THE AUTOMATED GENERATION OF ARCHITECTURAL FORM

WILLIAM J. MITCHELL

Assistant Professor
School of Architecture and Urban Planning
University of California, Los Angeles

Visiting Critic
Department of Architecture
Yale University

ABSTRACT

A systematic framework for discussion of automation of the solution of architectural problems is established, based on an examination of the concepts of solution representation, generation, and testing. Some of the more important implications of various different techniques and principles of representation, generation, and testing are then illustrated by a discussion of procedures for solution of some simple spatial arrangement games and puzzles. It is shown that when we attempt to write procedures for solution of the rather larger and more complex problems encountered in practical architectural design, we discover some quite severe limitations on our current ideas about architectural design automation. The major limitations are outlined, and progress made towards overcoming them recounted.

1. SOLUTIONS TO ARCHITECTURAL PROBLEMS -- REPRESENTATIONS, GENERATORS, AND TESTS

> If we were going to speak of the different species of animals, we should first of all determine the organs that are indispensable to every animal, as, for example, some organs of sense and instruments of receiving and digesting food, such as the mouth and stomach, besides organs of locomotion. Assuming now that there are only so many kinds of organs, but that there may be differences in them -- I mean different kinds of mouths, and stomachs, and perceptive and locomotive organ -- the possible combinations of these differences will necessarily furnish many varieties of animals. (For animals cannot be the same which have different kinds of mouths or ears). And when all the combinations are exhausted there will be as many sorts of animals as there are combinations of the necessary organs. (1)

In this passage, from the Politics, Aristotle puts forward a notion which has become a widely accepted basis for systematic description and discussion of problems of design and decision -- we define a set of dimensions of variation, describe the alternative possibilities along each of these dimensions, then consider the set of alternative combinations thus generated.

Applications of this principle have appeared and reappeared throughout history. The voluminous works of the thirteenth century Spanish mystic, Ramon Lull, describe a "universal" method, by means of which any aspect of the cosmos might be described in terms of combinations of "attributes of God," and represented using a special graphic notation. In the seventeenth century, the young Leibniz (apparently influenced by the work of Lull) described a combinatorial art of description and invention in De Arte Combinatoria (1666). In a letter to the Duke Johann Friedrich, in 1671, he claimed that it had aided him in the design of an extraordinary collection of objects: an adding machine and a ready-reckoner, telescopes and microscopes, pumps, ships, and submarines. Since Leibniz there have been several noteworthy attempts to develop procedures for the design of artifacts by the enumeration of combinatorial possibilities. Drawing on Charles Babbage's investigations of mechanical notation systems,

the engineering theorist Franz Reuleaux discussed the synthesis of machines by combination of various "kinematic elements" in different ways. In our own century, we have seen applications in a range of contexts, including design of vehicles of various types, rocket propulsion systems, and industrial processes. The major modern application, though, has not been as a method for the design of artifacts, but rather in dealing with problems of economic decision, where a set of conceivable decisions is generated by combinations of positions along vectors representing the relevant command variables in a situation. (2)

Now as designers, we are normally interested in situations where differences make a difference. Faced with a set of alternative possibilities, we seek those which are, in some sense, "better" than others. What kinds of enquiry procedures can we develop to discover them? Before we attempt to answer that, perhaps we should consider the more general question of how it is possible to enquire into the unknown at all, of how we may resolve this ancient riddle:

> And how will you inquire, Socrates, into that which you do not know? What will you put forth as the subject of inquiry? And if you find what you want, how will you ever know that this is what you did not know? (3)

Plato's Socrates at first dismissed this as a tiresome connundrum, but then acknowledged that it does raise an important issue. Unfortunately, his answer, (making use of the notion of the soul's prior existence) is not a very helpful guide to an understanding of design processes. If we consider the question at a much more pragmatic level though, we may observe that the writing of computer programs to perform such tasks is commonplace. In a paper published in 1956, John McCarthy (4) provided a particularly lucid discussion of how this is possible. He began by clearly distinguishing between "proposed solutions" and "solutions" to a problem, and discussed a class of problems described as "well-defined," in which

> ...there is a test which can be applied to a proposed solution. In case the proposed solution is a solution, the test muct confirm this in a finite number of steps.

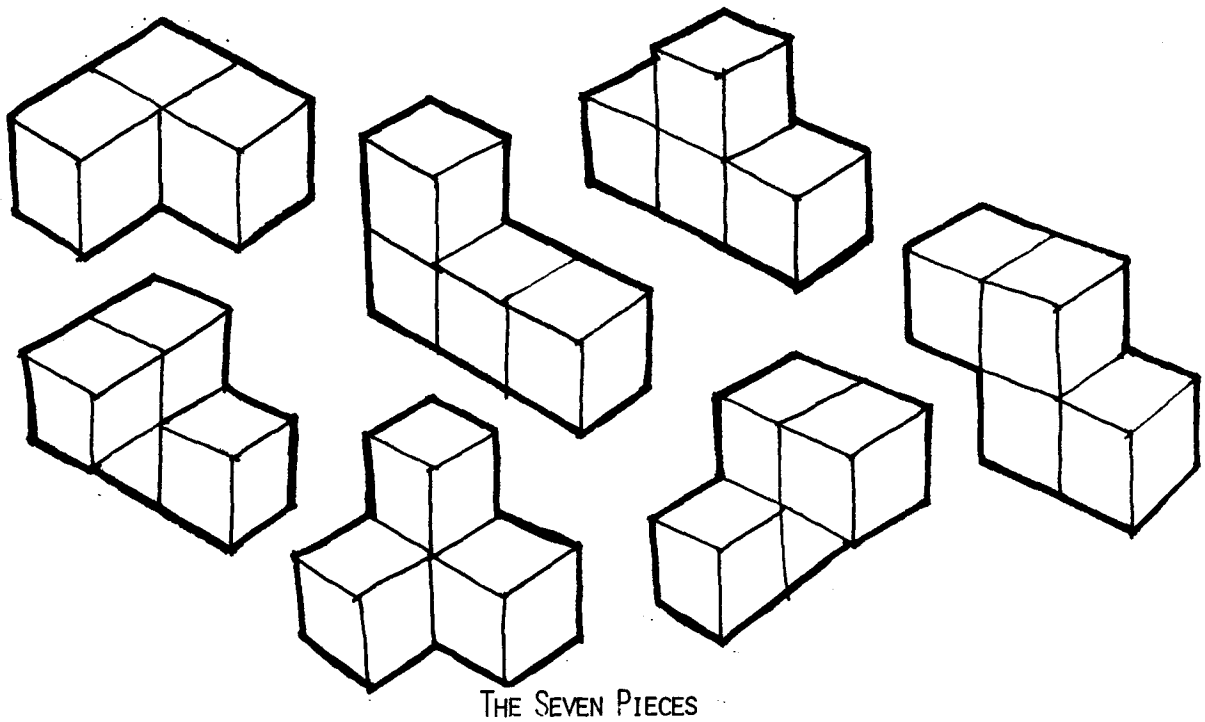
Using this notion, we can develop procedures capable of solving certain problems in this class, consisting of the following parts:

1. Some convenient and economical means of representing any proposed solution; i.e., of giving it a name.
2. A method of generating proposed solutions.
3. A test which indicates whether a proposed solution is a solution.

