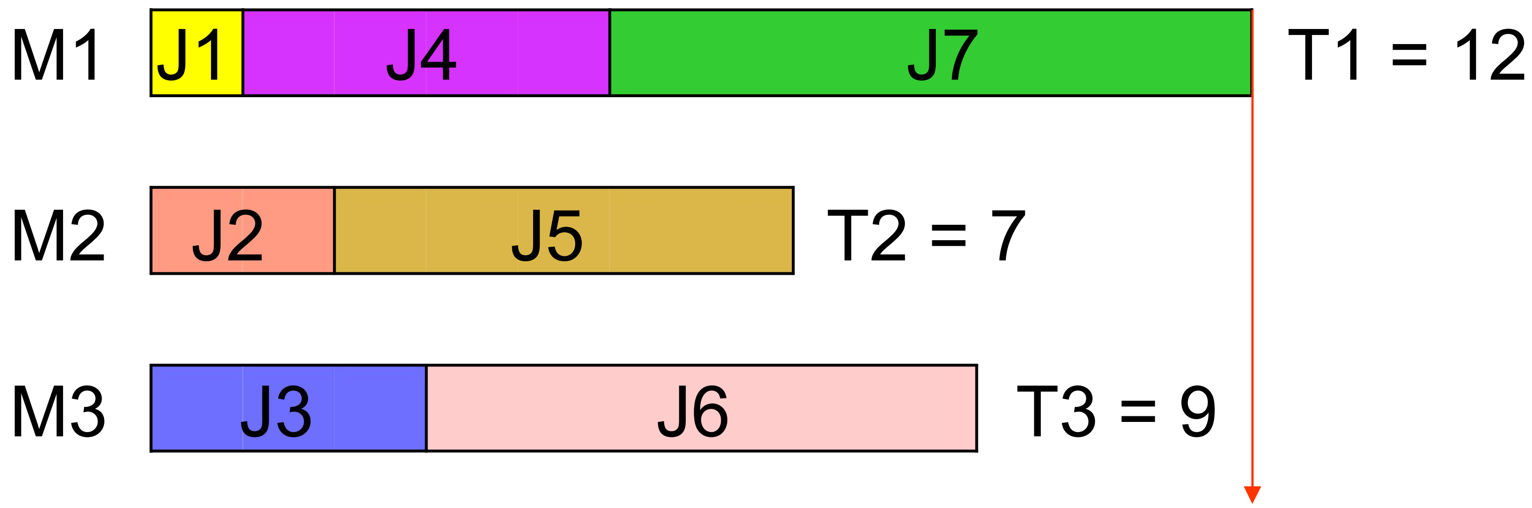
Load Balancing Problem

Input: m identical machines: M1, M2, ..., Mm

n jobs: J1, J2, ..., Jn

Processing time of each job: t_j ($j = 1, 2, ..., n$)

Example: 3 machines and 7 jobs (t_j = 1, 2, 3, 4, 5, 6, 7)

