Co-evolution and Emergence in Design

J. Poon and M. L. Maher

Key Centre of Design Computing
Department of Architectural and Design Science
University of Sydney NSW 2006 Australia
Email: (josiah, mary)@arch.usyd.edu.au

Abstract

Evolution as a metaphor borrowed from nature can be used to describe a design process. However, this has generally been applied to the evolution of a solution which assumes the problem does not change throughout the process. This is a naive assumption in design because the problem indeed changes. This paper considers the evolution of both the problem and solution and introduces co-evolutionary design. This paper proposes two approaches to implementing co-evolutionary design and also addresses the related issues of evaluation and termination in a computational model. Finally, the paper considers how a co-evolutionary system can generate and recognize emergent structure and behaviour.

Keywords: co-evolutionary design, emergence, genetic algorithms, evolutionary systems

1 Introduction

Nature gives inspiration to research scientists. She is used to model both the evolutionary characteristics of a design process and to provide a framework for the development of computational methods, e.g. genetic algorithms (Goldberg 1989), evolutionary programming (Fogel, Owens & Walsh 1966) etc. Therefore, it seems appropriate to implement a nature-inspired design process with a nature-inspired computational paradigm. In section 2, co-evolutionary design is introduced where the assumption of having a fixed goal (problem) is removed. The problem is allowed to change over time. The co-evolutionary design model leaves us to consider its implications on evaluation and termination. Another important research agenda follows, which is emergence in the context of co-evolutionary design. Emergence of behaviours and structures are discussed in section 3. Experiments are reported in section 4. Finally, the paper is closed by the conclusion in section 5.

2 Problem-Design Exploration as Co-evolution

A design process is traditionally viewed as a sequential process model from the formulation of the problem to the synthesis of solutions. In his book, Simon (1981) regards design as a state-space search where a problem leads to solution. However, to find *the* solution is seldom a one-off activity. To be more practical, there are many versions of solution generated during the course of design, where each current one is, in general, an improvement over the previous one. This kind of synthesis of solutions can be viewed as an evolutionary system over time.

The view of design as state-space search has dominated the research direction of the AI-in-Design community for some time. This is an attractive assumption because what is once a complex human activity is reduced to a relatively manageable computing task.

However, this simplified view faces a lot of challenges (Corne, Smithers & Ross 1994), (Gero 1994), (Maher & Poon 1996). The major criticism is on the assumption that a problem is defined once-and-for-all. This is definitely not true for design. The central tenet behind the opposing views is that design should be considered as an iterative process where there is interplay between problem reformulation and solution generation.

According to the evolutionary design process model offered by Hybs & Gero (1992), the formulation of functional requirements is to define expected behaviour, B_e, which is represented as the problem space. The solution space can be considered to contain structure elements where the design process is to search the right combination of structure elements to satisfy the requirements, B_e. The behaviour exhibited by the current structural combination (B_s) is compared against B_e in the evaluation process. Reformulation, which is defined as $S \rightarrow B_e$, is conducted if necessary.

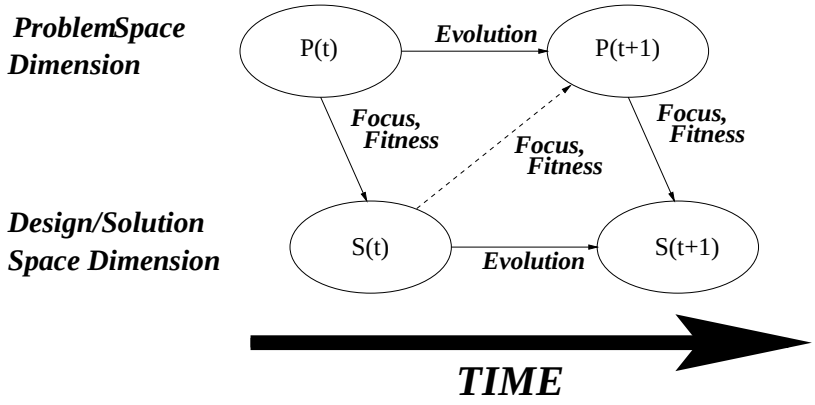


Fig. 1. Co-evolution of problem-space and design solution-space

Maher & Poon (1996) propose to model this problem-design exploration as co-evolution. This is graphically illustrated in Figure 1 as the interaction of problem space (the required behaviour) and solution space (the potential structural combinations). The diagram highlights the co-evolution of the behaviour-space with the structure-space over time and has the following characteristics:

