7/14

A DECISION THEORETIC APPROACH TO THE GENERAL LAYOUT PROBLEM

by

Pasu Loharjun

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in
Industrial Engineering and Operations Research

APPROVED:

Subhash C. Sarin, Chairman

Marvin H. Agee

Wolter J. Fabrycky

Charles J. Malmborg

Marilyn S. Jones

March, 1986

Blacksburg, Virginia

A DECISION THEORETIC APPROACH TO THE GENERAL LAYOUT PROBLEM

by

Pasu Loharjun

Subhash C. Sarin, Chairman

Industrial Engineering and Operations Research

(ABSTRACT)

This research is devoted to the development of a multiobjective facility layout creation methodology. This methodology seeks to extend the scope of existing computerized and manual layout creation methods by capturing a greater level of both intuitive and quantitative inputs in a method applicable for moderate to large-scale problems. To do this, an extended theoretical basis for decision theoretic models applicable to layout design is described. Using these models as an evaluation basis, a new optimizing layout creation strategy is developed and a decision support system for its implementation is presented. The new layout creation method is computationally attractive, and based on extensive computational experience, is found to give better solutions than those generated by CORELAP.

DEDICATION

This work is dedicated to my Grandfathers,

to whom education was always so important.

I think they would have been proud.

ACKNOWLEDGEMENTS

The completion of a Ph.D. dissertation has been a monumental goal in my life. There are many people without whom I would not have been able to reach this goal.

I would like to thank all the members of my advisory committee for their valuable assistance throughout the various stages of this research. I wish to especially thank Dr. Charles J. Malmborg for introducing me to this research topic and giving useful insight during the early phase of this research. I would also like to thank Dr. Marvin H. Agee for his constructive comments.

Foremost, I would like to thank Dr. Subhash C. Sarin, my chairman, for his valuable guidance, suggestions, insight and timely encouragement in carrying out this research and preparing the final draft.

I would also like to thank Drs. Wolter J.Fabrycky and Robert D. Dryden for awarding me Graduate Teaching Assistantships during 1983-1985.

A special and personal regards to my parents for their love and continuous support during my education at Virginia Tech.

TABLE OF CONTENTS

1. INTRODUCTION 1

1.1 Statement of Problem 1

1.2 Objectives of the Research 3

1.3 Organization of the Dissertation 4

2. LITERATURE REVIEW 6

2.1 Review of Literature on Facility Layout Design Approaches 6

2.1.1 Computerized Approaches 10

2.1.2 Multicriteria Approaches 35

2.1.3 Graph Theoretic Approaches 38

2.1.4 Mathematical Modeling Approaches 39

2.1.5 Summary and Conclusion 44

2.2 Literature Review of Decision Theory 45

2.2.1 Multiattribute Value Theory 46

2.2.2 Measurable Multiattribute Value Functions . . 52

2.2.3 Multiattribute Value Function: A Binary Case . 54

2.2.4 Summary and Conclusion 56

3. DECISION THEORETIC APPROACHES TO THE GENERAL PLANT LAYOUT PROBLEM 58

3.1 Introduction 58

3.1.1 Statement of Need for the Alternative Approach 58

3.1.2 Objectives of the Alternative Approach 59

3.1.3 Overview of the Chapter 61

3.2 Measurement Considerations 61

3.2.1 The Multiattribute Nature of the Facility Layout Problem 62

3.2.2 The Need to Consider the Mixed Continuous and Discrete Multiattribute Problem 63

3.2.3 The Insufficiency of Present Axiomatic Systems 64

3.2.4 An Axiomatic System for the Mixed Case 68

3.2.5 Summary and Conclusion 69

3.3 Application of Decision Theory to the Layout Design Problem 69

3.3.1 Extension of the REL Construct to Incorporate Multiattribute Value Functions 71

3.3.1.1 Efficient Assessment of the Value Functions 73

3.3.1.2 Overview of Assessment Software 74

3.3.2 Utilization of the Multiattribute Value Functions in Layout Creation 75

3.3.2.1 Description of the Procedure 75

3.3.2.2 Advantages Over Existing Methods 76

3.3.3 Overall Multiattribute Evaluation of Layout Alternatives 76

3.4 The Main Steps of the Proposed Procedure for layout Design 77

3.5 Summary and Conclusions 83

4. A DECISION SUPPORT SYSTEM FOR FACILITY LAYOUT . . . 84
4.1 The Man Machine Concept of Decision Making Applied to Facility Layout 85
4.1.1 Description of Comparative Advantages 85
4.1.2 Overview of the Decision Support System . . . 86
4.2 Description of the Decision Support System 87
4.2.1 Data Requirements 87
4.2.2 User Interaction 89
4.2.3 Assumptions of the Modeling Approach 89
4.2.4 Interfacing the Modules of the System 90
4.2.5 Advantages of the System 91
4.3 Summary and Conclusions 92

5. AXIOMATIC BASIS FOR MIXED ATTRIBUTE CASES 93
5.1 Notations and basic definitions 94
5.1.1 Alternative space 94
5.1.2 Weak ordering 94
5.1.3 Essentialism 95
5.1.4 Archimedean property 95
5.1.5 Double cancellation 96
5.1.6 Solvability 96
5.2 Mixed additive case using compensatory independence 96
5.2.1 Definitions of independence conditions 97
5.2.1.1 Preferential independence 97
5.2.1.2 Mutual preferential independence 97
5.2.1.3 Compensatory independence 98

5.2.2 Theorem 98

5.2.3 Proof 99

5.3 Mixed additive case using Epsilon solvability . 102

5.3.1 Definition of epsilon solvability 102

5.3.2 Theorem 104

5.3.3 Proof 105

5.4 Mixed multilinear case 108

5.4.1 Theorem 108

5.4.2 Proof 109

5.5 Mixed multiplicative case 112

5.5.1 Theorem 112

5.5.2 Proof 113

5:6 Mixed additive interdependent case 115

5.6.1 Theorem 115

5.6.2 Proof 117

5.7 Summary and Conclusion 120

6. METHODOLOGY FOR CREATING LAYOUT ALTERNATIVES . . 121

6.1 Layout design as a quadratic assignment problem 122

6.2 Literature Review of Procedures to Solve the QAP 124

6.2.1 Exact methods employing Branch and Bound . . 125

6.2.2 Exact methods not employing Branch and Bound 131

6.2.3 Heuristic Methods 136

6.3 A Solution procedure for the determination of the initial layout 142

6.4 A procedure for determining the Final Layout . . 151

6.5 The overall layout design procedure 153

6.6 An Example Problem 155

6.6.1 Description of the Example Problem 155

6.6.2 Layouts obtained using CORELAP and ALDEP . . 157

6.6.3 Results of experimentation with the proposed procedure 167

6.7 Summary and conclusion 184

7. SUMMARY, CONCLUSION, AND RECOMMENDATIONS 185

7.1 Summary and Conclusion 186

7.1.1 A decision theoretic approach to layout design 186

7.1.2 Justification of the decision theoretic approach 187

7.1.3 Solution procedures for the quadratic assignment problem 188

7.1.4 Decision support system for the layout design problem 189

7.1.5 Results of Experimentation 191

7.2 Recommendation 191

BIBLIOGRAPHY 193

APPENDIX A. DESCRIPTION OF PROGRAMS IN THE DECISION SUPPORT SYSTEM 199

APPENDIX B. A COMPUTER RUN OF THE EXAMPLE PROBLEM . 203

APPENDIX C. RANDOM MATRIX GENERATOR 246

VITA 247

LIST OF ILLUSTRATIONS

Figure 1.1 Principle categories of factors for for consideration in facility design 2

Figure 2.1 Layout example problem 12

Figure 2.2 Flow chart of CRAFT 14

Figure 2.3 CRAFT's creation process 15

Figure 2.4 Flow chart of PLANET 19

Figure 2.5 PLANET's layout creation process 20

Figure 2.6 Flow chart of PLOP 21

Figure 2.7 Flow chart of QUAINT 22

Figure 2.8 Output from INLAYT and S-ZAKY, Flow chart 26

Figure 2.9 Flow chart of CORELAP 29

Figure 2.10 Flow chart of CORELAP's selection rule 30

Figure 2.11 CORELAP's layout creation process 31

Figure 2.12 ALDEP's layout creation process 32

Figure 2.13 Sample output from RMA Comp I 34

Figure 2.14 Bazaraa's modeling aaaproach to the layout problem 42

Figure 2.15 DISCON's process 43

Figure 3.1 Multiattribute value function classifications 66

Figure 4.1 Man-machine interaction for facility layout problem 88

Figure 6.1 Flow chart of the layout design procedure. 156

Figure 6.2 Criteria specification 158

Figure 6.3 Department specification and size 159

Figure 6.4 Interdepartmental personnel flow between departments 160

Figure 6.5 Closeness rating for Common personnel (Max 10,Min 0) 161

Figure 6.6 Closeness rating for Common data used (Max 10,Min 0) 162

Figure 6.7 Closeness rating for Contact necessary (Max 10,Min 0) 163

Figure 6.8 Closeness rating for Convenience (Max 10,Min 0) 164

Figure 6.9 Marginal value functions 165

Figure 6.10 Preference matrix 166

Figure 6.11 Layout created by CORELAP 168

Figure 6.12 Layout created by ALDEP 169

Figure 6.13 Initial layout and final layout (designated A1) when all departments are assumed to be unequal 171

Figure 6.14 Initial layout and final layout (designated B1) for the criterion of Interdepartmental flow 172

Figure 6.15 Initial layout and final layout (designated B2) for the criterion of Common personnel 173

Figure 6.16 Initial layout and final layout (designated B3) for the criterion of Common information 174

Figure 6.17 Initial layout and final layout (designated B4) for the criterion of Contact necessary 175

Figure 6.18 Initial layout and final layout (designated B5) for the criterion of Convenience 176

Figure 6.19 Initial layout and final layout (designated B6) when all criteria considered together 177

Figure 6.20 Scores of layouts created by CORELAP, ALDEP, and the proposed procedure under various conditions 178

Figure 6.21 Scores of layouts generated by CORELAP and the proposed procedure for problem of different size 181

Figure 6.22 Graph showing percentage improvement of the proposed procedure over CORELAP versus problem size 182

Figure 6.23 Performance curve of CORELAP and the proposed procedure versus problem size . . 183

1. INTRODUCTION

Layout design is one of the most crucial problem areas in manufacturing as well as most other business activities. The work of layout designers covers a wide range. Some idea of the complexity and scope of the tasks facing facility designers can be obtained by studying Figure 1.1 (Apple, 1977), which lists the categories of factors to be considered in the design of a facility. In general, the design of an efficient layout can be best achieved if the problem is approached in a logical, systematic manner. This research concerns the development of a general layout procedure which also considers these factors.

1.1 STATEMENT OF PROBLEM

Most of the research relating to the layout design problem is focused on the construction of the layout (refer to Chapter 2). Researchers have tried to find ways to locate functional departments or machines in order to maximize predefined objective functions. Generally, these objective functions are related to material handling cost and closeness ratings for department pairs. Manual methods refer to graphical methods that attempt to arrange and/or move departments within a facility until a satisfactory layout plan

1. Building
2. Business trends
3. Communications
4. Community
5. Competition
6. Costs
7. Customer
8. Distribution
9. Convenience
10. Economic
11. Ecology
12. equipment
13. expansion
14. Financial
15. Fire protection
16. Flexibility
17. flow
18. Government/Legal
19. Grounds
20. Health
21. Inspection
22. Intangibles
23. Location
24. Long range planning
25. Maintenance
26. Management policy
27. Manufacturing methods
28. Market
29. Materials
30. Material handling
31. Offices
32. Organization
33. Packaging
34. Packing
35. Personnel
36. Pollution
37. Processes
38. Product
39. Production control
40. Quality control
41. Receiving
42. Refuse
43. Safety
44. Security
45. Services
46. Shipping
47. Site
48. Storage
49. Supervision
50. Throughput
51. Time frame
52. Transportation
53. Unions
54. Utilities
55. Warehousing
56. Waste
57. Work methods
58. Work standards
59. Yards
60. Zoning

Figure 1.1 Principal categories of factors for consideration in facility design

is achieved. Such methods may be suitable and effective for small problems with few departments or machines, but they are generally not applicable for moderate to large-size problems. Also, the use of manual methods may not be practical for generating and evaluating many possible layouts, even when the problem size is small. Most present computer methods are actually the implementation of manual methods on the computer. They can easily handle large problems and generate many different layouts but can only consider a very restricted, precisely defined set of objectives. Thus, they fail to effectively utilize the intuition of designers in layout creation and improvement. A need exists, therefore, to develop a methodology applicable to moderate and large-size problems that can better incorporate human creativity, and take maximum advantage of existing hardware technology.

1.2 OBJECTIVES OF THE RESEARCH

Recognizing the complexity and multiobjective nature of the layout problem, decision models for designing and evaluating facility layouts are developed in this research. The models are decision theoretic ones which utilize multiattribute decision theory in measuring both qualitative and quantitative layout factors in a systematic fashion. Such models are applied to the facility design problem in this research in achieving the following objectives:

1. Development of an appropriate theoretical foundation of decision theoretic models applicable to facility design.

2. Development of an optimizing layout creation technique that captures the objectives represented within decision theoretic models applicable to the facility design problem.

3. Development of a practical implementation technology for the optimal layout creation methodology and several heuristic variations of it.

1.3 ORGANIZATION OF THE DISSERTATION

This dissertation is organized into seven chapters. Chapter 2 describes some of the literature on layout design approaches which include computerized, multicriteria, graph theoretic and mathematical modeling approaches. The purpose of the review is to present the state of the art of existing computerized and analytical approaches to facility design. Also, in the second chapter, the literature on decision theory is surveyed. This provides the necessary background and describes the base from which new theoretical results are derived in this research effort. In Chapter 3, measurement considerations motivating the need for theoretical extensions are described. This includes demonstration of the insufficiency of present decision theory when applied to the facility layout problem. This chapter also describes a decision theoretic approach for the layout design problem. A decision support system for the general layout problem is described in Chapter 4. This includes a description of the man-machine interaction to solve the layout problem and the description of the software developed. In Chapter 5, an alternative axiomatic system defined on mixed attributes is presented. Theories based on the axiomatic system for additive, multilinear, and multiplicative forms are presented. In addition, the theory for an interdependent case is included. In Chapter 6, the methodology for finding relative department locations is described, and the layout problem is formulated as a quadratic assignment problem. A solution procedure which is a blend of common sense, an exact method, and pairwise interchanges of departments is presented. Chapter 6 also gives the final layout creation process and results of applying the decision support system on an example problem. Chapter 7 summarizes the research and suggests future research areas and possible improvements.

2. LITERATURE REVIEW

2.1 REVIEW OF LITERATURE ON FACILITY LAYOUT DESIGN APPROACHES

Numerous studies are devoted to facility layout, which involves complex decisions and a large amount of data, since it is one of the most crucial aspects of manufacturing. An organized approach to the problem developed by Muther (1961) has received considerable visibility in the literature and in practice due to the success derived from its application in solving a large variety of layout problems. The approach is referred to as Systematic Layout Planning or simply SLP. SLP has been applied to a variety of problems involving production, transportation, supporting services, and office activities. This approach provides guidelines in the analysis, search, and evaluation phases of the facility design process. The analysis phase involves understanding the facility layout problem, and determining the space and relationships between facilities. The search phase develops layout alternatives based on information from the analysis phase. Prior to the introduction of the computer and mathematical modeling approaches to aid in the problem of facility design, the development of layout alternatives was based on the general functional classification of a facility. The general functional classification of a facility refers to whether a facility can be classified as a job shop, flow shop, group technology shop, etc. Reed (1961) suggested three facility arrangements as follows:

1. Spiral method: The objective in using the spiral method is to arrange departments in such a manner that materials may flow directly from one department into the next department. The spiral method can be summarized in the following 9 steps (Malmborg and Sarin, 1984):

   a. Rank department pair flow volumes or adjacency importance of department pairs in descending order.

   b. Enter the department pair with the highest ranking into the layout.

   c. Determine the next highest ranking between a department pair and enter that department into the layout.

   d. Update the layout to maximize flow between adjacent departments without violating feasibility or other important constraints.

   e. Repeat steps (c) and (d) until all departments are included in the current solution.

f. Adjust the layout to reflect actual department areas.

g. Compute the inefficiency rating for the incumbent layout solution as: 100*flow between non adjacent departments/total flow.

h. Save the incumbent solution and generate alternative solutions repeating steps (a) through (g).

i. Select the best layout using the lowest inefficiency rating or other criteria.

2. Straight-line method: The objective is to reduce the total handling distance of the work piece. The method is to try to place departments in such a way that the flow is unidirectional from receiving to shipping departments.

3. Travel charting: This technique is for any plant arrangement when product characteristics do not allow the establishment of production lines. It is the basis of modern facility layout approaches when material handling cost is the major consideration. The travel charting technique can be summarized in the following 6 steps (Malmborg and Sarin, 1984):

a. Establish an initial layout.

b. Determine the rectilinear distance between centroids of department pairs.

c. Prepare the distance matrix.

d. Cross multiply the distance matrix with the appropriate version of from-to chart to obtain the travel chart and compute the volume-distance product of the layout.

e. Adjust the layout to reduce the volume distance product.

f. Select the layout with the minimum volume distance product or use the volume distance product in association with other criteria to select the preferred layout.

In later approaches, the fact that the amount of information from the analysis phase is too large to be manually handled was recognized and the computer was used to aid in developing layout alternatives. The following sections investigate the various approaches which were proposed to aid and solve facility layout design problems.

2.1.1 COMPUTERIZED APPROACHES

Since the early 1960s many computerized approaches to layout design have been proposed in the literature. Some of them use qualitative information and are designed to determine the relative location of departments which can minimize the cost of handling materials for a given production schedule. The first type of approach is generally used when material flow is the dominant factor in a layout problem. Examples of the approaches using quantitative information are CRAFT, SPACECRAFT, and PLANET. However, some of these use qualitative information, such as the relationships between departments. When quantitative information cannot be easily determined due to, for example, variability of production schedules, etc., the use of qualitative information based approaches is advisable (Lee and Moore, 1967).

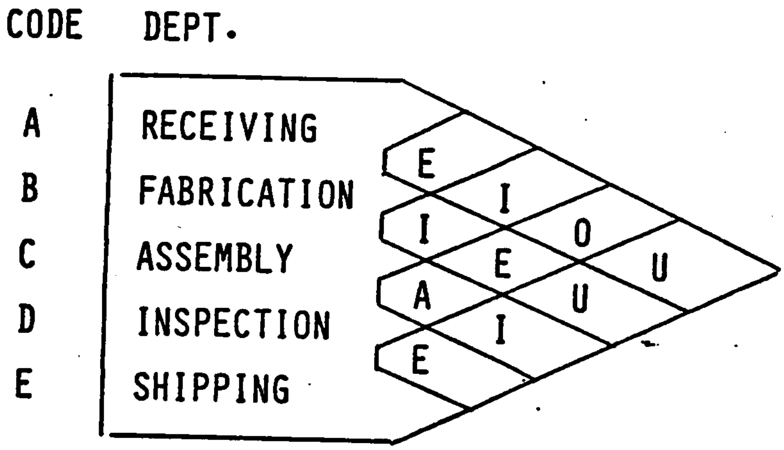
Aside from the classification of computer programs by the type of information used, computerized layouts are usually classified according to the way the final layout is produced. These groups are referred to as construction and improvement procedures (Moore, 1974). Construction procedures build up a layout from scratch, whereas improvement procedures require an initial layout which is then developed into a suboptimal solution (Francis and White,1974). In fact, this classification does not cover all computerized algorithms; for example, RUGR (Krejeirik,1969) is based on the mathematics of graph theory.

An example of a facility layout problem, as shown in Figure 2.1, will be used for demonstration purposes. The information for the problem includes department areas, REL chart, and from-to chart. The REL chart shows the degree of preference for any adjacent departmental pair, and A, E, I, O, and U represent the highest to the lowest preferences, respectively. The from-to chart shows the volume of material flow between any departmental pair.

The CRAFT methodology was first presented by Armour and Buffa (1963). It was later tested, refined and implemented by Buffa, Armour, and Vollmann (1964). The criteria employed in CRAFT is the minimization of the cost of item movement, where this cost is expressed as a linear function of the distance traveled. Since the criterion is one that is commonly used when the flow of material is a significant factor to be considered, CRAFT is referred to as a quantitative layout program. As such, it seeks an optimum design by making improvements in the layout in a sequential fashion. CRAFT first evaluates an initial (given) layout and then considers what the effect will be if the facilities' locations are interchanged. The objective function is the material handling cost which is the sum of the product between

DEPARTMENT	AREA (FT.^{2})	CODE
RECEIVING	4000	A
FABRICATION	8000	B
ASSEMBLY	6000	C
INSPECTION	6000	D
SHIPPING	4000	E

REL CHART:



FROM-TO CHART	(A) RECEIVING	(B) FABRICATION	(C) ASSEMBLY	(D) INSP.	(E) SHIPPING
(A) RECEIVING	-	140	80	20	0
(B) FABRICATION	0	-	80	100	0
(C) ASSEMBLY	0	0	-	160	80
(D) INSPECTION	0	40	80	-	160
(E) SHIPPING	0	0	0	0	-

Figure 2.1 Layout example problem

cost/unit distance and distance between departments for all facility pairs. The distances computed are the rectilinear distances between department centroids. CRAFT then considers exchanges of locations for those facilities which either are the same area or have a common border. The layout designer can have CRAFT consider 1) only pairwise interchanges, 2) only three way interchanges, 3) pairwise interchanges followed by three-way interchanges, 4) three-way interchanges, and 5) best of two-way and three-way interchanges. The heuristic of exchanging facilities continues until further improvement cannot be found. Later, the final layout is produced. Input data to CRAFT includes 1) initial spatial layout, 2) flow data, 3) cost data, and 4) number and location of fixed departments. Figure 2.2 depicts the flow chart of CRAFT. Figure 2.3 depicts CRAFT's layout creation process.

SPACECRAFT, developed by Johnson (1982), is another computerized method for allocating facilities within a multifloor building and retains CRAFT as a special case. Unlike CRAFT, where movement times are all assumed to be linear, SPACECRAFT allows the designer to specify characteristics of movement times. The improvement heuristic that SPACECRAFT and CRAFT utilize does not require linearity of costs, since it only compares and rank orders the costs. In addition to the data required by CRAFT, SPACECRAFT requires individual floor in-

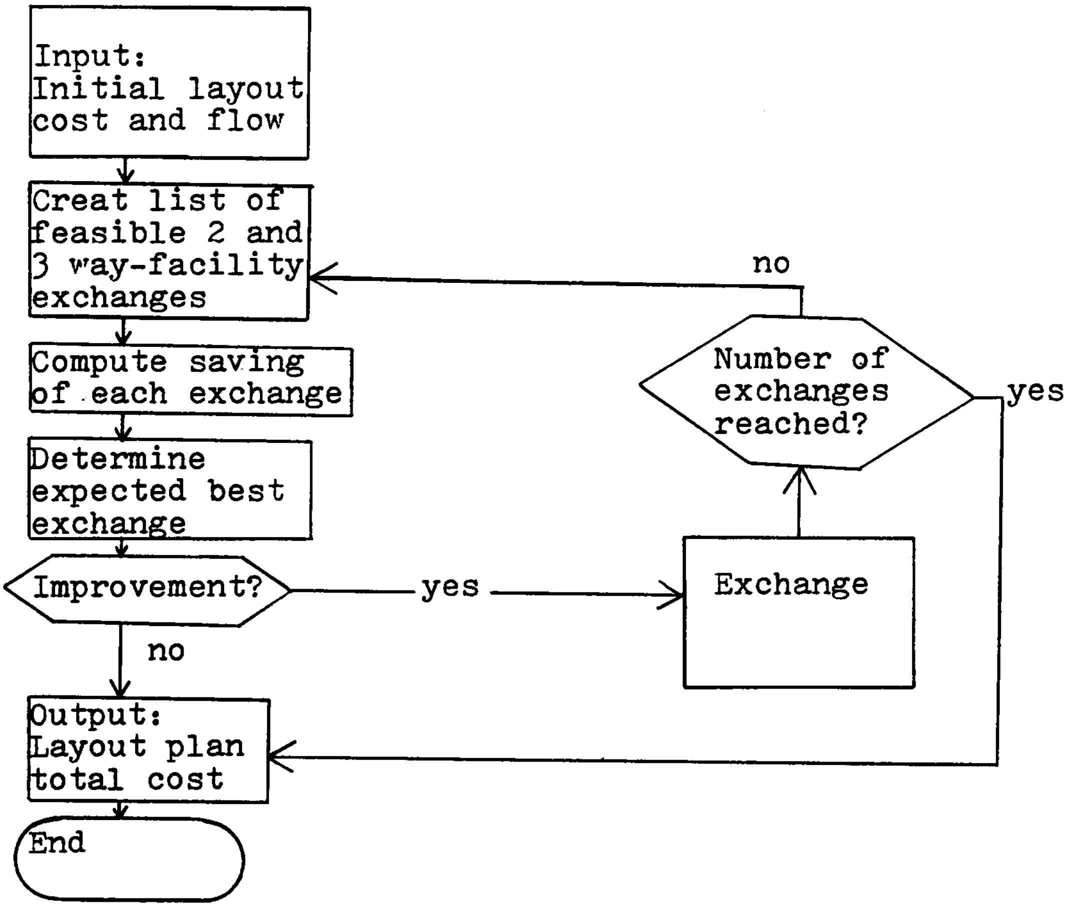
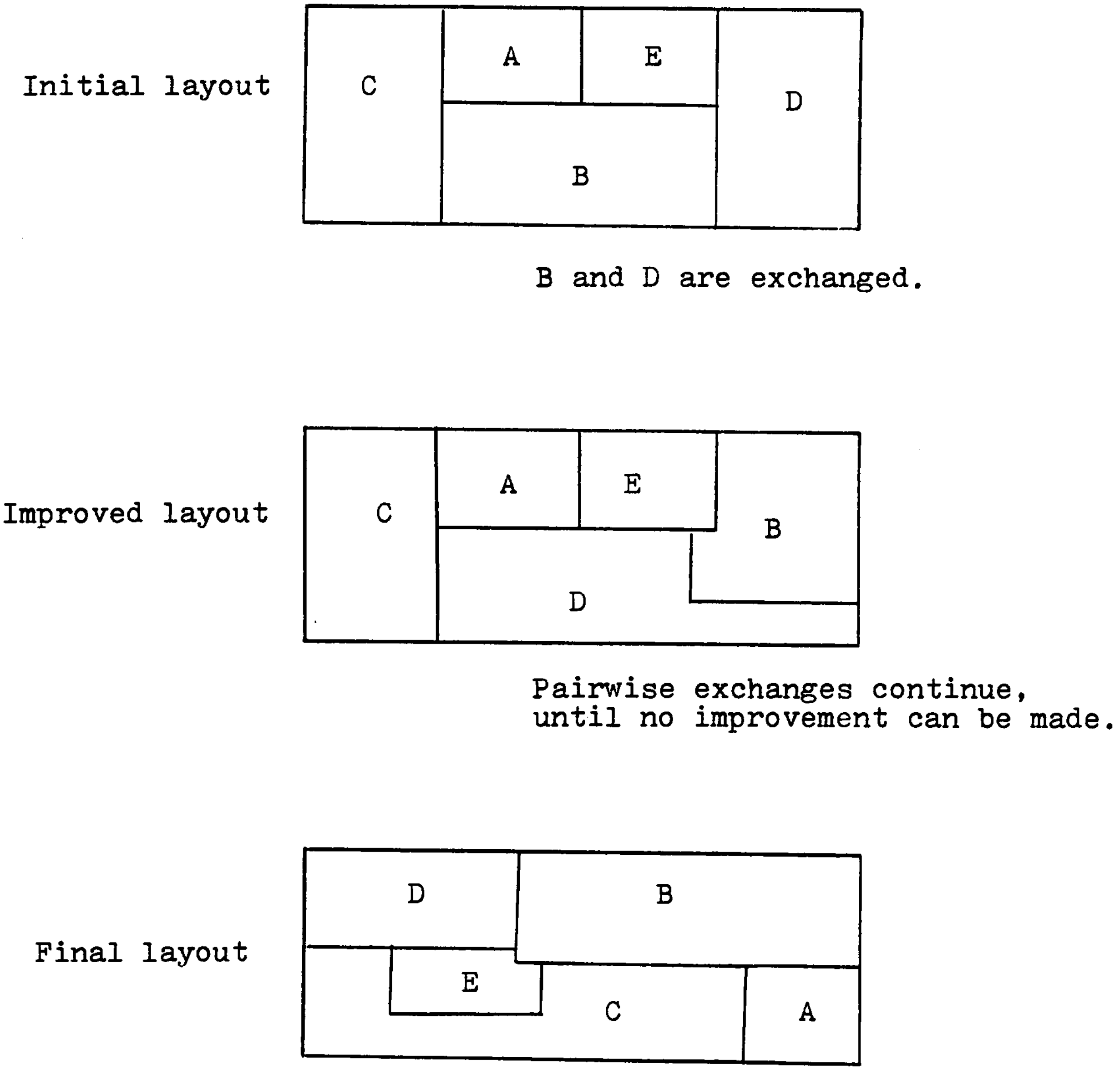


Figure 2.2 Flow chart of CRAFT



B and D are exchanged.

Pairwise exchanges continue, until no improvement can be made.

Figure 2.3 CRAFT's creation process

formation, cost of movement between floors, and specification of the movement times. SPACECRAFT is the only method which allocates departments to locations within a multifloor building and which considers nonlinear travel times.

PLANET, developed by Apple and Deisenroth (1980), is a computerized approach to facility layout design which utilizes material flow pattern information. There are two types of data used in the approach, facility information and material flow information. Each facility to be entered into the layout must be identified and the area requirements must be stated. Priority of each facility is also required for further use in the facility selection process. Unlike other computerized methods which require the from-to chart as input, PLANET utilizes an extended parts list instead. Extended parts list information includes part number, frequency of move, cost per move, and sequence of movement. These data are then translated into a format useful to the construction algorithm, specifically, a from-to chart which represents costs per unit distance. The objective function of the algorithm is to find a layout alternative with the lowest material handling cost possible. The material handling cost is the sum of the product of cost per unit distance times distance for all facility pairs where distances are rectilinear and are measured between facility centroids. The algorithm involves two phases, namely, selection and placement. The objective of the first phase is to select a facility to be a candidate for use in the placement phase. Three methods of selection are suggested:

Method A: First select a pair of facilities which has the highest cost per unit distance. The next facility is chosen from the unplaced list where the cost per unit distance between unplaced and placed facilities is the highest.

Method B: Identical to A for the first pair. The remaining selections are made by considering the relationships between each of the unplaced list and all facilities in the placed list. This is accomplished by letting the value in the placed list column be zero and by adding all the values for each unplaced row of the unplaced list. The facility with the highest sum is selected next.

Method C: Adding each row. Facilities are then ranked and selected according to their sum values.

During the second phase (placement), the selected facility is placed within the existing facilities. To find the best location, the selected facility is moved around the perimeter of the others until the location with minimum change in material handling cost is found. The flow chart of PLANET is depicted in Figure 2.4. PLANET's layout creation process is depicted in Figure 2.5.

CRAFT, one of the earliest and best improvement algorithms, has been reworked and implemented in new programs with the acronym PLOP2, PLOP3, PLOP4 (PLOP stands for Plant Layout Optimization Procedure). PLOP2 improves an initial layout by first calculating the change in total cost caused by exchanging the locations of all pairs of departments. It then exchanges the location of the pair of departments that gives the greatest reduction in total cost. It continues with this procedure until no further reduction can be made in the total cost. The general procedure on which PLOP2 is based is extended to three-way and four-way exchanges of facilities, which give rise to algorithms PLOP3 and PLOP4 (Lewis and Block,1980). Figure 2.6 depicts the flow chart of the PLOP algorithms.

QUAINT (Quadratic Assignment Implicit N-level Technique) is a modification of the PLOP series to reduce computing time while still maintaining the ability to consider swaps at the third and fourth exchange levels. QUAINT3 initially follows the same path as PLOP2, but when no two-way swap can give a further reduction in the total cost, PLOP3 is used to find a possible three-way swap. When a three-way swap is found, the program then reverts to PLOP2. With this procedure, QUAINT3

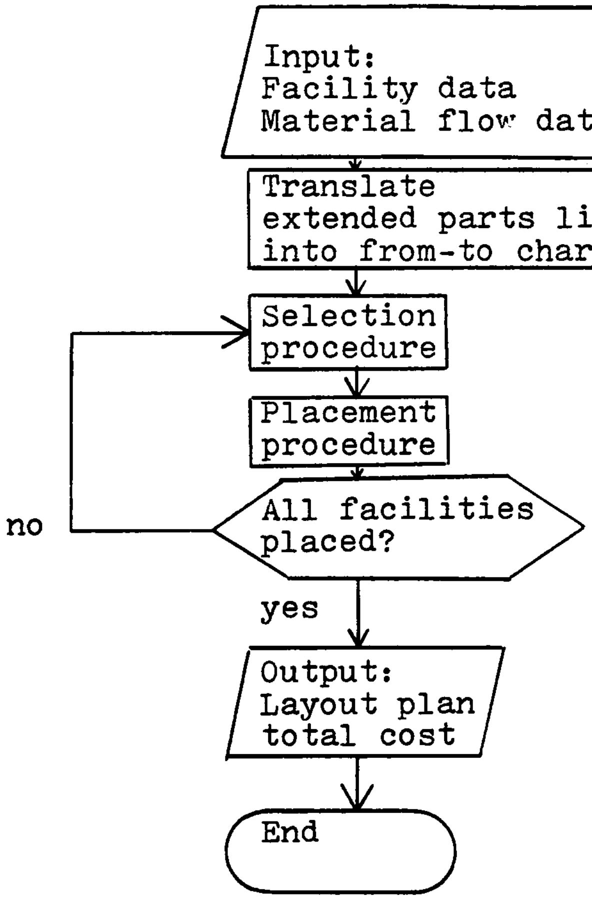


Figure 2.4 Flow chart of PLANET

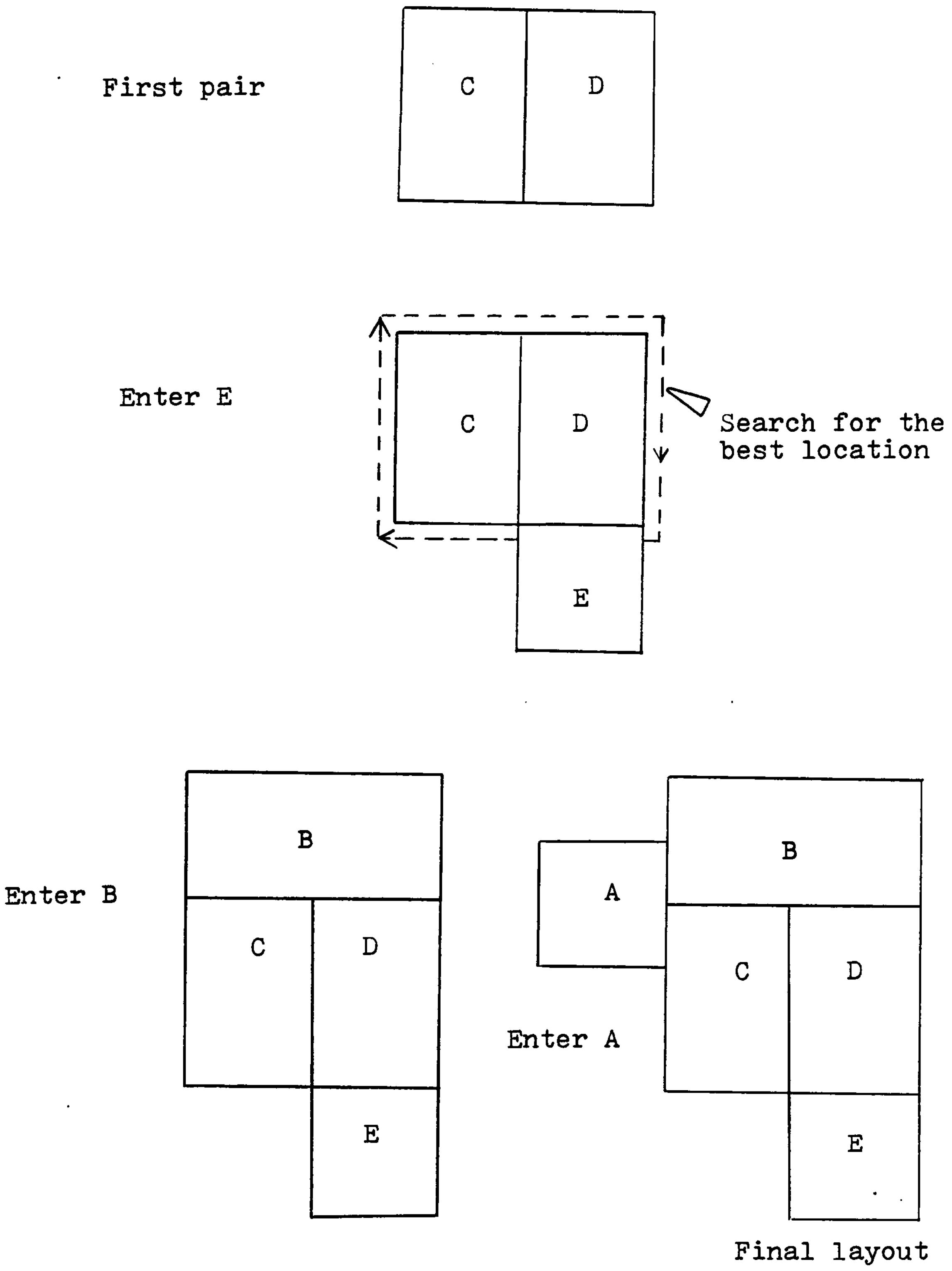
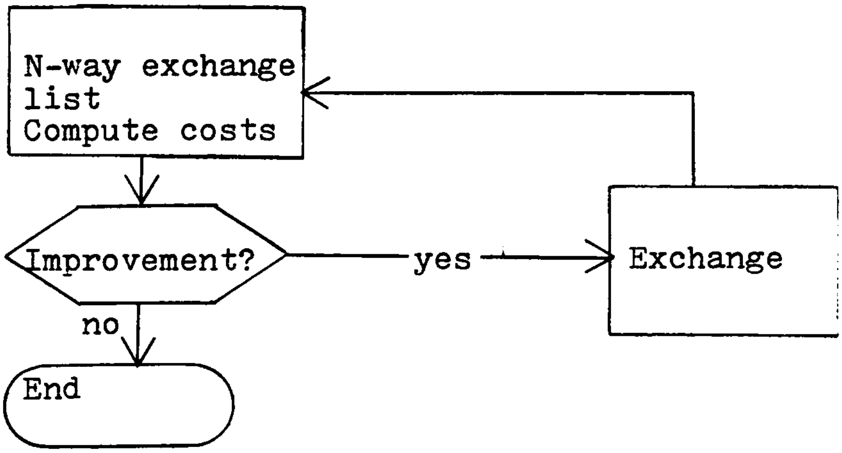
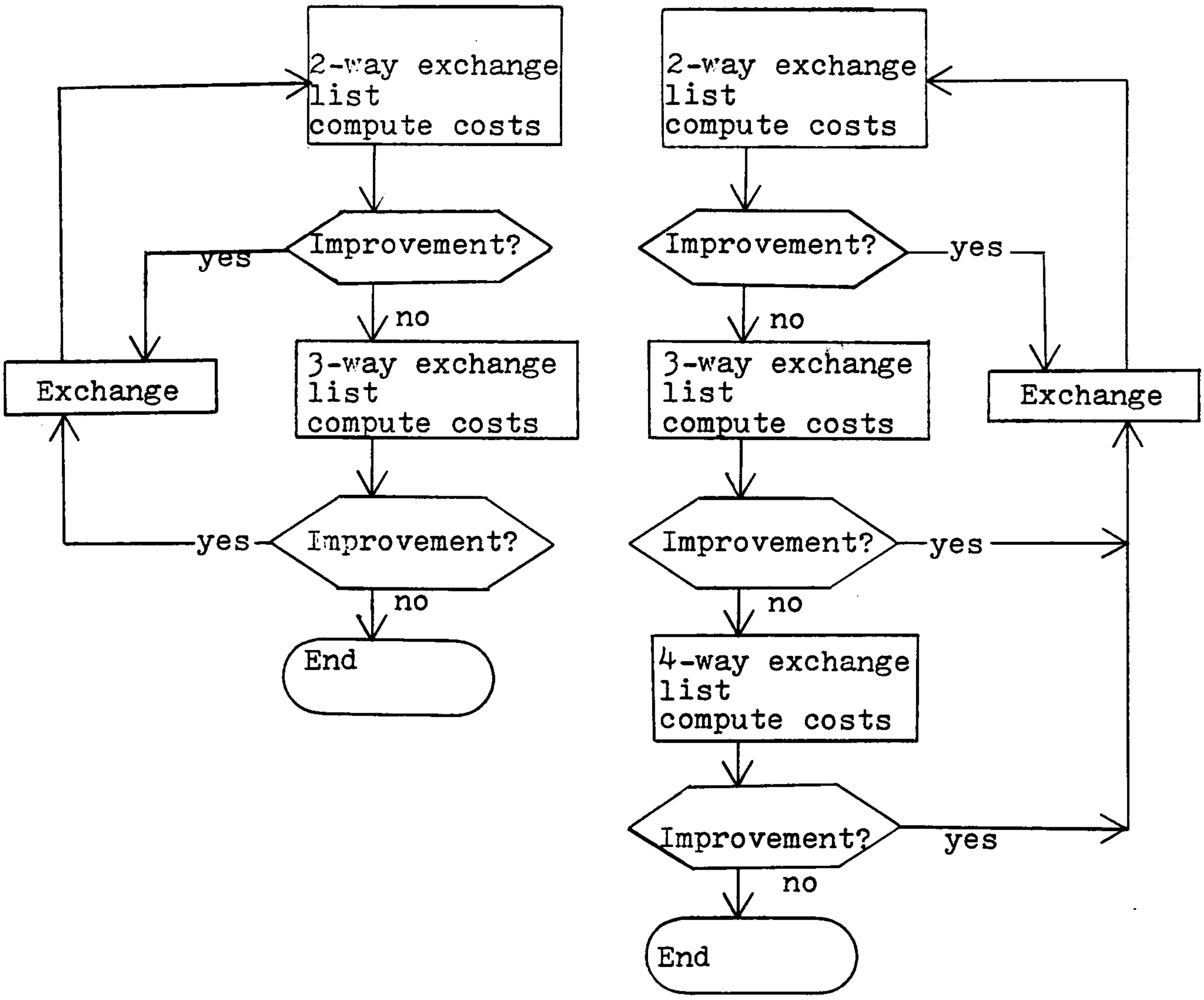


Figure 2.5 PLANET's layout creation process



N - 2,3,4 for PLOP 2,3,4
respectively

Figure 2.6 Flow chart of PLOP


2-way exchange list compute costs
Improvement?
yes
no
Exchange
3-way exchange list compute costs
Improvement?
yes
no
End
2-way exchange list compute costs
Improvement?
yes
no
3-way exchange list compute costs
Exchange
Improvement?
yes
no
4-way exchange list compute costs
Improvement?
yes
no
End

QUAINT 3

QUAINT 4

Figure 2.7 Flow Chart of QUAINT

has the advantage of being able to make three-way swaps without the disadvantage of increased computational time. The flow chart of QUAINT is depicted in Figure 2.7.

In 1980, O'Brien and Barr (1980) developed a program for layout design problems which utilizes a sophisticated graphics capability. The program is divided into two main procedures, construction and improvement. The first procedure (INLAYT) allows a layout to be constructed for situations such as a new factory or extension, or a major relayout of facilities. The inputs to this phase include the following:

1. The number of facilities to be considered,

2. The spatial array which consists of equal blocks (i.e., if 24 facilities are to be located, the designer must specify whether he wishes initially to consider a 6*4, 8*3, or 12*2 arrangement),

3. The volume of flow of material traveling between facilities, and

4. A flow factor. (This is a designer-specified variable between 0 and 1 to define the level of weighted flow between facilities which he considers significant.)

Unlike PLANET which requires facility areas, INLAYT does not require such information because it only assigns facilities in a spatial array.

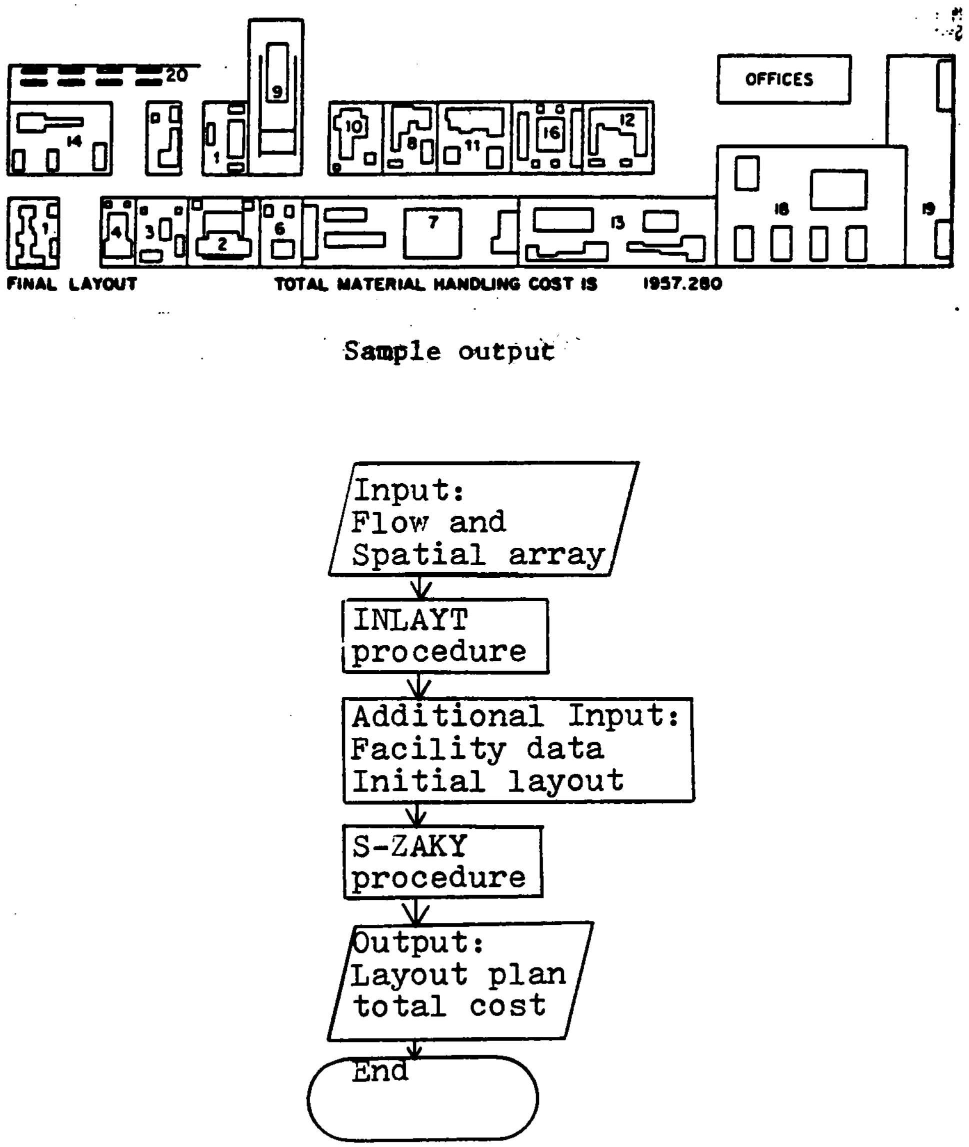
The construction procedure starts by ordering the facilities according to the sum of weighted flow in and out of each facility. Each facility is then ranked in order, and is grouped with those facilities for which the weighted flow of material equals or exceeds the maximum value of weighted flow found in the weighted flow matrix between any two facilities, multiplied by the flow factor. The information is presented on the graphic screen with the spatial array. The designer can use the light-pen to assign facilities to any location he wishes, based on information presented, and his judgment. The procedure ends when all facilities have been assigned to locations.

