Group Technology

END4650 – Material Handling Systems

Mehmet Güray Güler

Industrial Engineering Department

Yıldız Technical University

Group Technology

- Group technology (GT) is a manufacturing technique and philosophy to increase production efficiency by exploiting the “underlying sameness” of component shape, dimensions, process route, etc.
- GT is the realization that many problems are similar, and that by grouping similar problems, a single solution can be found to a set of problems thus saving time and effort. (Solaja 73)

Group Technology

REDUCTIONS

- Setup time
- Material handling cost
- Direct and indirect labor cost
- Throughput time
- Overdue orders
- Production floor space
- Raw material stocks
- In-process inventory

REDUCTIONS

- Capital expenditures
- Tooling costs
- Engineering time and costs
- New parts design
- New shop drawings
- Total number of drawings

Group Technology

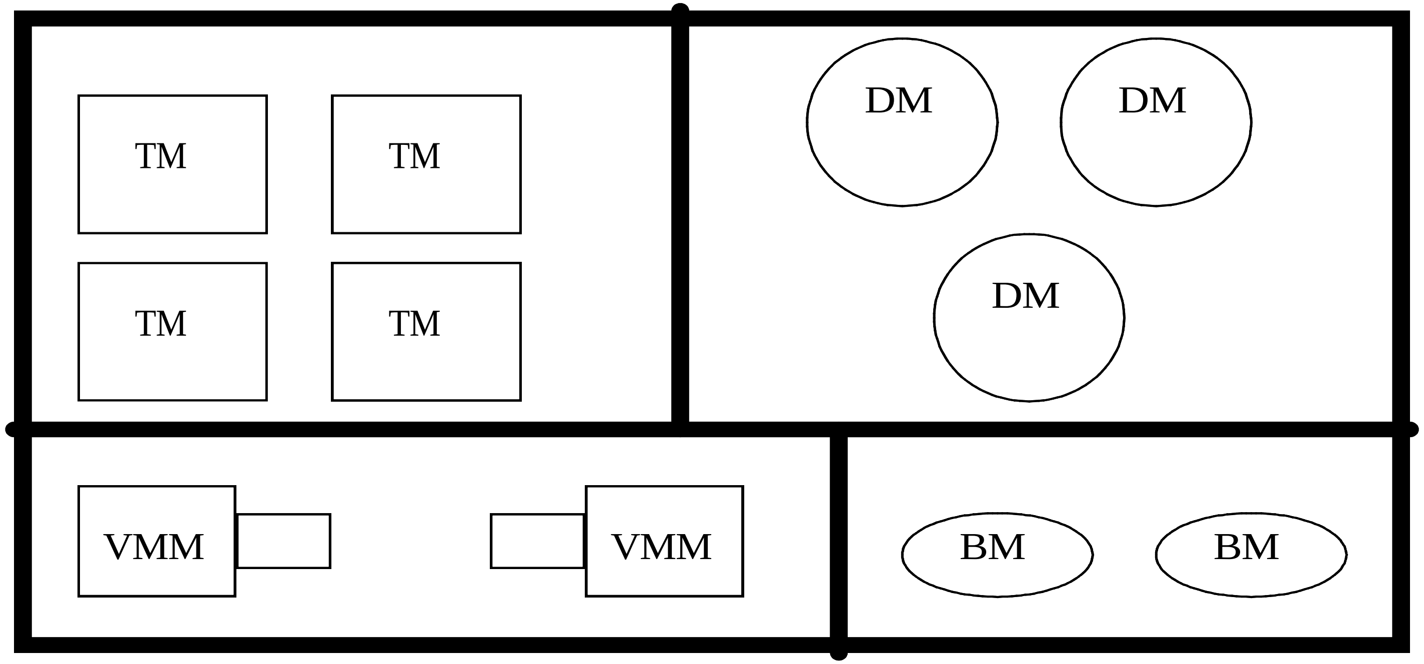
IMPROVEMENTS

- Quality
- Material Flow
- Machine and operator utilization
- Space Utilization
- Employee Morale