(i) There are two distinct search spaces: behaviour-space and structure-space.
(ii) These state spaces interact over a time spectrum.
(iii) Horizontal movement is an evolutionary process.
(iv) Diagonal movement is a search process where goals lead to solution. This can be the
   (a) Downward arrow: *"Problem leads to Solution"* or *synthesis* where $B_e \rightarrow S(B_s)$. The behaviour-space(t) is the design goal (the required behaviour) at time *t* and structure-space(t) is the solution space which defines the current search space for design solutions.
   (b) Upward arrow: *"Solution refocuses the Problem"* or *reformulation* where $S \rightarrow B_e$. The structure-space(t) becomes the goal and becomes the selection force to evaluate individuals in the behaviour-space at time *t+1* .

This model depicts two evolutionary systems. The evolutionary systems are the behaviour-space and the structure-space. The evolution of each space is guided by the most recent population in the other space. The basis for co-evolution is a simple genetic algorithm (GA) where special consideration is given to the representation and application of the fitness function so that the problem definition can change in response to the current solution space.

2.1 Co-evolutionary Algorithms

The co-evolution model is implemented using a modified genetic algorithm. The co-evolution of the design genes (solution space) and the fitness function (behaviour space) can be implemented with the following two approaches:

(i) **CoGA1: Combined Gene Approach**
A single composite genotype is formed by the combination of an expected behaviour and a design solution (Maher & Poon 1995). Since the fitness function is defined locally for each design solution, the measurement of a phenotype represents a local fitness value. Because of the unique representation, the basic GA is modified to have two phases in each generation. In the first phase (change of focus) of a generation, the genetic operators modify the behaviour part of a genotype, while the operators modify the solution part in the second phase (generation of alternatives). This approach can be viewed as a tightly-coupled, or a host-parasite, co-evolution where each parasite tries to adapt to a specific host.

(ii) **CoGA2: Interacting Population Approach**
Two spaces are modelled as two sets of genotypes and phenotypes: one for modelling expected behaviour and one for modelling design solutions (Maher & Poon 1996). Hence, fitness of individuals in the population of the problem requirements and population of design solutions is evaluated alternatively, i.e. one generation will have behaviours being evaluated and the other generation will have the structures evaluated. The current best individual from a population serves as the fitness measurement for individuals in another population in the next generation. This approach can be viewed as a loosely-coupled, or a prey-predator, co-evolution where the prey has to adapt to a variety of predators.

The emphasis of interaction between solution and requirements of these two approaches helps to identify complex interaction between structure variables, as well as the less attended behavior variables.

2.2 Fitness Evaluation

The performance of a solution is traditionally evaluated against a fixed goal. This is not so in a co-evolutionary design when the fitness function (goal) changes over time. A fitness function, in a co-evolutionary design, is important because it also represents how one space exerts its influence to the evolution of another space. The fitness function changes with time but it should maintain relevance to the initial expectation. Hence, we define a fitness function in a co-evolutionary paradigm to be a concatenation of initial requirements R_{init} and current best (behaviour or structure), CB(B or S), i.e. Fitness = R_{init} + CB (B or S). There are two important concepts here: the Current Best Structure (CBS) and Current Best Behaviour (CBB).

CBS is the best individual from the structure space after a fitness evaluation to the pool of plausible structures. This does not only serve as a passive end-product of an evaluation. CBS turns out to become an active participant in the evaluation, i.e. the reformulation of problem specifications, $S \rightarrow B_e$. The CBS is incorporated into the fitness function to evaluate the performance of phenotypes in the behaviour-space.

