Cell design, management and continuous improvement

Baker, Robert Peter

CELL DESIGN, MANAGEMENT AND CONTINUOUS IMPROVEMENT

A thesis submitted to the University of Durham for the degree of Master of Science

by

Robert Peter Baker

School of Engineering

October 1997

Declaration

I hereby declare that the work reported in this thesis has not been previously submitted for any degree. All material in this thesis is original except where indicated by reference to other work.

Abstract

A cellular manufacturing system is a shop floor that has been organised into groups of dissimilar machines producing groups of similar parts. Each group of machines is called a cell and each group of parts is called a part family. The main advantage of a cellular manufacturing system is low material handling, since ideally, a part need only travel to the cell it belongs to in order to be manufactured. If a cell can manufacture its part family without any member of that part family having to travel to another cell, then that cell is said to be independent.

In reality, cells are rarely independent and this causes many complications when trying to design a cellular manufacturing system. To address these complications, a strategy for cell design, management and continuous improvement was developed. This comprises three stages:

(i) Determine cell configurations.

(ii) Position cells and the workstations within them.

(iii) Carry out Capability Analysis to identify targets for continuous improvement.

Black Box Clustering is used to determine cell configurations by clustering a workstation-part matrix representation of routings. The Cellect layout tools identify the best position for each cell and the relative positions of the workstations within them based on material handling costs. This data combined with user interaction can be used to identify the precise locations of individual workstations. Capability Analysis is a methodology developed to assess groups of performance measures that should be

similar. It is used to create a list of targets for improvement, ranked in such a way that the target at the top of the list is the one in most need of improvement and the target at the bottom of the list is least important.

The three stages of cell design were implemented within a prototype software tool called Cellect. The heart of this system is a relational database that is used to manage the data required to run the system. Cellect was tested using industrial data.

Acknowledgements

I am grateful to my supervisor Dr. Paul Maropoulos for his guidance, encouragement and commitment throughout this research.

I am thankful to my colleagues with whom I was able to discuss my work. I am particularly grateful to Kelvin Revere, John Darlington, Neil Stutchbury, Alima Laguda, Hugh Bradley, Christos Emmanouilidis, Mike Betteridge and Zhihui Yao.

I am also grateful for the data supplied by Warner Electric.

Finally, I would like to thank my family, my girlfriend Jo Smith, and my friends Neil Kirby and Greg Bradshaw for their love and support.

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

There is a realisation amongst many manufacturers that manufacturing system design can no longer be the responsibility of the Production Engineering department alone. This is particularly evident within those firms practising *Concurrent Engineering* (CE), whereby the development of the manufacturing system is carried out simultaneously with product design. This is achieved by having representatives from Design, Manufacturing, Marketing, Purchasing and other functions working together as a CE team on a single product or product group. Whereas with traditional 'over the wall' engineering each function from design through to sales is done in isolation, with CE the aim is to develop a 'right first time' strategy to aid the rapid introduction of new products into the manufacturing system (Chanan and Menon 1994). This avoids, for example, an expensive manufacturing process having to be used when a less expensive option exists.

In much the same way that the advantages of cellular manufacturing stem from the fact that a whole product group is the responsibility of a team, the same principle, when implemented on the shop floor also brings with it certain advantages. These advantages arise as a result of grouping a large number of components into a smaller number of groups of similar parts. These groups can then be manufactured in cells of dissimilar workstations dedicated to them, with each cell managed ideally by one operator (Sekine 1992). This improves material handling and manufacturing system management. Note that in this Thesis, a workstation refers to a machine or other

facility that serves products by carrying out value adding operations. Figure 1.1 represents a comparison between the company-wide policy of CE and the shop floor policy of cellular manufacturing discussed above. Figure 1.2 shows how material flows within a traditional functional layout and the advantages in terms of material handling of converting to a cellular layout. Not only are material handling distances reduced, but the flow of material through the shop floor becomes smoother. Other advantages of cellular manufacturing are outlined in Chapter 2 and in Chapter 3 the shop floor layout of cellular manufacturing systems is discussed.

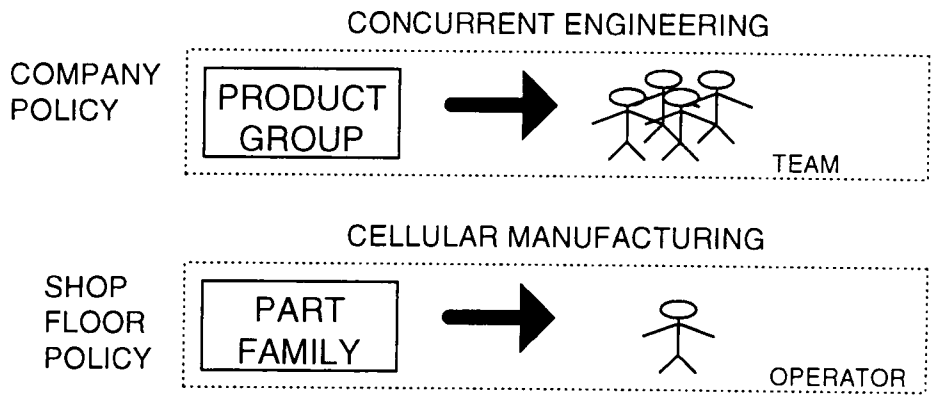


Figure 1.1: Comparison between concurrent engineering and cellular manufacturing

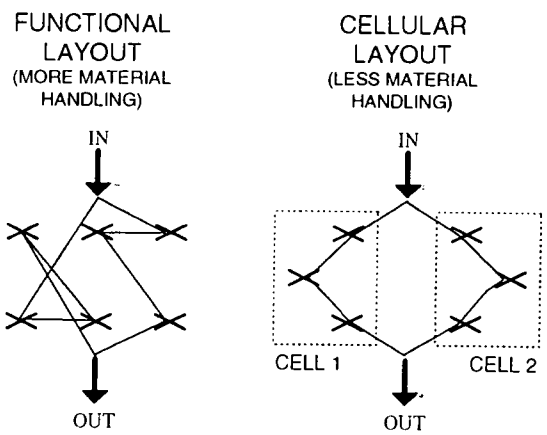


Figure 1.2: Material handling advantages of cellular layout over functional layout

In cellular manufacturing terms, a group of similar parts manufactured within a cell is called a part family. The categorisation of parts into part families is the single most important stage of forming a cellular manufacturing system. Traditionally, this has been done by identifying groups based on the physical similarity of their parts using a methodology called *Group Technology* (GT) and then allocating cells of workstations to the part families. The terms GT and cellular manufacturing are often used interchangeably, but in this work, the term GT will generally refer to the act of grouping parts, whereas cellular manufacturing will generally refer to the processes carried out by cells of workstations.

The thinking behind GT is by no means revolutionary and it has been reported that GT concepts were implemented as long ago as 2500 BC for the manufacture of stone cutting tools. This was done by grouping together similar tool shapes so that they could be made from the same blank (Koenigsberger 1972). For the manufacture of modern, more complex shapes, classification of parts into part families was traditionally carried out using systems such as BRISCH, CODE, MICLASS, OPITZ and KC-1 (Gallagher and Knight 1973, Bennett 1985, Ballakur and Steudel 1987, Keus et al 1977). However, such methods of classification and coding were highly data driven (Hyde 1981) which made them prone to errors from user input and unresponsive to factors such as design modifications. Of greater concern was the fact that such techniques were expensive and difficult to implement (Perrego et al 1995) and so it became apparent that other methods of determining cell configurations were required that needed the minimum amount of data already available. This led researchers to shift the emphasis away from classification schemes and instead use part routings to simultaneously group both parts into part families and workstations into

cells (King and Nakornchai 1982). These methods are collectively known as either *Production Flow Analysis* (PFA), *cluster analysis* or *cell formation algorithms* and are reviewed in greater depth in Chapter 2.

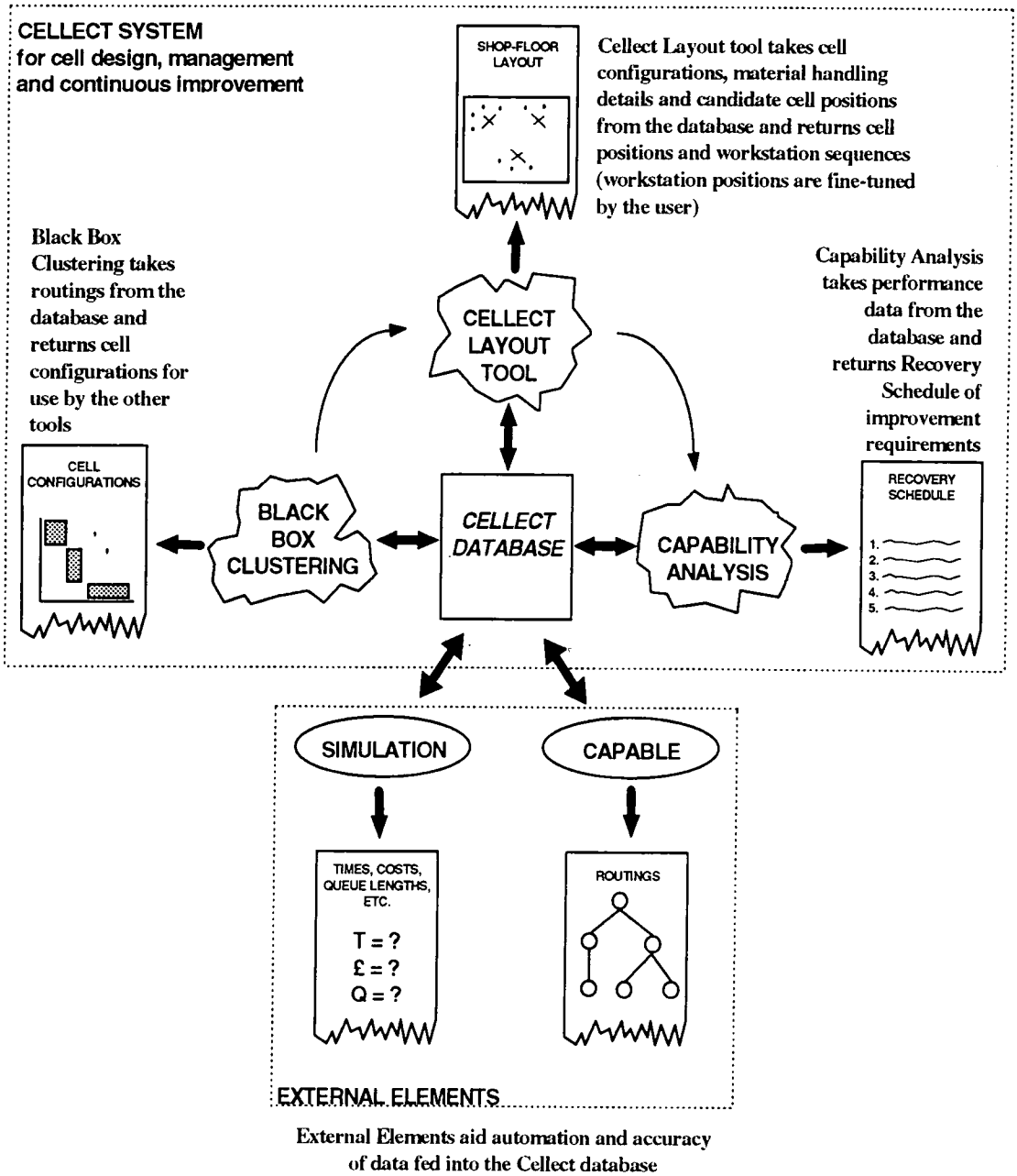


Figure 1.3: The Cellect system

None of the research undertaken into cellular manufacturing focuses on the need for a potential system to be used within a CE environment. To address this issue, the basis

of this research is the fact that to obtain the benefits of cellular manufacturing within a CE framework, a tool is required that can aid the design, management and continuous improvement of cellular manufacturing systems and be practical enough to be used by all members of a CE team. This not only means that a cell design system should be able to determine cell configurations and layouts as outlined above, but also be able to aid in determining ways of improving both the capability of the shop floor to manufacture its products and the capability of part designs to fit into a GT schema for efficient and easy manufacture.

To satisfy this need, a suite of software tools have been developed under the collective name of *Cellect*. Figure 1.3 is a representation of the Cellect system which is made up of the following elements.

1.2 Cellect Database

This is the nerve centre from which all the tools obtain their data and through which they share their results. Initial input comes from the planning system, "CAPABLE" (see Section 1.3.1.), Discrete Event Simulation systems and from CE team members. Results from data being processed by any of the elements are returned to the database which produces reports for analysis by CE team members. Updates resulting from actions taken by the CE team are inputted into the database for further processing in an iterative cycle that aids concurrent product and manufacturing system design.

1.3 External Elements

These support the Cellect activities by aiding automation and ensuring the accuracy of the data fed into the database. The inclusion of external elements as part of the Cellect

system is meant to highlight the fact that Cellect is supposed to complement, rather than replace, other cell design tools.

1.3.1 Modelling Tools

The Concurrent Engineering Support System (CAPABLE) currently being developed at Durham University is an object-oriented tool that determines possible routings for product designs by identifying the workstations required to manufacture the features that make up each component produced by the manufacturing system. By taking into consideration factors such as quality, cost and delivery rate (production time), the best set of routings can be selected (Bradley and Maropoulos 1997). These routings can be used to determine cell configurations (Chapter 2) and values corresponding to other factors assessed by CAPABLE can be used by the Capability Analysis tool (Chapter 4).

By creating a real-time model of the manufacturing system using simulation software, accurate queuing times, material handling times, machine utilisation and other details can be fed into the database for analysis by any of the other elements which in turn feed their results via the database back to simulation. With simulation, the user is able to analyse the effects product designs have on various manufacturing system configurations prior to committing resources (Morris and Tersine 1994, Massay et al 1995, Chan et al 1995). The usefulness of simulation to Cellect has already been recognised and research has been carried out at Durham University to establish a direct link to the WITNESS simulation tool (Higgins 1997 - see Chapter 6).

1.3.2 Data Acquisition Tools

By tagging a part with a bar-code or a kanban card, data about it can be stored either electronically or manually and updated each time an operation is carried out. Such data will provide information about the time taken for the part to travel between workstations and the times taken to set-up and process it. Data should also be recorded at each workstation to store down-times, defect rates and times, queue lengths, work in progress and so on. Correct auditing procedures should be in place to ensure the accuracy of data and standards such as those regulated by ISO 9000 can aid this process. However, the most effective way of ensuring employee co-operation in the data collection process is to ensure that system responsibility is passed to all members who are in contact with it (see Chapter 4).

1.4 Cellect Tools

These tools help to carry out the following three stages of cell design

(1) Clustering of a workstation-part matrix into a block diagonal form from which cell configurations can be determined

(2) Layout of the cells and the workstations within them

(3) Capability Analysis to identify targets for continuous improvement

1.4.1 Black Box Clustering (Chapter 2)

Part routings can be represented by a workstation-part matrix of ones and zeros where a one indicates that the part visits the corresponding workstation and a zero indicates

that it does not. Clustering algorithms attempt to rearrange this matrix so that the workstation-part visits (ones) are grouped into blocks from which workstation groups and part families can be identified. The clustering algorithm used by Cellect is *Black Box Clustering* which was developed to cluster the matrix, mark off the cells and determine the quality of the resulting block diagonal form, all without user interaction.

1.4.2 Layout (Chapter 3)

Having determined cell configurations, the next stage of the cell design process involves positioning the cells and the workstations within them on the shop floor. This activity requires that a large number of factors are taken into consideration. Many of these are difficult to quantify and hence model within a computer-based system. It was therefore decided to let the computer do what it is best at, namely number-crunching, to identify approximate positions of cells and the relative positions of the workstations within each one on the basis of material handling costs only. These activities are carried out by the Cellect Layout tools and it then becomes the responsibility of the user to identify the exact positions of the workstations based on qualitative factors such as safety, positions of walkways and AGV guides and so on. This activity allows the shop floor staff who will be using the workstations to be involved in their positioning.

1.4.3 Capability Analysis (Chapter 4)

The purpose of Capability Analysis carried out within this project is to examine the performances of the operations, parts, products, workstations and cells of a

manufacturing system and to assign a priority to each performance to form a ranked list of actions to be undertaken at either operation, part, product, workstation or cell level. Capability Analysis allows the comparison of different design and manufacturing factors within the same list. The analysis includes methods for representing factors measured at one level as performances at higher levels. By allowing performances to be represented at various levels, it is then possible to take a higher level target and identify corresponding lower level targets making it possible to pinpoint specific areas for improvement that will give the most overall benefit. Thus, the objectives of Capability Analysis are to obtain the most benefit from a given resource allocation, to build flexibility into the manufacturing system and to lower the time taken to work up to full production levels when introducing new products.

To demonstrate the use of Cellect, an example is presented in Chapter 5 that takes a set of data inputted into the Cellect database and demonstrates how this can be used to determine cell configurations, a shop floor layout and a set of improvement targets from Capability Analysis. This chapter also describes in detail the implementation of the Cellect tools using Microsoft's Visual FoxPro version 1.3. This is done by showing in table form, the information required by the database and each of the Cellect tools and also the output from each of the tools. Finally, Chapter 6 concludes the Thesis and suggests scope for further work.

1.5 Objectives of the Thesis

As outlined above, the rest of this Thesis is divided into 5 chapters (Chapters 2 to 6). Each chapter covers a separate subject, and as such will have its own literature review that will outline previous relevant work. Because the aim of the project was to develop

a computer-based system for the design, management and continuous improvement of cellular manufacturing systems, the research presented in the following chapters will refer to the Cellect system outlined in Figure 1.3. However, it is the intention that the methods discussed herein should also be applicable to those organisations that do not have the resources to establish an integrated cell design system such as Cellect. To this end, the Cellect layout tools and Capability Analysis have been designed to be implemented manually and this is shown using simple examples in Chapters 3 and 4. Manual implementation of these methods is only practical if the data sample is small enough. If this is to be the case then only a proportion of the shop floor should be used for the analysis. Conversations with senior production staff at local companies that have implemented cellular manufacturing have revealed that reorganising the shop floor into cells is more acceptable to management, who have to justify the resources required to carry out the change, if in the first instance only part of the shop floor is reorganised. Although Black Box Clustering described in Chapter 2 can only be implemented using a computer, if the data set is small enough, then PFA may be carried out manually (Burbidge 1975). Layout Design and Capability Analysis can then be carried out on paper or using a spreadsheet.

However, there comes a point when the data set is too large to allow manual analysis and it is at this point that a system such as Cellect becomes justifiable. This is particularly true when analysing *what-if* scenarios when assessing products not yet manufactured by the organisation or when assessing the benefits of purchasing state-of-the-art resources that the organisation does not yet own. In this case using a database system such as Visual Fox Pro 1.3 (Hentzen 1995) is the key to efficient data storage and retrieval. The implementation and use of such a system is described in

Chapter 5. To obtain accurate data necessary for assessing *what-if* scenarios, it is also advantageous to integrate discrete event simulation into the analysis. Research has been carried out to extend Cellect in this way (Higgins 1997) and this is described in Chapter 6.

The reader should bear in mind that each of the three stages of cell design can be carried out in isolation. For example, Capability Analysis can be carried out without first carrying out Black Box Clustering and layout design. Thus, it is hoped that this Thesis will present a set of generic tools that can be applied to various situations and in varying degrees of complexity. In this way it is intended that the research presents a set of practical tools that may be used by Industry to carry out cell design, management and continuous improvement.

1.6 Related Publications

This thesis presents the author's own work except for appropriately acknowledged related work. The methodologies, algorithms and developed software have been documented in refereed papers. These are as follows:

- "An Automatic Clustering Algorithm Suitable For Use By A Computer-Based Tool For The Design, Management And Continuous Improvement Of Cellular Manufacturing Systems" [Baker R.P. and Maropoulos P.G. (1997a)] presents the Black Box Clustering algorithm described in Chapter 2.

- "Manufacturing Capability Assessment for Cellular Manufacturing Systems" [Baker R.P. and Maropoulos P.G. (1997b)] presents the Capability Analysis methodology described in Chapter 4.

- "Cell Design, Management and Continuous Improvement, Part 1: Cell Configurations and Layouts" [Baker R.P. and Maropoulos P.G. (1997c)] describes the implementation of Black Box Clustering (Chapter 2) and the Cellect Layout Tools (Chapter 3).

- "Cell Design, Management and Continuous Improvement, Part 2: Capability Analysis for Continuous Improvement" [Baker R.P. and Maropoulos P.G. (1997d)] describes the implementation of Capability Analysis (Chapter 4) within Cellect and also describes how Capability Analysis can be implemented manually.

2. Black Box Clustering

2.1 Introduction

Group Technology has been developed to enable the manufacture of similar parts at dissimilar machines. The workstations (for the purpose of this Thesis, a workstation is defined as any point at which manufacturing operations take place and includes assembly points as well as production machines) that together manufacture a given part family are collectively known as a cell. The benefits of cellular manufacture identified by Wemmerlöv and Heyer (1989), Sekine (1992), Hey (1988) and Harrison (1992) include:

- Simplified planning in that parts need only be scheduled to their given cells,
- Reduced work in progress inventory,
- Reduced set-up time,
- Reduced throughput time,
- Reduced material handling,
- Increased part standardisation,
- Reduced part proliferation,
- Improved workstation performance,
- Reduced cost of indirect labour,
- Lower inspection costs, and

- Improved morale of workers due to greater responsibility and the increased variety of their tasks.

These benefits become greater as the cells become more independent since then, the performance of a given cell does not depend on the performance of other interacting cells. However, totally independent cells rarely exist, particularly in high product variety environments. Because of this fact, much research has taken place over the last two decades to find methods to cluster parts and workstations so that the cells produced are as independent as possible. The most common way of identifying cell configurations is to use *Production Flow Analysis* (PFA) whereby cells and part families are determined by grouping similar production routings. This is the first stage of the cell design process described in Chapter 1 and forms the basis of the research described in this chapter.

2.2 Cluster Analysis

Identifying cell configurations requires grouping similar parts and the machines they visit. Originally this was done by classification and coding: similar parts had similar codes and were hence allocated the same group of machines. This method suffered because coding was complex and time consuming. As such, many firms were reluctant to adopt the Group Technology methods required to form cells (Bennett 1985).

Burbidge (1975) first suggested that cells could be identified from part routings by using a [1,0] matrix; a one indicates that the part visits the workstation and a zero indicates that it does not (see Figure 2.1). By swapping rows and columns, the matrix can then be converted into a block diagonal form (BDF - see Figure 2.2) where each

block represents a cell and associated part family. Obtaining a BDF is the primary objective of PFA.

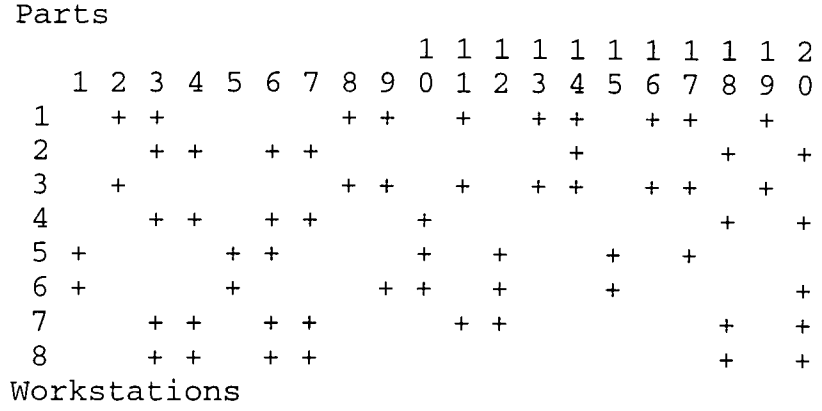


Figure 2.1 Example machine part incidence matrix (discussed in Section 2.4)

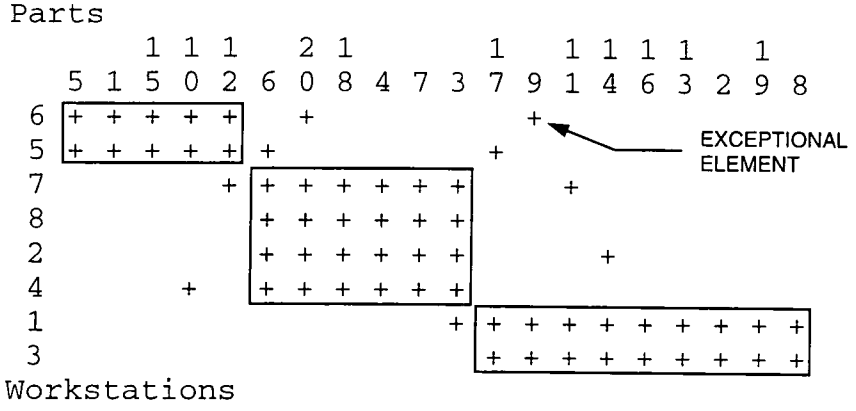


Figure 2.2: BDF for example matrix

When Burbidge first proposed this system, he suggested that PFA could be done manually for matrices containing up to 2000 parts. Researchers, however, soon found that this was not the case. This is because cells are rarely independent as some parts may have to visit two or more cells to get access to the workstations required for their manufacture. A visit by a part to a cell other than the one containing its associated part family is called an exceptional element (see Figure 2.2). The more exceptional elements there are, the more difficult PFA becomes.

As a result of the difficulty in obtaining a BDF, the last two decades have seen a large number of computer-based clustering algorithms published. The basic types are described below.

2.2.1 Similarity Coefficients

The similarity of two workstations is a function of the number of parts they have in common and vice-versa. This can be expressed by a similarity coefficient, S:

$$S = \frac{\text{Number of parts using both workstations}}{\text{Number of parts using both} \; + \; \text{Number of parts using either}} \qquad \textbf{Equation 2.1}$$

Similarity coefficient algorithms (Mc Auley 1972, DeWitte 1980, Waghodekar and Sahu 1984) convert matrices of similarity coefficients between workstations and parts into dendrograms. User-defined threshold values divide the dendrograms into cells. The sizes and independence of the resulting cells are dependent on the threshold values used. In order that it may be used by shop floor staff, the Cellect clustering algorithm should work without any intervention or expertise on the part of the user. Because of the need to identify threshold values, these algorithms cannot work without some form of user interaction and are hence unsuitable for use within Cellect.

2.2.2 Mathematical Methods

Under this heading come algorithms that model PFA as a set of functions that have to be maximised or minimised (Adil et al 1996) as well as those that have been developed as graph partitioning models which utilise similarity coefficients of one sort or another (Chen and Irani 1993, Kazerooni et al 1995). Modelling is, however, only

the first stage in developing mathematically based clustering algorithms since these models also have to be solved. This is done using a number of novel methods that have been adapted from other applications, including genetic algorithms (Goldberg 1989), simulated annealing (Chen et al 1995) and neural networks (Malakooti and Yang 1995, Chen and Cheng 1995). The need for dedicated expertise and/or software to use these methods makes them impractical for use within this research given that clustering is only the first stage of the overall cell design and management process.

2.2.3 Resources-Based Clustering

These methods include algorithms that aim to minimise costs (Askin and Subramanian 1987), utilisation (Ballakur and Steudel 1987) and material flow (Harhalakis et al 1990). Within Cellect, however, the priority at the clustering stage is to obtain the best possible BDF for the given routings, regardless of other considerations. Optimising the efficiency of the cells is then done at the continuous improvement stage using Capability Analysis methods. Hence, it was felt that there is no point in compromising BDF quality in favour of resources considerations when these will be taken into account at a later stage, as explained in Chapter 4.

2.2.4 Matrix Ordering Methods

The algorithm referred to by most authors on the subject of PFA is Rank Order Clustering (ROC - King 1980). This is because it is a simple algorithm that lends itself well to computerisation. Since the workstation-part matrix is made up of ones and zeros (according to whether or not a given part visits a given workstation), each

column and row can be converted into a binary word. The columns are rearranged according to their binary word rankings and binary word values are then determined for rows which are re-arranged in the same way as columns. This process is repeated until further ordering cannot take place. There are two main disadvantages of ROC. Firstly, the use of binary words to weight the lines of the matrix results in cluster dispersion, particularly when there are a lot of exceptional elements. Secondly, despite being easy to program, the use of binary words severely restricts the size of the problem that can be computed; each line is represented by a number that equals twice that of the previous line - it takes only 47 lines to reach the average PC's largest number limit. These problems have been tackled to some extent by algorithms such as ROC2 (King and Narconchai 1982) and MODROC (Chandrasekharan and Rajagopalan 1986a) but require user interaction to function and hence do not satisfy the requirements of this project.

2.2.5 The Basis Of The Cellect Clustering Algorithm

To overcome all the problems associated with ROC, a set of matrix ordering algorithms under the heading of Method of Moments have been developed by Mukhopadhyay et al (1995). Of particular interest is the End Load Ratio (ELR) algorithm that models each line of the matrix as weightless beams that are reordered according to their centre of mass values in much the same way as reordering takes place according to binary words in ROC. Clusters are still prone to dispersion due to the effect that exceptional elements have on ELR values, but because these values are not affected by position weightings (as with binary words) the effects are by no means as severe as with ROC. Also, the algorithm is not restricted by computational power

in the same way that ROC is. The final advantage, in terms of the requirements for this project, is that matrix rearrangement carried out by this algorithm requires no user interaction, though the user still needs to mark off the matrix to identify cell configurations. Hence, given that this algorithm is both effective and straightforward, ELR is used as the basis of the clustering algorithm developed in this project.

It should be noted that the four types of algorithm identified above are not distinct since algorithms can fall within two or more types. Perrego et al (1995) developed an algorithm which can best be described as a hybrid of similarity coefficient, resources-based and matrix ordering. Chandrasekharan and Rajagopalan (1986b, 1987) used graph partitioning theory (mathematical methods) to carry out matrix ordering to fit ideal seeds which can be thought of as BDF templates that represent likely cell configurations. Also, the algorithms referred to as examples when describing these types are by no means exclusive. Over the last two decades literally hundreds of algorithms have been developed and it is beyond the scope of this Thesis to describe them all.

