EVOLVING BUILDING BLOCKS FOR DESIGN USING GENETIC ENGINEERING: A FORMAL APPROACH.

JOHN S. GERO AND VLADIMIR A. KAZAKOV
Key Centre of Design Computing,
Department of Architectural and Design Science,
The University of Sydney, NSW 2006 Australia.
e-mail:{john,kaz}@arch.su.edu.au

Abstract. This paper presents a formal approach to the evolution of a representation for use in a design process. The approach adopted is based on concepts associated with genetic engineering. An initial set of genes representing elementary building blocks is evolved into a set of complex genes representing targeted building blocks. These targeted building blocks have been evolved because they are more likely to produce designs which exhibit desired characteristics than the commencing elementary building blocks. The targeted building blocks can then be used in a design process. The paper presents a formal evolutionary model of design representations based on genetic algorithms and uses pattern recognition techniques to execute aspects of the genetic engineering. The paper describes how the state space of possible designs changes over time and illustrates the model with an example from the domain of two-dimensional layouts. It concludes with a discussion of style in design.

1. Introduction

There is an increasing understanding of the role that a design language and its representation play in the efficiency and efficacy of any design process which uses that language (Coyne *et al.,* 1990; Gero *et al.,* 1994). A recurring issue is what is the appropriate granularity of a language. If building blocks which constitute the elements of a design map onto a design language then the question becomes what is an appropriate scale for those building blocks in terms of the final design. At one extreme we have parameterised representations where the structure of a design is fixed, all the variables which go to define a design are predefined and what is left is to determine the values of those variables. This defines a very small design space, small in terms of all the possible designs which might be able to be produced for that design situation. At the other extreme we have elementary building blocks which can be combined in a very large variety of ways and which, as a

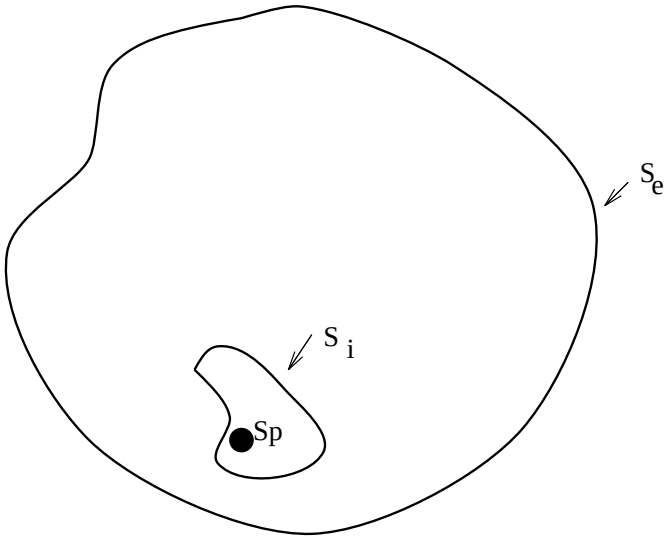

Figure 1. S_e is the design space produced by all the possible combinations of the elementary building blocks, S_p is the design space produced by all the combinations of the values of the parameterised variables, S_i is the design space of interesting designs for the design situation.

consequence define a very large design space, the vast part of which covers designs which are likely to be uninteresting in terms of the current design situation. The designs produced by the parameterised design representations are a subset of those capable of being produced by the elementary building block representation, Figure 1. Examples of building block representations include constructive systems such as design grammars as exemplified by shape grammars (Stiny, 1980b). Examples of parameterised variable representations include a wide variety of design optimization formulations (Gero, 1985).

The advantage of the use of the elementary building blocks representation is the coverage of the entire design space they provide, whereas the advantage of the parameterised variable representation is the efficiency with which a solution can be reached.

We present here a formal approach which generates a targeted representation of a design problem. A targeted representation is the one which closely maps on to the problem at hand. As an example consider a layout planning problem in architectural design. One representation may be at the material molecular level, where molecules can be combined to make a variety of materials and particular combinations in space produce physical objects; here the potential solution space includes designs which bear no relations to architecture. A targeted representations may be to represent rooms such that the potential solution space primarily includes designs which are all recognizably architectural layouts.

In order to simplify our analysis we consider designs which are assembled from

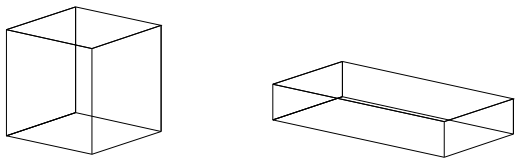

Figure 2. The set of building blocks for Froebel's kindergarten gifts (Stiny, 1980b).

some finite collection of spatial elements (we call them *building blocks* or *components*) along with assembly rules. It is assumed that the assembly rules do not affect the components - the design object is a union of non-overlapping building blocks. We start with some set of building blocks which we call *elementary* components. It is assumed that they cannot be decomposed into any smaller ones. We call a set of building components and assembly rules a *representation* of the design problem and the set of elementary components and corresponding rules the *basic* representation. We call it a representation because it implicitly defines the set of all designs (design state space) which can be produced using this set of building blocks and assembly rules.

The kindergarten gifts of Froebel (Stiny, 1980b) is a typical example of such types of design problem. One of many possible elementary representations and assembly rules for it is shown in Figures 2 and 3. One can easily extend it by adding further elementary building blocks and/or further assembly rules.

