NOVATEUR PUBLICATIONS
INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY [IJIERT]
ISSN: 2394-3696
VOLUME 7, ISSUE 5, May-2020

JOB SHOP AND FLEXIBLE JOB SHOP SCHEDULING PROBLEMS: SCHEDULING OPERATIONS

AJAY KUMAR AGARWAL
[1]Ph.D. Research Scholar, Mechanical Engineering Department,
I.K.G.P.T.U, Kapurthala, Jalandhar (Punjab), India
*ajaymechengineer@gmail.com

DR. RAKESH KUMAR
[2]Ph.D. Research Guide and Associate Professor,
Mechanical Engineering Department, S.B.S.S.T.C, Ferozepur (Punjab), India
*rakesh1607@gmail.com

ABSTRACT
This research paper explains the record of scheduling operations in mechanized services above the last 100 years. Understanding the traditions that production sequencing cum scheduling has been done is critical to analyze the existing production sequencing cum scheduling schemes and finding the traditions to get better them. The paper wraps not only the tools used to sustain decision-making in really existed world production arrangement although also made the changes in the production sequencing cum scheduling systems. The objective of the paper is to assist the production schedulers, production engineers and production researchers to recognize the true nature of production sequencing cum scheduling in active dynamic manufacturing systems and to persuade them to consider how production sequencing cum scheduling systems can be enhanced even supplementary. This section not only reviews the array of concepts and approaches used to get better production sequencing and scheduling although also demonstrate their timeless significance.

KEYWORDS: Scheduling operations, Production scheduling, Decision-making, Dynamic manufacturing systems, Computer-based scheduling

SCHEDULING TRIBULATIONS IN INDUSTRIAL OPERATIONS

The scheduling tribulations vary in industrial operations during production of a product as per demand, objective and utilization need of customers. The broadly scheduling tribulations in industrial operations can be classified as

- Job Shop Scheduling.
- Personnel Scheduling
- Facilities Scheduling
- Vehicle Scheduling and Routing
- Project Management
- Dynamic versus Static Scheduling

THE HIERARCHY OF PRODUCTION DECISIONS

The logical sequence of operations in factory planning corresponds to the following sequence

- All planning initiates with the primary demand forecast.
- Primary demand forecasts are the root for the top stage (aggregate) planning.
- The Master Production Schedule (MPS) is the outcome of disaggregating aggregate plans from down to the individual item stage.
- Based on the MPS, MRP is used to find out the size and timing of component and subassembly production of all products.
- Meticulous shop floor schedules are requisite to meet the production plans ensuing from the MRP.

Figure 1: Chain of Command of Production Decisions

SCHEDULING FUNCTIONS

- **Role of Scheduling**
  - Allocation of (scarce) resources over **time**
  - Decision-Making process with a goal of optimizing one/more objectives.

- **Elements**
  - Resources
    - Variety of Operational Machines in a Mechanical Workshop
    - Airstrips at an Airport
    - Squads at a Construction site
    - Central Processing Units in a Computer environment
  - Tasks
    - Production operations
    - Take-offs and landings
    - Construction stages
    - Computer programs
  - Objectives
    - Minimizing the completion time of the last performed task
    - Minimizing the number of tasks completed just after their respective due dates.

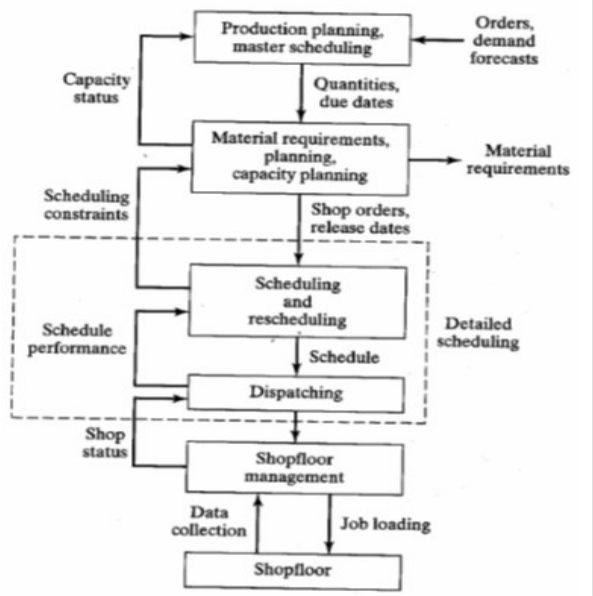


Figure 2: Information Flow Diagram of a Scheduling Function in an Enterprise

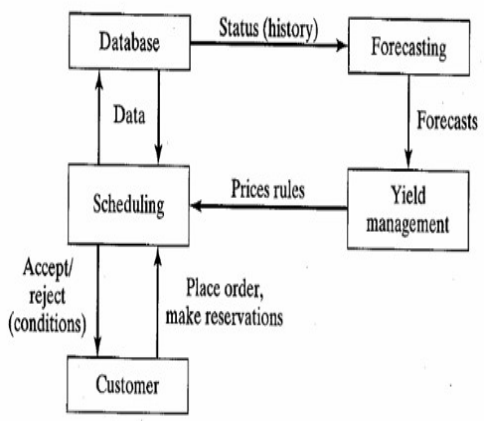


Figure 3: Information Flow Diagram of a Scheduling Function in a Service Enterprise