To summarise, the problem with many of the algorithms above is that they require user input to set the various parameters needed for them to function. Setting any of these parameters requires knowledge of the algorithm and experience in its use. This reduces the opportunity of using such algorithms within multi-disciplinary product and factory development teams. It is a requirement, however, that Cellect should be straightforward to use, particularly as PFA is only one part of its functions. Further more, with the advent of Concurrent Engineering tools that can quickly generate multiple routing options (such as CAPABLE - Bradley and Maropoulos 1997) it is necessary to carry out PFA on a large number of different machine-part matrices. As

such, a need exists for a fully automatic clustering algorithm that given a [1,0] input matrix is able to identify cell configurations and for each cell produce a list of workstations and parts. In short, clustering should be a 'black box' with which the user has no interaction and it was upon this premise that the clustering algorithm described in the next Section was designed.

2.3 Black Box Clustering Algorithm

Black Box Clustering (so called because it works behind the scenes with no user interaction) takes a workstation-part matrix (with workstations as rows, parts as columns and where each element equals one if the given part visits the corresponding workstation and zero otherwise), converts it into block-diagonal form (BDF), marks off the cells and provides measures of the BDF quality. It is made up of the three phases described below:

2.3.1 Phase 1: Converting the [1,0] machine-part matrix into a BDF

Phase 1 is a hybrid algorithm made up of the following three steps:

(a) End Load Ratio (ELR) clustering

(b) Tightening up of the clusters using similarity coefficient based weights

(c) ELR clustering without interference of exceptional elements

ELR clustering carried out as step a of Phase 1 models each row or column of the [1,0] matrix as a beam with a unit mass placed where a 1 exists. The ELR of the beam

is used to identify whereabouts on the matrix the row or column belongs and is calculated as follows.

$$M_j = \sum_{i=1}^{i=I} a_i \qquad \textbf{Equation 2.2}$$

$$X_j = \sum_{i=1}^{i=I} i.a_i \qquad \textbf{Equation 2.3}$$

$$C_j = \frac{X_j}{M_j} \qquad \textbf{Equation 2.4}$$

where;

i is an element number (or position) on a line or beam, j within a workstation-part incidence matrix,

I is the total number of elements on j,

a_i is the value (1 or 0) of i,

M_j is the mass of j,

X_j is the moment of j, and

C_j is the ELR value of j

ELR values are calculated and ranked for rows (workstations) which are rearranged to match the positions of these ranked values. The same is done for columns (parts) then for rows again and so on until no more ranking can take place. This procedure is shown in Figure 2.3.

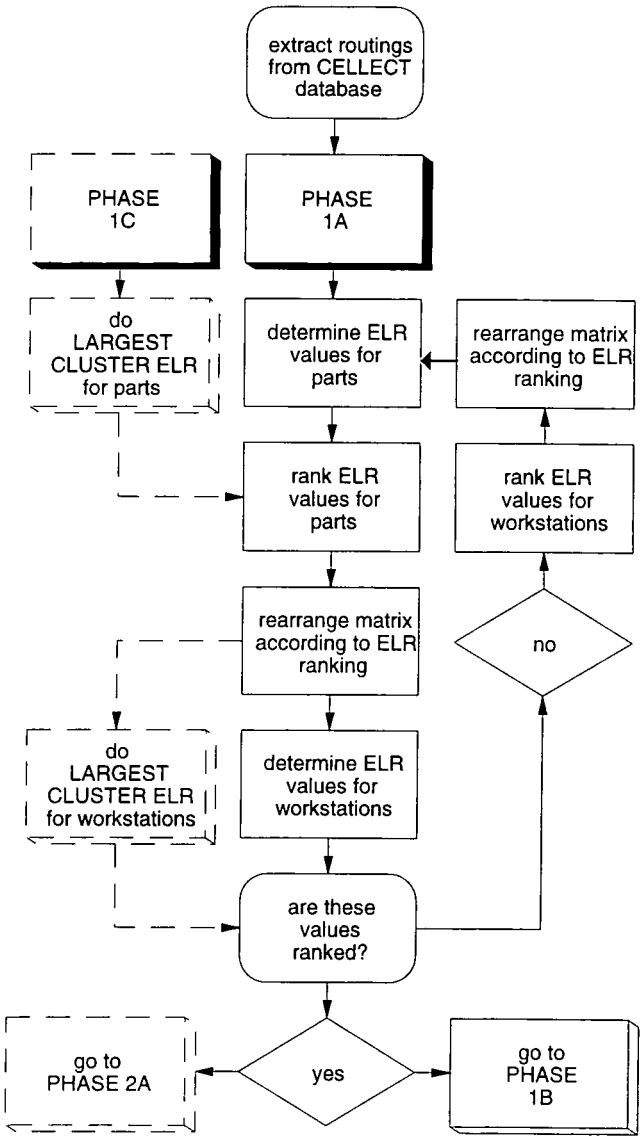


Figure 2.3: Phase 1, step a

In Figure 2.3, routings are converted into a machine-part incidence matrix. ELR values (Equation 2.4) are calculated for workstations and the rows of the matrix are reordered according to these ELR values. The same is done for the columns of the matrix which represent the parts. This procedure starts again for rows and continues to produce a better BDF with each iteration. When, after ELR calculation, the rows or columns remain ranked (that is, reordering the matrix according to ELR values

produces no further changes), then matrix reordering cannot proceed and the analysis advances to Phase 1, step b. Dashed lines starting with the box containing 'PHASE 1C' represent stages of Phase 1, step c of this algorithm. The matrix reordering methodology for Phases 1a and 1c are shown together because the algorithm is the same in both cases. The only difference is that Phase 1c carries out matrix reordering using a different type of ELR value, as will be described later on in this sub-section.

The Phase 1a procedure described thus far tends to produce excellent BDFs, but the results are affected by the presence of exceptional elements; the more exceptional elements there are, the more difficult it is to get a BDF. Similarity coefficients identify which two lines (rows or columns) should be together regardless of the presence of exceptional elements. Given two lines, the similarity coefficient between them is as follows.

$$S_{ij} = \frac{N_{ij}}{\max[N_i, N_j]} \qquad \textbf{Equation 2.5}$$

where;

S_{ij} is the similarity coefficient between, say, two parts i and j,

N_{ij} is the number of workstations visited by both i and j, and

N_i, N_j are the number of workstations visited by i and the number visited by j

Note: The above is applied to workstations as well as parts

The problem with rearranging the BDF according to similarity coefficients alone, even after carrying out ELR, is that the block diagonal structure of the BDF tends to get dispersed. However, a similarity coefficient based weighting can be used to identify

the 'closeness' of two lines by combining their ELR values and the similarity coefficient between them as in the following equation.

$$W_{ij} = C_i + C_j(1 - S_{ij})$$ **Equation 2.6**

where;

W_{ij} is the weighted similarity coefficient between two lines i and j,

C_i, C_j are the ELR values for i and j, and

S_{ij} is the similarity coefficient between i and j

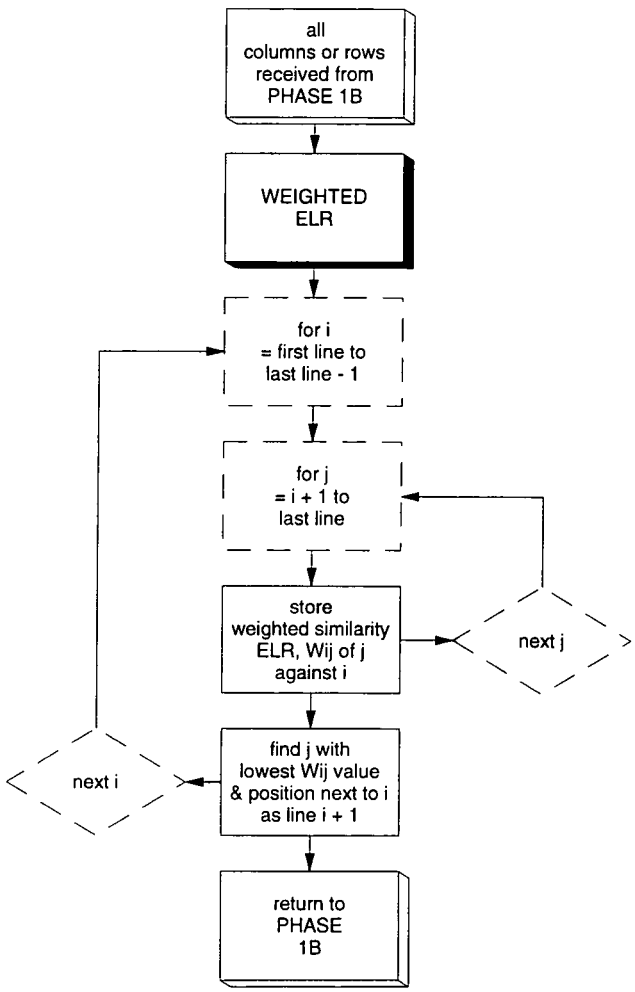


Figure 2.4: Positioning matrix lines according to weighted ELR values

Figure 2.4 shows how the matrix is scanned and reorganised to position lines with lowest weighted ELR values next to each other. The procedure for step b is as follows:

(i) carry out the steps in Figure 2.4 for parts

(ii) determine ELR values for workstations

(iii) carry out the steps in Figure 2.4 for workstations

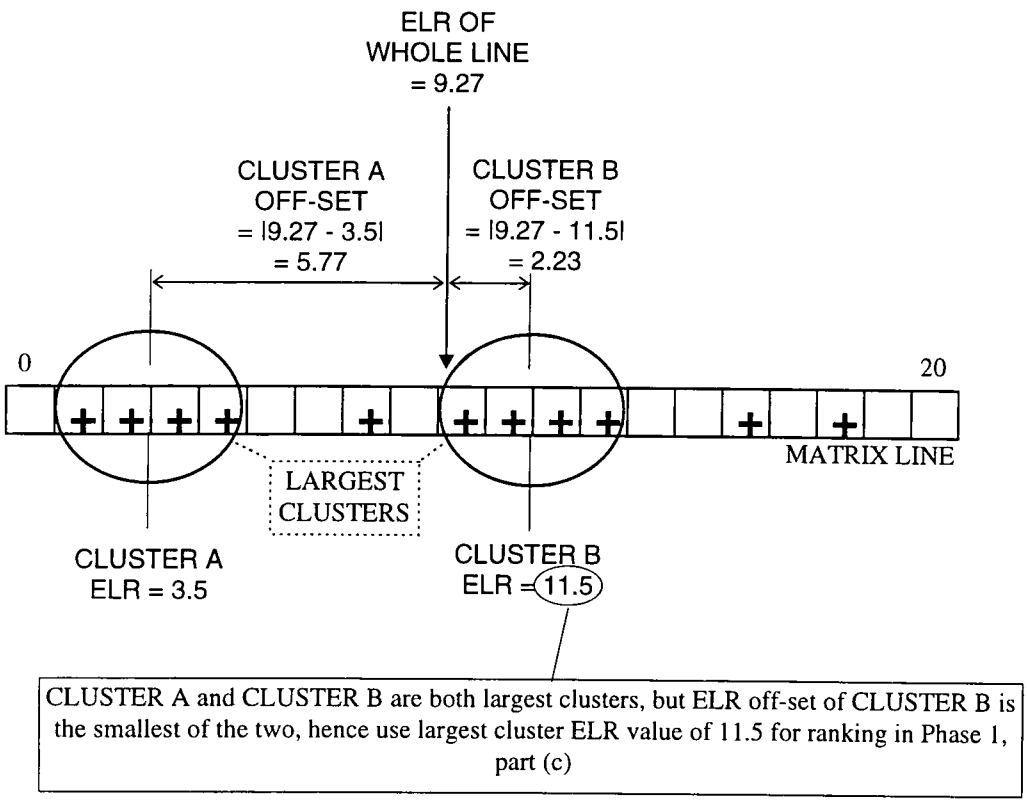


Figure 2.5: Determining the largest cluster ELR value of a matrix line

After carrying out step b, the matrix will generally not be in as much of a diagonal form as it was after step a was carried out; the grouping, however, will be slightly better. These tighter groups are used in step c to obtain the final BDF by carrying out ELR clustering using only the ELR values of the largest, uninterrupted group of ones

in each line. Where there are more than one largest clusters of equal size, the ELR off-set (absolute value of the difference between the ELR of the cluster and the ELR of the whole line) is used to identify which of the clusters' ELR values should be used. By carrying out this procedure, the algorithm ignores the effect of exceptional elements and clusters what appears at this stage to be the most likely workstation groups/part families. This procedure is called identifying the *largest cluster ELR*, an example of which is provided in Figure 2.5.

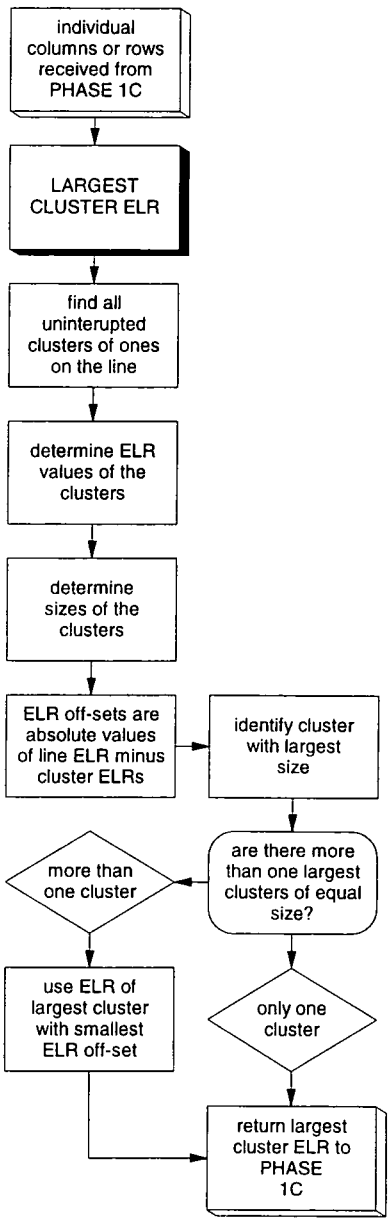


Figure 2.6: Determining largest cluster ELR value to be used in Phase 1, step c

In Figure 2.5 there are two clusters of equal size which can be used as the largest cluster in the analysis of Phase 1c. However, the diagram shows that the cluster nearest to the ELR of the whole line is the one that is selected for the analysis. It is this value that is used in the ranking described in Figure 2.3 (dashed lines starting with the box containing 'PHASE 1C'). When in Figure 2.3 the algorithm advances to 'do LARGEST CLUSTER ELR for parts' or 'do LARGEST CLUSTER ELR for workstations', then the procedure demonstrated in Figure 2.5 is carried out. The algorithm for this procedure of finding the largest ELR is shown in Figure 2.6. As already described and shown in this figure, for each line of the matrix the largest cluster is identified and if more than one of these is identified then the ELR off-set decides which cluster is used. The ELR of the selected cluster is used in the ranking of Figure 2.3.

2.3.2 Phase 2: Marking off the BDF

Phase 2 requires the marking off of the matrix into blocks. With most matrix reordering algorithms this is done by hand. However, in order to be able to analyse a large number of routing options, it is desirable that this should be done automatically. In this algorithm, marking off is done 'hands-free' using the following three steps:

(a) Create an initial matrix of candidate cell corners

(b) Use rules to identify other candidate cell corners and to eliminate unwanted cell corners

(c) Mark off and merge cells

Step a attempts to find breaks between distinct part families and machine groups of the BDF. For example, Figure 2.7 shows a near-perfectly structured BDF where candidate part families are the blocks formed as links in a chain stepping down in the horizontal direction from the top-left to the bottom-right. Similarly, candidate workstation groups step across in the vertical direction. The links when joined together should span across all the parts in the case of candidate part families and all the workstations for candidate workstation groups. Where candidate part families and candidate workstation groups intersect is marked as a candidate cell corner: squaring-off from one corner to the next defines the boundary of each cell. The reader should note that the middle circle in Figure 2.7 corresponds to an intersection between a candidate part family containing three parts and a candidate workstation group containing one workstation. Although the existence of this candidate workstation group is not clear in the figure, the reader should be aware that it must exist as a link between the second and third candidate workstation groups.

In Figure 2.7, the candidate cell corners are sufficient enough to produce satisfactory cells. However, for more complex matrices with indistinct blocks and a large number of exceptional elements, step b is required to identify candidate cell corners from matrix configurations such as those in Figure 2.8. These configurations are identified in order to eliminate candidate cell corners where these cause single-workstation clusters and to ensure that the top, left-hand corner of the BDF is a cell corner. The procedure for step b is outlined in Figure 2.9.

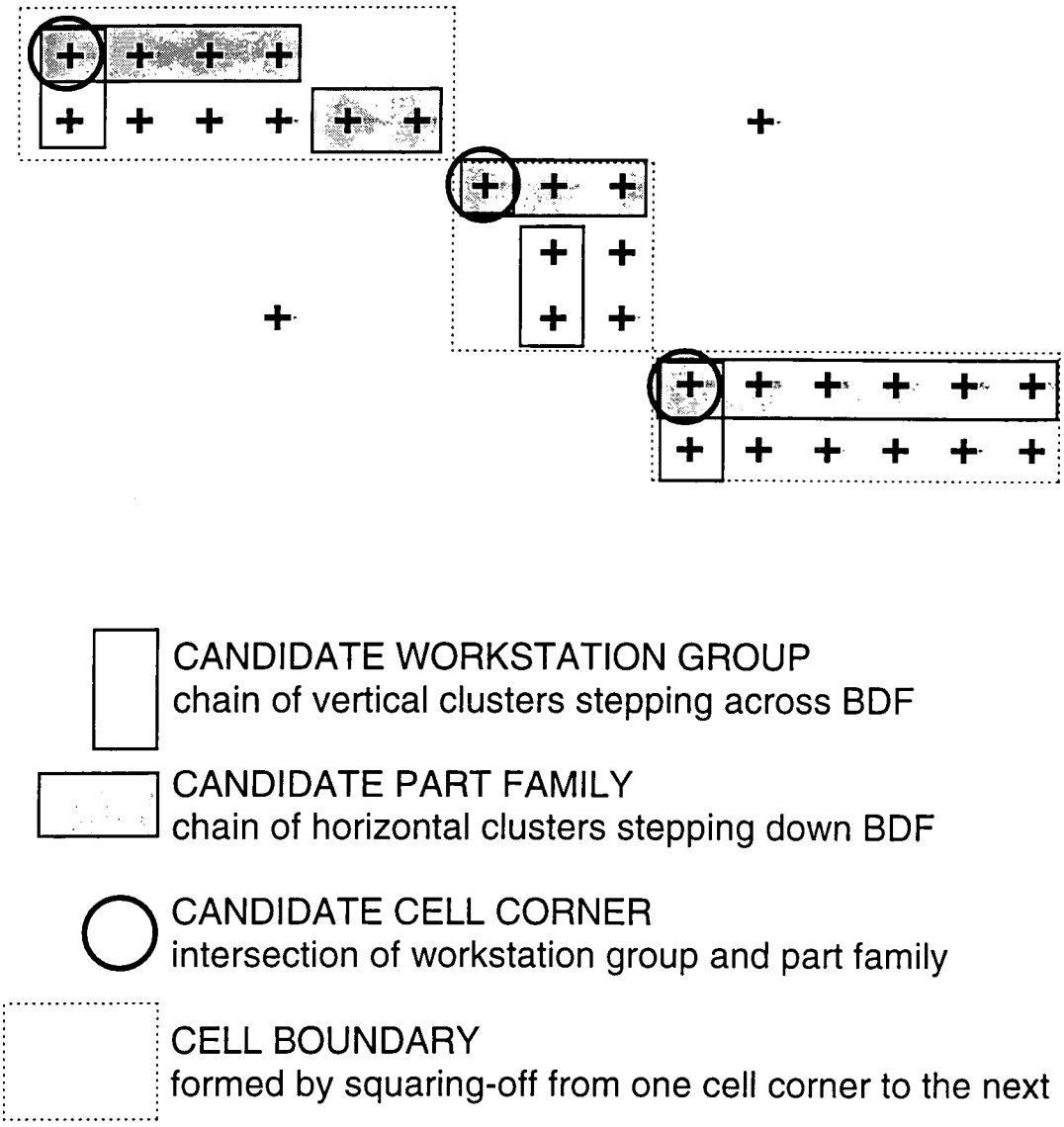
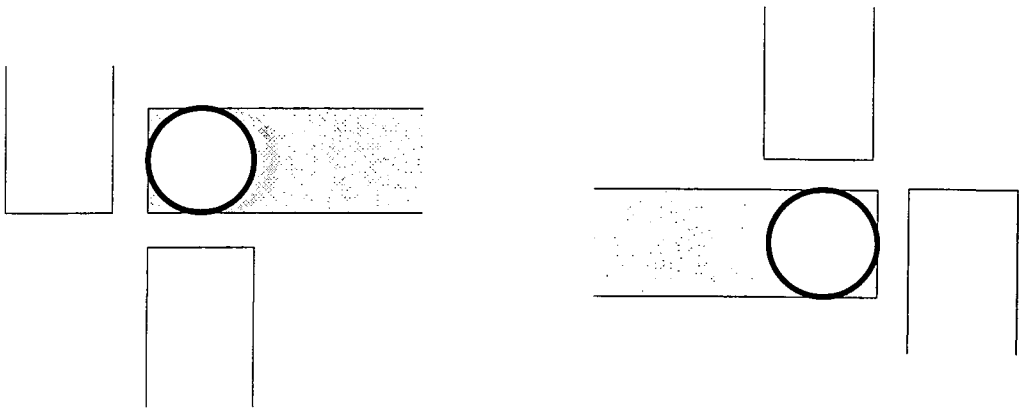


Figure 2.7: Identifying candidate cell centres

CANDIDATE PART FAMILY ELEMENT
SURROUNDED ON TWO SIDES BY CANDIDATE
WORKSTATION GROUP ELEMENTS BECOMES
CANDIDATE CELL CORNER



A ZERO SURROUNDED ON ALL SIDES BY
ONES BECOMES CANDIDATE CELL
CORNER

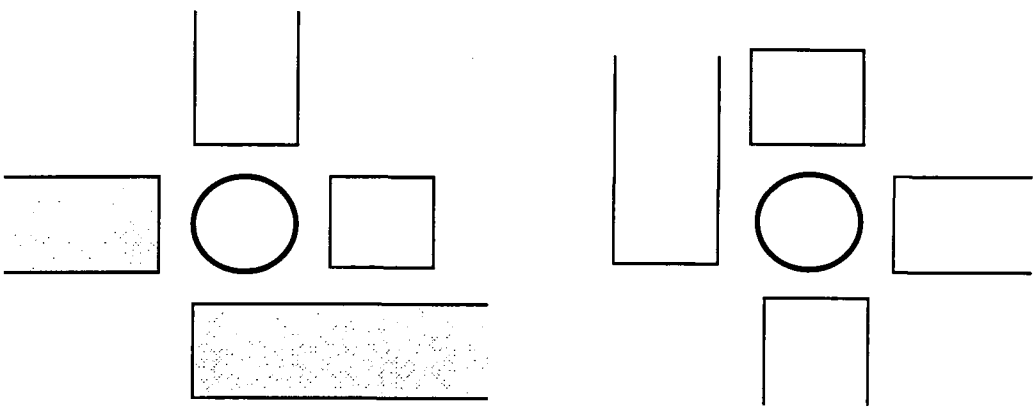


Figure 2.8: Rules to rationalise candidate cell centres

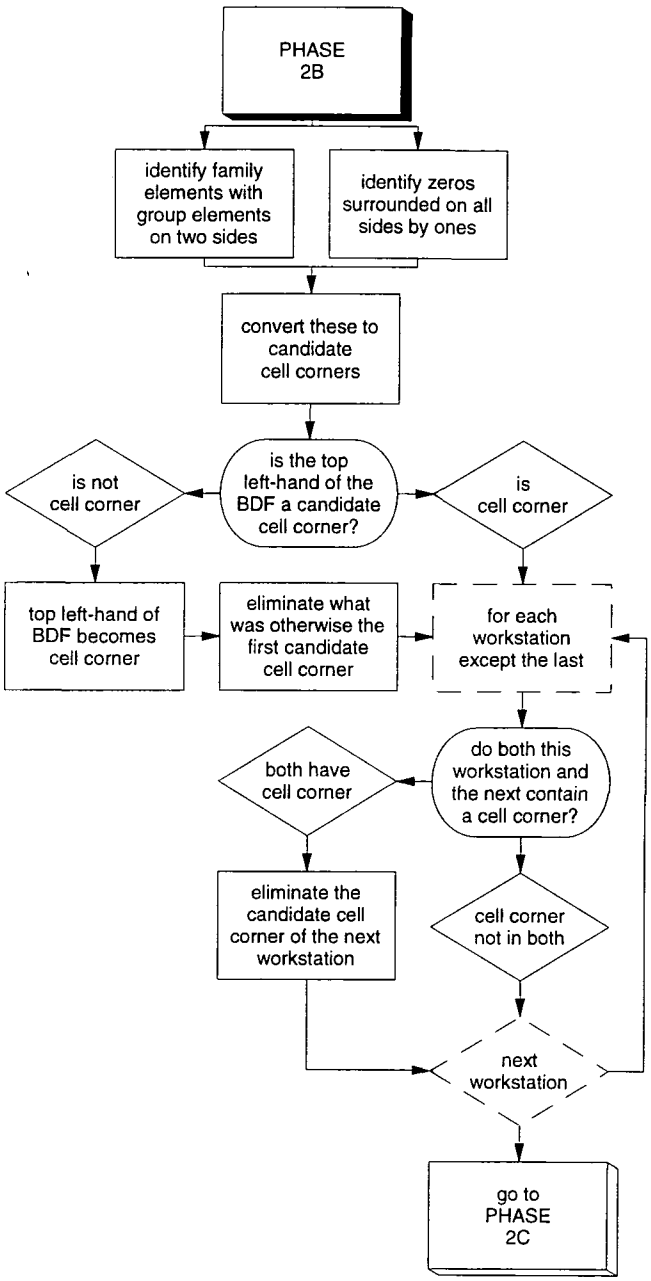


Figure 2.9: Phase 2, step b

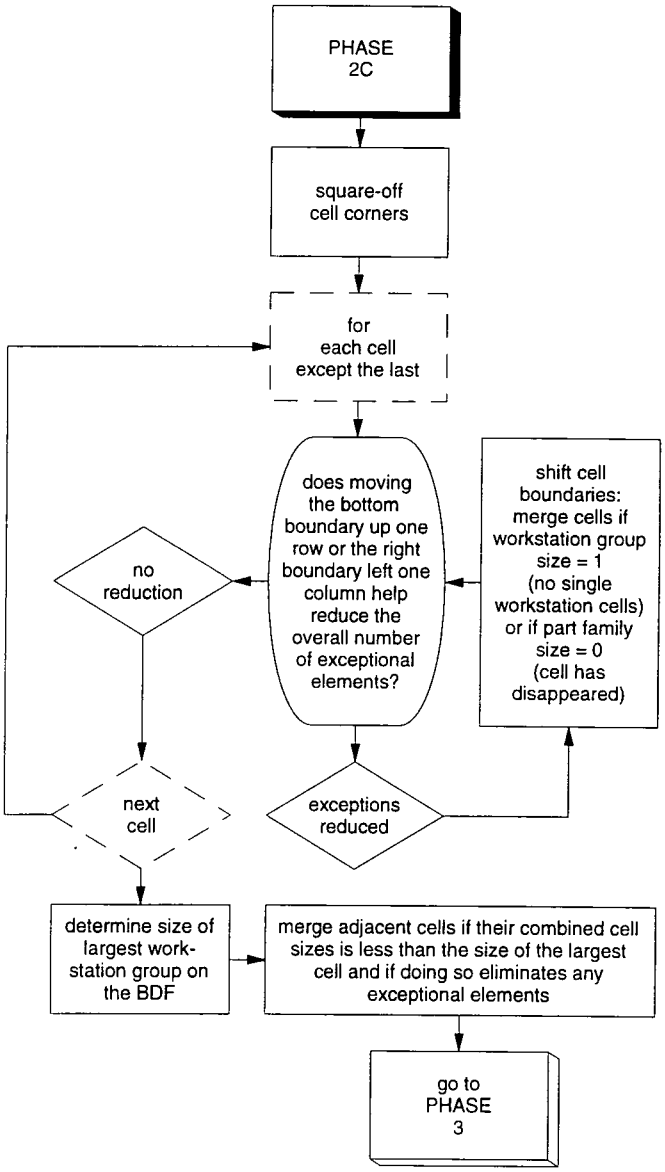


Figure 2.10: Phase 2, step c

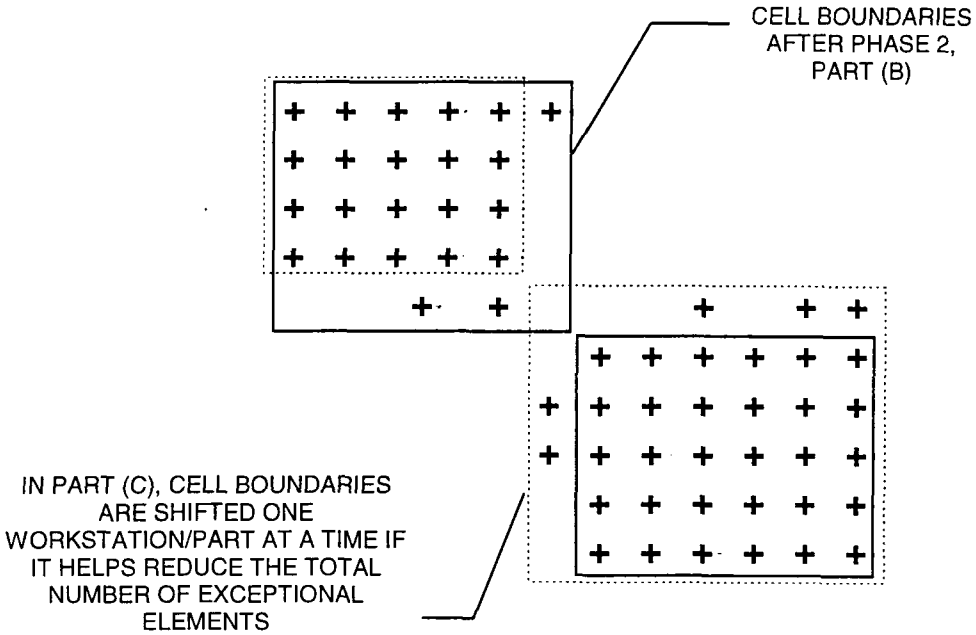


Figure 2.11: Shifting cell boundaries to reduce exceptional elements

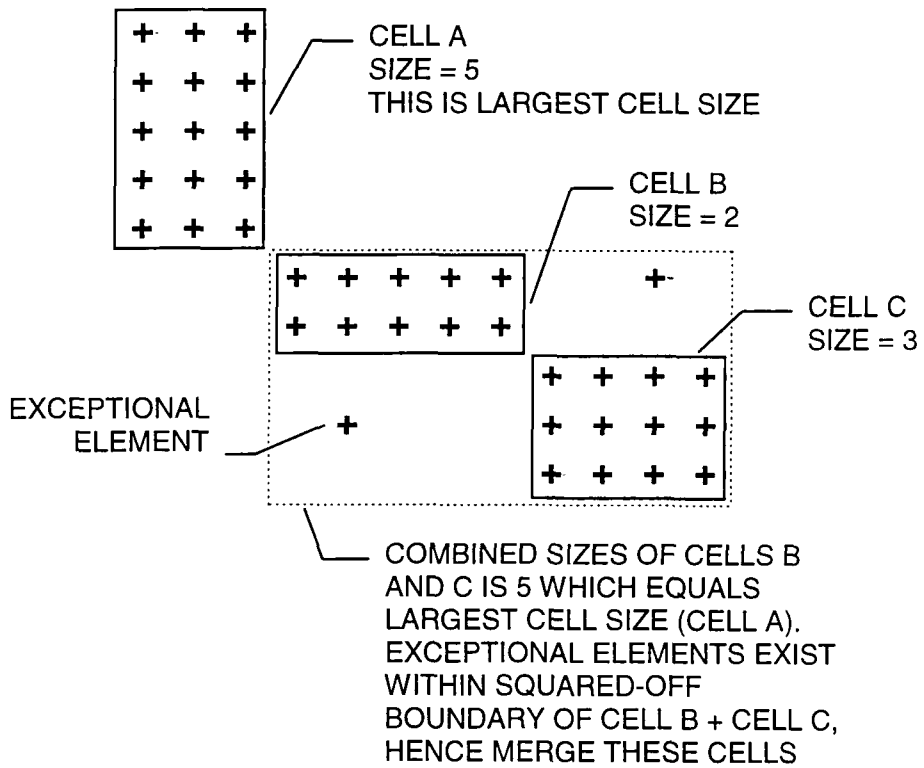


Figure 2.12: Merging cells to reduce exceptional elements

Step c of Phase 2 uses the remaining candidate cell corners to identify BDF boundaries. The overall procedure is shown in Figure 2.10. The objective of this step of the algorithm is, for each cell, to move cell boundaries one line at a time if each movement helps reduce the number of exceptional elements as shown in Figure 2.11. After this has been carried out, a largest cell size is identified and used to merge cells if it helps reduce exceptional elements as demonstrated in Figure 2.12.

