Chapter 1

The Production Paradigm

Evolution of Production Systems

- Ancient Systems
  - basic planning, organizations and control
  - specialization of labor
- Feudal Systems
  - hierarchical system (delegation)
  - land and labor as production input
- European System
  - double entry bookkeeping, cost accounting
  - Industrial Revolution: specialization, mass markets, mass production
- American System
  - interchangeable parts
  - steam power
  - assembly lines
The Competitive Environment

- Status Quo of the American (and European) System (late 80s):
  - production driven system
  - cost efficient production as the main goal
  - high quality standardized goods
  - Market is taken as given

- Change towards a market-driven system
  - more sophisticated consumers
  - short product life cycles
  - product variety increases
  - global competition and heterogeneous markets

Production Systems

- Input → Output
- manufacturing firms
- service companies: Universities

- flow process in two parts:
  - physical material
  - information

- coordination also with suppliers and distributors: supply chain management: recent emphasis on bi-directional information flow
Production Systems

Production Information System

The PPC function integrates material flow using the information system. Integration is achieved through a common database.
Building Blocks

- Objectives:
  - Quality
  - Cost
  - Time

- These might be seen as the fundamental objectives of the firm
- induced by these objectives one might observe various subordinate objectives at different levels and parts of the company
  - more variability, high inventory
  - low unit costs, high inventory
  - high throughput, less variability
  - short cycle times, high inventory

- Important to understand effects of individual incentives!

Building Blocks

- Physical Arrangement
  - production volume and product variety determine layout
    - job shop (low-volume, high customized)
      - process or functional layout
    - flow shop (high-volume)
      - product layout
Building Blocks

Organizational Arrangements

- **Functional Structure**: input oriented
- **Divisional Structure**: output oriented (projects, services, programs, locations) strategic business units
- **Matrix Structure**: one person-two bosses (input & output oriented)
Organizational Arrangements

Divisional Structure

CEO

  Engineering  Engineering  Engineering  Marketing  Marketing  Marketing  Control  Control  Control

Organizational Arrangements: Matrix

<table>
<thead>
<tr>
<th>Marketing</th>
<th>Engineering</th>
<th>Prod.</th>
<th>Purchasing</th>
<th>Finance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prod. A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prod. B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prod. C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Production Planning and Control (PPC)

- Integrated-material-flow-based information system
- based on a feedback loop (control theory)
- management of deviations
- art of selecting the appropriate mix of management technologies
- impact of organizational structure, life-cycle effects

Building Blocks

- Planning horizons

Operational Planning
- Hour
- Day
- Week
- Month
- Year

Strategic Planning
- Year
- Years

Tactical Planning
## Building Blocks

### Types of Decisions

<table>
<thead>
<tr>
<th></th>
<th>Long (strategic) top management</th>
<th>Intermediate (tactical) middle management</th>
<th>Short (operational) operational management</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time</strong></td>
<td>three to ten years</td>
<td>six months to three years</td>
<td>one week to six months</td>
</tr>
<tr>
<td><strong>Unit</strong></td>
<td>dollars; hours</td>
<td>dollars; hours; product line; product family</td>
<td>individual products; product family</td>
</tr>
<tr>
<td><strong>Inputs</strong></td>
<td>aggregate forecast; plant capacity</td>
<td>intermediate forecast; capacity and production levels taken from long range plan</td>
<td>short range forecast; work force levels, processes; inventory levels</td>
</tr>
<tr>
<td><strong>Decisions</strong></td>
<td>capacity; product; supplier needs; quality policy</td>
<td>work force levels; processes; production rates; inventory levels; contracts with suppliers; quality level; quality costs</td>
<td>allocation of jobs to machines; overtime; undertime; subcontracting; delivery dates for suppliers; product quality</td>
</tr>
</tbody>
</table>

Chapter 2

### Market Driven Systems


**Market driven systems**

**The Wheel of Competitiveness**

- **Customer Satisfaction/Expectations**
- Quality
- Cost
- Time
- Management Role
- Integration
- Role Design
- Scope

**The Wheel of competitiveness**

- **Hub: the customer**
  - individual rather than average customer
  - fast changing expectations
  - little loyalty
  - 'Internal customers': any operation is the customer of the previous operation
The Wheel of competitiveness

The Delivery Cycle:

Quality → Time → Cost

or

Quality

Time

Cost

The Wheel of Competitiveness

The Support Circle

- Scope (Supplier - Producer - Relationship)

- Integration
  - looking at the system rather than a component
  - product and process design

- Flexibility
  - volume
  - process (setups)

- Design
  - function, life, form and effective manufacture
The Wheel of Competitiveness

The Support Cycle:
- Simplicity (KISS)
- Variability
  - deterministic manufacturing
  - Factory Physics (Hopp/Spearman)
- Pull
  - physical flow
  - information flow
  - the essence of pull production is to do things upstream only when requested downstream

The Wheel of Competitiveness

The Support Cycle
- Waste/Value
  - “doing it right the first time”
  - value-adding activities
  - cost adding activities
- Improvement
  - Integrated and Continues Improvement
  - Kaizen, ...
- Management role
  - change process
  - commitment
  - participation
  - goals
The Wheel of Competitiveness

The Support Cycle
- Employee role
  - involvement
  - development

The impact circle
- Efficiency: make things right
  - local
  - ration of output to input
- Effectiveness: requirements of the total system

Implementation

Integrated Production Systems
- best applied in the medium-variety, medium-volume range
- information integration is key aspect

3 leading approaches
- Cellular Manufacturing Systems (CMS)
- Flexible Manufacturing Systems (FMS)
- Computer Integrated Manufacturing Systems (CIM)
Integrated Production Systems

**Cellular Manufacturing Systems**
- manned or unmanned cells
- produce a family of parts that have similar processes
- group technology (see Basic Course: OMA)
- organized in a u-shaped layout in which multifunctional workers perform the required operations
Integrated Production Systems

Flexible Manufacturing Systems
- integration of
  - manufacturing or assembly processes
  - automated material flow systems
  - computer communication
  - control
- computer control system does:
  - production control
  - scheduling
  - flow control
  - machine control
- reaction to real time status data
- automotive and electronics industry

Computer Integrated Manufacturing (CIM)
- broader scope than CMS
- use information technology to coordinate business functions with product development, design and manufacturing
- 'bridges' between FMS islands
Market driven systems

Integration Process
- teamwork
- concurrent engineering
  - life cycle engineering
  - product and process design are considered together
  - cross functional teams
- TQM
- World class manufacturing
- Lean production (Toyota, production floor focus)
- Agile manufacturing (enterprise view)

Chapter 3

Problem Solving
Problem Solving

- Current state $\rightarrow$ goal state
- **impact:** should be worth the resource
- ability to measure the gap
- ability to close the gap
- solve or dissolve

Problem Solving

- Problem Identification
  - Symptoms
  - Problem mission
  - mission will be translated into goals and objectives
  - problem owners: people who must live with the solution
  - Assumptions
  - Initial Problem Statement
Problem Solving

Identify the Problem
- Owners
- Problem Solver
- Need / Opportunity
- Mission
- Assumptions

Understand the Problem
- System
- Owners
- Problem Solver
- Assumptions
- Problem Characteristics
- Problem Validation

Develop the Model
- Problem Solver
- Assumptions
- Data
- Modeling Concepts
- Representations
- Boundaries
- Objective
- Constraints
- Internal Validation

Interpret the Solution
- Owners
- Problem Solver
- Interpretation
- Robustness
- Validate Solution
- Judgement

Solve the Model
- Problem Solver
- Resources
- Algorithms
- External Validation

Implementation
- Owners
- Problem Solver
- Presentation
- Acceptance & Commitment
- Training
- Parallel Operation
- Feedback

Understand the Problem
- The systems perspective
  - Analysis
  - Synthesis
- Goals
- Problem Characteristics
  - one-time - recurrent
  - level of detail
- Validate Understanding
Problem Solving

* Develop a model
  - Model representation
    - iconic
    - analog
    - symbolic
  - Data
  - Modeling concepts
    - Boundaries
    - Objectives
    - Constraints
  - Relationships
  - Assumptions and Involvement
  - Internal validation

Problem Solving

* Solve the Model
  - External validation
  - Simplification
  - Solution Strategy
    - Exact
    - Heuristic
    - Simulation

* Interpret the solution
  - robustness
  - plausibility

* Implementation
Example: MaTell – Identify

- MaTell produces telephones: desk phones, wall phones, answering machines
- All 3 products are made at a single plant
- Customers cannot buy the products because they are unavailable

Is there a problem?

What is the problem mission?

Who are the owners of this problem?

Assumptions?

Initial problem statement:
- Current state: Some customers who want our product cannot get them.
- Goal state: Deliver a product to all of our customers who want one.
- Problem: How can we provide products to all out customers?

Example: MaTell - Understand

- variety of ways to provide more products
  - build a new plant
  - expand the existing plant
  - subcontracting
  - ...

- actual production system
  - fabrication department - assembly department
  - 15000 wall phones (W), 17000 desk phones (D), 5000 answering machines (A) per week
  - plant works a three eight-hour shifts a day, seven days a week
  - fabrication: 135 hours per week
  - assembling: 163 hours per week

- new problem owner: production department

- 2 strategies:
  - using capacity more effectively
  - reducing the time a product spends in assembly
**Example: MaTell - Develop**

- **data available**: time it takes to make each product in the fabrication and assembly department
  - 1000 desk phones: 2.5 hours fabrication, 3 hours assembly
  - 1000 wall phones: 4 hours fabrication, 3 hours assembly
  - 1000 answering machines: 6 hours fabrication, 14 hours assembly
- **objective**: 
  
  \[
  W + D + A
  \]

  - total fabrication time: 
    
    \[
    4 W + 2.5 D + 6 A
    \]
  - total assembly time: 
    
    \[
    3 W + 3 D + 14 A
    \]
  - marketing department: at most 30,000 desk phones, 30,000 wall phones and 12,000 answering machines can be sold per week.
- **assumptions**:
  - Demand will continue at the same levels or higher for some time
  - The number of products made is a good measure for increasing the throughput.
  - There is a linear relationship between products and fabrication (assembly) time.
  - Data are accurate.

**Example: MaTell – Solve / Interpret**

- **Solve using Excel spreadsheet / Solver**
- **Is the new mix more or less profitable?**
- **margins**: $2.20 (D), $2.00 (W), $7.00 (A)
- **alternative objective**:
  
  \[
  2.2 D + 2 W + 7 A
  \]
- **add lower bounds**: 10 (D), 10 (W), 4 (A)
Example: MaTell – Implementation

- present the solution
  Though the spreadsheet was not used to get the solution, it would be a good way to introduce the LP solution

- acceptance relatively easy (owners were involved)

- commitment may be more difficult, but only few resources needed (LP package, training for the planner)

- check the system from time to time (conditions may change)

Problem Solving

- Work to do:

- Examples: 3.12 abcd, 3.19 ab, 3.30abc, 3.36abc, 3.41 abc, 3.46
Chapter 5

Aggregate Planning

Supply Chain Planning Matrix

<table>
<thead>
<tr>
<th>Long-term</th>
<th>Medium-term</th>
<th>Short-term</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategic Network Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Master Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Material Requirements Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Production Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distribution Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scheduling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transport Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Demand Planning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Demand Fulfilment &amp; ATP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Production Management 44
Supply Chain Planning Matrix

Aggregate Planning

Example:
- one product (plastic case)
- two injection molding machines, 550 parts/hour
- one worker, 55 parts/hour
- steady sales 80,000 cases/month
- 4 weeks/month, 5 days/week, 8h/day
- how many workers?

in real life constant demand is rare
- change demand
- produce a constant rate anyway
- vary production
Aggregate Planning

- **Influencing demand**
  - do not satisfy demand
  - shift demand from peak periods to nonpeak periods
  - produce several products with peak demand in different periods

- **Planning Production**
  - Production plan: how much and when to make each product
  - rolling planning horizon
  - long range plan
  - intermediate-range plan
    - units of measurements are aggregates
    - product family
    - plant department
    - changes in workforce, additional machines, subcontracting, overtime,...
  - Short-term plan

---

Aggregate Planning

- **Aspects of Aggregate Planning**
  - Capacity: how much a production system can make
  - Aggregate Units: products, workers,...
  - Costs
    - production costs (economic costs!)
    - inventory costs(holding and shortage)
    - capacity change costs
Aggregate Planning

- Spreadsheet Methods
- Zero Inventory Plan
  - Precision Transfer, Inc. Produces more than 300 different precision gears (the aggregation unit is a gear!).
  - Last year (=260 working days) Precision made 41.383 gears of various kinds with an average of 40 workers.
  - 41.383 gears per year
  - 40 x 260 worker-days/year = 3,98 -> 4 gears/worker-day

Aggregate demand forecast for precision gear:

<table>
<thead>
<tr>
<th>Month</th>
<th>January</th>
<th>February</th>
<th>March</th>
<th>April</th>
<th>May</th>
<th>June</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demand</td>
<td>2760</td>
<td>3320</td>
<td>3970</td>
<td>3540</td>
<td>3180</td>
<td>2900</td>
<td>19,670</td>
</tr>
</tbody>
</table>

- Holding costs: $5 per gear per month
- Backlog costs: $15 per gear per month
- Hiring costs: $450 per worker
- Lay-off costs: $600 per worker
- Wages: $15 per hour (all workers are paid for 8 hours per day)
- There are currently 35 workers at Precision
- Currently no inventory

Production plan?
Aggregate Planning

- Zero Inventory Plan
  - produce exactly amount needed per period
  - adapt workforce

![Graph showing change in workforce over months]

Production Management 51
Aggregate Planning

- **Level Work Force Plan**
  - backorders allowed
  - constant numbers of workers
  - demand over the planning horizon
  - gears a worker can produce over the horizon

- $\frac{19670}{(4 \times 129)} \approx 38.12 \Rightarrow 39$ workers are always needed

Aggregate Planning

- Inventory: January: $3276 - 2760 = 516$
- February: $516 + 3120 - 3320$
- March: $316 + 3588 - 3670 = -66!$ - Backorders: $66 \times $15 = $990

![Graph showing inventory and backorders over months]

Production Management 53
Aggregate Planning

- **no backorders are allowed**
  - workers = cumulative demand/(cumulative days x units/workers/day)
  - January: 2760/(21 x 4) = 32.86 -> 33 workers
  - February: (2760+3320)/[(21+20) x 4] = 37.07 -> 38 workers.
  - March: 10.050/(64 x 4) = 40 workers
  - April: 13.590/(85 x 4) => 40 workers
  - May: 16.770/(107 x 4) => 40 workers
  - June: 19670/(129 x 4) => 39 workers

Example Mixed Plan

- The number of workers used is an educated guess based on the zero inventory and level work force plans!
Spreadsheet Methods Summary

<table>
<thead>
<tr>
<th></th>
<th>Zero-Inv.</th>
<th>Level/ BO</th>
<th>Level/ No BO</th>
<th>Mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hiring cost</td>
<td>4950</td>
<td>1800</td>
<td>2250</td>
<td>3150</td>
</tr>
<tr>
<td>Lay-off cost</td>
<td>7800</td>
<td>0</td>
<td>0</td>
<td>4200</td>
</tr>
<tr>
<td>Labor cost</td>
<td>59856</td>
<td>603720</td>
<td>619200</td>
<td>593520</td>
</tr>
<tr>
<td>Holding cost</td>
<td>0</td>
<td>4160</td>
<td>6350</td>
<td>3890</td>
</tr>
<tr>
<td>BO cost</td>
<td>0</td>
<td>7110</td>
<td>0</td>
<td>990</td>
</tr>
<tr>
<td>Total cost</td>
<td>611310</td>
<td>616790</td>
<td>627800</td>
<td>605180</td>
</tr>
<tr>
<td>Workers</td>
<td>33</td>
<td>39</td>
<td>40</td>
<td>35</td>
</tr>
</tbody>
</table>

