

Scheduling of Jobs

IE 3265 – POM

R. Lindeke

Spring 2005



Focus of Job Scheduling:

- Scheduling to Control Bottleneck Machines
 - The n on 1 problem
- Scheduling of Parallel Machines
 - The n on m_{equal} problem
- Scheduling of Serial Processes
 - N on 2 or n on m problem
- Scheduling Goals
 - Thru-put and make span control
 - Lateness control
 - Minimization of WIP
 - Minimization of Idle times

Issues in Scheduling (sequencing of Jobs)

- Based on Established Job Priorities and the way arrivals are treated:
 - Static Arrival means we only enter jobs into consideration at fixed points in time – uses job accumulation
 - Dynamic Scheduling means that jobs are dispatched as they arrive with continuously revised job priorities
- Types and numbers of workers and machines available
 - Most scheduling algorithms are machine-limited (assumes labor is always available but machine time is the problem)
 - Newer algorithms are studying Labor-limited systems (due to cells and multi-tasking workers)
- Internal Flow patterns
- Priority Rules for job allocation

Typical (Common) Sequencing Rules that we should consider

- **FCFS**. First Come First Served. Jobs processed in the order they come to the shop.
- **SPT**. Shortest Processing Time. Jobs with the shortest processing time are scheduled first.
- **EDD**. Earliest Due Date. Jobs are sequenced according to their due dates.
- **CR**. Critical Ratio. Compute the ratio of processing time of the job and remaining time until the due date. Schedule the job with the smallest CR value next.

Examining the scheduling of n on 1 – or the handling of the bottleneck machine

- We will examine several approaches and select ones that best meet varying priority decision making
 - Like Min. Mean Flow Time
 - Average Job tardiness
 - Number of Tardy Jobs
- Let's try FCFS, SPT, EDD and CR

Example:

- Machine shop has 5 unprocessed jobs (J1, J2, J3, J4, J5) numbers by order they entered Bottleneck machines queue:

Job #	Process Time	Due Date
1	11	61
2	29	45
3	31	31
4	1	33
5	2	32

Using FCFS:

Sequence	Comp. Time	D. Date	Tardiness
J1	11	61	0
J2	40	45	0
J3	71	31	40
J4	72	33	39
J5	74	32	42
Totals	268		121

- Mean Flow Time: $(268)/5 = 53.4$
- Avg Tardiness: $(121)/5 = 24.2$
- # Tardy Jobs: 3

SPT:

Sequence	Comp. Time	D. Date	Tardiness
J4	1	61	0
J5	3	45	0
J1	14	31	0
J2	43	33	10
J3	74	32	42
Totals	135		52

- Mean Flow Time: $(135)/5 = 27$.
- Avg Tardiness: $(52)/5 = 10.4$
- # Tardy: 2

EDD:

Sequence	Comp. Time	D. Date	Tardiness
J3	31	31	0
J5	33	32	1
J4	34	33	1
J2	63	45	18
J1	74	61	13
Totals	235		33

- Mean Flow Time: $(235)/5 = 47$.
- Avg Tardiness: $(33)/5 = 6.6$
- # Tardy: 4

CR: This is an Iterative Process using this model:

- Set Current Time (sum of time of all scheduled jobs so far)
- Compute:
$$CR = \frac{(Due_Date) - (Cur_Time)}{Pr._Work_Remaining}$$
- Model Starts with current time = 0
- Current time updates after each selection by adding scheduled Process Time to current time

Try it:

JOB	Pr. Time	D. Date	CR
Current Time = 0			
1	11	61	5.546
2	29	45	1.552
3	31	31	1.00
4	1	33	33
5	2	32	16

JOB	Pr. Time	D. Date	CR
Current Time = 31			
1	11	61	2.727
2	29	45	.483
4	1	33	2
5	2	32	0.5

Continuing (CR)

JOB	Pr. Time	D. Date	CR
Current Time = 60			
1	11	61	0.091 do last
4	1	33	-27*
5	2	32	-14**

JOB	C. Time	D. Date	Tardy
Summary			
3	31	61	0
2	60	45	15
4	61	31	28
5	63	33	31
1	74	32	13
Total:	289		87

