How to make design optimization more useful to designers

John S Gero and Somwrita Sarkar

Key Centre of Design Computing and Cognition, University of Sydney, Australia

ABSTRACT: This paper presents an approach to making design optimization tools more useful to designers. Design optimization is the selection of the best design solution from alternative solutions to a given problem. Design optimization tools require that all the variables, parameters, goals and constraints be defined prior to the execution of any optimization process. This narrows the scope of design optimization, as the user has to reformulate the problem if any of these change during the designing process. The situatedness paradigm in design proposes the interactions and processes between the designer/ design tool and the environment as the basis for designing. A design system based on the situatedness paradigm can handle changing variables and constraints, and can reconfigure the solution space based on an assessment of the current conditions. This paper proposes an alternate approach to design optimization based on the situatedness paradigm, where an adaptive situated optimization framework can automate the task of problem and solution reformulation as part of the optimization process to produce better solutions. The framework is demonstrated through an example using genetic algorithms as an optimization technique.

Conference theme: Digital
Keywords: Design optimization, situatedness, situated design, genetic algorithms

INTRODUCTION

Design optimization is the search for the “best” solution in a defined solution state space. Any point in such a solution space represents the structure of a possible solution, defined by a configuration of variables. The aim of the optimization process is to reach a particular configuration of variables defining a structure which minimizes or maximizes defined expected behaviours. Design optimization takes the view that the variables defining the structure of the design and its corresponding behaviours are fixed a priori to the application of the optimization process. The goal, which is expressed in terms of optimizing defined behaviours, remains unchanged during any optimization run. The solution space to a given problem is fixed at the commencement of the process. Any solution that is optimal will be found within this fixed solution state space. If the problem needs reformulation, or a different technique of optimization, the user has to reformulate the problem separately from the application of any optimization algorithm. Thus, the solution depends on the modelling judgment applied by the user.

1. LIMITATIONS IN CURRENT APPROACHES TO DESIGN OPTIMIZATION

Design optimization supports routine designing, where all the requirements are known at the outset of the design task. In conceptual designing, all requirements are not explicitly defined at the beginning of the process. Some requirements and information about the design are discovered through the process of designing itself. In conceptual designing goals variables and constraints can change, giving rise to a modified solution space. This cannot be handled by traditional models of optimization.

Most artificial intelligence applications in design computing have been based on the premise that the tool is unchanged by the patterns of its use (Gero 2003). The world is assumed to be fixed and well defined and knowledge is an objective, static given. Design optimization theory and techniques and routine designing have traditionally been based on this world view.

This paper proposes an alternate approach to design optimization to make it more useful for designers by having the optimization tool adapt itself to the problem’s behaviour derived through the interactions between the tool and the problem. The approach is founded on concepts from situated cognition (Clancey 1997) and situatedness (Gero 2003).

2. SITUATEDNESS IN DESIGN

In the situatedness paradigm (Gero 2003) the interactions and processes between the designer, design tool and the environment are part of the basis for developing the design rather than only the knowledge encoded in the tool. The *situation* is a construction resulting from the designer/ design tool’s internal perceptions and conceptions. The situation is the world within which a designer or system operates and is not fixed. The act of designing and the design are based on the interpretations of the internal and external environment as perceived by the tool. Both the external and internal worlds are dynamic and change in response to the processes of designing. These changes and responses are the basis for the tool developing its knowledge base. Knowledge is partially constructed from the tool’s specific, situation-based experiences. As the tool builds its experiences, it forms concepts and uses its experiences to “ground” useful concepts

and discards those that are not useful. Each design tool may construct different situations out of the environment, so the concepts which these tools learn may differ from tool to tool depending on the experiences each tool is exposed to.

3. THE FUNCTION – BEHAVIOUR – STRUCTURE FRAMEWORK

The function – behaviour – structure framework (Gero 1990) is an ontology of the processes of designing. All design variables belong in one of the three sets:

a) *Function* variables describe what the design is for; function is a property ascribed to a design in terms of its purpose and the role it plays.
b) *Behaviour* variables describe how the design behaves or performs as a system, in order to accomplish the specified goal; behaviour is a property that is derivable from the structure of the design and is a transformation relationship between the structure and function of the design, rendering the function realizable through a specific structure producing specific behaviours.
c) *Structure* variables describe the various elements of the design and the relationships between them; structure is the basis for generating a behaviour which accomplishes the ascribed functions.

