Operations Management I

Scheduling

Dong-Ho Lee

Department of Industrial Engineering
Hanyang University

Scheduling

Introduction
Performance Measures
Workforce Scheduling
Operations Scheduling

- What is Scheduling
- Basic Types (according to application areas)
- Gantt Charts

- Classification
- Basic Results

Hopp and Spearman, 2008, **Factory Physics**, McGraw Hill. **(Sections 15.1/15.2)**
Krajewski and Ritzman, 2005, **Operations Management**, Prentice Hall. **(Chapter 17)**

Scheduling

Introduction

What is Scheduling (1)

- Definition

  Allocation of resources over time to perform a collection of tasks
  (a critical link between planning and execution phases of operations)

  ◄- - - - Sequencing: establishing the order in which the jobs waiting in the queue for a resource have to processed

  e.g., machine scheduling
  - Resources ≡ machines
  - Tasks ≡ jobs

- Two aspects

  ✓ Theoretical aspect ◄- - - - Scheduling theory
  ✓ Practical aspect

  Development of mathematical (optimization) models that related to scheduling (quantitative approach in concise mathematical form)

Scheduling

- **Introduction**

What is Scheduling (2)

- Basic Terms
  - ✓ Operation

    An elementary task to be performed

    ◄- - - - Processing time: duration for an operation
    (include setup time if it is sequence-independent)

  - ✓ Job

    Set of operations that are interrelated by precedence restrictions

  - ✓ Routing

    Ordering of operations (with specification of machines)

Scheduling

- **Introduction**

Basic Types (according to application areas)

- Demand scheduling

  Assigning customers to a definite time of order fulfilment

  e.g., appointments, reservations, etc.

- Workforce scheduling

  Determining when employees work
  (specify the on-duty and off-duty periods for each employee over a certain time period)

  e.g., crew scheduling

- Operations scheduling

  Assigning jobs to workstations or employees to jobs for specified time periods

  e.g., production scheduling

Scheduling

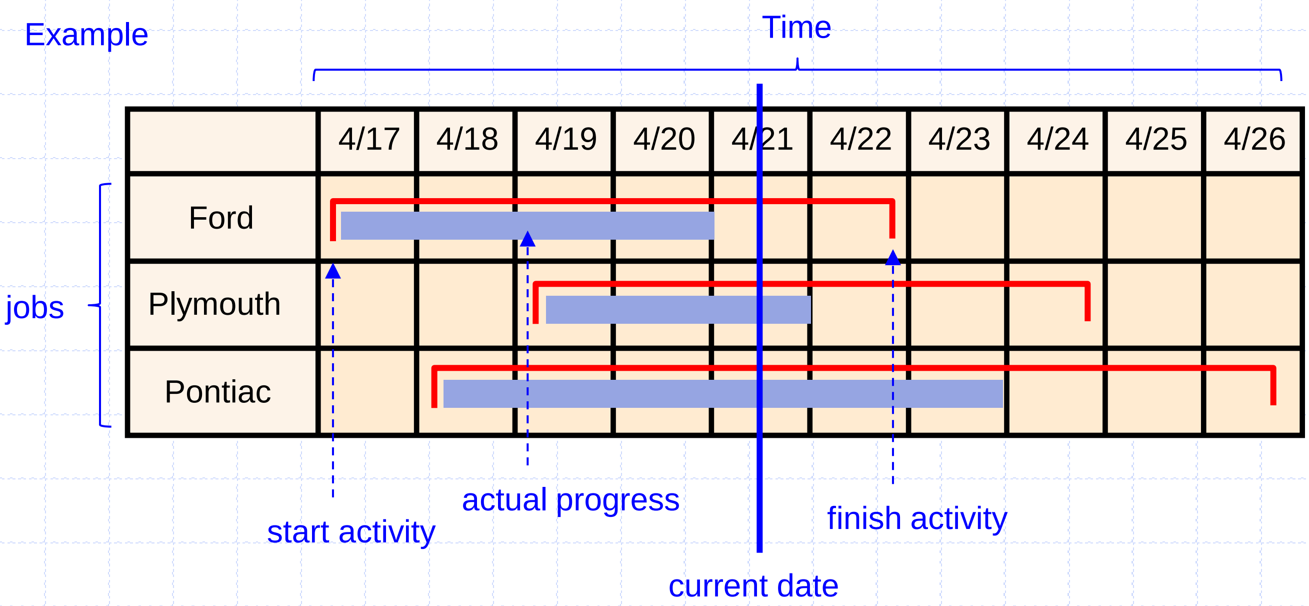
Introduction

Gantt Charts (1)

- Gantt progress chart

  Displays the current status of each job or activity relative to its scheduled completion date

