INT. J. PROD. RES., 1995, VOL. 33, NO. 4, 893–913

Machine-part family formation with the adaptive resonance theory paradigm

C. DAGLI†* and R. HUGGAHALLI‡

The ART1 neural network paradigm employs a heuristic where new vectors are compared with group representative vectors for classification. ART1 is adapted for the cell formation problem by reordering input vectors and by using a better representative vector. This is validated with both test cases studied in literaure as well as synthetic matrices. Algorithmns for effective use of ART1 are proposed. This approach is observed to produce sufficiently accurate results and is therefore promising in both speed and functionality. For the automatic generation of an optimal family formation solution a decision support system can be integrated with ART1.

1. Introduction

Group technology (GT) seeks to identify and exploit similarities of product design and manufacturing processes throughout the manufacturing cycle (Hyer 1984, Hyer and Wemmerlov 1989, Groover 1987). Implementation of GT requires the recognition of resemblances between parts that are being manufactured. There are three general ways of identifying common design and manufacturing attributes (Groover 1987). In 'visual inspection' (Banerjee and Redford 1982), physical design attributes are compared manually and therefore it is the least accurate and most strenuous way of identifying groups. Machine vision, a research area of current interest, is focused towards making visual inspection more efficient. In 'classification and coding', each part's attributes are comprehensively coded, and a classification is performed based on these codes. The main problems with this method are those of finding an optimal coding scheme for the given situation and the time required for code assignment and comparison. The 'production flow analysis' method simplifies recognition by considering only the manufacturing attributes derived from route sheets. A number of other analysis techniques called machine-part family/cell formation techniques based on manufacturing attributes have been developed since production flows analysis was proposed by Burbridge (1977). The subject of this paper is a new and an efficient analysis method directed towards machine-part family formation.

1.1. *The machine-part family formation problem*

The assignment of a similar group of parts to a cell of machines having common processing characteristics results in 'cellular manufacturing'. Machine-part family

Revision received March 1994.

†Engineering Management Department, University of Missouri-Rolla, Rolla MO-65401, USA.

‡Department of Electrical and Computer Engineering, University of South Carolina, Columbia SC-29208, USA.

*To whom correspondence should be addressed.

0020–7543/95 $10·00 © 1995 Taylor & Francis Ltd.

formation is necessary for the implementation of cellular manufacturing. The machine-part matrix is a binary matrix representing the relations between machines and parts. If the elements of the matrix are given by a_{ij} where i is the machine number and j is the part number, then $a_{ij}=1$ indicates that part j is processed at machine i or conversely that machine i processes part j. It is the starting point for the development and implementation of most cell formation techniques. Hence, such a matrix must be made available from process plans and routeing sheets. The objective of most machine-part family formation algorithms is to rearrange rows and columns of the machine-part matrix in such a way that the resultant matrix has all the '1' elements in the matrix clustered in groups in a block diagonal fashion. Each cluster in the rearranged form of the matrix would then indicate a part group and the corresponding machine group. During this clustering process, 'exceptional' parts are generated when a part has to be processed in more than one cluster or machine cell. In order to reduce the inter-cellular movement of such parts, machines are duplicated, thus in effect achieving a trade-off. Numerous small examples that clearly demonstrate this concept have been presented in literature (Huggahalli 1991, Dagli and Huggahalli 1991). In practical situations, very large matrices can be encountered. Several alternative solutions must then be evaluated to obtain optimal machine-part families. Also, there are several performance criteria such as total material handling cost, average machine utilization, average-set-up time and manufacturing lead time for which a solution must be evaluated. Hence, the need for techniques to obtain machine-part clusters as efficiently as possible and for a set of heuristics to obtain at least close-to-optimal solutions.

1.2. *Existing approaches*

To obtain an optimal solution in terms of a minimum cost, it is necessary for most algorithms to be iteratively implemented. Serial algorithms proposed in literature take significant time for each iteration. Highly parallel systems such as neural networks can be used, to be able to observe and compare different solutions in a short time. In light of the computational deficiencies of existing algorithms, even when implemented on hardware, the unmatched speed of neural networks must be considered for machine-part family formation problems where large amounts of information must be processed in a short time. A detailed review of the approaches proposed thus far has been presented in King and Nakornchai (1982), Haggahalli (1991) and Dagli and Huggahalli (1991).

Self-organizing or unsupervised neural networks have the significant advantage of minimal computation times. Each pattern that represents a part or a machine need only be applied once for classification. Hence, the time complexity is proportional to the total number of patterns being classified. Besides time complexity, connectivity (the numbr of network connections) is also important. Moon proposed an interactive activation and competition model (Moon 1990) in which a neural network is first established based on part similarities, machine similarities and the machine-part relations as obtained from the machine-part matrix. Then parts and machines are grouped using the neural network which activates uniquely for each part group and machine group. Since this model requires the computation of similarities between every pair of parts and machines and very little information is known about the performance of the model, other methods of application of neural networks to the cell formation problem must be explored to achieve better efficiency. Connectivity of such a neural network can also be very high. Two other preliminary works on the application of neural networks to cellular manufacturing have also been published recently (Kaparthi and Suresh 1992, Malave and Ramachandran 1991). The approaches are fundamentally similar to the one presented in this paper and the classification improvements suggested in the following sections are very much applicable.

1.3. *Outline of the ART1 approach*

The paper proposes the use of a neural network paradigm called the binary Adaptive Resonance Theory paradigm (ART1) for machine-part family formation. The essential function of an ART1 network is to classify a set of binary vectors into groups based on a specified degree of similarity. Each group of similar patterns (or vectors) is represented by a pattern. New patterns are compared with these representative patterns before being classified into a certain group. Thus, ART1 may be directly applied to classify parts characterized by the column vectors, and machines characterized by the row vectors in the machine-part matrix.

In this paper, it is first shown that the ART1 paradigm in its original form does not provide the expected results. The classification depends largely on the order in which vectors are sequentially applied to the network. Also, a deficient learning policy that gradually diminishes the effectiveness of the representative pattern leads to a number of inappropriate classifications and a larger number of groups than necessary. The problems encountered with the basic ART1 can be attributed to one primary factor: the high sensitivity of the paradigm to the heuristically chosen degree of similarity. This sensitivity can be reduced by applying the input vectors in the order of decreasing density (measured simply by the number of '1's in the vector) and by retaining only the vector with the greater density as the representative pattern. These modifications lead to a great improvement in the correctness of classification.

The method is then validated with test cases studied in literature, as well as with synthetically generated matrices. A procedure for obtaining the optimal cell formation solution is also proposed.

2. A neural network solution

2.1. *Neural networks*

Neural networks are massively parallelized computer systems (Simpson 1990, Wasserman 1989) that have the ability to learn from experience and adapt to new situations besides being extremely fast—particularly when implemented in hardware. Since their inception, numerous application have been introduced. Manufacturing is one of the areas in which extensive use of neural networks has been proposed (Dagli 1994). A number of neural network paradigms can be used as classifiers. The most important feature of a neural network for this application, is that it must be self-organizing. No desired output exists for each input vector, hence paradigms such the back propagation cannot be effectively used for this purpose. The counter-propagation and the ART paradigms can be applied directly to this problem. It is observed that the ART1 (binary ART) paradigm has a resemblance to the similarity coefficient methods (King and Nakornchai 1982, Sahay and Seifoddini 1987, Seifoddini 1989a, 1989b, Seifoddini and Wolfe 1986).