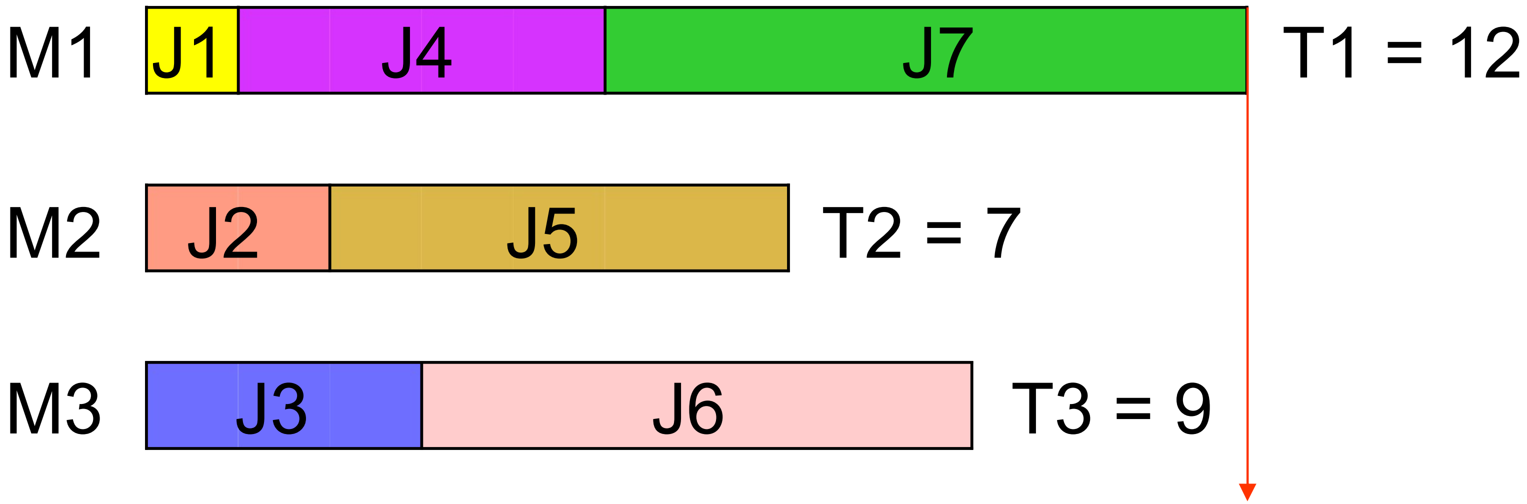


Makespan T = max {T1, T2, T3} = 12

Q. What is the best assignment ?

A. The assignment with the minimum makespan.

Example: 3 machines and 7 jobs (t_j = 1, 2, 3, 4, 5, 6, 7)



Makespan T = max {T1, T2, T3} = 12

Optimal Solution:

M1: ??, ...

M2: ??, ...

M3: ??, ...

Optimal Makespan $T^* = ??$

Greedy Algorithm

Assign a job to the machine with the smallest load in an arbitrary order of jobs.

Simple Example: Two Machines and Three Jobs

J1: 20 **J2: 20** **J3: 40**

If the three jobs are assigned in the order of J1, J2, J3:

M1:	**J1: 20**	**J3: 40**	Makespan = 60
M2:	**J2: 20**		

If the three jobs are assigned in the order of J3, J2, J1:

M1:	**J3: 40**		Makespan = 40
M2:	**J2: 20**	**J1: 20**	

Question: What is the average makespan by this algorithm?

Greedy Algorithm

Assign a job to the machine with the smallest load in an arbitrary order of jobs.

Q: How good is this greedy algorithm?

The obtained makespan T is not worse than $2T^*$ where T^* is the optimal makespan ($T \le 2T^*$): 2-approximation