Example



Scheduling

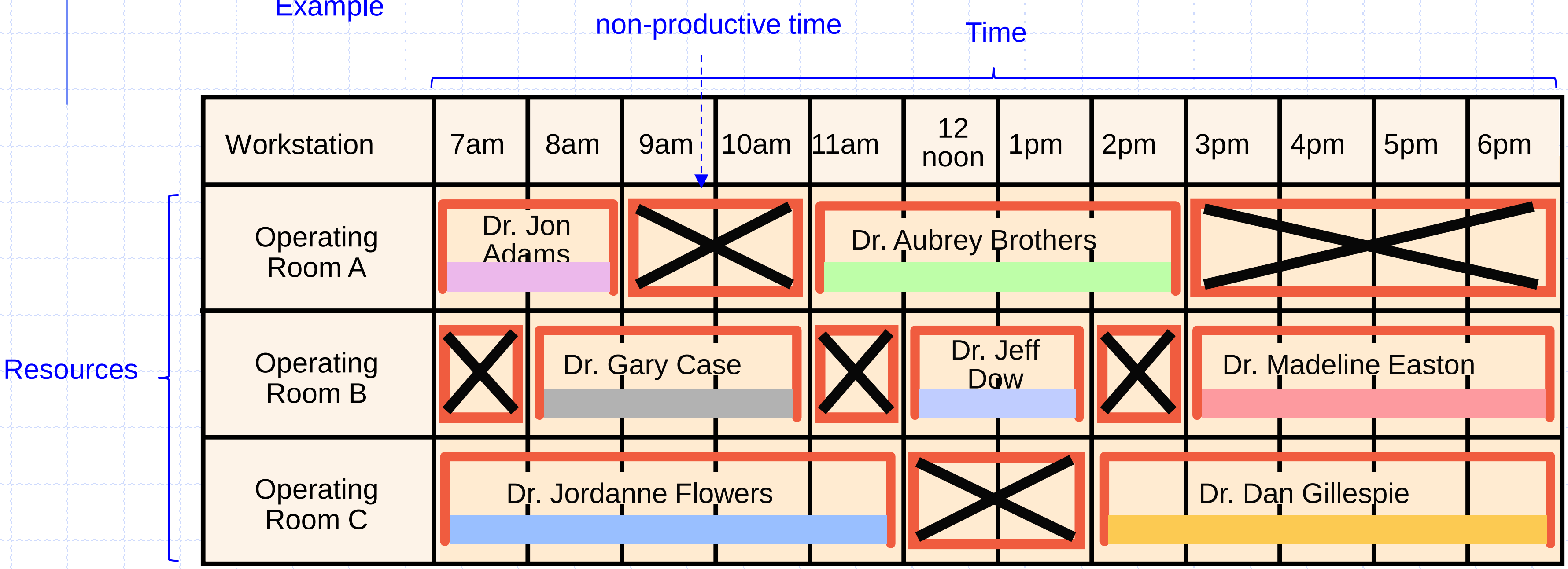
Introduction

Gantt Charts (2)

- Gantt workstation chart

  Displays the loads on resources and the non-productive time

Example



Scheduling

Performance Measures

Overview

- Maximizing system throughput (production rate)
- Maximizing utilization

  ← Good for cost accounting, provided that the equipment is utilized to increase revenue. Otherwise, it increases inventory, not profits.

  ← Minimizing makespan

- Minimizing work-in-process (WIP)

  ← Better responsiveness to the customer, improving quality, etc.

  ← Minimizing flow time

- Meeting due-dates

  ← Directly from customers (make-to-order environment)
  Material requirements for other manufacturing/service processes

  ← Minimizing lateness, tardiness
  Minimizing the maximum tardiness
  Minimizing the number of tardy jobs, etc.

Strike a profitable balance among conflicting goals of production and service systems

e.g., utilization ↑ → WIP ↑

Scheduling

◈ Performance Measures

Details (1)

- Makespan

  The time it takes to finish a fixed number of jobs
  (completion time of last job – starting time of first job)

  ←---- makespan ↓ → production rate (throughput) ↑
  → utilization ↑

✓ Mathematical representation

$M = \max\{C_j\}$ ←---- maximum of the completion times of a given set of jobs

C_j completion time of job j
(the time at which the processing of job *j* is finished)

Scheduling

- **Performance Measures**

Details (2)

- Flow time ← cycle time, throughput time, sojourn time, etc.

  The amount of time that a job spends in the system
  (from release of a job to completion)

  ← Reducing WIP ≡ Reducing the amount of time that a job spends in the system

  Little's law

  $L = \lambda \cdot W$ (WIP = throughput × cycle time)

✓ Mathematical representation

$F_j = C_j - r_j$ ←

C_j completion time of job j
(the time at which the processing of job *j* is finished)

r_j ready time of job *j* (= arrival time)
(the point in time at which job j is available for processing)

Scheduling

◈ Performance Measures

Details (3)

- Meeting due dates

  ✓ Lateness

  Amount of time by which the completion time of job *j* exceeds its due date

  $$L_j = C_j - d_j$$

  C_j completion time of job j
  (the time at which the processing of job *j* is finished)

  d_j due date of job *j*

  Average lateness has little meaning.
  - Jobs are finished near their due dates (good)
  - Every job is finished very late and one job is very early (bad)

  ✓ Tardiness

  $$T_j = \max\{0, L_j\}$$

  ✓ Earliness (just-in-time environment)

  $$E_j = \max\{0, -L_j\}$$

Scheduling

Performance Measures

Details (4)

- Aggregate performance measures
  - ✓ Makespan $M = \max\{C_j\}$
  - ✓ Mean flow time

$$\bar{F} = \frac{\sum_{j=1}^{n} F_j}{n} \longleftrightarrow \text{Total flow time } \sum_{j=1}^{n} F_j$$

  - ✓ Mean tardiness

$$\bar{T} = \frac{\sum_{j=1}^{n} T_j}{n} \longleftrightarrow \text{Total tardiness } \sum_{j=1}^{n} T_j$$

  - ✓ Maximum tardiness

$$T_{\max} = \max_{1 \le j \le n} \{T_j\}$$

  - ✓ Number of tardy jobs

$$N_T = \sum_{j=1}^{n} \delta(T_j) \longleftarrow \delta(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \end{cases}$$

Schedules are generally evaluated by aggregate quantities that involve information about all jobs, resulting in one-dimensional performance measures

- Function of completion times in a schedule

$$Z = f(C_1, C_2, \ldots, C_n)$$

Scheduling

Workforce Scheduling

Basic Model (1)

- Overview

  Workforce scheduling for a company that operates seven days a week

- Problem description

  ✓ Decision

  Determining when employees work (over a week)
  (specify the on-duty and off-duty day for each employee)

  ✓ Objective

  Minimize the amount of total slack capacity
  (≡ maximize the workforce utilization)

  ✓ Constraint

  Provide each employee with two consecutive days off

  e.g., Mon-Tue-Wed-Thu-Fri (on duty)
  Sat-Sun (off duty)

Scheduling

Workforce Scheduling

Basic Model (2)

- Procedure (Heuristic)

Step 1. From the schedule of net requirements for the week, find all the pairs of consecutive days that exclude the maximum daily requirements. Select the unique pair that has the lowest total requirements for the two days. (Ties are broken with a rule or arbitrary.)