Gero's (1990) FBS model differentiates between actual and expected behaviours. Actual behaviour is the behaviour derived from the structure, while the expected behaviour defines the behaviour which the structure should achieve. A set of 8 design processes relates these variables. These processes are: formulation, synthesis, analysis, evaluation, reformulation and finally the production of the design description. This framework defines the main processes for designing. It does not include the concepts of the situatedness paradigm and a model of design based on this framework would still depend on explicitly encoded knowledge.

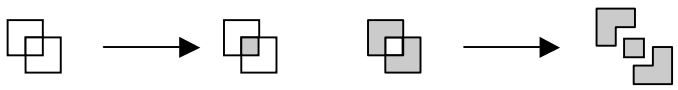
4. THE SITUATED FUNCTION – BEHAVIOUR – STRUCTURE FRAMEWORK

The situatedness paradigm has been formally incorporated into the FBS framework by Gero and Kannengiesser (2004). In this modified framework, in addition to the three defined sets of variables – function, behaviour and structure, three environments are defined which contain the variables and the processes of designing relating them. The *external world* is the world where the design tool effects its actions, and is composed of representations outside the design tool. The *interpreted world* is a construction inside the design tool in terms of sensory experiences, percepts and concepts. These are internal representations of parts of the external world with which the design tool interacts. The *expected world* is the world the actions in the interpreted world are hypothesized to produce.

Three kinds of processes link the three worlds and the FBS variables contained in them. *Interpretation* is the process by which variables in the external world are sensed and transformed into experiences, percepts and concepts in the interpreted world. *Focusing* selects aspects of the interpreted world and uses them in the expected world as current goals, leading to actions which, if taken, will produce the desired states (goal fulfillment) in the external world. An *action* produces a change in the external world based on the current goals in the expected world. The three worlds and the set of related variables and processes form the situation at a given time.

5. EMERGENCE AND INTERPRETATION

The situated FBS model lays the foundation for an alternate approach to design optimization. In this FBS framework, the design and the designing process are products of interpretations. In any a specific interpretation, the tool may detect emergent characteristics in an existing design representation that are a product of that situation. *Emergent* features in a design are those features which were not intentionally put in or conceived of by the designer at the time of problem formulation. It is an "unintended consequence of moves" (Schon and Wiggins 1992), which the designer/ design tool perceives from a current representation as a new feature which is implicit in the representation. Figure 1 shows examples of emergent features in designs. The superimposition of two squares results in the emergence of three new features, which the system learns as concepts by the process of re-interpretation.



(Stiny 1980)

Figure 1: Emergence in design representations: one small square and two L-shapes emerge from two overlapping squares

The tool may re-interpret and re-represent (Karmiloff-Smith 1992) the existing design and may construct a new representation, which transforms implicit emergent features into explicit design features (Saunders and Gero 2000). A *construction* here implies the act of the system detecting a useful emergent feature and then transforming its current structure to incorporate the representation of the new feature as usable design knowledge in other situations.

Emergent features may be of two kinds: one that may be constructed directly by the design tool out of the existing knowledge within it, and the other that may not be constructed out of its current knowledge. In the example in Figure 1, the small square which emerges from the superposition is known to the system as a concept, but the L-shapes are new features which are unknown to the system, and cannot be produced by it directly. The former class of emergent features may prove useful in grounding higher level combinations of lower level relationships and elements. The latter class of emergent features is more interesting and has the potential to expand or reconfigure the solution space. A new emergent feature that cannot be directly produced by the tool as a design element or relationship, if detected and re-represented successfully will have the capacity to produce an expanded solution space containing new kinds of solutions that were not possible before (Gero and Kazakov, 1998).

The processes of interpretation and concept formation are linked to the process of knowledge representation within the tool. Any expansion in the tool's capacity to re-interpret will also depend directly on the flexibility provided by the knowledge representation structure which the tool uses. The representation structure must be able to incorporate and represent new concepts.
This approach forms the basis of a situated generative design optimization system. In conceptual designing stages, a situated design optimization tool may assist the designer in generating new kinds of optimized solutions within a single optimization procedure, without the user having to reformulate the problem externally. The tool learns new concepts from the results of its iterations and applies them to reconfigure the solution space for the next iteration. This is similar to the human process of designing, where designers use sketches not just as external representations but also as a mode of interaction with the developing design to discover new ideas, what Schon and Wiggins (1992) have described as "the interaction of making and seeing".