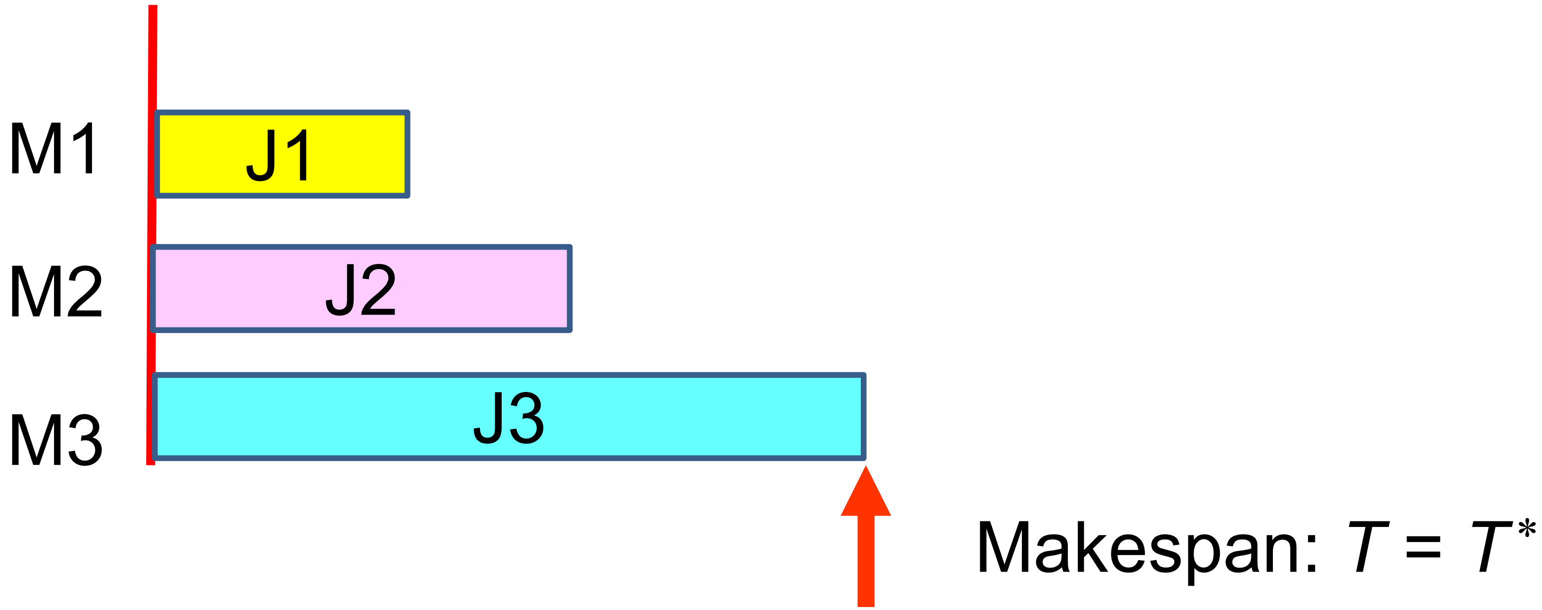
“<“ holds

Greedy Algorithm

Assign a job to the machine with the smallest load in an arbitrary order of jobs.

Q: How good is this greedy algorithm?

When the number of jobs is the same as or smaller than the number of machines, the optimal value is obtained by this algorithm: $T = T^*$ where T is the obtained makespan by the greedy algorithm and T^* is the optimal makespan.