The second phase, the improvement procedure (S-ZAKY), begins from an initial layout which can be an existing one, or that developed using INLAYT or any other methods. The basic data required are: 1) the facility specifications including names and areas, 2) the relative locations of departments, 3) the volume of flow of material between facilities, 4) the detail of departments with all machinery and other components, and 5) the position of set-down and pick-up points.

Like CRAFT, the algorithm considers the savings in material handling costs which would result from interchanging the locations of pairs of facilities, but at each iteration, the improvement to be obtained from interchanging not one, but three pairs of facilities, is made. The interchange of three pairs was adopted after considerable investigation as representing the best compromise between the improvements obtained and the computing time required. The changes in location of facilities accepted at each iteration provide the modified layout to be considered by the following iteration. The algorithm continues until it reaches its suboptimum solution in which no further improvements from interchanging facilities can be found.

Finally, the computer produces a detail layout plan, unlike other computer programs which produce only relative locations of facilities. Figure 2.8 shows a sample of output obtained from the INLAYT and S-ZAKY, and the macro flow chart of the program.

Since all of these computerized approaches utilize quantitative information for their input, the objective is to find a layout with the lowest total material handling cost. We next consider computerized approaches utilizing qualitative information.



Sample output

Flow chart

Figure 2.8 Output from INLAYT and S-ZAKY, Flow chart

CORELAP, presented by Lee and Moore (1967), is a computerized construction program based on facility relationships. The input requirement for CORELAP includes: 1) relationship chart, 2) number of facilities, 3) area of each facility, and 4) weights for the relationship chart entries. The objective of CORELAP is to produce a layout which satisfies most relationships. This is measured by the total closeness rating which is defined as

$$TPR = \sum_{i=1}^{n} \sum_{j=1}^{n} P_{ij} a_{ij}$$

where

P_{ij} = closeness rating for facility i and j

a_{ij} = 1 if facility i and j are adjacent

= 0 otherwise

Although the weights for the REL chart are arbitrarily defined, CORELAP starts by selecting a facility which is first placed in the layout. Next, a facility is selected and placed in the layout where the highest closeness rating is achieved with the department being placed. The process continues until all facilities have been placed in the layout. The process of facility design, including facility selection and placement, is heuristic and does not guarantee the optimal total closeness rating. Later, new versions of CORELAP were developed, including CORELAP8 and Interactive CORELAP.

CORELAP8 uses the same concept as the original version, but provides useful features such as the length-to-width ratio of the final layout. On the other hand, Interactive CORELAP provides flexibility in rearranging a layout. Figure 2.9 depicts the flow chart of CORELAP. Figure 2.10 depicts the flow chart of the facility selection rule used in CORELAP. Figure 2.11 depicts CORELAP's layout creation process.

ALDEP is a construction and improvement program developed by Seehof and Evans (Seehof and Evans,1967). Although it uses data similar to CORELAP, its construction procedure is different from CORELAP. ALDEP develops a layout design by randomly selecting a facility and placing it in the layout. Next, the relationship chart is scanned, and a facility with the highest closeness rating is placed in the layout. The ALDEP placement procedure is designed to avoid extreme irregularity in the shape of borders of the layout periphery, by using the vertical scan method of placing facilities. Basically, the layout area is filled by using vertical strips having a specified width and length equal to the depth of the layout. Figure 2.12 depicts ALDEP's layout creation process. This process is continued until all facilities are placed. The improvement proceeds by repeating the entire process and comparing the resulting score with the existing score. Inputs to ALDEP are similar to that for CORELAP. In addition, the user specifies the number of layouts to be generated.

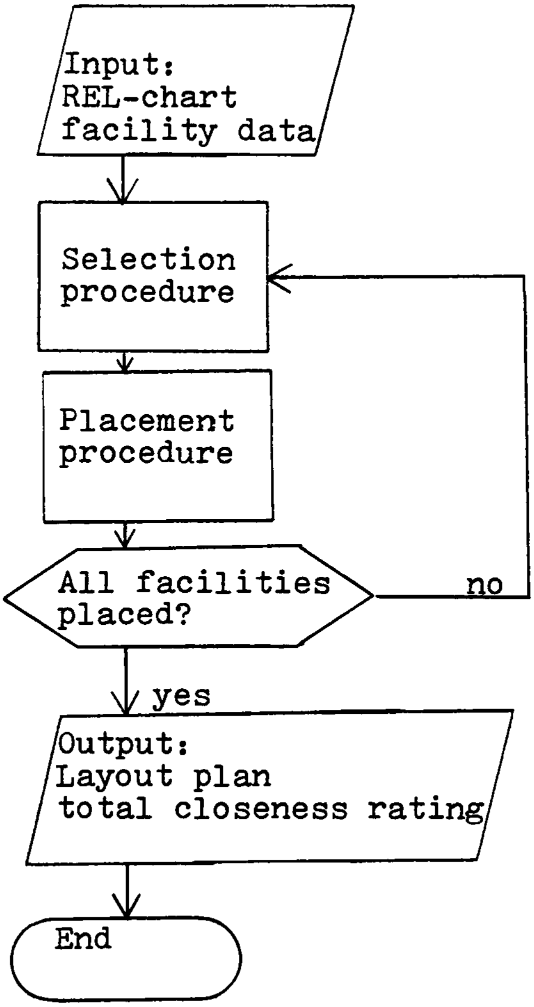


Figure 2.9 Flow chart of CORELAP

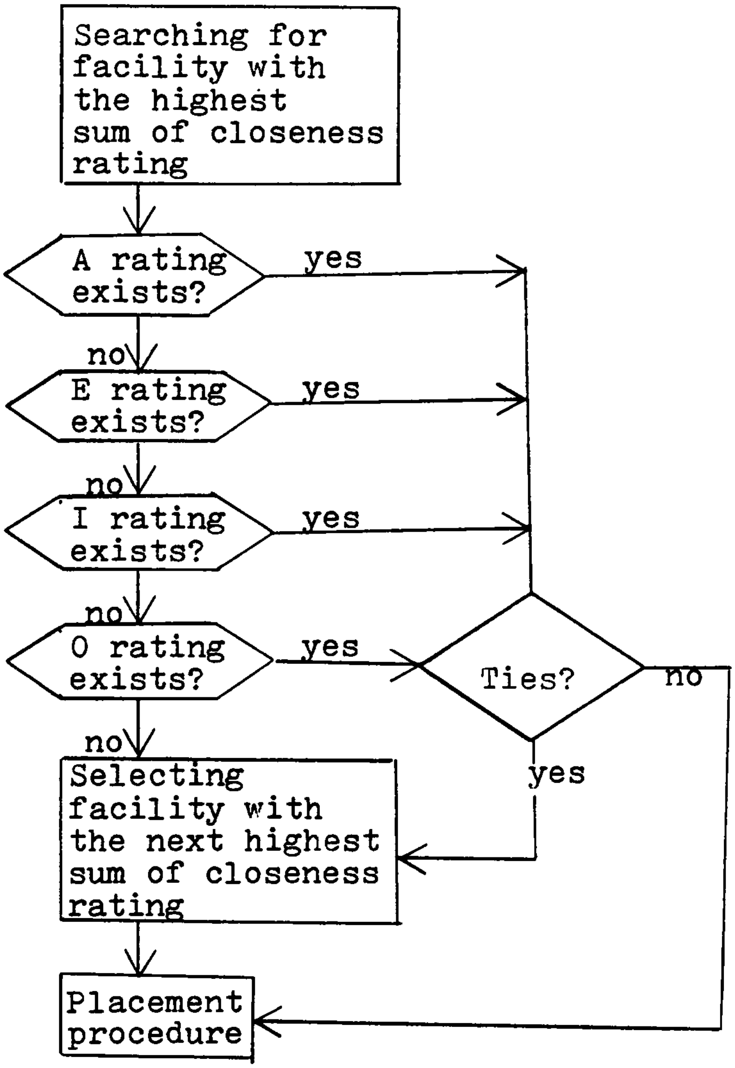

Searching for facility with the highest sum of closeness rating
A rating exists?
yes
no
E rating exists?
yes
no
I rating exists?
yes
no
O rating exists?
yes
Ties?
no
yes
no
Selecting facility with the next highest sum of closeness rating
Placement procedure

Figure 2.10 Flow chart of CORELAP's Selection Rule

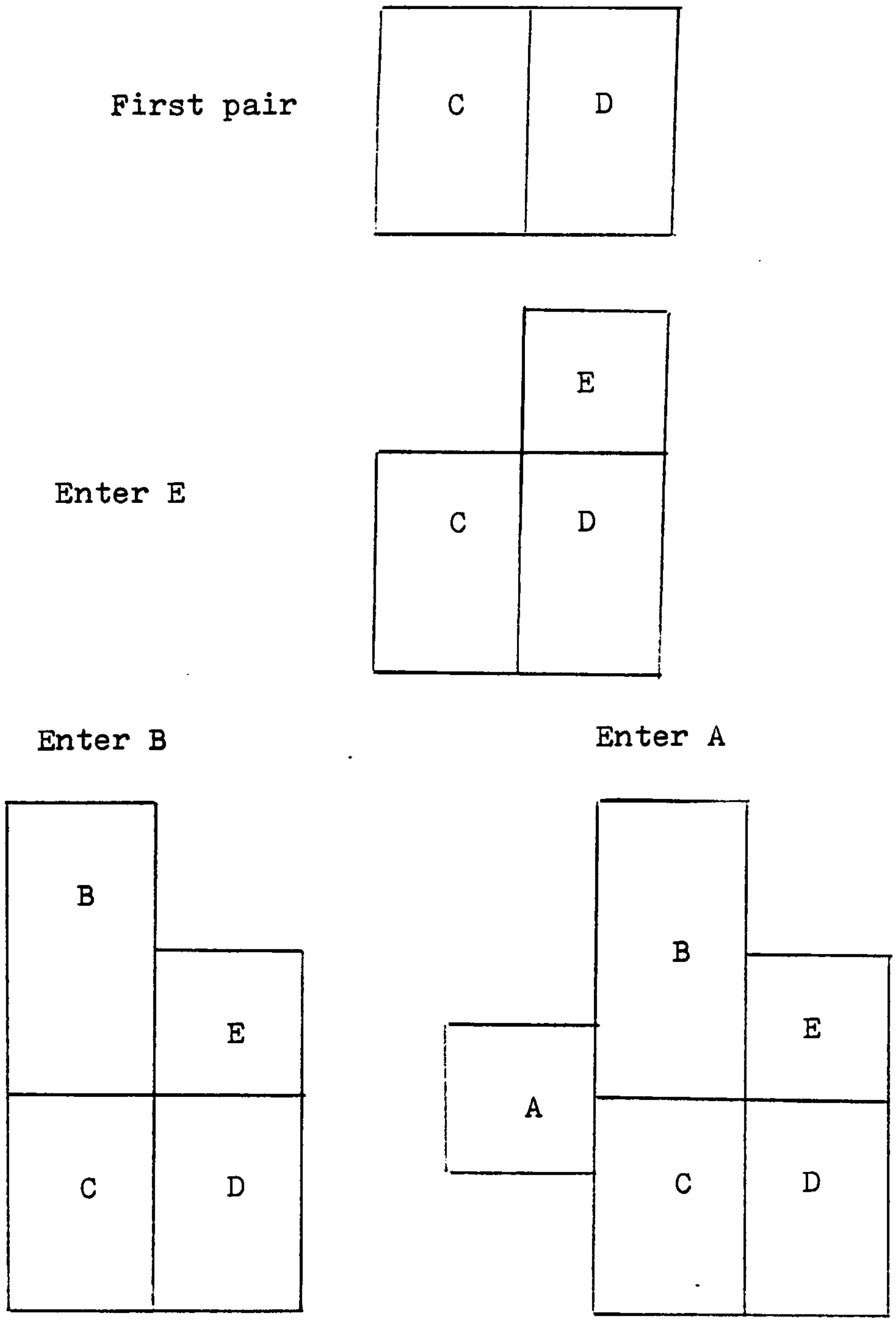


Figure 2.11 CORELAP's layout creation process

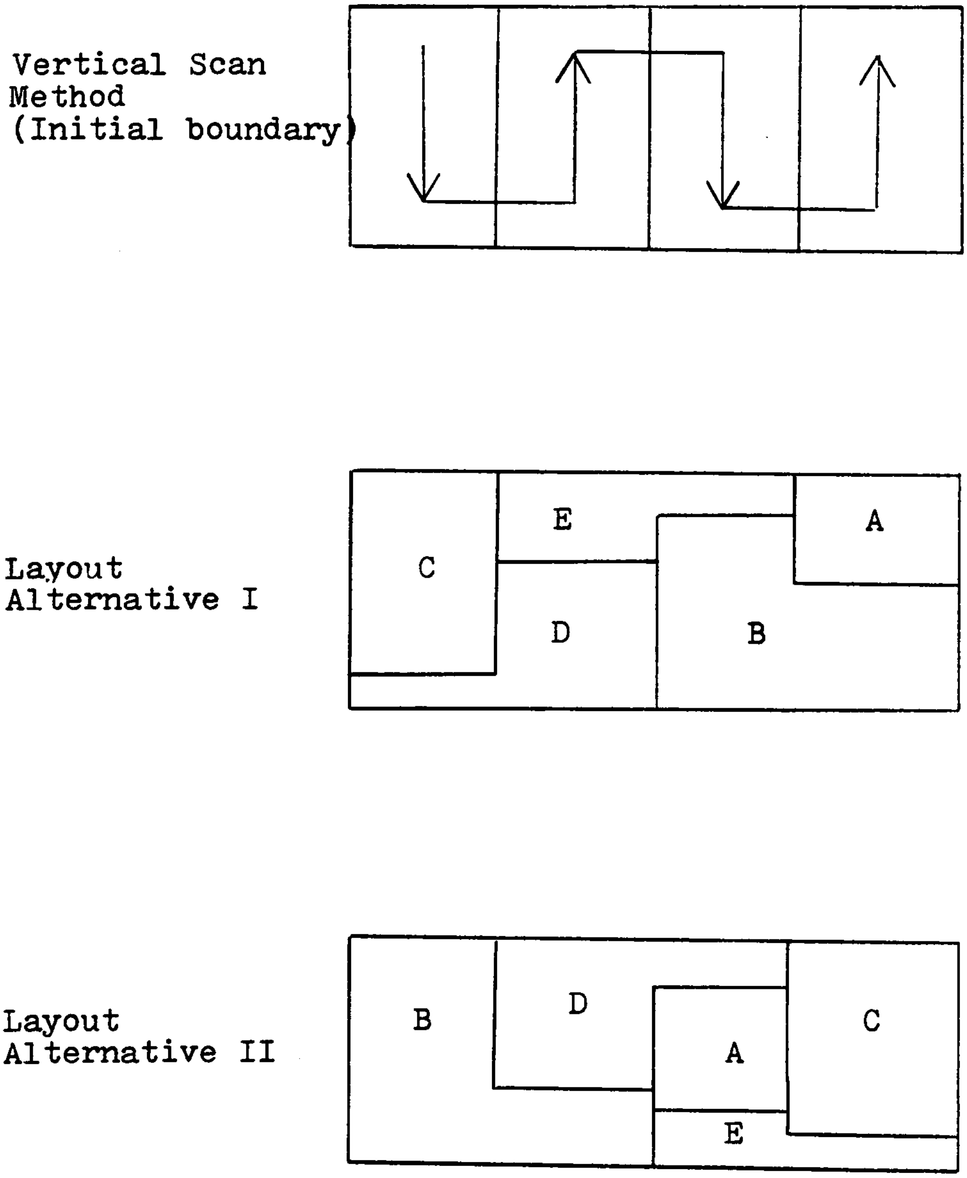


Figure 2.12 ALDEP's layout creation process

ALDEP will generate multiple layouts and select the one with the highest total closeness rating.

RMA Comp I is another approach which utilizes the relationship chart. Developed by the staff of Richard Muther and Associates (Muther,1970), it uses closeness relationships as input data in a similar manner as ALDEP and CORELAP do. The logic of selecting facilities is also similar to that used in CORELAP. RMA Comp I is designed to do by computer what Systematic Layout Planning (SLP) does manually. This program recognizes that further manual adjustment of the output is necessary in order to develop a practical, workable layout, so it does not try to make a printout showing facilities adjacent to each other as CORELAP and ALDEP do. Its printout is a space relationship diagram, which attempts to maximize all closeness rating relationships as shown in Figure 2.13.

The approaches described above were developed primarily by academicians. The information related to such approaches was available through academic journals as cited. Since the late 1970s, computer software specially designed for facilities planning and design began to be available commercially. This was largely due to the availability of computers with graphics capabilities. Filley (1985) categorizes the software based on such technologies as follows: 1) decision support systems (DSS), 2) computer-aided design (CAD) and 3) manage-

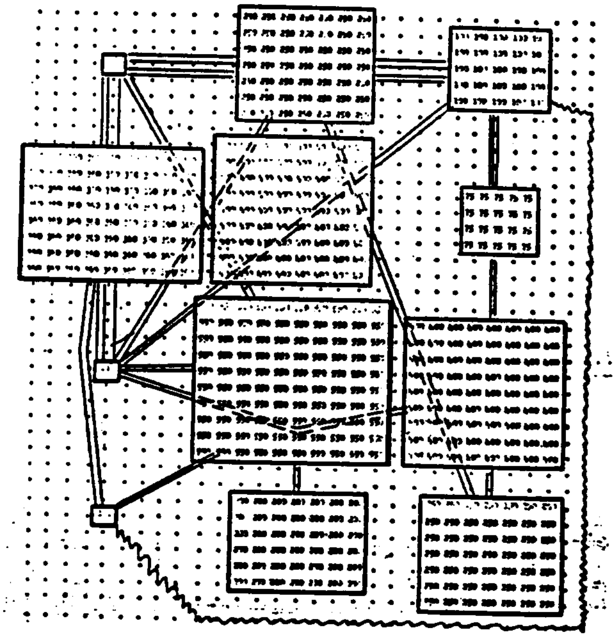


Figure2.13 Sample output from RMA Comp I

ment information systems (MIS). Each of these technologies generally serves a corresponding facilities function, as follows:

1. DSS-facilities planning. DSS helps make information more usable. DSS can be applied to issues such as whether to refurnish or build a new facility, or whether to own or lease a facility.

2. CAD-facility design. Color and 3-D graphics are now commonly found in any computer system. Thus, real time graphic simulations of robotics, material handling systems, warehousing and production systems can be modeled. This is the objective of most CAD systems.

3. MIS-facilities management. MIS is typically used to produce regular reports on assets or facilities utilization.

2.1.2 MULTICRITERIA APPROACHES

Multicriteria approaches refer to those where both qualitative and quantitative criteria are simultaneously considered. Rosenblatt (Rosenblatt,1979) presented a combined quantitative and qualitative approach to the plant layout design problem. The two objectives, quantitative and qualitative, which may be conflicting, are to minimize the material handling cost and maximize a closeness rating. However, these may also be combined into one objective function.

The minimization of material handling cost is formulated as follows:

$$\text{Min } C = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} a_{ijkl} X_{ij} X_{kl}$$

Subject to

$$\sum_{i=1}^{n} X_{ij} = 1 \qquad j=1,\ldots,n$$

$$\sum_{j=1}^{n} X_{ij} = 1 \qquad i=1,\ldots,n$$

where

X_{ij} = 1 if facility i is assigned to location j

= 0 otherwise

$a_{ijkl} = f_{ij} d_{kl}$ if i=k or j=l

$= f_{ii} d_{jj} + c_{ij}$ if i=k and j=l

c_{ij} = cost per unit time associated directly with facility i to location j

d_{jl} = distance from location j to location l

f_{jk} = work flow from facility i to facility k

The maximization of closeness rating can be formulated as follows:

$$\text{Max } R = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} w_{ijkl} X_{ij} X_{kl}$$

Subject to

$$\sum_{i=1}^{n} X_{ij} = 1 \qquad j=1,\dots,n$$

$$\sum_{j=1}^{n} X_{ij} = 1 \qquad i=1,\dots,n$$

$$X_{ij} = 0 \text{ or } 1 \quad \text{for any } i \text{ and } j$$

where

$w_{ijkl} = r_{ik}$ if location j and l are adjacent
= 0 otherwise

r_{ik} = closeness rating of facility i and facility k

Thus, a multi-objective formulation is

$$\text{Min } Z = b_1 C - b_2 R$$

Subject to

$$\sum_{i=1}^{n} X_{ij} = 1 \qquad j=1,\dots,n$$

$$\sum_{j=1}^{n} X_{ij} = 1 \qquad i=1,...,n$$

$$X_{ij} = 0 \text{ or } 1 \quad \text{for any } i \text{ and } j$$

$$b_1 + b_2 = 1 \text{ and } b_1, b_2 \geq 0$$

where b_1 and b_2 are weights assigned to the total cost flow and total rating score, respectively.

When an alternative is created, the total material handling cost, C, and the total closeness rating, R, can be calculated. These two values are combined through the multi-objective function. This results in a score which represents the degree of satisfaction of the layout alternative. This approach allows a designer to systematically choose the best layout alternative based on both quantitative and qualitative information.

Dutta and Sahu (Dutta and Sahu,1982) also consider the multiobjective problem. They developed a heuristic procedure for solving the previously described problem. The procedure takes an initial layout and improves it using a pairwise exchange routine.

2.1.3 GRAPH THEORETIC APPROACHES

Graph theoretic approaches refer to approaches using graph theory in optimizing closeness ratings of adjacent facilities. The Graph theory approach was first introduced by Seppanen and Moore (Seppanen and Moore,1970) to solve layout design problems. It attempts to allocate facilities in the layout so as to give an optimal total closeness rating. First, the relationship chart is converted into a graph which is later categorized as planar or nonplanar. The planar graph can be converted into a block plan layout which gives an optimal total closeness rating.

Seppanen and Moore (Seppanen and Moore,1975) realized the difficulties in recognizing the planarity of graphs. Thus, they introduced a linear string representation and the concept of grammar as basic tools for symbolizing trees and planar graphs. Algorithms based on graph theory to maximize closeness ratings as well as to convert planar graphs into block diagrams were suggested.

2.1.4 MATHEMATICAL MODELING APPROACHES

Mathematical modeling approaches refer to approaches where quantitative considerations of a facility layout problem are used to construct mathematical models and to represent quantitative characteristics of the problem. Francis and White summarized the various mathematical modeling approaches to the layout design problem (Francis and White, 1974). These models have the objective of minimizing material handling cost which is the sum of the product of distance and cost for all facility pairs. Generally, the layout design problem is to assign facilities to locations and minimize total cost. This problem can be formulated as;

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{h=1}^{n} c_{ijkh} X_{ik} X_{jh}$$

Subject to

$$\sum_{i=1}^{n} X_{ik} = 1 \qquad k=1,\ldots,n$$

$$\sum_{k=1}^{n} X_{ik} = 1 \qquad i=1,\ldots,n$$

$$X_{ik} = 0 \text{ or } 1 \qquad \text{for any } i \text{ and } k$$

where

c_{ikjh} = cost of having facility i located at location k and facility j located at location h

X_{ik} = 1 ,facility i is located at location k

= 0 ,otherwise

This formulation is a quadratic assignment problem to which there are no efficient exact methods which can solve realistically-sized problems.

Consequently, Bazaraa (1975) considered the layout design problem and formulated the problem as a quadratic set covering problem. His formulation can handle single or multistory buildings, facilities with regular or irregular shapes, designing a layout from scratch, or adding new facilities to an existing layout. The solution is provided by a branch and bound optimization procedure. Figure 2.14 shows Bazaraa's modeling approach to the layout problem.
Chapter 6 will discuss the quadratic assignment problem in detail.

DISCON is a new approach to the layout problem developed by Drezner (Drezner,1980). Its objective is to find a layout with least total cost. However, the DISCON procedure assumes that facilities have circular shapes, and that distance between facilities is measured from center to center by euclidean distance, instead of rectangular distance as in other traditional layout design programs. The DISCON algorithm includes two phases, dispersion and concentration. The dispersion phase has the circles (facilities) disperse from the origin, as the whole system expands. The purpose of the dispersion phase is to find good initial conditions. The final solution in this phase gives good starting points for the second phase which is the concentration phase. At the end of the first phase, the areas are not touching. In the second phase, this formation is concentrated or tightened.

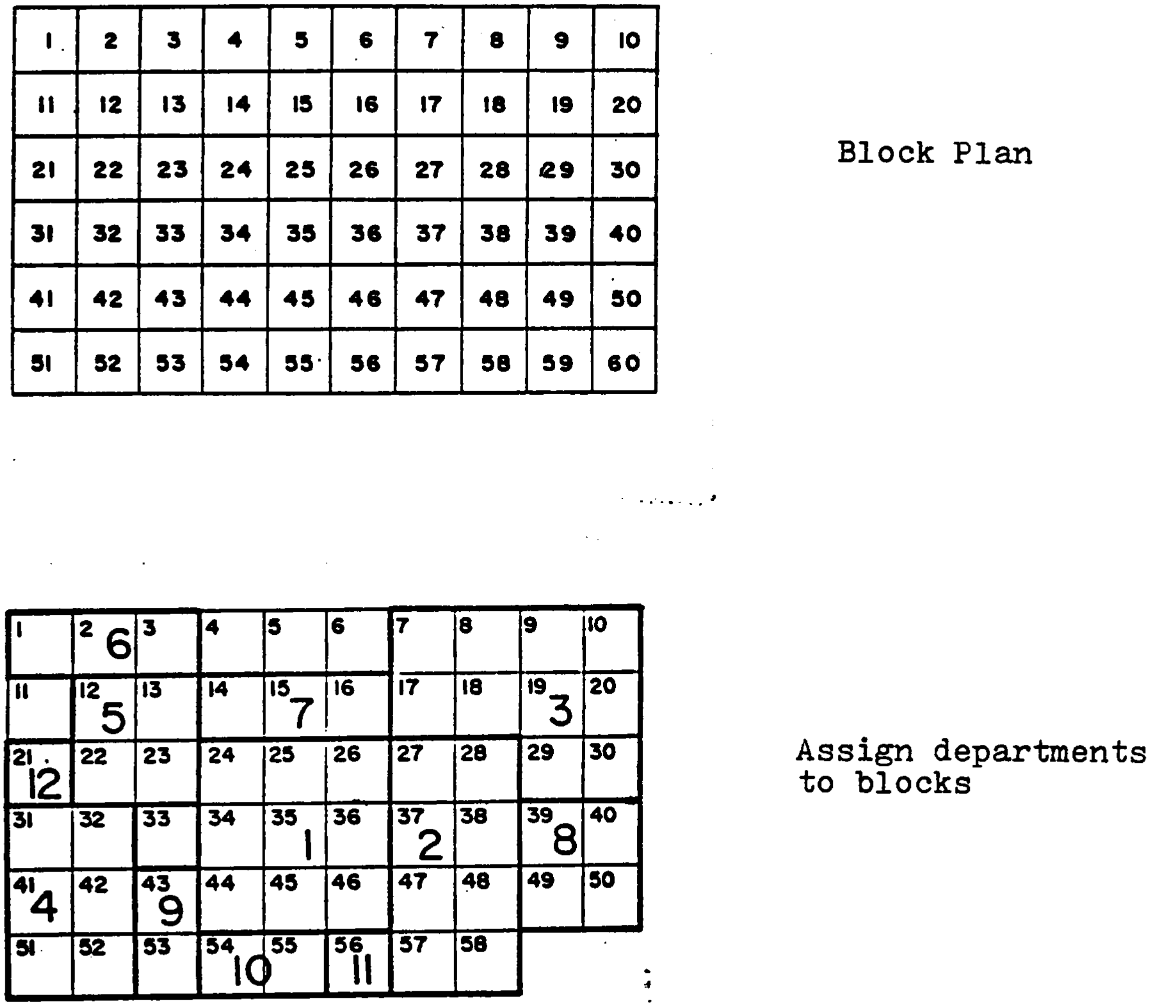
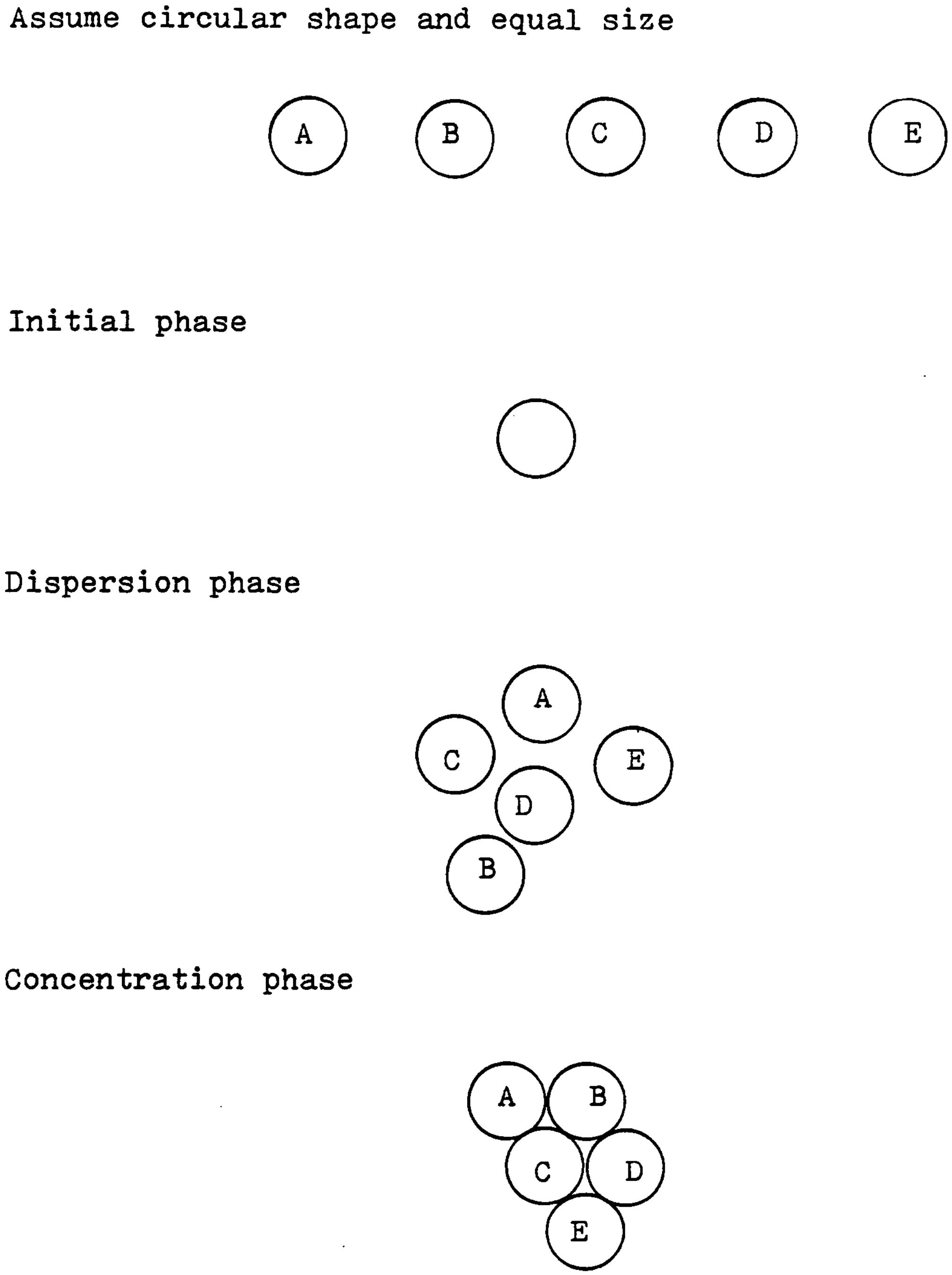


Figure 2.14 Bazaraa's modelling approach to the layout problem



Assume circular shape and equal size

Initial phase

Dispersion phase

Concentration phase

Figure 2.15 **DISCON's process**

The algorithms in these two phases are based on the Lagrangian Differential Gradient method suggested by Drezner (1980). Despite the strange assumptions, Drezner successfully used DISCON in an office layout design, where the assumptions of circle-shaped facilities and euclidean distance were acceptable (Drezner,1980). Figure 2.15 depicts DISCON's process.

2.1.5 SUMMARY AND CONCLUSION

The section 2.1 has included the literature review in the area of facility layout design. Four different kinds of approaches to the layout problem have been surveyed. The computerized approaches are designed to solve the layout problem with the objective of either minimizing material handling cost or maximizing the total closeness rating. Heuristic procedures used in the layout creation process were suggested to optimize predefined objective functions. The preferred heuristic procedures are typically determined by the measure of their computing time. The better computerized softwares are usually determined by graphics capabilities and user friendliness. Graph theoretic approaches are the basis of computerized heuristic procedures. Mathematical modeling approaches are still in the developing stages. The problem with mathematical modeling approaches is that there is no efficient exact methods which can solve realistically-sized problems. Multicriteria approaches are attempts to include both qualitative and quantitative criteria in the same objective function. The multicriteria approaches fail to standardize the scales of both criteria. Thus, the objective function may not consistently represent the designer's preference. The next section surveys decision theory that is useful in modeling the multicriteria facility layout problem.

2.2 LITERATURE REVIEW OF DECISION THEORY

One of the most common cases in the evaluation of alternatives is for entities to be composed of two or more components, each of which affects the evaluation in question. For example, a decision maker wants to choose which car to buy. The decision must be based on two components (attributes), such as cost and performance. The attributes in this example cannot readily be combined. However, there exist theories leading to the construction of measurement scales for composite objects which preserve their observed order with respect to the relevant attributes (e.g. preference, cost, and performance). The theories that lead to simultaneous measurement of the alternatives and their components are referred to as conjoint measurement theory (Krantz et. al., 1971), multiattribute value theory (Keeney and Raiffa, 1976), multiattribute utility theory (Keeney and Raiffa, 1976), topology theory (Fishburn,1970), and measurable multiattribute value theory (Dyer and Sarin,1979).

Generally, there are three forms of the aggregation function used to measure alternatives: 1) additive; 2) multilinear; and 3) multiplicative. These forms represent the decision maker's preference. There are more sophisticated forms, such as the polynomial (mostly applied in physics, Krantz et. al., 1971), but we restrict ourselves to these three. In most decision making situations, additive representations have proven to be the most successful in representing a decision maker's behavior due to their robustness (Dawes and Corrigan,1974; Einhorn and Hogart,1975).

2.2.1 MULTIATTRIBUTE VALUE THEORY

Krantz et. al. (1971) presented a classic theorem for the case of additive multiattribute value models. Before describing this theorem, we need to introduce the attribute space $X_1, \ldots, X_n$ which describes alternatives and follow it with some definitions of assumptions underlying the theorem:

1. Weak ordering: All the alternatives can be rank ordered.

2. Independence: $\prod_{i \varepsilon N} X_i$, $N=\{1,\ldots,n\}$ is independent iff, $\prod_{i \varepsilon N} X_i$, for fixed choices of X_i designated as a_i, $i \varepsilon N-M$ for some $M \subset N$, is unaffected by those choices.

3. Restricted solvability: $\prod_{i \varepsilon N} X_i$ satisfies restricted solvability iff, for each $i \varepsilon N$, whenever

   $b_1 \ldots \bar{b}_i \ldots b_n \geq a_1 \ldots a_i \ldots a_n \geq b_1 \ldots \underline{b}_i \ldots b_n$,

   then there exists $b_i \varepsilon X_i$ such that

   $b_1 \ldots b_i \ldots b_n \approx a_1 \ldots a_i \ldots a_n$

4. Archimedean property: Incremental variation in the level of each individual attribute is measurable in terms of measurable incremental variations in the level of other attributes.

5. Essentialism: The states of each attribute must exercise some influence on the preference for alternatives.

Any $\prod_{i \varepsilon N} X_i$, $N \geq 3$, is called an additive conjoint structure if all the five conditions hold. A major result from the theorem of Krantz is that if $X_1,\ldots,X_n$, $n\geq3$, is an n-component, additive conjoint structure, then there exist real-valued functions v_i on X_i, $i \varepsilon N$, such that for all a_i, $b_i \varepsilon X_i$,

$$a_1 \ldots a_n \geq b_1 \ldots b_n \text{ iff } \sum_{i=1}^{n} v_i(a_i) \geq \sum_{i=1}^{n} v_i(b_i).$$

The following is an outline of the corresponding proof:

First, it will be assumed that all components are essential. That is, every attribute exercises some influence on the preference value. The next step is to show that there is a symmetric substructure for which the theorem is true. Let x_0, $x_1' \varepsilon X_i$ be such that $x_1' > x_0$ and w_0', $w_1 \varepsilon X_j$ such that $w_1 > w_0'$. If $x_0w_1 \approx x_1'w_0'$, the elements are accepted as given. If $x_1'w_0' > x_0w_1$, then since $x_0w_1 > x_0w_0'$, there exists, according to restricted solvability, an $x_1 \varepsilon X_i$ such that $x_1 > x_0$ and $x_1w_0' \approx x_0w_1$. If $x_0w_1 > x_1'w_0'$, then a similar argument shows that $w_0 \varepsilon X_j$ exists such that $w_1 > w_0$, $x_1'w_0 \approx x_0w_1$. Dropping all primes, y_1, $y_0 \varepsilon X_k$ can also be constructed such that $w_0y_1 \approx w_1y_0$, and by independence, it follows $x_0y_1 \approx x_1y_0$. By continuing inductively, it can be seen that each two-component substructure bounded by these elements is symmetric in addition to fulfilling the original conditions.

Each X_i, X_j has an additive representation v_i, v_j. In fact the $v_1, \ldots, v_n$ may be selected such that any pair is additive over the symmetric substructure. To show this, it is sufficient to show for any distinct i, j, k that the group operations induced on j by i and k are identical. The $v_j(x)$ values are assigned such that $v_j(wo_jy)$, where o_j represents concatenation of j component (Krantz,et.al., 1971), equals $v_j(w) + v_j(y)$. By independence, for any $x \varepsilon X_j$, $\bar{x}_i\bar{x}_j\pi_k(x) \approx \bar{x}_ix\bar{x}_k \approx \pi_i(x)\bar{x}_j\bar{x}_k$. Using this and independence, for x, z ε

X_j, $\pi_i(z)(xo_k z)\bar{x}_k \approx \pi_i(z)x\pi_k(z) \approx \bar{x}_i(xo_i z)\pi_k(z) \approx \pi_i(z)(xo_i z)\bar{x}_k$, whence $xo_k z \approx xo_i z$. Thus, if v_i, v_j, and v_k are additive representation, there is a linear transformation of the latter pair into v_j, v_k. Finally, it is not difficult to show that v_i, v_k is also additive. Suppose that x, z ε X_i, w, r ε X_j, and y, s ε X_k are all within the symmetric substructure and that $xwy \geq zrs$, then it can be shown that

$v_1(x) + v_2(w) + v_3(y) \geq v_1(z) + v_2(r) + v_3(s)$

This is obvious if $x \geq z$, $w \geq r$, and $y \geq s$, so it can be assumed that at least one inequality is reversed. With no loss of generality, it may also be assumed that either

(1) $x \geq z$, $w \geq r$, $s > y$ or

(2) $x > z$, $w < r$, $s > y$.

(1) If $xy \geq zs$, then $v_i(x) + v_k(y) \geq v_i(z) + v_k(s)$ and from $w \geq r$, $v_j(w) \geq v_j(r)$, so that the result follows by addition of inequalities. On the other hand assume $zs > xy$. Since $xs > zs$ (by (1) above), by invoking restricted solvability, it implies that there exists a b ε x_k such that $zs \approx xb$. Thus, $xwy \geq zrs \approx xrb$, so $wy \geq rb$, whence $v_j(w) + v_k(y) \geq v_j(r) + v_k(b)$. From $xb \approx zs$, $v_i(x) + v_k(b) = v_i(z) + v_k(s)$. Adding the two inequalities and subtracting $v_k(b)$ yields the result.

(2) From $r > w$ and $s > y$, $xws > xwy \geq zrs > zws$. Thus, by restricted solvability, there exists c $\approx$ X_i such that $cws \approx xwy \geq zrs$. Thus, $v_i(x) + v_k(y) = v_i(c) + v_i(s)$ and $v_i(c)$ +

$v_j(w) \geq v_i(z) + v_j(r)$. The result follows by adding inequalities and subtracting $v_j(c)$.
Since strict preference inequalities go into strict numerical inequalities, the converse follows. A simple induction extends this result to any n. To extend additivity, use triple cancellation, reducing $x \approx z$ to the case where x is in the symmetric substructure and at most one z_i is outside it.

The original version of this theorem was presented by Debreu (Debreu,1960). Some of his topological assumptions were replaced by more general axioms, for example, solvability and the archimedean property.

Keeney and Raiffa (Keeney and Raiffa,1976) presented conditions for the additive value functions which are abstractions of the axioms developed by Krantz, et. al (1971). Two independence conditions, namely preferential independence and mutual preferential independence, are defined below.

The set of attributes Y is preferentially independent (PI) of the complementary set Z, iff
(Y',Z") > (Y",Z') implies (Y',Z") > (Y",Z") for all Z ,Y ,Y
where $Y = \{X_1,...,X_S\}$, $Z = \{X_{S+1},...,X_N\}$

The attributes $X_1,...,X_N$ are mutually preferentially independent (MPI) if every subset Y of these attributes is preferentially independent of its complementary set of evaluators.

Given attributes $X_1,\ldots,X_N$, an additive value function

$$V(X_1,\ldots,X_N) = \sum_{i=1}^{N} V(X_i)$$

exists if and only if the alternatives are mutually preferentially independent (Keeney and Raiffa,1976).

The MPI and PI assumptions are very useful to identify additive value functions. However, the number of preferential independence conditions to verify gets very large as the number of attributes gets only moderately large. In general, if there are n attributes, there are n(n-1)/2 pairs of attributes that must be preferentially independent of their respective complements. Gorman saves much potential work by allowing verification of MPI using only n-1 sets of preferential independence conditions (Gorman,1968a; Gorman,1968b). His results are stated as follows:

Let Y and Z be subsets of the attribute set $S = \{X_1,\ldots,X_n\}$ such that Y and Z overlap, but neither is contained in the other, and such that the union Y U Z is not identical to S. If Y and Z are each preferentially independent of their respective complements, then the following sets of attributes,

1. $Y \cup Z$

2. $Y \cap Z$

3. $Y-Z$ and $Z-Y$

4. $Y-Z \cup Z-Y$

are each preferentially independent of their respective complements. This saves a lot of work in verifying preferential independence conditions.

2.2.2 MEASURABLE MULTIATTRIBUTE VALUE FUNCTIONS

The term measurable value function is used, since the differences in the strength of preference between pairs of alternatives or, more simply, the preference difference between the alternatives can be ordered.

Measurable multiattribute value functions provide an alternative to cumbersome assessment procedures, since difference consistency is so intuitively appealing that it could be assumed in most applications (Dyer and Sarin,1979).

Dyer and Sarin (Dyer and Sarin,1979) present the following conditions:

1. Preferential independence (PI)

2. Mutual preferential independence (MPI)

3. Difference consistent - The set of mutually preferentially independent attributes $X_1,\ldots,X_n$ is difference consistent if, for all w_i, $x_i \in X_i$
   $(w_i,\bar{w}_i) \geq (x_i,\bar{w}_i)$
   iff $(w_i,\bar{w}_i),(x_i,\bar{w}_i) \geq^* (x_i,\bar{w}_i),(x_i,\bar{w}_i)$
   for some $\bar{w}_i \in \bar{X}_i$ and for any $i \in \{1,\ldots,n\}$ and if $w \approx x$ then $wy \approx^* xy$ or $yw \approx^* yx$ or both for any $y \in X$.

4. Difference independent - The attribute X_i is difference independent of $\bar{X}_i$ if, for all w_i, $x_i \in X_i$ such that
   $(w_i,\bar{w}_i) \geq (x_i,\bar{w}_i)$ for some $w_i \in X_i$ and
   $(w_i,\bar{w}_i),(x_i,\bar{w}_i) \approx^* (x_i,\bar{x}_i),(x_i,\bar{x}_i)$ for any $\bar{x}_i \in \bar{X}_i$.

In addition to the four conditions, structural and technical conditions such as solvability, Archimedean, and essentialism must hold. Thus, there exist additive value functions over the alternative set. Fishburn (Fishburn,1970) presented a similar result, but used persistence conditions which are similar to the difference independence conditions.

For multilinear and multiplicative cases, weak difference independent is defined. X_I is weak difference independent of $\bar{X}_I$, if given any w_I, x_I, y_I, z_I ε X_I and some $\bar{w}_I$ ε $\bar{X}_I$ such that

$(w_I,\bar{w}_I),(x_I,\bar{w}_I) \geq^* (y_I,\bar{w}_I),(z_I,\bar{w}_I)$ and

$(w_I,\bar{x}_I),(x_I,\bar{x}_I) \geq^* (y_I,\bar{x}_I),(z_I,\bar{x}_I)$ for any x_I ε X_I.

When X_I is weak difference independent of $\bar{X}_I$, then

$v(x_I,\bar{x}_I) = g(\bar{x}_I) + h(\bar{x}_I)v(x_I,\bar{w}_I)$

for any x_I, $\bar{x}_I$, and w_I.

2.2.3 MULTIATTRIBUTE VALUE FUNCTION: A BINARY CASE

In some decision situations, alternatives represent combinations of binary attribute states. In the case of continuously measured attribute states, the axiomatic problem is solved (Fishburn,1970; Krantz, et. al,1971; Keeney and Raiffa,1976; Dyer and Sarin,1979). Deutsch and Malmborg solved the case of binary attribute states (Deutsch and Malmborg,1984). The results show the insufficiency of solvability axioms for an additive conjoint structure in the case of binary attributes.

The Archimedean property requires that all strictly bounded standard sequences be finite. The indivisibility of binary attributes precludes the existence of standard sequences. Thus, the Archimedean property cannot hold on a binary alternative set. They also show that the restricted solvability axiom cannot be defined for the binary case.

Alternative conditions for binary attributes are defined as follows:

1. Discrete Weak Difference Independence (DWDI)
   $(e_i{}^*, \bar{e}_i), (e_i^\circ, \bar{e}_i) \geq^* (e_i^\bullet, \bar{e}_i), (e_i^\circ, \bar{e}_i)$ implies that
   $(e_i{}^*, \bar{e}_i{}'), (e_i^\bullet, \bar{e}_i{}') \geq^* (e_i^\bullet, \bar{e}_i{}'), (e_i^\bullet, \bar{e}_i{}')$ for any $\bar{e}_i{}' \varepsilon \bar{E}_i$.

2. Mutual Discrete Difference Independence (MDWDI) - The binary attribute set E satisfies MDWDI if given
   $e_I, e_I{}', e_I{}'', e_I{}'''$ ε E_I and e_I,
   $(e_I, \bar{e}_I), (e_I{}', \bar{e}_I) \geq^* (e_I{}'', \bar{e}_I), (e_I{}''', \bar{e}_I)$ implies that
   $(e_I, \bar{e}_I{}'), (e_I{}', \bar{e}_I{}') \geq^* (e_I{}'', \bar{e}_I{}'), (e_I{}''', \bar{e}_I{}')$ for any $\bar{e}_I$, $\bar{e}_I{}'$ where $I \subset N$.

3. Discrete Difference Consistency (DDC) - The binary attribute set E satisfies DDC if it satisfies monotonicity and if for all $I \subset N$ and
   $\bar{e}_I \varepsilon \bar{E}_I$, $e \approx e'$ implies that $e, e'' \approx^* e', e''$.

4. Discrete Compensatory Independence (DCI) - This condition is defined as follows:
   If $(e_I, e_J, \bar{e}_{IJ}) > (e_I{}', e_J{}', \bar{e}_{IJ})$ and $e_I{}' \geq e_J$
   then $(e_I, \bar{e}_I), (e_I{}', \bar{e}_I) \geq^* (e_J{}', \bar{e}_J), (e_J, \bar{e}_J)$
   for all $\bar{e}_{IJ} \varepsilon \bar{E}_{IJ}$, $\bar{e}_I \varepsilon \bar{E}_I$, $\bar{e}_J \varepsilon \bar{E}_J$ and $I, J \subset N$.