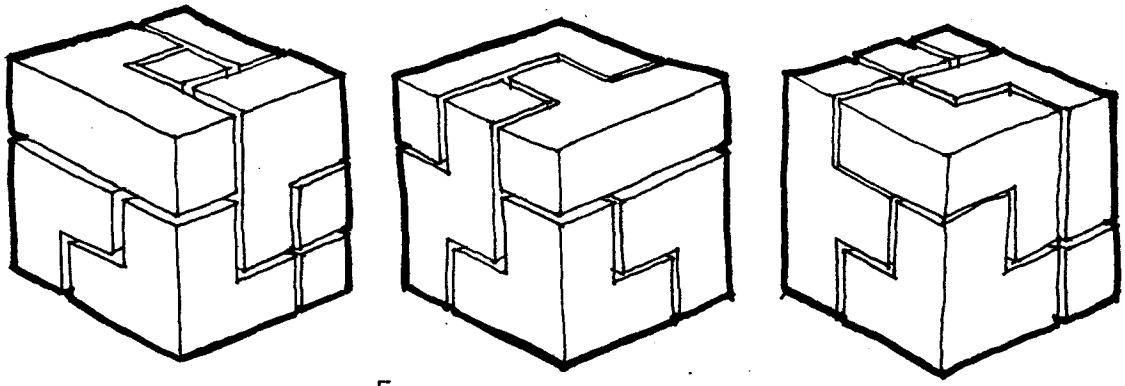
Perhaps the most widely utilized approach to the problem of representation of proposed solutions is to make use of the notion of multidimensional spaces of proposed solutions, in which each dimension corresponds to a design choice to be made, and each point in the space stands uniquely for one proposed solution. In other words, the set of points in the space is the set of all proposed solutions. One simple form of generator, then, is a procedure which does nothing more than successively select for testing some sequence of these points until one which represents an acceptable solution is discovered. It might, for instance, begin to systematically enumerate all possibilities, or randomly select points.

Generation processes of this type may, in fact, be thought of an implementations of strategies for searching within the space of proposed solutions. Imagine that the generator possesses a pointer which is used to indicate a point, and a set of operators which can be used to move the pointer from one point to another. By applying a sequence of operators, part of the space of proposed solutions is searched. Beginning at any arbitrary point, the strategies available to the generator may then, in fact, be represented by a game tree, in which nodes represent points and branches operators.

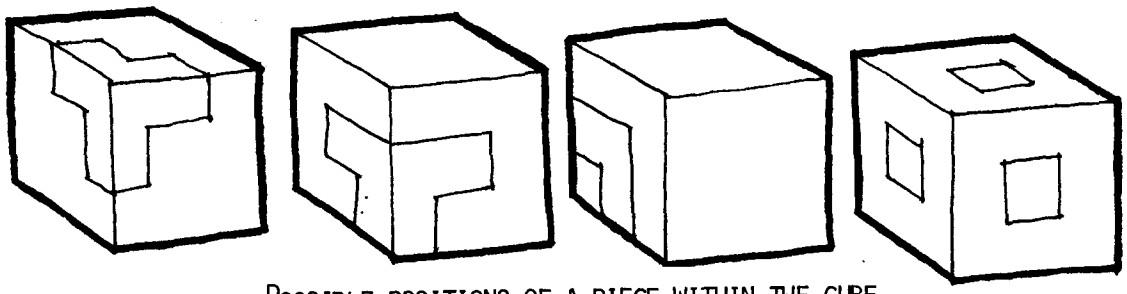
This tree representation makes it clear that there is a question of division of labor between generator and test to be considered. We can design strategies which examine large numbers of points, many of which may be "bad," and assign most of the discriminatory power to the test, or conversely, attempt to develop a generator which is more discerning, so that comparatively few complete alternatives need actually be considered (perhaps, even only one). Non-trivial design problems, however, are characterized by truly immense spaces of proposed solutions, and even using the largest and fastest computers, only very small fractions of the whole can ever actually be searched. Thus, only reasonably "intelligent" generators are of any real practical interest, and the design of generators possessing particular required properties becomes a major issue in computer-



THE SEVEN PIECES



EXAMPLES OF ASSEMBLED CUBES



POSSIBLE POSITIONS OF A PIECE WITHIN THE CUBE

FIGURE 1: A CUBE ASSEMBLY PUZZLE

aided design. (5)

Such schemes, in given situations, may or may not in principle be capable of producing solutions. A moment's reflection indicates that there are three possible types of situations, which may be described as follows:

1. Under-specification; our test allows a multitude of satisfactory solutions.
2. Unique, or almost-unique specifications; one, or a very few satisfactory solutions.
3. Over-specification; no solution which passes the text exists.

Utilizing these notions of solution representation, generation, and testing, we may now summarize the issues which we encounter in the design of designers as follows:

1. What are our design elements, and what universe of possibilities do they generate?
2. What kinds of representation schemes will be most appropriate for our purposes?
3. What division of labor between generator and test will we adopt?
4. What specific properties will then be desirable in our generator? How will it operate?
5. What criteria should our test incorporate?
6. How does our designer proceed in situations involving under or over specification?

This paper is specifically concerned with architectural design at the level of form manipulation and arrangement, that is with situations where spaces and physical components are the design elements, and examines these questions within that context.

2. SOME SIMPLE SPATIAL ARRANGEMENT PROCEDURES

It is often illuminating to play with simple toys. One of the best ways to begin to understand the implications of these questions in relation to architectural design is to write procedures for solution of some spatial arrangement games and puzzles.

Figure 1 shows the seven components with which a popular game marketed under the name of "Soma" (6) is played. The object is to assemble these components into some predefined form -- most commonly a 3 x 3 x 3 cube. It has been shown that there are something more than one million different ways of producing such a cube; the problem is under-specified. However, if we add some further requirements, such as specifying how the first piece is to be placed, then the number of alternatives may be drastically reduced. The problem becomes much more closely specified. (Most other common spatial arrangement puzzles; e.g., jigsaws, are truly uniquely specified.) Some cube-building starting positions which we can specify even appear to result in an over-specified situation (that is, no solution possible), although this is not always easy to prove.