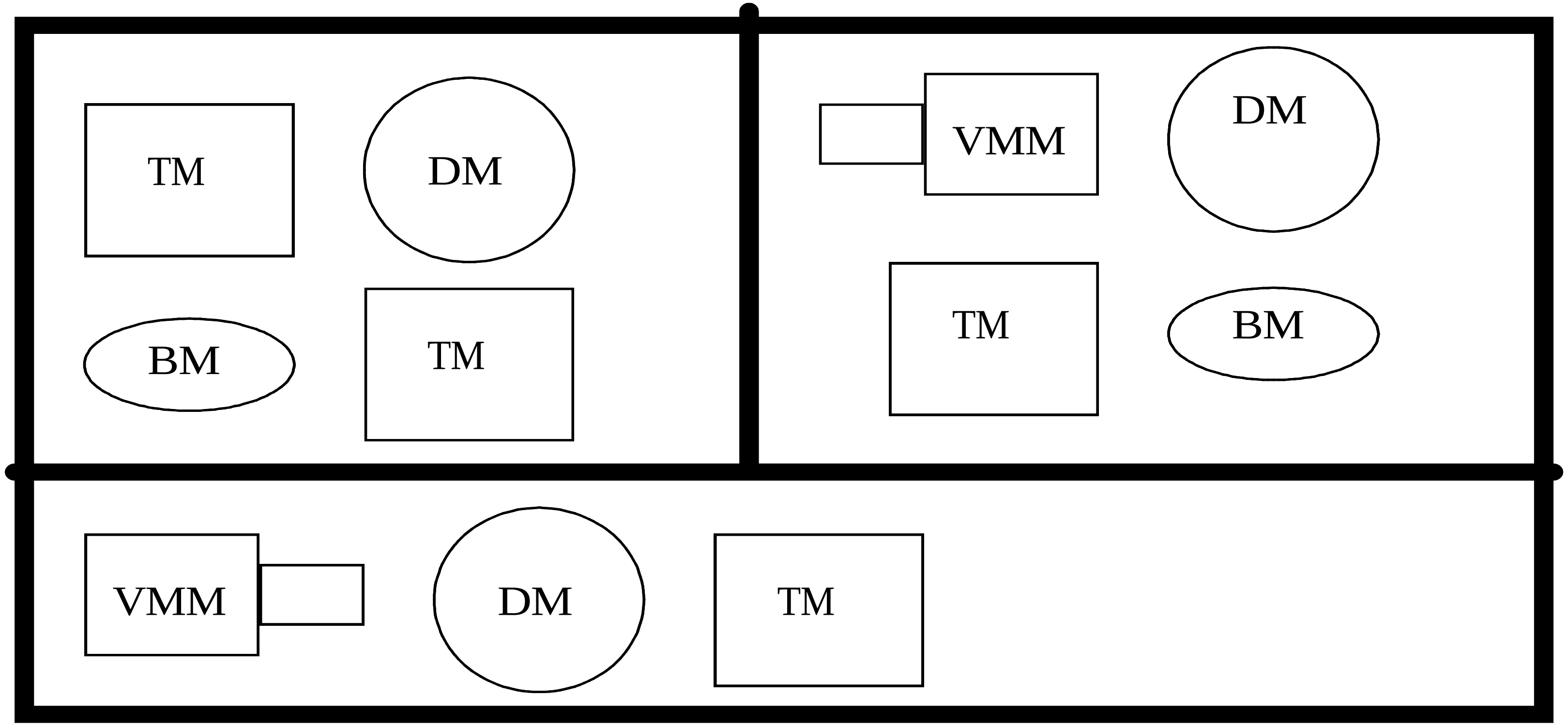
BENEFITS

- Easier to justify automation
- Standardization in design
- Easier, more standardized process plans

Process layout



Group Technology Layout



Sample part-machine processing indicator matrix

$$
[a_{ij}] = \begin{array}{ccc}
 & & \begin{array}{ccccccc} \text{M} & \text{a} & \text{c} & \text{h} & \text{i} & \text{n} & \text{e} \\ M_1 & M_2 & M_3 & M_4 & M_5 & M_6 & M_7 \end{array} \\
\begin{array}{c} \\ \text{P} \\ \text{a} \\ \text{r} \\ \text{t} \\ \\ \end{array} & \begin{array}{c} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} & \left[\begin{array}{ccccccc} 1 & & & 1 & & 1 & \\ & 1 & 1 & & 1 & & \\ & & & 1 & & 1 & \\ & 1 & 1 & & & & \\ & & 1 & & & & 1 \\ & 1 & & & 1 & & 1 \end{array} \right]
\end{array}
$$