When the four conditions hold, there exists an additive representation over the alternative set

$$v(e) = \sum_{i=1}^{n} c_i e_i .$$

Multilinear form requires DWDI. When e_i is DWDI to its complement, we have

$$v(e_i, \bar{e}_i) = g(e_i) + h(e_i)v(e_i', \bar{e}_i)$$

The MDWDI must hold for multiplicative form of

$$\lambda v(e) + 1 = \prod_{i=1}^{n} (\lambda v_i(e_i) + 1)$$

Readers who are interested in the proof of these theorems are referred to Deutsch and Malmborg (1984).

2.2.4 SUMMARY AND CONCLUSION

Three essential results including those pertaining to: multiattribute value theory, measurable multiattribute value functions, and multiattribute value functions for the binary case have been presented. Multiattribute value theory represents the case when all the attributes are continuous. The measurable multiattribute value function theory provides an alternative axiomatic system that is applicable to continuous attributes but based on different independence concepts. The multiattribute value function theory for the binary case is useful in many decision making situations since the nature of attributes in many problems may not be continuous. As we will show in the next chapter, layout design is not in general a continuous attribute problem.

In the next chapter, we will show that the facility layout problem represents a special class of decision problems for which no existing axiomatic foundation has been defined. Further, it is shown that decision analysis concepts can be easily applied to the layout problem in the context of optimal and heuristic strategies for layout creation. In addition to developing such axiomatic foundations and layout creation strategies, a decision support technology for their implementation is presented in Chapter 4.

3. DECISION THEORETIC APPROACHES TO THE GENERAL PLANT LAYOUT PROBLEM

3.1 INTRODUCTION

In the previous chapter, literature in the area of facility design and measurement theory was surveyed. This survey provided background in both of these areas. In this research, measurement theory is applied to evaluate and design layout alternatives. As will be described in this chapter, measurement theory is used to construct models that represent the layout designer's preferences. Proper implementation of these models leads to an efficient facility design approach.

3.1.1 STATEMENT OF NEED FOR THE ALTERNATIVE APPROACH

The layout design problem is complex in nature. It involves many conflicting criteria such as improving worker safety, enhancing system flexibility, improving labor satisfaction, and reducing material handling cost, material flow, and distance travelled. Some of these criteria are quantitative in nature while others are qualitative. Typically, layout design approaches consider the criteria of reducing material handling cost, material flow or distance travelled. Approaches, such as CRAFT and SUPERCRAFT, utilize quantitative information regarding material flow, cost, and distance. Layouts designed by these approaches require major adjustments of the final plan to satisfy other qualitative criteria. ALDEP and CORELAP utilize qualitative information regarding the relationships among activities (REL chart). The designer provides the degree of importance for pairs of departments to be adjacent. The objective is then to create a layout which satisfies as many of these relationships as possible.

An approach is needed that explicitly addresses the multicriteria nature of the facilities design problem. One possibility is to utilize decision theory to convert qualitative and quantitative information into readily measurable forms. This approach should also provide a precise way of utilizing closeness ratings so that they can be applied effectively in the evaluation and construction phases of layout design.

3.1.2 OBJECTIVES OF THE ALTERNATIVE APPROACH

Recognizing the multicriteria nature of the facility layout problem, general objectives of this research effort are to develop:

1. a methodology to design and evaluate facility layouts, and

2. a suitable technology to implement such a methodology.

The methodology is a decision theoretic approach which utilizes measurement theory in measuring both quantitative and qualitative criteria in a systematic fashion. The first step of the methodology is the development of an extension of the REL construct to incorporate multiattribute value functions. The extension of the REL construct is to provide a more precise statement of closeness ratings. These values can better represent designer preferences. Following this, measurement theory is utilized in combining closeness rating scores, material handling cost and other specified criteria into a single value. This value represents overall merit for a layout alternative.

The second objective is achieved by designing a proper decision support system to implement the methodology. The decision support system is developed by studying the man machine interaction with the objective of utilizing both man and machine effectively. The system is then programmed on an IBM PC for implementation.

3.1.3 OVERVIEW OF THE CHAPTER

In the section that follows, measurement considerations used for preference modeling and evaluation of layout designs are described. These will address the multiattribute nature of facility layout problems, including the need to consider both continuous and discrete attributes in problems, the insufficiency of the present axiomatic system, and an alternative axiomatic system for the mixed case. This is followed by a description of a decision theoretic approach to facility layout which includes an extension of the REL construct, utilization of multiattribute value functions in layout creation, and overall multiattribute evaluation of layout alternatives.

3.2 MEASUREMENT CONSIDERATIONS

The purpose of this section is to define facility design as a multiattribute problem and investigate it from a measurement theory point of view. We also show that the problem cannot be modeled using any existing axiomatic system. An alternative axiomatic system is then suggested.

3.2.1 THE MULTIATTRIBUTE NATURE OF THE FACILITY LAYOUT PROBLEM

In a facility layout problem, the designer must decide where to locate facilities in order to satisfy a set of criteria. Such criteria include material handling cost, flexibility for product change, flexibility for expansion, safety, labor satisfaction and supervision. There are numerous studies devoted to the criterion of minimizing material handling costs. The criterion is well defined and can usually be measured effectively. There are many mathematical models developed that can represent this measurement. However, other criteria mentioned above cannot be measured objectively because they are ill-defined and/or too complex to be modeled. This type of criteria are rated subjectively by the experienced designer. Facility design is therefore a multiattribute problem since both well defined and ill defined criteria should be considered when designing layout alternatives. Explicit recognition of such criteria has the potential to lead to an effective layout design methodology.

3.2.2 THE NEED TO CONSIDER THE MIXED CONTINUOUS AND DISCRETE MULTIATTRIBUTE PROBLEM

Given that facilities layout is a multiattribute problem, we can take a closer look at each attribute or criterion. Suppose that the design criteria under consideration are material handling cost, flexibility, safety, and ease of supervision. All of these criteria are relevant for most facility design problems. Material handling cost is a continuous attribute, since any small change in arrangement usually results in an increase or decrease in material handling cost. On the other hand, flexibility is a subjective criterion and must be rated by the designer. The specification of flexibility may be easily perceived by the designer if the scale of flexibility is divided into discrete intervals like low, medium, and high. The designer can then accurately rate flexibility based on experience. If flexibility is scaled between 0 and 100, the designer would have difficulties relating scaling values to his experience. Thus, such criteria should assume approximate discrete values. A similar assumption can also be applied to the safety criteria. Consequently, we can conclude that a facility layout problem involves both continuous and discrete attributes.

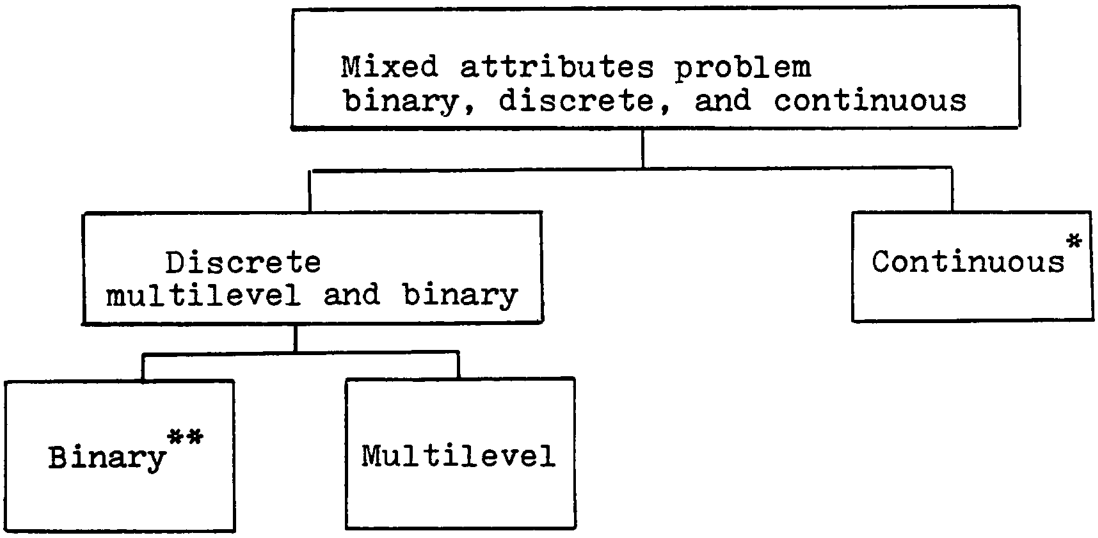
3.2.3 THE INSUFFICIENCY OF PRESENT AXIOMATIC SYSTEMS

Axiomatic systems for additive, multiplicative, and multilinear decision model forms have been developed by Krantz, et. al.(1971). Keeney and Raiffa (1976) have presented conditions which are the abstract of those axioms. These conditions are mutual preferential independence (MPI) and preferential independence (PI). These conditions can be used to verify the consistency of additive, multiplicative, or multilinear preference models with a decision maker's stated preferences. In addition to these conditions, there are four technical conditions which must hold for an additive model (Krantz, et. al., 1971). These are referred to as weak ordering, essentialism, the Archimedean property, and restricted solvability. Deutsch and Malmborg (1984) demonstrated the insufficiency of the axiomatic system stated above for application to the binary attribute problem. They showed that two technical conditions, namely, weak ordering and essentialism, can be defined for the binary multiattribute case, while the restricted solvability and the Archimedean property cannot be defined. This leads to the conclusion that the existing axiomatic system can apply to only continuous attribute values. Thus, they developed an alternative axiomatic system for the binary multiattribute case. The development resulted in four conditions: 1) discrete weak difference independence (DWDI), 2)mutual discrete weak difference independence (MDWDI), 3) discrete difference consistency (DDC), and 4) discrete compensatory independence (DCI).

To demonstrate the insufficiency of the present axiomatic systems for the mixed attribute problem, it is enough to show the insufficiency of Krantz's additive conjoint structure for the binary case and the alternative axiomatic system (Deutsch and Malmborg, 1984) for the mixed attribute case. The binary attribute case as well as the continuous case is a special case of the mixed attribute case. The structure of the multiattribute classification is shown in Figure 3.1.

Before proceeding with the discussion, we first need to establish some terminology. Let $E= E_1,\ldots,E_n$ be a consequence space where E_i is the i^{th} binary attribute and $(e_1,\ldots,e_n)$ is a specific consequence. The attribute E_i can assume only two levels given by the most preferred state, $e_i*=1$, or the least preferred state, $e_i=0$, for $i \varepsilon N=\{1,\ldots,n\}$. The term E is defined to mean $\prod_{i \varepsilon I} E_i$, $I \subset N$ and e_i is a specific level of E_i.

Now, by essentialism, $(e_i*, e_j*, \bar{e}_{ij}) > (e_j*, \bar{e}_i^{\bullet})$, which thereby requires that $V_i(e_i*) > V_j(e_j^{\bullet})$ and $V_i(e_i*) > V_j(e_j^{\circ})$, and thus



* Krantz, et. al. (1971), Keeney and Raiffa (1976)

** Deutsch and Malmborg (1984)

Figure 3.1 Multiattribute value function classifications

$V_i(e_i*)+V_j(e_j*) > V_i(e_i^\circ)+V_j(e_j^\circ) = 0$, for an arbitrary zero point of the interval scaled V values. However, if $(e_i*,\bar{e}_i^\circ) \geq (e_j*,\bar{e}_j^\circ)$, then by transitivity and independence we know that $V_i(e_i*) \geq V_j(e_j*)$ for $i,j \in N$. To extend this result somewhat, suppose that there exist $e_i,e_i' \in E_i$, $e_j,e_j' \in E_j$ and $e_I,e_I' \in E_I$, such that $(e_i,e_j,e_I) \geq (e_i',e_j',e_I')$. The question essentially asks if the following condition holds, that is:

(1) $V_i(e_i)+V_j(e_j)+V_I(e_I) \geq V_i(e_i')+V_j(e_j')+V_I(e_I')$.

This result follows directly if $e_i \geq e_i'$, $e_j \geq e_j'$, and $e_I \geq e_I'$. The other possibilities are one of the two general forms, namely,

(2) a: $(e_i \geq e_i',\ e_j \geq e_j',\ e_I' > e_I)$, b: $(e_j \geq e_j',\ e_I \geq e_I',\ e_i' > e_i)$ or c: $(e_i \geq e_i',\ e_I \geq e_I',\ e_j' > e_j)$

and

(3) x: $(e_i \geq e_i',\ e_j' > e_j,\ e_I' > e_I)$, y: $(e_j \geq e_j',\ e_i' > e_i,\ e_I' > e_I)$ or z: $(e_I \geq e_I',\ e_i > e_i',\ e_j' > e_j)$.

For the cases described by (2), restricted solvability defined on E does not preclude (1). For example, consider version a. Here, if $(e_i,e_I,e_j^\circ) \geq (e_i',e_I',e_j^\circ)$, then $V_i(e_i) + V_I(e_I) \geq V_i(e_i') + V_I(e_I')$, and since $e_j \geq e_j'$, then $V_j(e_j) \geq V_j(e_j')$ and (1) is obtained by adding these two inequalities. Alternatively, if $(e_i',e_I',e_j^\circ) \geq (e_i,e_I,e_j^\circ)$, then by restricted solvability there exists e_I'' $\in$ E_I such that $(e_i',e_I',e_j^\circ) \approx (e_i,e_I'',e_j^\circ)$ so that $(e_j,e_I,e_i^\circ) \geq (e_j',e_I'',e_i^\circ)$

implying that $V_j(e_j) + V_I(e_I) \geq V_j(e_j') + V_I(e_I")$. Using the result that $(e_i',e_I',e_j^\circ) \approx (e_i,e_I",e_j^\circ)$, we obtain $V_i(e_i) + V_I(e_I") = V_i(e_i') + V_I(e_I')$. When added to the previous inequality, (1) is obtained upon subtracting $V_I(e_I")$ from both sides of the resultant inequality. However, a problem arises with respect to cases described by (3). Consider version x from which it follows that $(e_i,e_j,e_I') > (e_i,e_j,e_I) \geq (e_i',e_j',e_I') > (e_i',e_j,e_I)$. Restricted solvability requires that there exists $e_i" \in E_i$ such that $(e_i",e_j,e_I') \approx (e_i,e_j,e_I) \geq (e_i',e_j',e_I')$. However, (1) in this case implies that $e_i>e_i'$ and $(e_i",e_j,e_I') \approx (e_i,e_j,e_I)$ implies that $e_i>e_i"$ for version x. This is impossible since it implies that $e_i> e_i"> e_i'$ and attributes can assume only two levels by definition. Thus, restricted solvability with respect to binary attributes cannot be defined to show that E has an additive conjoint structure and an alternative axiomatic basis for value functions is needed in this case.

3.2.4 AN AXIOMATIC SYSTEM FOR THE MIXED CASE

Most of the facilities design problems of interest have been shown to be mixed attribute problems and due to the insufficiency of the present axiomatic systems for this case, we develop an axiomatic system for the mixed case. The detail of the alternative axiomatic development is one major objective of this research. The development of axiomatic systems is presented in Chapter 5.

3.2.5 SUMMARY AND CONCLUSION

This section has shown that the facility layout problem is a multiattribute problem having both continuous and discrete attributes. We refer to this problem as the mixed continuous and discrete multiattribute problem. The insufficiency of the present axiomatic system to support the mixed problem has been demonstrated. In the course of this research efforts will be directed toward development of an alternative axiomatic system defined on a mixed attribute set for additive, multiplicative, and multilinear forms.

3.3 APPLICATION OF DECISION THEORY TO THE LAYOUT DESIGN PROBLEM

As stated earlier, facilities design involves both quantitative and qualitative factors. However, there are few reported research efforts that apply a decision theoretic approach to this problem. Most of the approaches are directed to quantitative factors. Some approaches, e.g. ALDEP and CORELAP, consider qualitative factors by using the activity relationship chart, and they construct layout alternatives based on activity relationship information. Few studies, for example Rosenblatt's (1980), consider both factors. Rosenblatt used scores from each objective function multiplying those by weights to reflect the importance of each objective. This was then used in an additive scoring model.

However, from the measurement theory point of view, the method of combining the two objectives suggested by Rosenblatt (1980), cannot really represent the merit of the layout alternative, since the two objectives have different scaling standards. In order to combine the two objectives, measurement theory must be applied to convert both scaling standards into the same type, and find an appropriate combination rule, for example, additive and multilinear. Section 3.3.3 is devoted to this subject.

Aside from the role of measurement theory stated in the previous paragraph, it can also be applied in constructing the REL chart. Traditionally, all pairwise combinations of relationships are evaluated, and a closeness rating (A,E,I,O,U, or X) is assigned to each facility pair. Then, arbitrary values are assigned to closeness ratings, for example, CORELAP uses A=6, E=5, I=4, O=3, U=3, and X=1. Again this fails to represent an accurate closeness rating since the numerical values are arbitrarily assigned. Section 3.3.1 is devoted to the use of measurement theory in constructing the REL chart.

3.3.1 EXTENSION OF THE REL CONSTRUCT TO INCORPORATE MULTIATTRIBUTE VALUE FUNCTIONS

The proposed extension of the REL will provide more insight into the facility layout problem and more meaningful numerical values for closeness ratings.

It is a fact that the closeness rating assigned to any pairwise combination of facilities is typically based on multiple criteria. The purpose of extending the REL construct is to incorporate such reasons. This extension may be viewed as considering a set of criteria which can be satisfied by having adjacent facilities. Such criteria can include convenience, common personnel, ease of supervision, flow of materials, etc. Some of them are listed in Appendix D.

A preference model based on the adjacency criteria is presented. The model can have additive, multiplicative, or multilinear forms depending on the satisfaction of the independence conditions with respect to the decision maker's preferences. Let $X_1,...,X_n$ be the criteria under consideration, for example, convenience, common personnel, ease of supervision, etc. If the conditions across the attributes for the additive form hold, we have a preference value for any paired facility i,j as follows:

$$\text{Preference value } (p_{ij}) = c_1V_1(X_{ij1})+...+c_nV_n(X_{ijn})$$

where

c_k = scaling constant describing the relative importance of attribute k,

$V_k(X_{ijk})$ = marginal value function describing the preference effects of variation in the level of individual attribute k of department pair i,j.

The designer inputs desired levels of each attribute for each facility pair. For n facilities with m attributes, there are a maximum of m*n*(n-1)/2 specifications. For 40 facilities and 4 attributes, there are a maximum of 3120 specifications. However, the actual number of specifications would not reach this number, since not all attributes in any pair i,j are perceived as relevant and/or sufficiently important. The results from the model are numerical values representing closeness ratings. The method presented provides a finer scale than the existing A, E, I, O, U, X rating and arbitrarily assigned numerical values.

3.3.1.1 EFFICIENT ASSESSMENT OF THE VALUE FUNCTIONS

Theoretically, the value function for each attribute may not be assessed independently due to potential interdependence among attributes; thus, an assessment of joint value functions should be conducted. This joint value function is similar to a joint probability function. When appropriate independence conditions across all attributes hold, we are able to evaluate and assess value functions independently; such a function is called a marginal value function. The efficient way of evaluating marginal value functions is the mid-value splitting technique (Keeney and Raiffa, 1976); another one is the SMART technique due to Edwards (1971).

Upon completing the assessment of marginal value functions, we assess scaling constants. The scaling constants represent an importance weight for attributes. The assessed marginal value functions and scaling constants as well as an aggregation form represents the closeness rating function for the problem.

3.3.1.2 OVERVIEW OF ASSESSMENT SOFTWARE

An assessment software package is developed for the implementation of the method. The software is capable of assessing marginal value functions and scaling constants for an additive preference model. In addition, the software utilizes interactive graphics to simplify the assessment procedure. The procedure consists of the following general steps.

1. Collect information about the attributes.

2. Rate the attributes on a scale between 1 to 100.

3. Normalize the ratings determined in step 2.

4. If reassessment of scaling constants is required, go back to step 2. Otherwise, proceed with the assessment of marginal value functions.

5. Develop marginal value functions.

6. Store the scaling constants and value functions.

Once the preference matrix has been completed, layout creation proceeds. This is described in the next section.

3.3.2 UTILIZATION OF THE MULTIATTRIBUTE VALUE FUNCTIONS IN LAYOUT CREATION

This section describes how closeness ratings can be used in the layout creation process. Then, advantages of the proposed method over existing ones are discussed.

3.3.2.1 DESCRIPTION OF THE PROCEDURE

Having developed the preference values for all pairwise facilities from the previously described methodology, we use these values as evaluators in the layout creation stage. To evaluate the degree of satisfaction, total preference (TPR) is defined as follows:

$$TPR = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}/d_{ij}$$

where

d_{ij} = rectilinear distance between department centroids.

p_{ij} = closeness rating of pair i,j.

Our goal is to create a layout which gives the highest TPR possible since this indicates fulfillment of adjacency requirements. A computer program will be developed to create facility layouts using these closeness ratings. The software will utilize a closeness rating matrix and TPR as evaluators during the creation process. The initial layout creation procedure is described in chapter six.

3.3.2.2 ADVANTAGES OVER EXISTING METHODS

Existing software, such as CORELAP and ALDEP, use a closeness rating which is arbitrarily assigned. The scales used are divided into 6 levels, A, E, I, O, U, and X. They use a similar selection and placing rule as illustrated in the proposed method. However, it is more likely that ties in the selection and placing process would occur as a consequence of their rough scaling. The method presented utilizes a finer scaling system. Thus, closeness ratings from our proposed method more closely represent the actual preference of the designer. In situations where most facilities are required to be adjacent, the finer scaling system can easily distinguish which facility pairings are preferable.

3.3.3 OVERALL MULTIATTRIBUTE EVALUATION OF LAYOUT ALTERNATIVES

In the overall evaluation step, several evaluation criteria can be defined. Two of them are total preference rating and material handling cost. Others may be flexibility, ease of expansion, and appearance. The following evaluation model is used which provides ranking scores.

$$S_i = \sum_{j=1}^{n} a_j v_j(c_{ij})$$

where

S_i = score for alternative i,

c_{ij} = specification of criterion j for alternative i,

a_j = scaling constant of criterion j in the overall evaluation model, and

$v_j(c_{ij})$ = marginal value function of criteria j.

The computer calculates the total preference rating and total material handling cost for every alternative. Evaluations for subjective criteria are provided by the designer. These evaluations are aggregated by the evaluation model. Layout alternatives are then ranked based on their scores. The layout with the highest score represents the most preferred layout.

3.4 THE MAIN STEPS OF THE PROPOSED PROCEDURE FOR LAYOUT DESIGN

The procedure described above for the layout design problem can be summarized in the following eight steps.

Step 1. Determination of closeness rating.

This step evaluates closeness ratings of pairs of departments.

Step 2. Determination of value functions and scaling constants.

In this step, the marginal value functions and scaling constants for closeness related criteria are evaluated.

Step 3. Determination of preference ratings.

The closeness ratings of step (1) are converted in this step into preference scales called preference ratings.

Step 4. Determination of preference matrix.

The preference ratings of step (3) are then converted into preference value as described in section 3.3.1.

Step 5. Layout creation.

The layout is then created as described in section 3.3.2, utilizing the preference matrix of step (4) as evaluators.

Step 6. Assessment of value functions and scaling constants for the overall decision criteria.

Next, the designer generates value functions and scaling constants for the overall decision criteria.

Step 7. Computations of final scores.

Finally, scores for ranking layout alternatives are computed as described in section 3.3.3 using the information gathered in steps (5) and (6).

Next, we illustrate these steps on an example problem. Consider a problem containing four departments. There are two closeness related criteria of interest, namely, the flow of material and the necessary contact between departments. Their values are as follows.

Flow

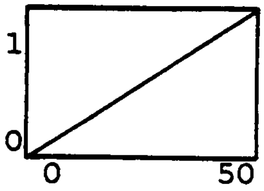
	1	2	3	4
1	-	50	30	10
2	50	-	5	20
3	30	5	-	10
4	10	20	10	-

Necessary contact

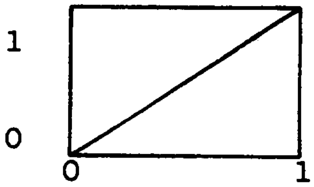
	1	2	3	4
1	-	0	1	0
2	0	-	0	1
3	1	0	-	0
4	0	1	0	-

The flow of material between departments i and j is measured here in terms of the number of trips made between the departments. For example, the material flow between departments 1 and 2 is represented by 100 trips with 50 trips each way. The necessary contact between departments is important because of technological constraints and is represented above by 0 and 1. For example, a contact is necessary between departments 1 and 3 and is not necessary between departments 1 and 2. This constitutes step (1) described above. Next, information is gathered regarding the designer's value functions and scaling constants. Assume these are as follows for the two criteria selected for this problem.

Flow
scaling constant=0.3



Contact necessary
scaling constant=0.7



The preference functions map closeness ratings into preference ratings, for example, 50 trips in the value function for flow represents preference rating of 1 and 0 trip represents preference rating of 0. Scaling constants represent weights of each criteria. The scaling constants for the criteria of flow and necessary contact are 0.3 and 0.7 respectively. In other words, the necessary contact is more than twice as important than the flow. Using these functions, the closeness ratings of the above example are mapped into the following preference ratings.

Flow

	1	2	3	4
1	-	1.0	0.6	0.2
2	1.0	-	0.1	0.4
3	0.6	0.1	-	0.2
4	0.2	0.4	0.2	-

Contact necessary

	1	2	3	4
1	-	0	1	0
2	0	-	0	1
3	1	0	-	0
4	0	1	0	-

The preference ratings of the two criteria shown above are next combined into single values, called preference values, using the scaling constants. The combination rule of the two criteria for this example problem is an additive one. For example, the preference value between departments 1 and 3 is 0.6(0.3) + 0.1(0.7) = 0.88. The preference values so obtained are shown below.

Preference value matrix

	1	2	3	4
1	-	0.30	0.88	0.08
2	0.30	-	0.03	0.82
3	0.88	0.03	-	0.06
4	0.06	0.82	0.06	-

Next, the layout alternatives are created and the preference value matrix is used as described in section 3.3.2 for evaluating alternatives. In this example, two layout alternative are created which have the TPR of 3.5 and 2.6 as shown below.

Layout alternative #1
TPR=3.5

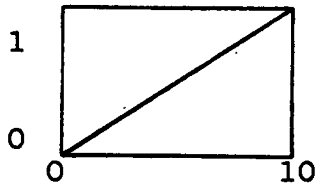
1	4
3	2

Layout alternative #2
TPR=2.6

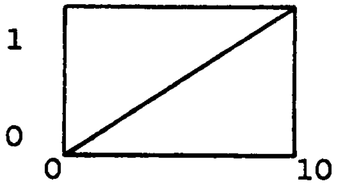
1	2
4	3

The TPR score represents the preference rating of the above two alternatives when all the closeness related criteria are considered. In case some other subjective criteria should be considered for the overall decision, we need to determine value functions and scaling constants for these criteria. For the example problem, value functions and scaling constants for the closeness related criteria and some other subjective criteria are as follows.

Closeness related criteria
scaling constant=0.6



Other subjective criteria
scaling constant=0.4



The TPR score for layout alternatives were computed earlier. The designer then rates each alternative based on a scale of 1 to 10. In this example, assume layout alternatives 1 and 2 are rated as 8 and 6, respectively. The two scores namely, TPR and the designer's rating, are mapped into preference scales. Both the TPR and the designer's rating scores are then combined. The results are shown below regarding the overall preference rating of the two layout alternatives in consideration.

$$S(1) = 0.6(0.35) + 0.4(0.80) = 0.53$$
$$S(2) = 0.6(0.26) + 0.4(0.60) = 0.40$$

The designer can now select the final layout based on these scores. In this case, layout alternative 1 is better.

3.5 SUMMARY AND CONCLUSIONS

This chapter has described a decision theoretic approach to a facility layout problem. First, the extension of the REL chart is presented by using multiattribute value theory to construct a closeness rating function for any pair of adjacent facilities. The results from the closeness rating values are represented in a finer scale than those suggested in such approaches as CORELAP and ALDEP. A finer scale and TPR presented would capture designer preferences better than the methods available in the literature, since the TPR presented includes relative distances between any pair of departments. Finally, an overall evaluation of layout alternatives is suggested by constructing a scoring model across evaluation criteria. The model is based on multiattribute value theory.

4. A DECISION SUPPORT SYSTEM FOR FACILITY LAYOUT

Most approaches to facility design have been unsuccessful as a consequence of not involving the designer sufficiently in the design and evaluation process. Therefore, there is a need for a decision support system to aid design and evaluation of facility layouts that involves the designer. Such a decision support system should recognize that the decision maker's judgment is a critical component to be utilized in the layout creation process.

The first step is to identify the decision criteria. There are different criteria and different degrees of impact for criteria in various design situations. Later, the layout alternatives are designed based on the criteria defined. The objective of the decision support system is not to construct an approach and let it create the layout alternative, but to allow human judgment to have an important role through the design process. Consequently, the designer may utilize his judgment to arrive at an ideal layout. His judgments are determined through the use of decision models, multiattribute value functions for REL and an overall multiattribute evaluation function, as presented in section 3.3.

4.1 THE MAN MACHINE CONCEPT OF DECISION MAKING APPLIED TO FACILITY LAYOUT

Since humans find it difficult to integrate a large amount of information in a consistent manner, what is needed is an aid to help overcome these specific limitations. Computers can be used to effectively moderate these limitations. They have the capability of quickly converting a large amount of information, through use of a man made model, into simple and easy to interpret forms. The user then proceeds with an action, based on his judgment and information provided by the computer. Again the computer performs tasks corresponding to the designer's reaction. These interactions continue until an acceptable output or outcome is met. Thus, the man and machine concept is applied to aid facility design. The following subsections describe comparative advantages of man versus machine. Later, the overview of the decision support system represented by a man-machine interaction diagram is presented.

4.1.1 DESCRIPTION OF COMPARATIVE ADVANTAGES

Due to the complex nature of the facility layout problem, a system applied for solving the problem must have the following capabilities.

1. Perform fixed logic - search and construct a layout which gives the lowest material handling cost and handle a large amount of data.

2. Utilize human creativity and experiences - have the decision maker evaluate the more complicated and ill-defined criteria.

The interactive computer system developed in this research has both capabilities, and effectively combines these human and machine capabilities. In a layout design problem, human capabilities are necessary for the evaluation of ill-structured criteria, creative design, and a realistic and practical layout. On the other hand, machine capabilities are needed for the creation of the initial layout according to fixed logic which includes searching and calculating, computing material handling cost, total preference, and final evaluation scores. If we combine human and machine capabilities properly, we have an effective system to design and evaluate a facility layout.

4.1.2 OVERVIEW OF THE DECISION SUPPORT SYSTEM

The decision support system is implemented on a microcomputer system. The IBM PC system was chosen because of its wide spread use. Figure 4.1 shows the man-machine interaction, and the information flow at each interaction stage. A computer program is designed following this concept.

The key to a good decision support system is to enhance human performance by providing the designer with the right information at the right time without overloading him. Generally, a user should not perform any tasks that can be programmed more effectively for the computer.

4.2 DESCRIPTION OF THE DECISION SUPPORT SYSTEM

The purpose of this section is to describe the computer software developed for this research. The software is based on the man-machine interaction diagram in Figure 4.1.

4.2.1 DATA REQUIREMENTS

The initial data requirements of the program are facility specifications that include facility lengths, widths, names, and requirements for all facility pairs. The value function components, which are scaling constants and marginal value functions, are also required as initial input. Later in the evaluation stage, scaling constants and value functions for overall decision criteria are required for a complete evaluation.

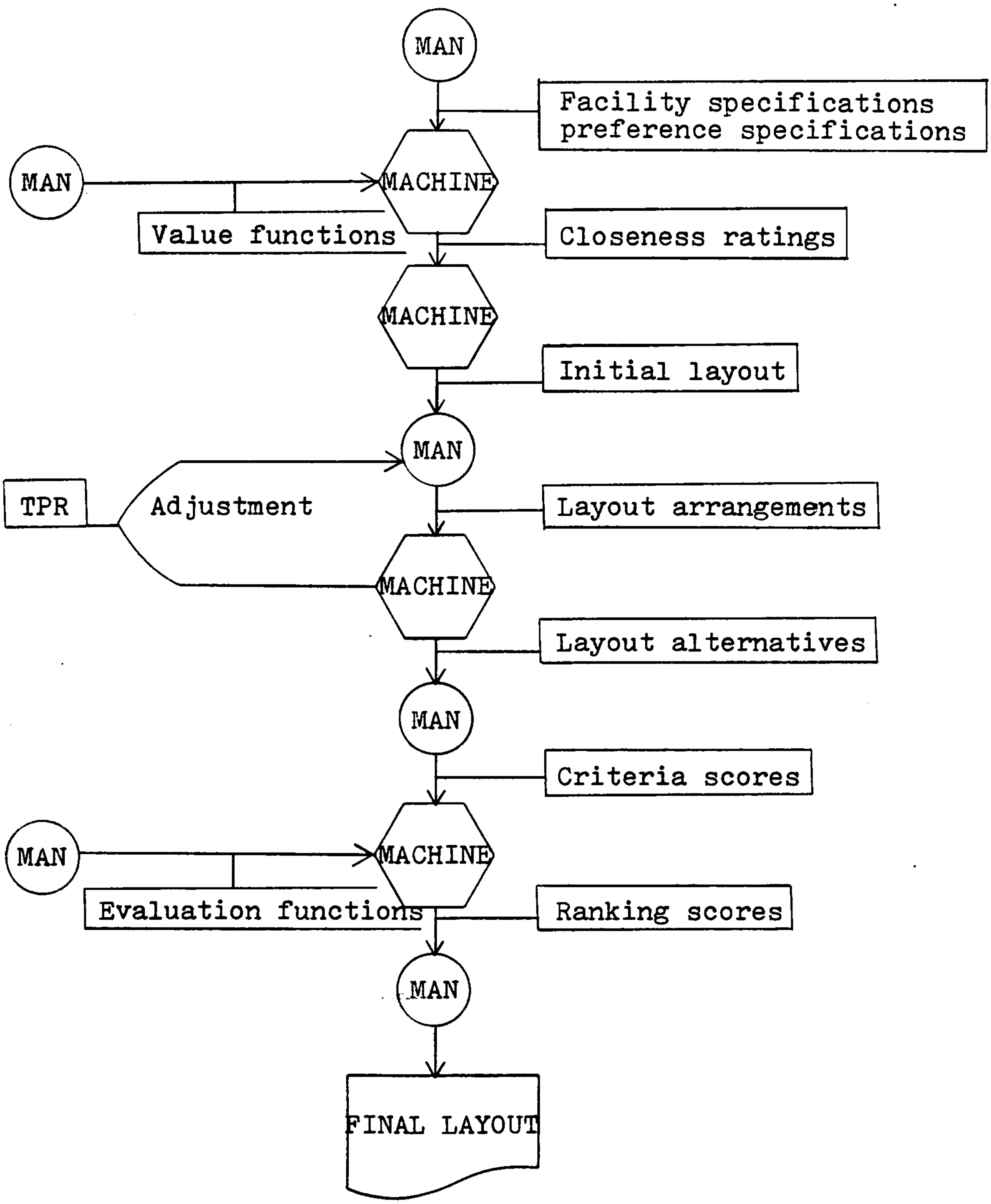


Figure 4.1 Man-machine Interaction for Facility Layout Problem

These requirements are considered to be fixed data, since they represent characteristics of the problem and the designer's behavior.

4.2.2 USER INTERACTION

The software is designed to enable the designer to interact at any stage of the program. The designer can switch to any part of the program to change or edit data as he wishes. However, the most important interaction occurs during the layout creation phase where the designer provides the locations of facilities based on his judgment. This interaction allows the designer to use his creativity in arranging the facilities based on other criteria. Then his judgment will move towards optimal design.

4.2.3 ASSUMPTIONS OF THE MODELING APPROACH

The major assumptions that underlie the software development are as follows.

1. Preference models are additive - this allows the user to assess value functions independently.

2. Distance between department centroids is rectilinear

3. Department sizes are equal - this is the assumption used in the methodology for finding relative department locations.

4.2.4 INTERFACING THE MODULES OF THE SYSTEM

The software package is divided into three phases which link to eleven subprograms. These are:

1. Data Acquisition

    a. Scaling constant assessment

    b. Marginal value function assessment

    c. Facility specification

    d. Adjacency requirement input

    e. Preference matrix computation

2. Layout construction

    a. Initial layout construction

    b. Layout editing

3. Evaluation

   a. Scaling constant assessment

   b. Marginal value function assessment

   c. Criteria specification

   d. Final scores computation

The designer can gain access to any part of the main menu. From the submain menu, the user can go back to the main menu or run any modules in that menu.

4.2.5 ADVANTAGES OF THE SYSTEM

The proposed system has the following major advantages over existing computer aided facility layout design systems.

1. It utilizes the man-machine performance concept which results in a better decision support system for the designer.

2. It includes layout creation and evaluation in one package.

3. It is implemented on a microcomputer.

4. It considers the multicriteria nature of the facility layout problem.

5. It uses an extension of the REL construct to measure closeness ratings.

4.3 SUMMARY AND CONCLUSIONS

This section has described the decision support system designed for the facility layout problem. It is, in fact, an interactive computer system designed to be a tool for layout design. The capabilities of man and machine applied to a facility layout problem were also discussed. The decision support system which shows interaction between man and machine during the design process was outlined. Furthermore, the development of the software was discussed in terms of input requirements, user interaction, assumptions of the modeling approach, and module interfacing.

5. AXIOMATIC BASIS FOR MIXED ATTRIBUTE CASES

In this chapter, theoretical results for the multiattribute values of mixed attributes are presented. The results pertain to the following cases:

1. Mixed additive case, using

   a. compensatory independence

   b. epsilon solvability

2. Mixed multilinear case

3. Mixed multiplicative case

4. Mixed additive interdependent case

Notations and basic definitions are first stated. Next, a theorem for each case is presented and it is followed by a proof.

5.1 NOTATIONS AND BASIC DEFINITIONS

5.1.1 ALTERNATIVE SPACE

Let E denote an alternative set which consists of many attributes. The attributes can be either discrete, binary, or continuous. Thus, alternative E can be represented in attribute space as follows.

$$E = e_1, \ldots, e_n, x_1, \ldots, x_m$$

where

e_i represents discrete attribute i

x_j represents continuous attribute j

e_i represents level of e where $v_i(e_i)=0$

e_i* represents level of e where $v_i(e_i*)=1$

x_j represents level of x where $v_j(x_j)=0$

x_j* represents level of x where $v_j(x_j*)=1$

$v_i(e_i)$ is marginal value function of discrete attribute i

$v_j(x_j)$ is marginal value function of continuous attribute j

5.1.2 WEAK ORDERING

Consider three alternative sets of attributes designated by $E', E'', E* \varepsilon E$. These are said to follow a weak ordering if and only if, the following two axioms are satisfied

1. Connectedness: Either $E' \geq E''$ or $E'' \geq E'$

2. Transitivity: If $E' \geq E''$ and $E'' \geq E^*$, then $E' \geq E^*$

In short, discrete, continuous, or mixed vector descriptions of alternatives can be ranked without intransitivity such that $E' \geq E'' \geq E^*$

5.1.3 ESSENTIALISM

Consider the case of two attributes $E= x_1, x_2$. Component x_1 is essential iff there exist $a,b \in x_1$ and $c \in x_2$ such that not ($ac \approx bc$). A similar definition holds for x_2. In other words, essentialism requires that the states of each attribute must exercise some influence on the preference for alternatives.

5.1.4 ARCHIMEDEAN PROPERTY

The Archimedean property requires that all strictly bounded standard sequences be finite. That is, for any set N of consecutive integers, a set $\{a_i | a_i \in x_1, i \in N\}$ is a standard sequence iff there exist $p,q \in x_2$ such that not ($p \approx q$) and for all $i,i+1 \in N$, $a_i p \approx a_{i+1} q$. A parallel definition holds for the second component.

5.1.5 DOUBLE CANCELLATION

An alternative E= x_1, x_2 satisfies double cancellation provided that, for every a,b,f ε x_1 and p,q,x ε x_2, if ax $\geq$ fq and fp $\geq$ bx, then ap $\geq$ bq. More details of this definition can be found in Krantz, et. al. (1971).

5.1.6 SOLVABILITY

An alternative E= x_1, x_2 satisfies unrestricted solvability provided that, given three of the four values of a,b,p, and q such that a,b ε x_1 and p,q ε x_2, the fourth exists so that ap $\approx$ bq. It satisfies restricted solvability provided that:

1. Whenever there exist a,$\bar{b}$,$\underline{b}$ ε x_1 and p,q ε x_2 for which $\bar{b}$q $\geq$ ap $\geq$ $\underline{b}$q, then there exists b ε x_1 such that bq $\approx$ ap;

2. A similar condition holds on the second component.

5.2 MIXED ADDITIVE CASE USING COMPENSATORY INDEPENDENCE

The following result is for an additive function of mixed discrete and continuous attributes. The main condition used in this result is compensatory independence which is an application of work done by Deutsch and Malmborg (1985) for the binary attribute case.

5.2.1 DEFINITIONS OF INDEPENDENCE CONDITIONS

5.2.1.1 PREFERENTIAL INDEPENDENCE

The attribute set E= $e_1,\ldots,e_n,x_1,\ldots,x_m$ is preferential independence iff, for e_i', e_i'' ε e_i, $(e_i',\bar{e}_i') \geq (e_i'',\bar{e}_i')$ for some e_i' ε e_i implies that $(e_i',\bar{e}_i'') \geq (e_i'',\bar{e}_i'')$ for every $\bar{e}_i''$ ε $\bar{e}_i$. This must hold for any discrete and continuous attribute. In short,

$(e_i',\bar{e}_i') \geq (e_i'',\bar{e}_i')$ iff $(e_i',\bar{e}_i'') \geq (e_i'',\bar{e}_i'')$, and
$(x_j',\bar{x}_j') \geq (x_j'',\bar{x}_j')$ iff $(x_j',\bar{x}_j'') \geq (x_j'',\bar{x}_j'')$.

Note that $\bar{e}_i$ is the e_i complement, $\bar{x}_j$ is the x_j complement.

5.2.1.2 MUTUAL PREFERENTIAL INDEPENDENCE

The attribute set E= $e_1,\ldots,e_n,x_1,\ldots,x_m$ is preferential independence iff, for e_I', e_I'' ε e_I, where I is a set of attributes, $(e_I',\bar{e}_I') \geq (e_I'',\bar{e}_I')$ for some e_I' ε e_I implies that $(e_I',\bar{e}_I'') \geq (e_I'',\bar{e}_I'')$ for every $\bar{e}_I''$ ε $\bar{e}_I$. This must hold for any set of discrete and continuous attributes. In short,

$(e_I',\bar{e}_I) \geq (e_I'',\bar{e}_I')$ iff $(e_I',\bar{e}_I'') \geq (e_I'',\bar{e}_I'')$
$(x_J',\bar{x}_J') \geq (x_J'',\bar{x}_J')$ iff $(x_J',\bar{x}_J'') \geq (x_J'',\bar{x}_J'')$
$(e_I',x_J',\overline{ex}_{IJ}') \geq (e_I'',x_J'',\overline{ex}_{IJ}')$ iff

$(e_I', x_J', \overline{ex}_{IJ}'') \geq (e_I'', x_J'', \overline{ex}_{IJ}'')$

for all $I \subset N$, $J \subset N$.

5.2.1.3 COMPENSATORY INDEPENDENCE

This condition is defined as follows;

If $(e_I', e_J', \bar{x}_{IJ}') \geq (e_I'', e_J'', \bar{x}_{IJ}')$ and $(e_J'', \bar{e}_J') \geq (e_J', \bar{e}_J')$ then $(e_I', \bar{e}_I'), (e_I'', \bar{e}_I') \geq^* (e_J'', \bar{e}_J'), (e_J', \bar{e}_J')$

for all $\bar{e}_{IJ}$, e_I, e_J and $I, J \subset N$

5.2.2 THEOREM

When a mixed attribute set satisfies technical conditions, namely, weak ordering, strictly bounded, essentialism, and also the three independence conditions of preferential independence, mutual preferential independence, and compensatory independence, there exists a real valued function v such that E' > E" if and only if

$$\sum_{i=1}^{n} v(e_i') + \sum_{i=1}^{n} v(x_j') \geq \sum_{i=1}^{n} v(e_i'') + \sum_{i=1}^{n} v(x_j'')$$

where $v(e_i) = c_i v_i(e_i)$

$v(x_j) = k_j v_j(x_j)$

$c_1+\ldots+c_n+x_1+\ldots+x_m = 1$

$v_i(e_i)$, $v_j(x_j)$ = marginal value functions

c_i, k_j = scaling constants

A theorem for the binary attributes and additive case is presented by Deutsch and Malmborg (1985). The above theorem differs from theirs due to the considerations of mixed attributes. That is, both discrete and continuous attributes are considered here. Next, we present a proof of this theorem. This proof closely follows the line of reasoning used by Deutsch and Malmborg (1985) in their proof.

5.2.3 PROOF

Consider the case where n=2 and m=2. That is, E= e_1,e_2,x_1,x_2. We assume that E' > E".

Thus $(e_1',e_2',x_1',x_2') > (e_1",e_2",x_1",x_2")$.

If $e_1' > e_1"$, $e_2' > e_2"$, $x_1' > x_1"$, $x_2' > x_2"$

==> $v(e_1') > v(e_1")$, $v(e_2') > v(e_2")$, $v(x_1') > v(x_1")$, and $v(x_2') > v(x_2")$.

Thus we have

$c_1v_1(e_1') + c_2v_2(e_2') + k_1v_1(x_1') + k_2v_2(x_2') >$

$c_1v_1(e_1") + c_2v_2(e_2") + k_1v_1(x_1") + k_2v_2(x_2")$

If $e_1' > e_1"$, $e_2' > e_2"$, $x_1' > x_1"$, $x_2' < x_2"$

$\Longrightarrow v(e_1') > v(e_1''), v(e_2') > v(e_2''), v(x_1') > v(x_1'')$, and $v(x_2') < v(x_2'')$.

By compensatory independence and mutual preferential independence,

$(e_1',e_2',x_1',x_2') > (e_1'',e_2'',x_1'',x_2'')$ and $x_2' < x_2''$.

Then

$(e_1',e_2',x_1',C_{121}), (e_1'',e_2'',x_1'',C_{121}) >^* (x_2'',C_2),(x_2',C_2)$
$\Longrightarrow v(e_1') + v(e_2') + v(x_1') + v(C_{121}) - v(e_1'') - v(e_2'') - v(x_1'') - v(C_{121}) > v(x_2') + v(C_2) - v(x_2'') - v(C_2)$.

By rearranging these terms, we have

$v(e_1') + v(e_2') + v(x_1') + v(x_2') > v(e_1'') + v(e_2'') + v(x_1'') + v(x_2'')$.

If $e_1' > e_1'', e_2' < e_2'', x_1' > x_1'', x_2' < x_2''$
$\Longrightarrow v(e_1') > v(e_1''), v(e_2') < v(e_2''), v(x_1') > v(x_1'')$, and $v(x_2') < v(x_2'')$.

By compensatory independence and mutual preferential independence,

$(e_1',e_2',x_1',x_2') > (e_1'',e_2'',x_1'',x_2'')$
and $e_2' < e_2'', x_2' < x_2''$

Then

$(e_1', x_1', C_{11}), (e_1'', x_1'', C_{11}) >^* (e_2'', x_2'', C_{22}), (e_2', x_2', C_{22})$
$\Longrightarrow v(e_1') + v(x_1') + v(C_{11}) - v(e_1'') - v(x_1'') - v(C_{11}) >$
$v(e_2'') + v(x_2'') + v(C_{22}) - v(e_2') - v(x_2') - v(C_{22}),$

By rearranging these terms, we have

$v(e_1') + v(e_2') + v(x_1') + v(x_2') > v(e_1'') + v(e_2'') + v(x_1'')$
$+ v(x_2'').$

If $e_1' > e_1''$, $e_2' < e_2''$, $x_1' < x_1''$, $x_2' < x_2''$
$\Longrightarrow v(e_1') > v(e_1'')$, $v(e_2') < v(e_2'')$, $v(x_1') < v(x_1'')$, and $v(x_2') < v(x_2'')$.