Day	M	T	W	T	F	S	Su
Number of employees	8	9	2	12	7	4	2

Pair with the lowest total requirements

Step 2. Assign the employee the selected pair of days off. Subtract the requirements satisfied by the employee from the net requirements for each day the employee to work.

e.g., an employee is assigned Sat and Sun off.

Day	M	T	W	T	F	S	Su
Number of employees	7	8	1	11	6	4	2

Step 3. Repeat steps 1 and 2 until all the requirements have been satisfied or a certain number of employees have been scheduled,

Scheduling

- **Workforce Scheduling**

Basic Model (3)

Day	M	T	W	T	F	S	Su
Number of employees	6	4	8	9	**10***	3	2

Maximum requirement

Example

Employee	Requirements (remaining)							Work days						
	M	T	W	T	F	S	Su	M	T	W	T	F	S	Su
1	6	4	8	9	10*	3	2	X	X	X	X	X	Off	Off
2	5	3	7	8	9*	3	2	X	X	X	X	X	Off	Off
3	4	2	6	7	8*	3	2	X	X	X	X	X	Off	Off
4	3	1	5	6	7*	3	2	Off	Off	X	X	X	X	X
5	3	1	4	5	6*	2	1	X	X	X	X	X	Off	Off
6	2	0	3	4	5*	2	1	Off	Off	X	X	X	X	X
7	2	0	2	3	4*	1	0	X	X	X	X	X	Off	Off
8	1	0	1	2	3*	1	0	X	X	X	X	X	Off	Off
Tie breaks 9	0	0	0	1	2*	1	0	Off	X	X	X	X	X	Off
10	0	0	0	0	1*	0	0	X	X	X	X	X	Off	Off
				Capacity (C)				7	8	10	10	10	3	2
				Requirements (R)				6	4	8	9	10	3	2
				Slack (C – R)				1	4	2	1	0	0	0

Scheduling

◈ Operations Scheduling

Classification

- With respect to the behavior of jobs
  - ✓ Static scheduling

    Jobs arrive simultaneously, no more jobs will arrive until the end of scheduling horizon.
  - ✓ Dynamic scheduling

    New jobs arrive over time.
- With respect to the system configuration
  - ✓ Single machine scheduling
  - ✓ Parallel machine scheduling
  - ✓ Flow shop scheduling
  - ✓ Job shop scheduling

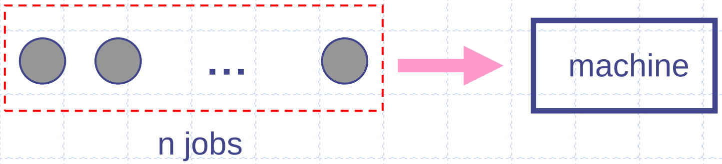
✓ Deterministic scheduling
✓ Stochastic scheduling

Scheduling

◈ Operations Scheduling

Single Machine Scheduling – Overview

- Configuration



- Assumptions (for basic single machine scheduling)

✓ Zero ready times
✓ Sequence independent setup times
✓ Deterministic job descriptors
✓ No preemption

A set of n independent, single operation jobs is available for processing simultaneously at time zero. (static scheduling)

Once an operation begins, it proceeds without interruption
(With preemption: preempt-resume and preempt-repeat)

Pure sequencing problem

Total number of distinct solutions = n!
(permutation schedule)

One-to-one correspondence between a sequence of *n* jobs and a permutation of job indices 1, 2, …, *n*

Scheduling

Operations Scheduling

Single Machine Scheduling – Basic Results (1)

- Minimizing makespan

Total time to complete all the jobs does not depend on the ordering.

Total time = makespan

$\sum_{j=1}^{n} t_j$ processing time of job j

- Minimizing mean flow time

In the basic single machine problem, mean flow time (total flow time) is minimized by shortest processing time (SPT) sequencing.

$$\bar{F} = \frac{\sum_{j=1}^{n} F_j}{n}$$

$$t_{[1]} \le t_{[2]} \le t_{[3]} \le \ldots \le t_{[n]}$$

[5] = 2 the fifth job in sequence in job 2

Jobs are ordered according to their processing times, with the shortest job first and longest job last.

✓ Related properties

- Mean flow time is directly proportional to WIP
(A schedule that minimizes mean flow time also minimizes WIP)

Scheduling

Operations Scheduling

Single Machine Scheduling – Basic Results (2)

- Minimizing mean lateness

  In the basic single machine problem, mean lateness is minimized by SPT sequencing.

$$\bar{L} = \frac{\sum_{j=1}^{n} L_j}{n}$$

$$L_j = C_j - d_j$$

$$T_j = \max\{0, L_j\}$$

$$L_{\max} = \max_{1 \le j \le n} \{L_j\} \qquad T_{\max} = \max_{1 \le j \le n} \{T_j\}$$

- Minimizing maximum lateness/tardiness

  In the basic single machine problem, the maximum job lateness and the maximum job tardiness are minimized by earliest due date (EDD) sequencing. .

$$d_{[1]} \le d_{[2]} \le d_{[3]} \le \ldots \le d_{[n]}$$

$d_{[1]}$ due date of the first job in sequence

- Minimizing mean tardiness (total tardiness)

  Difficult to solve for large sized problems
  (various optimization techniques, heuristics)

Scheduling

- **Operations Scheduling**

Single Machine Scheduling – Basic Results (3)

- Minimizing the number of tardy jobs

✓ Optimal algorithm (by Hodgson (Moore))

Step 1. Place all jobs in EDD order

Step 2. If no jobs are late, optimal. Otherwise, go to Step 3.

Step 3. Identify the first job that is late (*k*th job)

Step 4. Identify the longest job among the first *k* jobs. Remove it and re-compute the completion times. (Removed jobs are late jobs.) Go to Step 2.

