COMPUTER AIDED ARCHITECTURAL DESIGN WITH SPECIAL REFERENCE TO SPATIAL PLANNING AND SYNTHESIS OF BUILDINGS

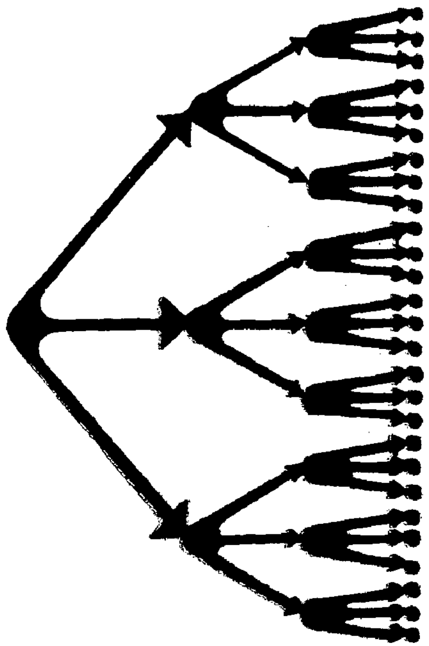
A DISSERTATION

Submitted in partial fulfilment
of the requirements for the award of the degree
of
MASTER OF ARCHITECTURE

By
S. SUKUMAR

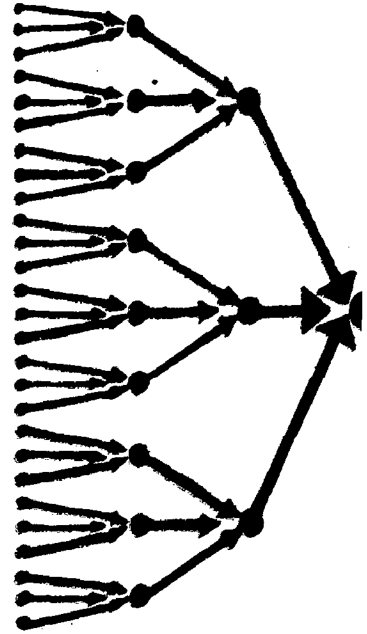
DEPARTMENT OF ARCHITECTURE AND PLANNING
UNIVERSITY OF ROORKEE
ROORKEE 247 667 (INDIA)

March, 1985



COMPUTER AIDED ARCHITECTURAL DESIGN

S. SUKUMAR — M. ARCH — 1983-84
UNIVERSITY OF ROORKEE.



CERTIFICATE

Certified that the dissertation entitled **'COMPUTER AIDED ARCHITECTURAL DESIGN WITH SPECIAL REFERENCE TO SPATIAL PLANNING AND SYNTHESIS OF BUILDINGS'**, which is being submitted by **Shri S.SUKUMAR** in partial fulfilment of the requirements for the award of the degree of **MASTER OF ARCHITECTURE,** in the Department of Architecture, University of Roorkee, Roorkee, is a record of the student's own work carried out by him under our supervision and guidance. The matter embodied in this dissertation has not been submitted for the award of any other degree or diploma.

This is to further certify that he has worked for a period of eight months from August 1984 to March 1985 for the preparation of this dissertation at this University.

(Dr. G.C. NAYAK)
Professor,
Department of Civil Engineering,
University of Roorkee,
ROORKEE.

(KULDIP.C.KAMBO) 2/4-85
Professor,
Department of Architecture & Planning,
University of Roorkee,
ROORKEE.

Roorkee
March 1985.

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to Prof. Kuldip C. Kambo for his invaluable guidance and for providing the necessary words of encouragement whenever they were due.

To DR. G. C. Nayak, I offer my profound thanks, for the assistance rendered in the coding and execution of the computer programs and for the enthusiasm shown by him throughout the endeavour.

I take this opportunity to acknowledge the help and suggestions offered by various members of the faculty of the Department of Architecture and Planning, University of Roorkee, Roorkee.

Last but not the least, I would like to thank all my various friends and colleagues, for having shown such great affection and warmth, in the short span that we spent together.

(S. SUKUMAR)

P R E F A C E

Computer Aided Architectural Design is very much in its infancy in India. There is practically very little work done in this field although interest in computing science and technology has been kindled to a remarkable extent. It is high time architects came out of the rut of time-honoured design methods and took a fresh look at the world around them.

I have chosen this area of spatial planning and synthesis as this forms the essence of an architect's work. Researchers and other interested professionals all over the world are still attempting to evolve a near perfect strategy which would simulate the complex working mechanism of an Architect's brain. Many researchers have, in fact, lost interest in this area, after a couple of decades of dedicated work, realising that problems that require the synthesis of many demands, mostly with subjective values, are best handled by the human brain. Here, then, lies the perfect challenge of attempting to find a solution where none exists.

This dissertation is a humble endeavour towards providing myself and others dedicated to this area of specialisation, with a solid foundation for further research.

ROORKEE. S. SUKUMAR

CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER-4

CHAPTER-5

ABSTRACT

This dissertation is an attempt at providing the background material for further research in the field of Spatial Planning and Synthesis in Computer Aided Architectural Design. An attempt has been made at reviewing the available literature and at compiling the various formulations in term of objectives and constraints and the solution synthesis procedures currently in use.

The author has had to rely heavily on material published about a decade back due to a paucity of current literature.

An endeavour has been made at coding and executing some of the programs available and at evaluating the technique practically. The logical extension of this dissertation would be the working out a design methodology incorporating computer as aids.

1. PREAMBLE

The contemporary Architect practises in a very complex environment. He is faced with a predicament which threatens to assume phenomenal proportions. Problems have surfaced within the profession itself and pressures have mounted from outside, the nature of buildings has become more and more complicated; there are restricted time schedules; there is an abundance of data and very little time to analyse it. Absence of definite design standards relevant to conditions in India only add to the burden.

On the other hand, pressures have developed in the form of the discerning public - the clients and the users, who have become more and more demanding. The premium on land has also necessi tated the need for maximum utilization of space. It is inconceivable that the Architect's mind can really absorb and hold for recall a significant amount of the information, which is available to his creative pallet, with any reliability.

It is believed that the extension of the human ability to comprehend, react and respond can be found in the development of new techniques in design/planning, especially computer applications.

The computer has become an integral component of the Twentiety century and it does have the potential to reshape the entire practice of Architectural Design, as it has affected every other phase of human lives. The computer's capacity for storage/retrieval, its phenomenal speed and its ability to display selected information in unique ways make it an indispensible tool for the Architect.

One of the recurring problems faced in the time-honoured Design process is that valuable data is lost within a project and from project to project. The computer extends the Architect's memory so that he is able to have access to any and all information within seconds. The Architect can store/retrieve information in any format he chooses.

The computer's ability to edit/update information eliminates the storage of erroneous and redundant data.

One of the oft quoted characteristics of the computer is its phenomenal speed. If, by using the computer, the Architect is able to shorten the Design process, it means savings for the clients and in turn is translated into lesser working costs for the users of the facility.

The fourth aspect is the capability of the computer to display information is selected ways. It can produce information in the form of printed matter, voluminous reports, abstracts, or in the form of two dimensional or three dimensional graphics, This aspect can be of immense help to the Architect.

The flexible application of computer techniques allows the Architect to develop a multilayered, multi-faceted approach to design rather than a single design philosophy.

It is essential to remember that the computer does not design. What it does is to eliminate the haphazard approach to design and hence influence the quality of the design. Computers are not meant to replace man but are machines meant to relieve him from wasting valuable time in doing routine tasks which can be done faster and more accurately on a computer.

2. THE SCOPE OF THE STUDY, STATEMENT OF OBJECTIVES AND REVIEW OF LITERATURE :

2.1 THE SCOPE :

Although the computer has been used extensivly abroad in the various aspects of environmental design*, the author has decided to restrict himself to one particular area of study, namely, 'SPATIAL PLANNING AND SYNTHESIS'. Space planning constitutes the heart of any Architectural Design methodology. It is one of the basic tasks an Architect undertakes in the design process to create spatial order for an architectural problem. It deals with solution generation to satisfy the problem requirements.

The farsighted goal of this project is an endeavour towards the development of a design methodology incorporating computers as aids. This methodology should be valid for circumstances that prevail in India.

The logical starting point for a study of this nature would be an analysis of techniqués which have been developed abroad. This analysis constitutes a definition of the scope and limitations of those techniques..

---

* Refer APPENDIX II for details of the State of the art of Computer Applications in Environmental Design.

2.2 STATEMENT OF OBJECTIVES :

The broad objective of this study is to provide a solid foundation for further research in the area of Spatial Planning and Synthesis. The development of a workable design methodology incorporating computers as aids, and radically changing the haphazard approach to design would be the ultimate aim of an endeavour of this nature.

The immediate objectives which are suggested are :-

* Analysis of the existing techniques used in spatial planning.

* Determination of the potentials and limitations.

2.3 REVIEW OF LITERATURE :

The literature dealing with computer Applications in Architecture gave a very clear idea of the state of the art of Computer Aided Architectural Design. Some examples illustrate the point.

Computers in Architectural Design :

By David Campion (1968), written at the time when computer applications in architecture were a novelty, describes tentatively, the capabilities of the computer and possible applications in the field of architecture.

The Automated Architect :

By N. Cross (1977), a more confident statement of the impact of computers on architectural design. It deals more with the psychological impact of computers. It brings to light some questions as to whether computer aided design improves either the quality of the end product or the quality of the designers working life.

Spatial Synthesis in Computer Aided Building Design by Charles, M. Eastman (1975), is a collection of papers, providing a detailed survey of the current state of the art in having the computer contribute to spatial synthesis problems. Most of the papers present enough detail to allow comparison, replication and extension.

Perhaps, the most exhaustive and comprehensive compilation, the result of several years of research has been done by Dr. Kaiman Lee of the Environmental Design and Research Centre, Massachusetts.

His interest in this field of computer applications in environmental design began with his Master's thesis in Architecture from the Iowa State University, U.S.A. in 1969. This deals with the computer as an Architectural Design Tool : an exploration into certain multistorey building layouts.

Beginning with this, he has published several compilations.

Step towards an Integrated Design System for the Architect/Planner deals with an interactive approach to Computer Aided Design System (1973).

Computer Aided Spatial Planning (1976) reviews the state of art of Spatial Planning using computers.

Computer Program in Environmental Design (1974) is an exhaustive compilation of computer programs developed all over the world.

Bibliography of the Computer Applications in Environmental Design (1973) is a compilation of literature available on the topic.

Computer Aided Architectural Design by William J. Mitchell (1977) provides a comprehensive introduction to the fundamentals of computer aided architectural design for the student of architecture, for the architect in practice and the computer professional.

The Architecture Machine by Nicholas Negroparte (1975) deals with a series of experiments conducted by the author to build machines that are capable of artificial intelligence, the era of robot architects.

By far, the most upto date publication is by Reynolds, R.A. (1980), Computer methods for architects. This provides an insight into the current trends and the development of practical aids in the profession, rather than conceptual ideas.

This is a review of some of the literature.

3. STATE OF THE ART OF COMPUTER AIDED SPACE PLANNING

3.1 SPACE PLANNING DEFINED

Space planning is one of the basic task which an architect undertakes in the design process, to create spatial order for an architectural problem. It deals with solution generation to satisfy the problem requirements. Traditionally, the architect derives a spatial arrangement through intuitive interpretation of the ideal relationship between elements. Upon closer analysis, it is quite apparent that this intuition takes on a few fundamental steps (Figure 3.1) :

(a) Identifies each element involved and defines the relationships between each pair of elements:

(b) Establishes for each element the required area, and any specific configurations desired.

(c) Diagrams elements relationships by relating various elements to each other graphically as bubble diagrams.

(d) Transforms bubble diagrams into space relationship layouts by incorporating the area required for each element. This layout becomes a scaled drawing.

(e) Evaluates alternative arrangements according to program constraints, such as functional requirements, project budget and aesthetic considerations.

SPACE PLANNING :

1. IDENTIFY EACH ELEMENT
   - DRAWING
   - DINING
   - KITCHEN
   - BEDROOM
   - TOILET

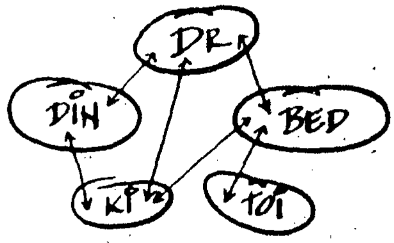
   DEFINE RELATIONSHIPS

   DR.
   DI.
   KI.
   BED.
   TOI.

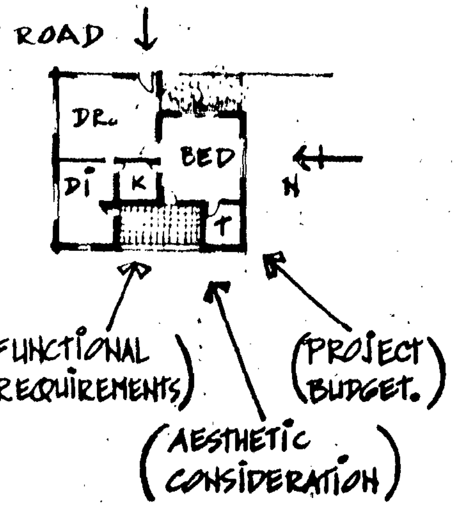
2. ESTABLISH FOR EACH ELEMENT – REQUIRED AREA & ANY SPECIFIC CONFIGURATION.

   DRAWING ROOM :
   AREA : 12'x 15'
   ACCESS FROM ROAD.
   ORIENTATION : EAST.

3. REPRESENT ELEMENT RELATIONSHIPS AS BUBBLE DIAGRAMS.



4. TRANSFORM BUBBLE DIAGS. INTO SCALED SPACE LAYOUT.



5. EVALUATE ALTERNATIVES ACCORDING TO PROGRAM CONSTRAINTS

Figure 3-1

COMPUTER AIDED ARCHITECTURAL DESIGN

S. SUKUMAR – M. ARCH - 1983-84
UNIVERSITY OF ROORKEE

However, as the complexity of element requirements multiplies, the task of arriving at an optimum solution or generating alternatives for evaluation becomes less manageable as well as time consuming. Since most architectural problems are usually overconstrained, no solution can possibly satisfy all criteria. Optimum solutions are compromises where the conflicts are minimized. The high speed digital computer can be an invaluable aid in generating solutions to the space allocation problem. With its large, accurate memory and low computational time, the computer can be used to generate as well as evaluate solutions.

3.2 COMPUTER AIDED SPACE PLANNING

Computer aided space allocation relates to the application of digital computers to the location of facilities within a two dimensional or three dimensional space. It has its origin in facility layout and location and has received considerable attention from architects, urban planners, economists, operations researchers, regional scientists and engineers. Each brings to the subject a different interpretation of the problem and different approaches to its solution. At a regional scale, space planning consists of determining the optimum location and distribution of various land use patterns, transportational facilities, warehouses and other large scale regional elements.

At an urban scale, space planning includes the distribution of urban transportation systems, the location of city services and the layout of buildings in a large urban complex.

At an architectural scale, space planning refers to the optional distribution of physical spaces within a building.

3.3 FORMULATION OF ARCHITECTURAL SPATIAL SYNTHESIS PROBLEMS

In general terms the problem may be stated as follows. given a data structure capable of representing a range of building designs find a state of the data structure (i.e., a particular design solution) such that specified objectives and/or constraints are complied with. From this formulation it can be seen that the quality or usefulness of a solution generated depends on three things :-

(a) Capability of the data structure to represent an appropriate range of potential solutions at an appropriate level of detail.

(b) Capability of the constraints and/or objectives to accurately and comprehensively reflect important design criteria.

(c) Capability of the solution generation procedure to efficiently find solutions which best fit the criteria.

The alternative formulations (in terms of objectives and constraints) of automated spatial synthesis problems is as follows.

3.3.1 Quadratic Assignment Formulation

The earliest and perhaps the still most popular formulation of the architectural spatial synthesis problem is as a quadratic assignment problem. This formulation is very general and may be employed for a wide variety of spatial arrangements problems in which circulation cost or some directly analogous objective is to be minimized. In addition to architectural problems, it may be used for such tasks such as location of warehouses or other facilities within transportation networks, regional land use planning and the backboard **wiring** problem in electronics.

An example of quadratic assignment problem as formulated by Koopmans and Beckmann (1957) is as follows. The basic task is to allocate a set of indivisible facilities to a set of locations in such a way that the following objective is minimised.

$$\text{Total Circulation Cost} = \sum_{i=1}^{n} \sum_{j=1}^{n} G_{ij}\, C_{ij}$$

where n = number of facilities to be assigned

($<$ number of location)

G_{ij} = measure of distance between pairs of located facilities i and j.

C_{ij} = a measure of circulation cost per unit distance between i and j.

G_{ij} = a measure of distance between pairs of located facilities i and j.

C_{ij} = a measure of circulation cost per unit distance between i and j.

In architectural floor planning applications, the set of locations is commonly taken as the set of cells in a square grid [Fig. 3.2 (a)]. This grid is of course, readily represented by a two-dimensional array of appropriate dimensions. The entire grid might be employed, or a polygon representing a building outline, lot limits, etc [Figure 3.2(b)]. Each space to be located in the plan is then considered to be composed of an appropriate number of square modul s according to floor area requirement [Figure 3.2 (c)]. Each square module is represented by an integer defining the space to which it belongs, so that the floor plan layout problem is represented as a problem of assigning integers to location in a two-dimensional array. Values for G_{ij} may be computed directly from array subscripts.

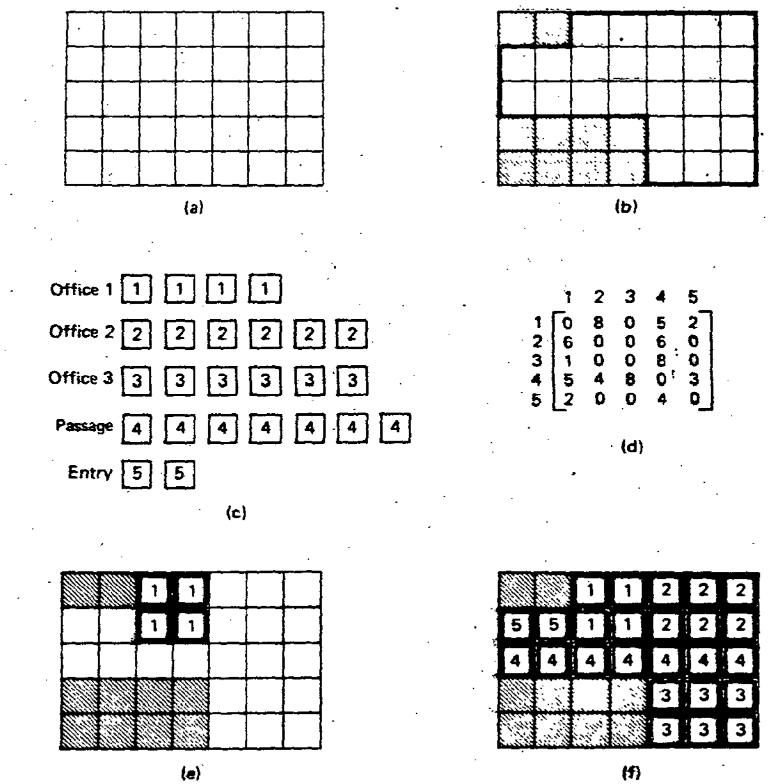
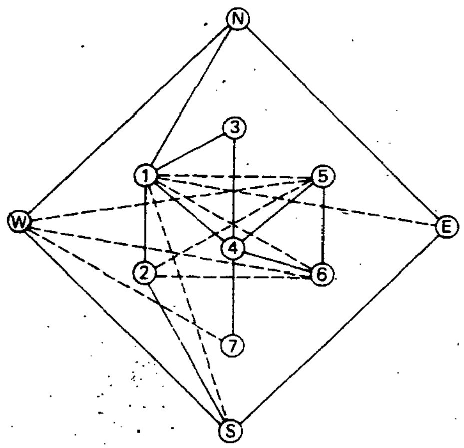


Figure 3-2
Floor plan layout as a quadratic assignment problem. (a) Grid of locations. (b) Building boundary within the grid. (c) Modules to be assigned. (d) Interaction matrix. (e) Preassigned space. (f) One of the many possible assignments of modules to locations.

Values for C_{ij} may be taken from an interaction matrix of circulation data, [Figure 3.2 (d)].
A quadratic assignment formulation is appropriate only in situations where circulation efficiency or some directly analogous objective is regarded as the primary determinant of the plan. This might conceivably be more or less the case in facilities such as industrial plants, warehouses, offices, hospitals, educational facilities, libraries and departmental stores.

3.3.2 Adjacency Requirements Graph Formulation

An alternate approach to formulation of floor plan layout problems is to define a set of adjacency requirements between spaces which must be met. In other words, to specify an adjacency requirements graph which must be a subgraph of the dual graph of the plan. This graph may be encoded in matrix form. (Figure 3.3) An adjacency requirements graph may be employed in conjunction with almost any technique for representation of built form. The adjacency requirements graph formulation contrasts with the quadratic assignment formulation in two important ways. First it explicitly specifies in detail the structure of the spatial system, rather than requiring optimization of an aggregated measure of the systems performance. This makes it more natural and appropriate for general use in architectural design. Second, it relies upon constraints rather than an objective to identify acceptable solutions. This means that, depending



1 Living room
2 Kitchen
3 Bathroom
4 Hall
5 Bedroom 1
6 Bedroom 2
7 Bedrocm 3

Solid line = required adjacency
Dashed line = prohibited adjacency

Figure 3-3
An adjacency requirements graph for a floor plan layout problem.

on the properties of the constraints, there may be multiple acceptable solutions, a unique solution, or no solution.

Dimensional constraints and objectives :

An adjacency requirements graph, by itself, usually provides insufficient definition of the characteristics of acceptable solutions. It is normally necessary to give information on dimensional, proportion, and shape constraints as well, if a useful result is to be generated.

Dimensional constraints may be stated by means of equalities for example,

length = x,

or in terms of inequalities, for example,

length > x_1

length < x_2.

Typically, dimensional constraints are applied to the following properties either of a particular object, space or of the building as a whole: length, width, perimeter, floor area, surface area, volume, maximum and minimum allowable distance.

Ratio constraints and objectives:

In some cases, important properties of built forms may be described by means of ratios of dimensions or area.

For example, the plan proportion of a rectangular room is defined as the ratio of length : width and it may be desired to impose proportion constraints in order to prevent the room either from becoming too long and narrow or too square. For non-rectangular objects, proportion constraints may be applied to the bounding rectangle (Figure 3.4).

Since compactness of built form clearly relates in a general way to construction cost and operating efficiency of a building, attempts are some times made to optimise some kind of compactness ratio such as perimeter, floor area, or surface : volume ratio (Figure 3.5).

Shape constraints :

The usual approach to constraining the shape of a two-dimensional object, such as the plan of the room, is to employ some measure of similarity to a reference shape such as a rectangle. A minimum bounding rectangle is drawn around the shape and the shape's percentage coverage of its minimum bounding rectangle is calculated. The value is low if the shape is very irregular, and 100% if it is in fact rectangular, (Figure 3.6).

Constraint graphs:

Some appropriate combination of adjacency, dimensional ratio and shape objective and constraints is very often sufficient to appropriately describe the criteria for a spatial synthesis non-rectilinear shape.

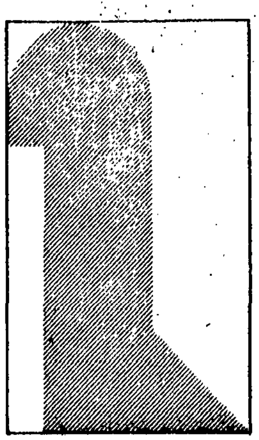


Figure 3-4

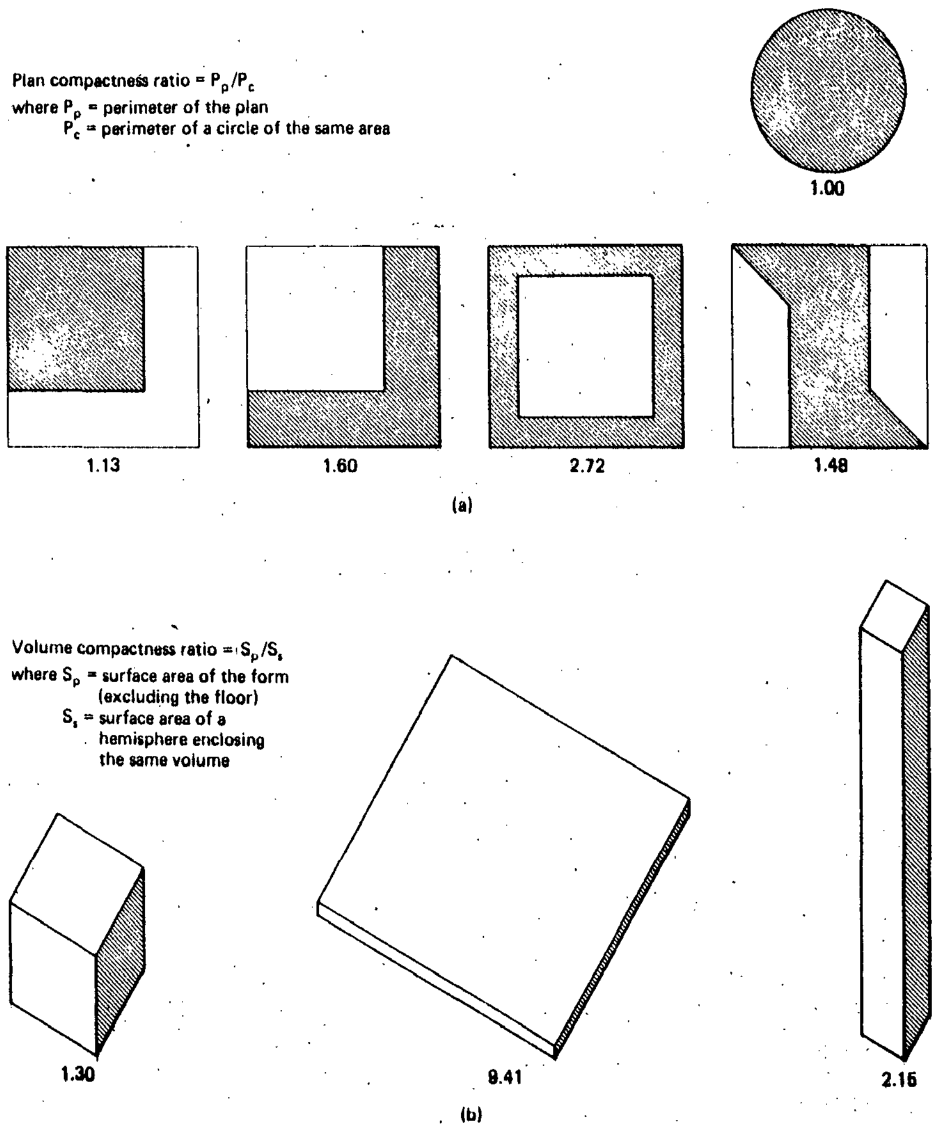


Figure 3-5

Compactness ratios. (a) Plan compactness ratios for several shapes enclosing the same areas. (b) Volume compactness ratios for several forms enclosing the same volumes.

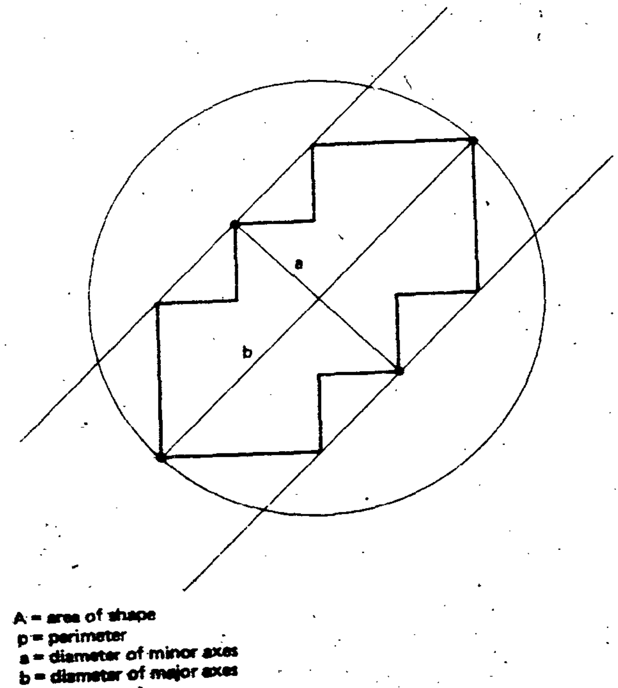
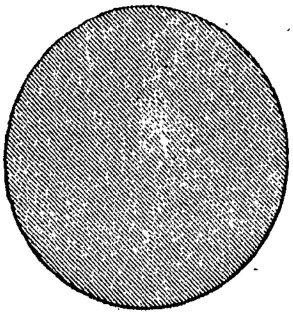
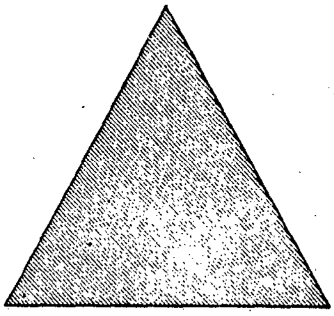


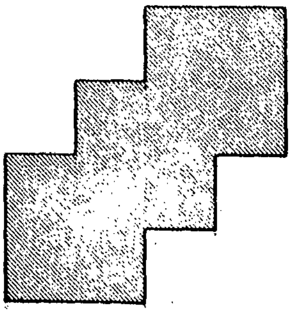
Figure 3-6
Some measures of shape (after Haggett and Chorley (1969)). (a) Variables employed in shape measures.



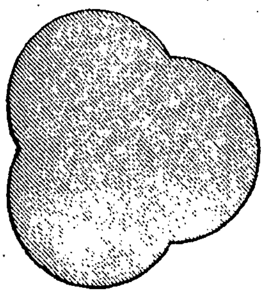
Garvey house, Urbana, Illinois



Adams house, Vinita, Oklahoma



Wilson house, Pensacola Bay, Florida



Trinity Baptist Church, Duncan, Oklahoma

	Form ratio $4a/\pi a^2$	Circularity ratio $4\pi a/p^2$	Elongation ratio $2/a\sqrt{\frac{a}{\pi}}$	Ellipticity ratio $4a/\pi ab$
Garvey house, Urbana, Illinois	1:00	1.00	1.00	1.00
Adams house, Vinita, Oklahoma	0.74	0.60	0.86	0.74
Wilson house, Pensacola Bay, Florida	1.59	0.49	1.26	0.80
Trinity Baptist Church, Duncan, Oklahoma	0.91	0.91	0.95	0.91

Figure 3-6

(b) Plan shapes of some projects by Bruce Goff (no common scale).

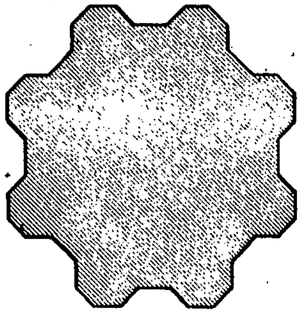
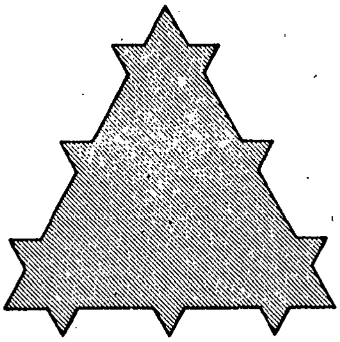
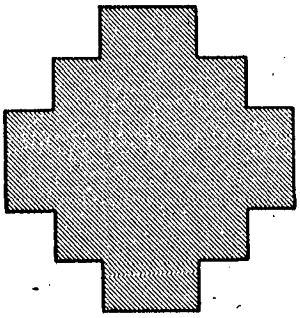
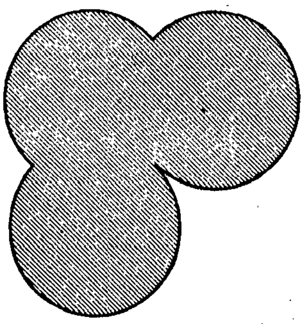
Plan	Form ratio $4a/\pi a^2$	Circularity ratio $4\pi a/p^2$	Elongation ratio $2/a\sqrt{\frac{a}{\pi}}$	Ellipticity ratio $4a/\pi ab$
 Nicol house, Kansas City, Missouri	1.00	0.70	1.00	1.00
 Gutman house, Gulfport, Mississippi	0.67	0.39	0.82	0.67
 Pollock house, Oklahoma City	0.85	0.52	0.92	0.85
 Mc Cullough house, Wichita Falls, Texas	1.62	0.91	1.27	1.21

Figure 3-6
(c) Plan shapes of some projects by Bruce Goff (no common scale).

problem. But in many problems it may be important to consider a variety of types of relations between objects in addition to adjacency/non adjacency. This is particularly the case in site planning, and furniture and equipment layout problems, in which discrete objects are relatively sparsely distributed within a spatial domain. Typical examples of further relations which might be important are that an object should not be visible from another object, or two or more objects should be aligned, that two objects should be symmetrical about some axis, and so on.

Where a variety of different types of relations between objects must be dealt with, a constraint graph is an appropriate technique to employ (Figure 3.7).

3.3.3 Synctactic Formulations :

Another approach to describing complex patterns of relational requirements is to formally define a shape grammar which incorporates an appropriate set of relational rules. A shape may be defined as a finite arrangement of lines on a planar surface.

One shape is a subshape of a second shape if and only if every part of the first shape is part of the second shape. Using these concepts of shape and subshape, a shape grammar may now be defined as follows.

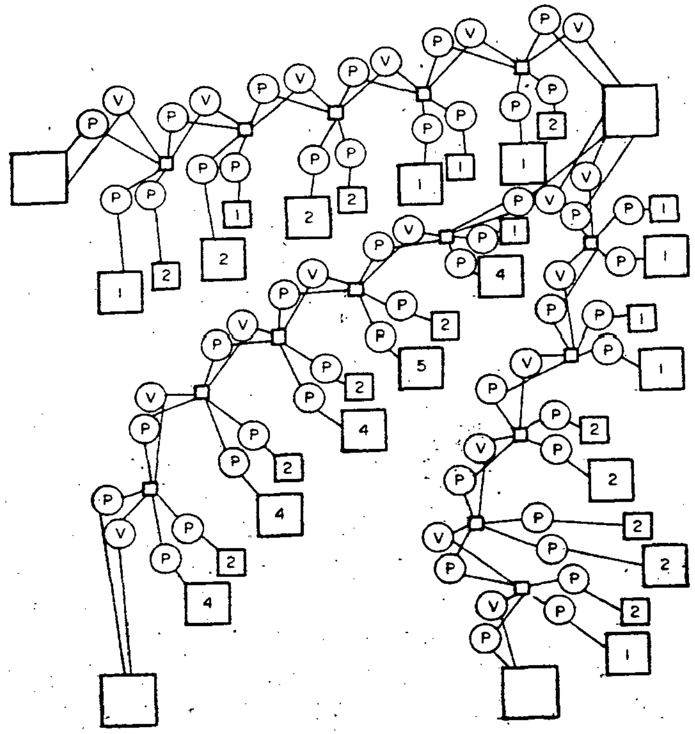


Figure 3-7

A constraint graph for a site layout problem (from Weizapfel and Handel (1975)). Squares represent housing units and other buildings, and the characters in the small circles are constraint labels. A label P indicates that the buildings linked by the constraint should be near to each other, and a label V that there should be visual access.

A shape grammar has four parts:

(a) V_T is a finite set of shapes.

(b) V_M is a finite set of shapes.

(c) R is a finite set of shape rules of the form $u \rightarrow V$, where u and V are shape made up of shapes in V_T or shapes in V_M. The shape a must have a subshape made of shapes in V_M.

(d) I is a shape made up of shapes in V_T or in V_M. The shape I must have a subshape made of shapes in V_M.

The language defined by the shape grammar is the set of shapes generated, by the shape grammar.(Figure 3.8).

3.4 TYPES OF SOLUTION SYNTHESIS PROCEDURES:

3.4.1 Strong and weak solution synthesis procedures

In general solution synthesis procedures may be classified as strong and weak procedures. The more information available, the better the results that can be obtained. Strong results imply strong information demands and weak demands can yield only weak results.

A typical example of a strong problem solving method is the simplex method for solving linear programming problems in economics. This method is easily implemented as a computer program which produces the results very

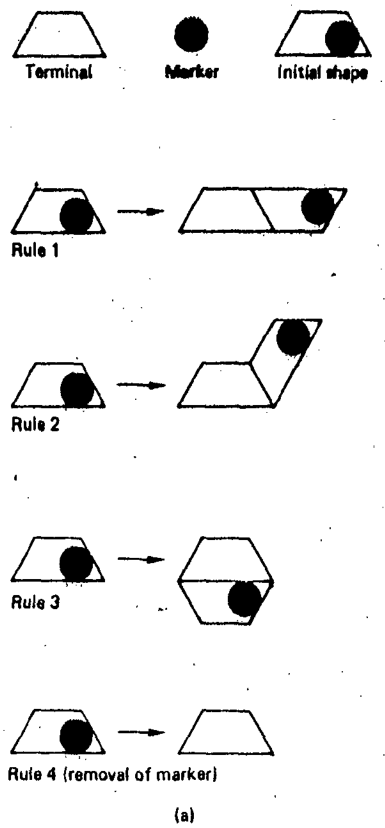
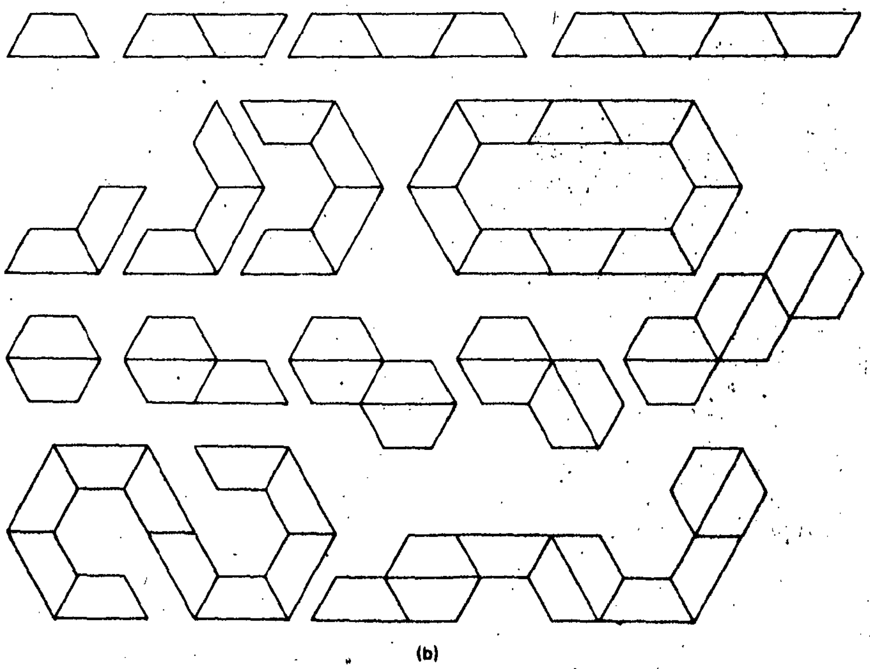


Figure 3-8
A shape grammar for arrangements of half-hexagon tables. (a) A shape grammar SG. (b) Some shapes in the language defined by SG.



quickly and cheaply. But it imposes some extremely stringent requirements. Thus its applicability is restricted to a very narrow and specific domain of problems.

A typical example of a weak method is the random-search method for finding a path through a maze. It is extremely inefficient and will normally result in exploration of numerous dead-ends before a satisfactory solution is found. But it requires absolutely no knowledge of the structure of the maze and it can be applied equally appropriately to any maze with any structure.

3.4.2 Procedures Applicable to Architectural Spatial Synthesis Problems:

Architectural spatial synthesis involves a spectrum of problems types, ranging from those to which very strong methods are applicable, to those for which only weak and general methods may be employed. Strong methods tend to be applicable only to those tasks which are usually thought of as being rather narrow and technical, while many broad central problems respond only to weak methods. For this reason, important automated architectural spatial synthesis techniques often relate rather less to typical engineering and operations research methods than might be expected, and rather more to the weak and general methods characteristic of artificial intelligence systems. The range of synthesis procedures discussed below are roughly in order from the weakest and most general to the strongest and more specific.

3.4.3 Generate-And-Test Procedure:

The weakest and most general of all problem solving methods is the generate-and-test procedure. Perhaps the most famous example of successful use of a generate and test procedure occurred in Thomas Edison's development of the incandescent lamp. It is reported that he systematically tested some 1,600 potential flament materials before finding the one most suitable. The difficulty is that in real design problems of any magnitude or complexity, astronomically long sequences of trials would be needed to be cycled though before a satisfactory solution was discovered.