2.3.3 Phase 3: Determining the quality of the BDF produced

An important function of Cellect will be its ability to be used in conjunction with CAPABLE. This function is enhanced by the fact that Black Box Clustering can work without user interaction thus allowing a number of different routing options to be analysed quickly and easily.

This approach differs from that taken by Adil et al (1996) and Kasilingam and Lashkari (1991) who attempt to cluster workstations and parts by identifying from a list of alternatives, those processes most conducive to workstation cell and part family grouping. This is because manufacturing costs and quality chain considerations are more important than the ability of routings to cluster. As such, CAPABLE identifies the most suitable set of routings based on cost, time and quality criteria which are passed on to Cellect to be clustered. If the routings are unsuitable for cellular manufacturing then the next best set of routings are clustered. If the routings are suitable for clustering despite the existence of exceptional elements and other inefficiencies, then Capability Analysis is used to identify where improvements could be made. CAPABLE and simulation can then be used to determine to what extent the

improvements can be carried out. Thus, at this stage it is necessary to assess only the quality of the BDF and consider other factors when carrying out Capability Analysis.

A variety of measures can be used to help identify the most suitable set of routings for cellular manufacture from the BDF alone (Chandrasekharan and Rajagopalan 1986b, Kandiller 1994). The ones used by Black Box Clustering are as follows:

(i) Total number of cells

(ii) Sizes of smallest and largest cells

(iii) percentage of exceptional element ({*total number of exceptional elements / total number of 'ones'*} * 100)

(iv) BDF density ({*total number of 'ones' within all the cells divided by total number of elements within all the cells*} * 100)

2.4 Algorithm Validation

The algorithm detailed in Section 2.3 was originally programmed in C and tested with the matrices used by authors of other clustering algorithms. In each case, a BDF was formed in less than a second and the results were as good as that of the authors' (in terms of the performance measures detailed above). Because the matrices used by these authors are usually selected to highlight the performance of their own particular algorithms (Kandiller 1994), the fact that Black Box Clustering can produce BDFs as good as that of the authors' algorithms demonstrates that this algorithm is versatile enough to cope with varied workstation-part input matrices in terms of size and complexity. To highlight this point, it is worth looking at some examples.

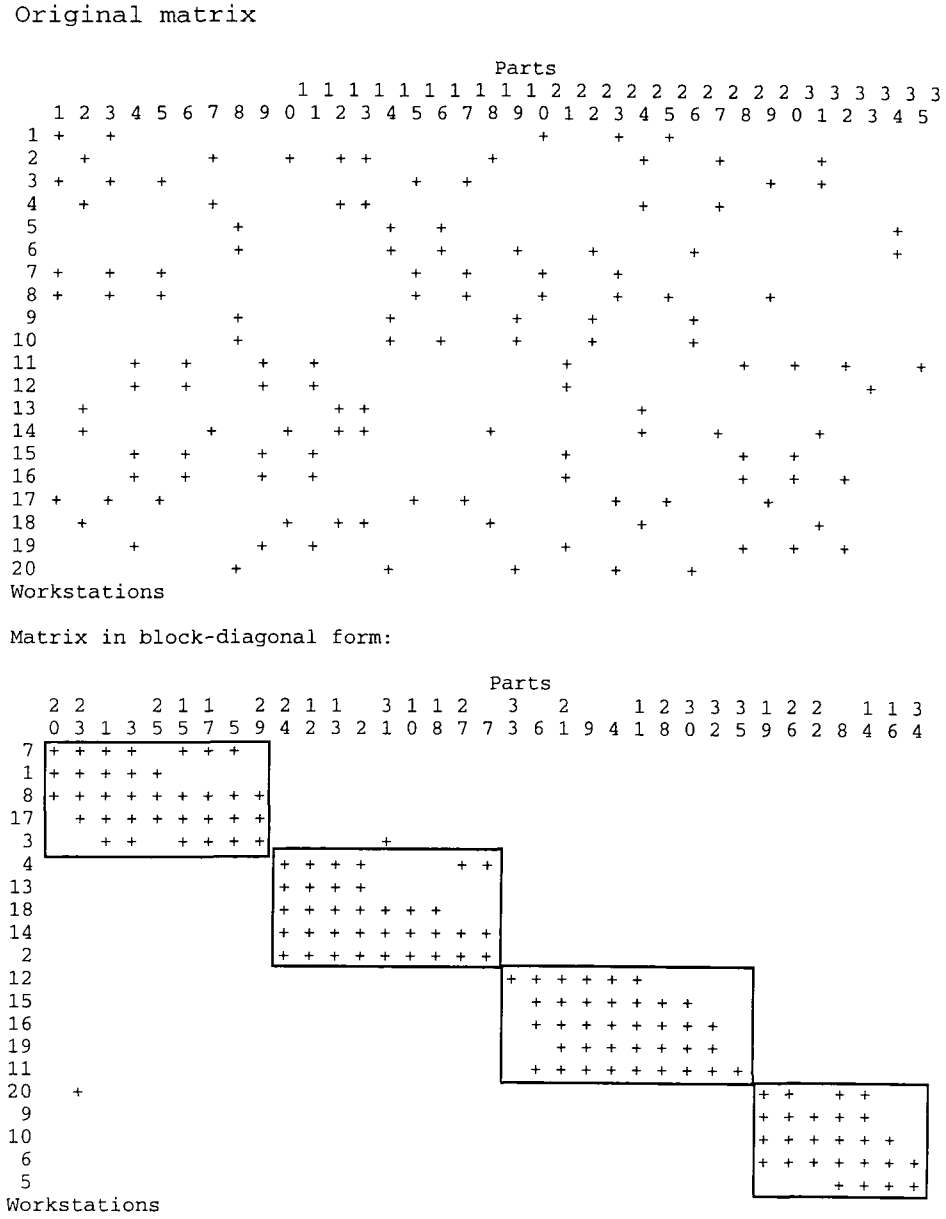
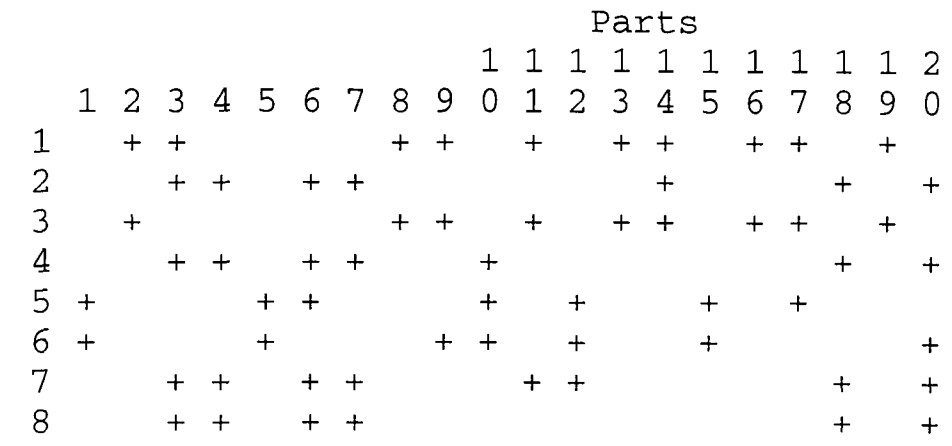


Figure 2.13: Black Box Clustering example 1

The first is that of Carrie (1973) used by Chandrasekharan and Rajagopalan (1986b) (Figure 2.13) which is the largest, but also the simplest example in that the blocks formed are very distinct with only two exceptional elements. Only Phase 1a is necessary to form the BDF which can be marked off using Phase 2a and the squaring-off in 2c. Thus for this simple example, only a small proportion of the algorithm

makes any difference; Phases 1b, 1c and 2b make no difference to the resulting BDF and its marking off.

Original matrix



Matrix in block-diagonal form:

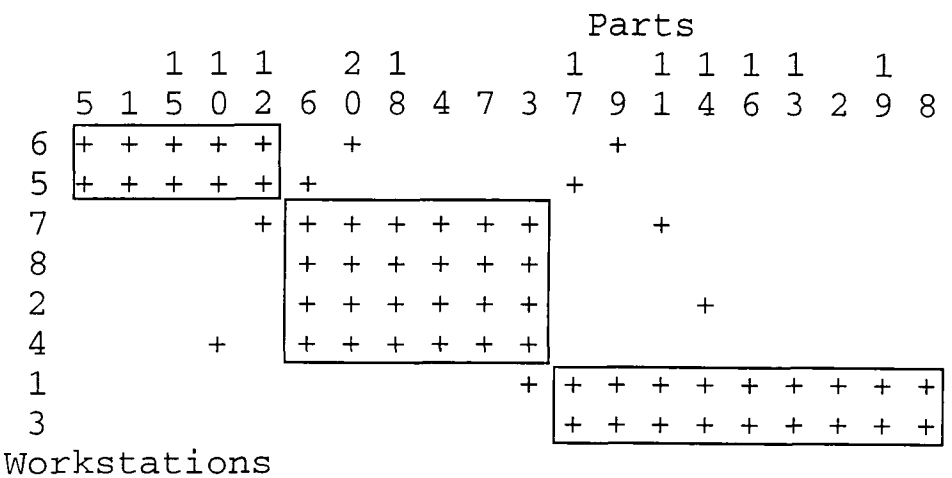


Figure 2.14: Black Box Clustering example 2

The next example is that of Chandrasekharan and Rajagopalan (1986a) (Figure 2.14) which has a much higher proportion of exceptional elements (14.75%), but because of its size and the distinctness of its blocks is easy to convert into a BDF using only Phase 1a. However, due to the positions of the exceptional elements within the matrix,

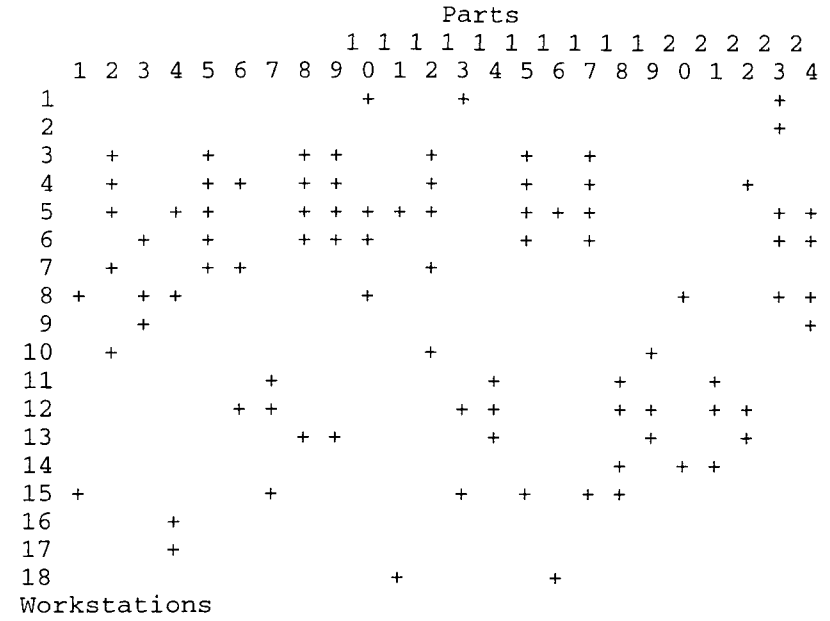
marking off using Phase 2a is insufficient to produce optimal cells. Using Phase 2b, this problem is overcome with the rules for converting recognisable matrix configurations into cell corners. The reader may wish to identify cell corners using Figure 2.8 described in Section 2.3.2 (note that only the top left diagram in Figure 2.8 applies).

As demonstrated with the above two examples, not all the steps of Phase 1 may be required to form a BDF. An example where this is not the case is that of Chen and Irani (1993) who test their algorithms using a complex matrix which Black Box Clustering tackled to produce cells as good as their best algorithm.

Figure 2.15 shows how this matrix developed through Phase 1 steps a to c to form the BDF shown. In Phase 1a, the matrix is reorganised according to ELR rankings alone, but it can be seen from the very indistinct BDF produced that Phase 1a alone is not enough to identify cell configurations. Phase 1b uses weighted similarity coefficients to tighten up the blocks and it can be seen in

Figure 2.15 that although the crosses have become more grouped, the BDF has become disrupted. Hence the requirement now is to restructure the BDF whilst maintaining the grouping developed thus far. In Phase 1c, the ELR values of the largest cluster of each line is used to carry out this task to produce a reasonably well structured BDF. It can be seen from the final BDF that the large number of exceptional elements and the low density of the blocks within the BDF are the complications requiring that all the steps in Phase 1 and all the rules in Phase 2 are used to identify cell configurations. It is because of its ability to adapt to situations of varying complexity that makes BBC such a versatile algorithm.

Original matrix



After Step a of Phase 1:

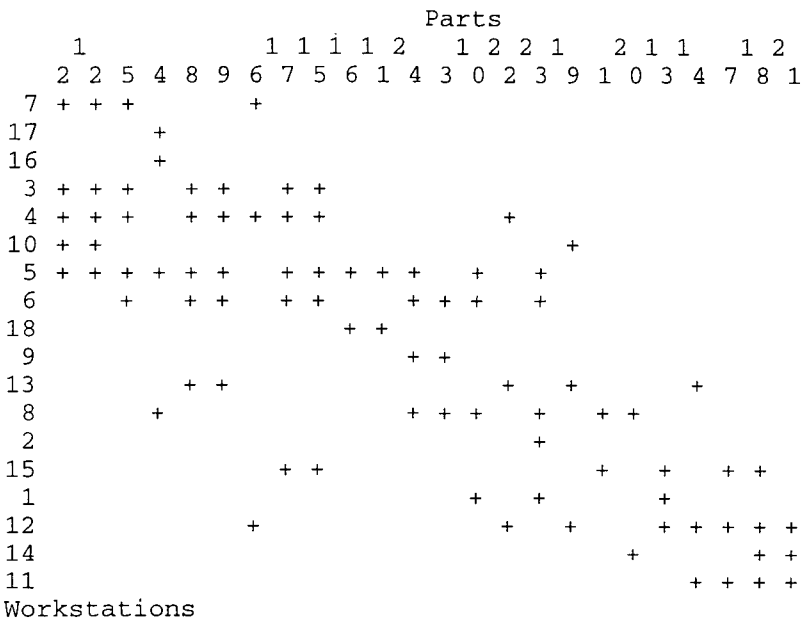
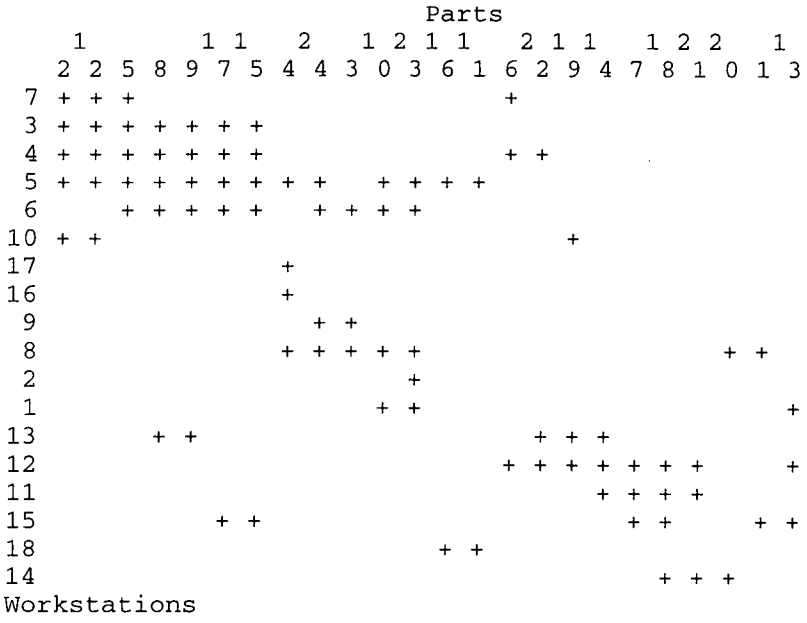


Figure 2.15 (continued on next page)

After Step b of Phase 1:



Matrix in block-diagonal form (after Step c of Phase 1) and marked off by Phase 2:

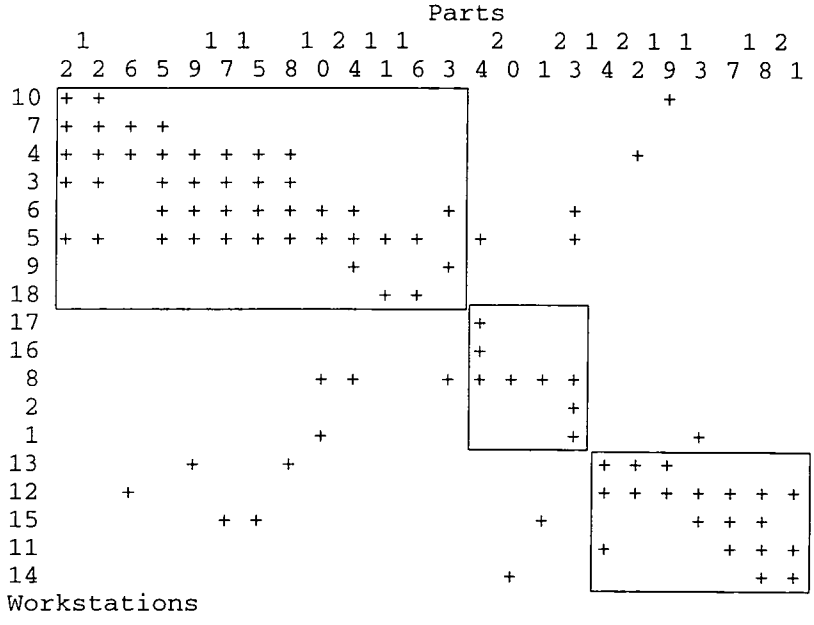


Figure 2.15: Black Box Clustering example 3

2.5 Summary

Black Box Clustering (BBC) is an algorithm that has been developed to be used by Cellect as part of an integrated database tool for the design, management and continuous improvement of cellular manufacturing systems. BBC carries out the first stage of the three stages of cell design described in Chapter 1, namely the identification of cell configurations. These are the groups of workstations that make up each cell and the groups of parts that represent each part family belonging to each cell. The aim of the algorithm is to effectively carry out clustering of workstation-part matrices into block diagonal forms (BDF), from which workstation groups and part families can be identified. The major restriction imposed by the requirements of Cellect was that Black Box Clustering should not rely on user interaction. This was achieved by developing a modified end load ratio (ELR) algorithm aided by the use of similarity coefficients. Marking off the resulting BDF to identify cell configurations was carried out using rules to identify cell corners between which the matrix is squared-off. The algorithm was tested with the matrices used by authors of other clustering algorithms and in each case, Black Box Clustering performed as well as the algorithm against which it was tested, hence demonstrating its versatility.

The integration of Black Box Clustering within Cellect is described in detail in Chapter 5. In the next chapter, the second stage of the cell design process is discussed. This involves the positioning of the cells and the workstations within them on the shop floor.

3. Cellect Layout Tools

3.1 Limitations of Automatic Layout Algorithms

Once cells configurations have been determined, it is necessary to position the cells and the workstations within them for optimal flow of parts. In this way the cell layout problem can be divided into two sub-problems:

(a) Intercell layout for positioning of cells on the shop floor, and

(b) Innercell layout for positioning of workstations within each cell.

These sub-problems are not mutually exclusive (Arvindh and Irani 1994) and must be carried out by taking into consideration factors such as:

(i) factory constraints,

(ii) material handling needs,

(iii) that it may be impractical to move certain workstations,

(iv) that workstations need to be positioned in a U-shape where practical for optimal flow of parts within the cell, and

(v) that workstations need to be positioned for ease of intercell flow of parts.

The more popular layout algorithms divide the factory into departments or blocks and attempt to position the blocks for maximum benefit of *quantitative* factors such as cost, distance and time and *qualitative* factors such as environmental considerations, ease of material and work force movement, and worker safety. Examples of such algorithms that were originally developed to design functional (job-shop)

manufacturing systems include CRAFT (Buffa et al 1964), CORELAP (Lee and Moore 1967) and ADELP (Seehof and Evans 1967). Other algorithms have also been developed specifically for the design of cellular system layouts (Abdou and Dutta 1990, Irani et al 1992, Leskowsky et al 1987, Tam and Li 1991, Jajodia et al 1992 and Tam 1992).

Overall, the major concern with existing layout algorithms is that the user is involved only in the early stages of the analysis. Since it is the shop floor staff who will work in the new layout, it is they who should decide the final locations of the workstations. Thus, in this respect, it is far better to determine approximate positions of workstations considering only quantitative factors (in particular, material handling cost) and then let the users consider qualitative factors during the physical process of machine positioning. In this way, the consideration of qualitative factors can be carried out far more accurately than if done using a computer, with the further advantage that the shop floor staff will feel some responsibility for the project (anon 1989).

However, because these algorithms consider the most important factor, namely ease of material movement, in terms of cost, distance and time functions, they do provide a useful insight into determining relative positions of cells and the workstations within them, but the overall layout produced should at best only be considered as a guide. As stated above, this is because the user needs to be far more involved in the assessment of qualitative factors which, generally, are too difficult to model using a computer.

With this in mind, the approach used by Cellect is to divide the shop floor into a set of points that define possible locations of cells (such as points on a grid) and use these to determine *approximate* positions of cells by assigning each cell to a point. The aim of cell positioning is to minimise overall material handling cost. As a separate task, it is

also necessary to determine the *relative* positions of workstations within each cell by identifying the sequence of workstations that gives the lowest overall material handling cost per unit distance. With this information the user can then manually determine precise workstation positions whilst taking into account all the qualitative factors that are so difficult to model. Thus, layout design is carried out in the two stage process described in the previous paragraph: analysis considering only quantitative factors then positioning considering also qualitative factors. The following sections describe how this is carried out.

3.2 The Cellect Layout Tools

The Cellect Layout Tools aim to provide the user with approximate positions of cells and the relative positions of the workstations within them. There are two layout tools, the first assigns cells to predefined positions on the shop floor and the second determines the sequence of workstations within each cell. Having carried out the two algorithms, the user is able to use the data produced along with the experience of all those involved in the layout project to determine the precise positions of the workstations in the analysis.

The quantitative factors chosen are material handling time, cost of material handling per unit time, and speed of material handling. As will be shown in Sections 3.2.1 and 3.2.2, these factors can be combined to form a material handling cost of a cell or workstation. These factors have been chosen to reflect the main objective of the analysis which is to reduce the overall amount of material handling on the shop floor. Including any other factors muddies this main objective. However, as stated above,

other factors have to be taken into account by the user at the end of the analysis in order to ensure a feasible, if not optimum solution.

The following subsections discuss the two Cellect Layout Tools. These are the Cell Positioning Tool and the Sequencing Tool.

3.2.1 Cell Positioning Tool

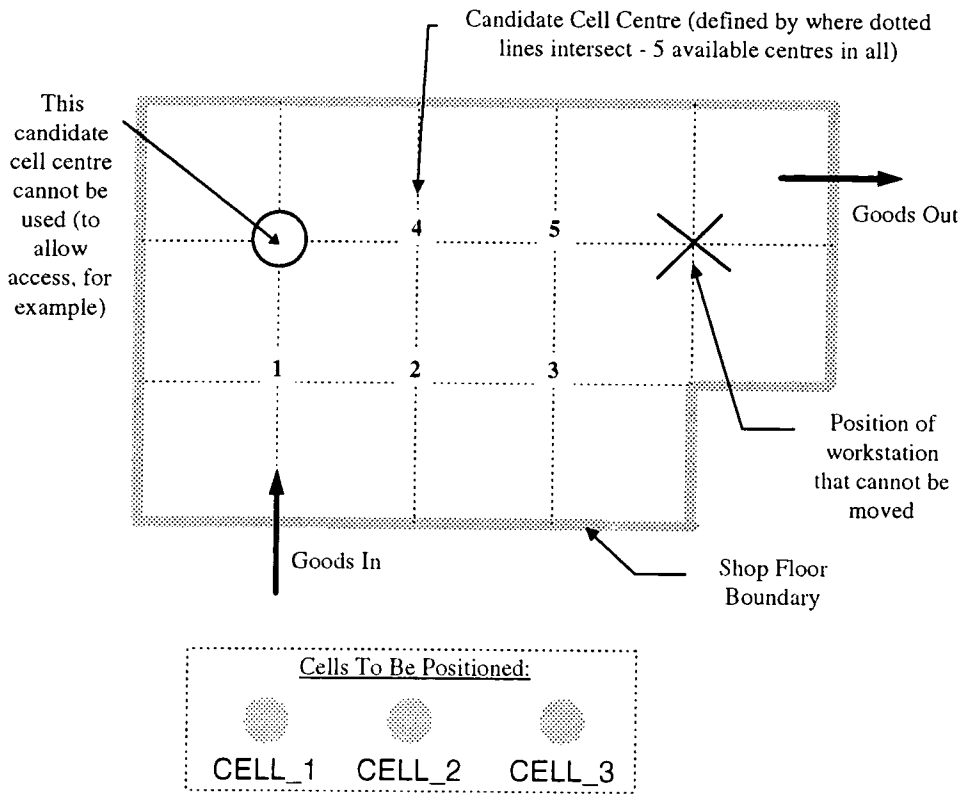


Figure 3.1: Example outlining the cell positioning problem

Firstly, the user identifies positions of candidate cell centres. These are likely positions of cells which initially can be entered semi-automatically as a grid of points with each point representing a candidate cell centre. Any points where a cell cannot be placed is deleted from the analysis by the user. Next, the user assigns co-ordinates to goods in/out and to workstations that should not be moved as shown in Figure 3.1.

The cell positioning tool determines for each cell the total material handling cost between existing positioned workstations (including goods in/out) and each of the candidate cell centres for all parts that visit the workstations belonging to the cell. This is demonstrated in Figure 3.2 in which CELL_1 is temporarily placed at candidate cell centre 2 as a step in the analysis (the analysis requires that each cell be positioned at each cell centre). The total material handling cost of all parts travelling between goods in/out or the already positioned workstation is calculated based on city-block (non-diagonal) travel. To determine material handling costs, it is assumed that each workstation within the cell has the co-ordinates of the candidate cell centre being tested. For workstations not yet positioned, material handling costs are zero.