2.2. *The ART1 approach—the difference*

The ART1 approach (Simpson 1990, Wasserman 1989) can be thought of as a direct mapping of the natural way of observing similarities. Most objects in the real world in the area of neural networks are represented as patterns. A pattern may be a 256 × 256 pixel image, or simply a binary vector. When humans encounter an object, an attempt is made to associate the current encounter with previous encounters—the effect is that of relating and classifying new experiences with old ones. Similarly, in the ART1 and several other neural networks, a new pattern is compared with a set of distinct stored patterns. The new pattern is then associated (or classified) with the stored pattern that is most similar to the new pattern.

First, part patterns can be classified by the ART1 to obtain part groups and then machine patterns can be classified to obtain machine groups. The machine-part matrix can be arranged by placing parts within a part group adjacently and repeating the same for machines. The resulting matrix can be inspected for bottleneck machines and the number of exceptional cells be minimized.

Analoguous to the similarity coefficient in similarity coefficient methods, a degree of similarity (or just similarity) between a new pattern and the stored pattern is used. Similarity is defined as,

$$\frac{\text{number of `1's in the same position in both patterns}}{\text{number of `1's in the new pattern}}$$

Thus, the similarity in this case is a measure to ensure whether the new pattern is properly classified or not. When an input pattern is applied to the network, stored patterns (each stored pattern representing a group) compete for it. The stored pattern closest to the input pattern is selected and the similarity between the two is computed. Once computed, the similarity is compared with a pre-specified theshold. A different threshold can be specified for the classification of parts and machines. A different degree of clustering is obtained for each threshold (as in the similarity coefficient method). If the similarity exceeds the threshold, a heuristic is used to change the stored pattern that is representative of its group in order to reflect the influence of the new pattern on the existing state. The performance of the ART1 greatly depends on this heuristic.

In a hardware implementation of the ART1, the stored patterns compete for the input pattern in parallel, resulting in major savings in time. Also, the parts and machines are classified after just a single application of each vector. The ART1 paradigm supports on-line learning, new parts and machines can be immediately classified and scheduled on the shop floor. Thus, besides being significantly different, the approach offers a greater efficiency of classification and can potentially form the core of an intelligent manufacturing system. The integration of ART1 with an expert system is proposed in § 5.1.

2.3. *Group Technology application*

The ART1 code at first classifies the column vectors (by our convention), based on their similarity. In the intermediate matrix, the similar columns are placed next to each other. This achieves a degree of clustering of the '1' elements of the matrix. In the next step, the row vectors, are applied as inputs to the ART1 which likewise classifies them. Re-ordering the rows, such that similar rows are adjacent, leads to a final matrix which shows further clustering. Figure 1 depicts this process. As the two groupings are independent, a significantly faster alternative is to group column vectors and row vectors simultaneously. The disadvantage is that additional processing power and memory will be required. Note that the intermediate matrix

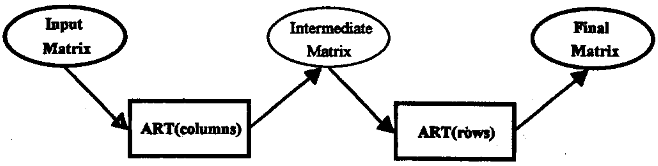


Figure 1. Cell formation with the ART1 paradigm.

serves only as an observation point and need not be the input to either column grouping or row grouping.

3. The ART1 solution

3.1. *ART1 Architecture*

The ART1 architecture consists of two layers of neurons, the comparison layer and the recognition layer as shown in Fig. 2 (Wasserman 1989). In the comparison layer, each neuron has three inputs, a feedback signal from the recognitionl layer—an element p of the vector P, the gain signal G and an element x of the input vector X. The gain G is 0 if any element of the vector P is 1. The output of these neurons is 1, if any two of its three inputs is 1 (the two-thirds rule). Note that if G is 0, the output is simply the AND of x and p. The output of the comparison layer thus obtained is a vector C.

The binary input vector X is applied at the comparison layer and initially passes through the layer unchanged as the binary vector C. This is because, initially G is set to 1 and all elements of vector P are zero. The vector C is the input vector to the

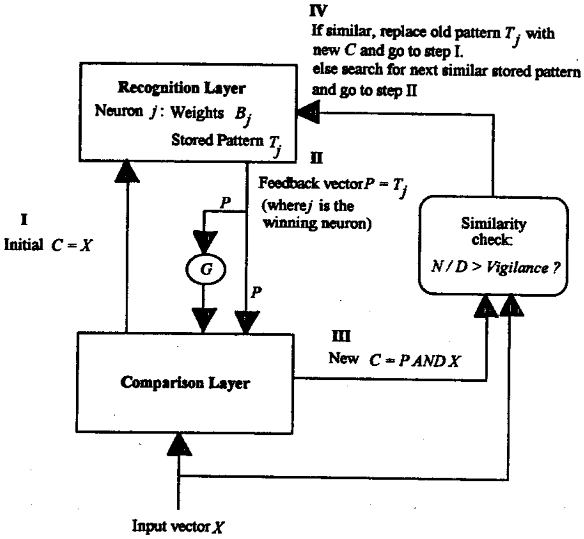

IV
If similar, replace old pattern T_j with new C and go to step I.
else search for next similar stored pattern and go to step II
Recognition Layer
Neuron j: Weights B_j
Stored Pattern T_j
II
Feedback vector $P = T_j$
(where j is the winning neuron)
P
I
Initial $C = X$
G
P
Similarity check:
$N/D > Vigilance$?
III
New $C = P AND X$
Comparison Layer
Input vector X

Figure 2. The ART1 model.

recognition layer. The weights corresponding to these inputs form the vector B_j at the jth neuron. For each neuron in the recognition layer, the dot product,

$$NET_j=(B_j \cdot C) \tag{1}$$

is computed. The output of the neuron with the highest NET value becomes the 'winning' neuron and all the other outputs of the recogniton layer neurons are suppressed to 0.

Each neuron in the recognition layer is associated with a stored pattern. These patterns are stored in two forms, the weight vectors B_j which are not binary and the vector T_j, a binary vector. The elements of B_j are all typically initialized to the same low value, while all elements of T_j are initialized to 1. If the associated neuron wins, and a similarity check is satisfied, the weights are adjusted to the normalized values of the elements of the vector C as follows,

$$b_{ij}=(Lc_i)/(L-1+\sum_k c_k) \tag{2a}$$

$$T_j=\begin{Bmatrix} 1 & \text{if } b_{ij}>0 \\ 2 & \text{otherwise} \end{Bmatrix} \tag{2b}$$

where c_i is the ith component of the comparison layer output vector C, j is the number of the winning recognition layer neuron, b_{ij} is the weight corresponding to the ith component of vector C and L is a constant that is typically set equal to 2. The effect of doing so is that, the weights corresponding to the 1's in C are increased, while the other weights are forced to zero. This is done to increase the chance of a similar vector to be detected at this neuron, while decreasing the chance of accepting a dissimilar vector.

With each neuron in the recognition layer, there also exists a vector, T_j, the elements of which are one's for every non-zero value of the elements of B_j. If j is the winning neuron then the vector that is fed back to the comparison layer P equals T_j. Application of the two-thirds rule at this layer, G being forced to 0 if any element of P is 1, gives us the AND of the elements of vector P and the elements of the vector X, which is the new value of vector C. The number of ones in the resultant vector (N) divided by the number of ones in the input vector $X(D)$, gives us the similarity S, between the input vector and the vector to be stored at neuron j in the form of B_j and T_j. The value of S must exceed a pre-determined threshold value called the vigilance parameter, for the input vector to be classified into a sepcific category. The concept is similar to using thresholds in the similarity coefficient method. If the condition is satisfied, then equation (2) is implemented, otherwise, the second layer neuron with the next highest NET is tested in the same manner. When an input vector is stored in conjunction with a recognition layer neuron for the first time, we say that an 'exemplar' is created, also implying that a new category has been formed. Later when similar vectors are applied, they are 'recognized' by comparison with this exemplar. If a sufficient match is assessed by the similarity check, the two vectors are 'AND'ed and the resulting vector X' (new C) is stored in the form of the vector B_j and its binary version T_j.