Rearranged part-machine processing indicator matrix

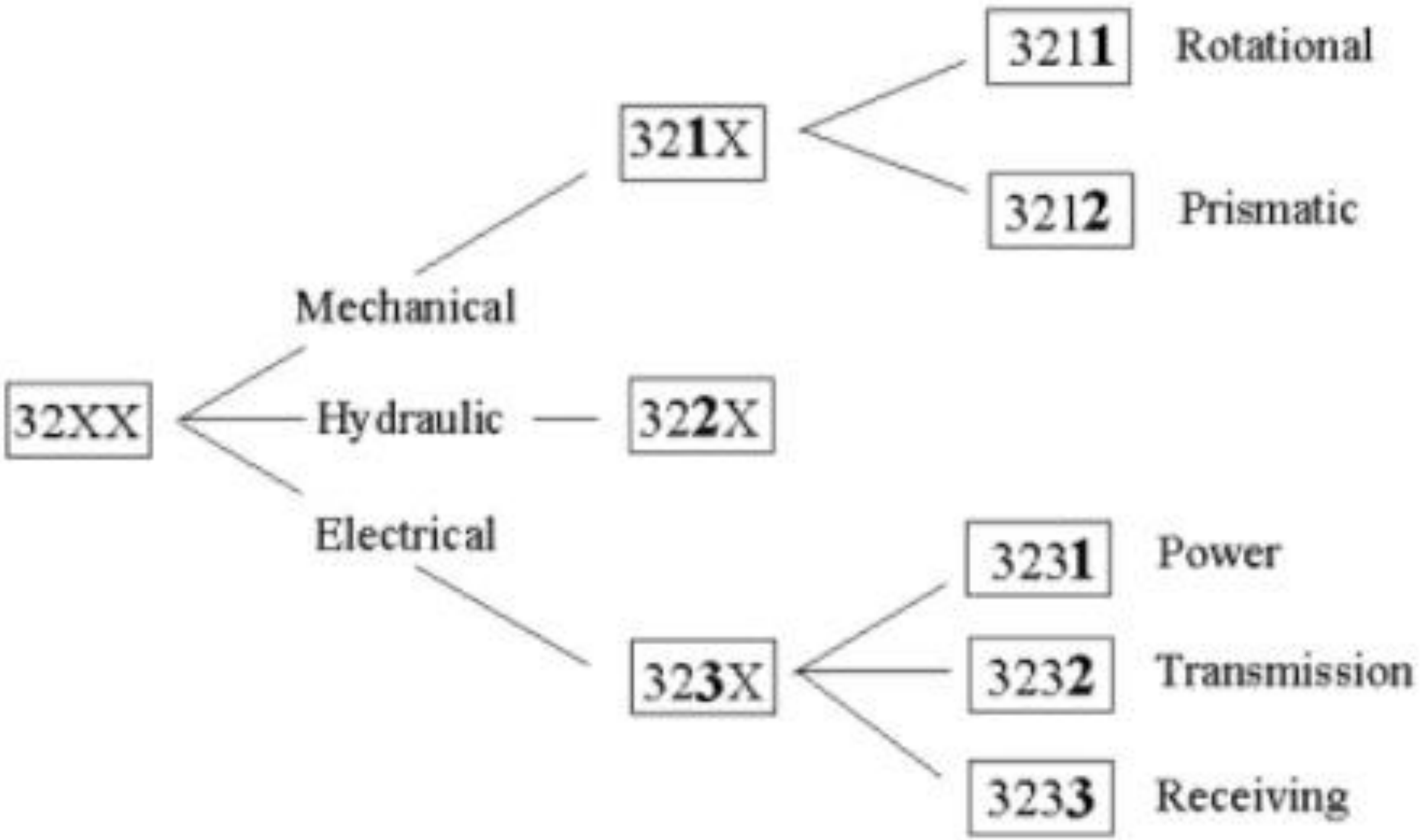
$$
[a_{ij}] = \begin{array}{ccc}
 & & \begin{array}{ccccccc} \text{M} & \text{a} & \text{c} & \text{h} & \text{i} & \text{n} & \text{e} \\ M_1 & M_4 & M_6 & M_2 & M_3 & M_5 & M_7 \end{array} \\
\begin{array}{c} \\ \text{P} \\ \text{a} \\ \text{r} \\ \text{t} \\ \\ \end{array} & \begin{array}{c} P_1 \\ P_3 \\ P_2 \\ P_4 \\ P_5 \\ P_6 \end{array} & \left[\begin{array}{ccc|cccc} 1 & 1 & 1 & & & & \\ & 1 & 1 & & & & \\ \hline & & & 1 & 1 & 1 & \\ & & & 1 & 1 & & \\ & & & & 1 & & 1 \\ & & & 1 & & 1 & 1 \end{array}\right]
\end{array}
$$

Classification and Coding Schemes

- Hierarchical
- Non-hierarchical
- Hybrid

1. <u>Hierarchical structure (or Monocode)</u>

- A code in which each digit <u>amplifies</u> the information given in the previous digit
  - Difficult to construct
  - Provides a deep analysis
  - Usually for permanent information



Structure of Monocode

Classification and Coding Schemes

2. Non-hierarchical (or Polycode, or chain-type structure)

- Easier to accommodate change
- The interpretaion of each symbol in the sequence is always the same.
- It does not depend on the value of the preceding symbols
  - Each symbol is independent of each other.

3. Mixed Coding

- Mixed of hierarchical and non- hierarchical

Digit	Class of feature	Possible value of digits			
		1	2	3	4
1	External shape	Cylindrical without deviations	Cylindrical with deviations	Boxlike	• • •
2	Internal shape	None	Center hole	Brind center hole	• • •
3	Number of holes	0	1–2	3–5	• • •
4	Type of holes	Axial	Cross	Axial cross	• • •
5	Gear teeth	Worm	Internal spur	External spur	• • •
⋮	⋮	⋮	⋮	⋮	⋮

Clustering Approaches

Clustering Approach Algorithms

- Rank order clustering
- Bond energy
- Row and column masking
- Similarity coefficient
- Mathematical Programming

Rank Order Clustering Algorithm

- The rank order clustering (ROC) algorithm
  - determines a binary value for each row and column
  - rearranges the rows and columns in descending order of their binary values
  - Identifies the clusters

Rank Order Clustering Algorithm

- **Step 1:** Assign binary weight $BW_j = 2^{m-j}$ to each column j of the part-machine processing indicator matrix.

- **Step 2:** Determine the decimal equivalent DE of the binary value of each row i using the formula

$$DE_i = \sum_{j=1}^{m} 2^{m-j} a_{ij}$$

- **Step 3:**
- Rank the rows in decreasing order of their DE values.
- Break ties arbitrarily.
- Rearrange the rows based on this ranking.
- If no rearrangement is necessary, stop; otherwise go to step 4.