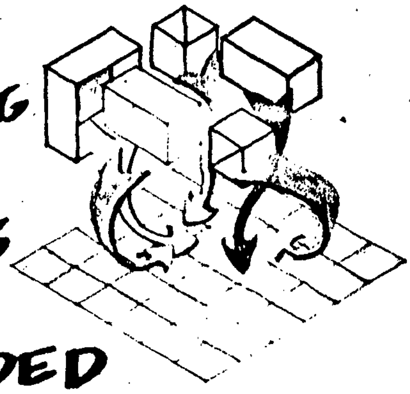
(a) Exhaustive generate and test:

Simple "brute force" enumeration of the elements have been used as a procedure for automated generation of solutions to certain strictly limited classes of floor plan layout problems. Mitchell, Steadman and Ligget (1976) have enumerated all topologically distinct mosaics of rectangles packing together to fill a larger containing rectangle, and suggest this can be a useful basis for an approach to the design of small houses.

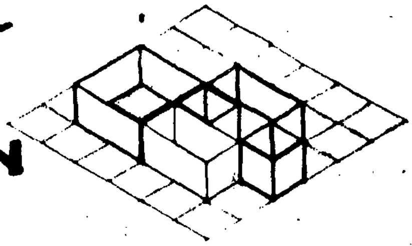
(b) Random generate and test: (Figure 3.9)

Rather more frequently than exhaustive enumeration, random or constrained random sampling of the elements have

RANDOM TECHNIQUE.



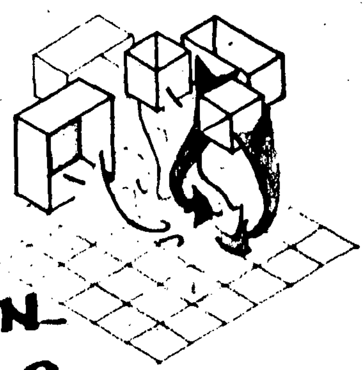
THE PROCESS OF LAYING OUT SPACES IS TO TREAT THEM AS UNIT ELEMENTS AND DROP THEM DOWN RANDOMLY ON A BOUNDED AREA.

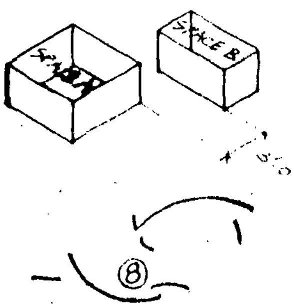


THOUSANDS OF LAYOUTS - GENERATED.

BETTER LAYOUTS CHOSEN BASED ON SCORES.

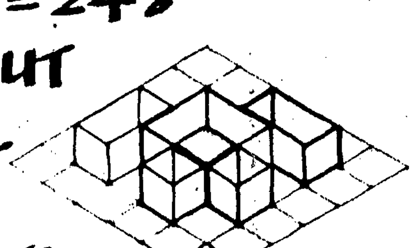
EXAMPLE:





IF DISTANCE BETWEEN TWO ELEMENTS = 3'0" & RELATIONSHIP RATING = 8,

SCORE = 3X8 = 24.



TOTAL SCORE FOR LAYOUT IS SUM OF SCORES OF ALL PAIRS OF ELEMENTS.

COMPUTER OUTPUTS ONLY BETTER LAYOUTS WITH LOWEST SCORES.

Figure 3-9

been employed in automated solution of floor-plan and site-plan layout problems by generate and test. The ALDEP floor plan developed by Seehof and Evans (1967) uses a simple square grid representation of floor plans, and attempts to assign nodules to grid location such that adjacency requirements between spaces are met. A random sampling strategy, in conjunction with some very simple assembly rules, is employed to very rapidly and cheaply generate plans for consideration (Figure 3.10).

A particularly interesting application of random sampling techniques to investigation of site plans has been reported in a thesis by Laios (1974). Laios' program sequentially places housing units on a site by attaching new units to the edges of units already placed in position (Figure 3.11).

Teicholz (1969), Fromboluti (1971) and Velez-Jahn (1972) also describe programs which produce floor and site plan arrangements by use of random number generators in conjunction with some simple assembly rules.

3.4.4 Improvement Procedures :

(a) Blind variation and selective retention :

A straight forward extension of the generate-and-test approach is to utilize an evolutionary or improvement

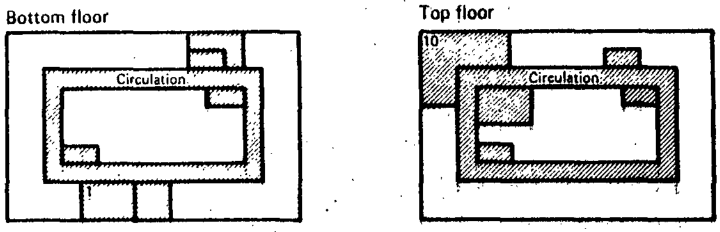
Number	Area (sq. ft.)	Area (10ft. X 10 ft. modules)
1	610	6
2	1537	15
3	2532	25
4	2417	24
5	1721	17
6	3321	33
7	1630	16
8	3239	32
9	2014	20
10	2024	20
11	2210	22

(a)

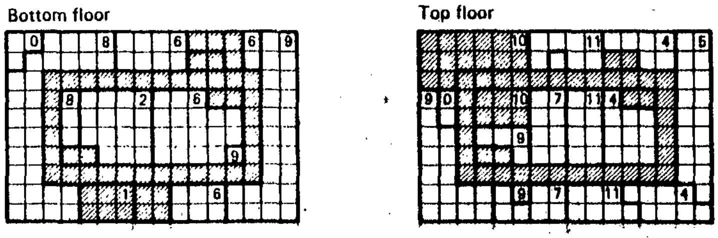
	1	2	3	4	5	6	7	8	9	10	11
1	X	A	D	B	D	D	D	C	C	D	F
2	A	X	D	C	D	D	D	D	D	D	D
3	D	D	X	D	D	C	C	D	D	B	C
4	D	D	D	D	B	D	D	D	D	D	D
5	C	D	D	A	X	D	D	B	D	D	B
6	D	D	B	D	D	X	A	D	D	D	D
7	B	C	D	D	B	D	X	D	C	D	D
8	D	C	D	D	C	D	D	X	D	D	D
9	B	D	D	C	D	D	D	D	X	C	D
10	D	D	D	D	D	D	D	D	D	X	C
11	F	D	D	D	D	D	D	D	D	C	X

Code	Interpretation
A	Absolutely essential to be located near department.
B	Essential to be located near department.
C	Important to be located near department.
D	Optional to be located near department.
F	Undesirable to be located near department.
X	No preference of department to itself.

(b)



(c)



(d)

Figure 3-10.

Typical result produced by the ALDEP floor plan layout program (after Seehof and Evans (1967)). (a) Department areas. (b) Interaction matrix. (c) Building outline and preassigned departments (hatched). (d) Layout generated by a run of ALDEP.

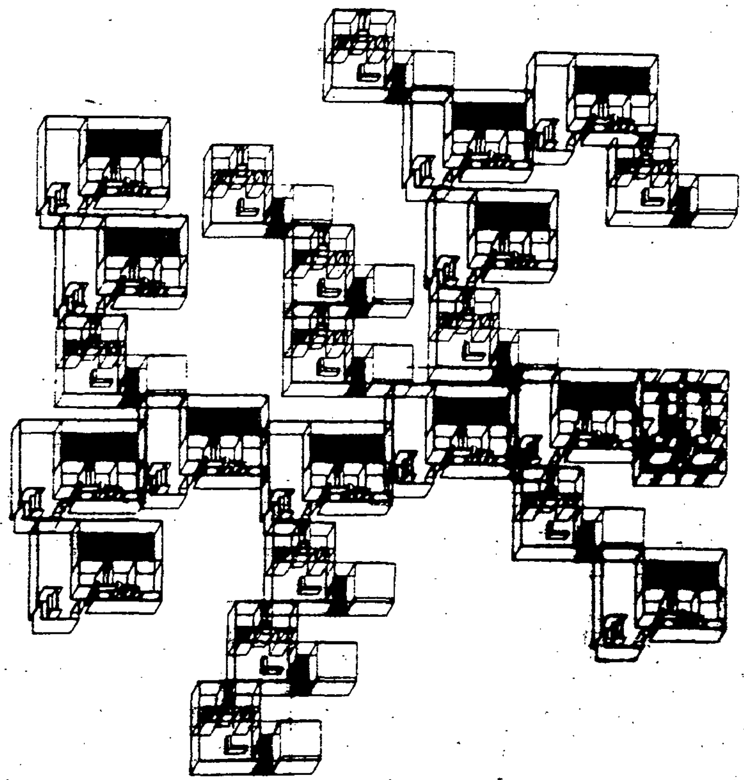


Figure 3-11
Site plan produced by a constrained random generation process (Dimitrious Laios, School of Architecture and Urban Planning, UCLA).

strategy, in which an initially given design is incrementally improved by means of some sequence of small modifications. The concept of an imprcvement strategy is perhaps best introduced by means of an anology. Imagine that you are lost on a hillside on a very dark right and that you can only see a few feet in any direction. Your objective is to reach the highest peak in the neighbourhood. A simple (though not very clever) strategy would be the following :

1. Walk a few feet in any randomly chosen direction from your present position P to a new position P_1.
2. Stop, and assess whether P_1 is higher than P.
3. If so, call P_1, P and repeat the process, or else.
4. Retrace steps to P and choose a new direction and try again.

Assuming that only one peak exists in the neighbourhood, this strategy will necessarily guide you to it. (Figure 3.12)

This is essentially the Darwinian model of evolution, in which mutations are randomly generated, but only those which are in some sense 'better' survive. It has been suggested by numerous authors, notably Popper (1961) and Campbell (1960), that this is a very general and fundamental type of problem-solving process. (Figure 3.13)

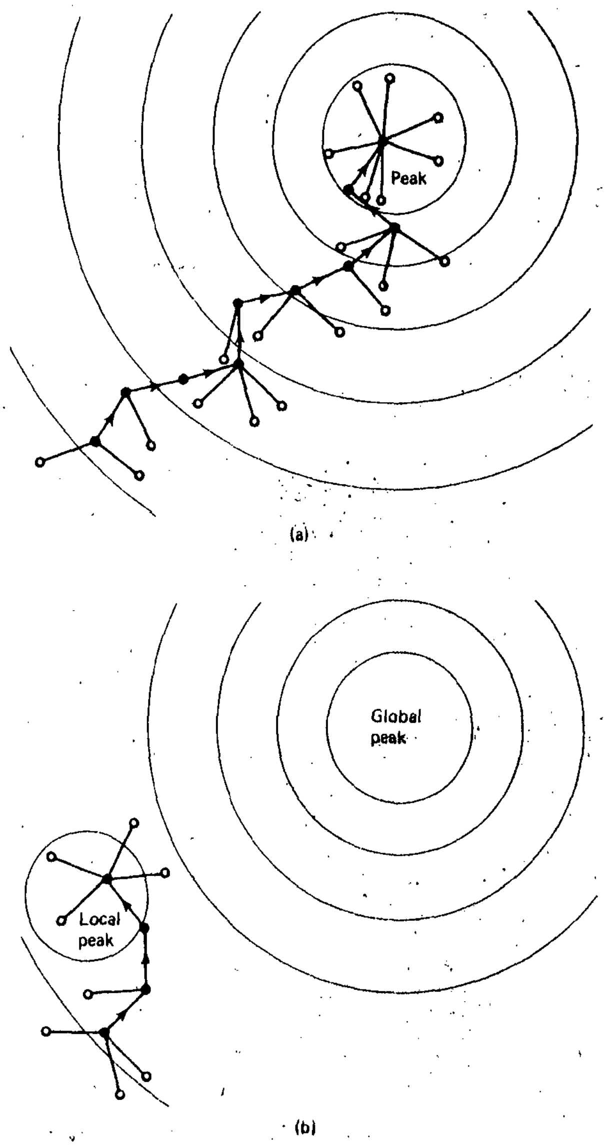
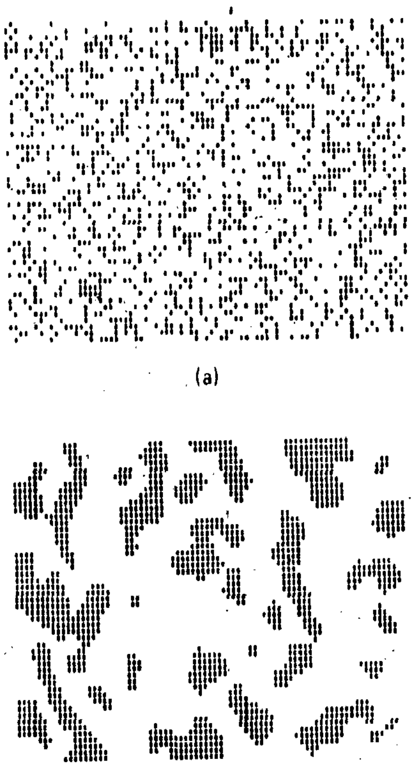


Figure 3-12
A simple improvement procedure for finding the peak of a hill. (a) Typical path generated by a simple improvement procedure for finding a peak. (b) Where more than one peak exists in the neighborhood the procedure may only find a local peak.



(a)

(b)

Figure 3-13

Application of an improvement procedure to a spatial arrangement problem. (a) Initial random field. (b) Result produced by transposition procedure. The system had reached effective stability after approximately 200,000 attempted transpositions for approximately 900 successes.

(b) Greatest improvement procedures :

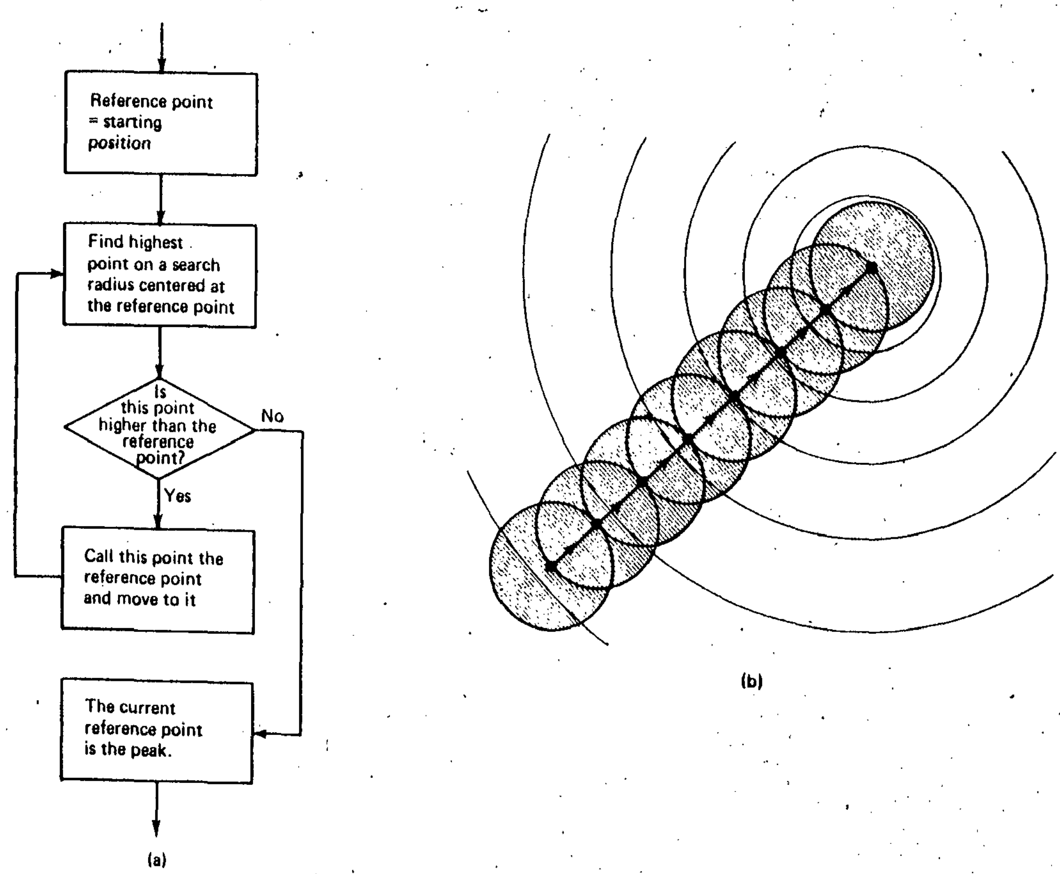
An obvious improvement over blind variation and selective retention, is to head in the direction of 'greatest improvement', rather than in a randomly chosen direction (Figure 3.14)

'Greatest improvement' procedures have been quite extensively used in floor plan layout problems. One of the earliest and best known of these programs is CRAFT by Armour and Buffa(1963). CRAFT utilizes a square grid representation of a floor plan, and employs the quadratic assignment formulation of a floor plan layout problem. It begins with an initial layout input by the user, and computes the quadratic function value for that layout. Subject to certain simple restrictions, which insure that rooms will not be split up, all possible transpositions of pairs of square modules are then considered. The transposition which results in the greatest inprovement in the objective function is executed. This cycle is repeated until no further significant improvements in the value of the objective function can be made (Figure 3.15)

Numerous variations on the basis CRAFT principle have been developed. Cinar (1968, 1975) has implemented a version CRAFT-3D which deals with multistoreyed layouts. Lew and Brown (1970) describe a version which allows the areas of spaces to vary between specified limits. Vollmann, Nugent, and Zartler (1968) have developed a version for

Figure 3-14

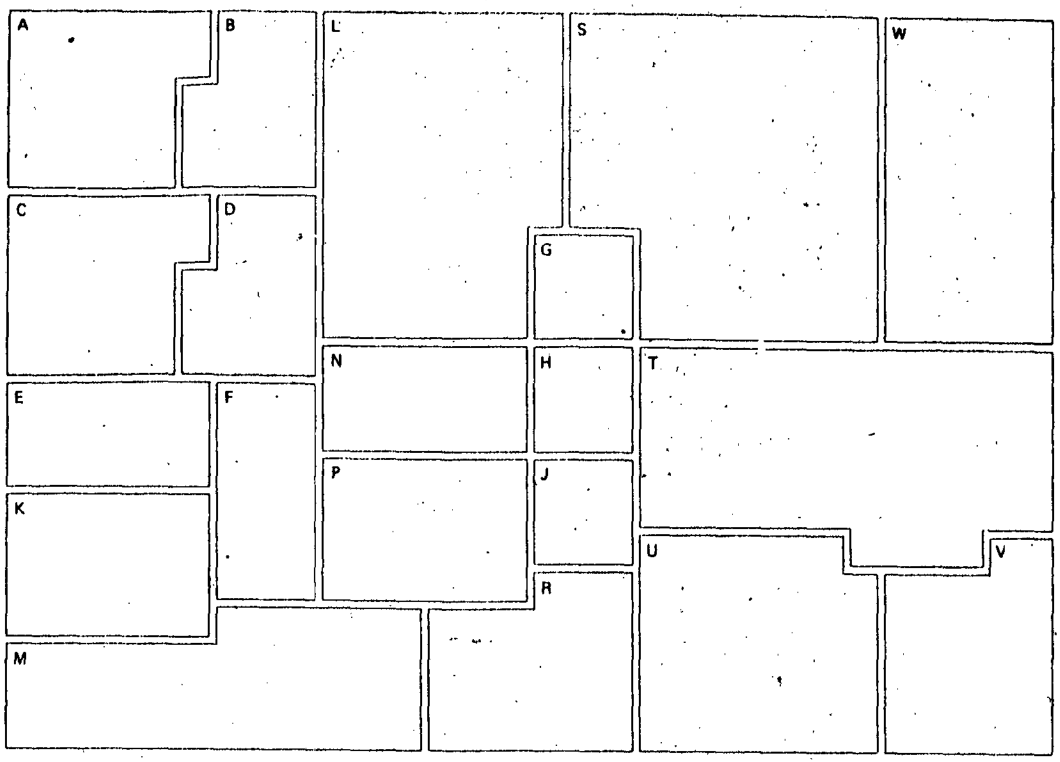
A "greatest improvement" procedure for finding the peak of a hill. (a) Flow diagram of procedure. (b) Typical path traced out in moving to a peak. Dots represent reference points and shaded circles represent search radii.



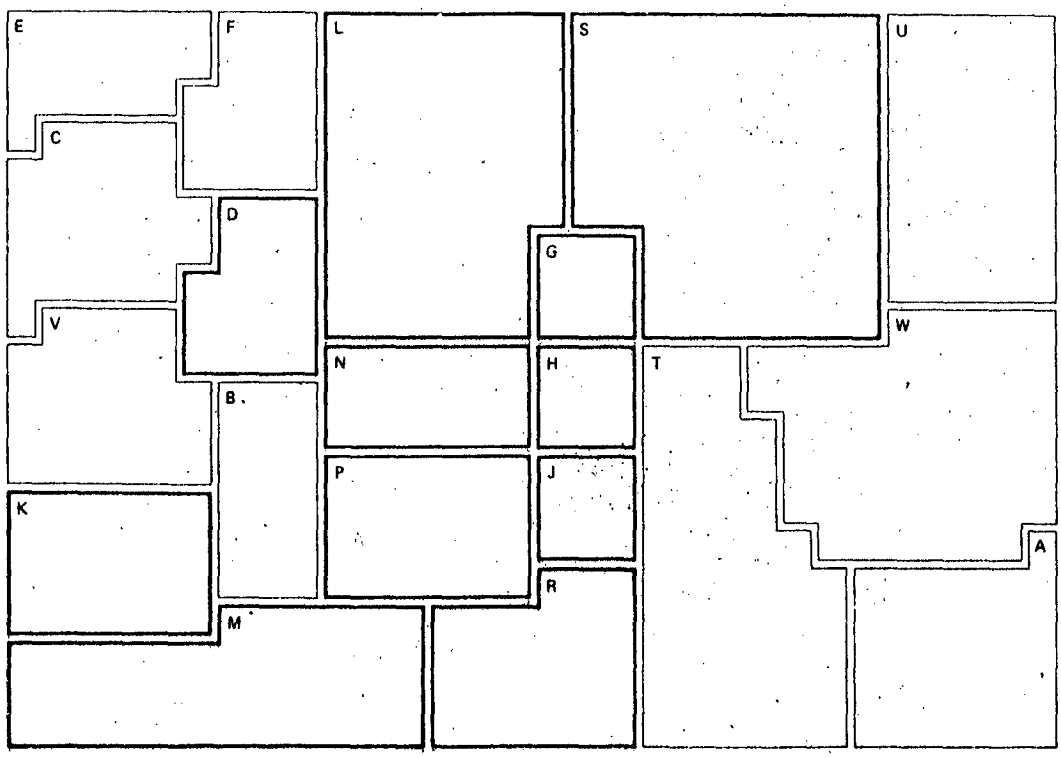
	A	B	C	D	E	F	G	H	J	K	L	M	N	P	R	S	T	U	V	W
A	0.	1.800	1.200	0.	0.	0.	0.	0.	0.	1.040	1.120	0.	0.	1.200	0.	0.	0.	0.	0.	0.
B	1.800	0.	0.960	24.450	0.780	0.	13.950	0.	1.200	1.350	0.	0.	0.	0.	0.	0.	0.	0.	6.900	0.
C	1.200	0.960	0.	0.	0.	2.210	0.	0.	3.150	3.900	0.	0.	0.	13.050	0.	0.	0.	0.	13.650	0.
D	0.	24.450	0.	0.	1.080	5.700	7.500	0.	2.340	0.	0.	1.400	0.	0.	0.	0.	0.	1.500	15.750	0.
E	0.	0.780	0.	1.080	0.	0.	2.250	1.350	0.	1.560	0.	0.	0.	0.	1.350	0.	0.	0.	0.	0.
F	0.	0.	2.210	5.700	0.	0.	6.150	0.	0.	0.	0.	0.450	0.	0.	0.	0.	0.	1.050	0.	0.
G	0.	13.950	0.	7.500	2.250	6.150	0.	24.000	0.	1.870	0.	0.	0.	0.960	0.	0.	0.	1.650	0.	3.750
H	0.	0.	0.	0.	1.350	0.	24.000	0.	0.	0.	0.	0.	0.600	0.	0.	0.	0.	0.	7.500	33.450
J	0.	1.200	3.150	2.340	0.	0.	0.	0.	0.	0.	0.	0.	0.	7.500	0.	0.	7.500	0.	0.	0.
K	1.040	1.350	3.900	0.	1.560	0.	1.870	0.	0.	0.	0.360	12.000	0.	18.600	1.920	0.	0.	0.	5.250	0.
L	1.120	0.	0.	0.	0.	0.	0.	0.	0.	0.360	0.	2.250	0.	3.000	0.960	22.500	–0.	–0.	–0.	0.
M	0.	0.	0.	1.400	0.	0.450	0.	0.	0.	12.000	2.250	0.	0.	0.	1.650	0.	15.000	0.	8.400	0.
N	0.	0.	0.	0.	0.	0.	0.	0.600	0.	0.	0.	0.	0.	8.000	1.040	6.000	0.	0.	0.	0.
P	1.200	0.	13.050	0.	0.	0.	0.960	0.	7.500	18.600	3.000	0.	8.000	0.	9.750	0.	0.	0.900	0.	0.
R	0.	0.	0.	0.	1.350	0.	0.	0.	0.	1.920	0.960	1.650	1.040	9.750	0.	0.	5.250	0.	0.	0.
S	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	–0.	0.	6.000	0.	0.	0.	12.000	0.	0.	0.
T	0.	0.	0.	0.	0.	0.	0.	0.	7.500	0.	–0.	15.000	0.	0.	5.250	12.000	0.	0.	7.500	0.
U	0.	0.	0.	1.500	0.	1.050	1.650	0.	0.	0.	–0.	0.	0.	0.900	0.	0.	0.	0.	4.650	0.
V	0.	6.900	13.650	15.750	0.	0.	0.	7.500	0.	5.250	–0.	8.400	0.	0.	0.	0.	7.500	4.650	0.	0.
W	0.	0.	0.	0.	0.	0.	3.750	33.450	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

(b)

Figure 3-15



(c)

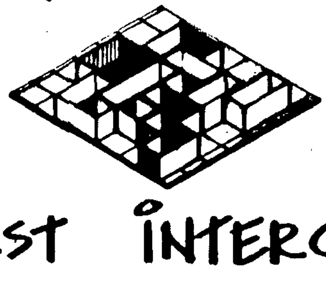


(d)

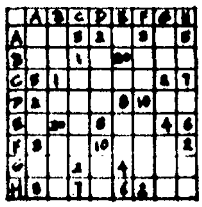
Figure 3-15

INTERCHANGE TECHNIQUE

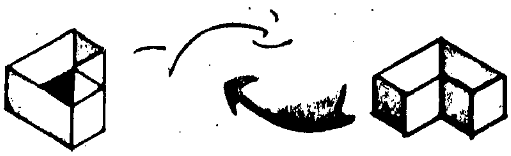
EXISTING LAYOUT



CORRESPONDING COST MATRIX.

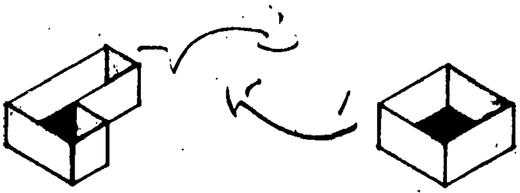


FIRST INTERCHANGE:



NEW LAYOUT SCORE COMPARED WITH PRE-SPECIFIED ONE. BETTER LAYOUT RETAINED.

SECOND INTERCHANGE:



THE ABOVE PROCESS CONTINUED UNTIL

A) THE TIME LIMIT FOR THE PROGRAM RUN IS REACHED.

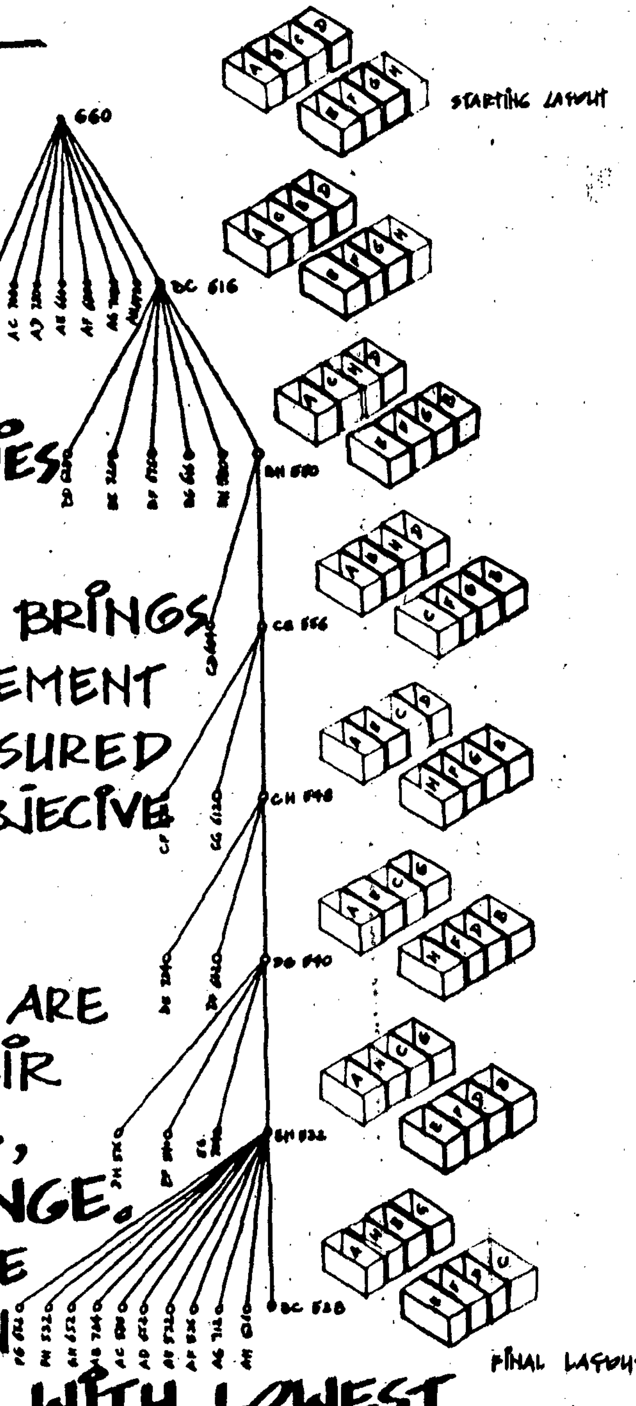
B) A SPECIFIED NUMBER OF ARRANGEMENTS TRIED.

C) ALL ARRANGEMENTS HAVE BEEN CHECKED.

Figure 3-16

IMPROVEMENT:

INITIAL LAYOUT — GENERATED RANDOMLY OR PREDETERMINED BY THE USER.



PAIRS OF ACTIVITIES OF SWAPPED.

ANY SWAP WHICH BRINGS ABOUT AN IMPROVEMENT IN THE LAYOUT, MEASURED IN TERMS OF THE OBJECTIVE FUNCTION — **RETAINED.**
IF NOT THE PAIRS ARE RETURNED TO THEIR FORMER POSITIONS, **NEXT INTERCHANGE.**
ASSUME COST MAY BE OBJECTIVE FUNCTION.
THE FINAL SWAP WITH LOWEST SCORE — THE 'BEST' SOLUTION.

Figure 3-17

COMPUTER AIDED ARCHITECTURAL DESIGN
S. SUKUMAR — M.ARCH-1983-84
UNIVERSITY OF ROORKEE.

assigning people to offices in an existing building layout. (Figure 3.18) Willoughby (1975) has produced a discussion of further applications to practical problems. Quite a different 'best improvement' strategy is employed by the IMAGE systems (Weinzapfel and Handel 1975). IMAGE represents spaces as rectangular parallelepipeds, each given dimensions and a location and rotation within a coordinate system. As in CRAFT, an initial configuration is input by the user. A constraint graph defines solution criteria in terms of such properties, such as proximity, alignment, visual access, circulation, etc. between parallelepipeds. Dimensional constraints are also employed. The synthesis procedure makes incremental changes to each parallelepiped in succession, temporarily holding all others constant. Properties which may be altered are dimensions, location, and rotation. (Figure 3.19)

3.4.5 Improvement Procedures Compared With Generate-And-Test:

Generate-and-test
- applicable to virtually any type of problem.
- guaranteed to find a solution - if one exists.
- general, but weak method.

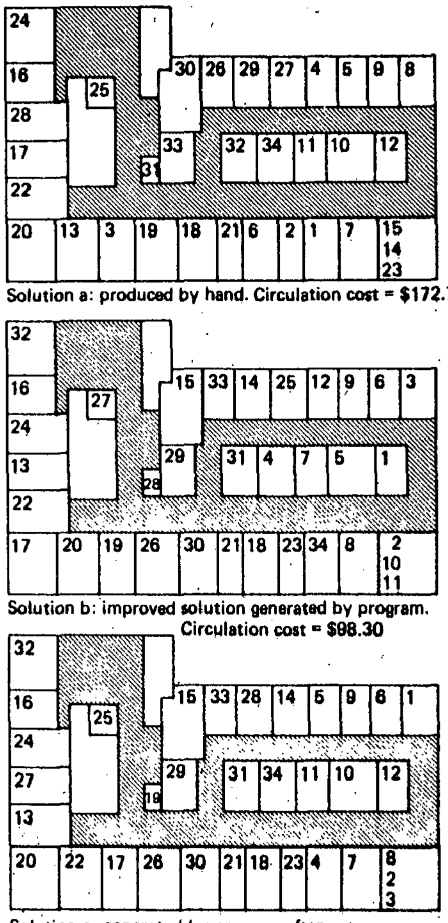
Improvement
- can converage on a good solution quite quickly, but the domain of problem types is limited.
- stronger, but less general
- limited to situations where we can devise some satisfactory means of

Figure 3-18

Some typical arrangements generated by Vollmann, Nugent and Zartler's (1968) pairwise switching program.

An oil company building

1 Geologist A
2 Geologist B
3 Geologist C
4 Landman
5 Geologist D
6 Geologist E
7 Land secretary
8 Geologist F
9 Geology secretary
10 Files A
11 Files B
12 Files C
13 Regional manager secretary
14 Production clerk A
15 Production clerk B
16 Legal secretary
17 Assistant division manager
18 Engineer A
19 Engineer B
20 Division manager
21 Engineer C
22 Engineer D
23 Production clerk C
24 Attorney
25 Receptionist
26 Production supervisor
27 Production clerk D
28 Accounting clerk
29 Office manager
30 Production secretary
31 Xerox
32 Storage area
33 Files D
34 Files E



Solution a: produced by hand. Circulation cost = $172.78

Solution b: improved solution generated by program. Circulation cost = $98.30

Solution c: generated by program after 10, 11, 12, 13, 16, 20, 24, 26, 31, 32, 33, 34 fixed in position in response to various functional constraints. Cost = $103.82

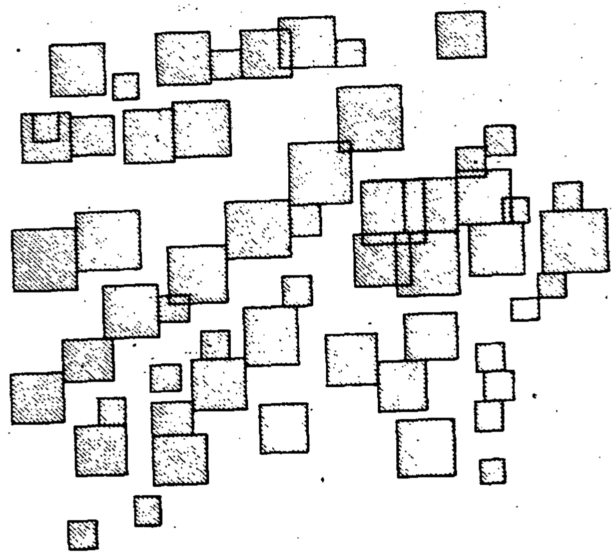


Figure 3-19
A typical result produced by the IMAGE system: a site layout generated in response to the constraint graph shown in figure 13.7 (after Weinzapfel and Handel (1975)).

comparitively evaluating proposals to determine which is 'closer' to the desired result.

3.4.6. Heuristic Procedures :

Principle of heuristic search:

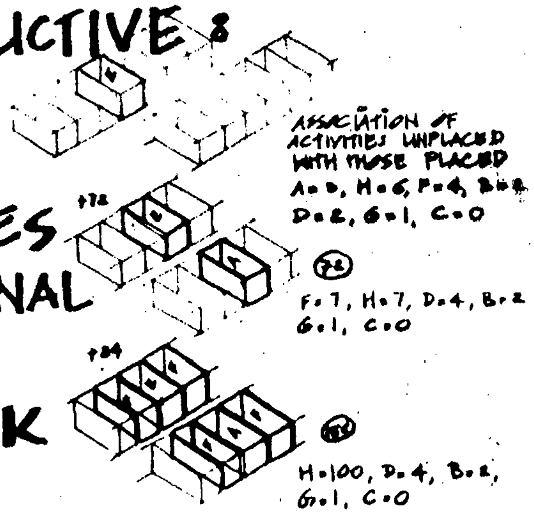
Both the generate-and-test and improvement procedures that have so far been discussed required the data structure to be fully specified before any evaluation of a potential solution takes place. On the contrary, the heuristic search procedure is characterised by solution generation in a sequence of stages, with evaluations based on the partially specified state of the data structure being made at each step. (Figure 3.20).

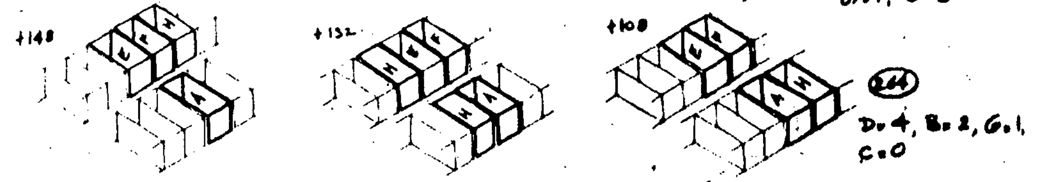
This type of serial decisions making process is conveniently represented by a state-action tree in which the root vertex represents the initial state of the data structure, branches representing alternative operations which may be performed, internal vertices representing partially specified states (incomplete proposed solutions) and terminal vertices representing full specified states.

A heuristic search procedure works by the utilizing a prior knowledge about the structure of the problem, combined with knowledge gained from evaluations performed upon partially - specified states of the data structure, in order to select an appropriate sequence of operations to execute (i.e., an appropriate path through the tree).

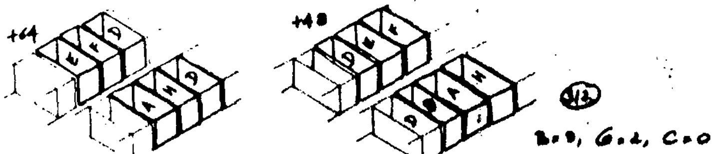
HEURISTIC SEARCH:

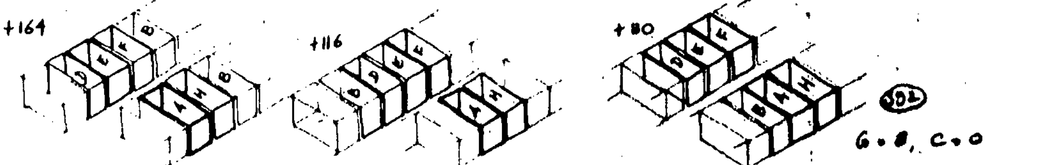
ADDITIVE OR CONSTRUCTIVE:
USE SOME CRITERION
TO DETERMINE
ORDER OF ACTIVITIES
PLACED IN 2-DIMENSIONAL
OR 3-DIMENSIONAL
SPATIAL FRAMEWORK



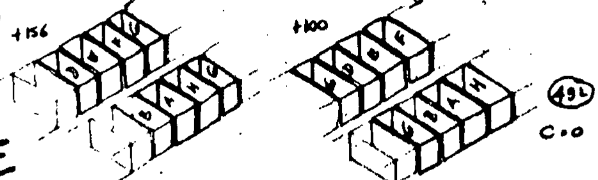


USUALLY, MOST
ACTIVE SPACE





PLACED NEXT TO
THE FIRST,
BASED ON THE
HIGHEST ASSOCIATION
VALUE WITH INITIAL
ACTIVITY. OBJECTIVE
CRITERION OF PLACING THE SELECTED
ACTIVITY MAY BE 'COST'.