3.2. *Drawbacks of the basic ART1*

The ART1 paradigm as described in the previous section does not provide satisfactory results when applied in its basic form. The drawbacks of the basic ART1 are stated as follows:

(1) The stored patterns grow sparser as more input vectors are applied. This effect can be minimized by adjusting the vigilance for different runs of the program to obtain a set of different solutions. However, the classifications almost never result in the block diagonal form.

(2) The classification process and result are completely dependent on the order in which the input vectors are applied.

(3) A basic difficulty is encountered in the determination of the vigilance parameters. A higher vigilance parameter implies that the groups formed will be smaller with vectors that are more similar. At the same time, there is also an increase in the number of cells. This implies that the inter-cellular costs will increase while the intra-cellular costs will decrease. If the vigilance is low, larger groups are formed in fewer numbers and the costs are reversed. The vigilance value thus plays a critical role. But no obvious method of detemining this value for an optimal machine-part cell formation exists. Other similarity coefficient methods had the same problem.

These shortcomings of the ART1 make it unsuitable for the direct application of this paradigm to the classification problem. Also, the bottleneck machine problem is not solved. Even if a block diagonal form is achieved, the paradigm does not prescribe any method of dealing with exceptional elements.

3.3. *Improving performance*

For an improved performance, a few modifications must be made. In this section, we propose two schemes to counter the first two drawbacks mentioned in §3.2. The remaining are dealt with in §4.

(1) In the first scheme, when a comparison between two vectors is successful, instead of storing X', the result of 'AND'ing vectors X and P, (when the two-thirds rule is applied), the vector having the higher number of one's among X' and T_j must be stored. This ensures that stored patterns become denser, rather than sparser as the classification progresses. If both vectors contain, the same number of one's then, any of the vectors can be stored by convention. The chances of improper classification due to the comparison with sparse stored patterns which are not similar to any vectors applied until then, are minimized.

(2) The second scheme involves the pre-processing of the machine-part matrix, to compensate for the dependency of the ART1 on the sequence of application of the vectors. The vectors are applied in a sequential manner, and if sparser vectors are encountered first, those vectors that group well with the later denser vectors will be classified improperly. This is the crux of the second drawback mentioned in §3.2. To avoid this, why not re-order the vectors according to the number of 1's in each vector, and apply them in order of descending number of 1's to the network? The effect of doing this is the 'absorption' of the sparser vectors into the denser patterns that have already been stored.

It is interesting to note the significance of the second scheme in light of the actual process of grouping new machines with existing machines and new parts with existing parts. Similarities between new machines and existing machines are more easily seen if the characteristics of the existing machines are a superset of the characteristics of the new machines. The 'superset' is essentially a greater enumeration of possible characteristics of a class of machines. The same observation applies to parts and in general, to objects.

4. Near-optimal algorithms and results

The results of application of the ART1 approach to some test cases are presented in this section. The observations are used to propose heuristics for finding the optimal machine-part family formation solution and to complement the capabilities of the ART1 paradigm itself.

4.1. *Formalizing the cell formation procedure*

4.1.1. *Determination of near-optimal vigilances*

In order to determine optimum vigilance values, the objective of the classification may be specified in another form. It is clear that the block diagonal form must be achieved to obtain proper distinct clusters of parts and machines. But what characterizes the block diagonal form from any other type of clustering, is the equal number of machine groups and part groups that are formed. If we label the vigilances for column vector and row vector classification as ρ_1 and ρ_2 respectively, and if N is the number of part groups and M the number of machine groups, then N and M generally increase with ρ_1 and ρ_2 respectively. However, their rates of increase depend on the vector set. Also, there may be more than one set of ρ_1 and ρ_2 for which $N=M$.

It is difficult to test the matrix for very small increments in the vigilance parameter. In any case, classification is not sensitive to very small increments. That is, satisfactory clusters can be obtained by varying vigilances in large discrete steps. The results obtained in this way can be adjusted even manually, to obtain distinct clustering.

4.1.2. *Duplication of bottleneck machines*

An algorithm for automatically duplicating bottleneck machines can now be proposed in the following steps,

Step 1 Separate all part groups and machine groups. Identify each block as,

$$b_1, b_2, \ldots b_{n \times m}$$

where n is the number of part groups and m is the number of machine groups. n is also the number of sub-matrices formed due to the part groups.

Step 2 For each block, identify the vector with the highest density (representative vector).

Step 3 Compare all representative vectors with one another. (A degree of similarity must be chosen).

Step 4 Group similar representative vectors. Each grouping results in the duplication of a bottleneck machine.

Step 5 Repeat Steps 2, 3, 4 for remaining members of each block.

Step 6 Repeat Steps 2, 3, 4, 5 for remaining smaller matrices.

The main advantage of this algorithm is that, the existing ART1 code itself can be used for Steps 3 and 4. A low vigilance may be used to allow machines in different groups to be classified in Step 4. However in this algorithm, duplications are not justified by a cost/performance analysis. A pure block diagonal form can be easily achieved, but the number of duplications needed for the pure block diagonal form can be extremely high. If the only criterion is performance then the number of exceptional cells must be minimized at any cost. In such a case, a pure block diagonal form is the optimal solution. In general, machine cost too is an important factor which restricts the number of duplications if it exceeds the cost saving due to lesser material handling. Other restrictions on the maximum allowable cell size and scheduling difficulties are also imposed. The simple algorithm just presented will not suffice. In the following section an algorithm for obtaining an optimal solution is presented. The details of cost/performance analysis are beyond the scope of this paper and were discussed in Gupta and Tomkins (1982), Kumar and Vannelli (1986), Flynn (1987), Sahay and Seifoddini (1987) and Wei and Gaither (1990).

4.1.3. *An algorithm for a near-optimal solution*

The implementation of the following algorithm will complement the capabilities of the ART1 program and leads to an integrated system for optimal machine-part family formation solutions.

Step 1 Find optimal vigilance (as in §4.1.1.).

Step 2 Obtain final matrix using the optimal vigilances.

Step 3 Separate part groups and machine groups. Identify each block as,

$$b_1, b_2, \dots b_{n \times m}$$

where n is the number of part groups and m is the number of machine groups. n is also the number of sub-matrices formed due to the part groups.

Step 4 In each sub-matrix, find and mark blocks with the highest degree of clustering, where 'degree of clustering' is defined as,

$$\frac{\text{number of '1's in a block}}{\text{dimension of the block}}$$

This step can also be done manually as such blocks can be identified visually.

Step 5 In each block, except for the marked blocks in Step 4, identify machines corresponding to the highest number of exceptional elements within the block. Let the machines and the corresponding exceptional elements be,

$$\begin{array}{cc} M_1 & e_1 \\ M_2 & e_2 \\ \vdots & \vdots \\ M_{n \times m} & e_{n \times m} \end{array}$$

It must be noted that by having exceptional elements in several part groups, some of $M_1 \dots M_{n \times m}$ may be the same.