For each part, the material handling cost between two workstations (or between a workstation and any other pre-defined point) is calculated from information within the Cellect database using the following formulae:

$$cd = \frac{Pct}{bv} \qquad \textbf{Equation 3.1}$$

$$d = |x1 - x2| + |y1 - y2| \qquad \textbf{Equation 3.2}$$

$$C_p = c_d.d \qquad \textbf{Equation 3.3}$$

where;

c_d is the total cost per distance per production period of a given part using a given method of material handling (£/m),

P is the period demand for the part,

c_t is the cost per minute of the material handling method (£/min),

b is the transfer batch size of the part,

v is the speed of the material handling method (m/min),

d is the city-block distance between two workstations (m),

(x_1, y_1) are the co-ordinates of first workstation (m, m),

(x_2, y_2) are the co-ordinates of second workstation (m, m), and

C_p is the material handling cost of the part between the two workstations per production period (£)

Note that in Equation 3.1, the cost per unit time, c_t is used as the cost variable. This was chosen as opposed to cost per distance, because it is more appropriate to determine material handling cost over a period of time from historical data, especially when taking into account time-dependent factors such as depreciation. From the discussion above, the total material handling cost of a given cell at a given candidate cell centre is:

$$C_c = \sum_{p=1}^{p=P} C_p \qquad \textbf{Equation 3.4}$$

where;

C_c is the total material handling cost (£) of a given cell, c at a given candidate cell centre,

p is a part that visits c,

P is the total number of parts that visit c, and

C_p is calculated as in equation 3.3 for p travelling between any positioned workstation (or other pre-defined point) and the candidate cell centre

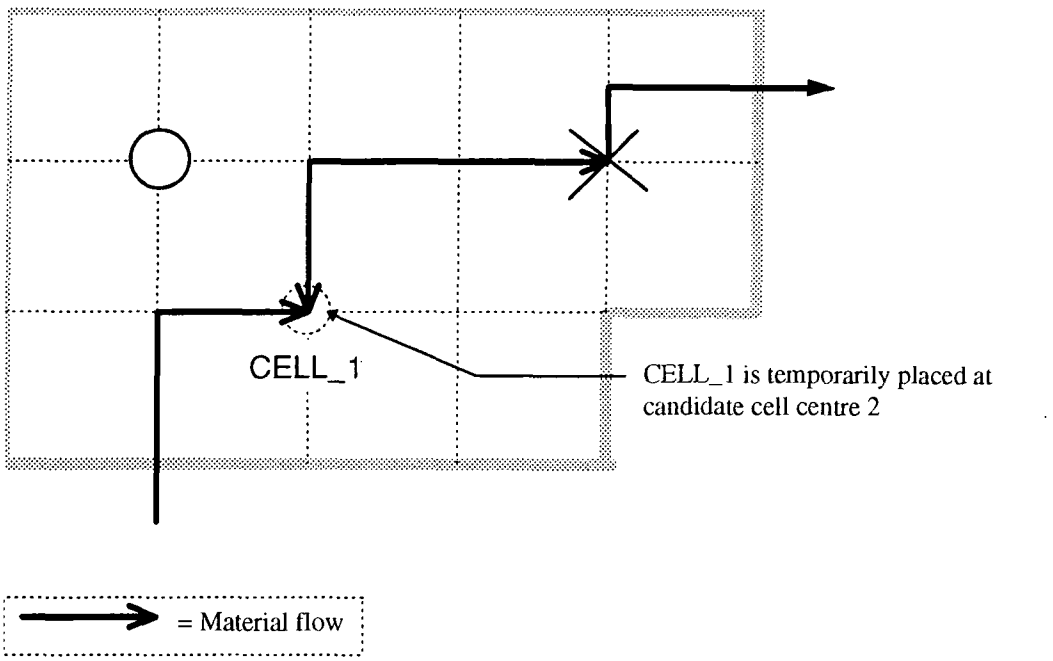


Figure 3.2: Material handling at a cell positioned at a candidate cell centre

Each cell is positioned at each candidate cell centre to determine material handling cost. The result for this example is shown in Table 3.1.

Candidate Cell Centre	CELL_1	CELL_2	CELL_3
1	**500 (L)**	**800 (L) (P)**	750
2	600	1000	800
3	750	1250	900
4	750	1100	700
5	900	1050	**600 (L)**

Table 3.1: Material handling matrix

For each cell, the candidate cell centre that gives the lowest material handling cost is identified. In Table 3.1, these are the elements in the table that are shown in bold, with (L) next to the total material handling cost (referred to as (L)-allocation). Out of these, the (L)-allocation with the highest total material handling cost is the one that is positioned at its corresponding candidate cell centre. In Table 3.1, this is the element that has (P) next to the (L)-allocated total material handling cost in CELL_2 (this

could be referred to as the (P)-allocation). Figure 3.3 shows the result of the (P)-allocation of CELL_2 on the shop floor. This method is used because by first determining (L)-allocations, the most suitable candidate cell centre is identified for each cell. The (P)-allocation then identifies the cell for which positioning at the corresponding candidate cell centre would be most cost-effective.

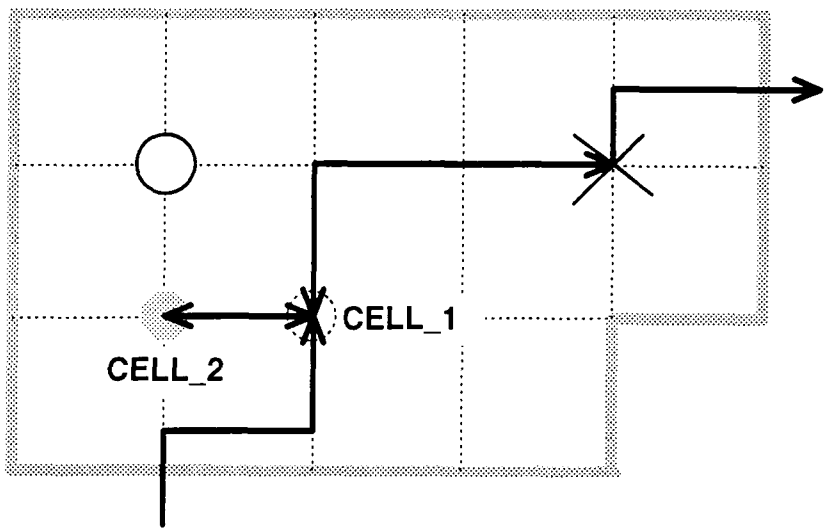


Figure 3.3: Effect on analysis when a cell has been positioned

All workstations within the (P)-allocated cell are assigned the co-ordinates of the corresponding candidate cell centre. This cell centre is now excluded from the analysis, and the procedure for assigning the remaining cells to candidate cell centres (as described in the second paragraph of this sub-section) continues until all cells have been positioned. Note that once a cell has been assigned a candidate cell centre, the workstations within that cell are used in the analysis to take into account intercell travel of parts (Figure 3.3). This should highlight the reason why (P)-allocation takes place only one cell at a time. Rather than attempting to position all the cells in one go, positioning one cell at a time means that as more information becomes available during

the progression of the analysis, the option of allowing unallocated cells to be positioned at various candidate cell centres is maintained.

The main advantage of this method for identifying cell layouts is the fact that it is run only once, allowing the user to position cells one by one; once a cell has been positioned, it remains at its allocated location. In this way, the layout algorithm proposed may be implemented manually, rather than having to use dedicated software such as Cellect. It may appear that the logic of this algorithm is flawed because initially cells are positioned without consideration for flows to and from other cells. However, to understand this the reader may wish to consider how a layout may be designed without the strict methodology of an algorithm. Initially, the first place a cell would be positioned is where there is the greatest concentration of material flow. In Table 3.1, all (L)-allocations are either at goods-in or goods-out, which is where one would expect most material flow to occur. Having identified where the cell is to be positioned, the next decision is to determine which cell to allocate to that position. If goods-in has been chosen as the candidate location, it then follows that the cell to be positioned near goods-in is the one that has the most material flowing between it and goods-in. This is the (P)-allocation discussed above. Once a cell has been positioned near goods-in then material flow to and from that cell should also be considered. As such the next cell will probably be positioned next to goods-in and the already positioned cell. This is the logic behind the further iterations that take place. The algorithm described simply presents this discussion in a more formal and generic manner.

3.2.2 Sequencing Tool

The sequencing of workstations within a cell is an independent task and as such this step of the layout algorithm can be carried out before or after the positioning of cells on the shop floor. This is not an ideal situation, because it does not take into account the positions of workstations in other cells and hence it cannot be carried out to minimise the effects of intercell travel. However, in an effective cellular manufacturing environment, cells should be as independent as possible to reduce the flow of material between cells. To aid in satisfying this objective, an effective method for determining cell configurations should be employed (such as the Black Box Clustering method discussed in the previous chapter) and where intercell travel exists, this should be eliminated by either changing the design of parts that travel between cells or by improving the capability of cells so that they can manufacture their part families independently. The Capability Analysis methods discussed in the next chapter address the issue of finding targets for improving cells in this way and ranking these targets in order of their priority for improvement. Thus, it is perhaps reasonable to carry out the tasks of cell positioning and workstation sequencing independently, particularly as these two methods act only as tools to aid (rather than carry out) shop floor layout design.

Workstation sequencing is carried out to minimise total material handling cost per unit distance. By doing this, assumptions are made that the workstations within a cell are of the same size and that the distances between them are equal. This obviously is not the case, but the idea here is not to determine the *exact* positions of the workstations within the cell, but to identify their *relative* positions. With this in mind, the first step is

to form a matrix of material handling cost per unit distance of parts travelling between all the workstations within the cell. This is done with information from the Cellect database using Equations 3.1 and 3.5:

$$Cs = \sum_{p_{cw}=1}^{p_{cw}=P} cd \qquad \textbf{Equation 3.5}$$

where;

C_s is the total material handling cost per unit distance (£/m) between two workstation, w_1 and w_2 in a given cell, c,

p_{cw} is a part that travels between w_1 and w_2,

P is the total number of parts that travel between w_1 and w_2, and

c_d is calculated as in equation 3.1

By minimising C_s between any two workstations, an attempt is being made to ensure that part flow through the cell is as smooth as possible (unidirectional). As such, if smooth part flow is the only factor of importance, the cost element can be eliminated from Equation 3.1. However, this is included to resolve a situation where a possibility exists for the allocation of more than one workstation to a given position within the sequence. An example of an innercell material handling cost matrix is shown in Figure 3.4.

```
Workstations to sequence
are A, B, C, D, E

Material handling costs
per unit metre are:

    A   B   C   D
   **************
B * 25
C * 5    9
D * 12  31  16
E * 16  18   3  10
```

Figure 3.4: Example sequencing problem

Having formed the matrix in Figure 3.4, the next stage is to identify the element within the matrix that has the highest overall total material handling cost per metre. The

```
The first workstations
to be sequenced are
those corresponding to
the element with the
highest value

    A   B   C   D
   **************
B * 25
C * 5    9
D * 12  31  16
E * 16  18   3  10

Workstation sequence so
far is : B-D
```

Figure 3.5: Sequencing of the first two workstations

workstations corresponding to this element will be the first to be sequenced. This is shown in Figure 3.5. In this example, workstations B and D have been sequenced together.

The next workstation to be sequenced can be placed beside workstation B or workstation D, but not between them. This decision is made be identifying which unsequenced workstation has the highest material handling cost per metre between itself and workstation B and which unsequenced workstation has the highest material handling cost per metre between itself and workstation D. Figure 3.6 shows that workstation A should be placed next to workstation B and workstation C should be placed next to workstation D. It has to be stressed that only two options exist for the allocation of workstations to a sequence. Thus, the sequencing of workstation E next to B is not an option even though the material handling cost per metre is greater than that for the possible sequencing of C next to D. This option is excluded, because workstation A has already been selected as the option for workstation B due to its greater material handling cost per metre.

```
B-D combination no longer
in analysis

Find elements with highest
values for sequencing next
to B or D

    A   B   C   D
   **************
B * 25
C * 5    9
D * 12      16
E * 16  18   3  10

Possible sequencing is A-B
or D-C

Sequencing takes place only
one workstation at a time:
A-B(25) > D-C(16) so
sequence A next to B

Workstation sequence so far
is : A-B-D
```

Figure 3.6: Sequencing of the third workstation

Although two options exist for the allocation of workstations to a sequence, to allow the analysis to sequence workstation A with workstation C if required, sequencing should only take place one workstation at a time. This is determined by which of the two options has the highest value in its corresponding element. In this case, workstation A is sequenced next to workstation B (Figure 3.6), whereas workstation C is left in the matrix as an unallocated option.

Figure 3.7 and Figure 3.8 show that sequencing continues in the same way until all workstations have been allocated a position. Note that in Figure 3.7, the decision to sequence workstation E next to A, as opposed to sequencing workstation C next to D is purely arbitrary. The algorithm could be modified to resolve such conflicts by perhaps allocating the workstation for which the number of parts transferred is greatest, or for which the profit contribution (discussed in the next Chapter) of the parts being transferred is greatest, or according to any other user-defined criteria. The final sequence of workstations in this example is E, A, B, D and C

```
B-? and D-A combinations
no longer in analysis

Find elements with highest
values for sequencing next
to A or D

     A    B    C    D
    **************
B *
C * 5
D *            16
E * 16          3   10

Possible sequencing is E-A
or D-C

Sequencing takes place
only one workstation at a
time:
E-A(16) = D-C(16) so
arbitrarily choose to
sequence A next to B

Workstation sequence so
far is : E-A-B-D
```

Figure 3.7: Sequencing of fourth workstation

```
A-? and D-E combinations
no longer in analysis

Find elements with highest
values for sequencing next
to E or D

     A    B    C    D
    **************
B *
C *
D *           16
E *            3

Possible sequencing is C-E
or D-C

Determine where remaining
workstation is to be
positioned
D-C(16) > C-E(3) so
sequence C next to D

Final sequence of
workstations is : E-A-B-D-
C
```

Figure 3.8: Sequencing of last workstation

(this is shown in Figure 3.8).

3.3 User Interaction With The Cellect Layout Tools

Initially, the user is required to ensure that the Cellect database is up to date and in particular has the appropriate material handling data, positions of input/output points, co-ordinates of any workstations that have been fixed or already positioned and a list of candidate cell centres. The cell positioning tool then determines approximate positions of cells on the shop floor by positioning cells at suitable candidate cell centres and the sequencing tool determines relative positions of the workstations within each cell.

Once the Cellect Layout Tools have completed their tasks, the user is then required to define the exact locations of the workstations. These will be positioned as near as possible to their corresponding cell centre, in the sequence prescribed by the tool and if possible, in a U-Shape to aid material handling and use by multi-skilled operators (Sekine 1992). Once the workstations have been located, it may be of benefit to identify new candidate cell centres as the exit/entry points of the cells (or use their actual centres) and then rerun the Cell Positioning Tool to confirm that cells are ideally positioned.

3.4 Summary

This Chapter concentrated on the second stage of cell design, namely the positioning of the cells on the shop floor. It described two cost-based algorithms, called the Cellect Layout Tools, for determining approximate cell positions and the sequence, or relative positions, of the workstations within them. It then becomes the responsibility of the

user to determine exact locations of the workstations by taking into consideration other, mainly qualitative factors.

In Chapter 5, the Cellect Layout Tools are applied to an industrial example. Before that, in the next chapter, Capability Analysis methods for determining targets for continuous improvement will be discussed, an objective of which is to increase the independence of cells. As has been discussed, this is a key requirement for accurate use of the Cellect Layout Tools.

4. Capability Measurement for Cellular Manufacturing Systems

4.1 The Need For Manufacturing Capability Analysis

Cell design does not end when workstations have been positioned. Because of the ever-changing environment within which a manufacturing organisation operates, it is essential to define procedures for the measurement of capabilities in order to identify the organisation's ability to adapt to the various situations it may encounter. In other words, it is necessary to identify the ability of the manufacturing organisation to achieve its strategic objectives. This chapter concentrates on one aspect of assessing this ability and provides a methodology to determine how well a manufacturing organisation is performing in terms of cellular manufacturing objectives. Measures of performance in terms of these objectives are used to identify the organisation's manufacturing capability.

Capability is the extent to which an organisation is achieving required performances with respect to pre-defined criteria. Capability Analysis (CA) are the methods for determining capability.

The calculation of capabilities is essential for the efficient management and continuous improvement of a cellular manufacturing system and, in general terms, is achieved in three steps: 'data recording, data analysis and problem solving' (Schonberger 1986). More specifically, this can be interpreted as:

(i) Identify the criteria required for the analysis within the context of the organisation's strategic objectives and extract the data that define these criteria.

(ii) Carry out Capability Analysis with respect to the identified criteria such that realistic targets for improvement are determined.

(iii) Carry out improvements and update data for further analysis.

This Chapter will present a methodology for carrying out CA. Section 4.2 discusses the selection of factors for the analysis, Section 4.3 discusses previous methods of assessing performances, Section 4.4 examines the objectives of CA, Section 4.5 discusses the details of the CA proposed in this research and Section 4.6 demonstrates CA using a simple example. A more realistic implementation of CA is discussed in Chapter 5.

4.2 Deciding What to Measure

The performance measures used must be chosen to reflect the context within which the analysis takes place. This is recognised in terms of general business performance measurement by Schonberger (1986), Lynch and Cross (1991) and Maskell (1991) who all emphasise the need for strategic objectives to be reflected within performance measures. More specifically, within the scope of cell design, Williams et al (1993) determine cell definitions and criteria by first establishing 'essential company features, aims and philosophies'. The performance measures used within Cellect reflect the objectives of cellular manufacturing discussed in the previous chapters and are presented in Section 4.5.1.

4.3 A Common Method of Analysis for all Performance Measures

Performance measuring has traditionally been carried out within a company using management accounting which provides financial information to managers at various levels and customised to suit the needs of the parties to which they apply (Lanigan 1992). Disadvantages of management accounting have been outlined by Maskell (1991) who includes in his list the fact that this form of measurement is restricted by the needs of financial accounting and as such is inflexible, irrelevant and impedes progress in world class manufacturing.

Within the context of manufacturing systems and cell design, performances have traditionally been determined using mathematical methods (Askin and Stanbridge 1993, Logendran 1993, Benjaafar 1995) and simulation (Morris and Tersine 1994, Massay et al 1995, Chan et al 1995). These methods suffer because they do not tell the user how to improve the system; they simply identify the effects of different changes. Attempts have also been made to carry out cell design using expert systems such as the rule-based system of Basu et al (1995) and the simulation/neural net/rule-based hybrid system of Chen and Sagi (1995). These provide suitable frameworks to carry out cell design that consider strategic objectives as well as technical factors. However, none of these systems attempt to identify capabilities and hence are unsuitable for every day use within a changing manufacturing environment into which new products are being introduced and where demands change as the different products progress through the various stages of their life cycles.

4.4 Objectives of Capability Analysis

The purpose of CA presented in this project is to provide a generic methodology for comparing performances of different factors measured in different ways. The aim is to provide a list of targets for improvement prioritised in such a way so that the target at the top of the list is the one in most need of improvement and the target at the bottom of this list is the one with least need for improvement. The concepts of CA will now be briefly summarised within the context of the above discussion. These concepts and the terms in *italics* are explained fully in Section 4.5.

4.4.1 Performance Indices in Capability Analysis

The steps required to compare different types of data within the manufacturing system are as follows.

- Referring back to Section 4.1, it can be seen that the first step of CA is to identify the criteria to be included in the analysis.
- In general terms, each specified criterion is a *capability factor* and the measure (or value) of any item of data for that capability factor is its *capability score*.
- A *collated* group of capability scores should all be similar. Preferably all capability scores within a collated group should be the same as the group's *required capability score*.
- Capability scores are defined so that any score above the required capability score is a *capability deficiency* and any score below is a

capability excess. Capability analysis targets only those scores that have a capability deficiency.

- In order that CA can be used for analysing different capability factors using the same methodology, a *marginal capability* is defined. This is a capability score expressed as a percentage of the *required capability score* for the *collated* group to which the capability score belongs.

- By taking into account the *improvement potential* of the capability score and importance to the manufacturing system (identified using *factor weighting* and *profit weighting*), a marginal capability can be converted into a *priority confidence score* (PCS).

The PCS is in effect a performance index, the advantages of which are highlighted by Lynch et al (1991) and are management summary, showing overall performance and not overwhelming with detail. The same authors also give disadvantages of performance indices as burying critical information and obscuring needed actions. CA overcomes these disadvantages by being structured in such a way that performances are shown at various *capability levels* such that higher levels provide greater overview and lower levels provide greater detail. This is done by having a *recovery schedule* for each capability level. A recovery schedule is a list of priority confidence scores for a given capability level ranked so that the highest PCS (target in most need of improvement) is at the top of the recovery schedule and the lowest PCS is at the bottom. By using the concept of *transparency* it is possible to take a target that is

shown at a higher level and break it down into its constituent targets at more detailed, lower levels.

4.4.2 Carrying Out Improvements

Continuous improvement should be carried out within a defined perspective. According to the Law of Diminishing Returns, increases in benefits become smaller for proportional increases in resources allocated to a given task (Woolfe et al 1987). Hence, for example, it is both costly and ineffective to aim for a set-up time of zero for all operations when the single-minute changeover defined by Shingo (1986) may be adequate. In the same way, it is necessary to avoid the Icarus paradox, whereby actions that contributed to success are extended to the point where they cause decline (Miller 1990). Thus, for each group of capability scores that should be similar (the *collated* group) there should also be identified a cut-off point beyond which resource allocation for improvement is inadvisable. This is the *required capability score* and forms the focus of all CA activities. Required capability scores, although not ideal measures, should reflect the best improvement that the system is capable of achieving at that time. Within Cellect, a required capability score for a collated group of capability scores is defaulted to the best capability score within that group. In other words, if the user is not able to supply Cellect with a level of performance that the system *should* be achieving then required capability scores are the best levels of performance that the system *is presently* achieving. If the default required capability scores are unsatisfactory then the user can alter them to reflect more realistic or ambitious improvement requirements using reference data such as the results from previous improvement projects, vendor data or benchmarking data.

CA does not implicitly address the human aspects of continuous improvement. Of particular importance is the fact that in order to carry out the improvements that are the targets in the recovery schedule, it is necessary to have in place a management structure that allows all employees to be involved in the activities that streamline the manufacturing organisation. With a tool such as CA that relies on different types of information, a method of feedback through the various management levels should be in place so that every employee feels he or she is part of the system and is thus encouraged to suggest and carry out improvements to better the organisation (anon 1989). It is probably because cellular manufacturing makes such a method of feedback more practical that there is little resistance to the change that takes place when converting from a functional to a cellular layout (Wemmerlov and Hyer, 1989).

According to Fan and Gassmann (1995), improvements are tackled more effectively if employees are able to measure the amount of improvement that takes place. In this way it is a desirable feature of CA that all improvements are carried out to attain specific levels of performance, as defined by the required capability scores. More specifically, the use of required capability scores that are generally identified from successful improvements (best capability scores) provides a method that allows continuous improvement to be carried out in a piece-meal fashion and allows progress to be assessed against specific targets. Rather than stating that an overall improvement in set-up time is required, it is far better to state that 'our present target is to reduce the set-up time of workstation x by y seconds'. Further more, the fact that required capability scores, particularly those defined as best capability scores, are continually reassessed shifts the management of improvement away from achieving overall improvement for all workstations (for example) to targeted improvement of individual

workstations. This suits the 'situational management' approach of Lean Production methods that renders manufacturing practices tacit (Johnston 1995). It is only in this way that effective cellular manufacturing can take place and it is for this reason that the abolishment of planning-centred production is Sekine's (1992) first objective for achieving one-piece production.

The above considerations have all been taken into account when developing the concepts of CA now discussed in detail.

4.5 Capability Analysis

The purpose of CA carried out within this project is to examine the performances at the operation, part, product, workstation and cell levels of a manufacturing system and assign a priority to each performance to form a ranked list of actions to be undertaken at each of those levels. The objectives of CA are:

(i) To obtain the most benefit from a given resource allocation.

(ii) To build flexibility into the manufacturing system.

(iii) To create responsive systems lowering the time taken to work up to full production levels when introducing new products.

The following subsections discuss the concepts and terminology of CA.

4.5.1 Capability Factor

A capability factor is a criterion for capability assessment.

The capability factors used within Cellect are operation cost, processing time, set-up time per part, number of defects, time spent producing defects, cost of defect production, workstation down time, labour time, labour cost, part similarity, workstation usage, nominal part lead time, material handling distances, material handling times and material handling costs. These are defined and discussed in Section 4.5.4.

4.5.2 Capability Score (s)

Capability score is a value of a capability factor.

For example, 0.25 minutes per part is a capability score for the set-up time capability factor of a given operation, 10 minutes per production period is the capability score for the down time for maintenance capability factor of a given workstation, and 300 metres per production period is the capability score for the intercell travel distance capability factor of a given cell. In order to ensure the consistency of CA, particularly when defining improvement potential (explained later in Section 4.5.8), capability factors should be defined so that their optimum capability scores equal zero. For example, for effective cellular manufacturing, each workstation within a cell should serve all the parts belonging to the cell's part family. Let

$$m_p' = \frac{pw}{pcw} \qquad \textbf{Equation 4.1}$$

where;

m_p' is the part similarity of workstation, w (initial definition),

p_w is the number of parts that visit w, and

p_{cw} is the number of parts that visit the cell, c to which w belongs

This should be adequate enough to describe the extent to which a given workstation serves a cell's part family. Given that it is desirable to ensure that the block diagonal form of a workstation/part matrix is dense (see Chapter 2), an optimum condition exists in terms of the part similarity capability factor if each workstation serves all members of the cell's part family. To describe this ideal condition in such a way that optimum capability score is zero, it is necessary to define capability scores for the part similarity capability factor as

$$m_p = 1 - m_p' \qquad \textbf{Equation 4.2}$$

where;

m_p is part similarity capability score

Because in equation 4.1, an optimum situation occurs when part similarity is one ($p_w = p_{cw}$), equation 4.2 is used to define the part similarity capability factor used in this project. With this definition, optimum capability score for part similarity capability factor will equal zero, as required to ensure CA consistency.

Capability scores used in Cellect are discussed in greater depth in Section 4.5.4.

4.5.3 Units And Capability Levels

A cellular manufacturing system, as modelled in this project, is made up of *cells*. Each cell contains a group of *workstations* and each workstation carries out a set of *operations*. A manufacturing system can also be considered as containing *products* and part families (defined as the group of parts belonging to a *cell* - see Chapter 2) made up of *parts* manufactured by a set of *operations*.

The model of a manufacturing system in this project comprises five units. These are cell, workstation, product, part and operation. Units represent the elements of manufacturing that CA seeks to improve.

Capability factors are defined so that their capability scores correspond only to the unit that the capability factor seeks to address. For example, travel distance for a given part is defined as the total distance travelled by material between the operations required to manufacture the part. Travel distance for a product, on the other hand, is defined as the total distance travelled by all of the parts that make up the product.

A capability level is a group of capability factors that address a given unit. Hence, in this project, there are five capability levels: one for each unit.

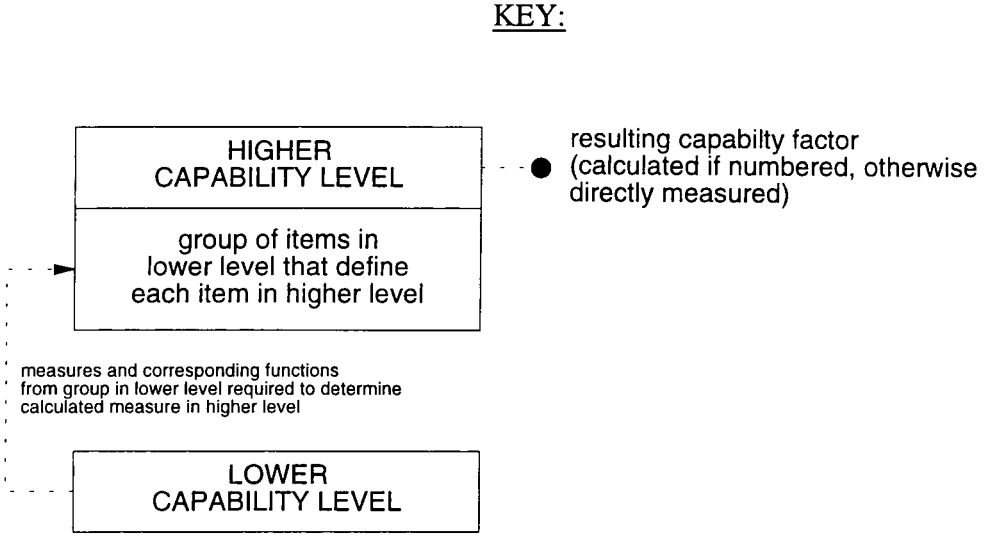
Capability scores can only be measured at capability level. For example, workstation usage capability scores can only be measured at workstation level, operation cost capability scores can only be measured at operation level, intercell travel distance capability scores can only be measured at cell level and so on. However, CA also provides a methodology to allow capability factors measured at one capability level to be represented at higher levels. These are the calculated capability scores discussed in the next section. For example, CA is able to determine the material handling cost capability of an operation relative to other operations that should be similar and then determine the material handling cost capabilities of the operation's corresponding workstation, cell, part and product. The abstraction of data at various levels can be expressed in terms of one level being higher than another. For the above example, a material handling cost for a cell (sum of all material handling costs associated with the cell) is more aggregate than the material handling cost associated with an operation taking place in the cell. The same applies to the other cell level capability factors as

will be shown in the next section. Hence, cell level is said to be higher than workstation level.

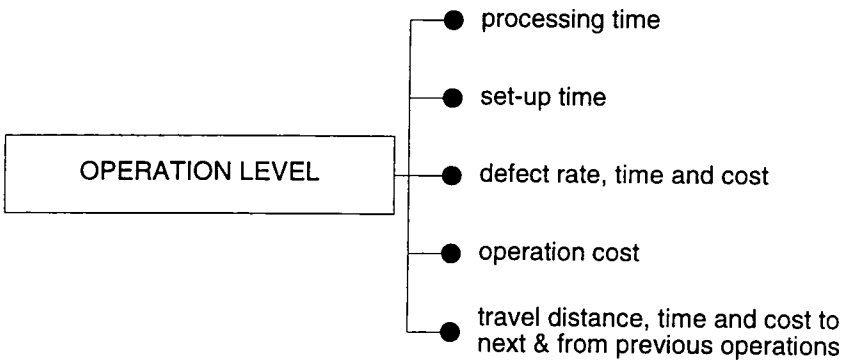
4.5.4 Calculated Capability Scores

A calculated capability score is a high level capability score derived from lower level capability scores.