SCHEDULING FUNCTION IN AN ENTERPRISE

Scheduling consists of

- **MRP (Materials Requirement/Resource Planning):** The MRP tells us the capacities of products to produce in each instance bucket. However, MRP does not make any hypothesis regarding the resources (i.e. labour, raw materials, machines etc.) presently accessible in the factory. E.g. two different components have to be manufactured in the similar division. How headed for scheduling them?
- **ERP (Enterprise Resource Planning)**

These components compromise the combination of below terms from initial to final stage of production time with stipulated dispatch time for a customer:

- Processing time (p_{ij})
- Release date (r_j)
- Due date (d_j)
- Weight (w_j)
- Notations
  - $\alpha \mid \beta \mid \gamma$
  - α Machine Environment
    - Single Machine (1 machine)
    - Identical machines in parallel (P_m)
    - Machines in parallel with different speeds (Q_m)
    - Unrelated machines in parallel (R_m)
    - Flow shop (F_m) (m machines in series)
    - Flexible flow shop (FF_c) (c stages with possible Identical machines)
    - Job shop (J_m) (recrc for recirculation in β field)
    - Flexible Job shop (FJ_c)
    - Open shop (O_m) (scheduler can determine route)
  - β Processing characteristics and constraints
    - Release dates (r_j)
    - Sequence dependent setup times (s_{jk})

- Machine specific sequence dependent setup times (s_{ijk})
- Premeptions (prmp)
- Precedence constraints (prec)
- Breakdowns (brkdwn)
- Machine eligibility restrictions (M_j)
- Premutation (prmu)
- Blocking (block)
- No-wait (nwt)
- Recirculation (recrc)

➢ γ Objectives
- Makespan (C_{max})
- Max lateness (L_{max}); $L_j = C_j - d_j$
- Total weighted completion time ($\Sigma w_j C_j$)
- Discounted total weighted completion time [$\Sigma w_j(1-e^{-rC_j})$]
- Total weighted tardiness ($\Sigma w_j T_j$)
- Weighted numeral of tardy jobs ($\Sigma w_j U_j$)

➢ Examples

➢ $F_m \mid p_{ij} = p_j \mid \Sigma w_j C_j$

CLASSES OF SCHEDULES

➢ **Non-delay Schedule:** A feasible schedule is known as non-delay if no machine is kept idle while an operation is waiting for processing (i.e. it prohibits the unforced idleness).

➢ **A scheduling anomaly:** Consider a $P_2 \mid prec \mid C_{max}$ with the following processing times:

Table 1: Processing Operational Times of a Schedule of Operations in a Production Shop

j	1	2	3	4	5	6	7	8	9	10
$\mathbf{p_j}$	8	7	7	2	3	2	2	8	8	15

Characteristics of the Job Shop Scheduling Problem

- Job Arrival Pattern
- Numeral and Variety of Machines
- Numeral and skill level of workers
- Flow Patterns
- Assessment of Alternative Regulations

Objectives in Job Shop Scheduling

- Meet due dates
- Diminish work-in-process (WIP) inventory
- Diminish average flow time
- Capitalize machine/worker utilization
- Diminish set-up times for changeovers
- Diminish direct production and labor costs

(Note: These objectives can be inconsistent which may generally vary as per real industrialization process)

Terminology

- **Flow shop:** n jobs processed through m machines in the identical succession and progression.
- **Job shop:** the sequencing of jobs through machines may be diverse, and there may be numerous operations on several machines.
- **Parallel processing vs. sequential processing:** parallel processing means to the operational machines are indistinguishable.

- **Flow time of job i:** Time elapsed from commencement of first job until the finishing point of job i.
- **Makespan:** Flow process time of the job completed last.
- **Tardiness:** The positive difference among the completion time and the due date.
- **Lateness:** The difference (may be negative) among the completion time and due date.

Common Sequencing Rules

- **FCFS (First Come First Served).** Jobs processed in the array of they approach to the production shop.
- **SPT (Shortest Processing Time).** Jobs which amid the shortest operational processing time period especially they are scheduled very first.
- **EDD (Earliest Due Date).** Jobs are sequenced as per their pre-planned due dates.
- **CR (Critical Ratio).** Determine the ratio of operational processing time of the job as well as the remaining proceeding time until their due date. Schedule the job amid the leading Critical Ratio worth subsequently.

Optimal Sequencing & Scheduling Resolutions intended for a Single Operational Machine

- The rule that diminishes the mean flow time of all proceeding jobs is SPT **[1]**.
- The following criteria are equivalent:
  - Mean flow time
  - Mean waiting time.
  - Mean lateness
- **Lawler's algorithm [2]** diminishes the maximum flow time subject to precedence constraints.
- **Moore's algorithm [3]** diminishes the numeral of tardy jobs.

Optimal Sequencing & Scheduling Resolutions intended for Multiple Operational Machines

- The optimal resolution for scheduling n jobs on two machines is forever a permutation schedule (that is, jobs are done in the identical order on both machines). (This is the base for a Johnson's algorithm.)
- For three machines, a permutation schedule is still optimal if we restrict attention to total flow time only. Under rare circumstances, the two machine algorithm can be used to solve the three machine case.
- When scheduling two jobs on m machines, the problem can be resolved by graphical means.

