ELSEVIER

European Journal of Operational Research 107 (1998) 175–192

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

Optimal solution of cellular manufacturing system design: Benders' decomposition approach

Sunderesh S. Heragu [a,*], Ja-Shen Chen [b]

[a] *Department of Decision Sciences and Engineering Systems, Rensselaer Polytechnic Institute, 110, 8th Street, Troy, NY 12180-3590, USA*

[b] *Department of Business Administration, Yuan-Ze Institute of Technology, Chung-Li, 320 Taiwan*

Received 1 November 1996; accepted 1 April 1997

Abstract

In this paper, a mathematical model for cellular manufacturing system (CMS) design which incorporates three critical aspects – resource utilization, alternate routings, and practical constraints, is presented. The model is shown to be NP-complete. A linear, mixed-integer version of the model which not only has fewer integer variables compared to most other models in the literature, but one that also permits us to solve it optimally using Benders' decomposition approach is presented. Some results that allow us to solve the problem efficiently as well as computational results with Benders' decomposition algorithm and a modified version are presented.

Keywords: Cellular manufacturing; Optimization; Decomposition; Mathematical programming

1. Introduction

One of the basic problems in the design of a cellular manufacturing system (CMS) is that of cell formation. This involves identifying: (i) parts with similar processing requirements; (ii) the set of machines that can process these parts, and (iii) dedicating the set of machines to the manufacture of the parts. The set of machines is called a machine cell and the corresponding set of parts is a part family. There are a number of benefits in identifying machine cells and corresponding part families. These are outlined in several sources including Wemmerlov and Hyer (1989). Industry has recognized these benefits and a large number of companies have implemented CMSs throughout the world, especially in the last decade.

Academia has attempted to solve the cell formation problem using several techniques. One of the techniques discussed most is the clustering technique. It involves identification of block diagonal clusters by rearranging the rows and columns of a given 0-1 part-machine incidence matrix $[a_{ij}]$ (see Fig. 1(a) and (b)). This matrix represents the relations between parts and machines – if a_{ij} is 1, it indicates that part i visits

*Corresponding author. Fax: +1-518-276-8227; e-mail: herags@rpi.edu.

S0377-2217/98/$19.00
PII S0377-2217(97)00256-7

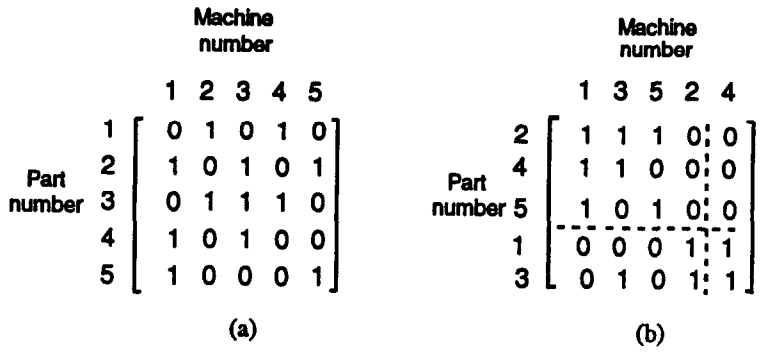

Fig. 1. 0-1 Part–machine incidence matrix.

machine j; a '0' indicates that part i does not visit machine j. There are some well known clustering techniques for the cell formation problem. These include similarity coefficient algorithm by McAuley (1972), ROC (rank order clustering) algorithms by King (1980) and King and Nakornchai (1982), direct clustering algorithm by Chan and Milner (1982). These and other algorithms have been reviewed in Vakharia (1986), Wemmerlov and Hyer (1986), and more recently in Singh (1993). In addition, there are papers dealing with other practical issues such as machine utilization, consideration of various costs, safety factors, upper bound on machine cells, size of each cell and capacity. For example, see Ballakur and Steudel (1987), Askin and Subramanian (1987), Heragu and Gupta (1994) and Grznar et al. (1994). Such papers have been reviewed in Heragu (1994) and Offodile et al. (1994).

Recently, researchers have recognized the importance of considering three critical aspects in the CMS design problem. The first aspect has to do with the explicit consideration of 'voids' in the final block diagonal structure. Chandrasekharan and Rajagopalan (1986) and Kumar and Chandrasekharan (1990) argued that in addition to minimizing the number of nonzero elements outside the final block diagonal structure in Fig. 1(b), we must also attempt to minimize the number of zero elements (voids) inside the blocks. Although they proposed two quantitative measures – grouping efficiency and its improved version, grouping efficacy, to evaluate goodness of CMS design in terms of number of exceptional elements and number of voids in the final block diagonal structure, they did not propose any technique for the problem. Adil et al. (1997) and Srinivasan and Narendran (1991) developed approaches that explicitly considered minimization of voids and exceptional elements. However, both are heuristic approaches and do not consider important practical constraints.

Throughout this paper we use the term 'resource utilization of parts' rather than 'voids' because we feel the term 'voids' is unclear to most readers. Moreover, minimizing the so-called voids is equivalent to maximizing the utilization of resources in the cell visited by each part. The first aspect discussed above is in contrast to previous cluster formation techniques which only attempted to minimize the number of nonzero elements (exceptional elements) outside the final block diagonal structure.

The second aspect involves consideration of alternate routings while designing a CMS. Unlike traditional manufacturing systems in which each part typically has only one routing or process plan, in advanced manufacturing systems, the consideration of alternate routings has become an attractive practice for production managers. It provides planning flexibility and allows us to improve throughput rates. For example, if a part has multiple process plans, and it turns out that one of the machines in the primary process plan is busy or down for repair, we are able to route the part using an alternate plan. There are only a limited number of papers dealing with this aspect. Kang and Wemmerlov (1993) provided a list and brief discussion of most of these papers. However, to date, the objective of most papers dealing with alternate routings is only for route selection, i.e., they only try to find the best route for each part from the available routes so as to minimize the intercell movement of parts. Once the best route is determined using one of the available alternate plans, the remaining are discarded and not considered further.

The third aspect that has long been ignored is the consideration of practical constraints while designing a CMS. Examples of such constraints are: capacity, safety, budget, upper limit on the number of cells and number of machines in each cell. A thorough discussion of these constraints is provided in Heragu and Gupta (1994).

In this paper, we expand the model in Adil et al. (1997) to consider the three aspects discussed previously, develop an efficient linear, mixed-integer version of the model and solve it optimally. It should be noted that existing techniques (most of which do not consider the issues and constraints addressed in this paper) are heuristic in nature.