6. USING EVOLUTIONARY SYSTEMS

We will model the situated design optimization framework as an evolutionary system, with its ontological basis in the situated FBS framework. We will demonstrate the framework by an example which uses genetic algorithms. There are several reasons for choosing evolutionary systems as an exemplar for the framework. Evolutionary systems exploit parallel information processing techniques, can deal with populations of solutions and are robust performers in search problems (Goldberg 1989). In design situations, designers may be interested in the optimal as well as other, near optimal "good" solutions, as they may contain multiple features of interest which the optimal solution may not capture, and which may turn out to be useful concepts to exploit later in other design problems. Since evolutionary systems have the capacity to process populations of solutions over generations, the process is potentially knowledge rich in this sense, and may be exploited to gain information about the design and designing process as a whole rather than just a best solution.
In the situated evolutionary design optimization framework, the structure of the design solution is represented by the genotype and phenotype, and behaviour is represented by the fitness or performance criteria used to assess the phenotype. Reformulation or reinterpretation of the structure of a design solution is possible by assessing either the genotype or the phenotype. The improvement of structure based on genotype assessment and modification based on the principles of genetic engineering has been demonstrated (Gero and Kazakov 2001). In the genetic engineering approach, the genotypes for "good" solutions were analyzed for recurrent emergent features and other genotypes were engineered to incorporate these useful features.
In this paper, we explore the assessment of the phenotypes for those recurrent or novel emergent features, which lead to fit solutions. These may be reverse engineered using inductive processes and incorporated into the genotypes as well as the generic knowledge base of the tool. An advantage of analyzing the phenotype is that a diagrammatic analysis is possible, which is related to the way human designers re-interpreting their sketches or drawings. A system incorporating these concepts is shown in Figure 2.

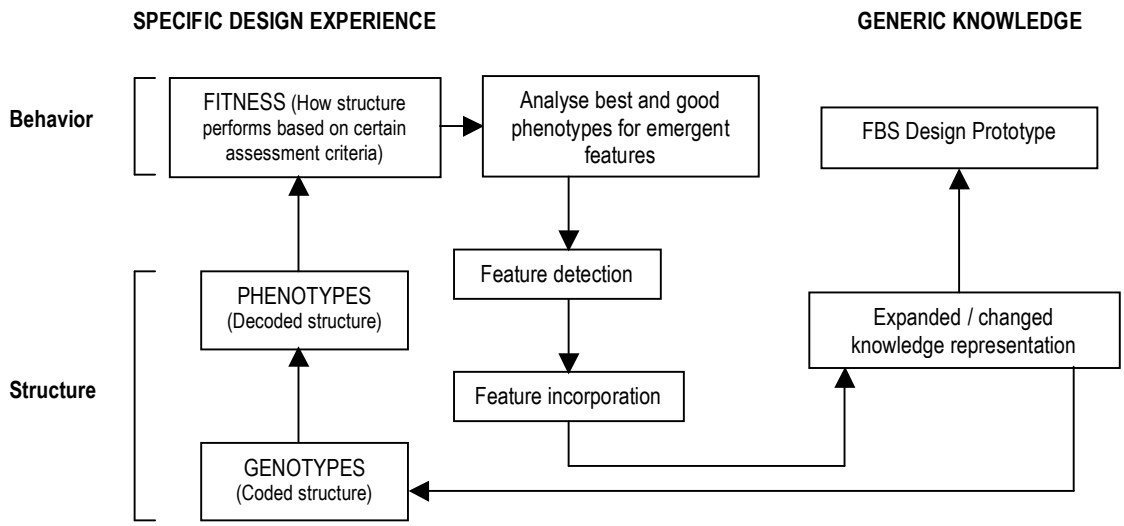


Figure 2: An evolutionary optimization system with an adaptive knowledge representation mechanism

7. THE SITUATED DESIGN OPTIMIZATION FRAMEWORK

We develop an optimization system using a genetic algorithm and its three basic operators – selection, crossover and mutation, making it situated by using the situated FBS processes. Although FBS processes incorporate the transformations and reformulations of all three categories – structure, behaviour and function, this paper will look only at structure reformulations. Figure 3 elaborates the general structure shown in Figure 2, and shows a situated design optimization framework. We demonstrate an example of this framework in the next section.
Structure is represented by phenotypes and genotypes. Behaviour is defined as the degree of novelty and usefulness of the emergent features within the environment. If the emergent feature is one that may be generated by existing knowledge, the feature is assigned low fitness in terms of novelty. If its presence in a solution leads to "good" solution

structures, it is assigned a high fitness in terms of usefulness. The combination of both behaviours determines a structure's usefulness.
A generator is developed, which produces the genotypes representing the alternatives for solution structures. Two primary gene groups are defined as part of the external problem formulation – an element group and a relationship group. The element group defines the primary elements which are available to the generator to create solutions. The relationship group defines the primary relationships which are available to the generator for operations on the elements. A genotype, G, is represented as

$$G = \{E_i, R_j\}$$
$$E_i = \{e_1, e_2, \ldots, e_n\}$$
$$R_j = \{r_1, r_2, \ldots, r_m\}$$

where E_i represents the primary elements of structure and R_j represents the primary relationships operating on these elements.