Stochastic Scheduling: Static Case

- **Single machine case**. Suppose that processing times are casual variables. If the objective is to diminish the average weighted flow time, jobs are sequenced according to anticipated weighted Shortest Processing Time. That is, if job times are $t_1, t_2, \ldots$, and the particular weights are $u_1, u_2, \ldots$ then job i leads job i+1 if

$$E(t_i) / u_i < E(t_{i+1}) / u_{i+1}.$$

- **Multiple Machines.** Entails the hypothesis that the allocation of job times is exponential, (memoryless possessions). Presume parallel operational processing of n jobs on two machines. Then the optimal sequence is headed for schedule the jobs as per primary operational preference related to their LEPT (Longest Expected Processing Time).
- **Johnson's algorithm** for scheduling n jobs on two machines in the deterministic case has a natural extension to the stochastic case as long as the job operational times are exponentially distributed.

Stochastic Scheduling: Dynamic Analysis

- When jobs arrive to the job shop with dynamism over time, queuing theory provides a means of analyzing the results. The standard M/M/1 queue applies to the case of purely random arrivals to a single machine for random operational processing times.
- If the selection discipline does not depend on the flow times, the mean flow times are the same, but the variance of the flow times will differ.
- If job times are realized while the job connects the queue rather than when the job goes into its desired service, Shortest Processing Time generally results in lowest expected flow time.

Single Operational Machine Deterministic Scheduling Models

Jobs: $J_1, J_2, \ldots, J_n$

Suppositions:

- The machine is forever accessible right through the scheduling period.
- The machine cannot process more than one job at a time.
- Each job must expend on the machine a prearranged length of time.

$$S(t)=\begin{cases} k & \text{if job } J_k \text{ is processed at time } t \\ 0 & \text{if no job is processed at time } t \end{cases}$$

Figure 4: Single Operational Machine Deterministic Scheduling Models

Necessities that may obstruct the feasibility of an operational schedules:

- precedence constraints
- no preemptions
- release dates
- deadlines

Whether few numerous feasible operational schedules exist? NP hard

Objective function f is used comparable evaluation of different operational schedules.
f(S) < f(S') whenever operational schedule S is considered to be superior than S' problem of diminishing f(S) beyond the set of feasible operational schedules.

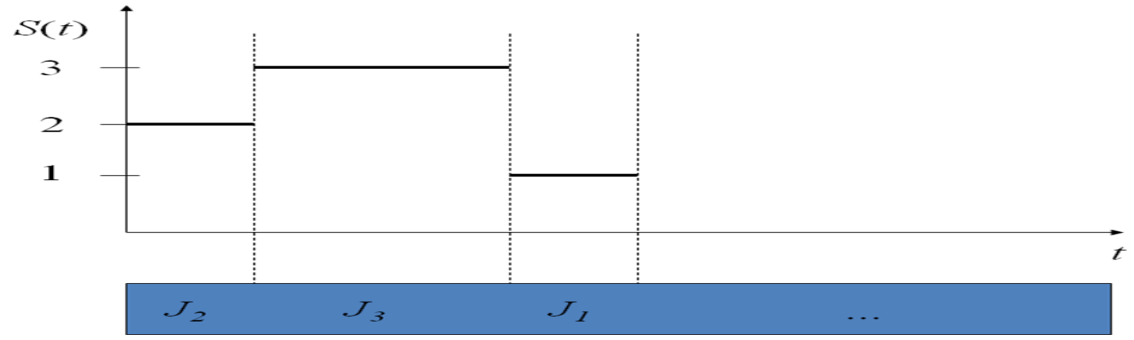
Scheduling Models Related to an Objective

A. Completion Time Models

Contents

1. An algorithm which gives an optimal schedule with the minimum total weighted completion time, $1 \| \sum w_j C_j$

   Theorem. The weighted shortest processing time first rule (WSPT) is optimal for $1 \| \sum w_j C_j$



WSPT: jobs are prearranged in declining order of w_j / p_j

The next follows inconsequentially:

The problem $1 \| \sum C_j$ is resolved by a sequence S with jobs arranged in non declining array of operational processing times.

Proof.

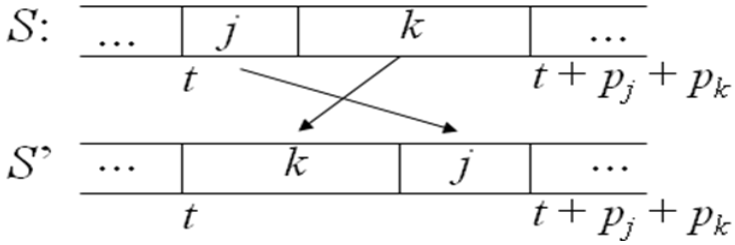
By the mode of a contradiction.

S is an operational schedule, irrespective of WSPT, which is optimal.

j and k are two adjacent jobs such that

$$\frac{w_j}{p_j} < \frac{w_k}{p_k}$$

which implies that $w_j p_k < w_k p_j$



S: $(t+p_j)\, w_j + (t+p_j+p_k)\, w_k = t\, w_j + p_j\, w_j + t\, w_k + \mathbf{p_j w_k} + p_k\, w_k$
S': $(t+p_k)\, w_k + (t+p_k+p_j)\, w_j = t\, w_k + p_k\, w_k + t\, w_j + \mathbf{p_k w_j} + p_j\, w_j$

the completion time for S' < completion time for S **contradiction!**

2. An algorithm which gives an optimal operational schedule with the least overall weighted completion time when the jobs are subject to precedence relationship that take the form of chains, 1 | chain | $\sum w_j C_j$

chain 1: $1 \rightarrow 2 \rightarrow \ldots \rightarrow k$
chain 2: $k+1 \rightarrow k+2 \rightarrow \ldots \rightarrow n$

Lemma. If

$$\frac{\sum_{j=1}^{k} w_j}{\sum_{j=1}^{k} p_j} > \frac{\sum_{j=k+1}^{n} w_j}{\sum_{j=k+1}^{n} p_j}$$

the chain of jobs 1,...,k precedes the chain of jobs k+1,...,n.