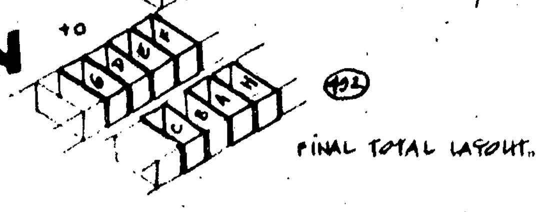


Figure 3-20

COMPUTER AIDED
ARCHITECTURAL
DESIGN

S. SUKUMAR — M.ARCH 1983-84
UNIVERSITY OF ROORKEE

The details of different heuristic search procedures, vary, depending on the types of knowledge they utilize. In the extreme case, no evaluations are performed at internal vertices and no special knowledge at all is utilized to select operations ; the tree is simply explored in a systematic or random fashion, (Equivalent to generate-and-test). In the most powerful, sophisticated procedures are employed to perform evaluations at internal vertices and thus guide the search.

Breadth-first and depth first search methods :

Exploration of a tree can either be conducted in a breadth-first or depth-first fashion, or by some combination of the two. In pure breadth-first search, all the vertices at one level in the tree are generated before any vertices at the next level are considered (Figure 3.21)

In pure depth-first search, a path through the tree is first explored to the end, then further alternatives are generated by a systematic backtracking procedure.

Heuristic information and branch selection :

In principle, both breadth first and depth first search will exhaustively enumerate potential solution. In practice, the enormous extent of the tree usually makes exhaustive search infeasible. It is necessary to make use of some strategy for rapidly eliminating large portions of the tree from consideration.

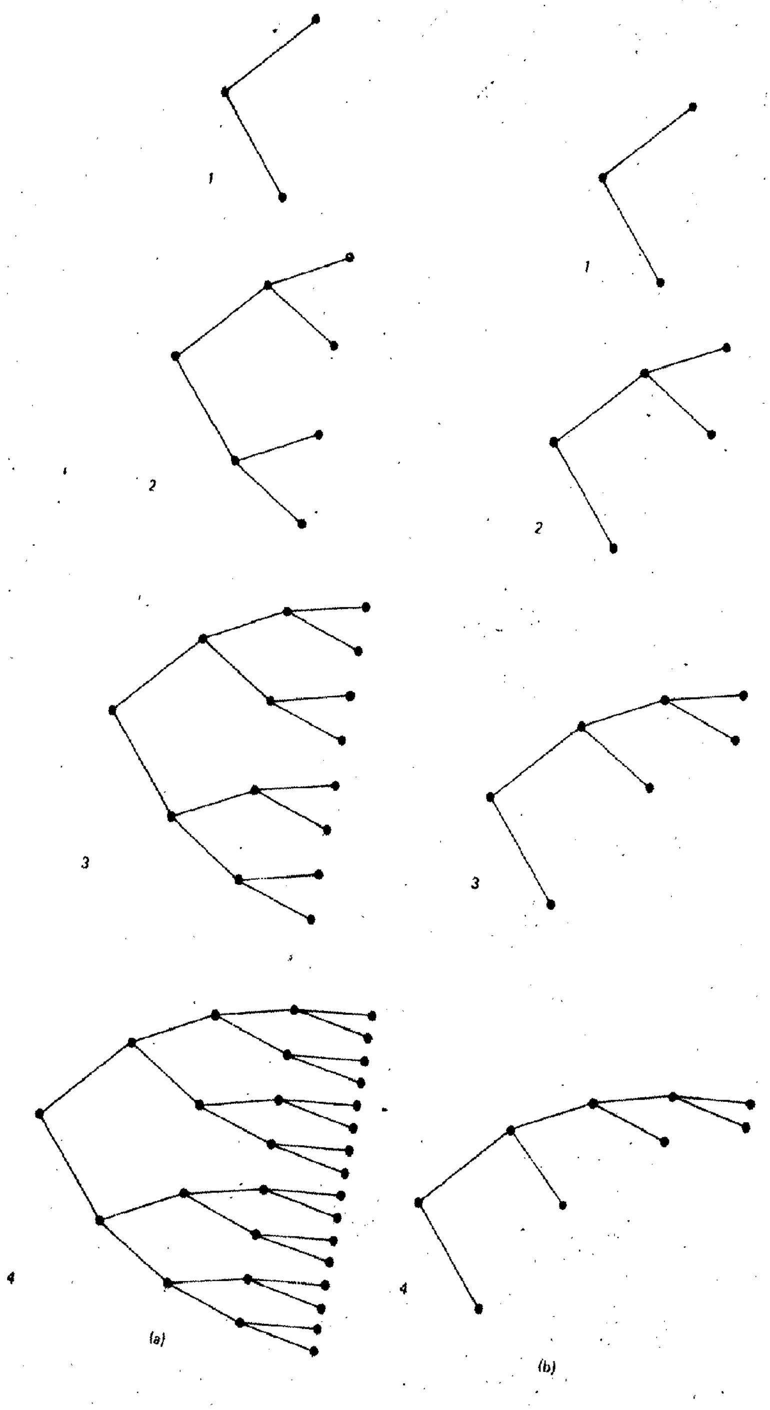
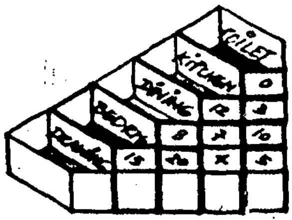


Figure 3-21
Breadth-first and depth-first search procedures (operating in this example on a binary tree). (a) Breadth-first. (b) Depth-first.

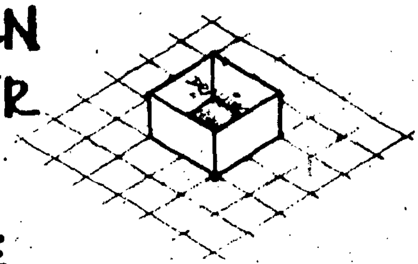
NEIGHBORING TECHNIQUE.

PRE-SPECIFIED RELATIONSHIP MATRIX:

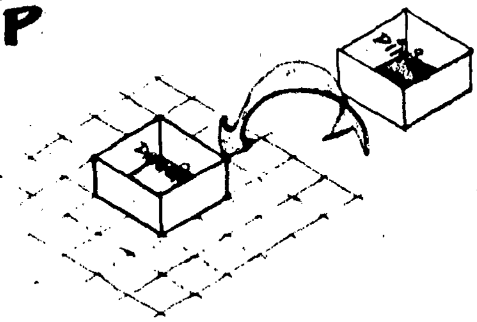
THE SPACE WITH THE HIGHEST TOTAL RELATIONSHIP RATE SELECTED.



SELECTED SPACE THEN PLACED IN THE CENTER OF ALLOWED AREA.

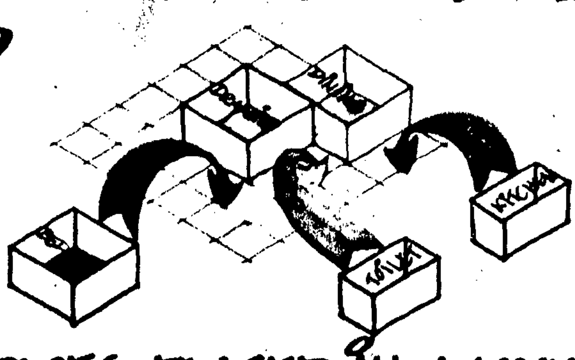


THE SPACE WITH THE HIGHEST RELATIONSHIP RATING TO THE SPACE IN CENTER SELECTED.



THIS SPACE PLACED NEXT TO PREVIOUS ONE.

THE SPACE WITH HIGHEST RATING TO EITHER SELECTED & PLACED IN PROPORTION TO CORRESPONDING RATINGS.



THE SAME PROCEDURE FOLLOWED UNTIL ALL SPACES PLACED IN LAYOUT.

Figure : 3-22

COMPUTER AIDED ARCHITECTURAL DESIGN

S. SUKUMAR — M. ARCH · 1983-84
UNIVERSITY OF ROORKEE.

MULTI-CONSTRAINED TECHNIQUE.

INITIAL LAYOUT

CONSIDERATION OF SEVERAL RELATIONSHIP PATTERNS.

<ADJACENCY>

PRESENTS A SOLUTION.
GENERATES ARRANGEMENTS BY MODIFYING SPACES.

<SATISFIED>

REDUCE THE VIOLATATION OF RELATIONSHIPS CONSTRAINING SPACES.

<DISTANCE>

<SATISFIED>

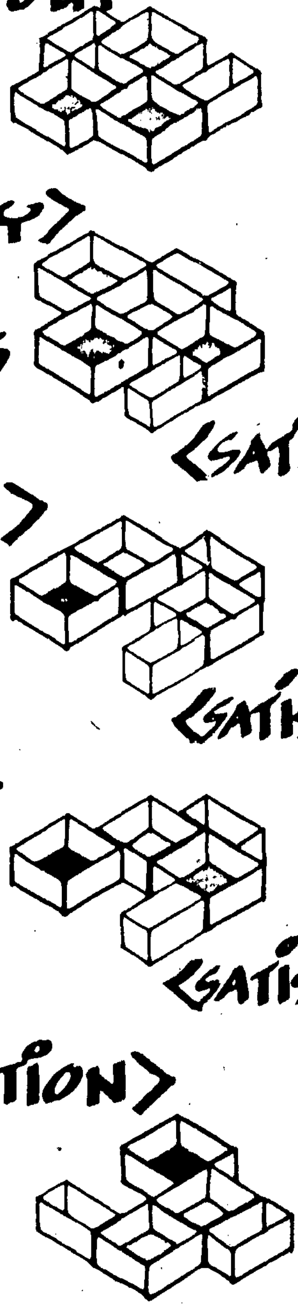
<ACCESS>

RECYCLES THE SPACES

A CONFIGURATION REACHED : SATISFIES ALL SPECIFIED RELATIONSHIPS.

<SATISFIED>

<ORIENTATION>



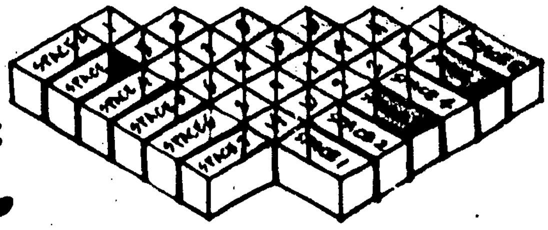
FINAL LAYOUT. <SATISFIED>

Figure 3-23

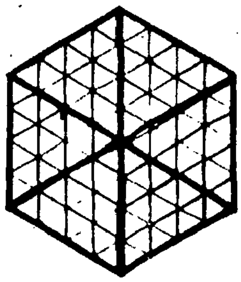
COMPUTER AIDED ARCHITECTURAL DESIGN
S. SUKUMAR – M. ARCH 1983-84
UNIVERSITY OF ROORKEE

VECTOR TECHNIQUE:

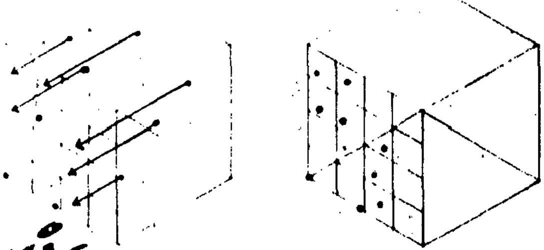
INTERRELATIONSHIP BETWEEN SPACES:
A MATRIX OF NUMBERS: RELATIVE DISTANCES.



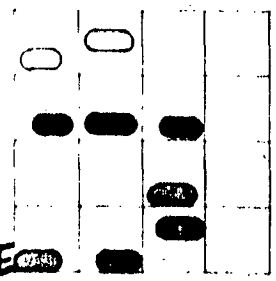
FUNCTIONAL ELEMENTS REPRESENTED AS POINTS.



THE ELEMENT WITH LOWEST TOTAL DISTANCE RELATIVE TO OTHERS, PLACED IN THE ORIGIN OF 3D COORDINATE SYSTEM.
THE ELEMENT WITH SHORTEST DISTANCE TO FIRST-PLACED ON X-AXIS.



ELEMENT WITH LOWEST DIST. FROM EITHER PLACED ON X-Y PLANE.
NEXT ELEMENT: IN X-Y PLANE OR 3D SPACE.



ALL ELEMENTS LOCATED AND PROJ. ON X-Y PLANE
AREAS GIVEN & BUBBLE DIAG. GENERATED.

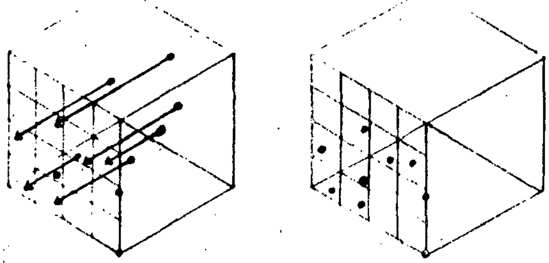


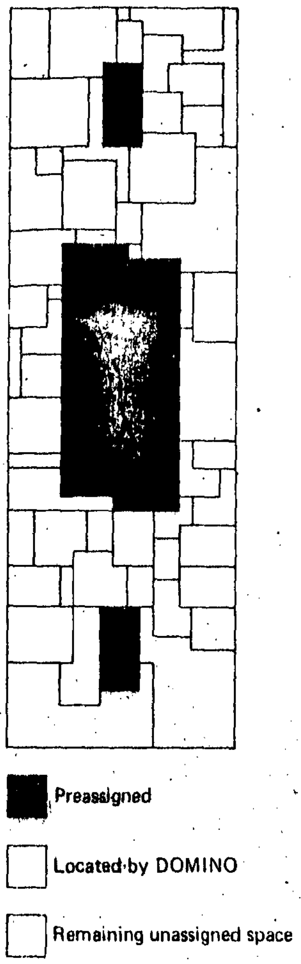
Figure 3-24

Numerous floor plan layout programs which employ plausible selection rules or the type to locate modules within a square grid have been developed. Some of the best known are programs bv Whitehead and Eldars (1964). Lee and Moore (1967). Spillers (1970). Willoughby (1970). and Mitchell and Dillon (1972) (Figure 3.25)

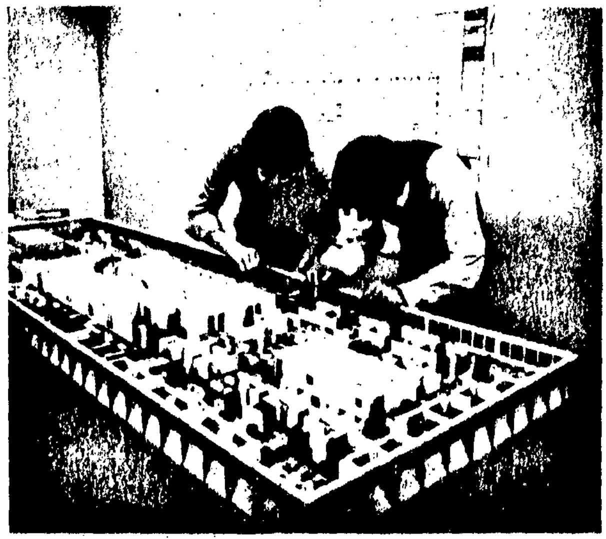
For solution of floor plan layout problems formulated in quadratic assignment terms, various heuristic strategies using selection rules based upon mathematics analysis of properties of the state - action tree have been employed. Stategies have been developed for computing mean expected values of the objective function for available branches at each decision point, and then selecting the branch with the lowest value. Figure 3.26 uses a trivial problem of arranging 4 shaded square in a row to illustrate one such strategy graphically.

Backtracking :

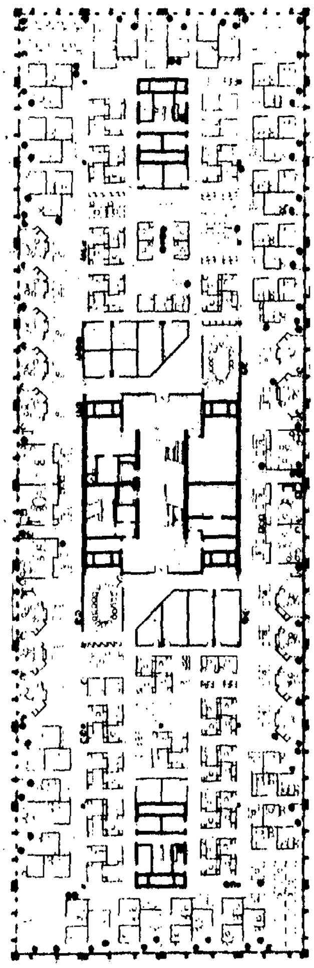
None of the programs discussed so far engage in any backtracking through the state-action tree, a decision is never reconsidered once it is made. It is often desirable, in heuristic search programs, to incorporate a capability to recognize that a particular sequence of decisions has led to a 'dead-end' from which no satisfactory solution can result, and to backtrack to a previous vertex and restart the search along an alternative branch.



(a)



(b)



(c)

Figure 3-25

Application of DOMINO, an empirical heuristic floor plan layout program, to planning an office floor (Morganelli-Heumann and Associates, Los Angeles). (a) Block layout generated by DOMINO. (b) Block layout used as a basis for detailed layout. (c) Final plan.

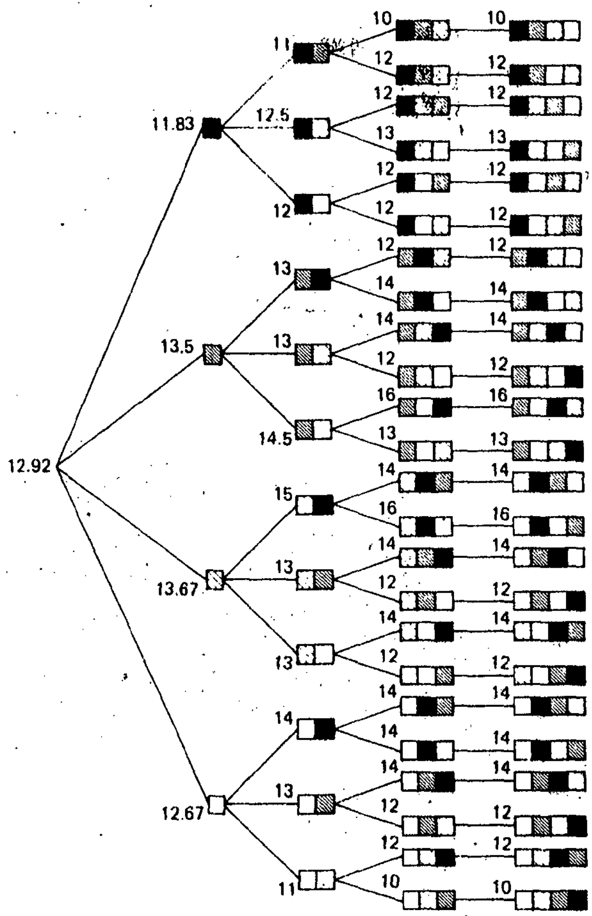


Figure 3-26
State-action tree for a quadratic assignment problem, with the mean expected value for the objective function at each branch shown.

This procedure of backtracking is characteristic of human problem - solving processes. (Figure 3.27) The basic pattern is as follows:

1. A sequence of decisions is made until it becomes obvious that a dead end has been reached ;
2. The configuration is partially destroyed by removal of pieces ;
3. Another sequence of decisions is made until either a solution is found or it again become obvious that a dead end has been reached.

This pattern is repeated until a satisfactory solution results. (Figure 3.28).

Planning :

One characteristic shared by all the heuristic search programs discussed so far is that they make one decision at a time, without 'planning ahead', by breaking down the total problem into sub problems.

One of the first heuristic search spatial synthesis programs to incorporate partitioning of the problem into sub-problem was developed by Beaumont (1976). A sophisticated use of planning is contained in a more redent program called DPS (Design Problem Solver) developed by Pfefferkorn (1975). DPS arranges furniture or equipment in a room in accordance with a constraint graph specifying proximity,

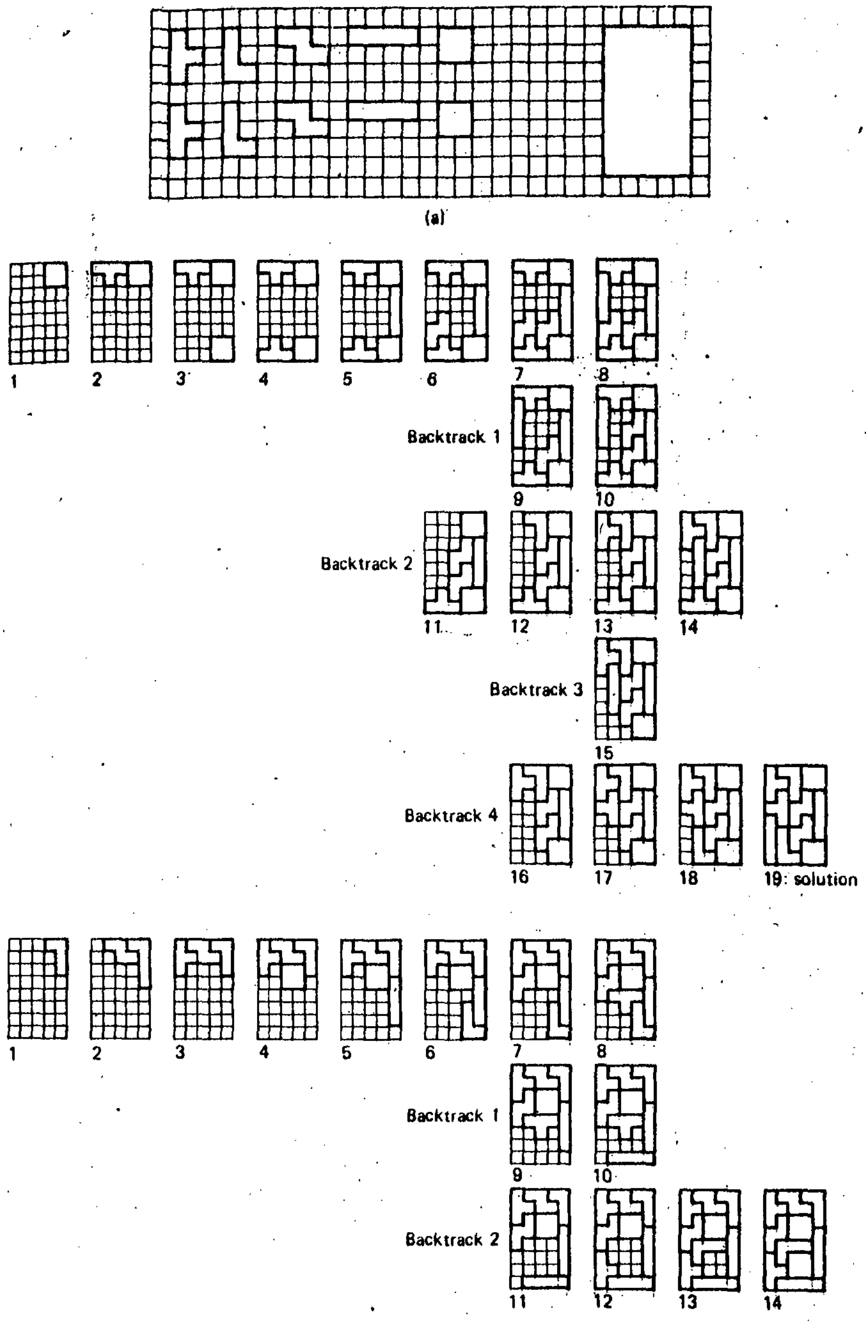
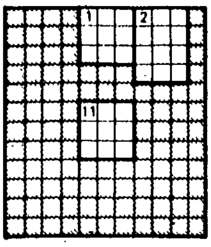
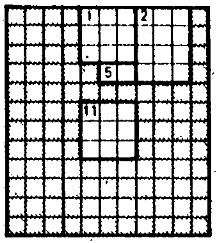


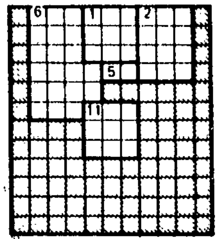
Figure 3-27
Backtracking in human problem solving. (a) Problem: pack polyominoes on the left into the rectangle on the right. (b) Two solution protocols.



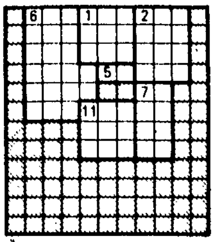
(a) 2 successfully added to 1.



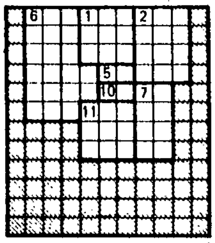
(b) 5 successfully added to 1.



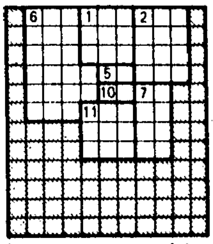
(c) 6 successfully added to 1.



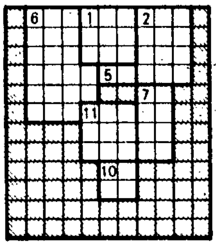
(d) 7 successfully added to 11.



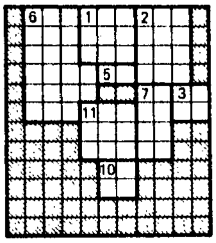
(e) Unsuccessful attempt to add 10 to 11: backtrack



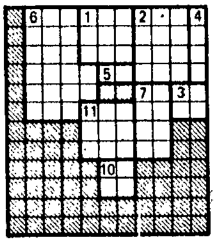
(f) Second unsuccessful attempt to add 10 to 11: backtrack



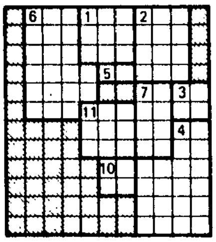
(g) Successful attempt to add 10 to 11



(h) Successful attempt to add 3 to 2



(i) Unsuccessful attempt to add 4 to 3: backtrack



(j) Successful attempt to add 4 to 3

Figure 3-28

Backtracking by the DOMINO floor plan layout program (Mitchell and Dillon 1972). The program attempts to add new spaces to the perimeters of located spaces to which there are high interactions. Backtracking is necessary if there is insufficient room for the added space at the chosen perimeter location. A new perimeter location is then selected and tried.

separation and other requirements. At each step of the solution generation procedure, an attempt is made to insert a new element into the design such that no constraints are violated. The design is complete and satisfactory when all elements have been successfully inserted in this way.

3.4.7 Capabilities and Applications of Heuristic Search Programs :

Heuristic search procedures are quite strong techniques. They can often deliver very good solutions for most problems to which they are applied, and can do so reasonably economically. But in general, the stronger heuristic search procedures are fairly limited in the types of problems to which they are applicable. This is because they often gain their strength by exploiting very specific knowledge about a particular type of problem.

Most of the easly applications of heuristic search programs in architecture were to floor plan layout problems involving arrangements of square modules. At Carnegie-Mellon University, two very successful heuristic search programs for laying out furniture or equipment within spaces have been developed. These are GSP (Eastman 1971, 1972, 1973) and DPS (Pfefferkorn 1975). The Harness Hospital Integrated Computer Aided Design System incorporates a heuristic search automated layout program called HAPA which locates standard departments around a circulation route defined by the user to produce complete Multi-storey hospital layouts.

3.4.8 Nonlinear and Linear Programming :

Even stronger but correspondingly less general techniques than heuristic search are nonlinear programming and linear programming methods. These methods were initially developed for solutions of economic, operations research, and engineering problems, but they have found some interesting applications to floor plan synthesis problems.

(a) Non Linear Programming :

Non linear programming has been used in conjunction with dimensionless representations of floor plans to generate optimum dimensioned layouts with respect to some cost criterion and subject to certain functional constraints (Mitchell 1974, 1975, Sauda 1975, Mitchell, Steadman and Liggett 1976, McGorern 1976) (Figure 3.29).

(b) Linear Programming :

Standard linear programming techniques are applicable and will efficiently generate a solution or report infeasibility where this type of dimensioning problem can be formulated in terms of the optimization of a linear objective function subject to linear constraints. Typical linear objectives are maximization or minimization of

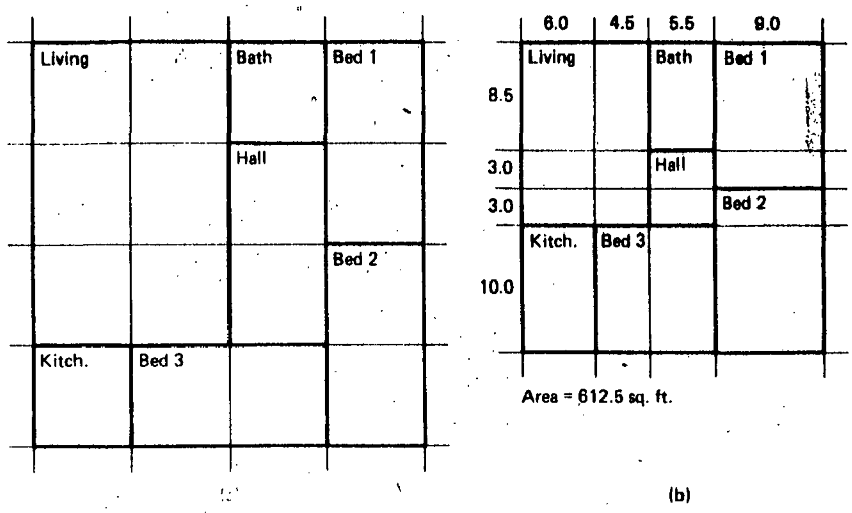


Figure 3-29
An example of application of nonlinear programming to a floor plan layout problem. (a) Dimensionless representation of layout. (b) Optimum dimensions found by nonlinear programming.

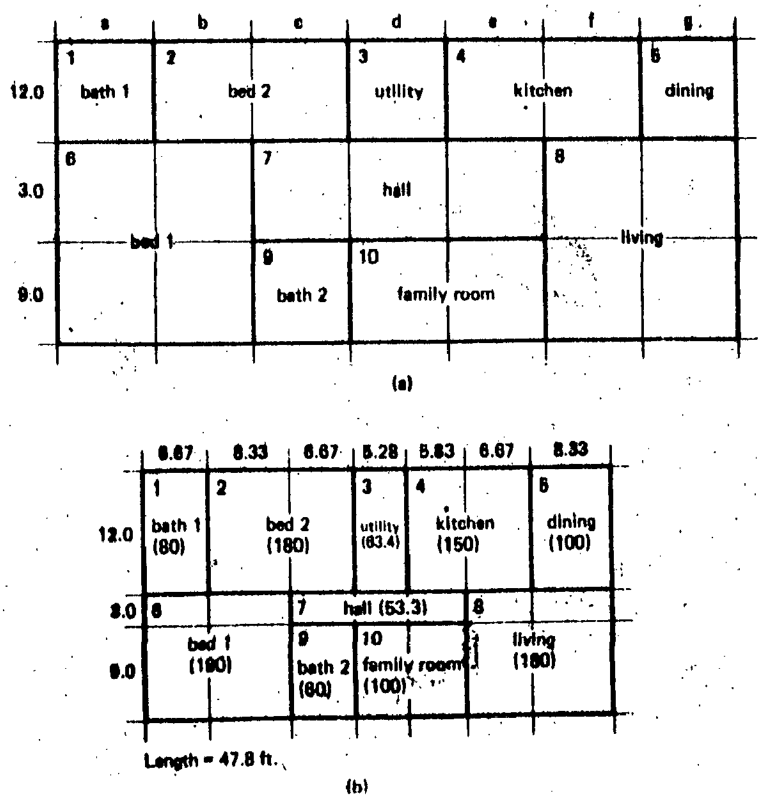


Figure 3-30
An example of application of linear programming to a floor plan layout problem. (a) Dimensionless representation of a "double wide" (24 ft.) mobile home unit, with dimensioning vector along short side fixed. (b) Optimum solution drawn to scale.

overall plan length, width, perimeter or proportion ratio. Typical linear programming constraints are upper and lower bounds, on allowable lengths, widths, perimeters, and proportion ratios of individual rooms and of the overall plan envelope.

An immediate obvious limitation of the linear programming approach is that area constraints are non-linear and thus cannot be incorporated in a linear programming formulation. Furthermore, objectives representing such important properties as construction cost, heat loss, etc. are also functions of the floor area.

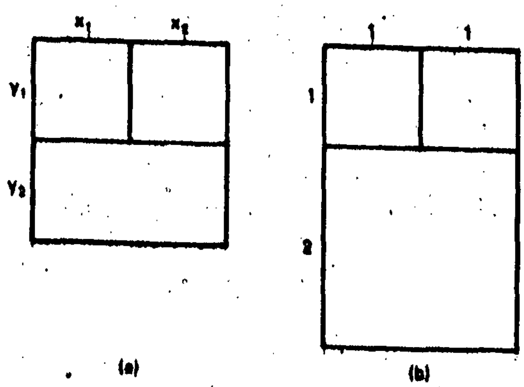
Linear programming procedures are much more powerful than any of the techniques discussed previously. They guarantee to produce the optimum solution, and they do so with great efficiency. The price which must be paid, however, is that their application is restricted to a relatively narrow range of problem types (Figure 3.30).

3.4.9 Analytical Procedures :

Analytical procedures are the most powerful of all. Non trivial architectural spatial synthesis problems which can be solved analytically are rare but contrary to general opinion, they do exist (Figure 3.31 and 3.32).

Algebraic solution of a simple proportioning problem.

Figure 3-31



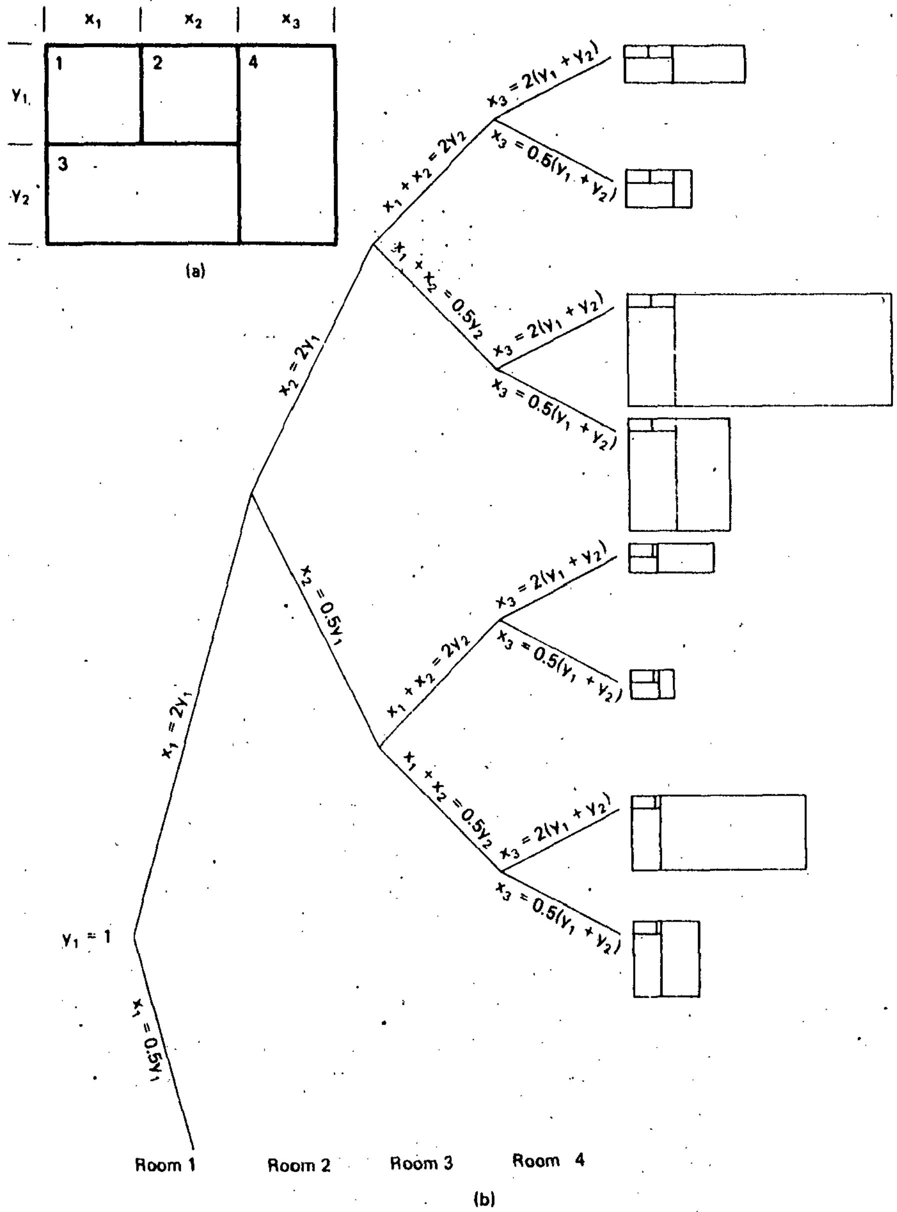


Figure 3-32

Proportioning problem for a plan (a) where all rooms are to be 2:1. (a) Dimensionless plan. (b) Systematic generation of correctly proportioned plans by solution of simultaneous linear equations.

These procedurés described in general above give a very comprehensive idea of the range of potential approaches to computer aided spatial synthesis. Various other authors have made classifications based on their research. Tabor(1970) has distinguished two primary procedures for computer aided spatial allocation : additive and permutational. Nugent et.al, (1968), and March and Steadman (1971) have classified procedurés into constructive and improvement. Dr. Kaiman Lee (1976) has classified procedures into :-

1. Interchange techniques
2. Neighbouring techniques
3. Random techniques
4. Vector techniques
5. Multiconstrained techniques.

Eric Teicholz (1976) has listed them as under :

1. Neighbour-searching
2. Interchange
3. Random
4. Hierarchical
5. Optimization.

It can be noticed that most researchers and authors have classifid procedures, more or less, on similar lines. Current work being done today reflect a basic understanding of these procedures and systems being developed, involve 'hybrid' procedures, where two or more procedures are concatenated into one. The general trend is towards making spatial synthesis more user-oriented, incorporating the salient features of the existing procedures and the elimination of bottlenecks and limitations encountered. A lot of work is being done in the field of two-dimensional and three-dimensional graphics and the generation of solutions through iteractive graphic programming, utilizing the capabilities of man and machine.

4. ANALYSIS OF A COMPUTER AIDED SPACE SYNTHESIS PROGRAM :

The logical extension of the previous chapter would be an in-depth analysis of certain space planning, synthesis programs developed by a few authors to give a thorough understanding of the technique involved, the salient features and the limitations. The program under study can be categorized as a heuristic procedure, quite strong in its execution, but not very general in character. The abstract is as follows :

PROGRAM NAME	:	COMPUTERIZED MULTISTOREYED BUILDING LAYOUT
CODE NAME	:	COMSBUL
KEY WORDS	:	Multistorey build, layout plan, relate chart, circulate cost, heuristic, matrix, architect system, rational neighbour space allocation.
AUTHOR	:	Kaiman Lee
DATE OF STATUS	:	Completed in August 1969, at IOWA STATE UNIVERSITY.
SCOPE	:	COMSBUL, describes a computer program approach to the layout of certain multi-storey buildings. The room with the highest total relationship rating t is selected and placed in the centre of

the matrix layout. The room with the highest relationship rating to the initial space will be placed next to it. The process continues until the specified floor area is filled. Then the program starts with the next floor level and so on.

INPUT	:	The input data consists of the maximum length to width ratio of the plan, the side unit of the module, area of each room, the number of floors, the maximum area of each, and the relationship rating matrix.
OUTPUT	:	The computer prints out on a line printer floor layouts of every floor level along with the arrangement of the input data.
LIMITATION	:	The computer program accepts upto 35 rooms and upto 20 relationship ratings. The upper limit of the number of floors is 99. The maximum size of the layout matrix is 39 x 39 modules.
SOFTWARE	:	FORTRAN IV
HARDWARE	:	IBM 360/50, CDC 6400, PDP 15, COMPUTEK 400 C.R.T., CLEVITE 4800 PRINTER.

AVAILABILITY : The computer program deck and writeup may be obtained by writing to Dr.Kaiman Lee, Environmental Design of Reséarch Centre, 940, Park Square., Building, Boston, Massachusetts 02116.

The program was initially developed in partial fulfillment of the requirements for the degree of Master of Architecture at the IOWA State University of Science and Technology, Iowa.

4.1 THE UNDERLYING CONCEPT OF THE PROGRAM :

4.1.1 Circulation Cost Concept :

There is a large number of interrelationships between the various activities in a building. It seems impossible to reconcile all types of relationships so that the optimum is achieved for each. In most types of buildings, the cost of people walking from place to place tends to be a predominant part of the total cost of providing and operating the buildings.

Whitehead and Eldars (1964) made a study of a hospital operating theatre suite and stated the following :-

Studies show that, on an average, people working within the suite spend 38% of their working time in walking between rooms and, if relative salaries are taken into account, this represents 34% of the total salary cost of staff time.

Dr. Lynn Moseley states the 'circulation cost' concept as follows :-

The relationship between two or more activities may be assumed to depend on two factors, that of the degree of movement between them, and the distance over which this movement takes place.

Circulation cost of the relationship = Total traffic x access distance.