Aggregate Planning

* Linear Programming Approaches to Aggregate Planning

Parameters:
T... Planning horizon length

t ... Index of periods, t=1,2,..., T

\( D_t \) ... forecasted number of units demanded in period \( t \)

\( n_t \) ... number of units that can be made by one worker in period \( t \)

\( C^p_t \) ... cost to produce one unit in period \( t \)

\( C^w_t \) ... cost of one worker in period \( t \)
Aggregate Planning

$C^H_t$ … cost to hire one worker in period $t$

$C^L_t$ … cost to lay off one worker in period $t$

$C^I_t$ … cost to hold one unit in inventory in period $t$

$C^b_t$ … cost to backorder one unit in period $t$

Aggregate Planning

Decision Variables:

$P_t$ … number of units produced in period $t$

$W_t$ … number of workers available in period $t$

$H_t$ … number of workers hired in period $t$

$L_t$ … number of workers laid off in period $t$

$I_t$ … number of units held in inventory in period $t$

$B_t$ … number of units backordered in period $t$
Aggregate Planning

Constraints: work, Capacity, force, material

\[ P_t \leq n W_t \quad t = 1, 2, \ldots, T \]

\[ W_t = W_{t-1} + H_t - L_t \quad t = 1, 2, \ldots, T \]

\[ \text{net inventory this period} = \text{net inventory last period} + \text{production this period} - \text{demand this period} \]

\[ I_t - B_t = I_{t-1} - B_{t-1} + P_t - D_t \]

Costs

\[ \sum_{t=1}^{T} (C^P_t P_t + C^W_t W_t + C^H_t H_t + C^L_t L_t + C^I_t I_t + C^B_t B_t) \]

Example: Precision Transfer

- **Planning horizon**: 6 months \( T = 6 \)
- **Costs do not vary over time**: \( C^P_t = 0 \)
- **\( d_t \)**: days in month \( t \)
- **\( C^W_t = $120d_t \)**
- **\( C^H_t = $450 \)**
- **\( C^L_t = $600 \)**
- **\( C^I_t = $5 \)**

- **We assume that no backorders are allowed!**
- **no production costs and no backorder costs are included!**

**Demand**

<table>
<thead>
<tr>
<th>January</th>
<th>February</th>
<th>March</th>
<th>April</th>
<th>May</th>
<th>June</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>2760</td>
<td>3320</td>
<td>3970</td>
<td>3540</td>
<td>3180</td>
<td>2900</td>
<td>19,670</td>
</tr>
</tbody>
</table>

Production Management
Linear Program Model for Precision Transfer

Minimize
\[
252W_1 + 240W_2 + 270W_1 + 252W_4 + 290W_5 + 264W_6
+ 450(H_1 + H_2 + H_3 + H_4 + H_5 + H_6)
+ 600(L_1 + L_2 + L_3 + L_4 + L_5 + L_6)
+ 5(I_1 + I_2 + I_3 + I_4 + I_5 + I_6)
\]

subject to

(Production-capacity constraints)
\[
P_1 \leq 8W_1, \quad P_2 \leq 8W_2, \quad P_3 \leq 8W_3, \quad P_4 \leq 8W_4, \quad P_5 \leq 8W_5, \quad P_6 \leq 8W_6
\]

(Work-force constraints)
\[
W_1 = 35 + H_1 - L_1, \quad W_2 = W_1 + H_2 - L_2, \quad W_3 = W_2 + H_3 - L_3,
W_4 = W_3 + H_4 - L_4, \quad W_5 = W_4 + H_5 - L_5, \quad W_6 = W_5 + H_6 - L_6.
\]

(Inventory-balance constraints)
\[
I_1 = I_1 - P_1, \quad I_2 = I_2 - P_2 - 3320, \quad I_3 = I_3 - P_3 - 3079,
I_4 = I_4 + P_4 - 3540, \quad I_5 = I_5 + P_5 - 3180, \quad I_6 = I_6 + P_6 - 2900
\]

(Non-negativity constraints)
\[
P_1, P_2, P_3, P_4, P_5, P_6, W_1, W_2, W_3, W_4, W_5, W_6, H_1, H_2, H_3, H_4, H_5, H_6,
L_1, L_2, L_3, L_4, L_5, L_6, I_1, I_2, I_3, I_4, I_5, I_6 \geq 0
\]

Aggregate Planning

LP solution (total cost = $600 191.60)

<table>
<thead>
<tr>
<th></th>
<th>Production</th>
<th>Inventory</th>
<th>Hired</th>
<th>Laid off</th>
<th>Workers</th>
</tr>
</thead>
<tbody>
<tr>
<td>January</td>
<td>2940.00</td>
<td>180.00</td>
<td>0.00</td>
<td>0.00</td>
<td>35.00</td>
</tr>
<tr>
<td>February</td>
<td>3232.86</td>
<td>82.86</td>
<td>5.41</td>
<td>0.00</td>
<td>40.41</td>
</tr>
<tr>
<td>March</td>
<td>3877.14</td>
<td>0.00</td>
<td>1.73</td>
<td>0.00</td>
<td>42.14</td>
</tr>
<tr>
<td>April</td>
<td>3540.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>42.14</td>
</tr>
<tr>
<td>May</td>
<td>3180.00</td>
<td>0.00</td>
<td>0.00</td>
<td>6.01</td>
<td>36.14</td>
</tr>
<tr>
<td>June</td>
<td>2900.00</td>
<td>0.00</td>
<td>0.00</td>
<td>3.18</td>
<td>32.95</td>
</tr>
</tbody>
</table>

Production Management 64
**Aggregate Planning**

### Rounding LP solution

<table>
<thead>
<tr>
<th></th>
<th>January</th>
<th>February</th>
<th>March</th>
<th>April</th>
<th>May</th>
<th>June</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days</td>
<td>21</td>
<td>20</td>
<td>23</td>
<td>21</td>
<td>22</td>
<td>22</td>
<td>129</td>
</tr>
<tr>
<td>Units/Worker</td>
<td>84</td>
<td>80</td>
<td>92</td>
<td>84</td>
<td>88</td>
<td>88</td>
<td>516</td>
</tr>
<tr>
<td>Demand</td>
<td>2760</td>
<td>3320</td>
<td>3970</td>
<td>3540</td>
<td>3180</td>
<td>2900</td>
<td>19670</td>
</tr>
<tr>
<td>Workers</td>
<td>35</td>
<td>41</td>
<td>42</td>
<td>42</td>
<td>36</td>
<td>33</td>
<td>229</td>
</tr>
<tr>
<td>Capacity</td>
<td>2940</td>
<td>3280</td>
<td>3864</td>
<td>3528</td>
<td>3168</td>
<td>2904</td>
<td>19684</td>
</tr>
<tr>
<td>Capacity - Demand</td>
<td>180</td>
<td>-40</td>
<td>-106</td>
<td>-12</td>
<td>-12</td>
<td>4</td>
<td>14</td>
</tr>
<tr>
<td>Cumulative Difference</td>
<td>180</td>
<td>140</td>
<td>34</td>
<td>22</td>
<td>10</td>
<td>14</td>
<td>400</td>
</tr>
<tr>
<td>Produced</td>
<td>2930</td>
<td>3280</td>
<td>3864</td>
<td>3528</td>
<td>3168</td>
<td>2900</td>
<td>19670</td>
</tr>
<tr>
<td>Net inventory</td>
<td>170</td>
<td>130</td>
<td>24</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>336</td>
</tr>
<tr>
<td>Hired</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>Laid Off</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>Costs</td>
<td>89050</td>
<td>101750</td>
<td>116490</td>
<td>105900</td>
<td>98640</td>
<td>88920</td>
<td>600750</td>
</tr>
</tbody>
</table>

### Practical Issues

- 100,000 variables and 40,000 constraints
- LP/MIP Solvers: CPLEX, XPRESS-MP, ...

### Extensions

#### Bounds

- $I_i \leq I^U_i$
- $I^L_i \leq I_i \leq I^U_i$
- $L_i \leq 0.05W_i$

#### Training

$W_i = W_{i-1} + H_{i-1} - L_i$
Aggregate Planning

**Transportation Models**
- supply points: periods, initial inventory
- demand points: periods, excess demand, final inventory

- \( n_t \) = capacity during period \( t \)
- \( D_t \) = forecasted number of units demanded in period \( t \)
- \( C_t^p \) = the cost to produce one unit in period \( t \)
- \( C_t^i \) = the cost to hold one unit in inventory in period \( t \)

<table>
<thead>
<tr>
<th>t</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>capacity ( n_t )</td>
<td>350</td>
<td>300</td>
<td>350</td>
</tr>
<tr>
<td>demand</td>
<td>200</td>
<td>300</td>
<td>400</td>
</tr>
<tr>
<td>production costs</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>holding costs</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

initial inventory: 50
final inventory: 75
## Aggregate Planning

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Ending inventory</th>
<th>Excess capacity</th>
<th>Available capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Beginning inventory</strong></td>
<td></td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td><strong>Period 1</strong></td>
<td>50</td>
<td>10</td>
<td>12</td>
<td>50</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td><strong>Period 2</strong></td>
<td>-</td>
<td>-</td>
<td>300</td>
<td>11</td>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td><strong>Period 3</strong></td>
<td>-</td>
<td>-</td>
<td>350</td>
<td>12</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td><strong>Demand</strong></td>
<td>200</td>
<td>300</td>
<td>400</td>
<td>400</td>
<td>75</td>
<td>75</td>
</tr>
</tbody>
</table>

### Extension:
- **overtime**: overtime capacity is 90, 90, and 75 in period 1, 2, and 3;
- **overtime costs** are $16, $18, and $20 for the three periods respectively;
- **backorders**: units can be backordered at a cost of $5 per unit-month;
  production in period 2 can be used to satisfy demand in period 1

---

**Production Management**

69

70
### Aggregate Planning

#### Disaggregating Plans
- Aggregate units are not actually produced, so the plan should consider individual products
- Disaggregation
- Master production schedule

#### Questions:
- In which order should individual products be produced?
  - Example: shortest run-out time \( R_i = I_i / D_i \)
- How much of each product should be produced?
  - Example: balance run-out time

### Table

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Ending inventory</th>
<th>Excess capacity</th>
<th>Available capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beginning inventory</td>
<td>0</td>
<td>25</td>
<td>25</td>
<td>4</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td><strong>Period 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regular time</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>0</td>
<td>350</td>
</tr>
<tr>
<td>Overtime</td>
<td>16</td>
<td>18</td>
<td>20</td>
<td>22</td>
<td>0</td>
<td>90</td>
</tr>
<tr>
<td><strong>Period 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regular time</td>
<td>16</td>
<td>11</td>
<td>13</td>
<td>15</td>
<td>0</td>
<td>350</td>
</tr>
<tr>
<td>Overtime</td>
<td>23</td>
<td>18</td>
<td>20</td>
<td>22</td>
<td>40</td>
<td>90</td>
</tr>
<tr>
<td><strong>Period 3</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regular time</td>
<td>22</td>
<td>17</td>
<td>12</td>
<td>14</td>
<td>0</td>
<td>300</td>
</tr>
<tr>
<td>Overtime</td>
<td>30</td>
<td>25</td>
<td>20</td>
<td>22</td>
<td>0</td>
<td>75</td>
</tr>
<tr>
<td><strong>Demand</strong></td>
<td>400</td>
<td>300</td>
<td>400</td>
<td>75</td>
<td>130</td>
<td>1305</td>
</tr>
</tbody>
</table>
Aggregate Planning

- Advanced Production Planning Models
  - Multiple Products
  - Same notation as before
  - Add subscript i for product i

Objective function

$$\min \sum_{i=1}^{T} \left( C_i^p W_t + C_i^H H_t + C_i^L L_t + \sum_{i=1}^{N} C_i^P P_t + C_i^I I_t \right)$$

subject to

$$\sum_{i=1}^{N} \left( \frac{1}{n_i} \right) P_{it} \leq W_t \quad t = 1, 2, \ldots, T$$

$$W_t = W_{t-1} + H_t - L_t \quad t = 1, 2, \ldots, T$$

$$I_{it} = I_{i-1} + P_{it} - D_t \quad t = 1, 2, \ldots, T; \quad i = 1, 2, \ldots, N$$

$$P_{it}, W_t, H_t, L_t, I_{it} \geq 0 \quad t = 1, 2, \ldots, T; \quad i = 1, 2, \ldots, N$$
Aggregate Planning

- **Computational Effort:**
  - 10 products, 12 periods: 276 variables, 144 constraints
  - 100 products, 12 periods: 2436 variables, 1224 constraints

---

Example: Carolina Hardwood Product Mix
- Carolina Hardwood produces 3 types of dining tables;
- There are currently 50 workers employed who can be hired and laid off at any time;
- Initial inventory is 100 units for table 1, 120 units for table 2 and 80 units for table 3;

<table>
<thead>
<tr>
<th></th>
<th>t</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>costs of hiring</td>
<td>420</td>
<td>410</td>
<td>420</td>
<td>405</td>
<td></td>
</tr>
<tr>
<td>costs of lay off</td>
<td>800</td>
<td>790</td>
<td>790</td>
<td>800</td>
<td></td>
</tr>
<tr>
<td>costs per worker</td>
<td>600</td>
<td>620</td>
<td>620</td>
<td>610</td>
<td></td>
</tr>
</tbody>
</table>
Aggregate Planning

The number of units that can be made by one worker per period:

<table>
<thead>
<tr>
<th></th>
<th>Table 1</th>
<th>Table 2</th>
<th>Table 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>200</td>
<td>300</td>
<td>260</td>
</tr>
<tr>
<td>2</td>
<td>220</td>
<td>310</td>
<td>255</td>
</tr>
<tr>
<td>3</td>
<td>210</td>
<td>300</td>
<td>250</td>
</tr>
<tr>
<td>4</td>
<td>200</td>
<td>290</td>
<td>265</td>
</tr>
</tbody>
</table>

Forecasted demand, unit cost and holding cost per unit are:

<table>
<thead>
<tr>
<th>Demand</th>
<th>Unit costs</th>
<th>Holding costs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Table 2</td>
<td>Table 3</td>
</tr>
<tr>
<td></td>
<td>Table 1</td>
<td>Table 2</td>
</tr>
<tr>
<td>1</td>
<td>3500</td>
<td>5400</td>
</tr>
<tr>
<td>2</td>
<td>120</td>
<td>150</td>
</tr>
<tr>
<td>3</td>
<td>125</td>
<td>150</td>
</tr>
<tr>
<td>4</td>
<td>120</td>
<td>150</td>
</tr>
</tbody>
</table>

Minimize

\[ 600W_1 + 600W_2 + 620W_3 + 620W_4 + 420H_1 + 410H_2 + 430H_3 + 408H_4 + 380W_1 + 380W_2 + 790W_3 + 790W_4 + 125T_1 + 150T_2 + 200T_3 + 125T_4 + 520T_5 + 205T_6 + 125T_7 + 140T_8 + 205T_9 + 1045 + 110T_1 + 112T_2 + 111T_3 + 112T_4 + 110T_5 + 111T_6 + 112T_7 + 111T_8 + 111T_9 \]

Subject to

\[
\begin{align*}
P_1 + P_2 & \leq W_1, \\
P_2 + P_3 & \leq W_2, \\
2P_1 & \leq W_3, \\
2P_2 & \leq W_4.
\end{align*}
\]

\[ \begin{array}{c}
W_1 = 50 \quad I_1, \\
W_2 = W_1 + I_2, \\
W_3 = W_2 + I_3, \\
W_4 = W_3 + I_4.
\end{array} \]

\[ \begin{align*}
I_1 & = 1400, \\
I_2 & = 300, \\
I_3 & = 1200, \\
I_4 & = 1000, \\
I_5 & = 1200, \\
I_6 & = 1200, \\
I_7 & = 1200, \\
I_8 & = 1200.
\end{align*} \]
Aggregate Planning

