Chapter 2 : Production planning

ENSIIE - Operations Research Module

Dimitri Watel (dimitri.watel@ensiie.fr)

2024

2 machines job shop scheduling problem

Some important things to note

- The machine M_1 can focus on at most one job at a time
- The machine M_2 can focus on at most one job at a time
- The machines M_1 and M_2 can be parallelized
- Each bear must go through the machine M_1 before M_2.
- The time of each job on each machine depends on the job .

2 machines job shop scheduling problem

There are 8 jobs to be done.
The following table gives, for each job and each machine the time (in minutes) the machine needs to complete the job.

$t(j_i, M_j)$	j_1	j_2	j_3	j_4	j_5	j_6	j_7	j_8
M_1	10	30	20	60	40	40	50	30
M_2	15	50	100	30	10	90	20	40

Question : how to minimize the production time of the 8 jobs?

The Johnson algorithm

$t(j_i, M_j)$	j_1	j_2	j_3	j_4	j_5	j_6	j_7	j_8
M_1	10	30	20	60	40	40	50	30
M_2	15	50	100	30	10	90	20	40

- Partition the jobs into the two following sets:
  - $A = \{j_i, t(j_i, M_1) \leq t(j_i, M_2)\}$
  - $B = \{j_i, t(j_i, M_1) > t(j_i, M_2)\}$
- Let S_A be the list of elements in A sorted by $t(j_i, M_1)$ in **ascending** order
- Let S_B be the list of elements in B sorted by $t(j_i, M_2)$ in **descending** order
- Return S_A concatenated with S_B in that order

(see the board for an example)

The Johnson algorithm

Property

The Johnson algorithm is optimal.

However

- The algorithm does not work if we add some constraints (this job must be done before this one; the machine M_1 must be stopped during y minutes each time it is used more that x minutes, ...)
- The algorithm does not work if we add one or more machines, be can be adapted to some cases...

What if we remove the ordering constraint

Some important things to note

- The machine M_1 can focus on at most one job at a time
- The machine M_2 can focus on at most one job at a time
- The machines M_1 and M_2 can be parallelized
- ~~Each bear must go through the machine M_1 before M_2.~~ A bear cannot go through the machine M_1 and M_2 at the same time.
- The time of each job on each machine depends on the job .

Optimal time

For readability, we write $a_i = t(j_i, M_1)$ and $b_i = t(j_i, M_2)$

Let $T_1 = \sum_i a_i$.

Let $T_2 = \sum_i b_i$.

Let $M = \max_i a_i + b_i$.

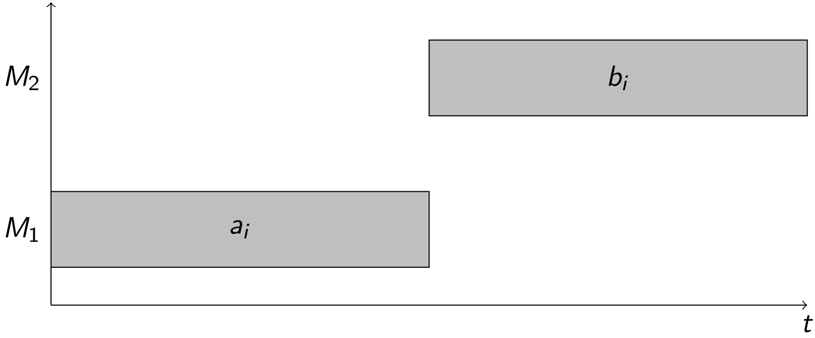
Theorem

The optimal time is max T_1, T_2, M.

How to achieve this time?