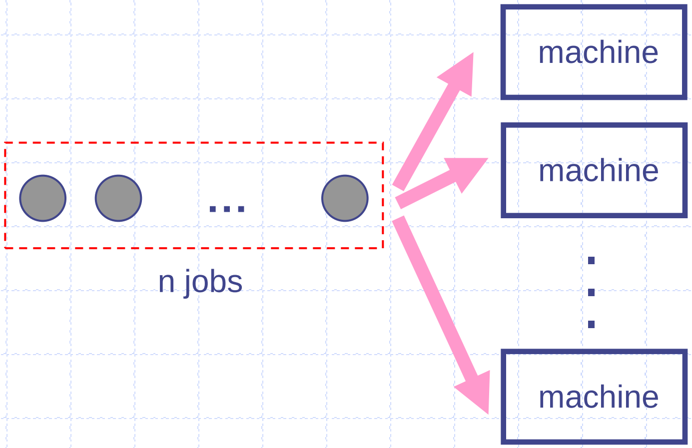
Scheduling

Operations Scheduling

Parallel Machine Scheduling – Overview

- Configuration

  ✓ m different machines in parallel



✓ How to allocate and sequence the n jobs to optimize a given performance measure? (allocation and sequencing)

- Assumptions (for basic parallel machine scheduling)

  ✓ Identical parallel machines
  ✓ Zero ready times
  ✓ Sequence independent setup times
  ✓ Deterministic job descriptors
  ✓ No preemption

Scheduling

Operations Scheduling

Parallel Machine Scheduling – Basic Results (1)

- Minimizing makespan ← - - - - Bin-packing problem (no simple method to obtain the optimal solution)

✓ Heuristic algorithm – LPT heuristic

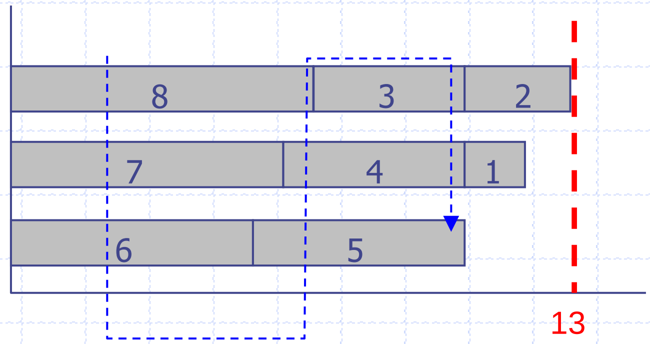
Step 1. Construct an LPT (longest processing time) ordering of the jobs

Step 2. Schedule the jobs in order, each time assigning a job to the machine with the least amount of processing already assigned.

e.g.

Job	1	2	3	4	5	6	7	8
Processing time	1	2	3	4	5	6	7	8

- LPT order: 8 → 7 → 6 → 5 → 4 → 3 → 2 → 1



- Minimizing mean tardiness (total tardiness)
- Minimizing maximum lateness/tardiness
- Minimizing the number of tardy jobs

  Difficult to solve for large sized problems
  (various optimization techniques, heuristics)

Scheduling

Operations Scheduling

Parallel Machine Scheduling – Basic Results (2)

- Minimizing mean flow time

✓ Optimal algorithm ◄- - - - An extension of the SPT sequencing in single machine

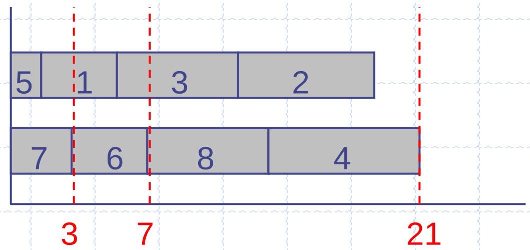
Step 1. Construct SPT ordering of all jobs.

Step 2. To the machine with the least amount of processing already allocated, assign the next job on the ordered list of jobs. (arbitrary tie breaks). Repeat until all jobs are assigned.

e.g.

Job	1	2	3	4	5	6	7	8
Processing time	4	7	6	8	1	4	3	6

- Two machines (m = 2)
- SPT order: 5 → 7 → 1 → 6 → 3 → 8 → 2 → 4



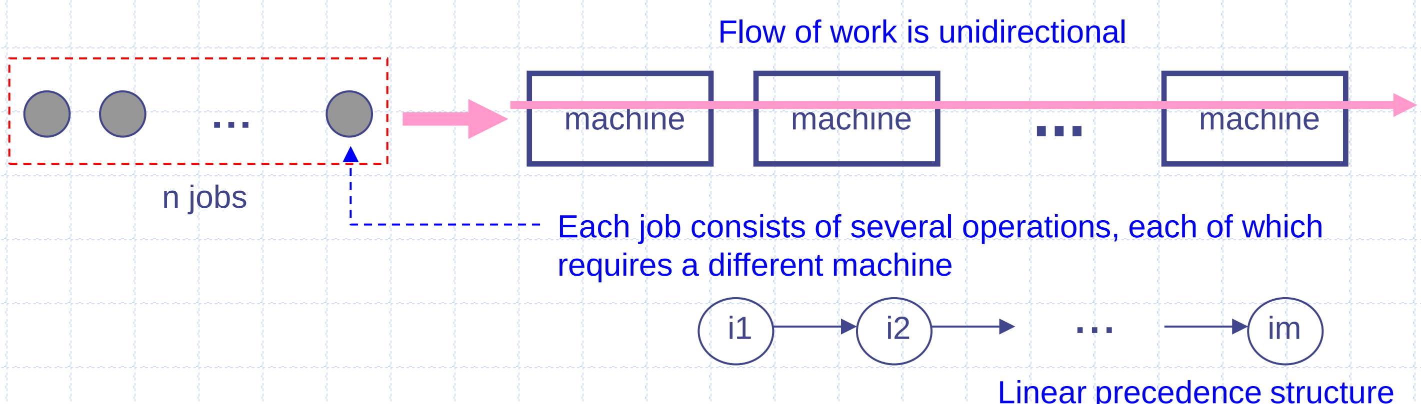
Job	Flow time
1	5
2	18
3	11
4	21
5	1
6	7
7	3
8	13

Scheduling

◈ Operations Scheduling

Flow Shop Scheduling – Overview

- Configuration
  - ✓ m different machines in series



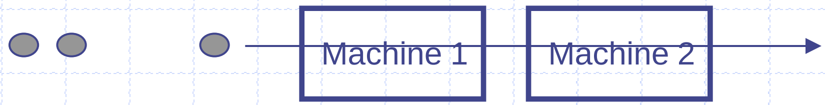
- ✓ How to sequence the n jobs at each machine to optimize a given performance measure?