Targeted representations

A great variety of designs can be produced within a basic representation. Usually the designer is interested in some particular class of designs. Assume we have some additional set of composite building blocks and an additional set of assembly rules to handle them. We can calculate the number of these composite building blocks which can be found in all possible designs in that particular class and the number of elementary building blocks used to build the rest of these designs (each elementary building block should be counted only once as a member of composite building or elementary building block, the largest composite blocks are counted first and the elementary blocks are counted last). Then we can calculate the frequency of usage of these composite building blocks and elementary building blocks in the entire design space. The same values can be calculated for all designs which have the property or properties we are interested in. If the frequency of the usage of the composite building blocks is much higher for the designs of interest than for all designs built from the elementary building block and the frequency of elementary components usage is much lower than that of the composite building blocks for the design space of interest then we can use the composite building blocks instead

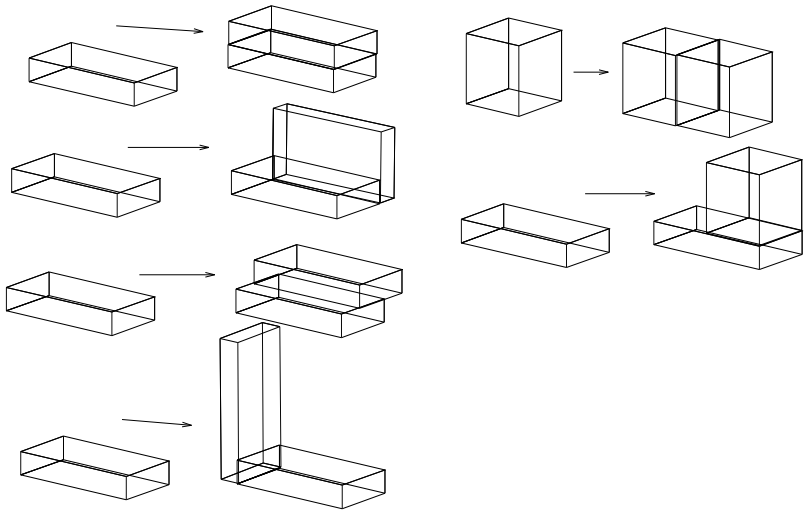

Figure 3. The set of six assembly rules for Froebel's kindergarten gifts.

of elementary one to produce designs of interest with much higher probability. In other words a representation exists which maps into the area of interest of the entire design space. Let us call it the *targeted* representation for the particular class of designs. Obviously different targeted representations can be produced which correspond to different sets of composite building blocks. We characterize these representations by their "complexity" which is defined recursively as: 0-complexity for the basic representation, 1-complexity for the representation whose building blocks are assembled from elementary building blocks, 2-complexity for the representation whose building blocks are assembled from the building blocks of 0-complexity and 1-complexity, etc. Assume an evolution occurs in an abstract space of complex representations: initially only elementary building blocks exist then a cycle proceeds when a new set of composite building blocks is produced from the ones which are currently available. Then a representation of i-complexity (and building blocks of i-complexity) simply means that composite building blocks of this representation have been produced during i-th step of this evolution.

Different composite building blocks of the same i-complexity may contain different numbers of elementary building blocks: for example, assume some building block of 3-complexity contains 3 elementary building blocks and one of the composite building blocks of 4-complexity is assembled from 2 building blocks of 3-complexity and thus contains 6 elementary components and another one is assembled from one block of 3-complexity and one block of 0-complexity and thus contains 4 elementary components. It is also clear that because there are different ways to assemble the same composite building block it may be produced multiple

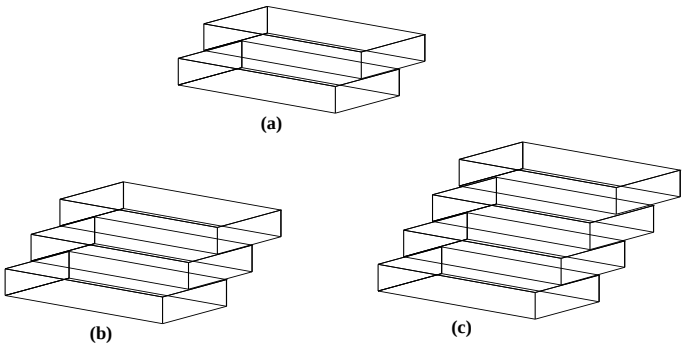

Figure 4. The set of composite building blocks of different complexity for building a staircase; (a) 1-complexity, (b) and (c) 2-complexity.

times in representations of different complexity level during the evolution.

The search for a reasonably good design using the basic representation is very difficult because significant part of the search effort is wasted in the search of unuseful parts of the design space. If the targeted representation is used instead of elementary one the probability of producing designs of interest becomes much higher, the design space becomes smaller and the design problem less complicated and easier to deal with. The approach presented in this article automatically generates the hierarchy of more and more complex building blocks (in general); ones which are more and more close to the targeted representations which are capable of producing better and better designs.

Assume we work with the representation of the kindergarten blocks shown in Figures 2 and 3 and are trying to design a two-level building with walking access from one floor to the next. The search for a design with this property is quite difficult because only a very small fraction of all feasible objects exhibits it and the probability of discovering the combination of building blocks which makes a staircase during the search is low. However, if we add a composite object of 1-complexity (Figure 4) and corresponding assembly rules Figure 5 to the representation we increase this probability, and if we add a composite building block with 2-complexity (Figure 4) then this probability increases further.

Genetic engineering

Genetic engineering, as used in this paper, is derived from genetic engineering notions related to human intervention in the genetics of natural organisms. In the genetics of natural organisms we distinguish three classes: the genes which go to make the genotype, the phenotype which is the organic expression of genotype, and the fitness of the phenotype in its environment. When there is a unique identifiable fitness which is performing particularly well or particularly badly amongst all the fitness of interest we can hypothesize that there is a unique cause for it and

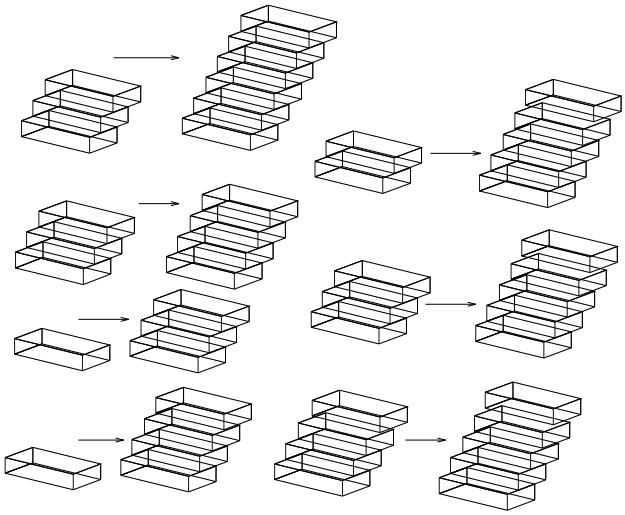

Figure 5. The set of additional assembly rules for handling composite building blocks.

that this unique cause can be directly related to the organism's genes which appear in a structured form in its genotype. Genetic engineering in concerned with locating those genes which produce the fitness under consideration and in modifying those genes in some appropriate manner. This is normally done in a stochastic process where we concentrate on populations rather than on individuals.