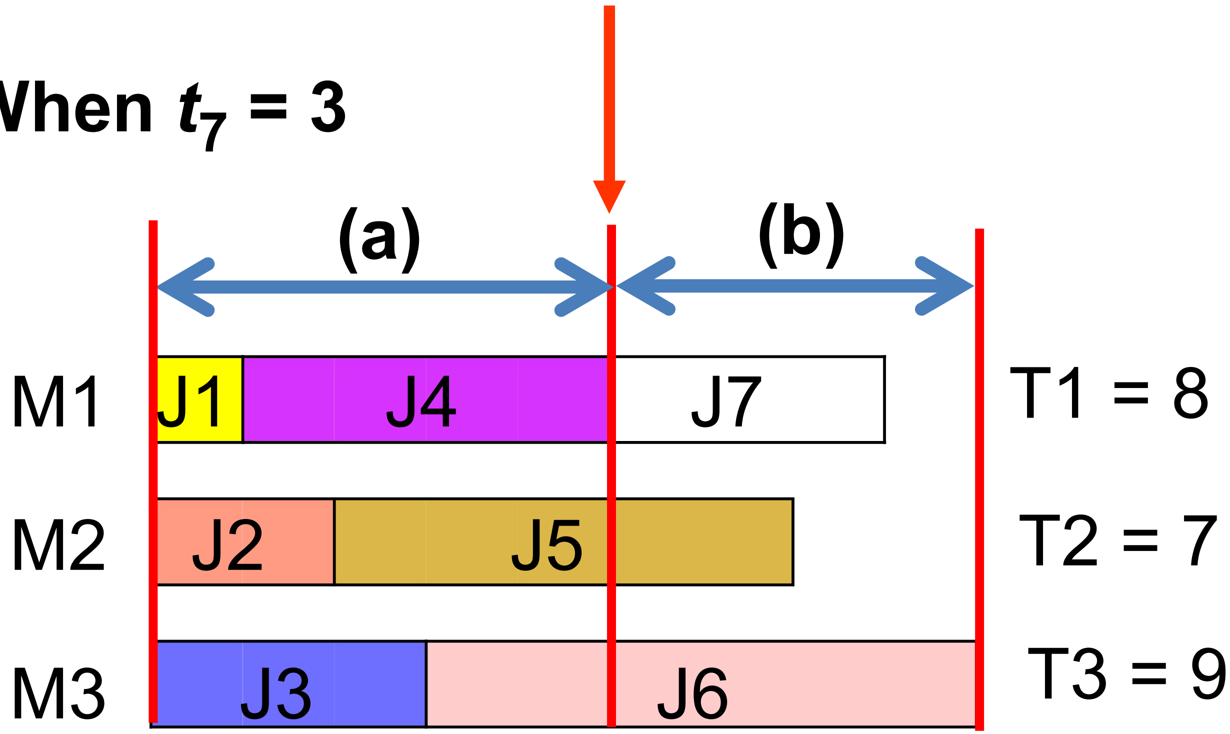


When the number of jobs is larger than the number of machines:
The obtained makespan T is not worse than $2T^*$ where T^* is the optimal makespan ($T \le 2T^*$): 2-approximation

$$(a) < \frac{1}{m}\sum_{j=1}^{n} t_j \le T^* \qquad (b) \le \max_{j=1,2,\dots,m} \{t_j\} \le T^*$$

The smallest load just before the last job assignment.

When $t_7 = 3$



When the number of jobs is larger than the number of machines:

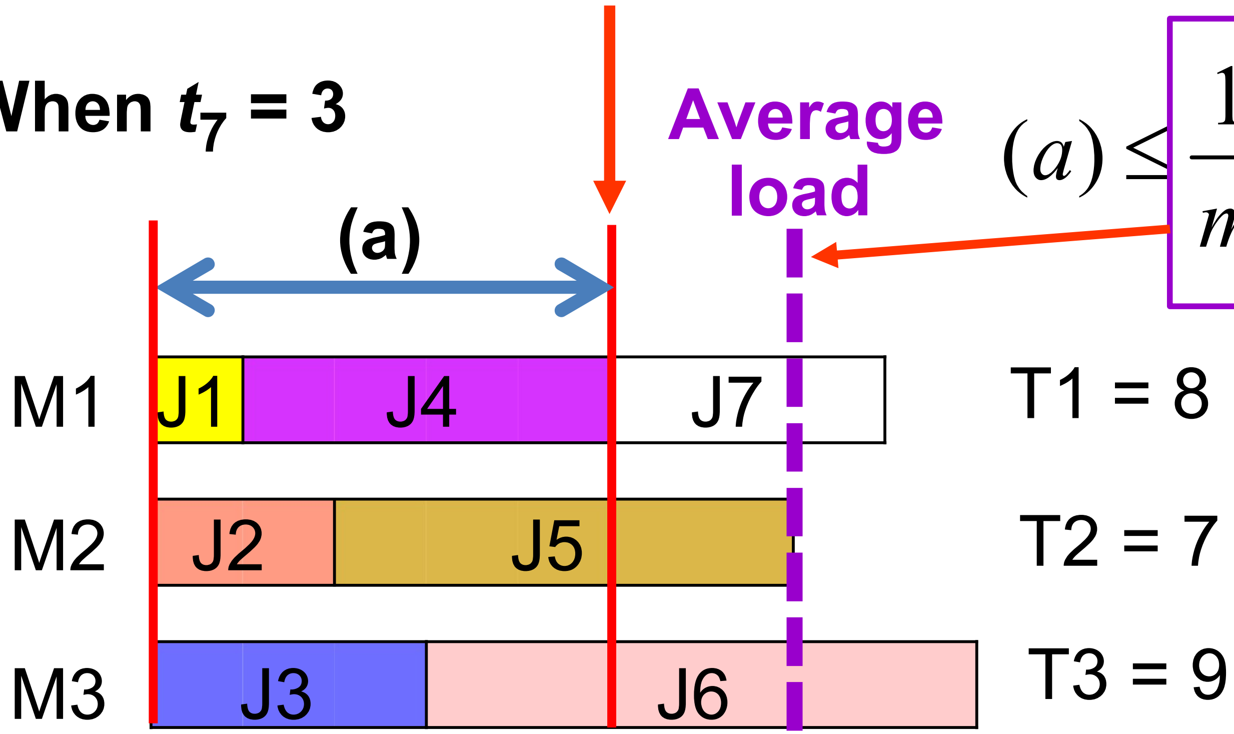
The obtained makespan T is not worse than $2T^*$ where T^* is the optimal makespan ($T \le 2T^*$): 2-approximation

$$(a) < \frac{1}{m}\sum_{j=1}^{n} t_j \le T^*$$	$$(b) \le \max_{j=1,2,\ldots,m} \{t_j\} \le T^*$$

The smallest load just before the last job assignment.