2. Model description

The mathematical model presented in this section assumes that: (1) product mix and product demand information is known a priori and constant, and (2) operation requirements for each part are known. The notation used is provided below:

Parameters

i, j, k part index, machine index, cell index, respectively
a_{ij} part–machine processing indicator
c_i intercell movement cost per unit of part i
v_i number of units of part i
t_{ij} time for processing one unit of part i on machine j
u_{ij} cost of part i not utilizing machine j
T_j available operating time on machine j
o_{ij} number of times part i requires operation on machine j
$M_{\min}$ minimum number of machines permitted in a cell
$M_{\max}$ maximum number of machines permitted in a cell
S_1 sets of machine pairs that cannot be located in the same cell
S_2 sets of machine pairs that must be located in the same cell
p total number of part types
q total number of machines
r maximum number of cells permitted

Decision variables

N_j number of units of machine j required
x_{ik} variable indicating whether or not part i is processed in cell k
y_{jk} $\begin{cases} = 0 & \text{if machine } j \text{ is not in cell } k \\ = 1 & \text{otherwise} \end{cases}$

In the above notation, the parameter a_{ij} is set to 0 if machine j is not required for part i. Otherwise, it is set to a value greater than 0 but less than or equal to 1, depending upon whether part i is partially or entirely processed on machine j. Similarly, the decision variable x_{ij} takes on a value of 0 if part i is not processed in cell k. Otherwise, it takes a value greater than 0 but less than or equal to 1, depending upon whether part i is partially or entirely processed in cell j.

2.1. Problem formulation

The problem of determining the required number of machines to satisfy capacity constraints so that the total purchase cost is minimized, is trivial and can be obtained by using the following rule:

$$N_j = \left[\frac{\sum_j v_i a_{ij} t_{ij}}{T_j}\right],$$

where $[\bullet]$ is the smallest integer greater than or equal to $\bullet$.

Model I (minimize cost related to intercell movement and resource under-utilization)

$$\text{minimize} \quad \sum_{i=1}^{p}\sum_{j=1}^{q}\sum_{k=1}^{r} c_i v_i o_{ij} a_{ij} x_{ik}(1 - y_{jk}) + \sum_{i=1}^{p}\sum_{j=1}^{q}\sum_{k=1}^{r} u_{ij} v_i (1 - a_{ij}) x_{ik} y_{ik} \tag{1}$$

$$\text{subject to} \quad \sum_{k=1}^{r} x_{ik} = 1, \qquad i = 1, 2, \ldots, p, \tag{2}$$

$$\sum_{k=1}^{r} y_{jk} = 1; \qquad j = 1, 2, \ldots, q, \tag{3}$$

$$y_{sk} + y_{tk} \leqslant 1, \qquad k = 1, 2, \ldots, r, \{s, t\} \in S_1, \tag{4}$$

$$y_{sk} - y_{tk} = 0, \qquad k = 1, 2, \ldots, r, \{s, t\} \in S_2, \tag{5}$$

$$M_{\min} \leqslant \sum_{j=1}^{m} y_{jk} \leqslant M_{\max}, \quad k = 1, 2, \ldots, r, \tag{6}$$

$$0 \leqslant x_{ik} \leqslant 1, \qquad i = 1, 2, \ldots, p, \; k = 1, 2, \ldots, r, \tag{7}$$

$$y_{jk} = 0 \text{ or } 1, \qquad j = 1, 2, \ldots, q, \; k = 1, 2, \ldots, r. \tag{8}$$

The objective function (1) minimizes the total cost of intercell movement as well as the cost of resource under-utilization. Note that u_{ij}, the under-utilization penalty for part i–machine j pair is a function of part i's volume (v_i), so that the resource under-utilization weight is not insignificant in comparison to the intercellular movement weight in the objective function. Constraint (2) requires that each part be allocated to only one of r cells. Constraints (3) and (8) ensure each machine can only be assigned to one cell. Constraint (4) ensures that the machine pairs included in S_1 cannot be placed in the same cell. Similarly, constraint (5) ensures machine pairs in S_2 have to be placed in the same cell. Constraint (6) specifies the minimum and maximum number of machines allowed in any cell.

2.2. Special features of model I

Model I has several special features not found in other papers. These are discussed below.

Consideration of alternating routings: In model I, we assume all the alternate routes will be used during production runs, some more often than others, to cope with machine down time and also to introduce planning flexibility. Use of alternate routes improves machine utilization and system throughput rate. To include alternate routings in our model, elements of the part–machine incidence matrix-a_{ij}'s, are considered as real variables between 0 and 1. Traditionally, they have been given binary values. For example, if part 1 has two routes with the first through machines 1, 3, 5 and the second through machines 1, 4, 5, and the probability of using the two routes is 0.8 and 0.2 respectively, then in the part-machine matrix, we assign $a_{11} = 1$, $a_{13} = 0.8$, $a_{14} = 0.2$, and $a_{15} = 1$.

Consideration of resource utilization: In model I, we explicitly consider resource utilization. A penalty cost is given for parts which do not or partially utilize machines in the same cell. The value of u_{ij} is determined by the designer. If high machine utilization is desired, then u_{ij} is set to a large value; otherwise it is set

to a lower value. By properly assigning values to u_{ij}, one can introduce flexibility to the model and form large loose cells or small tight cells. Once again, our approach is different from that found in most other papers which consider only the intercell movement cost and not resource under-utilization cost.

Consideration of practical constraints: Practical constraints are introduced in model I via constraints (4)–(6). Safety and technological considerations are provided in constraints (4) and (5) and upper/lower limits on cell size are imposed by constraint (6). To understand the safety/technological aspects, let us consider a situation where a heat treatment station and a forging station must be placed adjacent to each other for safety reason. If these stations are placed in S_2 (sets of machine pairs that must be located in the same cell), constraint (5) will ensure the two stations are grouped in the same cell. Similarly, painting and welding stations usually cannot be placed adjacently. Again, if this pair of stations is placed in S_1 (sets of machine pairs that cannot be located in the same cell), constraint (4) will enforce the two stations to be located in different cells. Some researchers (e.g. Song and Hitomi, 1992) impose such constraints indirectly by placing an upper limit on cell size and by trial and error seek solutions which satisfy the safety/technological constraints.

Imposition of upper limit on the number of cells: We do not assume that the number of cells is predetermined. In practice, designers usually do not know the number of cells that would yield the best CMS design. Thus, it is unrealistic to set the maximum number of cells to a given value (Kusiak, 1987). Hence, in our approach, the optimal number of cells is determined by the model. The optimal number of cells is equal to the initial assigned number of cells minus the number of empty cells which will be known from the solution. The initially assigned number of cells is any large number and can be set equal to the maximum number of machines in the problem. Of course, if the user wants to set the maximum number of cells to a specific value less than the optimal number determined by the model, it can be easily done and in fact, makes our model easier to solve. If set to a value larger than the optimal, the solution returned will have the optimal number of cells, i.e. less than what the user requested.