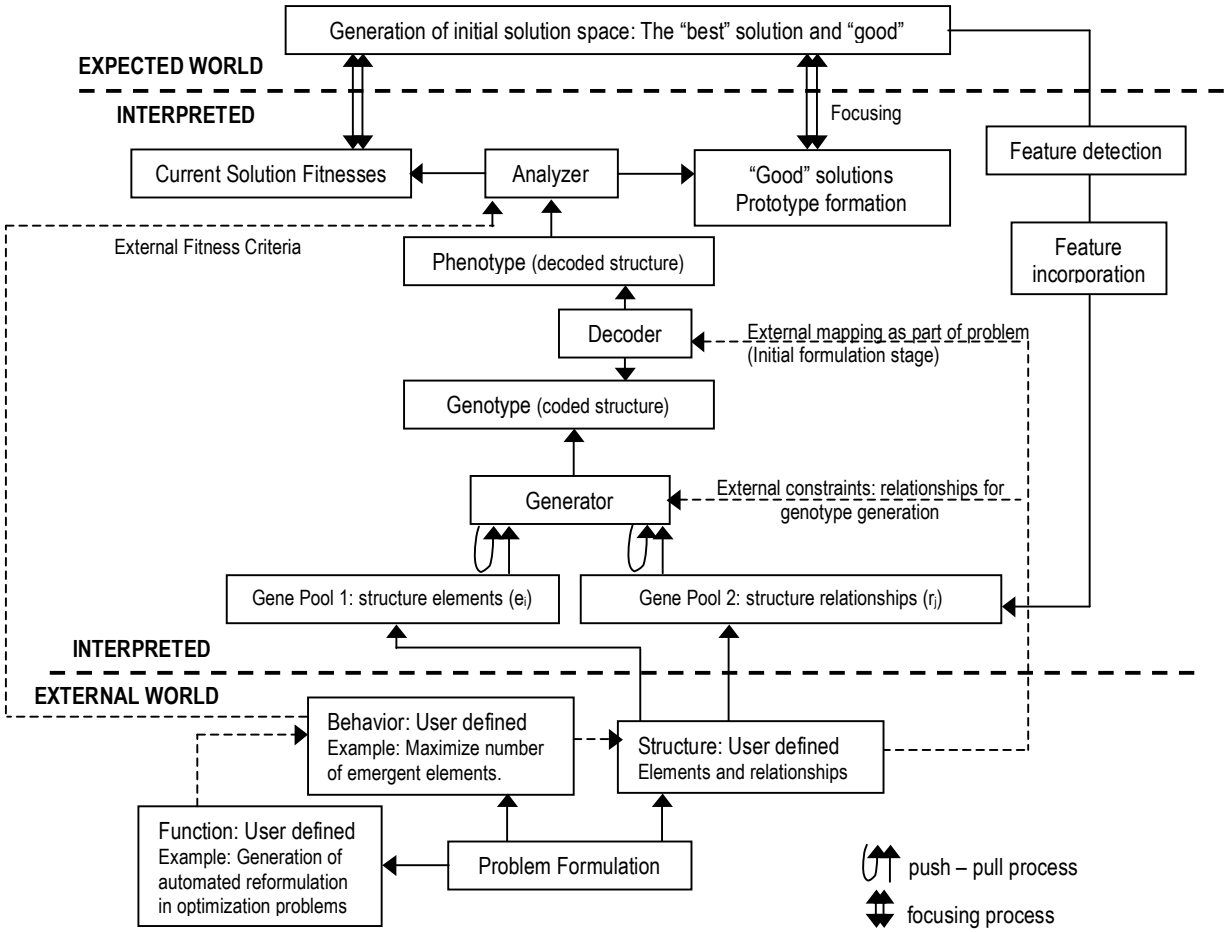


Figure 3: The situated design optimization framework

The initial mapping between the genotype and the phenotype as well as the knowledge required to generate the genotypes is specified as part of the problem formulation in the external world by the user. This is the knowledge base on which the tool builds in a situated way through its experiences, to induce changes in the generator. The mapping between the genotype and the phenotype is used by the decoder to generate a phenotype. The phenotypes or structures pass through an analyzer, which keeps track of the optimal and all the other "good" solutions in each generation, searching for any recurrent features in the structures. These recurrent features, emergent or otherwise, represent knowledge constructed out of primary elements. They are coded as constructed members of the primary element and relationship gene groups. The tool uses the FBS prototype model to incorporate the knowledge derived from specific experiences into a generic database. This information from the analyzer is fed back into the prototype, the gene pools and the generator, for use in the next generation of solutions.