When $t_7 = 3$

$$(a) \le \boxed{\frac{1}{m}\sum_{j=1}^{n-1} t_j} < \frac{1}{m}\sum_{j=1}^{n} t_j \le T^*$$



When the number of jobs is larger than the number of machines:

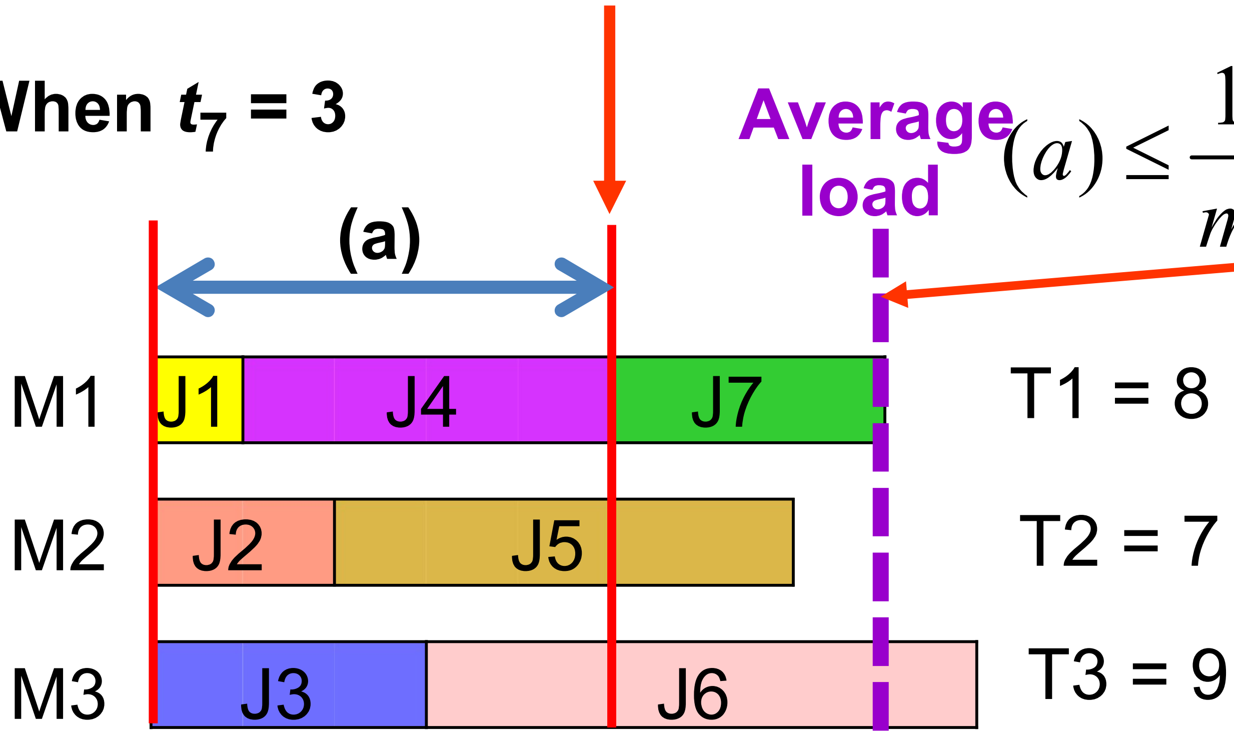
The obtained makespan T is not worse than $2T^*$ where T^* is the optimal makespan ($T \le 2T^*$): 2-approximation

$$(a) < \frac{1}{m}\sum_{j=1}^{n} t_j \le T^*$$

$$(b) \le \max_{j=1,2,\ldots,m} \{t_j\} \le T^*$$

The smallest load just before the last job assignment.