Organisms which perform well (or badly) in the fitness of interest are segregated from these organisms which do not exhibit that fitness or do so only in a minimal sense. This bifurcates the population into two groups. The genotypes of the former organisms are analysed to determine whether they exhibit common characteristics which are not exhibited by the organisms in the latter group (Figure 6). If they are disjunctive, these genes are isolated on the basis that they are responsible for the performance of the fitness of interest. In natural genetic engineering these isolated genes are either the putative cause of positive or negative fitness. If negative then they are substituted for by "good" genes which do not generate the negative fitness. If they are associated with positive fitness they are reused in other organisms. It is this later purpose which maps on to our area of interest.

One can interpret the problem of finding the targeted set of building blocks as an analog of the genetic engineering problem: finding the particular combinations of genes (representing elementary building blocks) in genotypes which are responsible for the properties of interest of the designs and regular usage of these gene clusters to produce designs with desired features.

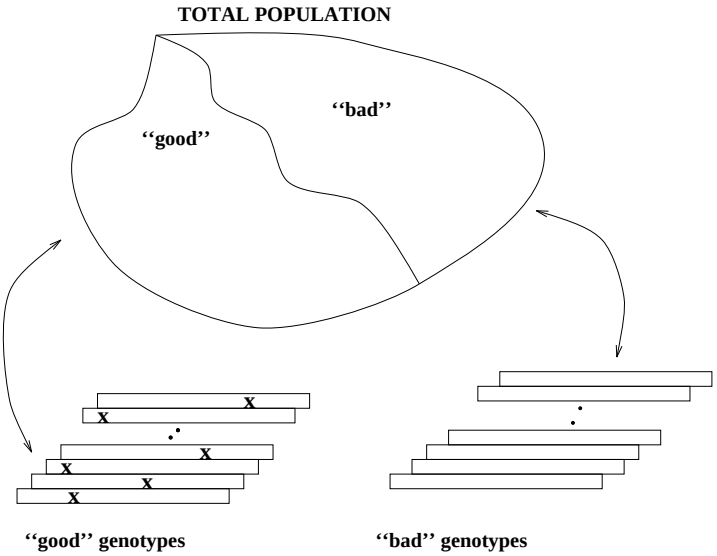

Figure 6. The genotypes of the “good” members of population all exhibit gene combinations, X, which are not exhibited by the genotypes of the “bad” members. These gene combinations are the ones of interest in genetic engineering.

2. Building blocks

Thus, we establish that different building blocks define different design state spaces (which are, in their turn, the subsets of the entire basic design space). More formally we assume that for the design space of interest a set of composite building blocks exists which is sufficient to build any design of interest from it (or which are sufficient to build a significant part of any of these designs of interest).

We search for these building blocks using the consequence of the assumption made in the introduction about frequencies of composite components usage: on average the sampling set of designs with the desired characteristics (the “good” ones) contains more of such composite building blocks than the sampling set of designs that do not have these characteristics (the “bad” ones). In some cases it is even true in a deterministic sense - that only the designs which can be built completely from some set of composite building blocks possess the objective characteristics, all the rest of the entire basic state space does not have them. One can easily come up with corresponding examples.

In the next section we describe an evolutionary algorithm which generates “good” and “bad” sampling sets using the current set of building block (set of elementary block at the beginning) and use genetic engineering concepts to determine new composite blocks which are closer to the “targeted” ones than the current set of building blocks. These two steps proceed in cycle while the “good” sampling set converges to the sampling set from the desired design state space and the set of

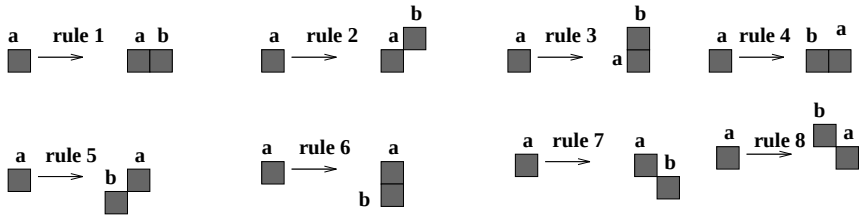

Figure 7. The assembly (transformation) rules used in the example.

complex building blocks comes closer and closer to the targeted set.

If the basic assumption about more frequent use of some composite building blocks to generate the particular class of designs is not true for some problem then the targeted representation for this problem does not exist and the algorithm which is proposed below will not generate an improved representation but will be equivalent to the algorithm for solving the routine design problem (Gero and Kazakov, 1995) and will simply generate the improved designs.

If the sequence of assembly actions is coded as a real vector then the problem of finding the complex building blocks becomes the problem of finding the key patterns in the coding vector - the combinations of codes within it which are likely to be associated with the property of interest in the designs. The vast arsenal of pattern recognition methods can be used to solve this problem. Essentially they are just search methods for subsets in a coding sequence which on average are more frequently observed in objects with desired characteristics than in the rest of the population.

Let us illustrate the execution of the cycle just outlined using a simple 2-dimensional graphical example. We will describe it in more detail later but for now on it is sufficient to say that there is only one elementary block here - the square and that a design is assembled from cubes using the 8 rules shown in Figure 7. Any design can be coded as a sequence of these rules used to assemble it. Assume we are trying to produce a design which has the maximum number of holes in it and that each design contains not more than 20 squares. We start the cycle by generating a set of coding sequences and corresponding designs Figure 8. Then we notice that a number (4) of the designs have the maximal number of holes (designs 1, 2, 4, and 7 - the "good" sampling set) contain the composite building block A and that for three of them their coding sequences contain the pattern $\{2, 8, 5\}$. We also notice that only a few (none in this case) of the designs without holes (designs 3,5,8 and 10 - the "bad" sampling set) contain this block and none contain this pattern in their coding sequence. Then we can generate the next population of coding sequences using the identified sequence $\{2, 8, 5\}$ as a new rule which uses the composite building block A in the design. Assuming that we employ some optimization method to generate this new population we can expect that the "good" sampling set from the new pop-

