Scheduling

Notes 6

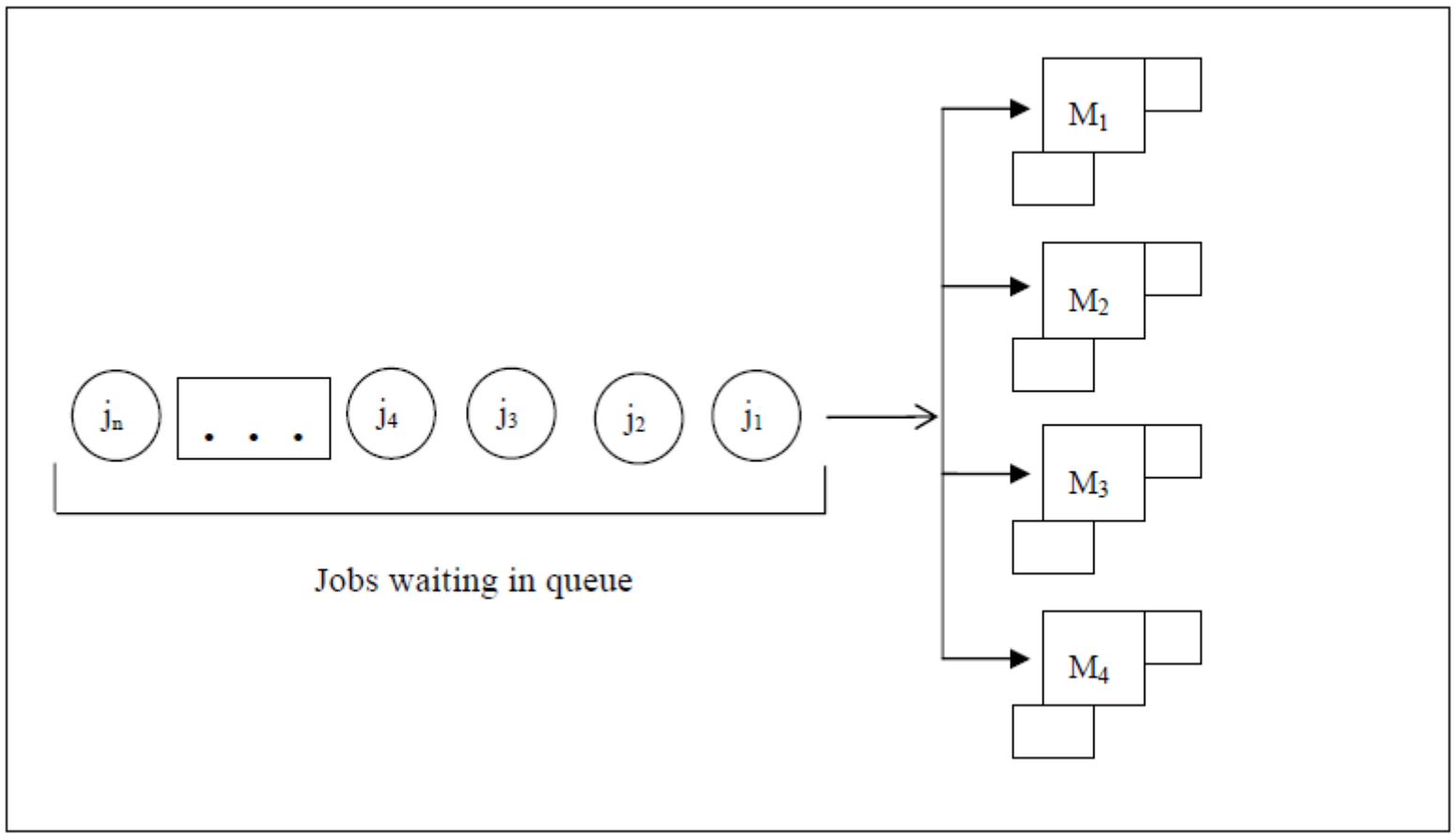
Extension of Johnson's Rule to 3 Machine 4/3/F/Cmax

Jobs	M1	M2	M3
1	6	5	7
2	9	3	3
3	5	4	8
4	8	2	4

Optimal Sequence = ?
Cmax = ?

Parallel Machine Scheduling

- In parallel machine scheduling environment, similar number of machines are available in multiple numbers and jobs can be scheduled over these machines simultaneously.
- Each job has one operation and the job can be processed on any of the machines.
- The time to process a job is the same on any machine.
- Generally, the numbers of machines are lower than the numbers of jobs.


j_n
j_4
j_3
j_2
j_1
Jobs waiting in queue
M_1
M_2
M_3
M_4

- Parallel machines can be grouped under four types:
- Same types with same speed
- Same types with different speed
- Different types with same speed
- Different types with different speed

- The machines are capable of processing all of the n jobs.
- m machines can process continuously.
- The jobs are processed only on one machine.
- The processing times of the jobs are known.

