Control
and Cybernetics
VOL. 15 (1986) No. 2

A new cross-decomposition algorithm: the GPM. Comparison with the bond energy method

by

H. GARCIA

J. M. PROTH

INRIA-LORRAINE
Château du Montet 54500 Vandoeuvre
France

A new cross-decomposition method, called GPM, is proposed. Starting from a 0–1 matrix, the GPM leads to a set of non encroaching blocks in such a way that:
1) the number of 1-values in the blocks is as high as possible,
2) the number of 0-values outside the blocks is also as high as possible.
The GPM is then compared with the bond energy method (BEM).

0. Introduction

A job-shop being given, the approach of Group Technology is to find a partition of the machines in production subsystems, and a partition of the parts in the same number of part families in such a way that, if possible, the parts belonging to the same part family are manufactured on the same production subsystem, and only on it.

The first research in that way has been made by Burbidge (1963). Mc Auley (1972) has proposed the single linkage cluster analysis method. Mc Cormick et al. (1972) have developed the bound energy method (BEM). The rank order clustering method (ROC) has been proposed by King (1980).

This paper is devoted to a new method, called GPM, which solves the above mentioned Group Technology problem.

We first recall the characteristics of the problem. In the second paragraph, we introduce a criterion, and set the problem as an optimisation problem. We then prove a fundamental result which is used to build up a new algorithm.

The last paragraph is devoted to the comparison between the GPM and the BEM, because the BEM seems to be the best adapted method for solving the Group Technology problem.

1. Setting of the problem

We consider a set of n objects and a set of m items, called X and Y respectively. The weight of the object i is denoted by w_i.

We define the matrix $A=[a_{i,j}]$, $i=1, ..., n$; $j=1, ..., m$, as follows:

$$a_{i,j}=\begin{cases}1 \text{ if the item } j \text{ holds for the object } i\\ 0 \text{ otherwise}\end{cases} \tag{1}$$

We denote by W_q the set of pairs (P_X^q, P_Y^q), where $P_X^q=\{X^1, ..., X^q\}$ is a partition of X into q subsets and $P_Y^q=\{Y^1, ..., Y^q\}$ is a partition of Y into q subsets.

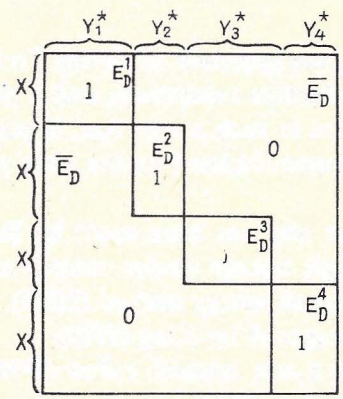
We are looking for a couple (P_X^{*q}, P_Y^{*q}), where $P_X^{*q}=\{X^{*1}, ..., X^{*q}\}$ and $P_Y^{*q}=\{Y^{*1}, ..., Y^{*q}\}$, in such a way that:

1. $E_D=\{a_{ij}/(i,j) \in \bigcup_{r=1}^{q} (X^{*r}, Y^{*r})\}$ contains the maximal number of 1-values "distributed as well as possible among the different blocks:

$$E_D^r=\{a_{i,j}/(i,j) \in (X^{*r}, Y^{*r})\}"$$

2. $\bar{E}_D=\{a_{ij}/a_{ij} \notin E_D\}$ contains the maximal number of 0-values.

In terms of Group Technology, if X is a set of parts and Y a set of machines, the problem consists in finding a partition of the set of parts into q part families and a partition of the set of machines into q production subsystems in such a way that, for $r=1, ... q$, each part belonging to the r-th part family is mainly manufactured using the machines of the r-th production subsystem.



MATRIX B

Fig. 1

In other words, the objective is to obtain a matrix like B (see Fig. 1). starting from A by permutation of rows and columns in an adequate manner.

In the next paragraph, we choose a criterion, set the problem, and propose a fast algorithm which contributes to a "good" solution of this problem.