Step 6 Check for cost/performance trade-offs through existing techniques (King and Nakornchai 1982, Flynn 1987, Wei and Gaither 1990 and Huggahalli 1991). Production volume, scheduling techniques, processing times, set-up times are some of the factors to be considered. Duplicate machine M_y corresponding to e_y whose cost is $Max\{C(e_1), \dots C(e_{n \times m})\}$ where C is a cost function, if justified by analysis.

Step 7 Remove M_y from the set $(M_1 \dots M_{n \times m})$ and replace this machine by a machine with the highest number of exceptional elements in block y.

Step 8 Repeat 6 and 7 until duplication of any other machine cannot be justified.

The algorithm facilitates the identification of bottleneck machines and the exceptional elements in the matrix. Thus it acts as an interface between the ART1 program and a decision making system that performs the cost/performance analysis. The algorithm has been implemented on a 43 × 16 matrix (King and Nakornchai 1986, Sahay and Seifoddini 1987, Huggahalli 1991, Dagli and Huggahalli 1991) as well as a 90 × 36 matrix (King and Nakornchai 1986, Huggahalli 1991, Dagli and Huggahalli 1991) discussed in the next section.

4.2. *A 90 × 36 matrix*

A 90 part, 36 machine matrix adopted from King and Nakornchai (1982) was used to further validate the functioning of ART1 and the algorithm just presented. The initial matrix is shown in Table 1.

Using ART1, several classifications were obtained by varying the column and row vigilances. If we observe a graph showing the cluster variation, since the sensitivity of ART1 now depends on the 'representativeness' of the most dense vector, the sparsity of the matrix leads to a number of outliers. The number of part groups and machine are inflated owing to such classifications. If we consider only groups with at least three members, a different but more true cluster variation is obtained as shown in Fig. 3. From the graph, the (column vigilance, row vigilance)

```
   000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
   000000000111111111122222222223333333333444444444455555555556666666666777777777788888888889
   123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
 1                1                 1      1        1                          1
 2                                                                                          1
 3
 4
 5         1    1         1            1  1   1  1        1 11    1111   1         1 1 1 1 1
 6    1                  1     1        1    1    11  1                1    1        1  1
 7                                                                                 1   1
 9                                    1                                                1
10
11           1                1       1  1 1
12
13                                    1                                      1
14                                         1               1       1    1      11
15                   1                                                      1
16        1       1     11       1         1          1    1     111   1    1  1    1   1
17                                                                                1      1
18                                                                                 1 1
19                                                          1   1     1 1  11     1         1111
20                                                                               1  1 1 11 1
21                                                             1  1  1                 11 1  1
22                1     11                 1          1    1      11  11    1  11 1       11 1
23                           1
24
25          11     1     11                  1          1   1       1   1    1  11          111
26    1              1                 1      1     1      1       1             111    1  11 1
27          1            11                 1           1   1       1    1   1  1 11      11111
28          11     1    111       1  1                                        1      1 1 1
29                             1                                1
30          1      1     11                 1                        1          11 11 1      11
31            11       1        1       1    1       1     1     1                   1 1 1
32    111111 1 1 1     1     1 1 1 1    11   111  1  11   11  1 1  1  1  1  1  1  11    1     1
33       1     111              1  1    1    1    1  1    1      1          1           1
34   1       1 1 1  1        1 1 1 1 1       1 1          1  1  1    1   1111 1    1    1 1
35                1            1    1              1     1 1 1        1             1 1
36          11           11                      1  1   1              1  11       1 11  1
```

Table 1. Initial 90 × 36 matrix.

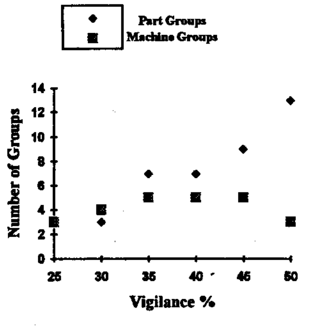


Figure 3. Variation of the number of clusters with vigilance—90 × 36 matrix.

```
   000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
   792274788806678567847803414081352412234566677000112334555670001133885681675134562244123230
   901270668985937238280025271941657938913406824457695376078151360249156435719407120445838625
32 11          1   11    1 1 111111111 111111 1 111 11 111 1 1 11
34 1      1   11       1     1111  1 1 11111  11   1 11   1   1             1
33                           1 111  111 1 1   1  1      1   1     1
 6        1    1  1   1             1  1 1     11                        1              11
31                           1  111 1 1             1       1    1   11  1
 1     1                     1                     1     1         1
11                           1  11    1                             1
 9                        1    1
13                                1                           1
 3
 4
 8
10
12
24
 5      1   1 1  1            1             1          1   1         11111 1111 1   11
36 1 11      1    1  111      1                                      11    11         1
28   11      1                1                1      1              11  11    1   1
19  1  1   11 1  1     1          1                                        111
21        1 1 1                1                                         1   1    1
20        11        1                                                  11  1
 7                                                                     11
22 1111111 1    11 1                         1                         11 11     1
27 1111 111111 1111 1                                                  11
16   11 11      1  1                         1    1                  1111 1    1 11
25  1111 1 111  1 1          11              1                         11 1
26 11  1   1     1 11 1  1111  1  1
30 1111111  111      1  1                                                 1
14     111     1                                                       11
15              1                                                                      1
17 1      1
 2  1
35                   1  1         11        1             1                   1 1          11
18                   1  1
29                                  1       1
23                                                                                        1
```

Table 2. 90 × 36 matrix—final matrix (vigilances = 0·3 and 0·3).

combinations that need to be considered for further inspection are (0·25, 0·25), (0·25, 0·3), (0·3, 0·25), and (0·3, 0·3), since the number of machine and part groups formed are close at these vigilances.

The combination (0·3, 0·3) results in the least number of inappropriate classifications and is therefore chosen. The final matrix obtained with these vigilances is shown in Table 2. The part groups and machine groups are separated column-wise giving 3 sub-matrices A, B and C (the inappropriate classifications are included in sub-matrix C), and considering 4 machine groups, a total of 12 blocks. The blocks may then be numbered (Step 3) as follows,

Sub-matrix A: b_1, b_2, b_3 and b_4

Sub-matrix B: b_5, b_6, b_7 and b_8

Sub-matrix C: b_9, b_{10}, b_{11} and b_{12}.

The degree of clustering for each block is computed and the following three blocks are marked (Step 4),

Marked blocks: b_3, b_5 and b_{10}.

In Step 5, in all but the marked blocks, the machines with the highest number of exceptional elements in a block are identified (32, 36, 35, 5, 25, 35, 31, 25, 35) and the corresponding exceptional elements identified (7, 8, 2, 4, 3, 4, 3, 8, 2).

With the simple criterion that a machine must be duplicated if within a block it introduces more than 2 exceptional elements, Step 6 is implemented by duplicating machine 36 (having the highest number of exceptional elements—8). In Step 7, machine 36 is then replaced in the set by machine 19 which has the next highest number of exceptional elements—7, in the block. Steps 6 and 7 are repeated until no other machine has more than 2 exceptional elements in the block. The final result is shown in Table 3. This example is discussed in detail in Huggahalli (1991).