By compensatory independence and mutual preferential independence,

$(e_1', e_2', x_1', x_2') > (e_1'', e_2'', x_1'', x_2'')$ and
$e_2' < e_2''$, $x_1' < x_1''$, $x_2' < x_2''$.

Then

$(e_1', C_1), (e_1'', C_1) >^* (C_{212}, e_2'', x_1'', x_2''), (C_{212}, e_2', x_1', x_2'),$
$\Longrightarrow v(e_1') + v(C_1) - v(e_1'') - v(C_1) > v(C_{212}) + v(e_2'') +$
$v(x_1'') + v(x_2'') - v(C_{212}) - v(e_2') - v(x_1') - v(x_2').$

By rearranging these terms, we have

$v(e_1') + v(e_2') + v(x_1') + v(x_2') > v(e_1'') + v(e_2'') + v(x_1'')$
$+ v(x_2'').$

For other possibilities

$$e_1' > e_1'', e_2' > e_2'', x_1' < x_1'', x_2' > x_2''$$
$$e_1' > e_1'', e_2' < e_2'', x_1' > x_1'', x_2' > x_2''$$
$$e_1' < e_1'', e_2' > e_2'', x_1' > x_1'', x_2' > x_2''$$
$$e_1' > e_1'', e_2' > e_2'', x_1' < x_1'', x_2' < x_2''$$
$$e_1' < e_1'', e_2' < e_2'', x_1' > x_1'', x_2' > x_2''$$
$$e_1' < e_1'', e_2' > e_2'', x_1' > x_1'', x_2' < x_2''$$
$$e_1' < e_1'', e_2' < e_2'', x_1' < x_1'', x_2' > x_2''$$
$$e_1' < e_1'', e_2' > e_2'', x_1' < x_1'', x_2' > x_2''$$
$$e_1' < e_1'', e_2' > e_2'', x_1' < x_1'', x_2' < x_2''$$
$$e_1' < e_1'', e_2' < e_2'', x_1' < x_1'', x_2' > x_2''$$
$$e_1' < e_1'', e_2' < e_2'', x_1' > x_1'', x_2' < x_2''$$

For general n, the proof follows in a similar manner. QED.

5.3 MIXED ADDITIVE CASE USING EPSILON SOLVABILITY

Next, we present an alternative axiom for the mixed additive case. Instead of using compensatory condition, epsilon solvability is defined and used.

5.3.1 DEFINITION OF EPSILON SOLVABILITY

Due to the discreteness of some attributes, there are limited combinations for a set of discrete attributes $(e_1,...,e_n)$.

As a result, it is possible that the complement of an attribute may not be able to achieve strict equality. That is, even though the full magnitude of a change in any given attribute can be offset by its complement, it may not be possible to offset this change exactly. Whenever,

$$(e_1',\ldots,e_i^*,\ldots,e_n',x_1',\ldots,x_m') \geq$$
$$(e_1'',\ldots,e_i'',\ldots,e_n'',x_1'',\ldots,x_m'') \geq$$
$$(e_1',\ldots,e_i,\ldots,e_n',x_1',\ldots,x_m')$$

then there exists an expression

$$(e_1'',\ldots,e_i'',\ldots,e_n'',x_1'',\ldots,x_m'') \approx$$
$$(e_1',\ldots,e_i',\ldots,e_n',x_1',\ldots,x_m')$$

which is used to characterize the restricted solvability and which must be adjusted to;

$$(e_1'',\ldots,e_i'',\ldots,e_n'',x_1'',\ldots,x_m'') \approx\varepsilon$$
$$(e_1',\ldots,e_i',\ldots,e_n',x_1',\ldots,x_m')$$

where ε implies the error in calling the above expression a strict equality due to the discreteness of attribute e_i.

5.3.2 THEOREM

When a mixed attribute set satisfies;

1. Mutual preferential independence
2. Essentialism
3. Weak ordering
4. Archimedean property
5. Epsilon solvability,

then there exists a value function v on e_i and x_j for i=1,...,n and j=1,...,m such that $(e_1',\ldots,e_n',x_1',\ldots,x_m') > (e_1'',\ldots,e_n'',x_1'',\ldots,x_m'')$ iff

$$\sum_{i=1}^{n} v(e_i) + \sum_{j=1}^{m} v(x_j) > \sum_{i=1}^{n} v(e_i) + \sum_{j=1}^{m} v(x_j)$$

Malmborg (1981) has used epsilon solvability for the case of binary additive. The theorem presented here is for the case of mixed additive where discrete and continuous attributes are considered. In the following section, we present a proof of this theorem. The proof closely follows the line of reasoning used by Malmborg (1981) in his proof.

5.3.3 PROOF

Consider an attribute pair E_i, E_j. Let e_i°, $e_i{*} \in E_i$ and e_j, $e_j{*} \in E_j$ where $v(e{*})=1$, $v(e^{o})=0$, $e{*}$ is the highest level and e^{o} is the lowest level. We also define "o" as the concatenation of alternatives by which level of attributes are combined.

If $(e_j{*}oe_i{*}) > (e_j^\circ oe_i^\circ)$ and essentialism holds, then, $v(e_j{*}) > v(e_j^\circ)$, $v(e_i{*}) > v(e_i^\circ)$, and thus

$$v(e_j{*}) + v(e_i{*}) > v(e_j^\circ) + v(e_i^\circ) = 0$$

If $(e_i{*}oe_j^\circ) > (e_i^\circ oe_j{*})$, then we know from double cancellation and entity independence that $e_i{*} > e_j{*}$, which implies that $v(e_i{*}) > v(e_j{*})$. Furthermore, $(e_i{*},e_j^\circ) > (e_i^\circ,e_j{*})$ implies that

$$v(e_i{*}) + v(e_j^\circ) > v(e_i^\circ) + v(e_j{*})$$

which is reduced to

$$v(e_i{*}) + 0 > 0 + v(e_j{*})$$

giving the previous result. This result can be shown for any attribute pair E_i, E_j; $i,j=1,...,n$. Given that $v(e_i)$, $v(e_j)$ are additive for any attribute pair, we can go on to prove additivity for the entire attribute set. To do this, suppose that e_i, $e_i' \varepsilon E_i$; e_j, $e_j' \varepsilon E_j$; e_a, $e_a' \varepsilon \bar{E}_{ij}$; and $(e_i,e_j,e_a) > (e_i',e_j',e_a')$.

It then follows that

$$v(e_i) + v(e_j) + v(e_a) > v(e_i') + v(e_j') + v(e_a') \qquad (1)$$

This is obvious if $e_i > e_i'$, $e_j > e_j'$, and $e_a > e_a'$, so assume that at least one inequality is reversed. We can assume two other possibilities;

$$e_i>e_i',\ e_j>e_j',\ e_a'>e_a \qquad (2)$$

or

$$e_i>e_i',\ e_j'>_j,\ e_a'>e_a \qquad (3)$$

For (2), if $(e_i o e_a) > (e_i' o e_a')$, then $v(e_i) + v(e_a) > v(e_i') + v(e_a')$, and from $e_j>e_j'$, $v(e_j)>v(e_j')$.

The result follows from combining inequalities. We, therefore, assume that $(e_i' o e_a') > (e_i o e_a)$. By epsilon solvability, there exists a $e_b \varepsilon \bar{E}_{ij}$ such that $(e_i' o e_a') \approx_\varepsilon (e_i o e_b)$ (If $\bar{E}_{ij}$ is a continuous attribute, there exist $e_a \varepsilon \bar{E}_{ij}$ instead of e_b). As a result,

$$(e_i o e_j o e_a) > (e_i' o e_j' o e_a') \approx\varepsilon\ (e_i o e_j' o e_b)$$

so that $(e_j o e_a) > (e_j' o e_b)$, which implies;

$$v(e_j) + v(e_a) > v(e_j') + v(e_b) \tag{4}$$

Using the result that $(e_i o e_b) \approx\varepsilon\ (e_i' o e_a')$, suggests

$$v(e_i) + v(e_b) \approx v(e_i') + v(e_a'). \tag{5}$$

Combining Eq. (4) and (5) gives

$$v(e_i)+v(e_b)+v(e_j)+v(e_a) > v(e_i')+v(e_a')+v(e_j')+v(e_b).$$

or

$$v(e_i) + v(e_j) + v(e_a) > v(e_i') + v(e_j') + v(e_a'). \tag{6}$$

For (3), we know that, $e_j'>e_i$ and $e_a'>e_a$ and $(e_i o e_j o e_a') > (e_i o e_j o e_a) > (e_i' o e_j o e_a') > (e_i' o e_j o e_a)$. By epsilon solvability, there exists $e_c\ \varepsilon\ \overline{E}_{ja}$ such that

$$(e_c o e_j o e_a') > (e_i o e_j o e_a) > (e_i' o e_j' o e_a').$$

This suggests that

$$v(e_i) + v(e_a) \approx v(e_c) + v(e_a') \tag{7}$$

and

$$v(e_c) + v(e_j) > v(e_i') + v(e_j') \tag{8}$$

Combining Eqs. (7) and (8) yields

$$v(e_i)+v(e_a)+v(e_c)+v(e_j) \geq v(e_c)+v(e_a')+v(e_i')+v(e_j').$$

or

$$v(e_i) + v(e_j) + v(e_a) \geq v(e_i') + v(e_j') + v(e_a') \tag{9}$$

By induction, these results can be extended to any partition of the attribute set, and thus additivity for the mixed case exists when epsilon solvability holds. QED.

5.4 MIXED MULTILINEAR CASE

The mixed multilinear case is the same as the additive case, except a weaker set of assumptions hold.

5.4.1 THEOREM

For an alternative set where e_i is preferentially independent of $\bar{e}_i$ for i=1,...,n, and x_j is preferentially independent of

$\bar{x}_j$ for j=1,...,m, there exists a multilinear function form across the attribute set. Deutsch and Malmborg (1984) and Keeney and Raiffa (1976) have considered the cases of binary multilinear and continuous multilinear, respectively. The case presented here is a more general case invloving mixed multilinear. A proof of this theoren is presented next, closely following the line of reasoning used by Deutsch and Malmborg (1985) in their proof.

5.4.2 PROOF

From preferential independence, it follows that for any e_i and e_i, the order of preference difference $(e_i', \bar{e}_i)$, $(e_i'', \bar{e}_i)$ depends only on the value of attribute E_i. Thus, the functions $v(e_i', \bar{e}_i')$ and $v(e_i', \bar{e}_i'')$ are strategically equivalent, and as a consequence, $v(e_i', \bar{e}_i')$ is a linear transformation of $v(e_i', \bar{e}_i'')$. Thus,

$$v(e_i', \bar{e}_i') = g(\bar{e}_i') + h(\bar{e}_i')v(e_i', \bar{e}_i''), \text{ for any i.} \quad (1)$$

This is the property referred to by Dyer and Sarin (1979) as conditional cardinality. If $e_i' = e_i^o$ and $\bar{e}_i'' = \bar{e}_i^o$, we then have $g(\bar{e}_i') = v(e_i^o, \bar{e}_i')$.

Substituting this into equation (1) gives

$$v(e_i', \bar{e}_i') = v(e_i^o, \bar{e}_i') + h(\bar{e}_i')v(e_i', \bar{e}_i^o), \text{ for any i.} \quad (2)$$

Equation (2) can be rewritten as

$$v(e') = v(\bar{e}_i') + h(\bar{e}_i')v(e_i'), \text{ for any } i. \qquad (3)$$

Note that $v(e_i') = c_i v_i(e_i')$, and let $d_i(\bar{e}_i') = c_i h(\bar{e}_i')$. This yields:

$$v(e') = v(\bar{e}_i') + d(\bar{e}_i')v_i(e_i'). \qquad (4)$$

Let $e_i' = e_i*$, $v_i(e_i*) = 1$, then

$$d_i(\bar{e}_i') = v(e_i*,\bar{e}_i') - v(e_i^\circ,\bar{e}_i'). \qquad (5)$$

Substituting (5) into (4) and rearranging,

$$\begin{aligned} v(e') = {} & v_i(e_i')v(e_i*,\bar{e}_i') \\ & +[1-v_i(e_i')]v(e_i^\circ,\bar{e}_i'), \text{ for any } i \end{aligned} \qquad (6)$$

Equation (6) with i=1,

$$v(e') = v_1(e_1')v(e_1*,\bar{e}_1') + [1-v_1(e_1')]v(e_1^\circ,\bar{e}_1') \qquad (7)$$

Also equation (6) with i=2,

$$v(e') = v_2(e_2')v(e_2^*,\bar{e}_2') + [1-v_2(e_2')]v(e_2^o,\bar{e}_2') \tag{8}$$

By letting $e_1=e_1^*$ and $e_1=e_1^o$ in equation (8), we have

$$\begin{aligned} v(e_1^*,\bar{e}_1') = {} & [v_2(e_2')v(e_1^*,e_2^*,\bar{e}_{12}') \\ & + [1-v_2(e_2')]v(e_1^*,e_2^o,\bar{e}_{12}')] \end{aligned} \tag{9}$$

and

$$\begin{aligned} v(e_1^o,\bar{e}_1') = {} & [v_2(e_2')v(e_1^o,e_2^*,\bar{e}_{12}') \\ & + [1-v_2(e_2')]v(e_1^o,e_2^o,\bar{e}_{12}')] \end{aligned} \tag{10}$$

Substituting (9) and (10) into (7),

$$\begin{aligned} v(e') &= v_1(e_1')v(e_1^*,\bar{e}_1') + [1-v_1(e_1')]v(e_1^o,\bar{e}_1') \\ &= v_1(e_1') \\ &\quad [v_2(e_2')v(e_1^*,e_2^*,\bar{e}_{12}') + [1-v_2(e_2')]v(e_1^*,e_2^o,\bar{e}_{12}')] \\ &\quad + [1-v_1(e_1')] \\ &\quad [v_2(e_2')v(e_1^o,e_2^*,\bar{e}_{12}') + [1-v_2(e_2')]v(e_1^o,e_2^o,\bar{e}_{12}')] \\ &= v(e_1^o,e_2^o,\bar{e}_{12}') \\ &\quad + [v(e_1^*,e_2^o,\bar{e}_{12}') - v(e_1^o,e_2^o,\bar{e}_{12}')]v_1(e_1') \\ &\quad + [v(e_1^o,e_2^*,\bar{e}_{12}') - v(e_1^o,e_2^o,\bar{e}_{12}')]v_2(e_2') \\ &\quad + [v(e_1^*,e_2^*,\bar{e}_{12}') \\ &\quad - v(e_1^*,e_2^o,\bar{e}_{12}') - v(e_1^o,e_2^*,\bar{e}_{12}') \\ &\quad + v(e_1^o,e_2^o,\bar{e}_{12}')]v_1(e_1')v_2(e_2') \end{aligned}$$

Repeating the procedure for both discrete and continuous attributes, we get the final result as follows:

$$v(e') = \sum_{i=1}^{n} c_i v_i(e_i') + \sum_{i=1}^{n} \sum_{j>i} c_{ij} v_i(e_i') v_j(e_j')$$

$$+ \sum_{i=1}^{n} \sum_{j>i} \sum_{l>j} c_{ijl} v_i(e_i') v_j(e_j') v_l(e_l')$$

$$+ \ldots + c_{123\ldots,n}\, v_1(e_1') \,\ldots\, v_n(e_n')$$

where

$$c_i = v(e_i{*}, \bar{e}_i^\circ)$$

$$c_{ij} = v(e_i{*}, e_j{*}, \bar{e}_{ij}^\circ) - c_i - c_j$$

$$c_{ijl} = v(e_i{*}, e_j{*}, e_l{*}, \bar{e}_{ijl}^\bullet) - c_{ij} - c_{il} - c_{jl} - c_i - c_j - c_l$$

$$c_{1\ldots n} = v(e*) - \sum_i c_{1\ldots(i-1)(i+1)\ldots n} \ldots - \sum_{i,j>i} c_{ij} - \sum_i c_i$$

The proof is therefore completed. QED.

5.5 MIXED MULTIPLICATIVE CASE

The multiplicative case is the case in which a stronger set of assumptions than those for the multilinear case but weaker than the additive case hold.

5.5.1 THEOREM

For an alternative set where e_I is preferentially independent of $\bar{e}_I$, and x_J is preferentially independent of $\bar{x}_J$, for any possible set of I,J, there exists a multiplicative function form across the attribute set.

The cases of binary multiplicative and continuous multiplicative have been discussed by Deutsch and Malmborg (1984) and Keeney and Riaffa (1976), respectively. The one presented here is a more general mixed multiplicative case. A proof of the mixed multiplicative theorem is presented next. The proof closely follows the line of reasoning used by Deutsch and Malmborg (1984) in their proof.

5.5.2 PROOF

From mutual preferential independence it follows that, for any e_I and $\bar{e}_I$, the order of preference differences, $(e_I',\bar{e}_I)$, $(e_I'',\bar{e}_I)$ depends only on the values of the attributes in E_I. Thus, the functions $v(e_I',\bar{e}_I')$ and $v(e_I',\bar{e}_I'')$ are strategically equivalent. It follows that $v(e_I',\bar{e}_I'')$ is a linear transformation of $v(e_I',\bar{e}_I'')$ of the form:

$$v(e_I',\bar{e}_I') = v(\bar{e}_I') + h(\bar{e}_I')v(e_I',\bar{e}_I'') \qquad (1)$$

This is the property referred to by Dyer and Sarin (1979) as condition cardinality. Now, let $e_I' = \bar{e}_I'$ and $e_I' = e_i'$. We can write (1) as

$$v(e_i',\bar{e}_i') = g(e_i') + h(e_i')v(e_i'',\bar{e}_i'), \text{ for any } i. \qquad (2)$$

Assume that $e_i'' = e_i^\circ$, we then obtain

$$v(e') = g(e_i') + h(e_i')v(e_i'), \text{ for any } i, \qquad (3)$$

where $v(\bar{e}_i') = v(e_i^\circ,\bar{e}_i')$.

Setting all $e_i' = e_i^\circ$, except e_1 and e_j, where j-2,...,n, we have

$$\begin{aligned} v(e_1',e_j') &= g(e_1') + h(e_1')v(e_j') \qquad (4) \\ &= g(e_j') + h(e_j')v(e_1'), \end{aligned}$$

which yield $[h(e_1')-1]/g(e_1') = [h(e_j')-1]/g(e_j')$, $j=2,\ldots,n$. From this, if $g(e_i')=0$, then $h(e_i')=1$. It follows that

$$h(e_i') = \lambda\, g(e_i') + 1, \text{ for any } i \qquad (6)$$

We can repeatedly use equation (3) to obtain

$$v(e') = g(e_1') + h(e_1')v(e_1^\circ,\bar{e}_1')$$

$$= g(e_1') + h(e_1')[g(e_2') + h(e_2')v(e^\circ_1, e^\circ_2, \bar{e}_{12}')]]$$

$$= g(e_1') + h(e_1')g(e_2') + \ldots. + \qquad (7)$$

$$h(e_1') \ldots h(e'_{n-1})g(e_n').$$

Substituting (6) into (7) yields

$$v(e') = g(e_1') + [\lambda g(e_1') + 1]\ g(e_2') + \ldots +$$

$$\{[\lambda g(e_1') + 1] \ldots [\lambda g(e'_{n-1}) + 1]\}g(e_n')$$

$$= g(e_1') + \sum_{j=2}^{n} \prod_{i=1}^{j-1} [\lambda g(e_i') + 1]g(e_j') \qquad (8)$$

We then multiply both sides of (8) by λ, and add 1 to each side,

$$\lambda v(e') + 1 = \lambda g(e_1') + 1 + \sum_{j=2}^{n} \prod_{i=1}^{j-1} [\lambda g(e_i') + 1]g(e_i')\lambda$$

$$\lambda v(e') + 1 = \prod_{i=1}^{n} [\lambda g(e_i') + 1] \qquad (9)$$

and $g(e_i') = c_i v_i(e_i')$.

This completes the proof. QED.

5.6 MIXED ADDITIVE INTERDEPENDENT CASE

The result is this section deals with the additive form when attributes have overlapping dependencies.

5.6.1 THEOREM

Let $N=\{1,...,S,S+1,...,n\}$, where $(n\geq3)$, denote a unidimensional attribute set where $i=1,...,S$ are discrete attributes and $i=S+1,...,n$ are continuous attributes. Suppose N is decomposed into the overlapping partition, $Y_1,...,Y_m$, where, $(n>m\geq3)$, and this m-component partition satisfies the following:

1. Mutual preferential independence among partitions
2. Essentialism
3. Weak ordering
4. Archimedean property
5. Epsilon solvability

Then, for all e ,$e' \varepsilon E$, there exist real valued functions $v(Y_1),...,v(Y_m)$, such that

$$e > e' \Longrightarrow v(e) = \sum_{i=1}^{m} v(Y_i) > v(e') = \sum_{i=1}^{m} v(Y_i'),$$

where

$$v(Y_1) = f(Y_1)$$

and

$$v(Y_i) = f(Y_i) + \sum_{k=1}^{i-1} (-1)^k \sum_{1\leq i_1 <...<i_k <i} f(Y_{i_1} \cap ... \cap Y_{i_k}).$$

A theorem for the case of continuous, additive, and interdependent attributes is presented by Deutsch and Malmborg (1983). The one presented here is for the case of mixed, additive, and interdependent attributes. The theorem stated above is proved here by induction. It closely follows the line of reasoning used by Deutsch and Malmborg (1983) in their proof.

5.6.2 PROOF

The proof of the theorem has two parts. The first part is to show that the partition $Y_1,\ldots,Y_m$ is an m-component additive conjoint structure. The result can be proved by the approach used earlier to prove the mixed case with epsilon solvability.

The second part of the proof is to show that the $v(Y_i)$ functions are of the appropriate form. To see this, consider that an n-dimensional alternative can be represented as $(e_1,\ldots,e_n)$, or equivalently as $(Y_1 U \ldots\ldots U\ Y_m)$. Hence, it is sufficient to show

$$v(e) = f_{Y_1 U\ldots UY_m}(Y_1 U\ldots UY_m) = \sum_{i=1}^{m} v(Y_i).$$

From the assumption that

$f(YUZ) = f(Y)+f(Z)-f(Z\cap Y)$, we have, for m=4,

$$f(Y_1 \cup (Y_2 \cup Y_3 \cup Y_4)) = f(Y_1) + f(Y_2 \cup Y_3 \cup Y_4) - f(Y_1 \cap (Y_2 \cup Y_3 \cup Y_4)) \quad (1)$$

$$f(Y_2 \cup (Y_3 \cup Y_4)) = f(Y_2) + f(Y_3 \cup Y_4) - f(Y_2 \cap (Y_3 \cup Y_4)) \quad (2)$$

$$f(Y_3 \cup Y_4) = f(Y_3) + f(Y_4) - f(Y_3 \cap Y_4) \quad (3)$$

Substituting (3) into (2) yields

$$\begin{aligned} f(Y_2 \cup (Y_3 \cup Y_4)) = {} & f(Y_2) + f(Y_3) + f(Y_4) - f(Y_3 \cap Y_4) \quad (4) \\ & - f(Y_2 \cap (Y_3 \cup Y_4)). \end{aligned}$$

Substituting (4) into (1) yields

$$\begin{aligned} f(Y_1 \cup (Y_2 \cup Y_3 \cup Y_4)) = {} & f(Y_1) + f(Y_2) + f(Y_3) + f(Y_4) \quad (5) \\ & - f(Y_2 \cap (Y_3 \cup Y_4)) - f(Y_1 \cap (Y_2 \cup Y_3 \cup Y_4)) \end{aligned}$$

$$\begin{aligned} f(Y_1 \cap (Y_2 \cup Y_3 \cup Y_4)) = {} & f(Y_1 \cap Y_2 \cap Y_3 \cap Y_4) + f(Y_1 \cap Y_2) + f(Y_1 \cap Y_3) \quad (6) \\ & + f(Y_1 \cap Y_4) - f(Y_1 \cap Y_2 \cap Y_3) \\ & - f(Y_1 \cap Y_3 \cap Y_4) - f(Y_1 \cap Y_2 \cap Y_4) \end{aligned}$$

$$\begin{aligned} f(Y_2 \cap (Y_3 \cup Y_4) = {} & f(Y_2 \cup Y_3 \cup Y_4) + f(Y_2 \cap Y_3) - f(Y_2 \cap Y_4) \quad (7) \\ & - 2f(Y_2 \cap Y_3 \cap Y_4) \\ = {} & f(Y_2 \cap Y_3) - f(Y_2 \cap Y_4) - f(Y_2 \cap Y_3 \cap Y_4) \end{aligned}$$

Substituting (6) and (7) into (5),

$$\begin{aligned} f(Y_1 \cup Y_2 \cup Y_3 \cup Y_4) = {} & f(Y_1) + f(Y_2) + f(Y_3) + f(Y_4) \\ & - f(Y_1 \cap Y_2) - f(Y_1 \cap Y_3) - f(Y_1 \cap Y_4) \\ & - f(Y_2 \cap Y_3) - f(Y_2 \cap Y_4) - f(Y_3 \cap Y_4) \\ & + f(Y_1 \cap Y_2 \cap Y_3) + f(Y_1 \cap Y_3 \cap Y_4) \\ & + f(Y_1 \cap Y_2 \cap Y_4) + f(Y_2 \cap Y_3 \cap Y_4) \\ & - f(Y_1 \cap Y_2 \cap Y_3 \cap Y_4) \end{aligned}$$

$$\begin{aligned} = {} & \sum_{i=1}^{4} f(Y_i) - \sum_{1 \leq i_1 < i_2} f(Y_{i_1} \cap Y_{i_2}) \\ & + \sum_{1 \leq i_1 < i_2 < i_3} f(Y_{i_1} \cap Y_{i_2} \cap Y_{i_3}) \\ & - f(Y_1 \cap Y_2 \cap Y_3 \cap Y_4). \end{aligned}$$

By induction, we can show that, for any m,

$$f(Y_1 \cup \ldots \cup Y_m)$$

$$= \sum_{i=1}^{m} f(Y_i) - \sum_{1 \leq i_1 < i_2} f(Y_{i_1} \cap Y_{i_2}) \ldots (-1)^{m-1} f(Y_m \cap \ldots \cap Y_m).$$

Since $v(Y_1) = f(Y_1)$, and

$$v(Y_i) = f(Y_i) + \sum_{k=1}^{i-1} (-1)^k \sum_{1 \leq i_1 < \ldots < i_k < i} f(Y_{i_1} \cap \ldots \cap Y_{i_k}),$$

then

$$\sum_{i=1}^{m} v(e_i) = \sum_{i=1}^{m} f(Y_i)$$

$$- \sum_{1\leq i_1 < i_2} f(Y_{i_1} \cap Y_{i_2}) \ldots (-1)^{m-1} f(Y_1 \cap \ldots \cap Y_m).$$

Thus, $f(Y_1 U \ldots U Y_m) = \sum_{i=1}^{m} v(Y_i)$

This completes the proof. QED.

5.7 SUMMARY AND CONCLUSION

In this chapter, we have presented results for the mixed, discrete and continuous attributes. Appropriate sets of conditions are used to verify the aggregation of the attribute set. The axiomatic system presented here is an application of Deutsch and Malmborg (1984). It is generalized here so that, now, it can be used for any case, namely, continuous, binary, discrete, or mixed cases. This chapter does not attempt to provide procedures for the verification of the conditions used, but presents results which underly appropriate function forms.

6. METHODOLOGY FOR CREATING LAYOUT ALTERNATIVES

In this chapter the methodology of using closeness preference ratings to determine relative department locations is described. First, the layout design problem is formulated as a quadratic assignment problem and some solution procedures suggested in the literature for solving the quadratic assignment problem are reviewed. Then, a heuristic procedure based on the solution of a small size quadratic assignment problem is described to obtain an initial layout which gives locations of various departments. This initial layout is used as a guideline to obtain the final layout by taking into consideration the actual department sizes. The process of constructing the final layout is also described. An example problem is presented to demonstrate the proposed procedure. CORELAP and ALDEP are also applied to the same problem and the results obtained from all three procedures are compared. A detailed comparison between the results obtained using CORELAP and the proposed procedure is then presented. The proposed procedure is shown to consistently give better layouts than those obtained using CORELAP.

6.1 LAYOUT DESIGN AS A QUADRATIC ASSIGNMENT PROBLEM

In the previous chapter we described a methodology of measuring accurate closeness ratings. Now, we develop a procedure for layout construction based on the objective of maximizing closeness ratings. The problem of determining an optimal layout can be formulated as a quadratic assignment problem (QAP) as follows:

$$\text{Max} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} c_{ijkl} x_{ij} x_{kl}$$

Subject to

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

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

where

x_{ij} = 1, when department i is assigned to location j

= 0, otherwise

n = total number of departments

p_{ik} = closeness rating between departments i and k

d_{jl} = rectilinear distance between locations j and l

and

$c_{ijkl} = p_{ik}/d_{jl}$. The symbol, c_{ijkl}, represents the closeness rating between a department pair i,k, depending upon the rectilinear distance between the locations that they are assigned to. The greater the distance between departments, the smaller is the closeness rating between them.

In the above formulation, the objective function represents total closeness rating between departments. The first constraint ensures that only a single department is assigned to a location and the second constraint ensures that a department is assigned to only one location. This formulation assumes that the departments are of the same size. In case the departments are of different sizes, the above formulation becomes

$$\text{Max} \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{n} \sum_{l=1}^{m} c_{ijkl} x_{ij} x_{kl}$$

Subject to

$$\sum_{j=1}^{n} x_{ij} = s_i \qquad i=1,\ldots,n$$

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

where,

$x_{ij} = 1$, when department i is assigned to location j

= 0, otherwise

s_i = number of blocks for department i

m = total number of blocks

and

$c_{ijkl} = p_{ik}/(d_{jl}s_is_k)$ = closeness rating between a department pair i,k.

6.2 LITERATURE REVIEW OF PROCEDURES TO SOLVE THE QAP

Generally, the quadratic assignment problem (QAP) seeks to assign n departments to n mutually exclusive locations, so as to minimize the total quadratic interaction term. In our case, the objective is that of maximization. The QAP was first formulated when Koopmans and Beckmann (1957) studied problems concerning the allocation of plants to sites. Subsequently, considerable work has been done by mathematicians and engineers alike on problems of a similar nature. However, the problem still remains of considerable interest due to the variety of possible solution approaches and the number of applications it has in various fields.

The nonlinearity and combinatorial nature of the problem creates difficulties in developing efficient solution techniques. The number of possible permutations for the problem grows exponentially with an increment in the value of n. Exhaustive enumeration for even a modest size problem is computationally infeasible. For example, Nugent, et. al. (1968) estimate that for n=12 (for which n!=4.79*10) the total enumeration on an IBM 7090 computer would require three years of cpu time.

Solution methodologies for the QAP can be classified under two categories, namely, exact and heuristic methods. Exact methods are those which determine the optimal QAP solution. Heuristic methods generally determine a suboptimal solution.

6.2.1 EXACT METHODS EMPLOYING BRANCH AND BOUND

These are exact procedures for assigning n departments to n locations by enumerating, in a controlled manner, all the possible location-department permutations. Many of the permutations are handled implicitly, thus reducing computational effort.

The general method employed is as follows. Starting from node zero at which no department is fixed, branches are created to systematically partition the solution space. Each branch represents the allocation of a department to a location or vice versa and it terminates in a node. A node in the enumeration tree represents a subproblem at which the solution is restricted to satisfy the conditions imposed on the branches connecting it to node zero. Thus, a path from node zero to the bottom of the tree prescribes allocation of departments to locations. Once a department or a location is assigned to any particular node it is labelled. The unassigned departments or locations are label free. Creating branches until all the departments are fixed to all possible locations, is equivalent to enumerating all the location-department permutations. However, this is avoided by computing lower bounds (upper bound for the maximization problem). This is defined as follows. The lower bound (Upper bound) at any partial placement (corresponding to a node) is a value which indicates that every possible combination of the free departments will yield an objective value higher (lower) than this value in case of the minimization (maximization) problem.

The lower bound (upper bound) at any partial placement can be broken down into three components:

1. A value b_1 which gives the interaction cost (closeness rating) between the assigned department themselves.

2. A value b_2 which a lower bound (upper bound) on the interaction between the assigned departments and the unassigned departments.

3. A value b_3 which is a lower bound (upper bound) on the interaction between the unassigned departments themselves.

Thus, if the objective function value of some known feasible complete placement is lower (higher) than the lower bound (upper bound) of a partial placement, then the node corresponding to this partial placement, and its descendants, can be eliminated from further consideration. This process of eliminating further branches from a node is termed fathoming the node. The best known objective function value of a complete placement is referred to as the upper bound (lower bound) or the incumbent value. This underlies the importance of lower bounds in a branch and bound procedure. Better lower bounds (upper bounds) fathom a greater portion of the enumeration tree thereby reducing the computational effort. The various exact methods employing branch and bound presented in the literature differ primarily in the methods used for branching and for the computations of bounds. Next, we briefly describe these methods.

A class of methods proposed for QAP is termed single assignment algorithms. All algorithms in this class deal with assigning a particular department to a location at each node. The algorithms vary primarily in the method of computing the lower bound, and the branching procedure.

Gilmore (1962) presented an algorithm which is a semi-enumerative one. The algorithm associates a particular location or department to each level in the branching process. For each location, the distances from all other locations are computed and they are arranged in a non-increasing order. The departments/locations are then assigned to each level in this order. The algorithm starts from the top of the tree assuming an initial solution. Search is performed using the depth first strategy. Branches are fathomed when lower bounds are found to be greater than the incumbent solution. The lower bounds are calculated by the previously described method. The search backtracks to the next higher level and resumes. The efficiency of the algorithm depends largely upon the initial solution. The algorithm is not computationally feasible for n larger than 15.

Lawler (1963) presented a technique to calculate the lower bound of the cost function based on replacing the binary variable $x_{ij}= 0$ or 1 by $x_{ij}\geq 0$. The resulting problem is then solved by a quadratic programming technique. The branching process is the same as Gilmore's.

Graves and Whinston (1970) identified the statistical properties of the cost function. They used these statistical properties in conjunction with a general enumerative procedure to define a solution algorithm. Using the idea of confidence level enumeration, this algorithm allows for an effective treatment of QAP. The branching process is basically the same as Gilmore's.

Unlike the single assignment algorithms, pair assignment algorithms assign pairs of departments, i and j, to locations k and p. Land (1963), Gavett and Plyter (1966), and Pierce and Crawston (1971) have developed algorithms in this regard. For the case of n departments and n locations, there are n(n-1)/2 pairs of departments and n(n-1)/2 pairs of locations. Thus, given four departments A, B, C and D, and four locations 1, 2, 3 and 4, there are the following possible pairs of departments and locations:

Department pairs: (A,B) (A,C) (A,D) (B,C) (B,D) (C,D)

Location pairs: (1,2) (1,3) (1,4) (2,3) (2,4) (3,4)

A feasible assignment of these pairs does not necessarily provide a feasible solution to the original problem. To illustrate this, suppose department pair (A,B) is assigned to location pair (1,2) and the department pair (A,C) is assigned to location pair (3,4). This is not a feasible solution since department A is assigned to two locations. Hence, additional constraints are introduced in the linear pair assignment problem so that a solution to this linear assignment problem gives a feasible solution to the original problem. The solution technique proceeds level by level by fixing a new pair of departments to a pair of locations.

Land (1963) used a simple assignment matrix to establish the cost lower bounds on the branches of the tree. However, the method involved enumeration of the set of n! feasible solutions, a task which becomes impossible even for moderate values of n. Gavett and Plyter (1966) also used simple assignment matrix but instead of employing only a column-reduced matrix, they used a row and column-reduced matrix to establish the lower bounds. Both algorithms proceed level by level in the search tree, committing one new pair to the solution at each level, and backtracking to the next higher level in the search tree. Pierce and Crawston (1971) introduced the pair-exclusive algorithm method. This method eliminates some of the feasible solutions at each stage of the stepwise process. The solution procedure uses a technique similar to that of the pair assignment algorithm to determine pair assignments. If the relaxed problem of assigning pairs produces an infeasible assignment, then some conflicting assignments exist which should not be part of the optimal solution. The search tree then branches into as many nodes as there are conflicting assignments, removing one conflicting assignment at each node created. Having branched, lower bounds are computed at each descendent node. For this purpose, an assignment problem with the additional pair exclusion constraints is solved at each node. If this does not result in a feasible solution, and if the lower bound does not exceed the upper bound at any node, then further branching is necessary. However, if a feasible solution is obtained, then that node is fathomed. At any stage, the procedure selects another node with the lowest bound and proceeds as before. When all nodes are fathomed the algorithm terminates.

The branch and bound methods have been found to be computationally feasible only for problems of size twelve or less number of departments. Consequently, other methods have also been developed which we discuss next.

6.2.2 EXACT METHODS NOT EMPLOYING BRANCH AND BOUND

These approaches use techniques such as integer programming, quadratic integer programming, cutting plane algorithms and decomposition principles to solve the QAP.

First, Cabot and Francis (1970) showed that for the QAP with linear constraints, the optimal solution is the extreme point of the convex set of feasible solutions. They presented a procedure for solving such a problem. Their methodology involved first determining a related linear program containing the original set of constraints. The extreme-point-ranking approach of Murty (1968) was then applied to this linear program to obtain an optimum solution to the QAP.

Love and Wong (1976) formulated the QAP as a binary mixed integer program. This formulation is as follows.

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}(R_{ij}+L_{ij}+A_{ij}+B_{ij})$$

Subject to

$$R_{ij}-L_{ij} = x_i - x_j \qquad i=1,\ldots,n-1$$

$$A_{ij}-B_{ij} = y_i - y_j \qquad j=i+1 \qquad .$$

$$x_i + y_i = \sum_{k=1}^{n} s_k a_{ik} \qquad i=1,\ldots,n$$

$$x_i - y_i = \sum_{k=1}^{n} d_k a_{ik} \qquad i=1,\ldots,n$$

$$\sum_{k=1}^{n} a_{ik} = 1 \qquad i=1,\ldots,n$$

$$\sum_{i=1}^{n} a_{ik} = 1 \qquad k=1,\ldots,n$$

where a_{ik}= 1 when department i is assigned to location k

= 0, otherwise

n = number of departments

w_{ij} = flow (interaction) between i and j

R_{ij} = horizontal distance between departments i and j if i is to the right of j, and is zero otherwise

L_{ij} = horizontal distance between departments i and j if i is to the left of j, and is zero otherwise

A_{ij} = vertical distance between departments i and j if i is to above of j, and is zero otherwise

B_{ij} = vertical distance between departments i and j if i is to below of j, and is zero otherwise

(x_i, y_i) = location of department i, i=1,...,n

s_k = sum of coordinates of location k

d_k = difference of coordinates of location k

The first four constraint types guarantee that none of the departments are overlapped. The last two constraint types ensure that one department must be assigned to only one location. This formulation has n^2 binary variables and (n^2+3n) constraints. It can be solved using integer programming codes. However, it is restricted to relatively small problems ($n \leq 9$) due to the number of constraints. It has the advantage of not requiring quadratic assignment code.

Another well known method used for solving QAP involves the construction of an equivalent, but much larger linear assignment problem with several side constraints. The drawback of this method lies in the weakness of the bounds obtained by solving the linear problem. Kaufman and Broechx (1978) suggest an alternate linearization using Glover's linearization technique. The QAP then reduces to the following linear problem:

$$\text{Min} \sum_{i=1}^{n} \sum_{k=1}^{n} w_{ik}$$

Such that

$$\sum_{i=1}^{n} x_{ik} = 1 \qquad k=1,\dots,n$$

$$\sum_{k=1}^{n} x_{ik} = 1 \qquad i=1,\dots,n$$

$$C_{ik}x_{ik} + \sum_{j=1}^{n} \sum_{i=1}^{n} f_{ij}d_{kl}x_{ik} - w_{ik} \leq C_{ik}$$

$$x_{ik} \in \{0,1\}$$

$$w_{ik} \geq 0$$

where

$$w_{ik} = x_{ik} \sum_{j=1}^{n} \sum_{l=1}^{n} f_{ij}d_{kl}x_{jl}$$

$$C_{ik} = \text{MAX}\left(\sum_{j=1}^{n} \sum_{l=1}^{n} f_{ij}d_{kl}, 0\right)$$

w_{ik} = total cost of assigning department i to location k.

f_{ij} = interaction between department i and department j.

d_{kl} = distance between location k and location l.

x_{ik} = 1, department i is assigned to location j

= 0, otherwise.

The third constraint of the formulation ensures that any assignment of department i to location k results in minimum of total cost.

Glover's method introduces n new continuous variables and n new constraints. The problem can be solved using a code for mixed integer programming. Furthermore, solution method is suggested for the linearized QAP which is based on Bender's decomposition.

Bazaraa and Sherali (1980) consider a formulation similar to that of Kaufman and Broeckz (1978). They applied Bender's partitioning scheme to solve the model. This approach decomposes the model into a linear integer master problem and a linear subproblem. At each iteration, the master problem generates a point X_A, based on which a suitable subproblem is solved to generate a cut-constraint. This cut is then appended to the other constraints in the master problem and the solution to the latter is updated. The procedure hence alternates between the master problem and the subproblem until a suitable termination criterion is met in a finite number of steps.

6.2.3 HEURISTIC METHODS

A heuristic method is a common sense approach to provide approximate solutions to problems. They remain the only effective methods of solving large sized problems which exist in reality. A heuristic, however, is not an exact approach and it is usually not possible to guarantee or verify optimality. Typically, solutions are closely sub-optimal.

The heuristics for QAP existing in the literature can be classified as follows.

1. Construction procedures - These methods select departments to be placed on locations one-at-a-time sequentially. This continues until all departments have been located using certain rules. One method in this regard suggested by Rajgopal (1985) is summarized below.

Let (D,L) be a partial assignment where L represents a set of locations and D(L) represents the departments that are assigned to the locations in set L. Let

$$A_i = \sum_{p \text{ not} \varepsilon\ L} (d_{ip}+d_{pi})$$

$$B_j = \sum_{q \text{ not} \varepsilon\ D(L)} (p_{jq}+p_{qj})$$

where d_{ip} = distance between locations i and p

p_{jq} = interaction between department j and q

Then assign an index i with minimal value A_i to an index j with maximal value B_j. The procedure is repeated until all departments have been assigned.

2. Improvement procedures - These methods start with a feasible arrangement of departments to locations, and then systematically improve that solution. A method in this regard is to perform a pairwise interchange between departments.

Hillier (1963) used this method for QAPs with the grid structure. According to this method the maximum movement of a department either, left, up or down is first determined. This is termed the maximum number of steps, p. The p step movements of all the departments is considered and the cost of all these p step movements to their corresponding sites is determined without the counter movement of the department occupying the site. The interchange of departments is considered with the department that offers the maximum cost reduction. The interchange is implemented only if it results in a decreased cost. The move numbers are then updated and the procedure continues until no more p step improvements are possible. Next p is decreased by one and the process is repeated. This procedure was further improved by Hillier and Connors (1966) by considering in the preliminary moves only those improvements which decreased the cost by at least a specified amount.

Armour, Buffa and Vollman (1963) developed the well known method of CRAFT. From an arbitrary starting solution, they computed the decrease in cost by interchanging each pair of departments. Only those pair interchanges that resulted in the minimum cost were implemented. This method terminates when no more improvement is possible.

Vollman, Nugent, and Zartler (1966) suggested the following two phase technique which was a considerable improvement over CRAFT. In the first phase, the two facilities which contributed most to the cost were selected. All pairwise interchanges with one of the chosen departments were computed and the largest reduction was implemented. The department interchanged was then disregarded from further consideration. When no further improvements were possible with the first department chosen, the procedure was repeated with the second department. Again two departments with the greatest contributions to cost were chosen and the process was repeated. This continued until no further improvements resulted. The procedure then switched over to the second phase, where all possible pairwise interchanges were attempted. The entire procedure was repeated once more and then terminated.

Pagels (1966) used a discrete optimization technique adopted by Reiter and Sherman (1965) to develop a heuristic. All possible pairwise interchanges were done on a random layout giving a local (2-opt) minimum. This was used to generate a second starting layout giving a second local (2-opt) minimum. The user was then asked to provide a value for the probability of not finding the optimal solution at termination, this was then used to determine the number of iterations to be performed.

Los (1978) proposed the following technique to improve upon CRAFT. All possible exchanges were evaluated only once. At each exchange the computed exchange values were updated recursively, thus reducing the computational effort for all subsequent iterations.

Patel, Dewald, and Cote (1976) used the following concept to reduce computational effort. They created algorithms to partition the departments into groups of departments in such a way that the interactions between the groups was minimized. It was desirable to have an equal number of departments in each group. Highly interconnected departments were put together and referred as groups or supernodes. This was done until the required number of supernodes were formed. The locations were then divided into superlocations and supernodes were allotted to superlocations and improved by pairwise interchange. After this placement, pairwise interchange was conducted within the elements of each supernode only, thus reducing the computational effort considerably.

Other improvement schemes are more complex than the pairwise interchange technique described before. They are the combinations of several other methods. For example, Burkard and Stratman (1978) alternated between a branch and bound approach and an improvement exchange routine. Bazaraa and Sherali (1980,1982) implemented Benders partitioning method, and a disjunctive cutting plane method on a mixed integer formulation of the QAP. The details of some of these procedures are discussed next.

The heuristic of Gaschutz and Ahren's (1968) imbeds the problem into a graph theoretic framework in which the nodes represent the departments. In this graph, departments with a positive interaction are connected by links having weights proportional to the level of interaction. A continuous relaxation technique is then employed to determine coordinates of the nodes such that the sum of weighted square deviations of the nodes from each other is minimized. This is subject to a normalization constraint that the sum of the x and the sum of the y coordinates should each equal zero, and the sum of their squares should each equal unity. An iterative scheme is used to solve this problem. The resulting coordinates are then used to match the departments onto the location grid.

Burkard and Stratmann (1978) alternated between a branch and bound approach and an improvement exchange routine. Their method called VERBES, terminate when no further improvement is possible. This method yielded very good results.

Bazaraa and Sherali (1980,1982) were able to obtain solutions of equivalent and better quality than those obtained by Burkard and Stratmann (1978) for the test problems. They implemented Benders partitioning method, and a disjunctive cutting plane method on a mixed integer formulation of the QAP. They empirically demonstrated the prohibitiveness of using these methods as exact algorithms, and developed theoretical lower bounds on computational complexity in some cases. They then converted these algorithms to heuristic procedures which were able to produce better quality solutions on test problems than any of the previous methods.

Mirchandani and Obata (1979) also employed Gaschutz and Ahrens (1968) method along with Hall's (1970) algorithm for solving the squared deviation problem as a preprocessor in a rotating grid algorithm. In their method, several trial solutions were obtained by attempting to match the locations of the placement-grid as it rotated into several positions superimposed upon Gaschutz and Ahrens' solution. This matching was performed by using an assignment problem to obtain a least deviation from the Gaschutz and Ahrens' solution. A pairwise exchange was then performed over each resulting solution. In addition, the mixed-exchange scheme discussed earlier was used to improve the solution.

Rajgopal (1985) divided the problem of n departments/locations into several subproblem of size less than six departments/locations. Each subproblem was then solved using a branch and bound method. Later, a pairwise interchange routine is performed on the overall problem. The method is a polynomial time heuristic with respect to the size of the problem.

6.3 A SOLUTION PROCEDURE FOR THE DETERMINATION OF THE INITIAL LAYOUT

In this section we develop a new heuristic solution procedure for the layout problem formulated as a QAP. Most of the exact algorithms presented so far in the literature, for the solution of QAP, do well for small problems. These procedures require substantial computer programming memory and computer running time for a realistic size problem. In contrast, heuristic methods require less computer memory and computation time. The principal aim of the heuristic developed here is to provide a good solution in a reasonable amount of time when run on a personal computer system.