Consideration of part volume and intercell movements: Most existing approaches implicitly assume that the volume of parts manufactured is the same for each part. Unlike Adil et al. (1997) and many other researchers, we use part volume information in determining the intercell traffic. We also assume that the cost of intercell movement is the same between any pair of cells for a given part. However, this cost need not necessarily be the same for all parts. This is because the intercell movement cost depends on the material handling devices used which may be different for different parts. Although our model ignores the distance between cells, we feel this underestimation does not cause serious problems for two reasons. First, there is a large fixed cost involved in moving material between cells and the variable (distance) component is typically small. If this is not true in a problem, we can reduce the weight given to intercellular movement in the objective function. Second, the number of intercell moves is generally small. Furthermore, since the layout of cells is not known at this stage, it is impossible to determine the exact intercell movement cost.

Relaxing integer restrictions on key decision variables: Although no integer restrictions are placed on decision variable x_{ik}'s, it can be shown that there is at least one optimal solution to the cell formation model such that all x_{ik}'s take on binary values. This is discussed in Section 3.4. Further, a straightforward algorithm can easily identify this solution by inspection. Hence, integer restrictions are not necessary for x_{ik} variables but only for y_{jk} variables. As a result, our formulation has substantially fewer integer variables (equal to number of machines times number of cells) than other formulations in the literature. Hence, large problems which have thus far been solved heuristically, can now be solved optimally.

2.3. Dealing with multiple copies of machines

In practical applications, it is common to have multiple machines of a given type. This aspect is considered in many papers on CMS design. However the multiple machines of a given type are used to identify machine cells/part families so that the number of exceptional elements is minimized. In fact, some authors even suggest that we purchase additional machines of a given type and duplicate them in appropriate cells to minimize such exceptional elements and thereby reduce intercell material handling costs (for example, see

Seifoddini and Wolfe, 1986). We feel that such an approach which calls for high capital expenditures to minimize small operating costs is not practical. Moreover, the main reason we have multiple machines is to ensure that the system has adequate capacity to process all the parts. As a result, in our analysis of the CMS design problem, we first determine the required number of machines of each type to satisfy the capacity constraint and work with these numbers to design a CMS.

Like most existing cell formation models and algorithms, model I implicitly assumes that there is only one unit of each machine type. In practice, this is generally true for most machine types, but not all. There are typically a few machine types of which multiple copies are available. Of course, as discussed earlier, this information can be determined using the N_j formula and therefore is known prior to solving model I. If multiple units of one or more machine types are indeed available, we suggest that one of the following three approaches be used.

(1) Distribute jobs a priori to multiple copies of machines and then solve model I.
(2) Solve model I first assuming there is only one copy of each machine type and then deal with multiple copies of machines appropriately.
(3) Consider only machine types with single copy while solving model I and then deal with multiple copies of machines appropriately.

In the first approach, we require the user to assign a_{ij} values based on part volume and available capacity for each part that requires processing on a machine type with multiple copies. This is reasonable, since the user provides all other a_{ij} values (i.e. those pertaining to machine types of which there is a single copy). Ideally, it is better to set a_{ij}'s as decision variables, but it will introduce further nonlinearity in our already quadratic model I. Hence, we suggest that a_{ij} values be chosen according to plant manager's experience. For example, suppose that part types 1, 2, and 3 require processing on machine type 1 of which we have two copies. Further, let the production volume of parts 1, 2 and 3 be such that one unit of machine type 1 is adequate to process part types 1 and 3 and the other unit to process part type 2. Then each copy of machine type 1 is treated as a separate machine type say 1, 2. Two columns corresponding to these are added to the $[a_{ij}]$ matrix and a_{11}, a_{22}, and a_{31} values are set to 1. For this $[a_{ij}]$ matrix, we can obtain an optimal solution via model I.

In the second approach, we solve model I assuming that there is only one unit of each machine type. We then assign the available multiple copies of appropriate machine types to cells such that machine capacity requirements in each cell are satisfied and the intercell movement cost and resource under-utilization costs are minimized. This can be done rather easily.

In the third approach, we only consider the machine types with single copy and disregard all machine types with multiple copies while solving model I. After model I is solved, we then assign multiple copies of machines to cells such that intercell movement cost and resource under-utilization cost are minimized and machine capacity requirements in each cell are satisfied. The third approach is better than the second one because machine types with multiple copies are often bottleneck machines that are required by many parts. For such problems, the second approach may provide a solution with several large, loose cells. A large loose cell is one with large number of parts and machines and low resource utilization.

2.4. Model modification

Model I is nonlinear. Most nonlinear problems are usually much harder to solve optimally than linear problems. Hence, we reformulate it as a mixed-integer linear programming model by introducing a new set of variables z_{ijk} to replace the $x_{ik}y_{jk}$ product terms. The new model is shown below.

Model II (Mixed-integer problem)

$$\text{minimize} \quad \sum_{i=1}^{p}\sum_{j=1}^{q}\sum_{k=1}^{r} c_i v_i o_{ij} a_{ij} x_{ik} + \sum_{i=1}^{p}\sum_{j=1}^{q}\sum_{k=1}^{r} (u_{ij}(1-a_{ij}) - c_i v_i o_{ij} a_{ij}) z_{ijk} \tag{9}$$

subject to (2)–(8)

$$z_{ik} \leqslant x_{ik}, \qquad i = 1,2,\ldots,p,\ j = 1,2,\ldots,q,\ k = 1,2,\ldots,r, \tag{10}$$

$$z_{ijk} \leqslant y_{jk}, \qquad i = 1,2,\ldots,p,\ j = 1,2,\ldots,q,\ k = 1,2,\ldots,r, \tag{11}$$

$$x_{ik} + y_{jk} - z_{ijk} \leqslant 1, \qquad i = 1,2,\ldots,p,\ j = 1,2,\ldots,q,\ k = 1,2,\ldots,r, \tag{12}$$

$$0 \leqslant z_{ijk} \leqslant 1, \qquad i = 1,2,\ldots,p,\ j = 1,2,\ldots,q,\ k = 1,2,\ldots,r. \tag{13}$$

Model I without constraints (4)–(6) reduces to the clustering problem which can be relaxed into a minimum k-connected graph theory problem with extra constraints (King and Nakornchai, 1982). The latter is known to be NP-complete (Garey and Johnson, 1979). Hence, model I is NP-complete as well.

3. Solution algorithm – Benders' decomposition approach

Due to the combinatorial nature of the cell formation problem, more heuristic than optimal algorithms have been developed. Since the number of integer variables in our formulation is relatively smaller than those in others, we use Benders' decomposition approach to obtain optimal solutions for problems which have thus far been solved heuristically. Benders' decomposition approach was proposed to solve mixed-integer problems by Benders (1962). The basic idea is to iterate between two problems, a subproblem and a master problem, both of which are derived from the original mixed-integer problem. If the objective function value of the optimal solution to the master problem is greater than or equal to that of the subproblem, we terminate the iteration and obtain the optimal solution of the original mixed-integer problem. Otherwise, we add Benders' cuts, one at a time to the master problem, and solve it until the termination criteria are met.

