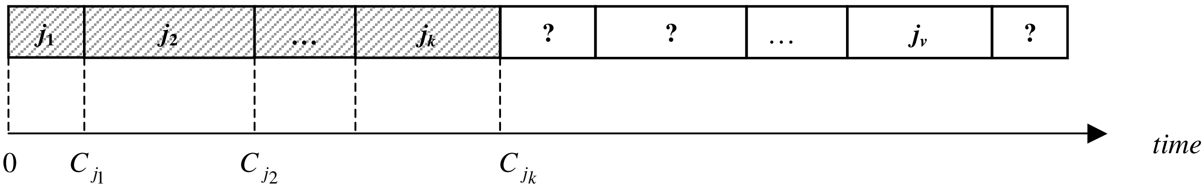
Step 2: To assign the second job in the sequence, we branch from the each of these four nodes to three possibilities: branching from (1,*,*,*) gives (1,2,*,*), (1,3,*,*), and (1,4,*,*); branching from (2,*,*,*) gives three possibilities (2,1,*,*), (2,3,*,*), and (2,4,*,*), etc.

Step 3: Assigning the job to be processed in the third position immediately fixes the last job.

This process is represented by a branching tree. Each node of a tree corresponds to a partial schedule with several jobs assigned to the first positions and the remaining jobs unassigned. To avoid full enumeration of all job permutations, we calculate in each step the *Lower Bound* (*LB*) of the value of the objective function for each partial schedule.

Calculation of the Lower Bound is implemented in a different way for different problems.

Consider problem $1||\Sigma T_j$ and a partial schedule obtained at Step k after k jobs have been assigned to the first k positions.



- For the jobs assigned to the first k positions, their actual tardiness is determined as

$$T_{j_u} = \max\{C_{j_u} - d_{j_u}, 0\}.$$

- For any unscheduled job j_v, its tardiness T_{j_v} cannot be less than $\max\{C_{j_v} - d, 0\}$, where $\boldsymbol{d}$ is the maximum due date among the unscheduled jobs:

$$T_{j_v} \geq \max\{C_{j_v} - d, 0\}.$$

Consider an artificial problem where the unscheduled jobs have a common due date

$$\boldsymbol{d} = \max\{d_{j_{k+1}}, d_{j_{k+1}}, \ldots, d_{j_n}\}.$$

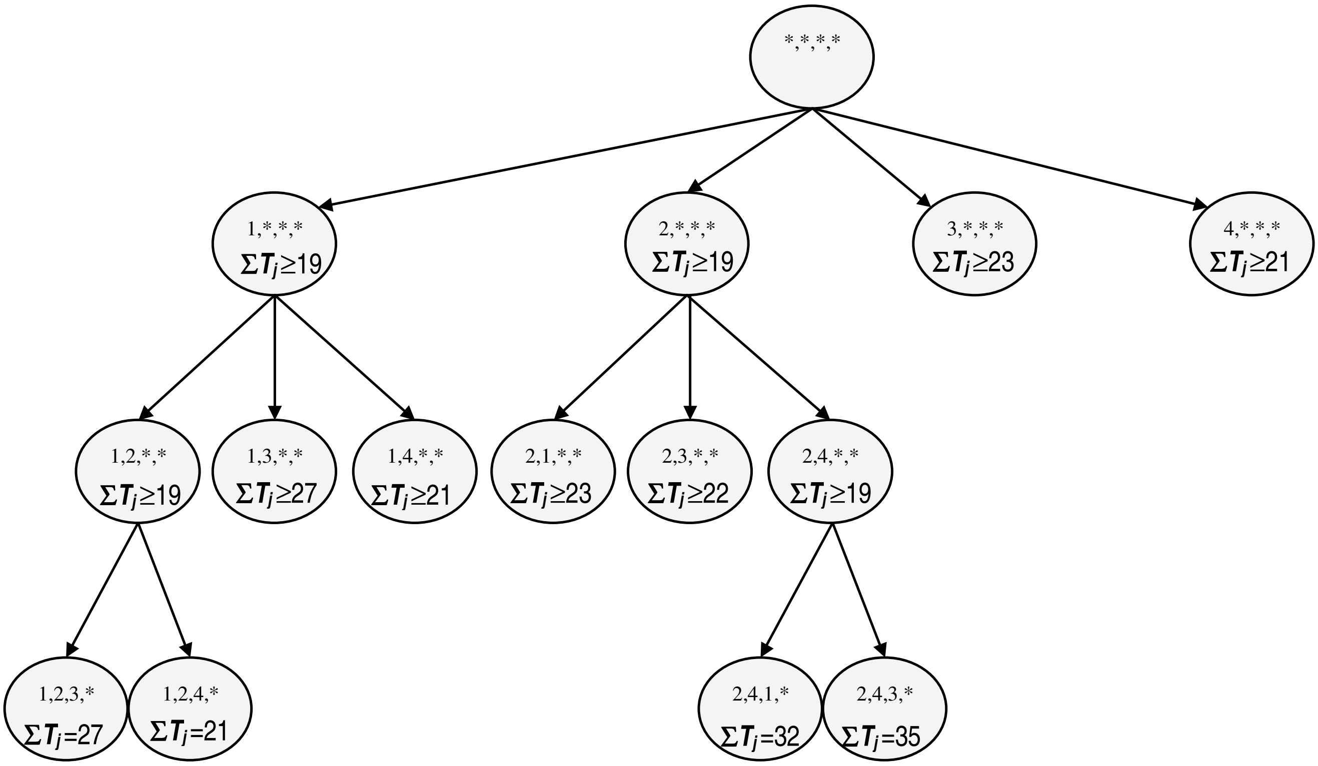
The minimum total tardiness with respect to common due date $\boldsymbol{d}$ can be found by sequencing the unscheduled jobs in the SPT order[1]. Thus completion times C_{j_v} of the unscheduled jobs and their tardiness T_{j_v} can be found by applying the SPT-rule.