The proposed heuristic is of a construction type and can be coupled with either a pairwise interchange routine or another improvement routine described below to improve solution quality. The following construction routines can be used to build a starting solution which can subsequently be subjected to improvement.

Construction routine 1:

Let (D,L) represent a partial assignment where L represents a locations set and D(L) represents the departments set which are assigned to the locations set L. Let

$$A_i = \sum_{p \text{ not}\varepsilon\ L} (d_{ip}+d_{pi})$$

$$B_j = \sum_{q \text{ not}\varepsilon\ D(L)} (p_{jq}+p_{qj})$$

where $i \notin L$, $j \notin D(L)$. Assign location i with minimal value A_i to department j with maximal value B_j. Then replace L by L U {i} and compute new values of A_i and B_j. The process continues until all departments have been assigned to locations. This routine is similar to that suggested by Rajgopal (1985) and described in section 6.2.3 except that the sets A_i and B_j are updated here whenever a department is assigned to a location.

Construction routine 2:

Let k represent the order of department/location to be introduced into the layout. The first step is performed as in construction routine 1 to find the first assignment. Let k=1. If an i_1 was assigned to a j_1, then, at the next step, assign an i with minimal$(d_{ii_1}+d_{i_1i})$ to a j with maximal$(p_{jj_1}+p_{j_1j})$. Set k=k+1. Then, i_1 and j_1 are replaced by i_k and j_k and the process continues until all departments have been assigned to locations.

Construction routine 3:

Let (D,L) represent a partial assignment where L and D are as defined earlier. Also determine A_i and B_j as before. That is,

$$A_i = \sum_{p \text{ not} \varepsilon \ L} (d_{ip}+d_{pi})$$

$$B_j = \sum_{q \text{ not} \varepsilon \ D(L)} (p_{jq}+p_{qj}),$$

where $i \ \varepsilon \ L$, $j \ \varepsilon \ D(L)$. Rank A_i in the increasing order of its values. Assign location i with minimal value A_i to department j with maximal value B_j. Make $L = L \cup \{i\}$ and update

D(L). Select the next higher A_i value from the ordered list and assign a department $j \varepsilon \{1,...,n\}$-D(L) to this i such that

$$\sum_{p \varepsilon L} p_j / d_{ip}$$

is maximal. Denote the department j which is assigned to location i by D(i). The process repeats until all departments/locations have been assigned.

These three construction algorithms are used to create initial layouts. Each of them is designed to accommodate various types of preference matrices. Construction algorithm 1 gives better solutions when the positive entries in the preference matrix are not much different from each other, as this method considers aggregate rating of a department with others. Construction algorithm 2 gives better solutions when only a few departments are dominating and have large values of the corresponding entries in the preference matrix, since it searches for a department pair with the highest preference value among those available and puts those departments next to each other. In the case where several entries in the preference matrix are sufficiently larger than the others, construction algorithm 3 gives better solutions, since it selects a department to enter the layout which maximizes the total preference value of the layout obtained so far.

Improvement routine:

The improvement routine suggested here is designed to produce a good solution in a reasonable amount of time. Since the routine is implemented on a microcomputer system, the computer memory and computation speed are two major concerns.

The improvement routine is a blend of solving a subproblem using the exact method and alternating it with a pairwise interchange routine described below. The size of the subproblem for the exact method is arbitrarily chosen here to be 5, due to the concerns of computation time and computer memory for larger subproblem size. The details of the improvement routine are described next.

Let L_d be an ordered list of locations corresponding to the values of A_i and n be the number of locations under consideration (which is also equal to the number of departments).

1. Initialize $S=L_d$, $n_s=n$ and denote by f(i) the facility which is currently fixed on location site i, i=1,...,n.

2. Let $n_d=\min\{n_s,5\}$ and denote by S_d the top n_d locations in S. Accordingly, define $S_f= \{p:p=f(i) \text{ for } i \in S_d\}$. Solve for an exact solution of $QAP(S_f,S_d)$. Update f(i), $i \in S_d$. The QAP is solved here using exhaustive search as the subproblem size is relatively small, namely 5.

In case a larger subproblem size is selected, the QAP can be solved by a more effective method.

3. Set $n_s=n_s-k$. That is, consider the next subproblem from among the n_s-k locations where the assignment of departments to the first k locations is fixed. A large value of k will give relatively small overlap among the subproblems, while a small value of k will result in relatively large computation time. A value of k=2 was found to be a good compromise. If n ≤4, then stop. Otherwise, remove the top two locations from S and go to step (2).

The solution obtained above can further be improved using a pairwise interchange routine described in the following section.

Pairwise interchange routine:

Let f(i)=j for i=1,...,n and j=1,...,n as an initial solution, where f(i)=j represents facility j being assigned to location i.

1. Let i_1,i_2 represent a location pair.

2. Set $i_1=1$, $i_2=2$. Calculate total preference. Let $k=i_2$.

3. Interchange locations of facilities $f(i_1)$ and $f(i_2)$. Calculate total preference. If the new total preference is greater than the previous one, the interchange is successful. Otherwise, go to the next step.

4. If i_1=n-1, i_2=n, then terminate the process.

5. If i_2=n, then go to step (6). Otherwise, $i_2=i_2+1$, and go to step (3).

6. Set $i_1=i_1+1$, i_2=k+1, k=i_2 and go to step (3).

Implementation of the Proposed Procedure to the Case of Unequal Department Size:

Next, we discuss the application of the above procedure to the case of unequal department sizes. As was discussed earlier in section 6.1, unequal departments can be represented as several blocks of a basic unit size. If the above heuristic method is directly applied to the blocks corresponding to each department, it is possible that the final solution may give an infeasible solution. That is, blocks of the same department may not be adjacent. In order to force the blocks of the same department to be adjacent, a penalty function concept can be applied. That is, a very large value of c_{ijkl} is added in the objective function as a penalty among blocks of the same department, to force them to be adjacent. Consequently, the formulation is as follows:

$$\text{Max} \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{n} \sum_{l=1}^{m} c_{ijkl} x_{ij} x_{kl}$$

Such that

$$\sum_{j=1}^{m} x_{ij} = s_i \qquad i=1,\ldots,n$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j=1,\ldots,m$$

where

x_{ij} = 1, when department i is assigned to location j

= 0, otherwise

and

$c_{ijkl} = p_{ik} /(d_{jl} s_i s_k)$ = closeness rating for the department pair i,k; this is a function of preference rating and rectilinear distance of relative locations; c_{ijkl} = M (a very large value), when i=k.

p_{ik} = closeness rating between departments i and k

d_{jl} = rectilinear distance between locations j and l

s_i = number of blocks for department i

m = number of total blocks

n = number of departments

By selecting an appropriate value of M, the proposed method can be applied effectively. A modification needs to be made in the implementation of the proposed method. In the proposed method, a set of five locations and their corresponding departments are chosen and are rearranged by solving the QAP. In the case of departments of unequal sizes, it is possible that only a subset of blocks belonging to a department are contained in the selected set of five locations. Consequently, in the improvement routine, the subset of blocks of a department, that are contained in the selected set, will not move from their current locations due to a large value of closeness rating, c_{ijkl}, between them and their counterparts outside the selected set. A modification is therefore made in the algorithm to either include or exclude blocks of the same department in the selected set. A step by step strategy for the selection of the location set is as follows.

Step 1: Set n=1.

Step 2: The n^{th} location in the set is chosen as described in the improvement routine.

Step 3: Check whether more than one block is required for the department in the n^{th} location. If so, go to step 4. Otherwise, set n=n+1 and repeat step 3 until the location set is filled.

Step 4: Assume that the department selected requires m blocks. If m is greater than the number of locations remaining in the location set, go to step 2 and select the next location on the ordered list as described in the improvement routine; otherwise, go to step 5.

Step 5: Assign the locations corresponding to the selected department to the location set. Set n=n+m, then go to step 2.

By applying the above strategy, we can assure that, when the QAP is solved, all blocks are free to move. This yields a better solution than obtained by directly applying the proposed routine to the case of unequal size departments.

6.4 A PROCEDURE FOR DETERMINING THE FINAL LAYOUT

The layout determined by the method of section 6.3 is termed an initial layout or blockplan because it only gives locations of various departments. From this we need to construct the final layout, taking into consideration the actual department sizes. A procedure to do that is as follows.

Step 1: Take the department from the first location of row one in the blockplan and locate it in the upper left hand corner of the actual layout plan. Take the department from the second location of row one in the blockplan and place it next to the previously located department. The same process continues until departments in the first row of the blockplan are all placed. If it occurs that departments cannot fit in the layout, go to step 2, otherwise, continue the same process for row two, and so on until all departments are placed in the final layout or a department cannot fit in the layout. In which case, go to step 2.

Step 2: Check whether an acceptable adjustment in the shape of the department can be made so that the department can now fit in the layout. If so, alternate the department shape and rearrange it and go back to step 1; otherwise, go to step 3.

Step 3: Either remove the first or the last department of the row in consideration and rearrange the rest of the departments in that row to fit in the layout. Place the first or the last department (whichever selected) below the second or the second last department, respectively. If it is feasible to do so, then continue with step 1 in the next row; otherwise, discard the blockplan and try other blockplans.

These steps provide a general guideline to convert an initial layout into a final layout. However, some layout problems involve departments of various sizes in which case the differences in department sizes should be considered apriori in the generation of the initial layout as described earlier in this chapter. This strategy reduces conflicts in locating departments later on in the final layout creation process.

6.5 THE OVERALL LAYOUT DESIGN PROCEDURE

The overall layout design procedure can now be summarized as is depicted in Figure 6.1. It consists of four phases, namely, data acquisition, initial layout creation, final layout creation, and overall decision. A decision support system is developed based on these phases. Figure 6.1 depicts the layout design phases as well as the corresponding computer packages for the decision support system.

The data acquisition phase is intended to gather all the necessary information for the layout problem under consideration. This information, for instance, pertains to facility specification and criteria rating. The computer package corresponding to this phase is an interactive data acquisition package. The package interacts with the designer in assessing preference functions, acquiring facility information, and in preparing the preference matrix to be used in the next phase.

The initial layout creation phase utilizes the preference matrix generated in the data acquisition phase to create a layout depicting relative department locations. These relative department locations are used in the final layout creation phase to determine a layout based on actual department sizes. In the initial layout creation phase the layout problem is formulated as a quadratic assignment problem. A heuristic procedure that is based on a subset of the QAP is developed for the creation of the initial layout. The final layout creation phase is interactive in nature and permits the user to use subjective criteria that may not have been included in the analysis and also his or her insight to create or fine tune the final layout. The computer package allows the designer to manipulate department locations and orientations and to compute layout scores while the layout is being adjusted. The listing of the program is included in Appendix A.

The overall evaluation phase uses the preference scores corresponding to the layouts created, computer ratings for other criteria besides the closeness related criteria, and then aggregates these scores into a single value for each layout.

A layout with the highest value is then selected. A listing of the complete computer package is shown in Appendix A.

6.6 AN EXAMPLE PROBLEM

Next, we describe an example problem. The purposes of presenting this example is three fold: (1) to illustrate the generation of the initial and the final layouts by the proposed procedure, (2) to show that the creation of initial layouts by maximizing a multicriteria function yields better solutions than those obtained by maximizing only a single criterion, and (3) to compare the final layout to that obtained using CORELAP and ALDEP.

6.6.1 DESCRIPTION OF THE EXAMPLE PROBLEM

This example problem is adapted from Francis and White (1974). The town council of Blacksburg has decided to construct a new municipal building to replace the old building. The problem is to locate departments in a given space. A study has been made and flow data for the number of daily interdepartmental personnel contacts are collected. Besides, the hypothesized closeness related criteria which includes common personnel, common data used, contact necessary, and convenience to visitors are evaluated. Also, scaling constants and marginal value functions which represent impor-

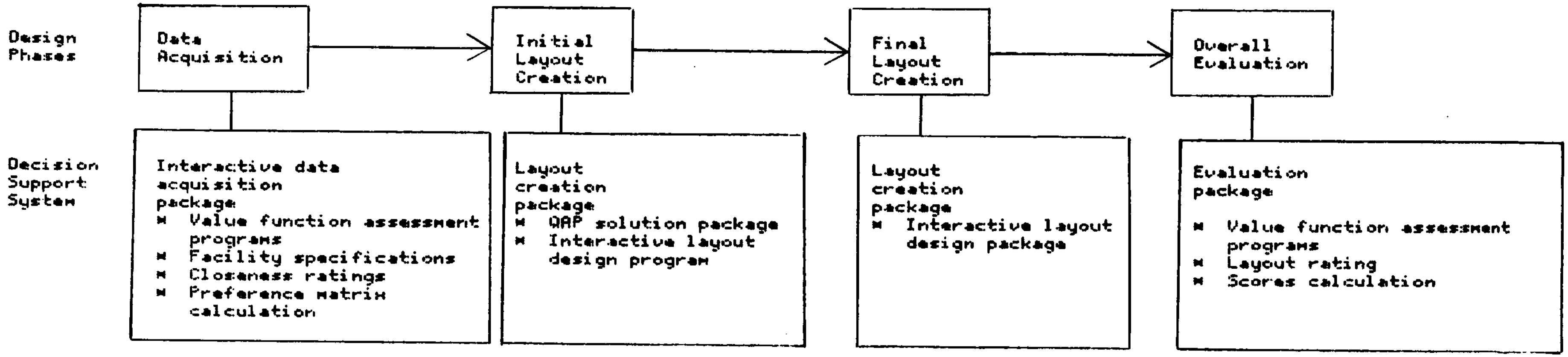


Figure 6.1 Flow chart of the layout design procedure

tance of these criteria are evaluated. These data are given in Figures 6.2 to 6.9.

Figure 6.2 presents closeness related criteria, their minimum and maximum levels, and the degrees of importance (scaling constants). Department sizes and names are given in Figure 6.3. The interdepartmental personnel flow data are presented in Figure 6.4. Figures 6.5, 6.6, 6.7, and 6.8 present closeness ratings for common personnel, common data used, contact necessary and convenience, respectively. The last four criteria are hypothetical ones and are subjectively measured. Marginal value functions for all criteria are shown in Figure 6.9. The preference matrix that is obtained as a result of combining marginal value functions, scaling constants, and closeness ratings is presented in Figure 6.10.

6.6.2 LAYOUTS OBTAINED USING CORELAP AND ALDEP

The layout generation procedures CORELAP and ALDEP, as described in Chapter 2, are first applied to solve the example problem. The measure (designated by TPRA) used to compare layouts generated by different procedures is "total preference per unit distance", and is defined as

$$\text{TPRA} = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij} / d_{ij}$$

Closeness related criterion	Minimum value	Maximum value	Scaling constant
Flow	0	70	.3
Common personnel	0	10	.15
Common information	0	10	.15
Contact necessary	0	1	.3
Convenience	0	2	.1

Figure 6.2 Criteria Specification

Department number	Department name	Size in feet (length,width)
1	Police department	40,38
2	Jail	33,20
3	Court room	64,60
4	Judge's chamber	20,15
5	License Bureau	30,30
6	Treasurer's office	30,30
7	Welfare office	30,30
8	Health department	30,30
9	Public work and sanitation	30,30
10	Engineer's office	50,46
11	Recreation department	30,30
12	Mayor's office	30,10
13	Town council's chambers	30,25

Figure 6.3 Department specification and size

		Department												
	#	1	2	3	4	5	6	7	8	9	10	11	12	13
	1	-	30	70	4					10			10	
D	2		-	8	2									
e	3			-	13									
p	4				-					10				
a	5					-	44			20				
r	6						-	20		20			30	
t	7							-	40	35				
m	8								-	46	40	10		
e	9									-	30	5		
n	10										-	10		
t	11											-		
	12												-	14
	13													-

Figure 6.4 Interdepartmental personnel flow between departments expressed as the number of persons traveling between department pairs

		Department												
	#	1	2	3	4	5	6	7	8	9	10	11	12	13
	1	-	5	2										
D	2		-	8										
e	3			-	10									
p	4				-									
a	5					-								
r	6						-							
t	7							-						
m	8								-					
e	9									-	5	5		
n	10										-			
t	11											-		
	12												-	
	13													-

Figure 6.5 Closeness rating for Common personnel (Max 10,Min 0)

		Department												
	#	1	2	3	4	5	6	7	8	9	10	11	12	13
	1	-												
D	2		-											
e	3			-	10									
p	4				-									
a	5					-	10							
r	6						-	8		8				
t	7							-	9					
m	8								-	7		3		
e	9									-	10			
n	10										-			
t	11											-		
	12												-	
	13													-

Figure 6.6 Closeness rating for Common data used (Max 10,Min 0)

	Department												
#	1	2	3	4	5	6	7	8	9	10	11	12	13
1	-	1											
D 2		-	1										
e 3			-	1									
p 4				-									
a 5					-								
r 6						-							
t 7							-						
m 8								-					
e 9									-				
n 10										-			
t 11											-		
12												-	1
13													-

Figure 6.7 Closeness rating for Contact necessary (Max 10,Min 0)

		Department												
	#	1	2	3	4	5	6	7	8	9	10	11	12	13
	1	-												
D	2		-											
e	3			-										
p	4				-									
a	5					-	2	1	1					
r	6						-	2	2	2				
t	7							-	2	2				
m	8								-	2		2		
e	9									-	2			
n	10										-			
t	11											-		
	12												-	
	13													-

Figure 6.8 Closeness rating for Convenience (Max 10,Min 0)

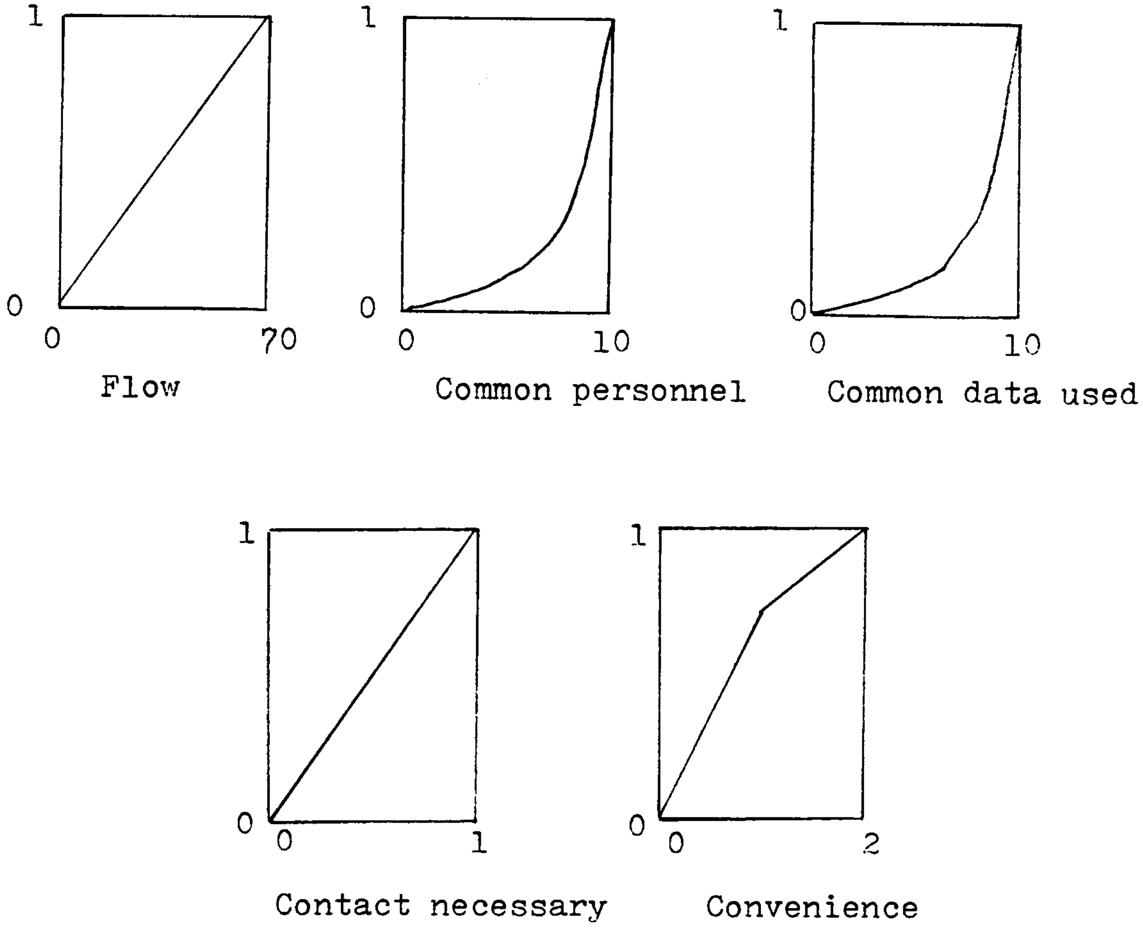


Figure 6.9 Marginal Value Functions

		Department												
	#	1	2	3	4	5	6	7	8	9	10	11	12	13
	1	.00	.44	.31	.02	.00	.00	.00	.00	.04	.00	.00	.04	.00
D	2	.44	.00	.36	.01	.00	.00	.00	.00	.00	.00	.00	.00	.00
e	3	.31	.36	.00	.66	.00	.00	.00	.00	.00	.00	.00	.00	.00
p	4	.02	.01	.66	.00	.00	.00	.00	.00	.04	.00	.00	.00	.00
a	5	.00	.00	.00	.00	.00	.44	.08	.08	.09	.00	.00	.00	.00
r	6	.00	.00	.00	.00	.44	.00	.20	.10	.20	.00	.00	.13	.00
t	7	.00	.00	.00	.00	.08	.02	.00	.29	.25	.00	.00	.00	.00
m	8	.00	.00	.00	.00	.08	.10	.29	.00	.31	.17	.15	.00	.00
e	9	.04	.00	.00	.04	.09	.20	.25	.31	.00	.39	.04	.00	.00
n	10	.00	.00	.00	.00	.00	.00	.00	.17	.39	.00	.04	.00	.00
t	11	.00	.00	.00	.00	.00	.00	.00	.15	.04	.04	.00	.00	.00
	12	.04	.00	.00	.00	.00	.13	.00	.00	.00	.00	.00	.00	.36
	13	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.36	.00

Figure 6.10 Preference matrix

TPRA is the sum of preference ratings of department pairs divided by the distances between them. TPRA is used here to compare the layouts generated by CORELAP, ALDEP, and the proposed procedure because the previously described measure of TPR (see section 3.3.2.1) is not applicable to the layouts generated by CORELAP and ALDEP.

The layout created by CORELAP is shown in Figure 6.11. The TPRA value of the layout is 247. Figure 6.12 shows the layout created by ALDEP and the TPRA value of this layout is 142. Note that the layout generated by ALDEP gives a smooth boundary and departments can assume other shapes besides being rectangular, whereas CORELAP generates a zig-zag boundary for the departments and the department shapes are always rectangular.

6.6.3 RESULTS OF EXPERIMENTATION WITH THE PROPOSED PROCEDURE

Next, we present results obtained using the proposed procedure. The results are presented both for the case of equal and unequal department sizes. Figure 6.13 shows the initial layout and the final layout (A1) created by the proposed procedure when the departments are assumed to be unequal during the generation of the initial layout. Figures 6.14, 6.15, 6.16, 6.17, 6.18, and 6.19 depict initial and final layouts created under several conditions by the proposed

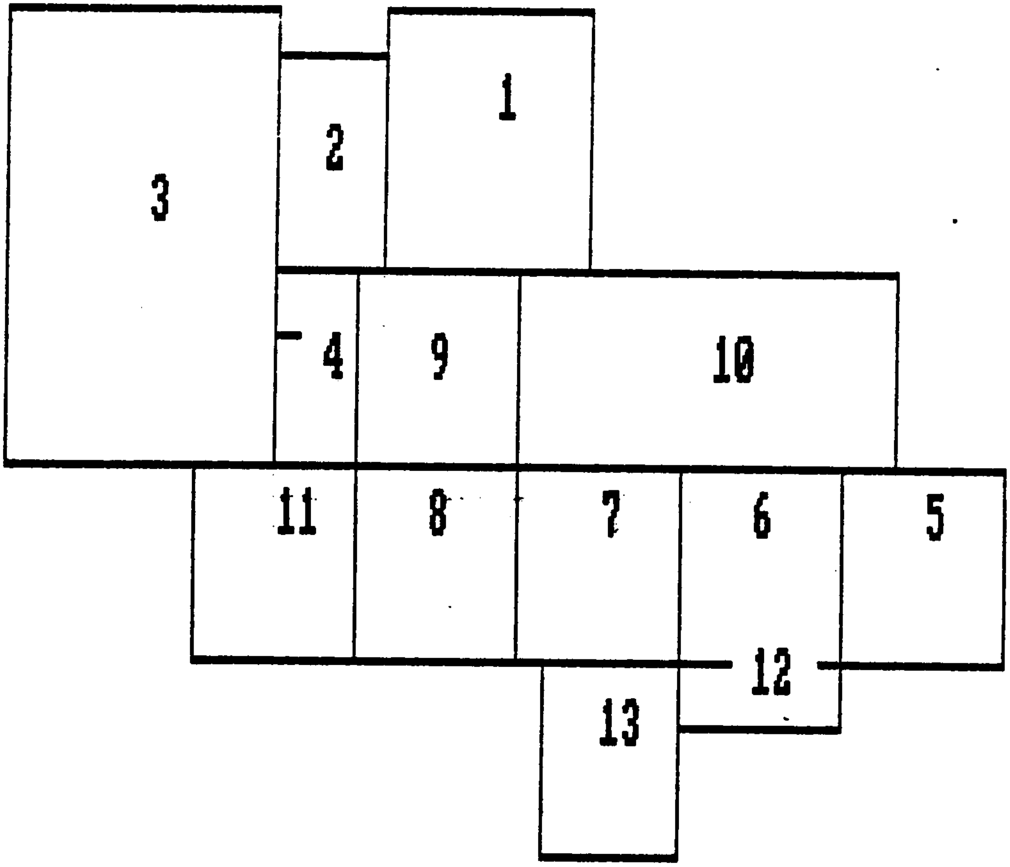


Figure6.11 Layout created by CORELAP

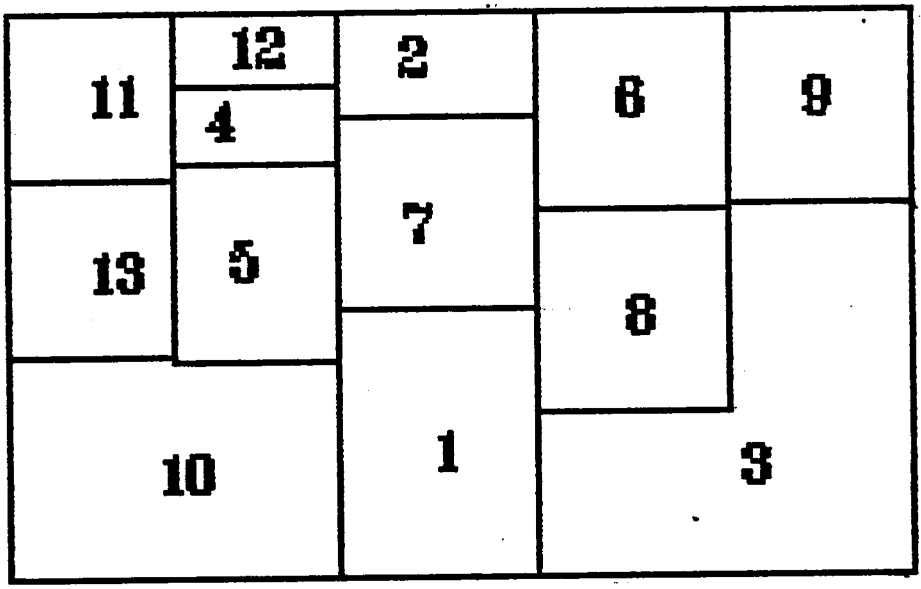


Figure6.12   Layout created by ALDEP

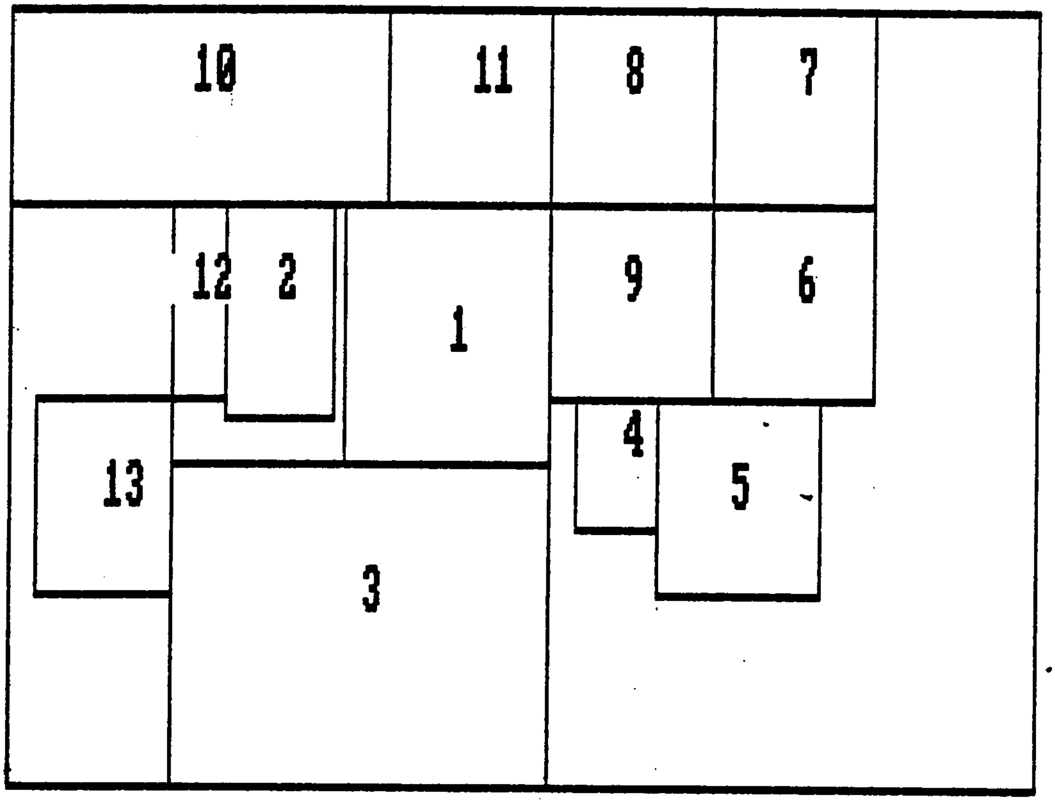
procedure when the departments are assumed to be of equal sizes during the generation of the initial layout. Figure 6.14 depicts the layout (B1) obtained when only the first closeness criterion (flow) is in consideration. Similarly, Figures 6.15, 6.16, 6.17, and 6.18 depict layouts designated B2, B3, B4, and B5 when the criterion of common personnel, common data, necessary contact, and convenience, respectively, are in consideration individually. Figure 6.19 represents the layout (B6) obtained when all criteria are in consideration together.

All scores related to the layouts depicted in Figures 6.11 to 6.19 are summarized in Figure 6.20. The TMAX score shown in Figure 6.20 is computed by making the scaling constant of the considered criterion to be 1.0 and the others to be zero. It also shows values of TPR for the layouts generated by the proposed procedure under various conditions. It is clearly indicated by Figure 6.20 that better results are obtained when all criteria are considered together.

A detailed experimentation is next carried out as depicted in Figure 6.21. Seventy preference matrices of problems of sizes 9 to 23 departments are randomly generated by the matrix generator shown in Appendix C. Both CORELAP and the proposed procedure are used to create layouts for these matrices. Figure 6.21 presents scores of layouts created by

10	10	11	8	7
12	2	1	9	6
13	3	3	4	5

Initial layout

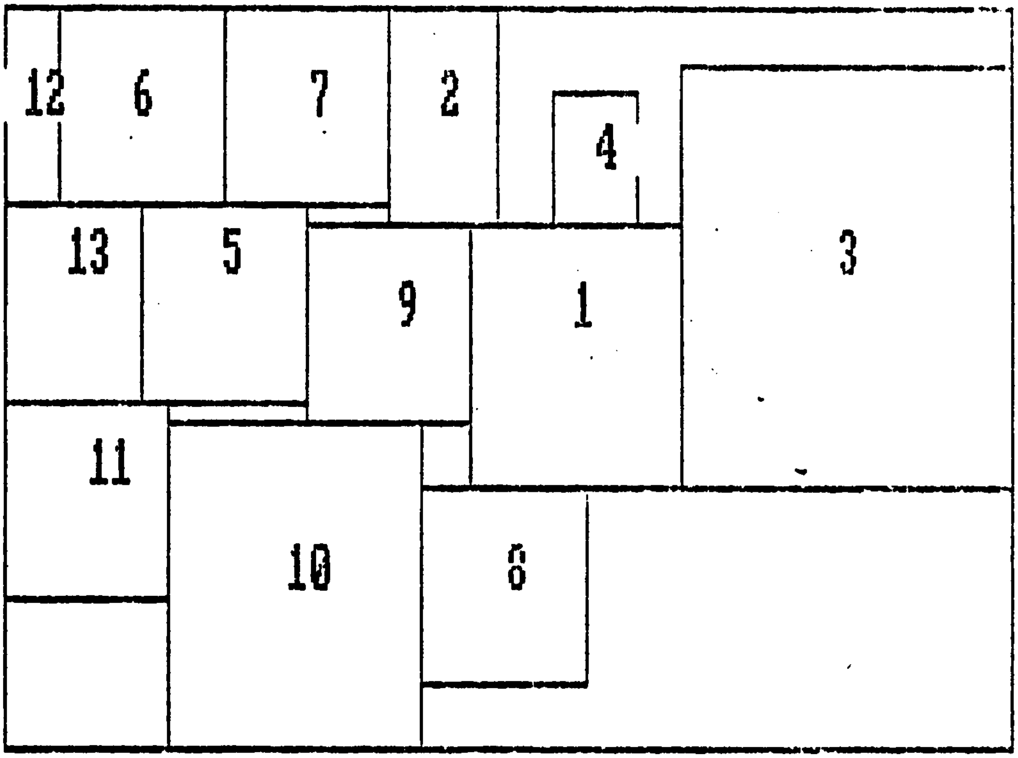


Final layout

Figure 6.13 Initial layout and final layout (designated A1) when all departments are assumed to be unequal

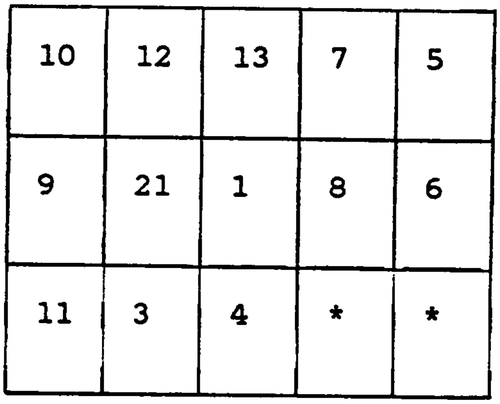
12	6	7	2	4
13	5	9	1	3
11	10	8	*	*

Initial layout



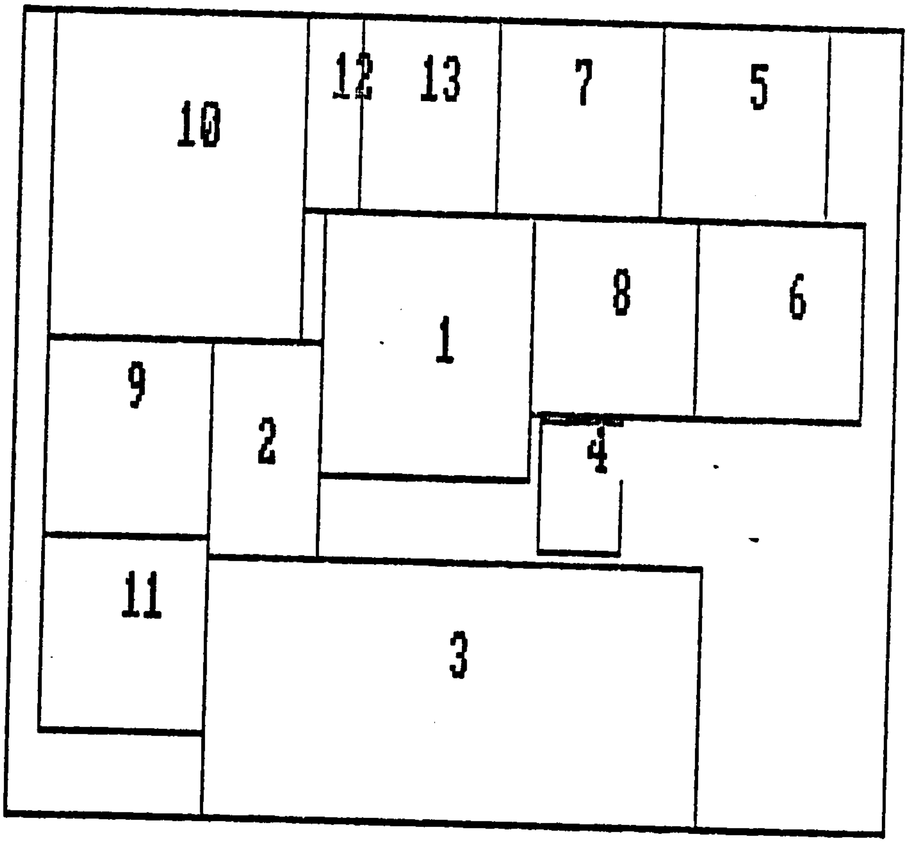
Final layout

Figure 6.14 Initial layout and final layout (designated B1) for the criterion of Interdepartment flow



10	12	13	7	5
9	21	1	8	6
11	3	4	*	*

Initial layout

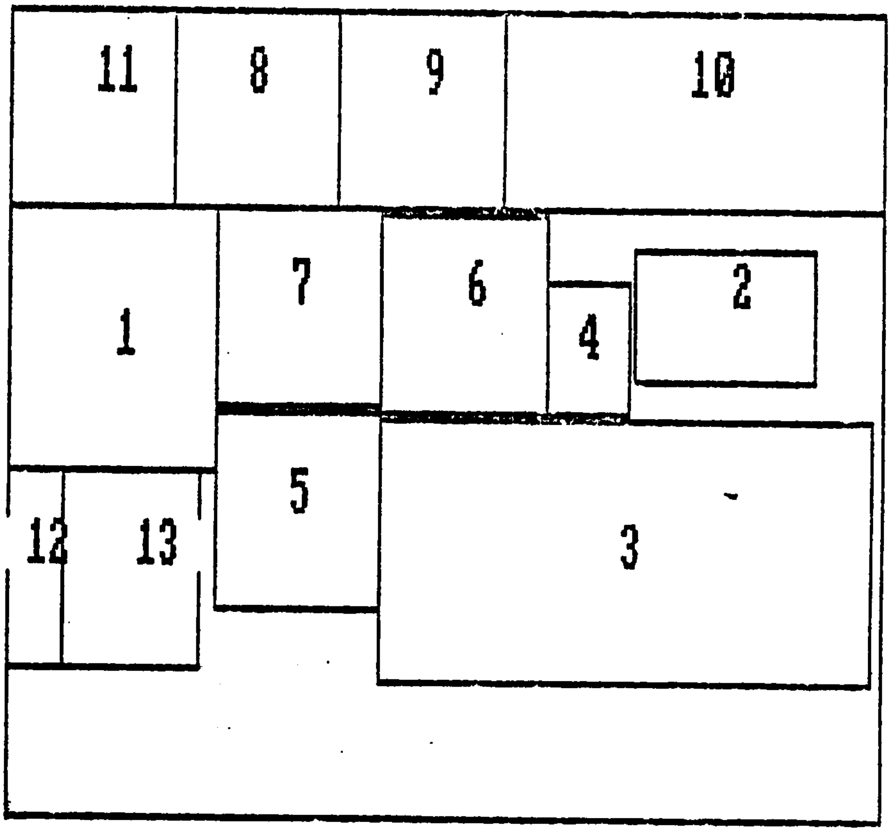


Final layout

Figure 6.15 Initial layout and final layout (designated B2) for the criterion of Common personnel

11	8	9	10	2
1	7	6	4	3
12	13	5	*	*

Initial layout

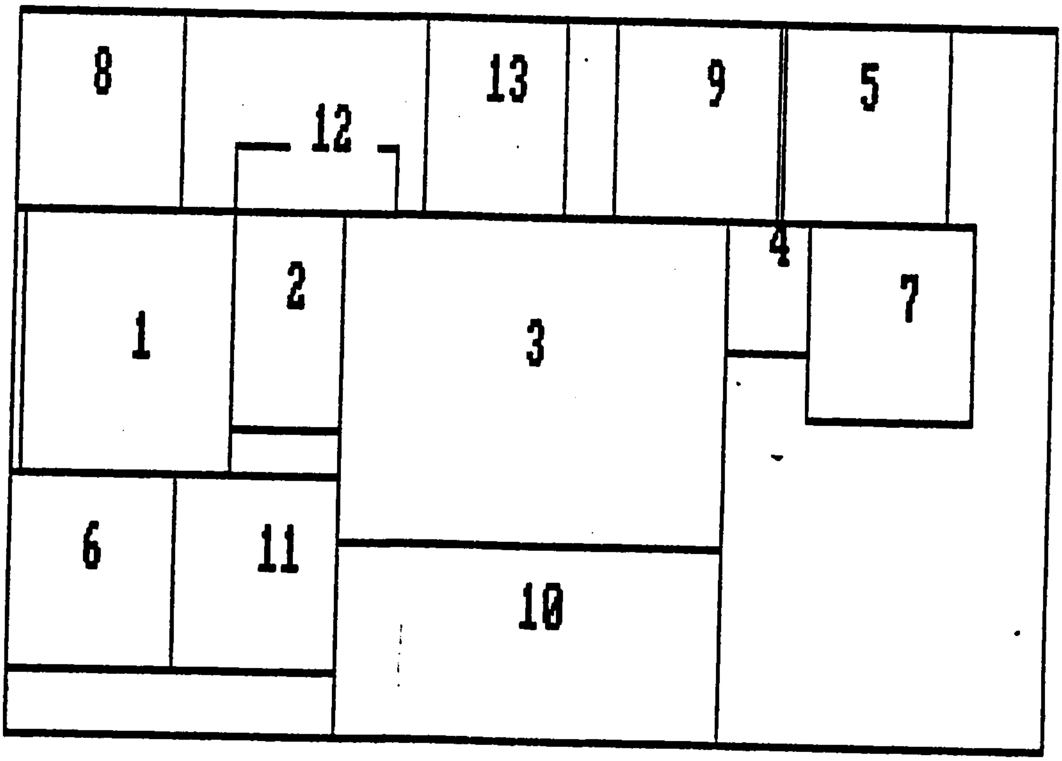


Final layout

Figure 6.16 Initial layout and final layout (designated B3) for the criterion of Common information

8	12	13	9	5
1	2	3	4	7
6	11	10	*	*

Initial layout

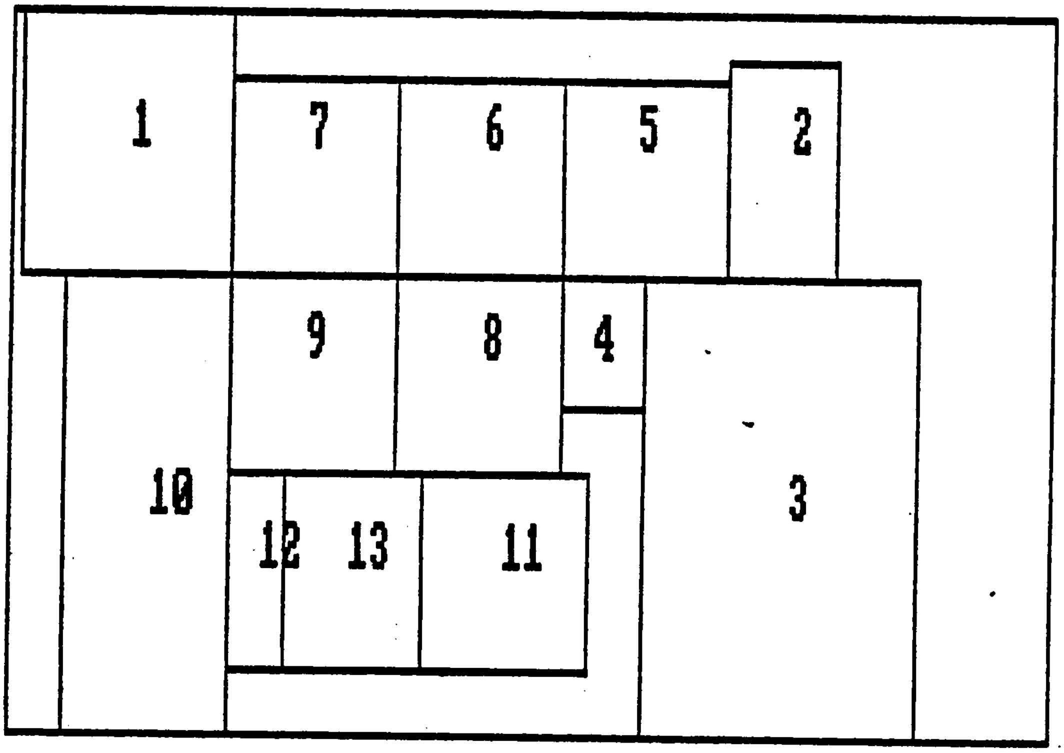


Final layout

Figure 6.17 Initial layout and final layout (designated B4) for the criterion of Contact necessary

1	7	6	5	2
10	9	8	4	3
12	13	11	*	*

Initial layout

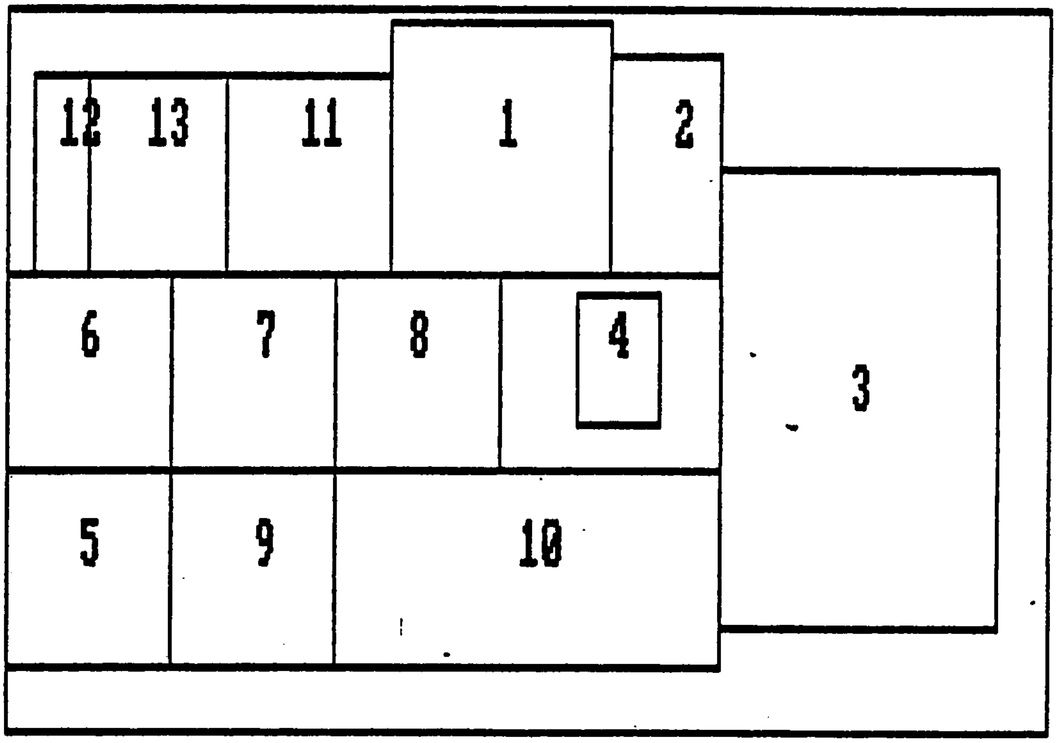


Final layout

Figure 6.18 Initial layout and final layout (designated B5) for the criterion of Convenience

12	13	11	1	2
6	7	8	4	3
5	9	10	*	*

Initial layout



Final layout

Figure 6.19 Initial layout and final layout (B6) when all criteria are considered together

Layout Generation Procedure	Score		
	TMAX	TPRA	TPR
CORELAP	-	247	-
ALDEP	-	142	-
Proposed procedure assuming unequal size departments and all criteria considered together (Layout A1)	-	273	10.04
Proposed procedure assuming equal size departments for			
Flow criterion only (Layout B1)	12.78	240	8.60
Commmon personnel criterion only (Layout B2)	2.89	207	7.70
Commmon information criterion only (Layout B3)	6.97	236	8.10
Contact necessary criterion only (Layout B4)	8.00	149	6.10
Convenience criterion only (Layout B5)	17.67	225	7.90
All criterion together (Layout B6)	-	264	9.10

Figure 6.20 Scores of Layouts Created by CORELAP,ALDEP, and the Proposed Procedure Under Various Conditions.