When $t_7 = 3$



$$(a) \le \frac{1}{m}\sum_{j=1}^{n-1} t_j < \boxed{\frac{1}{m}\sum_{j=1}^{n} t_j} \le T^*$$

When the number of jobs is larger than the number of machines:

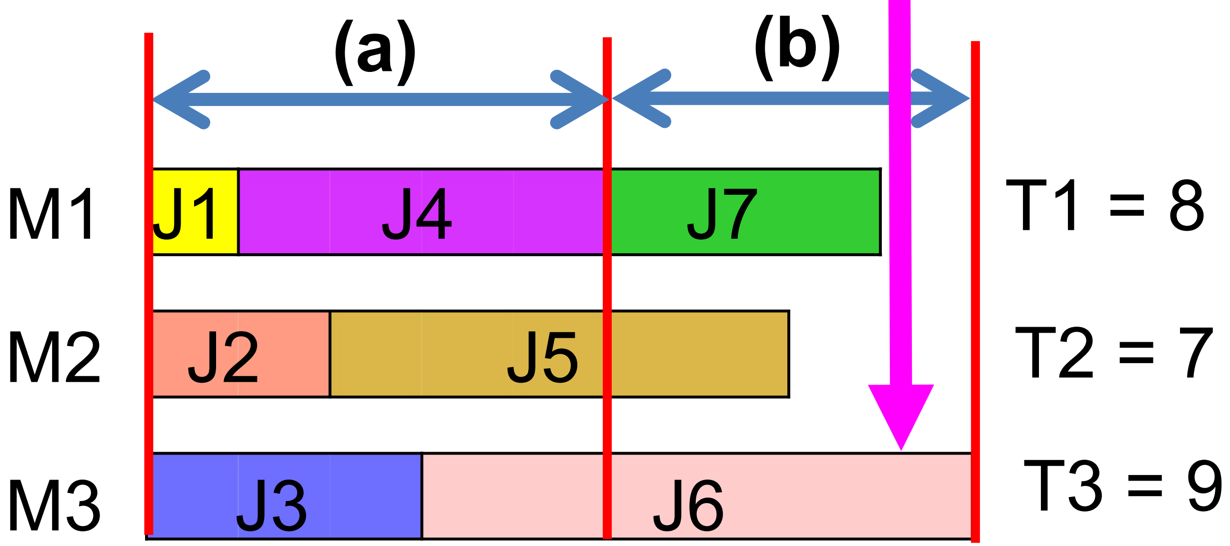
The obtained makespan T is not worse than $2T^*$ where T^* is the optimal makespan ($T \le 2T^*$): 2-approximation

$$(a) < \frac{1}{m}\sum_{j=1}^{n} t_j \le T^* \qquad (b) \le \max_{j=1,2,\dots,m} \{t_j\} \le T^*$$

The last job at the machine with the largest makespan.