**Multiple Products and Processes**

- $T$ = horizon length, in periods
- $N$ = number of products
- $K$ = number of resource types
- $t$ = index of periods, $t = 1, 2, \ldots, T$
- $i$ = index of products, $i = 1, 2, \ldots, N$
- $k$ = index of resource types, $k = 1, 2, \ldots, K$
- $D_{it}$ = forecasted number of units demanded for product $i$ in period $t$
- $m_{it}$ = number of different processes available to make product $i$
- $A_{it}$ = amount of resource $k$ available in period $t$
- $a_{ijkt}$ = amount of resource $k$ required by one unit of product $i$ if produced by process $j$
- $C_{ijt}^p$ = cost to produce one unit of product $i$ using process $j$ in period $t$
- $C_{it}^l$ = cost to hold one unit of product $i$ in inventory for period $t$

The decision variables are

- $P_{ijt}$ = number of units of product $i$ produced by process $j$ in period $t$
- $I_{it}$ = number of units of product $i$ held in inventory at the end of period $t$

The linear programming formulation is

Minimize $\sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j=1}^{m_i} (C_{ijt}^p P_{ijt} + C_{it}^l I_{it})$

subject to

$\sum_{j=1}^{m_i} a_{ijkt} P_{ijt} \leq A_{ikt}$ \hspace{1cm} $t = 1, 2, \ldots, T; \; k = 1, 2, \ldots, K$

$I_{it} = I_{it-1} + \sum_{j=1}^{m_i} P_{ijt} - D_{it}$ \hspace{1cm} $t = 1, 2, \ldots, T; \; i = 1, 2, \ldots, N$

$P_{ijt}, I_{it} \geq 0$ \hspace{1cm} $t = 1, 2, \ldots, T; \; i = 1, 2, \ldots, N; \; j = 1, 2, \ldots, m_i$
Aggregate Planning

- Example: Cactus Cycles process plan
- CC produces 2 types of bicycles, street and road;
- Estimated demand and current inventory:

<table>
<thead>
<tr>
<th>t</th>
<th>initial inventory</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>street b.</td>
<td>100</td>
<td>1000</td>
<td>1050</td>
<td>1100</td>
</tr>
<tr>
<td>road b.</td>
<td>50</td>
<td>500</td>
<td>600</td>
<td>550</td>
</tr>
</tbody>
</table>

- Available capacity (hours) and holding costs per bike:

<table>
<thead>
<tr>
<th>Capacity (hours)</th>
<th>Holding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine</td>
<td>Worker</td>
</tr>
<tr>
<td>1</td>
<td>8600</td>
</tr>
<tr>
<td>2</td>
<td>8500</td>
</tr>
<tr>
<td>3</td>
<td>8800</td>
</tr>
</tbody>
</table>

Aggregate Planning

- Process costs (process1, process2) and resource requirement per unit:

<table>
<thead>
<tr>
<th></th>
<th>Process1</th>
<th>Process2</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>Street</td>
<td>Road</td>
</tr>
<tr>
<td>1</td>
<td>72</td>
<td>85</td>
</tr>
<tr>
<td>2</td>
<td>74</td>
<td>88</td>
</tr>
<tr>
<td>3</td>
<td>75</td>
<td>84</td>
</tr>
<tr>
<td>Machine hours required</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>Worker hours required</td>
<td>10</td>
<td>12</td>
</tr>
</tbody>
</table>
Aggregate Planning

Minimize

\[ \begin{align*}
& 7.2P_{11} + 101P_{12} + 88P_{21} + 90P_{22} + 74P_{11} + 78P_{12} + 95P_{22} \\
& + 75P_{31} + 70P_{32} + 64P_{41} + 82P_{42} \\
& + 7I_{11} + 6I_{12} + 7I_{21} + 7I_{22} + 7I_{23} \\
\end{align*} \]

subject to

\[ \begin{align*}
5P_{11} + 4P_{12} + 8P_{21} + 6P_{22} & \leq 8600, \\
5P_{12} + 4P_{12} + 8P_{22} + 6P_{22} & \leq 8600, \\
5P_{32} + 4P_{12} + 8P_{22} + 6P_{22} & \leq 8600, \\
10P_{11} + 8P_{12} + 12P_{21} + 9P_{22} & \leq 17000, \\
bP_{1,2} + bP_{2,2} + bP_{2,2} + bP_{2,2} & \leq 16500, \\
bI_{1,3} + bI_{1,3} + bI_{2,3} + bI_{2,3} & \leq 17200, \\
I_{11} = 100 + n_{11} + n_{121} - 1000, & \quad I_{21} = 50 + n_{21} + n_{221} - 500, \\
I_{12} = 100 + n_{122} + P_{122} - 1000, & \quad I_{22} = 100 + n_{222} + P_{222} - 600, \\
I_{13} = I_{12} + n_{113} + P_{113} - 1000, & \quad I_{23} = I_{22} + n_{213} + P_{213} - 550, \\
n_{ij}, I_{it} \geq 0 & \quad t = 1,2,3; \quad i = 1,2; \quad j = 1,2 \\
\end{align*} \]

Aggregate Planning

\[ \text{solution: Objective Function value} = \$8,534.166 \]

<table>
<thead>
<tr>
<th>Street Bicycle</th>
<th>Road Bicycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td>Process</td>
</tr>
<tr>
<td>t</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>900</td>
</tr>
<tr>
<td>2</td>
<td>1050</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>
Aggregate Planning - Extensions

Hopp/Spearman, S. 522-540

Notation:

\( X_{it} \) ... amount of product i produced in period t

\( r_i \) ... net profit from one unit of product i

\( S_{it} \) ... amount of product i sold in period t

\( a_{ij} \) ... time required on workstation j to produce one unit of product i

\( c_j \) ... capacity of workstation j in period t in units (consistent with \( a_{ij} \))

Backorders

\[
\max \sum_{i=1}^{m} \sum_{t=1}^{T} r_i S_{it} - h_i I^+_i - \pi_i I^-_i
\]

subject to

\[
d_i \leq S_{it} \leq \bar{d}_i \quad \text{for all } i, t
\]

\[
\sum_{i=1}^{m} a_{ij} X_{it} \leq c_j \quad \text{for all } j, t
\]

\[
I^-_i = I^+_{i-1} + X_{it} - S_{it} \quad \text{for all } i, t
\]

\[
I^+_i = I^-_{i-1} - I^-_i \quad \text{for all } i, t
\]

\[
X_{it}, S_{it}, I^+_i, I^-_i \geq 0 \quad \text{for all } i, t
\]
Aggregate Planning - Extensions

**Overtime**

\[ I'_j = \text{cost of one hour of overtime at workstation } j \]
\[ O_{jt} = \text{overtime at workstation } j \text{ in period } t \text{ in hours} \]

\[
\max \sum_{t=1}^{T} \left\{ \sum_{i=1}^{m} \left( r_{it} S_{it} - h_{it} I^+_i - \pi_i I^-_i \right) + \sum_{j=1}^{n} I'O_{jt} \right\}
\]

subject to

\[
\sum_{i=1}^{m} a_{ij} X_{it} \leq c_{jt} + O_{jt} \quad \text{for all } i, t
\]

\[ X_{it}, S_{it}, I^+_i, I^-_i, O_{jt} \geq 0 \quad \text{for all } i, t \]

---

**Yield loss**

\[ \alpha, \beta, \gamma \ldots \text{fraction of output that is lost} \]
\[ y_j \ldots \text{cumulative yield from station } j \text{ onward} \]

\[ \text{(including station } j) \text{ for product } i \]

we must release \( \frac{d}{y_j} \) units of \( i \) into station \( j \)
Aggregate Planning - Extensions

- **Basic model + Yield loss extension (no backorders)**

\[
\max \sum_{i=1}^{r} \sum_{j=1}^{m} (r_i S_{ij} - h_i I_{ij})
\]

subject to

\[
d_{ij} \leq S_{ij} \leq \bar{d}_{ij} \quad \text{for all } i, t
\]

\[
\sum_{i=1}^{m} g_i x_{ij} \leq c_{jt} \quad \text{for all } j, t
\]

\[
I_{ij} = I_{ij-1} + X_{ij} - S_{ij} \quad \text{for all } i, t
\]

\[
X_{ij}, S_{ij}, I_{ij} \geq 0 \quad \text{for all } i, t
\]

---

Aggregate Planning - Workforce Planning

- **Single product, workforce resizing, overtime allocation**

- **Notation**

  - \( b \): number of man-hours required to produce one unit of product
  - \( l \): cost of regular time in dollars/man-hour
  - \( l' \): cost of overtime in dollars/man-hour
  - \( e \): cost to increase workforce by one man-hour per period
  - \( e' \): cost to decrease workforce by one man-hour per period
  - \( W_t \): workforce in period \( t \) in man-hours of regular time
  - \( H_t \): increase in workforce from period \( t-1 \) to \( t \) in man-hours
  - \( F_t \): decrease in workforce from period \( t-1 \) to \( t \) in man-hours
  - \( O_t \): overtime in period \( t \) in hours
Aggregate Planning - Workforce Planning

LP formulation:
maximize net profit, including labor, overtime, holding, and hiring/ firing costs
subject to constraints
on sales, capacity,...

\[
\begin{align*}
\text{maximize } & \sum_{t=1}^{T} \left\{ S_t - h_t I_t - l_t W_t - l_t' O_t - e_t H_t - e_t' F_t \right\} \\
\text{subject to } & \quad d_t \leq S_t \leq \bar{d}_t \quad \text{for all } t \\
& \quad a_j X_t \leq c_{jt} \quad \text{for all } j, t \\
& \quad I_t = I_{t-1} + X_t - S_t \quad \text{for all } t \\
& \quad W_t = W_{t-1} + H_t - F_t \quad \text{for all } t \\
& \quad b X_t \leq W_t + O_t \quad \text{for all } t \\
& \quad X_t, S_t, I_t, O_t, W_t, H_t, F_t \geq 0 \quad \text{for all } t
\end{align*}
\]

AP-WP Example

- Revenue: 1000$
- Worker capacity: 168h/ month
- Initially 15 workers
- No initial inventory
- Holding costs: 10$/unit/ month
- Regular labor costs: 35$/ hour
- Overtime: 150% of regular
- Hiring costs: 2500$ (2500/ 168 ~ 15$ per man-hour)
- Lay-off costs: 1500$ (1500/ 168 ~ 9$ per man-hour)
- No backordering
- Demands over 12 months: 200, 220, 230, 300, 400, 450, 320, 180, 170, 170, 160, 180
- Demands must be met! (S=D)
AP-WP Example (cont.)

* Determine over a 12 month horizon:
  - Number of workers: \( W \)
  - Output: \( X \)
  - Overtime use: \( O \)
  - Inventory: \( I \)
  - (\( H, F \) are additional choice variables in the model)

Aggregate Planning - Workforce Planning

<table>
<thead>
<tr>
<th>Parameter</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>300</td>
<td>290</td>
<td>300</td>
<td>290</td>
<td>400</td>
<td>400</td>
<td>500</td>
<td>500</td>
<td>500</td>
<td>500</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>a</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>b</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>H</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decision Variables</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>H</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

| Objective: Profit | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Production Management 93
Aggregate Planning-Summary

The following scenarios have been discussed:

- **single product, single resource, single process**
  find: workforce, output, inventory (w. or w/o backorders)

- **multiple products, single resource, single process**
  find: workforce, all outputs, all inventories (w. or w/o backorders)

- **multiple products, multiple resources, multiple processes** (workforce given)
  find: all outputs, all inventories, use of processes

Aggregate Planning-Summary

The following scenarios have been discussed:

- **multiple products, multiple workstations**
  (workstation capacities given)
  find: all sales, all outputs, all inventories (w. or w/o backorders)

- **multiple products, multiple workstations**
  find: all sales, all outputs, all inventories (w. or w/o backorders), OT

- **single product, multiple workstations, one resource**
  find: workforce, all sales, all outputs, all inventories (w. or w/o backorders), OT
Aggregate Planning

- Work to do:

- Examples: 5.7, 5.8abcdef, 5.9abcd, 5.10abcd, 5.16abcd, 5.21, 5.22, 5.29, 5.30

Replace capacity columns of table in problem 5.29 with

<table>
<thead>
<tr>
<th>Month</th>
<th>Machine</th>
<th>Worker</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1350</td>
<td>19000</td>
</tr>
<tr>
<td>2</td>
<td>1270</td>
<td>19000</td>
</tr>
<tr>
<td>3</td>
<td>1350</td>
<td>19500</td>
</tr>
</tbody>
</table>

- Minicase BF SWING II

Chapter 7

Production, Capacity and Material Planning
Production, Capacity and Material Planning

- **Production plan**
  - quantities of final product, subassemblies, parts needed at distinct points in time

- **To generate the Production plan we need:**
  - end-product demand forecasts
  - Master production schedule

- **Master production schedule (MPS)**
  - delivery plan for the manufacturing organization
  - exact amounts and delivery timings for each end product
  - accounts for manufacturing constraints and final goods inventory

Based on the MPS:

- **Rough-cut capacity planning**

- **Material requirements planning**
  - determines material requirements and timings for each phase of production
  - *detailed capacity planning*
Production, Capacity and Material Planning

- End-Item Demand Estimate
- Master Production Schedule (MPS)
- Material Requirements Planning (MRP)
- Material Plan
- Purchasing Plan
- Shop Orders
- Shop Floor Control
- Updates
- Updates

Master Production Scheduling

- Aggregate plan
- Demand estimates for individual end-items
- Demand estimates vs. MPS
  - inventory
  - capacity constraints
  - availability of material
  - production lead time
  - ...
- Market environments
  - make-to-stock (MTS)
  - make-to-order (MTO)
  - assemble-to-order (ATO)
Master Production Scheduling

**MTS**
- produces in batches
- minimizes customer delivery times at the expense of holding finished-goods inventory
- MPS is performed at the end-item level
- production starts before demand is known precisely
- small number of end-items, large number of raw-material items

**MTO**
- no finished-goods inventory
- customer orders are backlogged
- MPS is order driven, consists of firm delivery dates

Master Production Scheduling

**ATO**
- large number of end-items are assembled from a relatively small set of standard subassemblies, or modules
- automobile industry
- MPS governs production of modules (forecast driven)
- Final Assembly Schedule (FAS) at the end-item level (order driven)
- 2 lead times, for consumer orders only FAS lead time relevant
**Master Production Scheduling**

- **MPS - SIBUL manufactures phones**
  - three desktop models A, B, C
  - one wall telephone D
  - MPS is equal to the demand forecast for each model

<table>
<thead>
<tr>
<th>WEEKLY MPS (FORECAST)</th>
<th>Jan</th>
<th>Feb</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Week</td>
<td>Week</td>
</tr>
<tr>
<td>Product</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Model B</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>Model C</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>Model D</td>
<td>600</td>
<td>600</td>
</tr>
<tr>
<td>weekly total</td>
<td>3100</td>
<td>3000</td>
</tr>
<tr>
<td>monthly total</td>
<td>12200</td>
<td></td>
</tr>
</tbody>
</table>

**MPS Planning - Example**

- MPS plan for model A of the previous example:
  - Make-to-stock environment
  - No safety-stock for end-items

\[ I_t = I_{t-1} + Q_t - \max(F_t, O_t) \]

- \( I_t \) = end-item inventory at the end of week \( t \)
- \( Q_t \) = manufactured quantity to be completed in week \( t \)
- \( F_t \) = forecast for week \( t \)
- \( O_t \) = customer orders to be delivered in week \( t \)

<table>
<thead>
<tr>
<th>INITIAL DATA Model A</th>
<th>Jan</th>
<th>Feb</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Week</td>
<td>Week</td>
</tr>
<tr>
<td>Current Inventory</td>
<td>1600</td>
<td></td>
</tr>
<tr>
<td>orders Ot</td>
<td>1200 800 300 200 100</td>
<td></td>
</tr>
</tbody>
</table>
Master Production Scheduling

**Batch production: batch size = 2500**

- \( Q_t = \max \{0, I_t \} - \max \{F_t, O_t \} \)
- \( Q_t = 0, \) if \( I_t > 0 \)
- \( Q_t = 2500, \) otherwise

- \( I_1 = \max \{0, 1600 \} - \max \{1000, 1200 \} = 400 > 0 \)
- \( I_2 = \max \{0, 400 \} - \max \{1000, 800 \} = -600 < 0 \Rightarrow Q_2 = 2500 \)
- \( I_2 = 2500 + 400 - \max \{1000, 800 \} = 1900, \) etc.