Let us now consider how we might design a procedure for building these "Soma" cubes. Each piece has only a small number of possible positions within the cube (see figure 1). We may then conveniently represent each possible position of each piece by use of seven arrays of integers:

```
Piece 1 (position 1) ....Piece 1
                                (position n1)
. . .
. . .
Piece 7 (position 7) ....Piece 7
                                (position n7)
```

The most straightforward generation procedure is systematic enumeration of all conceivable sombinations. Now the cube consists of 27 cubical cells, and no cell can be simultaneously occupied by more than one piece. Thus our test for a solution is to check that each cell is uniquely occupied.

This procedure will discover all 1,105,920 possible cubes in a finite number of steps. But it is obviously very inefficient; a vast number of "unsatisfactory" solutions will be generated and examined along the way. Such complete enumeration is a trivial solution procedure for a wide range of problems. Where we are interested in practically applicable procedures, however, we must impose some efficiency requirement, that is, a limitation on the number of steps to be executed.

Sometimes we can achieve the requisite efficiency by designing our procedure to take advantage of some special knowledge that we have, or can discover, about the properties of our problem. For instance, we know that certain positions within a cube of any given "Soma" piece cannot possibly coexist with certain positions of another, since one or more of the 27 cells would be simultaneously occupied by both pieces. We can take advantage of this fact in designing a much more efficient procedure for building "Soma" cubes. Conceptualize the process as a

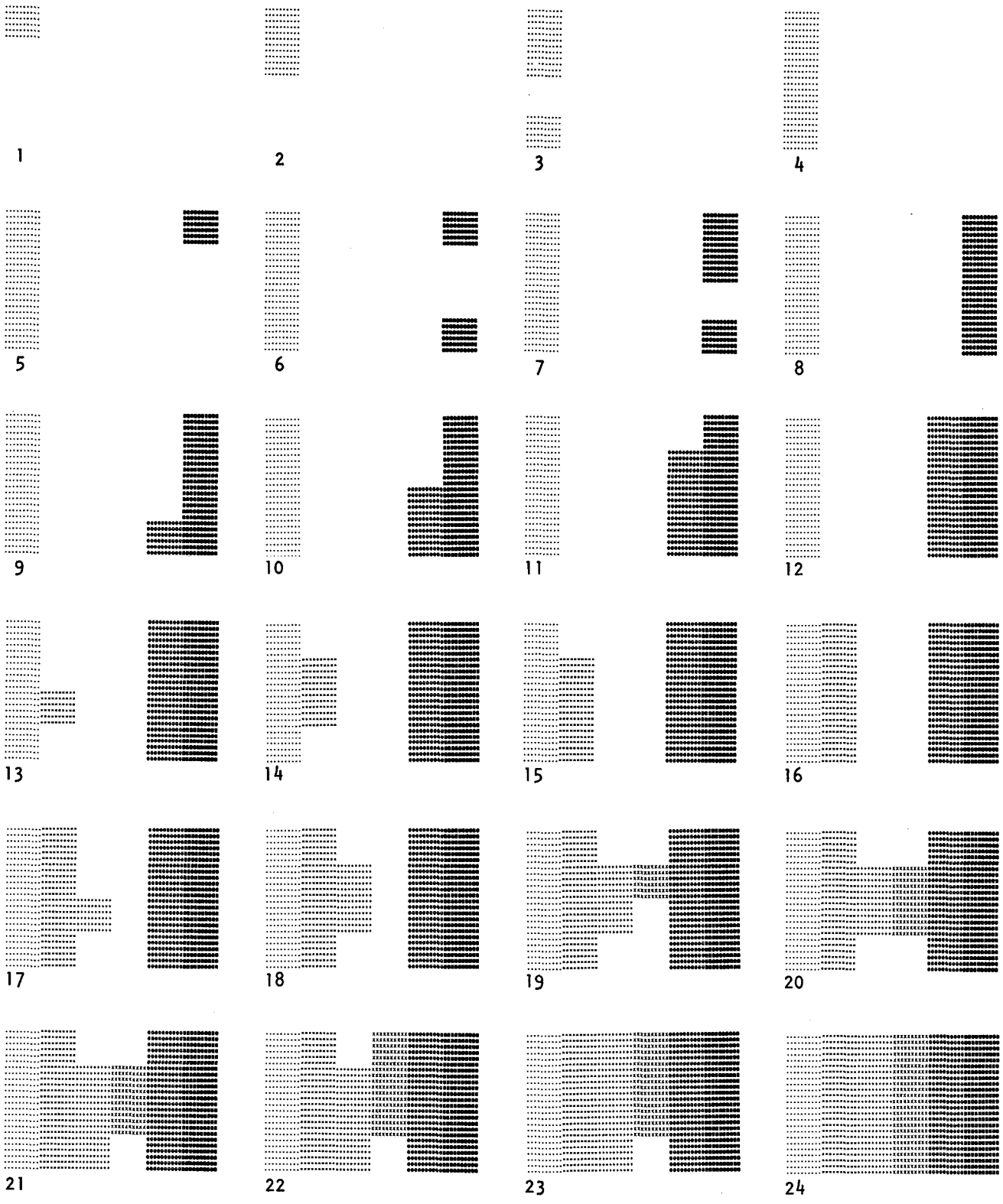


FIGURE 2: SOLUTION OF THE GREY SQUARE PLACEMENT PROBLEM

decision tree in which each level represents a location decision for one piece (there will be seven such levels), each branch a possible location of a piece, the root and internal nodes incomplete proposed solutions, and the end nodes complete proposed solutions. Now various of the nodes will contain "incompatibilities," and it follows from the structure of the tree that any node springing either directly or indirectly from these nodes must also contain those same incompatibilities. Thus we can design a procedure to systematically explore all branches of the tree, but incorporate within it a capacity to recognize nodes containing incompatibilities, and to backtrack to the next "good" node -- so by implicit rejection drastically reducing the number of branches which need actually be explored.

In many practical circumstances though, in order to keep the number of steps in our procedure at some reasonable level, we may need to sacrifice some performance quality...to abandon the certainty that our procedure will ultimately solve the problem in favour of the expectation that a solution will probably be reached in some reasonable time (i.e., use some heuristic strategy), or to relax our criteria of what constitutes an acceptable solution (satisfice rather than optimize), or both (7).

Figure 2 illustrates the performance of a computer program following a slightly uncertain, but very efficient, strategy in dealing with a fairly simple two-dimensional spatial arrangement problem. It operates on a 6 x 4 square modular grid within which 24 squares, shaded in different tones of grey are to be located. The design goal is to produce a configuration containing only smooth tonal transitions (e.g., a black square is well located if it is adjacent to black or dark grey squares, but not if it is adjacent to whites or light greys), or more precisely, to minimize an objective which is a function of separation of squares on the grey scale and actual physical separation in the configuration. This program employs an implicit enumeration approach, making use of probability calculations to assess which placements seem most likely to lead to a configuration in which the objective is minimized (8). In the example shown in figure 2 it guessed well! But it can, and does at times, make mistakes.