2. A fast cross-decomposition method

2.1. The problem

Using the previous notation, we set the problem as follows:
find a couple (P_X^{*q}, P_Y^{*q}) in such a way that:

$$h \sum_{(i,j) \in \bigcup_{r=1}^{q} (X^{*r}, Y^{*r})} w_i a_{ij} + (1-h) \sum_{(i,j) \notin \bigcup_{r=1}^{q} (X^{*r} Y^{*r})} w_i (1-a_{ij}) =$$

$$= \operatorname{Max}_{(P_X^q, P_Y^q) \in W_q} \left\{ h \sum_{(i,j) \in \bigcup_{r=1}^{q} (X^r, Y^r)} w_i a_{ij} + (1-h) \sum_{(i,j) \notin \bigcup_{r=1}^{q} (X^r, Y^r)} w_i (1-a_{ij}) \right\} \quad (2)$$

where $h \in [0, 1]$.

Suppose that we are able to find (P_X^{*q}, P_Y^{*q}) which verifies (2). The result depends on the value of h, and for some values of h, one or more elements of P_X^{*q} and P_Y^{*q} may be empty.

For instance, if $h=1$, it is easy to see that:

$$\begin{cases} X^{*1}=\{1, ..., n\}, & X^{*2}=X^{*3}= ... =X^{*q}=\emptyset \\ Y^{*1}=\{1, ..., m\}, & Y^{*2}=Y^{*3}= ... =Y^{*q}=\emptyset \end{cases}$$

is an optimal solution of the problem, according to the criterion (2).

Then, if the constraint:

$$\left.\begin{matrix} X^{*r} \neq \emptyset \\ Y^{*r} \neq \emptyset \end{matrix}\right\} r=1, ..., q \quad (3)$$

holds, we often will have to try various values of h before obtaining a solution which verifies (3). A good strategy to reach such a value of h is the following (ε below is a given constant, e.g. the occuracy with which computer hold numbers):

1. Set $h^1=1/2$
2. For $i=2, 3, 4, ...$
   - 2.1. Set $\varepsilon_i=1/2^i$
   - 2.2. If h^{i-1} leads to a solution which does not verify (3), then
     - 2.2.1. if $\varepsilon_i<\varepsilon$, then
       - 2.2.1.1. We don't reach a solution which verifies (3)
       - 2.2.1.2. End of the process
     - 2.2.2. If $\varepsilon_i \geqslant \varepsilon$, then
       - 2.2.2.1. Set $h^i=h^{i-1}-\varepsilon_i$
       - 2.2.2.2. End of the loop 2.

2.3. If h^{l-1} leads to a solution which verifies (3), then

2.3.1. We keep the solution

2.3.2. End of the process.

In the following, we always use the same process in order to choose a good value of h.

2.2. A good solution knowing an initial partition $P^q_{X,1}$

We suppose that there exists a partition of the set of rows of A in q non-empty subsets:

$$P^q_{X,1}=\{X^1_1, X^2_1, ..., X^q_1\}$$

We will show that it is possible to obtain a good solution of (2) starting from this initial partition. We first prove the fundamental result.

2.2.1. The fundamental result

THEOREM 1. *Let* $P^q_{X,K}=\{X^1_K, X^2_K, ..., X^q_K\}$ *and* $P^q_{Y,K}=\{Y^1_K, Y^2_K, ..., Y^q_K\}$ *be partitions of* X *and* Y *respectively.*

1. For $j=1, 2, ..., m$, we compute:

$$y^r_K(j)=h\sum_{i\in X^r_K} w_i\, a_{ij}+(1-h)\sum_{i\notin X^r_K} w_i\,(1-a_{ij})$$

for $r=1, 2, ..., q$

and we denote by r^j_K the integer which verifies:

$$y^{r^j_K}_K(j)=\max_{r=1,2,...,q} y^r_K(j) \tag{4}$$

We then assign the item j to the subset $Y^{r^j_K}_{K+1}$

Finally, we obtain the partition:

$$P^q_{Y,K+1}=\{Y^1_{K+1}, Y^2_{K+1}, ..., Y^q_{K+1}\}$$

in which some elements may be empty.

2. For $i=1, 2, ..., n$, we compute:

$$x^s_K(i)=h\sum_{j\in Y^r_K} a_{ij}+(1-h)\sum_{j\notin Y^r_K}(1-a_{ij})$$

for $s=1, 2, ..., q$

and we denote by s^i_K the integer which verifies:

$$x^{s^i_K}_K(i)=\max_{s=1,2,...,q} x^s_K(i) \tag{5}$$

We then assign the object to the subset $X^{s^i_K}_{K+1}$

Finally, we obtain the partition:

$$P^q_{X,K+1}=\{X^1_{K+1}, X^2_{K+1}, ..., X^q_{K+1}\}$$

in which some elements may be empty.