Rank Order Clustering Algorithm

- **Step 4:** For each rearranged row of the matrix, assign binary weight $BW_i = 2^{n-i}$.
- **Step 5:** Determine the decimal equivalent of the binary value of each column j using the formula

$$DE_j = \sum_{i=1}^{m} 2^{n-i} a_{ij}$$

- **Step 6:**
- Rank the columns in decreasing order of their DE values.
- Break ties arbitrarily.
- Rearrange the columns based on this ranking.
- If no rearrangement is necessary, stop; otherwise go to step 1.

Rank Order Clustering – Example

		M_1	M_2	M_3	M_4	M_5	M_6	M_7	
Binary weight		64	32	16	8	4	2	1	Binary value
	P_1	1			1		1		74
	P_2		1	1		1			52
$[a_{ij}] =$	P_3				1		1		10
	P_4		1	1					48
	P_5			1				1	17
	P_6		1			1		1	37

Rank Order Clustering – Example

$$
\begin{array}{r c l c}
 & & \begin{array}{ccccccc} M_1 & M_2 & M_3 & M_4 & M_5 & M_6 & M_7 \\ 32 & 28 & 26 & 33 & 20 & 33 & 6 \end{array} & \text{Binary weight} \\
[a_{ij}] = & \begin{array}{c} P_1 \\ P_2 \\ P_4 \\ P_6 \\ P_5 \\ P_3 \end{array} & \begin{bmatrix} 1 & & & 1 & & 1 & \\ & 1 & 1 & & 1 & & \\ & 1 & 1 & & & & \\ & 1 & & & 1 & & 1 \\ & & 1 & & & & 1 \\ & & & 1 & & 1 & \end{bmatrix} & \begin{array}{c} 32 \\ 16 \\ 8 \\ 4 \\ 2 \\ 1 \end{array}
\end{array}
$$

Binary value

Rank Order Clustering – Example

$$
\begin{array}{lc}
 & \begin{array}{ccccccc} M_4 & M_6 & M_1 & M_2 & M_3 & M_5 & M_7 \end{array} \\
\text{Binary weight} & \begin{array}{ccccccc} 64 & 32 & 16 & 8 & 4 & 2 & 1 \end{array}
\end{array}
$$

$$
[a_{ij}] = \begin{array}{c|ccccccc|}
 & M_4 & M_6 & M_1 & M_2 & M_3 & M_5 & M_7 \\
P_1 & 1 & 1 & 1 & & & & \\
P_2 & & & & 1 & 1 & 1 & \\
P_4 & & & & 1 & 1 & & \\
P_6 & & & & 1 & & 1 & 1 \\
P_5 & & & & & 1 & & 1 \\
P_3 & 1 & 1 & & & & &
\end{array}
\quad
\begin{array}{c}
\textcolor{red}{\text{Binary value}} \\
112 \\ 14 \\ 12 \\ 11 \\ 5 \\ 96
\end{array}
$$

Rank Order Clustering – Example

$[a_{ij}] =$

	M_4	M_6	M_1	M_2	M_3	M_5	M_7	Binary weight
Binary value	48	48	32	14	12	10	3	
P_1	1	1	1					32
P_3	1	1						16
P_2				1	1	1		8
P_4				1	1			4
P_6				1		1	1	2
P_5							1	1

Bond Energy Algorithm

- Bond Energy algorithm (BEA) is a heuristic that attempts to maximize the sum of the bond energies for each element (i,j) in the part-machine processing indicator matrix $[a_{ij}]$
- The bond energy is defined so that a matrix with clusters or diagonal blocks of 1s will have a larger bond energy compared with the same matrix with rows and columns arranged so that the 1s are uniformly distributed throughout the matrix.
- The bond energy for elemen (i,j) is given by :
  - $a_{ij}(a_{i,j+1} + a_{i,j-1} + a_{i+1,j} + a_{i-1,j})$
  - i.e., you multiply the cell with its neighbours.

Bond Energy Algorithm

Step 1: Set i=1. Arbitrarily select any row and place it.

Step 2: Place each of the remaining n-i rows in each of the i+1 positions (i.e. above and below the previously placed i rows) and determine the row bond energy for each placement using the formula

$$\sum_{i=1}^{i+1} \sum_{j=1}^{m} a_{ij}(a_{i-1,j} + a_{i+1,j})$$

Select the row that increases the bond energy the most and place it in the corresponding position.

Bond Energy Algorithm

Step 3: Set $i=i+1$. If $i < n$, go to step 2; otherwise go to step 4.

Step 4: Set $j=1$. Arbitrarily select any column and place it.

Step 5: Place each of the remaining m-j rows in each of the $j+1$ positions (i.e. to the left and right of the previously placed j columns) and determine the column bond energy for each placement using the formula

$$\sum_{i=1}^{n}\sum_{j=1}^{j+1} a_{ij}(a_{i,j-1} + a_{i,j+1})$$

Step 6: Set $j=j+1$. If $j < m$, go to step 5; otherwise stop.