If the quantity of traffic between two activities is recorded over a standard period of time and is regarded as a constant, the variable factor which is minimized in order to produce an optimum cost is distance. Whitehead and Eldars (1965) in their paper, The planning single-storey layouts, describe a computer program to achieve space allocation using this concept. In subsequent papers Whitehead and Agraa (1967 and 1968) suggest improvements and the addition of certain other parameters to the program.

COMSBUL bases its space allocation logic on the concept advocated by Lynn Moseley and developed by Whitehead and Eldars.

4.1.2 Assumptions Made :

1. The more traffic two activities generate, the closer together they should be located, while the distance between them is minimized.

2. The more times an activity participant travels outside a building, the closer to the ground floor this activity should be.

3. The higher the salary of a staff member, the lesser the distance he should walk. Consequently, his room should be located more centrally in relation to the other rooms.

4. The larger the area of a room, the more central this room should be, so that it has a bigger circumference to which other rooms may attach.

In conclusion :- The number of trips between the two activity areas are modified by three factors, i.e., the internal and internal/external traffic, the comparative salary level, and the areas of the rooms. The numerical values after modification are referred to as 'circulation cost'.

4.2 DATA REQUIREMENTS :

4.2.1 Accommodation Schedule :

The procedure for collecting data concerning movement interrelationships can be summarized as follows :-

1. Choose a building of similar type to the one to be designed.

2. Observe the movement of each type of staff member within the building for a representative period.

3. Record the movements in terms of the activities undertaken as well as in terms of the points visited.

4. Examine the movements and reasons for them to determine whether they can be eliminated as unnecessary or the reasons can be removed by changes in organisation or by providing changed or additional facilities.

5. Modify the data taking into account any assumed changes in activities or facilities.

6. Multiply the number of journeys between each pair of rooms counted for each type of staff by the number of staff of that type.

7. Modify the number of journeys for each type of staff by a factor representing the relative cost (salary plus overheads) of that type of staff in relation to the average.

For example :-

	Total Annual cost ($) (Salary and overhead charges	FACTOR
Surgeon	14,400	1.6
Staff	8,850	1.0
Student Nurse	4,500	0.5

If a hundred actual journeys between two rooms are made by the surgeon and by the student nurse, the results will be :

Surgeon	- 100 x 1.6	= 160
Student Nurse	- 100 x 0.5	= 50

Thus introducing a weighting factor based on cost (Figure 4.2).

8. Determine a convenient unit floor area (say 10 ft x 10 f = 100 sq. feet). The number of units of floor area (hereafter referred to as elements) required for each room is found. The number of journeys found previously is now modified by the number of elements for the two rooms under consideration. For example,

Number of journeys	= 100
Room A	Room B
4 elements	2 elements

The modified number of journeys between rooms A and B is

100 x (4 + 2) = 600 (Figure 4.3).

It is possible to derive a general model.

T_i = Number of persons in each type of staff for the two rooms under consideration.

N_i = Total number of journeys made by each persons

$= K_{ij} N_{ij} + N_{ic}$

where K_{ij} = Ratio of cost of travel between input and internal traffic.

N_{ij} = Total number of in/out journeys by each person.

N_{ic} = Total number of internal journeys by each person.

S_i = Factor representing the relative salary level of a type of staff for the two rooms under consideration.

K = A constant used to modify A.

A = Total number of floor units for the two rooms under consideration.

Thus the 'circulation cost' is

$$C.C. = \Sigma [T_i \times N_i \times S_i] \times KA$$

The value of T_i, N_{ij}, N_{ic}, S_i and A are finite and can be obtained. The values of K_{ij} and K can only be determined through experience and statistics.

4.3 THE COMPUTER PROGRAM :

4.3.1 COMSBUL Logic :

The overall logic used in COMSBUL is heuristic. It arrives at a logical plan layout which is optimum based on the criterion of circulation cost. The program allows the inclusion of :

(a) Any factor, not necessarily related to circulation cost, by which it is thought desirable to differentiate between different classes of people.

(b) Any factor by which it is considered necessary to amend the total relationship between spaces. It is therefore a general, rather than a very specialized tool for the arrangement of spaces in relation to one another.

4.3.2 Main Algorithm (Figure 4.1) :

Basically there are two questions which are asked systematically. They are 'which room has the privilege of being placed next into the layout', and 'how is this room entered into the layout'. The main algorithm asks and answers these questions in a heuristic manner.

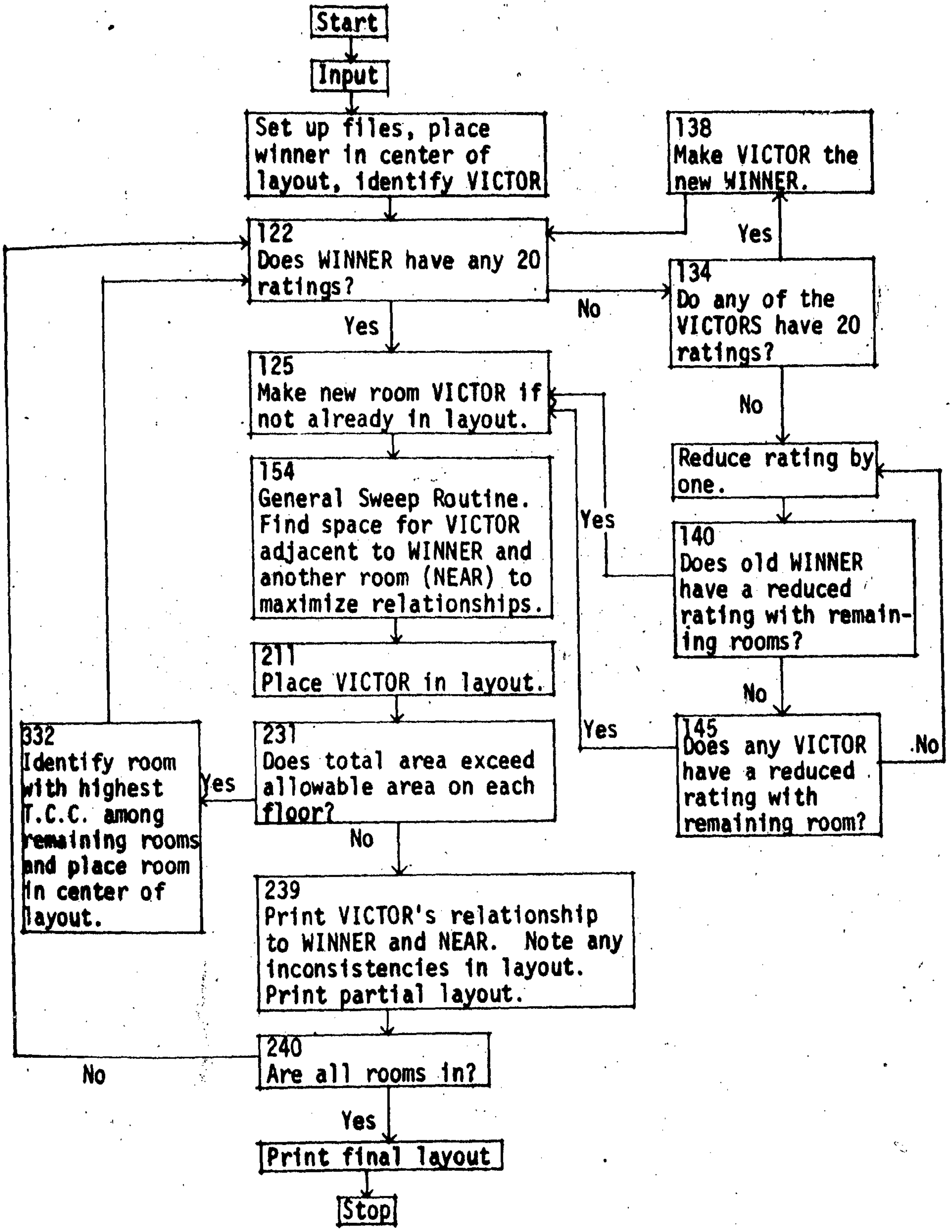
The room with the highest T.C.C. (Total circulation cost) is selected and placed in the centre of the layout matrix. A heuristic search is conducted of the remaining rooms to see which of them has a 20 rating with the room already placed. This is then placed in the layout. The process continues until all rooms are placed within the layout or the allowable area on the floor is reached, where upon it places the rest of the rooms on the next floor.

4.3.3 COMSBUL Input :

The program will accept problems involving upto 35 rooms. The scale of the printout is determined by the user.

The input data is as follows :

1. The number of rooms.



COMSBUL overall logic flow chart

Figure 4-1

2. The size of the module (unit square).
3. Length to width ratio (maximum).
4. Area requirements for each of the rooms.
5. Number of floor levels.
6. The circulation cost chart (symmetrical matrix form) (Figures 4.4, 4.5, 4.6).

4.3.4 COMSBUL Output :

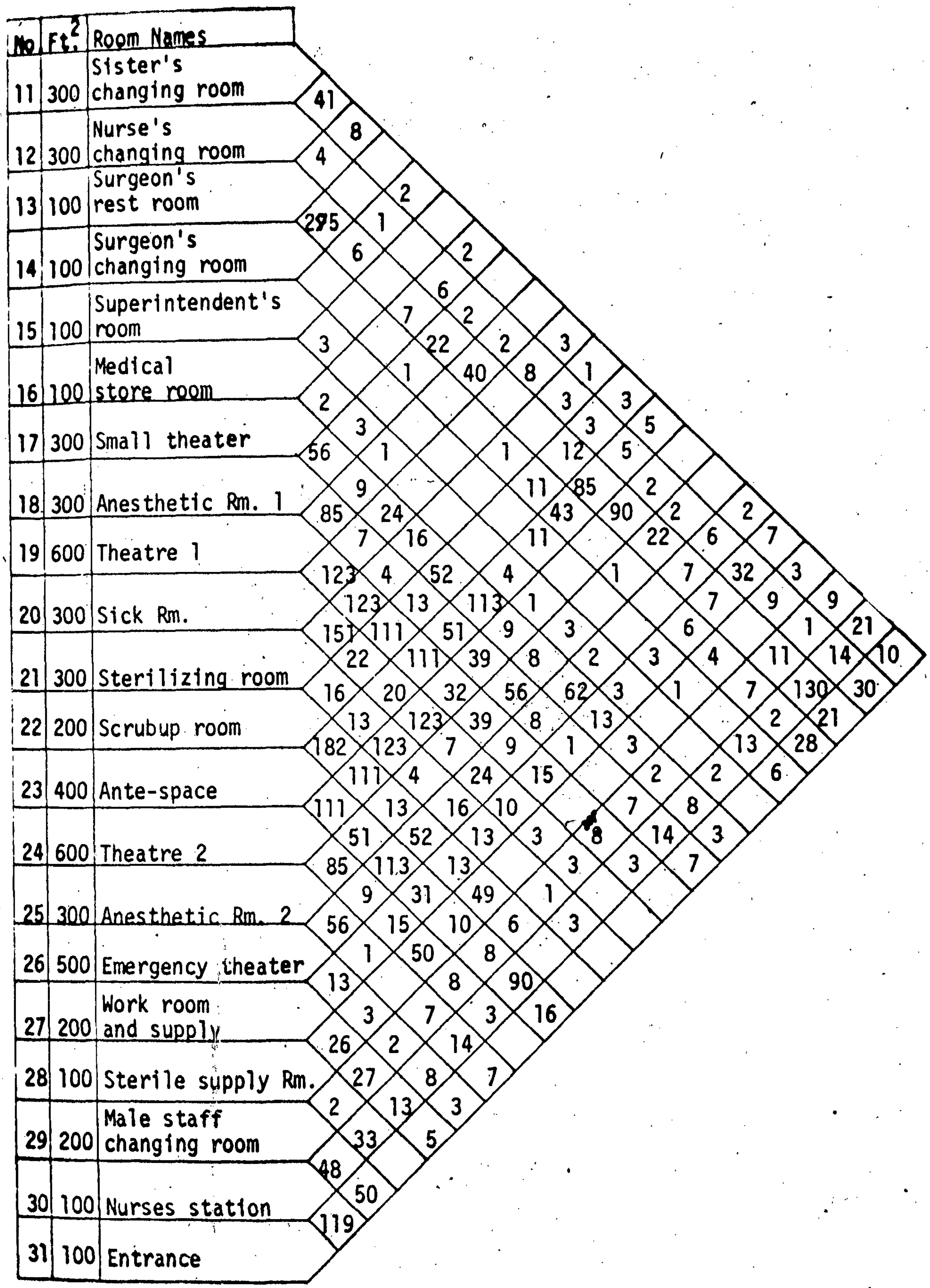
The computer outputs of the final layouts for the floor levels, rooms being indicated as two digit numbers. Each numbers represents the module (say 10 ft x 10 ft.). The groups of elements in the location matrix can then be outlined to indicate the locations of the rooms(Figure 4.7 and 4.8).

The designer's task in using this procedure is to amend the optimum solution to take into account restrictions such as site limit, height of buildings, fixed location of certain rooms etc.

The last step is to convert the theoretical layout into a practicable form (Figure 4.9).

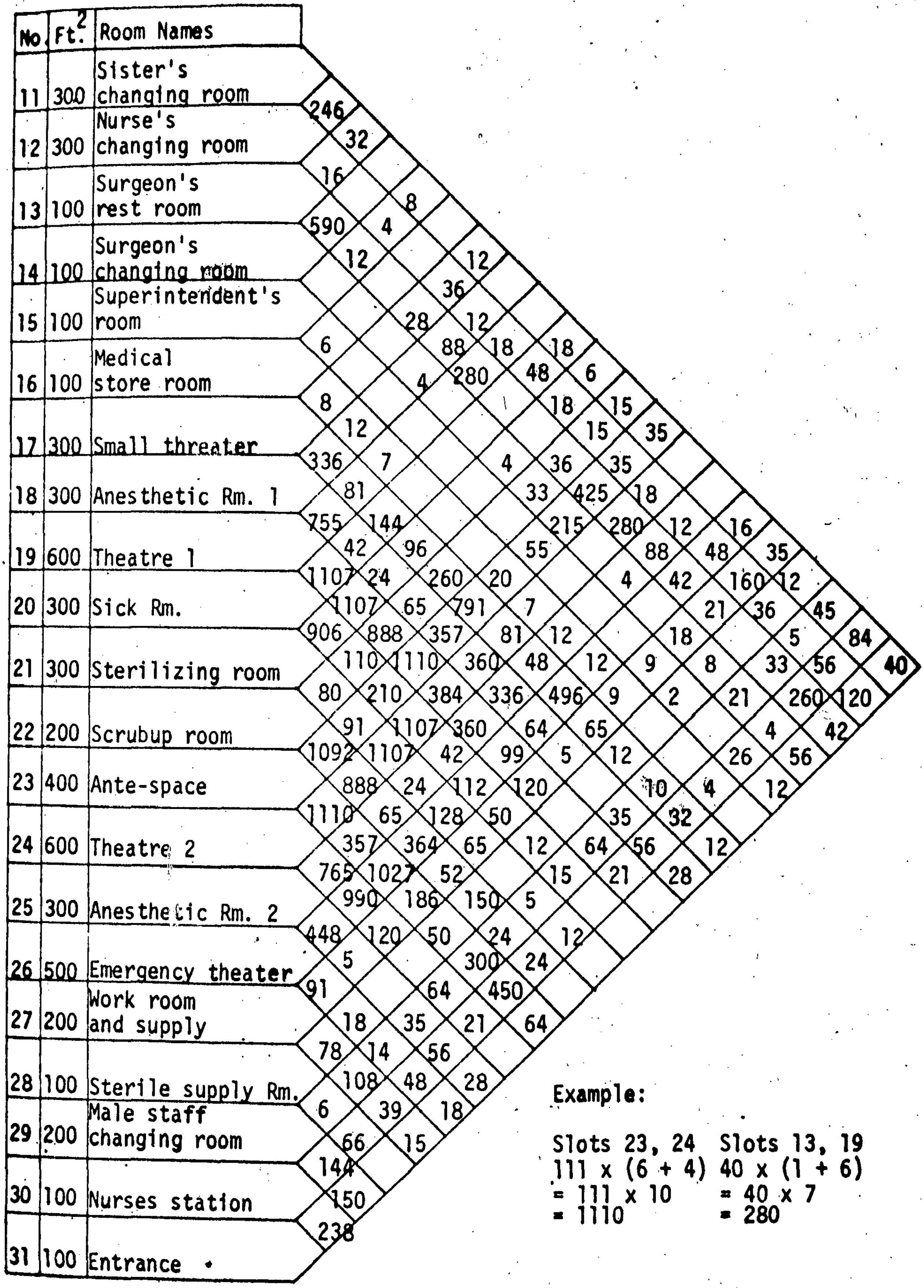
4.4 DESIGN PROBLEMS THAT CAN BENEFIT FROM THE USE OF THIS CONCEPT :

Generally any type of building which has a great deal of movement within can benefit from the use of the circulation cost concept.



Circulation cost chart (cost factor incorporated)

Figure 4-2



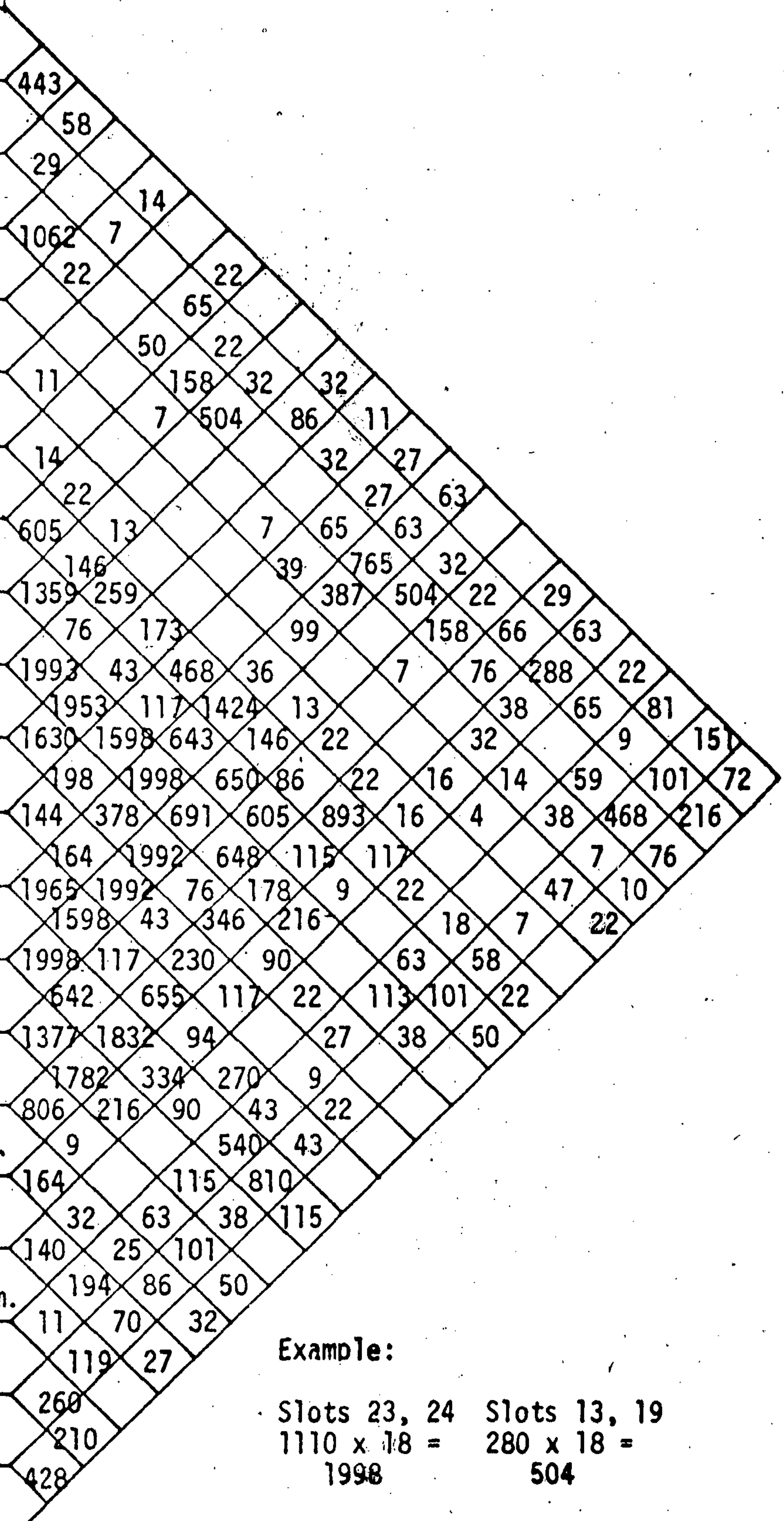
Example:

Slots 23, 24	Slots 13, 19
111 x (6 + 4)	40 x (1 + 6)
= 111 x 10	= 40 x 7
= 1110	= 280

Circulation cost chart 2 (area factor being incorporated)

Figure 4-3

No	Ft²	Room Names
11	300	Sister's changing room
12	300	Nurse's changing room
13	100	Surgeon's rest room
14	100	Surgeon's changing room
15	100	Superintendent's room
16	100	Medical store room
17	300	Small threater
18	300	Anesthetic Rm. 1
19	600	Theatre 1
20	300	Sick Rm.
21	300	Sterilizing room
22	200	Scrubup room
23	400	Ante-space
24	600	Theatre 2
25	300	Anesthetic Rm. 2
26	500	Emergency theater
27	200	Work room and supply
28	100	Sterile supply Rm.
29	200	Male staff changing room
30	100	Nurses station
31	100	Entrance



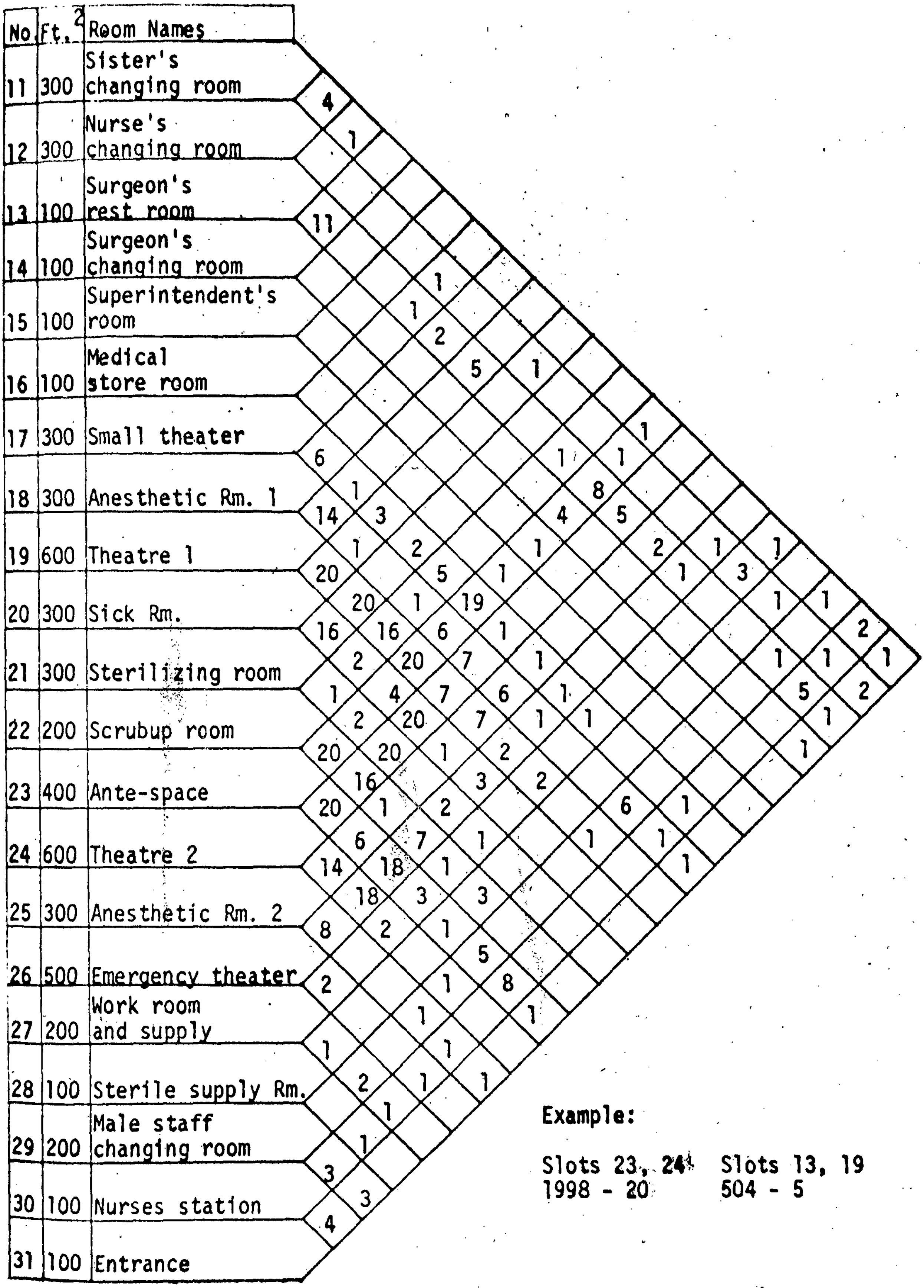
Example:

Slots 23, 24 — 1110 x 18 = 1998

Slots 13, 19 — 280 x 18 = 504

Circulation cost chart 3 (increased to maximum of 2000 ratings)

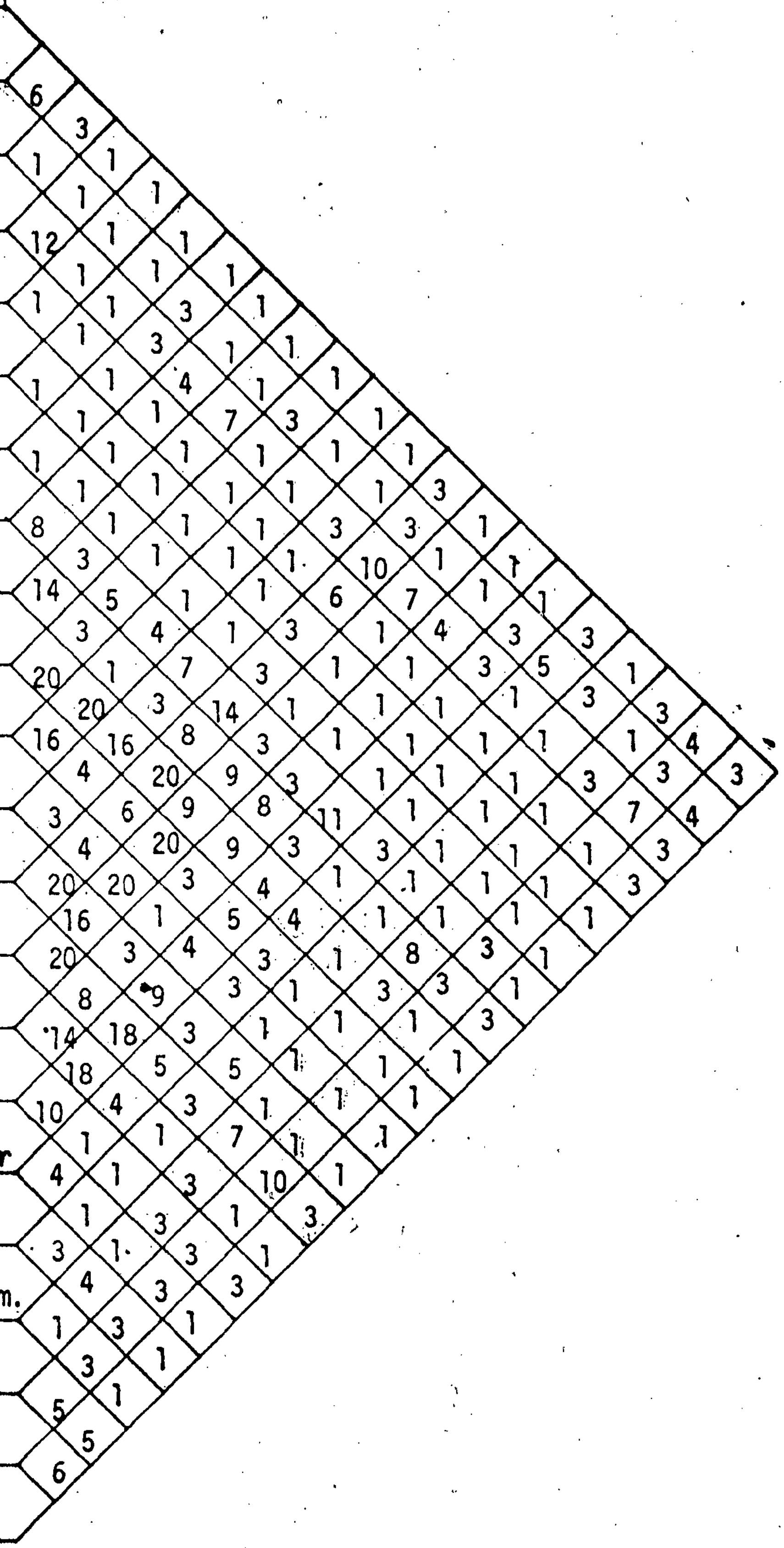
Figure 4-4



Final circulation cost chart (reduced to 20 ratings)

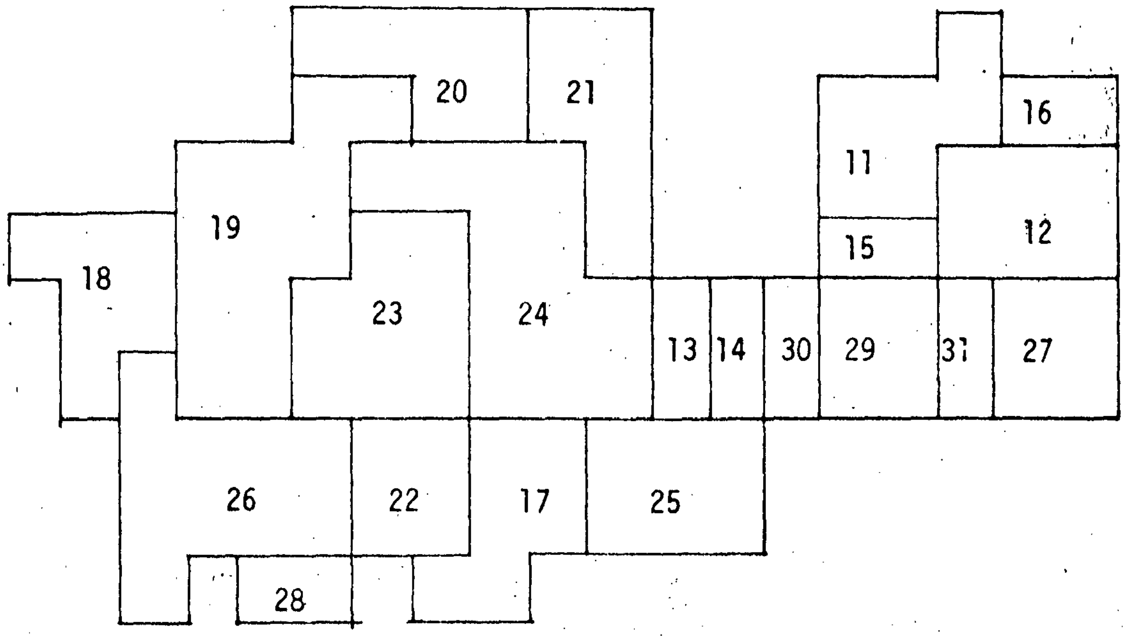
Figure 4-5

No.	Ft.^{2}	Room Names
11	300	Sister's changing room
12	300	Nurse's changing room
13	100	Surgeon's rest room
14	100	Surgeon's changing room
15	100	Superintendent's room
16	100	Medical store room
17	300	Small theater
18	300	Anesthetic Rm. 1
19	600	Theatre 1
20	300	Sick Rm.
21	300	Sterilizing room
22	200	Scrubup room
23	400	Ante-space
24	600	Theatre 2
25	300	Anesthetic Rm. 2
26	500	Emergency theater
27	200	Work room and supply
28	100	Sterile supply Rm.
29	200	Male staff changing room
30	100	Nurses station
31	100	Entrance



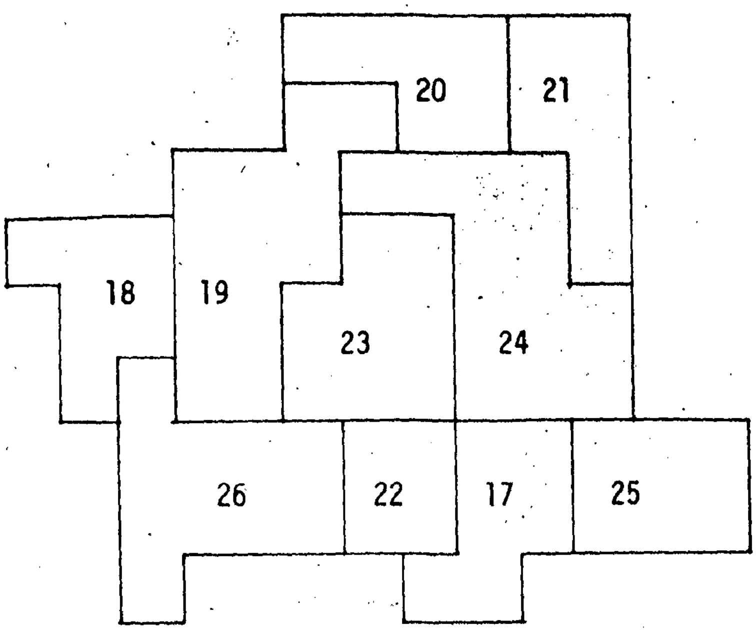
Final circulation cost chart (increased to fill unused ratings)

Figure 4-6

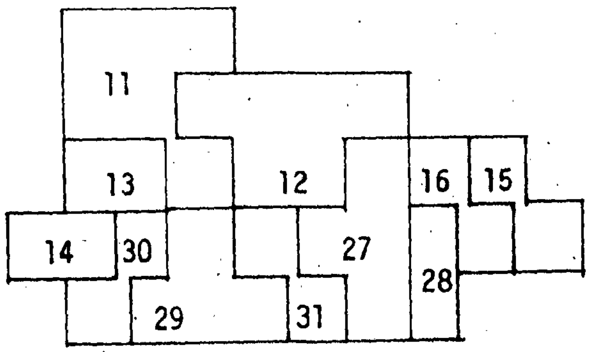


Single story computer layout
No scale

Figure 4-7

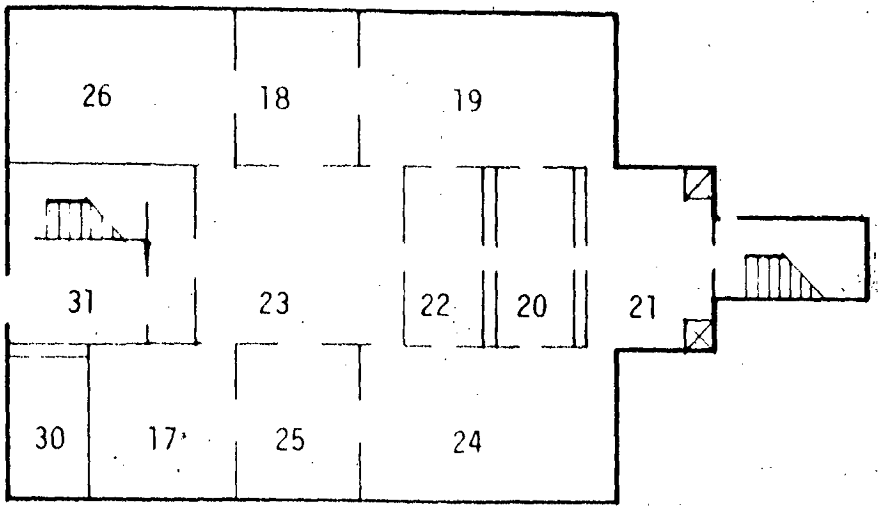


Ground floor computer output

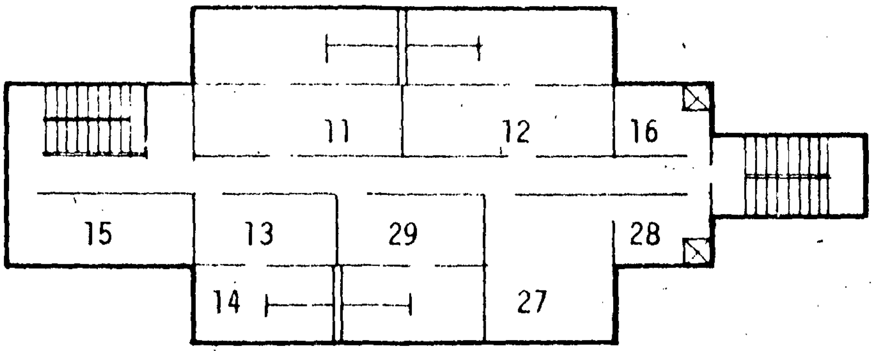


Second floor computer output
No scale

Figure 4-8



Ground floor plan



Second floor plan
Scale - 1" = 20'

Figure 4-9

In case of a hospital, it is usually understood that the communication efficiency of material, personnel and so on is a prime factor in evaluating the workability of a plan layout.

In an office building, lines of communication can be expressed in the physical placement of staff work stations. People who need to communicate by travelling should be close to each other. Other types of layouts may be, factories, laboratories, schools, libraries , or even kitchens.

4.5 LIMITATIONS :

Since the circulation cost matrix is the main data and is dependant on the traffic generated between rooms, the traffic pattern information needs to be carefully generated. To carry out surveys of similar buildings already in existence will be economical only when large building program is being carried out and would certainly be beyond the resources of most private offices.

Since certain optimization procedures are undertaken only one output can be generated per program run. To get alternate solutions, certain factors have to be modified.

The program lacks generality and the architect feels left out. There is no man-machine interaction. Only when the layout is generated does the architect enter the picture.

The applicability of the program is more suitable to buildings that generate excessive traffic and is rather wasted in other general buildings.

5. A MULTI CONSTRAINED TECHNIQUE FOR CONSTRUCTING TWO DIMENSIONAL SPATIAL ARRANGEMENTS :

5.1 PREAMBLE :

The following program is a development of the work done by Charles, M. Eastman at the Institute of Physical Planning, Carnegie-Mellon University, Pittsburgh, Pennsylvania, U.S.A. Mr. Eastman has spent several years in the research and development of spatial planning techniques. The set of routines suggested by him can be incorporated as the basic machinery for defining, manipulating and testing of two-dimensional arrangements of objects.

The object of the current work is to develop the main controlling program which would systematically develop a two-dimensional spatial layout from the data provided. The idea is to make it as user-oriented as possible to simulate the conditions under which an architect generally works when achieving spatial layouts through conventional means.

5.2 THE UNDERLYING MODEL OF DESIGN ACTIVITIES(FIGURE 6.1) :

The program incorporates two classes of elements, namely, Objects and Spaces. A Space is a bounded empty domain, while an Object is a physical entity or activity assigned to a space. In other words, Spaces are like empty boxes and Objects are entities assigned to positions within the boxes.

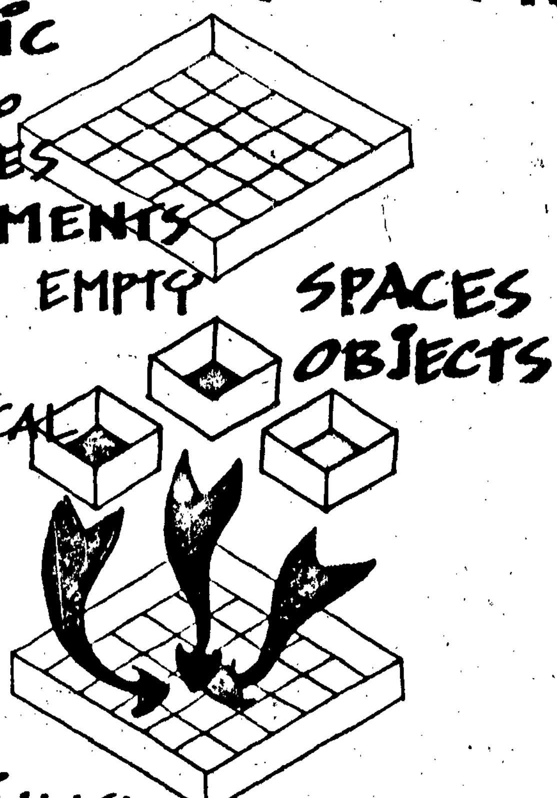
A SPATIAL SYNTHESIS PROGRAM

BASED ON A HEURISTIC SEARCH PROCEDURE.

THE PROGRAM INCORPORATES **TWO CLASSES OF ELEMENTS**

A 'SPACE' – A BOUNDED EMPTY DOMAIN.