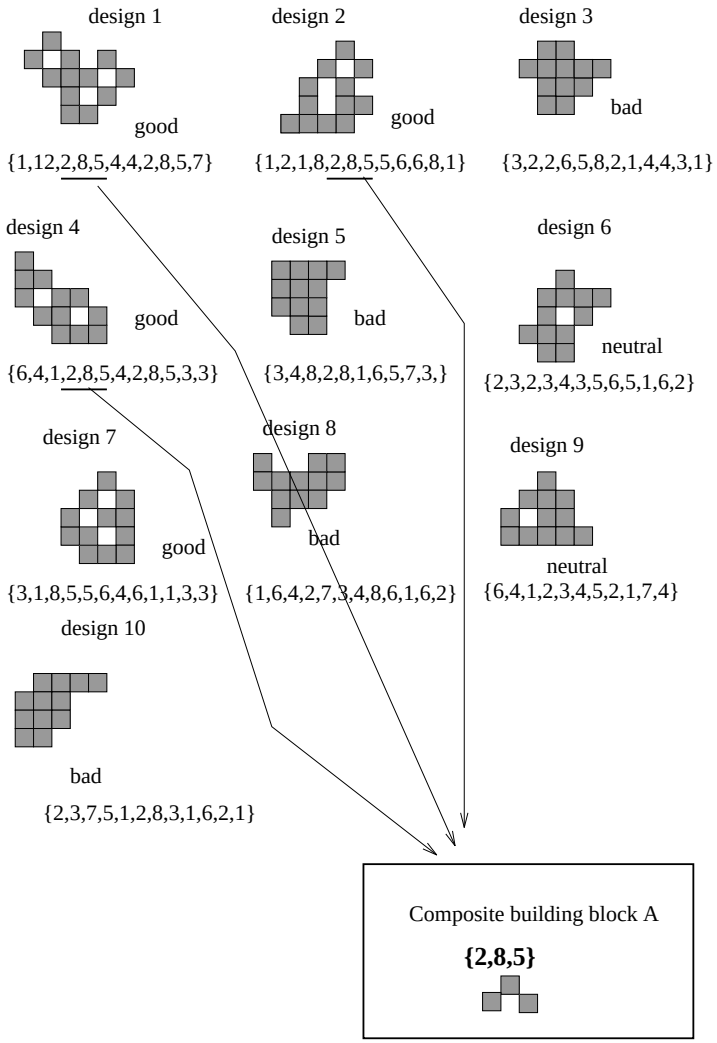

Figure 8. The identification of the pattern $\{2, 8, 5\}$ and corresponding composite building block A in the genotypes of "good" designs.

ulation is better than the previous one (that is, the designs which belong to it have on average more holes than the ones from the previous "good" sampling set). Then we again try to identify the patterns which are more likely to be found in designs from this "good" sampling set than from the "bad" one. This time these patterns may contain the previously identified patters as a component. Then we generate a new population of designs using these additional pattern sequences of rules as an additional assembly rule and so on.

The sizes of the sampling sets in realistic systems is likely to be much larger

than the ones in this example and much more sophisticated techniques (Pearson and Miller, 1992) should be employed to single out these key patterns.

3. Evolving building blocks

For a more formal analysis of the evolution of the building blocks we use the shape grammar formalism (Stiny, 1980a). It consists of an ordered set of initial shapes and an ordered set of shape transformation rules which are applied recursively. A particular design x within the given grammar is completely defined by a control vector v which defines the initial shape and transformation rules applied at each stage of recursive shape generation. According to the discussion in the Introduction we consider a particular class of shape grammar similar to the kindergarten grammar (Stiny, 1980b), where any shape is a non-overlapping union of building blocks and feasible shape transformations are addition, replacement or deletion of the building blocks.

Let $B = \{b_0, b_1, \ldots, b_n\}$ be a set of n currently available building blocks, and $R = \{r_0, r_1, \ldots, r_m\}$ be a set of m assembly rules applicable to these blocks. Then the control vector $v^i = \{b^i, r_1^i, r_2^i, \ldots, r_{N_i}^i\}$, $b^i \in B$, $r_j^i \in R$, $j = 1, \ldots, N_i$, $i = 1, \ldots, M$ defines the population of M designs $x(v^i)$, $i = 1, \ldots, M$. The length of the control vector $\{v^i\}$, N_i is a variable.

If we add new complex building block $b_{n+1} = x(\{b^{n+1}, r_1^{n+1}, r_2^{n+1}, \ldots, r_k^{n+1}\})$ and new assembly rules $r_{m+1}, \ldots, r_{m+l^{n+1}}$ for its handling then we get a new extended set of rules $R = R \cup_{j=1,l^{n+1}} \{r_{m+j}\}$, $B = B \cup \{b_{n+1}\}$, $n = n + 1$ and $m = m + l^{n+1}$. Now we can produce the design $x(v)$ which corresponds to the vector v whose components belong to the extended B and R. Note that the additional building blocks and assembly rules are generated recursively: they are completely defined in terms of the previous R and B.

We assume that the design problem has a quantifiable objective vector-function $F_k(x)$, $k = 1, \ldots, p$ and can be formulated as optimization problem

$$F(x(v)) = F(v) \to \max \tag{1}$$

The problem (1) over the representation with a fixed set of building components and assemble rules can be solved using any of optimization methods (Gero and Kazakov, 1995) but the stochastic algorithms like genetic algorithms (Holland, 1975) and simulated annealing (Kirkpatrick *et al.,* 1983) look most promising at the moment. We have chosen the genetic algorithm.