Bond Energy Algorithm – Example

- **Step 1:** Set i=1 and arbitrarily choose row 2:

$$
\begin{array}{l c}
 & \text{Column} \quad 1 \quad\; 2 \quad\; 3 \quad\; 4 \\
\text{Row} & \\
1 & \\
2 & \\
3 & \\
4 &
\end{array}
\quad
\begin{array}{r}
\\
\\
\left[\begin{array}{cccc}
1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0
\end{array}\right]
\end{array}
$$

Bond Energy Algorithm – Example

Row Selected	Where Placed	Row Arrangement	Row Bond Energy	Maximize Energy
1	Above Row 2	1 0 1 0 0 1 0 1	0	No
1	Below Row 2	0 1 0 1 1 0 1 0	0	No
3	Above Row 2	0 1 0 1 0 1 0 1	4	Yes
3	Below Row 2	0 1 0 1 0 1 0 1	4	Yes
4	Above Row 2	1 0 1 0 0 1 0 1	0	No
4	Below Row 2	0 1 0 1 1 0 1 0	0	No

Bond Energy Algorithm – Example

- The end of the row allocation is the following:

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

- Now start for the column allocation.
- Arbitrarily choose Column1 and continue the algorithm:

Bond Energy Algorithm – Example

Column Selected	Where Placed	Column Arrangement	Column Bond Energy	Maximize Energy
2	Left of Column 1	0 1 0 1 1 0 1 0	0	No
2	Right of Column 1	1 0 1 0 0 1 0 1	0	No
3	Left of Column 1	1 1 1 1 0 0 0 0	4	Yes
3	Right of Column 1	1 1 1 1 0 0 0 0	4	Yes
4	Left of Column 1	0 1 0 1 1 0 1 0	0	No
4	Right of Column 1	1 0 1 0 0 1 0 1	0	No

Bond Energy Algorithm – Example

- The end of the column allocation is the following:

1	1	0	0
1	1	0	0
0	0	1	1
0	0	1	1

Row &Column Masking Algorithm

Step 1: Draw a horizontal line through the first row. Select any 1 entry in the matrix through which there is only one line.

Step 2: If the entry has a horizontal line, go to step 2a. If the entry has a vertical line, go to step 2b.

Step 2a: Draw a vertical line through the column in which this 1 entry appears. Go to step 3.

Step 2b: Draw a horizontal line through the row in which this 1 entry appears. Go to step 3.

Step 3: If there are any 1 entries with only one line through them, select any one and go to step 2. Repeat until there are no such entries left. Identify the corresponding machine cell and part family. Go to step 4.

Step 4: Select any row through which there is no line. If there are no such rows, STOP. Otherwise draw a horizontal line through this row, select any 1 entry in the matrix through which there is only one line and go to Step 2

Row &Column Masking Algorithm – Example

$$
[a_{ij}] = \begin{array}{ccc}
 & & \begin{array}{ccccccc} \mathrm{M} & \mathrm{a} & \mathrm{c} & \mathrm{h} & \mathrm{i} & \mathrm{n} & \mathrm{e} \\ M_1 & M_2 & M_3 & M_4 & M_5 & M_6 & M_7 \end{array} \\
\begin{array}{c} \\ \mathrm{P} \\ \mathrm{a} \\ \mathrm{r} \\ \mathrm{t} \\ \\ \end{array} & \begin{array}{c} P_1 \\ P_2 \\ P_3 \\ P_4 \\ P_5 \\ P_6 \end{array} & \begin{bmatrix}
1 & & & 1 & & 1 & \\
 & 1 & 1 & & 1 & & \\
 & & & 1 & & 1 & \\
 & 1 & 1 & & & & \\
 & & 1 & & & & 1 \\
 & 1 & & & 1 & & 1
\end{bmatrix}
\end{array}
$$

Row &Column Masking Algorithm – Example

The lines are drawn in the order that are shown at the left or at the bottom.

$$
\begin{array}{cccccccccc}
 & & & \text{M} & \text{a} & \text{c} & \text{h} & \text{i} & \text{n} & \text{e} \\
 & & & M_1 & M_2 & M_3 & M_4 & M_5 & M_6 & M_7 \\
 & & P_1 & 1 & & & 1 & & 1 & & 1 \\
 & \text{P} & P_2 & & 1 & 1 & & 1 & & & \\
[a_{ij}] = & \text{a} & P_3 & & & & 1 & & 1 & & 5 \\
 & \text{r} & P_4 & & 1 & 1 & & & & & \\
 & \text{t} & P_5 & & & 1 & & & & 1 & \\
 & & P_6 & & 1 & & & 1 & & 1 & \\
 & & & 2 & & & 3 & & 4 & &
\end{array}
$$

Row &Column Masking Algorithm – Example

The lines are drawn in the order that are shown at the left or at the bottom.