Such heuristic strategies, in general, depend on the insight that, where we cannot achieve some goal directly, we can often move towards it via the achievement of some sequence of appropriately selected sub-goals. Where a unique optimum solution exists, and our goal is to discover it, then all well-selected sequences of sub-goals should lead to the same result. But where many equivalent solutions exist (or where many may be regarded as effectively equivalent due to adoption of a satisficing attitude), then strategies based on different sub-goal sequences may lead to characteristically different equivalent solutions.

Figure 3 graphically illustrates some effects of sub-goal selection in this type of spatial arrangement problem. The task attacked is a variation of the grey square problem, in which we reduce the number of shades to two, black and white. Beginning from the same random configuration, three computer programs (each operating as shown in the flow diagram) were run to produce the results shown. Each used exactly the same sequence of pseudo-random numbers in selecting squares for transposition, so the differences between the configurations generated result entirely from differences in sub-goal definition (9).

The grey-square arrangement problem is worth examination from another viewpoint too; it strikingly illustrates the effects of alternate representations on the ease with which a problem may be solved. Two obvious representations for the 6 x 4 problem are as follows:

(1) Name each matrix position uniquely with an integer between 1 and 24. Similarly, name each grey square. Now, any proposed solution may be represented as a mapping of the set of grey squares onto the set of positions, thus for example:

POSITIONS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
SQUARES	10	4	16	11	17	5	19	10	1	18	12	21	22	6	13	7	14	2	23	25	3	24	8	9

(This is the representation technique utilized in the computer program)

(2) Simple literal representation -- appropriately-toned squares in a 6 x 4 modular grid as follows:

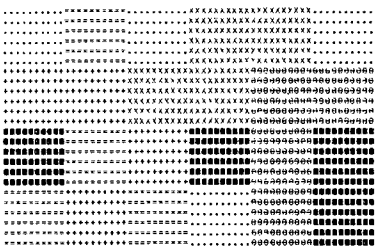
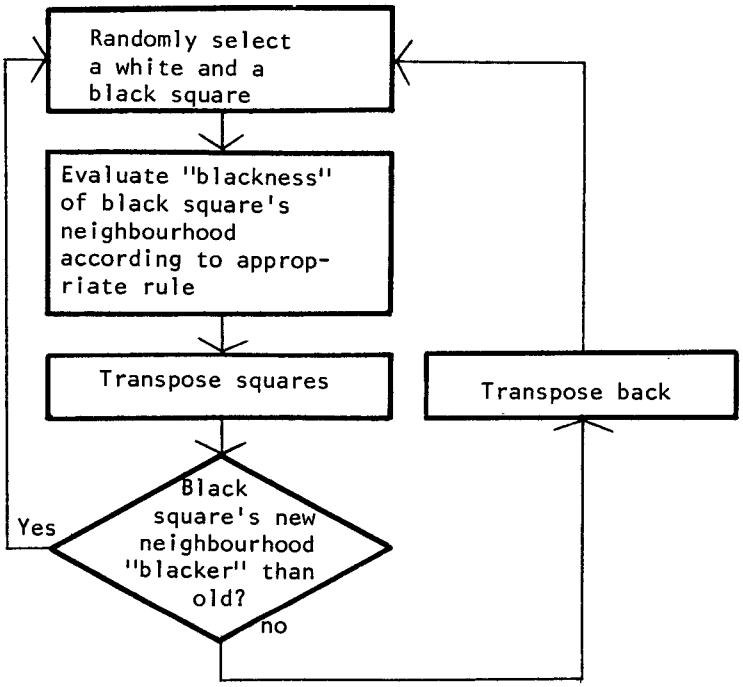
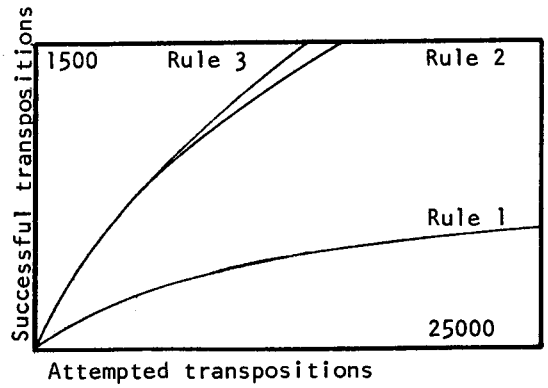


FIGURE 3: EFFECTS OF SUB-GOAL SELECTION

THE SUB-GOAL IS, IN EACH CASE, TO TRANSPOSE A BLACK SQUARE INTO A "BLACKER" NEIGHBOURHOOD, BUT THE RULES FOR EVALUATING NEIGHBOURHOODS DIFFER. UNDER RULE 1, THE NUMBER OF IMMEDIATE BLACK NEIGHBOURS IS COUNTED. UNDER RULE 2, ALL BLACK SQUARES WITHIN A RADIUS OF 10 SQUARES ARE CONSIDERED, AND THEIR EFFECTS ARE INVERSELY PROPORTIONAL TO THEIR DISTANCES FROM THE CENTER OF THE FIELD. RULE 3 DIFFERS FROM RULE 2 ONLY IN THAT THE RADIUS IS EXTENDED TO 20 SQUARES.

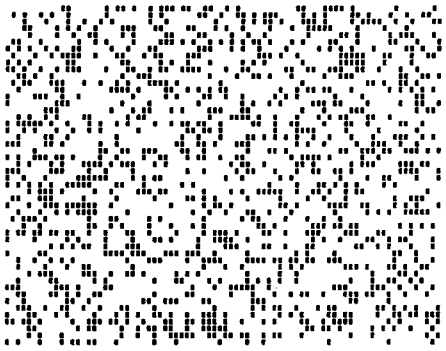


THE PROCEDURE USED

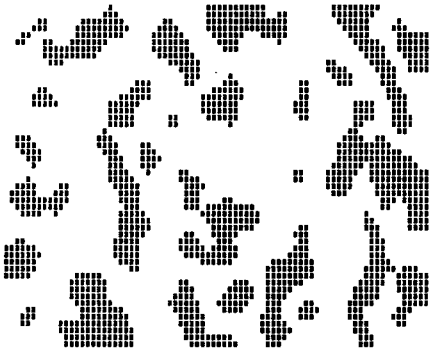


COMPARISON OF NUMBERS OF ATTEMPTED AND SUCCESSFUL TRANSPOSITIONS

NOTE THAT THE CURVE FOR RULE 1 RAPIDLY FLATTENS (AS THE DISCRETE CLUMPS APPEARING IN RESULT 1 BEGIN TO FORM); THE PROCEDURE BECOMES STUCK ON A LOCAL OPTIMUM. BUT PROCEDURES EMPLOYING RULES 2 AND 3 DO NOT PRODUCE SUCH PRONOUNCED CLUMPING, AND THEIR CORRESPONDING CURVES ARE MUCH STEEPER; THESE PROCEDURES DO NOT BECOME STUCK QUITE SO RAPIDLY.