The evolutionary method has the following structure :

Algorithm

(0). **Initialization.** Set counter of iteration $k = 0$. Take the set of elementary building blocks $B = \{b_0, \ldots, b_n\}$ and corresponding assembly rules R. Generate some

random population of $v^{i,0}$, calculate $x(v^{i,0})$ and $F(x(v^{i,0}))$. Set the relative thresholds for the design's ranking $0 < A_b < A_g < 1$; they are used during an evolution stage to divide the design into "good", "bad" and "neutral" sampling sets, that is, the parts of population which exhibit (A_g) best, (A_b) worse and intermediate relative fitness level.
(1) **Evolution of complex building blocks.** For every component of the objective function F_k divide the population into 3 groups :
"good" ($F_k(x) > \max_{i=1,M} F_k(x_i) - A_g * (\max_{i=1,M} F_k(x_i) - \min_{i=1,M} F_k(x_i))$,
"bad" ($F_k(x) < \min_{i=1,M} F_k(x_i) + A_b * (\max_{i=1,M} F_k(x_i) - \min_{i=1,M} F_k(x_i))$,
and "neutral" (the rest of population).
Determine J combinations $b_{n+j} = x(\{b^j, r_0^j, r_1^j, \ldots, r_{k^j}^j\})$, $j = 1, \ldots, J$ of the current building blocks which distinguish the "good" sampling set from the "bad" one statistically significantly using any one of the pattern recognition algorithms. Add it to the current set of building blocks $B = B \cup_{j=1,J} \{b_{n+j}\}$. Add corresponding new assembly rules to R.
(2) **Generation of new population.** Compute new population using available information about current population $v^{i,k+1} = G((v^{i,k}, x(v^{i,k}), F(x(v^{i,k}))$. The components of $v^{i,k+1}$ belong to the new extended B and R. The G depends on the optimization method employed. If the genetic algorithm has been chosen then $v^{i,k+1}$ is to be calculated using standard crossover and mutation operations. Because the updated grammar includes the grammar from the previous generation the search method guarantees that the new population is better than the previous one (at least no worse) and the new "good" sampling set is closer to sampling set of the design state space of interest.
(3) Repeat steps (1) and (2) until the stop conditions are met.

The stop conditions usually are the termination or slowing down of the improvement in F and/or the end of new building blocks generation.

4. Example

Evolving the targeted representation
As an example we take the problem of the generation of a 2-dimensional block design on a uniform planar grid (derived from (Gero and Kazakov, 1995)). There is just one elementary component here - a square and the eight assembly rules (transformation rules in terms of a shape grammar) which are shown in Figure 7. If the position where the current assembly rule tries to place the next square is already taken then all the squares along this direction are shifted to allow the placement of new square. It is assumed that the transformation rule at the i-th assembling stage is applied to the elementary block added during the $(i-1)$-th stage. The characteristics of interest are geometric properties of the generated design. In order to demonstrate the idea, assume that the generated design can not consist of more than 32

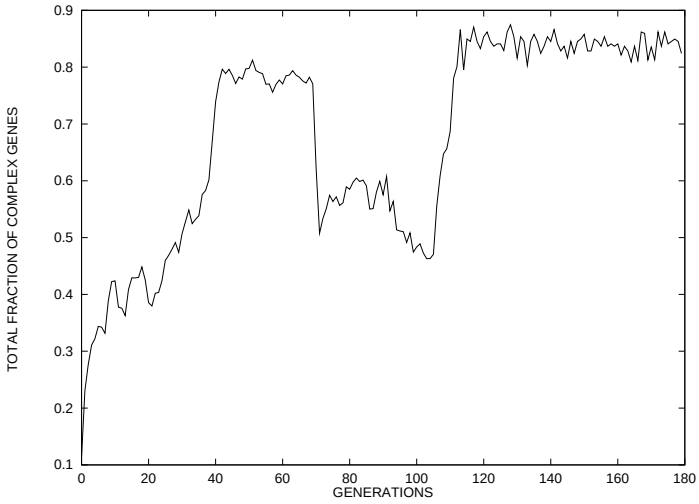

Figure 9. The fraction of composite building blocks in the total pool of building blocks used to assemble the population vs. generation number. The objective function has two components: the area of closed holes and the number of connections between holes and the outside space. The initial set of building blocks contains only elementary building blocks. Evolution proceeds until it naturally dies off.

elementary components. We generate a new population during the stage (2) of the Algorithm using the modification of the simple genetic algorithm tailored to handle multidimensional objective functions (Gero and Kazakov, 1995). We implement a very simple pattern recognition algorithm based on the statistical frequency analyses of double and triple element building blocks with a high cut-off threshold for the acceptance of the patterns. For more complex systems more sophisticated technique is needed.

During the first iteration we begin with the set of building blocks which contains only the elementary ones and search for the designs with maximal area of enclosed holes and maximal number of connections between the holes and outside space. The evolution was allowed to proceed until a stable condition was reached. The result are shown in Figures 9 and 10. By plotting the fraction of the complex building blocks in the total pool of building blocks used to assemble the population at different generations Figure 9, one can see how complex building blocks become dominant and how its fraction reaches a stable level after 110-120 iterations. The fractions of building blocks of different complexity in the total pool at different generation are shown in Figure 10. One can see that during the first 40 generations the total fraction of composite building blocks arises monotonically. For the first 10 generations this rise is completely provided by the increasing number of 1-complexity composite building blocks in the population. Then (from 15 to 30 generations) the fraction of 1-complexity blocks remains stable but the number of 2-complexity building blocks increases and provides the continuing increase

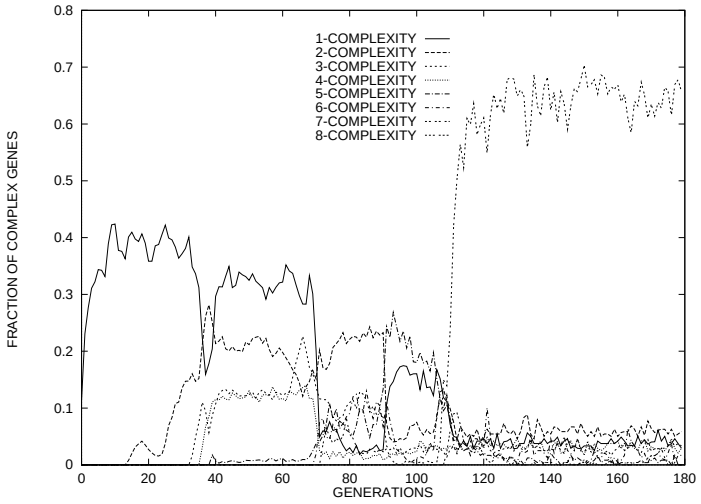

Figure 10. The fraction of composite building blocks with different complexities in the total pool of the building blocks used to assemble the population vs. generation number. This figure shows the building blocks of different complexities which are summed to produce the total fraction shown in Figure 9.