3.1. Deriving primal/dual and master problems

To apply Benders' decomposition, we need to derive a primal problem (P) from model II by fixing the values of integer variables y_{jk}'s to yield a feasible solution and thus ignoring the constraints with only y_{jk}, i.e. (3)–(6)

(P) – *Primal problem*

$$\min \quad \sum_i \sum_j \sum_k c_i v_i o_{ij} a_{ij} x_{ik} + \sum_i \sum_j \sum_k \left(u_{ij} v_i (1 - a_{ij}) - c_i v_i o_{ij} a_{ij}\right) z_{ijk} \tag{14}$$

$$\text{s.t.} \quad z_{ijk} - x_{ik} \leqslant 0 \qquad \forall\, i,j,k, \tag{15}$$

$$z_{ijk} \leqslant y_{jk} \qquad \forall\, i,j,k, \tag{16}$$

$$x_{ik} - z_{ijk} \leqslant 1 - y_{jk} \qquad \forall\, i,j,k \tag{17}$$

and (2), (7), (13).

Next, we derive the dual problem (D) from the primal.

(D) – *Dual problem*

$$\max \quad \sum_i \sum_j \sum_k 0 l_{ijk} + \sum_i \sum_j \sum_k (-y_{jk}) m_{ijk} + \sum_i \sum_j \sum_k (y_{jk} - 1) n_{ijk} + \sum_i s_i \tag{18}$$

$$\text{s.t.} \quad \sum_j l_{ijk} + 0 m_{ijk} + (-1) \sum_j n_{ijk} + s_i \leqslant \sum_j (c_i v_i a_{ij} o_{ij}) \quad \forall\, i,k, \tag{19}$$

$$-l_{ijk} - m_{ijk} + n_{ijk} + 0 s_i \leqslant u_{ij} v_i (1 - a_{ij}) - c_i v_i o_{ij} a_{ij} \quad \forall\, i,j,k, \tag{20}$$

$$l_{ijk}, m_{ijk}, n_{ijk} \geqslant 0 \quad \forall\, i,j,k, \tag{21}$$

$$s_i \text{ free} \qquad \forall\, i. \tag{22}$$

The dual problem is linear with four sets of real variables l_{ijk}, m_{ijk}, n_{ijk}, and s_i corresponding to constraint sets (15)–(17) and (2), respectively. Note that constraints (2) and (15) force $x_{ik} \leqslant 1$ and $z_{ijk} \leqslant 1$. We therefore do not need dual variables corresponding to the upper bounds in constraints (7) and (13). Last, we formulate the master problem.

(M) – *Master Problem*

$$\begin{aligned} &\text{minimize} && Z \\ &\text{subject to} && (3)\text{–}(6), (8), \\ & && Z \geqslant \sum_j \sum_k \left(\sum_i (\tilde{n}_{ijk} - \tilde{m}_{ijk}) \right) y_{jk} + \sum_i \tilde{s}_i - \sum_i \sum_j \sum_k \tilde{n}_{ijk}, \\ & && Z \geqslant 0. \end{aligned} \tag{23}$$

$\tilde{m}_{ijk}, \tilde{n}_{ijk}, \tilde{s}_i$ in (M) are optimal values of the variables m_{ijk}, n_{ijk}, s_i obtained by solving (D). Although the master problem is a mixed-integer problem, it can be easily converted to a pure 0-1 integer problem by rewriting the only continuous variable Z in the master problem as a function of additional binary variables to make it a pure integer problem as discussed in most introductory Operations Research textbooks, for example Taha (1992). Only the constraints with y_{jk} terms in model II are considered in the above model; constraint (23) is Benders' cut which is derived from the objective function of the dual problem.

3.2. Benders' decomposition algorithm

Formally, Benders' decomposition algorithm is as follows:

Algorithm 1

Step 0. Set LB $= -\infty$, UB $= +\infty$, $y_{j1} = 1$ for $\forall j$ and $y_{jk} = 0$ for $\forall j \neq 1, k$ in (D).

Step 1. Solve (D) and set the current optimal solution value $\hat{z} =$ UB if $\hat{z} <$ UB. If LB $\geqslant$ UB, stop. We have the optimal solution of the original mixed-integer problem. Otherwise go to Step 2.

Step 2. Update (M) by adding a new constraint (23) to it, solve the model and obtain optimal solution z^* and optimal values for integer variables y_{jk}. Set LB $= z^*$ and update y_{jk}s in the dual problem. Go to Step 1.

Further interpretation of the algorithm is provided below. First, solution of the master problem may yield no feasible solution for the original problem. In that case we need to generate an extreme ray and include it in the master problem. However, in our model we can guarantee that the master problem generates only feasible y_{jk}'s, because we included constraints with y_{jk} terms from model I (i.e. constraints (3)–(6)) in the master problem.

Second, the solution we get from master problem is a lower bound on the original problem's objective function value. For each iteration, we add a stronger cut to the master problem to improve the lower bound. If the lower bound is greater than or equal to the optimal solution in (D), we get optimal solution of the original problem. Of course, the master problem is always feasible since the original problem is feasible.

Last, the computational efficiency of the algorithm mainly depends on the number of iterations required and time needed to solve the master problem in each iteration. Although the master problem can be converted into a pure integer programming problem, it still takes a considerable effort to solve it optimally in

each iteration. Therefore, a variant of Benders' decomposition presented in Geoffrion and Graves (1974) is used. Instead of solving (M) optimally in each iteration, we solve only until the first feasible solution is found with value below UB $-\epsilon$, $\epsilon > 0$, where UB is the best solution of (D) among the iterations, and ϵ is an allowed error margin. Of course, the master problem has to be modified as shown below.

3.3. The modified Benders' decomposition algorithm

The master problem can be modified as

(M′) – *Modified master problem*

$$\min \quad \sum_i \sum_j \sum_k -y_{jk} m_{ijk} + \sum_i \sum_j \sum_k (y_{jk} - 1) n_{ijk} + \sum_i s_i \tag{24}$$

s.t. (3)–(6), (8) and

$$\sum_i s_i - \sum_i \sum_j \sum_k \tilde{n}_{jik} + \sum_j \sum_k \left(\sum_i (\tilde{n}_{ijk} - \tilde{m}_{ijk}) \right) y_{jk} \leqslant (\mathrm{UB} - \epsilon). \tag{25}$$

(M′) is a pure integer problem. The objective function is adopted from the dual objective. However, it can be arbitrarily assigned without affecting the solution. Also, it can be proved that if $\epsilon > 0$, constraint (25) will ensure that the same integer solution will not be regenerated. The modified algorithm is stated below.