CORELAP and the proposed procedure and the percentage improvement of the proposed procedure over CORELAP. A graph which depicts the percent improvement (%) of the proposed procedure over CORELAP as a function of the problem size (n) is shown in Figure 6.22. The graph shows a decreasing trend in percentage improvement when the problem size is small (n=9 to 14). This trend ceases at the medium size problem (n=14 to 15) and becomes an increasing trend for larger problem size (n=16 to 23). The nature of this relationship between the performance of CORELAP and the proposed procedure may be explained with reference to Figure 6.23 which speculates relative performances of both of these procedures, compared to the optimal solution, as a function of problem size. The exact performance of these procedures are difficult to measure due to the unavailability of the optimal solution for layouts when n>5. For n=<2, both procedures generate optimal layouts. The proposed procedure continues to generate optimal layouts for n=<5 because the problem is solved exactly. As the problem size increases, the performance of both the procedures first decrease and then tend to stabilize. Since CORELAP is basically a construction routine, its performance is lower than that of the proposed procedure. The performance of CORELAP decreases more uniformly than that of the proposed procedure. Because of this nature of the graphs, the difference between the performances of CORELAP and the proposed procedure first decreases and then increases with an increase in problem size.

Next, we discuss how the total preference ratings (TPR) (described in section 3.3.3) can be used in the overall decision process to choose the best layout alternative. The TPR preference function must first be evaluated as described in section 3.3.3. In order to evaluate a preference function, an upper bound value must be determined for which a preference rating of 1 is asssumed. We show here one way of computing an upper bound of TPR. Let Tmax be the TPR value when the scaling constant of criterion k is set to 1.0 and the scaling constants of other criteria are set to 0. The upper bound is then computed as follows:

$$UB = \sum_{k=1}^{n} Tmax_k sc_k$$

where

sc_k = actual scaling constant of attribute k.

Thus, for the example problem of Figure 6.20,

$UB = 12.78(.3)+2.89(.15)+6.97(.15)+8.00(.3)+17.67(.1)$

$= 9.48$

A similar method can be applied to find the upper bound of TPRA if TPRA is used as the overall decision criterion.

Problem Size (n)	TPRA of Proposed Procedure	TPRA of CORELAP	Percentage Improvement (%)	Avarage Percentage Improvement
9	212.66	198.00	7.40	
9	240.16	226.00	6.27	
9	227.16	205.00	10.81	
9	193.83	183.33	5.73	
9	239.66	225.00	6.52	
9	239.16	224.16	6.69	
9	260.66	241.00	8.16	7.37
10	249.23	232.76	7.08	
10	289.20	272.86	5.99	
10	236.40	222.23	6.38	
10	250.10	238.00	5.08	
10	198.00	183.57	7.86	6.48
11	178.90	167.50	6.81	
11	185.34	175.46	5.63	6.22
12	214.00	198.80	7.65	
12	187.73	175.40	7.03	
12	360.02	340.29	5.80	
12	355.51	338.26	5.10	
12	327.07	310.91	5.20	
12	351.46	332.73	5.63	
12	388.61	368.66	5.41	
12	375.23	359.60	4.35	
12	344.43	335.83	2.56	
12	401.96	375.40	7.08	5.41
13	189.00	180.73	4.58	
13	251.97	239.17	5.35	4.96
14	494.33	486.70	1.57	
14	422.40	411.96	2.53	
14	437.53	424.73	3.01	
14	470.56	459.26	2.46	
14	474.60	461.46	2.85	
14	484.70	462.20	4.87	
14	463.73	442.19	4.87	
14	503.63	485.50	3.73	
14	457.40	439.36	4.11	3.33
15	570.20	542.63	5.08	
15	381.60	370.00	3.14	4.11
16	615.43	587.90	4.68	
16	403.10	387.10	4.13	4.41
17	428.90	411.00	4.36	
17	632.71	602.01	5.10	
17	648.37	620.45	4.50	
17	623.42	599.85	3.93	
17	678.85	652.38	4.06	
17	621.13	607.63	2.22	
17	641.85	607.65	5.63	
17	711.41	678.68	4.82	4.33
18	698.93	662.96	5.43	
18	476.85	456.00	4.57	5.00
19	843.22	806.51	4.55	
19	483.06	460.28	4.95	4.75
20	847.88	798.00	6.25	
20	895.35	844.40	6.03	
20	858.55	831.59	3.24	
20	905.54	864.57	4.74	
20	849.26	807.48	5.17	
20	895.44	841.16	6.45	5.32
21	956.24	888.48	7.63	
21	906.08	848.83	6.74	
21	949.94	898.07	5.78	
21	917.13	858.53	6.83	
21	898.39	843.72	6.48	
21	957.04	893.03	7.17	6.77
22	1026.36	960.16	6.89	
22	975.86	914.68	6.69	6.79
23	1095.73	1036.98	5.67	
23	1066.51	992.11	7.50	
23	1180.61	1100.29	7.30	
23	1071.44	1003.23	6.80	
23	1014.80	947.58	7.09	6.87

Figure 6.21 Scores of Layouts generated by CORELAP and the Proposed Procedure for problem of different size.

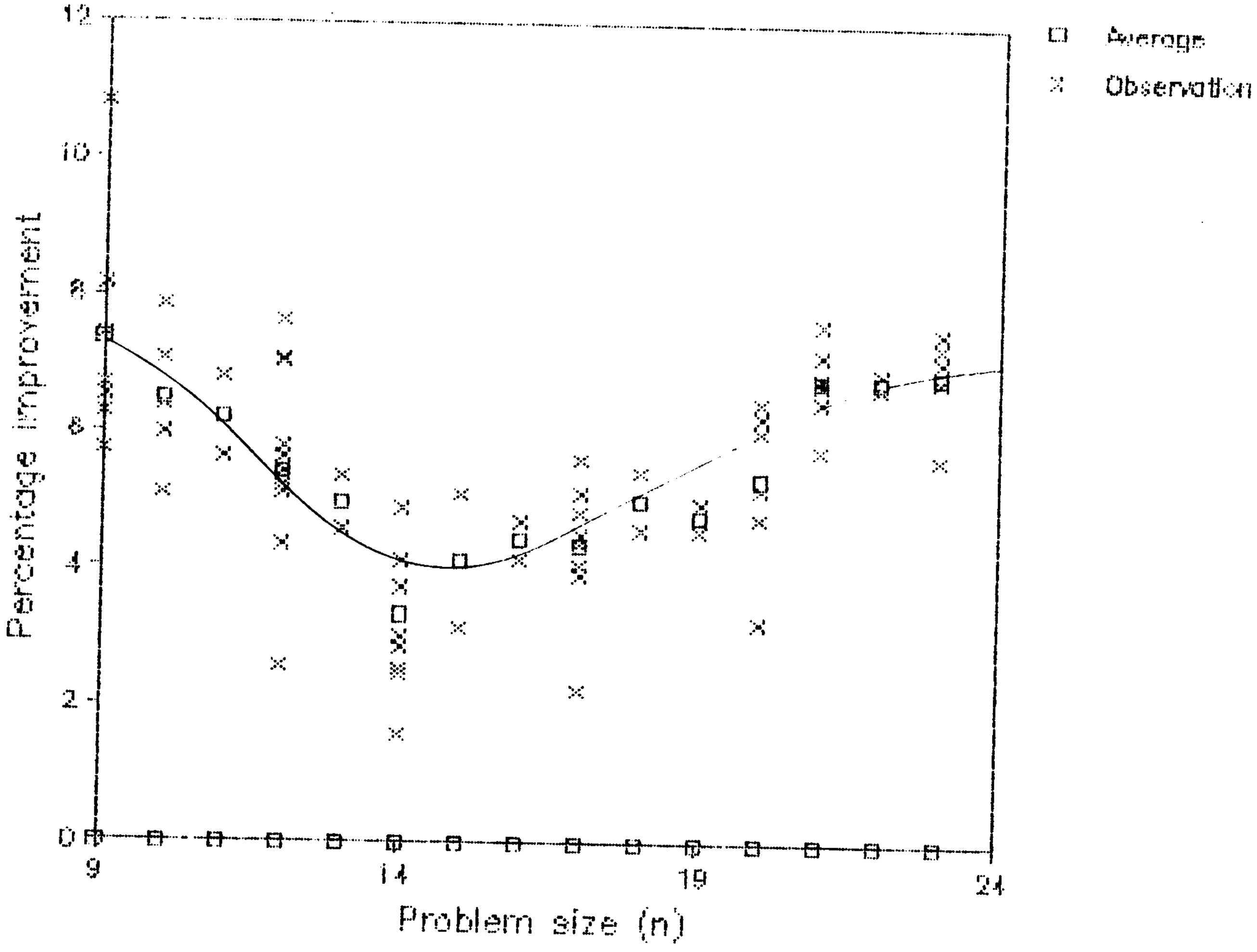


Figure 6.22 Graph showing percentage improvement of the proposed procedure over CORELAP versus problem size (n)

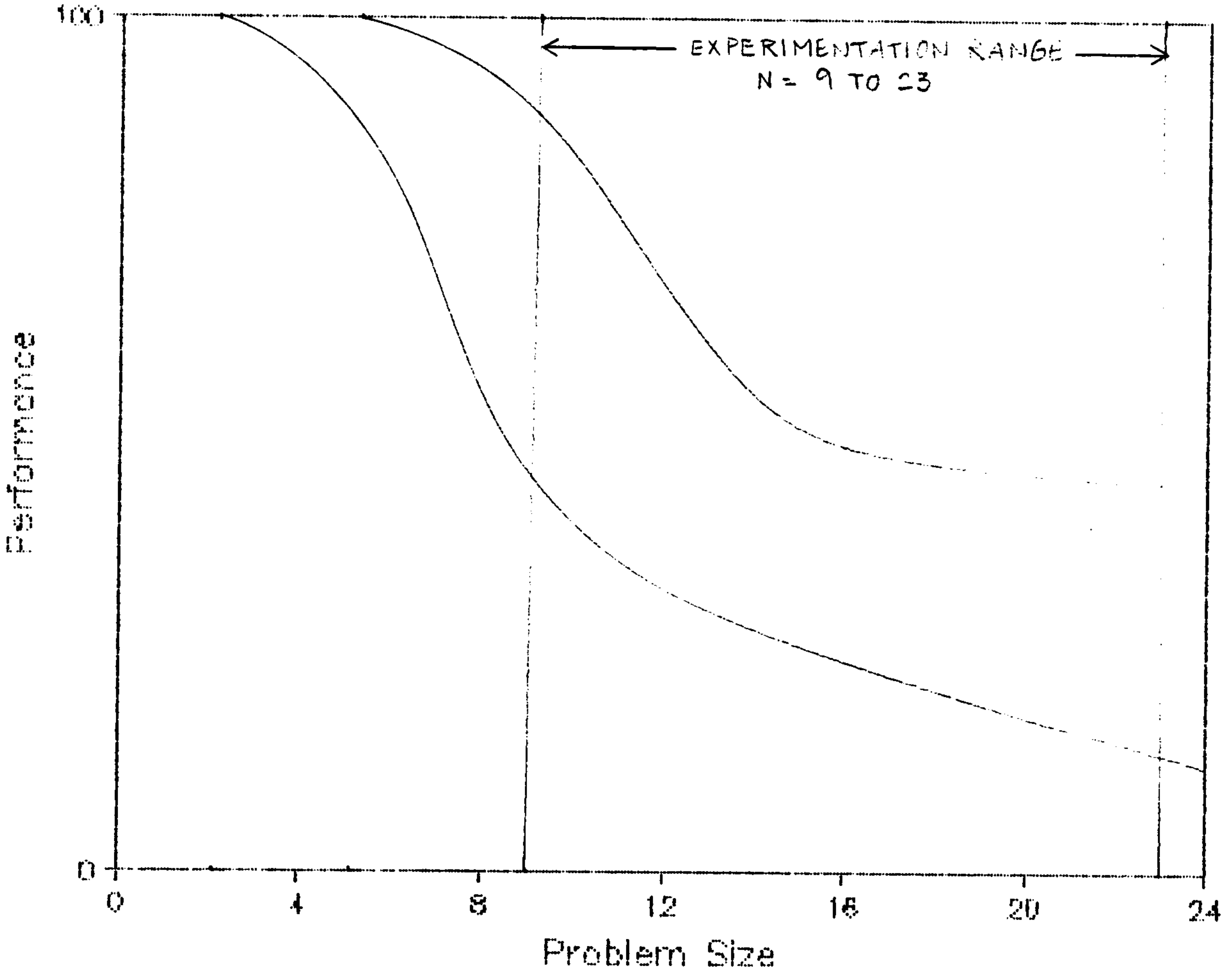


Figure 6.23 Performance curves of CORELAP and the proposed procedure versus problem size

6.7 SUMMARY AND CONCLUSION

In this chapter a method of determining relative department locations was developed. This method is based on the formulation of the layout problem as a quadratic assignment problem. The proposed procedure for the solution of the QAP is specially designed in order to be implemented on an IBM PC. Three construction and two improvement routines are programmed on the IBM PC as a part of the layout design package that is developed. The quality of the layout obtained can be improved by running through the improvement routine several times. The results from this methodology are used to create a final layout where actual department sizes and other criteria are considered. A procedure to create the final layout was also described. Finally, the results of using the initial and the final creation process on the example problem were presented. The proposed procedure gives layouts which are consistently better than those obtained using CORELAP. In addition, the proposed method allows the designer to interact during the design process to make appropriate adjustments on the final layout.

7. SUMMARY, CONCLUSION, AND RECOMMENDATIONS

This study has presented a decision theoretic approach and a computerized decision support system for solving the general layout design problem. The research was divided into seven tasks as follows.

1. Review of the literature on layout design approaches.

2. Review of the literature on decision theory.

3. Development of a decision theoretic approach to the layout design problem.

4. Review of the literature on quadratic assignment problem.

5. Development of heuristics for solving the quadratic assignment problem.

6. Development of new theoretical results to justify the decision theoretic approach taken.

7. Development of a computerized decision support system.

This chapter summarizes the study and gives suggestions for further research.

7.1 SUMMARY AND CONCLUSION

A major motivation of the research came from the fact that most of the layout design approaches reported in the literature are concerned only with economic justification, basically, that of material handling cost. Some of the studies consider qualitative factors and use closeness ratings to measure these factors. Generally, a scale of A, E, I, O, and U is used to represent closeness ratings between pairs of departments. This research developed a layout design approach which accounts for multiple criteria in a systematic fashion. The results presented in Chapter 6 have shown that this multicriteria approach gives better solutions than a single-criterion approach.

7.1.1 A DECISION THEORETIC APPROACH TO LAYOUT DESIGN

Layout design has rarely been addressed as a multicriteria problem. Chapter 3 of this study presented an approach for layout design, taking multicriteria into account. This approach is a systematic way of measuring the closeness rating for any departmental pair. A function representing the closeness rating between any departmental pair is constructed. Since the layout criteria may be measured on different scales, they are converted into a preference scale. The criteria are then combined by either additive, multiplicative, or multilinear functions. The combined criterion, for any departmental pair, is then rated. This results in closeness ratings of department pairs. The departments are then assigned to locations so that the sum of closeness ratings is maximized. A set of overall evaluation criteria is used to evaluate the final layout.

7.1.2 JUSTIFICATION OF THE DECISION THEORETIC APPROACH

The approach developed in this study involves criteria which are measured on various scales. The first task in this case is to convert these scales into a unique scale such that all criteria can be combined. The combining of functions may take several forms, for example, additive, multilinear, and multiplicative. In order to take any of these forms, the attributes (criteria) set must satisfy some conditions. In the literature, axiomatic systems defined for attributes, which are measured on continuous scales, have been developed. However, these axiomatic systems are not applicable for the case where attributes are discrete or binary (Deutsch and Malmborg, 1984). The insufficiency of the present axiomatic system for the discrete and binary cases was presented in Chapter 3. In Chapter 5, the axiomatic system for mixed attributes which includes continuous, binary, and discrete attributes was presented. The necessary theoretical results for the additive, multilinear, and multiplicative forms, based on this axiomatic system, were presented.

7.1.3 SOLUTION PROCEDURES FOR THE QUADRATIC ASSIGNMENT PROBLEM

In Chapter 6, a method of determining relative department locations based on closeness ratings was presented. The objective of the method was to optimize closeness ratings. This involved formulation of the problem as a quadratic assignment problem which attempts to allocate departments to locations. Three construction and two improvement heuristics were developed and presented.

The successful heuristics reported in the literature are computationally intensive and require a tremendous amount of computer memory. The solution procedures for the quadratic assignment problem are specially designed by this research so that they can be implemented on an IBM PC with 256K RAM. Since the computer has limited calculation speed and computer memory, the improvement heuristic developed is designed as a polynomial time heuristic which is effective with respect to the quality of solutions obtained, while at the same time not being computationally intensive.

7.1.4 DECISION SUPPORT SYSTEM FOR THE LAYOUT DESIGN PROBLEM

The decision support system developed in this research, to aid the layout design is an interactive computer software which utilizes both the human and machine capabilities. In a layout design problem, the human is necessary for the evaluation of an ill-structured criterion, creative design, and practical design. The machine is needed for handling large amounts of information pertaining to material handling costs and closeness ratings for the various departments involved in a problem. The computer software package was developed based on the decision support system presented in Chapter 4. This package is divided into three phases as follows.

1. Data Acquisition: This phase prepares primary information for the layout problem. This includes five programs; namely, the scaling constant assessment program, the marginal value function assessment program, the facility specification program, the adjacency requirement program, and computing the preference matrix program. Two of these programs, namely, the scaling constant assessment program and the marginal value function assessment program, utilize the IBM PC graphic capability to aid the assessment procedure.

2. Layout construction: This phase includes two programs, namely, the initial layout creation program and the final layout creation program. They utilize information prepared in phase 1, and generate layout alternatives using the proposed algorithm. The initial layout creation program determines relative department locations by using three construction and two improvement heuristics. The improvement heuristics are based on the solution to a quadratic assignment problem and pairwise interchanges among departments. Any combination of these heuristics can be utilized to solve the problem. Results from the initial layout creation program merely give relative department locations since it assumes equal department sizes. Consequently, the purpose of the final layout creation program is to create layout alternatives with exact department sizes using the results from the initial layout creation program.

3. Overall evaluation: This phase consists of four programs to prepare data and to compute final layout alternative scores. There are two assessment programs, namely, the scaling constant assessment program and the marginal value function assessment program. These two programs prepare value functions for the overall decision criterion. The other two programs compute the final scores.

7.1.5 RESULTS OF EXPERIMENTATION

An extensive experimentation was conducted to test the performance of the proposed procedure for the layout design problem. The results indicate that this procedure is easy to use and it generates better solutions than those generated by CORELAP.

7.2 RECOMMENDATION

This section lists some topics for future study and improvement as follows.

1. Decision theory: An axiomatic basis for value functions defined on a mixed attributes set was presented in chapter 5. The compensatory conditions defined must hold to allow the additive value function form to be used. Should the condition be verified, the number of all possible verifications become very large when the function involves multilevel attributes. Gorman (1968) has done some work to reduce the number of independent verifications for the axiomatic system defined on continuous attributes. A similar work concerning our axiomatic system should be investigated in order to simplify the verification procedure. Otherwise, alternative approaches or axiomatic systems for the mixed attribute set will be a promising research area.

2. Computer package: The computer package has been developed as a tool to aid layout design. The additive form is used for the combination of criteria. The total preference function is formulated as a quadratic assignment problem. An improvement can be made in the program by developing a better data preparation scheme to accommondate unequal size departments. Currently, some of the data preparation for unequal size departments is done manually. In addition, the computer package should provide other value function forms, for example, multilinear and multiplicative. Another potential improvement concern providing more user-friendly commands in the program.

3. Other applications of decision theory: Typically, problems in manufacturing system design involve several economic and performance related criteria. In this research, we have developed results for multicriteria functions involving discrete and continuous attributes. These were demonstrated in this research specifically on the layout design problem. However, the same functions, and modifications thereof, can be investigated for application to other multicriteria problems.

BIBLIOGRAPHY

Apple, J. M., 1977. Plant Layout and Material Handling 3rd Edition, John Wiley & Sons, New York.

Apple, J. M. and Deisenroth, M. P., 1980. "A Computerized Plant Layout Analysis and Evaluation Technique (PLANET)", Proceeding AIIE 1980, pp.121-127.

Armour, G. C. and Buffa, E. S., 1963. "A Heuristic Algorithm and Simulation Approach to Relative Location of Facilities", Management Science, 9(1), pp.294-309.

Bazaraa, M. S., 1975. "Computerized Layout Design: A Branch and Bound Approach", AIIE Transactions, 7(4), pp.432-438.

Bazaraa, M. S. and Goode, J. J., 1977. "The Travelling Salesman Problem: A Duality Approach", Mathematical Programming, v13, pp.221-237.

Bazaraa, M. S. and Sherali, H. D., 1980. "Benders Partitioning Scheme Applied to a New Formulation of the Quadratic Assignment Problem", Naval Research Logistics Quarterly, v27, pp.29-41.

Bazaraa, M. S. and Sherali, H. D., 1982. "On the Use of Exact and Heuristics Cutting Methods for the Quadratic Assignment Problem", Journal of Operations Research Society, v33, pp.991-1003.

Block, T. E., 1978. "A New Construction Algorithm for Facilities Layout", Journal of Engineering Production, n2, pp111-

Buffa, E. S., Armour, G. C., and Vollmann, T. E., 1964. "Allocating Facilities with CRAFT", Harvard Business Review, 42(2), pp.136-159.

Burkard, R. E. and Stratmann, K. H., 1978. "Numerical Investigation on the Quadratic Assignment Problem", Naval Research Logistics Quarterly, v25, n1, pp.129-248.

Cabot, V. A. and Francis, R. L., 1970. "Solving Certain Nonconvex Quadratic Minimization Problems by Ranking the Extreme Points", Operations Research, v18, pp.82-86.

Dawes, R. M. and Corrigan, B., 1974. "Linear Models in Decision Making", Psychological Bulletin, pp.97-106.

Debreu, G., 1960. "Topological Methods in Cardinal Utility Theory", Mathematical Method in the Social Science, pp.16-26.

Deutsch, S J. and Malmborg, C. J., 1984. "An Axiomatic Basis for Multiattribute Value Functions Defined on Binary Attributes", Working Paper, School of Industrial Engineering and System Engineering, Georgia Tech and Department of Industrial Engineering and Operations Research, Virginia Tech.

Deutsch, S. J. and Malmborg, C. J., 1985. European Journal of Operational Research, v19, n2.

Deutsch, S. J. and Malmborg, C. J., 1985. "Interdependent additive conjoint structures", European Journal of Operational Research, v13, n4.

Drezner, Zvi, 1980. "DISCON: A New Method for the Layout Problem", Operations Research, 28(6), pp.1375-1384.

Dutta, K. N. and Sahu, S., 1982. "A Multigoal Heuristic for Facilities Design Problem: MUGHAL', Int. J. Prod. Res., 20(2), pp.147-154.

Dyer, J. S. and Sarin, R. K., 1979. "Measurable Multiattribute Value Functions", Operations Research, 27(4), pp.810-822.

Einhorn, H. J. and Hogart, R. M., 1975. "Unit Weighting Schemes of Decision Making", Organizational Behavior and Human Performance, 13, pp.171-192.

Filley, R. D., 1985. "Three Emerging Computer Technologies Boost Value of, Respect for Facilities Function", Industrial Engineering, 17(5), pp.27-39.

Fishburn, P. C., 1970. Utility Theory for Decision Making, John Wiley & Sons Inc., New York.

Foulds, L. R. and Robinson, D. F., 1978. "Graph theoretic Heuristic for the Plant Layout Problem", Int. J. Prod. Res., 16(1), pp.27-37.

Francis, R. L. and White, J. A., 1974. Facility Layout and Location: An Analytical Approach, Prentice-Hall Inc., Englewood Cliffs, New Jersey.

Gardiner, P. C., 1977. "Decision Spaces", IEEE Transactions on Systems, Man, and Cybernetics, SMC-7(5), pp.340-349.

Gaschutz, G. K. and Ahrens, J. H., 1968. "Suboptimal Algorithm for the Quadratic Assignment Problem", Naval Research Logistics Quarterly, v15, pp.49-62.

Gavett, J. W. and Plyter, N. V., 1966. "The Optimal Assignment of Facilities to Locations by Branch and Bound", Operations Research, v14, pp.210-232.

Gilmore, P. C., 1962. "Optimal and Suboptimal Algorithms for the Quadratic Assignment Problem", SIAM Journal on Applied Mathematics, v10, pp.305-313.

Gorman, W. M., 1968a. "The Structure of Utility Functions", Review of Economic Studies, 35, pp.367-390.

Gorman, W. M., 1968b. "Conditions for Additivity Separabilities", Econometric, 36, pp.605-609.

Graves, G. W. and Whinston, A. B., 1970. "An Algorithm for the Quadratic Assignment Problem", Management Science, v17, pp.453-471.

Hall, K. M., 1970. "An r-Dimensional Quadratic Placement Algorithm", Management Science, v17, pp.219-229.

Hillier, F. S., 1963. "Quantitative Tools for Plant Layout Analysis", The Journal of Industrial Engineering, v14, pp.33-40.

Hillier, F. S. and Connors, M. M., 1966. "Quadratic Assignment Problem Algorithms and The Locations of Indivisible Facilities", Management Science, v13, pp.42-57.

Johnson, R. V., 1982. "SPACECRAFT for Multi-floor Layout Planning", Management Science, 28(4), April, pp.406-417.

Kaufman, L. and Broeckx, F., 1978. "An Algorithm for the Quadratic Assignment Problem Using Benders' Decomposition", European Journal of Operational Research, v2, pp.207-211.

Keeney; R. L. and Raiffa, H., 1976. Decisions with Multiple Objectives: Preferences and Value Tradeoffs, John Wiley & Sons, New York.

Koopmanns, P. C. and Beckmann, M. 1957. "Assignment Problems and the Location of Economic Activities", Econometrica, v25, pp.53-76.

Krantz, C. H., Luce, R. D., Suppes, P., and Tversky, A., 1971. Foundations of Measurement, Volume I: Additive and Polynomial Representations, Academic Press, New York.

Krejeirik, M., 1969. "RUGR Algorithm', Computer Aided Design, Autumn.

Land, A. N., 1963. "A Problem of Assignment with Interrelated Costs", Operational Research Quarterly, v14, pp.185-198.

Lawler, E. L., 1963. "The Quadratic Assignment Problem", Management Science, v19, pp.586-599.

Lee, R. C. and Moore, J. M., 1967. "CORELAP-Computerized Relationship Layout Planning", Journal of Industrial Engineering, 18(3), pp.195-200.

Lewis, W. P. and Block, T. E., 1980. "On the Application of Computer Aids to Plant Layout", Int. J. Prod. Res., 18(1), pp.11-20.

Los, M., 1977. "Simultaneous Optimization of Land Use and Transportation: A Synthesis of the Quadratic Assignment Problem and of the Optimal Network Problem", Regional Science and Urban Economics.

Love, R. L. and Wong, J. Y., 1976. "Solving Quadratic Assignment Problems with Rectangular Distances and Integer Programming", Naval Research Logistics Quarterly, 23(3), pp.623-628.

Malmborg, C. J., 1981. Ph.D Dissertation, Georgia Tech.

Malmborg, C. J. and Sarin, S. C., 1984. Plant Layout and Design Course Work Book, Department of Industrial Engineering and Operation Research, Virginia Tech.

Mirchandani, P. B. and Obata, P., 1979. "Locational Decisions with Interactions between Facilities: The Quadratic Assignment Problem - A Review", Working Paper PS-79-1, Rensselaer Polytechnic Institute.

Moore, J. M., 1974. "Computer Aided Facilities Design: An International Survey", Int. J. Prod. Res., 12(2).

Muther, R. M., 1955. Practical Plant Layout, McGraw-Hill Inc., New York.

Muther, R. M., 1961. Systematic Layout Planning, Industrial Education Institute, Boston, Mass.

Muther, R. and McPherson, K., 1970. "Four Approaches to Computerized Layout Planning", Industrial Engineering, 2(2), pp.39-42.

Nugent, C. E., Vollmann, T. E., and Ruml, J., 1968. "An Experimental Comparison of Techniques for the Assignment of Facilities to Locations", Operations Research, v16, pp.150-173.

O'Brien, C. and Barr, E. Z. A., 1980. "An Interactive Approach to Computer Aided Facility Layout", Int. J. Prod. Res., 18(2), pp.201-211.

Patel, A. M. et. al., 1976. "Prepartioning as a Means of Enhancing Pairwise Interchange Solution to Quadratic Assignment Problems", Computer and Operations Research, v3, pp.49-56.

Pegels, C. C., 1966. "Plant Layout and Discrete Optimization", International Journal of Production Research, v5, pp.81-92.

Pierce, J. F. and Crowston, W. B., 1971. "Tree Search Algorithms for the Quadratic Assignment Problems", Naval Research Logistics Quarterly, v18, pp.1-36.

Rajgopal, P., 1985. "A Flexible Construction and Improvement Heuristic for the Quadratic Assignment Problem", Master Thesis, Virginia Tech.

Reed, R. Jr., 1961. Plant Layout, Irwin Inc.

Rosenblatt, M. J., 1979. "A facilities Layout Problem: A Multi-goal Approach", Int. J. Prod. Res., 17(4), pp.323-332.

Roucairol, C., 1978. "A Reduction Method for the Quadratic Assignment Problems", 3rd symposium Uber Operations Research, University Mannheim, October.

Seehof, J. M. and Evans, W. O., 1967. "Automated Layout Design Program", Journal of Industrial Engineering, 18(12), pp.690-695.

Seppanen, J. and Moore, J. M., 1970. "Facilities Planning with Graph Theory", Management Science, 17(4), December, pp.242-253.

Seppanen, J. and Moore, J. M., 1975. "String Processing Algorithms for Plant Layout Problems", Int. J. Prod. Res., 13(3), pp.239-254.

Smith, H. C. T., 1975. "A Computational Comparison of an Improved Pair Assignment Algorithm and a Pair Exclusion Algorithm for the Quadratic Assignment Problem", AD-A019799.

Vollmann, T. E., Nugent, C. E., and Zartler, R. L., 1968. "A Computerized Model for Office Layout", The Journal of Industrial Engineering, v19, pp.321-329.

APPENDIX A. DESCRIPTION OF PROGRAMS IN THE DECISION SUPPORT SYSTEM

MAIN: MAIN program accesses to three programs-MAIN1, MAIN2, and MAIN3.

MAIN1: This is a Data Acquisition control access program which allows the user to link to MAIN11, MAIN12, MAIN13, MAIN14, and MAIN15.

MAIN2: This is a Layout Construction control access program which links to MAIN21 and MAIN22.

MAIN3: This is a Evaluation control access program which links to MAIN31, MAIN32, MAIN33, and MAIN34.

MAIN11: This program helps in interactively assessing scaling constants of closeness related criteria.

MAIN12: This program helps in interactively assessing marginal value functions of the closeness related criteria.

MAIN13: This program acquires department specifications which include department names and sizes.

MAIN14: This program acquires criteria rating for all department pairs.

MAIN15: This program uses data acquired in MAIN11, MAIN12, MAIN13, MAIN14, and MAIN15 to compute the preference matrix. This matrix is used in the Layout Construction phase.

MAIN21: This program creates initial layouts which give relative department locations. The designer must specify block plan layout. Any combination of construction routines (CR10, CR20, or CR30) and improvement routines (IMP1 and CQAP) can be selected. Then, the computer produces initial layouts and compute their scores.

MAIN22: This program allows the designer to create final layouts using the actual department sizes.

MAIN31: This program assesses scaling constants of the overall decision criteria.

MAIN32: This program assesses marginal value functions of the overall decision criteria..

MAIN33: This program acquires criteria rating for all layout alternatives.

MAIN34: This program produces final scores of all layout alternatives.

CR10: Construction routine #1.

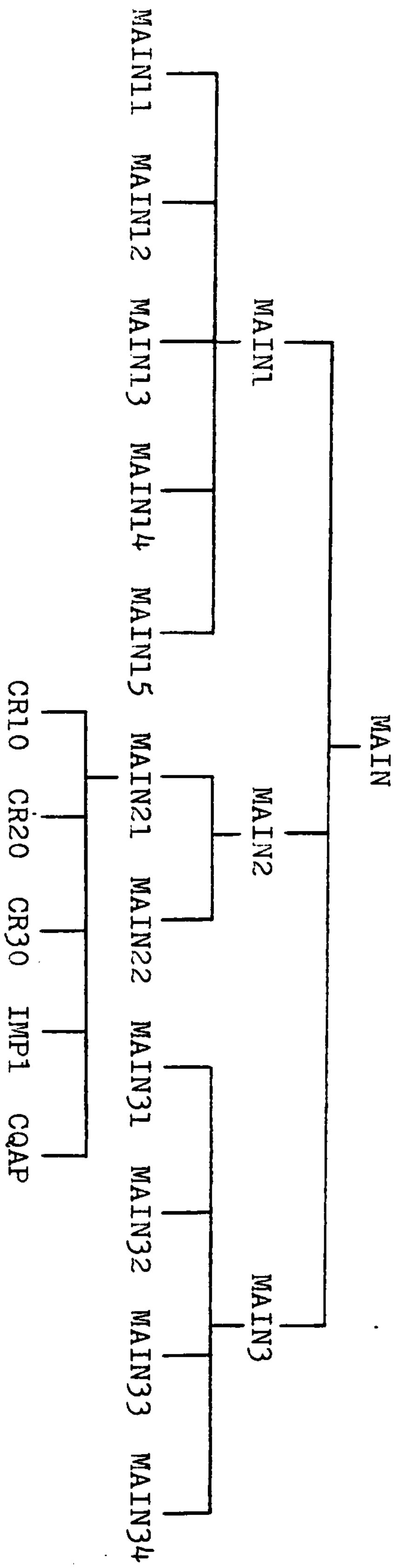
CR20: Construction routine #2.

CR30: Construction routine #3.

IMP1: Pairwise interchange routine.

CQAP: Specially designed improvement routine.

The macro-structure of the layout design package is shown on next page.



Macro-Structure of the Layout Design Package

APPENDIX B. A COMPUTER RUN OF THE EXAMPLE PROBLEM

Diagrams 1, 2, 3, and 4 are shown on the following pages, depict keys to accessing each screen. Legend N1.N2 : N1 represents program number and N2 represents the screen number created by each program. For example, see diagram 2, press pf8 to go from 12.6 to 1.1, press 2 and <return> to go from 1.1 to 12.1.

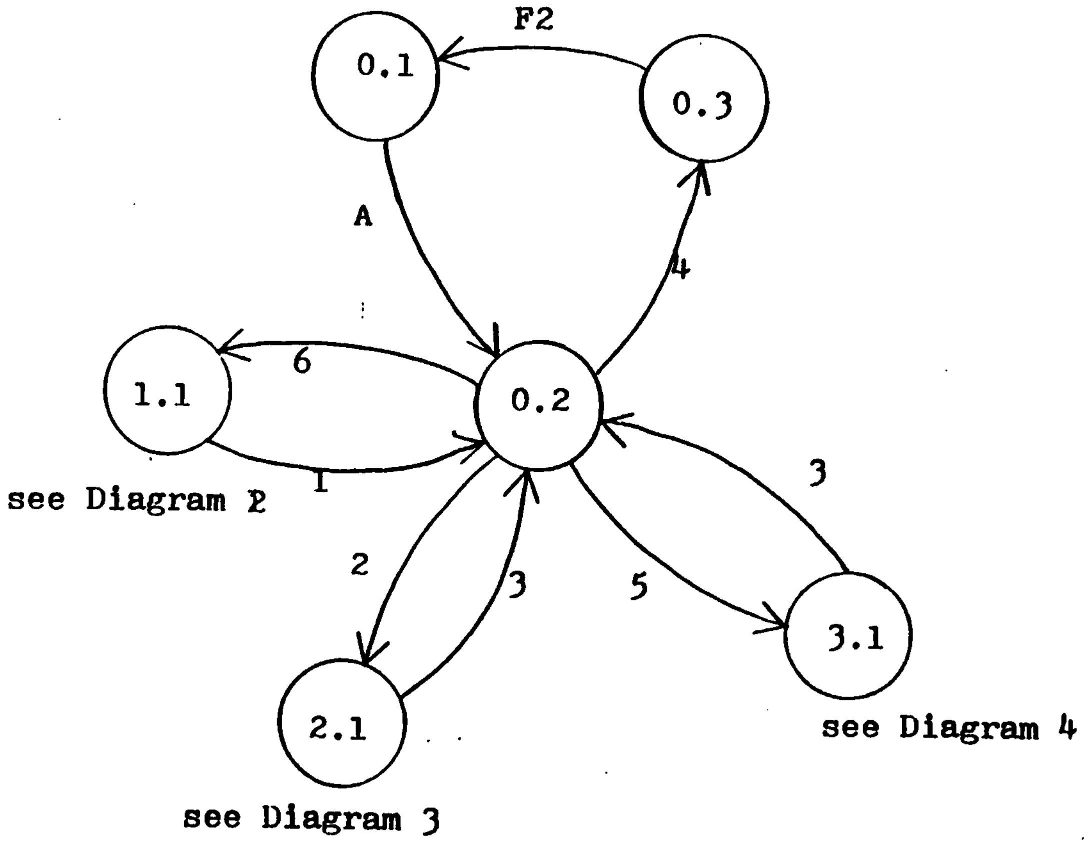

F2
0.1
0.3
A
4
6
1.1
0.2
1
see Diagram 2
3
2
3
5
3.1
2.1
see Diagram 4
see Diagram 3

Diagram 1

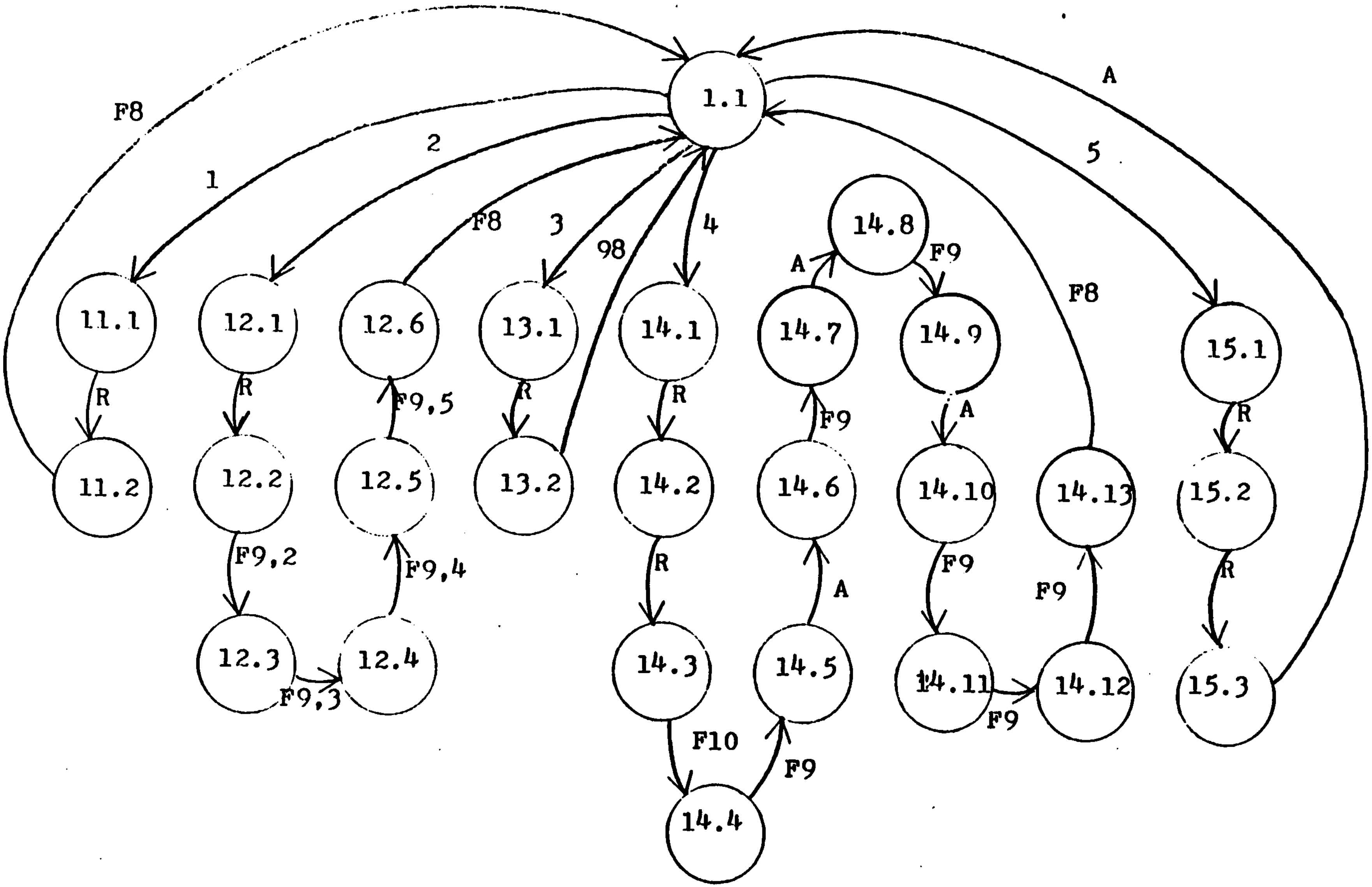

1.1
F8
1
2
F8
3
98
4
A
5
F8
11.1
R
11.2
12.1
R
12.2
F9,2
12.3
F9,3
12.4
F9,4
12.5
F9,5
12.6
13.1
R
13.2
14.1
R
14.2
R
14.3
F10
14.4
F9
14.5
A
14.6
F9
14.7
A
14.8
F9
14.9
A
14.10
F9
14.11
F9
14.12
F9
14.13
15.1
R
15.2
R
15.3

Diagram 2

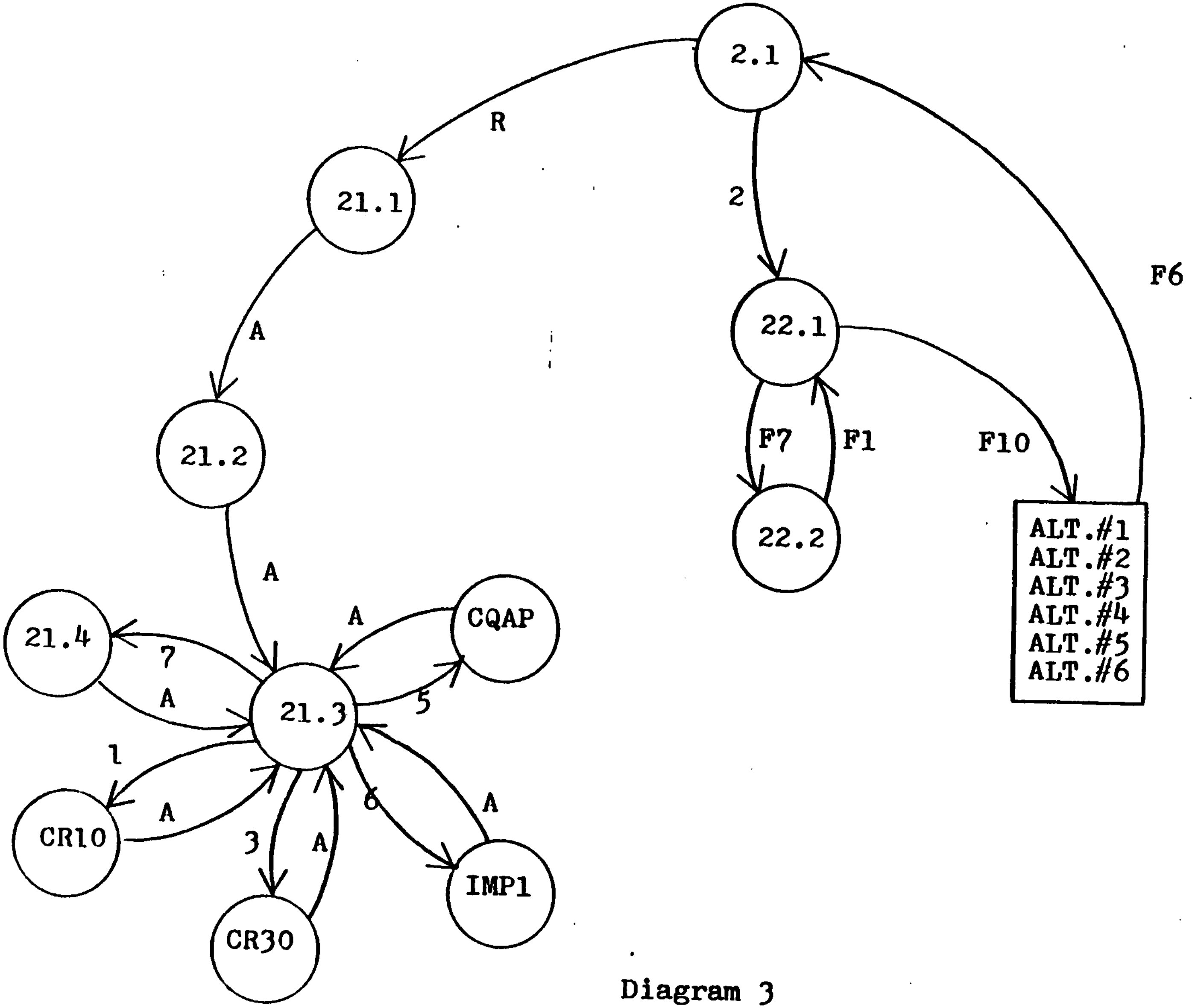

2.1
R
2
21.1
A
22.1
F6
21.2
F7
F1
F10
22.2
ALT.#1
ALT.#2
ALT.#3
ALT.#4
ALT.#5
ALT.#6
A
A
CQAP
21.4
7
A
21.3
5
1
A
CR10
3
A
6
IMP1
CR30