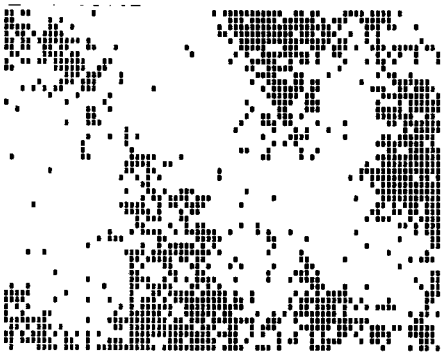


INITIAL RANDOM FIELD



CONFIGURATION PRODUCED EMPLOYING RULE 1. (204761 ATTEMPTS FOR 890 SUCCESSES)

CONFIGURATION PRODUCED EMPLOYING RULE 2. (15903 ATTEMPTS FOR 1500 SUCCESSES)



CONFIGURATION PRODUCED EMPLOYING RULE 3. (13859 ATTEMPTS FOR 1500 SUCCESSES)

Now imagine how we ourselves might go about manually solving the problem using each. Using representation 1, it looks formidable indeed, since there are no immediately obvious relations between the mapping which we see and the actions which we might take to improve it. We are faced with a large and intractable integer programming problem. Using representation 2, on the other hand, the problem becomes almost trivially simple, since actions which will result in improvement are directly evident as soon as we examine it (10).

3. THE AUTOMATED GENERATION OF ARCHITECTURAL FORM: SOME EXAMPLES

If a computer can be programmed to solve simple spatial arrangement on problems such as these, how effectively can it then also be programmed to solve much more complex problems, such as those encountered in architectural design? There is now quite an extensive literature on this question, and numerous programs (mostly concerned with plan layout) have been developed. It is worth examining some typical examples (11).

One popular approach has been to model floor-plan layout as an quadratic assignment problem. The first useful model along these principles appears to have been formulated by Koopmans and Beckman (12), who introduced the convention of dividing space into a number of discrete modules, or locations, then considered the problem of assigning the various spatial units to these modules in such a way as to satisfy a given set of constraints, and optimize some objective function. The objective is normally some function of distance and "interaction weight" between spatial units. Interaction weight is either some objective quantity, such as communication volume and cost, or is assigned subjectively on some scale. Interaction data is normally input in the form of an interaction matrix. Brotchie (13) gives a concise and clear description of this type of model.

Until recently, most attempts at development of space-planning programs followed this approach. Since there are no known feasible algorithms for optimal solution of large integer-programming problems of this type, the aim has been to develop heuristic procedures leading to reasonably good (although not necessarily optimal) solutions within reasonable computation time.

Approaches fall into two broad classes:

(1) Constructive procedures. These begin with an empty field, and proceed by locating one spatial unit after another, in succession (14).

(2) Improvement procedures. These commence with some given configuration, and attempt to modify it in order to produce a better configuration (15).

Various building types, in which communication costs are an important design factor, have been modelled in these terms: factories, offices (particularly bürolandschaft), warehouses, hospital floors, etc. This approach to floor-planning has not found widespread acceptance, however (outside of very narrowly specialized applications):

(1) Since it has not yet been possible to develop sufficiently powerful integer programming techniques for dealing with such large assignment problems successfully.

(2) Since most of the programs developed allow requirements to be specified only in a very crude, unsatisfactory, and incomplete way (normally only by interaction matrix).

(3) Since the internal (matrix) representations of built space which are used are generally far too crude to be useful, ignoring as they do doors, walls, and circulation systems, for instance.

An alternative type of model of floor planning, recently rather thoroughly investigated by Grason (16), is based on a dual linear graph representation of configurations. The potential usefulness of this particular scheme derives from its exploitation of a particular property of planar linear graphs; if nodes stand for rooms, and connecting line segments for required adjacencies, then this graph is the dual of another linear graph corresponding to a floor plan fulfilling those adjacency requirements. By introducing colour, direction, and weight on the edges, different types of adjacencies (north, sourth, east, or west), and length of floor plan wall segments may also be represented.

Several computational procedures operating on various linear graph representations to

produce floor layouts have been reported (17). These appear capable of producing good results with small problems, but great difficulties are encountered as the numbers of rooms increase. Proponents of this type of scheme seem to have devoted little discussion to the question of its adequacy and relevance in actual design situations. However, at least one investigator has suggested that it may be applicable in the generation of minimum-standard house plans.

Another approach, usually attempted for site-planning purposes, is based on the image of superimposition of successive coloured overlaps. A grid is superimposed on the site, th then for each module in the grid, numerical values corresponding to some measures of "suitability" for a particular building purpose are entered. Soil type, land cost, adequacy of drainage, degree of natural obstruction, and slope are commonly used. From these values, some type of combined "utility" for each module is computed, and buildings are located in those areas possessing greatest "utility" for building.

These overlay schemes mostly seem to depend on the assumption that the various different scales of "suitability" employed are independent and additive. Before much faith could be placed in them, it would need to be demonstrated that this very unrealistic assumption does not, in fact, produce unacceptable distortion of results. Examples of this approach are given in Willoughby (18) and Ward (19).

One further direction of development, which appears to possess great potential, has been suggested by repeated successes, since the early sixties, of attempts to build programs which can reasonably successfully play such complex games as chequers and chess. Such programs are generally rather complex themselves, written in list-processing languages (c.f., for instance, the assignment programs, which are almost exclusively coded in FORTRAN or similar languages) and are characterized by a capability to assess in considerable detail, and with considerable sophistication, the implications of a particular situation, and on this basis, to select and execute one of a wide range of available operations. Unlike more conventional programs, which are generally fairly limited in their branching, the branching of such programs is complex and unpredictable...responding to the complexity and unpredictability of situations encountered in the course of play. It is an intuitively appealing argument that similar flexibility of strategy will be necessary in any really successful space-planning program. Eastman (20) gives an excellent discussion, with examples, of the principles and possibilities.

4. LIMITATIONS OF OUR APPROACHES TO THE AUTOMATED GENERATION OF ARCHITECTURAL FORM

I think we might all agree that the programs described so far are generally quite limited in their conception and scope. This is representative of the current state of the art. It is useful to discuss these limitations in terms of the list of fundamental questions posed earlier.