$$
[a_{ij}] = \begin{array}{cc}
 & \begin{array}{ccccccc} \mathrm{M} & \mathrm{a} & \mathrm{c} & \mathrm{h} & \mathrm{i} & \mathrm{n} & \mathrm{e} \end{array} \\
 & \begin{array}{ccccccc} M_1 & M_2 & M_3 & M_4 & M_5 & M_6 & M_7 \end{array} \\
\begin{array}{cc} & P_1 \\ \mathrm{P} & P_2 \\ \mathrm{a} & P_3 \\ \mathrm{r} & P_4 \\ \mathrm{t} & P_5 \\ & P_6 \end{array} &
\left[\begin{array}{ccccccc}
1 & & & 1 & & 1 & \\
 & 1 & 1 & & 1 & & \\
 & & & 1 & & 1 & \\
 & 1 & 1 & & & & \\
 & & 1 & & & & 1 \\
 & 1 & & & 1 & & 1
\end{array}\right]
\begin{array}{c} \\ 1 \\ \\ 5 \\ 8 \\ 6 \end{array} \\
 & \begin{array}{ccccccc} & 2 & 3 & & 4 & & 7 \end{array}
\end{array}
$$

Row &Column Masking Algorithm – Example

The resulting matrix…

$$[a_{ij}] = \begin{array}{cc} & \begin{array}{ccccccc} \mathrm{M} & \mathrm{a} & \mathrm{c} & \mathrm{h} & \mathrm{i} & \mathrm{n} & \mathrm{e} \\ M_1 & M_4 & M_6 & M_2 & M_3 & M_5 & M_7 \end{array} \\ \begin{array}{cc} & P_1 \\ \mathrm{P} & P_3 \\ \mathrm{a} & P_2 \\ \mathrm{r} & P_4 \\ \mathrm{t} & P_5 \\ & P_6 \end{array} & \left[\begin{array}{ccc|cccc} 1 & 1 & 1 & & & & \\ & 1 & 1 & & & & \\ \hline & & & 1 & 1 & 1 & \\ & & & 1 & 1 & & \\ & & & & 1 & & 1 \\ & & & 1 & & 1 & 1 \end{array}\right] \end{array}$$

Similarity Coefficient Algorithm

- The similarity coefficient (SC) between two machines is calculated by:

$$s_{ij} = \frac{\sum_{k=1}^{n} a_{ki} a_{kj}}{\sum_{k=1}^{n} \left(a_{ki} + a_{kj} - a_{ki} a_{kj} \right)},$$

$$\text{where } a_{ki} = \begin{cases} 1 \text{ if part } k \text{ requires machine } i \\ 0 \text{ otherwise} \end{cases}$$

Similarity Coefficient Algorithm

- Each machine is placed in its own cell
- We compute the SC values for each machine pair.
- We place a machine pair in a new cell if its SC value is above a user selected threshold value. Ties are broken arbitrarily.
- This procedure gives a new solution with fewer cells such that one or more cells have two machines in them.
- Then treat each cell as a machine and determine the new set of SC values for the machine pairs, cell pairs and machine-cell pairs.
- SC values for machine pairs are calculated with the previous equation.

Similarity Coefficient Algorithm

- For cell pairs
  - Determine the SC value between each machine in the first cell and every other in the second.
  - The **largest** amongst there values is the SC value of this cell pair.
- Using a new (lower) threshold value, decide whether or not to combine two machines or cells into one as before.

Similarity Coefficient Algorithm

$$[a_{ij}] = \begin{array}{cc|ccccccc|}
 & & \text{M} & \text{a} & \text{c} & \text{h} & \text{i} & \text{n} & \text{e} \\
 & & M_1 & M_2 & M_3 & M_4 & M_5 & M_6 & M_7 \\
\hline
 & P_1 & 1 & & & 1 & & 1 & \\
\text{P} & P_2 & & 1 & 1 & & 1 & & \\
\text{a} & P_3 & & & & 1 & & 1 & \\
\text{r} & P_4 & & 1 & 1 & & & & \\
\text{t} & P_5 & & & 1 & & & & 1 \\
 & P_6 & & 1 & & & 1 & & 1 \\
\hline
\end{array}$$

Similarity Coefficient Algorithm

- **Iteration 1**: Use threshold value of 4/6 = 0.66

Machine Pair	SC Value	Combine into one cell?
{1,2}	0/4=0	No
{1,3}	0/4=0	No
{1,4}	1/2	No
{1,5}	0/3=0	No
{1,6}	1/2	No
{1,7}	0/3=0	No
{2,3}	1/2	No
{2,4}	0/5=0	No
{2,5}	2/3	Yes
{2,6}	0/5=0	No
{2,7}	1/4	No
{3,4}	0/5=0	No
{3,5}	1/4	No
{3,6}	0/5=0	No
{3,7}	1/4	No
{4,5}	0/4=0	No
{4,6}	2/2=1	Yes
{4,7}	0/4=0	No
{5,6}	0/4=0	No
{5,7}	1/3	No
{6,7}	0/4=0	No

Similarity Coefficient Algorithm

- **Iteration 2:** Threshold is 3/6 = 0.5

Machine/Cell Pair	SC Value	Combine into one cell?
{1, (2,5)}	0	No
{1, (4,6)}	1/2	Yes
{1,3}	0	No
{1,7}	0	No
{(2,5), (4,6)}	0	No
{(2,5), 3}	1/2	Yes
{(2,5), 7}	1/3	No
{(4,6), 3}	0	No
{(4,6), 7}	0	No
{3,7}	1/4	No