Summarizing from CR analysis:

- Mean F. Time: $(289)/5 = 57.8$
- Mean Tardiness: $(87)/5 = 17.4$
- # Tardy: 4

Comparing the Algorithm's for Bottleneck Sequencing:

- Total Makespan is independent of sequencing algorithm
- SPT will guarantee minimum Mean Flow time
- Minimize Maximum lateness with EDD model
- Minimize # Tardy jobs starts with EDD, see text – Moore's Algorithm – for rule to do this.
- Minimizing Mean Tardiness uses Wilkerson-Irwin Algorithm

Wilkerson-Irwin Algorithm:

- Step 1: initialize jobs in **EDD order**.
- Compare 1st two jobs in EDD list if $\max(t_a, t_b) \leq \max(d_a, d_b)$ assign a to Col 1 and b to Col 2.
 - Else assign SPT task to Col. 1 other to Col. 2. Place the 3rd job from EDD list to Col. 1.
- Step 2: Compare 1st job to 2nd job to see if 1st will join 2nd on tentative scheduled list. If $t_1 \leq t_2$ **-OR-** $F_1 + \max(t_1, t_2) \leq \max(d_1, d_2)$ move Job 1 to col. 1, Job 2 to col. 2 and next job on EDD list to col. 1.
 - If no more jobs are in EDD list, add current col. 1 and col. 2 jobs to schedule and stop. Else repeat Step 2. If both tests fail go on to step 3.

Wilkerson-Irwin Algorithm:

- Step 3: Put Job in Col. i back to top of EDD job list and move Col. i job to col. i . Compare last col. i job to this col. i job to see if col. i job will join the tentative schedule.
 - If $t_i \leq t_i$ -OR- $F_i - t_i + \max(t_i, t_i) \leq \max(d_i, d_i)$ move col. i job to col. i and select next 2 jobs off EDD list. Return to Step 2. If both tests fail go to Step 4.
- Put job in Col. i back to top of EDD list. Assign last scheduled job back to the Col. i . Return to Step 3.
 - If no jobs are on scheduled list, put Job in Col. i on schedule list. Let 1st two jobs on EDD list become Col. i and Col. i jobs. Return to Step 2.

Lets apply it to our 5 Job Example

Sequence	Comp. Time	D. Date	Tardiness
J3	31	31	0
J5	33	32	1
J4	34	33	1
J2	63	45	18
J1	74	61	13
Totals	235 (47)		33 (6.6)

Step by Step:

Col. 1	Col. 2	Col. 3	Scheduled	By Step?
J3	J5	J4		1
$t_5 \leq t_4$ (no); $F_5 + \text{MAX}(t_5, t_4) \leq \text{MAX}(d_5, d_4)$ (yes)				2
J3, J5	J4	J2	J3	
$t_4 \leq t_2$ (yes); $F_4 + \text{MAX}(t_4, t_2) \leq \text{MAX}(d_4, d_2)$ (no)				2
J3, J5, J4	J2	J1	J3, J5	
$t_2 \leq t_1$ (no); $F_2 + \text{MAX}(t_2, t_1) \leq \text{MAX}(d_2, d_1)$ (no)				2
J3, J5, J4	J1		J3, J5	
$t_4 \leq t_1$ (yes); $F_4 - t_4 + \text{MAX}(t_4, t_1) \leq \text{MAX}(d_4, d_1)$ (yes)				3
J3, J5, J4, J1			J3, J5, J4, J1, J2	

Summary:

Sequence	Comp. Time	D. Date	Tardiness
J3	31	31	0
J5	33	32	1
J4	34	33	1
J1	45	61	0
J2	74	45	29
Totals	217		31