<table>
<thead>
<tr>
<th>MPS</th>
<th>Jan</th>
<th>Feb</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current Inventory = 1600</strong></td>
<td>Week</td>
<td>Week</td>
</tr>
<tr>
<td>orders Qt</td>
<td>1200</td>
<td>800</td>
</tr>
<tr>
<td>inventory It</td>
<td>1600</td>
<td>400</td>
</tr>
<tr>
<td>MPS Qt</td>
<td>2500</td>
<td>2500</td>
</tr>
<tr>
<td>ATP</td>
<td>400</td>
<td>1400</td>
</tr>
</tbody>
</table>

**Master Production Scheduling**

- **Available to Promise (ATP)**
  - \( ATP_1 = 1600 + 0 - 1200 = 400 \)
  - \( ATP_2 = 2500 - (800 + 300) = 1400, \) etc.

- Whenever a new order comes in, ATP must be updated

- **Lot-for-Lot production**

<table>
<thead>
<tr>
<th>MPS</th>
<th>Jan</th>
<th>Feb</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current Inventory = 1600</strong></td>
<td>Week</td>
<td>Week</td>
</tr>
<tr>
<td>orders Qt</td>
<td>1200</td>
<td>800</td>
</tr>
<tr>
<td>inventory It</td>
<td>1600</td>
<td>400</td>
</tr>
<tr>
<td>MPS Qt</td>
<td>0</td>
<td>600</td>
</tr>
<tr>
<td>ATP</td>
<td>400</td>
<td>0</td>
</tr>
</tbody>
</table>
Master Production Scheduling

MPS Modeling
- differs between MTS-ATO and MTO
- find final assembly lot sizes
- additional complexity because of joint capacity constraints
- cannot be solved for each product independently

Master Production Scheduling

Make-To-Stock-Modeling

\[
Q_i = \text{production quantity of product } i \text{ in period } t \\
I_i = \text{Inventory of product } i \text{ at end of period } t \\
D_i = \text{demand (requirements) for product } i \text{ in time period } t \\
a_i = \text{production hours per unit of product } i \\
h_i = \text{inventory holding cost per unit of product } i \text{ per time period} \\
A_i = \text{set-up cost for product } i \\
G_i = \text{production hours available in period } t \\
y_i = 1, \text{if set-up for product } i \text{ occurs in period } t (Q_i > 0)
\]
**Master Production Scheduling**

- **Make-To-Stock-Modeling**

\[
\begin{align*}
\text{min } & \sum_{a} \sum_{t=1}^{T} (A_{i} y_{it} + h_{i} I_{it}) \\
I_{i,t+1} + Q_{a,t} - I_{it} = D_{it} & \quad \text{for all } (i,t) \\
\sum_{a} a_{it} Q_{a,t} \leq G_{t} & \quad \text{for all } t \\
Q_{a,t} - y_{it} \sum_{k=1}^{T} D_{k,t} \leq 0 & \quad \text{for all } (i,t) \\
Q_{a,t} \geq 0; I_{it} \geq 0; y_{it} \in \{0,1\}
\end{align*}
\]

- **Assemble-To-Order Modeling**

- **two master schedules**
  - MPS: forecast-driven
  - FAS: order driven
- **overage costs**
  - holding costs for modules and assembled products
- **shortage costs**
  - final product assembly based on available modules
  - no explicit but implicit shortage costs for modules
  - final products: lost sales, backorders
Master Production Scheduling

- m module types and n product types
- \(Q_{kt}\) = quantity of module k produced in period t
- \(g_{kj}\) = number of modules of type k required to assemble order j

**Decision Variables:**
- \(I_{kt}\) = inventory of module k at the end of period t
- \(y_{jt}\) = 1, if order j is assembled and delivered in period t; 0, otherwise
- \(h_k\) = holding cost
- \(\pi_{jt}\) = penalty costs, if order j is satisfied in period t and order j is due in period t' (t'<t); holding costs if t' > t

Master Production Scheduling

- **Assemble-To-Order Modeling**
  \[
  \min \sum_{k=1}^{m} \sum_{t=1}^{L} h_k I_{kt} + \sum_{j=1}^{n} \sum_{t=1}^{L} \pi_{jt} y_{jt}
  \]
  subject to
  \[
  I_{kt} = I_{k,t-1} + Q_{kt} - \sum_{j=1}^{n} g_{kj} y_{jt} \quad \text{for all (k, t)}
  \]
  \[
  \sum_{j=1}^{n} a_j y_{jt} \leq G_t \quad \text{for all t}
  \]
  \[
  \sum_{t=1}^{L} y_{jt} = 1 \quad \text{for all j}
  \]
  \[
  I_{kt} \geq 0; \quad y_{jt} \in \{0,1\} \quad \text{for all (j, k, t)}
  \]
Master Production Scheduling

**Capacity Planning**
- Bottleneck in production facilities
- Rough-Cut Capacity Planning (RCCP) at MPS level
- Feasibility
- Detailed capacity planning (CRP) at MRP level
- Both RCCP and CRP are only providing information

### MPS:

<table>
<thead>
<tr>
<th>Product</th>
<th>January</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1000 1000 1000 1000</td>
</tr>
<tr>
<td>B</td>
<td>- 500 500 -</td>
</tr>
<tr>
<td>C</td>
<td>1500 1500 1500 1500</td>
</tr>
<tr>
<td>D</td>
<td>600 - 600 -</td>
</tr>
</tbody>
</table>

#### Bill of capacity (min)

<table>
<thead>
<tr>
<th></th>
<th>Assembly</th>
<th>Inspection</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>20 2</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>24 2.5</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>22 2</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>25 2.4</td>
<td></td>
</tr>
</tbody>
</table>

#### Capacity requires (hr)

<table>
<thead>
<tr>
<th></th>
<th>Week</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly</td>
<td>1133 1083 1333</td>
</tr>
<tr>
<td>Inspection</td>
<td>107 104 128</td>
</tr>
</tbody>
</table>

- Weekly capacity requirements?
- Assembly: 1000*20 + 1500*22 + 600*25 = 68000 min = 1133,33 hr
- Inspection: 1000*2 + 1500*2 + 600*2.4 = 6440 min = 107,33 hr etc.
- Available capacity per week is 1200 hr for the assembly work center and 110 hours for the inspection station;
Master Production Scheduling

- Infinite capacity planning (information providing)
- Finding a feasible cost optimal solution is a NP-hard problem
- If no detailed bill of capacity is available: capacity planning using overall factors (globale Belastungsfaktoren)
  - Required input:
    - MPS
    - Standard hours of machines or direct labor required
    - Historical data on individual shop workloads (%)
- Example from Günther/Tempelmeier
  - C133.3: overall factors

Master Production Scheduling

Capacity planning using overall factors

<table>
<thead>
<tr>
<th>Product</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100</td>
<td>80</td>
<td>120</td>
<td>100</td>
<td>120</td>
<td>60</td>
</tr>
<tr>
<td>B</td>
<td>40</td>
<td>-</td>
<td>60</td>
<td>-</td>
<td>40</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product</th>
<th>Work on critical machine</th>
<th>Work on non-critical machine</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>4</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

Historic capacity requirements on critical machines:
- 40% on machine A
- 60% on machine B
Master Production Scheduling

in total 500 working units are available per week, 80 on machine a and 120 on machine b;

Solution:
overall factor = time per unit x historic capacity needs

product A:
machine a: 1 x 0,4 = 0,4
machine b: 1 x 0,6 = 0,6

product B:
machine a: 4 x 0,4 = 1,6
machine b: 4 x 0,6 = 2,4

Master Production Scheduling

capacity requirements: product A

<table>
<thead>
<tr>
<th>machine</th>
<th>week</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td>40</td>
<td>32</td>
<td>48</td>
<td>40</td>
<td>48</td>
<td>24</td>
</tr>
<tr>
<td>b</td>
<td></td>
<td>60</td>
<td>48</td>
<td>72</td>
<td>60</td>
<td>72</td>
<td>36</td>
</tr>
<tr>
<td>other</td>
<td></td>
<td>200</td>
<td>160</td>
<td>240</td>
<td>200</td>
<td>240</td>
<td>120</td>
</tr>
</tbody>
</table>

capacity requirements: product B

<table>
<thead>
<tr>
<th>machine</th>
<th>week</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
<td>64</td>
<td>-</td>
<td>96</td>
<td>-</td>
<td>64</td>
<td>-</td>
</tr>
<tr>
<td>b</td>
<td></td>
<td>96</td>
<td>-</td>
<td>144</td>
<td>-</td>
<td>96</td>
<td>-</td>
</tr>
<tr>
<td>other</td>
<td></td>
<td>80</td>
<td>-</td>
<td>120</td>
<td>-</td>
<td>80</td>
<td>-</td>
</tr>
</tbody>
</table>
### Master Production Scheduling

#### Total Capacity Requirements

<table>
<thead>
<tr>
<th>Machine</th>
<th>Week 1</th>
<th>Week 2</th>
<th>Week 3</th>
<th>Week 4</th>
<th>Week 5</th>
<th>Week 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>104</td>
<td>32</td>
<td>144</td>
<td>40</td>
<td>112</td>
<td>24</td>
</tr>
<tr>
<td>b</td>
<td>156</td>
<td>48</td>
<td>216</td>
<td>60</td>
<td>168</td>
<td>36</td>
</tr>
<tr>
<td>other</td>
<td>280</td>
<td>160</td>
<td>360</td>
<td>200</td>
<td>320</td>
<td>120</td>
</tr>
</tbody>
</table>

#### Graphical Representation

- **a** (max 80)
- **b** (max 120)
- **other** (max 300)
Master Production Scheduling

**Capacity Modeling**
- Heuristic approach for finite-capacity-planning
- Based on input/output analysis
- Relationship between capacity and lead time

- $G =$ work center capacity
- $R_t =$ work released to the center in period $t$
- $Q_t =$ production (output) from the work center in period $t$
- $W_t =$ work in process in period $t$
- $U_t =$ queue at the work center measured at the beginning of period $t$, prior to the release of work
- $L_t =$ lead time at the work center in period $t$

**Lead time is not constant**

**Assumptions:**
- Constant production rate
- Any order released in this period is completed in this period

\[
Q_t = \min \left\{ G, U_{t-1} + R_t \right\}
\]
\[
U_t = U_{t-1} + R_t - Q_t
\]
\[
W_t = U_{t-1} + R_t = U_t + Q_t
\]
\[
L_t = \frac{W_t}{G}
\]
Master Production Scheduling

**Example**

<table>
<thead>
<tr>
<th>Period</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>G (hr/week)</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>R (hours)</td>
<td>20</td>
<td>30</td>
<td>60</td>
<td>20</td>
<td>40</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Q (hours)</td>
<td>30</td>
<td>30</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>U (hours)</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>8</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>W (hours)</td>
<td>30</td>
<td>30</td>
<td>60</td>
<td>44</td>
<td>48</td>
<td>52</td>
<td>52</td>
</tr>
<tr>
<td>L (weeks)</td>
<td>0.83</td>
<td>0.83</td>
<td>1.67</td>
<td>1.22</td>
<td>1.33</td>
<td>1.44</td>
<td>1.44</td>
</tr>
</tbody>
</table>

Material Requirements Planning

**Inputs**
- master production schedule
- inventory status record
- bill of material (BOM)

**Outputs**
- planned order releases
  - purchase orders(supply lead time)
  - workorders(manufacturing lead time)
Material Requirements Planning

Legend:
S/A = subassembly
PP = purchased part
MP = manufactured part
RM = raw material

MRP Process
- goal is to find net requirements (trigger purchase and work orders)
- explosion
  - Example:
    - MPS, 100 end items
    - yields gross requirements
- netting
  - Net requirements = Gross requirements - on hand inventory - quantity on order
  - done at each level prior to further explosion
- offsetting
  - the timing of order release is determined
- lotsizing
  - batch size is determined
### Material Requirements Planning

#### Example 7-6

![Diagram of Telephone assembly process]

#### PART 11 (gross requirements given)

<table>
<thead>
<tr>
<th>week</th>
<th>current</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduled receipts</td>
<td></td>
<td>400</td>
<td>700</td>
<td>200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>projected inventory balance</td>
<td></td>
<td>1200</td>
<td>1600</td>
<td>1700</td>
<td>900</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>planned receipts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>planned order release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Net requirements?**
- Net requ. (week 2) = 600 – (1600 + 700) = -1700 => Net requ. (week 2) = 0
- Net requ. (week 3) = 1000 – (1700 + 200) = -900 => Net requ. (week 3) = 0
- Net requ. (week 4) = 1000 – 900 = 100 etc.
Material Requirements Planning

Assumptions:
- lot size: 3000
- lead time: 2 weeks

<table>
<thead>
<tr>
<th>week</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheduled receipts</td>
<td>400</td>
<td>700</td>
<td>200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>projected inventory balance</td>
<td>1200</td>
<td>1600</td>
<td>1700</td>
<td>900</td>
<td>2800</td>
<td>900</td>
<td>1900</td>
<td>2900</td>
</tr>
<tr>
<td>planned receipts</td>
<td></td>
<td>-3000</td>
<td></td>
<td></td>
<td>3000</td>
<td></td>
<td>3000</td>
<td></td>
</tr>
<tr>
<td>planned order release</td>
<td>3000</td>
<td></td>
<td>3000</td>
<td></td>
<td>3000</td>
<td></td>
<td>3000</td>
<td></td>
</tr>
</tbody>
</table>

Material Requirements Planning

*Multilevel explosion*

<table>
<thead>
<tr>
<th>part number</th>
<th>description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>base assembly</td>
<td>1</td>
</tr>
<tr>
<td>121</td>
<td>housing S/A</td>
<td>1</td>
</tr>
<tr>
<td>123</td>
<td>rubber pad</td>
<td>4</td>
</tr>
<tr>
<td>1211</td>
<td>key pad</td>
<td>1</td>
</tr>
</tbody>
</table>

- lead time is one week
- lot for lot for parts 121, 123, 1211
- part 12: fixed lot size of 3000
### Material Requirements Planning

#### MRP Updating Methods

- **MRP systems operate in a dynamic environment**
- **Regeneration method**: the entire plan is recalculated
- **Net change method**: recalculates requirements only for those items affected by change

<table>
<thead>
<tr>
<th>Product</th>
<th>February</th>
<th>Updated MPS for February</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Week</td>
<td>Week</td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>300</td>
</tr>
<tr>
<td>B</td>
<td>300</td>
<td>400</td>
</tr>
<tr>
<td>C</td>
<td>800</td>
<td>1000</td>
</tr>
<tr>
<td>D</td>
<td>500</td>
<td>700</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Net Change for February</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week</td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
</tbody>
</table>
Material Requirements Planning

*Additional Netting procedures*
- implosion:
  - opposite of explosion
  - finds common item
- combining requirements:
  - process of obtaining the gross requirements of a common item
- pegging:
  - identify the item's end product
  - useful when item shortages occur

Lot Sizing in MRP
- minimize set-up and holding costs
- can be formulated as MIP
- a variety of heuristic approaches are available
- simplest approach: use independent demand procedures (e.g. EOQ) at every level
Material Requirements Planning

MIP Formulation

Indices:
- \( i = 1 \ldots P \) label of each item in BOM (assumed that all labels are sorted with respect to the production level starting from the end-items)
- \( t = 1 \ldots T \) period \( t \)
- \( m = 1 \ldots M \) resource \( m \)