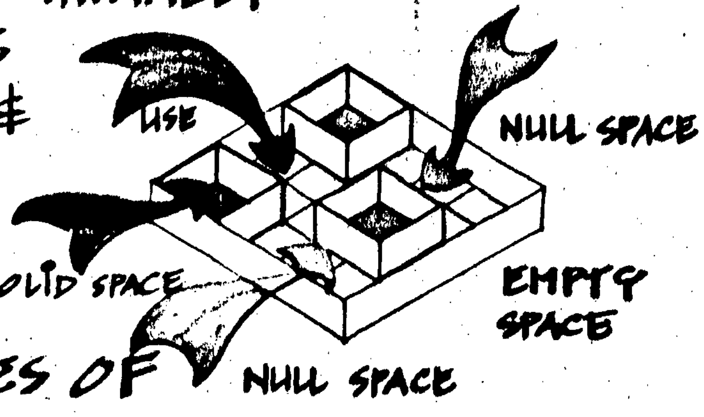
AN 'OBJECT' – A PHYSICAL ENTITY OR ACTIVITY ASSIGNED TO A 'SPACE'.



'SPACE' – AN EMPTY BOX

'OBJECTS' – ENTITIES

SPACES & OBJECTS INITIALLY DEFINED IN TERMS OF THEIR SHAPES & PROPERTIES.



SEMANTICS FOR TREATING **FOUR DIFFERENT TYPES OF SPACES** – INCORPORATED.

EMPTY SPACE: AVAILABLE SPACE.

SOLID SPACE: OCCUPIED BY OBJECT.

USE SPACE: OCCUPIED TEMPORARILY.

NULL SPACE: LEFT-OVER SPACE.

COMPUTER AIDED ARCHITECTURAL DESIGN

S. SUKUMAR – M.ARCH 1983-84
UNIVERSITY OF ROORKEE

Figure 6-1

In order to define and assign meaning to these elements, the program incorporates four different types of conditions.

Empty Space is the available area within a Space.
Solid represents space occupied permanently by an Object. Example - Walls, furniture, columns etc.

Use Spaces denotes spaces occupied temporarily. Example, door swings, etc.

Null Space is the complement to the shape of a Space or Object. Null Space in a Space may not be used for any purpose, while in an Object it depicts the area not included in the Object.

Spaces are made of Empty Spaces. Objects are made of Solids, Use Spaces or some combination of the two. In both cases Null Space is automatically defined around them.

Solids are permanent assignments to a space and cannot overlap anything other than Empty Space.

Use Spaces are temporary assignments of a Space.

5.3 THE PROGRAM CONCEPT :

Initially the user defines a set of Objects and Spaces in terms of their properties and shape. Their shape is defined as a template. The locating of an object is the mapping of an Object template into a Space template, similar to tracing. Removing an Object is the reverse mapping. Only one mapping of each Object into a Space is allowed.

The user initially defines the Empty Spaces and maps the Objects one by one into the bounded domain, by means of coordinating vectors. The tests provided in the program all evaluate spatial arrangements according to some criterion. Either they are satisfied or they fail. The tests check for adjacency, distance, sight and orientation of one object to the space, or of one object to another.

5.4 REPRESENTATION OF ELEMENTS :

The representation used to depict Objects and Spaces allows elements to be defined as any combination of rectangles in two dimensions. The rectangles need not be adjacent. Thus one Object may be the whole set of columns on the interior of a building.

Each element is identified by an index. The index of Spaces begins with 100. The index of Objects ranges from 1 to 99. The following representations are used to denote what occupies a particular space.

1 - 99	:	Empty Space
501 - 900	:	Use Space
9001 - 9400	:	Solid Space
9999	:	Null Space

Each Object and Space is assigned a unique value for its different types of space. An Object's use space, is denoted by 500 plus the Object index. For Spaces, Empty Space is denoted by their index minus 100.

Included in the definition of each Object and Space may be upto 5 reference points, referred to by an index. The points must be defined within the shape defined. Also included in the definition is an index of each side that borders its rectangular null.

5.5 PROGRAM INPUT :

The user must initially define the following :-

1. The maximum number of objects.
2. The maximum number of spaces.
3. The array size of the object - defining the maximum complexity.
4. The array size of the space - defining the maximum complexity.

The input for representation of the objects and spaces consists of the following :

1. Index of the element

   Index of objects : 1 to number of objects.

   Index of Spaces : 101 to (number of spaces + 100).

2. Definition of Object or Space

(a) Type of Space - E for Empty Spaces
U for use space
S for solid space.

(b) Left X coordinate.

(c) Right X coordinate.

(d) Top Y coordinate.

(e) Bottom Y coordinate.

Ten rectangles may be defined to describe each object or space.

All shapes are upper and left justified.

3. Reference points on the elements :
Upto 5 reference points may be specified for each element.
These are defined by the X and Y coordinates from the origin of the element (Top, left corner).

5.6 BASIC DEFINITION AND OPERATION OF EACH ROUTINE :

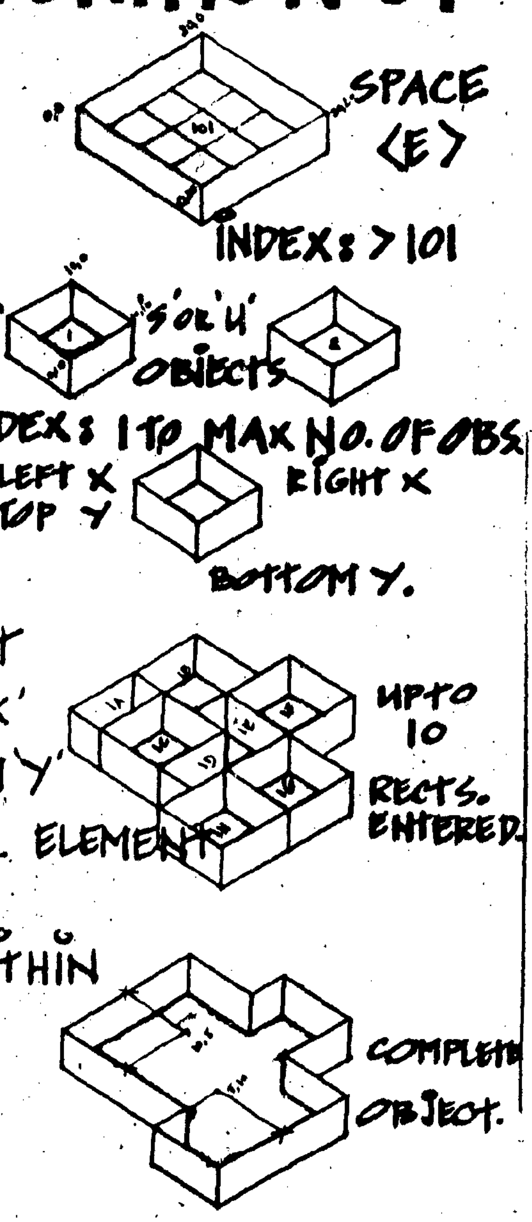
5.6.1 Initial Definition of Objects and their Shapes - Subroutine INIT (Figure 6.2) :

* The user initially defines objects and spaces through this routine.

* Each call to INIT initiates a series of READ statements that define an element from data.

INIT: INITIAL DEFENITION OF ELEMENTS AND THEIR SHAPES:

FIRST DATA IMAGE - INDEX OF SPACE OR OBJECT.
DEFINE SPACE USAGE.
'E' - EMPTY SPACE
'U' - USE SPACE
'S' - SOLID SPACE
CO-ORDINATES OF ELEMENT
LEFT 'X' CO-ORDINATE - RIGHT 'X'
TOP 'Y' CO-ORDINATE - BOTTOM 'Y'
UPTO 10 RECTANGLES FOR ELEMENT DEFENITION
5 POINTS REFERENCED WITHIN ELEMENT.



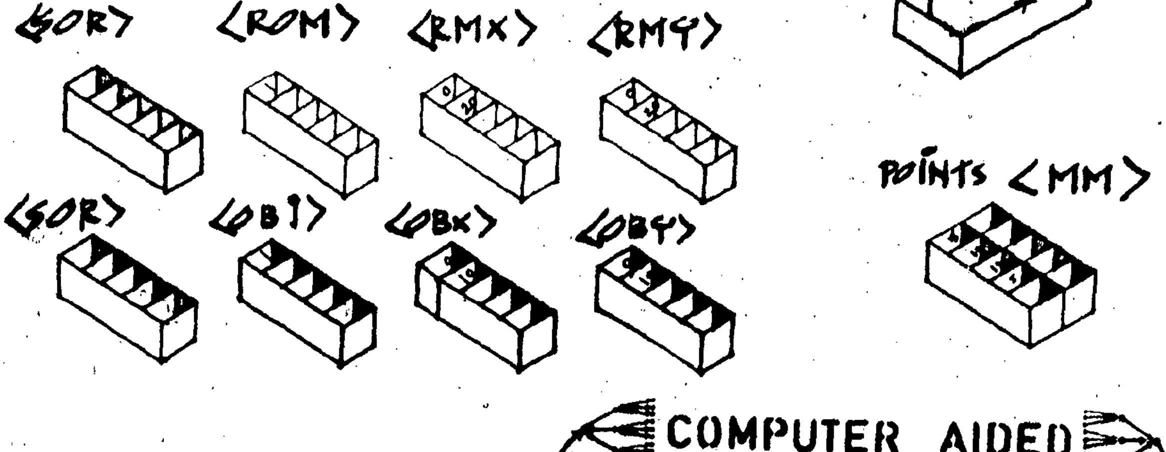
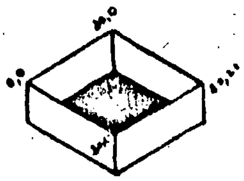


Figure 6-2

COMPUTER AIDED ARCHITECTURAL DESIGN

S. SUKUMAR - M. ARCH - 1983-84
UNIVERSITY OF ROORKEE

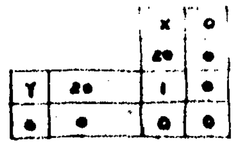
DEBUG: DISPLAYING AN ELEMENT.



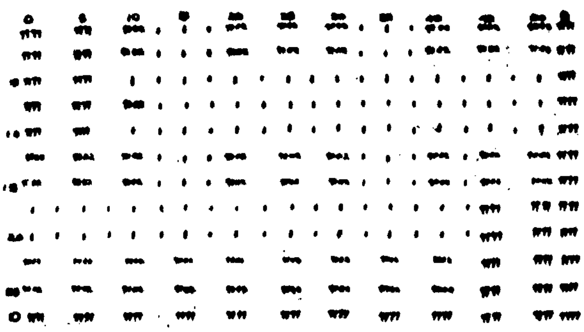
REQUIRES ONLY INDEX OF SPACE OR OBJECT

PRINTS OUT INTERIM ALPHA-NUMERIC OUTPUT - QUITE LEGIBLE GRAPHICALLY.

INDEX OF SPACE OR OBJECT



DISPLAYED OBJECT



REPRESENTATION

EMPTY SPACE - 1 TO 99

USE SPACE - 501 TO 900

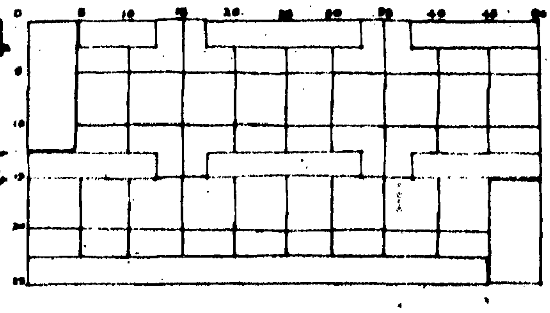
SOLIDS - 9001 TO 9400

NULL SPACE - 9999

ZERO VECTORS ADDED TO REPRESENTATION - TO DENOTE CARTESIAN CO-ORDINATES.

A LIST OF REFERENCE POINTS IN THE ELEMENT(S) ARE ALSO GIVEN.

DISPLAYING A LAYOUT



SCALED DRAWING OF LAYOUT.

WHEN IMPLEMENTED IN CONJUNCTION WITH SOME GRAPHIC HARDWARE SCALED DRAWINGS POSSIBLE.

COMPUTER AIDED ARCHITECTURAL DESIGN
S SUKUMAR— M.ARCH-1983-84
UNIVERSITY OF ROORKEE

Figure 6-4

* The subroutine call provides the element index, which also indicates if the element is a space or an object.

* The user defines a sufficient number of rectangles (not greater than 10) to describe the element.

* The routine reads all data into a STO array, removes duplicate values for X and Y dimensions and stores the X dimensions in OBX (for objects) RMX (for spaces) and Y dimensions in OBY (for objects), RMY(for spaces). The type of space is also stored. The reference points are stored in a MM array.

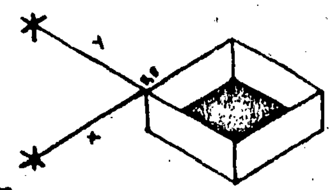
5.6.2 Displaying an Element-Subroutine DEBUG (Figure 6.4) :

This routine provides the interim alpha-numeric output, if no graphic hardware is available. This routine only requires the index of the object or the space. It prints out the corresponding variable domain array, with zero vectors at the top and left. After the array, a list of reference points in the element are also given. If a space is output and it includes one or more objects, the objects are designated in the space array by their code.

5.6.3 Locating an Object in a space - Subroutine INCL (Figure 6.3) :

This routine takes as arguments an object, the X and Y coordinates of the origin point of the object where it

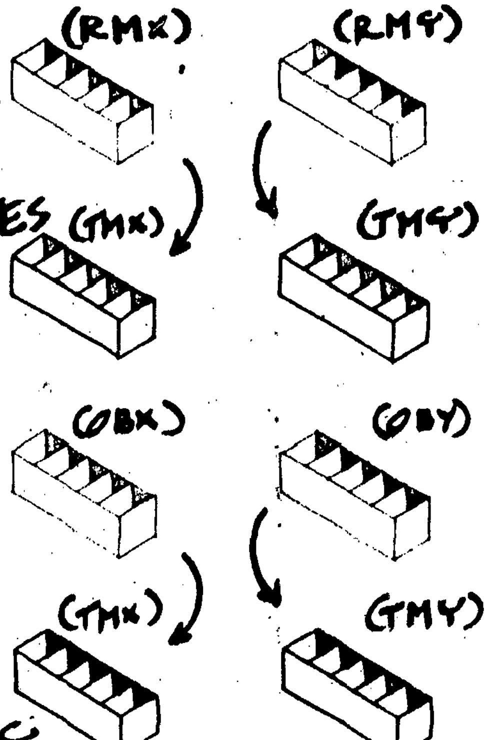
INCL: LOCATING AN OBJECT IN A SPACE.



ARGUMENTS: THE INDEX OF AN OBJECT, X & Y CO-ORDINATES OF THE ORIGIN POINT OF THE OBJECT WHERE IT IS TO BE LOCATED, AND THE SPACE IN WHICH IT IS TO BE LOCATED.

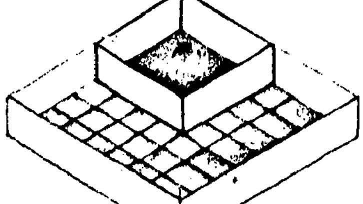
THE PROGRAM GENERATES THREE EXPANDED TEMORARY ARRAYS.

SUBROUTINE **CREATE** IS CALLED TWICE TO MAP THE PARTITIONS MATCHING THE **OBJECT** AND THE **SPACE.**



THE CELLS OF THE ARRAY ARE FILLED, FIRST BY MAPPING IN VALUES CORRESPOINDING TO THE **SPACE** AND THEN **OBJECT.**

THIS IS ACHIEVED BY TWO CALLS TO SUBROUTINE **PUT.**



ILLEGAL OVERLAPS ARE CHECKED.

COMPUTER AIDED ARCHITECTURAL DESIGN
S SUKUMAR – M. ARCH · 1983-84
UNIVERSITY OF ROORKEE.

Figure 6-3

is to be located, and the space in which it is to be located. INCL, maps the designated object into the appropriate space and returns the updated space, together with auxiliary book-keeping on the status of the object.

INCL functions in a manner similar to INIT. It generates three expanded temporary arrays with partitions translated by the X and Y coordinates of both Space and Object. This is accomplished by the subroutine CREATE. The cells of the array are then filled, first by mapping in values corresponding to the space and then the object. This is achieved by two calls to the subroutine PUT. As the object is entered, each cell is checked to see that there are no illegal overlaps. After the object is entered and if all cells are legal, the temporary arrays are mapped back into the space arrays. If a cell is encountered with illegal overlaps, all operation halt and the original Space array is returned with a message of failure.

5.6.4 Rotating an Object - Subroutine ROT :

This routine rotates an object in increments of 90°. It takes as arguments the object index and the number of clockwise 90° rotations desired. Minus values for the number of rotations are accepted, resulting in counter clockwise rotations. With 90° rotations the effective range is four and other values are changed to the modulus.

Any rotation is made in a single step, not by iterated calls to a single 90^{0} rotation. Each is achieved by the proper mapping of the object arrays into temporary arrays then back again.

5.6.5 Removing an Object from a Space - Subroutine OUT :

In order to relocate an Object, it must first be removed from its current location. This is achieved by OUT. OUT subtracts the Object array from its current position in the non-zero part of the Space array. The Space and location of each Object are stored in the MM array and these are automatically retrieved by OUT. After the subroutine is executed another subroutine, CON, is applied to reduce the Space array to its smallest non-redundant size. It is called twice, once for reduction in the X coordinate, then again for the Y coordinate.

5.6.6 Testing for Adjacency - Subroutine ADJ :

This routine evaluates whether two different elements are adjacent. It more precisely evaluates whether all of a specified side of the second element has a common border with any side of the first element. The second element must be an object, while the first may be a space or object. The routine prints out the result of the test and assigns to INFO, 0 for passing, 19 for failure.

5.6.7 Test for a Clear View between elements - Subroutine SIT :

This routine evaluates whether there is a clear view between portions of two elements. The elements may be a Space or Objects. Input to the routine are the two element indices, and optionally, two points on each. A subroutine of SIT, named STI, orders the points into a quadrangle. This quadrangle is then scanned by another subroutine SCT, to determine if any solid or Null Space is blocking the view within it. If it is, 23 is assigned to INFO, or else 0 is assigned for an acceptable arrangement. If no points are provided on one or both elements, SIT assigns points designating the full width of the element, thus requiring that it be completely in view (This option of not defining points is only applicable to an object. Points must be specified for any designated space).

5.6.8 Test for distance between parts of two elements - Subroutine DI2 :

This routine evaluates the straight line distance between the points on two elements. It takes as input two element indices (Spaces or Objects), a maximum allowed distance and optionally, a point on one or both elements. If points are assigned, DI2 computes the distance between them and compares this with the allowed limit. If the point on one or both elements is not assigned, the routine will define them as the point on the element resulting in the minimum distance to the other element. If one of the elements is a Space, the point on it must be defined, in this case, the point is not optional. If DI2 fails it returns 29 in INFO.

5.6.9 Test for Orientation of one Object with regard to another - Subroutine ORT :

This routine checks whether a specified side of one object is oriented toward another object. It takes as input two objects and a side of the second one. By riented toward', is meant that the designated side of the second Object is one of the two (out of four orientation) that is facing less than 180° from the first object. Failure is indicated by 31 or else 0.

5.6.10 Limitations :

These routines have been used in the implementation of two large computer-aided design projects and were originally developed for one of them, General Space Planner. They have value for student exercises and research in those problem areas requiring some capability in designing spatial arrangements.

The routines and the structure behind them are limited but efficient is processing time and memory requirements.

The representation of objects is limited to shapes that can be approximated by an adjacent set of rectangles. The routines included only allow rotation in increments of ninety degrees. On the other hand, there are no limitations on the dimensions of the objects and they may be quite complex in shape.

The memory used in storing any arrangement is a function of its complexity, not its dimensions. Operation on the representation are also efficient.

6.0 COMPUTER AIDED ARCHITECTURAL DESIGN - IN RETROSPECT :

Past and Present

Architects have in the past viewed computers with automatic suspicion or hostility, and have found little use for them in the practice. In contrast to this, other allied professionals in the construction team have made extensive use of computer and are rapidly increasing that use. For example, the structural engineers have utilised small inexpensive computers to handle their laborious calculations and have been able to pay more attention to general principles and to the exploration of alternative solutions.

The service engineers have found computers capable of solving problems in the design of optimal pipe and duct network and carrying out extensive checks on heating and lighting provision. They have been able, therefore to specify economic, yet adequate solution of greater competency.

Quantity surveyors have made use of computer libraries of standard phrases from which bills of quantities may be generated.

The larger contractors normally utilize the computers in construction management, to make sure that they have the optimum number of men and machines on site and to organise the ordering of materials at the correct time. The architects hostility and suspicion seems to be a reaction from the enthusiasm of earlier years. There was an intense feeling that computers were going to transform the profession as they were transforming other trades. In the 1970's, for example, engineering design was undergoing a complete restructuring, as calculations could be done both automatically and almost immediately. The effect was that the design team could be more compact and could work on a higher plane, thinking in terms of principles rather than specific, so that work could become more interesting and requirring rather different skills. The resultant design would naturally be of a higher quality than the normal equivalent.

At that time, there seemed every good reason to believer that the computers would have the same effect within the architectural professional also. By the 1980's this transformation did not happen and after a series of disappointment the general feeling was that th architects work is too complex and too intuitive to be significantly aided by the computer.

6.1 A REVIEW OF COMPUTER APPLICATIONS IN ARCHITECTURE :

Computers were first introduced in the 1950's and initially they were used for scientific calculations and for straight forward large scale business uses such as payroll production. In 1963 Ivan Sutherland at MIT introduced the famous SKETCHPAD system. This system allowed the user to directly draw on the screen of a television like device connected to a computer.

Three years later, William Newman, at Imperial College in London, developed a system for specifically architectural applications. From library of building elements, the user was able to assemble a plan on the screen. From the assembled plan, the computer produced a list of room areas, compiled a schedule of building elements, calculated the heat loss from the structure and assessed the natural and artificial lighting levels.

It was natural that such a system excited architects and everyone expected system like these to revolutionise the architectural practice in general.

At about the same period, a number of design theoretician like Christopher Alexander, L. Bruce Archer , and Christopher Jones began to make their presence felt. They attempted to find the basis of design from which systematic rules could be evolved to enable design to be more logical and less intuitive. Their methods often required the use of computer, to reduce to manageable form large tables of the interaction of each activity area with all other activity areas, or to apply complex mathematical optimisation techniques.

These ideas, attracted a great deal of attention, and people felt that the introduction of such techniques would remove much of the necessity for creative ability and accurate intuition on the part of the designer.

Another factor favouring the sense of optimum was the building boom in the 1970's and the increase in the Architects work loads. Labour too was expensive and hard to find. There was therefore both the incentive and the finance to develop solutions that would reduce the dependance on manual effort.

The period between late 1960's and early 1970's saw a spurt in attempts to use the computer. The architectural profession too was not averse to the introduction of new ideas and watched with eagerness the several conferences on computer aided architectural design, the several publications that infiltrates the market and the several reports and investigations into the different aspects of computer application in the construction industry. Architects confidently awaited the revolutionising of their profession.

To a certain context this did happen. In England, for example, the architecture department of the U.K's. West Sussex County Council developed an almost complete design system using computers as aids applied to the industrialized building system SCOLA. The U.K. Government's Department of Health and Social Security, introduced in 1969 a whole hospital design concept named HARNESS.

There were many other attempts to introduce computers both on a large and a small scale, but within a few years it became obvious that there was not going to be a revolution. When architects attempted to use computers for themselves they found that it was a full-time task with few rewards. A list of new techniques had to be learned and problem solved, more of the problems arising out of the handling of the machines and the relatively slow speed of the machines. There were not many software packages available, and those developed concentrated on problems requiring a lot of calculations, such as beam design, daylight factors, heat loss and gain, problems that were on the periphery of an architects interest. The central problems that occupied most of an architect time were not tackled at all. In addition, these programs generally required an enormous amount of data collection and preparation and the results did not justify the time and effort spent.

Another disappointment came in the shape of the much awaited draughting system that were slated to replace the drawing boards with television screens. These turned out to be extremely expensive and initially ruled out their use in any but government supported organisations. The draughting speed too was not much faster than manual draughting. Larger drawings had to be broken down into smaller section before they could be input and the drawings were displayed in thick and clumsy lines.

The building design software packages produced very superficial and naive designs. In most cases, they attempted to optimise a single factor, circulation cost being the most popular choice. They did not take into consideration the thousands of other factors that must be taken into account in producing a probable design.

As a result of these disappointment, some of the ambitious computer projects quietly closed down. In England, the West Sussex County Council went back to manual methods in 1974. The Dept. of Health and Social Security abandoned the HARNESS system in 1975. Most architects who had tried to use computer gave up, frustrated that the results did not justify the effort and cost.

As a consequence, there is understandably a good deal of resistance to computers in the professions and this is perhaps a correct and hard headed reaction to the facts. However progress has been going on quietly all the time and computers, can now be of real help to the architect, not by way of revolutionary solutions but in straightforward boosts to the working methods.

There have been advances in computer technology and architecture ,the prime achievement being the reduction in costs and increased power of the machinery. These basic improvements in cost and power have made it possible to write much more useful and more easily used programs.

With the growth in the number of computers, they have become much more accessible. Another result of the improvements in technology has been the vast mprovement in the ability of the computer to handle drawings. Initially the computer could handle only numerical application but demand from the business world caused the introduction of textual processing. Efforts made by Sutherland and others to express a graphical structure in terms of numbers succeeding and today several manufacturers are making computers expressively for graphical applications. A number of architectural firms are now using such machines.

For a program to be accepted now-a-days, it must fill a genuine need, be well written and documented and easy to use. As a consequence the program will be generally a long one and so slow and expensive to produce. For this reason, few serious programs are now written for particular problems in particular offices. There has grown up a sizeable market in the off-the-peg computer programs and a number of bodies now exist to evaluate and to publicise them.

In general, it may be stated that attitudes to computing today are much more professional. Expectation are not so high and there is recognition of the fact that computers do not offer adequate solution to some types of problems. The greater experience and more disciplined approach in conjunction with better and cheaper machinery have firmly established the viability of computers in many applications.

7. CONCLUSION AND SUGGESTIONS FOR FURTHER RESEARCH :

7.1 BENEFITS OF USING COMPUTERS :

Over the years, attempts have been made to use computers to aid virtually every task in the construction process. The programs currently available that are of interest to the architect cover a very wide range. They include programs that provide assistance at a very general level, such as those that give rough costings at the feasibility stage, and programs that help with very precise and well defined activities such as the production of working details.

Most of the advantages of computers derive from their being able to carry out long and repetitive calculations and comparisons, very much faster than human can. This greatly increased speed means that certain results can be produced that would take a prohibitively long time, perhaps even years, by manual method. The architect can use these results to gain more insight into a design and thus produce a better and more economical building than he could otherwise do. Alternatively the design period can be shortened in order to complete the building sooner, or the design team can be reduced, thus increasing efficiency and making staff savings.

7.2 DRAWBACKS OF USING COMPUTERS

Computers are not an unmixed blessing. They are unsuited or less suited, to certain kinds of problems and they inevitably introduce difficulties of their own. The nature of the problem or of the building or its scale, might preclude the efficient use of computers. They may impose extra and unfamiliar duties upon the architect, and they may disrupt the traditional way of working.

Obviously, a problem whose solution depends at least partly on subjective judgements is not normally suited to computer solution. For example, aesthetic valuation, or the considerations of human behaviour and relationships, cannot be adequately be dealt with by computers. Because much of the power of computers lies in their ability repeatedly to apply the same process, the building itself must contain a reasonable amount of repetition. If this is the case, the information process used in one situation can be used in many others, so producing an increase in efficiency. Many architects are troubled by the fear that this need for repetition might prove detrimental to the design.

As a conclusion it may be stated that at present there are two principal ways of using the computer in building design, in a generative way, to produce a room layout for example, and in an analytical way, to check the viability of a proposed design. Although the generative approach is the one that has traditionally received all the attention, and indeed is still the subject of much research, it has been completely abandoned by practising architects because the results produced are inadequate. This is mainly because it is not possible to build into a computer program many of the most important factors that must be considered in design. However if the architect produce a design and the computer then checks such aspects of it as it is able to, it is possible to create, a very powerful symbiosis of man and machine.

Analytical procedures, both non-dynamic (evaluation) and dynamic (simulation) have been applied and were successful. It is the generative procedure which has been attempted and found lacking. Hence it is this area of generative design, or spatial synthesis, which has potential for further research. This study is an attempt at reviewing the various formulations, techniques and procedures in automated spatial planning and synthesis to provide a foundation for continued research.

APPENDIX - 1

1.0 DESCRIPTIONS OF BUILDING TOPOLOGY AND GEOMETRY*

In a traditional design process, drawings (principally plans, elevations and sections) are employed to represent the shapes, dimensions, locations, connections, and relations of components and spaces in the building. Drawings are normally employed in performance of those design functions which require spatial reference, relational, and performance data, and which require the designer to check for spatial consistency. If these functions are to be automated, some way must be found to incorporate sufficient shape, dimensional locational, and relational information into data structures used to describe buildings in the computer.

One obvious expedient is merely to add extra fields for geometric data of various kinds to the types of data structures used for non geometric representation, as discussed. For example, fields for room shape, dimension, and location could be added to the records of a programmatic data file. Indeed in one rather simple-minded sense, the task of building planning in response to a given brief is a task of generating the information to fill these extra fields. But although such a straightforward technique for describing building geometry may be suitable for some purposes, it tends to prove inadequate as a basis for automated performance of the complex spatial synthesis and evaluation functions involved

---

* REFERENCE : Computer Aided Architectural Design
William J. Mitchell (1977).

in architecturel design. This Chepter discusses various alternative approaches which may be taken to the design of data structures of capable of representing building topology and geometry for these purposes.

1.1 TYPES OF GEOMETRIC DESCRIPTION

Different types of geometric description can be distinguished according to :

The type of geometric element upon which the description based,

The particular geometric and topological attributes of entities to be described,

The types of geometric and topological relations
9 between entities to be represented,

The level of detail at which the description is made,
The geometric assumptions upon which the description method is based.

1.1.1 <u>Elements</u>

A fishing net, according to the old joke, can be regarded either as a pattern of strings or as a pattern of holes. It depends upon how you look at it. Similarly, there are many different ways of looking at a building. We might choose to regard it as an assemblage of physical components, like columns, beams, etc., or of bounding surfaces like the planes of walls, floors, and ceilings, or (as in a conventionsl architectural drawing) of bounding lines marking edges and intersections of surfaces, or of enclosed functional volumes like rooms, or of abstract building blocks like square or cubic modules. Descriptions based upon different types of elements are suitable for different purposes. But the choice of geometric elements is not simply a technical question. In a very direct sense, the choice of elements begins to establish a language of architectural design.

1.1.2 Attributes

For most applications, it is necessary to describe the primary geometric properties of overall length, width, height, and orientation, and location in the buiding, of each element. Sometimes it is necessary to include detailed descriptions of shape properties. It may be worthwhile to explicitly store some secondary geometric data, like volume, surface area, etc. For engineering computations, it is often necessary to store non-geometric physical attributes like weight, U-value, etc.

1.1.3 Relations

For many applications, it is necessary to be able to determine which elements are adjacent to any specified element. Adjacency data may either be explicitly stored, implicitly represented by location of elements in the data structure, or computed as required from element dimension and location data. Other types of relations that it may be important to have the capability of determining are distances between elements, alignment, symmetry, intersection, visibility of one element from another, etc. Sometimes relations other than adjacency are explicitly stored, but it is more usual to compute them as required.

1.1.4 Level of detail

In a traditional design process, a general progression in level of detail of drawings tends to take place as design decisions are made. Early sketc hes may be at 1 :200 and show very little detail. Walls may be indicated by a single line, openings may be omitted etc., More developed sketches and working drawings may move upto 1 :100 or even 1 :50. Wall thicknesses are indicated, locations of openings are now shown, individual construction elements are more carefully differentiated, and so on.

When considering techniques for computer representation of building geometry, the stage of the design process for which the representation is intended must be kept in mind. A representation which incorproates too much or too little geometric information, or data at too high or too low a level of detail, will not be successful.

1.1.5 Geometric assumptions

Considerable efficiencies can be achieved in geometric description if some general geometric assumptions can be made, for example, that all angles are 90^{o}, that dimensions vary in modular increments, that all curves are segments of circles, or that all objects are convex polyhedra. Conversely, unless builbing form is developed within the framework of a disciplined geometry, it becomes very difficult both to describe and to construct.

1.2 RANGE OF APPROACHES

In response to the considerations discussed above, diverse approaches to structuring geometric descriptions of buildings have evolved. These can be grouped into five broad categories :

Representations based upon regular grids and lattices.
Representations based upon variably-dimensioned grids and lattices.

Polygon and polyhedron representations.
Dual-graph representations.
Smith diagram representations.

1.3 REPRESENTATIONS BASED ON REGULAR GRIDS AND LATTICES

Geometric representations based upon regular and semi-regular grids and lattices are among the easiest to understand and the most straightforward to implement. Hence they form an

:opriate starting point for discussion of geometric descriptic
uniques.

Plans of buildings are often constructed within square
:ectangular modular grids, (Figure 1.1) and less commonly witl
angular, hexagonal, or more complex grids. (Figure 1.2)
ending this concept to three dimensions, lattices composed of
es or rectangular parallelpipedes are often employed in the
eration of architectural forms.

.1 Square grids

Any arbitrary two-dimnsional shape can be represented
any required level of accuracy by a pattern of binary
nents (black and white squares, or 1's and 0's) wihin a
-dimensional square grid , (Figure 1.3) Each cell of the
d contains 1 bit of information, so the amount of information
uired to represent a form in this way is equal to m X n bits,
re m is the number of rows in the grid and n the number of
umns. The amount of information required rises exponentially
the level of accuracy is increased; if, for example, the widt
a cell is halved, the number of cells in the grid is quadrupl

Further data can be recorde by utilizing different
bers to represent different properties of the surface, for
mple, tone, as shown, Figure 1.4.Walls and other bounding
ments are usually not explicitly represented. A boundary is
licitly defined when adjacent cells contain different
egers. (Figure 1.4)

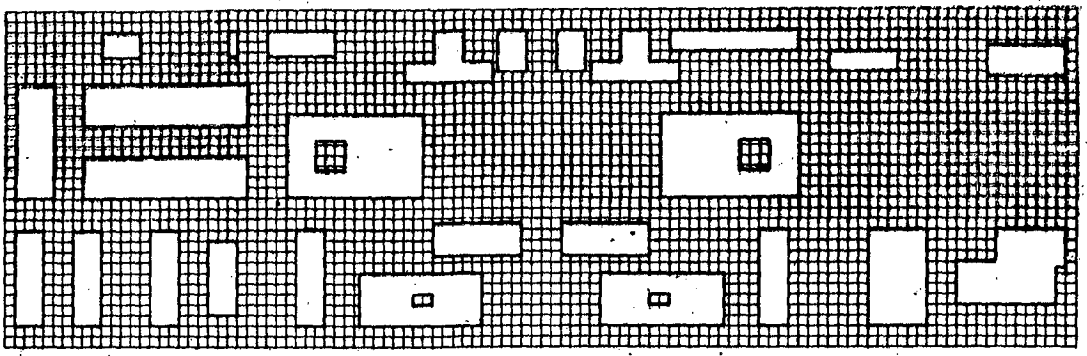


Figure 1-1
Use of a square grid in building design: site plan for the Illinois Institute of Technology, architect Ludwig Mies van der Rohe, 1940.

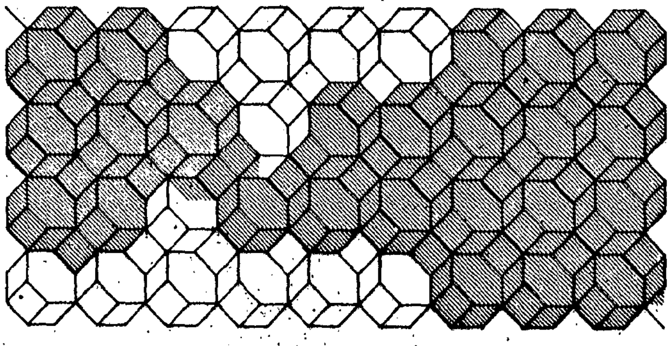


Figure 1-2
Plan torms of two projects by Walter Netsch, illustrating use of complex grids.

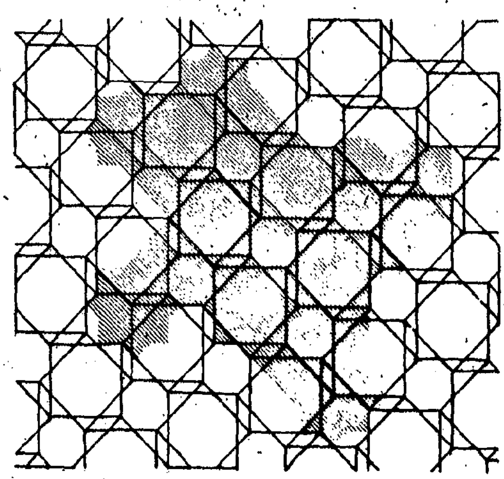
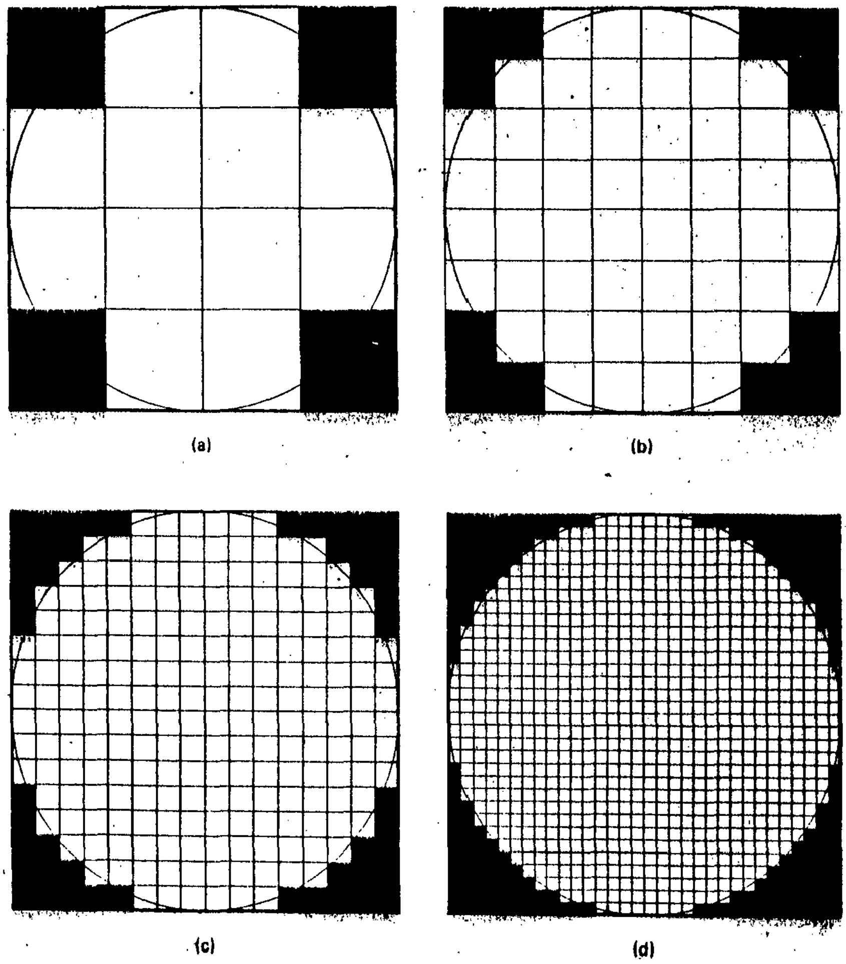


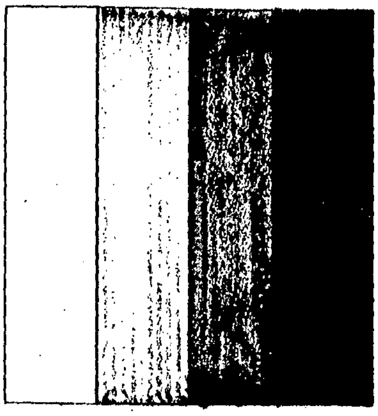
Figure
Project by O. Mathias Ungers for the Roosevelt Island housing competition, New York, 1974, illustrating use of a cubic module.

Figure 1-3

Representations of a circle within a square grid. (a) 16 bits. (b) 64 bits. (c) 256 bits. (d) 1024 bits.