- There are two important constraints
- Precedence constraints: To process a job, the jobs before that job should be processed.

- Technological constraints: It means doing a job because of technological capabilities of machines.

- From a theoretical point of view parallel machine is a generalization of the single machine.
- The three principal objectives are the minimization of the makespan, the total completion time, and the maximum lateness.
- In practice, people have to deal with the problem of balancing the load on machines; by minimizing the makespan.

- The scheduling of parallel machines is as a two step process.
- First, the determination of which jobs have to be allocated to which machines,
- Second, the determination of the sequence of the jobs allocated to each machine.
- With the makespan objective only the allocation process is important.
- Due to the complex nature of scheduling problems many heuristics algorithms are developed to determine near-optimal schedule.

Heuristic Algorithms

- The term **heuristic** is used for methods which find solutions among all possible ones ,but they do not guarantee that the best will be found. These methods, usually find a solution close to the best one and they find it fast and easily. Sometimes these algorithms can be accurate,that is they actually find the best solution, but the algorithm is still called heuristic until this best solution is proven to be the best.
- Heuristic refers to a problem-solving method executed through learning-based techniques and experience.
- Now, some problems are hard and you may not be able to get an acceptable solution in an acceptable time. In such cases you often can get a not too bad solution much faster, by applying some arbitrary choices (educated guesses): that's a **heuristic**.
- A heuristic is still a kind of an algorithm, but one that will not explore all possible states of the problem, or will begin by exploring the most likely ones.
- An Algorithm is an arrangement of all around characterized directions for doing a specific assignment. It must be sound and finish. That means it must give you the right answer and it has to work for all cases.

- A heuristic algorithm is one that is designed to solve a problem in a faster and more efficient way than traditional methods by sacrificing optimality, accuracy, precision, or completeness for speed. Heuristic algorithms often times used to solve NP-complete problems, a class of decision problems. In these problems, there is no known efficient way to find a solution quickly and accurately although solutions can be verified when given.
- Heuristic algorithms are most often employed when approximate solutions are sufficient and exact solutions are necessarily computationally expensive.
- Genetic algorithm, tabu search, particle swarm optimization are examples of some heuristics.

Preemptive and Non-preemtive Scheduling

- Tasks are usually assigned with priorities. At times it is necessary to run a certain task that has a higher priority before another task although it is running. The running task is interrupted for some time and resumed later when the priority task has finished its execution. This is called **preemptive scheduling.**
- In **non-preemptive scheduling**, a running task is executed till completion. It cannot be interrupted.