in the total fraction of composite building blocks. From generations 40 to 70 this total fraction is stable with approximately half of building blocks of 1-complexity and half of 2, 3 and 4-complexities. Then the number of 1-complexity blocks and total number of complex blocks declines sharply and from 70 until 110 generation a transitional process occurs with a complex redistribution of populations between representations with different complexities. At the end of this period the building blocks of 8-complexity saturate the population when the fractions of the other complex building blocks are shifted towards a noise level only. One of the evolution paths in the space of complex building blocks is shown in Figure 11 (a). Some of the designs produced are shown in Figure 11 (b). Here arrows show which previously evolved composite building blocks are used to assemble the new building block. The 0-complexity block and its contributions are omitted. As we already noted composite blocks of the same complexity level sometimes have different numbers of elementary components. Coincidently, the 5-complexity block is reproduced again in the representations of 6-, 7- and 8- complexities and is one of the dominant blocks at the end of the evolutionary process.

Using targeted representation.

The set of targeted building blocks evolved during this process is then used as an initial set of building blocks during the second experiment when we produce the designs with maximal total area of holes inside and maximal number of connections between these holes inside the structure. Here the fitnesses are close to but

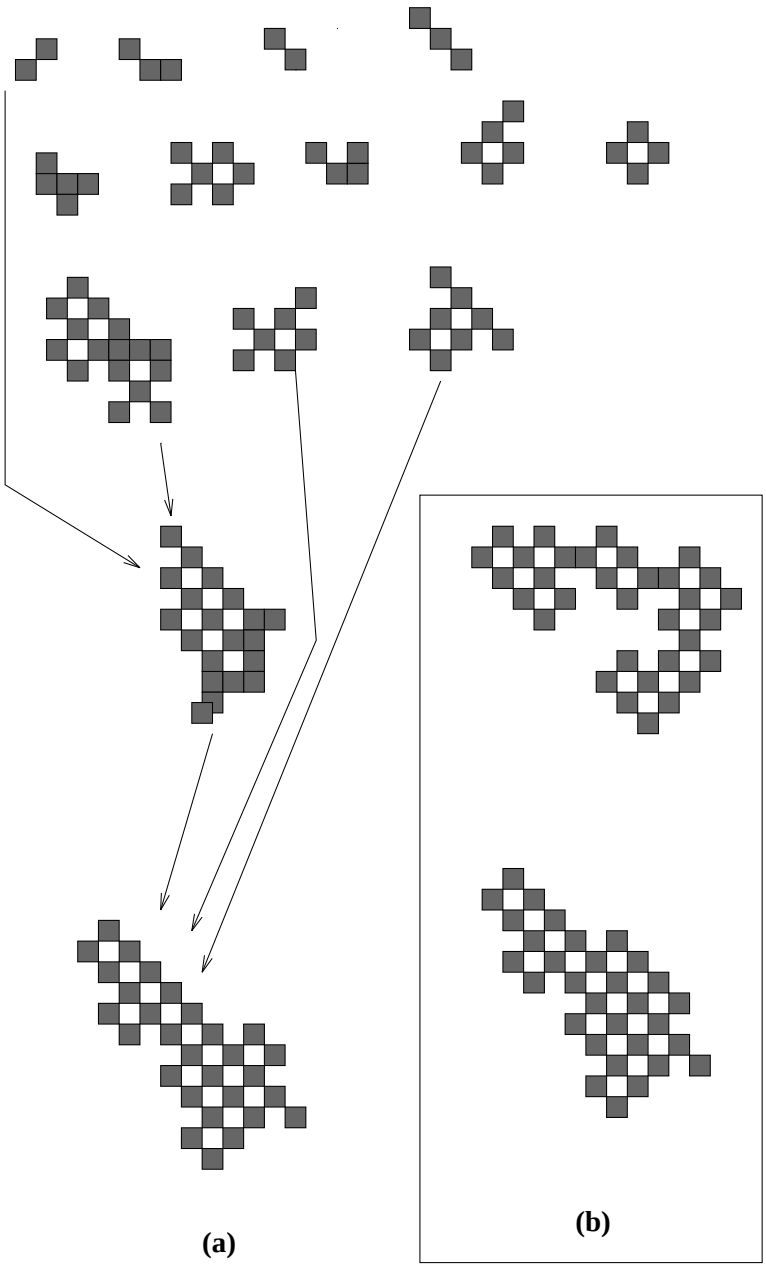

Figure 11. (a) An example of the evolutionary paths in the evolution of a complex building block, (b) some of the designs produced using the set of evolved complex blocks.

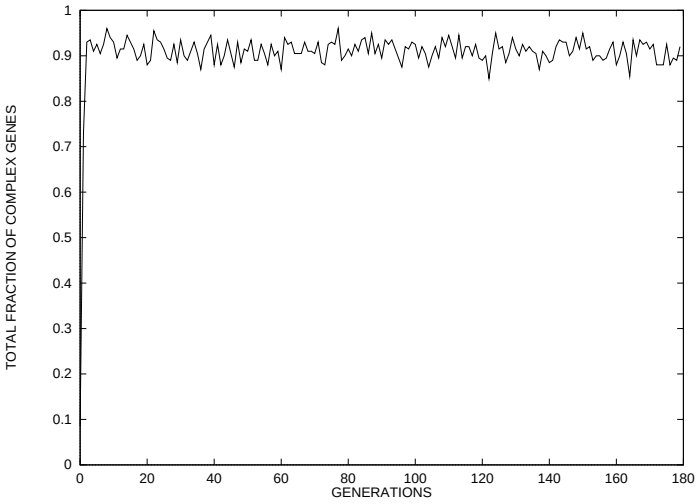

Figure 12. The fraction of composite building block in the total pool of building block used to assemble the population vs. generation number . In this experiment the objective function is the number of closed holes and the number of connection between the closed holes inside the structure. The initial set of building blocks is inherited from the first experiment and is the targeted representation.

not the same as those used to evolve the targeted representation. This experiment is used to test whether the targeted representation is likely to be used more than the original, elementary building blocks. If the targeted representation is used rather than the elementary building blocks then we have achieved our goal of evolving a representation can be used to produce designs which exhibit desired characteristics more readily. The results are shown in Figures 12 and 13. One can see that the fraction of the composite building blocks used to produce these designs reaches the saturation level during the first few iterations. The visible redistributions of the population between the composite building blocks of 5, 6 and 7-complexities are purely superficial - this redistribution occurs between the same composite building blocks which are present in all these representations. Evolution of the representation does not occur during this experiment - no new complex building block were evolved. This can be interpreted as an indication of closeness of the targeted representations for both problems. So if the targeted representation is evolved for one set of objectives then it can be usefully applied to any of the objective sets which are only slightly different to it.