Let l* satisfy

$$\frac{\sum_{j=1}^{l^*} w_j}{\sum_{j=1}^{l^*} p_j} = \max_{1 \le l \le k} \left(\frac{\sum_{j=1}^{l} w_j}{\sum_{j=1}^{l} p_j} \right)$$

ρ factor of chain 1,...,k
l* is the job that determines the ρ factor of the chain

Lemma. If job l* determines ρ (1,...,k) , then there exists an optimal operational sequence that processes the jobs 1,...,l* one subsequent to another without disruption by jobs of commencing supplementary chains.

Algorithm

Whenever the machine is free of operational charge, select the one with the highest ρ factor amongst the remaining operational chains. Process this selected chain upto together with the job l* that concludes its ρ factor.

Example.

chain 1: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$

chain 2: $5 \rightarrow 6 \rightarrow 7$

Table 2: Processing Operational Times of a Schedule of Operations in a Production Shop

jobs	**1**	**2**	**3**	**4**		**5**	**6**	**7**
$\mathbf{w_j}$	6	18	12	8		8	17	18
$\mathbf{p_j}$	3	6	6	5		4	8	10

ρ factor of chain 1 is determined by job 2: (6+18)/(3+6)=2.67
ρ factor of chain 2 is determined by job 6: (8+17)/(4+8)=2.08
chain 1 is selected: jobs 1, 2

ρ factor of the remaining part of chain 1 is determined by job 3: 12/6=2
ρ factor of chain 2 is determined by job 6: 2.08
chain 2 is selected: jobs 5, 6

ρ factor of the remaining part of chain 1 is determined by job 3: 2
ρ factor of the remaining part of chain 2 is determined by job 7: 18/10=1.8
chain 1 is selected: job 3

ρ factor of the remaining part of chain 1 is determined by job 4: 8/5=1.6
ρ factor of the remaining part of chain 2 is determined by job 7: 1.8
chain 2 is selected: job 7

job 4 is scheduled last

the final schedule: **1, 2, 5, 6, 3, 7, 4**

$1 \mid prec \mid \sum w_j C_j$

* Polynomial time algorithms for the more complex precedence constraints than the simple chains are developed.
* The scheduling problems with arbitrary precedence relation are NP hard.
* $1 \mid r_j, prmp \mid \sum w_j C_j$ preemptive version of the WSPT rule does not always lead to an optimal solution, the problem is NP hard
* $1 \mid r_j, prmp \mid \sum C_j$ preemptive version of the SPT rule is optimal
* $1 \mid r_j \mid \sum C_j$ is NP hard

Summary

* $1 \mid\mid \sum w_j C_j$ WSPT rule
* $1 \mid chain \mid \sum w_j C_j$ a polynomial time algorithm is given
* $1 \mid prec \mid \sum w_j C_j$ with arbitrary precedence relation is NP hard
* $1 \mid r_j, prmp \mid \sum w_j C_j$ the problem is NP hard
* $1 \mid r_j, prmp \mid \sum C_j$ preemptive version of the SPT rule is optimal
* $1 \mid r_j \mid \sum C_j$ is NP hard

Due date related objectives:

B. Lateness Models

Contents

1. Lawler's algorithm provides an optimal operational schedule with the least cost, h_{max} when the jobs are subject to precedence relationship, 1 | prec | h_{max}

Lawler's Algorithm

- Backward algorithm provides an optimal operational schedule for 1 | prec | h_{max}

$$h_{max} = \max (h_1(C_1), \ldots, h_n(C_n))$$

h_j are non declining cost functions

Notation

Makespan, $C_{max} = \sum p_j$ completion of operational process of the last job

J set of jobs already scheduled as they have to be processed during the pre-allotted time interval

$$\left[C_{\max} - \sum_{j \in J} p_j, C_{\max} \right]$$

J^C complement of job set J, set of jobs still to be process scheduled

$J' \subseteq J^C$ set of jobs that can be operational scheduled instantly previous to set J (schedulable jobs)

Lawler's Algorithm for 1 | | h_{max}

Step 1.

$J = \varnothing$

$J^C = \{1,\ldots,n\}$

$k = n$

Step 2.

Let j* be such that

$$h_{j^*}\left(\sum_{j \in J^C} p_j \right) = \min_{j \in J^C} h_j \left(\sum_{j \in J^C} p_j \right)$$

Place j* in J in the k-th order position

Delete j* from J^C

Step 3.