$$LB = \underbrace{\left(T_{j1} + T_{j2} + \ldots + T_{jk}\right)}_{\text{scheduled jobs}} + \underbrace{\left(T_{jk+1} + T_{jk+2} + \ldots + T_{jn}\right)}_{\text{unscheduled jobs}}$$

*Sequencing the unscheduled jobs in the SPT order and replacing their original due dates d_{j_v} by a large artificial due date **d** is done temporarily in order to determine the Lower Bound for the current partial schedule.*

[1] We use here an auxiliary result: an optimal schedule for problem $1|\, d_j = d \,|\, \Sigma T_j$ with equal due dates can be obtained by SPT-rule. A task to prove this rule will be included in Coursework 2.

The resulting branch-and-bound tree is represented in the following figure.
An example of calculating the lower bounds is given below.

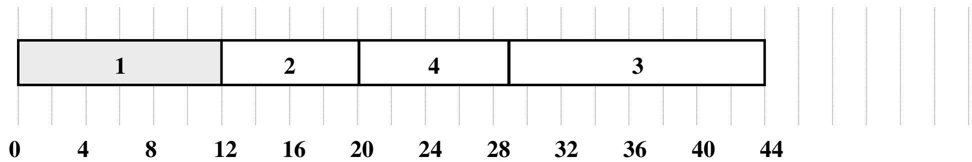


Calculating Lower Bounds

In the partial schedule (1,*,*,*), tardiness of job 1 is

$$T_1=\max\{C_1-d_1,0\}=\max\{12-16,0\} = 0.$$

Total tardiness of the remaining jobs 2, 3, and 4 is not less than the minimum total tardiness calculated with respect to the large common due date $d=\max\{d_2, d_3, d_4\}=27$. In accordance with Exercise 1, the jobs 2, 3, 4 should be sequenced in the SPT-order:



$$T_2+ T_3+ T_4 \geq \max\{C_2-d,0\} + \max\{C_3-d,0\} + \max\{C_4-d,0\} =$$
$$=\max\{20-27,0\} + \max\{44-27,0\} + \max\{29-27,0\} = 0+17+2 = 19.$$

It follows that $T_1+T_2+ T_3+ T_4 \geq 0+19$ and this is the lower bound LB for the partial schedule (1,*,*,*).

Advantages of Branch & Bound algorithm:

— Finds an optimal solution (if the problem is of limited size and enumeration can be done in reasonable time).

Disadvantages of Branch & Bound algorithm:

— Extremely time-consuming: the number of nodes in a branching tree can be too large.

The algorithm finds the first complete schedule and then tries to improve it. Often developing the “promising” branches may lead to a huge number of offsprings that finally may not give an improvement. Thus the size of the tree may grow exponentially without improving the best solution obtained.