8. EXAMPLE

8.1 Formulation

The process of formulation is used to generate an initial design state space based on the problem formulation in the external world. The user in the external world formulates the problem in terms of behaviour and structure variables. In the optimization framework, structure is defined in terms of the element gene group E_i and the relationship gene group R_j. Some examples of the elements and transformation relationships between them are shown in Table 1. The

framework is general and can incorporate any type of elements and relationships. The goal is to have an optimization system that can automate reformulation of the solution state space through the incorporation of emergent structure features derived from previous iterations. One of the possible example behaviours which decide the fitness criteria for "goodness" would then be defined as: Maximize the number of new emergent structures, novel or useful. If we use the structure and relationships in Table 1 we could specialize this to be those which are closed bounded shapes or cluster shapes.

Table 1: Primary Gene Groups

Primary element gene group (E_i)	Primary relationships gene group (R_j)
• e_1 = point (x, y) • e_2 = line (L) • e_3 = square (S) of n units side length	• r_1 = N (Introduce a new element) • r_2 = n (n units) • r_3 = U (move element up) • r_4 = D (move element down) • r_5 = L (move element left) • r_6 = R (move element right) • r_7 = O (Rotate element) • r_8 = F (Reflect element)

The generator produces combined genes by choosing randomly from the element gene group and the relationship gene group. The crossover operation is defined as a crossover between two genotypes, with the constraint that the crossover site can only fall between the genes and not inside a single gene (since the single gene is a combination of elements and relationships operating on them, this will not be disrupted here). Let us assume the generator produces a combined gene by choosing exactly one member from the element group, a square, and one member each from the relationship group to operate on this element. We show a manual demonstration of the generation of new emergent concepts.
Let the generator choose a square of 2 units. It then randomly chooses relationship genes to produce genotypes, which can be flexible in length. The decoder operates on these genotypes to generate phenotypes. The analyzer then calculates fitness for these phenotypes by detecting emergent features and analyzing them for the required characteristics. The high fitness phenotypes, which in this example are the ones which contain new closed bounded shapes or cluster shapes, are sifted out of the population and transferred to the expected world, as the initial solution state space. Figure 4 shows an example of the generation of a "design".

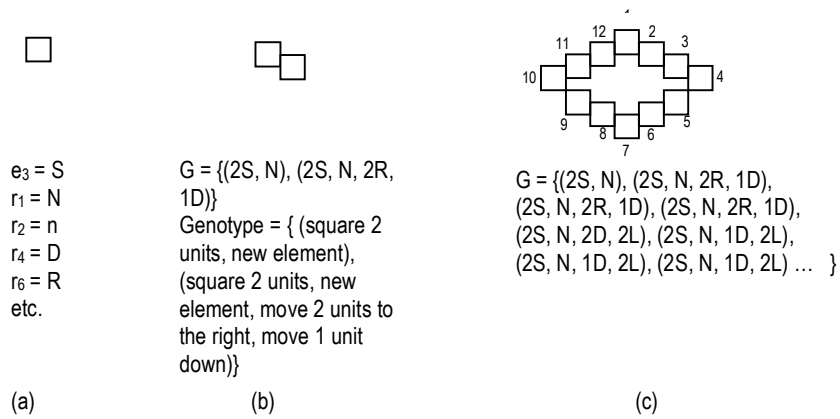


Figure 4: Generation of a "design": (a) generator randomly chooses element e_3 and relationships r_1, r_2…r_j (b) genotype with 2 genes; (c) genotype with 12 genes; each gene is a combination of an element and relationships operating on it.

Starting with a single structure element, the genotype is generated by selecting relationships randomly and applying them to the element as a transformation. A single gene in the genotype is a combination of structure elements and relationships operating on them. In this example, only a single element of structure is chosen (as a kind of external constraint applied by the user on the generator) and various relationships are applied on this structure. A series of genes form the genotype. The decoder operates on the genotype to produce the graphic phenotype as shown in Figure 6.

8.2 Synthesis

Synthesis is defined as the process of transforming expected behaviour into external structure. In the optimization framework, the initial solution state space contains the best and the "good" phenotypes, which are expressed in the external world as solution representations. The system provides the representation of the best as well as other good solutions, as this will render them open to re-interpretation, and some useful knowledge may be derived from them. It is possible that the re-interpretation and re-representation of a near optimal structure can generate new emergent features.