If $J^C = \varnothing$ then Stop

else $k = k - 1$

go to Step 2

Example

Table 3: Processing Operational Times of a Schedule of Operations in a Production Shop (no precedence relationships between jobs)

jobs	**1**	**2**	**3**
$\mathbf{p_j}$	2	3	5
$\mathbf{h_j (C_j)}$	$1 + C_1$	$1.2C_2$	10

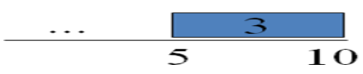
$J = \varnothing$ $J^C = \{1, 2, 3\}$ jobs at a standstill to be scheduled

$C_{max} = 10$

$h_1(10) = 11$

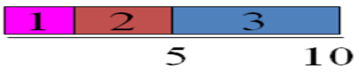
$h_2(10) = 12$

$h_3(10) = \mathbf{10}$



Job 3 is scheduled last and has to be processed in [5, 10].

J = {3} J^C = {1, 2} jobs at a standstill to be scheduled
C_{max} = 5
$h_1(5)$ = **6**
$h_2(5)$ = **6**
Either job 1 or job 2 may be processed before job 3.



Two schedules are optimal: **1, 2, 3** and **2, 1, 3**

Lawler's Algorithm for 1 | prec | h_{max}
Step 1.
J = ∅, J^C = {1,...,n}
J' the set of all jobs with no successors
k = n
Step 2.
Let j* be such that

$$h_{j^*}\left(\sum_{j\in J^C} p_j\right) = \min_{j\in J'} h_j\left(\sum_{j\in J^C} p_j\right)$$

Place j* in J in the k-th order position
Delete j* from J^C
Modify J' to represent the set of jobs which can be scheduled instantaneously before set J.

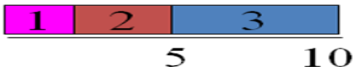
Step 3.
If J^C = ∅ then Stop
else k = k - 1
go to Step 2

Example
What will happen in the previous example if the precedence 1 → 2 has to be taken into account?
J = ∅
J^C={1, 2, 3} at a standstill to be scheduled
J'={2, 3} have no successors
C_{max} = 10
$h_2(10)$ = 12
$h_3(10)$ = **10**

J = {3}
J^C={1, 2} at a standstill to be scheduled
J'={2} can be scheduled instantaneously before J
C_{max} = 5
$h_2(5)$ = **6**

J = {3, 2} J^C={1} J'={1}
$h_1(2)$ = **3**

Optimal schedule: 1, 2, 3,
h_{max} = 10



Note:
1 || L_{max} is the special case of the 1 | prec | h_{max}

where $h_j = C_j - d_j$

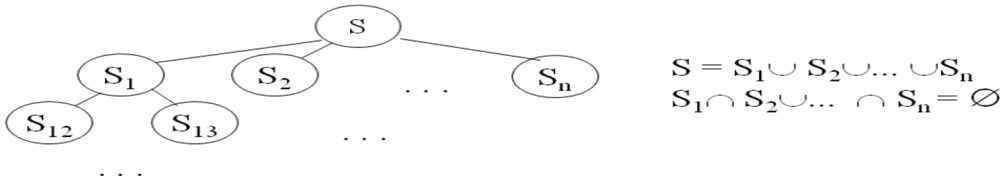
algorithm results in the schedule that orders jobs in increasing order of their due dates - earliest due date first rule (EDD)

$1 \mid r_j \mid L_{max}$ is NP hard, branch-and-bound is used

$1 \mid r_j, prec \mid L_{max}$ similar branch-and-bound

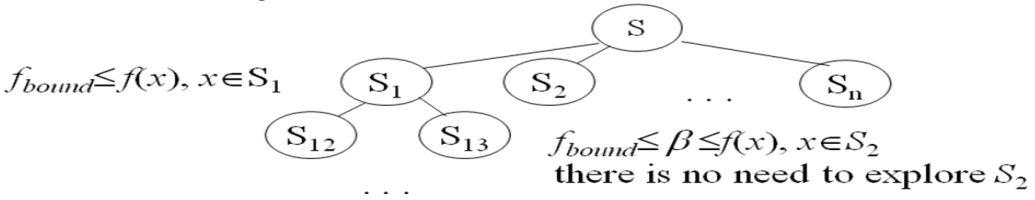
C. Branch-and-bound algorithm for the scheduling problems with the objective to minimise lateness, $1 \mid r_j \mid L_{max}$

- Search space can grow very large as the number of variables in the problem increases!
- Branch-and-bound is a heuristic that works on the idea of successive partitioning of the search space.



- We require several means for attaining a lower bound on the charge for any meticulous solution (the task is to diminish the operational cost of production of a product).

Branch-and-bound algorithm



Step 1.

Initialise $P = \cup S_i$ (conclude the partitions)

Initialise f_{bound}

Step 2.

Remove best partition S_i from P

Reduce or subdivide S_i into S_{ij}

Update f_{bound}

$P = P \cup S_{ij}$

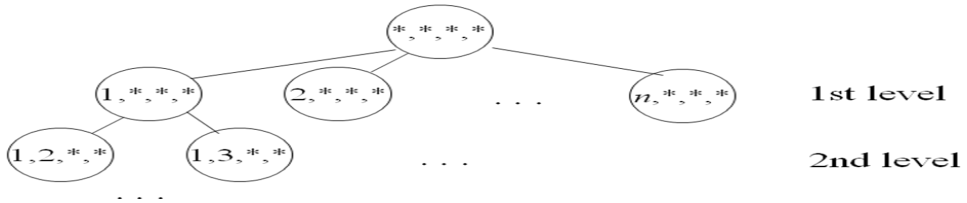
For all $S_{ij} \in P$ do

if lower bound of $f(S_{ij}) > f_{bound}$ then remove S_{ij} from P

Step 3.