Similarity Coefficient Algorithm

- **Iteration 3:** Threshold is 2/6 = 0.33

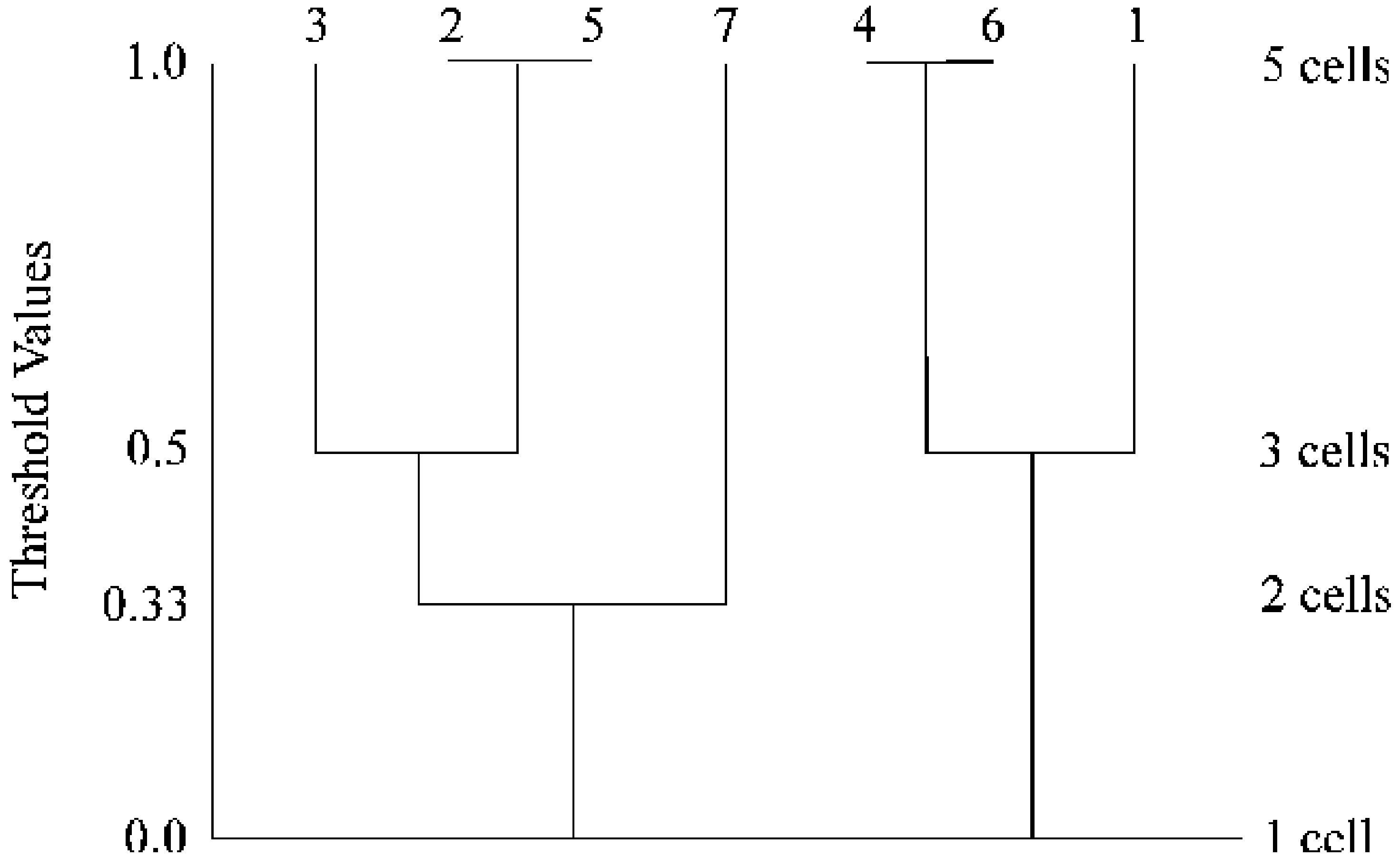
Machine/Cell Pair	SC Value	Combine into one cell?
{(1,4,6) (2,3,5)}	0	No
{(1,4,6), 7}	0	No
{(2,3,5), 7}	1/3	Yes

- **Iteration 4:** Threshold is 0.01

Machine/Cell Pair	SC Value	Combine into one cell?
{(1,4,6) (2,3,5,7)}	0	No

Similarity Coefficient Algorithm

- Dendogram.