Parameters:
- \( \Gamma(i) \) set of immediate successors of item \( i \)
- \( \Gamma^{-1}(i) \) set of immediate predeccessors of item \( i \)
- \( s_i \) setup cost for item \( i \)
- \( c_{ij} \) quantity of item \( i \) required to produce item \( j \)
- \( h_i \) holding cost for one unit of item \( i \)
- \( a_{im} \) capacity needed on resource \( m \) for one unit of item \( i \)
- \( b_{im} \) capacity needed on resource \( m \) for the setup process of item \( i \)
- \( L_{mt} \) available capacity of resource \( m \) in period \( t \)
- \( o_{cm} \) overtime cost of resource \( m \)
- \( G \) large number, but as small as possible (e.g. sum of demands)
- \( D_{it} \) external demand of item \( i \) in period \( t \)

Decision variables:
- \( x_{it} \) delivered quantity of item \( i \) in period \( t \)
- \( I_{it} \) inventory level of item \( i \) at the end of period \( t \)
- \( O_{mt} \) overtime hours required for machine \( m \) in period \( t \)
- \( y_{it} \) binary variable indicating if item \( i \) is produced in period \( t \) (=1) or not (=0)

Equations:

\[
\begin{align*}
\min & \sum_{i=1}^{P} \sum_{t=1}^{T} (s_i y_{it} + h_i I_{it}) + \sum_{t=1}^{T} \sum_{m=1}^{M} o_{cm} O_{mt} \\
I_{it} &= I_{it-1} + x_{it} - \sum_{j \in \Gamma^{-1}(i)} c_{ij} x_{jt} - D_{it} \quad \forall i, t \\
& \quad x_{it} - G y_{it} \leq 0 \quad \forall i, t \\
\sum_{i=1}^{P} (a_{mi} x_{it} + b_{mi} y_{it}) &\leq L_{mi} + O_{mt} \quad \forall m, t \\
x_{it}, I_{it}, O_{mt} &\geq 0, \quad y_{it} \in \{0,1\} \quad \forall i, m, t
\end{align*}
\]
Material Requirements Planning

**Multi-Echelon Systems**
- Multi-echelon inventory
- Each level is referred as an echelon
- "Total inventory in the system varies with the number of stocking points"
- Modell (Freeland 1985):
  - Demand is insensitive to the number of stocking points
  - Demand is normally distributed and divided evenly among the stocking points,
  - Demands at the stocking points are independent of one another
  - A (Q,R) inventory policy is used
  - Service level (fill rate) is applied
  - Q is determined from the EOQ formula

**Reorder point in (Q,R) policies:**
- i: total annual inventory costs (%)
- c: unit costs
- A: ordering costs
- τ: lead time
- στ: variance of demand in lead time

**Given a fill rate β choose z(β) such that:**
\[
L(z) = \int_{z}^{\infty} (y - z) \phi(y) dy = \frac{(1 - \beta)Q}{\sigma_{\tau}}
\]
\[\phi: \text{density of } N(0,1) \text{ distribution; } L(z): \text{standard loss function}\]
Material Requirements Planning

- **Safety stock:** \( s = z \cdot \sigma_t \)
- **Reorder point:** \( R = D_t + z \cdot \sigma_t \)
- **Order quantity:** \( Q = \text{EOQ} = \sqrt{\frac{2AD}{ic}} \)
- **Average inventory:** \( \bar{I}(1) = \frac{Q}{2} + s \)
  
  \( \bar{I}(n) = \text{average inventory for } n \text{ stocking points} \)
  
  \( \bar{I}(1) = \frac{1}{2} \sqrt{\frac{2AD}{ic}} + z \sigma_t \)
Material Requirements Planning

for two stocking points:
demand at each point: $D/2$
variance of lead-time demand: $\sigma_r^2 / 2$
standard deviation is: $\sigma_r / \sqrt{2}$

average inventory at each stocking point is:
\[
\frac{1}{2} \sqrt{\frac{2AD}{ic}} + \frac{z\sigma_r}{\sqrt{2}} = \frac{1}{\sqrt{2}}(Q/2 + s)
\]

Material Requirements Planning

the average inventory for two stocking point is:
\[
\bar{I}(2) = 2 \left[ \frac{1}{\sqrt{2}}(Q/2 + s) \right] = \sqrt{2}(Q/2 + s) = \sqrt{2} \cdot \bar{I}(1)
\]
\[
\bar{I}(n) = \sqrt{n} \cdot \bar{I}(1)
\]
for each level the safety stock is: $s/\sqrt{n}$
the total safety stock is $\sqrt{n} \cdot s$
Material Requirements Planning

Example: At the packaging department of a sugar refinery:

A very-high-grade powdered sugar:

Sugar-refining lead time is five days;
Production lead time (filling time) is negligible;
Annual demand: \( D = 800 \) tons and \( \sigma = 2.5 \)
Lead-time demand is normally distributed with \( D\tau = 16 \) tons and \( \sigma\tau = 3.54 \) tons
Fill rate = 95%
\( A = 50, \ c = 4000, \ i = 20\% \)

Material Requirements Planning

Inventory at level 0 and 1? Safety stock?

\[
Q = \sqrt{\frac{2AD}{ic}} = \sqrt{\frac{2 \times 50 \times 800}{800}} = 10 \ tons
\]

\( \beta = 0.95 \Rightarrow z = 0.71 \)
\( s = z\sigma\tau = 0.71 \times 3.54 = 2.51 \) tons

Suppose we keep inventory in level 0 only, i.e., \( n = 1 \):

\[
I(1) = \frac{Q}{2} + s = \frac{10}{2} + 2.51 = 7.51 \ tons
\]

Suppose inventory is maintained at both level 0 and level 1, i.e., \( n = 2 \):

\[
I(2) = \sqrt{2I(1)} = 10.62 \ tons
\]

The safety stock in each level is going to be:

\[
\frac{s}{\sqrt{2}} = \frac{2.51}{\sqrt{2}} = 1.77 \ tons
\]
Material Requirements Planning

MRP as Multi-Echelon Inventory Control
- continuous-review type policy (Q, R)
- hierarchy of stocking points (installation)
- installation stock policy
- echelon stock (policy): installation inventory position plus all downstream stock
- MRP:
  - rolling horizon
  - level by level approach
  - bases ordering decisions on projected future installation inventory level

Material Requirements Planning

- All demands and orders occur at the beginning of the time period
- orders are initiated immediately after the demands, first for the final items and then successively for the components
- all demands and orders are for an integer number of units
- T = planning horizon
- τ_i = lead time for item i
- s_i = safety stock for item I
- R_i = reorder point for item I
- Q_i = Fixed order quantity of item i
- D_t = external requirements of item i in period t
Material Requirements Planning

Installation stock policies \((Q,R)\) for MRP:
- A production order is triggered if the installation stock minus safety stock is insufficient to cover the requirements over the next \(\tau)\) periods.
- An order may consist of more than one order quantity \(Q\).
- If lead time \(\tau)\) = 0, the MRP is equal to an installation stock policy.
- Safety stock = reorder point.

Echelon stock policies \((Q,R^e)\) for MRP:
- Consider a serial assembly system.
- Installation 1 is the downstream installation (final product).
- The output of installation \(i\) is the input when producing one unit of item \(i-1\) at the immediate downstream installation.
- \(w_i\) = installation inventory position at installation \(i\).
- \(I_i\) = echelon inventory position at installation \(i\) (at the same moment).

\[ I_i = w_i + w_{i-1} + \ldots + w_1 \]

- A multi-echelon \((Q,R)\) policy is denoted by \((Q,R^e)\).
- \(R^e\) gives the reorder point for echelon inventory at \(i\).
Material Requirements Planning

\[ R_{1}^e = s_1 + D_{t1} \]
\[ R_{i}^e = s_i + D_{ti} + R_{i-1}^e + Q_{i-1} \]

**Example:**

**Two-level system, 6 periods**

\[ I_0^0 = 18, I_0^1 = 38, R_1^e = 20, R_2^e = 34, Q_1 = 10, Q_2 = 30 \]

**D = 2 (Item 1), \( \tau_1 = 1, \tau_2 = 2 \)**

---

<table>
<thead>
<tr>
<th>Period</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demand</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

**Item 1**

| Level w1 | 18 | 24 | 24 | 24 | 29 | 28 |
| Production | 10 | 0 | 0 | 0 | 10 | 0 |
| Level w2 | 10 | 10 | 10 | 10 | 30 | 10 |
| Production | 0 | 0 | 30 | 0 | 9 | 0 |

**Item 2**

Suppose now that five units were demanded in period 2:

<table>
<thead>
<tr>
<th>Period</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demand</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

**Item 1**

| Level w1 | 23 | 21 | 19 | 27 | 25 | 23 |
| Production | 0 | 0 | 10 | 0 | 10 | 0 |

**Item 2**

| Level w2 | 10 | 10 | 30 | 30 | 30 | 30 |
| Production | 30 | 0 | 0 | 0 | 0 | 0 |
Material Requirements Planning

Lot Size and Lead Time
- lead time is affected by capacity constraints
- lot size affects lead time

Batching effect
- an increase in lot size should increase lead time

Saturation effect
- when lot size decreases, and set-up is not reduced, lead time will increase

Expected lead time can be calculated using models from queueing theory (M/G/1)

\[
L = \text{lead time} \\
L = \frac{\left(\frac{\lambda}{\mu}\right)^2 + \lambda^2 \sigma^2}{2\lambda(1 - \frac{\lambda}{\mu})} + \frac{1}{\mu}
\]

\[\lambda = \text{mean arrival rate} \]
\[\mu = \text{mean service rate} \]
\[\sigma^2 = \text{service time variance} \]
Material Requirements Planning

\[ D_j = \text{demand per period for product } j \]
\[ t_j = \text{unit production time for product } j \]
\[ S_j = \text{set-up time for product } j \]
\[ Q_j = \text{lotsize for product } j \]

mean arrival rate of batches:
\[ \lambda = \sum_{j=1}^{n} \lambda_j = \sum_{j=1}^{n} \frac{D_j}{Q_j} \]

mean service time:
\[ \frac{1}{\mu} = \frac{\sum_{j=1}^{n} \lambda_j (S_j + t_j Q_j)}{\sum_{j=1}^{n} \lambda_j} \]

service - time variance:
\[ \sigma^2 = \frac{\sum_{j=1}^{n} \lambda_j (S_j + t_j Q_j)^2}{\sum_{j=1}^{n} \lambda_j} - \left( \frac{1}{\mu} \right)^2 \]

Production Management 157

Material Requirements Planning
Material Planning

Work to do: 7.7ab, 7.8, 7.10, 7.11, 7.14 (additional information: available hours: 225 (Paint), 130 (Mast), 100 (Rope)), 7.15, 7.16, 7.17, 7.31-7.34

Chapter 8

Operations Scheduling
Operations Scheduling

Scheduling is
- the process of organizing, choosing and timing resource usage to carry out all the activities necessary to produce the desired outputs at the desired times, while satisfying a large number of time and relationship constraints among the activities and the resources (Morton and Pentico, 1993).

Schedule specifies
- the time each job starts and completes on each machine, as well as any additional resources needed.

A Sequence is
- a simple ordering of the jobs.
Operations Scheduling

- Determining a **best** sequence
- 32 jobs on a single machine
- $32!$ Possible sequences approx. $2.6 \times 10^{35}$
  - suppose a computer could examine one billion sequences per second
  - it would take $8.4 \times 10^{15}$ centuries
- real life problems are much more complicated
- Scheduling theory helps to
  - classify the problems
  - identify appropriate measures
  - develop solution procedures

---

Algorithmic complexity

- an efficient algorithm is one whose effort of any problem instance is bounded by a polynomial in the problem size, e.g. # of jobs
- minimal spanning tree can be solved in at most $n^2$ iterations
- $n$: number of edges
- $O(n^2)$

- if effort is exponential $O(2^n)$ the algorithm is not efficient
- branch and bound algorithm for 0/1 variables

- NP-hard problems: no exact algorithm in polynomial time is known, e.g. Traveling salesman problem
- Heuristics are usually polynomial algorithms tailored to the specific problem structure
Operations Scheduling

- **Scheduling Theory (Background)**
- **Jobs are**
  - activities to be done
  - processing time known
  - in general continuously processed until finished (preemption not allowed)
  - due date
  - release date
  - precedence constraints
  - sequence dependent setup time
  - processed by at most one-machine at the same time
Operations Scheduling

- **Machines (resources)**
  - single machine, parallel machines
  - flow shop:
    - each job must be processed by each machine exactly once
    - all jobs have the same routing
    - a job cannot begin processing on the second machine until it has completed processing on the first
  - assembly line
  - job shop:
    - each job may have a unique routing
  - open shops:
    - job shops in which jobs have no specific routing
    - re-manufacturing and repair

- **Measures**
  - profit, costs
  - it is difficult to relate a schedule to profit and cost
  - **regular measure** is a function of completion time
    - function only increases if at least one completion time in schedule increases

- \( n \) = number of jobs to be processed
- \( m \) = number of machines
- \( p_{ik} \) = time to process job \( i \) on machine \( k \)
- \( r_i \) = release date of job \( i \)
- \( d_i \) = due date of job \( i \)
- \( w_i \) = weight of job \( i \) relative to the other jobs
Operations Scheduling

- $C_i = \text{the completion time}$
- $F_i = C_i - r_i, \text{the flowtime}$
- $L_i = C_i - d_i, \text{lateness of job } i$
- $T_i = \max\{0, L_i\}, \text{tardiness of job } i$
- $E_i = \max\{0, -L_i\}, \text{earliness of job } i$
- $\delta_i = 1, \text{if job } i \text{ is tardy (} T_i > 0\text{)}$
- $\delta_i = 0, \text{if job } i \text{ is on time (} T_i = 0\text{)}$

$$C_{\text{max}} = \max_{i=1,n}\{C_i\}, \text{makespan}$$
$$L_{\text{max}} = \max_{i=1,n}\{L_i\}, \text{maximum lateness}$$
$$T_{\text{max}} = \max_{i=1,n}\{T_i\}, \text{maximum tardiness}$$

Operations Scheduling

- **Common proxy objectives**
  - total flowtime
  - total tardiness
  - makespan
  - maximum tardiness
  - number of tardy jobs
  - if not all jobs are equally important weights should be introduced

- minimizing total completion time is equivalent to minimizing total flowtime or minimizing total tardiness
Operations Scheduling

**Algorithms:**
- exact algorithms often based on (worst case scenario) enumeration (e.g. Branch and Bound, Dynamic Programming)
- heuristic algorithm judged by quality (difference to the optimal solution) and efficacy (computational effort)
- worst-case bounds are desirable to motivate use of a certain heuristic

Consider the following four-job, three-machine job-shop scheduling problem:

<table>
<thead>
<tr>
<th>Job</th>
<th>Op.1</th>
<th>Op.2</th>
<th>Op.3</th>
<th>Release Date</th>
<th>Due date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4/1</td>
<td>3/2</td>
<td>2/3</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>1/2</td>
<td>4/1</td>
<td>4/3</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>3/3</td>
<td>2/2</td>
<td>3/1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>3/2</td>
<td>3/3</td>
<td>1/1</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

**Assume the following sequences:**
- 2-1-4-3 on M1
- 2-4-3-1 on M2
- 3-4-2-1 on M3
**Operations Scheduling**

Gantt Chart (machine oriented)

<table>
<thead>
<tr>
<th>M1</th>
<th>2</th>
<th>4</th>
<th>3</th>
<th>1</th>
<th>4</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>M2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>M3</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Operations Scheduling**

\[ C_1 = 14, C_2 = 11, C_3 = 13, C_4 = 10 \]

The makespan is

\[ C_{\text{max}} = \max\{C_1, C_2, C_3, C_4\} = \max\{14, 11, 13, 10\} = 14 \]

The total flowtime is

\[ \sum_i F_i = 14 + 11 + 13 + 10 = 48 \]
Operations Scheduling

The lateness and the tardiness of a job:

- \( L_i = 14 - 16 = -2 \)
- \( T_i = \max \{ 0, -2 \} = 0 \)
- \( L_i = 11 - 14 = -3 \)
- \( T_i = \max \{ 0, -3 \} = 0 \)
- \( L_i = 13 - 10 = 3 \)
- \( T_i = \max \{ 0, 3 \} = 3 \)
- \( L_i = 10 - 8 = 2 \)
- \( T_i = \max \{ 0, 2 \} = 2 \)