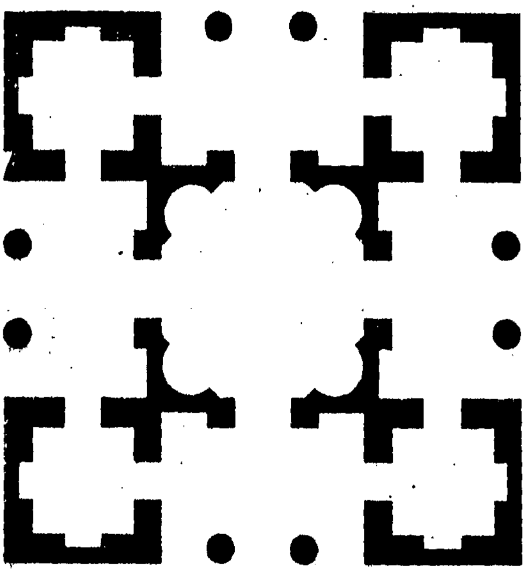


(a) (b) (c) (d)

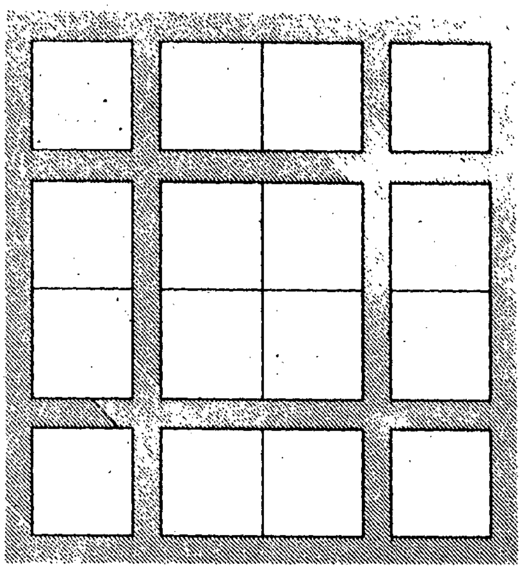


0	0	20	20	40	40	60	60
0	0	20	20	40	40	60	60
0	0	20	20	40	40	60	60
0	0	20	20	40	40	60	60
0	0	20	20	40	40	60	60
0	0	20	20	40	40	60	60
0	0	20	20	40	40	60	60
0	0	20	20	40	40	60	60

(a)



Project from Serlio's The Book of Architecture (London, 1611)



Schematized representation of the layout

1 2 2 3
4 5 5 6
4 5 5 6
7 8 8 9

Schematized representation encoded

(b)

Figure 1-4

Encoding forms as integer arrays. (a) Encoding a grey scale picture as an integer array. (b) Encoding a floor plan as an integer array.

Most of programs employing this representation have been written in FORTRAN, ALGOL, PL/1 or other high-level general purpose languages. The array handling facilities of these languages make it very straightforward and convenient to store plans encoded in this way as simple two-dimensional integer arrays, and this has almost invariably been the approach taken.

The two-dimensional array approach to representation of building plans is simple but powerful, because it effectively utilizes the structure of the array itself to implicitly represent positions and adjacencies of spatial elements.

The principal disadvantage of the simple two-dimensional integer array representation of floor plans is that it consumes a great deal of storage. A 100 module x 100 module grid, for example, requires 10,000 words of memory. To make matters worse, as we have seen, the number of modules required to describe a floor plan of given area grows exponentially as the dimensions of the module are decreased. Since many of the operations performed upon this type of representation involve iterating through the array and performing some operation upon each individual cell, growth in size of the array also tends to imply substantial increases in computation time.

To overcome the problem of excessive memory requirements, two variations of the simple two-dimensional array representation may be employed : hierarchical arrays and a form of string representation.

In a hierarchical array, grid cells are recursively subdivided into smaller cells as required to represent those portions of the plan in which fine detail occurs, as shown in Figure 1.6

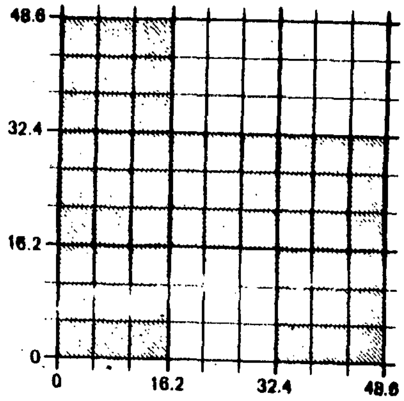
A number of variants of the string representation can be constructed. The basic principle underlying all of them is that the two-dimensional array can be unfolded into a one-dimensional arrary, then compressed (Figure 1.7) It can be seen that the resultant one-dimensional array consists of sequences of identical integers. The array can then be compressed by replacing each of these sequences by just two integers, the first giving the total number of identical integers in the sequence, and the second giving the integer from which the sequence is composed.

1.3.2 Rectangular grids

The concept of the square grid representation can easily be extended to allow for rectangular grid cells. This may be an advantage in situations where design of a floor plan is based upon use of a rectangular module, and it is desired to have the cells in the computer representation correspond to the module. It is merely necessary to store coefficients for the x-dimension and y-dimension of the module, and to apply these coefficients when calculating distances, areas, centroid locations, etc.

Figure 1-5

Harness hospital department laid out on a 5.4m planning grid located within 16.2m structural bays.



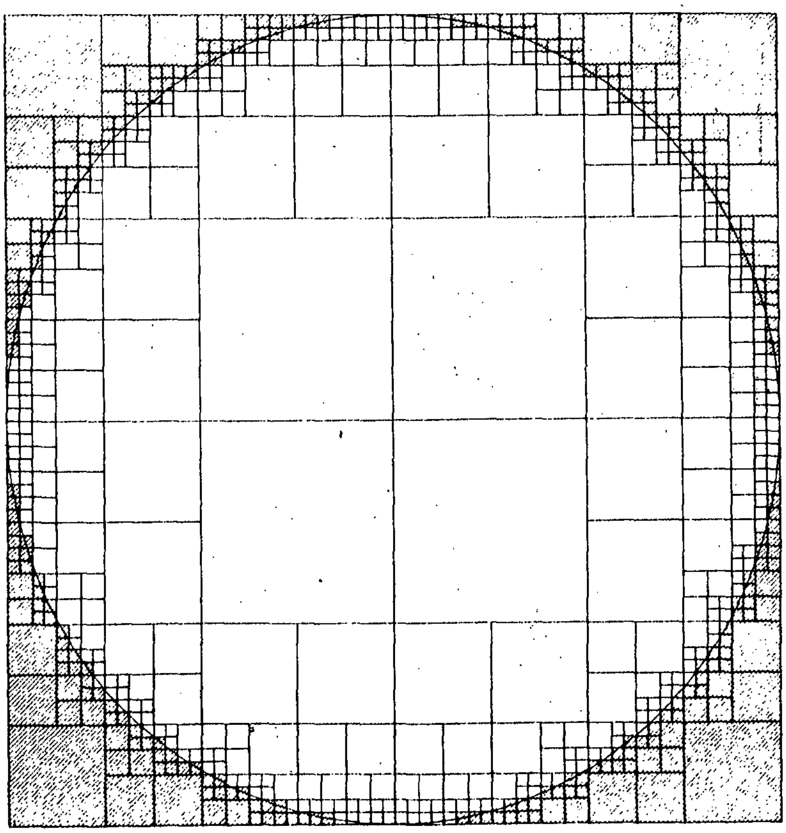
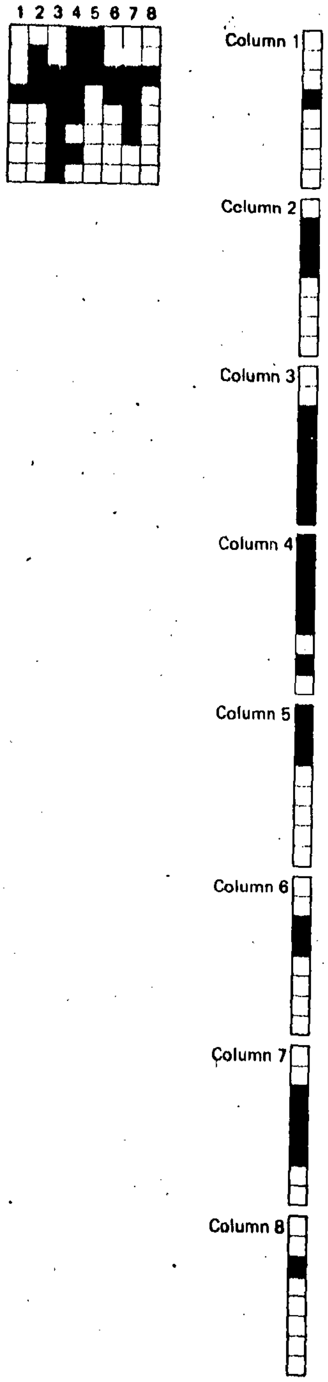


Figure 1-6

Representation of a plan form by a hierarchical array.



(a) (b)

Number	Type
3	0
1	1
5	0
3	1
6	0
11	1
1	0
1	1
1	0
3	1
7	0
2	1
6	0
4	1
4	0
1	1
5	0

(c)

Figure 1-7
Compressed string representation. (a) Two-dimensional array representation of a form. (b) Two-dimensional array unfolded into a one-dimensional string. (c) Compressed format for storage of string.

1.3.3 Cubic Lattices :

By utilizing three-dimensional instead of two-dimensional arrays, three-dimensional forms can be represented using the integer array technique (Figure 1.8). It should be noted, however, that the exponential growth in storage requirements with decreasing cell size is even more rapid in this case; every halving of the side-length results in an eight-fold increase in the number of cells. Hierarchical arrays and string representations may also be utilized in the three-dimensional case, and coefficients may be applied to produce a lattice composed of rectangular parallelpipeds.

1.3.4 Non-rectilinear grids :

In order to discuss representation of plan forms utilizing non-rectilinear grids, a brief excursion into the theory of symmetrical network patterns will be necessary.

Consider equally-spaced parallel rows of equidistant points, joined by straight lines, as illustrated in Figure 1.9. This may be termed a parallelogram system of equivalent points. A particular parallelogram system may be described by giving the side length x and y, and the angle θ between these sides. Forms represented as patterns of cells in a parallelogram system may be transformed by varying x, y and θ [Figure 1.10(a)]. This technique of transformation of form by varying the patameters of a parallelogram system was known to Durer, who

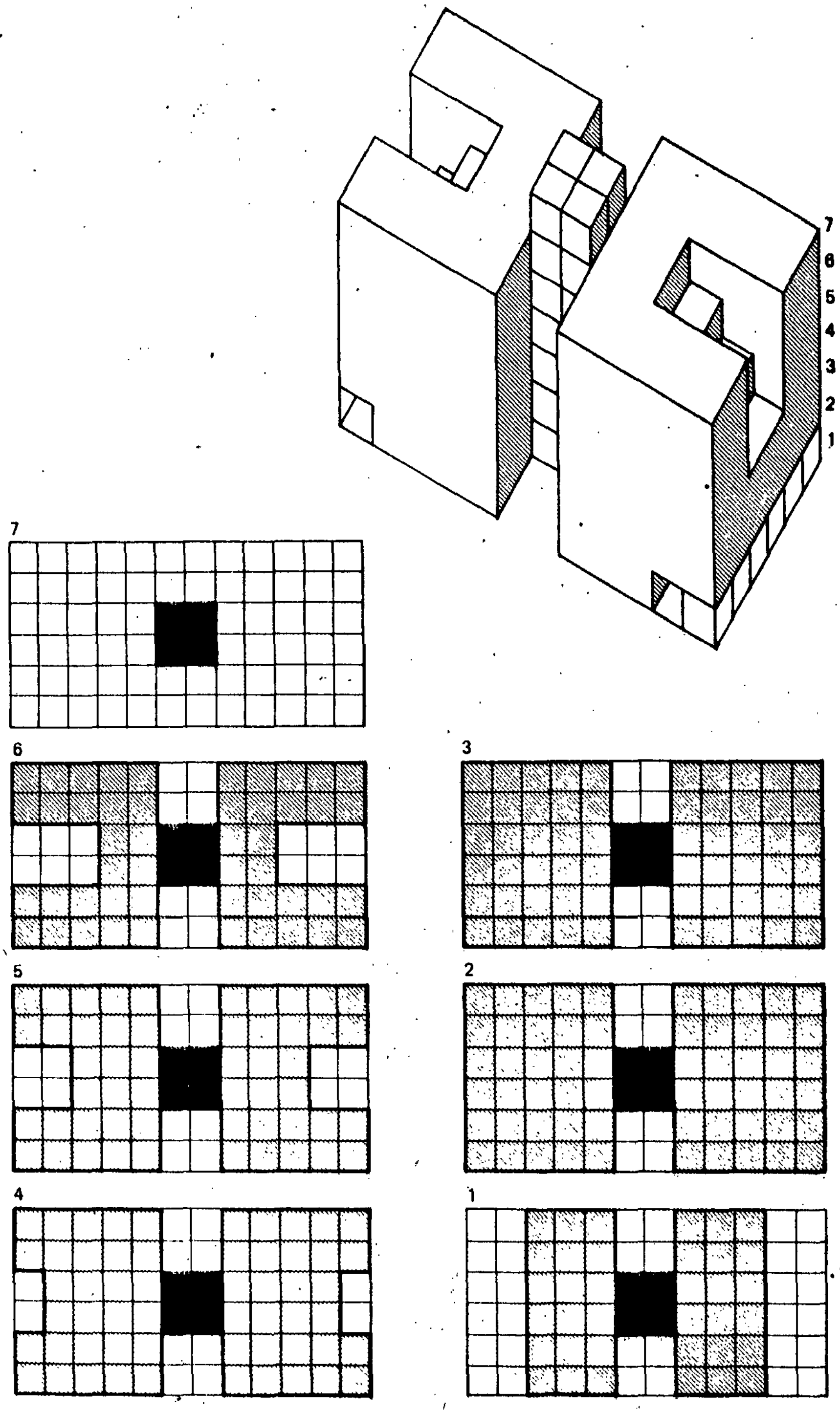


Figure 1-8
Representation of a built form as an assemblage of cubic modules.

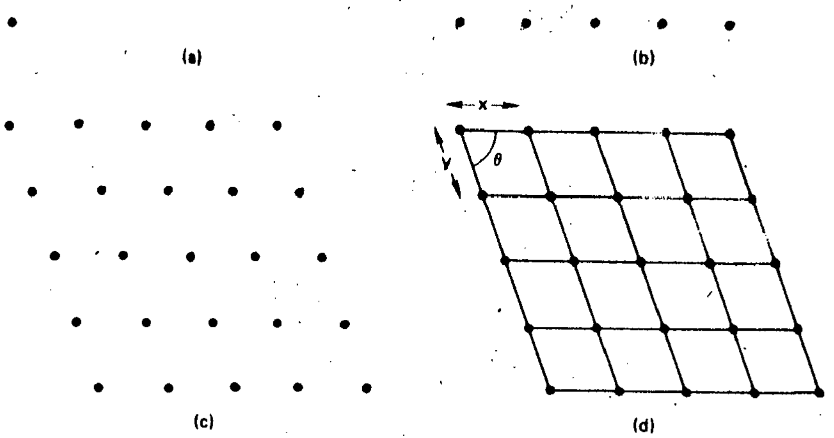


Figure 1-9

Generation of a parallelogram system of equivalent points. The particular system is described by the side lengths x and y and the angle θ. (a) Point. (b) Point translated along one axis to generate a row of equally spaced points. (c) Row translated along another axis to generate equally spaced rows. (d) Points connected by straight lines drawn parallel to the translation axes.

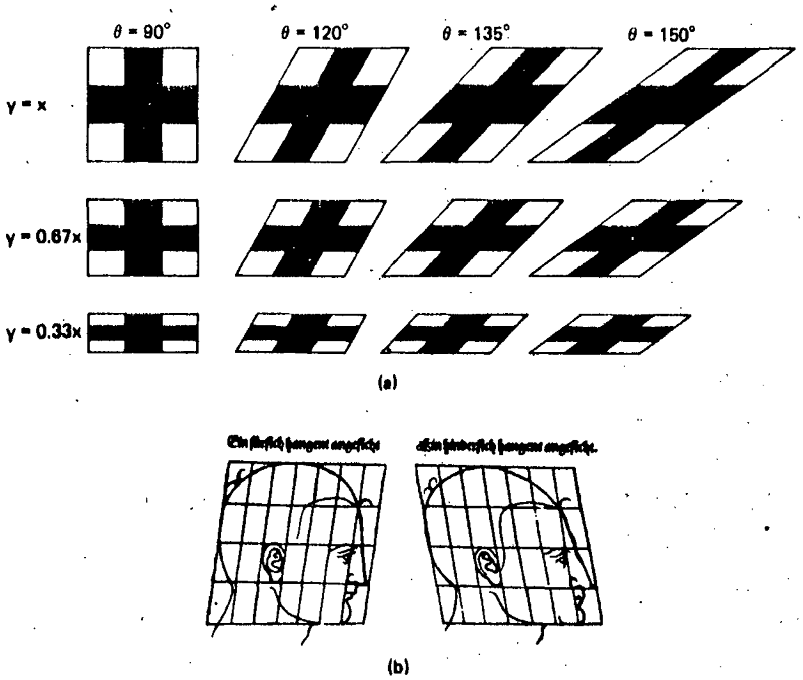


Figure 1-10

Transformation of forms by varying the parameters of parallelogram systems. (a) Tableau of plan forms produced by varying the parameters of y and θ. (b) Dürer's method of transforming faces by varying the parameters of a parallelogram system (illustration from the *Four Books of Human Proportions*).

employed it to generate caricatures. (Figure 1.10 (b)) It can be shown that there are only five distinct types of parallelogram systems of equivalent points. Each particular cell in any of these systems may be uniquely identified by means of two or three coordinates.

A method for description of any floor plan constructed within a parallelogram system can now be shown. The steps are as follows :

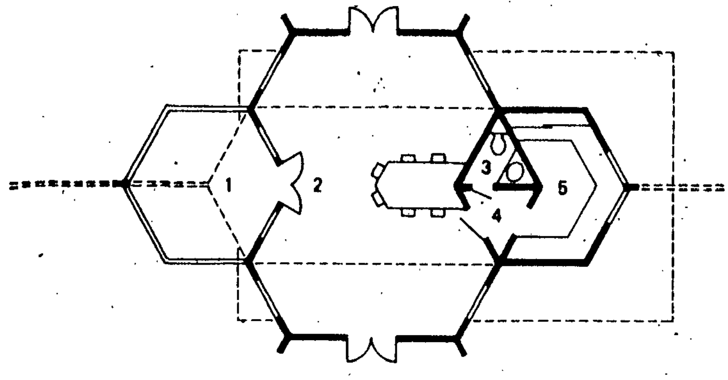
1. Describe the parallelogram system by means of the parameters x, y and θ, and assign coordinates to the cells.
2. Assign integers to cells to represent different areas, as with the square grid representations discussed previously.
3. Store these integers in the corresponding locations of a two-or three dimensional array.

An example is illustrated in Figure 1.11.

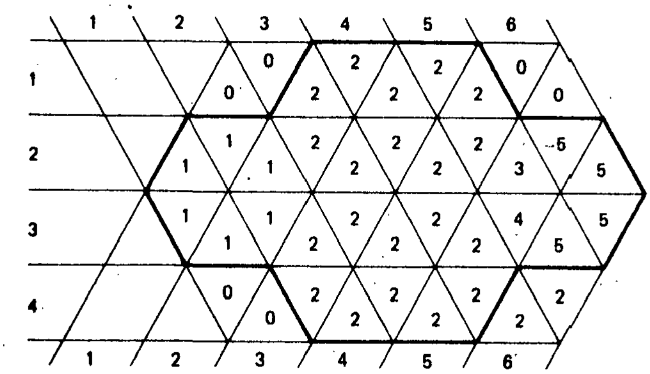
Further network of equal figures may be obtained by means of additional operation (Figure 1.12).

1.3.5 Complex shapes formed by combination of equal figures :

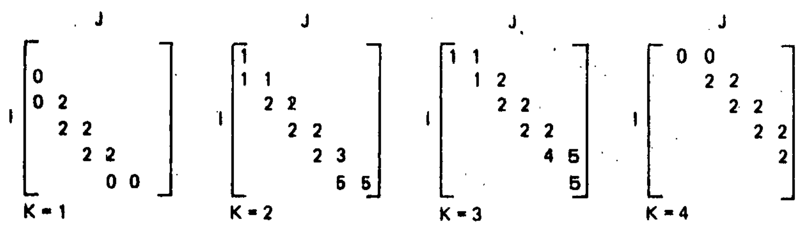
In the types of representations which we have been discussing, a complex shape is always represented as an aggregation of adjacent equal polygons or polyhedra. For example, a room in a floor plan might be described as an aggregation of square.



(a)



(b)



(c)

Figure 1-11

Encoding and storage of a plan which is based upon a triangular grid. (a) Project for a houseboat by Frank Lloyd Wright. (b) Representation of the plan by integers in a triangular grid. (c) Storage of the representation in a three-dimensional array.

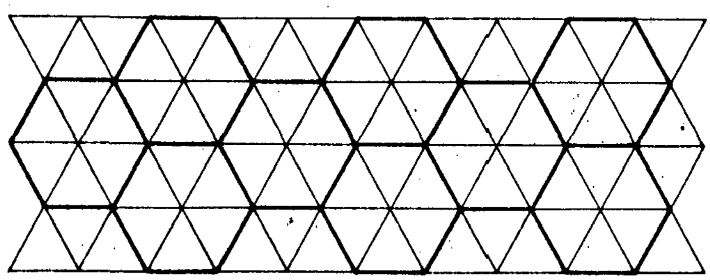


Figure 1-12.

Generation of a hexagonal network from a triangular network.

From an architectural point of view, one of the most interesting families oc complex shapes formed from equal elements is the family of polyominoes. A polyomino is an edge-connected assemblage of squares. All the distinct polyominoes of up to six squares are shown in Figure 1.13.

By generalization of the concept of a polyomino, polyiamonds are edge-connected assemblages of equilateral triangles, polyhexes are edge-connected assemblages of regular hexagons, polycubes are edge-connected assemblages of cubes, and so on.

1.4 VARIABLY DIMENSIONED GRIDS AND LATTICES

1.4.1 Dimensionless representation of rectilinear shapes

Consider the three rectilinear objects shown in Figure.1.14 Although their dimensions and proportions are quite different, they clearly have some shape properties in common: they are all 'U-shaped' objects. The observation that dimensionally very different objects can be discerned to have common shape properties suggests that it may be possible to develop descriptive techniques in which shape and dimensions are specified separately. This can be accomplished by use of a dimensionless representation in conjunction with dimensioning vectors.

The concept of a dimensionless representation of a rectilinear shape is perhaps best explained by the following example. Consider again the series of apparently similar forms shown in figure. The reason for this apparent similarity becomes obvious if we surround each form by a rectangular

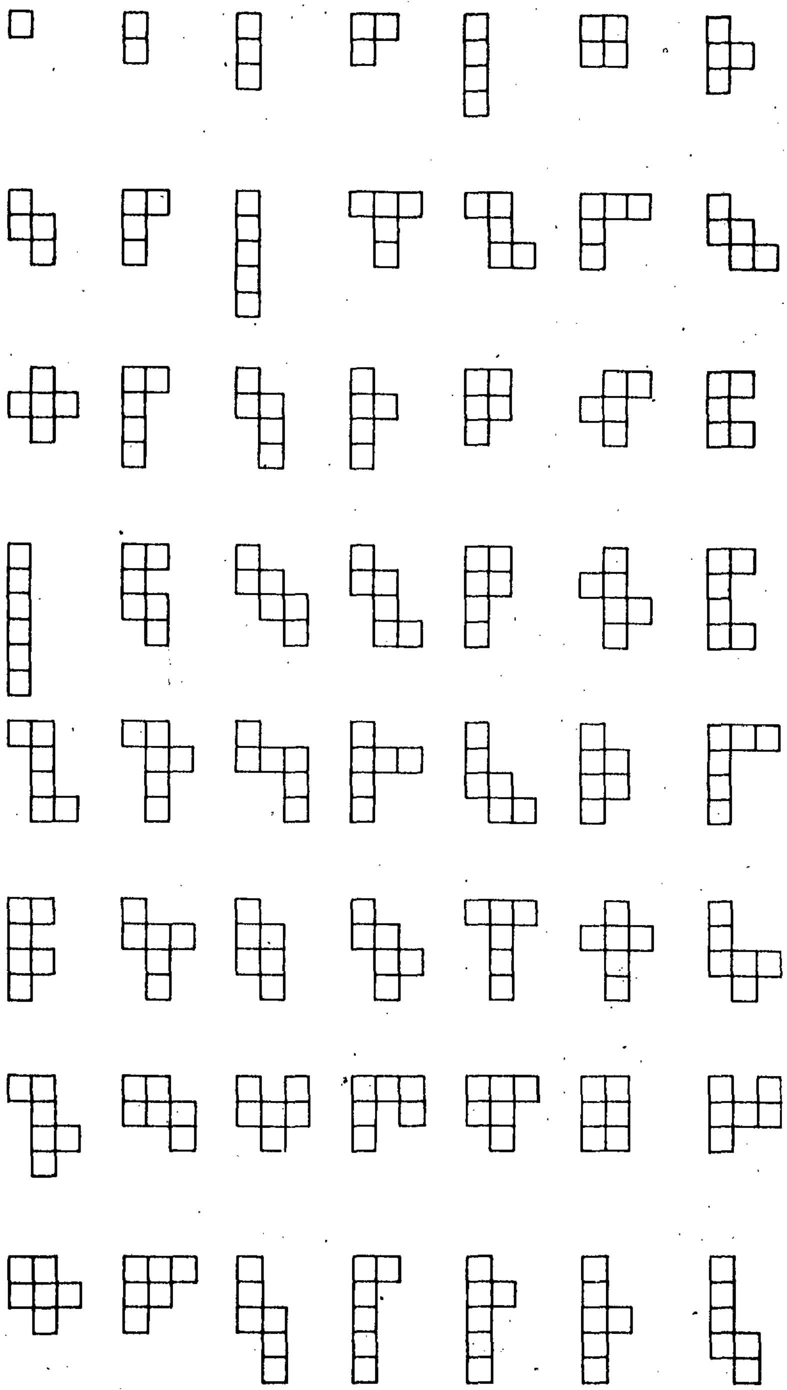
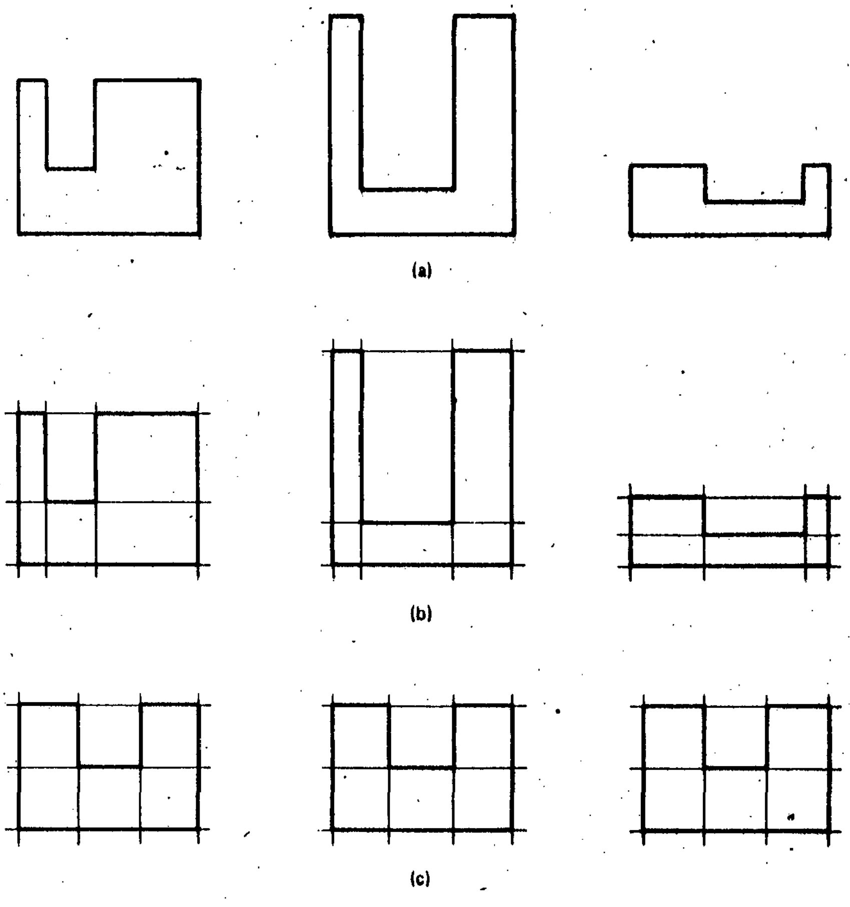


Figure 1-13
The 56 polyominoes composed of 6 or fewer squares.

Figure 1-14
Dimensionless representations of similar rectilinear shapes. (a) Several U-shaped objects. (b) Imposition of minimum rectangular gratings. (c) Transformation to dimensionless representations.



frame, and superimpose a minimum rectangular grating. This minimum rectangular grating is formed by drawing horizontal and vertical lines across the frame in such a way that all vertices of the original form lie on an intersection, and each grating line intersects at least one vertex. It can be seen that the minimum grating for each form, in the example shown, consists of six rectangular cells. If we now adjust the dimensions of the minimum gratings so that each cell in the grating becomes square, as shown in figure, we find that each of the differently-dimensioned shapes reduces to the same U-shaped figure. Such a representation of a form within a uniform square grating may be termed its dimensionless representation.

The dimensionless representation of a rectilinear shape can be numerically encoded as a two-dimensional array of 1's and 0's. March (1976) has suggested that this array can then be unfolded to form a one-dimensional string of 1's and 0's, in other words a binary number. This binary number can then be re-expressed in octal, decimal or hexadecimal form. The encoding of the floor plan of one of Le Corbusier's Maison Minimum, using this scheme, is illustrated.

1.4.2 Dimensioning vectors

The dimensional properties of a particular rectilinear shape can be specified by defining X and y dimensioning vectors. (Figure 1.15). The elements of the X dimensioning vector describe the widths of the columns, and the elements of the Y

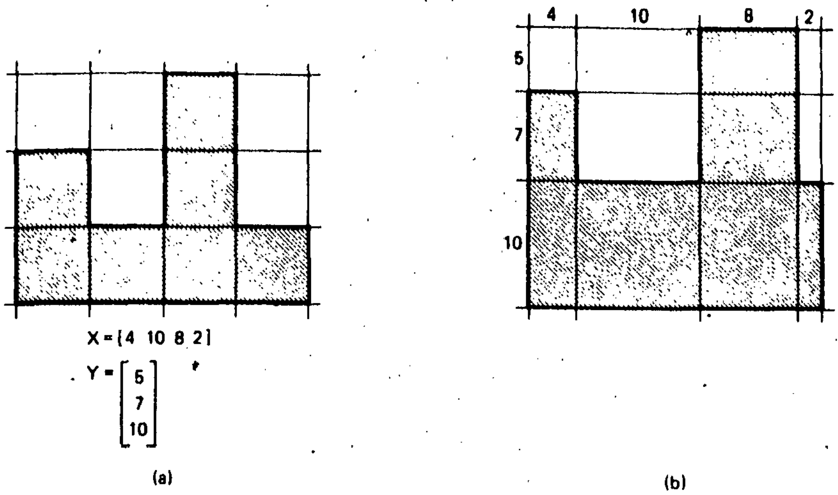


Figure 1-15
Description of a shape by a dimensionless representation plus dimensioning vectors. (a) Dimensionless representation plus dimensioning vectors. (b) Corresponding shape drawn to scale.

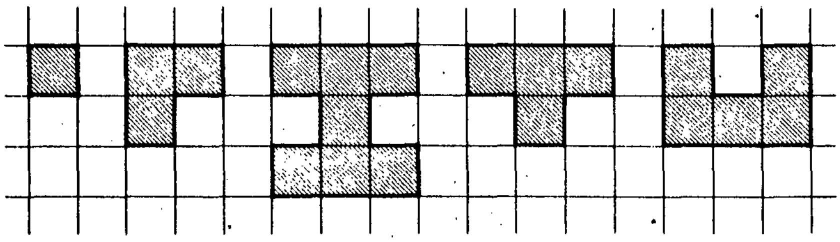


Figure 1-16
Dimensionless representations of standard structural sections.

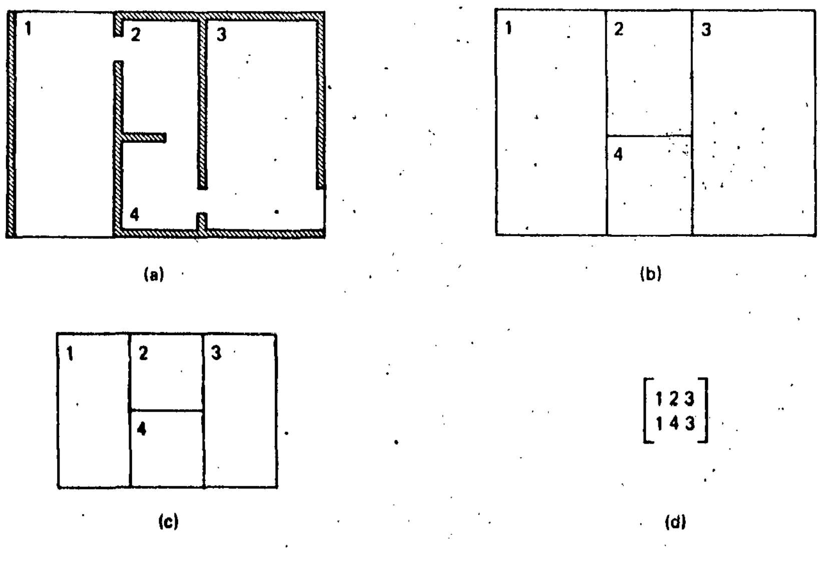


Figure 1-17
Use of an integer array to store a dimensionless representation of a floor plan. (a) Initial floor plan. (b) Outline floor plan. (c) Dimensionless representation of outline floor plan. (d) Integer array encoding dimensionless representation of outline floor plan.

dimensioning vector describe the widths of the rows. Given a dimensionless representation and dimensioning vectors, the dimensioned shape can be constructed. Applying different dimensioning vectors to a dimensionless representation will produce a family of shapes in which the constituent rectangles have different dimensions and areas, but the adjacency relations between these rectangles remain constant. Well-known examples of such families of shapes are the standard structural sections; rectangles, angles, I-sections, T-sections, and channels. (Figure 1.16)

1.4.3 Rectilinear floor plans in dimensionless form

It is straightforward to extend the technique of dimensionless representation for use in description of floor plans. Instead of just using 1's and 0's to represent solids and voids, as in the examples discussed so far, we may use different integers to distinguish different rooms in a lay-out, as illustrated. (Figure 1.17). This approach has two obvious advantages over the simple square grid representation which was introduced earlier: it can gain considerable storage economics, and it allows dimensions to vary continuously rather than in fixed modular increments. Of course, a price of increased complexity of manipulation operations must be paid.

A third, less obvious advantage of this method of representation is that the separation of shape-description from dimensional information allows the exhaustive enumeration of generic solutions to floor plan layout problems. Consider for example the class of floor plans generated by dissection of a rectangle into rectangles. For dissection into any given number of rectangles, the number of distinct dissections is limited, and there exists a simple algorithm for exhaustively enumerating these distinct dissections. All the distinct dissections into upto six rectangles are illustrated, in dimensionless representation (Figure 1.18). Ranges of specific floor plans can be produced by mapping activities into the cells of these dissections and applying dimensioning vectors.

1.4.4 Three-dimensional rectilinear forms :

The technique of form description by means of dimensionless representation of shape plus dimensioning vectors may be extended to three-dimensional rectilinear forms. A three-dimensional minimum grating is employed, together with X,Y and Z dimensioning vectors. Several examples of dimensionless representations of buildings are illustrated in Figure 1.19 and the method for numerically encoding dimensionless representations of three-dimensional rectilinear forms is demonstrated in Figure 1.20.

1.4.5 Non-rectilinear forms :

Non-rectilinear polygons and floor plans may also be reduced to dimensionless representations (Figure 1.21). Note, however, that in the non-rectilinear case a number of different orientations of the frame with respect to the plan are possible and that each different orientation will produce a different

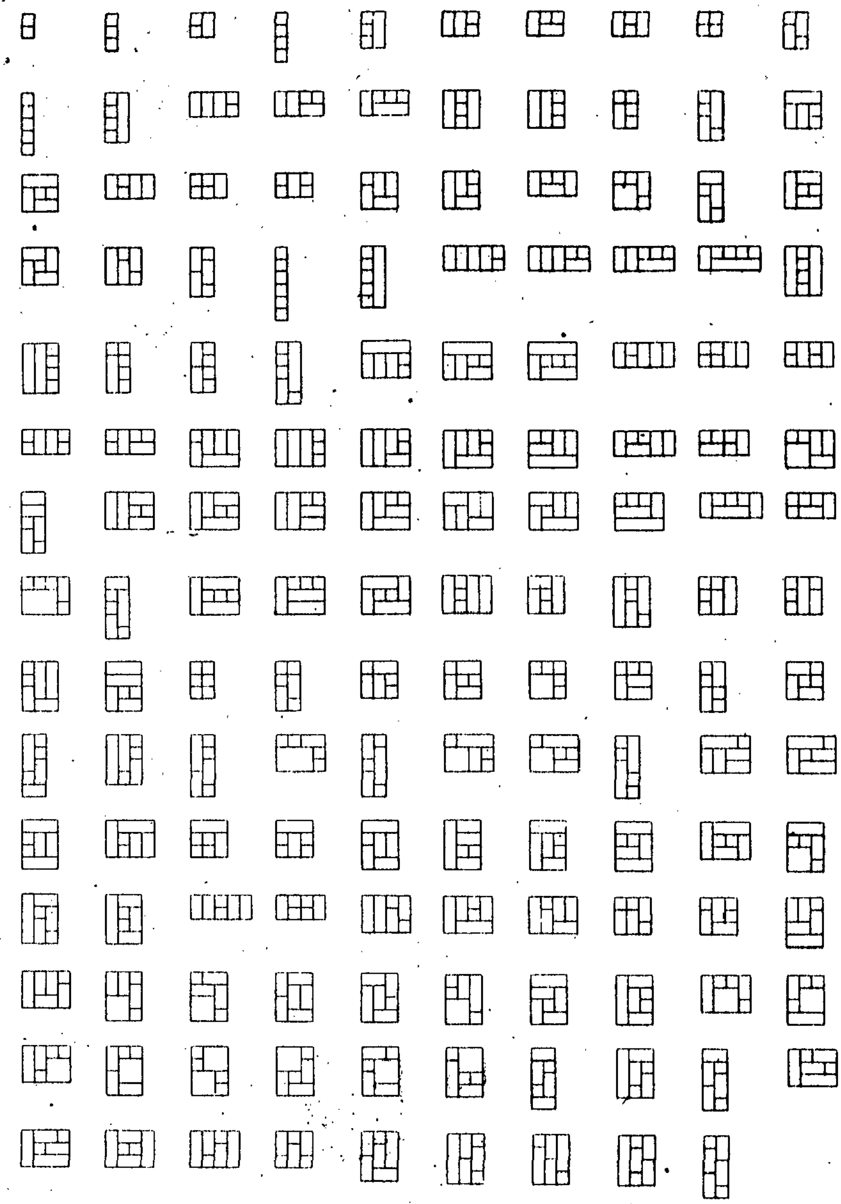


Figure 1-18

Dimensionless representations of all the dissections of a rectangle into six or fewer rectangles (from Mitchell, Steadman and Liggett, 1976).