If here not termination conditions exist at that time go to Step 2

- Branch-and-bound algorithm for $1 \mid r_j \mid L_{max}$
  - Solution space contains n! schedules (n is number of jobs). Total enumeration is not viable!



Branching rule:
k-1 level, $j_1, \ldots, j_{k-1}$ are scheduled,
j_k need to be considered if no job at a standstill to be scheduled cannot be processed before the release time of j_k and that is:

$$r_{j_k} < \min_{l \in J} (\max(t, r_l) + p_l)$$

J set of jobs not yet scheduled
t is time when j_{k-1} is completed

Lower bound:

- Preemptive earliest due date (EDD) rule is optimal for $1 \mid r_j \text{ prmp} \mid L_{max}$
  A preemptive schedule will have an utmost lateness not superior than a non-preemptive schedule.

- If a preemptive EDD rule provides a non preemptive schedule then all nodes with a superior lower bound are able to be disregarded.

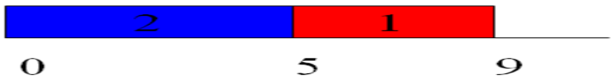
Example

Table 4: Processing Operational Times of a Schedule of Operations in a Production Shop

jobs	**1**	**2**
$\mathbf{p_j}$	4	5
$\mathbf{r_j}$	3	0
$\mathbf{d_j}$	4	6

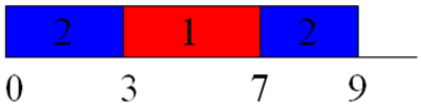
- Non-preemptive schedules

$L_1=3$
$L_2=6$
$L_{max}=6$



$L_1=5$
$L_2=-1$
$L_{max}=5$

- Preemptive schedule obtained using EDD



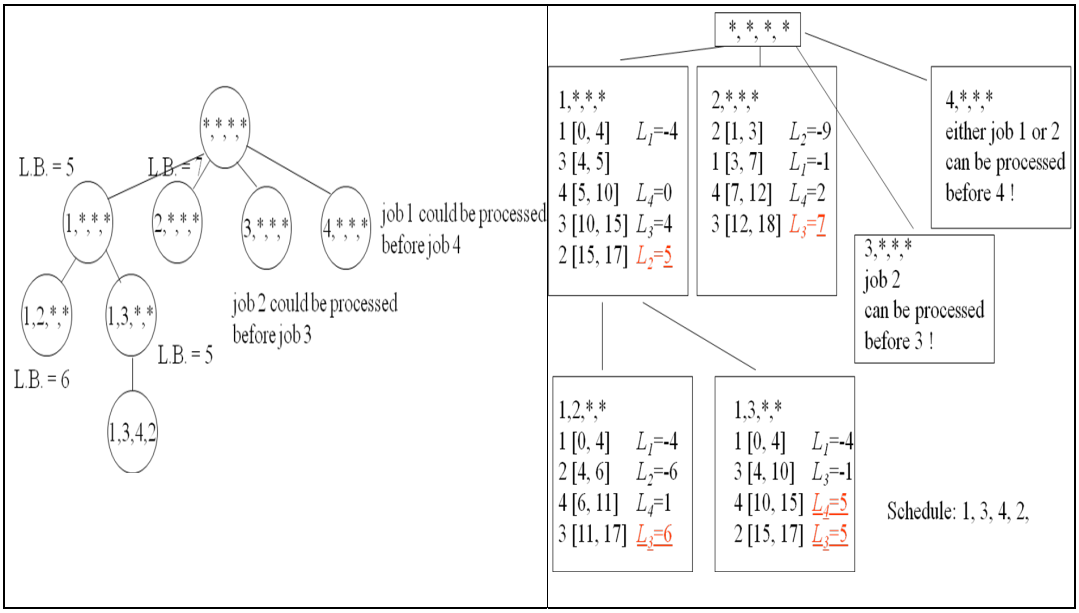
$L_1=3$
$L_2=3$
$L_{max}=3$

✉ the lowest L_{max} !

EExample

Table 5: Processing Operational Times of a Schedule of Operations in a Production Shop

jobs	**1**	**2**	**3**	**4**
$\mathbf{p_j}$	4	2	6	5
$\mathbf{r_j}$	0	1	3	5
$\mathbf{d_j}$	8	12	11	10



Summary

*	1 | prec | h_{max} , h_{max}=max($h_1(C_1)$, ... ,$h_n(C_n)$),	Lawler's algorithm
*	1 || L_{max}	EDD rule
*	1 | r_j | L_{max}	is NP hard , branch-and-bound is used
*	1 | r_j , prec | L_{max}	similar branch-and-bound
*	1 | r_j, prmp | L_{max}	preemptive EDD rule

D. (i) Tardiness Models

Contents

1. Moore's algorithm which provides an optimal schedule with the least numeral of tardy jobs, 1 || $\sum U_j$
2. An algorithm which gives an optimal schedule with the least total tardiness, 1 || $\sum T_j$

Moore's algorithm for 1 || $\sum U_j$

Optimal schedule has this form $j_{d1},...,j_{dk}$, meet their due dates EDD rule
$j_{t1},...,j_{tl}$ do not meet their due dates

Notation

J set of jobs already scheduled

J^C set of jobs still to be scheduled

J^d set of jobs already considered for scheduling, but which have been discarded because they will not meet their due date in the optimal schedule

Step 1.