The total lateness is:

\[
\sum L_i = (-2) + (-3) + 3 + 2 = 0
\]

The total tardiness is:

\[
\sum T_i = 0 + 0 + 3 + 2 = 5
\]

The maximum tardiness is:

\[
T_{\text{max}} = \max \{ 0, 0, 3, 2 \} = 3
\]

Tardy jobs have \( \delta_i = 1 \), so the number of tardy jobs is:

- \( T_1 = 0 \Rightarrow \delta_1 = 0 \)
- \( T_2 = 0 \Rightarrow \delta_2 = 0 \)
- \( T_3 > 0 \Rightarrow \delta_3 = 1 \)
- \( T_4 > 0 \Rightarrow \delta_4 = 1 \)

Operations Scheduling

- **Single Machine Scheduling**
- **Minimizing Flowtime**

**Problem data**

<table>
<thead>
<tr>
<th>Job i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_i )</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

**Sequence:** 1-2-3-4-5

**Total Flowtime?**

- \( F = p_1 + (p_1 + p_2) + (p_1 + p_2 + p_3) + \ldots + (p_1 + p_2 + \ldots + p_n) \)
- \( F = np_1 + (n-1)p_2 + \ldots + p_n \)
Operations Scheduling

- Theorem. SPT sequencing minimizes total flowtime on a single machine with zero release times.
- Proof. We assume an optimal schedule is not an SPT sequence.

\[ p_i > p_j \]
\[ TF(S) = TF(B) + (t + p_i) + (t + p_i + p_j) + TF(A) \]
\[ TF(S') = TF(B) + (t + p_j) + (t + p_j + p_i) + TF(A) \]
\[ TF(S) - TF(S') = p_i - p_j > 0 \]

SPT-rule \( \Rightarrow \) sequence: 2-4-3-1-5

\[ C_1 = 11 \]
\[ C_2 = 2 \]
\[ C_3 = 7 \]
\[ C_4 = 4 \]
\[ C_5 = 15 \]

Total flowtime = total completion time = 39

- SPT rule also minimizes
  - total waiting time
  - mean # of jobs waiting (mean work in progress)
  - total lateness
- Why?
Operations Scheduling

- Minimize weighted Flow-time: \( \sum_{i=1}^{n} w_i F_i \)

- weighted SPT (WSPT): order ratios \( \frac{p_i}{w_i} \) (nondecreasing)

- exact algorithm for weighted flow-time with zero release time (completion time)

Operations Scheduling

Weighted Flowtime

WSPT scheduling

\( w_1 = 1, w_2 = 4, w_3 = 3, w_4 = 1, w_5 = 3 \)

the processing-time-to-weight ratio gives: 4; 0.5; 1; 2; 1.33

the WSPT sequence is the following: 2-3-5-4-1

\( C_1 = 15 \)
\( C_2 = 2 \)
\( C_3 = 5 \)
\( C_4 = 11 \)
\( C_5 = 9 \)

the value of weighted flowtime is

\[ \sum_{i=1}^{5} w_i F_i = 76 \]
**Operations Scheduling**

- **Maximal Tardiness and Maximal Lateness**
  - due date oriented measure
  - earliest due date sequence (EDD)
  - EDD minimizes
    - Maximal Tardiness and
    - Maximal Lateness

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Due date</td>
<td>16</td>
<td>10</td>
<td>7</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>Proc. Time</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

- EDD-sequence: 5-3-4-2-1
- Tardiness of the jobs is (0, 0, 2, 1, 0)

**Number of Tardy Jobs**

- **Hodgson’s algorithm**
  - **Step 1.** Compute the tardiness for each job in the EDD sequence. Set NT=0, and let k be the first position containing a tardy job. If no job is tardy go to step 4.
  - **Step 2.** Find the job with the largest processing time in positions 1 to k.
    
    \[
    \text{Let } p_{[j]} = \max_{j=1,k} p_{[i]} \quad \text{then } j^* = [j] \]
  
  - **Step 3.** Remove job \( j^* \) from the sequence, set \( N_T = N_T + 1 \), and repeat Step 1.
  - **Step 4.** Place the removed \( N_T \) jobs in any order at the end of the sequence.

**This sequence minimizes the number of tardy jobs**
Operations Scheduling

- Consider the previous example:
  - EDD-sequence: 5-3-4-2-1

  - Step 1: The tardiness is (0, 0, 2, 1, 0) ⇒ Job 4 in the third position is the first tardy job;
  - Step 2: The processing times for jobs 5, 3 and 4 are 4, 3, 2, respectively; ⇒ largest processing time for job 5
  - Step 3: Remove job 5, goto step 1
  - Step 1: EDD-sequence is 3-4-2-1; completion times (3, 5, 7, 11) and tardiness (0, 0, 0, 0) ⇒ Go to step 4
  - Step 4: schedule that minimizes the number of tardy jobs is 3-4-2-1-5 and has only one tardy job: Job 5

Minimize the weighted number of tardy jobs!
NP-hard Problem
Heuristic approach: processing-time-to-weight ratio (not exact!)

Consider the previous example with the following weights:
\[ w_1 = 1, w_2 = 4, w_3 = 3, w_4 = 1, w_5 = 3 \]
- EDD-sequence was 5-3-4-2-1
- Step 1 first tardy job is job 4
- Step 2 the processing-time-weight-ratio for jobs 5, 3 and 4 are 4/3, 3/3 and 2/1
- Step 3 Remove job 4
- Step 1 EDD-sequence is 5-3-2-1 with no tardiness
- Step 4 new schedule 5-3-2-1-4 has one tardy job: job 4 with weight 1
Operations Scheduling

- Minimize Flowtime with no tardy jobs
  - for all jobs to be on time, the last job must be on time
  - schedulable set of jobs contain all jobs with due dates greater than or equal to the sum of all processing times
  - Start from the end and choose the job with the largest proc time among the schedulable jobs, schedule this job last, remove from the list and continue
  - Optimal algorithm! (corresponding alg. For weighted flowtime is only heuristic)

Problem data

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_i</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>due date</td>
<td>16</td>
<td>11</td>
<td>10</td>
<td>9</td>
<td>12</td>
</tr>
</tbody>
</table>

Step 1: Sum of the processing time is 15
  - Job 1 has a due-date greater to 15 => schedule x-x-x-x-1

Step 2: Sum of the remaining processing-times is 11
  - Job 5 has a larger processing time => schedule x-x-5-1

Step 3: remaining processing time is 7
  - All remaining jobs have due dates at least that big
  - => choose the one with the largest processing time => x-x-3-5-1

Step 4: Continue =>2-4-3-5-1
Operations Scheduling

- **Minimizing total Tardiness**
- General single-machine tardiness problem is NP-hard

- **Heuristic approach for the weighted problem (Rachamadugu/Morton)**
- If all jobs are tardy, minimizing weighted tardiness is equivalent to minimizing weighted completion time, which is accomplished by the WSPT sequence.

- Weight-to-processing-time ratio is used

- Slack of job i, \( S_i = d_i - (p_i + t) \) where \( t \) is the current time

Operations Scheduling

- A job should not get full WTPTR “credit” if its slack is positive
  \( S_i^+ = \max\{0, S_i\} \)

- Average processing time of the jobs:
  \( p_{av} = \frac{1}{n} \sum_{i=1}^{n} p_i \)

- Ratio of the slack to the average processing time of jobs:
  \( S_i^+ / p_{av} \)
  which is the number of average job lengths until job j is tardy

- Weight of a job is discounted by an exponential function:  \( \exp(-S_i^+ / \kappa p_{av}) \)
Operations Scheduling

Define the priority of job $i$ by

$$\gamma_i = \left( \frac{w_i}{p_i} \right) e^{-[S_i^+ / \kappa \cdot p_{\text{av}}]}$$

$\kappa$ is a parameter of the heuristic to be chosen by the user (e.g. $\kappa = 2$)

Sequence jobs in descending order of priorities.

Rachamadugu and Morton (1982) R&M Heuristics:

- The owner of Pensacola Boat Construction has currently 10 boats to construct;
- If PBC delivers a boat after the delivery date, a penalty proportional to both the value of the boat and the tardiness must be paid.

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g(1)$</td>
<td>6</td>
<td>12</td>
<td>6</td>
<td>10</td>
<td>3</td>
<td>11</td>
<td>9</td>
<td>11</td>
<td>13</td>
<td>7</td>
</tr>
<tr>
<td>$w(1)$</td>
<td>4</td>
<td>1</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>9</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>$t(1)$</td>
<td>26</td>
<td>20</td>
<td>32</td>
<td>35</td>
<td>30</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>53</td>
<td>54</td>
</tr>
</tbody>
</table>

- How should PBC schedule the work to minimize the penalty paid?
Operations Scheduling

- Penalty is weighted tardiness where weights measure the value of the boat.
- \( \kappa = 2 \)
- Calculate: \( p_{av} = 9 \)

**Job 1:**

\[
\gamma_1 = \left( \frac{W_1}{P_1} \right) e^{-\left( S/(\kappa p_{av}) \right)} = \left( \frac{4}{8} \right) e^{-\left( (26-8)/(2\times9) \right)} = 0,5e^{-1} = 0,18
\]

---

### Table 1: Jobs

<table>
<thead>
<tr>
<th>Jobs</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_i )</td>
<td>6</td>
<td>12</td>
<td>5</td>
<td>10</td>
<td>3</td>
<td>11</td>
<td>9</td>
<td>11</td>
<td>13</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>( w_i )</td>
<td>4</td>
<td>1</td>
<td>6</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>8</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>( d_i )</td>
<td>26</td>
<td>32</td>
<td>36</td>
<td>38</td>
<td>40</td>
<td>50</td>
<td>51</td>
<td>53</td>
<td>64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \gamma_i )</td>
<td>0.6</td>
<td>0.35</td>
<td>0.25</td>
<td>0.35</td>
<td>0.25</td>
<td>0.35</td>
<td>0.25</td>
<td>0.35</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \gamma_i/p_{av} )</td>
<td>0.88</td>
<td>1.44</td>
<td>1.54</td>
<td>1.64</td>
<td>1.74</td>
<td>1.84</td>
<td>1.94</td>
<td>2.04</td>
<td>2.14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>priority</td>
<td>0.18</td>
<td>0.03</td>
<td>0.24</td>
<td>0.17</td>
<td>0.05</td>
<td>0.05</td>
<td>0.06</td>
<td>0.09</td>
<td>0.17</td>
<td>0.01</td>
<td></td>
</tr>
</tbody>
</table>

---

### Table 2: Jobs

<table>
<thead>
<tr>
<th>Jobs</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>gamma_i</td>
<td>0.24</td>
<td>0.18</td>
<td>0.125</td>
<td>0.09</td>
<td>0.07</td>
<td>0.06</td>
<td>0.05</td>
<td>0.047</td>
<td>0.03</td>
</tr>
<tr>
<td>p_i</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>11</td>
<td>13</td>
<td>9</td>
<td>3</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>( C_i )</td>
<td>6</td>
<td>14</td>
<td>24</td>
<td>35</td>
<td>48</td>
<td>57</td>
<td>60</td>
<td>71</td>
<td>83</td>
</tr>
<tr>
<td>( d_i )</td>
<td>32</td>
<td>26</td>
<td>35</td>
<td>61</td>
<td>53</td>
<td>50</td>
<td>48</td>
<td>28</td>
<td>64</td>
</tr>
<tr>
<td>T_i</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>22</td>
<td>23</td>
<td>55</td>
</tr>
<tr>
<td>w_i</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>8</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>w_i/T_i</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>35</td>
<td>22</td>
<td>92</td>
<td>55</td>
</tr>
</tbody>
</table>

---

Production Management 192
Operations Scheduling

Minimizing Earliness and Tardiness with a Common Due-Date

\[ Z = \sum_{i=1}^{n} (E_i + T_i) \]

- this is not a regular measure
- assume common due date: \( d_j = D \)

- Number jobs in LPT sequence: \( p_1 \geq p_2 \geq \cdots \geq p_n \)
- choose \( j^* = n/2 \) or \( n/2+0.5 \)

- if \( p_1 + p_3 + \cdots + p_j \leq D \) then the following sequence is optimal: 1 - 3 - 5 - 7 - \ldots - n - \ldots - 6 - 4 - 2

---

Operations Scheduling

Example: 10 Jobs with common due-date 80

<table>
<thead>
<tr>
<th>Jobs</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>proc Time</td>
<td>8</td>
<td>18</td>
<td>11</td>
<td>4</td>
<td>15</td>
<td>5</td>
<td>23</td>
<td>25</td>
<td>10</td>
<td>17</td>
</tr>
</tbody>
</table>
Operations Scheduling

- if \( p_1 + p_2 + \cdots + p_n > D \) then apply a heuristic (by Sundararaghavan & Ahmed, 1984)

- **Step 0:** Set \( B = D; A = \sum_{i=1}^{n} p_i - D; k = b = 1; a = n; \) use the LPT sequence

- **Step 1:** If \( B > A \):
  assign job \( k \) to position \( b \)
  \( b := b + 1 \)
  \( B := B - p_k \)

  else
  assign job \( k \) to position \( a \)
  \( a := a - 1 \)
  \( A := A - p_k \)

- **Step 2:** \( k := k - 1; \) if \( k \leq n \) go to step 1.

Operations Scheduling

Problems with non-zero release time

- Non-zero release times typically makes scheduling problems much harder, e.g. SPT does in general not minimize total flowtime

- **Heuristic Approach:**
  - At each time t determine the set of **schedulable jobs**: jobs that have been released but not yet processed.
  - Choose from the schedulable jobs according to some rule (e.g. SPT for minimizing flowtime)
Operations Scheduling

* Preemption allowed:

<table>
<thead>
<tr>
<th>j</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>12</td>
<td>2</td>
<td>0</td>
<td>11</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>p</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>t=0</td>
<td>rp</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=2</td>
<td>rp</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=3</td>
<td>rp</td>
<td>4</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=4</td>
<td>rp</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=6</td>
<td>rp</td>
<td>3</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=9</td>
<td>rp</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=10</td>
<td>rp</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=11</td>
<td>rp</td>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=12</td>
<td>rp</td>
<td>8</td>
<td>6</td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=18</td>
<td>rp</td>
<td>8</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=24</td>
<td>rp</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Operations Scheduling

* Minimizing makespan with non-zero release time and tails
* Given n jobs with release times $r_i$, processing times $p_i$, and tails $n_i$

* Schrage Heuristics:
  1. Start at $t=0$
  2. Determine schedulable jobs
  3. If there are schedulable jobs select the job $j^*$ among them with the largest tails, otherwise $t=t+1$ goto 1.
  4. Schedule $j^*$ at $t$
  5. If all jobs have been scheduled stop, otherwise set $t = t + p_{j^*}$, goto 1.
Operations Scheduling

- **Schrage Heuristics Example**: 6 jobs with release times and tails

<table>
<thead>
<tr>
<th>j</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_j</td>
<td>12</td>
<td>2</td>
<td>0</td>
<td>11</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>p_j</td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>n_j</td>
<td>21</td>
<td>9</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>

- **Minimize makespan!**

Operations Scheduling

- Denote by SJ the set of schedulable jobs and by S the scheduled sequence

  - Step 1. t = 0, SJ = {3}, S = <3>, t = 3, C_{max} = 5
  - Step 2. t=3, SJ = {2}, S = <3-2>, t = 7, C_{ma} = 16
  - Step 3. t = 9, SJ = {5}, S = <3-2-5>, t = 11, C_{ma} = 18
  - Step 4. t=11, SJ = {4, 6}, S = <3-2-5-6>, t = 13, C_{ma} = 23
  - Step 5. t=13, SJ = {1, 4}, S = <3-2-5-6-1>, t = 21, C_{ma} = 42
  - Step 6. T=21 SJ = {4}, S = <3-2-5-6-1-4>, t = 27, C_{ma} = 42

- **Schrage heuristic is in general not optimal, e.g. B&B model can be used as an exact algorithm**
Operations Scheduling

Minimizing Set-Up Times

- sequence-dependent set-up times
- the time to change from one product to another may be significant and may depend on the previous part produced
- \( p_{ij} \) = time to process job \( j \) if it immediately follows job \( i \)

Examples:
- electronics industry
- paint shops
- injection molding

Minimizes makespan
- problem is equivalent to the traveling salesman problem (TSP), which is NP-hard.