The modified ART1 performs extremely well, producing results that are very much comparable to the ROC2 algorithm and similarity coefficient methods. The ROC2 algorithm produced, for the 43 × 16 matrix, a solution that used 5 additional

```
    000000000000000000000000000 00000000000000000000000000000000000000000000 0000000000000000000
    792274788806678567847803411 40813524122345666770001123345556700011332330 8856816751345622442
    901270668985937238280025278 19416579389134068244576953760781513602493825 1564357194071204456
32                              111111111 111111 1 111 11 111 1 1 11
34                              1111  1 1 11111  11   1 11   1   1                  1
33                              1 111  111 1 1   1  1      1   1     1
 6                                     1  1 1     11                    11       1
31                              1  111 1 1             1       1    1
 1      1                       1                     1     1         1
11                              1  11    1                             1
 9                           1      1
13                                  1                           1
 3
 5                               1             1          1   1
28                               1                1      1
35                                   11        1             1            11          1 1
29                                    1       1
26                                  1  1
25                              11              1

31                                                                             11  1
 5                                                                             11111 1111 1   11
36                               1                                             11    11         1
28                                                                             11  11    1   1
19                                   1                                             111
21                                1                                              1   1    1
20                                                                             11  1
 7                                                                             11
22                                                                               11 11     1
27                                                                               11
16                                                                             1111 1    1 11
25                                                                               11 1
30                                                                                  1
14                                                                               11
23                                                                                             1

32  11          1   11    1 1
34  1      1   11       1
 6         1    1  1   1
 5       1   1 1  1
36  1 11      1    1  111
28    11      1
19   1  1   11 1  1     1
21         1 1 1
20         11        1
 2  1111111 1    11 1
27  1111 111111 1111 1
16    11 11      1  1                            1    1
25   1111 1 111  1 1
26  11  1   1     1 11 1  1111
30  1111111  111      1  1
14      111     1
15               1            1
17  1      1
 2   1
18                    1  1
35                    1  1
 4
 8
10
12
24
```

Table 3. 90 × 36 matrix—block diagonal form after duplication.

machines and had only one exceptional element (King and Nakornchai (1982). The method employed by Sahay and Seifoddini (1987) produced 6 additional machines and 8 exceptional elements. The result of applying the modified ART1 algorithm showed 5 additional machines and 4 exceptional elements (ignoring part 36's claim to exclusiveness).

For the 90 × 36 matrix, ROC2's solution had 18 duplications with 6 remaining exceptional elements, while the solution shown in Table 3 has 21 duplications and 11 exceptional elements. Thus, speed apart, the ART1 shows classification accuracy that is comparable to those of previous algorithms.

4.3. *Validation with synthetic matrices*

A synthetic matrix generation (SMG) program facilitates the validation and refinement of the ART1 approach. Two main ways in which synthetic matrix generation is beneficial are,

(1) large matrices can be automatically generated. Typing a matrix of size 200 × 100 into the computer can take days!
(2) special cases can be simulated easily. This advantage renders debugging and refinement easier tasks to accomplish.

The program uses the standard UNIX random number generating function, rand( ) extensively. The user is given control over the density of specific regions in

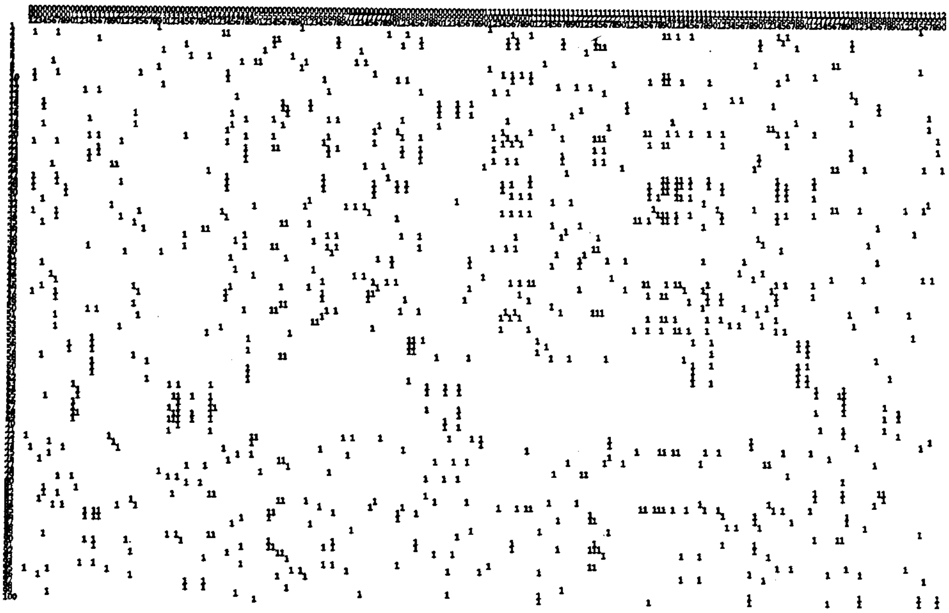


Table 4. Initial matrix created by synthetic matrix generator—matrix without bottleneck machines.

the matrix through a set of control parameters. The program is explained in detail in Huggahalli (1991). The results of simulating some specific cases of a 200 × 100 matrix are now presented.

4.3.1. *Matrix without bottleneck machines*

The SMG was used to generate a 200 part, 100 machine input matrix for which the final matrix is a pure block diagonal form. The input (initial) matrix shown in Table 4 is obtained by ordering the rows and columns of the final matrix.

The classification obtained through the ART1 program for the vigilance combinations of (0·2, 0·2) is shown in Table 5. The final matrix obtained show that if a pure block diagonal form exists, it can be obtained with a higher probability with lower vigilance values. The final matrix obtained using (0·2, 0·2) shows 10 distinct clusters. The representative vector problem explained in § 3.2 produces 5 inappropriate exceptional elements.

4.3.2. *Matrix with two bottleneck machines*

As a second special case, two bottleneck machines (16 and 26) were introduced into the 200 × 100 matrix (Table 6 shows the initial matrix created by SMG). The objective of this experiment was to use the ART1 to correctly identify the bottleneck machines.

The bottleneck machines had a drastic effect on the classifications. Since the row vectors corresponding to the bottleneck machines can be expected to have a greater density the ART1 makes them the representative vectors. At low row vigilances, many row vectors tend to group with these representative vectors leading to a number of inappropriate classifications. However, especially if the bottleneck machines are few in number, they can be separated from the other machines at higher vigilances as shown in Fig. 4. In the final matrix the bottleneck machines can be expected to collect at the top of the rows as shown in Table 7 (Vigilance combination of (0·2, 0·5)). The classification can be performed after removing the bottleneck machines in order to distinctly identify machine cells.

4.3.3. *Very large matrices*

The modified ART1 paradigm has been further improved through 'intermediate learning' by Dagli and Sen (1992) and, applied to very large matrices. Their paper considers a 1200 × 200 case and a 2000 × 200 case. The results of this effort are shown in Table 8 and Table 9 respectively.

4.4. *Classification time*

The performance of the simulation program is comparable to the performance of other algorithms in terms of computational effort and time and therefore the software itself can be used for cell formation. The time taken for processing matrices of various sizes is shown in Fig. 5. Further reduction in computational time can be achieved by performing part grouping and machine grouping in parallel at the cost of extra memory. The program is user-interactive, and offers flexibility, in choosing the vigilance parameters, accepting matrices of very large order, obtaining traces of the classification process and in easy modification of the ART1 to test new schemes. Implementation of this program is thus straightforward.

At this stage, it must be emphasized that the ART1 program is only a neural network simulation. The software can be definitely used to obtain optimal machine-part cell formation solutions, but true advantage of the method will be realized when implemented in hardware.