Then, either the partitions $(P^q_{X,K+1}, P^q_{Y,K+1})$ are better than the partitions $(P^q_{X,K}, P^q_{Y,K})$ with regard to the criterion (2), or the value of the criterion (2) is the same for both solutions.

Proof.

1. Let us consider $j \in \{1, 2, ..., m\}$ and suppose that:

$$j \in Y^r_K \text{ with } r \in \{1, 2, ..., q\}$$

If $r=r^j_K$, then the contribution of the item j to the critetion (2), knowing $P^q_{Y,K}$, remains the same.

If $r \neq r^j_K$, then this contribution increases (see (4)) and $P^q_{Y,K+1}$ leads to a greater value of the criterion (2) than $P^q_{Y,K}$ (knowing $P^q_{X,K}$).

Finally, $(P^q_{X,K}, P^q_{Y,K+1})$ is either better than $(P^q_{X,K}, P^q_{Y,K})$, or leads to the same value of the critetion (2). (6)

2. Using the same arguments, it is easy to show that the solution $(P^q_{X,K+1}, P^q_{Y,K+1})$ is either better than $(P^q_{X,K}, P^q_{Y,K+1})$, or leads to the same value for the criterion (2). (7)

3. From (6) and (7), we conclude that either the solution $(P^q_{X,K+1}, P^q_{Y,K+1})$ is better than the solution $(P^q_{X,K}, P^q_{Y,K})$, or that the value of the criterion (2) is the same for both solutions.

This completes the proof. ■

2.2.2. *The algorithm*

We suppose that the initial partition $P^q_{X,1}=\{X^1_1, X^2_1, ..., X^q_1\}$ is known.

We propose the following algorithm for a given value of $h \in [0, 1]$.

1. Set $K=2$
2. Compute $P^q_{Y,K}$ starting from $P^q_{X,K-1}$ (see (4))
3. Compute $P^q_{X,K}$ starting from $P^q_{Y,K}$ (see (5))
4. If $[(K=2)$ or $((P^q_{X,K}, P^q_{Y,K})$ is different from $(P^q_{X,K-1}, P^q_{Y,K-1}))]$, go to 2, else $(P^q_{X,K}, P^q_{Y,K})$ is a "good" solution.

THEOREM 2. *The algorithm given above converges.*

Proof. Obvious, if we consider that:

1. the number of partitions of X and Y is finite,
2. the solution $(P^q_{X,K}, P^q_{Y,K})$ is different at each step or the algorithm stops.

REMARKS

1. The previous algorithm leads to a solution which depends on the initial partition $P^q_{X,1}$. We then have to make several trials in order to obtain a solution close to the optimal one.

2. The initial partition $P^q_{X,1}$ can be obtained using the "nuées dynamiques" method. In this method, the number q has to be chosen by the user (see Diday, Lemaire, Pouget and Testu (1982)). We summarize this method in the next paragraph.

2.2.3. The "nuées dynamiques" method

This method begins with the search of q initial points (i.e. q initial rows of [A]) which are as far as possible one from the other, q being the number of subsets belonging to $P^q_{X,1}$. We use the Euclidean distance. In the software we are working with, the first point is chosen at random. We then compute the number of points for which the distance to the first one is less than ε, ε being chosen by the user. This number is called the density of the first point. We then cancel the first point and the points used to compute its density, and restart the process by choosing at random a point in the remaining set of points, and so on. The process stops if the remaining set of points is empty. The q initial points are those with the highest densities.

The second step of the "nuées dynamiques" method can be described as follows (n is the total number of points) according to, Diday, Lemaire, Pouget and Testu (1982):

1. For $k=1, 2, \ldots$
   - 1.1.1. For $i=1, 2, \ldots, n$
     - 1.1.1. For $p=1, \ldots, q$
       - Compute $d_{i,p}$, distance between the p-th initial point and the i-th point of the initial set of points.
     - 1.1.2. Search $d_{i,p^*} = \underset{p=1,\ldots,q}{\text{Min}}\ d_{i,p}$
     - 1.1.3. Set the i-th point in the class represented by the p^*-th initial point
   - 1.2. Set $q=q^1$, where $q^1 \leqslant q$ is the number of non-empty classes obtained
   - 1.3. Compute the inertia center of each of the q classes to replace the previous initial points
   - 1.4. If the classes are the same as the previous ones, go to 2, else go to 1 (next value of k)
2. End of the process.