Scheduling

Operations Scheduling

Flow Shop Scheduling – Minimizing Makespan (1)

- Two-machine flow shop – Johnson's problem



t_{j1} processing time of job j on machine 1
t_{j2} processing time of job j on machine 2

✓ How to sequence the jobs at both machine?

✓ Assumptions (for basic flow shop scheduling)
- Zero ready times
- Sequence independent setup times
- Deterministic job descriptors
- No preemption

→ Same sequence on both machines

Scheduling

Operations Scheduling

Flow Shop Scheduling – Minimizing Makespan (2)

- Two-machine flow shop – Johnson's problem
  - ✓ Optimal algorithm

Step 1. Find the minimum processing time among unscheduled jobs.

Step 2a. If the minimum in Step 1 occurs on machine 1, place the associated job in the first available position in sequence. (Ties broken arbitrarily.) Go to Step 3.

Step 2b. If the minimum in Step 1 occurs on machine 2, place the associated job in the last available position in sequence. (Ties broken arbitrarily.) Go to Step 3.

Step 3. Remove the assigned job from consideration and return to Step 1 until all sequence positions are filled.

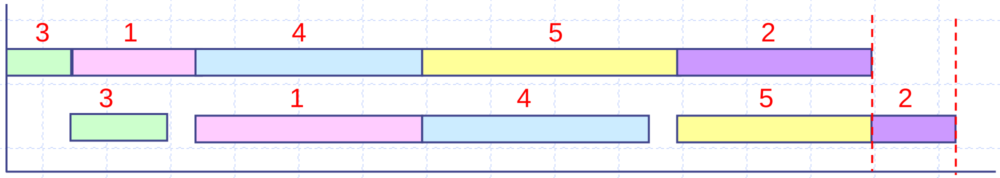
Scheduling

Operations Scheduling

Flow Shop Scheduling – Minimizing Makespan (3)

- Two-machine flow shop – Johnson's problem

Job	1	2	3	4	5
Processing time on machine 1	4	6	2	7	8
Processing time on machine 2	7	3	3	7	6



3 → 1 → 4 → 5 → 2

Step 1. Unscheduled Jobs = {1, 2, 3, 4, 5}
The minimum processing time = 2

Step 2. Job 3 on machine 1 (First available position)
3 --------

Step 3. Unscheduled Jobs = {1, 2, 4, 5}

Step 1. The minimum processing time = 3

Step 2. Job 2 on machine 2 (last available position)
3 -------- 2

Step 3. Unscheduled Jobs = {1, 4, 5}

Step 1. The minimum processing time = 4

Step 2. Job 1 on machine 1 (first available position)
3 1 -------- 2

Step 3. Unscheduled Jobs = {4, 5}

Step 1. The minimum processing time = 6

Step 2. Job 5 on machine 2 (last available position)
3 1 -------- 5 2

Scheduling

Operations Scheduling

Flow Shop Scheduling – Minimizing Makespan (4)

- m-machine flow shop
  - ✓ Difficult to solve for large sized problems (various optimization techniques, heuristics)
  - ✓ Types of schedules
    - ➢ Permutation schedule

      A schedule with the same job order on all machines

      ←---- A schedule that is completely characterized by a single permutation of the job indices (n! for n jobs)

      e.g., m = 2 permutation schedule by the Johnson's algorithm
    - ➢ Nonpermutation schedule

      All possible schedules

      ←---- $(n!)^m$ for n jobs and m machines

Scheduling

Operations Scheduling

Flow Shop Scheduling – Minimizing Makespan (5)

- m-machine flow shop
  - ✓ Heuristics (for permutation schedule)
    - ➢ CDS (Campbell, Dudek and Smith) algorithm (1970)

      A multistage use of Johnson's rule

      - Stage 1 $\left.\begin{aligned} t_{j1}' &= t_{j1} \\ t_{j2}' &= t_{jm} \end{aligned}\right\} \longrightarrow$ Apply Johnson's rule
      - Stage 2 $\left.\begin{aligned} t_{j1}' &= t_{j1} + t_{j2} \\ t_{j2}' &= t_{j,m-1} + t_{jm} \end{aligned}\right\} \longrightarrow$ Apply Johnson's rule

        ⋮
      - Stage i $\left.\begin{aligned} t_{j1}' &= \sum_{k=1}^{i} t_{jk} \\ t_{j2}' &= \sum_{k=1}^{i} t_{j,m-k+1} \end{aligned}\right\} \longrightarrow$ Apply Johnson's rule

        ⋮
      - Stage m – 1

      Select the best one among *m* -1 schedules

- Minimizing mean flow time (total flow time)
- Minimizing mean tardiness (total tardiness)
- Minimizing maximum lateness/tardiness
- Minimizing the number of tardy jobs

Difficult to solve for large sized problems
(various optimization techniques, heuristics)

Scheduling

Operations Scheduling

Flow Shop Scheduling – Minimizing Makespan (6)

- m-machine flow shop
  - ✓ Heuristics (for permutation schedule)
    - ➢ NEH (Nawaz, Enscore and Ham) algorithm (1983)

Step 1. Order n jobs by decreasing order of processing times on the machines.

Step 2. Take the first two jobs and schedule them in order to minimize the partial makespan

Step 3. For $k = 3$ to n, do

The current partial schedule contains $k – 1$ jobs.
Insert kth jobs at the place which minimizes the makespan among the k possible positions available.