Likewise, CBB represents the individual which has the highest fitness score in the behaviour space pool after assessment by the current fitness function. This individual is also used in one of the evaluations, i.e. the synthesis of solution, $B_e \rightarrow S(B_s)$. The CBB is empty at the beginning of the design process. However, it becomes more complex as time progresses.

The balance between the initial requirements and the current best is another implementation issue, i.e. whether R_{init} and CB (B|S) should have equal weight, or should CB (B|S) should be more emphasized as time proceeds? This should be determined by the designer as whether a solution should be more sensitive to current best requirements or a solution should not deviate too much away from the initial requirements.

2.3 Termination

One concern with the implementation of co-evolution in a computational paradigm is the problem of termination. The challenge lies in how a program knows when is the "*right*" time to stop. The potential candidate solutions might include:

– *Initial Problem, P_{init}*. The search is terminated when a generated solution satisfies the initial problem (requirements). This seems a logical answer as the program finds what we want. However, this has thrown away the knowledge accumulated in the CBB or CBS.
– *Time*. When the running time of a computational system exceeds a user-specified time, it stops. This is a fair approach because, on the deadline date, even though

a human designer may find improvements can be made, he/she needs to submit the design.
- *Equilibrium*. When the results from both the behaviour space and structure space consistently show little variations over the past *N* generations, then it may be a signal to stop the design process. This indicates the two spaces are in a balanced ecology setting.
- *Repeating Pattern*. If the process does not converge to *a* solution, but several results repeat themselves, then this may be an appropriate time to stop. Such phenomenon may also indicate the desired results fall on a pareto-optimal surface. In terms of design, this is a signal that the problem can be satisfied by quite a few solutions, the designer has attempted them all and begins to run out of ideas.

3 Emergence

The co-evolutionary model aims to show the interaction between the problem (or the behaviour) space and solution (or the structure) space. If important and characteristic features of the two spaces are useful to the design goal, they tend to dominate the genotypes in a population. This is no exception in nature, emergence occurs where new species are found and they exhibit characteristic traits. Emergence is an important research issue in biology as well as in creative design. This has recently drawn the attention of research workers in the *design community*, e.g. (Gero & Yan 1994), (Gero, Damski & Jun 1995), (Edmonds & Soufi 1992) etc. Here are a few definitions on emergence offered by the *design community*.

> "*A property that is only implicit, i.e. not represented explicitly, is said to be an emergent property if it can be made explicit.*" (Gero 1992)

> "*... drawing might be thought of as a visual image together with an associated description that imposes structure upon it. Thus a drawing may be thought of as a structured entity. From this perspective, an emergent shape occurs when a revised description, or structure, is discovered ...*" (Edmonds & Soufi 1992)

> "*... emergent subshapes are... emergent entities and relationships – ones that they never explicitly input ...*" (Mitchell 1993)

The definitions offered from the *design community* are usually applied to shape only. Hence, attempts are made to borrow definitions of emergence from other research communities to enrich our understanding. Since the *ALife* (Artificial Life) research community has also put emergence high on its research agenda, it is beneficial to survey what they have offered, particularly about emergent behaviour.

> "*... emergent properties ... collections of units at a lower level of organization, through their interaction, often give rise to properties that are not the mere superposition of their individual contributions ...*" (Taylor 1990)

> "*The key idea is that functionality is made to emerge as a global side-effect of some intensive, local interactions among components that make up the system ...*" (Maes 1990)

> "*Emergent functionality means that a function is not achieved directly by a component or a hierarchical system of components, but indirectly by the interaction of more primitive components among themselves and with the world.*" (Steels 1991)

After reviewing definitions from *ALife*, an alternate definition of emergence in design is offered. Emergence is defined here as a *"global pattern as a result of local interactions of low level units"*. This alternative definition helps us to look at emergence in design from a different perspective. In the next section, emergence is addressed at the behaviour level. It is first argued that emergent behaviour, which is a result of the local interaction of genes, is goal-oriented. The issue of genes interaction is further elaborated in the Section 3.2. This is followed by a study to the several approaches which can identify a "good" gene-pair. In Section 3.4, emergence of behaviour and structure, and the framework of co-evolutionary design are integrated to give a complete picture.