4.1 DESIGN ELEMENTS

In certain types of problems, such as electrical circuit design, this question may involve few difficulties; the designer may simply be presented with a set of standard components to be combined together in some way. Since buildings appear generally to be composed of fairly similar basic parts and spaces, it seems reasonable to take a similar approach -- to treat architectural design as a set of decisions about locations and dimensions for various component parts of a building. Many of the programs discussed so far, for instance, have been concerned with combining sets of rectangles (i.e., "rooms") into plane configurations possessing certain desired properties. It is not difficult to imagine generalization of this approach to three dimensions and more complex forms (21).

But there are many possible schemes for subdividing the built environment into "atomic" components. We might accept the "rooms," "walls" and "buildings," etc. of conventional thought and speech, the "behavior settings" of Barker and his associates (22), the standardized components of a building system (23), the "patterns" of Alexander (24), or any of a host of other alternatives. Which is most appropriate as a basis for automated design procedures is far from clear at this point.

4.2 REPRESENTATION SCHEMES

Perhaps the most pressing difficulty is in finding an appropriate scheme for representation of the geometrical characteristics of buildings -- the sizes and shapes of spaces and forms. Since the early sixties there have existed, it is true, a number of computer graphics languages, using which it is fairly easy to describe quite complex forms, and display drawings of these forms, in various ways, on the face of a cathode ray tube.

Boundaries of physical components are defined by lists of line segments, lists of attributes of the component may be added, and operations such as insertion, deletion, combination, transformation, and rotation are readily performed. But whilst this approach is well-adapted to the task of generating displays, this type of data structure does not facilitate some common and important types of processing required in architectural design, for example, testing whether a given object will fit within a particular empty space (25).

An alternative approach, quite commonly enployed, is to use two or three-dimensional arrays, in which subscripts define Cartesian coordinates, and values define the element (room, physical component, empty space) occupying that position. Normally, cells are of equal size, but more elaborate schemes are also possible. This is a representation evidently much more satisfactorily oriented towards locational and arrangement problems of this type encountered in architectural design (note that almost all the programs discussed thus far use simple two-dimensional array representations), but it is equally clear that where the sizes of the arrays are even moderately large, then core and processing demands may become intolerable (26).

Further, less obvious, approaches to the representation of built form include the use of strings and linear graphs of various types (27). Linear graph techniques have proven advantageous as a means of representing assemblages of rectangles and rectangular parallelpipeds in a particularly economical fashion (28).

4.3 SOLUTION GENERATION

On the surface of it, it seems that we should be able to utilize many of the discoveries of modern applied mathematics in constructing very powerful and effection solution generation procedures. Spectacular successes in the use of optimization techniques in other fields suggest this. But a closer investigation of the question leads to some dismaying conclusions. The most powerful optimization techniques depend, for their successful operation, on our goals being capable of expression in particular ways (e.g., as a linear objective function), or on the space of proposed solutions possessing various desirable properties. We have simply failed, as yet, to find methods for modelling most of our important architectural and urban design problems in ways such that these techniques may be applied (29)

Since we can rarely apply conventional optimization techniques, the most fruitful approaches seem to be:

(1) A division of labor between generator and test such that a large search-tree is systematically explored, and most of the discriminatory power lies with the test.

(2) Use of heuristic programming techniques.

(3) Use of interactive systems.

Exhaustive search is the most obvious (and normally least practicable) example of the first approach. The trees are simply far too big. Most of the programs discussed so far have relied on the second, utilizing heuristic procedures of various degrees of sophistication. (The principles of such heuristic programming for design problems have been discussed by Eastman (30). But in many ways, the path of development of interactive systems, where the human operator generates and the machine tests, seems most promising.

If we opt for interactive design systems, there are some interesting questions of the division of tasks between man and machine to be considered. At one extreme, we may assign all the generation to the human operator, and all the testing to the machine; e.g., the INTU-VAL system described by Kamnitzer (31). Conversely, the machine can generate and present a sequence of alternatives for inspection and evaluation. The ALDEP floor-layout program, described by Seehof and Evans (32), essentially works on this principle (although it is not interactive). Finally, we may imagine the rather attractive alternative of a system which allowed for a considerable degree of flexibility in this division.

4.4 TESTS OF SOLUTIONS

We need to be able to specify test criteria in clear, complete, and unambiguous ways. In many problems of engineering or economics this is a relatively straightforward business. We might, for instance, simply say that certain variables should take on certain ranges of values, or that some objective function should be maximized or minimized. But in architectural design problems, several characteristic difficulties are encountered.

Firstly, we usually find that we have multiple objectives, and furthermore, some of