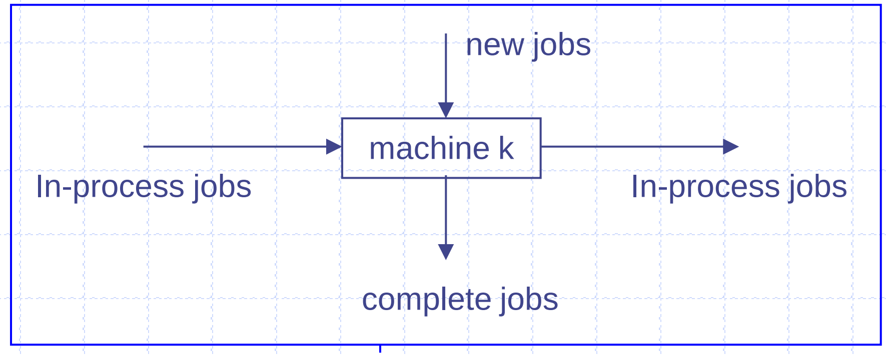
- ✓ Meta-heuristics
  - ➢ Simulated annealing (SA)
  - ➢ Tabu search (TS)
  - ➢ Genetic algorithm (GA), etc.

Scheduling

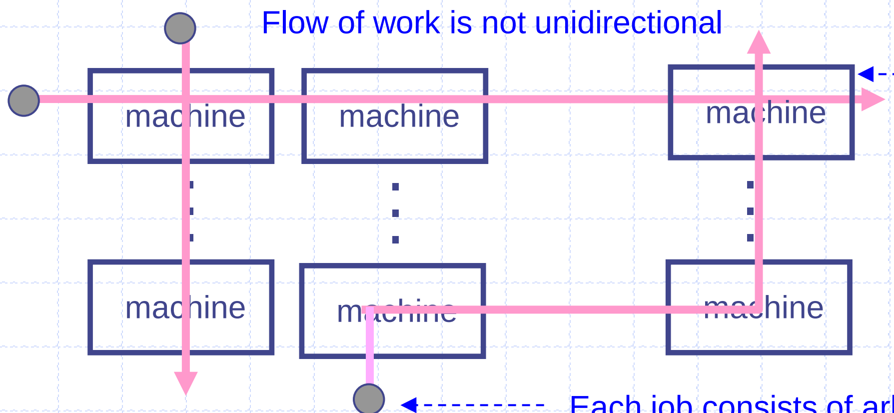
Operations Scheduling

Job Shop Scheduling – Overview (1)

- Configuration



Flow of work is not unidirectional



Each job consists of arbitrary number of operations, each of which requires a machine.
(Routing: set of machine assignments for a given job)

✓ Triplet (i, j, k): jth operations of job i is processed at machine k

✓ How to sequence the jobs at each machine to optimize a given performance measure?

An extension: FMS scheduling (with alternative machines)

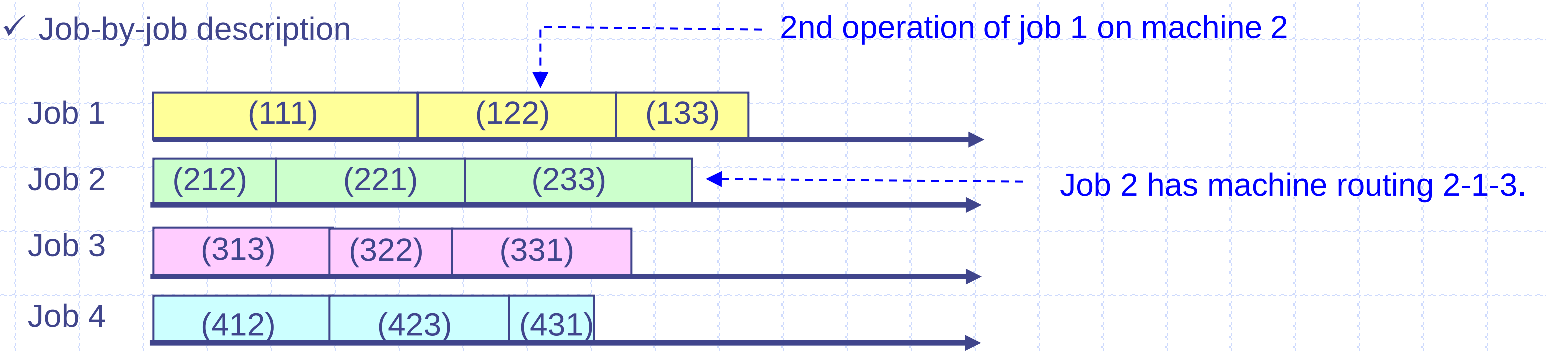
Scheduling

Operations Scheduling

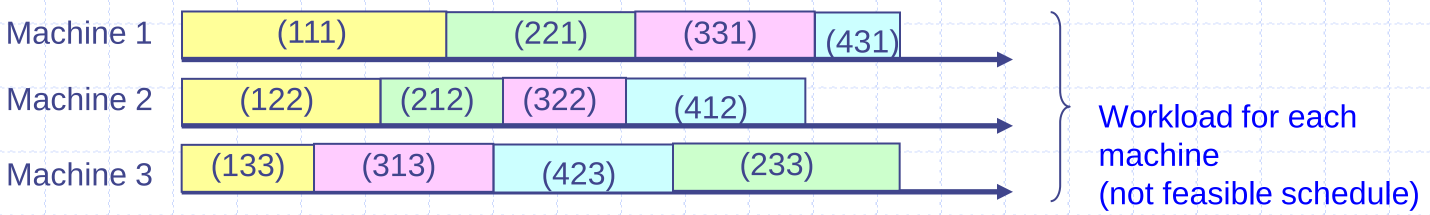
Job Shop Scheduling – Overview (2)

- Graphical Description – Gantt charts (infeasible schedule)