Case 1 : $M \geq T_1, T_2$

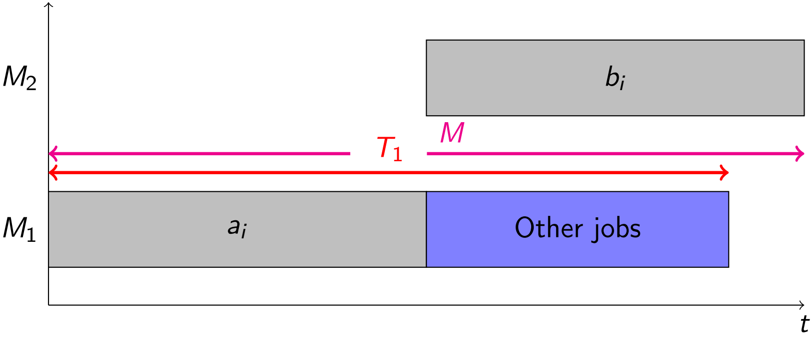
Let i such that $M = a_i + b_i$. Then the following solution is optimal



1 : Place the job i

Case 1 : $M \geq T_1, T_2$

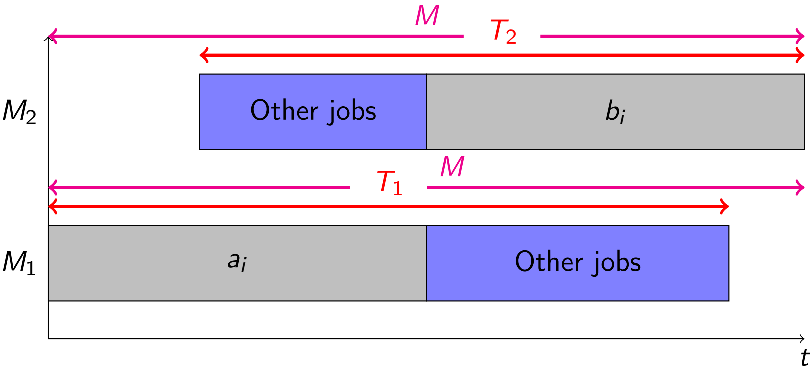
Let i such that $M = a_i + b_i$. Then the following solution is optimal



2 : The left space on M_1 is enough for the other jobs as $M \geq T_1$.

Case 1 : $M \geq T_1, T_2$

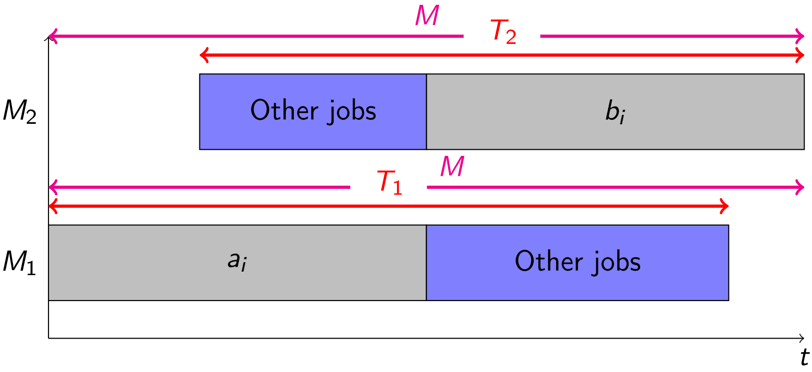
Let i such that $M = a_i + b_i$. Then the following solution is optimal



3 : The left space on M_2 is enough for the other jobs as $M \geq T_2$.

Case 1 : $M \geq T_1, T_2$

Let i such that $M = a_i + b_i$. Then the following solution is optimal



No task is done on the same time on the two machines.

Case 2 : $M < T_1$ or T_2

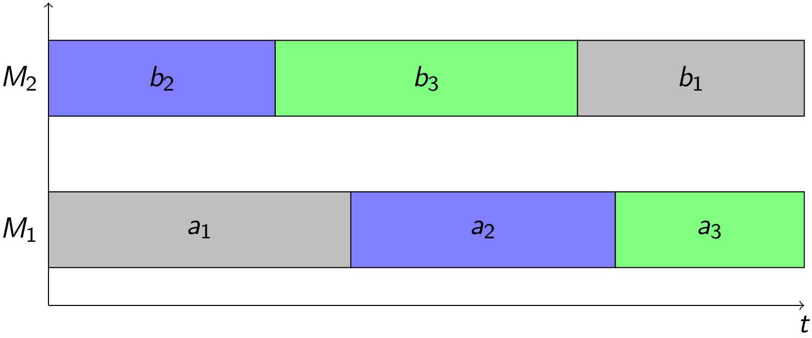
We can assume that $T_1 = T_2$

If $T_1 < T_2$, we add $T_2 - T_1$ fictive jobs with $a_i = 1$ and $b_i = 0$.
If $T_2 < T_1$, we add $T_1 - T_2$ fictive jobs with $a_i = 0$ and $b_i = 1$.
Thus $T_1 = T_2$.