8.3 Analysis and Evaluation

Analysis is defined as the derivation of interpreted behaviour from a synthesized external structure. Evaluation is the process in which this actual behaviour is compared with the ideal expected behaviour. In the optimization framework, phenotypes are analyzed for recurrent sub-structures which may be produced out of a combination of known relationships, and the presence of which within a solution leads to high fitness. These are interpreted as new structure

elements and incorporated as a new gene in the gene groups, Figure 4. In the next run, these genes can be used as regular elements of structure. In the example in Figure 6, the combination of 12 genes produces a structure from which a new structure element and relationships can emerge. The tool can discover a diagonal relationship of the form (U + R) = D_R, i.e. a move up added to a move right produces a diagonal move to the right. The tool can also learn other such transitive, commutative relationships, and group them as new relationships which can be applied directly in future applications of the tool. The phenotype itself is interpreted as a new structure element and incorporated into the gene groups as a new gene which can be reused in the next iterations.

8.4 Re-formulation of structure

Re-formulation of an existing structure leads to a different structure with possibly different behaviours. The current phenotype structures are analyzed to detect emergent features within them which are novel, i.e. structures that cannot be generated from the already known relationships and elements. This is the process of re-interpretation. This is followed by the process of re-representation of an emergent feature. Re-interpretation of the structures in Figures 5, 6 and 7 lead the analyzer to detect a number of possible novel emergent features which are re-represented as new elements. These elements are again tested for usefulness by the defined fitness criteria, and the highly fit ones are incorporated as new genes in the existing gene groups.

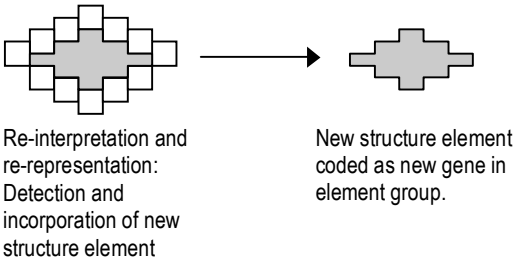


Figure 5: Emergent novel features: re-interpretation of existing phenotype to detect an emergent feature; followed by re-representation

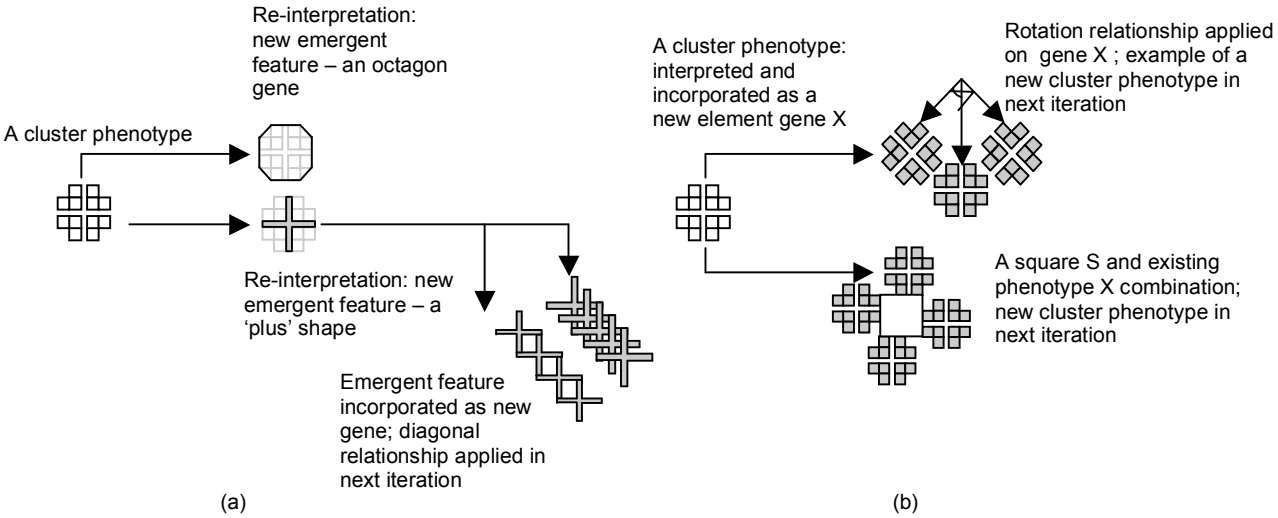


Figure 6: Emergent novel features: (a) cluster shape phenotype re-represented as two new shapes, which are incorporated as genes for next iteration; (b) cluster phenotype as new gene undergoes rotation and combination with a square to produce new cluster phenotypes