Diagram 3

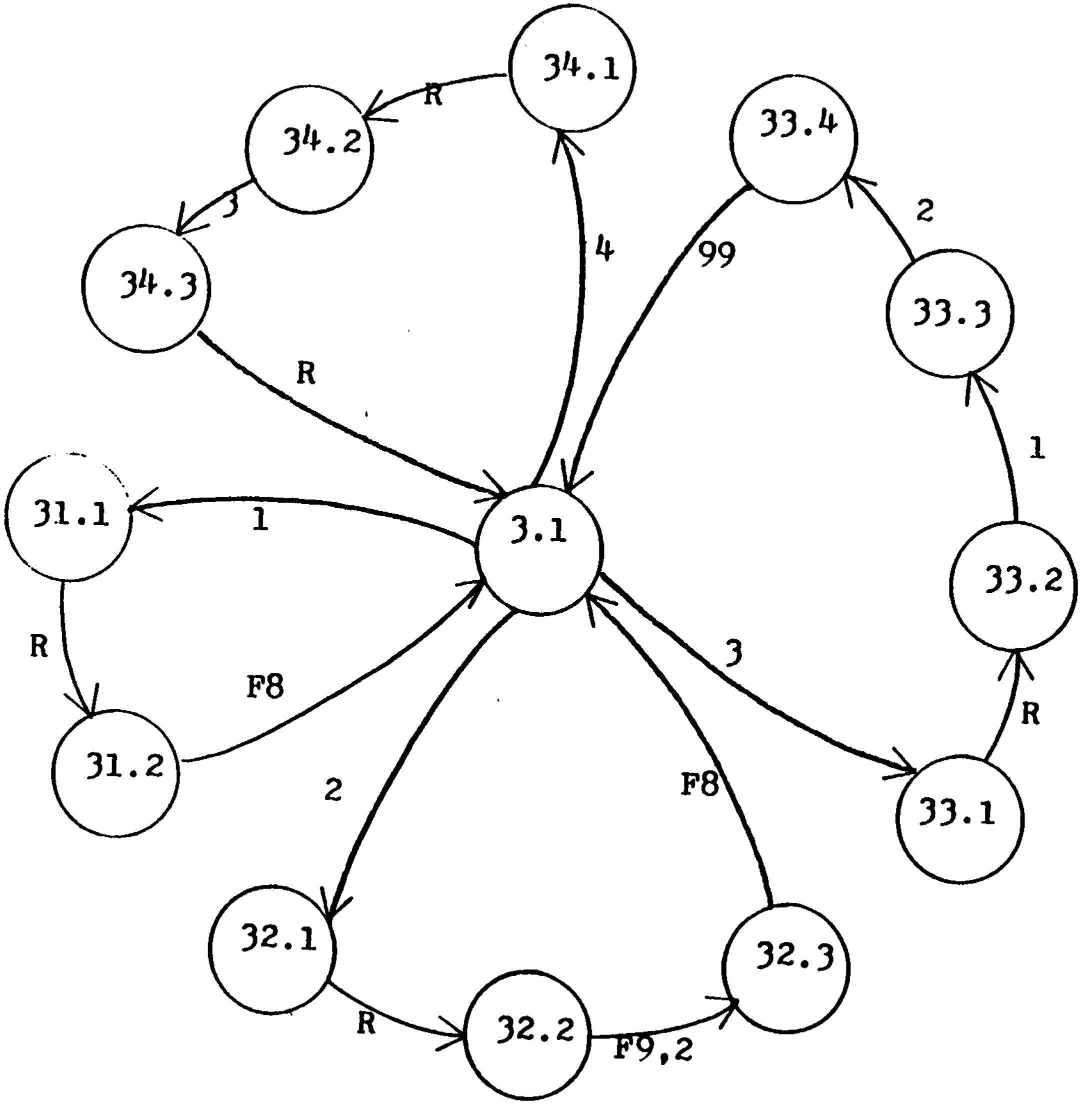
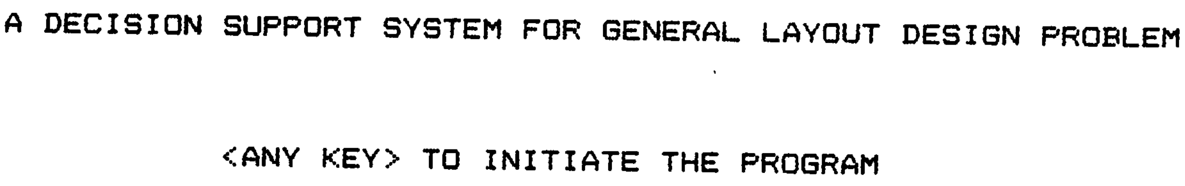
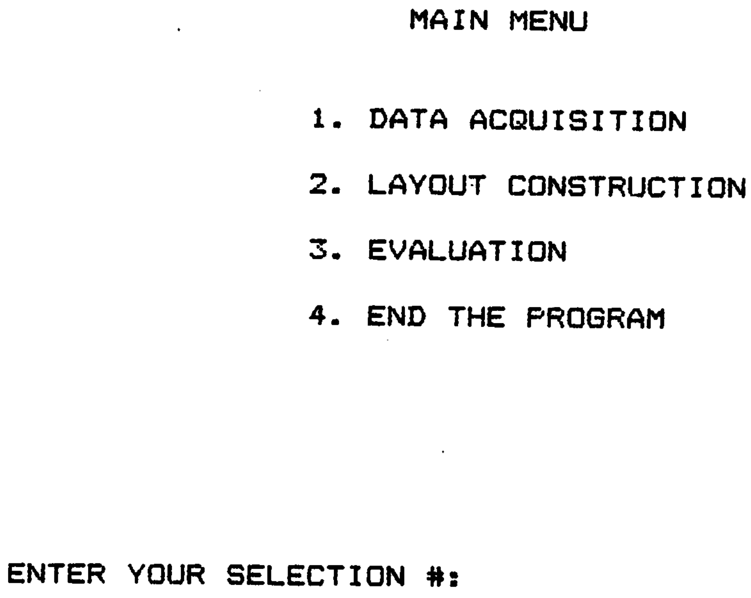

34.1
R
34.2
3
34.3
4
33.4
99
2
33.3
R
1
31.1
1
3.1
33.2
R
F8
3
R
31.2
2
F8
33.1
32.1
32.3
R
32.2
F9,2

Diagram 4



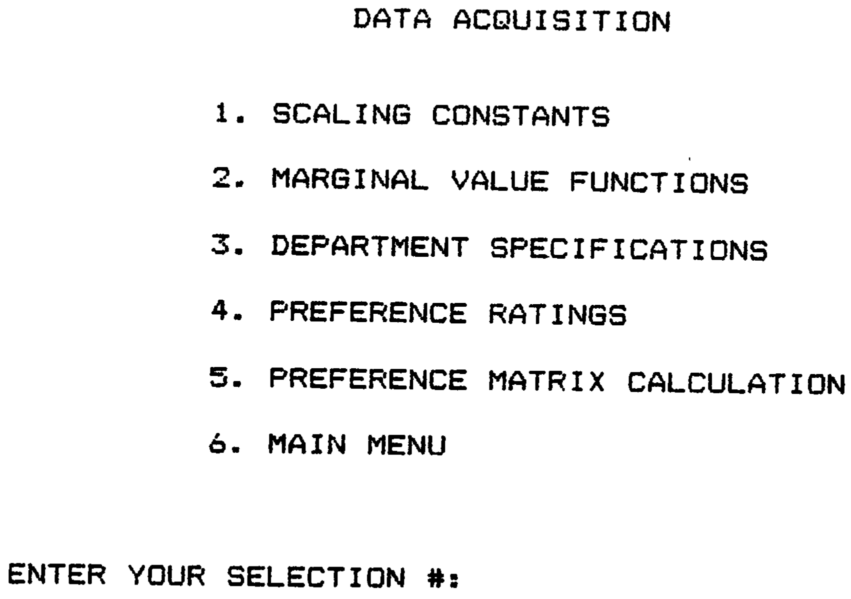
```
A DECISION SUPPORT SYSTEM FOR GENERAL LAYOUT DESIGN PROBLEM

           <ANY KEY> TO INITIATE THE PROGRAM
```

:0.1



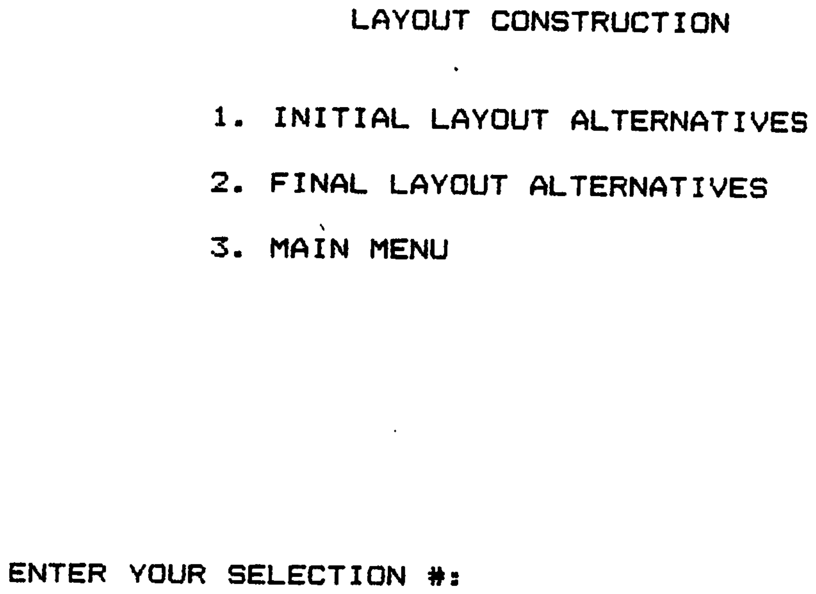
```
                      MAIN MENU

                 1. DATA ACQUISITION

                 2. LAYOUT CONSTRUCTION

                 3. EVALUATION

                 4. END THE PROGRAM



 ENTER YOUR SELECTION #:
```

:0.2



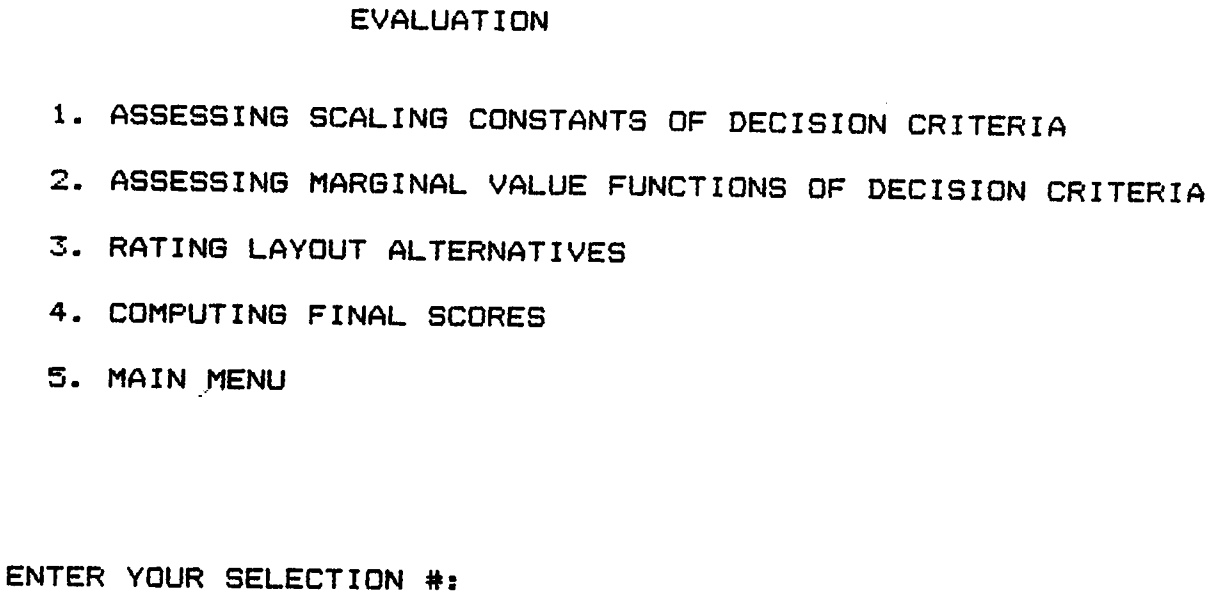
```
              DATA ACQUISITION

        1. SCALING CONSTANTS

        2. MARGINAL VALUE FUNCTIONS

        3. DEPARTMENT SPECIFICATIONS

        4. PREFERENCE RATINGS

        5. PREFERENCE MATRIX CALCULATION

        6. MAIN MENU


ENTER YOUR SELECTION #:
```

:1.1



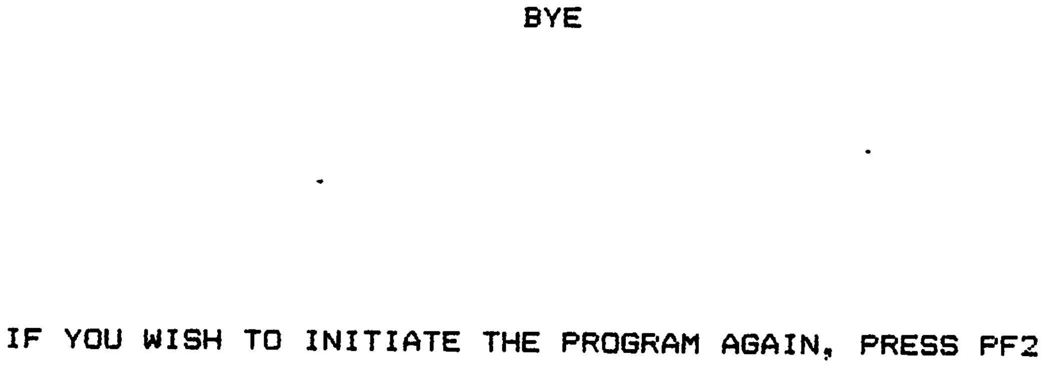
```
              LAYOUT CONSTRUCTION

        1. INITIAL LAYOUT ALTERNATIVES

        2. FINAL LAYOUT ALTERNATIVES

        3. MAIN MENU




ENTER YOUR SELECTION #:
```

:2.1



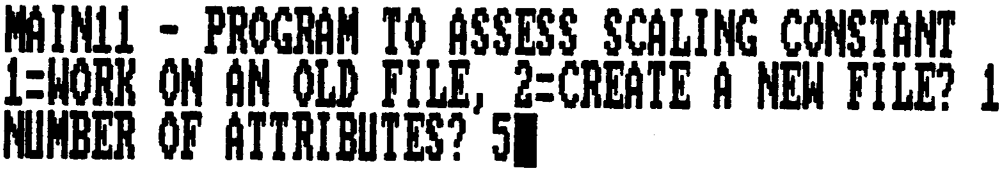
```
                    EVALUATION

   1. ASSESSING SCALING CONSTANTS OF DECISION CRITERIA

   2. ASSESSING MARGINAL VALUE FUNCTIONS OF DECISION CRITERIA

   3. RATING LAYOUT ALTERNATIVES

   4. COMPUTING FINAL SCORES

   5. MAIN MENU



ENTER YOUR SELECTION #:
```

:3.1



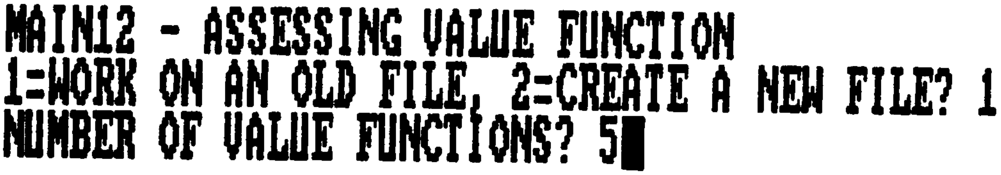
```
                              BYE




  IF YOU WISH TO INITIATE THE PROGRAM AGAIN, PRESS PF2
```

:0.3



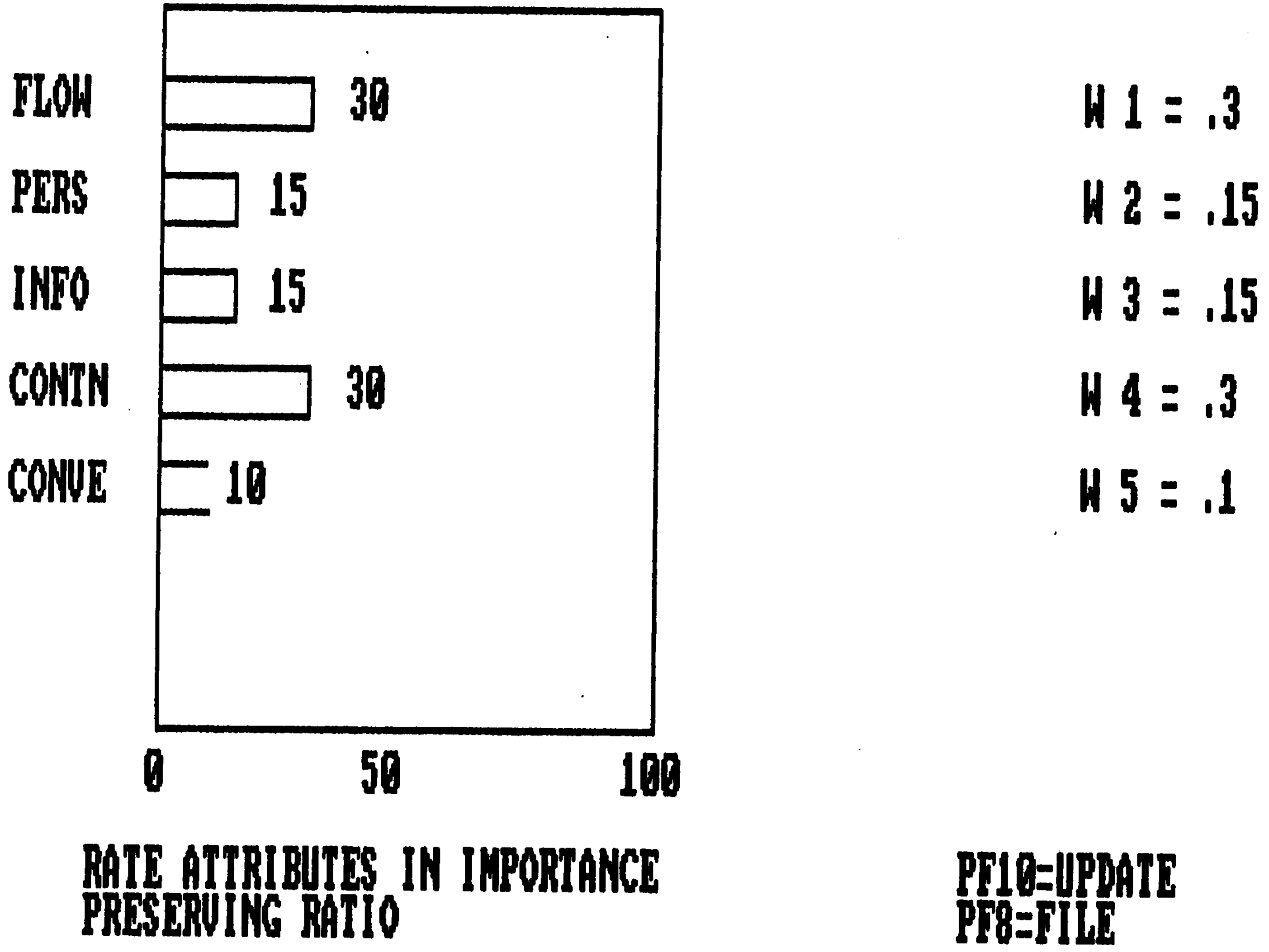
```
MAIN11 - PROGRAM TO ASSESS SCALING CONSTANT
1=WORK ON AN OLD FILE, 2=CREATE A NEW FILE? 1
NUMBER OF ATTRIBUTES? 5
```

:11.1



```
MAIN12 - ASSESSING VALUE FUNCTION
1=WORK ON AN OLD FILE, 2=CREATE A NEW FILE? 1
NUMBER OF VALUE FUNCTIONS? 5
```

:12.1



RATE ATTRIBUTES IN IMPORTANCE
PRESERVING RATIO

W 1 = .3

W 2 = .15

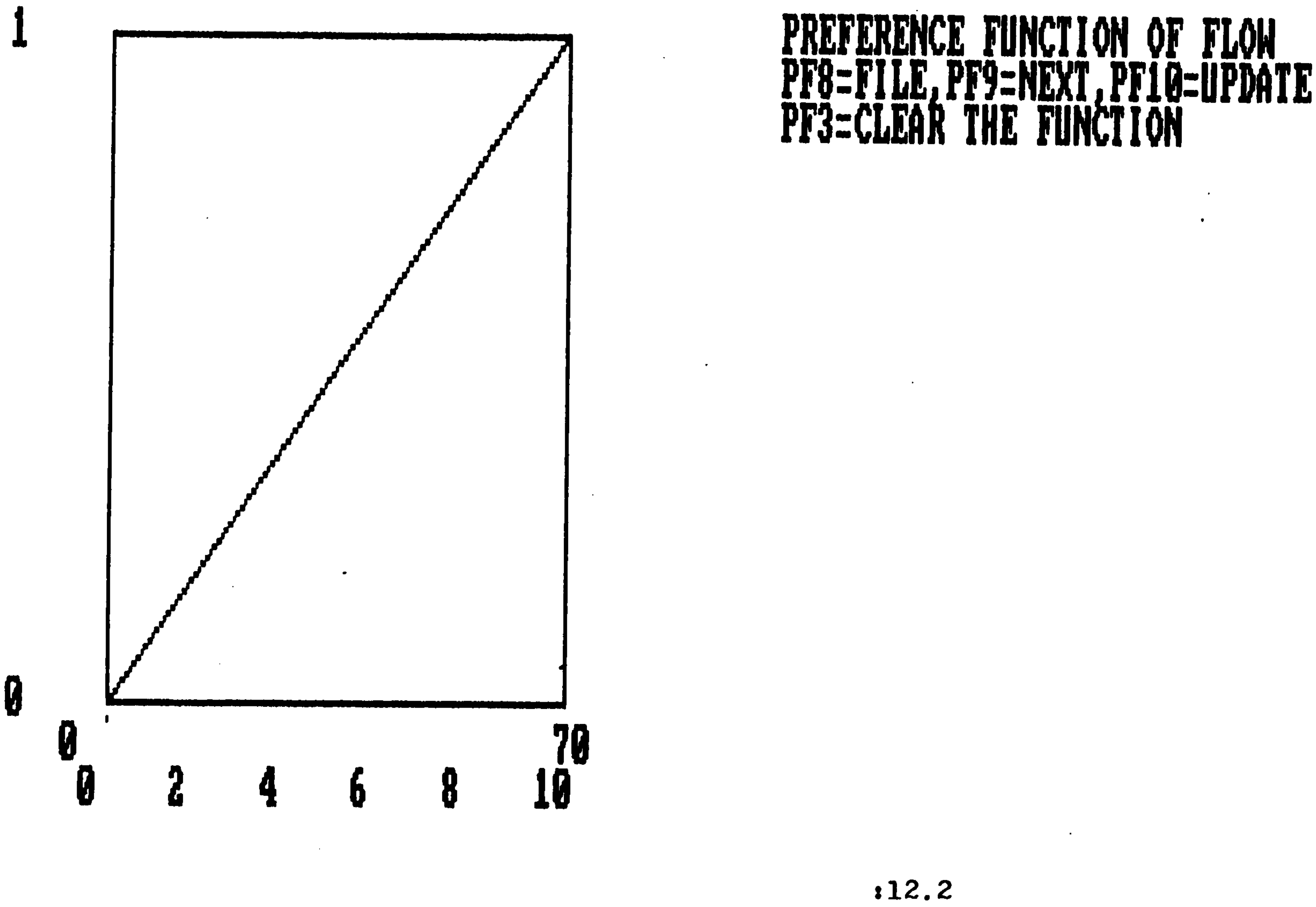
W 3 = .15

W 4 = .3

W 5 = .1

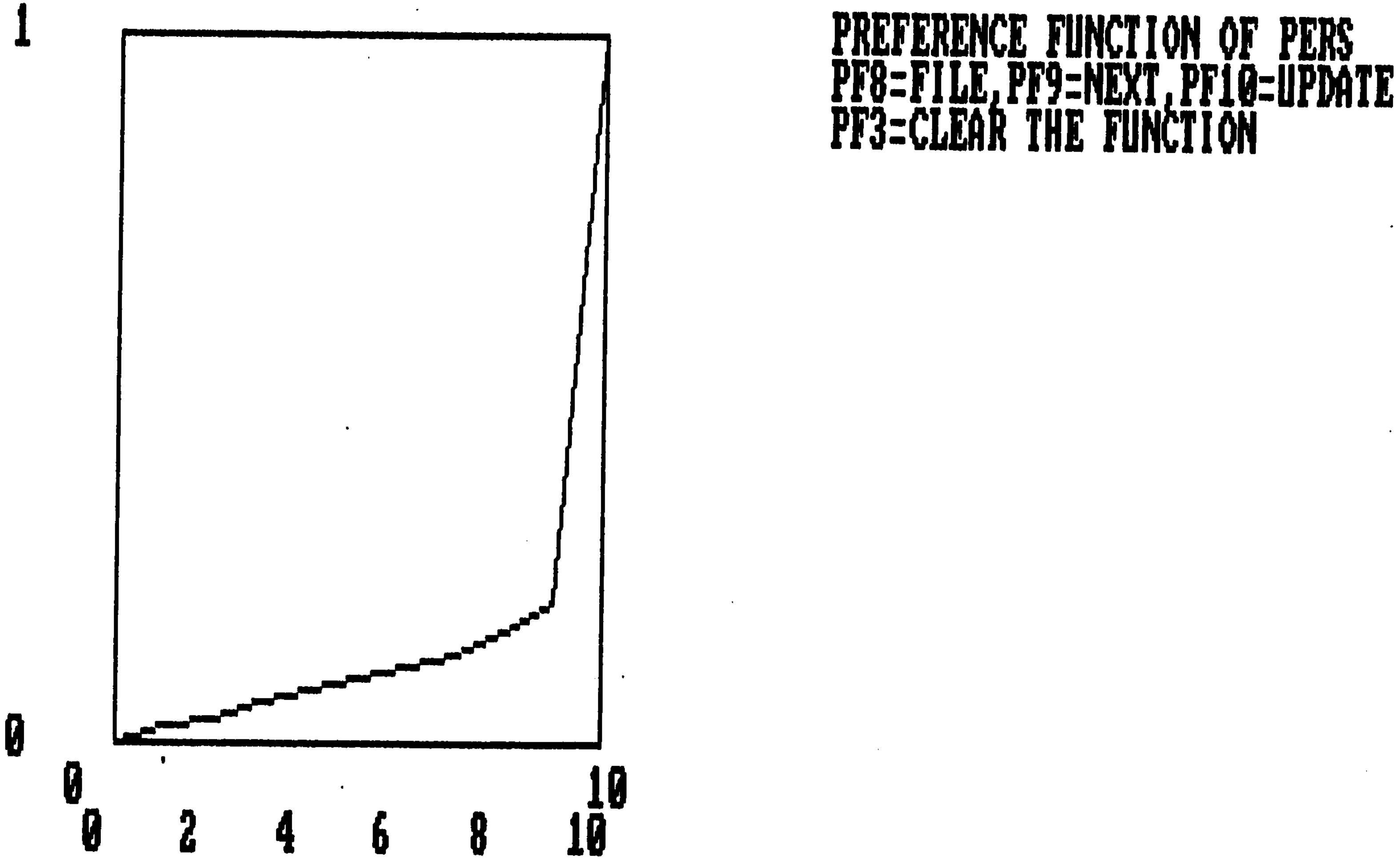
PF10=UPDATE
PF8=FILE

:11.2



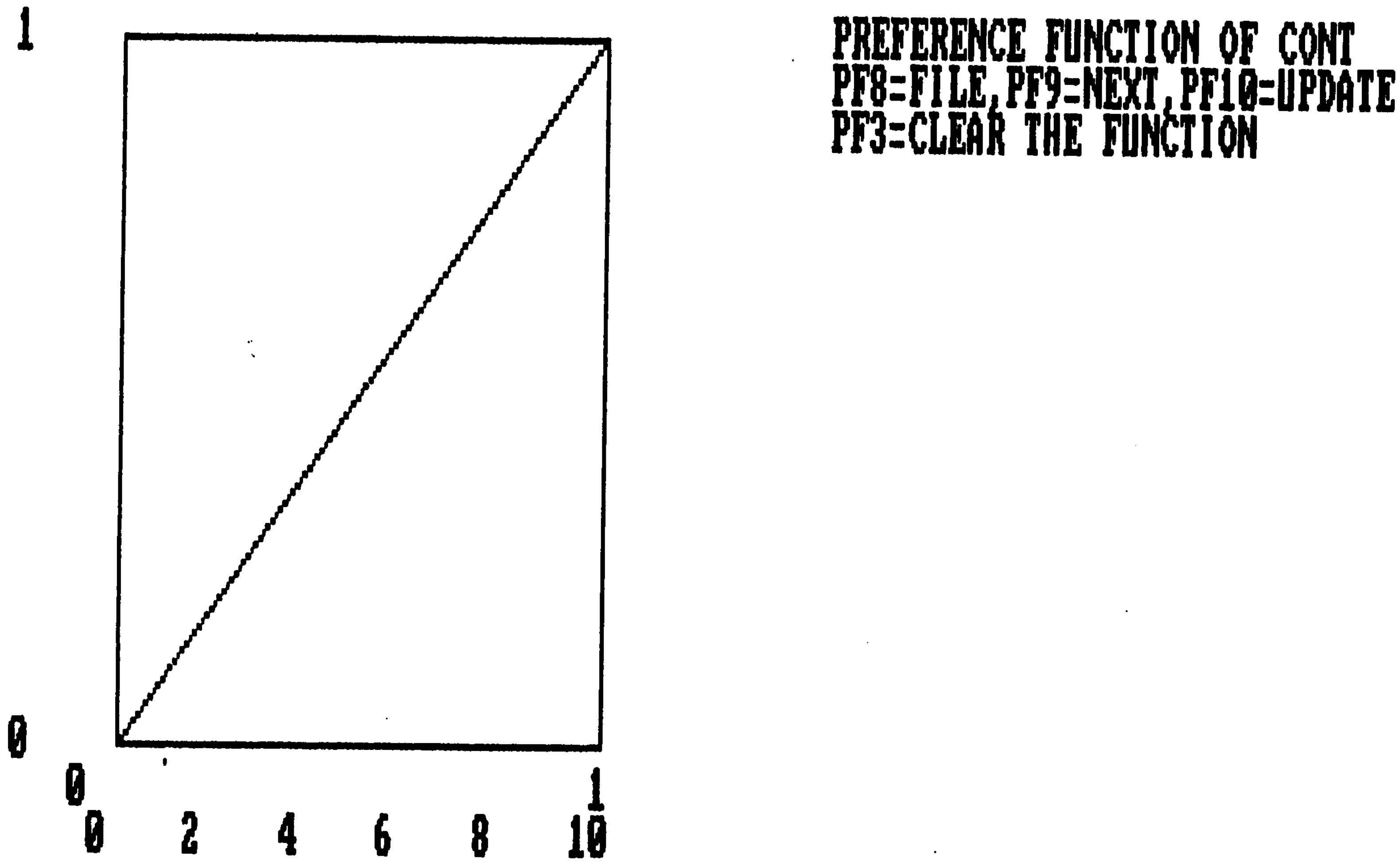
PREFERENCE FUNCTION OF FLOW
PF8=FILE,PF9=NEXT,PF10=UPDATE
PF3=CLEAR THE FUNCTION

:12.2



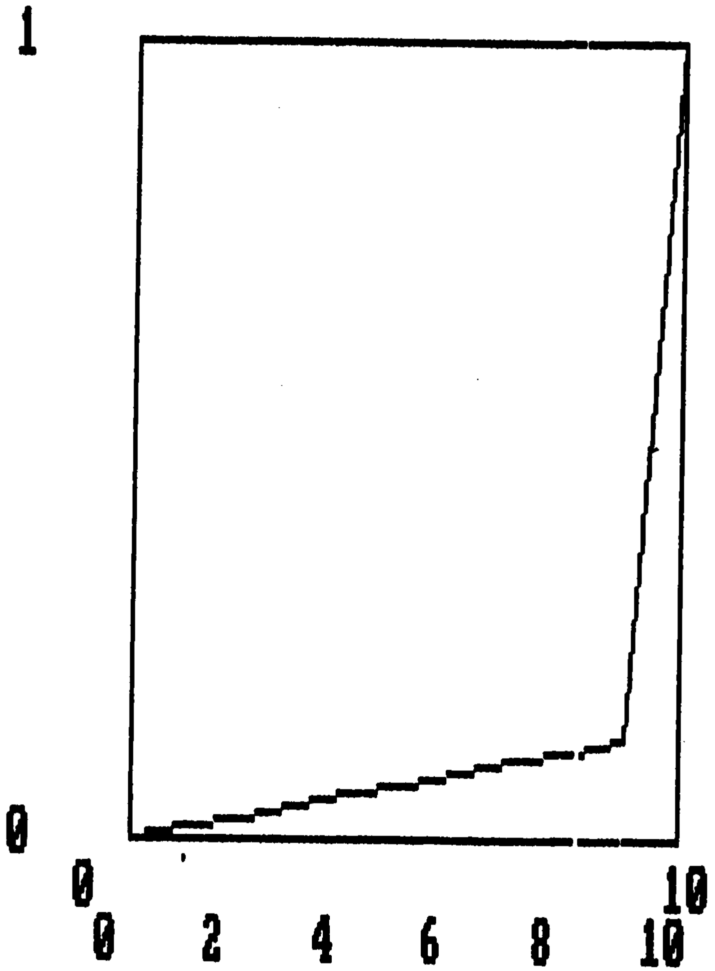
PREFERENCE FUNCTION OF PERS
PF8=FILE,PF9=NEXT,PF10=UPDATE
PF3=CLEAR THE FUNCTION

:12.3



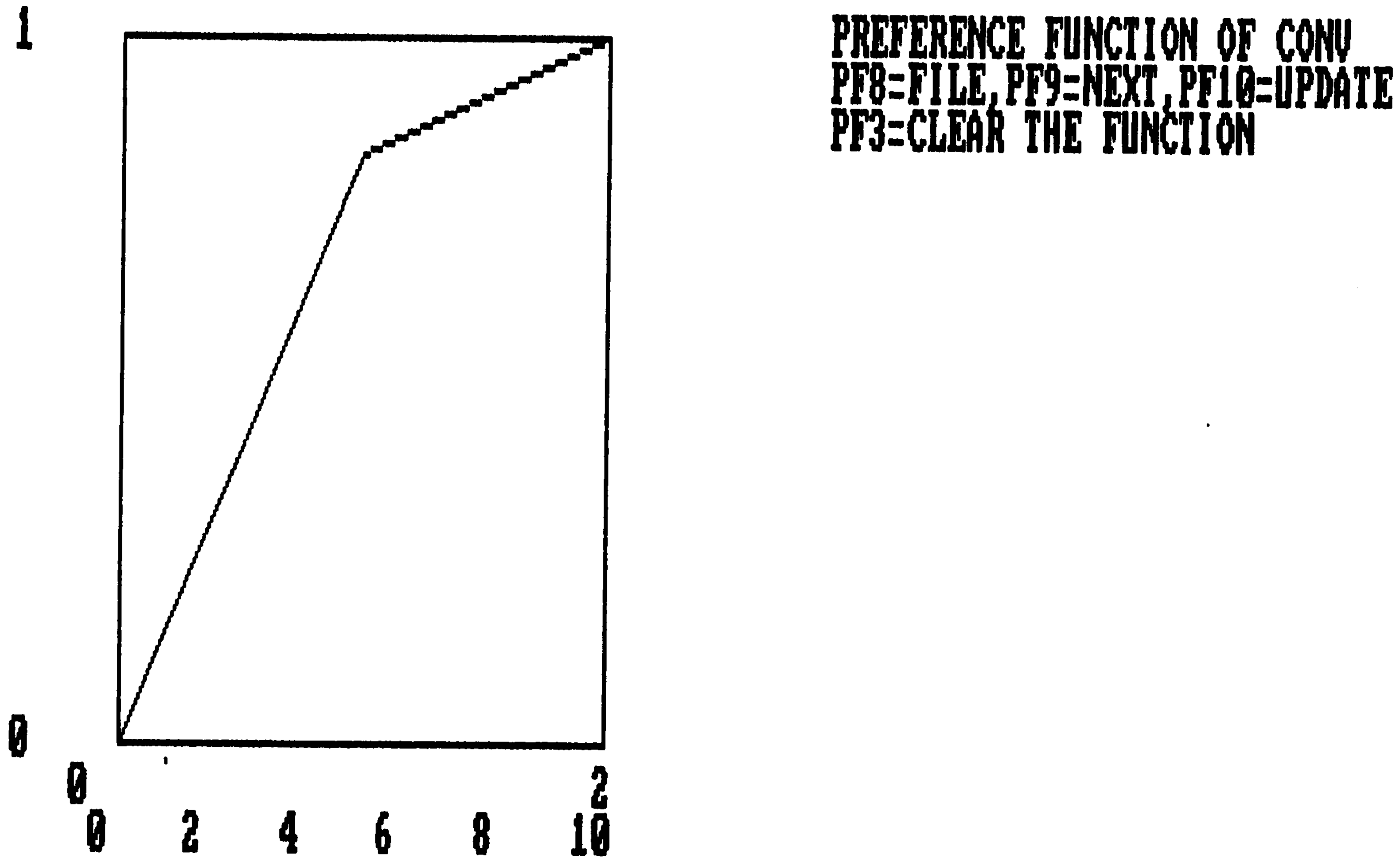
```
PREFERENCE FUNCTION OF CONT
PF8=FILE,PF9=NEXT,PF10=UPDATE
PF3=CLEAR THE FUNCTION
```

:12.4



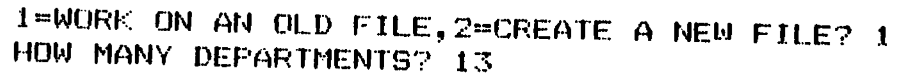
```
PREFERENCE FUNCTION OF INFO
PF8=FILE,PF9=NEXT,PF10=UPDATE
PF3=CLEAR THE FUNCTION
```

:12.5



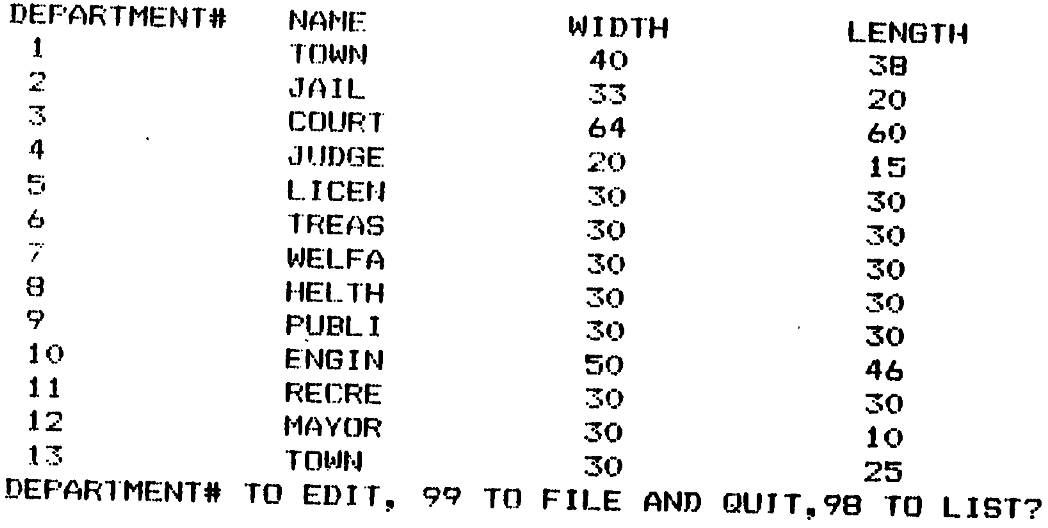
PREFERENCE FUNCTION OF CONV
PF8=FILE,PF9=NEXT,PF10=UPDATE
PF3=CLEAR THE FUNCTION

:12.6



```
1=WORK ON AN OLD FILE,2=CREATE A NEW FILE? 1
HOW MANY DEPARTMENTS? 13
```

:13.1



```
DEPARTMENT#    NAME          WIDTH        LENGTH
  1            TOWN           40           38
  2            JAIL           33           20
  3            COURT          64           60
  4            JUDGE          20           15
  5            LICEN          30           30
  6            TREAS          30           30
  7            WELFA          30           30
  8            HELTH          30           30
  9            PUBLI          30           30
 10            ENGIN          50           46
 11            RECRE          30           30
 12            MAYOR          30           10
 13            TOWN           30           25
DEPARTMENT# TO EDIT, 99 TO FILE AND QUIT,98 TO LIST?
```

:13.2

```
CRITERIA SPECIFICATION PROGRAM
NUMBER OF DEPARTMENTS AND NUMBER OF ATTRIBUTES? 13,5
```

:14.1

```
1=WORK ON THE OLD FILE, 2=CREATE A NEW FILE? 1
```

:14.2

```
PRESS PF10 TO START
```

:14.3

```
 1
CRITERIA(K)#                    1                 PF10=UPDATE,PF9=LIST,PF8=FILE
DEPARTMENT(I)#                  1
DEPARTMENT(J)#                  1
CURRENT SPEC.  0
```

:14.4

```
          1    2    3    4    5    6    7    8    9   10   11   12   13
 1        0   30   70    4    0    0    0    0   10    0    0   10    0
 2        0    0    8    2    0    0    0    0    0    0    0    0    0
 3        0    0    0   13    0    0    0    0    0    0    0    0    0
 4        0    0    0    0    0    0    0    0   10    0    0    0    0
 5        0    0    0    0    0   44    0    0   20    0    0    0    0
 6        0    0    0    0    0    0   20    0   20    0    0   30    0
 7        0    0    0    0    0    0    0   40   35    0    0    0    0
 8        0    0    0    0    0    0    0    0   46   40   10    0    0
 9        0    0    0    0    0    0    0    0    0   30    5    0    0
10        0    0    0    0    0    0    0    0    0    0   10    0    0
11        0    0    0    0    0    0    0    0    0    0    0    0    0
12        0    0    0    0    0    0    0    0    0    0    0    0   14
13        0    0    0    0    0    0    0    0    0    0    0    0    0
<ANY KEY> TO RETURN


CRITERIA(K)#    1
```

:14.5

```
 1
CRITERIA(K)#                   2              PF10=UPDATE,PF9=LIST,PF8=FILE
DEPARTMENT(I)#                 1
DEPARTMENT(J)#                 1
CURRENT SPEC.   0
```

:14.6

```
        1   2   3   4   5   6   7   8   9   10  11  12  13

 1      0   5   2   0   0   0   0   0   0   0   0   0   0
 2      0   0   8   0   0   0   0   0   0   0   0   0   0
 3      0   0   0   10  0   0   0   0   0   0   0   0   0
 4      0   0   0   0   0   0   0   0   0   0   0   0   0
 5      0   0   0   0   0   0   0   0   0   0   0   0   0
 6      0   0   0   0   0   0   0   0   0   0   0   0   0
 7      0   0   0   0   0   0   0   0   0   0   0   0   0
 8      0   0   0   0   0   0   0   0   0   0   0   0   0
 9      0   0   0   0   0   0   0   0   0   0   0   0   0
 10     0   0   0   0   0   0   0   0   0   5   5   0   0
 11     0   0   0   0   0   0   0   0   0   0   0   0   0
 12     0   0   0   0   0   0   0   0   0   0   0   0   0
 13     0   0   0   0   0   0   0   0   0   0   0   0   0
<ANY KEY> TO RETURN


CRITERIA(K)#   2
```

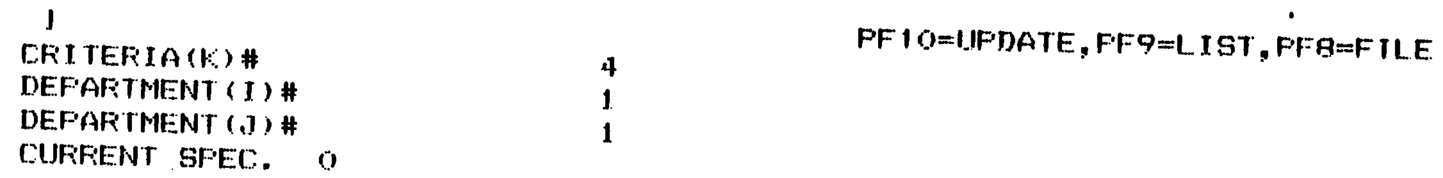
:14.7

```
 1
CRITERIA(K)#                          3                  PF10=UPDATE,PF9=LIST,PF8=FILE
DEPARTMENT(I)#                        1
DEPARTMENT(J)#                        1
CURRENT SPEC.    0
```

:14.8

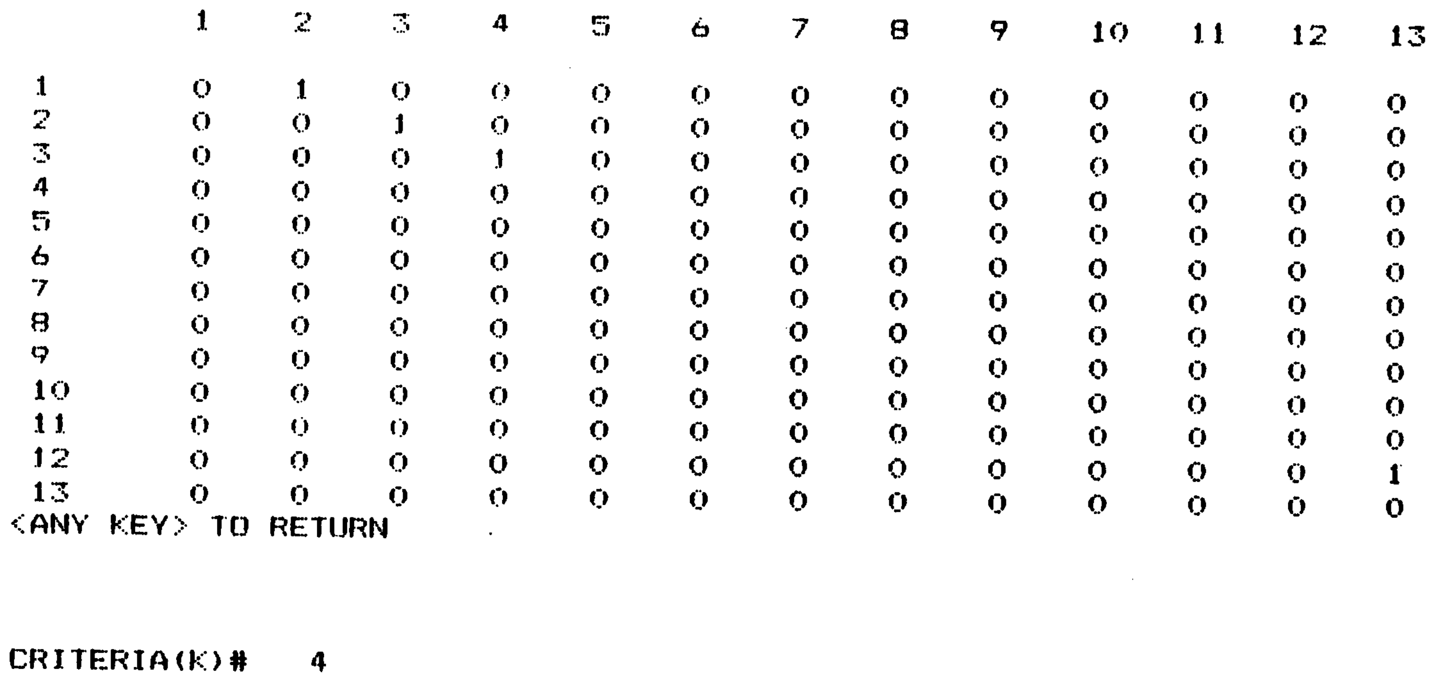
```
          1    2    3    4    5    6    7    8    9   10   11   12   13
 1        0    0    0    0    0    0    0    0    0    0    0    0    0
 2        0    0    0    0    0    0    0    0    0    0    0    0    0
 3        0    0    0   10    0    0    0    0    0    0    0    0    0
 4        0    0    0    0    0    0    0    0    0    0    0    0    0
 5        0    0    0    0    0    0    0    0    0    0    0    0    0
 6        0    0    0    0    0   10    0    0    0    0    0    0    0
 7        0    0    0    0    0    0    8    0    8    0    0    0    0
 8        0    0    0    0    0    0    0    9    0    0    0    0    0
 9        0    0    0    0    0    0    0    0    7    0    3    0    0
10        0    0    0    0    0    0    0    0    0   10    0    0    0
11        0    0    0    0    0    0    0    0    0    0    0    0    0
12        0    0    0    0    0    0    0    0    0    0    0    0    0
13        0    0    0    0    0    0    0    0    0    0    0    0    0
<ANY KEY> TO RETURN


CRITERIA(K)#    3
```