3.1 Emergent Behaviours are Goal-Oriented

Since the formulation of functional requirements is to define expected behaviour, B_e (Hybs & Gero 1992), emergent behaviour must be goal-oriented. This complex behaviour cannot emerge outside the context of a fitness function. For example, if the design goal is to maximize the area of a floor plan, taking no explicit consideration of the appropriateness of room arrangement, the fitness evaluation to a design solution only derives a numerical result about the area of a floor plan. Since there is no way to assess the quality of a floor plan, there will not be any complex behaviour of room arrangement emerging from the design process. If emergent behaviours of room arrangement are sought, the fitness function must be modified to cater for room arrangement as part of its evaluation.

To find an emergent behaviour is to find the interaction of unit behaviours which contributes to the global fitness. When Fogel (1995) discusses the identification of "good" building blocks in the context of genetic algorithms, he suggests that there are no viable credit assignment algorithms for isolated genetic structures or behavioural traits. These elements are highly integrated. According to his arguments, credit assignment does not exist in evolution, it is a human construction. However, it is also true that in biological systems, diseases are discovered to relate to certain genes in a chromosome. Hence, the credit assignment should not be thrown away completely, both in natural and artificial worlds. In this situation, we propose that emerging behaviour patterns derived from this computation are interpreted as a *statistical correlation* between the complex evolved behaviour and the fitness of the phenotype.

3.2 Interactions Between Genes

Our alternative definition of emergence emphasises on *"local interactions of low level units"*. Adjacency is considered to be a kind of these interactions. In Gero & Schnier (1995), they refer "adjacency" to be the consecutive drawing commands

which draw the floor plan. In their example, only neighboring genes are allowed to interact because of an implicit assumption. They assume the evolved structure is an ordered pair where $(a, b) \neq (b, a)$.

The ordered pair adjacency interaction is a good approach if a chronological or spatio-relationship is required, otherwise, this can be relaxed to keep the pair in a set, i.e. $\{a, b\}$. To further relax the original approach (Gero & Schnier 1995) is to break another implicit assumption. Their consideration of "adjacency" assumes the genes to be arranged in a linear manner. The linear presentation of genes is a human notation. The chemical notation for water, H_2O, does not imply the *actual* configuration of the molecule. In fact, a water molecule is not arranged as H-H-O, but H-O-H. Hence, we can treat the behaviour variables to float around in a space and can interact with other variables, either strongly or weakly. Since there is no fixed *a priori* arrangement of genes, local interaction to neighboring genes has many more interactions than a linear arrangement.

The consideration of how two behaviour variables interact depends upon the necessity of chronological control and the basic assumption about genes arrangement. In summary, the interaction scheme between two behaviour variables can be classified as the following patterns with decreasing constraints:

(a) linear arrangement of genes and an evolved gene is an ordered pair
(b) linear arrangement of genes and an evolved gene is a set
(c) no predefined sequence of genes and an evolved gene is an ordered pair
(d) no predefined sequence of genes and an evolved gene is a set

3.3 Evolving New Genes

The output of the evolving representation of Gero & Schnier (1995) is a set of new evolved complex representations from genetic cycles which statistically correlate to "good" phenotypes. As a new evolved gene builds upon previous basic genes, the complexity of an evolved gene increases. The genotypes in the appropriate space is incrementally restructured by the replacement of these evolving elements. Since emergence has been defined as a recognition of collective phenomena resulting from local interactions of low level units, a complex evolving representation can thus be classified as an emergent representation. To follow this line of argument, a complex evolving behaviour is an emergent behaviour.