Algorithm 2

Step 0. Set UB $= +\infty$, $y_{j1} = 1$ for $\forall j$ and $y_{jk} = 0$ for $\forall j \neq 1, k$ in (D).

Step 1. Solve (D); set optimal solution $= \hat{z}$ and update UB $= \min\{\mathrm{UB}, \hat{z}\}$.

Step 2. Update (M′) by changing the UB value if necessary and adding a new constraint (25). Obtain a feasible solution to (M′) and update y_{jk} values. Go to Step 1. If (M′) is infeasible, terminate. We get an ϵ-optimal solution.

3.4. On solving the dual

While the simplex algorithm or interior point method can be used to solve the dual problem, we can take advantage of the problem structure and solve the dual more efficiently. This is especially important for large problems (with thousands of variables and constraints) that have to be solved several times before we get the optimal solution. The procedure we develop for solving the dual problem is based on the following propositions.

Proposition 1. *The primal problem* (P) *is always feasible and bounded for given* y_{jk}*'s satisfying constraint sets (3)–(6) and (8).*

Proof. The primal problem (P) is derived from model II which is equivalent to model I with given y_{jk}'s satisfying constraint sets (3)–(6) and (8). Thus, if model I is feasible and bounded, the primal problem will also be feasible and bounded.

Model I is formulated to find the minimum cost related to intercell movement and resource under-utilization. It is easy to see that model I is always feasible if the practical constraints are not violated, i.e. constraint sets (3)–(6) and (8) are satisfied. This is because each part can be arbitrarily assigned to any part family and each machine can be randomly assigned to any machine group. In addition, since the cost related to the part and machine assignments are finite and nonnegative, any feasible solution to model I must be bounded. Therefore, we can conclude that the primal problem is also feasible and bounded for given y_{jk}'s satisfying constraint sets (3)–(6) and (8). □

Corollary 1. *The dual problem* (D) *is feasible and bounded and has at least one optimal solution.*

Proposition 2. *There exists at least one set of optimal variables in the dual such that* $\tilde{m}_{ijk} = 0$, if $y_{jk} = 1$ and $\tilde{n}_{ijk} = 0$, if $y_{jk} = 1$, *for all* i, j, k.

Proof. From corollary 1, we know the dual has at least one optimal solution. Suppose an optimal variable set exists with some $\tilde{m}_{ijk}s > 0$ when $y_{jk} = 1$ and some $\tilde{n}_{ijk}s > 0$ when $y_{jk} = 0$. Then, another optimal variable set with $\tilde{m}_{ijk} = 0$ when $y_{jk} = 1$ and $\tilde{n}_{ijk} = 0$ when $y_{jk} = 0 (\forall i, j, k)$ can be derived as shown below.

From (18) and (19), if an optimal variable set exists with some $\tilde{n}_{ijk}s > 0$ when $y_{jk} = 0$, then we can decrease the value of $\tilde{n}_{ijk}$ to 0 and s_i by $\tilde{n}_{ijk}$ and retain optimality as well as feasibility. Notice that reducing the value of $\tilde{n}_{ijk}s$ to 0 in (20) will not violate feasibility, and therefore, the optimal values of l_{ijk}, $\tilde{m}_{ijk}$ need not be changed.

In addition, if an optimal variables set exists with some $\tilde{m}_{ijk}s > 0$ when $y_{jk} = 1$, it is easy to see from (18), (19) and (20) that we can reduce the value of $\tilde{m}_{ijk}s$ to 0, correspondingly increase the value of $\tilde{l}_{ijk}$ and then reduce the value of $\bar{s}_i$. Because $\tilde{l}_{ijk}$, m_{ijk} and s_i have coefficients of 0, −1 and 1 in the objective function, respectively, the dual problem will remain feasible and optimal. Therefore, we can conclude that there exists at least one set of optimal variables in the dual such that $\tilde{m}_{ijk} = 0$, if $y_{jk} = 1$ and $\tilde{n}_{ijk} = 0$, if $y_{jk} = 1$, for all i, j, k. □

Proposition 3. *The optimal solution of the dual problem can be found by inspection.*

Proof. From Proposition 2, we know that the dual objective function value is equal to $\sum s_i$. To make s_i as large as possible, we need to maximize the value of n_{ijk} and minimize the value of l_{ijk}, due to constraint (19).

Algorithm 3 provided below shows how the optimal solution can be obtained by inspection.

Algorithm 3

Step 0. Set $i = j = 1$.

Step 1. For $k = 1, 2, \ldots, r$, let $\text{RHS}_{ijk} = u_{ij}v_i(1 - a_{ij}) - c_iv_io_{ij}a_{ij}$.
$\text{RHS}_{ijk} \geqslant 0$ and $y_{jk} = 1$, set $\tilde{l}_{ijk} = 0$, $\tilde{m}_{ijk} = 0$, $\tilde{n}_{ijk} = \text{RHS}_{ijk}$.
If $\text{RHS}_{ijk} \geqslant 0$ and $y_{jk} = 0$, set $\tilde{l}_{ijk} = 0$, $\tilde{m}_{ijk} = 0$, $\tilde{n}_{ijk} = 0$.
If $\text{RHS}_{ijk} < 0$ and $y_{jk} = 1$, set $\tilde{l}_{ijk} = -\text{RHS}_{ijk}$, $\tilde{m}_{ijk} = 0$, $\tilde{n}_{ijk} = 0$.
If $\text{RHS}_{ijk} < 0$ and $y_{jk} = 0$, set $\tilde{l}_{ijk} = 0$, $\tilde{m}_{ijk} = -\text{RHS}_{ijk}$, $\tilde{n}_{ijk} = 0$.

Step 2. Set $j = j + 1$. If $j \leqslant nm$, go to Step 1.

Step 3. Set $i = i + 1$. If $i \leqslant np$, set $j = 1$ and go to step 1.

Step 4. For $i = 1, 2, \ldots, p$, set $\tilde{s}_i = \min\{\sum_j (c_iv_ia_{ij}o_{ij}) + \sum_j \tilde{n}_{ijk} - \sum_j \tilde{l}_{ijk}, k = 1, 2, \ldots, r\}$. The optimal solution of the dual Z is then $\sum_i \tilde{s}_i$. STOP.

To verify that the dual solution obtained using Algorithm 3 is optimal, we apply the property of strong duality to find the corresponding primal solution and then show the primal and dual solutions have the same value as discussed below.