:14.9



```
 J
CRITERIA(K)#                     4                             PF10=UPDATE,PF9=LIST,PF8=FILE
DEPARTMENT(I)#                   1
DEPARTMENT(J)#                   1
CURRENT SPEC.   0
```

:14.10



```
           1    2    3    4    5    6    7    8    9   10   11   12   13

 1         0    1    0    0    0    0    0    0    0    0    0    0    0
 2         0    0    1    0    0    0    0    0    0    0    0    0    0
 3         0    0    0    1    0    0    0    0    0    0    0    0    0
 4         0    0    0    0    0    0    0    0    0    0    0    0    0
 5         0    0    0    0    0    0    0    0    0    0    0    0    0
 6         0    0    0    0    0    0    0    0    0    0    0    0    0
 7         0    0    0    0    0    0    0    0    0    0    0    0    0
 8         0    0    0    0    0    0    0    0    0    0    0    0    0
 9         0    0    0    0    0    0    0    0    0    0    0    0    0
 10        0    0    0    0    0    0    0    0    0    0    0    0    0
 11        0    0    0    0    0    0    0    0    0    0    0    0    0
 12        0    0    0    0    0    0    0    0    0    0    0    0    1
 13        0    0    0    0    0    0    0    0    0    0    0    0    0
<ANY KEY> TO RETURN


CRITERIA(K)#    4
```

:14.11

```
 1
CRITERIA(K)#                      5                        PF10=UPDATE,PF9=LIST,PF8=FILE
DEPARTMENT(I)#                    1
DEPARTMENT(J)#                    1
CURRENT SPEC.   0

                                                                     :14.12


          1    2    3    4    5    6    7    8    9    10   11   12   13

 1        0    0    0    0    0    0    0    0    0    0    0    0    0
 2        0    0    0    0    0    0    0    0    0    0    0    0    0
 3        0    0    0    0    0    0    0    0    0    0    0    0    0
 4        0    0    0    0    0    0    0    0    0    0    0    0    0
 5        0    0    0    0    0    2    1    1    0    0    0    0    0
 6        0    0    0    0    0    0    2    2    2    0    0    0    0
 7        0    0    0    0    0    0    0    2    2    0    0    0    0
 8        0    0    0    0    0    0    0    0    2    0    2    0    0
 9        0    0    0    0    0    0    0    0    0    2    0    0    0
 10       0    0    0    0    0    0    0    0    0    0    0    0    0
 11       0    0    0    0    0    0    0    0    0    0    0    0    0
 12       0    0    0    0    0    0    0    0    0    0    0    0    0
 13       0    0    0    0    0    0    0    0    0    0    0    0    0
<ANY KEY> TO RETURN


CRITERIA(K)#   5

                                                                     :14.13
```

```
HOW MANY DEPARTMENT, ATTRIBUTES? 13,5
```

:15.1

```
1=OLD MATRIX, 2=RECALCULATE? 1
```

:15.2

```
          1    2    3    4    5    6    7    8    9   10   11   12   13
   1    0.0000.4420.3050.0170.0000.0000.0000.0000.0430.0000.0000.0430.000
   2    0.4420.0000.3590.0090.0000.0000.0000.0000.0000.0000.0000.0000.000
   3    0.3050.3590.0000.6560.0000.0000.0000.0000.0000.0000.0000.0000.000
   4    0.0170.0090.6560.0000.0000.0000.0000.0000.0000.0000.0000.0000.000
   5    0.0000.0000.0000.0000.0000.4390.0830.0830.0860.0000.0000.0000.000
   6    0.0000.0000.0000.0000.4390.0000.2020.1000.2020.0000.0000.1290.000
   7    0.0000.0000.0000.0000.0830.2020.0000.2900.2500.0000.0000.0000.000
   8    0.0000.0000.0000.0000.0830.1000.2900.0000.3120.1710.1490.0000.000
   9    0.0430.0000.0000.0430.0860.2020.2500.3120.0000.3920.0350.0000.000
  10    0.0000.0000.0000.0000.0000.0000.0000.1710.3920.0000.0430.0000.000
  11    0.0000.0000.0000.0000.0000.0000.0000.1490.0350.0430.0000.0000.000
  12    0.0430.0000.0000.0000.0000.1290.0000.0000.0000.0000.0000.0000.360
  13    0.0000.0000.0000.0000.0000.0000.0000.0000.0000.0000.0000.3600.000
<ANY KEY> TO MAIN1
```

:15.3

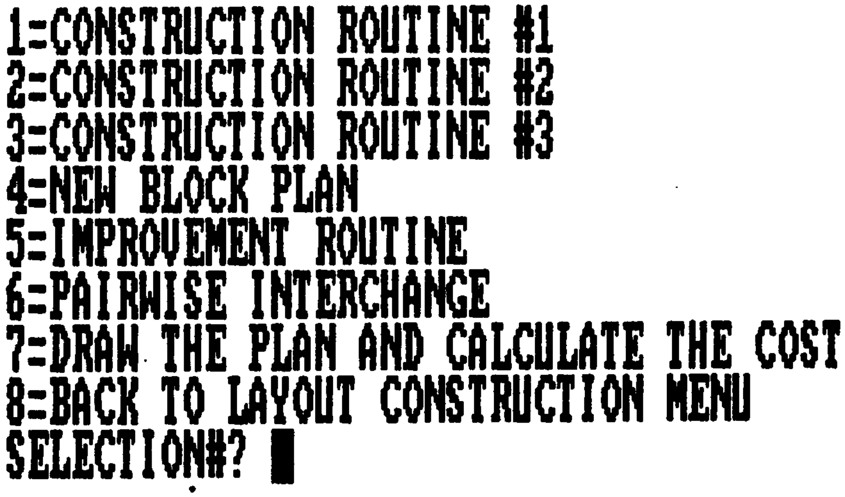
```
BLOCK PLAN     3 * 5
(ANY KEY) TO CONTINUE
```

1	2	3	4	5
6	7	8	9	10
11	12	13	*	*

:21.1

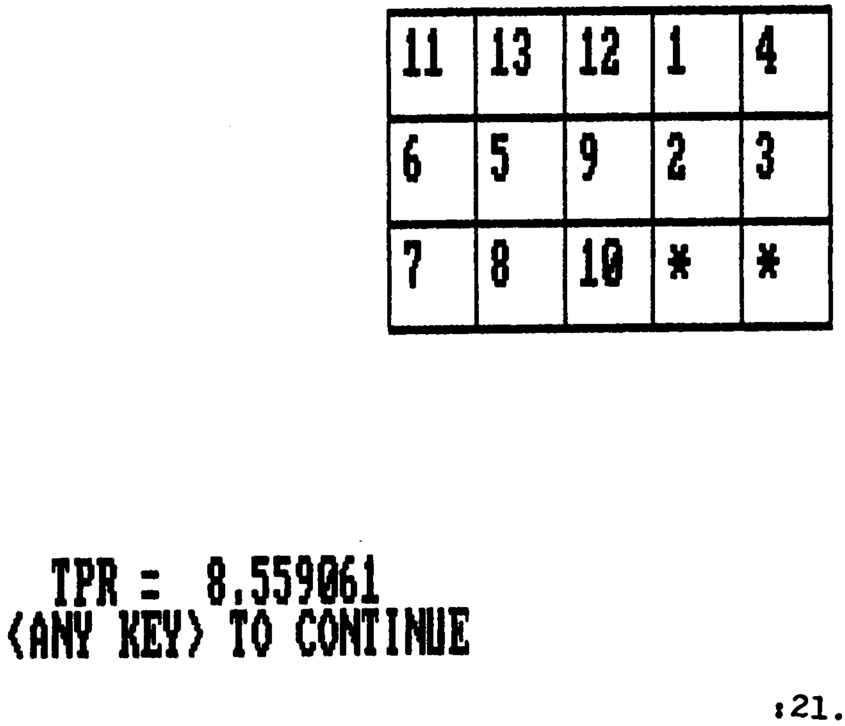
```
          1    2    3    4    5    6    7    8    9    10   11   12   13

1         0    1    2    3    4    1    2    3    4    5    2    3    4
2         1    0    1    2    3    2    1    2    3    4    3    2    3
3         2    1    0    1    2    3    2    1    2    3    4    3    2
4         3    2    1    0    1    4    3    2    1    2    5    4    3
5         4    3    2    1    0    5    4    3    2    1    6    5    4
6         1    2    3    4    5    0    1    2    3    4    1    2    3
7         2    1    2    3    4    1    0    1    2    3    2    1    2
8         3    2    1    2    3    2    1    0    1    2    3    2    1
9         4    3    2    1    2    3    2    1    0    1    4    3    2
10        5    4    3    2    1    4    3    2    1    0    5    4    3
11        2    3    4    5    6    1    2    3    4    5    0    1    2
12        3    2    3    4    5    2    1    2    3    4    1    0    1
13        4    3    2    3    4    3    2    1    2    3    2    1    0
DISTANCE MATRIX
<ANY KEY> TO CONTINUE
```

:21.2



```
1=CONSTRUCTION ROUTINE #1
2=CONSTRUCTION ROUTINE #2
3=CONSTRUCTION ROUTINE #3
4=NEW BLOCK PLAN
5=IMPROVEMENT ROUTINE
6=PAIRWISE INTERCHANGE
7=DRAW THE PLAN AND CALCULATE THE COST
8=BACK TO LAYOUT CONSTRUCTION MENU
SELECTION#? █
```

:21.3



11	13	12	1	4
6	5	9	2	3
7	8	10	*	*

```
  TPR =  8.559061
<ANY KEY> TO CONTINUE
```

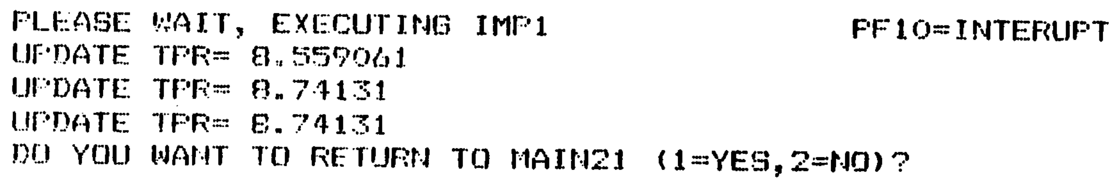
:21.4

```
EXECUTING CONSTRUCTION ROUTINE #1, PLEASE WAIT
LOCATION 5     DEPARTMENT 11
LOCATION 10    DEPARTMENT 13
LOCATION 4     DEPARTMENT 12
LOCATION 9     DEPARTMENT 10
LOCATION 1     DEPARTMENT 5
LOCATION 3     DEPARTMENT 6
LOCATION 2     DEPARTMENT 7
LOCATION 6     DEPARTMENT 8
LOCATION 11    DEPARTMENT 9
LOCATION 7     DEPARTMENT 4
LOCATION 8     DEPARTMENT 3
LOCATION 12    DEPARTMENT 1
LOCATION 13    DEPARTMENT 2
<ANY KEY> TO RETURN TO MAIN21
```

:CR10

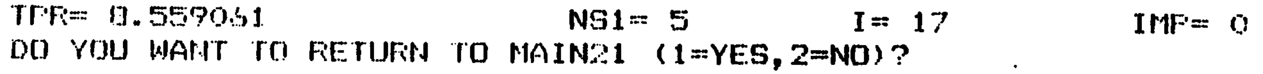
```
EXECUTING CONSTRUCTION ROUTINE #3, PLEASE WAIT
LOCATION 8     DEPARTMENT 9
LOCATION 13    DEPARTMENT 10
LOCATION 12    DEPARTMENT 8
LOCATION 11    DEPARTMENT 7
LOCATION 6     DEPARTMENT 6
LOCATION 7     DEPARTMENT 5
LOCATION 2     DEPARTMENT 13
LOCATION 3     DEPARTMENT 12
LOCATION 4     DEPARTMENT 1
LOCATION 9     DEPARTMENT 2
LOCATION 10    DEPARTMENT 3
LOCATION 5     DEPARTMENT 4
LOCATION 1     DEPARTMENT 11
<ANY KEY> TO RETURN TO MAIN21
```

:CR30



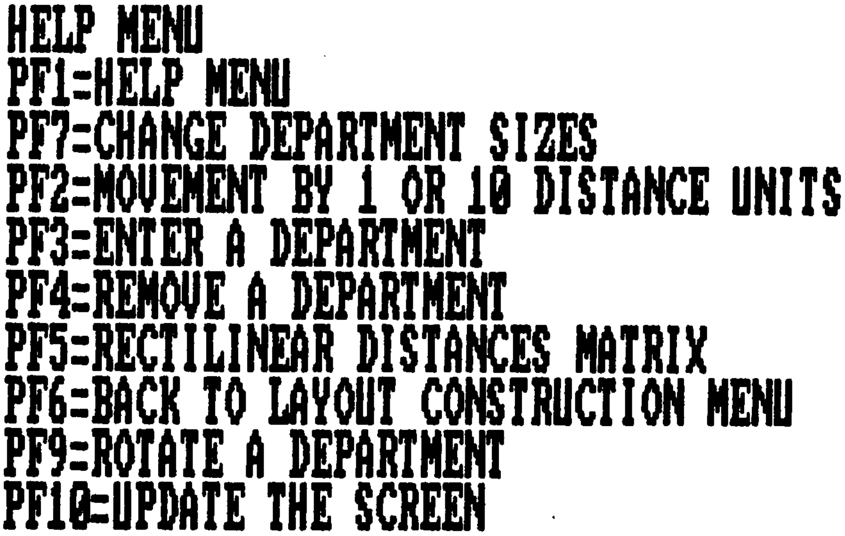
```
PLEASE WAIT, EXECUTING IMP1                     PF10=INTERUPT
UPDATE TPR= 8.559061
UPDATE TPR= 8.74131
UPDATE TPR= 8.74131
DO YOU WANT TO RETURN TO MAIN21  (1=YES,2=NO)?
```

:IMP1



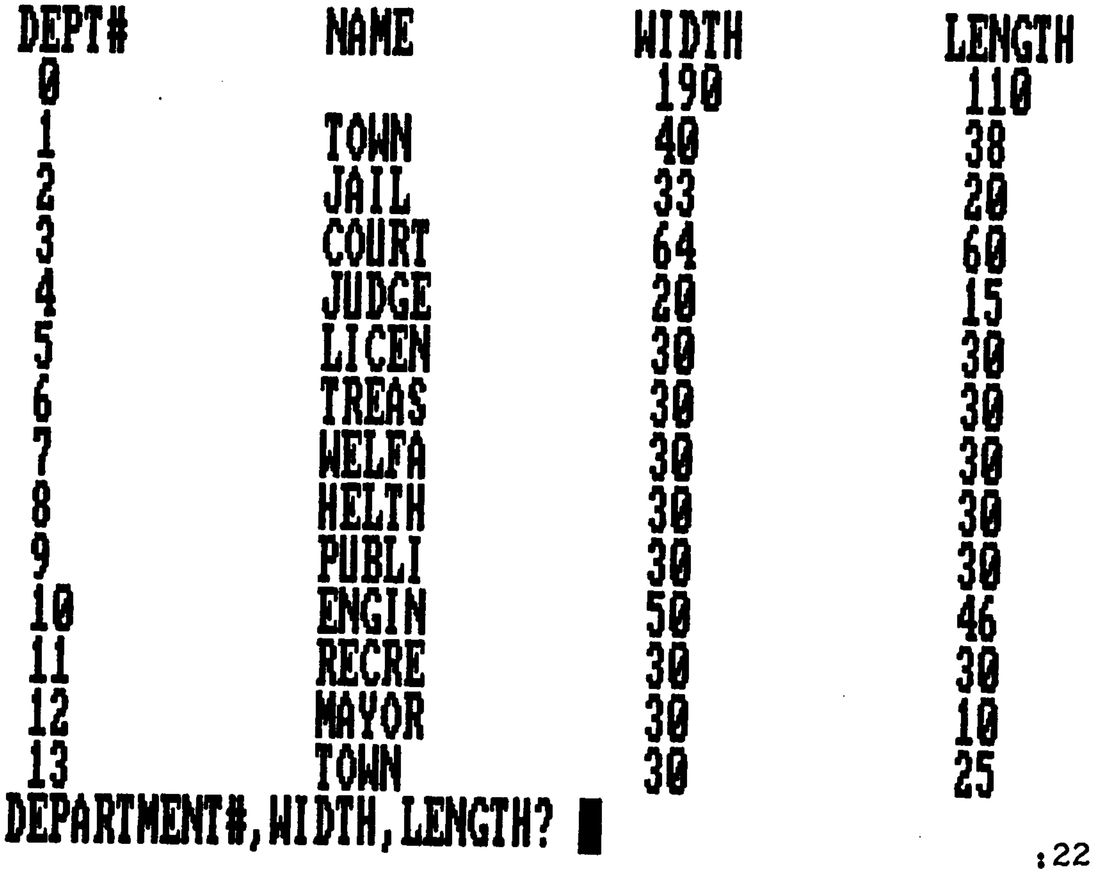
```
TPR= 8.559061             NS1= 5         I= 17          IMP= 0
DO YOU WANT TO RETURN TO MAIN21  (1=YES,2=NO)?
```

:CQAP



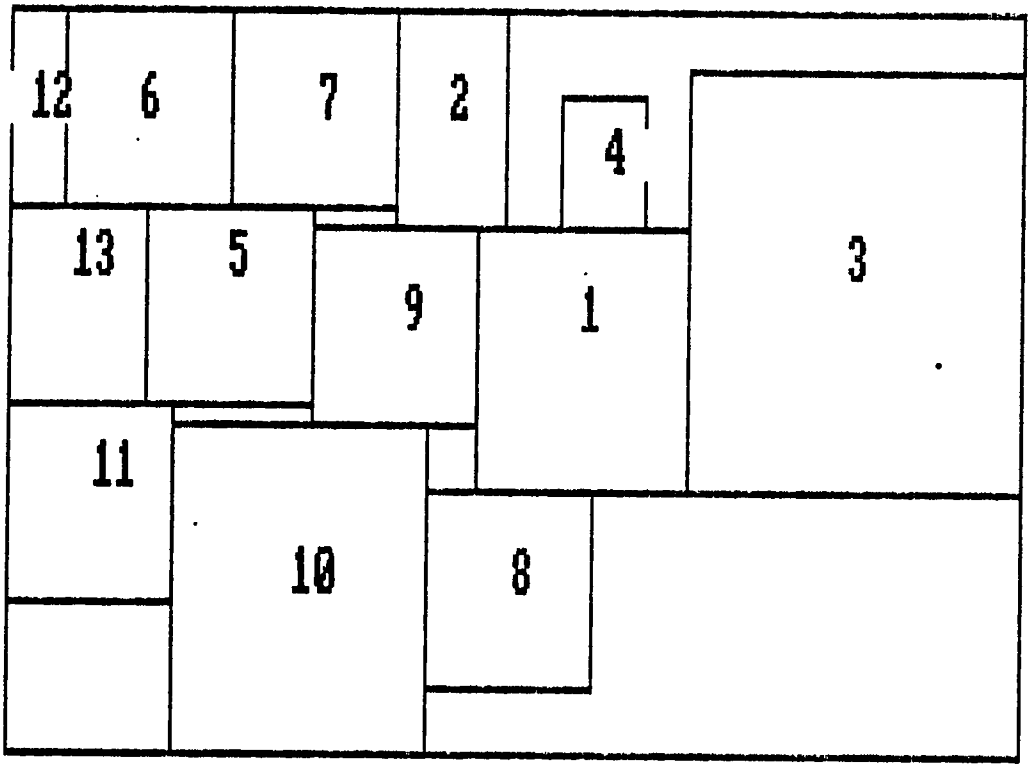
```
HELP MENU
PF1=HELP MENU
PF7=CHANGE DEPARTMENT SIZES
PF2=MOVEMENT BY 1 OR 10 DISTANCE UNITS
PF3=ENTER A DEPARTMENT
PF4=REMOVE A DEPARTMENT
PF5=RECTILINEAR DISTANCES MATRIX
PF6=BACK TO LAYOUT CONSTRUCTION MENU
PF9=ROTATE A DEPARTMENT
PF10=UPDATE THE SCREEN
```

:22.1

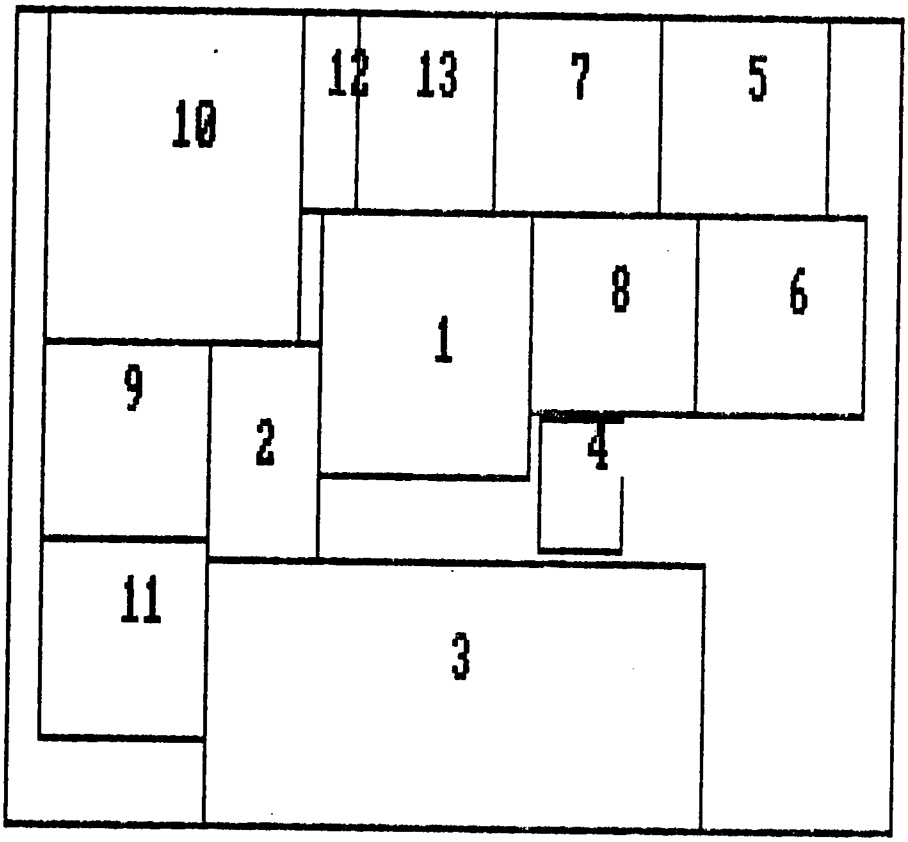


```
DEPT#         NAME          WIDTH         LENGTH
 0                           190           110
 1            TOWN           40            38
 2            JAIL           33            20
 3            COURT          64            60
 4            JUDGE          20            15
 5            LICEN          30            30
 6            TREAS          30            30
 7            WELFA          30            30
 8            HELTH          30            30
 9            PUBLI          30            30
 10           ENGIN          50            46
 11           RECRE          30            30
 12           MAYOR          30            10
 13           TOWN           30            25
DEPARTMENT#,WIDTH,LENGTH? █
```

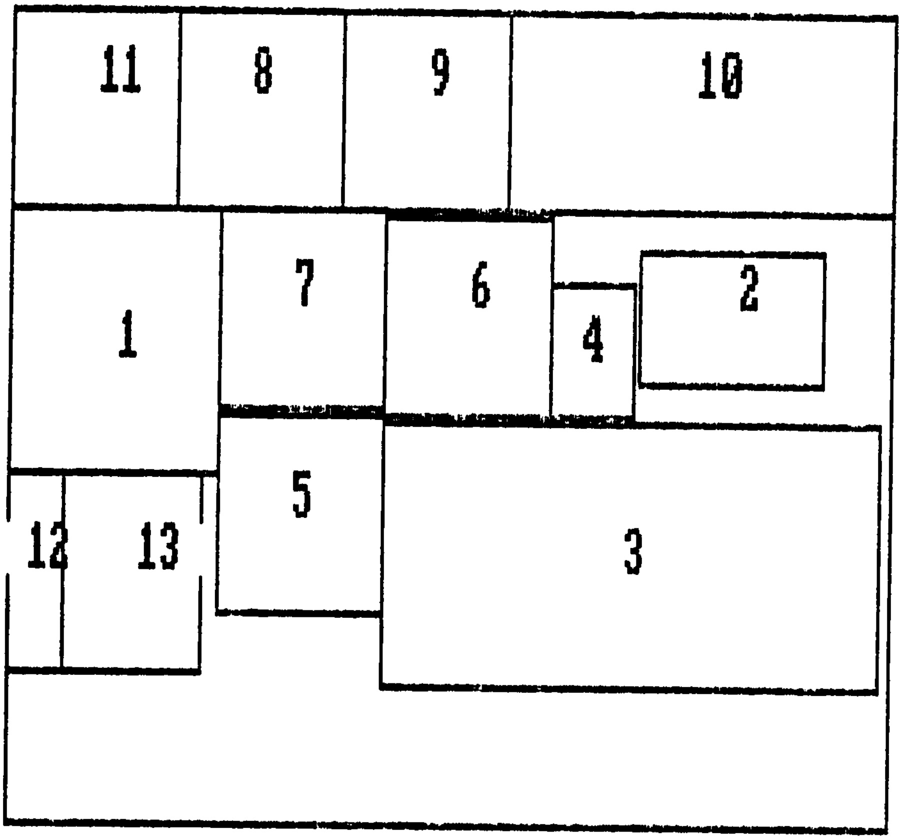
:22.2



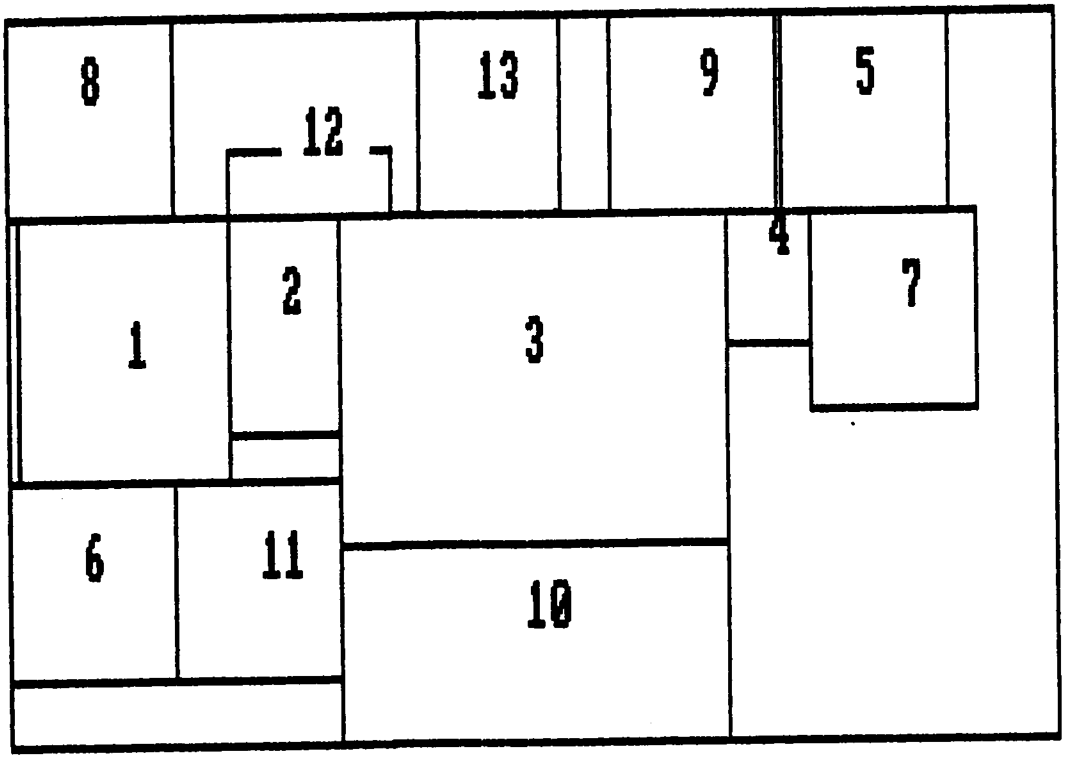
:ALT.#1



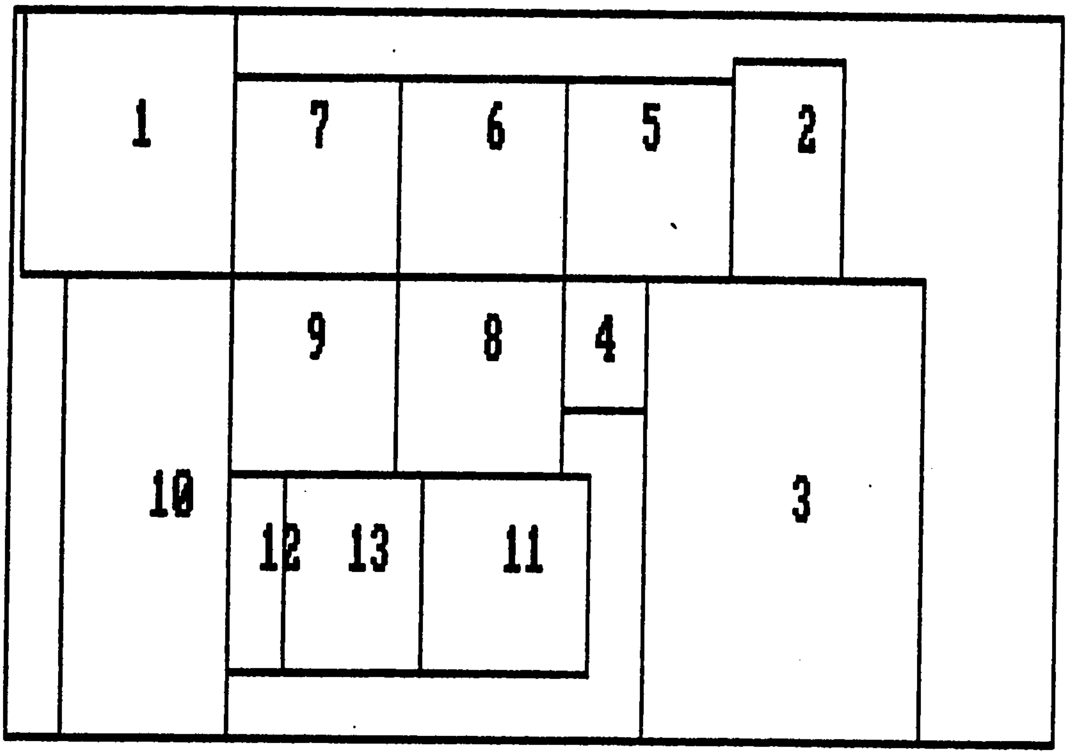
:ALT.#2



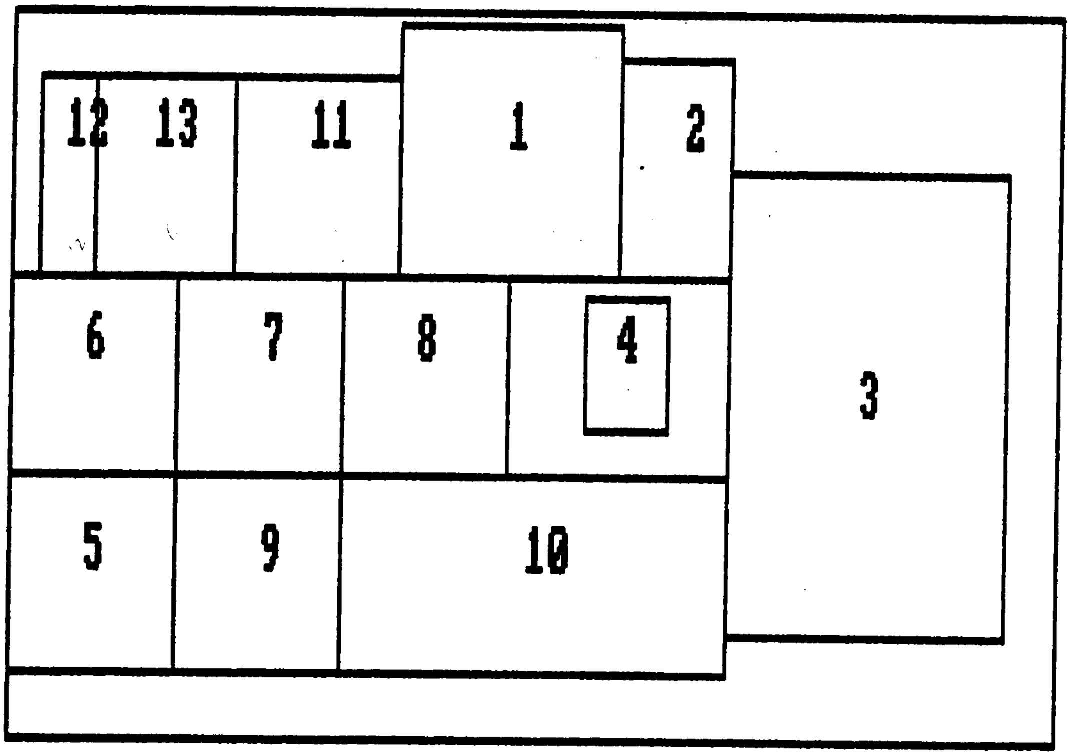
:ALT.#3



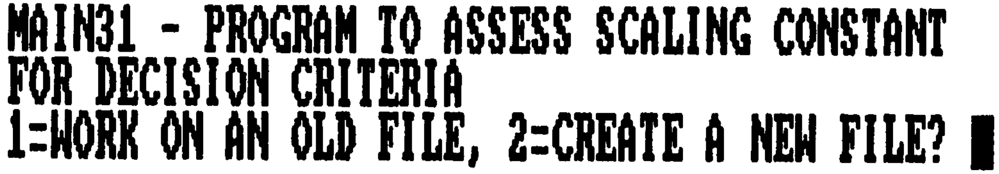
:ALT.#4



:ALT.#5

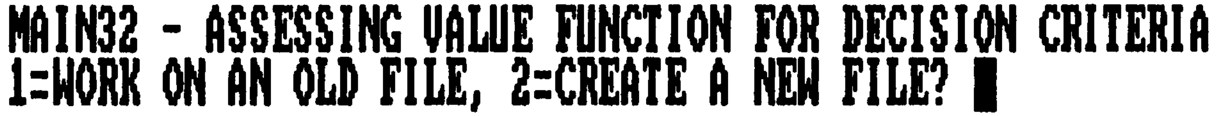


:ALT.#6



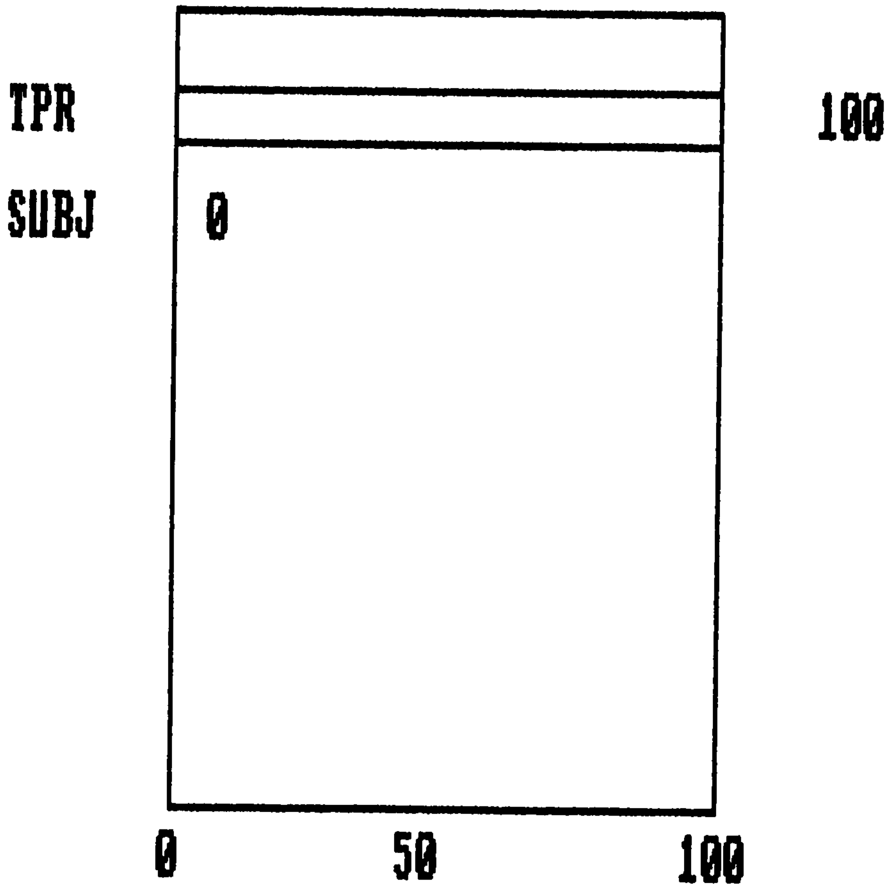
```
MAIN31 - PROGRAM TO ASSESS SCALING CONSTANT
FOR DECISION CRITERIA
1=WORK ON AN OLD FILE, 2=CREATE A NEW FILE? █
```

:31.1



```
MAIN32 - ASSESSING VALUE FUNCTION FOR DECISION CRITERIA
1=WORK ON AN OLD FILE, 2=CREATE A NEW FILE? █
```

:32.1



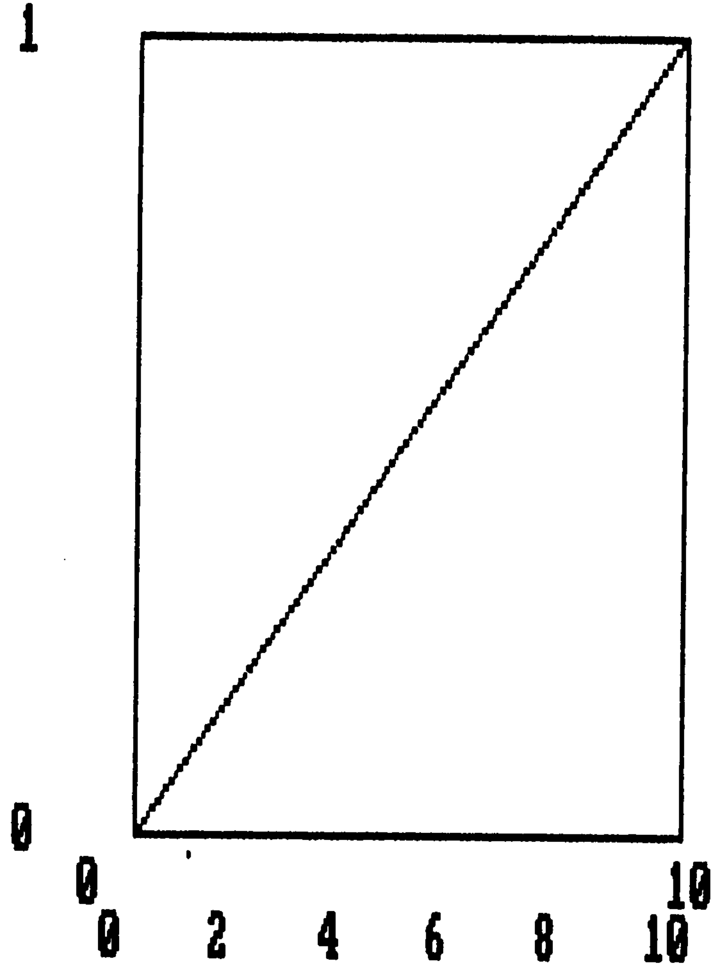
W 1 = 1

W 2 = 0

RATE ATTRIBUTES IN IMPORTANCE
PRESERVING RATIO

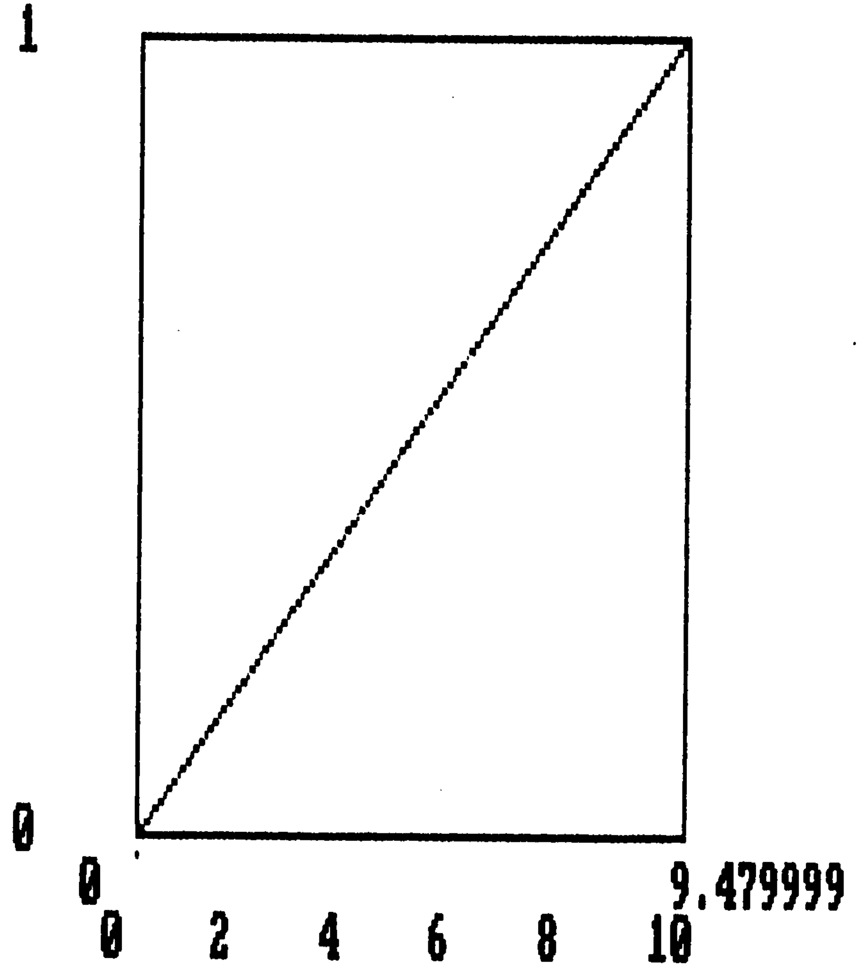
PF10=UPDATE
PF8=FILE

:31.2



PREFERENCE FUNCTION OF SUBJ
PF8=FILE,PF9=NEXT,PF10=UPDATE

:32.2



PREFERENCE FUNCTION OF TPR
PF8=FILE,PF9=NEXT,PF10=UPDATE

:32.3

```
1=WORK ON AN OLD FILE, 2=CREATE A NEW FILE? 1
HOW MANY CRITERIA,ALTERNATIVES? 2,6
```

:33.1

```
CRITERION#    NAME        MIN        MAX          SPEC.
 1            TPR         0           9.479999     7.79
 2            SUBJ        0          10            0
-----------------------------------------------------------
ALTERNATIVE#   3
CRITERION#    NAME        MIN        MAX          SPEC.
 1            TPR         0           9.479999     8.12
 2            SUBJ        0          10            0
-----------------------------------------------------------
ALTERNATIVE#   4
CRITERION#    NAME        MIN        MAX          SPEC.
 1            TPR         0           9.479999     6.1
 2            SUBJ        0          10            0
-----------------------------------------------------------
ALTERNATIVE#   5
CRITERION#    NAME        MIN        MAX          SPEC.
 1            TPR         0           9.479999     7.94
 2            SUBJ        0          10            0
-----------------------------------------------------------
ALTERNATIVE#   6
CRITERION#    NAME        MIN        MAX          SPEC.
 1            TPR         0           9.479999     9.01
 2            SUBJ        0          10            0
ALTERNATIVE# TO EDIT ANY ALTERNATIVE, 99=FILE AND QUIT,98=PRINT ALL?
```

:33.2

```
ALTERNATIVE      2
CRITERION#      NAME            MIN             MAX             SPEC.
 1              TPR              0               9.479999        7.79
 2              SUBJ             0               10              0
CRITERION       TPR              0               9.479999        7.79
NEW SPEC.? 7.79
CRITERION       SUBJ             0               10              0
NEW SPEC.?
ALTERNATIVE# TO EDIT ANY ALTERNATIVE, 99=FILE AND QUIT,98=PRINT ALL?
```

:33.4

```
ALTERNATIVE      1
CRITERION#      NAME            MIN             MAX             SPEC.
 1              TPR              0               9.479999        8.600001
 2              SUBJ             0               10              0
CRITERION       TPR              0               9.479999        8.600001
NEW SPEC.? 8.6
CRITERION       SUBJ             0               10              0
NEW SPEC.? 0
ALTERNATIVE# TO EDIT ANY ALTERNATIVE, 99=FILE AND QUIT,98=PRINT ALL?
```

:33.3

```
HOW MANY DECISION CRITERIA, ALTERNATIVES? 2,6
```

:34.1

```
ALTERNATIVE#     1               SCORE= .9071732
CRITERION#       NAME            MIN            MAX            SPEC.
 1               TPR              0              9.479999       0
 2               SUBJ             0              10             0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3? 2
ALTERNATIVE#     2               SCORE= .82173
CRITERION#       NAME            MIN            MAX            SPEC.
 1               TPR              0              9.479999       0
 2               SUBJ             0              10             0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3? 3
ALTERNATIVE#     3               SCORE= .8565401
CRITERION#       NAME            MIN            MAX            SPEC.
 1               TPR              0              9.479999       0
 2               SUBJ             0              10             0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3? 4
ALTERNATIVE#     4               SCORE= .64346
CRITERION#       NAME            MIN            MAX            SPEC.
 1               TPR              0              9.479999       0
 2               SUBJ             0              10             0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3?
```

:34'.2

```
ALTERNATIVE#    3              SCORE= .8565401
CRITERION#      NAME           MIN                MAX           SPEC.
 1              TPR             0                  9.479999      0
 2              SUBJ            0                  10            0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3? 4
ALTERNATIVE#    4              SCORE= .64346
CRITERION#      NAME           MIN                MAX           SPEC.
 1              TPR             0                  9.479999      0
 2              SUBJ            0                  10            0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3? 5
ALTERNATIVE#    5              SCORE= .8375529
CRITERION#      NAME           MIN                MAX           SPEC.
 1              TPR             0                  9.479999      0
 2              SUBJ            0                  10            0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3? 6
ALTERNATIVE#    6              SCORE= .950422
CRITERION#      NAME           MIN                MAX           SPEC.
 1              TPR             0                  9.479999      0
 2              SUBJ            0                  10            0
---------------------------------------------------------------------
ALT.# TO REVIEW, OTHERWISE RETURN TO MAIN3?
```

:34.3

APPENDIX C. RANDOM MATRIX GENERATOR

Program to generate random matrices is shown in this appendix. These matrices are used for testing performance of CORELAP and the proposed procedure.

```
10 REM RANDOM MATRIX GENERATOR
20 DIM P(25,25)
30 INPUT "NUMBER OF DEPT. AND FILE NAME";ND,N$
40 RANDOMIZE : CLS
50 FOR I=1 TO ND
60 FOR J=1 TO ND
70 IF I=J THEN P(I,J)=0 : GOTO 100
80 IF I>J THEN P(I,J)=P(J,I) : GOTO 100
90 X=RND*10 : P(I,J)=INT(X)
100 LOCATE I,J*2 : PRINT P(I,J)
110 NEXT
120 NEXT
130 OPEN N$+".DAT" FOR OUTPUT AS #1
140 FOR I=1 TO ND
150 FOR J=1 TO ND
160 WRITE #1,P(I,J)
170 NEXT
180 NEXT
190 CLOSE #1
200 END
```