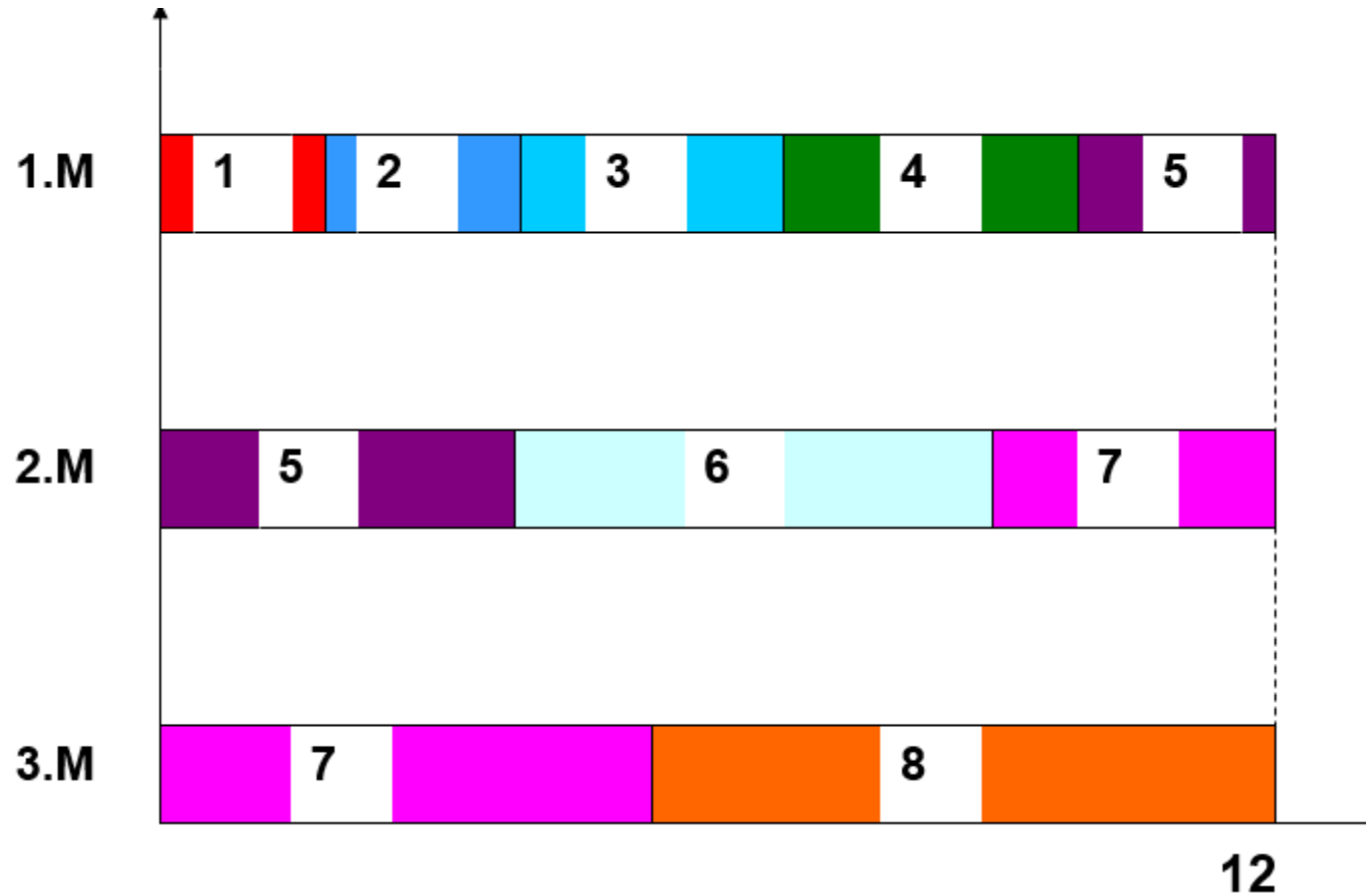
Preemptive Case

- If the jobs are preemptive, the problem can be solved as follows:
- Load per machine= $\frac{\sum_{i=1}^{n} P_i}{m}$
- **Step 1:** A job is assigned to the first machine.
- **Step 2:** Another unassigned job assigned to the same machine. This process continues until the processing time on that machine is equal to Cmax.
- **Step 3:** If the last assigned job to the first machine is not finished at the time of Cmax, the remaining processing time of that job will be assigned to the second machine and the process continues.
- If the last assigned job to the first machine is finished at the time of Cmax, then another job will be assigned to the second machine and the process continues until all jobs are assigned.