Suppose $y_{jk} = 1$ for some $j = j^*$ and $k = k^*$. Then based on Algorithm 3, dual constraint (19) corresponding to $k = k^*$ can be simplified as $s_i \leqslant v_i \sum_j u_{ij}(1 - a_{ij}) \forall i$. For all other k, i.e. $k \neq k^*$, (19) can be simplified as $s_i \leqslant c_iv_i \sum_j a_{ij}o_{ij}$ $\forall i$. Two conditions arise. Either

$$v_i \sum_j u_{ij}(1 - a_{ij}) \leqslant c_iv_i \sum_j a_{ij}o_{ij} \quad \text{or} \quad v_i \sum_j u_{ij}(1 - a_{ij}) > c_iv_i \sum_j a_{ij}o_{ij}.$$

For the first case, using complementary slackness property it can be shown that the corresponding primal variable $x_{ik^*} = 1$ and $x_{ik} = 0$ for all $k \neq k^*$. Since $z_{ijk} = x_{ik}y_{jk}$ and the value of y_{jk}'s is known, the value of z_{ijk} variables and hence the objective function value of the primal can be easily obtained. Furthermore, this

can be shown to be equal to Z – the objective function value of the dual. Hence, the x_{ik} values as determined above and $\tilde{s}_i$, $\tilde{l}_{ijk}$, $\tilde{m}_{ijk}$, $\tilde{n}_{ijk}$ values as obtained in Algorithm 3 are optimal to the respective problems. It is easy to show the same holds even for the second case, i.e., $v_i \sum_j u_{ij}(1 - a_{ij}) > c_i v_i \sum_j a_{ij} o_{ij}$. Any one $k \neq k^*$ can be arbitrarily fixed at 1 and the others including x_{ik^*} set to 0. Thus, in each case, a feasible solution to the primal such that $x_{ik} = 1$ for only one k can be obtained. From the structure of the dual problem, we notice that it is likely to have multiple optimal solutions. However, our interest here is to find one solution and then use it in the master problem. Algorithm 3 is adequate for this purpose.

3.5. Interesting phenomenon of model I

In Section 3.3, we showed how the dual problem can be solved without using the simplex method. In this section, we prove another important result that eliminates a large number of integer restrictions in model I and thereby makes it extremely efficient.

Remark 1. When y_{jk},s in model I are fixed so constraints (3)–(6) are satisfied, the resulting model will have at least one optimal solution in which x_{ik}'s are integers. Further, such a solution can be found by inspection.

Proof. Observe that model I with fixed y_{jk}'s can be rewritten as

$$\text{minimize} \quad \sum_i \sum_k \left(\sum_j \left(c_i v_i o_{ij} a_{ij} (1 - y_{jk}) + u_{ij} v_i (1 - a_{ij}) y_{jk} \right) \right) x_{ik} \tag{26}$$

subject to (2) and (7).

The above is a half assignment problem. It is well known that in any basic feasible solution (including the optimal one), each constraint will have at most one basic variable. Since the coefficient of each x_{ik} is 1 in constraint (2), the basic variable in each row will take on a value of 1 even in the final solution. Of course, the remaining variables will take a value of 0. Thus it can be guaranteed that there is at least one optimal solution for the above model such that x_{ik}'s are 0,1 integers. □

To find an optimal solution, we use the following algorithm.

Algorithm 4

Step 0. Set $i = 1$.

Step 1. Since y_{jk} is fixed, combine all the terms in (1) and select the smallest coefficient of $x_{ik}, k = 1, 2, \ldots, r$. Set the corresponding x_{ik} to 1 and all other x_{ik}'s to 0. Set $i = i + 1$.

Step 2. If $i \leqslant p$, go to Step 1. Otherwise STOP.

Remark 2. When y_{jk}'s are fixed so as to satisfy constraints (3)–(6), the solution to model II can be obtained by inspection.

Proof. Using Algorithm 4, we can determine values of the x_{ik} variables. Because the constraints in model II satisfy the condition $z_{ijk} = x_{ik} y_{jk}$, the y_{jk}'s are known, and model II is equivalent to model I, the value of z_{ijk} is given by the product $x_{ik} y_{jk}$. □

4. Numerical example

In this section, an example is provided to show how Benders' decomposition works for cell formation problems. The data of the example in Heragu and Kakuturi (1997) are modified for this purpose. This

example consists of 20 part types and seven machine types. The related costs, alternate routings information, and practical constraints are all considered and assumed. The data required are provided below.

1. Part–machine matrix (**A**) – including the information of part operation requirements, alternate routings, and number of visits required.
2. Material handling cost matrix (**C**).
3. Operation time matrix (**T**) – containing the processing time each part i requires on machine j.
4. Part quantity matrix (**V**) – indicating number of parts per batch.
5. Machine relations matrix (**R**)
6. Machine cost matrix ($\mathbf{B_J}$)
7. Resource under-utilization cost for each part–machine pair (u_{ij}) is 0.2.
8. The available budget is 20 units.
9. Available operation time on each machine type j (t_j) is 50 units.
10. The minimum and maximum number of machines in each cell is 0 and 7, respectively.

Machine Number

A =

Part No.	1	2	3	4	5	6	7
1	1	1*	0	0.8	0.2	0	0
2	0	0	0	0	0	1	1
3	0	0	0	0.85	0.15	0	0
4	1	1	0	1	0	1	0
5	1	0	0	1	0	0	0
6	0	0	1	0	0	1	1
7	0	1	0	1*	0	0	0
8	0	0	0	0	1	1	0
9	0.1	0.9	1	1	0	0	0
10	1	1	0	0	0	0	0
11	0	0	0	0	0	1	1
12	0	0	0	0	1	1	0
13	0	0	1	0	0	0	1
14	0	0	0	0	1	0	0
15	1*	0	0	0	1	1	0
16	1	0	0	0	1	1	0
17	0	0	0	0	1	1	0
18	0.8	0.2	0	1	0	0	0
19	1	1	0	0	0	0	0
20	0	0	1	0	0	0	1

C =

1
0.5
1
0.3
1
1
1
1
1
1
1.2
0.2
1
1
1
1
1
1
1
1

*Indicates two non-consecutive visits are required on the machines.