Figure 5 shows the phenotype produced by a genotype comprised of 12 element-relationship genes. The phenotype is then re-interpreted to detect a new closed bounded shape using a figure ground reversal mechanism. The new feature is a shape that is novel and cannot be produced by a direct combination of the known elements and relationships. Similarly, in Figure 6(a), a cluster gene produced by elementary combinations of squares is re-interpreted and then re-represented to produce two features: an octagon and a "plus" shape, both of which could not have been produced by the known element-relationship combinations. Figure 6(b) shows that the cluster phenotype is interpreted as a new gene X, and incorporated as a regular element into the gene group. In the next iteration, the generator can now choose it in combination with other known elements and relationships to generate phenotypes. Figure 7 shows an example of a cluster phenotype generated from the same 3 square element combination as Figure 6, but in an inverted reflected relationship. This produces a different cluster phenotype than the one generated in Figure 6. The same generic process is now able to produce shapes potentially different from the example above. The examples in Figures 8 and 9 show that slightly different relationships in starting iterations applied on similar elements have the capacity to produce widely differing structure combinations.

The novel emergent features, once detected and re-represented, are incorporated as new members of the elements group. The same relationships may be applied to these elements to produce more shapes, out of which some more novel features may emerge giving rise to a constantly expanding solution space.

For the examples, we, as external users, started with a given configuration of the elements, relationships and the mechanics by which the element and relationship genes combine to produce complex genes to produce a genotype. As the tool learns and modifies its concepts, these may change by using inductive mechanisms. In an inductive process, the tool generates new concepts out of specific design experiences, if these specific cases produce consistent good results. Elements and relationships, primary, generated or emergent, which occur repeatedly in good solutions, are induced as concepts. This may also result in some of the unused concepts, either defined by the user or discovered by the tool as part of the designing process, to be discarded. With time, design experiences are used to ground the useful concepts, and discard those that have not been used.

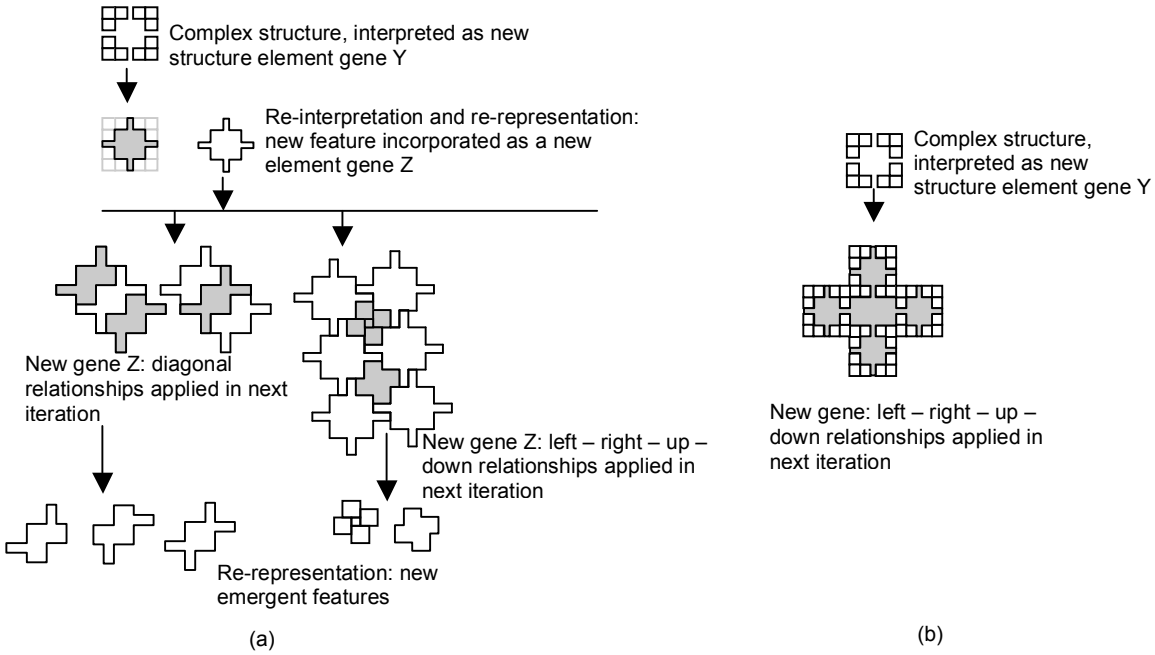


Figure 7: Emergent novel features: (a) cluster phenotype re-represented to produce a new shape, which is incorporated as a gene for next iteration; (b) cluster phenotype as a new gene combines with itself to produce a new cluster phenotype

There is a possibility that the same shape can be the product of many genotype “paths”, i.e. the same shape can be generated by many paths. This attribute supports a multiple representation mechanism for shapes. Since parallel processing of schema are supported by genetic algorithms, multiple representations of the same shape may be processed as independent representations, without causing any ambiguity. This attribute is useful in the designing process as different design situations may demand or use different forms of representation for the same shape.