Effects of incomplete evolution

In this experiment we repeat the first iteration but stop the evolution prematurely after only 60 generations. After this we repeat the second iteration using the evolved incomplete set of composite building blocks. The results are shown in Figures 14

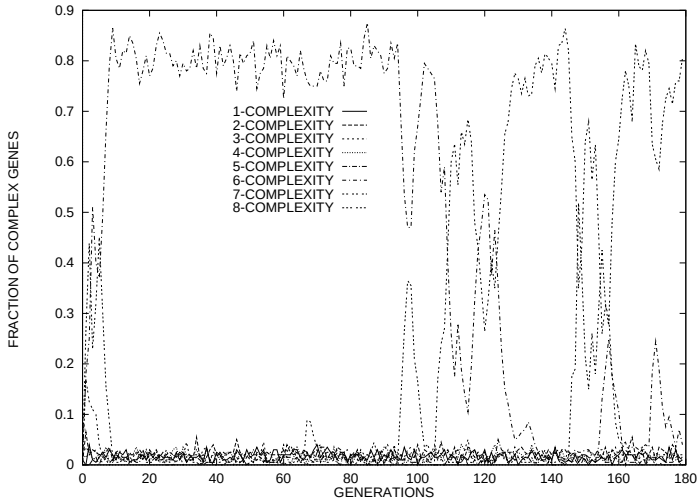

Figure 13. The fraction of composite building blocks with different complexities in the total pool of the building block used to assemble the population vs. generation number in the experiment.

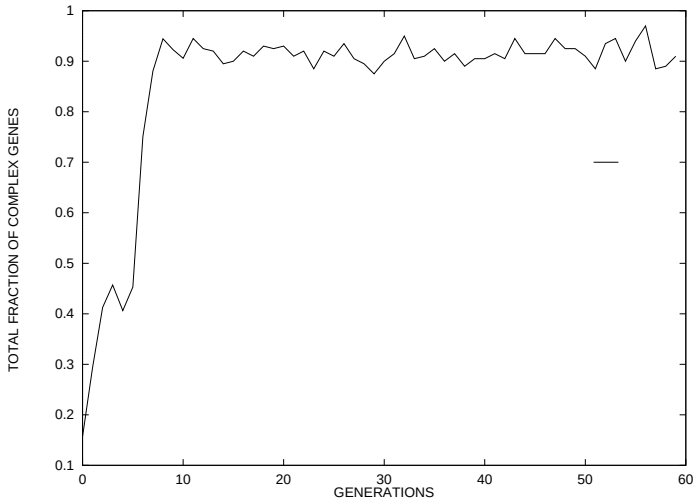

Figure 14. The fraction of composite building block in the total pool of building block used to assemble the population vs. generation number . In this experiment the objective function is the number of closed holes and the number of connections between the closed holes inside the structure. The initial set of building blocks is inherited from first iteration which has been prematurely terminated at generation 60.

and 15. In this case the evolution of the representation continues for about a further 10 generations and we end up with the same set of evolved composite building blocks. The saturation of the population with the composite building blocks is also completed after these 10 generations. Thus, one can start to evolve a representa-

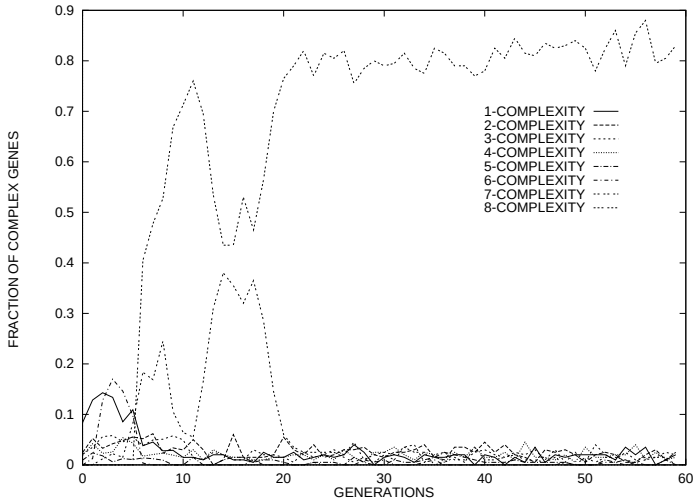

Figure 15. The fraction of composite building blocks with different complexities in the total pool of the building block used to assemble the population vs. generation number in the third experiment.

tion for one set of objectives and then continue it for another closely related set of objectives.

If we commence by treating the problem as one of finding improved designs then from a computational viewpoint this form of evolution speeds up the convergence to improved designs by up to 15% (in terms of the number of generations required) when compared with standard genetic algorithms. It appears that the use of a targeted representation can lead to the production of designs which are locally optimal.

However, if we use the completion evolution approach presented in the second experiment we get further improvements in performance. We will leave to the Discussion section further discussion of the other advantages of the approach presented.

5. Discussion

The analysis just presented can be easily extended to include general object grammars of types different to the kindergarten grammar. The proposed approach can be considered as an implementation of the simplest version of the genetic engineering approach to the generic design problem. From the technical point of view the algorithm presented is a mixture of a stochastic search method (which may be a genetic algorithm) and a pattern recognition technique.

The genetic engineering approach can be applied in a similar fashion to the problem of the generation of a "suitable" shape grammar (Gero and Kazakov, 1995) where the complex building blocks correspond to the evolved grammar rules.

As already mentioned in the analysis of the numerical experiment, the evolved representations are highly redundant - the same composite building blocks are evolved many times along the different branches of the evolutionary trees. The redundancy level of the current set of composite building blocks can be reduced in a number of different ways. The simplest is just to delete all the redundant copies from the current set. In the general case, we have to find the minimal representation of the subspace which can be generated using the current set of complex building blocks.