$J = \varnothing$

$J^d = \varnothing$

$J^C = \{1,...,n\}$

Step 2.

Let j* be such that

$$d_{j^*} = \min_{j \in J^C} \left(d_j\right)$$

Add j* to J

Delete j* from J^C

Step 3.
If

$$\sum_{j \in J} p_j \le d_{j^*}$$

then go to Step 4.
else let k* be such that

$$p_{k^*} = \max_{j \in J} (p_j)$$

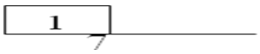
Delete k* from J
Add k* to J^d

Step 4.
If $J^d = \varnothing$ STOP
else go to Step 2.

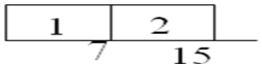
Example
Table 6: Processing Operational Times of a Schedule of Operations in a Production Shop

jobs	**1**	**2**	**3**	**4**	**5**
$\mathbf{p_j}$	7	8	4	6	6
$\mathbf{d_j}$	9	17	18	19	21

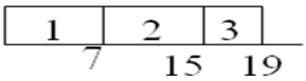
$J = \varnothing$, $J^d = \varnothing$, $J^C = \{1,...,5\}$
j*=1 $J = \{1\}$, $J^d = \varnothing$, $J^C = \{2, 3, 4, 5\}$, $t=7 < 9 = d_1$



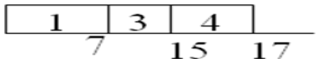
j*=2 $J = \{1, 2\}$, $J^d = \varnothing$, $J^C = \{3, 4, 5\}$, $t=15 < 17 = d_2$



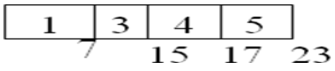
j*=3 $J = \{1, 2, 3\}$, $J^d = \varnothing$, $J^C = \{4, 5\}$, $t=19 > 18 = d_3$
k*=2 $J = \{1, 3\}$, $J^d = \{2\}$, $t=11$



j*=4 $J = \{1, 3, 4\}$, $J^d = \{2\}$, $J^C = \{5\}$, $t=17 < 19 = d_4$



j*=5 $J = \{1, 3, 4, 5\}$, $J^d = \{2\}$, $J^C = \varnothing$, $t=23 > 21 = d_5$
k*=1 $J = \{3, 4, 5\}$, $J^d = \{2, 1\}$, $t=16 < 21 = d_5$

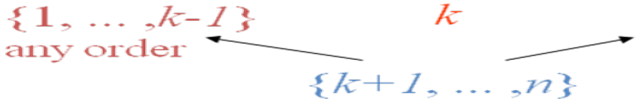


optimal schedule **3, 4, 5, 1, 2** $\sum U_j = 2$

(ii) The Total Tardiness
$1 \,||\, \sum T_j$ is NP hard

Lemma. If $p_j < p_k$ and $d_j < d_k$ then there exists an optimal sequential schedule in which job j is scheduled before job k.

$$d_1 \le ... \le d_n \text{ and } p_k = \max(p_1, ..., p_n)$$



Lemma. There exists an integer δ, $0 \le \delta \le n-k$ such that there is an optimal operational schedule S in which job k is preceded by jobs $j \le k + \delta$ and followed by jobs $j > k + \delta$.

$\{1, \ldots, k-1, k+1, \ldots, k+\delta\}$ $\{k+\delta+1, \ldots, n\}$ completion time of job k
any order any order

E. Sequence-Dependent Setup Problems

Principle of Optimality (Bellman 1956 [4])
An optimal policy has the property that whatever the initial state and the initial decision are; the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

Algorithm Dynamic programming procedure: recursively the optimal solution for some job set J starting at time t is determined from the optimal solutions to sub problems defined by job subsets of $S^* \subset S$ with start times $t^* \geq t$.

J(j, l, k) contains all the jobs in a set {j, j+1, ... , l} with processing time ≤ p_k
V(J(j, l, k) , t) total tardiness of the subset under an optimal sequence if this subset starts at time t

Initial conditions:
$V(\varnothing, t) = 0$
$V(\{ j \}, t) = \max (0, t + p_j - d_j)$

Recursive conditions:

$$V(J(j,l,k),\ t) = \min_{\delta} (V(J(j, k'+\delta, k'), t) + \max(0, C_{k'}(\delta) - d_{k'}) + V(\ J(k'+\delta+1, l, k'), C_{k'}(\delta)))$$

where k' is such that $p_{k'} = \max (p_{j'} \mid j' \in J(j, l, k))$

Optimal value function is obtained for V ({ 1,...,n }, 0)

Example

Table 7: Processing Operational Times of a Schedule of Operations in a Production Shop

jobs	**1**	**2**	**3**	**4**	**5**
$\mathbf{p_j}$	121	79	147	82	130
$\mathbf{d_j}$	260	266	266	336	337

$k' = 3,\ 0 \leq \delta \leq 2$
$d_{k'} = d_3 = 266$

$$V(\{1,2,\ldots,5\}),0) = \min \begin{cases} V(J(1,3,3),0) + 81 + V(J(4,5,3),347) \\ V(J(1,4,3),0) + 164 + V(J(5,5,3),430) \\ V(J(1,5,3),0) + 294 + V(\varnothing,560) \end{cases}$$