This process is situated because the solution structures and strategies, and gradually the knowledge base which the tool develops, are derivable only from what the tool experiences, interprets and re-represents. No coded knowledge, except for the primary elements and relationships and the external problem formulation in terms of an objective function, constraints and fitness criteria are provided to the tool. The situatedness paradigm, applied through the FBS ontology is used to re-interpret solutions and detect novel emergent features in existing solutions. These features are re-represented such that the system recognizes them as new elements, which it can use in the next iteration. The solution space that is generated in the next iteration will now incorporate these new elements that did not exist before. Since these new interpreted genes could not have been generated by the original elements without a situated re-interpretation and re-representation, it is evident that the solution space is reconfigured and contains new kinds of solutions only because of the situated way in which the design optimization process evolves.

As the tool runs iterates in the process of optimization, the solution spaces are modified with each design experience. As an optimization tool, the tool develops an additional attribute: the capacity to preserve new useful concepts that are generated from an interpretation of the design solutions and the designing process. This results in the tool developing task knowledge and domain knowledge, which it can use to produce better solutions. The optimization process may be controlled in two ways now, either by the user, or by the tool itself. The tool, through an adaptive incorporation of new concepts that produce better solutions, also develops the capacity for re-formulating the problem. This capacity to reformulate a problem, has, till now, been a task requiring human expertise. The situated design optimization framework can effectively automate this task of problem reformulation.

CONCLUSION

The framework developed for situated design optimization views design optimization as a process in which the tool can effectively produce new and better solutions, by restructuring the solution state space to incorporate emergent features

which are discovered as part of the process of designing. The generation of solution states in each iteration can depend upon the new knowledge discovered, re-interpreted and re-represented in the previous iterations. This form of derived knowledge can lead to the tool supporting design optimization tasks which otherwise require that the user reformulate the design problem. This reformulation is now part of the design process and can be either controlled by the tool's user or let run automatically to see where it leads. In each case an adaptive expansion of the solution space will only occur if the emergent structure features potentially produce better solutions than before. Manual examples of re-representation have produced expanded solution spaces that contain better designs than previous "optimal" designs (Gero and Kazakov, 1999).

Situated design tools are modelled with concepts from situated cognition derived from studies of human behaviour. It may be that such tools follow the user's development of knowledge about both a problem and the tasks involved in dealing with it, more closely than tools that do not adapt themselves.

ACKNOWLEDGEMENT

This research is supported by a Faculty of Architecture research scholarship and by a grant from the Australian Research Council.

REFERENCES

Clancey, W. (1997), *Situated Cognition: On Human Knowledge and Computer Representations*, Cambridge University Press, Cambridge

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

Gero, J.S. (2003), *Design tools as situated agents that adapt to their use*, *in* W Dokonal and U Hirschberg (eds), *eCAADe21,* eCAADe. Graz University of Technology, pp 177-180

Gero, J.S. (2003), *Situated computing: A new paradigm for design computing*, *in* A Choutgrajank, E Charoenslip, K Keatruangkamala and W Nakapan (eds), *CAADRIA03*, Rangsit University, Bangkok, pp 579-587.

Gero J.S. and Kannengiesser U. (2004) *The situated function-behaviour-structure framework*, *Design Studies* **25**(4): 373-391.

Gero J.S. and Kazakov V. (1998) *Adaptive expansion of search spaces using adaptive features* in J. Slaney, G. Antoniou, and M. Maher (eds) AI'98 Griffith University Brisbane, Australia, pp 25 – 36

Gero J.S. and Kazakov V. (1999) *An extrapolation process for creative designing* in G. Augenbroe and C. Eastman, (eds), Computers in Building, Kluwer, Boston, pp 263 – 274.

Gero, J.S. and Kazakov, V. (2001), *A genetic engineering extension to genetic algorithms*, *Evolutionary Systems* **9**(1): 71-92.

Goldberg, D.E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, USA.

Karmiloff-Smith A. (1992) *Beyond Modularity: A Developmental Perspective on Cognitive Science*, MIT Press

Schon, D. and Wiggins, G. (1992), *Kinds of seeing and their function in designing*, *Design Studies* **13**(2): 135-156