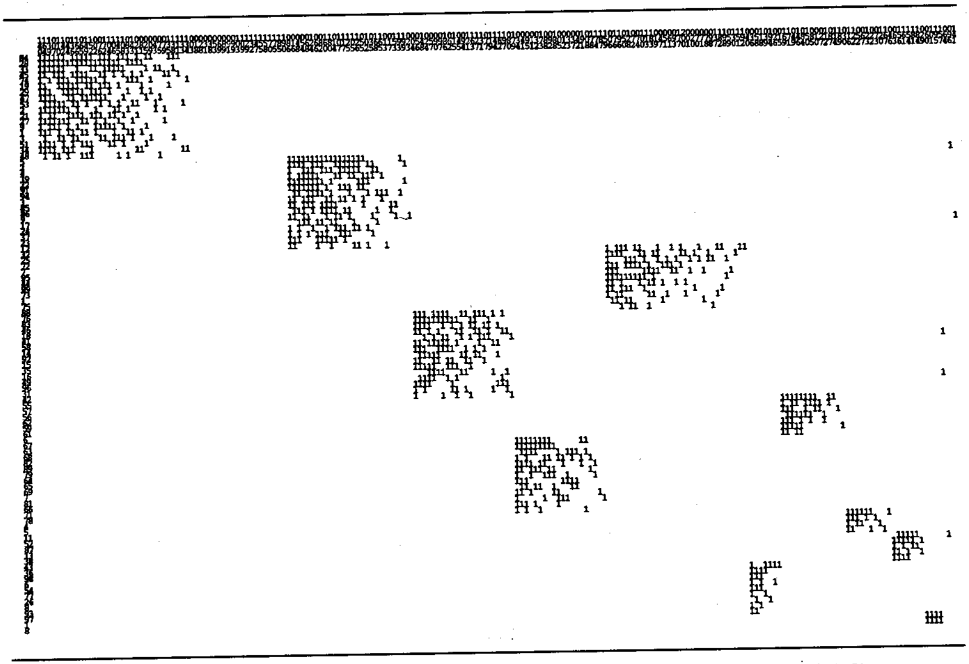


Table 5. Final matrix from the ART1 program (vigilances=0·2 and 0·2).

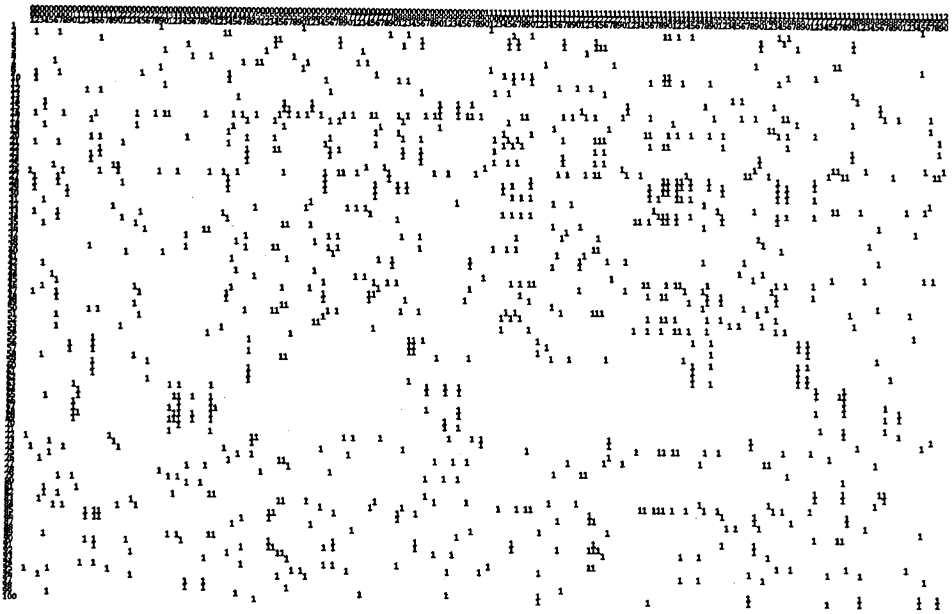


Table 6. Initial matrix created by synthetic matrix generator—matrix with two bottleneck machines.

5. Future work

Several improvements to the ART1 paradigm and enhancements for building an integrated group technology support system are possible as part of further research work. The overall problem posed by cellular manufacturing is very large and complex. The scope of this paper is restricted to a facility that encourages the formal implementation of cellular manufacturing in real time. Some of the issues that must be addressed before actual implementation are discussed in this section.

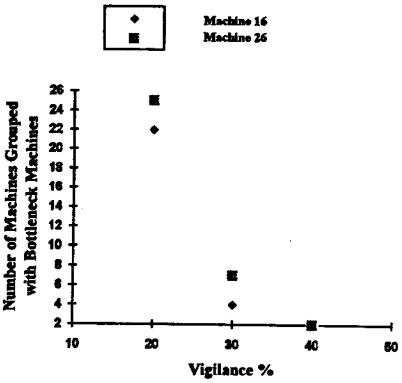


Figure 4. Separation of the bottleneck machines from other machines.

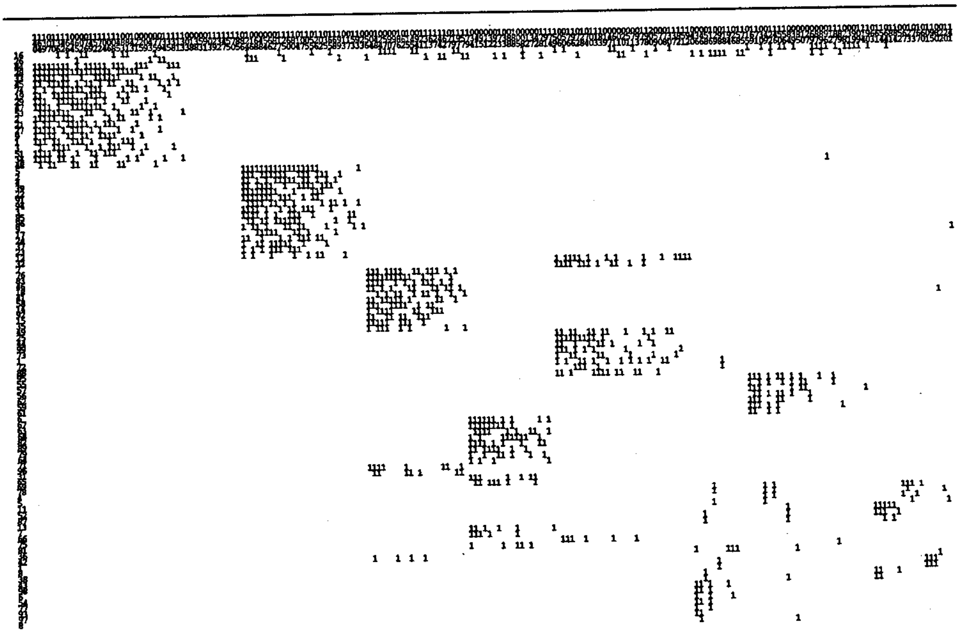


Table 7. Final matrix from the ART1 program (vigilances=0·2 and 0·5).