- Mean Flow Time: $(217)/5 = 43.4$
- Mean Tardiness: $(31)/5 = 6.2$
- # Tardy: 3

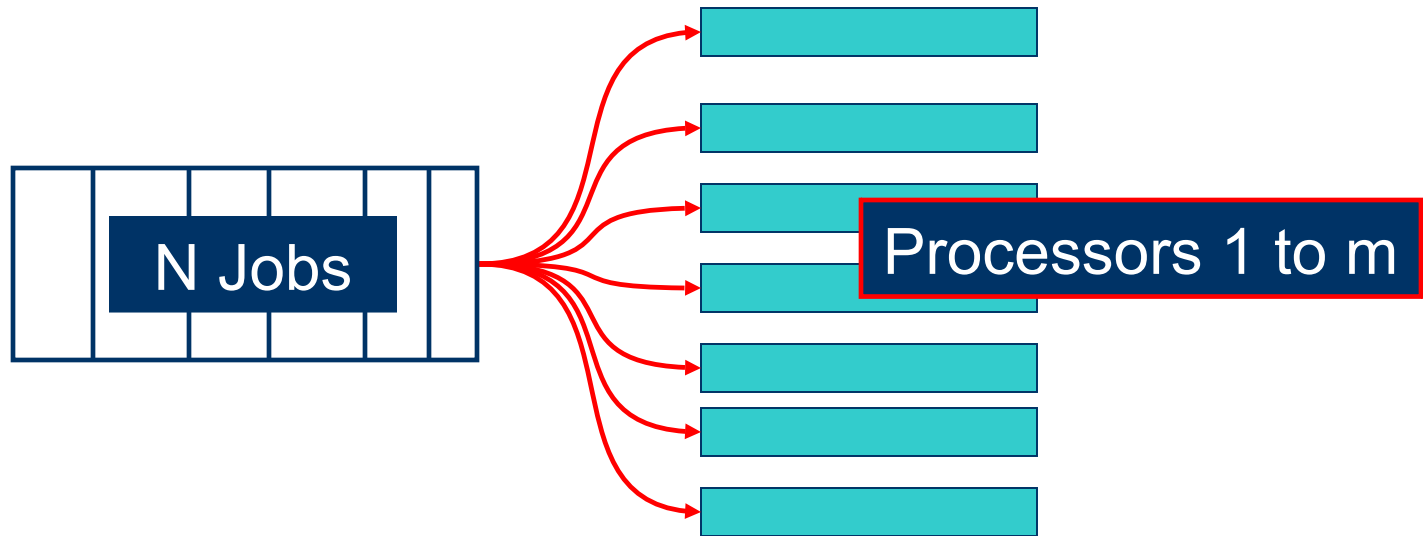
Some Issues:

- All of these algorithms ignore Constraints that may be desired/required
 - Valued Customers orders
 - Promises to others
 - Rush Jobs
 - Rework/Scrap, etc.
- These issues must be considered and the rest of jobs must be scheduled around them

You try one!

Job #	Process Time	Due Date
1	15	45
2	9	60
3	41	44
4	6	50
5	10	30

Looking at Parallel Machine Problem:



Algorithms to Minimize Mean Flow Time and/or Makespan:

- One For Minimizing Mean Flow:
 - Step 1: Sequence Jobs by SPT
 - Step 2: Take Jobs from list and assign to the processor with **least amount** of assigned time
 - Step 3: Continue thru all jobs
- One to Minimize Makespan while *controlling* Mean Flow:
 - Step 1: Sequence Jobs in Longest Processing time (LPT) order
 - Step 2: Take Jobs from list and assign to the processor with **least amount** of assigned time
 - Step 3: Reverse scheduled tasks on each processor to set the processing order

Lets try one:

Given Order		SPT Order		LPT Order	
Job #	Pr. Time	Job #	Pr. Time	Job #	Pr. Time
1	5	6	2	4	8
2	6	10	2	5	7
3	3	3	3	2	6
4	8	7	3	1	5
5	7	9	4	8	5
6	2	1	5	9	4
7	3	8	5	3	3
8	5	2	6	7	3
9	4	5	7	6	2
10	2	4	8	10	2

SPT Algorithm

Job #	Pr. Time	M1	M2	M3
6	2	#6 - 2		
10	2		#10 - 2	
3	3			#3 - 3
7	3	#7 - 5		
9	4		#9 - 6	
1	5			#9 - 8
8	5	#8 - 10		
2	6		#2 - 12	
5	7			#5 - 15
4	8	#4 - 18		