When $t_7 = 3$

$$(b) \le t_6 \le \max\{t_j\} \le T^*$$



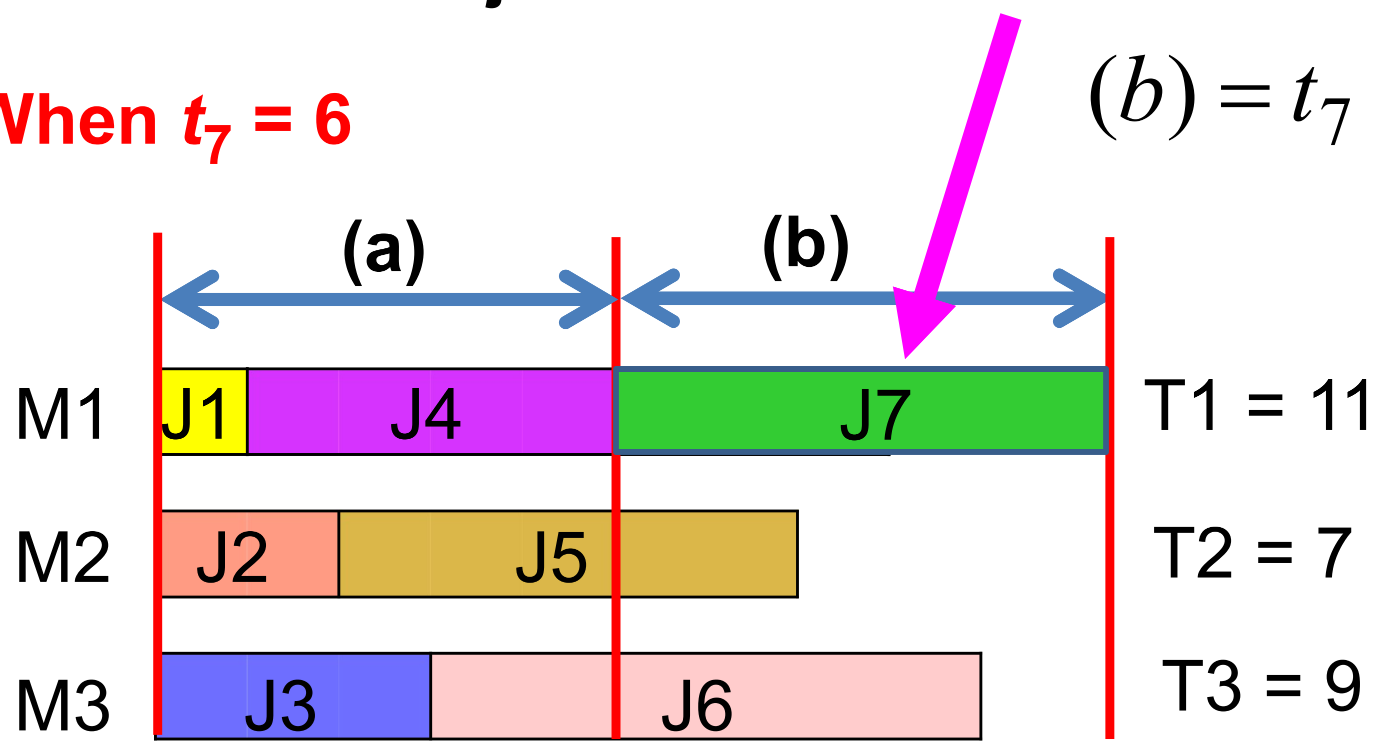
When the number of jobs is larger than the number of machines:
The obtained makespan T is not worse than $2T^*$ where T^* is the optimal makespan ($T \le 2T^*$): 2-approximation

$$(a) < \frac{1}{m}\sum_{j=1}^{n} t_j \le T^* \qquad (b) \le \max_{j=1,2,\dots,m} \{t_j\} \le T^*$$

The last job at the machine with the largest makespan.

When $t_7 = 6$

$$(b) = t_7 \le \max\{t_j\} \le T^*$$



```
List-Scheduling(m, n, t1,t2,...,tn) {
   for i = 1 to m {
      Li ← 0              ← load on machine i
      J(i) ← φ            ← jobs assigned to machine i
   }

   for j = 1 to n {
      i = argmink Lk             ← machine i has smallest load
      J(i) ← J(i) ∪ {j}          ← assign job j to machine i
      Li ← Li + tj               ← update load of machine i
   }
   return J(1), ..., J(m)
}
```

procedure GREEDY-BALANCE
  1 pass through jobs in any order.
  Assign job j to machine with current smallest load.
end procedure

Q: How tight is this upper bound?

Exercise 3-1:

Create two examples where the obtained makespan T is always the same as T^*. (Easy examples for the greedy algorithm).

Exercise 3-2:

Create two example where the obtained makespan T strongly depends on the order of jobs (i.e., the obtained makespan is much larger than T^* for some orders of jobs and the same as T^* for some other orders of jobs.

Exercise 3-3:

For each of the following three cases (i) $m = 2$ and $n = 3$, (ii) $m = 4$ and $n = 7$, and (iii) a general case with m machines and n jobs, create an example where the value of T/T^* is very large for the obtained makespan T by the greedy algorithm using a particular order of jobs. (If you can create an example where T/T^* is 2 (or approximately equal to 2), we can say that the upper bound $2T^*$ is tight).