5.1. *An integrated group technology support system*

The ART1 provides us an excellent classification methodology, particularly when the schemes proposed in Section 3.3, 4.1 and 4.2 are used. However, the problem of finding the minimum cost solution cannot be solved by the ART1. The ART1 provides the flexibility of considering various options for cell formation, but leaves the decision to a cost/performance analysis system. Integrating an expert system, with the ART1 can facilitate the automatic identification of at least close-to-optimal solutions. The ART1 can perform clustering with low initial vigilance parameters and provide part and machine groups formed, to an expert system. The groups thus formed are evaluated in terms of cost by the expert system. Vigilances are changed by suitable increments and a new classification is again obtained and evaluated. The expert system then compares cost figures of various cell formations and recommends a practically achievable global optimal solution. Sufficient range of the vigilance parameters can be covered so that the global minimum is not missed. Trade-offs needed between duplication of bottleneck machines and material handling costs are made automatically by the expert system. The expert system itself must be integrated with simulation software (since no known efficient analytic method for computing optimal solutions exist) for assessment of a cell formation and the scheduling involved. The expert system together with the simulation software forms a decision support system to be integrated with the ART1 program. The integrated system is depicted in Fig. 6.

5.2. *Improvements to the ART1 paradigm*

The only significant weakness of the ART1 paradigm is that of the 'representative vector'. The original paradigm used a heuristic that results in the gradual decay

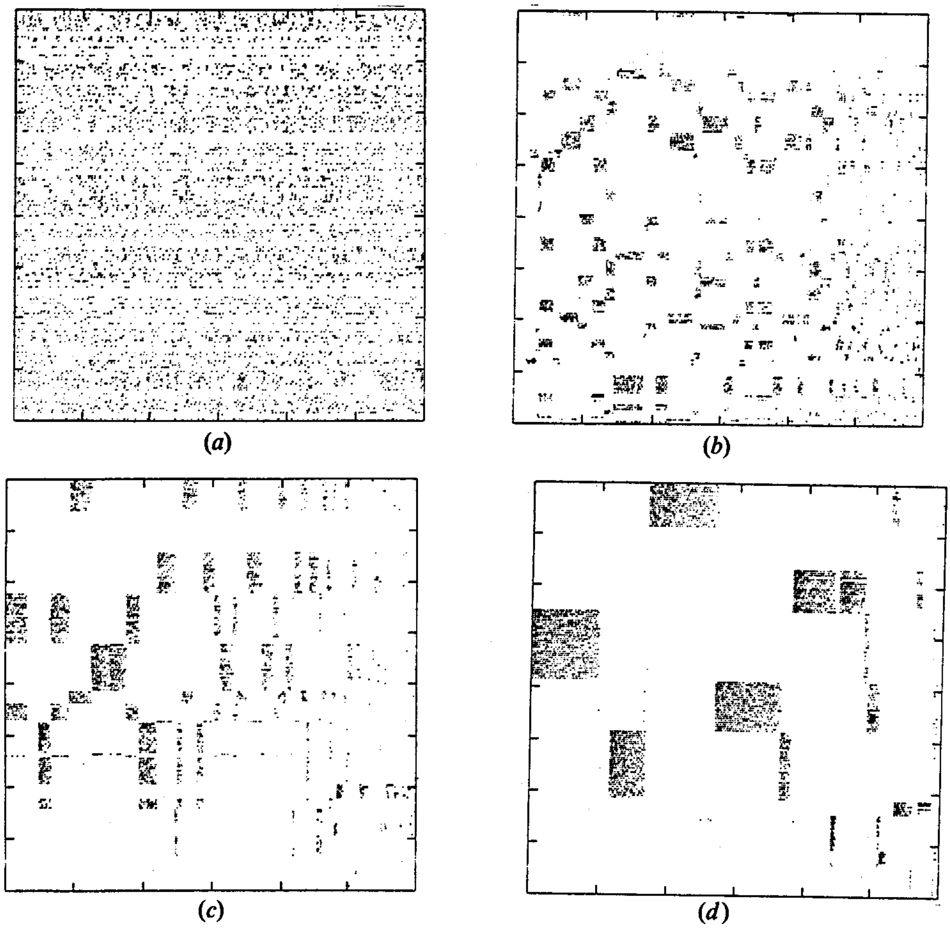


Table 8. A 1200 × 20 case (Dagli and Sen 1992). (*a*) Initial Matrix. (*b*) Result obtained from basic (original) ART1. (*c*) Result obtained using intermediate learning. (*d*) Result obtained using intermediate learning, pre-processing and modified ART1.

of the density of the representative vector. The poor classification accuracy was corrected by retaining the highest density vector as the representative vector at each comparison. The new heuristic though relatively very accurate, has the effect of introducing a few additional exceptional parts typically having one or two exceptional elements. Such parts may be eliminated very easily (by proper classification), particularly as they tend to collect towards the extreme right corner of the final matrix. Improvements that are planned are discussed in the following two sections.

5.2.1. *Fast learning vs slow learning*

The type of learning employed in this implementation of the ART1 is called fast learning. In fast learning, the application of an input pattern has an immediate effect on the weight vector particularly in the original heuristic. More specifically, the weights are changed according to equation (2), where for unmatched positions, the weights become zero. As a contrast, in slow learning, when there is a pattern match, the weights are incremented and decremented by small amounts to make the stored pattern slightly more similar to the input pattern. Slow learning if properly implemented can eliminate the representative vector problem since the representative vector has a resemblance to all input patterns experienced until the time of concern. Slow learning has a stability problem because the classification will be extremely sensitive to the initial weight values, the increment and decrement values used. As noted by Dagli and Sen (1992), the drawback of fast learning is that the category structure is too dependent on input presentation order. Intermediate learning was then suggested to partly alleviate this dependency. The results of their work on very large matrices are shown in Tables 8 and 9.

Most of the other binary, unsupervised, feedback recall paradigms employ learning schemes similar to slow learning and have varying stability characteristics. The efficiency of other neural network paradigms for this classification problem must therefore be determined.

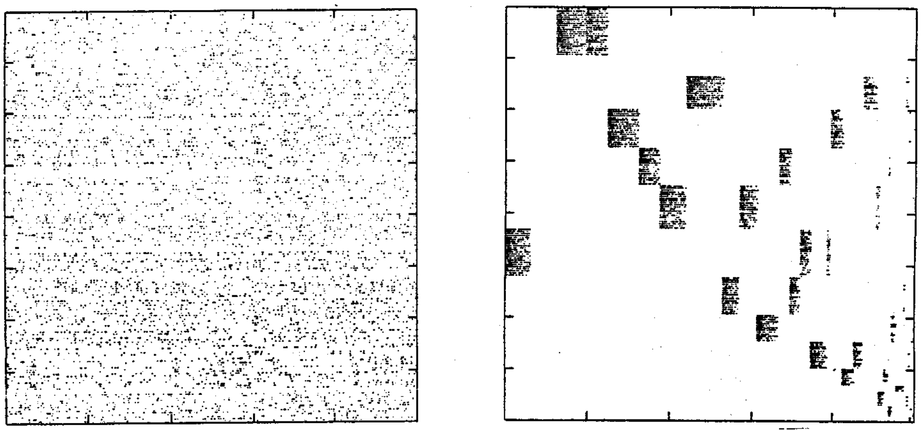
5.2.2. *Improving the software*

An important issue when building the integrated group technology support system is that of 'visualization'. Efficient visualization techniques are extremely important not only for user-interactiveness but also for easy manipulation of large matrices. The program was originally implemented on a HP-Apollo workstation's Domain environment wherein the windows allow both vertical (row-wise) as well as horizontal (column-wise) scrolling. Few windows allow both vertical (row-wise) as well as horizontal (column-wise) scrolling. Few window systems provide for column-wise scrolling, resulting in the wrap around of the row vectors. In any case, inspection and manipulation of large matrices is extremely tedious. Even if the entire procedure is automated, it is necessary to provide for manual intervention through a good graphical user interface.