Presented as Gantt Chart:

Makespan is total time start to finish: 18 days

ID

Computing Mean Flow

$$Mean_Flow = \frac{\sum_{M=1}^{tot_machines} M_Flow_M}{M}$$

$$M_Flow_M = \frac{\sum_{N=1}^{all_Jobs_Assigned} Comp_Time_N}{N}$$

$$M_Flow_1 = \frac{(2 + 5 + 10 + 18)}{4} = 8.75$$

$$Mean_Flow_{sch} = \frac{(8.75 + 6.67 + 8.67)}{3} = 8.03$$

LPT Approach:

Job #	Pr. Time	M1	M2	M3
4	8	#4 - 8		
5	7		#5 - 7	
2	6			#2 - 6
1	5			#1 - 11
8	5		#8 - 12	
9	4	#9 - 12		
3	3			#3 - 14
7	3	#7 - 15		
6	2		#6 - 14	
10	2		#10 - 16	

To schedule: **REVERSE ORDER** on each machine

Presented as Gantt Chart:

ID

Makespan is total time start to finish: 16 days

Computing Mean Flow:

- $M_Flow_2 = \square(2 + 4 + 9 + 16)/4 = 7.75$
- $M_Flow_{sys} = \square(8.33 + 7.75 + 4.67)/3 = 6.917$
- Now Considering Idle time:
 - SPT:
 - Total Idle is $6 + 3 = 9$
 - M_Idle is $9/3 = 3$
 - LPT:
 - Total Idle is $4 + 1 = 5$
 - M_Idle is $5/3 = 1.667$

Considering Lateness

- To minimize Maximum Tardiness:
 - Step 1: Sequence task by EDD
 - Step 2: Take tasks from EDD order list then schedule them on processor w/ **least assigned task time**
- To Reduce overall “Tardiness” typically we use a Slack Rule: $\text{Slack} = (\text{Date}_{\text{due}} - \text{Pr.}_\text{Time})$
- To reduce Mean Tardiness, make up several schedules using SPT, LPT, EDD, Slack
 - Apply Wilkerson-Irwin algorithm separately to each machine.
 - Select the sequence with the minimum mean tardy time



Looking at the Serial Machine Problem



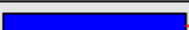

- In the 2-machine problem with 'n' different products, we have potentially: $(n!)^2$ different schedules – therefore, full enumeration is (usually) impossible
- Rules used in making decisions:
 - Minimizing Makespan (time from start of 1st job on M1 to completion of last job on M2)
 - Minimizing Mean Flow time
 - Minimize Mean Idle
 - Minimize measures of Tardiness

In the Serial Problem, Various Schedules are Compared with Gantt Charts:

- True if 'n' is 25 or less else the chart is unreadable!
- Lets try a simple 2 parts example $(2!)^2 = 4$ potential schedules – lets fully enumerate!

	M1	M2
Job 1	4	2
Job 2	1	4

Potential Schedule 1 of 4


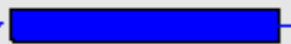


ID	Task Name	Start	Finish	Duration	Tue Apr 25											
					8	9	10	11	12	1	2	3	4	5	6	7
1	J1 M1	4/25/2006	4/25/2006	4h												
2	J2 M1	4/25/2006	4/25/2006	1h												
3	J1 M2	4/25/2006	4/25/2006	2h												
4	J1 M2	4/25/2006	4/25/2006	4h												

- Makespan = 10 units
- Mean Flow = $(4.5+8)/2 = 6.25$
- Mean Idle = $(4+5)/2 = 4.5$

Potential Schedule 2 of 4





- Makespan: 7
- Mean Flow: $(3+6)/2 = 4.5$
- Mean Idle: $(1+2)/2 = 1.5$

Potential Schedule 3 of 4

ID	Task Name	Start	Finish	Duration	Tue Apr 25													
					9	10	11	12	1	2	3	4	5	6	7	8	9	
1	J2 M1	4/25/2006	4/25/2006	.99h														
2	J1 M1	4/25/2006	4/25/2006	4h														
3	J1 M2	4/25/2006	4/25/2006	2h														
4	J2 M2	4/25/2006	4/25/2006	4h														