An evolved gene is due to the interaction of two local genes: the gene-pair. This new gene comes into existence when the contribution of a gene-pair, C_i, is considered to be significant to the "goodness" of phenotypes. A phenotype can be considered as the design solution. The internal representation is called a genotype. A mapping process is necessary to transform a genotype to a phenotype. A phenotype is classified as a "good" one when its fitness value, F_j, exceeds a user-specified threshold value, F_{min}.

There are many approaches to find a "good" gene-pair. One approach is to find the unit contribution of a gene in a "good" phenotype. This is done by dividing the fitness value of a phenotype by the length of its genotype (Equation 1). As a result,

the contribution of each gene-pair is a two "unit gene contribution". Since a gene-pair may appear more than once in a genotype, the contribution of this gene-pair for a particular genotype is a multiple of the gene-pair contribution (Equation 2). The overall contribution of a gene-pair is a summary of contribution among the "good" phenotypes (Equation 3). After gene-pairs from all "good" phenotypes are examined, they are sorted in the descending order of their contributions in Gero & Schnier (1995). If there are n gene-pairs in the list, the top 3% are classified as evolved genes and each of them are assigned with a unique gene number. These evolved genes are also added to the pool of basic genes. The corresponding gene-pairs in all phenotypes are rewritten by new evolved genes so that these pairs are not separated during crossover.

$$C_j^{unit} = \frac{F_j}{L_j} \tag{1}$$

$$C_{ij} = 2k_{ij}C_j^{unit} \tag{2}$$

$$C_i = \sum_{j=1}^{m} C_{ij} \tag{3}$$

$$C_i = \frac{\sum_{j=1}^{m} C_{ij}}{\sum_{j=1}^{m} k_{ij}} \tag{4}$$

where

i	is the i^{th} gene-pair
j	is the j^{th} "good" phenotype
m	is the number of "good" phenotypes
F_j	is the fitness value of j^{th} phenotype
L_j	is the length of genotype of the j^{th} phenotype
k_{ij}	is the number of occurrence i^{th} gene-pair in the j^{th} phenotype
F_{min}	is the minimum threshold fitness value of a phenotype to proceed to the analysis of its genotype
C_j^{unit}	is the fitness contribution of each unit gene in the j^{th} phenotype
C_{ij}	is the fitness contribution of i^{th} gene-pair in the j^{th} phenotype
C_i	is the fitness contribution of i^{th} gene-pair

Another approach is almost the same except there is a modification to Equation 3. This new approach normalizes the contribution of a gene-pair with respect to its number of times of appearance. This aims to eliminate the low performing but high frequency gene-pairs (Equation 4). However, these two approaches have an implicit assumption that each gene provides the same contribution in a genotype, C_i^{unit}. To overcome this limitation, a weight can be assigned for every gene to derive each unit strength. The drawback of weight assignment is three fold: (1) to assign a weight to each different gene is equally problematic, (2) to derive the weight of an evolved gene is another open-ended issue and (3) it assumes the gene-pair must be significant when two highly weighted genes are put together, which may not be true.

The third suggestion is a totally different approach. This approach only searches for the existence of a gene-pair and does not try to find its share to the fitness of a phenotype. The contribution of a gene-pair to a "good" phenotype is 1 if the gene-pair exists in the genotype, regardless the number of times it can be found in that genotype; otherwise, for that phenotype, the gene-pair scores a zero fitness value (Equation 5). The overall contribution of a gene-pair is a total of its contribution among the "good" phenotypes. Corollary, this approach looks at all the "good" phenotypes, and to count how many of their genotypes carry the concerned gene-pair. The threshold fitness value, F_{min}, has dual purpose. On one hand, the F_{min} is to determine whether a phenotype is good enough for further analysis. On the other hand, F_{min} is to calculate the minimum threshold population size, S_{min} (Equation 6). In other word, smaller the F_{min}, a larger S_{min} is required, and vice versa (Figure 2). A gene-pair is evolved *iff* this pair is found in at least S_{min} phenotypes in the pool where these phenotypes have fitness values exceeding F_{min} and carry the pair (Equation7).