3. The bond energy method

As far as it is known, this method has been developed by Mc Cormick et al. (1972).

We summarize it in this paragraph, using the previous notations.

Let us consider two consecutive columns of $[A]$, for instance j and $j+1$ $(0<j<m)$:

$$[A_{.,j}]=\begin{bmatrix} a_{1,j} \\ a_{2,j} \\ \vdots \\ a_{n,j} \end{bmatrix} \quad \text{and} \quad [A_{.,j+1}]=\begin{bmatrix} a_{1,j+1} \\ a_{2,j+1} \\ \vdots \\ a_{n,j+1} \end{bmatrix}$$

The bond energy measure between these colums is given by:

$$E(A_{.,j}, A_{.,j+1})=\sum_{i=1}^{n} a_{i,j} \cdot a_{i,j+1}$$

Similarly, the bond energy between two consecutive lines of A, for instance:

$$\left.\begin{aligned} [A_{i,.}]&=[a_{i,1}, a_{i,2}, \ldots, a_{i,m}] \\ [A_{i+1,.}]&=[a_{i+1,1}, a_{i+1,2}, \ldots, a_{i+1,m}] \end{aligned}\right\} \quad 0<i<n$$

is given by:

$$E(A_{i,.}, A_{i+1,.})=\sum_{j=1}^{n} a_{i,j} \cdot a_{i+1,j}$$

The bond energy method consists in classifying the rows and the columns in order to maximize the total bond energy of the matrix.

This problem is a quadratic assignment problem. The computing time increases exponentially with the size of the problem. It is the reason why Mc Cormick et al. have proposed a sub-optimal heuristic procedure which can be summarized as follows:

1. Choose one of the columns arbitrarily and place it at the rank 1
2. For $j=2$ to m

   j columns being placed, try placing each of the $m-j$ remaining columns in each of the i possible positions (between the first and the second columns already placed, between the second and the third, ..., and, finally, after the $(j-1)$-th column). Keep the column which leads to the maximal increase of the total bond energy and its position.
3. Choose one of the rows arbitrarily and place it at the rank 1
4. For $i=2$ to n

   Repeat on the rows the procedure described in point 2.

As it will be seen, the results depend on the choice of the first line and of the first column.

4. Comparison of the two methods

In what follows, the bond energy method will be called BEM, and the new method showed in the second paragraph will be called GPM (i.e. Garcia and Proth's method).

In order to make a comparison between BEM and GPM, we first have to choose significant criteria regarding the problem presented in the first paragraph.

4.1. The criteria

Two types of criteria seem to be significant.

4.1.1. Computation times

The first type of criteria deals with the computation times.

For the BEM method, we give the "CPU-equivalent time" which is a quantity proportional to the CPU-time needed to reach the solution.

We give two "CPU-equivalent times" for the GPM method. The first one corresponds to the time needed to obtain the initial partition $P^q_{X,1}$ (see 2.2.3.), and the second one corresponds to the time used to reach a "good" solution using the algorithm used in 2.2.2. The search of the q initial points using a density approach needs the bigest part of the "CPU-equivalent time" (between 70 and 90%). In practice, it is easy to spare this time either by choosing the initial points at random, or by giving it to the system.

4.1.2. Quality of the solution

Let us consider Figure 1.

For $k=1, 2, ..., q$, we define the density d_k of the block E^k_D by:

$$d_k = \sum_{(i,j)\in E^k_D} a_{i,j}/s_k$$

where s_k is the size (i.e. the number of elements) of E^k_D.

In the same way, the density of $\overline{E_D}$ is given by:

$$\bar{d} = \sum_{(i,j)\in \overline{E_D}} (1-a_{i,j})/\bar{s}$$

where $\bar{s}$ is the size of $\overline{E_D}$. In other words:

$$\bar{s} = n \cdot m - \sum_{k=1}^{q} s_k$$

(n an m are respectively the number of rows and the number of columns of [A]).