(Those jobs can be placed anywhere without generating a conflict.)

Case 2 : $M < T_1 = T_2$ and there are only 3 jobs

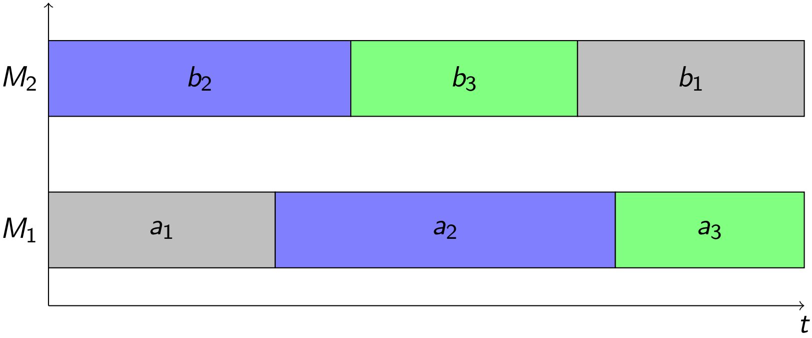
The following procedure finds an optimal solution.



If $a_1 \geq b_2$ and $b_1 \geq a_3$, the above solution does not generate any conflit.

Case 2 : $M < T_1 = T_2$ and there are only 3 jobs

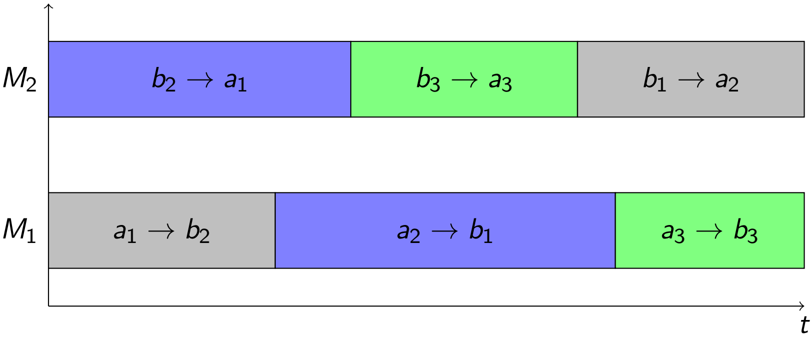
The following procedure finds an optimal solution.



If $a_1 < b_2$, there is a conflict (with the job 2). However, we can ignore this case by ...

Case 2 : $M < T_1 = T_2$ and there are only 3 jobs

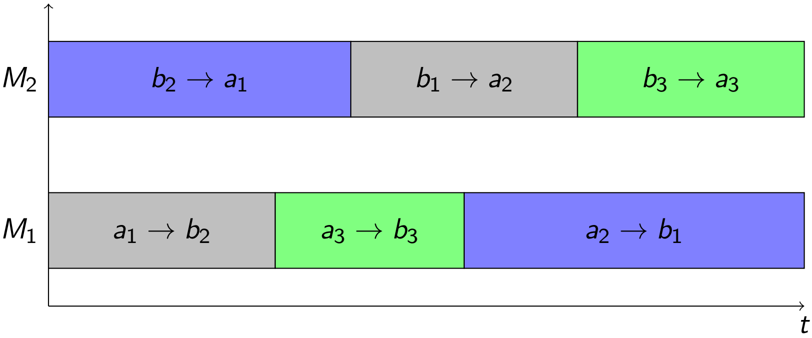
The following procedure finds an optimal solution.