Examples:
- electronics industry
- paint shops
- injection molding

minimizes makespan
- problem is equivalent to the traveling salesman problem (TSP), which is NP-hard.

Production Management 201

---

**SST(=shortest set-up time) heuristic**

A metal products manufacturer has contracted to ship metal braces each day for four customers. Each brace requires a different set-up on the rolling mill:

<table>
<thead>
<tr>
<th>Rolling mill set-up times</th>
<th>Job</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>( \infty )</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>( \infty )</td>
<td>4</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>6</td>
<td>( \infty )</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>4</td>
<td>( \infty )</td>
<td>( \infty )</td>
<td>( \infty )</td>
</tr>
</tbody>
</table>

*Job C cannot follow job D, because of quality problems

SST-heuristic:

**Step 1** starting arbitrarily by choosing one Job: A

**Step 2** B has the smallest set-up time following A; \( \Rightarrow \) A-B

**Step 3** C has the smallest set-up time of all the remaining jobs following B; \( \Rightarrow \) A-B-C

**Step 4** D is the last remaining job; \( \Rightarrow \) A-B-C-D-A with a makespan of \( 3 + 4 + 2 + 5 = 14 \)

Production Management 202
Operations Scheduling

- A regret based Algorithm
  - makespan must be at least as big as the \( n \) smallest elements
  - reduced matrix
    - row reduction
    - column reduction
    - sum of reduced costs = lower bound for TSP
  - find reduced matrix!

<table>
<thead>
<tr>
<th>Job</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>( \infty )</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>( \infty )</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>6</td>
<td>( \infty )</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>5</td>
<td>4</td>
<td>( \infty^* )</td>
<td>( \infty )</td>
</tr>
</tbody>
</table>

Operations Scheduling

- The reduced matrix has a zero in every row and column
- what happens if we do not choose \( j \) to follow \( i \)
- regret: lower bound on not choosing \( j \) to follow \( i \)

<table>
<thead>
<tr>
<th>Job</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>( \infty )</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>( \infty )</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>5</td>
<td>( \infty )</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>0</td>
<td>( \infty^* )</td>
<td>( \infty )</td>
</tr>
</tbody>
</table>
Operations Scheduling

Regret heuristic
Find the cycle sequence that minimizes the set-up time.

Set-up times

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>18</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>19</td>
<td>∞</td>
<td>9</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>18</td>
<td>∞</td>
<td>13</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>6</td>
<td>1</td>
<td>∞</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>17</td>
<td>1</td>
<td>13</td>
<td>∞</td>
<td>∞</td>
</tr>
</tbody>
</table>

Solution: TSP model – regret heuristic

Step 0 \( C(\text{max}) = 0 \) and \( L = 1 \)

Step 1 Reduce the matrix:

Reduced matrix

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Min</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>15</td>
<td>0</td>
<td>6</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>∞</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>9</td>
<td>∞</td>
<td>4</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>∞</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>0</td>
<td>12</td>
<td>16</td>
<td>∞</td>
<td>1</td>
</tr>
</tbody>
</table>

Step 2 Calculate the regret

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Min</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>15</td>
<td>0</td>
<td>6</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>∞</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>9</td>
<td>∞</td>
<td>4</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>∞</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>0</td>
<td>12</td>
<td>16</td>
<td>∞</td>
<td>1</td>
</tr>
</tbody>
</table>

Step 3 Choose the largest regret: 17

Step 4 Assign a job pair: Job 2 immediately follows job 5 (5-2)
\( L = 1+1; \)
We prohibit 2-5
Operations Scheduling

Step 1 Reduce the matrix

\[ C_{\text{max}} = 19 + 4 + 1 = 24 \]

Reduced Matrix

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>∞</td>
<td>1</td>
<td>∞</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>4</td>
<td>10</td>
<td>∞</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
</tbody>
</table>

Step 2 Calculate the regret

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>0(0)</td>
<td>0(1)</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>1</td>
<td>∞</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0(9)</td>
<td>4</td>
<td>10</td>
<td>∞</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>0(0)</td>
<td>0(2)</td>
<td>∞</td>
</tr>
</tbody>
</table>

Step 3 Choose the largest regret: 9

Step 4 Assign a job pair: 3-1

Prohibit 1-3

Step 1 Reduce the matrix: not possible

Matrix

<table>
<thead>
<tr>
<th>Job</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>∞</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>∞</td>
<td>0</td>
</tr>
</tbody>
</table>

Step 2 Calculate regret

<table>
<thead>
<tr>
<th>Job</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0(3)</td>
<td>0(1)</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>0(1)</td>
<td>1</td>
<td>∞</td>
</tr>
<tr>
<td>3</td>
<td>0(9)</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>0(0)</td>
<td>∞</td>
<td>0(2)</td>
</tr>
</tbody>
</table>

Step 3 Choose the largest regret: 3

Step 4 Assign job pair: 1-4; partial sequence: 5-2, 3-1-4

Prohibit 4-1 and 4-3 (to keep 3-1-4-3 from being chosen)

Final Matrix

<table>
<thead>
<tr>
<th>Job</th>
<th>3</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>4</td>
<td>∞</td>
<td>0</td>
</tr>
</tbody>
</table>

choose 2-3 and 4-5

-> sequence 3-1-4-5-2

the total set-up time is 24
**Operations Scheduling**

**Branch and Bound Algorithm**

1. Using the regret heuristic construct a (sub-)tree where each node represents the decision to let \( j \) follow \( i \) (\( j \rightarrow i \)) or to prohibit that \( j \) follows \( i \) (\( i \rightarrow j \)).

2. For each node a lower bound for the makespan is inferred from the regret heuristic.

3. Once a solution is obtained from the regret heuristic this is an upper bound for the optimal makespan. All nodes where the lower bound is above that level are pruned.

4. If all but one final node are pruned (or no non-pruned node can be further branched) this final node gives the optimal solution.

5. If 4. does not hold start again with 1. at one of nodes which are not pruned and can still be branched.

---

**Operations Scheduling**

**Branch and Bound Algorithm**

- All final nodes can be pruned: \( \text{opt. Solution has been found!} \)
Operations Scheduling

- **Single-Machine Search Methods**
  - Neighborhood Search
  - Simulated Annealing
  - Ant System
  - Tabu Search
  - ...

- **Neighborhood Search**
  - seed
  - Neighborhood
  - any heuristic can be used to produce an initial sequence

- adjacent pairwise interchange (API):
  - n-1 neighbors
  - 1-2-3-4-5-6-7-8-9
  - 1-2-3-4-6-5-7-8-9

- Pairwise interchange (PI):
  - n(n-1)/2 neighbors
  - 1-2-3-4-6-7-8-9

- Insertion (INS)
  - (n-1)^2 neighbors
  - 1-2-3-4-5-6-8-9

- Evaluation function
- Update function
Consider the following single-machine tardiness problem; Use the EDD sequence as the initial seed with an API neighborhood;

Data for neighborhood search

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing time</td>
<td>10</td>
<td>3</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>Due-date</td>
<td>15</td>
<td>16</td>
<td>24</td>
<td>30</td>
<td>35</td>
<td>37</td>
</tr>
</tbody>
</table>

Step 1: Construct the EDD sequence and evaluate its total tardiness. Set $i = 1$ and $j = 2$.

- The EDD sequence $S^*$: 1-2-3-4-5-6; tardiness-vector $\langle 0, 0, 5, 7, 6, 14 \rangle$

Step 2: Swap the jobs in the $i$-th and $j$-th position in $S^*$; the sequence is $S'$ with tardiness $T'$. If $T' < T$, go to step 4.

Step 3: $j = j + 1$: If $j > n$: go to step 5. Otherwise, $i = j - 1$ and go to step 2.

Step 4: Replace $S^*$ with $S'$; $i = 1, j = 2$; go to step 2.

Step 5: Stop; $S^*$ is a local optimal sequence.
### Operations Scheduling

#### Neighborhood search solution

<table>
<thead>
<tr>
<th>Jobs</th>
<th>Schedule</th>
<th>Tardiness</th>
</tr>
</thead>
<tbody>
<tr>
<td>i j</td>
<td>1 2 3 4 5 6</td>
<td>32 42 33 30 40 34</td>
</tr>
</tbody>
</table>

- 1 2 2 1 3 5 4 6
- 2 3 1 3 2 4 5 6
- 3 4 1 2 4 3 5 6
- 4 5 1 2 3 5 4 6

- 1 2 2 1 3 5 4 6
- 2 3 1 3 2 4 5 6
- 3 5 1 2 5 3 4 6
- 5 4 1 2 3 4 5 6
- 4 6 1 2 3 5 6 4

### Operations Scheduling

#### Single machine results

- Flowtime - SPT (E)
- Lateness - SPT (E)
- Weighted Flowtime - WSPT (E)
- Maximal Tardiness (Lateness) - EDD (E)
- Nb. Of tardy jobs - Hodgson (E)
- weighted nb. Of tardy jobs - modified Hodgson (H)
- No jobs tardy/flowtime - modified SPT (E)
- Tardiness - R&M (H)
- weighted Tardiness - R&M (H)
- makespan with non-zero release time and tails - Schrage (H)
- Sequence dependent - SST (H), regret (H), B&B (E)
Operations Scheduling

### Parallel Machines

- **Scheduling decisions:**
  - which machine processes the job
  - in what order

- **List Schedule**
  - to create a schedule, assign the job on the list to the machine with the smallest amount of work assigned.

  **Step 0.** Let $H_i=0, i=1,2,...,m$ be the assigned workload on machine $i$, $L=(\{1\},\{2\},...,\{n\})$ the ordered list sequence, $C_j=0, j=1,2,...,n$, and $k=1$

  **Step 1.** Let $j^*=L_k$ and $H_{i^*}=\min_{i=1,m}\{H_i\}$; Assign job $j^*$ to be processed on machine $i^*$, $C_{j^*}=H_{i^*}+p_{j^*}, H_{i^*}=H_{i^*}+p_{j^*}$

  **Step 2.** Set $k=k+1$, if $k>n$, stop. Otherwise go to step 1.

---

Minimizing flowtime on parallel processors

Consider a facility with 3 identical machines and 15 jobs that need to be done as soon as possible;

Processing times(after SPT):

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>16</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
</tbody>
</table>

**Optimal schedule:**

<table>
<thead>
<tr>
<th>Machine 1</th>
<th>Machine 2</th>
<th>Machine 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td>p(j)</td>
<td>C(j)</td>
</tr>
<tr>
<td>----</td>
<td>-----</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>17</td>
</tr>
<tr>
<td>10</td>
<td>13</td>
<td>30</td>
</tr>
<tr>
<td>13</td>
<td>16</td>
<td>46</td>
</tr>
</tbody>
</table>

Total flowtime = 372
Operations Scheduling

Minimize the makespan

Use a longest processing time (LPT) first list;
Assign the next job on the list to the machine with the least total processing time assigned.

Optimal schedule:

<table>
<thead>
<tr>
<th></th>
<th>Machine 1</th>
<th></th>
<th>Machine 2</th>
<th></th>
<th>Machine 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td>p(j)</td>
<td>C(j)</td>
<td>j</td>
<td>p(j)</td>
<td>C(j)</td>
</tr>
<tr>
<td>15</td>
<td>19</td>
<td>19</td>
<td>14</td>
<td>18</td>
<td>18</td>
</tr>
<tr>
<td>10</td>
<td>13</td>
<td>32</td>
<td>11</td>
<td>13</td>
<td>31</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>42</td>
<td>8</td>
<td>11</td>
<td>42</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>52</td>
<td>5</td>
<td>9</td>
<td>51</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>53</td>
<td>2</td>
<td>3</td>
<td>54</td>
</tr>
</tbody>
</table>

Optimal makespan: 52

Operations Scheduling

**Flow shops**
- all jobs are processed in the same order
- two machine makespan model: Johnson’s Algorithm
- Bound on makespan:
  \[
  C^*_{\text{max}} \geq \max \left( \min p_{i2} + \sum_{j=1}^{n} p_{j1}, \min p_{i1} + \sum_{j=2}^{n} p_{j2} \right)
  \]
- Formulate Johnson’s Algorithm
- For 2-machine Flow shops the optimal schedule is a Permutation Schedule, i.e. the job sequence is the same on every machine
Operations Scheduling

- **Makespan with more than two machines**
  - **Johnson’s algorithm will work in special cases, e.g. three machine problem where the second machine is dominated:**
    \[
    p_{ij} \leq \max(\min p_{i1}, \min p_{i3})
    \]
  - Formulate an artificial two machine problem with
    \[
    p'_{i1} = p_{i1} + p_{i2} \quad \text{and} \quad p'_{i2} = p_{i2} + p_{i3}
    \]
    and solve it using the Johnson algorithm gives the optimal solution for the three machine problem.

- **Heuristics for the m-machine problem**
  - **Cambell, Dudek and Smith (1970)**
  - convert a m-machine problem into a two machine problem
  - how?
    \[
    p'_{ik} = \sum_{j=1}^{k} p_{ij} \quad \text{and} \quad p'_{il} = \sum_{j=1}^{l} p_{ij}
    \]
  - Start with: k=1 and l=m; then k=2 and l=m-1; until: k=m-1 and l=2
  - m-1 schedules are generated
  - Use the best of these m-1 schedules
Operations Scheduling

Flow-shop heuristics

Processing data:

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>1</td>
<td>10</td>
<td>17</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>M2</td>
<td>13</td>
<td>12</td>
<td>9</td>
<td>17</td>
<td>3</td>
</tr>
<tr>
<td>M3</td>
<td>6</td>
<td>18</td>
<td>13</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>M4</td>
<td>2</td>
<td>18</td>
<td>4</td>
<td>6</td>
<td>16</td>
</tr>
</tbody>
</table>

Use the CDS heuristic to solve this problem.

(1) i.) Use the Johnson’s algorithm only for M1 and M4:

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>1</td>
<td>10</td>
<td>17</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>M4</td>
<td>2</td>
<td>18</td>
<td>4</td>
<td>6</td>
<td>16</td>
</tr>
</tbody>
</table>

[j] 1-2-5-4-3, C_{max} = 88

Next combine M1 and M2 to pseudomachine 1 and M3 and M4 to pseudomachine 2.

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM1</td>
<td>14</td>
<td>22</td>
<td>26</td>
<td>29</td>
<td>14</td>
</tr>
<tr>
<td>PM2</td>
<td>8</td>
<td>36</td>
<td>17</td>
<td>8</td>
<td>21</td>
</tr>
</tbody>
</table>

[j] 5-2-3-1-4, C_{max} = 85

Finally combine M1, M2 and M3 to pseudomachine 1 and M2, M3 and M4 to pseudomachine 2.