Example
$P_3||C_{max}$

Jobs	1	2	3	4	5	6	7	8
Pi	1	2	3	4	5	6	7	8

Load per machine= 36/3=12


1.M
1
2
3
4
5
2.M
5
6
7
3.M
7
8
12

Non-Preemtive SPT

- Minimizing Flowtime

Jobs	1	2	3	4	5	6	7	8
Pi	1	2	3	4	5	6	7	8

Machine 1		
Jobs	Pi	Ci
1	1	1
4	4	5
7	7	12

Machine 2		
Jobs	Pi	Ci
2	2	2
5	5	7
8	8	15

Machine 3		
Jobs	Pi	Ci
3	3	3
6	6	9

Total Flow time = 54

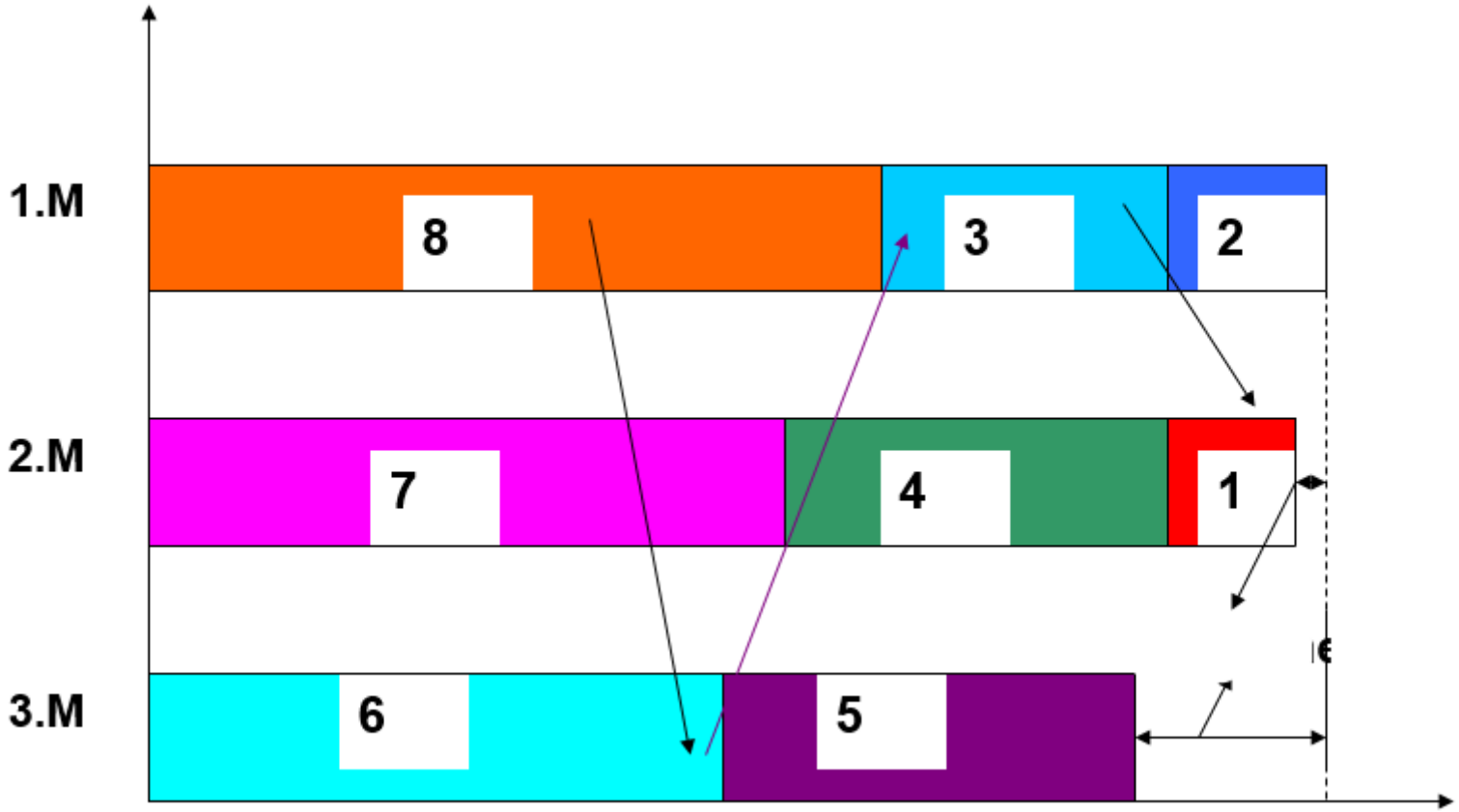
Non-Preemptive Case
Longest Processing Time (LPT) First Rule

- The problem Pm || Cmax is considered.
- The Makespan without Preemptions
- A common heuristics in parallel machine scheduling is LPT rule.
- Jobs are arranged in decreasing order of process times. The Longest Processing Time first (LPT) rule assigns at t = 0 the m longest jobs to the m machines. After that, whenever a machine is free the longest job among those not yet processed is put on the machine. This heuristic tries to place the shorter jobs more towards the end of the schedule, where they can be used for balancing the loads.

Example
$P_3||C_{max}$

Jobs	1	2	3	4	5	6	7	8
Pi	1	2	3	4	5	6	7	8

Order according to LPT: 8-7-6-5-4-3-2-1



Cmax=13
Load per machine= 36/3=12

- Load per machine= 36/3= 12

- Cmax optimal is 12.
- 13/12= 1.08
- LPT rule gives 8% more value than optimal.

Load Balancing Heuristics

	Jobs	Total Time
M1	J8, J4	12
M2	J7, J5	12
M3	J6,J3, J2,J1	12

Resources

- Sıralama ve Programlama, Hüseyin Başlıgil
- Çizelgeleme Ders Notları, Prof. Dr. Hüseyin Başlıgil
- Scheduling, Theory, Algorithms and Systems, Michael L. Pinedo, Third Edition, Springer, 2008.
- Algorithms for Sequencing and Scheduling, Ibrahim M. Alharkan
- web4.uwindsor.ca/users/b/baki%20fazle/Chapter_08_Lecture_12_to_19_w08_431_scheduling.ppt – Windsor University Operations Scheduling Lecture Notes
- https://fenix.tecnico.ulisboa.pt/downloadFile/282093452004307/5.1%20-%20Scheduling.pdf
- http://students.ceid.upatras.gr/~papagel/project/kef5_5.htm
- https://optimization.mccormick.northwestern.edu/index.php/Heuristic_algorithms
- https://www.careerride.com/OS-preemptive-scheduling.aspx
- https://www.techopedia.com/definition/5436/heuristic
- https://stackoverflow.com/questions/2334225/what-is-the-difference-between-a-heuristic-and-an-algorithm
- https://www.differencebtw.com/difference-between-algorithm-and-heuristic/