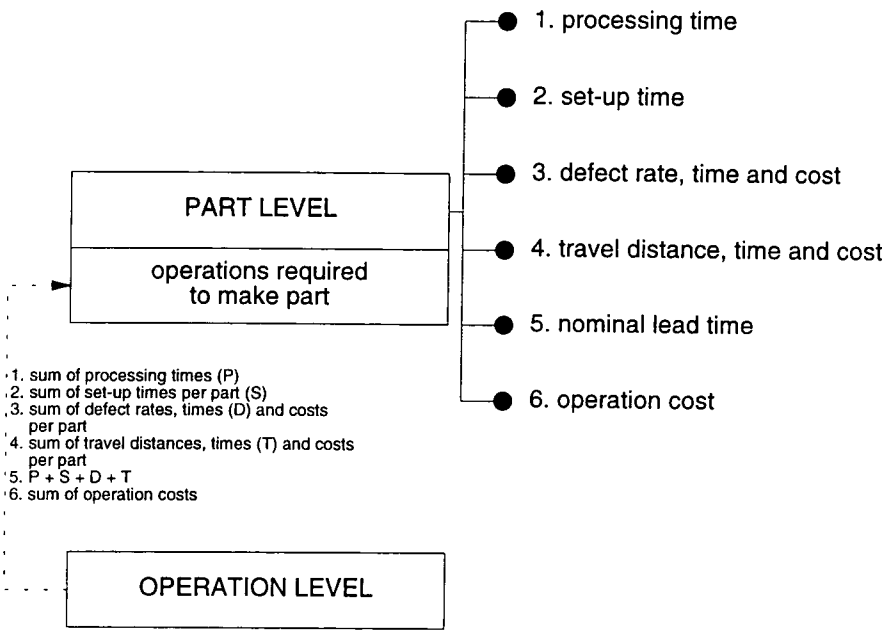
For example, the set-up time of a part is the sum of set-up times per operation of the all the operations required to manufacture the part; the intercell travel distance is the sum of all distances travelled by a cell's corresponding parts where those parts enter or leave the cell. Figure 4.1 shows capability scores for all capability factors at each capability level. The figure includes capability scores that are directly measured (those in the figure that are not numbered) as well as capability scores that are calculated (those in the figure that are numbered).



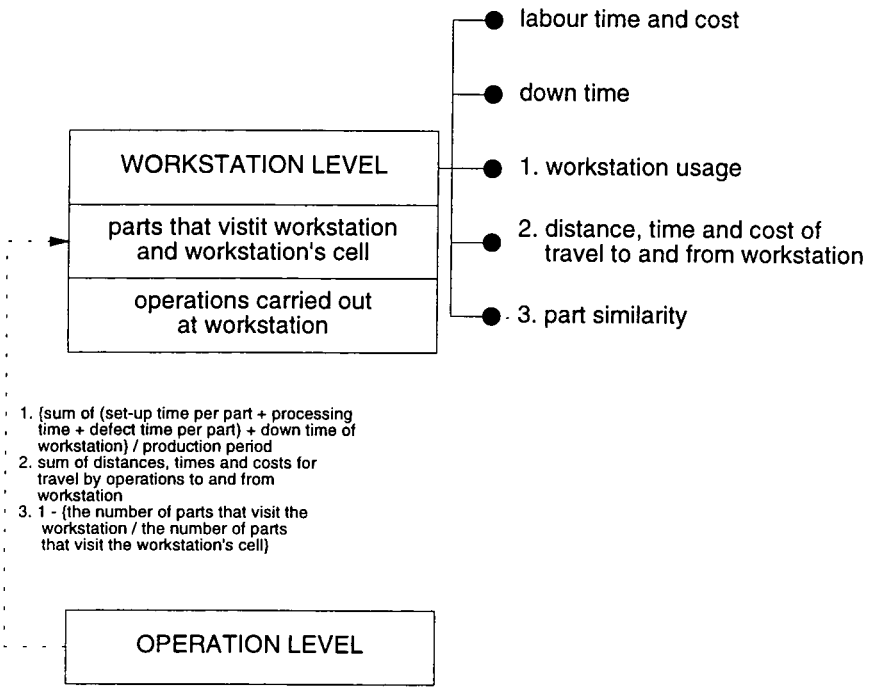
(A) OPERATION LEVEL:



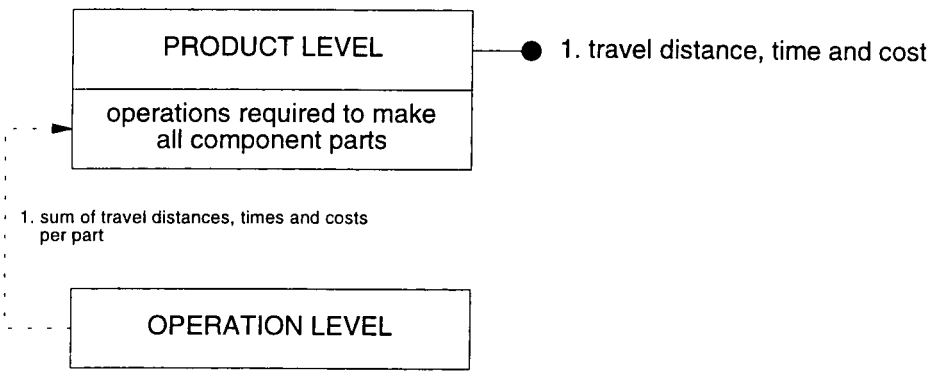
(B) PART LEVEL:



(C) WORKSTATION LEVEL:



(D) PRODUCT LEVEL:



(E) CELL LEVEL:

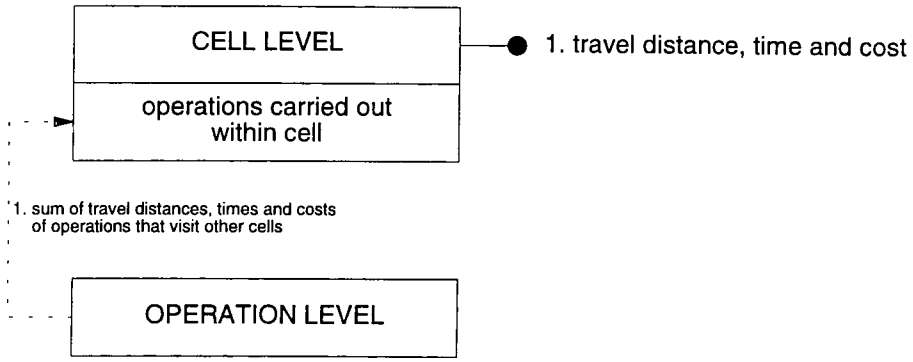


Figure 4.1: Directly measured and calculated capability scores used in Cellect

These capability factors have been selected to reflect the objectives of cellular manufacturing as outlined by Wemmerlov and Heyer (1989):

- to reduce set up times by using part family tooling and sequencing
- to reduce flow times by reducing set up and move times, wait times for moves and using small transfer batches
- to reduce inventories and market response times
- to develop independent sociological units conducive to team work

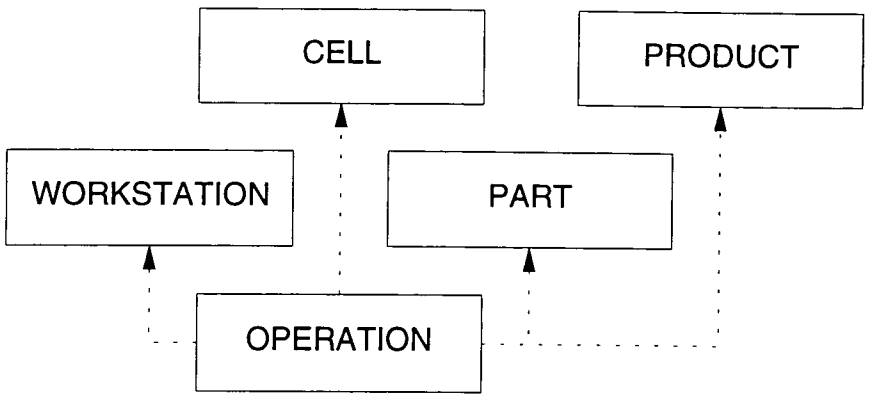


Figure 4.2: Summary diagram for calculated capability scores

Figure 4.2 summarises Figure 4.1 by showing the data flows between capability levels for calculated capability scores. Note how data flows from the lowest possible level (operation level) to ensure accuracy of calculated capability scores at higher levels. Referring back to Figure 4.1, this data flow happens in the following way.

(A) Operation Level Capability Scores

Operation level is the lowest, base level and as such no calculated capability scores can be determined from lower levels, since there are none. Hence, capability scores for all capability factors are determined directly at this level. These are:

- Processing time
- Set-up time
- Defect rate, time and cost
- Operation cost
- Travel distance, time and cost for material handling to the next operation
- Travel distance, time and cost for material handling from the previous operation.

(B) Part Level Capability Scores

Operation level capability factors are also represented at part level. Part level capability scores are calculated from the sums of corresponding capability scores for operations required to make each part. For example, the processing time capability score for a part is the sum of processing time capability scores for all the operations required to make the part and the material handling time capability score for a part is

the sum of operation level material handling time capability scores between the operations required to make the part. There is one additional capability factor at part level, which is *nominal part lead time*. A capability score for this factor is determined by summing the part level's processing time, set-up time, defect time and material handling time for a given part.

(C) Workstation Level Capability Scores

Calculating capability scores for workstation level cannot be done from operation level capability factors, except for material handling capability factors (distance, time and cost). This is because within the context of CA, these factors at workstation level make no sense. For example, if a workstation level processing time capability score was to be defined as the sum of operation level processing time capability scores for all the operations that take place at the workstation, then from the discussion in Section 4.4.2, it would be a requirement that all processing times of a collated group of workstations should, through the process of improvements, become the same. This clearly makes no sense, since different workstations carry out different activities and so there should be no attempt, for example, to force the processing times of a drill to be the same as those of an oven!

The capability factors for which capability scores are directly measured are *labour time*, *labour cost*, and *down time for maintenance*. There are also three capability factors that have capability scores that are calculated. These are *workstation usage*, *material handling performances (distance, time and cost)* and *part similarity*. Part similarity was discussed in Section 4.5.2. Material handling performances are simply sums of distances, times and costs of all operations travelling to and from the workstation in question. Workstation usage is defined as follows:

$$u = \frac{Dj + \sum_{i=1}^{i=I} Pi.ni.(sij + pij + dij)}{T}$$ **Equation 4.3**

where;

u is the usage of workstation j,

D_j is the down time of j per production period,

i is an operation that takes place at j,

I is the total number of operations that take place at j,

P_i is the period demand of the product belonging to i,

n_i is the number-off per product of the part belonging to i,

s_{ij} is the set-up time per part of i at j,

p_{ij} is the processing time of i at j,

d_{ij} is the down time per operation of i at j, and

T is the production period

Note that, although usage is calculated, more accurate data may be obtained through methods such as shop-floor data capture and simulation (see Chapters 1 and 6). The same also applies to nominal part lead time, measured at part level.

(D) Product Level and (E) Cell Level Capability Scores

At product and cell levels, *material handling distance, time and cost* are the only capability factors analysed. At product level, these scores are determined for material handling between the operations required to manufacture the product and at cell level they are determined for all operations entering or leaving the cell (intercell travel).

As the following sections will show, due to the generic nature of CA, other factors may be used to extend the scope of the analysis to reflect different strategic objectives, as discussed in Section 4.2.

4.5.5 Collating

Collating is the act of grouping together capability scores that should be similar.

For example, in Chapter 2, cell configurations were determined to identify groups of dissimilar workstations (cells) producing groups of similar parts (part families). The parts in a part family should be as similar as possible in terms of the processing conditions required to make them. Thus, the capability scores for each capability factor at part level, whether directly measured or calculated (see Figure 4.1) should all be similar within a given cell. The parts belonging to a product, on the other hand, can be wildly different and so a group of similar parts at product level cannot be defined. Hence, the capability scores for part level capability factors can be collated to cell level (each collated group is a part family), but not to product level. Collating for each capability level is defined as follows.

(a) Operation level collated groups. A collated group at operation level is the set of operations taking place at a given workstation. Capability scores for selected factors for operation level CA should be similar for all operations at the given workstation.

(b) Workstation level collated groups. A collated group at workstation level is the set of workstations belonging to a given

cell. Capability scores for selected factors for workstation level CA should be similar for all workstations at the given cell.

(c) Cell level collated groups. A collated group at cell level is all the cells in the manufacturing system. Capability scores for selected factors for cell level CA should be similar for all cells in the manufacturing system.

(d) Part level collated groups. A collated group at part level is all the parts belonging to the part family of a given cell. Capability scores for selected factors for part level CA should be similar for all parts in a given part family.

(e) Product level collated groups. A collated group at product level is all the products in the manufacturing system. Capability scores for selected factors for product level CA should be similar for all products in the manufacturing system.

Capability Level	*Collated To*	*Defined By*
Operation	Workstation	Operations taking place at a given workstation
Workstation	Cell	Workstations belonging to a given cell
Part	Cell	Parts belonging to a cell's part family
Cell	Cell	All cells in the analysis
Product	Product	All products in the analysis

Table 4.1: Collating activities within Cellect

It should be noted that capability scores can be collated to the same capability level at which they are measured. For example, intercell travel cost, which is a cell level capability factor, is compared with all the intercell travel costs in the system. This is because cells should be as independent as possible and it is in this respect that all cells are similar. This is why at cell level only material handling performances are assessed. These are intercell distances, times and costs (Section 4.5.4), and the requirement is that all these should be similar (as low as possible).

The above discussion is summarised in Table 4.1. Figure 4.3 is a summary diagram showing how data is collated within the whole Cellect system.

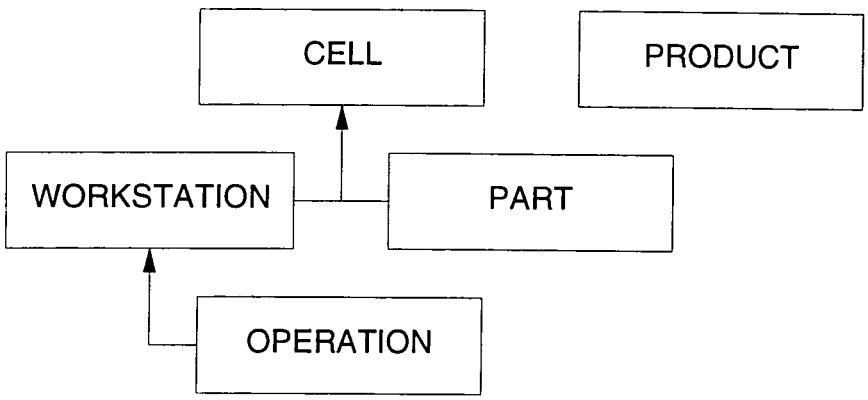


Figure 4.3: Summary diagram showing collating activities

4.5.6 Required Capability Score (s_r), Best Capability Score (s_a) and Worst Capability Score (s_z)

One of the objectives of CA is to improve the capability scores within a collated group to that set as the required capability score for that group. As discussed in Section 4.4.2, a required capability score in most cases need not be defined as the optimum capability score. Rather, it is far better, when carrying out improvements to aim for a

capability score that is attainable. Within a collated group, where all capability scores corresponding to a given capability factor should be similar, the capability score identified as being the best in that group should in most cases be set as the required capability score.

This is represented in Figure 4.4 which shows a set of collated capability scores. In other words, these capability scores corresponding to a given capability factor should all be similar. They are, however, as one might expect, different. As a result, it is possible to identify within this group a best measure and a worst measure.

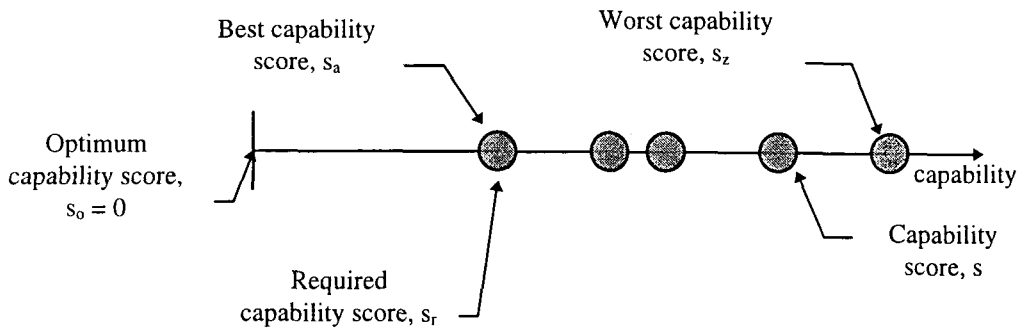


Figure 4.4: Representation of a group of collated capability scores

As stated in Section 4.5.2, capability scores should be defined so that the optimum capability score is zero. Thus, in Figure 4.4, capability scores increase from best to worst, with the best capability score nearest the optimum and the worst furthest away. It can be seen from this figure that, as previously stated, a sensible and attainable required capability score within this group of scores is the best capability score. The objective of an improvement team, therefore, is to lower the other capability scores to the level of this best capability score.

Targets for improvement within a collated group are those that have capability scores that are greater than the required capability score.

In certain circumstances, it may be better to set the required capability score as something other than the best capability score. For example, if all set-up times within a collated group are approximately the same, but unsatisfactory, then by carrying out SMED techniques (Shingo 1985), it is quite reasonable to eventually achieve single-minute changeovers. In this case, the required capability score would be lower than the best capability score (Figure 4.5).

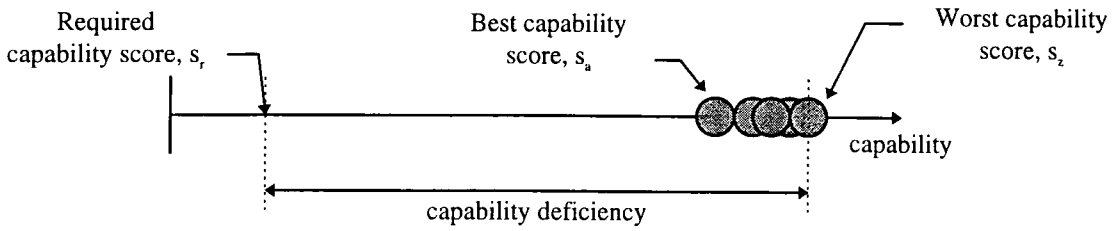


Figure 4.5: Effect of setting the required capability score nearer the optimum value

An example of a situation where a required capability score might be set to be greater than the best capability score is for workstation usage, defined in Section 4.5.4 as the proportion of time the workstation is running. This is a workstation level capability factor that is collated to cell level. Doing so implies that all usages within a cell should be similar. However, as shown in Figure 4.6, it is often the case that a cell may contain a workstation that is hardly ever used. If the usage of this workstation is 10%, then the required capability score for the collated group to which this workstation belongs would also be 10%. Since the aim of CA is to reduce capability scores to that set as the required, it would be necessary to invest resources to force all usages within this group down to 10%. This, of course, is unreasonable and a far more satisfactory usage for any workstation is around 80%, since then workstation capacity is not

exceeded but usage is at a level that ensures the workstation is busy and hence justifying its expenditure. Thus, to avoid unnecessary effort to force usages down to as far as 10%, it is far more reasonable to force the required capability score to be 80% (or thereabouts) and in this way avoid unnecessary allocation of resources. With this adjusted required capability score, the only targets for improvement (that is, those workstations that have a capability deficiency) are those workstations that have a usage that exceeds 80%, whereas those workstations that have capability excess (usage less than the required capability score of 80%) are excluded from the analysis. This situation is shown in Figure 4.6.

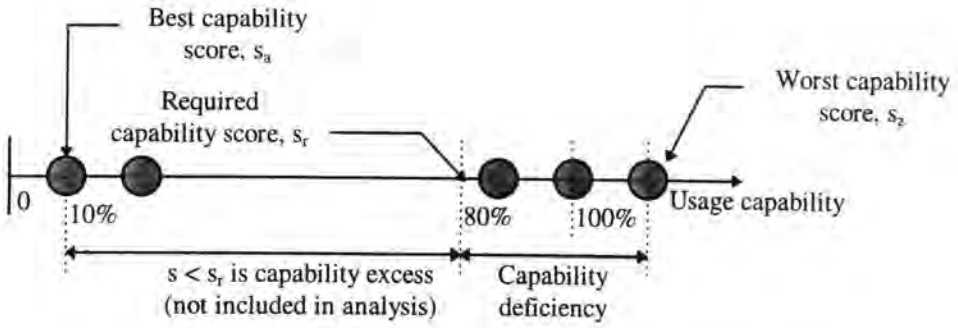


Figure 4.6: Effect of setting required capability score nearer worst capability score

An issue that might warrant further investigation as an alternative to the above is to define workstation usage capability scores as a function of required workstation usage. This would reduce user involvement in setting required capability scores, particularly when it is desirable to have workstation usage above a certain level to justify expenditure. For example, it may be more reasonable to define the usage required capability score for workstations within a given cell as a range between 30% to 80%

and anything outside that as unsatisfactory. Bearing in mind that optimum capability score should be zero, a conditional equation for usage capability scores might be:

$$s = u - 0.8 \quad \text{for } u \geq 0.8$$

$$s = 0.3 - u \quad \text{for } u \leq 0.3$$

$$s = 0 \quad \text{for } 0.3 < u < 0.8 \qquad \textbf{Equation 4.4}$$

where;

s is the capability score for workstation usage capability factor at a given workstation, and

u is workstation usage

Within Cellect, however, user set required capabilities cannot be defined as functions and so the above concept has not been implemented.

In all cases where the required capability score is to be different from the best capability score, in order to have effective continuous improvement, the required capability score must be an improvement over the worst capability score. To reduce the time spent inputting data, the required capability score within Cellect is by default set as the best capability score, although a facility exists to allow the user to change this.

4.5.7 Band Width (m_{rz}), Capability Deficiency (c) and Marginal Capability (c_m)

As discussed so far, within a collated group of capability scores which should be similar, there will be a worst capability score and a required capability score

(defaulted to the best capability score). The difference between the two capability scores is the band width (Figure 4.7):

$$s_{rz} = s_z - s_r \qquad \textbf{Equation 4.5}$$

where;

s_{rz} is the band width of a collated group of capability scores for a given capability factor,

s_z is the worst capability score in that collated group, and

s_r is the required capability score in that collated group

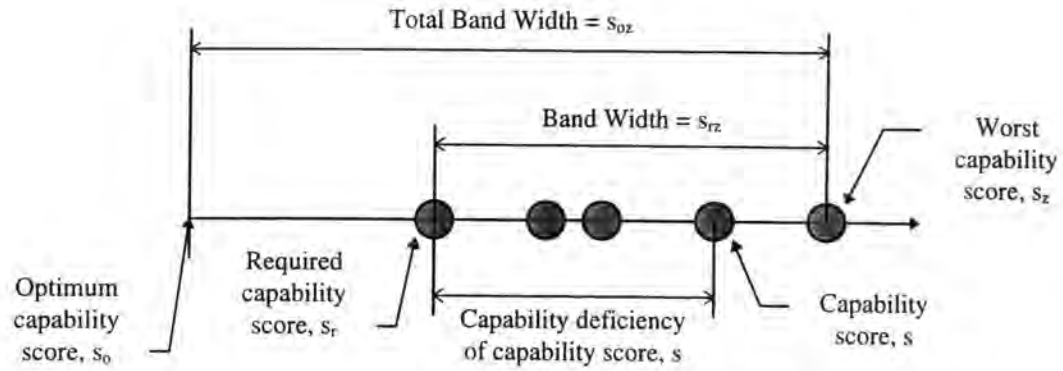


Figure 4.7: Capability Analysis concepts

In order to make planning easier and the control of the manufacturing system simpler, it is desirable to have as small a band width as possible. This is demonstrated by the concept of part families, where in order to make scheduling easier and the control of individual cells more predictable, the parts within a part family must have similar processing requirements. An extreme case of the part family concept is to be found within flow-manufacturing processes wherein workstations are generally dedicated to a single operation. This makes the whole manufacturing system controllable for the mass production of a small number of products. This is because at *operation* level,

where the number of operations at each workstation is one and the required capability score is the best capability score, the band width equals zero. Hence, at *operation level* there are no specific targets for improvement.

If a flow-manufacturing system is considered as a single cell, then workstation level capability factors such as down time and workstation usage are all collated together. Thus, it is easy to find targets for improvement, because for example, for the usage capability factor, the workstation with the worst usage capability score (highest usage) in the whole system (or cell) is the one targeted for improvement. However, usages are mainly made up of operation level capability scores (in particular, set-up time and processing time capability factors), for which band width equals zero so that although improvements are required at *workstation* level, no specific targets can be found at *operation* level. Thus in such organisations, where continuous improvement was born, the only improvement targets are those identified at workstation level, whereas there are no operation level targets and so operation level improvements are carried out in an ongoing, almost opportunistic manner.

For this example, in order to employ CA within a flow manufacturing environment for planned and targeted continuous improvement, required capability scores have to be set for the usage capability factor (as discussed in the previous section). Any workstation that has a usage above that level would be one that needed to be concentrated on. By looking at the operations of these targeted workstations, required capability scores could be set manually for factors such as set-up time, processing time, transfer times and so on. Efforts would then have to be directed to lowering the capability scores of these identified operations to the required levels and in this way

bring down the usage capability scores of the targeted workstations. This enables improvements to be less opportunistic and more focused.

Within a cellular manufacturing system that aims to provide the benefits of flow manufacturing within a medium variety, medium volume environment, the concept of band width is necessary to control and improve the effectiveness of the manufacturing system to handle a greater variety of products. The tools for reducing band width fall into two broad categories:

(a) methods for improving processes to provide consistency for a larger variety of operations, and

(b) methods for improving the design of products so that a smaller variety of parts is able to serve a greater number of products

For a given capability score within a collated group, the difference between it and the required capability score is the capability deficiency (as shown in Figure 4.7):

$$c = s - s_r \qquad \textbf{Equation 4.6}$$

where;

s is a capability score within a collated group,

s_r is the required capability score for that group, and

c is the capability deficiency of s

The worst capability score will have the greatest capability deficiency and it is this capability score that should be targeted when attempting to reduce band width (band width equals capability deficiency of the worst capability score). Given that CA aims to reduce band widths, if a capability score *other than the worst capability score* is

targeted and reduced, then this will have no effect on the band width. In fact, if it is reduced to less than the required capability score, then the band width will become greater, as will the need to reduce the worst capability score! (See Figure 4.8) It is also necessary to reduce all capability scores simultaneously. This is because band width becomes smaller only if the worst capability score is reduced, but still remains greater than the next worst capability score. Thus, targeting only the worst capability score will produce a limited benefit defined by the difference between it and the next worst capability score (this can be deduced from Figure 4.8), although the result of a reduction of the worst capability score will tend to have a knock-on effect on other capability scores in the collated group (for example, reducing the set-up time at a workstation to reduce the set-up capability score of a targeted operation will tend to reduce the set-up times of all the operations that take place at the workstation). Hence the following definition is valid:

Band width is the total amount of lacking capability and is defined as the difference between the required capability score and the worst capability score. Band width reduction is desirable and only takes place by targeting the capability score with the greatest capability deficiency. This will always be the worst capability score.

Further more, when band width has become small enough, it becomes necessary for the user to reduce the required capability score and force it nearer the optimum level. This idea is enforced by CA and discussed in the Section 5.8.

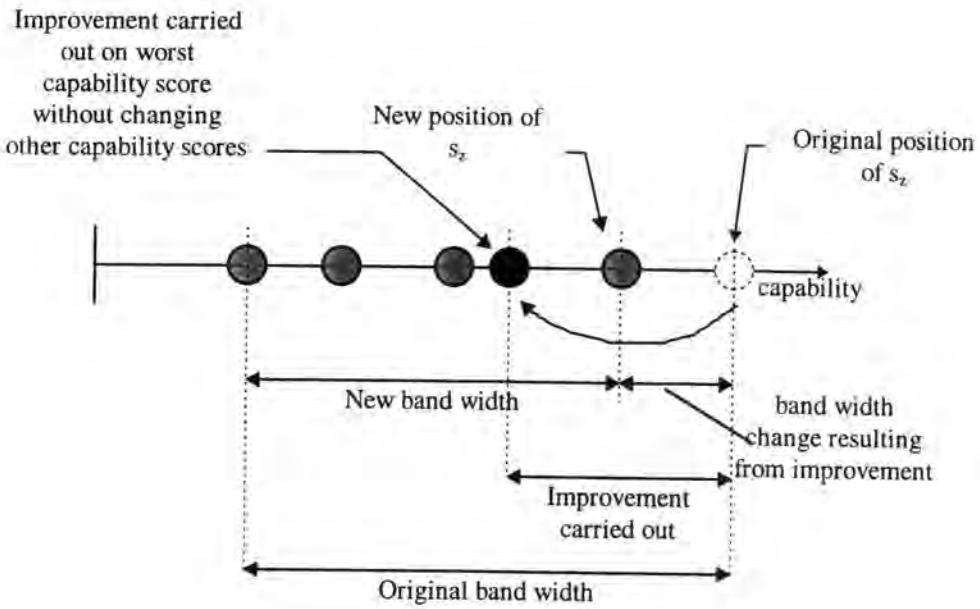


Figure 4.8: Limitation of reducing only the worst capability score

When comparing capability scores corresponding to different capability factors, in order that they can all be assessed in the same way, it is necessary to convert them into marginal capabilities.

A marginal capability is the capability deficiency as a proportion of band width, expressed as a percentage.

$$c_m = \frac{c}{s_{rz}}\% \qquad \textbf{Equation 4.7}$$

where;

c is the capability deficiency of a capability score, s within a collated group,

s_{rz} is the band width of the collated group, and

c_m is the marginal capability of s

At any particular level, given that all factors are equally important and that the necessity to reduce each capability score is the same, then ranking the marginal

capabilities for all capability scores corresponding to all capability factors in descending order will produce a list of improvement targets. This list is a basic recovery schedule.

A recovery schedule is a list of improvement targets ranked in order of priority.

At the top of the recovery schedule will be the worst capability scores and at the bottom will be the best capability scores. The most important aspect of the recovery schedule is the fact that capability scores corresponding to different capability factors can be compared alongside one another. Of course, different capability factors have different levels of importance and the necessity to reduce different capability scores varies according to how much improvement can actually be achieved. In the following subsections, these considerations are taken into account to form a realistic recovery schedule using priority confidence scores (PCS values). A PCS is a marginal capability weighted according to its need for improvement.

4.5.8 Optimum Capability Score (s_o), Total Band Width (s_{oz}) and Improvement Potential (I)

When working with marginal capabilities, particularly when comparing different capability factors with one another, the units of the individual capability scores become irrelevant. Because marginal capabilities are proportional, it is difficult to visualise how much improvement is actually achieved by reducing a band width by, say, 10%. An example of another issue that needs to be considered is that a one minute reduction in the set-up time of operations for a drill which has a worst capability score of 2 minutes is more significant than the same reduction for die changeovers at a press with a worst capability score of 8 hours.