The following values are significant to evaluate the quality of the solution:

a. the main density

$$S = \frac{\sum_{k=1}^{q} d_k + \bar{d}}{q+1}$$

b. the standard deviation of the densities

$$\sigma_S = \left[\frac{1}{q+1} \left\{ \sum_{k=1}^{q} (d_k - S)^2 + (\bar{d} - S)^2 \right\} \right]^{1/2}$$

In some cases, the BEM method leads to a matrix $[B]$ for which it is impossible to extract q blocks. In that case, we indicate that a solution does not exist.

4.1.3. The test

The matrices used to test the method are obtained using the following process:

1. We give the size of the matrix (i.e. n and m)
2. We give q, number of expected blocks
3. The number of rows and columns of each block is generated at random.
4. We give a "percentage of noise". It is the probability to have 0 in a given location of a block and 1 in a given location outside the blocks. The matrix $[B]$ is generated at random taking into account this percentage.
5. Finally, to obtain $[A]$, we mixt the rows and the columns of $[B]$ using a random process.

Some results are given in the following table.

Test number	Size of the matrix		% of noise	BEM				GPM				
	n rows	m colums		q	CPU equ. times	S	σ_S	First step CPU-t.	Second step CPU-t.	Total CPU-eq. time	S	σ_S
1	15	10	0	3	6.1	1	0	3	3.3	6.3	1	0
2	15	10	5	3	6	0.966	0.034	4.5	4.6	9.1	0.966	0.034
3	25	10	10	4	8.5	0.936	0.046	6.5	5.4	11.9	0.938	0.038
4	15	10	15	5	6.2	no solution	no solution	4.4	4.7	9.1	0.916	0.061
5	100	20	20	3	70	0.749	0.295	76	19	95	0.902	0.047

All the tests we have made lead to the same conclusion:

— The quality of the solution is good with both methods if the percentage of noise is close to zero. (The quality of the solution is good if S is close to 1 and σ is close to 0).

— The quality of the solution given by the BEM decreases very fast if the percentage of noise or/and the size to the matrix increases. On the contrary, the stability of the GPM is noteworthly.

— The CPU time is better with the BEM method than with the GPM method if we search the initial partition using the density approach. The conclusion is different if the user gives the initial points.

Conclusion

The aim of the GPM is to find a set of non encroaching blocks in a matrix by permutation of the rows and columns in an adequate manner.

As far as we know, the BEM was one of the best adapted method to reach this objective. It seems that the performances of the GPM are better than those of the BEM. This result is of high interest in Group Technology.

As we have shown, the result obtained using the GPM depends on the initial partition of the rows. Consequently, it depends on the first initial point, which is chosen at random, and on the process used to choose the other points. A possible way to the future research is to find a fast method which leads to a "good" initial partition.

References

[1] Diday E., Lemaire J., Pouget J., Testu F. Eléments d'Analyse des Données. Dunod, 1982.
[2] Burbidge J. L. Production flow analysis. *Prod. Engr.*, **42** (1963), 742.
[3] King J. R. Machine-component grouping using ROC algorithm. *Int. J. Prod. Res.* **18** 2, (1980) 213–231.
[4] Mc Auley J. Machine grouping for efficient production. *Prod. Engr.*, **51** (1972) 53.
[5] Mc Cormick W. T., Schweitzer P. J., White T. E. Problem decomposition and data reorganisation by a clustering technique *Opns. Res.*, **20** (1972), 993.

GPM — nowy algorytm dekompozycji skrośnej. Porównanie z metodą energii powiązań

Zaproponowano nową metodę dekompozycji skrośnej, nazywaną dalej GPM. GPM przeprowadza 0–1 macierz w zbiór wzajemnie rozłącznych bloków w ten sposób, że

1. liczba jedynek w blokach jest możliwie największa,
2. liczba zer poza blokami jest także możliwie największa.

Następnie porównano GPM z metodą energii powiązań BEM.

ГПМ — новый алгоритм сквозной декомпозиции. Сравнение с методом энергии связей.

Предложено новый метод сквозной декомпозиции далее называемы ГПМ. ГПМ переводит 0—1 матрицу в совокупность раздельных блоков, так что:

1. число единиц в блоках возможно больше.
2. число нулей вне блоков тоже возможно больше.

Затем сравнено ГПМ с методом энергии связей.