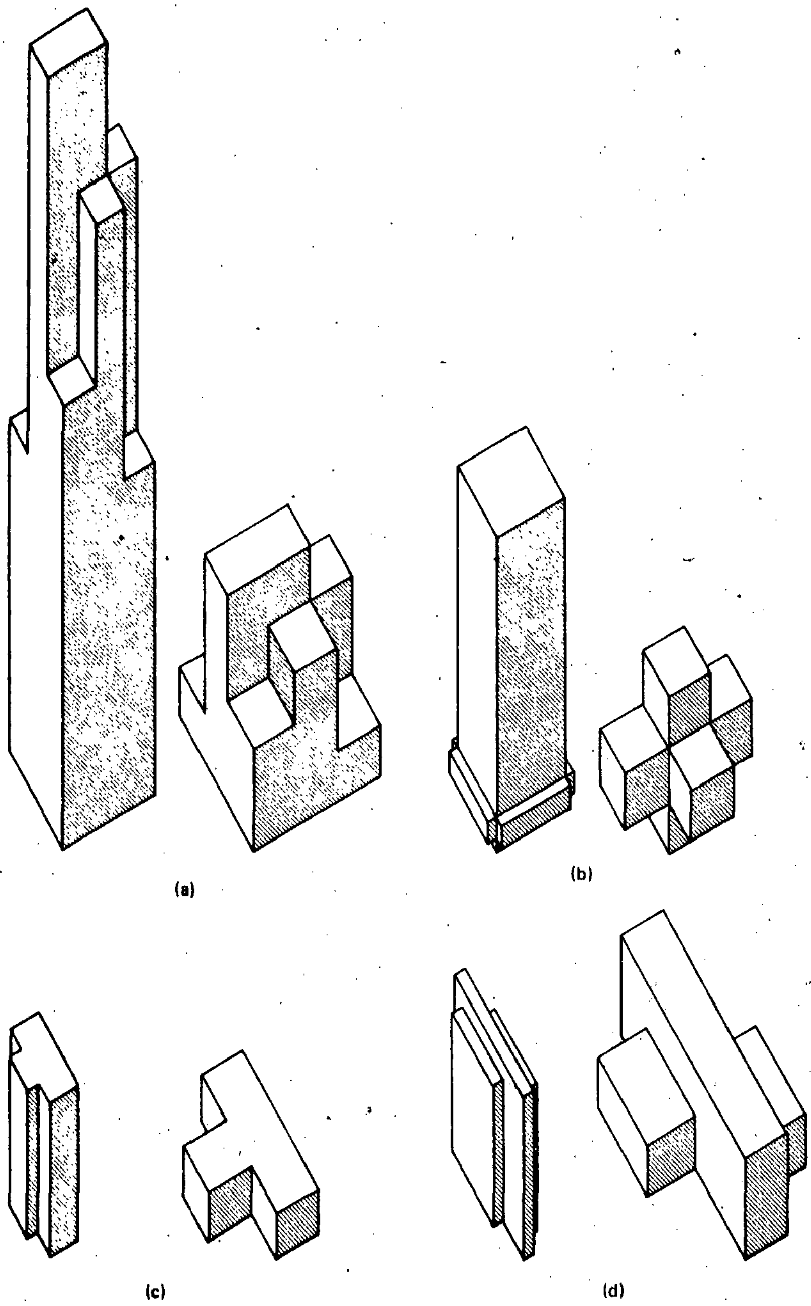


Figure 1-19
Examples of high rise office buildings and their dimensionless representations. (a) Sears Tower, Chicago (Skidmore, Owings, and Merrill). (b) Place Victoria, Montreal (Luigi Moretti and Pier Luigi Nervi). (c) One Charles Center, Baltimore (Ludwig Mies van der Rohe). (d) Thyssen-Rohrenwerke office, Dusseldorf (Hentrich and Perschnigg).

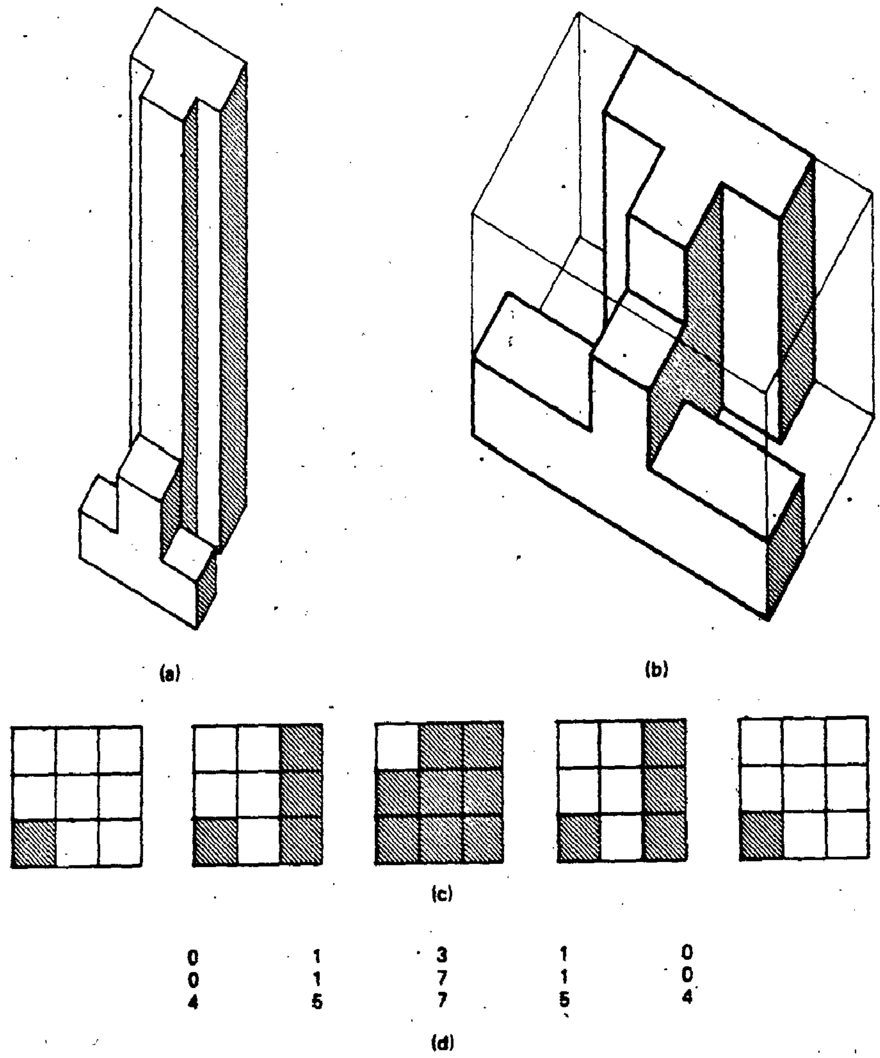


Figure 1-20
Numerically encoding the dimensionless representation of the Seagram Building, New York (after March 1976). (a) Scale drawing. (b) Dimensionless representation. (c) Representation of sections through the dimensionless representation as bit patterns. (d) Encoding of each row of each section as an octal digit.

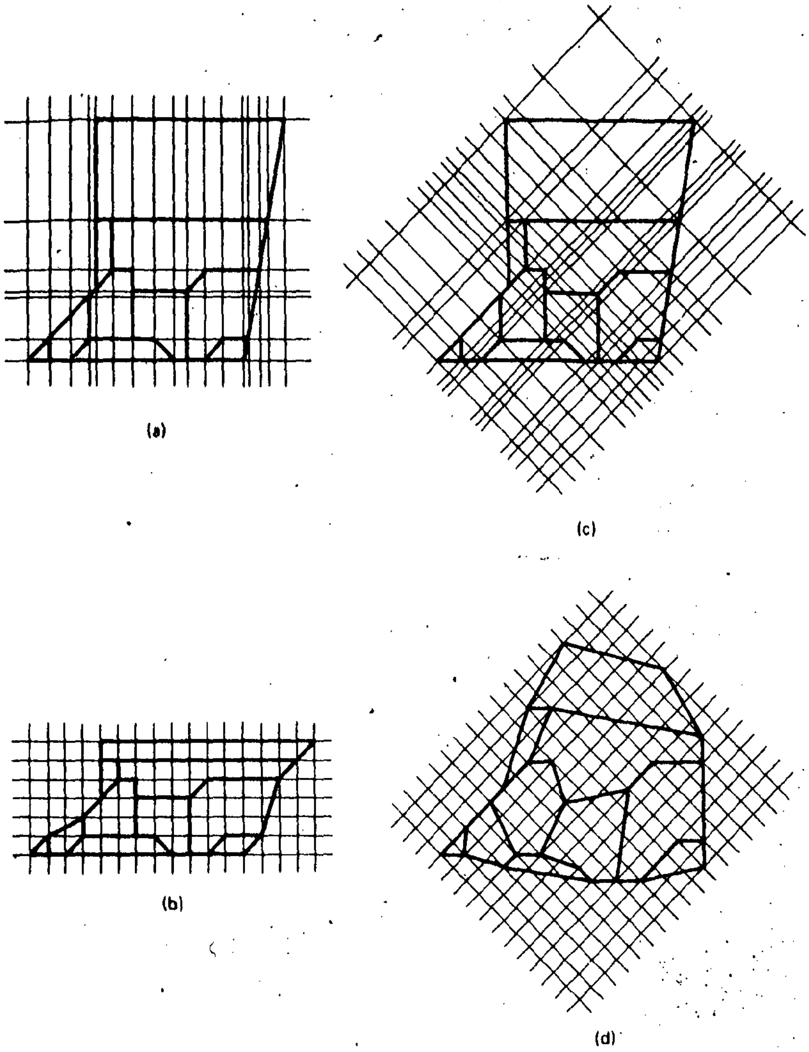


Figure 1-21

Dimensionless representation of a non-rectilinear plan (North Penn Visiting Nurse Association Headquarters, architects, Venturi and Short, 1960). (a) Grating aligned with north, south, and west walls. (b) Resultant dimensionless representation in a 6 X 16 grid. (c) Grating rotated 45° to align with northwest wall. (d) Resultant dimensionless representation in a 14 X 16 grid.

dimensionless representation. A further difference from the rectilinear case is that vertex angles do not necessarily remain constant as the dimensions of the grating are adjusted. Since walls need not necessarily lie along grating lines, the simple integer array form of representation cannot be employed.

1.5 POLYGON AND POLYHEDRON REPRESENTATIONS :

The representational techniques discussed so far have all been based on explicit representation of spatial domains (by integers representing spatial elements). This approach differs from that followed in traditional architectural drawings, where spatial domains are implicitly represented by drawing edge-lines which define the boundaries of domains. Systems of representation based upon storing edge-lines of forms and spaces may also be utilized for describing buildings in computer memory. These may be classed as polygon and polyhedron representations.

1.5.1 Points, Lines, Polygons, and Polyhedra :

Points are the basic building blocks of polygon and polyhedron representations. A point in space, relative to some given coordinate systc, can be represented by its coordinates (X,Y) in two-dimensional space or (X,Y,Z) in three-dimensional space. For reasons which will become clear as we proceed, it is conveient to think of these coordinate pairs or triples as row vectors (X,Y) or (X,Y,Z).

Since a straight line is defined by its end points, [Figure 1.22(a)], it can be represented as a 2 x 2 matrix composed of the two coordinate pairs which specify the end points :

$$\begin{matrix} x_1 & y_1 \\ x_2 & y_2 \end{matrix}$$

Lines composed of n straight segments can be represented by (n+1) points, [Figure 1.22(b)]. Similarly, curved lines can be represented to any desired degree of accuracy by approximation with sh t straight segments, then encoding in this format.

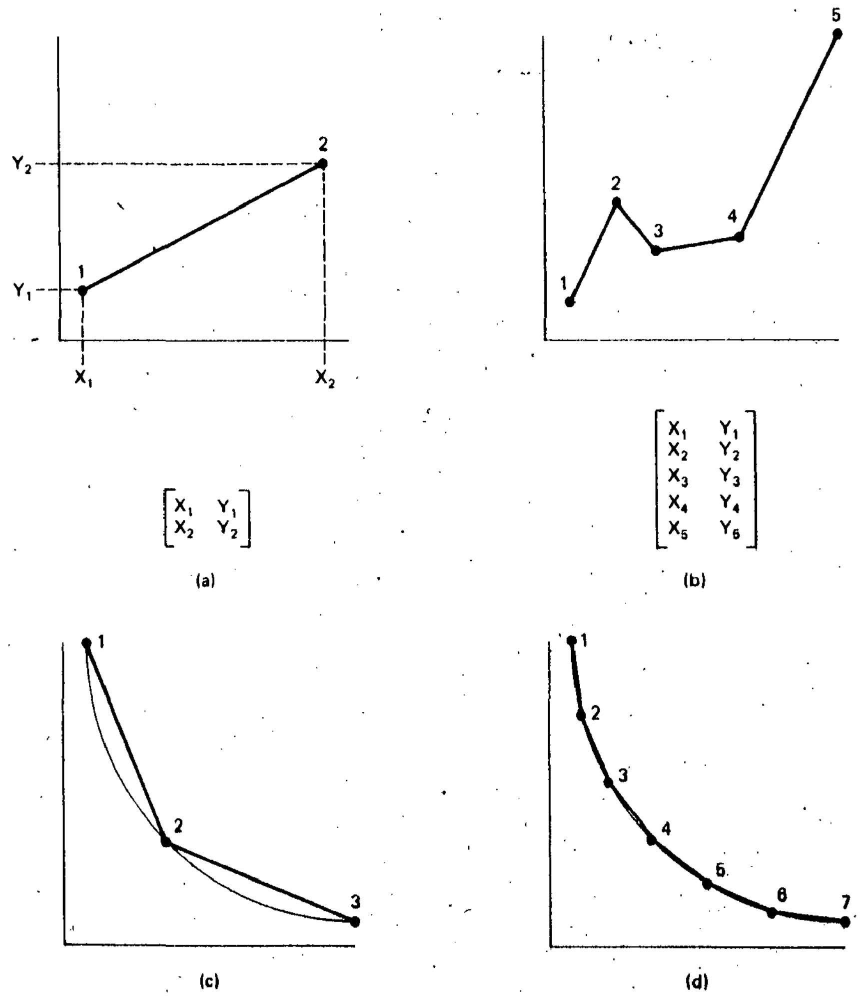
Polygons are defined by lines which begin and end at the same point. Thus polygons (and closed curves) can also be represented in this format, as shown in Figure 1.23.

The faces of polyhedra are polygons. A polyhedron can thus be represented by several point-matrices, each of which represents one of the faces, (Figure 1.24). Some redundancy results, however. Each line is represented twice, since it divides two faces, and each point is represented at least twice.

Point-matrices can readily be stored in a computer by means of arrays or lists. This type of representation is very widely employed in computer graphics systems, since arrays or

Figure 1-22

Representation of lines. (a) Representation of a straight line by the coordinates of its end points. (b) Representation of a line composed of straight segments by point-coordinates. (c) Approximation of a curved line by straight segments. (d) More accurate approximation by more segments.



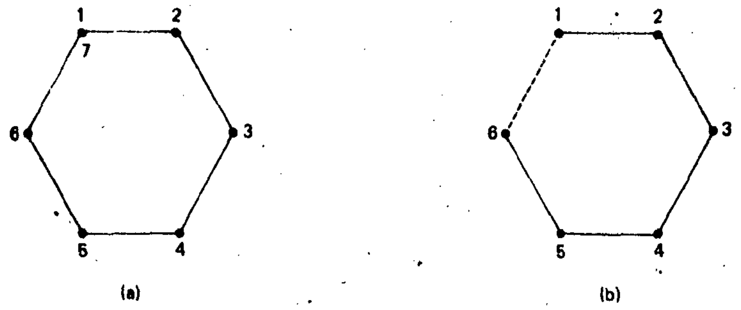


Figure 1-23

Representation of polygons. (a) Explicit closure. (b) Implicit closure.

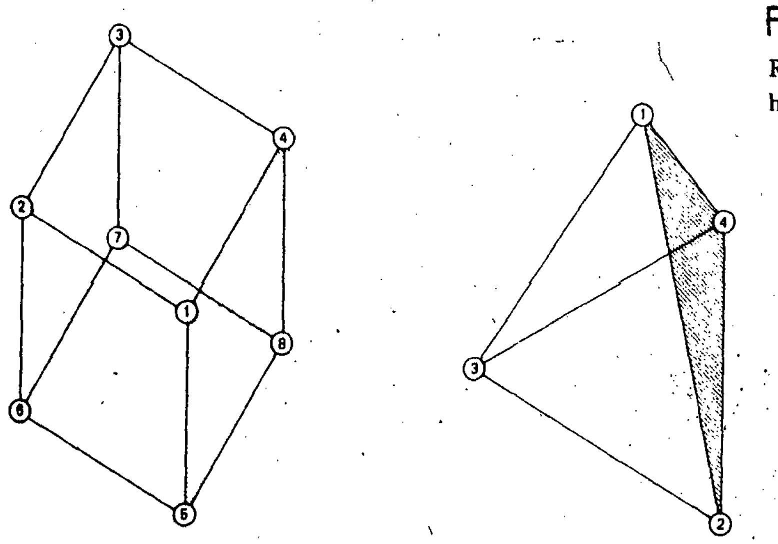


Figure 1-24

Representation of polyhedra.

lists storing point-matrices are readily converted into instructions to a graphics output device to generate a picture.

1.5.2 Sparse Matrix Techniques :

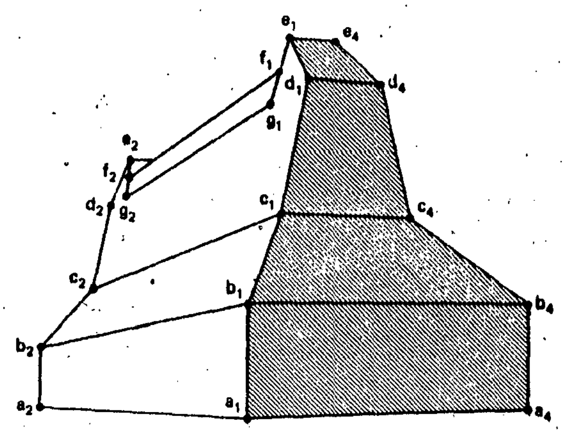
A disadvantage with this point matrix/adjacency matrix technique is that it consumes excessive storage when the figure is large, (Figure 1.25). For a figure composed of n points, an adjacency matrix of n^2 cells is required. Since the matrix is symmetrical, some economies can be achieved by storing only the entries on one side of the diagonal. But the number of cells still grows as $(n^2 - n)/2$. Much greater economies can be achieved by recognizing that the matrix will characteristically very sparse, i.e. it will consist mostly of 0's. The efficient storage of large sparse matrices is a common problem in computing, and numerous special techniques have been developed for this purpose.

Very great compression can be achieved by storing only subscript pairs identifying the non-zero entries. For examples the binary matrix

$$\begin{matrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{matrix}$$

can be described by the following list of subscript pairs :

(1,3), (2,5), (3,1), (5,2).



(a)

	x	y	z
a_1	−7.0	−25.0	0.0
a_2	−7.0	25.0	0.0
a_3	7.0	25.0	0.0
a_4	7.0	−25.0	0.0
b_1	−7.0	−25.0	5.0
b_2	−7.0	25.0	5.0
b_3	7.0	25.0	5.0
b_4	7.0	−25.0	5.0
c_1	−3.5	−20.0	8.75
c_2	−3.5	20.0	8.75
c_3	3.5	20.0	8.75
c_4	3.5	−20.0	8.75
d_1	−1.75	−20.0	15.0
d_2	−1.75	20.0	15.0
d_3	1.75	20.0	15.0
d_4	1.75	−20.0	15.0
e_1	−1.25	−15.0	17.5
e_2	−1.25	15.0	17.5
e_3	1.25	15.0	17.5
e_4	1.25	−15.0	17.5
f_1	−1.5	−14.0	16.25
f_2	−1.5	14.0	16.25
f_3	1.5	14.0	16.25
f_4	1.5	−14.0	16.25
g_1	−1.75	−13.0	15.0
g_2	−1.75	13.0	15.0
g_3	1.75	13.0	15.0
g_4	1.75	−13.0	15.0

(b)

	a_1	a_2	a_3	a_4	b_1	b_2	b_3	b_4	c_1	c_2	c_3	c_4	d_1	d_2	d_3	d_4	e_1	e_2	e_3	e_4	f_1	f_2	f_3	f_4	g_1	g_2	g_3	g_4
a_1		1		1	1																							
a_2	1		1			1																						
a_3		1		1			1																					
a_4	1		1					1																				
b_1	1					1		1	1																			
b_2		1			1		1			1																		
b_3			1			1		1			1																	
b_4				1	1		1					1																
c_1					1					1		1	1															
c_2						1			1		1			1														
c_3							1			1		1			1													
c_4								1	1		1					1												
d_1									1							1	1											
d_2										1					1			1										
d_3											1			1					1									
d_4												1	1							1								
e_1													1							1	1							
e_2														1					1			1						
e_3															1			1					1					
e_4																1	1							1				
f_1																	1					1		1	1			
f_2																		1			1		1			1		
f_3																			1			1		1			1	
f_4																				1	1		1					1
g_1																					1					1		
g_2																						1			1			
g_3																							1					1
g_4																								1			1	

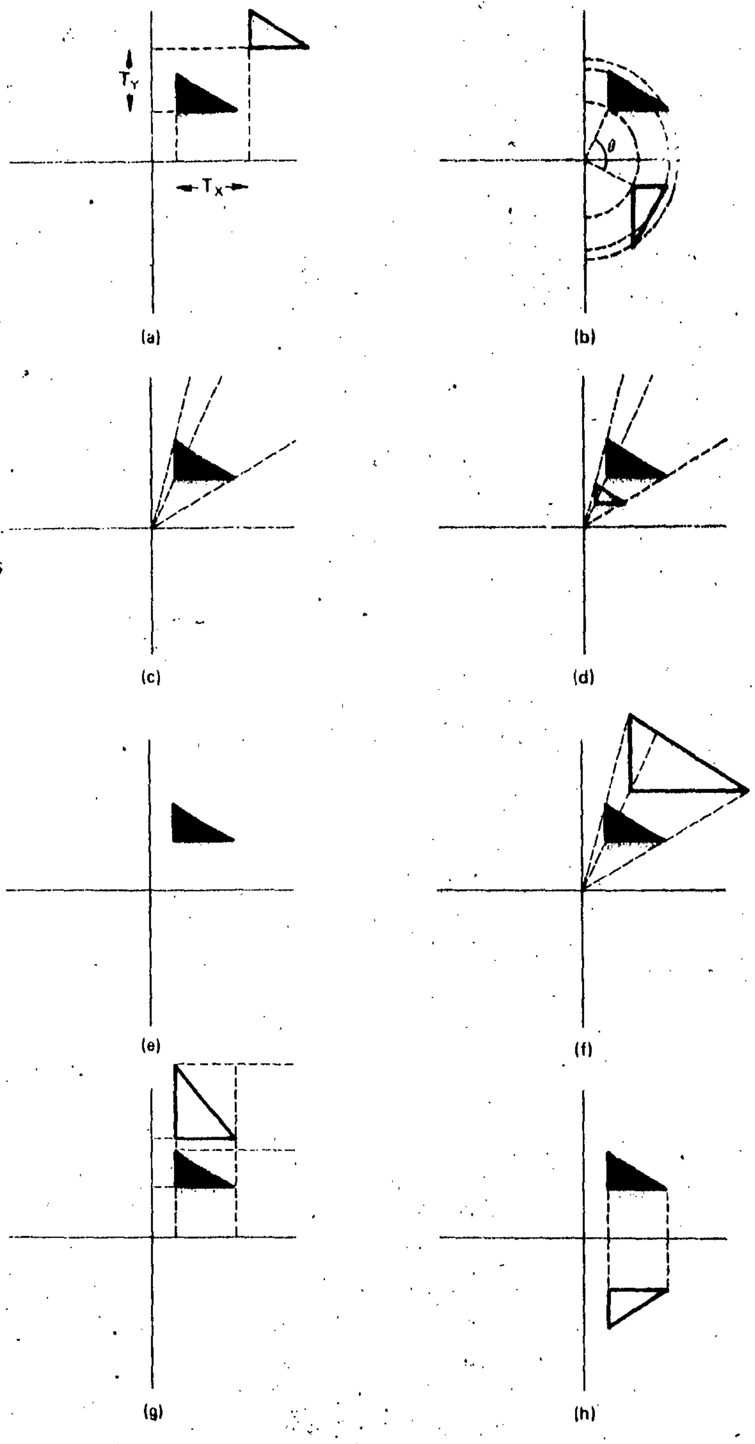
(c)

Figure 1-25

Description of forms by vertex list and adjacency matrix. (a) Erich Mendelsohn's Luckenwalde Hat Factory (dye works), 1921–23. (b) Vertex list. (c) Adjacency matrix (zero entries omitted for clarity).

Figure 1-26

Some common two-dimensional geometric transformations. (a) Translation. (b) Rotation about the origin. (c) Scaling $S_x = S_y = 0$. (d) Scaling $S_x = S_y$ $0 < S_x < 1$. (e) Scaling $S_x = S_y = 1$. (f) Scaling $S_x = S_y$ $S_x > 1$. (g) Scaling $S_x = 1$ $S_y > 1$. (h) Reflection across X axis. (i) Reflection across Y axis. (j) Reflection across X and Y axis.



An alternative to a simple linearly-linked list for this purpose is a ring structure, in which each non-zero entry points to both the next non-zero entry to the right, and the next non-zero entry below, and the last non-zero entry in a row or column points back to the first. Alternative representations of a floor plan, using linearly-linked and ring-structures to store the adjacency matrix, are illustrated, The ring structure is more complex to implement than the simple list, but it saves on the time taken to access an entry when the matrix is large.

1.5.3 Hierarchical Representations :

In the floor plan representation just described, wall segments and corners are the only elements that are explicitly identified. For many applications, it is necessary to also explicitly identify rooms, and to associate data (e.g., name, floor area, use, internal temperature, etc.) with rooms. This can be accomplished by use of three cross-linked lists :-

A room-list, containing data describing rooms, each element of which is cross-linked to three or more walls.

A wall-list, containing data describing wall segments, each element of which is cross-linked to two points.

A point-list, containing coordinates of vertices.

In effect, the floor plan is now represented as a hierarchy of spatial elements of different dimensionality. At the highest level are two dimensional polygons. Each polygon is bounded by one-dimensional lines, and each line is bounded by two zero-dimensional points. Data can be associated with different levels of the hierarchy as required : details of corners at level 0, properties of wall-segments at level 1, and properties of rooms at level 2. The associations of elements of lower and higher dimensionality are recorded by the cross-linkages.

1.6 DUAL-GRAPH REPRESENTATIONS :

Where representations are regarded as graphs, it becomes possible to exploit some results from graph theory in order to develop further descriptive techniques which may be useful for particular purposes.

1.6.1 Concept of a dual-graph :

Certain graphs possess the property of planarity. A graph G is said to be planar if there exists some geometric realization of G which can be drawn on a plane such that no two of its edges intersect. A graph that cannot be drawn on a plane without crossover between its edges is called non-planar, (Figure 1.27).

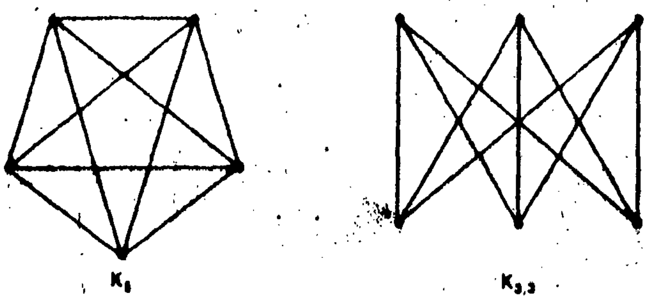


Figure 1-27
Kuratowski's nonplanar graphs.

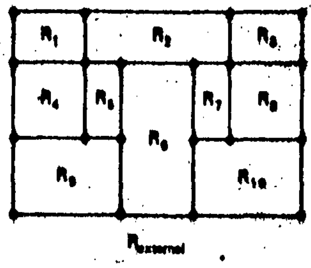


Figure 1-28
Floor plan represented as a planar graph, in which rooms correspond to internal regions, and the external region is of infinite extent.

Figure 1-29
The dual of a planar floor plan. (a) Dual of the floor plan shown in Figure 6.43. (b) External region divided into "north," "east," "south," and "west" regions by insertion of dummy "walls." (c) External region ignored.

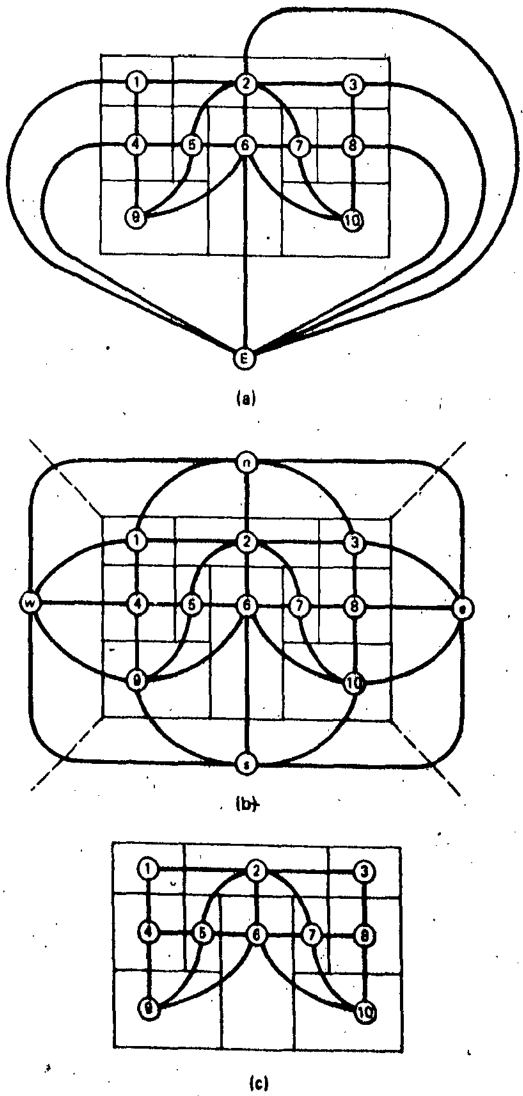
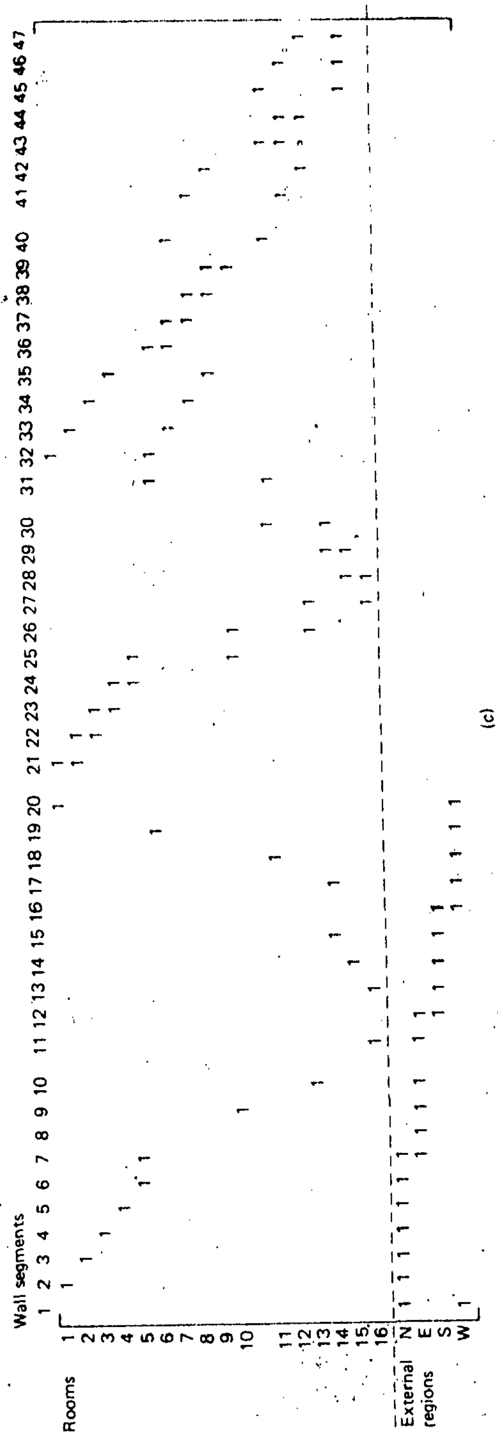
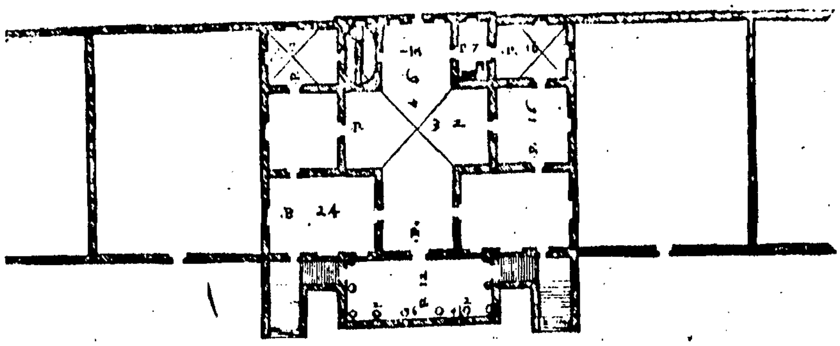


Figure 1-30

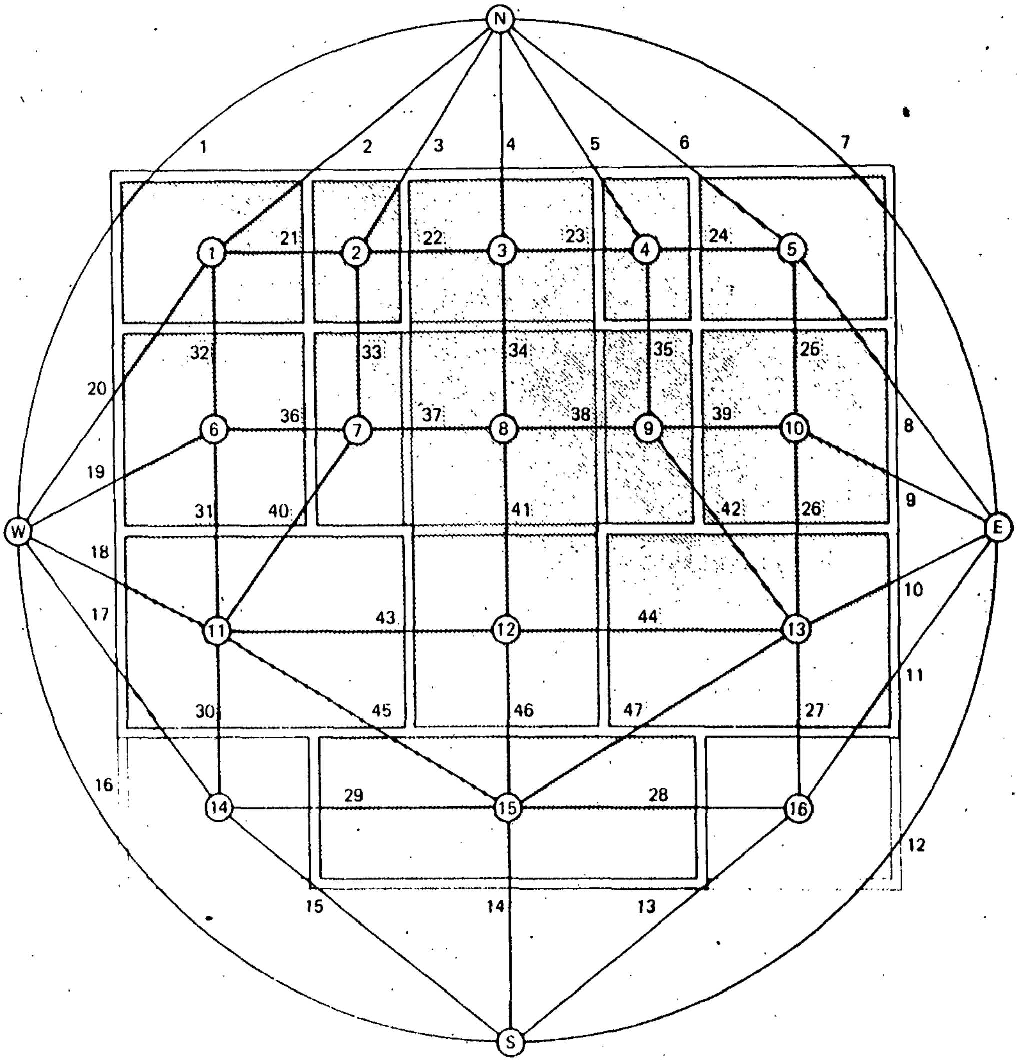
Representation of a floor plan by incidence matrix of dual graph plus room and wall segment data tables. (a) Plan of Palladio's Villa Malcontenta, 1560 (illustration from the *Quattro Libri*). (b) Schematized plan represented as a dual graph. Note the insertion of "dummy" wall segments in order to make all spaces rectangular. (c) Incidence matrix of a dual graph (zero entries excluded for clarity). (d) Room data table. (e) Wall segment data table.





(a)

Figure 1-30



(b)

Figure 1-30

Room	Centroid coordinates x	y	Area
1	4	26	48
2	10	26	24
3	16	25	48
4	22	26	24
5	28	25	48
6	4	18	64
7	10	18	32
8	16	18	64
9	22	18	32
10	28	18	64
11	4	10	96
12	16	10	64
13	28	10	96
14	4	3	48
15	16	3	96
16	28	3	48

(d)

Segment	Length
1	Dummy
2	8
3	4
4	8
5	4
6	8
7	Dummy
8	6
9	8
10	8
11	6
12	Dummy
13	8
14	16
15	8

(e)

Figure 1-30

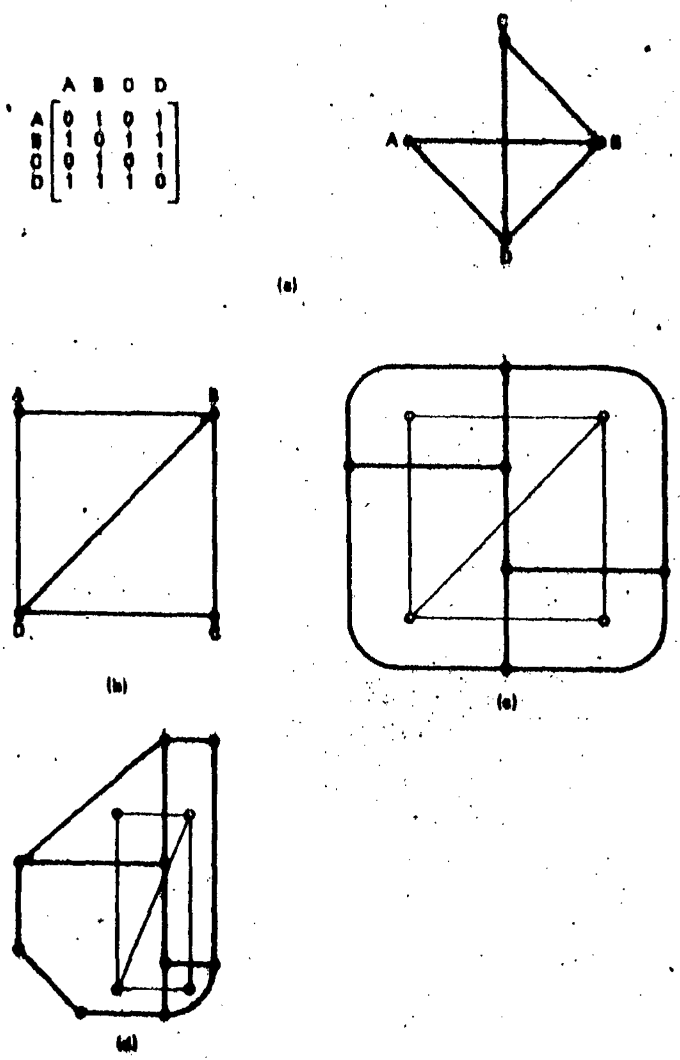


Figure 1-31
Procedure for constructing a floor plan corresponding to a specified adjacency requirements matrix. (a) Adjacency requirements matrix and its corresponding graph. (b) Graph unfolded in the plane. (c) Dual constructed (i.e. wall segments drawn in). (d) Shapes and dimensions of rooms adjusted.

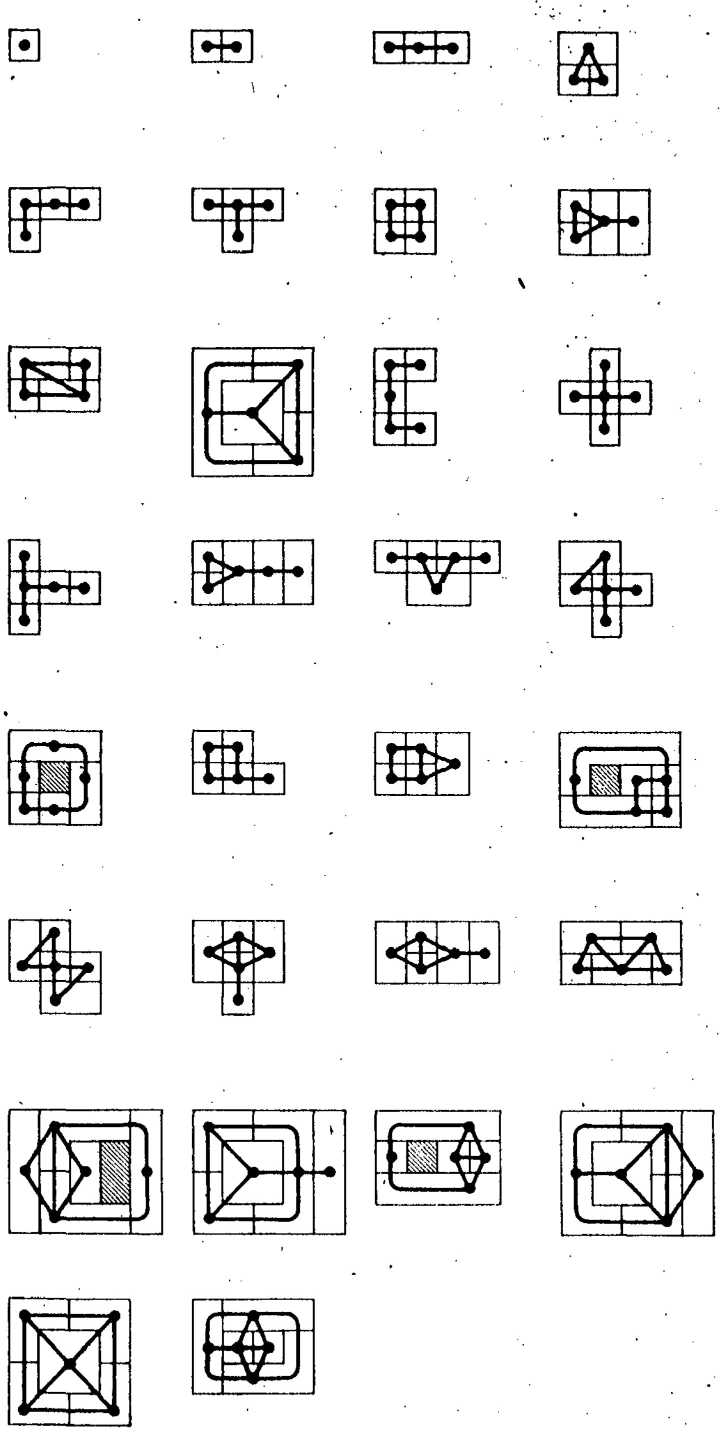


Figure 1-32
The simple connected planar graphs of five or fewer vertices, shown superimposed upon corresponding floor plans.