✓ Job-by-job description



✓ Machine-by-machine description



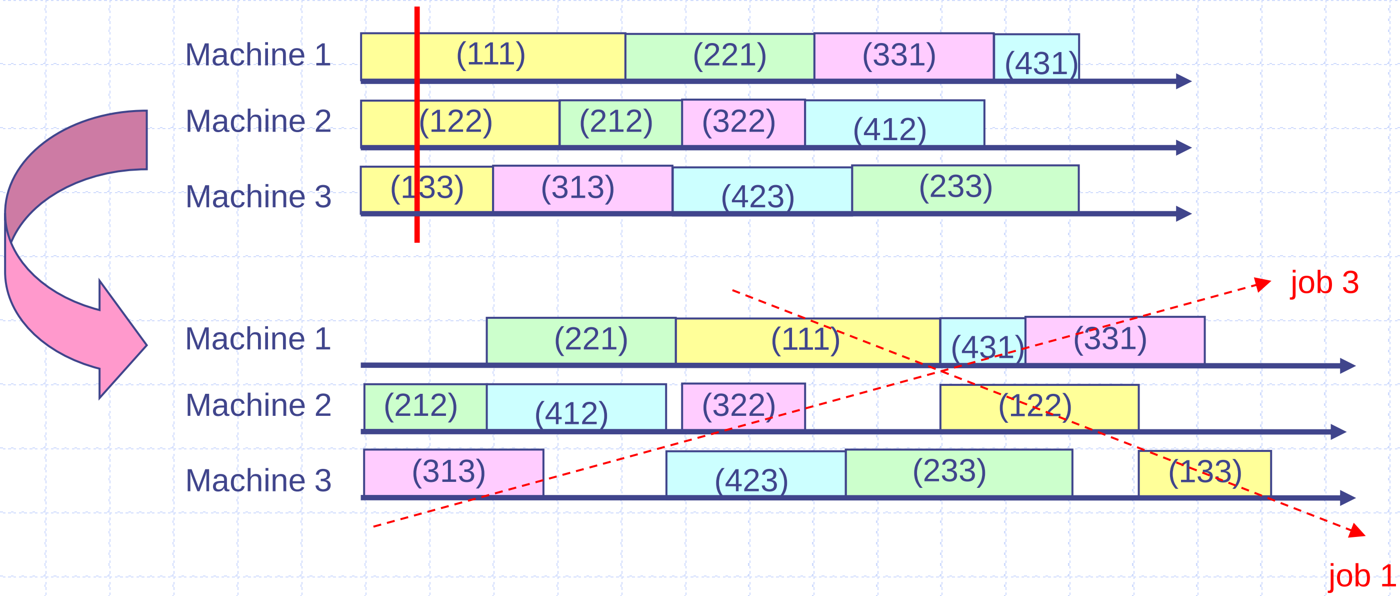
Scheduling

- **Operations Scheduling**

  Job Shop Scheduling – Overview (3)

  - Graphical Description – Feasibility conditions

    ✓ All operations of a job can be placed on one time axis in precedence order and without overlap. (operation precedence in a job)

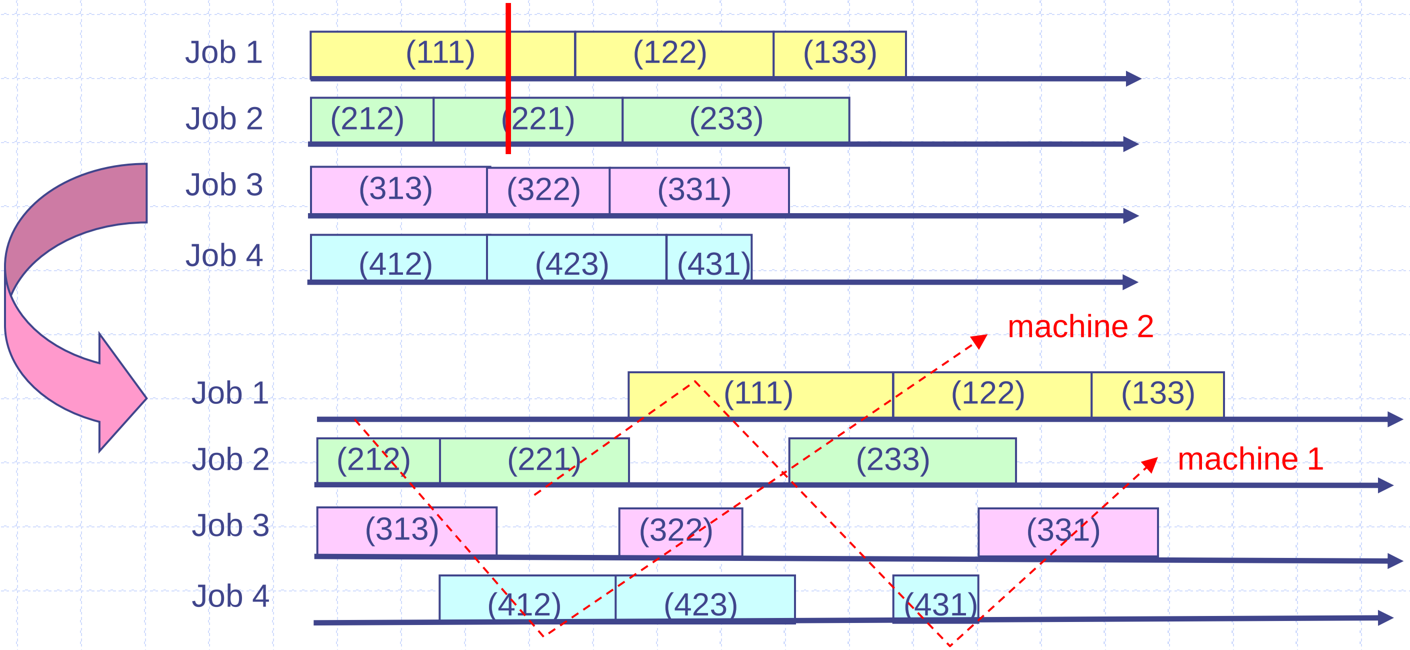


Scheduling

◆ Operations Scheduling

Job Shop Scheduling – Overview (4)

- Graphical Description – Feasibility conditions
  - ✓ No two jobs can occupy the same machine simultaneously. (resource constraint)



Scheduling

Operations Scheduling

Job Shop Scheduling – Minimizing Makespan (1)

- Two-machine problem

✓ Jackson's algorithm (optimal) ◂----- an extension of Johnson's algorithm (for two-machine flow shop)

Step 1. Define jobs into 4 sets.