V(J(1, 3, 3) , 0) = 0 **1, 2** $C_2 = 200 < 266 = d_2$
$T_1 + T_2 = 0$
2, 1 $C_1 = 200 < 260 = d_1$
$T_2 + T_1 = 0$

$C_3(0) - d_3 = 121 + 79 + 147 - 266 = 347 - 266 = 81$

V(J(4, 5, 3) , 347) **4, 5** $T_4 = 430 - 336 = 94$

$T_5 = 560 - 337 = 229$
$T_4 + T_4 = 317$

5, 4 $T_5 = 477 - 337 = 140$
$T_4 = 560 - 336 = 224$
$T_5 + T_4 = 364$

$C_3(1) - d_3 = 347 + 83 - 266 = 430 - 266 = 164$
$C_3(2) - d_3 = 430 + 130 - 266 = 560 - 266 = 294$

$V(J(1, 4, 3), 0) = 0$ achieved with the sequence **1, 2, 4** and **2, 1,** 4
$V(J(5, 5, 3), 430) = 223$
$V(J(1, 5, 3), 0) = 76$ achieved with the sequence **1, 2, 4, 5** and **2, 1, 4, 5**
$V(\varnothing, 560) = 0$

$$v(\{1,2,...,5\},0) = \min\begin{cases} 0+81+317 \\ 0+164+223 \\ 76+294+0=370 \end{cases}$$

Optimal sequences: **1, 2, 4, 5, 3** and **2, 1, 4, 5, 3**

Summary
* $1 \mid\mid \sum U_j$ forward algorithm
* $1 \mid\mid \sum w_j U_j$ is NP hard
* $1 \mid\mid \sum T_j$ is NP hard, pseudo-polynomial algorithm based on dynamic programming

Assembly Line Balancing
The characteristics of the Assembly Line Balancing problem are:
- A collection of n tasks must be completed on each item
- Tasks are assigned to stations. Tasks must be sequenced properly, and certain tasks may not be completed at the same station.
- The objective is to allocate tasks to stations to diminish the cycle time, C.
- The general problem is complex to solve optimally, but effective heuristics are available (the text discusses one known as the ranked positional weight technique.)

Schematic of a Typical Assembly Line

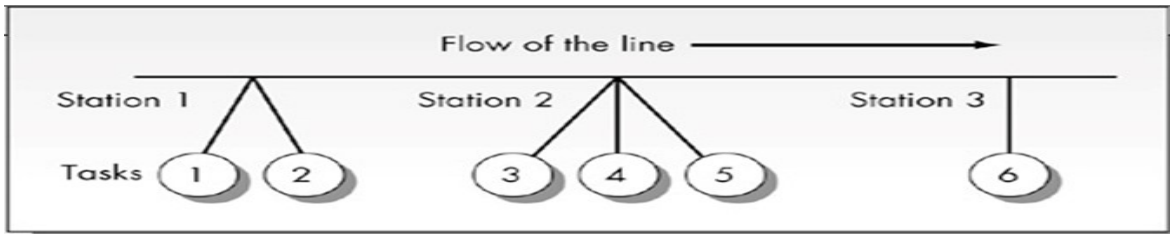


Figure 4: Schematic of a typical assembly line

CONCLUSION
Since the severance that established production scheduling as a distinctive production management function, the huge changes in production scheduling are due to mainly two key events. The first is to recognize the complex relationships between men, machines, orders, and time. The second is the irresistible powers of information technology to collect, visualize, process, and share data quickly and easily, which has enhanced all types of decision-making processes. These actions have led, in most places, to the decline of shop foremen, who used to rule factories, to software systems as well as optimization algorithms for production scheduling.

The shocking news is that many manufacturing firms have not taken benefits of these developments. They produce goods only to transport them to their customers, but the production sequencing and scheduling system is a broken down collected works of autonomous plans that are often ignored, periodic meetings wherever unreliable information is communal, expediters who jog from one calamity to another, and ad-hoc decisions made by peoples who cannot see the whole system. Production scheduling systems rely on human decision-makers and many of them need timely research guidance assistance.

This overview of scheduling operations of production methods should be valuable to those just beginning their study of production planning and control. In addition, practitioners as well as researchers should use this information to consider what has been truly useful to improve the production scheduling practices in the real-world.

ACKNOWLEDGEMENT

Inspiration and guidance are invaluable in all aspect of life, especially when it is academic. I acknowledge my gratitude to all those who has provided me timely help me in completing my research. I am highly obliged to Dr. Rakesh Kumar for directing me through his guidance for my research work and allowing me to write this as well as additional research papers for the technical knowledge of future technical researchers.

REFERENCES

Books

1) Scheduling, Theory, Algorithms, and Systems, Michael Pinedo, Prentice Hall, 1995, or new: Second Addition, 2002, Chapter 3.
2) Scheduling, Theory, Algorithms, and Systems, Michael Pinedo, Prentice Hall, 1995, Chapter 3.2 or new: Second Addition, 2002, Chapter 3.
3) Scheduling, Theory, Algorithms, and Systems, Michael Pinedo, Prentice Hall, 1995, Chapters 3.3 and 3.4 or new: Second Addition, 2002, Chapter 3.

Journal Articles

4) Bellman, R., (1956), “Mathematical aspects of Scheduling Theory”, Journal of Society of Industrial and Applied Mathematics 4, pp.168–205.