- Makespan = 11
- Mean Flow = $(3+9)/2 = 7.75$
- Mean Idle = $(6+5)/2 = 5.5$

Potential Schedule 4 of 4

ID	Task Name	Start	Finish	Duration	Tue Apr 25													
					9	10	11	12	1	2	3	4	5	6	7	8	9	
1	J1 M1	4/25/2006	4/25/2006	4h														
2	J2 M1	4/25/2006	4/25/2006	1h														
3	J2 M2	4/25/2006	4/25/2006	4h														
4	J1 M2	4/25/2006	4/25/2006	2h														

- Makespan: 11
- Mean Flow = $(4.5 + 10)/2 = 7.25$
- Mean Idle = $(6+5)/2 = 5.5$

Comparing:

- Upon Examination, it is obvious that Tentative Schedule 2 is the most favorable
 - Shortest Makespan, Min Mean Flow and Min Mean Idle
- Examining this schedule, we see that
 - 1. the Jobs are run directly from their 1st machine to the second upon completion
 - 2. The job with the shortest time on M1 was scheduled 1st
 - 3. The job with the shortest M2 time was scheduled last
- These decisions observations have been found (Johnson 1954) to provide an optimal schedule for the 2 machine serial problem

Johnson's Rule for n on 2 serial:

- Step 1: List all jobs with their M1 and M2 process times
- Step 2: Select the shortest processing time on the list
 - If a M1 time, schedule job 1st
 - If M2 time, schedule job LAST
 - Cross this job off list
- Repeat Step 2 through rest of job (however, 1st means after already scheduled “1sts” and last is before already scheduled lasts)
- Build optimal Schedule (Gantt Chart?) and compute Makespan, Mean Flow and Mean Idle

Lets Try an Example:

	M1	M2
J1	5	2
J2	1	6
J3	9	7
J4	3	8
J5	10	4

By Step 2: Select J2 and Schedule 1st (M1 time)
X row 2 out!

Continuing: (sch. is: 2 ----)

	M1	M2
J1	5	2
J2	1	6
J3	9	7
J4	3	8
J5	10	4

Select J1 with M2 short time – schedule 5th
X out row 1

Continuing: (sch. is 2 --- 1)

	M1	M2
J1	5	2
J2	1	6
J3	9	7
J4	3	8
J5	10	4

Select J4 short time on M1 Schedule it 2nd (1st!)
X out Row 4

Continuing: (sch. is 2-4 -- 1)

	M1	M2
J1	5	2
J2	1	6
J3	9	7
J4	3	8
J5	10	4

Select J5 short time is M2 schedule 2nd last

Now, since only J3 is left, schedule it 3rd last (M2 short time)

Summarizing:

ID	Task Name

Computing Measures:

- Makespan: is 30 working hours of time (6 am 4/25 finishing at 7 pm on 4/26)
- Mean Idle = $(2+3)/2 = 2.5$ units
- Mean Flow Time:

$$\begin{aligned} \text{MeanFlow} &= \frac{\left(\frac{1+4+13+23+28}{5} \right) + \left(\frac{7+15+22+27+30}{5} \right)}{2} \\ &= \frac{13.8 + 20.2}{2} = 17 \end{aligned}$$

Expanding The Technique to M machines

- We will explore an algorithmic method to control Makespan based on work by: Campbell, Dudek & Smith in *The Journal of Management Science*
- This method creates $M - 1$ possible schedules (M is # of serial processors)
 - This is of a possible group of $(n!)^M$ possible schedules
 - For 10 parts and 5 machines it is: 6.29×10^{32} !
- We essentially collapse the M machines into a series of “2” machine problems and apply Johnson’s Rule
 - So for 5 machine we evaluate 4 possible schedules

Stating the Algorithm:

- Define:
 - $t_{i,1}^*$ as the time to complete tasks on the “first machine” of the of the pseudo-Johnson Pair
 - $t_{i,2}^*$ as the time to complete tasks on the “second machine” of the of the pseudo-Johnson Pair
- Schedule 1 is then:
 - $t_{i,1}^* = t_{i,1}$ where we only first machine times are used for Johnson’s rule studies
 - $t_{i,2}^* = t_{i,2}$ where only the M machine times are used for Johnson’s rule studies

Stating the Algorithm: (cont.)