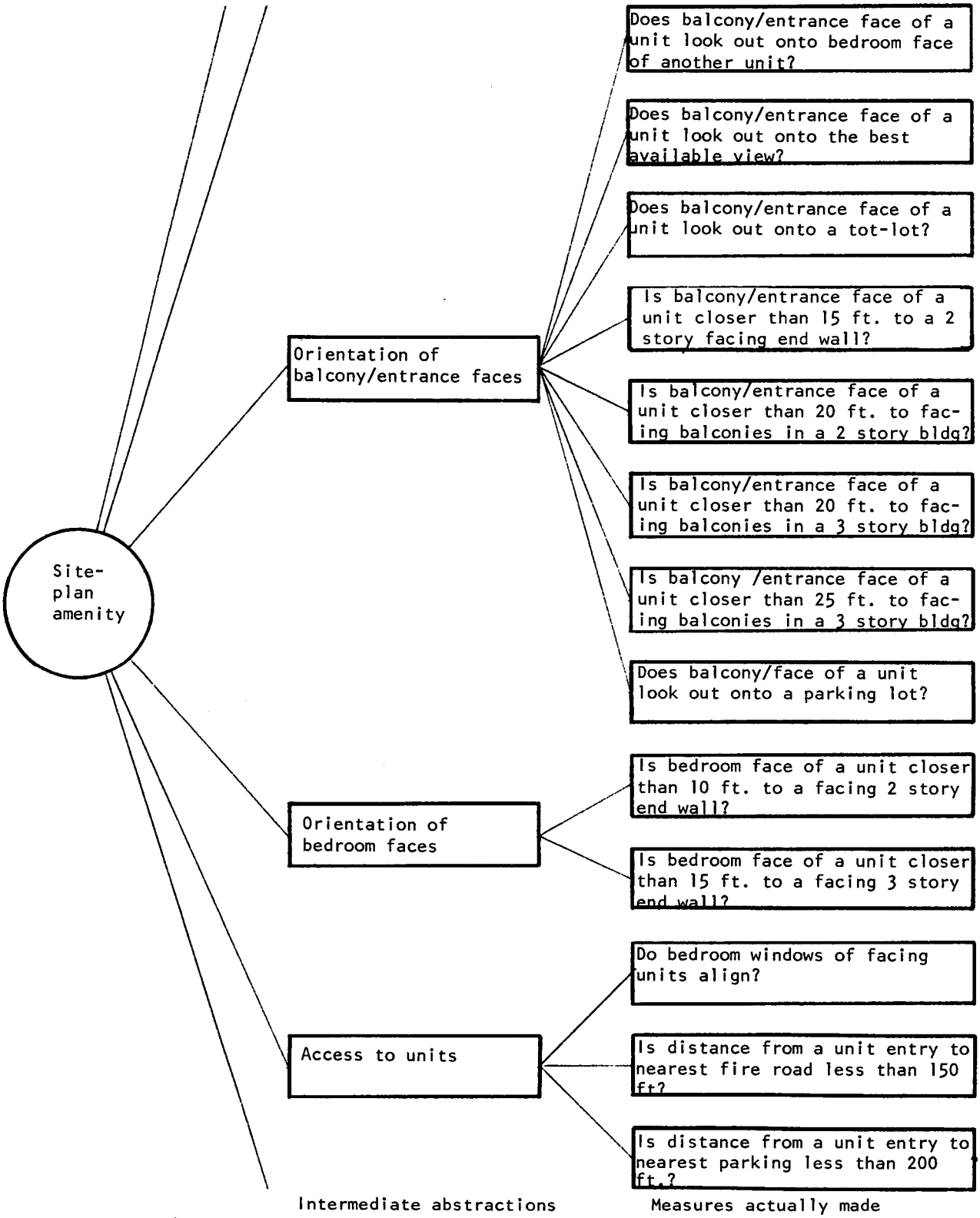


FIGURE 4: MEASURING SITE PLAN AMENITY

these objectives may be mutually contradictory, since all the various participants in a design situation are rather unlikely to have identical goals. In such situations, as the theory of games so elegantly demonstrates, we must often abandon the hope of discovering any unique "best" answer (33)

Secondly, we may not always be able to measure the variables that interest us directly, and we may be forced to rely on surrogates of various types. For instance, we might be concerned with a building's "safety," the best we can do is to check structural stability, protection of openings against intruders, adequacy of heating and ventilation, etc., and from our results infer whether an appropriate level of "safety" exists. Figure 4 shows in detail part of a set of surrogate measures of "site plan amenity" developed to form the basis of a site-planning computer program which I am currently developing. It is often far from easy to derive a set of surrogates by which to measure the quality of a physical environment, as the inadequacy of most building codes testifies. The technical problems involved in deriving and defining adequate sets of physical environmental quality measures have been discussed by Markus (34), Paul (35), and Wright (36). (Problems of a closely analogous nature are also encountered in broad social systems accounting. Gross (37) provides an extremely interesting discussion in that context.)

Even if, at any given point in time, we can resolve these difficulties, and give some fairly clear meaning to the notion of "answer" we may still encounter troubles, since our preferences and choices may of course be changeable and inconsistent. We do not always know what we want until we get it. Conversely, when we get what we want, we may decide that we did not want it after all. Frew (38) discusses preference shifts during the design process, and the ways in which these may affect the ultimate result.

4.5 DEALING WITH UNDER AND OVER SPECIFIED SITUATIONS

Unless a solution is truly uniquely specified, we need some mechanism for choosing between acceptable alternatives, or conversely choosing which constraints to disregard. This dilemma can be arbitrarily resolved, for instance by saying that we will accept the first, or the nth answer generated. An extremely popular resolution, which appears to have dominated much experimentation with computer-generated music, poetry, graphic design, and architectural design, is the use of procedures incorporating choice by random number generators. Hiller and Isaacson (39) discuss the types of choices which are made in musical composition, and the use of music-generating procedures based on random choice within a framework of defined rules. Borroff (40) reports on the generation of stanzas of poetry by a SNOBOL program incorporating grammatical and formal rules, and using words randomly selected from a limited, predefined vocabulary. Negroponte (41) describes some simple but fascinating devices, partially controlled by random number generators, which perform space and form design tasks.

There are considerable similarities between these procedures and certain of the compositional methods followed by musicians and poets. Studies of oral epic poetry, for instance, suggest that it depends on the restriction of the poet's means to a sparse and economical system -- great metrical regularity combined with the employment of a highly standardized vocabulary and set of oft-repeated formulas, lines, passages, and themes. The poet's task becomes largely one of making selections from amongst small and well-defined sets of alternatives (42). A comparison of the products of such machine and human procedures can be extremely interesting, as it forces us to focus very closely on the question of the distinction between form and content in works of art (43). It is quite conceivable that the machine procedures might (and in fact some do) produce "works" which are in some sense formally correct, or even elegant, but (unless we maintain an extreme formalist position that form becomes its own content), I think we must regard them as totally devoid of content and meaning. The fundamental difference is embodied in the difference in selection principles, one controlled by a human will and consciousness, and the other by an arbitrary mechanical device (44). Just how much of this human selection process is determinate, observable, and potentially imitatable seems a very open question.

5. CONCLUSION

I have discussed a wide range of technical and philosophical problems which we encounter in attempting to automate the solution of architectural problems. The obvious question now is, "Since it seems so difficult to program computers to perform these tasks, many of which human designers perform with comparative ease, why should we bother?" One answer is that such comparisons are premature and therefore irrelevant, since at this stage, we are really only groping for the first glimmerings of an understanding of the

potentials of the area. But we can also point to the possibility that a better understanding of the computer as a design medium will enable us to engage in exciting new modes of design. If, for example, we can develop systems which greatly speed the design process, and make it unnecessary for participants in that process to possess great technical skill, then more truly participatory design becomes possible. The goal is not so much to mechanize what we do now as to enable us to see the world in hitherto unimagined ways, to ask unasked questions, to explore unexplored possibilities in the spirit of Christian Morgenstern's eloquent piece of nonsense:

Es war einmal ein Lattenzaun,
mit Zwishenraum, hindurchzuschaun.

Ein Architekt, der dieses sah,
stand eines Abends plötzlich da --

und nahm den Zwishchenraum heraus
und baute draus ein grosses Haus...

One time there was a picket fence
with space to gaze from hence to thence.

An architect who saw this sight
approached it suddenly one night,

removed the spaces from the fence
and built of them a residence...

NOTES

(1) Politics, 1290, trans. Jowett.

(2) Frances Yates, The Art of Ramon Lull: An Approach to it Through Lull's Theory of the Elements, in Journal of the Warburg and Courtauld Institutes, Vol. 17, pp. 115-173 (1954).

Charles Babbage, Laws of Mechanical Notation, Pamphlet distributed at the Great Exhibition of 1851, reprinted in Morrison and Morrison (eds.) Charles Babbage and His Calculating Engines, Dover, 1961.

Franz Reuleaux, The Kinematics of Machinery, trans. A.B.W. Kennedy, Macmillan, London, 1876.

K.W. Norris, The Morphological Approach to Engineering Design in Jones and Thornley (eds.) Conference on Design Methods, Pergamon, 1963.