If $a_1 < b_2$, there is a conflict (with the job 2). However, we can ignore this case by ... renaming j_1 in j_2 and j_2 in j_1 and M_1 in M_2 and M_2 in M_1. Thus we have $a_1 \geq b_2$. (with the new values of a_1 and b_2)

Case 2 : $M < T_1 = T_2$ and there are only 3 jobs

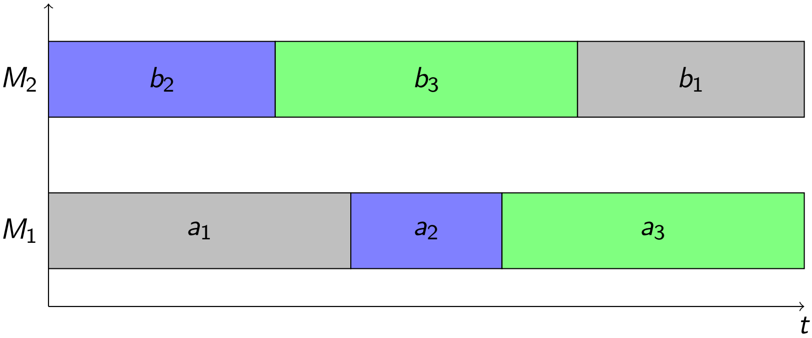
The following procedure finds an optimal solution.



If $a_1 < b_2$, there is a conflict (with the job 2). However, we can ignore this case by ... renaming j_1 in j_2 and j_2 in j_1 and M_1 in M_2 and M_2 in M_1. Thus we have $a_1 \geq b_2$. And we can try the solution of the 13th slide.

Case 2 : $M < T_1 = T_2$ and there are only 3 jobs

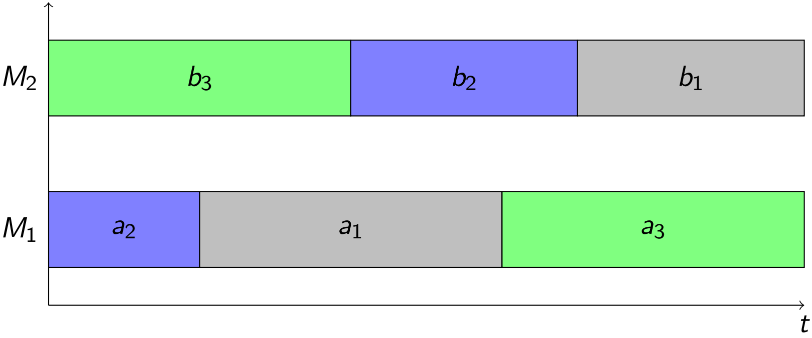
The following procedure finds an optimal solution.



If $a_1 \geq b_2$ but $b_1 < a_3$, there is a conflict (with the job 3).
However, ...

Case 2 : $M < T_1 = T_2$ and there are only 3 jobs

The following procedure finds an optimal solution.



If $a_1 \geq b_2$ but $b_1 < a_3$, there is a conflict (with the job 3).
However, ... the above solution works as $a_1 + a_3 > b_1 + b_2$.
(Recall : $a_3 \leq b_1 + b_2$ as $M < T_1, T_2$)

Case 2 : $M < T_1 = T_2$ and there are $n > 3$ jobs

Trick

We can always simplify the instance to get only 3 left jobs:

Create 3 empty sets J_1, J_2, J_3
for k from 1 to n **do**
  for p from 1 to 3 **do**
    if $\sum_{i \in J_p \cup \{k\}} a_i + b_i \leq T_1$ **then**
      Add k to J_p

Theorem

At the end, every job is either in J_1, J_2 or J_3.

Case 2 : $M < T_1 = T_2$ and there are $n > 3$ jobs

We then use the following algorithm:

- build the three sets J_1, J_2 and J_3
- create the jobs j_1', j_2' and j_3' where $a_i' = \sum_{j_k \in J_i} a_k$ and $b_i' = \sum_{j_k \in J_i} b_k$
- find a solution using the algorithm of Case 2.1
- replace the job j_1' by all the jobs in J_1 in any order. Do the same for j_2' and j_3'

At the end, there is no conflict as this would imply a conflict with one of the jobs $j_{1'}$, $j_{2'}$ or $j_{3'}$.

In the syllabus, to be seen in tutorials

- Adapt PERT and/or Metra Potential (MPM) with capacity constraints.
- Johnson algorithm with 3 machines M_1, M_2, M_3 such that
  - $\max_i t(j_i, M_2) \leq \min_i t(j_i, M_1)$
  - $\max_i t(j_i, M_2) \leq \min_i t(j_i, M_3)$.