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM1</td>
<td>20</td>
<td>40</td>
<td>39</td>
<td>31</td>
<td>19</td>
</tr>
<tr>
<td>PM2</td>
<td>21</td>
<td>48</td>
<td>26</td>
<td>25</td>
<td>24</td>
</tr>
</tbody>
</table>

[j] 5-1-2-3-4, C_{max} = 85

Operations Scheduling

Gantt Chart for the CDS schedule

![Gantt Chart](image)
Gupta - Heuristic

- Gupta (1972)
- Exact for 2-machine problem and 3-machine problem, where the 2nd machine is dominated

\[ e_i = \begin{cases} 
1 & \text{if } p_{ti} < p_{tm} \\
-1 & \text{if } p_{ti} \geq p_{tm}
\end{cases} \]

\[ s_i = \min_{k=1,...,n-1} \left( \frac{e_i}{p_{ti} + p_{ti+1}} \right) \]

- Sorting jobs with nonincreasing \( s_i \)

\((s_{i[1]} \geq s_{i[2]} \geq ... \geq s_{i[n]})\)

\[
\begin{array}{cccccccc}
\text{Job} & p_1+p_2 & p_2+p_3 & p_3+p_4 & \text{min} & e_i & s_i & [l] \\
1 & 14 & 19 & 8 & 8 & 1 & 0.12 & 1 \\
2 & 22 & 30 & 36 & 22 & 1 & 0.05 & 3 \\
3 & 26 & 22 & 17 & 17 & -1 & -0.06 & 4 \\
4 & 29 & 19 & 8 & 8 & -1 & -0.12 & 5 \\
5 & 14 & 8 & 21 & 8 & 1 & 0.12 & 2 \\
\end{array}
\]

Branch and Bound Approaches

- Machine based bounds
- Job based bounds
- Three machines

- Machine 1:

\[ C_{\text{max}}^* \geq H_1 + \sum_{i \in U} p_i + \min_{i \in U} \{p_{i2} + p_{i3}\} \]

- Machine 2:

\[ C_{\text{max}}^* \geq \max \left( \left( H_1 + \min_{i \in U} \{p_i\} \right) \cdot H_2 \right) + \sum_{i \in U} p_{i2} + \min_{i \in U} \{p_{i3}\} \]
**Operations Scheduling**

Machine 3:

\[ C^*_\text{max} \geq \max \left\{ H_1 + \min \{p_{i1} + p_{i2}\}, H_2 + \min \{p_{j1} + p_{j2}\}, H_3 \right\} + \sum p_{i3} \]

Job oriented bounds:

\[ C_{\text{max}} \geq H_1 + \max \left\{ \sum_{j=1}^{m} p_{ij} + \sum_{k \in U, k \neq i} \min \{p_{k1}, p_{k3}\} \right\} \]

---

**Operations Scheduling**

**B&B algorithm for minimizing makespan in multi-machine Flow Shops**

1. Create an initial incumbent solution, e.g. CDS heuristic upper bound
2. Starting at t=0 with a root node; branch the tree by generating a node for each schedulable jobs.
3. In each node calculate the lower bounds and prune the node if at least one exceeds the upper bound.
4. If a non-pruned final node exists at the lowest level take the corresponding solution as new incumbent, update the upper bound and do the corresponding pruning.
5. If all final nodes are pruned current incumbent is the optimal solution, otherwise branch at the node with the lowest lower bound and goto 3.
Operations Scheduling

Makespan permutation schedule for a three-machine flow-shop

Processing data:

<table>
<thead>
<tr>
<th>Machine</th>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>11</td>
<td>10</td>
<td>17</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>13</td>
<td>12</td>
<td>9</td>
<td>17</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>6</td>
<td>18</td>
<td>13</td>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

Solution:

Start with CDS algorithm: sequence: 1-2-3-4-5, C_max = 65

Initial lower bound:

M1: \( C_{\text{max}}^* \geq H_1 + (p_{11} + p_{12} + p_{13}) + \min\{ p_{14}, p_{15}, p_{24}, p_{25}, p_{34}, p_{35} \} \)

\[ = 0 + (1 + 10 + 17 + 12 + 11) + \min\{19, 30, 22, 19, 8\} = 51 + 8 = 59 \]

M2: \( C_{\text{max}}^* \geq \max\{H_1 + \min\{p_{11} + p_{21} + p_{31} + p_{41} + p_{51}\}, H_2\} \)

\[ + \{ p_{12} + p_{22} + p_{32} + p_{42} + p_{52} \} + \min\{ p_{13}, p_{23}, p_{33}, p_{43}, p_{53} \} \]

\[ = \max\{0 + \min\{1, 10, 17, 12, 11\}, 0\} \]

\[ + (13 + 12 + 9 + 17 + 3) + \min\{6, 18, 13, 2, 5\} \]

\[ = 1 + 54 + 2 = 57 \]

Job-based bounds are the following:

J1: \( C_{\text{max}}^* \geq H_1 + (p_{11} + p_{12} + p_{13}) + \min\{ p_{14}, p_{15}, p_{24}, p_{25}, p_{34}, p_{35}\} \)

\[ = 0 + (1 + 13 + 6) + (\min\{10, 18\} + \min\{17, 13\} + \min\{12, 5\} + \min\{11, 5\}) \]

\[ = 0 + 20 + (10 + 13 + 2 + 5) = 50 \]

Similarly, we have

J2: \( C_{\text{max}}^* \geq 61, J3: C_{\text{max}}^* \geq 57, J4: C_{\text{max}}^* \geq 60, J5: C_{\text{max}}^* \geq 45 \)

LB: 61, UB: 65
Operations Scheduling

1st level: Job 2 at first place: $H_1 = 10$, $H_2 = 22$, $H_3 = 40$

- $U = \{1, 3, 4, 5\}$
- $M_1: C_{\text{max}}^* \geq 59$
- $M_2: C_{\text{max}}^* \geq 66$, which is greater than the upper bound; thus we fathom the node;
- $J_3$, $J_4$ and $J_5$ at first place: we can fathom all of them;

2nd level: Consider Job 3: $H_1 = 18$, $H_2 = 27$, $H_3 = 40$, $U = \{2, 4, 5\}$

- $M_1: C_{\text{max}}^* \geq 59$
- $M_2: C_{\text{max}}^* \geq 61$
- $M_3: C_{\text{max}}^* \geq 65$, so we fathom the job; only job 2 remains unfathomed;

3rd level: Job 3: $H_1 = 28$, $H_2 = 37$, $H_3 = 57$, $U = \{4, 5\}$

- $M_1: C_{\text{max}}^* \geq 59$
- $M_2: C_{\text{max}}^* \geq 61$
- $M_3: C_{\text{max}}^* \geq 64$

Machine-bounds did not fathom the node; so we have to calculate job-based bounds:

- $J_4: C_{\text{max}}^* \geq 64$
- $J_5: C_{\text{max}}^* \geq 49$

$\Rightarrow$ best bound = 64; thus create nodes for $J_4$ and $J_5$

4th level: nodes $J_4$ and $J_5$ of level 3 will be fathomed; thus the algorithm is complete:

1-2-3-4-5 with a makespan of 65;
Operations Scheduling

- **Job Shops**
  - different routings for different jobs
  - precedence constraints
  - \((n!)^m\) possible schedules

Two machine job shops

- **Jackson (1956)**
- minimize makespan
  - Machine A: \{(AB), \{A\}, \{BA\}\}
  - Machine B: \{(BA), \{B\}, \{A,B\}\}

**Jackson’s algorithm**

<table>
<thead>
<tr>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Route/</td>
<td>BA</td>
<td>AB</td>
<td>BA</td>
<td>B</td>
<td>A</td>
<td>AB</td>
<td>B</td>
<td>BA</td>
<td>BA</td>
<td>AB</td>
</tr>
<tr>
<td>p(i)1</td>
<td>3</td>
<td>1</td>
<td>11</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>0</td>
<td>8</td>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td>p(i)2</td>
<td>8</td>
<td>10</td>
<td>13</td>
<td>1</td>
<td>0</td>
<td>8</td>
<td>6</td>
<td>10</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

Find a schedule that would finish all jobs as soon as possible!

Solution:
\(\{A\} = \{5\}, \{B\} = \{4,7\}, \{AB\} = \{2, 6, 10\}\) and \(\{BA\} = \{1, 3, 8, 9\}\)
### Operations Scheduling

**Johnson's algorithm for (AB):**

<table>
<thead>
<tr>
<th>Job</th>
<th>2</th>
<th>10</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>p(i)1</td>
<td>1</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>p(i)2</td>
<td>10</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

**Johnson's algorithm (reversed) for (BA):**

<table>
<thead>
<tr>
<th>Job</th>
<th>9</th>
<th>3</th>
<th>8</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>p(i)1</td>
<td>13</td>
<td>11</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>p(i)2</td>
<td>6</td>
<td>13</td>
<td>10</td>
<td>8</td>
</tr>
</tbody>
</table>

Sequence for A: 2-10-6-5-9-3-8-1
Sequence for B: 9-3-8-1-4-7-2-10-6

Makespan: 67

---

**Dispatching**

- **job shop scheduling**
- **dispatching rules**

**Basic idea:**
- Schedule an operation of a job as soon as possible.
- If more than one job is waiting to be processed by the same machine, schedule the one with best priority.

**Define:**
- \( A \) = set of idle machines
- \( J_k \) = the index of the last job scheduled on machine \( k \)
- \( U_k \) = the set of jobs that can be processed on machine \( k \)
- \( H_k \) = the completion time of the job currently processed on machine \( k \)
- \( u_i \) = the priority of job \( i \) at time \( t \)
Operations Scheduling

Step 0. Initialize: $t=0; \ H_k=0; k=1,2,...,m$; A={1,2,...,m}; $U_k=\{i|\text{operation 1 of } i \text{ is on machine } k, \ i=1,2,...,n\}; s_i=c_i=0$. Go to step 4.

Step 1. Increment t;

Let $t = \min_{k=1,m, k \in A} H_k$, and $K = \{k \mid H_k = t\}$

Step 2. Find the job or jobs that complete at time t and the machine released. Set $A = A \setminus K$.

Step 3. Determine the jobs ready to be scheduled on each machine; Let $U_k=\{i|\text{job } i \text{ uses machine } k \text{ and all operations of job } i \text{ before machine } k \text{ are completed}\}, \ k=1,2,...,m$. If U_k=0 for k=1,2,...,m, Stop. If U_k=0 for k\in A, go to Step 1.

Step 4. For each idle machine try to schedule a job; for each $k \in A$ with $U_k=0$,

let $i$ be the job with the best priority $u_{i_{opt}} = \min_{i \in U_k} u_i$.

Schedule job i* on machine k

Set $J_k = i$, $s_{ik} = t$, $c_{ik} = t + p_{i_{opt}k}$, $H_k = c_{ik}$

Remove the scheduled job from $U_k$

$U_k \leftarrow U_k \setminus \{i\}$

and the machine from A

$A \leftarrow A \setminus \{k\}$

Go to Step 1.
Operations Scheduling

Many priority measures possible:

- SPT
- FCFS
- MWKR (most work remaining)
- EDD
- EDD/OP
- SLACK, SLACK/OP
- Critical ratio: slack/remaining time
- ...

Quick Closures: job-shop dispatch heuristic

Quick Closure has four machines in the shop: (1) brake, (2) emboss, (3) drill, (4) mill. The shop has currently orders for six different parts, which use all the four machines, but in a different order.

Processing time:

<table>
<thead>
<tr>
<th>Operation</th>
<th>Job</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6/1</td>
<td>8/2</td>
<td>13/3</td>
<td>5/4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4/1</td>
<td>1/2</td>
<td>4/3</td>
<td>3/4</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3/4</td>
<td>8/2</td>
<td>6/1</td>
<td>1/3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5/2</td>
<td>10/1</td>
<td>15/3</td>
<td>4/4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3/1</td>
<td>4/2</td>
<td>6/4</td>
<td>4/3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>4/3</td>
<td>2/1</td>
<td>4/2</td>
<td>5/4</td>
<td></td>
</tr>
</tbody>
</table>

Finish all six parts as soon as possible!

Solution: We use a dispatch procedure with MWKR as the priority.
Operations Scheduling

Step 1: $t = 0, H_1 = H_2 = H_3 = H_4 = 0, A = \{1, 2, 3, 4\}, U_1 = \{1, 2, 5\}, U_2 = \{4\}, U_3 = \{6\}, U_4 = \{3\}; s_{ij} = c_{ij} = 0, i = 1, 2, 3, 4, 5, 6; \text{ and } j = 1, 2, 3, 4; \text{ Go to step 4.}

Step 4: $u_{10} = -(6+8+13+5) = -32, u_{20} = -12, u_{50} = -17; \text{ thus } s_{11} = 0, c_{11} = 0 + 6 = 6, H_1 = 6.

Remove job 1 from $U_1$, $U_1 = \{2, 5\}$ and machine 1 from $A$, $A = \{2, 3, 4\}.

Set $k = 2$; there is only one job in $U_2$ so we schedule it on machine 2; $i^* = 4$, $s_{41} = 0, c_{41} = 5, H_2 = 5, U_2 = \{\}$, and $A = \{3, 4\}.$

We schedule J6 and J3 on M3 and M4 (tab: $t = 0$ row). Go to step 1.

Step 1: $t = \min_{a_i} \min_{H_k} H_k = \min\{6, 5, 4, 3\} = 3$, and $K = \{k \mid H_k = 3\} = \{4\}; H_k \min$ is bold in the table;

Step 2: $J3$ completes at time 3 on $M4$, so $i^3 = \{i \mid J_k = i, k \in K\} = \{3\}, K = \{4\},$ and $A = \emptyset, U(4) = \{4\}.$ (tab: $t = 3$ row)

Step 3: $U1 = \{2, 5\}, U2 = \{3\}, U3 = U4 = \{\}; \text{ Since no jobs are waiting for M4, no jobs can be}$ scheduled to start at time 3; go to step 1 etc.
Chapter 10

Section 5.5: Bottleneck Scheduling

*Operations Scheduling*

<table>
<thead>
<tr>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Time: 0 10 20 30 40 50
Shifting Bottleneck Procedure

heuristic to minimize makespan for multiple machine job shops

Main idea:
1. for each job on each machine calculate the minimal amount of time needed before and after the processing of this job generates minimal makespan problem with release times and tails
2. for each machine solve this problem for each machine (e.g. Schrage heuristic) and determine the machine with the maximal makespan (bottleneck machine)
3. Fix the found sequence on the bottleneck machine, update release times and tails on the remaining machines and repeat 2. for the remaining machines until schedules for all machines have been determined

Shifting Bottleneck Procedure Example:
3 machines (M1, M2, M3), 3 jobs (1,2,3)

Job routings: 1: M1-M2-M3
2: M2-M3-M1
3: M2-M1-M3

Processing times:

<table>
<thead>
<tr>
<th>p_{ik}</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>
Operations Scheduling

Machine-Flow-Graph:

Problems with release times and tails for each machine:

M1:  
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_j</td>
<td>0</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>p_j</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>n_j</td>
<td>5</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

M2:  
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_j</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>p_j</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>n_j</td>
<td>2</td>
<td>6</td>
<td>4</td>
</tr>
</tbody>
</table>

M3:  
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>r_j</td>
<td>6</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>p_j</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>n_j</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>
Operations Scheduling

- Schrage heuristic gives the following solutions for the three machines:

- Machine 2 is bottleneck with $C_2 = 11$
- Fix sequence on machine 2

Operations Scheduling

- Machine-flow-graph:

- Update release times and tails on M1 and M3:

- M1:
  - $r_i$: 0, 5, 6
  - $p_i$: 3, 3, 3
  - $n_i$: 5, 0, 1

- M3:
  - $r_i$: 9, 2, 9
  - $p_i$: 2, 3, 1
  - $n_i$: 0, 3, 0
Operations Scheduling

Schrage heuristic for M1, M3:

both machines could be considered the bottleneck with $C=12$, fix sequence on M1

Updated machine-flow-graph:

update release time and tails and apply Schrage to M3. This gives $C_{\text{max}}=12$
Operations Scheduling

**Finite Capacity Scheduling**
- MRP systems generally assume constant lead times, ignore setups
- MRP plans might be unrealistic
- Traditionally hidden by inventory and excess capacity
- Reducing inventory and capacity makes finite capacity scheduling crucial
- Computer-assisted finite capacity scheduling systems rather than manual scheduling by foreman

**Work to do:** 8.3abcde, 8.4, 8.5, 8.6, 8.10, 8.14, 8.16, 8.18 (with the following due dates: 42, 50, 12, 63, 34, 36, 42, 54, 32)
  8.30ab, 8.32abc, 8.36ab, 8.43, 8.44, 8.49ab, 8.51ab, 8.56, 8.57 (apply shifting bottleneck procedure)

**Minicase: Ilana Designs**