Although CA does not attempt to achieve optimum capability scores, the above issues can be tackled by favouring those actions that make the most progress towards the optimum. To include this within CA, it is necessary to remember that since the objective of CA is band width reduction, the first capability score to be targeted in a collated group is the worst capability score.

The total band width is the amount of lacking capability the worst capability score has from the optimum:

$$s_{oz} = s_z - s_o \qquad \textbf{Equation 4.8}$$

where;

s_{oz} is the total band width of a collated group of capability scores,

s_z is the worst capability score of the collated group, and

s_o is the optimum capability score for the collated group

This is shown in Figure 4.7.

To indicate that more benefit can be achieved by having a required capability score nearer the optimum, it is necessary to define an improvement potential.

For a collated group, the improvement potential is the band width expressed as a proportion of total band width:

$$I = \frac{s_{rz}}{s_{oz}} \qquad \textbf{Equation 4.9}$$

where;

I is the improvement potential for a collated group of capability scores,

s_{rz} is the band width of the collated group, and

s_{oz} is the total band width of the collated group

Thus, if two capability scores are exactly the same but belong to different collated groups and all else is equal, then the capability score with the highest improvement potential should take precedence. Also, if a collated group has a very low improvement potential, then the required capability score should be altered so that it is nearer to the optimum.

To make matters more manageable within Cellect, optimum capability scores for all capability factors are set to zero (see Section 4.5.2):

$$s_o = 0 \qquad \textbf{Equation 4.10}$$

Hence,

$$I = 1 - \frac{sr}{sz} \qquad \textbf{Equation 4.11}$$

where;

s_r is required capability score for a collated group, and

s_z is the worst capability score for the collated group

4.5.9 Factor Weighting (W_F) and Profit Weighting (W_R)

Not all capability scores are the same and as such a weighting should be applied to indicate the importance of a given capability score to the analysis. The weightings used within Cellect fall into two categories. These are factor weightings and profit weightings.

Factor weightings indicate the importance of each capability factor to the analysis

These are constrained as follows:

$$\sum_{F=1}^{F=N_F} W_F = 1 \qquad \textbf{Equation 4.12}$$

where;

F is a capability factor in the analysis,

N_F is the total number of capability factors in the analysis, and

W_F is the factor weighting corresponding to each factor, F

Factor weightings are user-defined and must be chosen to reflect the objectives of the analysis. If, for example, the objective is to streamline material flow through the factory, then travel distance, time and cost factor weightings should be given the highest values.

The next type of weighting is profit weighting which attempts to indicate in financial terms the value of a given capability score. This could be defined in any number of ways, but

within Cellect, profit weighting is the net contributing profit of a unit as a proportion of the total net profit per production period:

$$R = \sum_{P=1}^{P=N_P} RP \qquad \textbf{Equation 4.13}$$

$$R_u = \sum_{P_u=1}^{P_u=N_{Pu}} RP_u \qquad \textbf{Equation 4.14}$$

$$W_R = \frac{Ru}{R} \qquad \textbf{Equation 4.15}$$

where;

R is the total net profit per production period,

R_u is the contributing net profit of a particular unit, u (operation, workstation, cell, part, or product),

W_R is the profit weighting of u

P_u is a product, P associated with u (the product a given operation serves at operation level, a product a given workstation serves at workstation level, a product a given cell serves at cell level, the product a given part belongs to at part level, or the product itself at product level),

N_{Pu} is the total number of products associated with u,

R_{Pu} is the net profit of P_u per production period,

P is any product in the analysis,

N_P is the total number of products in the analysis, and

R_P is the net profit of P per production period

R is constant and the same for all levels, whereas R_u is the same only for all capability factors for a particular unit. For example at operation level, only one product is served by each operation (within Cellect, an operation is attached to only one part) and hence R_u is the net profit of the product that is made up of the part that the operation serves. At cell level, it is not uncommon to see W_R equal to one, because a given cell can serve *all* products within the system so that R_u equals R.

4.5.10 Priority Confidence Scores (S), Transparency And The Target Identification Level

A priority confidence score (PCS) is a marginal capability that has been weighted to reflect the improvement potential, factor importance and financial value of the capability score being analysed.

For each capability score, the PCS is:

$$S = I \{ \alpha W_F + (1 - \alpha) W_R \} c_m \qquad \textbf{Equation 4.16}$$

where;

S is the PCS of a given capability score,

c_m, I, W_F and W_R are the marginal capability, improvement potential, factor weighting and profit weighting respectively, and

α is an emphasis parameter to force the analysis to be factor-oriented or profit-oriented and is constrained as follows:

$$0 \leq \alpha \leq 1 \qquad \textbf{Equation 4.17}$$

In Section 4.5.7, a simple recovery schedule was developed by ranking in descending order the marginal capabilities of various capability scores. By using PCS values instead of marginal capabilities, the recovery schedule is able to provide a much more realistic overview of the improvement requirements within a system.

Since the objective of the recovery schedule is to identify targets for improvement, to make it a useful tool, those targets have to be transparent and the recovery schedule as a whole must not be cluttered with irrelevant targets.

Transparency is the ability to take a target at one level and to identify constituent targets at lower levels.

To this end, it is more useful to have separate recovery schedules for each capability level. Transparency can then be achieved as follows:

1. Select a target from a high level recovery schedule.
2. Identify a group of related capability scores at a lower level.
3. Filter the recovery schedule for that lower level to show targets for this related group only .
4. Select a target from this filtered group and go to step 2.

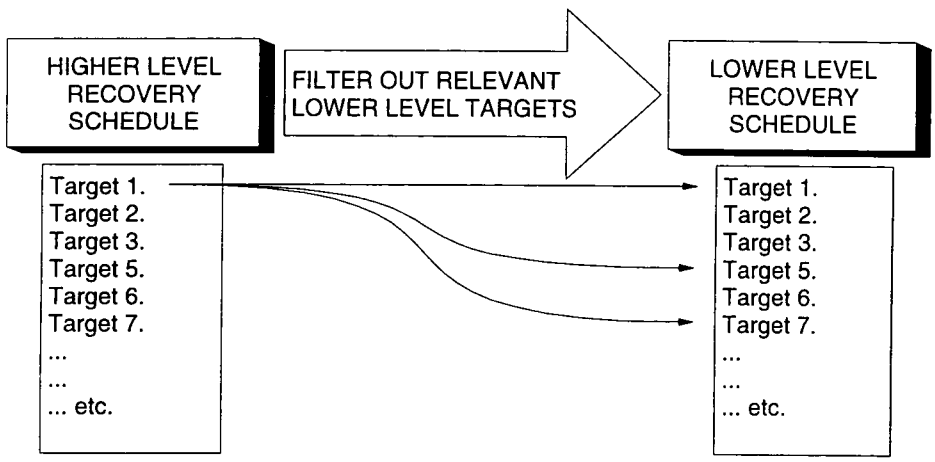


Figure 4.9: Using transparency to filter a lower level recovery schedule

This way, lower level targets can be identified to tackle a specific target at a higher level (Figure 4.9). For example, if at the top of a *cell level recovery schedule* there is a capability score corresponding to intercell travel cost, one group of related capability scores will be the travel costs to and from the workstations that make up the cell. These would be found in the *workstation level recovery schedule*. This workstation recovery schedule could then be filtered to show only those records for which (a) the

capability factor is intercell travel and (b) the workstations are those belonging to the cell being examined. This filtered workstation level recovery schedule would show in descending order of PCS values the following information: workstations, PCS values, marginal capabilities, capability scores, required capabilities and any other data that might aid the user in determining an improvement strategy. A filtering of the operation level recovery schedule for any of these workstation level capability scores would provide yet more detail to the analysis. Thus, in this way specific operations can be identified to improve capability scores for, what in this example, is a cell level target. In the next subsection, which shows how to represent capability factors measured at one level as performances at higher levels, the concept of transparency becomes even more useful.

To remove some of the less critical targets from a recovery schedule, a target identification level (TIL) can be employed.

A target identification level is a marginal capability below which targets are not included in the recovery schedule.

Using a TIL creates a recovery schedule showing only those targets that once tackled will yield significant improvements to the manufacturing system. The TIL should be set near 100% to emphasise the highest marginal capabilities, since as previously explained, an improvement is not productive within the objectives of CA if it does not lower any worst capability scores (100% marginal capabilities).

4.5.11 Representing Capability scores as Performances at Higher Levels

Another important requirement of CA is to be able to take capability factors that are either directly measured or calculated and represent them at a higher level. For example, set-up times are measured at operation level. It makes perfect sense to identify the set-up time of a given operation as being 25 seconds per part, but how is the set-up time of a workstation determined? A couple of possibilities might be the total or average set-up time of all the operations that take place at the workstation. This, however, does not take into account the fact that some workstations have fewer operations taking place at them, thus implying that to reduce workstation set-up time it is adequate to simply eliminate operations. Although this is true to a certain extent, a profitable manufacturing system is not one that does not have any operations taking place within it!

The answer to the above question is that it is not the set-up time of a workstation that is determined, but its set-up performance. PCS values identify the performances of operations such that one capability score is better than another because it has a lower PCS, irrespective of which workstation it belongs to. By averaging the PCS values for a given capability factor of all operations that take place at a given workstation, the performance of that workstation can be determined in terms of the PCS values of its corresponding operations. This is true for all capability levels such that:

$$S' = \frac{\sum_{T=1}^{T=N_T} S_T}{N_T} \qquad \textbf{Equation 4.18}$$

where;

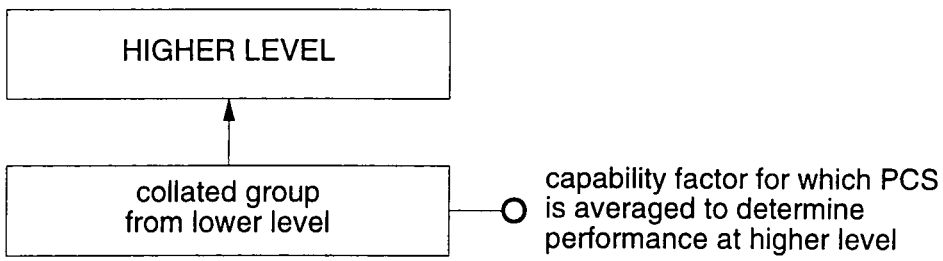
S' is a higher level performance of a given unit, u for a capability factor, F that is determined (directly or calculated) at a lower level, L,

T is a target in the level L recovery schedule, for which capability factor is F and unit is u,

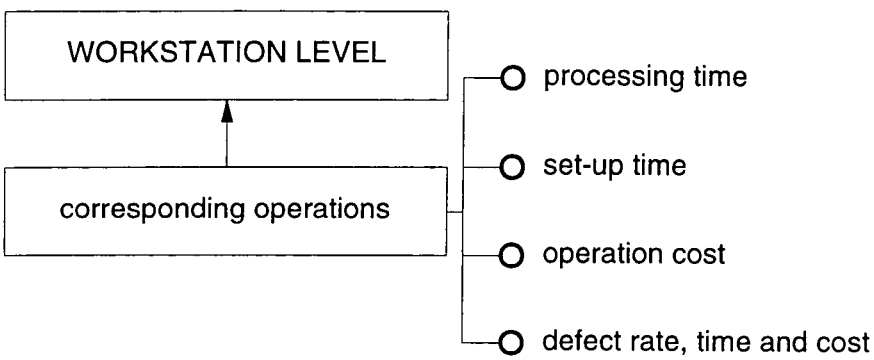
N_T is the total number of targets in the in the level L recovery schedule, for which capability factor is F and unit is u, and

S_T is the PCS value corresponding to T

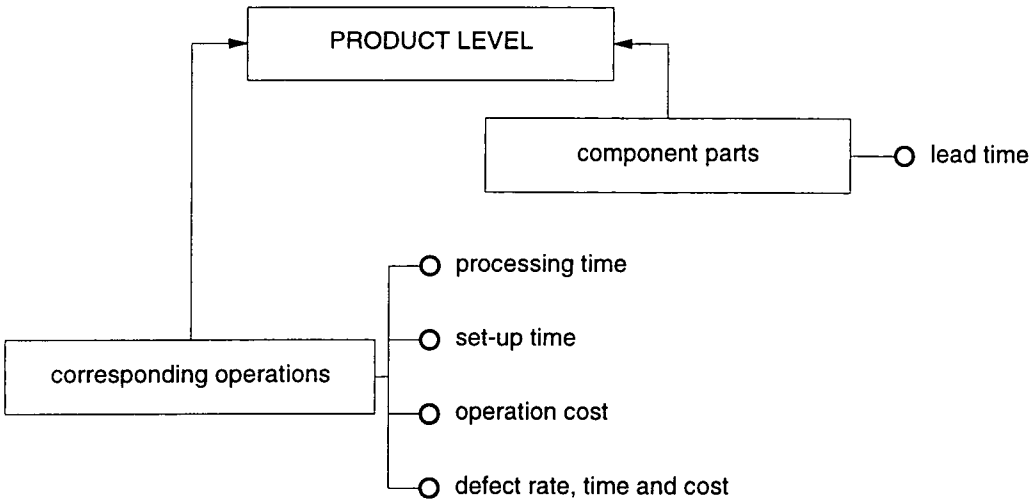
KEY:



WORKSTATION LEVEL:



PRODUCT LEVEL:



CELL LEVEL:

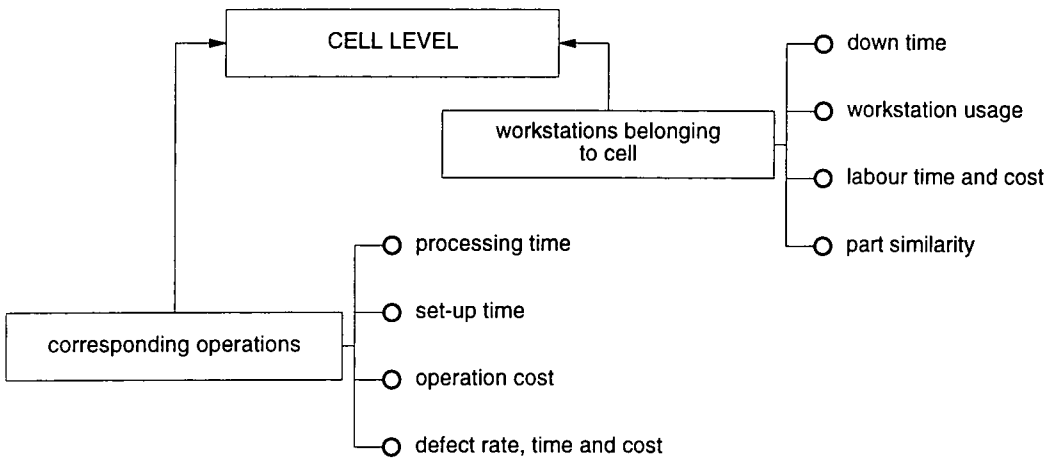


Figure 4.10: Lower level PCS values represented at higher levels

Thus, the set-up performance of a given workstation is the average set-up PCS value of all the operations that take place at the workstation. Figure 4.10 shows the performances that are determined within Cellect at each capability level for capability factors that are not directly measured or calculated at that level. Figure 4.11 is a summary diagram showing how data flows within Cellect to determine indirectly measured performances.

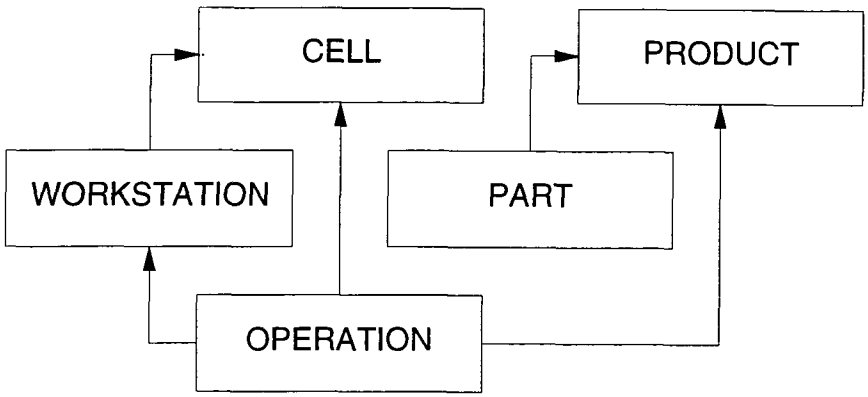


Figure 4.11: Summary diagram showing data flows for performances

Note that no operation level performances have been determined at part level. This is because, as described in Section 4.5.4, all operation level capability factors are represented at part level as capability factors for which capability scores have been calculated. Also, to make the analysis more comprehensible, performances are only determined from capability scores that are wholly responsible for corresponding performances at higher levels. For example, if a part *p* is made by two cells, *cell a* and *cell b*, then both *cell a* and *cell b* affect the nominal lead time of *p*. Thus, the nominal lead time performance of *cell a* is affected by the inadequate capabilities of *cell b* and vice-versa. This makes building the recovery schedule difficult, since then the cell level nominal lead time performances for both *cell a* and *cell b* would have to be represented as a *single* target which is not valid within the current software structure of Cellect (see Chapter 5). Besides, all the elements that contribute to a nominal part lead time are represented as cell level performances of operation level capability factors (see Section 4.5.4). Other possible performances that have been ignored in the current version of Cellect for the same reason as above are product level workstation

usage performance, part level workstation usage performance and workstation level nominal part lead time performance.

4.6 Capability Analysis Example

The following example is intended as an aid to understanding the concepts described in the previous sections. It shows how to determine the PCS value of a capability score for a given capability factor and then goes on to demonstrate how a recovery schedule is built up for four collated groups of capability scores, each of which is determined at one of two different capability levels. This example is also intended to act as an insight into how CA can be implemented manually.

4.6.1 Capability Levels and Capability Factors

Workstation level and operation level are the two capability levels chosen for this example. The capability factors for which capability scores can be determined (either directly or calculated) are as follows.

Operation Level	Workstation Level
Set-up time	Down time
Operation cost	Usage

4.6.2 Collating and Capability Scores

Cell design has taken place from which can be identified *groups* of workstations (cells) and *groups* of parts (part families). These are the groups into which capability scores are collated and in each collated group, performances should be similar. In this

example, capability scores for operation level capability factors corresponding to a given workstation should be similar and capability scores for workstation level capability factors corresponding to a given cell should be similar. Hence operation level capability scores are collated to workstation level and workstation level capability scores are collated to cell level (see Table 4.1).

A set of capability scores for a collated group of set-up times per part (this capability factor is measured at operation level) for all operations at a singe workstation, W1, are as follows:

Operation	Capability Score (set-up time per part at W1)
OpA	25
OpB	15
OpC	20
OpD	35

4.6.3 Band Width Diagram

The above capability scores can now be represented within a band width diagram, as shown in Figure 4.12.

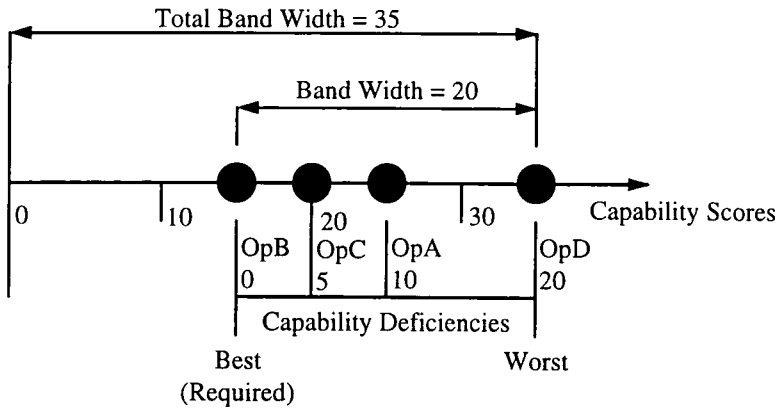


Figure 4.12: Band width diagram for example data

For the whole collated group, the following information can be identified from Figure 4.12: Band width diagram for example data:

Worst capability score, s_z: 35

Best capability score, s_a: 15

Required capability score, $s_r = s_a$: 15

Band width, $s_{rz} = s_z - s_r$: 20

Optimum capability score, s_o: 0

Total band width, $s_{oz} = s_z - s_o$: 35

Improvement potential, $I = \frac{Srz}{Soz}$ or $I = 1 - \frac{Sr}{Sz}$: 0.57

4.6.4 Determining PCS For A Capability Score

OpA will be looked at in more detail in order to demonstrate how a PCS is determined. The first stage is to determine a marginal capability corresponding to a capability score, s_{OpA} for OpA:

Capability Deficiency, $c = s_{OpA} - s_r$: 10

Marginal capability, $cm = \frac{c}{srz}\%$: 50 %

The next stage is to combine the marginal capability with the improvement potential, factor weighting, profit weighting and emphasis parameter. If OpA serves a part belonging to a product that brings in £1000 net profit per production period and the total net profit of all the products in the system is £4000 per production period, then the profit weighting W_R of OpA is 0.25. Let it also be assumed that the user has considered set-up time capability factor important enough to be allocated a factor weighting W_F of 0.8. The emphasis parameter is set at 0.5. PCS can now be determined as follows:

$$S = I\{\alpha W_F + (1 - \alpha)W_R\}c_m$$

$$= 0.57 * \{(0.5 * 0.8) + (0.5 * 0.25)\} * 50$$

$$= 15$$

The PCS values for each of the other capability scores at each level are calculated in the same way. Examples PCS values are as follows:

Operation Level	Set-up PCS	Operation cost PCS
OpA	15	12
OpB	0	15
OpC	7.5	20
OpD	30	0
Workstation Level	Down Time PCS	Usage PCS
W1	0	30
W2	12	0

W3	7	15

4.6.5 Initial Recovery Schedules

The PCS values corresponding to each capability factor are without units and so can be compared alongside one another. The result of ranking these values are recovery schedules for each capability level :

Operation Level Recovery Schedule

Operation	*Factor*	*S*	*s*	s_r	c_m
OpD	Set-up	30	35	15	100
OpC	Opn Cost	20	20	10	100
OpB	Opn Cost	15	17	10	70
OpA	Set-up	15	25	15	50

Workstation Level Recovery Schedule

W'station	*Factor*	*S*	*s*	s_r	c_m
W1	Usage	30	1.20	0.90	100
W3	Usage	15	1.09	0.90	75
W2	Down time	12	25	5	100
W3	Down time	7	15	5	50

Note that the full recovery schedule for each level is not shown. This is because a target identification level filters out marginal capabilities that are less than 50%. Note

also that in the operation level recovery schedule, the third and fourth targets have the same PCS values and are therefore ranked according to marginal capabilities. The user may wish to choose an alternative method for resolving such conflicts. For example, ranking according to improvement potential.

4.6.6 Representing PCS Values At Higher Levels

For each unit at a higher level (in this example, for each workstation), averaging related PCS values for lower level capability factors (in this example, averaging set-up time and operation cost PCS values of operations that take place at each of the workstations) allows lower level capabilities to be represented at higher capability levels.

In this example, OpA, OpB, OpC and OpD are carried out at W1 and so averaging out set-up time and operation cost PCS values of these operations will allow these capability factors to be represented at workstation level. To determine these averages, the data used is that from the recovery schedule:

W1 set-up time performance = (30 + 15) / 2 = 22.5

W1 operation cost performance = (20 + 15) / 2 = 17.5

These PCS values can now be added to the workstation level recovery schedule:

Workstation Level Recovery Schedule Including W1 Lower Level Performances

W'station	*Factor*	*S*	*s*	s_r	c_m
W1	Usage	30	1.20	0.90	100
W1	Set-up	22.5	-	-	-

W1	Opn Cost	17.5	-	-	-
W3	Usage	15	1.09	0.90	75
W2	Down time	12	25	5	100
W3	Down time	7	15	5	50

4.6.7 Transparency

Transparency is used to identify targets at a lower level that correspond to a target at a higher level. In this example, where the workstation recovery schedule includes average PCS values, it is necessary to identify the targets corresponding to those values. For example, in order to carry out transparency to identify the causes of the second most critical target in the workstation recovery schedule, it is necessary to filter the operation level recovery schedule to show only those targets for which (a) the capability factor is set-up time and (b) correspond to operations taking place at W1:

Operation Level Recovery Schedule Filtered To Show Only Those Targets Corresponding The Second Most Critical Target In The Workstation Level Recovery Schedule

Operation	*Factor*	*S*	*s*	s_r	c_m
OpD	Set-up	30	35	15	100
OpA	Set-up	15	25	15	50

From this it can be seen that in order to eliminate the second target in the workstation level recovery schedule, it is necessary to concentrate efforts to reduce the set-up times per part of OpD by 20 seconds and the set-up times per part of OpA by 10 seconds. Note also that reducing the set-up time of OpD by more than 10 seconds without reducing the set-up time of OpA will not provide any more benefit, since OpA will then have the worst capability score and will therefore become the focus of attention.

4.7 Summary

An important element in the design and management of cellular manufacturing systems is the ability to determine capabilities. This Chapter described the methodology of Capability Analysis (CA) to form a recovery schedule of targets for improvement ranked in order of priority. The main aspect of CA is the ability to compare different factors alongside one-another. This is done by determining priority confidence scores (PCS values) which are capability scores (measures of performance) represented as proportions of required capability scores adjusted to take into account improvement potential, factor importance and financial considerations. Recovery schedules are produced by ranking the PCS values for each capability level (cell, product, part, workstation and operation). Targets at a higher level can be broken down into more detailed targets at a lower level using the concept of transparency.

The main objectives of CA are:

- Compare unlike capability factors in the same recovery schedule
- Filter out the lower level causes of higher level targets

- Encourage improvements that provide the most benefit for a given resource allocation.

In the next Chapter, all the concepts so far discussed are implemented within Cellect. This is a software tool for the design, management and continuous improvement of cellular manufacturing systems, which amongst other activities, is able to carry out CA for all the factors and levels shown in Figure 4.1.

5. Cellect Software

5.1 Implementation

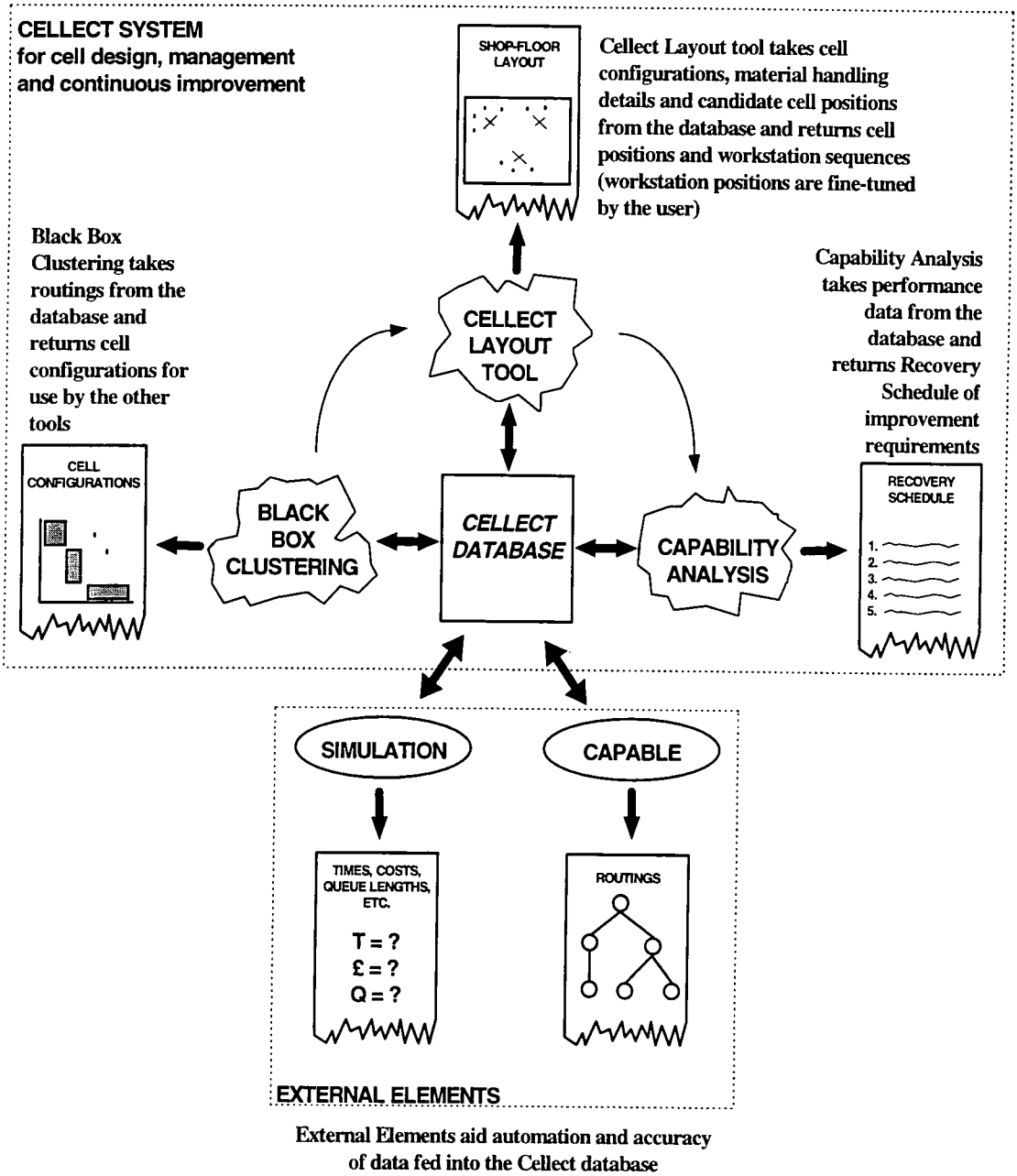


Figure 5.1: Representation of the Cellect system

To demonstrate the concepts discussed in the previous chapters, a software system has been developed that incorporates a database, the three Cellect tools (Black Box Clustering, Cellect Layout Tools and Capability Analysis) and a basic user interface.

Figure 1.3 (which was discussed in Chapter 1 and is reproduced above for the reader's convenience as Figure 5.1) shows the three Cellect tools connected to a common database. There are two main reasons for using a common database. These are:

(a) To prevent duplication of the data inputted thus saving time and making data entry less error prone, and

(b) To facilitate the sharing of data between the Cellect tools and to make each tool independent.