Machine Type

T = Part No.	1	2	3	4	5	6	7	V =
1	2	1	0	1	2	0	0	2
2	0	0	0	0	0	3	1	3
3	0	0	0	2	1	0	0	2
4	5	2	0	1	0	1	0	2
5	2	0	0	1	0	0	0	1
6	0	0	1	0	0	1	0.8	1
7	0	1	0	3	0	0	0	2
8	0	0	0	0	1	1	0	3
9	5	2	1	1	0	0	0	2
10	4	1	0	0	0	0	0	2
11	0	0	0	0	0	1	0.5	2
12	0	0	0	0	0.5	1	0	2
13	0	0	1	0	0	0	1	4
14	0	0	0	0	1	0	0	2
15	3	0	0	0	1	1	0	1
16	1	0	0	0	1	4	0	2
17	0	0	0	0	2	1	0	2
18	4	0.2	0	1	0	0	0	2
19	3	1	0	0	0	0	0	3
20	0	0	1	0	0	0	2	2

R = M/C Type	1	2	3	4	5	6	7	B_J =
1	O	O	O	O	O	O	O	1
2	O	O	X	A	O	O	O	0.5
3	O	X	O	O	O	O	O	1
4	O	A	O	O	O	O	O	0.3
5	O	O	O	O	O	O	O	1
6	O	O	O	O	O	O	O	1
7	O	O	O	O	O	O	O	1

The first row of **A** matrix indicates that 80% of part 1 follows the routing sequence 1-2-4-2 and 20% follows 1-2-5-2. It also indicates that part 1 requires two nonconsecutive operations on machine 2. The same interpretation can be made for all other rows of the **A** matrix. In machine relations matrix **R**, an 'A' indicates the corresponding two machines have to be grouped in the same cell, an 'X' indicates the two machines cannot be grouped in the same cell. An 'O' indicates there is no special location restriction for the machine pair. Therefore, for this problem, the **R** matrix shows that machines 2 and 4 have to be in the same cell; machines 2 and 3 cannot be in the same cell, and no restriction is applied to all other machine pairs.

To solve this problem, first, the N_j formula is used to find the number of units required for each machine type so as to satisfy capacity constraint. Table 1 shows the total processing time and the number of units required for each machine.

From Table 1, it can be seen that only one unit of each machine type is required. Also since the total machine purchase cost $-12 \cdot 4$ units is less than available budget 20, the budget constraint is satisfied. The next step is to solve model II. The maximum number of cells allowed, C_u is assigned a value of 4. Therefore model II consists of 640 real variables and 28 integer variables. After the dual and master problems are formulated, Algorithms 1 and 2 are used to solve the problem. The error margin ϵ for Algorithm 2 is set as 1. Therefore, the largest possible difference between the solution found by the modified Benders, decomposition algorithm and the optimal solution is 1. The solutions obtained using Algorithms 1 and 2 are identical and equal to 15.54. The solution includes three machine cells and corresponding part families. The fourth cell is a dummy cell which contains no parts or machines and our initial choice of $C_u = 4$ is justified. If the solution had four cells, then we would have to increase C_u by 1 and resolve model II until the number of cells in the solution is one less than the value assigned to C_u. The optimal solution of this cell formation problem and the corresponding part–machine matrix are provided below.

Cell 1:	Part family #1	Part 1, 3, 4, 5, 7, 9, 10, 18, 19
	Machine group #1	Machine 1, 2, 4
Cell 2:	Part family #2	Part 8, 11, 12, 14, 15, 16, 17
	Machine group #2	Machine 5, 6
Cell 3:	Part family #3	Part 2, 6, 13, 20
	Machine group #3	Machine 3, 7

Use of Algorithm 2 to solve the cell formation problem: Algorithm 2 is not an optimal algorithm. However, if the error margin is set to a small enough number, the solution obtained using Algorithm 2 is optimal (as shown in this example). In fact it can be proved that, for a totally unimodular case (i.e. all coefficients of the model are either −1, 0, 1), if the error margin is set to a fractional number, the solution obtained using Algorithm 2 is optimal.

Table 1
Results using N_j formula

Machine number j	Available operation time T_j	Total required processing time $\sum_j v_i a_{ij} t_{ij}$	Number of units required N_j
1	50	45.40	1
2	50	13.68	1
3	50	9.00	1
4	50	18.00	1
5	50	14.10	1
6	50	30.00	1
7	50	12.80	1

Discussion of maximum number of cells allowed, r: In this example, we assigned maximum number of cells allowed, r equal to 4 and obtained solution indicating three machine cells and one dummy cell. Therefore we can conclude that the optimal cell number, r^* is equal to 3. For any value of r greater than or equal to 3, the solution obtained using Algorithm 1 is an optimal solution.

Discussion of multiple machines: To demonstrate our approach of dealing with multiple machines, we change the available operation time of each machine type from 50 to 42 in the previous example. Now, from Table 1 we know the total required processing time on machine 1 is 45.4 which is large than 42. Thus an additional unit of machine 1 is required. In addition, the total machine purchasing cost now is 14.4. It is still less than available budget 20. Hence, the budget constraint is satisfied.

To deal with multiple machines, the first approach mentioned in Section 2.3 that distributes jobs to machine type 1 before solving model II is used here. Since we found from previous result that parts 15 and 16 require intercell movements due to machine 1, we assign the additional unit of machine 1 to these two parts.

Machine Number

A = Part No.	1	2	4	5	6	3	7
1	1	1*	0.8	0.2	0	0	0
3	0	0	0.85	0.15	0	0	0
4	1	1	1	0	1	0	0
5	1	0	1	0	0	0	0
7	0	1	1*	0	0	0	0
9	0.1	0.9	1	0	0	1	0
10	1	1	0	0	0	0	0
18	0.8	0.2	1	0	0	0	0
19	1	1	0	0	0	0	0
8	0	0	0	1	1	0	0
11	0	0	0	0	1	0	1
12	0	0	0	1	1	0	0
14	0	0	0	1	0	0	0
15	1*	0	0	1	1	0	0
16	1	0	0	1	1	0	0
17	0	0	0	1	1	0	0
20	0	0	0	0	0	1	1
2	0	0	0	0	1	0	1
6	0	0	0	0	1	1	1
13	0	0	0	0	0	1	1

* Indicates two non-consecutive visits are required on the machines.

Before solving model II for this revised problem, we need to ensure the capacities of machine 1 and 1′ are adequate. The revised total operation time on machine 1 is 40.2 and on machine 1′ is 5 and both are less than available machine operation time. Thus the capacity constraint is satisfied. To solve model II, the solution procedure described previously for the original example is used. The optimal solution is found equal to 13.6. The optimal solution of the cell formation problem is provided below.

Cell 1:	Part family #1	Part 1, 3, 4, 5, 7, 9, 10, 18, 19
	Machine group #1	Machine 1, 2, 4
Cell 2:	Part family #2	Part 8, 11, 12, 14, 15, 16, 17
	Machine group #2	Machine 1′, 5, 6
Cell 3:	Part family #3	Part 2, 6, 13, 20
	Machine group #3	Machine 3, 7

5. Computational performance

In this section, the computational performance of Algorithms 1 and 2 is analyzed using four test problems. OSL software was used on an IBM RISC 6000 machine to solve these problems. The first example is a test problem with five parts, four machines to illustrate use of Benders' decomposition for traditional cluster analysis discussed in most papers. The second example is to demonstrate the approach we propose for solving practical cell formulation problems. The data of the second problem are modified from Heragu and Kakuturi (1997) and all of the cost and time related values are assumed. The third and fourth examples are two classical cell formation problems which are widely discussed and adapted from Askin and Chiu (1990) and Burbidge (1975), respectively. The problem size is different in these examples and the degree of difficulty is in increasing order.