- Schedule 2 is then:
 - $t_{i,1}^* = t_{i,1}$ where the sum of the first two machine times are used for Johnson's rule studies
 - $t_{i,2}^* = t_{i,2}$ where the sum of the M^{th} and $M^{\text{th}}-1$ machine times are used for Johnson's rule studies
- Schedule K is:
 - $t_{i,1}^* = t_{i,1}$ where the sum of the first K machine times are used for Johnson's rule studies
 - $t_{i,2}^* = t_{i,2}$ where the sum of M^{th} to $M^{\text{th}}-K$ machine times are used for Johnson's rule studies
- And this continues thru the $M-1$ possible schedules

Trying One: (4 parts on 4 machines)

	M1	M2	M3	M4
J1	5	8	4	3
J2	7	9	5	8
J3	2	3	9	7
J4	6	1	6	4

We can expect 331776 possible schedules but the Campbell algorithm reduces it to $4! = 24$ that will have min(Makespan)!

Lets build the 1st of the 3 Possible Schedules:

	$t_{i,1}^*$	$t_{i,2}^*$
J1	5	3
J2	7	8
J3	2	7
J4	6	4

By Johnson's Rule the Sequence is: (J3-J2-J4-J1)
with Makespan of 38 units

Lets build the 2nd of the 3 Possible Schedules:

	$t_{i,1}^*$	$t_{i,2}^*$
J1	$5+8 = 13$	$4+3 = 7$
J2	$7+9 = 16$	$5+8 = 13$
J3	$2+3 = 5$	$9+7 = 16$
J4	$6+1 = 7$	$6+4 = 10$

Sequence is: (J3-J4-J2-J1) and Makespan is 40 →
so stay with 1st

Lets build the 3rd of the 3 Possible Schedules:

	$t_{i,1}^*$	$t_{i,2}^*$
J1	$5+8+4 = 17$	$8+4+3 = 15$
J2	$7+9+5 = 21$	$9+5+8 = 22$
J3	$2+3+9 = 14$	$3+9+7 = 19$
J4	$6+1+6 = 13$	$1+6+4 = 11$

Sequence is: J3-J2-J1-J4 and has a Makespan of 40
so keep 1st schedule

Gantt Charting the Solution:

ID

Task Name

1

J3 M1

Computing Mean Numbers:

- Mean Flow:

$$\begin{aligned} M.Flow &= \left(\frac{2+9+15+20}{4} \right) + \left(\frac{5+18+19+28}{4} \right) + \left(\frac{14+23+29+31}{4} \right) + \left(\frac{21+31+35+38}{4} \right) \\ &= \frac{(11.5+17.5+24.25+31.25)}{4} = 21.125 \end{aligned}$$

- Mean Idle:

$$M.Idle = \frac{18+17+16+16}{4} = 16.75$$

Considering Cells rather Than Job Shops!

- Make 1 Pass 1 rules apply
- In the last example, we would see a significant amount of idle time reduction
- In a Jobshop (if times represent batch production times), we must wait for the entire batch to be completed before they move to the next machine BUT, in Cellular (flow shops) once the first part in a batch is done, it can move to the next machine
- This would significantly reduce 'Up Front' Idle and erase much of the 'within' idle times too!

Thus ends the Lecture series:

- Please use the ideas presented during our lecture series wisely!
- Remember, in Management, none of the decisions we make are the **ONLY ANSWER!**
- When you make decisions, be prepared to back them up with data!
- Finally, as a manager, since you **must** make the decisions and the data available is sketchy → your decision will likely include a lot of “Gut Feel”
- So, Always Be **Bold** (but don’t forget to bring your towel!)