As can be seen from Figure 5.1, the database acts a central storage facility to which each of the tools can pass data and from which they can retrieve information. In order that the Cellect tools can be independent, it is thus necessary to have no data transfer between them and so instead any data movement is done only via the database. Before continuing with a description of Cellect, it is first necessary to explain some database terms (in this work, the term database refers to the modern definition of a *relational* database, which is a collection of data tables, as opposed to the traditional database which is a single data table).

- A database is a set of tables, with each table containing a category of information (workstations, parts, material handling and so on).
- Each table is made up of fields, with each field corresponding to an item of information (for example, in the workstations table,

'workstation name', 'cell that the workstation belongs to' and 'included in analysis' are fields).

- Each table is a collection of records, where each record contains entries in all the fields for a specific item (for example in the workstations table, a Cincinnati Sabre 750 is an item; the workstation name, cell that it belongs to, down time, labour cost, etc. together make up the record for this item).

The database is populated either by user input or by a program. Any data that has been entered automatically can be modified by the user. More about relational databases can be found in Hentzen (1995).

The programming of Cellect required the use of a relational database application developer. Microsoft Visual FoxPro version 3 was chosen because it has all the tools necessary to build an efficient relational database, has a suitable programming language and has object-oriented form-building tools.

The development of the Cellect application and the concepts described in the previous chapters are demonstrated using example data based on that required to manufacture two electro-mechanical products. This has been obtained from a local manufacturer. Where data could not be obtained, this was guessed.

5.2 Required Data

When setting up Cellect, the first task is to provide inputs into those fields that require data. The data required from the user is shown in Table 5.1(the *Table* column refers to the database table names; *Inputted Data* are the field names; a tick in *B*, *L* or *C*

indicate that the inputted data is used by Black Box Clustering (B), Cellect Layout Tools (L) or Capability Analysis (C); and *Notes* provides additional information to aid understanding)

Table	Inputted Data	B	L	C	Notes
Products	Product name	✓	✓	✓	
	Demand		✓	✓	per production period
	Net profit			✓	
Workstations	Workstation name	✓	✓	✓	this also includes exit/entry points and other locations where operations takes place
	Cell that workstation belongs to		✓	✓	default cell names are allocated to workstations automatically although the user may in some circumstances want to identify new cells by default, any workstation not belonging to a cell is given a cell name NONE - the user may want to force this cell name to a workstation
	Included in analysis?	✓	✓		by selecting whether or not a workstation is included in the analysis, the user can store data for machines not in the factory (eg state-of-art machines) this field can also be used to temporarily hide workstations from either Black Box Clustering or the Cellect Layout Tools
	x co-ordinate		✓	✓	either inputted to fix the position of a workstation that cannot be moved or fine-tuned after running the Cellect Layout Tools
	y co-ordinate		✓	✓	see above
	Labour grade required to operate workstation			✓	
	Labour time at workstation			✓	
	Down time of workstation			✓	
Parts	Product name	✓	✓	✓	to which part belongs
	Part name			✓	
	Part code	✓	✓	✓	unique identifier
	Part family that part belongs to			✓	as with cell names in the workstations table, the user may force a part into a pre-defined part family (the part will have to be excluded from the analysis)
	Number off		✓	✓	per parent product
	Batch size		✓	✓	

	Included in analysis	✓	✓		as with similar field in workstations table, the user can choose whether or not to include parts not yet designed
Operations	Part code	✓	✓	✓	each operation serves only one part
	Feature name			✓	identifies what process takes place
	Sequence			✓	each part requires a sequential set of operations - this field identifies where in that sequence the operation is (see Figure 5.2)
	Visiting point	✓	✓	✓	workstation or entry/exit point at which the operation takes place
	Previous point		✓	✓	
	Next point		✓	✓	
	Material handling method from previous point		✓	✓	
	Material handling method to next point		✓	✓	
	Transfer batch size from previous point		✓	✓	
	Transfer batch size to next point		✓	✓	
	Operation cost			✓	
	Processing time			✓	
	Set-up time per batch			✓	
	Defects			✓	per million parts
Material handling	Material handling method		✓	✓	
	Speed		✓	✓	metres per minute
	Cost		✓	✓	per minute
Labour	Labour grade			✓	
	Rate corresponding to grade			✓	
Candidate cell centres	x co-ordinate		✓		this table can be generated automatically if candidate cell centres are a grid of points (user then deletes unwanted cell centres)
	y co-ordinate		✓		
Capability factors	Factor name			✓	
	fc_oplev			✓	this is an index corresponding to how priority confidence scores for a given capability factor are calculated at cell level (1 - PCS determined at this level, 2 - marginal capabilities averaged to determine performances of workstation level capability factors, 3 - performances determined for operation level capability factors, 0 - capability factor not measured at this level)

	fc_prlev			✓	index for product level (1, 3 and 0 as above, 2 - performances determined for part level capability factors)
	fc_wslev			✓	index for workstation level (1 and 0 as above, 2 - equivalent to 3 above)
	fc_ptlev			✓	index for part level (1 and 0 as above)
	fc_oplev			✓	index for operation level (1 and 0 as above)
	order of data retrieval			✓	index corresponding to order in which information is retrieved to be stored in Capability Analysis arrays - please refer to listing in Appendix)
	Factor weighting			✓	
Required measures	Unit			✓	cell, workstation, product, part or operation
	Capability factor			✓	
	Required measure			✓	only add records to this table for required measures that are not best measures
Miscellaneous	Emphasis parameter			✓	
	Target identification level			✓	
	Production period				hours
Features	Feature name				this table is used only for error checking - when entering a record in the operations table, the workstation has to be capable of producing the operation's feature
	Workstation name				

Table 5.1: Data Cellect requires from user

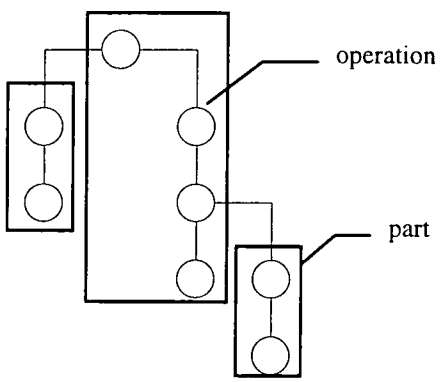


Figure 5.2: Each part should be defined as a sequence of operations

The procedure for inputting the data in Table 5.1 into the database is described below and is done through forms such as the one shown in Figure 5.3.

- Identify the products in the analysis and input relevant data.
- From routings, identify the workstations required. Into the workstations table input labour grade, operator time and down time. The 'included in analysis field' is Boolean and by default is set to true. Workstations can be added to the database but excluded from the analysis by setting included in analysis to false. If the workstation needs to be forced into a user-defined cell then a cell name should be provided and the workstation should be excluded from analysis prior to running Black Box Clustering. Set co-ordinates for workstations that should not be moved. Identify locations of other points parts visit and set the cell names for these to NONE (if positions are known, these should be provided and the points should be excluded from the analysis by setting 'included in analysis' to false; if positions are not known, then exclude from Black Box Clustering only). Ensure the labour table contains the labour grade entered. If it does not, an error occurs and the user is prompted to check the input.
- From product bill of materials and the routings determine inputs for the parts table. Note that the part code is a unique identifier that links a given part with a given product; if two products have the same part, then that part will have two part codes: one for each product. Also, as with workstations, a part may be excluded from the analysis by setting the 'included in analysis field' to false.

- Use the routings and manufacturing data to fill in the operations table. Cellect checks to see if the workstation at which the operation takes place is capable of producing the feature. This is done by determining whether or not the operation's workstation and feature are in the same record in the Features table. If an error occurs, make sure the feature can be produced by the workstation and if it can then add the feature/workstation combination to the features table. Errors also occur if part codes, workstation names and material handling methods are not in their corresponding tables.

- If the Cellect Layout Tools are to be used then ensure candidate cell centre co-ordinates are entered in the appropriate table. A program has been written to set candidate cell centre co-ordinates as a grid, or alternatively, the user can set these manually.

- In the capability factors table, the user need only be concerned with entering factor weights against corresponding factor names. The fc_cllev, fc_prlev, fc_wslev, fc_ptlev, fc_oplev and order of data retrieval fields are there to aid the developer (or super-user) when adding extra capability factors or extra capability levels. Capability Analysis is carried out by pulling data out of the database, storing it in arrays and then using generic methods to produce the Recovery Schedule. The program knows how to deal with each capability factor at a given capability level from the index fields and knows

when to pull out data from the value in the order of data retrieval field.

- If a required measure of a given capability factor for a given collated group is to be a value that is different to the best measure of that group then this can be specified.
- Finally, α, TIL and the production period need to be entered.

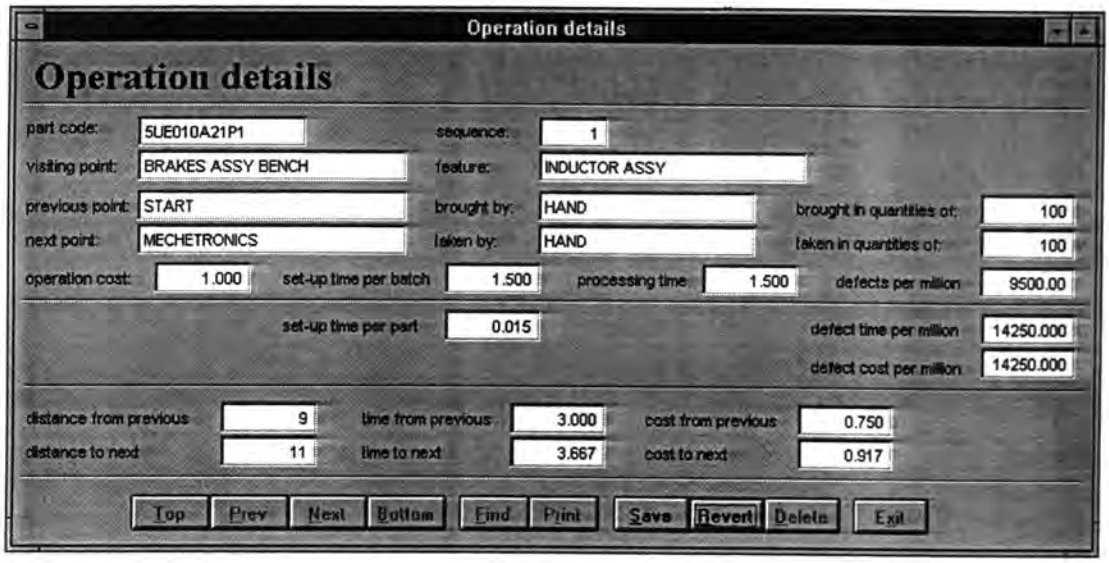


Figure 5.3: Example data entry form

The example data (see Section 5.1) was inputted into the database. This was a lengthy and painstaking procedure, made easier by the fact that data entry was done via customised data entry forms such as the one shown in Figure 5.3. These were modified versions of those produced automatically by the Fox Pro 'Form Wizard' and provided a common user interface that made accessing and updating data easy. The process of entering Cellect data manually highlighted the need for effective data management and also highlighted the benefits of tools such as CAPABLE, discrete

event simulation and shop floor data capture (see Chapter 1) to automate the accurate input of data.

5.3 Running the Cellect Tools

The main objective of this section is to identify the data required to run each Cellect tool and to identify the data that each tool supplies the database. This information is needed so that the user can decide what data it is necessary to update prior to running any of the tools. This section will also present the results that each tool produces from running the example data used in this project.

It is intended that when using Cellect to carry out cell design from scratch, Black Box Clustering, Cellect Layout Tools and Capability Analysis should be run in that order. This can be seen from the stages of cell design in Chapter 1, Figure 5.1 and the data required by each tool shown in Table 5.1. The following subsections are therefore presented in such a way so as to show how Cellect can be used to determine cell configurations, use the configurations to aid the user in positioning of workstations on the shop floor and identify improvements required based on the activities occurring on the redesigned shop floor.

5.3.1 Black Box Clustering

Table	Required Data	Notes
Workstations	Workstation name	
	Included in analysis	having this set to false for a workstation may exclude parts from the analysis (see below)
Parts	Part code	
	Included in analysis	some parts may be excluded from the analysis even if included in analysis is set to true - this happens if all operations for this part are carried out by workstations excluded from the analysis (included in analysis set to FALSE in the workstations table)
Operations	Part code	
	Visiting point	

Table 5.2: Cellect data required to run Black Box Clustering

Table	Data Inputted	Notes
Parts	Part family that part belongs to	the algorithm assigns cell names such as CELL_1, CELL_2, etc. parts excluded from the analysis are assigned to the NONE cell
	Matrix number	corresponds to numbers in Figures 5.4a and b
Workstations	Cell that workstation belongs to	as with family names
	Matrix number	as for parts

Table 5.3: Data supplied to the database by black box clustering

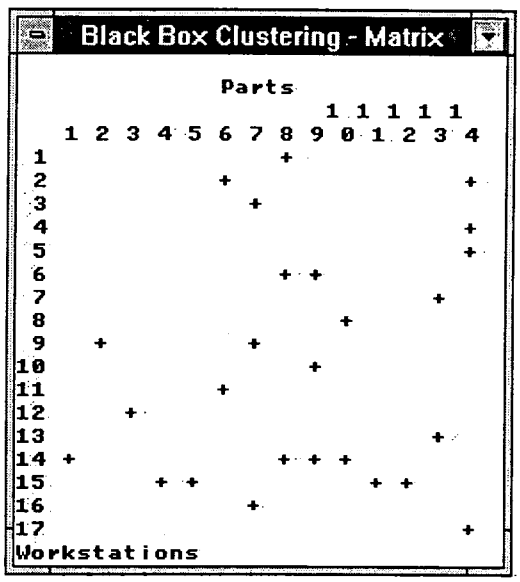


Figure 5.4: Initial workstation-part matrix

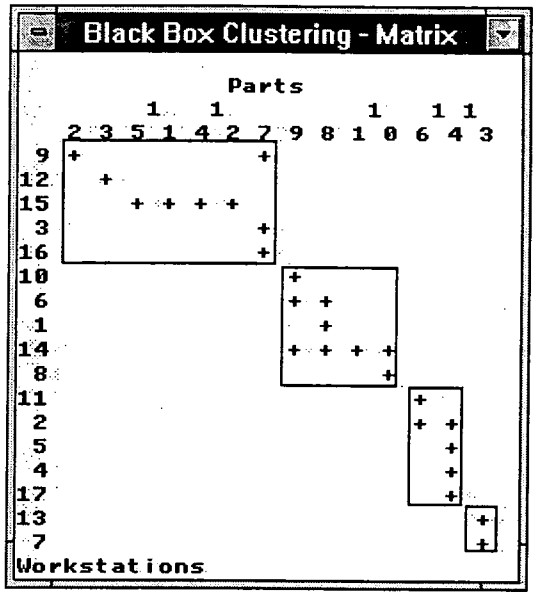


Figure 5.5: BDF produced after running BBC

Black Box Clustering takes the data in Table 5.2 and converts it into a workstation-part incidence matrix. The data used in this project produced the matrix shown in Figure 5.4. The algorithm then proceeds as described in Section 2.3 to produce a block-diagonal form (BDF) such as the one shown in Figure 5.5. BDF quality statistics (Section 2.3.3) are displayed and the user is prompted to select whether or not the database should be updated based on this information (for example, if BDF quality was poor, the user may want to try another routing option). If the user selects to update the database, the information supplied to the database is shown in Table 5.3. In this example, the statistics are as follows:

BDF Quality (Cellect Output)

Number of Cells : 4

Largest Cell: 5 Workstations

Smallest Cell: 2 Workstations

The matrix contains a total of 26 elements of which 0 (0.00%) are exceptions

```
BDF Density:                    38.81%
```

The above statistics and figures show that running BBC on the example data produced four totally independent cells. However, the average density of 39% is low and this can be seen in Figure 5.5 in which there are very few crosses in the first two blocks. This indicates that utilisation of the workstations in these cells *may* be low (Chandrasekharan and Rajagopalan 1986b). However, Cellect allows the user to view and alter cell configurations manually using forms such as the ones shown in Figure 5.6 and Figure 5.7. If so desired, the number of cells in the BDF could be increased by configuring cells as shown in Figure 5.8. In this figure, the resulting blocks are much more dense, but less independent as can be seen from the number of exceptional elements that now exist. It is felt that cell independence is more important than cell density and so the configurations produced automatically by BBC (Figure 5.5) is used for the rest of the analysis.

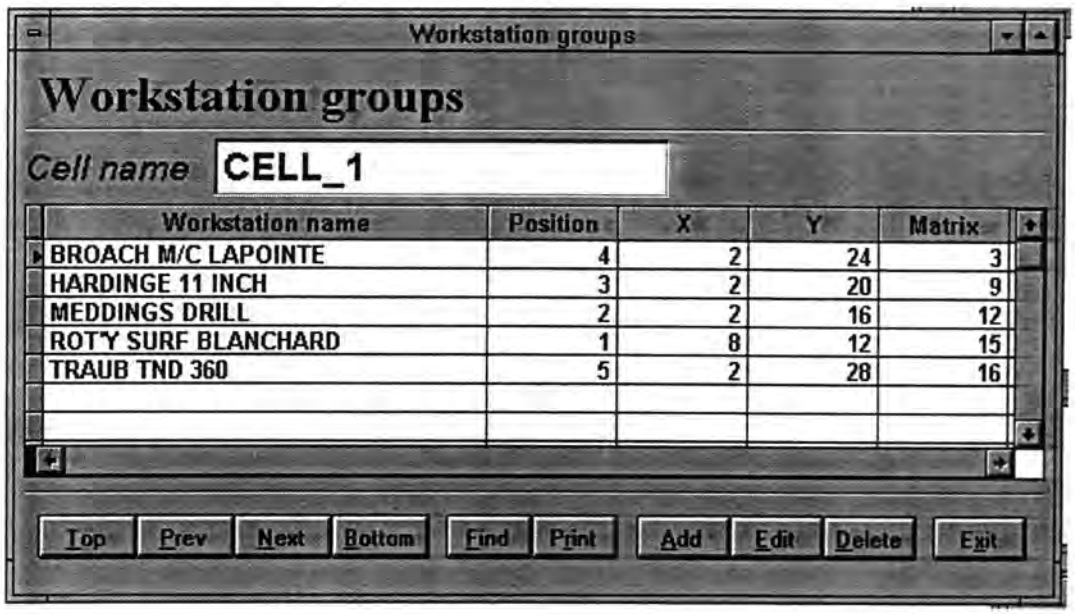


Figure 5.6: Workstation groups form

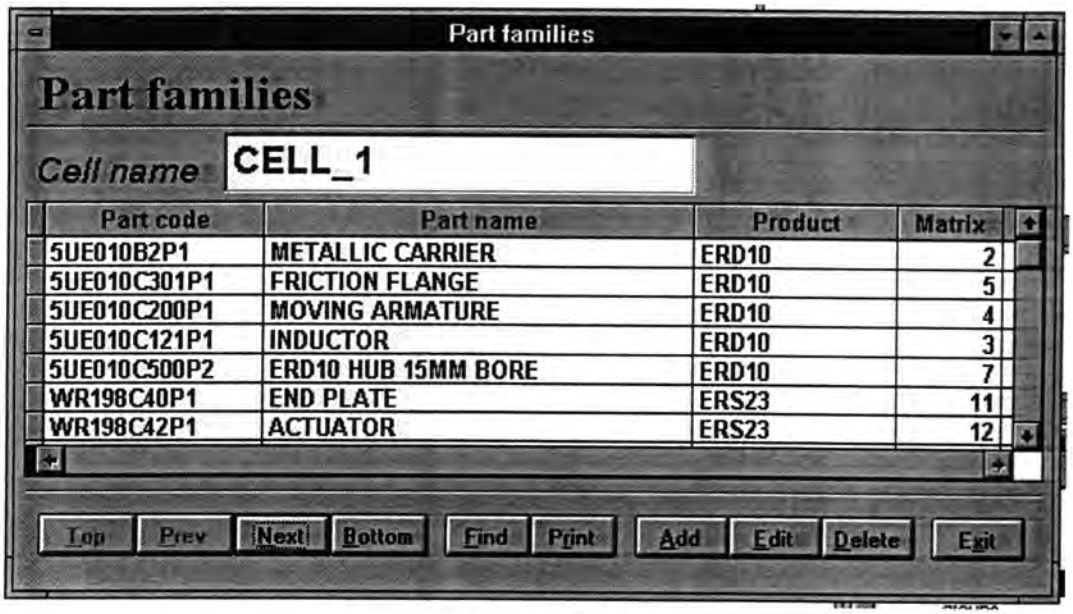


Figure 5.7: Part families form

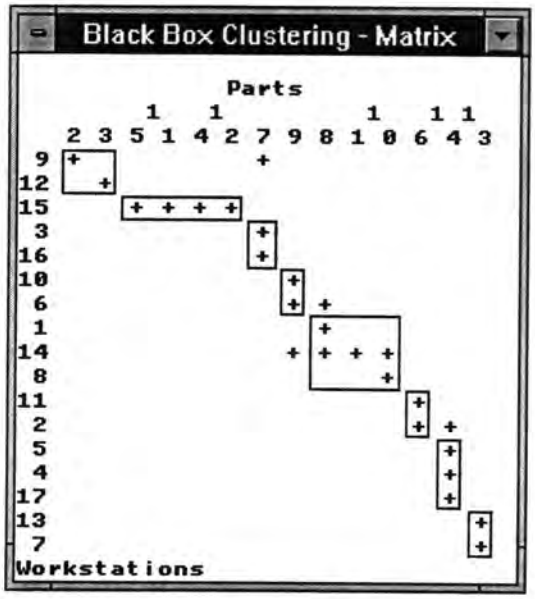


Figure 5.8: Possible alternative cell configurations for example BDF

5.3.2 The Cellect Layout Tools

As described in Chapter 3, the Cellect Layout Tools are made up of the Cell Positioning Tool and the Sequencing Tool. These can be run independently from each other (and the other Cellect tools). In the following description, Cell Positioning was

carried out first, then Sequencing. The example also shows how the positions of the candidate cell positions were updated to allow an iterative approach to solving the layout problem.

The data required to run the Cell Positioning Tool is the same as that required for the Sequencing Tool except that the Cell Positioning Tool also requires the positions of candidate cell centres. This data is shown in Table 5.4.

Table	Data Required	Notes
Workstations	Cell that workstation belongs to	
	Workstation name	
	X co-ordinate	for workstations that cannot be moved
	Y co-ordinate	as above
Candidate cell centres	X co-ordinate	
	Y co-ordinate	
Operations	Visiting point	
	Previous point	
	Next point	
	Material handling method from previous point	
	Material handling method to next point	
	Transfer batch size from previous point	
	Transfer batch size to next point	
Parts	Part code	
	Number off	
Products	Product name	
	Demand	
Material handling	Material handling method	
	Speed	
	Cost	

Table 5.4: Data required for Cellect layout tools

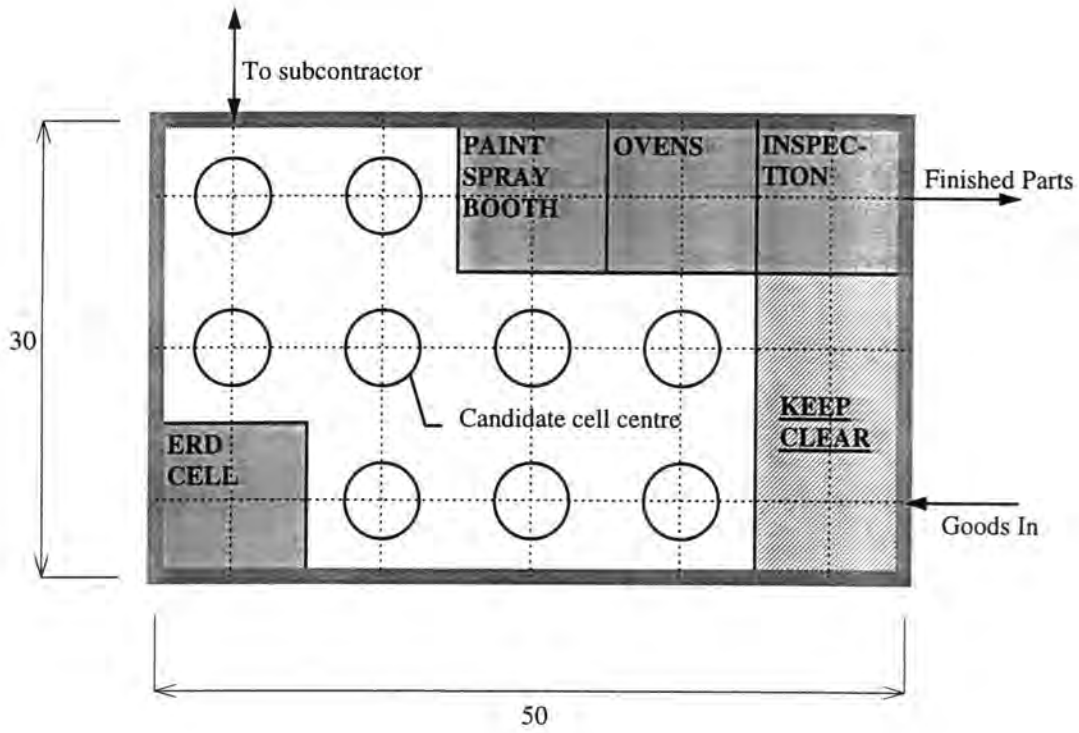


Figure 5.9: Shop floor prior to allocation of cells to candidate cell centres

Note that parameters can also be set to vary the grid of candidate cell centres, such as the grid size, the grid spacing and the grid start points, although at present this is not done via the database. Instead, these are altered in the source code of the grid producing program.

Figure 5.9 shows the example shop floor, the dimensions of which were 50 x 30 metres. For the list of candidate cell centres, a grid size of 10 metres squared was chosen and started 5 metres from each boundary. The name and positions of workstations that could not be moved were: paint spray booth (25, 25), ovens (35, 25), inspection (45, 25) and ERD cell (5, 5). These workstations were excluded from the analysis ('included in analysis' set to false as described in Section 5.2), which in this case means that their locations were used to determine travel costs of parts visiting them, but the Cell Positioning Tool is not able to override their co-ordinates.

Goods come in at point (50, 5) and exit the shop floor at point (50, 25). An entry/exit point for an adjoining, separate sub-contractor is at (5, 30). Points (45, 5) and (45, 15) are also deleted from the list of candidate cell centres to keep an area around the material entry point clear for access and for a set of partitioned offices.

With this data, the Cell Positioning Tool proceeds as described in Section 3.2.1. When complete, Cellect asks the user whether or not the database should be updated. If the answer is yes, the x and y co-ordinate fields for each workstation in the analysis (in the workstations table) are updated with the co-ordinates of their corresponding cell.

For the example data, the Cell Positioning Tool allocated cell centres as follows: CELL_1 (5, 25), CELL_2 (35, 5), CELL_3 (15, 25), CELL_4 (25, 5) and BRAKES (35, 15). From the number and estimated sizes of the workstations in each cell, it was decided that more appropriate cell centres might be as follows: CELL_1 (5, 20), CELL_2 (30, 5), CELL_3 (15, 20), CELL_4 (25, 15) and BRAKES (35, 15) (this is shown in Figure 5.10). To confirm these positions, the list of candidate cell centres was changed to the co-ordinates of the proposed cell centres and the Cell Layout Tool was re-run and showed that the new assignment of cell centres was indeed valid.

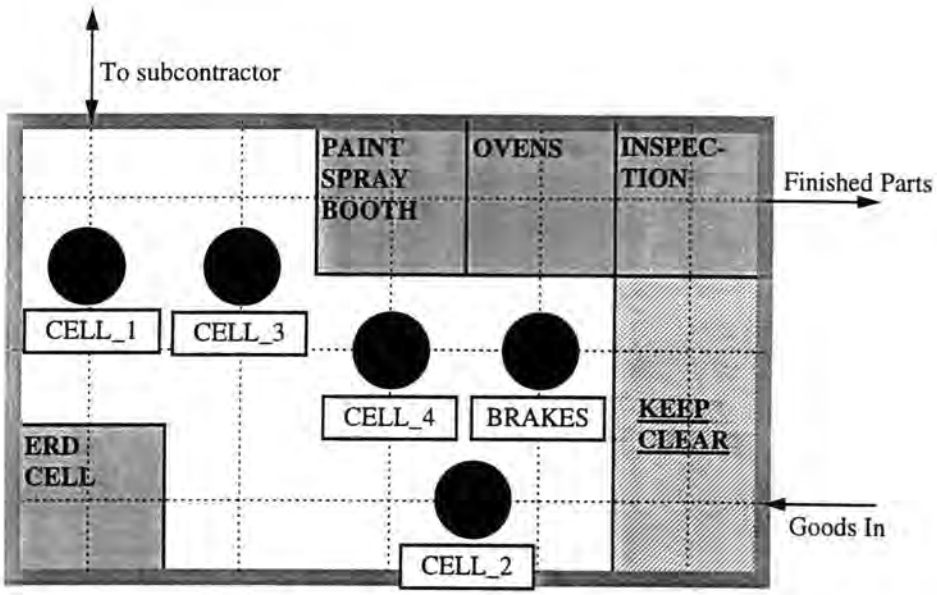


Figure 5.10: Final positions of cell centres

The Sequencing Tool requires all the data shown in Table 5.4, except the table of candidate cell co-ordinates. As far as the user is concerned, all that needs to be done is for the tool to be initiated and then once the algorithm described in Section 3.2.2 has completed, he or she needs to confirm whether or not the database ought to be updated. If the answer is yes, each workstation in the workstations table that has been included in the analysis is given a sequence number corresponding to its relative position within its cell.

The user now assign co-ordinates to the workstations within each cell based on where the workstation's cell centre is positioned (the results of the cell positioning tool) and the relative position of the workstation within the cell (the sequence numbers generated by the sequencing tool). Figure 5.11 shows the final arrangement of the workstations on the shop floor, the details of which can be shown via a form such as the one in Figure 5.6.