- ✓ Set {A}: jobs that are processed only on machine 1
- ✓ Set {B}: jobs that are processed only on machine 2
- ✓ Set {AB}: jobs that are processed on machine 1 and then machine 2
- ✓ Set {BA}: jobs that are processed on machine 2 and then machine 1

Step 2. Sequence jobs in {AB} with Johnson's algorithm.
Sequence jobs in {BA} with Johnson's algorithm.

Step 3. Schedule jobs as

- ✓ Machine 1: jobs in {AB} → jobs in {A} → jobs in {BA}
- ✓ Machine 2: jobs in {BA} → jobs in {B} → jobs in {AB}

Scheduling

Operations Scheduling

Job Shop Scheduling – Minimizing Makespan (2)

- Two-machine problem
  - ✓ Jackson's algorithm (optimal)

Example

Job	Machine number		Processing time	
	Oper 1	Oper 2	Oper 1	Oper 2
1	1	2	4	6
2	1		5	
3	1		4	
4	1	2	5	2
5	2	1	1	2
6	2		1	
7	2	1	7	8
8	2		3	
9	2	1	6	7
10	1	2	2	4

A = {2, 3}
B = {6, 8}
AB = {1, 4, 10}
BA = {5, 7, 9}

Machine	Job		
	1	4	10
1	4	5	2
2	6	2	4

Sequence by Johnson's algorithm 10 – 1 – 4

Machine	Job		
	5	7	9
2	1	7	6
1	2	8	7

Sequence by Johnson's algorithm 5 – 9 – 7

{A}

Sequence on machine 1: $\underbrace{10 – 1 – 4}_{\{AB\}}$ – 2 – 3 – 5 – 9 – 7 {BA}

{B}

Sequence on machine 2: $\underbrace{5 – 9 – 7}_{\{BA\}}$ – 6 – 8 – $\underbrace{10 – 1 – 4}_{\{AB\}}$

w_{ij} start time of operation *j* of job *i* (j = 1, 2,…J_i)

$\gamma^k_{(i,j)(i',j')}$ = 1 if operation *j* of job *i* precedes operation *j'* of job *i'* on machine *k*, and 0 otherwise $((i,j),(i',j') \in O_k)$

(O_k set of operations processed on machine *k*)

t_{ij} processing time of operation *j* of job *i*

M a large number

Scheduling

Operations Scheduling

Job Shop Scheduling – Minimizing Makespan (3)

- Two-machine problem
  - ✓ Integer programming model ← - - - - - - -
    - Branch and bound algorithm
    - Meta-heuristics

$$\text{Min Max } w_{iJ_i}$$

subject to

$$w_{ij} - w_{i,j-1} \geq t_{i,j-1} \quad \text{for all } i, j$$

$$w_{i'j'} - w_{ij} + M \cdot (1 - \gamma^k_{(i,j)(i',j')}) \geq t_{ij} \quad \text{for all } (i,j),(i',j') \in O_k$$

$$w_{ij} - w_{i'j'} + M \cdot \gamma^k_{(i,j)(i',j')} \geq t_{i'j'} \quad \text{for all } (i,j),(i',j') \in O_k$$

$$w_{ij} \geq 0 \quad \text{for all } i, j$$

$$\gamma^k_{(i,j)(i',j')} \in \{0,1\} \quad \text{for all } (i,j),(i',j') \in O_k \text{ and } k$$

Scheduling

Dispatching rules (priority rules)

- ✓ Blackstone, Jr., J. H., D. T. Philips, and D. L. Hogg, 1982, A state-of-the-art Survey of Dispatching Rules for Manufacturing Jobshop Operations, *International Journal of Production Research*, **20**, 27-45.
- ✓ Panwalkar, S. S. and W. Iskander, 1977, A survey of Scheduling Rules, *Operations Research*, **25**, 45-61.

Operations Scheduling

Job Shop Scheduling – Minimizing Makespan (4)

- Two-machine problem
  - ✓ Dispatching rules
    - Without due-date considerations

FCFS (first come first served)

processing time of operation *j*

SPT (shortest processing time): min t_j ($\rightarrow$ SPT^x)

MWKR (most work remaining): max r_j

LWKR (least work remaining): min r_j

remaining work of operation *j* (sum of processing times of the successor operations including itself)

MOPR (most operation remaining): max o_j

LOPR (least operation remaining): min o_j

remaining operations of operation *j* (number of successor operations including itself)

PWKR (processing time/work remaining): min t_j / r_j

POPR (processing time/operation remaining): min t_j / o_j

- Minimizing mean tardiness (total tardiness)
- Minimizing maximum lateness/tardiness
- Minimizing the number of tardy jobs

Difficult to solve for large sized problems
(various optimization techniques, heuristics))

Scheduling

Operations Scheduling

Job Shop Scheduling – Minimizing Makespan (5)

- Two-machine problem
  - ✓ Dispatching rules
    - With due-date considerations

EDD (earliest due date)

STR (slack time remaining)

min $(d_j - t - r_j)$

STR/OP (slack time remaining per operation)

min $[(d_j - t - r_j) / o_j]$

CR (critical ratio)

min $[(d_j - t) / r_j]$

CR > 1: early
CR < 1: late
CR < 0: already late

e.g., machine (processing time)

Job	operation		
	1	2	3
1	1(4)	2(3)	3(2)
2	2(1)	1(4)	3(4)
3	3 (3)	2 (2)	1(3)
4	2(3)	3 (3)	1 (1)