The introduction of ideas and methods from genetic engineering into design systems based on genetic algorithms opens up a number of avenues for research into both evolutionary-based design synthesis and into modified genetic algorithms. In design systems based on such modified genetic algorithms it is possible to consider two directions.

The first is to treat the sequence of the genes which results in certain behaviours or fitness performances as a form of 'emergence', emergence of the schema represented by that gene sequence. The use of the genetically engineered complex genes changes the properties over time of the state spaces which are being searched. This allows us to consider the process as being related to design exploration modelled in a closed world. The precise manner in which the probabilities associated with states in the state space change is not yet known. Clearly, this is also a function of whether a fixed length genotype encoding is used or not. If a variable length genotype encoding is used with the genetically engineered complex genes then the shape of the state space remains fixed but the probabilities associated with the states within it change. If a fixed length genotype encoding is used with the genetically engineered complex genes then the shape of the state space changes in addition to the probabilities associated with states in the state space.

The second is to treat the genetically engineered complex genes as a means of developing a representation for potential designs. A fundamental part of designing is the determination of an appropriate representation of the components which are used in the structure (Gero, 1990) of the design. This is part of that aspect of designing called 'formulation', ie the determination of the variables, their relationships and the criteria by which resulting designs will be evaluated. In most computer-aided design systems the components map directly on to variables. Further, in such systems the variables are specified at the outset, as a consequence there is an unspecified mapping between the solutions capable of being produced and the variables chosen to represent the ideas which are to be contained in the resulting designs. The genetic engineering approach described provides a means of automating the representation part of the formulation process. The level of granularity is determined by the stability condition of the evolutionary process or can be determined by the user. The targeted building blocks provide a high-level starting point for all later designs which are to exhibit the required characteristics as evidenced in the earlier designs. It is this latter requirement which is met by this formal method.

The following simple picture can be used to summarize the model described in

this paper. A group of children are playing with the "Lego" game using not more than 50 squares. They join them together and want to build the object with the largest number of closed spaces inside. After each child has built his or her object the supervisor tries to find a combination of squares which is present in many of the best designs but is present in none or only in a few of unsatisfactory designs. Then he makes this combination permanent by gluing its components together and adds a bunch of such permanent combinations to the pool of building elements available to the children. Then the children make another set of objects using these new building blocks as well as an old ones. The supervisor tries to find another "good" composite block and the process is repeated. Thus, two steps occur in each cycle: first children make a set of new designs from currently available blocks and combination of blocks and second the supervisor tries to single out the additional combination of blocks that should be employed. If there are no such combinations which distinguish "good" design from the "bad" ones then we will not get new combinations but only the improved designs.

Style

The choice of particular variables and configurations of variables is a determinant of the style of the design (Simon, 1975). The label 'style' can be used in at least two ways: either to describe a particular process of designing or as a means of describing a recognizable set of characteristics of a design. Thus, it is possible to talk about the 'Gothic' style in buildings or the 'high tech" style of consumer goods. Precisely what goes to make up each of these styles is extremely difficult to articulate even though we able to recognize each of these styles with very little difficulty. An appropriate question to pose is: how can we understand what produces a style during the formulation stage of a designing process? This brings us back to the concepts described in this paper.

> 'The history of taste and fashion is the history of preferences, of various acts of choice between different alternatives..... [But] an act of choice is only of symptomatic significance, is expressive of something only if we can really want to treat styles as symptomatic of something else, we cannot do without some theory of the alternatives' (Gombrich, quoted from (Simon, 1975)).

If we use a particular style as the fitness of interest then it should be possible to utilise the genetic engineering approach described in this paper to determine if there is a unique set of genes or gene combinations which is capable of being the progenitors of that style. For this to occur satisfactorily a richer form of pattern recognition will be needed than that alluded to here. We will need to be able to determine a wider variety of gene schemas in the genotypes of those designs which exhibit the desired style.

The use of genetic engineering in evolving schemas of interest opens up a potential subsymbolic model of emergence including the emergence of domain se-

mantics (Gero and Jun, 1995). Style can be considered as a form of domain semantics. This is of particular interest in design synthesis since, if domain semantics can be captured in a form such as described in this paper, then they can be readily used to synthesize designs which exhibit those semantics and even that style. This is analogous to the induction of a shape grammar which captures the characteristics of designer's style.

6. Acknowledgments

This work is directly supported by a grant from Australian Research Council, computing resources are provided through the Key Centre of Design Computing.

References

Coyne, R., Rosenman, M., Radford, A., Balachandran, M. and Gero, J. S.: 1990, *Knowledge-Based Design Systems*, Addison-Wesley, Reading.

Gero, J. S.: 1990, Design prototypes: a knowledge representation schema for design, *AI Magazine* **11**(4): 26–36.

Gero, J. S. (ed.): 1985, *Design Optimization*, Academic Press, New York.

Gero, J. S. and Jun, H.: 1995, Emergence of shape semantics of architectural shapes, *Technical Report*, University of Sydney, NSW 2006, Australia, Key Centre of Design Computing.

Gero, J. S. and Kazakov, V.: 1995, An exploration-based evolutionary model of a generative design process, *Microcomputers in Civil Engineering (to appear)* .

Gero, J. S. , Louis, S. and Kundu, S.: 1994, Evolutionary learning of novel grammars for design improvement, *AIEDAM* **8**(2): 83–94.

Holland, J.: 1975, *Adaptation in Natural and Artificial Systems*, University of Michigan, Ann Arbor.

Kirkpatrick, S., Gelatt, C. G. and Vecchi, M.: 1983, Optimization by simulated annealing, *Science* **220**(4598): 671–680.

Pearson, W. and Miller, W.: 1992, Dynamic programming algorithms for biological sequence comparison, *Methods in Enzymology* **210**: 575–601.

Simon, H.: 1975, Style in design, *in* C. Eastman (ed.), *Spatial Synthesis in Computer- Aided Building Design*, Applied Science, London, 287–309.

Stiny, G. : 1980a, Introduction to shape and shape grammars, *Environment & Planning B* **7**: 343–351.

Stiny, G.: 1980b, Kindergarten grammars: designing with Froebel's building gifts, *Environment & Planning B* **7**: 409–462.