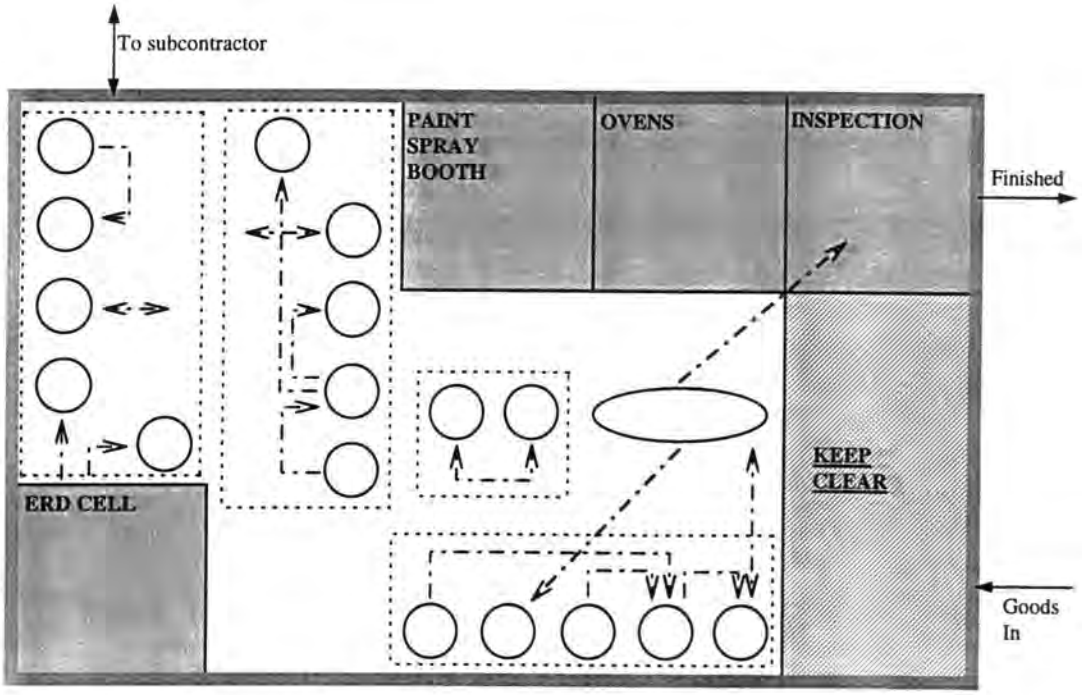


Figure 5.11: Workstation layout and some of the flows between them

The flows shown in Figure 5.11 indicate that the positioned cells are independent since material rarely travels out of the cells except when arriving from goods in or leaving to the finished goods stores. This was to be expected from the results of BBC described in Section 5.3.1 which produced cell configurations that had no exceptional elements. The only cell which has material travelling to other cells is the Brakes Assembly Cell (the oval shape in the figure). Because every part visits this cell, it was excluded from BBC since it was a cell in its own right. However, it was included in the cell positioning analysis in order to identify where it should be located. Material flow to other cells is unavoidable, because as stated, all parts visit this cell for final assembly. This fact also explains why the cell positioning tool chose this cell as the one that should be positioned nearest the finished goods store. Another notable aspect of the results shown in Figure 5.11 is that flows within each cell are generally unidirectional, suggesting that the workstations have been correctly sequenced.

5.3.3 Capability Analysis

So far, cell configurations have been determined by Black Box Clustering and workstations have been positioned on the shop floor according to these configurations using the Cellect Layout Tools. The final stage of Cellect is Capability Analysis which allows the user to compare manufacturing system performances to identify targets where improvements are required. In the previous chapter, it was stated that for certain capability factors, measures are *calculated* as opposed to *directly measured.* Hence, once workstation positions have been identified or if any data is changed, it is necessary to update calculated measures before carrying out Capability Analysis. In the current version of Cellect, this is done using an *Update* program which the user is prompted to run prior to running Capability Analysis or which can also be initiated at any other time. In the next chapter, a modification to Cellect is proposed that could eliminate the need for the Update program and instead have all calculated measures updated automatically. Figure 4.1 in the previous chapter shows capability factors, both directly measured and calculated, as well as how the calculated measures are determined and the data used to determine them. Thus, for ease of representation, in Table 5.5 which shows all the data required for Capability Analysis, the data required for calculated measures is not shown, since these can be identified from Figure 4.1.

Table	Data Required	Notes
Miscellaneous	Emphasis parameter	
	Target identification level	
Products	Product name	
	Demand	per production period
	Net profit	
	Total travel distance	travel distances, times and costs are for all parts belonging to the product
	Total travel time	

	Total travel cost	
Parts	Product name	
	Part name	
	Part code	
	Part family that part belongs to	
	Number off	
	Batch size	
	Nominal lead time	
	Total travel distance	
	Total travel time	
	Total travel cost	
Capability factors	Factor name	
	fc_oplev	see Notes in Table 5.1
	fc_prlev	
	fc_wslev	
	fc_ptlev	
	fc_oplev	
	order of data retrieval	
	Factor weighting	
Operations	Part code	
	Feature name	
	Sequence	
	Visiting point	
	Operation cost	
	Operation profit	profit of parent product of part served by operation
	Set-up time per part	
	Processing time	
	Defects	per million parts
	Time spent producing defects	per million parts calculated measure
	Cost of defect production	per million parts calculated measure
	Distance from previous operation	calculated measure
	Distance to next operation	calculated measure
	Time from previous operation	calculated measure
	Time to next operation	calculated measure
	Cost from previous operation	calculated measure
	Cost to next operation	calculated measure
Workstations	Workstation name	
	Cell that workstation belongs to	
	Workstation profit	calculated by Updates program as total profit of parent products for all parts visiting the workstation

	Down time of workstation	
	Labour time at workstation	
	Cost of labour	calculated measure
	Part similarity	calculated measure
	Usage	calculated measure
	Total distance from previous workstations	distances, times and costs are for all parts visiting the workstation (see Figure 5.8) calculated measure
	Total distance to next workstations	calculated measure
	Total time from previous workstations	calculated measure
	Total time to next workstations	calculated measure
	Total cost from previous workstations	calculated measure
	Total cost to next workstations	calculated measure
Cells	Cell name	this table is produced entirely by the Updates program
	Cell profit	calculated by Updates program as total profit of parent products for all parts visiting the cell
	Total distance from previous cells	distances, times and costs are for all parts visiting the cell (see Figure 5.8) calculated measure
	Total distance to next cells	calculated measure
	Total time from previous cells	calculated measure
	Total time to next cells	calculated measure
	Total cost from previous cells	calculated measure
	Total cost to next cells	calculated measure

Table 5.5: Data Cellect requires for Capability Analysis

Capability Analysis proceeds through the methodology described in the previous chapter to produce the recovery schedules, which in Cellect are all stored in one table called Targets. This table has the following data entered into it (non-applicable data is assigned the FoxPro .NULL. value): recovery schedule level (an identifier describing which level the target belongs to), cell name, workstation name, product name, part code, part name, feature name, sequence, capability factor, priority confidence score,

marginal capability, required measure, importance weighting, measure, required measure, worst measure, best measure, profit weighting and factor weighting.

Forms such as the one shown in Figure 5.12 can now be used to view specific recovery schedules. For the target shown in this figure, the following should be noted.

(i) The targeted operation is 'Cut and Weigh' of armature parts at the Moulding-Bench. Further information provided is that the part belongs to the ERS23 product and that the operation takes place within CELL_4. A part code and sequence number of the workstation are also provided.

(ii) The factor causing concern is the 'Operation Cost of Defects'. The marginal capability of 100% indicates that this is the worst capability score. This is confirmed by the fact that the capability score for this target and the worst capability score of the whole of the collated group that the target belongs to are both the same.

(iii) The difference between the capability score and the required capability score is very high. The required capability score, when compared with this worst capability score is near the optimum value of zero which explains why the improvement potential is near one. This high improvement potential suggests that this is an important target.

(iv) Although the factor weighting is low, the profit weighting is high, further suggesting that this is an important target.

(v) The current version of Cellect does not have a specially developed tool to carry out transparency. However, if for this example, the user wanted to identify related targets to find the cause of this target then the 'Find'

button could be pressed. A FoxPro form would be displayed and this could be used to identify the number of defects for this and other operations that take place at Moulding-Bench. This extra information may provide an insight to the cause of the problem.

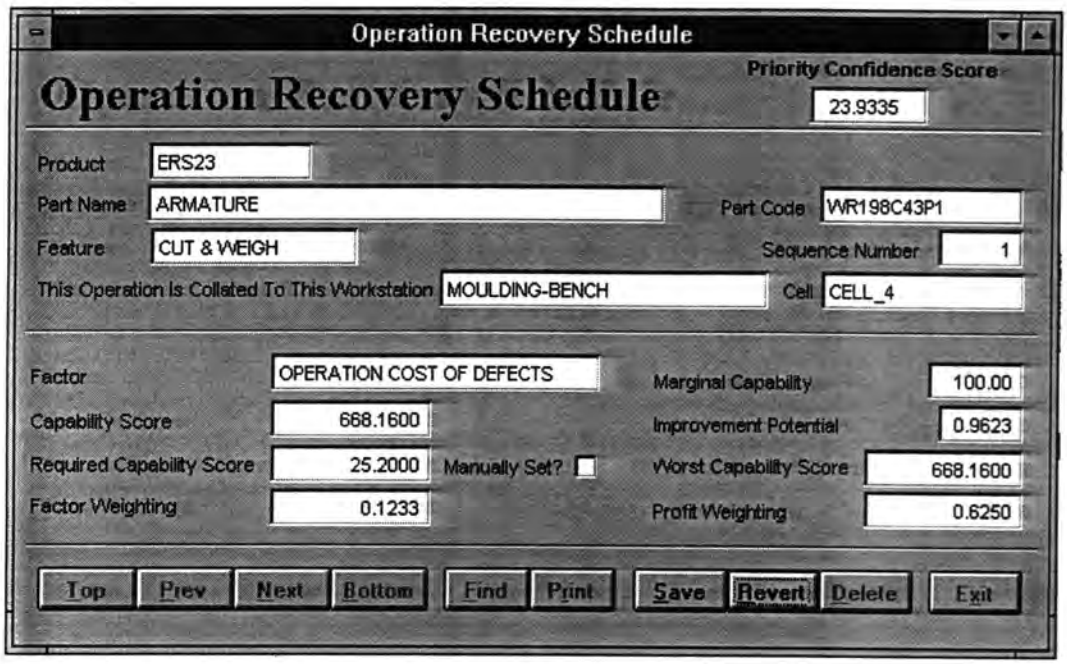


Figure 5.12: Operation level recovery schedule

5.4 Summary

This Chapter described the implementation of Cellect as a Visual FoxPro application. It concentrated on the structure of a relational database to act as the central information store for the three Cellect Tools. The running of Cellect was demonstrated using an industrial example.

6. Conclusions and Scope for Further Work

6.1 Discussion

The main objective of this project was to develop new methods and a prototype software tool to carry out cell design, management and continuous improvement. It was set up in response to the following needs:

(a) Cell design and improvement is an ongoing process involving both designers and shop-floor staff.

(b) The capability of a cellular manufacturing system should be assessed regularly.

(c) Cell design should be straight-forward.

(d) A single data source should be used to ensure data reliability and efficiency.

The name of the system developed to satisfy the above requirements is Cellect. It comprises three elements.

(i) Determining cell configurations.

(ii) Layout of the cells and the workstations within them.

(iii) Capability Analysis to identify targets for continuous improvement.

Each element represents a stage in the cell design, management and continuous improvement process. Cell design involves determining cell configurations. These are the groups of workstations that together form a cell and the groups of parts that

together make up a part family. An algorithm called Black Box Clustering (BBC) was developed to determine cell configurations from routings without intervention from the user. It has been demonstrated that BBC is able to identify clusters of workstations and parts that are at least equally as good as the results of a number of different algorithms developed by various authors.

Having identified cell configurations, the next stage of cell design is to position the workstations on the shop floor for efficient flow of material. It was felt that because of the large number of qualitative factors involved in this process, layout design should be done with as much shop floor involvement as possible. To this end, the Cellect Layout Tool determines approximate positions of cells and the sequence of workstations within them based on material handling data. With this information, the user is able to consider all qualitative factors to find the exact positions of workstations.

The final element of Cellect is Capability Analysis (CA). This was developed because cell configurations are rarely ideal. It is, therefore, essential to identify weaknesses within the manufacturing system to be acted upon by both shop floor staff and designers. Shop floor staff are able to eliminate weaknesses within the manufacturing system by improving the capabilities of processes, whereas designers are able to maintain the effectiveness of processes by designing products that can be manufactured without exceeding existing capabilities.

To allow these activities to take place, CA considers a number of factors that affect the performance of a manufacturing system to produce a list of targets called the recovery schedule. The concepts that constitute CA are based on the fact that capabilities can be represented as capability scores which can be divided into groups of capability scores that should be similar. By comparing capability deficiencies as a proportion of required

capability score for a given group, it is possible to compare different factors with one another.

The presentation of the recovery schedule allows the user to identify precisely capability deficiencies at a detailed level that contribute to a capability deficiency at a higher, more abstract level.

Microsoft Visual FoxPro version 3 is used for the implementation of Cellect. The focus of Cellect is a set of database tables from which all three tools can access data and into which they can input data. The use of the Cellect software was demonstrated with example data as follows.

Black Box Clustering

Relevant data was extracted in the form of a machine-part incidence matrix which was clustered into cell configurations using BBC. Default cell names were given to workstations to indicate the workstation groups that they belonged to and to parts to indicate the part families that they belonged to. All this information was inputted back into the database. The whole process occurred without user intervention.

Cellect Layout Tools

Cell configurations and material handling data was extracted from the database. Also extracted was data concerning the floor space used for positioning the workstations. This data was in the form of user-defined candidate cell centres. The Cell Positioning Tool allocated cells (workstation groups) to candidate cell centres and the Sequencing Tool identified the relative positions of workstations within each cell. This information was inputted into the database and presented to the user who was able to determine precise workstation positions whilst taking into consideration information about

qualitative factors which are difficult to model, yet readily available to him or her. Workstation co-ordinates were inputted into the database.

Capability Analysis

With all the workstations positioned, targets for improving the example manufacturing system were determined using CA. This was done by extracting from the database data that was relevant to the selected capability factors. This data was converted into capability scores. Using the CA methodology, capability scores were assessed with respect to: (a) user-defined and automatically set required capability scores, (b) user-defined capability factor weightings, (c) calculated improvement potentials and (d) calculated profit weightings. For each capability score this information was represented as a priority confidence score (PCS). These PCS values were ranked to form recovery schedules for each capability level.

Another feature of Cellect is that the database is able to store information about parts and workstations that do not exist within the manufacturing system. By choosing to include these in the analysis, the user is able to determine cell configurations, workstation positions and capabilities when using state-of-the-art machines or for products not yet designed.

6.2 Scope For Further Work

As discussed in Chapter 1, external elements allow the use of more accurate or approved data using methods such as simulation, shop floor data capture and the process planning methods of tools such as Durham University's CAPABLE. Of particular importance is the fact that within CA it is desirable that capability scores

should be directly measured as opposed to calculated. Although approved data direct from the shop floor is the preferred source of data, in the case of parts not yet manufactured, this information is not available and so specialist software that is able to simulate a manufacturing system to determine these capability scores is the next best option. It is thus suggested that data from simulation software should also be considered directly measured, since when data is not available from the shop floor and providing that the manufacturing system is modelled accurately, simulation provides the most reliable source of data for CA. This fact has been recognised in the research of Higgins (1997) which has provided a link between the Cellect database and the Witness simulation software. This research used the routings in the Cellect database to carry out automatic generation of factory models and provided the user with extra information such as queue lengths and waiting times. These were modelled based on three different statistical distributions for the arrival of parts at workstations.

It is therefore suggested that an area warranting further research is the integration of Cellect within a Computer Integrated Manufacturing (CIM) system to provide real-time CA without the need to run the Update program presently within Cellect (when data stored in a table is updated, FoxPro is unable to automatically update calculated values that rely on this updated data, hence the need for the Update program). A good start would be to have all calculations performed in a spreadsheet and have an OLE link (Microsoft's Object Linking and Embedding system for on-the-fly data transfer between applications) between the database and the spreadsheet. Instead of running an Update program, each time data is changed, the spreadsheet would register this and data such as PCS values would automatically be recalculated and reinputted into the database.

Another useful area of research would be identifying capability factors that allow CA to be extended. As mentioned in Chapter 4, CA methods are intended to be as generic as possible to allow other capability factors to be included in the analysis with the minimum of extra development time. Such factors could be those corresponding to a feature capability level. For example, feature similarity, required tooling, tool approach paths and manufacturing times and costs. It is worth bearing in mind that features do not collate to operation level (features in an operation can be different) nor to any other level and so a higher feature-type level needs to be defined to allow collating. For example, all through-holes would be collated together, as would milled shoulders, threaded features and so on.

There is much research still taking place to identify effective cell formation algorithms. This research should be monitored in order to identify further enhancements to BBC. Also, the Cellect Layout Tool could be modified to take into consideration the fact that layouts can be improved by taking into consideration intercell material flows. Also ignored in this algorithm is the fact that the size and shape of workstations may affect the amount of movement between them.

6.3 Concluding Remarks

Previous research into the design of cellular manufacturing systems has tended to concentrate on only one or two aspects of the problem or has not been specific enough. Those aspects that have been examined the most have been cell formation algorithms and layout design. Architectures for expert cell design systems have been proposed, but none of these have been implemented. Also, little has been done to address the issue of how to manage and improve the cellular manufacturing system

once it has been designed. This research showed that a need exists for an integrated cell design tool that is suitable for use by shop floor staff and that the methodologies should be simple enough to allow this tool to be implemented either manually or within a specialist software system. The novelty of this research is as follows:

(i) A formal definition of the three stages of cell design as determining cell configurations, layout design and capability analysis.

(ii) An algorithm to determine cell configurations without any user intervention.

(iii) A methodology to determine layouts of cells and the workstations within them using two algorithms combined with user interaction.

(iv) A methodology for assessing capabilities of cellular manufacturing systems to determine targets for improvement.

(v) A prototype database-driven software tool to carry out cell design, management and continuous improvement by implementing the methodologies and algorithms outlined above.

(vi) An overall cell design strategy suitable for use within a Concurrent Engineering environment.

References

Abdou G., and Dutta S.P. (1990), "An integrated approach to facilities layout using expert systems", *International Journal of Production Research*, Vol. 28, No. 4, pp. 685-708

Adil G.K., Rajamani D., and Strong D. (1996), "Cell formation considering alternative routings", *International Journal of Production Research*, Vol. 34, No. 5, pp. 1361-1380

anon (1989), "Transferring ownership and enthusiasm", *Engineering with Computers*, Vol. 8, No. 1, p. 29

Askin R.G. and Stanbridge C.R. (1993), *Modelling and Analysis of Manufacturing Systems*, Wiley

Askin R.G., and Subramanian S.P. (1987), "A cost based heuristic for group technology configuration", *International Journal of Production Research*, Vol. 25, pp. 101

Arvindh B., and Irani S.A. (1994), "Cell formation: the need for an integrated solution of the subproblems", *International Journal of Production Research*, Vol. 32, No. 5, pp. 1197 - 1218

Baker R.P. and Maropoulos P.G. (1997a), "An Automatic Clustering Algorithm Suitable For Use By A Computer-Based Tool For The Design, Management And Continuous Improvement Of Cellular Manufacturing Systems", *Computer Integrated Manufacturing Systems*, Vol.10, No.3

Baker R.P. and Maropoulos P.G. (1997b), "Manufacturing Capability Assessment for Cellular Manufacturing Systems", *International Journal of Production Research*, Vol 36, No 9, pp. 2511-2527

Baker R.P. and Maropoulos P.G. (1997c),"Cell Design, Management and Continuous Improvement, Part 1: Cell Configurations and Layouts", *International Journal of Advanced Manufacturing Technology*, submitted for publication

Baker R.P. and Maropoulos P.G. (1997d),"Cell Design, Management and Continuous Improvement, Part 2: Capability Analysis for Continuous Improvement", *International Journal of Advanced Manaufacturing Technology*, submitted for publication

Ballakur, A. and Steudal, H. J. (1987), "A within-cell utilization based heuristic for designing cellular manufacturing systems", *International Journal of Production Research*, Vol. 25, No. 5, pp. 639-665.

Bartlett H., Baxevanoglou A., and Kochhar A. K. (1994), "The application of systematic techniques to the re-layout of a low-volume manufacturing system", *Procedures of the Institution of Mechanical Engineers Part B*, No. 208, pp. 89-102

Basu A., Hyer N., Shtub A. (1995) "An expert system based approach to manufacturing cell design", *International Journal of Production Research*, Vol 33, No 10, pp. 2739-2755

Bennett, D. (1985), *Production Systems Design*, Butterworths

Benjaafar S. (1995) "Machine sharing and performance of cellular manufacturing systems", *Journal of Materials Processing Technology*, Vol 52, No 1, pp. 91-101

Bradley H.D. and Maropoulos (1997) "Aggregate process planning: a methodology for supporting concurrent engineering", *32nd International MATADOR Conference 1997* 10-11 July 1997, accepted for publication

Buffa E.S., Armour G.C., and Vollmann T.E. (1964), "Allocating facilities with CRAFT", *Harvard Business Review*, Vol. 42, No. 2, pp. 136-159

Burbidge J.L. (1975), *The Introduction of Group Technology*, John Wiley and Sons

Carrie A.S. (1973), "Numerical taxonomy applied to group technology and plant layout", *International Journal of Production Research*, Vol. 11, pp. 399

Chan F.T.S., Jayaprakash B. and Tang N.K.H. (1995) "Design of automated cellular manufacturing sustems with simulation modelling - a case study", *International Journal of Computer Applications in Technology*, Vol 8, Nos 1-2, pp. 1-11

Chanan S. S. and Menon U (1994), *Concurrent Engineering: Concepts, Implementation and Practice*, Chapman and Hall

Chandrasekharan M.P., and Rajagopalan R. (1986a), "MODROC: an extension of rank order clustering for group technology", *International Journal of Production Research*, Vol. 24, No. 5, pp. 1221-1233

Chandrasekharan M. P., and Rajagopalan R. (1986b), "An ideal seed non-hierarchical clustering algorithm for cellular manufacturing", *International Journal of Production Research*, Vol. 24, No. 2, pp. 451-464

Chandrasekharan M.P., and Rajagopalan R. (1987), "ZODIAC - an algorithm for concurrent formation of part-families and machine-cells", *International Journal of Production Research*, Vol. 25, No. 6, pp. 835-850

Chen S.J., and Cheng C.S. (1995), "A neural network-based cell formation algorithm in cellular manufacturing", *International Journal of Production Research*, Vol. 33, No. 2, pp. 293-318

Chen C.L., Cotruvo N.A., and Baek W. (1995), "A simulated annealing solution to the cell formation problem", *International Journal of Production Research*, Vol. 33, No. 9, pp. 2601-2614

Chen C.Y., and Irani S.A. (1993), "Cluster first-sequence last heuristics for generating block diagonal forms for a machine-part matrix", *International Journal of Production Research*, Vol. 31, No. 11, pp. 2623-2647

Chen F.F. and Sagi S.R. (1995) "Concurrent design of manufacturing cell and control functions: a neural network approach", *Intenational Journal of Advanced Manufacturing Technology*, Vol 10, No 2, pp. 118-130

DeWitte J. (1980), "The use of similarity coefficients in production flow analysis", *International Journal of Production Research*, Vol. 18, No. 4, pp. 503-514

Fan I.S., and Gassmann R. (1995), "Study of the practicalities of human centred implementation in a British manufacturing company", *Computer Integrated Manufacturing Systems*, Vol. 8, No. 2, pp. 151-154

Gallagher C.C. and Knight W.A. (1973), *Group Technology*, Butterworths

Goldberg D.E. (1989), *Genetic Algorithms In Search, Optimisation And Machine Learning*, Adison-Wesley

Harhalakis G., Nagi R., and Proth J.M. (1990), "An efficient heuristic in manufacturing cell formation for group technology application", *International Journal of Production Research*, Vol. 28, pp. 185-198

Harrison A. (1992), *Just-In-Time Manufacturing In Perspective*, Prentice Hall

Hassan M.M.D. (1995), "Layout design in group technology manufacturing", *International Journal of Production Economics*, Vol. 38, pp. 173-188

Hentzen W. (1995), *Programming Visual FoxPro 3.0*, Ziff-Davis Press (Emeryville, Ca.)

Hey E. J., (1988), *The Just-In-Time Breakthrough*, John Wiley and Sons

Higgins A. (1997), *Automating the Simulation of Manufacturing Systems by Integrating Database Technology and Discreet Event Simulation Modelling through Object Oriented Programming*, Final Year Report for the degree of Master of Engineering

Hyde W.F. (1981), *Improving Productivity by Classification Coding and Database Standardization: The Key to Maximizing CAD/CAM and Group Technology*, Marcel Dekker (New York)

Irani S.A., Cohen P.H., and Cavalier T.M. (1992), "Design of cellular manufacturing systems", *Journal of Engineering for Industry*, Vol. 114, No. 3, pp. 352-361

Johnston R.B. (1995), "Making manufacturing practices tacit: a case srudy of computer-aided production management and lean production", *Journal of the Operational Research Society*, Vol. 46, No. 10, pp. 1174-1183

Jojodia S., Minis I., Harhalakis G., and Proth J. (1992) "CLASS: Computerized layout solution using simulated annealing", *International Journal of Production Research*, Vol. 30, No. 1, pp. 95-108

Kandiller L. (1994), "A comparative study of cell formation in cellular manufacturing systems", *International Journal of Production Research*, Vol. 32, No. 10, pp. 2395-2429

Kasilingam R.G., and Lashkari R.S. (1991), "Cell formation in the presence of alternate process plans in flexible manufacturing systems", *Production Planning and Control*, Vol. 2, No. 2, pp. 135-141

Kazerooni M., Luong L.H.S., and Abhary K. (1995), "Machine chain similarity, a new approach for cell formation", *IMechE Production Automation Conference 1995*

Keus J.A., Rome C.P., and Van Zoelan, G.J. (1977), "Implementation of the group technology concept for the manufacture of 544 parts for electro-mechanical products with the aid of the MICLASS-package", *CIRP Manufacturing Systems*, No. 6, p. 167

King, J. R. (1980), "Machine-component group formation in production flow analysis: an algorithm using a rank order clustering algorithm", *International Journal of Production Research*, Vol. 18, pp. 213-232

King J.R., and Nakornchai V. (1982), "Machine-component group formation in group technology: review and extension", *International Journal of Production Research*, Vol. 20, No. 2, pp. 117-133

Koenigsberger, F. (1972), "The Use of Group Technology in the Industries of Various Countries", *CIRP Keynote Paper*

Lanigan M. (1992), *Engineers in Business: The Principles of Management and Product Design*, Addison-Wesley, Wokingham

Lee R.C., and Moore J.M. (1967), "CORELAP - COmputerized RElationship LAyout PLanning", *Journal of Industrial Engineering*, Vol. 18, No. 3, pp. 195-200

Leskowsky Z., Logan L., and Vannelli A. (1987), "Group technology decision aids in an expert system for plant layout", A. Kusiak (Ed.), *Modern Production Management Systems*, North-Holland, Amsterdam

Logendran R. (1993) "Methodology for converting a functional manufacturing system into a cellular manufacturing system", *International Journal of Production Economics*, Vol 29, No 1, pp. 27-41

Lynch R.L. and Cross K.F. (1991), *Measure Up! Yardsticks For Continuous Improvement*, Blackwell Business, Oxford

Malakooti B., and Yang Z. (1995), "A variable-parameter unsupervised learning clustering neural network approach with application to machine-part group formation", *International Journal of Production Research*, Vol. 33, No. 9, pp. 2395-2413

Maskell B.H. (1991), *Performance Measurement for World Class Manufacturing: a Model for American Companies*, Productivity Press, Cambridge Ma.

Massey L.L., Udoka S.J. and Benjamin C.O. (1995) "A simulator-based approach to cellular manufacturing system design", *Computers and Industrial Engineering*, Vol 29, pp. 327-331

McAuley J. (1972), "Machine grouping for efficient production", *Production Engineer*, Vol. 51, No. 2, pp. 53-57

Miller, D. (1990), *The Icarus Paradox: How Exceptional Companies Bring About Their Own Downfall: New Lessons in the Dynamics of Corporate Success, Decline and Renewal*, Harper Business

Morris J.S. and Tersine R.J. (1994) "A simulation comparison of process and cellular layouts in a dual rosource constrained environment", *Computers and Industrial Engineering*, Vol 26, No 4, pp. 733-741

Mukhopadhyay S. K., Gopalakrishnan A., and Kripalani M. K. (1995), "Moments-based clustering techniques for manufacturing cell formation", *International Journal of Production Research*, Vol. 33, No. 4, pp. 1091-1115

Perrego T.A., Petersen H.C., and Hahn W.F. (1995), "The Perrego algorithm: a flexible machine-component grouping algorithm based on group technology techniques", *International Journal of Production Research*, Vol. 33, No. 6, pp. 1709-1721

Schonberger R. (1986), *World Class Manufacturing*, Free Press, New York

Seehof J.M., and Evans W.O. (1967), "Automated layout design program", *Journal of Industrial Engineering*, Vol. 18, pp. 690-695

Sekine K. (1992), *One-Piece Flow: Cell Design for Transforming the Production Process*, Productivity Press

Shingo S. (1986), *Zero Quality Control: Source Inspection and the Poka-Yoke System*, Productivity Press

Tam K.Y. (1992), "A simulated annealing algorithm for allocating space to manufacturing cells", *International Journal of Production Research*, Vol. 30, No. 1, pp. 63-87

Tam K.Y., and Li S.G. (1991), "A hierarchical approach to the facility layout problem", *International Journal of Production Research*, No. 29, No. 1, pp. 165-184

Waghodekar P.H., and Sahu S. (1984), "Machine-component cell formation in group technology: MACE", *International Journal of Production Research*, Vol. 22, pp. 937

Wemmerlöv U., and Heyer N. L.(1989), "Cellular Manufacturing in the U.S. industry: a survey of users", *International Journal of Production Research*, Vol. 27, No. 9, pp. 1511-1530

William G.B., Davis L.E. and Butcher P.A. (1993) "Cell based manufacturing and measures of performance", *IFIP Transactions B - Applications in Technology*, Vol 13, pp. 561-568

Woolfe E., Tanna S., and Singh K. (1987), *Economics*, Hutchinson Education (London)