(3) Plato Meno 80, trans. Jowett.

(4) John McCarthy, The Inversion of Functions Defined by Turing Machines, in Shannon and McCarthy (eds.) Automata Studies, Princeton University Press, 1956, p. 177.

See also:

Herbert A. Simon, The Logic of Heuristic Decision Making in Nicholas Rescher (ed.), The Logic of Decision and Action, University of Pittsburgh Press, 1966.

(5) Herbert A. Simon, Style in Design, in Archea and Eastman (eds.) EDRA 2: Proceedings of the 2nd Annual Environmental Design Research Association Conference, Carnegie-Mellon University, Pittsburgh, October 1970.

(6) The game is manufactured by Parker Brothers, Inc., P.O. Box 900, Salem, Mass.

(7) Simon (1970) ibid.

(8) G.W. Graves and A.B. Whinston, An Algorithm for the Quadratic Assignment Problem, unpublished paper, Graduate School of Business Administration, University of California, Los Angeles.

(9) The theory of clumping in simple quadratic lattices is discussed in detail in S.A. Roach, The Theory of Random Clumping, Methuen, 1968.

(10) Simon (1966) ibid.

(11) The following paragraphs are summarized from an unpublished survey: William J. Mitchell, Summary of Approaches to Computer-Aided Space Planning, School of Architecture and Urban Planning, University of California, Los Angeles, February 1971.

(12) T.C. Koopmans and M. Beckman, Assignment Problems and the Location of Economic Activities, in Econometrica, Vol. 25, No. 1 (January 1957).

(13) John Brotchie, A General Planning Model, in Management Science, Vol. 16, No. 3 (1969).

(14) Examples are described in:

R.B. Lee and J.M. Moore, CORELAP -- Computerized Relationship Layout Planning in Journal of Industrial Engineering, Vol. 18, No. 3 (March 1967).

W.R. Spillers, An Algorithm for Space Allocation, in Proceedings: Fifth Annual ACM Urban Symposium, New York, August 1970.

(15) Examples are described in:

G.C. Armour and E.S. Buffa, A Heuristic Algorithm and Simulation Approach to the Relative Location of Facilities, in Management Science, Vol. 9, No. 2 (January 1963).

B. Whitehead and M.Z. Eldars, An Approach to the Optimum Layout of Single-Story Buildings, in Architects Journal, June 17, 1964.

(16) John Grason, Methods for the Computer-Implemented Solution of a Class of Floor-Plan Design Problems, unpublished Doctoral Dissertation, Carnegie-Mellon University, Department of Electrical Engineering, 1969.

(17) Grason, ibid.

Philip Steadman, The Automated Generation of Minimum Standard House Plans, in EDRA 2 ibid.
M. Krejcirik, Computer-Aided Plant Layout in Computer-Aided Design, Autumn 1969, pp. 7-19.

(18) T.M. Willoughby, Computer-Aided Design of a University Campus in Architects Journal, March 25, 1970.

(19) W.S. Ward, D.P. Grant, and A.J. Chapman, A PL/1 Program for Architectural Space Allocation in Proceedings: Fifth Annual ACM Urban Symposium, New York, August 1970.

(20) Charles M. Eastman, Problem Solving Strategies in Design, in Sanoff and Cohen (eds.), Proceedings of the First Annual Environmental Design Research Association Conference, Chapel Hill, June 1969.

(21) Philip Hendren, Experiments in Form, Using Computer Graphics, School of Architecture, University of Texas at Austin, 1968.

(22) Paul V. Gump, The Behavior Setting: A Promising Unit for Environmental Designers, Landscape Architecture, January 1971.

(23) William M. Newman, An Experimental Program for Architectural Design in Computer Journal, Vol. 9.

(24) Christopher Alexander, Sara Ishikawa, and Murray Silverstein, A Pattern Language Which Generates Multi-Service Centers, Center for Environmental Structure, Berkeley, California, 1968.

(25) For discussion of a typical line-segment based graphics system see:

Ivan E. Sutherland, Sketchpad: A Man-Machine Graphical Communication System in Proceedings: AFIPS 1963 Spring Joint Computer Conference.

The limitations of such systems for architectural purposes are discussed in:

Charles M. Eastman, Representations for Space Planning, in Communications of the ACM, Vol. 13, No. 4 (April 1970).

(26) Eastman, ibid.

(27) Eastman, ibid, and Grason, ibid.

(28) Lavette C. Teague, Network Models of Configurations of Rectangular Parallelpipeds, in Gary T. Moore (ed.) Emerging Methods in Environmental Design and Planning, M.I.T. Press, 1970.

(29) Similar problems at the urban design and planning scale are discussed in:

Britton Harris, Computation is Not Enough, in Proceedings: Second World Congress of Engineers and Architects, Tel-Aviv, December 1970.

(30) Eastman (1969) ibid.

(31) Peter Kamnitzer and Stan Hoffman, INTU-VAL: An Interactive Computer Graphic Aid for Design and Decision Making in Urban Planning, in EDRA 2, ibid.

(32) J.M. Seehof and W.O. Evans, Automated Layout Design Program in Journal of Industrial Engineering, December 1967.

(33) See for example:

Anatol Rapaport, Two-Person Game Theory, University of Michigan Press, p. 214.

(34) Thomas A. Markus, Optimization by Evaluation in the Appraisal of Buildings, in EDRA 2, ibid.

(35) W. Paul, Performance Specification:

Theory and Practice, in Architects Journal, 23 April 1969.

(36) James R. Wright, Performance Criteria in Buildings in Scientific American, Vol. 224, No. 3 (March 1971).

(37) Bertram M. Gross, The State of the Nation: Social Systems Accounting in Raymond A. Bauer (ed.) Social Indicators, M.I.T. Press, 1966, pp. 263-267.

(38) Robert S. Frew, The Nature of Preference Shift in Building Design, unpublished paper, Department of Systems Design, Waterloo University.

(39) Lejaren Hiller and Leonard Isaacson Experimental Music, in Sayre and Crosson (eds.), The Modelling of Mind, Simon and Schuster, 1968.

(40) Marie Borroff, Computer as Poet, in Yale Alumni Magazine, February 1971.

(41) Nicholas Negroponte, The Architecture Machine, M.I.T. Press, 1970.

(42) For a good brief description of the principles and process of oral composition see:

G.S. Kirk, The Songs of Homer, Cambridge University Press, 1962, pp. 55-101.

(43) Perhaps the best brief discussion of this question is given in:

Ernst Fischer, The Necessity of Art, Penguin, 1963, pp. 116-196.

(44) A.M. Turings classic Computing Machinery and Intelligence in Mind, Vol. LIX, No. 236 (1950) very briefly touches on the question of free will and random number generation.