To improve the speed of the ART1 software implementation, column and row classifications can be made simultaneously. Distributed computing wherein greater processing power and storage space can be leveraged is the key to such an implementation.

5.3. *Hardware implementation of ART1*

A hardware implementation of the ART1 can be used as co-processor in computers dedicated in group technology. The classification process then depends



(*a*) (*b*)

Table 9. A 2000 × 200 case (Dagli and Sen 1992). (*a*) Initial matrix. (*b*) Final matrix.

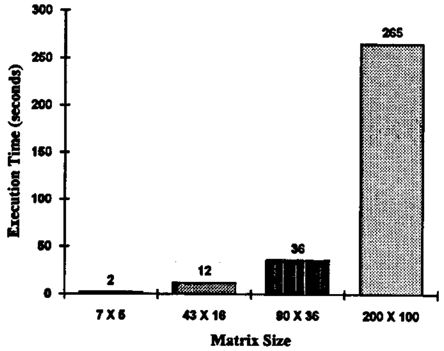


Figure 5. Execution time of the ART1 program.

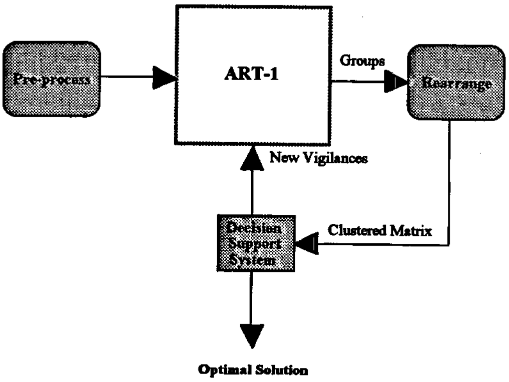


Figure 6. An integrated group technology support system.

essentially only on the time needed to sequentially apply all the input vectors. The neural network itself will be extremely fast and its time delay is negligible compared to other delays. The possibility of integrating other neural networks such as the Hopfield net for obtaining the global minimum of the total cost, must be explored. This would alleviate the intensive need for an expert system, whose interaction with the neural network will be much slower.

5.4. *Use in real time*

In batch manufacturing systems, when new parts are to be manufactured, or when the manufacture of some parts is to be stopped, the optimal cell formation solution can be quite different and inefficient for the current situation compared to the original solution. Provided we have a fairly flexible manufacturing system, it may be possible to regularly re-adjust the shop floor to suit our current needs without incurring excessive overhead expenses. In such a case, a neural network system can greatly facilitate the identification and evaluation of the numerous options that must be considered to obtain optimal layouts for the manufacturing system. Even if the plant layout cannot be immediately changed, the new parts can be correctly classified and various scheduling possibilities can be analysed through simulation. If not optimal, the best possible schedule under the restrictions can be employed.

6. Conclusions

The ART1 neural network paradigm has been successfully implemented for cell formation problems. The results compare favourably with popular existing algorithms such as King's ROC2 and similarity coefficient-based algorithms. Modification of the original paradigm and preprocessing of input vectors obtained from the machine-part matrix, were necessary to obtain drastic improvements in clustering. The paradigm and associated processing modules were simulated with a program and tested with several test cases. The ART1 can be used to obtain several part and machine classifications by varying the threshold parameters called column vigilance and row vigilance respectively. Vigilances that lead to approximately equal number of part groups and machine groups are used to obtain the final matrix. An algorithm for identifying bottleneck machines from the final matrix has been proposed and tested using matrices studied earlier in literature. A synthetic matrix generator has also been developed to facilitate further validation and refinement of the existing ART1 system. The work presented in this paper is expected to lead to an integrated group technology support system.

References

Banerjee, K. G., and Redford, A. H., 1982, Visual inspection of components for mechanized assembly. *International Journal of Production Research*, **20** (5), 545–553.

Burbridge, J. L., 1977, A manual method of production flow analysis. *Production Engineer*, **56**, 34–38.

Dagli, C., and Huggahalli, R., 1991, Neural Network Approach to Group Technology. In *Knowledge Based Systems and Neural Networks—Techniques and Applications*, R. Sharda, J. Y., Cheung, and N. J. Cochran (Eds) (New York: Elsevier), pp. 213–228.

Dagli, C., and Sen, C.-F., 1992, ART1 Neural Network Approach to Large-Scale Group Technology Problems, *Robotics and Manufacturing: Recent Trends in Research, Education and Applications*, **4**, M. Jamshidi, R. Lumia, J. Mulling and M. Shahinpoor (Eds) (New York: ASME Press).

Dagli, C., 1994, *Artificial Neural Networks for Intelligent Manufacturing* (London: Chapman and Hall).

El-Essawy, I. F. K., and Torrance, J., 1972, Component flow analysis—an effective approach to production systems design, *Production Engineer*, **51**, 165.

Flynn, B. B., 1987, The effects of setup time on output capacity in cellular manufacturing. *International Journal of Production Research*, **25** (12), 1761–1772.

Groover, M., 1987, *Automation, Production Systems and Computer Integrated Manufacturing*, 2nd edition (New Jersey: Prentice Hall).

Gupta, R., and Tomkins, J. A., 1982, An examination of the dynamic behaviour of part families in group technology. *International Journal of Production Research*, **20** (1), 73–86.

Huggahalli, R., 1991, A neural network approach to group technology, *MS thesis*, University of Missouri-Rolla.

Hyer, N. L., 1984, *Group Technology at Work*, Society of Manufacturing Engineers.

Hyer, L., and Wemmerlov, U., 1989, Group technology in US manufacturing industry: a survey of current practices. *International Journal of Production Research*, **27** (8), 1287–1304.

Kaparthi S., and Suresh, C., 1992, Machine-component cell formation in group technology: neural network approach. *International Journal of Production Research*, **30** (6), 1353–1368.

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

KUMAR, K. R., and VANNELLI, A., 1986, A method of finding minimal bottleneck cells for grouping part-machines families. *International Journal for Production Research*, **24** (2), 387–400.

KUSIAK, A., VANNELLI, A., and KUMAR, K. R., 1986, Clustering analysis: models and algorithms. *Control and Cybernetics*, **15** (2), 139–154.

MALAVE, C. O., and RAMACHANDRAN, S., 1991, Neural network-based design of cellular manufacturing systems. *Journal of Intelligent Manufacturing*, **2** (5), 305–314.

MOON, Y., 1990, Interactive activation and competition model for machine-part family formation. *Proceedings of the International Journal Conference on Neural Networks*, Washington D.C., **2**, pp. 667–670.

SAHAY, A. K., and SEIFODDINI, H., 1987, An algorithm for forming machine-component cell in group technology. *Proceedings of the 9th International Conference on Production Research*, **1**, pp. 1314–1321.

SEIFODDINI, H., and WOLFE, P. M., 1986, Application of the similarity coefficient method in group technology. *IIE Transactions*, September, pp. 271–277.

SEIFODDINI, H., 1989, Duplication process in machine cells formation in group technology. *IIE Transactions*, December, pp. 382–388.

SEIFODDINI, H., 1989, A note on the similarity coefficient method and the problem of improper machine assignment in group technology applications. *International Journal of Production Research*, **27** (2), 1161–1165.

SIMPSON, P. K., 1990, *Artificial Neural Systems-Foundations, Paradigms, Applications and Implementations* (New York: Pergamon), 1st edition.

WASSERMAN, D., 1989, *Neural Computing—Theory and Practice*, Van Nostrand Reinhold.

WEI, J. C., and GAITHER, N., 1990, An optimal model for cell formation decisions, *Decision Sciences*, **21** (2), 243–257.