A geometric realization of a planar graph divides the plane into regions, (Figure 1.28). Each region is bounded by a specific set of edges. There exist both internal regions of finite area, and an external region of infinite extent. Thus we can now charactezie a single storey floor plan as a geometric realization of a planar graph, in which edges represent wall segments, vertices represent intersections of wall segments, internal regions represent rooms, and the external region represents the exterior.

For any single-level floor plan there can be constructed a planar dual by the following procedure. Within each region (room) locate a vertex, as shown in Figure 1.29 (a). Connect vertices in adjacent regions by an edge. The resultant figure is the dual of the original figure. Duals of floor plans are often referred to as room adjacency graphs because they represent the adjacency relations between rooms. It is often convenient to divide the external region into several different regions by the insertion of dummy walls of infinite length, as illustrated, [Figure 1.29(b)], so that orientations of rooms are indicated. Alternatively, the external region may be ignored altogether, [Figure 1.29(c)].

1.7 SMITH DIAGRAMS :

1.7.1 The electrical network analogy :

If a floor plan is known to consist of a dissection of a rectangle into rectangles, then an alternative though closely related graph theoretic representation known as a 'Smith Diagram' may be utilized. A Smith diagram may be thought of as an 'electrical network' (Figure 1.33). East-west wall is represented by a 'terminal'. The 'potential' at each 'terminal' represents the distance from the south wall, and the 'current' in each link represents the width of the space that it traverses. Similarly of course, a Smith diagram can be drawn from west to east.

The most important special property of the Smith diagram representation is that it facilitates checks for dimensional consistency and compliance with dimensional constraints in a floor plan, since it can be shown that Kirchoff's laws for electrical networks must hold. We know from Kirchoff's first law that the current in a conductor is given by

$$\text{Current} = C\,(V_1 - V_2)$$

where

C = Conductance

V_1 = larger voltage

V_2 = smaller voltage

Interpreting this in terms of a rectangular floor plan, we see that $(V_1 - V_2)$ represents the length of the room, and C represents the proportion (ratio of length to width). The second law simply states that current entering a terminal must equal closely related graph theoretic representation known as a 'Smith Diagram' may be utilized. A Smith diagram may be thought of as an 'electrical network' (Figure 1.33). East-west wall is represented by a 'terminal'. The 'potential' at each 'terminal' represents the distance from the south wall, and the 'current' in each link represents the width of the space that it traverses. Similarly of course, a Smith diagram can be drawn from west to east.

The most important special property of the Smith diagram representation is that it facilitates checks for dimensional consistency and compliance with dimensional constraints in a floor plan, since it can be shown that Kirchoff's laws for electrical networks must hold. We know from Kirchoff's first law that the current in a conductor is given by

$$\text{Current} = C\,(V_1 - V_2)$$

where

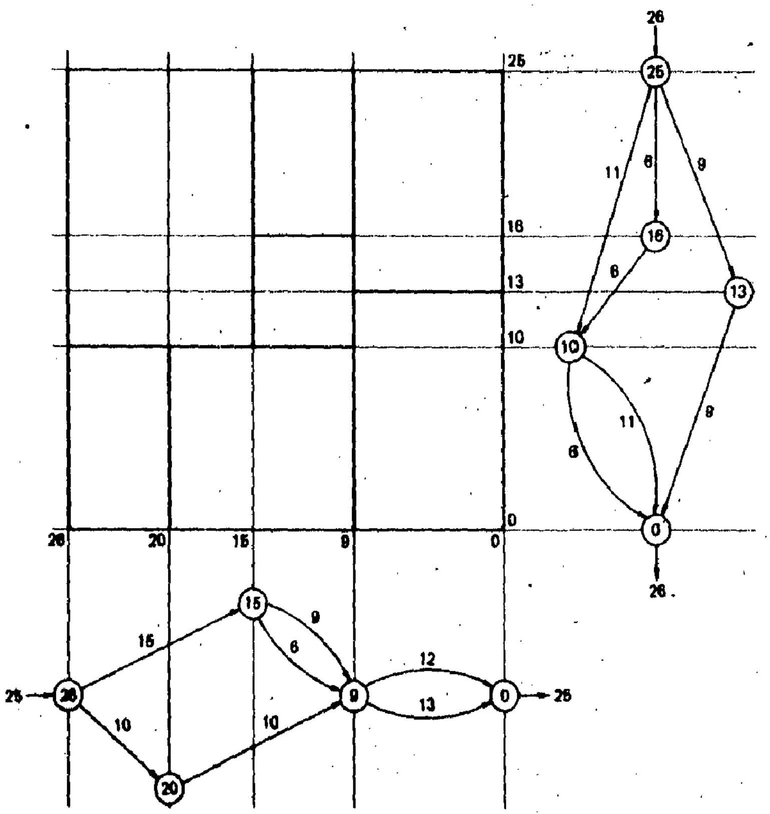
C = Conductance

V_1 = larger voltage

V_2 = smaller voltage

Interpreting this in terms of a rectangular floor plan, we see that $(V_1 - V_2)$ represents the length of the room, and C represents the proportion (ratio of length to width). The second law simply states that current entering a terminal must equal

Figure 1-33
Smith diagram representation of a rectangular floor plan. (a) East-west. (b) North-south. (c) Incidence matrix encoding.



(a) (b)

current leaving. In terms of the floor plan, this simply means that the sum of the widths of rooms on one side of a wall segment must equal the sum of the widths on the other.

A Smith diagram may be numerically encoded as an incidence matrix. Associated with each row (vertex) is a 'potential', and associated with each column (edge) is a 'current', as shown in Figure. This representation can then be stored in array form. The order in which the edges are arranged around a vertex in a Smith diagram indicates the order in which rooms occur, and this order must therefore be preserved in the incidence matrix. Using pencil and graph paper, the reader can soon convince himself that the complete plan can be reconstructed from this matrix.

Since any rectilinear plane shape may be dissected into rectangles, the Smith diagram representation can be generalized to deal with any rectilinear floor plan by the introduction of 'dummy' walls and spaces, as illustrated.

1.8 GRAPH-THEORETIC REPRESENTATIONS IN THREE DIMENSIONS :

Just as a floor plan may be treated as a planar graph, so may a section. All the graph theoretic representations that may be used for floor plans may also be employed for sections.

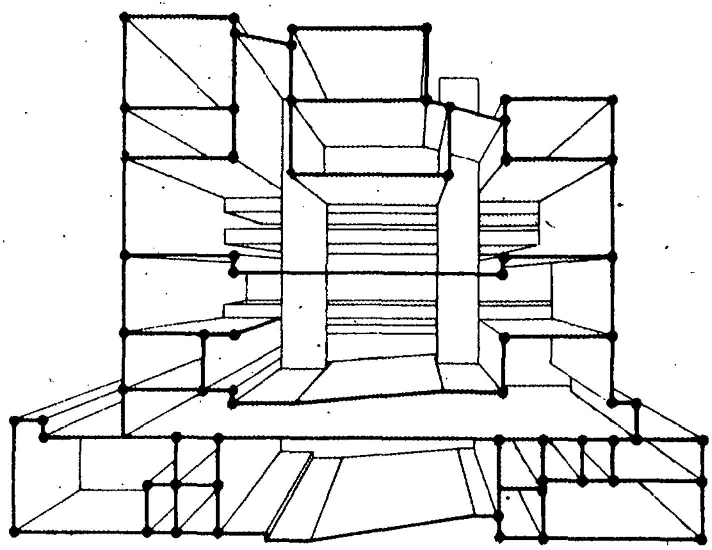
A building section is regarded as a graph in which floor and wall segments are edges, and joints are vertices, (Figure 1.34). The graph can then be encoded as an adjacency or incidence matrix, with associated dimensional and locational data, in the normal way. This is a particularly useful approach to describing the geometry of a building frame for purposes of structural analysis. The connectivity of members is explicitly recorded, data on the properties of members can be associated with edges, and data on the properties of joints can be associated with vertices.

A dual graph for the section can also be constructed, [Figure 1.34(b)]. In the same way that a dual-graph can be used to generate floor layouts that satisfy specified plan adjacency requirements, dual-graphs can also be employed to produce sections which satisfy specified adjacency requirements in a vertical plane.

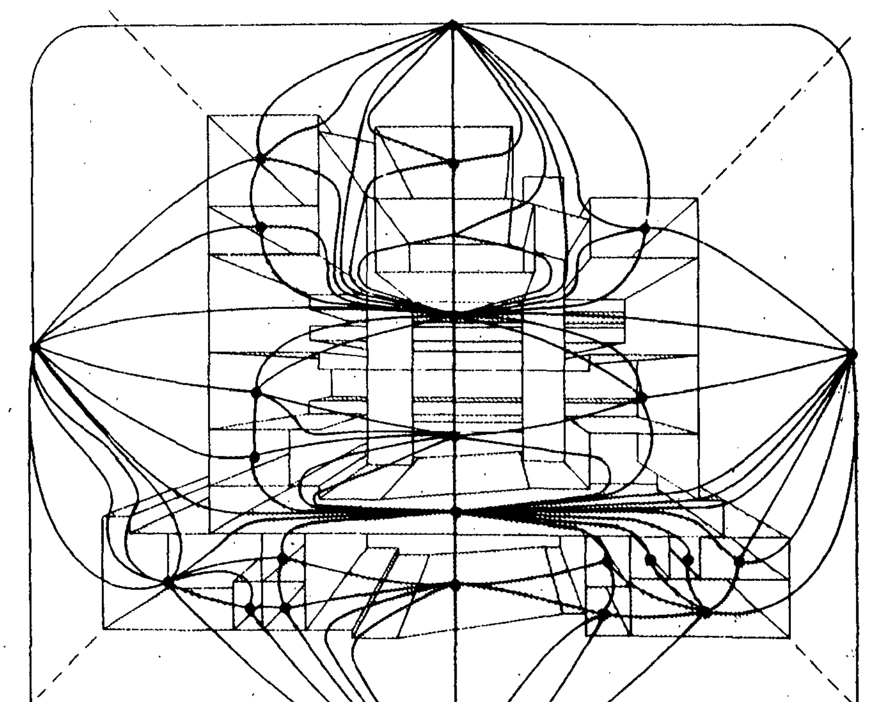
In the Smith diagram representation of a section, 'potentials' at vertices correspond to floor heights and 'currents' to the widths of spaces at the section plane.

Figure 1-34

Graph representation of the section of the Yale School of Art and Architecture building (architect Paul Rudolph, 1958–64). (a) Section represented as a graph. (b) Dual of the section.



(a)



(b)

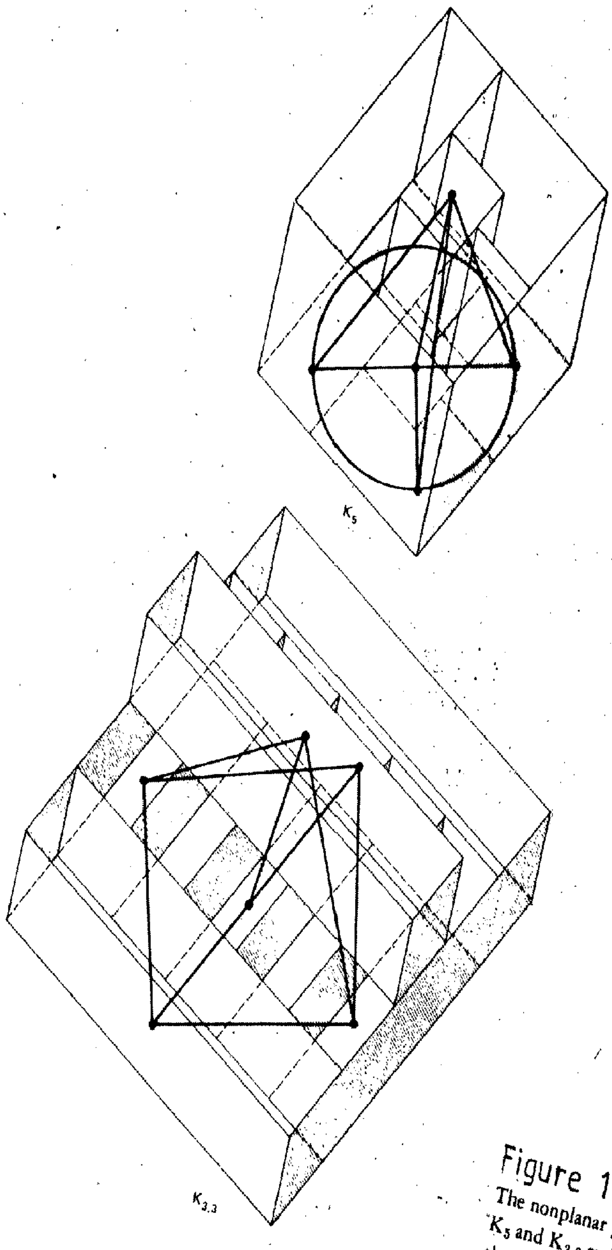
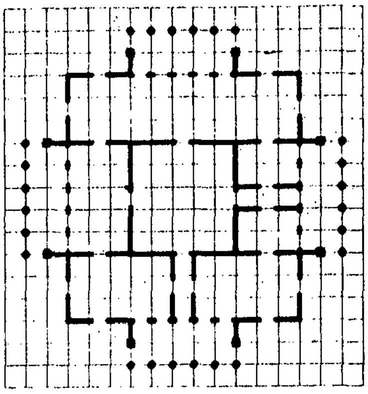
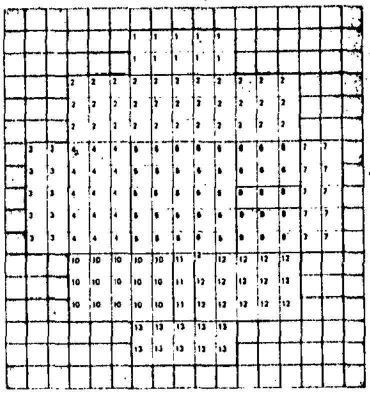


Figure 1-35
The nonplanar graphs K_5 and $K_{3,3}$ realized as three-dimensional adjacency graphs.



(a) Scheme for a villa from J. Gwilt's Rudiments of Architecture (1826)

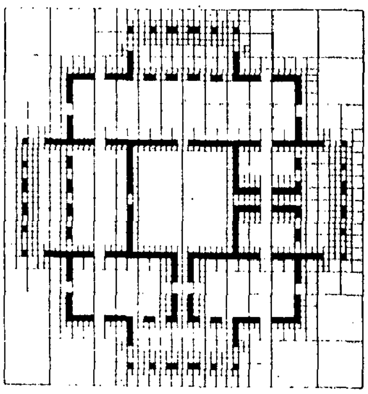


(b) Regular (square) grid representation

Advantages: *Conceptual simplicity*
Ease of implementation using array handling facilities of common programming languages
Spatial consistency is automatically maintained (since space is explicitly represented)
Many geometric computations, e.g. computing areas, testing for adjacency of spaces, *are made simple*

Disadvantages: Tends to consume excessive storage
Imposes a geometric discipline which may be inappropriate for many applications

Typical applications: Automated floor plan layout programs
Description of highly modular buildings

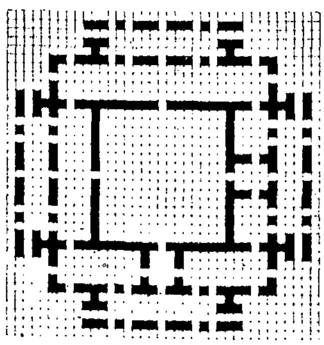


(c) Hierarchical array form of square grid representation

Advantages: Preserves many of the advantages of the simple square grid representation, while allowing relatively economical representation of fine detail

Disadvantages: Tends to be difficult to implement and manipulate, especially if the programming language employed does not incorporate the necessary facilities

Typical applications: No architectural applications reported
Mobile automata



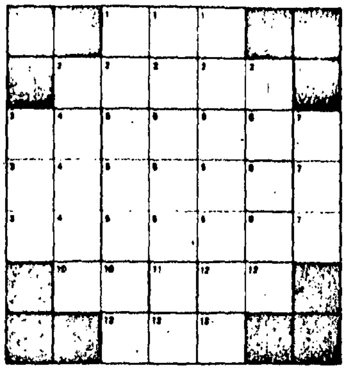
(d) Dimensionless representation (of solids and voids) in conjunction with dimensioning vectors

Advantages: Like the hierarchical array, preserves many of the advantages of the simple square grid representation, while allowing relatively economical representation of fine detail
Straightforward to implement and manipulate using array handling facilities of common programming languages

Disadvantages: Not well suited to description of non-rectilinear forms
Inefficient for description of very complex shapes

Typical applications: Comprehensive description of buildings of basically rectilinear geometry
Dimensional optimization programs

Figure 1-36
Comparison of different methods of geometric description.



(e) Dimensionless representation (of spaces) in conjunction with dimensioning vectors

Advantages: Extremely efficient method for description of rectilinear forms

Preserves most of the advantages of the simple square grid representation

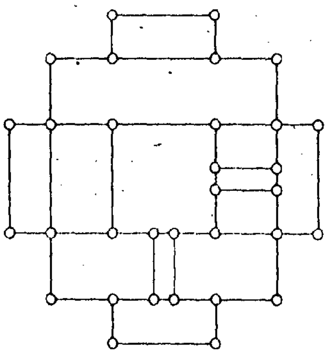
Separation of shape description from dimensional information facilitates efficient enumeration of generic plan forms

Disadvantages: Not well suited to description of non-rectilinear or complex shapes

Typical applications: *Comprehensive description of buildings* of basically rectilinear geometry

Automated floor plan layout programs

Dimensional optimization programs



(f) Polygon (or polyhedron) representation

Advantages: *Very general and flexible, imposes* few restrictions on geometry

A large amount of research has been devoted to development of data structures and algorithms for handling this type of representation

Well suited to production of graphic output

Disadvantages: Sophisicated polygon or polyhedron representations tend to require the support of complex and extensive software

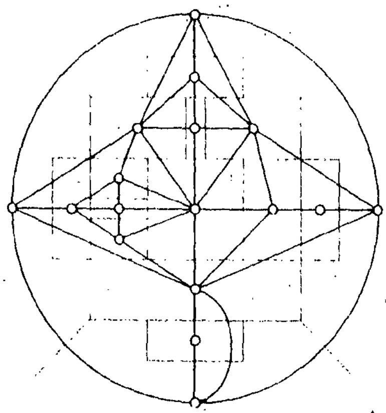
Implicit, rather than explicit representation of spaces causes serious difficulties in computing intersections, overlaps, adjacencies, etc.

Typical applications: Comprehensive building description

Graphics production applications

Spatial synthesis applications

Various engineering applications



(g) Dual graph representation

Advantages: Directly and efficiently represents adjacencies between spaces

May be used as a basis for certain floor plan layout techniques

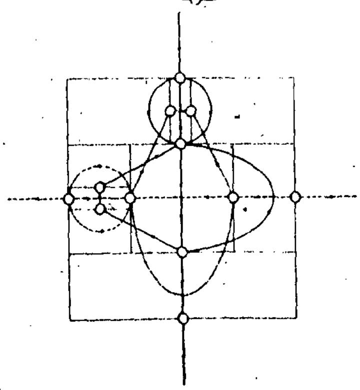
Can represent a floor plan concept before shape or dimension decisions have been made

Disadvantages: *Essentially describes topology,* and does not form a very convenient basis for developing a geometric description

Typical applications: Representation of very early layout concepts

Automated floor plan layout programs

Traffic or heat flow analysis



(h) Smith diagram (or Teague network) representation

Advantages: Very efficient for description of rectangular geometries

Allows exploitation of the "electrical network" analogy

Disadvantages: Restricted to rectangular geometries

Typical applications: Comprehensive description of buildings of basically rectangular geometry

Automated floor plan layout programs which exploit the "electrical network" analogy

Figure 1-36.

APPENDIX II

COMPUTER PROGRAMS IN ENVIRONMENTAL DESIGN*

Originator

University	135
A/E Firm	84
Service Bureau	72
Individual	18
Research/Government	17
Hardware	13

Status

Current	192
70 - 73	88
Before 70	54

Major Software

FORTRAN	274
SPECIAL	12
ALGOL	10
COBOL	9
BASIC	9
PL/1	8
APL	4
MATH PLUS	3

Major Hardware

IBM	183
UNIVAC	54
DEC	29
CDC	21
GE	11
ICL	11
ATLAS	9
NOVA	5
SIGMA	4
HONEYWELL	3
BURROUGH	1
INTERDATA	1

Core Size

33-64K	107
17 - 32K	95
Over 64K	62
Up to 16K	57

Availability

At Cost	147
Negotiation	84
Non-Proprietary	57
Not Available	44

Country

U.S.	245
England	35
Scotland	25
Canada	8
Israel	5
Australia	4
Argentina	3
France	2
Denmark	1
Germany	1
Switzerland	1
Turkey	1
Venezuela	1

Thirteen Areas of Application

Feasibility Study	39
Architectural Programg.	14
Space Planning	71
Two-Dimensional Graphics	30
3-Dimensional Graphics	32
Cost Control	50
Environmental Control	34
Circulation Analysis	20
Text Manipulation	19
Project Control	25
Office Management	22
Evaluation	28
Site Planning	68

*REFERENCE :- COMPUTER AIDED SPACE PLANNING - DR. LEE, KAIMAN (1976).

APPENDIX III

COMPUTER PROGRAMS IN ENVIRONMENTAL DESIGN

1. Activity Analysis Program/ Mike Gizinski and Dan Smith./ School of Design, North Carolina State University, Raleigh, North Carolina, 27607.

2. Additive Element Technique for space allocation./ William H.Parsons, Jr./ Thesis copy - library of Civil Engg., Mass. Institute Of Tech., Cambridge, Mass, 02139.

3. Allan/ Mr. Allan, Dept. Of Industrial Engg., Penn. State Univ./ Computer aided Design Lab. Dept. Of Arch. Engg., Penn. State Univ., Univ. Park, PA 16802.

4. Allocation Design Analysis Programming Technique./ I. Paul Lew./ Library of the School of architecture, Columbia Univ. New York, N.Y. 10027.

5. Alokat/ Allen Bernholtz and Steve Fosburg./ Allen Bernholtz,. 44B Ontario Street, Ottawa, Ontario, CANADA,

6. Alternative Selection Matrix./ Dalton-Dalton-Little./ Computer Coordinator, Dalton-Dalton-Little-Newport Inc., 3605 Warrensville Center Road, Cleveland, Cleveland, Ohio 4412.

7. An alyze Compose Display./ Franz S. Veit./ The cannon Partnership, 2170 whitehaven Rd, Grand Island, New York, N.Y. 14072.

8. Arang 11./ Charles Eastman/ Charles Eastman, School of Urban and public affairs, Carnegie-Mellon Univ., Schenley Park, Pittsburgh, PA 15213.

9. Archit./ Jerry Finrow and Robert L. Heilman./ Jerry Finrow Dept. Of Arch., Univ. Of Oregon, Eugene, Oregon 974033.

10. Architects Computer Graphics Aid./ Robert Wehrli, Max J. Smith, At Dept of Arch., Salt lake city, Utah.

11. Architectural Drawing System./ Michael Eiben and Maurice Ginardi./ Computer sercice, Inc., Marina City, Chicago. Illinois 60610.

12. Architectural Graphics Subroutine./ Ditto/ Ditto.

13. Architectural information Retieval System./ Ditto Ditto.

---

* REFERENCE :- COMPUTER PROGRAMS IN ENVIRONMENTAL DESIGN DR. LEE KAIMAN (1974).

14. Architectural information Storage and Retrieval System./ Thomas. S.Brown and Lawrence O. Sinkey./ Comp. Aided Des. Lab. Dept. Of Arch. Engg., Penn. State Univ., Univ. Park, Pen n 16802.

15. Arta./ An Interactive Animation Sequence./ L. Mezei and A. Zivian./ The Comp. Systems Research group, Univ. of Toronto, Toronto 181, Ontario, Canada.

16. Assig./ H. Sussock./ H. Sussock, Abacus Sercices, Dept. of Arch. and Building Science, Univ. of Strathclyde, 131 Rottenrow Glasgow G4 ONG, Scotland.

17. Build Form Manipulation./ Robin Th'ng and Malcolm Davies./ Abacus Services. Dept of Arch. and Building Science, Univ. of Strathclyde, 131 Rittenrow, Glasgow G4 ONG, Scotland.

18. Circe./ Andrew Bond./ Andrew Bond, The Computer Applications Group, Dept. of Arch., Univ. of Bristol, 91 Woodland Road, Bristol BS8 1US, England.

19. Circulation./ Computer Univ, Dept. of Arch., Leeda Polytechnic./ Computer Univ, Dept. of Arch. Studies, Leeds Poly., Leeds, England.

20. Cluster Analysis Program./ G.F.Bonham-Carter and Designs Systems Inc., Boston./ Design Systems, Inc., 34 Barberry Road, Lexington Mass. 02173.

21. Clustr./ Murray Milne./ Murray Milne, School of Arch. and Urban Plng., Univ. of Cal., L.A., California, 90024.

22. Compl 1 and 2./ Des. Sys. Inc./ Des. Sys. Inc.,34 Barberry Rd. Lexington, Mass. 02173.

23. Ark 2 System./ Herry, Dean and Stewart./ Computer/ Architect Coordinator, 955 Park Sq. Bldg., Boston, Mass. 02116.

24. Computer Accessed Library of layouts./ Krishan Mathur/ Abacus Services, Dept. of Arch. and Bldg. Sc., Univ. of Strathclyde, 131 Rootenrow, Glasgow, G4, ONG, Scotland.

25. Computer Aided Constraint Oriented approach to the design of home-units./ John. S.Gero and Ian James./ John Gero, Dept. of Arch. Sc., Univ. of Sydney, Syney, Australia.

26. Computer aided space allocation technique./ Donald Grant Arthur Chapman./ D.C. or A.C., School of Architecture, Cal Poly, San Luis Obispo, California, 93407.

27. Computer Aided Spatial Synthesis./ William, J.Mitchell/ William. J. Kitchell, School of Architecture and Urban Plg., Univ. of Cal., L.A., Cal.

28. Computer Aided system for Housing./ Krishan Mathur./ T.W. Mayer, Abacus Services, Dept. of Arch. and Bldg. Sc., Univ. of Strathclyde, 131 Rottenrow, Glasgow G4, ONG, Scotland.

29. Computer Assisted land Use planning./ Dieter Schindowski and H.J. Fey, Univ. of Dortmund, Dept. Raumpanung, Dortmund, Germany.

30. Computer Packing of Rectangular envelope with spaces of sides./ Krishan Mathur./ Abacus ....

31. Computerised Relationship Layout planning./ School of Arch., Cal. State Poly Univ./ Systems Engg. Group. Albert C.Martin and Assoc., P.O. Box 60147, L.A. Cal. 90060.

32. Contour analysis by random triangularation algorithm./ Idan Computers./ Idan Computers Ltd.,P.O.Box 33390, Tel Aviv. Israel. CARTA.

33. Decompit./ Robert Ray./ Robert Ray, Center for advanced Computation, Univ. of Illinois, Urbana, Illinois 61801.

34. Diprocod./ Arturo F.Montagu./ ......, Consejo National de Investigaciones Clentificas Y Techica, Address : Rivadavia 1971, Buenos Aires. R.Argentina.

35. Domino./ William J.Mitchell and Robert. L.Dillon./ W.JM. School of Architecture and Urban Plng., Univ. Of Cal., L.A., California 90024.

36. Evaluation of Problem Structu./ Charles F.Davis. 111 and Michael Kennedy./ Charles Davis 111 at Skidmore, Owings and Merrill, 30 Monroest, W.Chicago, Illinois, EVA, EVAPROBST.

37. FASTDRAW./ McDonnell Douglas Automation Company./ William J.Vickroy, Vice President. of marketing, McDonnell Douglas Automation Company, 5.O. Box 516, St. Louis, Missouri 63166.

38. Form And Annotaion Plot./ Dalton-Dalton-Dalton./ Little-Newport Inc., 3605 Warrensville Center Road., Cleveland, Ohio 44122. FORM.

39. Formal Space Planning Language./ Christos I Yessios./ ..... Institute Of Physical Planning., Carnegie-Mellon Univ., Pittsburgh, Pennsylvania. 15213. FOSPLAN II.

40. General Space Planner./ Charles E.Eastman./... Institute of Phusical Planning, Carnegie Mellon Univ., Pittsburgh, PA. 15213. GSP.

41. Generation of Random Access Site Plans./ Eric Teicholz 34,B arberry Road, Lexington, Mass. 02173. GRASP.

42. Genopt 2./ B.Whitehead./.....Dept. of Bldg. Engg. Muspratt Lab., Univ. of Liverpool, P.O. Box 147, Liverpool L69 3 BX, England.

43. Graph Manipulation Package For Space Layout./ John Grason./..... Dept. of Electrical Engg., CArnegie-Mellon Univ. Pittsburgh, Penn. 15213. GRAMPA.

44. Grid./ DAvid Sinton and Carl Steinitz./ Lab. For Computer graphics and spatial analysis, Gund Hall, Graduate school of Desig, Harvard Univ. Cambridge, Mass. 02138,

45. Heuristic Optimization of Topological layouts using high level interaction./ John S.Gero, Warren Julian and Neville Holmes./..... J.S.G., Dept. of Archl. Sc., Univ. of Sydney, Sydney, 2006, Australia.

46. Hidecs-Recomp Procedure./ Edward Bierstone and Allen B ernholtz./ A.B. 44B Ontario Street, Ottawa, Ontario, Canada.

47. Hierarchial Decomposition of a system with an associated Linear graph. HYPER AND HYPAP./ Kenneth W.Panzl, revised by Christopher Alexander and Gary T Moore./ Pror. Ch.Alex Dept. of Arch. Univ. of Cal., Berkeley, Cal. 94720.

48. Hle./ HSussock./....., Abacus services, Dept. of Arch. and Bldg. SC., Univ. of Strathclyde, 131 Rottenrow, Glasgow G4 ONG, Scotland.

49. House Design.'. Complex Housing Forms. LRHD./ Aart Bijl, D.F. Barnard, R.E. Handley, Et al./ Aart Bijl. Edinburgh CAAD studies, Dept. of Arch., Edinburgh Univ.,/ t 55 George Square, Edinburgh, Scotland.

50. House Design Game./ Luis. H.Summers./ Dr. Luis.H. Summers. Dept. of Arch. Engg., 101 Engg. 'A'., The Penn. State Univ. State College, PA 10802.

51. House Design. Scottish Special Housing Assoc., DH/ SSHA./ Aart Bijl .... Ref. 49).

52. Housing Evaluation and Layout Package./ HELP./ Thomas. W. Maver./ ......, Abacus Services. Ref 48).

53. Housing Project Planning System. HPPS./ The Advanced Plg. Research Group, Inc., Suite 207, 10400 Connecticut Ave., Kensing ton, MD. 20795.

54. HUNCH./ Nicholas Negroponts, James Taggare and Others./ N.N., The Arch. Machine Group, Dept. of Arch., M.I.T., Cambridge, Mass. 02138.

55. IMAGE./ Timot by E.Johnson, Guy Weinzapfel, John Perkins. Etc./ Dept. of Arch., Mass. Institute of Tech., Cambridge, Mass.

56. Implicit Enumeration Approach to Floor Plan Layout./ Robin Segerblom Lig get./ School of Arch. and U.Plg., Univ. of Cal., L.A. Cal 90024.

57. Interactive Computerized Relationship Layout Planning./ Robert. G.Lee., James. m.Moore, and Others./ Engg. Mgt. Associates, 302A Whittemore Hall, Blacksburg, Virginia. 24051.

58. Interior And./ Architectural Drafting System./ Man/Mac./ SLS Envirommetics./ Mr. Marshall A. Graham, The Computer Dept. SLS Environeties, 600 Madison Ave., N.Y., N.Y. 10022.

59. Interior Design Information System./ Charles Davis./ I.D.I.S./ ...., Skidmore, Owings and Merril, 400 Park Ave., New York, N.Y., 10022.

60. Interrelations Analysis Model./ IAM/Charles Davis./ Schraek, Library of the Dept. of computer Sc. or Architecture, Texas A. and M. Univ., College Station, Texas 77843.

61. LIFE3./ Philip A.Hendren./...., 2105 Scenic Drive, Austin Texas 78703.

62. LOCATπ/ Isao Oishi./ Prof. Isao Oishi, College of Architecture Virginia Polytechnic Instit. and State Univ., Blacksburg, VA 24061.

63. LOKAT./ Sponsored - Perkins and Will, Architects, Programmed-Dave Sinton and Steve Fosburg under the directionoof Alan Bern holtz./ Allan Bernholtz, 44B Ontario St., Ottawa, Ontario Canada.

64. Manipulation of Arbitrary Line Drawings./ SPARTA./ L.Mezei....., Dept. of Computer Sc., Univ. of Toronto, Toronto, Ontario, Canada.

65. Matrix Analysis Program-Three./ MATRAN-III./ William. R. Miller./ The Systems Engg. Group, Albert. C.Martin and Assoc. P.O. Box 60147, Terminal Annex, Los Angeles, Cal 90060.

66. OTOTROL./ Moritz Bergmeyer and Cal Opitz./ Lab. For Computer Graphics and Spatial Analysis, Gund Hall, GraduatesSchool of Desig, Harvard Univ., Cambridge, Mass. 02138.

67. Package For Hospital Arch., Simulation and Evaluatio./ Phase 1./ D.Kernohan, G. Ranking, G. Wallace, R. Walters./ D.Kernohan, Dept. of Arch. and Bldg. Sc., Univ. of Stratholyde, 131 Rottenrow, Glasgow G4 ONG, Scotland.

68. PLAN 6./ Kelly R.McAdams./...., P.O.Box 8294. Austin Texas, 78712.

69. PLANIT./ Glenn Lavine, Bank Building Corp. 1130 Hampton Ave. St. Louis, Missouri 63139.

70. Planning Module Selection System./ CRS Design Assoc. Data Center, and Caudil 1 Rowlett Scott, Inc./ CRSDA Data Center./ Design Assoc., Inc., 1100 Milam, Houston, Texas 77002. CRS.

71. Preliminary Design Area Generator./ PRELM./ Maurice Girardi and Michael Eiben./Computer Service Inc., Marina City, Chicago, Illinois 60610.

72. Rectangular Meshes : Their Uses and Control In Computer Produced Architectural Schemes./ Gonzalo Velez./ Abstracted from the proceedings of the annual conference of the association for computing machinery PP 745-755, August, 3-5,1971, Chicago, Ill.

73. Relationship Layout Technique./ RELATE./ Lester Gorsline Assoc., P.O. Box 6276, Terra Linda, Cal 94903.

74. RUMOR./ Sponsored - Milton Fund, Programmed - Katharine Kiernan. Supervision - Allan Bernholtz./ Allan Bernholtz, 44B Ontario Street, Ottawa, Ontarion Canada.

75. Scheduling Package And Computer Evaluation of Schools./ Spaces./ Robert Th'ng and Malcolm Davies./ ....or....., Abacus Services, Dept. of Arch. and Bldg. Sc. Univ. of Strathclyde 131 Rottenrow, Glasgow G4 ONG, Scotland.

76. Site Planning Oriented Computer Language./ Christos I Yessios./ C.I. Yessios, 392 Brown Hall, School of Arch., The Ohio State Univ. Columbus, Ohio./ 43210 or Institute of Phisical planning, Carnegie Mellon Univ. Pitts., PA 15213.

77. Space Organization Method./ Wally Rutes of Skidmore Owings and Merrill./ The Dept. of Civil Engg. and Graphics, Columbia Univ., New York, New York 10027.

78. Space Organisation Method - Interactive./ SOMI./ Sami Al-Banna and W.R. Spillers./..... Dept. of Civil Engg. andGGraphios, Columbia Univ., N.Y., N.Y., 10027.

79. Space - Plan./ Robert Krawezyk of C.F. Murphy and Assoc., Chicago and Prof. Elliot Dudnik,/ E. Dudnik, Dept. of Arch., Univ. of Illinois - Chicago Circle, Chicago Ill 60680.

B I B L I O G R A P H Y

1. Anselin, Luc - Estimation methods for spatial auto-regressive architecture - a study in spatial economics - 1980.

2. Auger, Boyd - Architect and the Computer - 1972.

3. Baxter, Richard, (Ed) Perraton, Jean and Baxter- Models evaluation and information systems for planners - 1974.

4. Baxter, Richard - Computers and statistical techniques for planners - 1976.

5. Bishop, Peter - Comprehensive computer studies - 1981.

6. Campion, David - Computers in Architectural Design - 1968.

7. Conference on small computers systems applications in construction - 1980.

8. Cowan, H.J. and others - Models in Architecture - 1968.

9. Cross, N. - Automated Architect - 1977.

10. Eastman, C.M. - Spatial Synthesis in computer aided buildings design - 1975.

11. Evans, N - The architect and the computer - a guide through the jungle - 1981.

12. Flores, Ivan - Data base architecture - 1981.

13. Gilol, Wolfgang K - Interactive computer graphics : data structure/algorithm and language - 1978.

14. Gray, Crispen - Fundamental conception in computer aided arch. : Storage and data structure.

15. Gutteridge, Bryan and Wainwright, F.R. - Computer in Arch. practice - 1973.

16. Handler, A.B. - Systems approach to architecture - 1970.

17. Harold, C.Mill - Information processes and computer programming - an introduction - 1973.

18. Harper, S.N. (ed) - Computers applications in arch. and engineering - 1965.

19. John S. Gero - Computer application in arch.

20. Kaiman Lee and others - Step toward an integrated design system for the architect/planner - 1973.

21. Lasseau, Paul - Graphic thinking for architects and designers - 1980.

22. Lee, Kaiman - Computer generated perspective drawings - 1974.

23. __________Bibliography of the Computer in Environmental design - 1973.

24. __________Computer aided space planning - 1976.

25. __________Computer as an arch. design tool - an exploration into certain multistoreyed building plan layouts - a thesis - 1969.

26. __________Computer program in environmental design - 1974.

27. __________Environmental impact statement : a reference manual for the architect/planner - 1974.

28. __________Evaluation, synthesis and development of an interactive approach to space allocation - 1975.

29. __________Interactive computer graphics in architecture - 1976.

30. __________State of the art of computer aided environmental design - 1975.

31. __________Performance specification of computer aided environmental design - 1975.

32. Lee Kaiman (Ed) - Computer aided architectural design - 16 Arck - 2 - Articles - 1973.

33. Lee, Kaiman and Masloff, Jacqueline - Energy Oriented computer programs for the design of monitoring of building - 1979.

34. Lee, Kaiman and Molerb, John - Environmental design evaluation - a matrix method - 1975.

35. Lee, R.C. and Moore, J.M. - Corelap - Computerised relationship layout planning - Journal of Industrial Engg. - 8th March 1967, No.3 - pp.195-200.

36. Mitchell, W.J. - Computer aided arch. design - 1977.

37. Negroponte, Nicholas - Architecture machine - towards a more human environment - 1970.

38. ________ (ed) - Reflection on computer aids to design and architecture - 1975.

39. Reynolds, R.A. - Computer methods for architects - 1980.

40. H.T. Smith and T.R.G. Green - 1980 - Human interaction with computers.

41. Sprouls, R. Clay - Computers - a programming problem approach - 1966.

42. Steven, J. Fenves - Computer method in Civil engineering - 1967.