A summary of the results for the four test problems is provided in Table 2. As mentioned previously, the number of integer variables is a product of machines and cells. From the table we notice that: (1) the CPU time increases dramatically when the number of integer variables increases; (2) modified Benders' decomposition approach can solve the problem much faster than the original algorithm, and (3) increasing the value of error margin, ϵ, can speed up the performance but the solution quality will decrease. Spending a few hours to solve a cell design problem optimally or near-optimally is considered reasonable because the cell design problem, unlike the planning problem, is not a routine job that is required to be solved frequently. Our experience with industrial experts shows that designers would rather spend more time to get

Table 2
Summary results of test problems 1–4

	Problem 1	Problem 2		Problem 3		Problem 4		
No. of parts	5	20		24		43		
No. of m/c types	4	5		14		16		
No. of m/cs	4	7		14		21		
No. of cells	2	3		3		4		
Approach used	Alg. 3	Alg. 3	Alg. 4	Alg. 3	Alg. 4	Alg. 4		
ϵ value, if applicable	–	–	$\epsilon = 1$	–	$\epsilon = 1$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 1$
CPU time	1.1 s	5 min	86 s	197 min	10 min	25 min	91 min	631 min
No. of iterations	10	59	60	118	97	102	154	560
Optimality	yes	yes	yes	yes	yes	no	yes	yes

an optimal or near optimal solution instead of using heuristic approach to get an inferior solution for an important system design problem that has ramifications on planning problems.

6. Conclusion

In this paper, a mathematical model was presented to incorporate three critical aspects: (1) resource utilization, (2) alternate routings, and (3) practical constraints while designing CMSs. Unlike most other papers, our model does not require us to enforce integer restrictions on a key variable set, thereby enabling us to formulate a mixed-integer problem with much fewer integer variables. The algorithms used to solve the model were coded using OSL, and optimal results were obtained for all but one example problem.

A cell decomposition approach has been developed to solve even larger problems. The cell decomposition approach, first, analyzes the part–machine relations; second, it decomposes the original system to several subsystems using existing techniques, and then uses the approach discussed in this paper to solve each subsystem. Details are covered in Chen and Heragu (1995).

References

Adil, G.K., Rajamani, D., Strong, D., 1997. Assignment allocation and simulated annealing algorithms for cell formation. IIE Transactions 29 (1), 53–67.

Askin, R.G., Chiu, K.S., 1990. A graph partitioning procedure for machine assignment and cell formation in group technology. International Journal of Production Research 28 (8), 1555–1572.

Askin, R.G., Subramanian, S.P., 1987. A cost based heuristic for GT configuration. International Journal of Production Research 25, 101–113.

Ballakur, A., Steudel, H.J., 1987. A within cell utilization based heuristic for designing cellular manufacturing systems. International Journal of Production Research 25 (5), 639–665.

Benders, J.F., 1962. Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik 4, 238–252.

Burbidge, J.L., 1975. The Introduction of Group Technology, Wiley, New York.

Chan, H.M., Milner, D.A., 1982, Direct clustering algorithm for group formation in cellular manufacturing. Journal of Manufacturing Systems 1, 64–76.

Chandrasekaran, M.P., Rajagopalan, R., 1986. An ideal seed non-hierarchical clustering algorithm for cellular manufacturing. International Journal of Production Research 24 (2), 451–464.

Chen, J., Heragu, S.S., 1995. Cellular manufacturing systems design: a cell decomposition approach for solving large scale real world problems, DSES Technical Report #37-95-437, Rensselaer Polytechnic Institute, Troy, NY 12180.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability, Freeman, San Francisco, CA.

Geoffrion, A.M., Graves, G.W., 1974. Multicommodity distribution system design by Benders decomposition. Management Science 20 (5), 822–844.

Grznar, J., Mehrez, A., Offodile, O.F., 1994. Formulation of the machine cell grouping problem with capacity and material movement constraints. Journal of Manufacturing Systems 13 (4), 241–260.

Heragu, S.S., 1994. Group technology and cellular manufacturing. IEEE Transactions on Systems, Man, and Cybernetics 24 (2), 203–215.

Heragu, S.S., Gupta, Y.P., 1994. A heuristic for designing cellular manufacturing facilities. International Journal of Production Research 32 (1), 125–140.

Heragu, S.S., Kakuturi, S.R., 1997. Grouping and placement of machine cells. IIE Transactions 29(7), 561–571.

Kang, S., Wemmerlov, U., 1993. A work load-oriented heuristic methodology for manufacturing cell formation allowing reallocation of operations. European Journal of Operational research 69 (3), 292–311.

King, J.R., 1980. Machine-component grouping in production flow analysis: an approach using a rank order clustering algorithm. International Journal of Production Research 18, 213–219.

King, J.R., Nakornchai, V., 1982. Machine-component group formation in group technology: review and extension. International Journal of Production Research 20 (2), 117–133.

Kumar, C.S., Chandrasekharan, M.P., 1990. Group efficacy: a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. International Journal of Production Research 28 (2), 233–243.

McAuley, J., 1972. Machine grouping for efficient production. The Production Engineer 51 (2), 53–57.
Offodile, O.F., Mehrez, A., Grznar, J., 1994. Cellular manufacturing: a taxonomic review framework. Journal of Manufacturing Systems 13 (3), 196–220.
Seifoddini, H., Wolfe, P.M., 1986. Application of the similarity coefficient method in group technology. IIE Transactions 18, 271–277.
Singh, N., 1993. Design of cellular manufacturing systems. European Journal of Operational Research 69 (3), 284–291.
Song, S., Hitomi, K., 1992. GT cell formation for minimizing the intercell parts flow. International Journal of Production Research 30 (12), 2737–2753.
Srinivasan, G., Narendran, T.T., 1991. Graphics – a nonhierarchical clustering algorithm for group technology. International Journal of Production Research 29, 463–478.
Taha, H.A., 1992. Operations Research: An Introduction, Macmillan, New York.
Vakharia, A.J., 1986. Methods of cell formation in group technology: a framework for evaluation. Journal of Operation Management 6 (3), 257–271.
Wemmerlov, U., Hyer, N.L., 1986, Procedures for the part family machine group identification problem in cellular manufacturing. Journal of Operations Management 6 (2), 125–147.
Wemmerlov, U., Hyer, N.L., 1989. Cellular manufacturing in the U.S. industry: a survey of users. International Journal of Production Research 27 (9), 1511–1530.