Mathematical Programming Approach

- Recall that SC between two <u>machines</u> is calculated by:

$$s_{ij} = \frac{\sum_{k=1}^{n} a_{ki} a_{kj}}{\sum_{k=1}^{n} \left(a_{ki} + a_{kj} - a_{ki} a_{kj} \right)},$$

$$\text{where } a_{ki} = \begin{cases} 1 \text{ if part } k \text{ requires machine } i \\ 0 \text{ otherwise} \end{cases}$$

Mathematical Programming Approach

- the similarity between two parts can be calculated by

$$s_{ij} = \frac{\sum_{k=1}^{m} a_{ik} a_{jk}}{\sum_{k=1}^{m} \left(a_{ik} + a_{jk} - a_{ik} a_{jk} \right)},$$

$$\text{where } a_{ik} = \begin{cases} 1 \text{ if part } i \text{ requires machine } k \\ 0 \text{ otherwise} \end{cases}$$

Mathematical Programming Approach

- We also have a dissimilarity coefficient:

$$d_{ij} = \left[\sum_{k=1}^{n} w_k \left| a_{ki} - a_{kj} \right|^r \right]^{1/r}$$

- *r* is a positive integer
- w_k is the weight for part *k*
- d_{ij} instead of s_{ij} to indicate that this is a dissimilarity coefficient
- Special case where w_k=1, for *k*=1,2,...,*n*, is called the Minkowski metric

- *r*=1, absolute Minkowski metric, and
- *r*=2, the Euclidean metric
- The absolute Minkowski metric measures the dissimilarity between ~~part pairs~~ machine pairs

Mathematical Programming Approach

- We will use the dissimilarity between the parts:

$$d_{ij} = \sum_{k=1}^{m} |a_{ik} - a_{jk}|$$

- Note that it simply takes the difference in two rows.

P – median problem

- Minimize $$\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij}$$

- Subject to $$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1,2,\ldots,n$$

$$\sum_{j=1}^{n} x_{jj} = P$$

$$x_{ij} \le x_{jj} \qquad i, j = 1,2,\ldots,n$$

$$x_{ij} = 0 \text{ or } 1 \qquad i, j = 1,2,\ldots,n$$

P – median problem

- P is a parameter that represents the number of part families desired
- The user must know P a priori.
- The user can solve the model for different values of P for which the minimum cost (or dissimilarity coefficient) solution is obtained
- X_{ij} is a decision variable that takes 0 or 1
  - 1 if part i belongs to part family j, otherwise it is 0.
- Constraint 1 ensures that each part belongs to one part family
- Constraint 2 specifies the desired number of families.
- Constraint 3 guarantess that a part i is assigned to part family j only when this family is formed.
- The objective function minimizes the overall dissimilarities of parts.

P – median problem: Example

- Consider the following machine-part matrix. using mathematical programming approach find the cell formation for 2 cells (i.e., P=2)

$$
\begin{array}{ccc}
 & & \begin{array}{ccccccc} \text{M} & \text{a} & \text{c} & \text{h} & \text{i} & \text{n} & \text{e} \\ M_1 & M_2 & M_3 & M_4 & M_5 & M_6 & M_7 \end{array} \\
[a_{ij}] = & \begin{array}{cc} & P_1 \\ \text{P} & P_2 \\ \text{a} & P_3 \\ \text{r} & P_4 \\ \text{t} & P_5 \\ & P_6 \end{array} & \begin{bmatrix} 1 & & & 1 & & 1 & \\ & 1 & 1 & & 1 & & \\ & & & 1 & & 1 & \\ & 1 & 1 & & & & \\ & & 1 & & & & 1 \\ & 1 & & & 1 & & 1 \end{bmatrix}
\end{array}
$$

P – median problem: Example

- The dissimilarity matrix is given by the following

$$[d_{ij}] = \begin{array}{c|cccccc|} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & 6 & 1 & 5 & 5 & 6 \\ 2 & 6 & 0 & 5 & 1 & 3 & 2 \\ 3 & 1 & 5 & 0 & 4 & 4 & 5 \\ 4 & 5 & 1 & 4 & 0 & 2 & 3 \\ 5 & 5 & 3 & 4 & 2 & 0 & 3 \\ 6 & 6 & 2 & 5 & 3 & 3 & 0 \end{array}$$

```
MIN   6 X12 + X13 + 5 X14 + 5 X15 + 6 X16 + 6 X21 + 5 X23 + X24 + 3 X25 +
      2 X26 + X31 + 5 X32 + 4 X34 + 4 X35 + 5 X36 + 5 X41 + X42 + 4 X43 +
      2 X45 + 3 X46 + 5 X51 + 3 X52 + 4 X53 + 2 X54 + 3 X56 + 6 X61 +
      2 X62 + 5 X63 + 3 X64 +  3 X65
SUBJECT TO
        C1)   X12 + X13 + X14 + X15 + X16 + X11 =    1
        C2)   X21 + X23 + X24 + X25 + X26 + X22 =    1
        C3)   X31 + X32 + X34 + X35 + X36 + X33 =    1
        C4)   X41 + X42 + X43 + X45 + X46 + X44 =    1
        C5)   X51 + X52 + X53 + X54 + X56 + X55 =    1
        C6)   X61 + X62 + X63 + X64 + X65 + X66 =    1
        C7)   X11 + X22 + X33 + X44 + X55 + X66 =    2
        C8)   X21 - X11 <=   0
        C9)   X31 - X11 <=   0
       C10)   X41 - X11 <=   0

            o     o     o

       C34)   X16 - X66 <=   0
       C35)   X26 - X66 <=   0
       C36)   X36 - X66 <=   0
       C37)   X46 - X66 <=   0
       C38)   X56 - X66 <=   0
```

P – median problem: Example Result

Variable	Value
X31	1
X42	1
X52	1
X62	1
X11	1
X22	1
Objective	7