$$C_i = \sum_{j=1}^{m} N_j \tag{5}$$

where

$$N = \begin{cases} 1 & \text{if gene-pair } i \text{ is in "good" phenotype } j \\ 0 & \text{if gene-pair } i \text{ is NOT in "good" phenotype } j \end{cases}$$

$$C_{min} = S_{min} \qquad \text{where} \qquad S_{min} = (1 - F_{min}) S_{pop} \tag{6}$$

where C_{min} is the minimum contribution of a gene-pair to be qualified as an evolved gene
S_{min} is the minimum threshold population size
S_{pop} is the population size of the pool

$$C_i \geq C_{min} \qquad \text{gene-pair } i \text{ EVOLVES} \tag{7}$$

This approach is best illustrated with an example. The example is in two scenarios for a population of 50 genotypes (i.e. S_{pop} = 50): a high threshold and a low threshold value. In the first scenario, the threshold is defined to be a high value, say, 0.9 (F_{min} = 0.9). A gene-pair is classified as an evolved gene if the pair occurs in at least 5 genotypes (C_{min} = S_{min} = 5), where these few individuals have fitness value no less than 0.9. If the threshold is lowered to 0.3 (F_{min} = 0.3) in another scenario, then the threshold population size goes up to 35 genotypes (S_{min} = 35). These two scenarios demonstrate the relationship and sensitivity of C_{min} (S_{min}) to F_{min}.

We highlight a few options to calculate the "goodness" of a gene-pair and it is not an exhaustive list. The last approach is preferred because it relies on the occurrence of a gene-pair alone and avoids the time consuming credit assignment problem. If a gene-pair is found in all the genotypes of a threshold size, this pair deems to be useful in the contribution to the fitness of its corresponding phenotype. Finding an

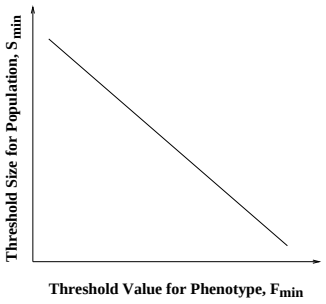


Fig. 2. Relationship between phenotype threshold value and threshold population size

appropriate algorithm for "good" gene-pairs is critical to the success of generating meaningful and useful emerging behaviour/structure.

3.4 Computational Emergence in Co-evolutionary Design

We now consider computational emergence as part of co-evolutionary design. Although Figure 3 shares the similar outline as in Figure 1, this figure further shows the emergence of behaviour and structure. The diagram can be divided into three parts, the behaviour dimension, the fitness function and the structure dimension.

At the start, individuals in the structure space are evaluated by the initial requirements alone. The individual phenotypes from the structure space are the manifestation of genotypes, which are composed from the basic genes. After this evaluation, phenotypes which achieve above a threshold value are classified as *"good"* individuals. Their corresponding genotypes then undergo further analysis. Useful gene-pairs are identified and each gene-pair is now treated as a gene on its own, thus new structure genes emerge. These evolved genes are now integrated to the pool of basic genes. They are also used to restructure the genotypes from the whole population in structure space. Meanwhile, the best phenotype is selected from the structure space to represent the CBS (current best structure), which combines with the R_{init} to form the fitness function for the behaviour space (shown as the round-corner rectangle which sits on the upward moving line from the structure dimension to the behaviour dimension).

A similar processing happens in the next generation, except the genetic operations are performed on behaviour space. The individuals in the behaviour space are evaluated by R_{init} and CBS. Behaviour genes that evolve at the end of the generation are also used to restructure the genotypes in the behaviour space. The best behaviour phenotype is identified as the CBB (current best behaviour). The CBB serves as part of the evaluation function with R_{init} for the structure space in the next generation.

Emergence of behaviour and structure is integrated to co-evolutionary design (Poon & Maher 1996). The evolved (emerging) genes can become more complex in each generation as an evolved gene may be composed of another evolved gene, thus, this is a kind of hierarchical structure of emerging genes.

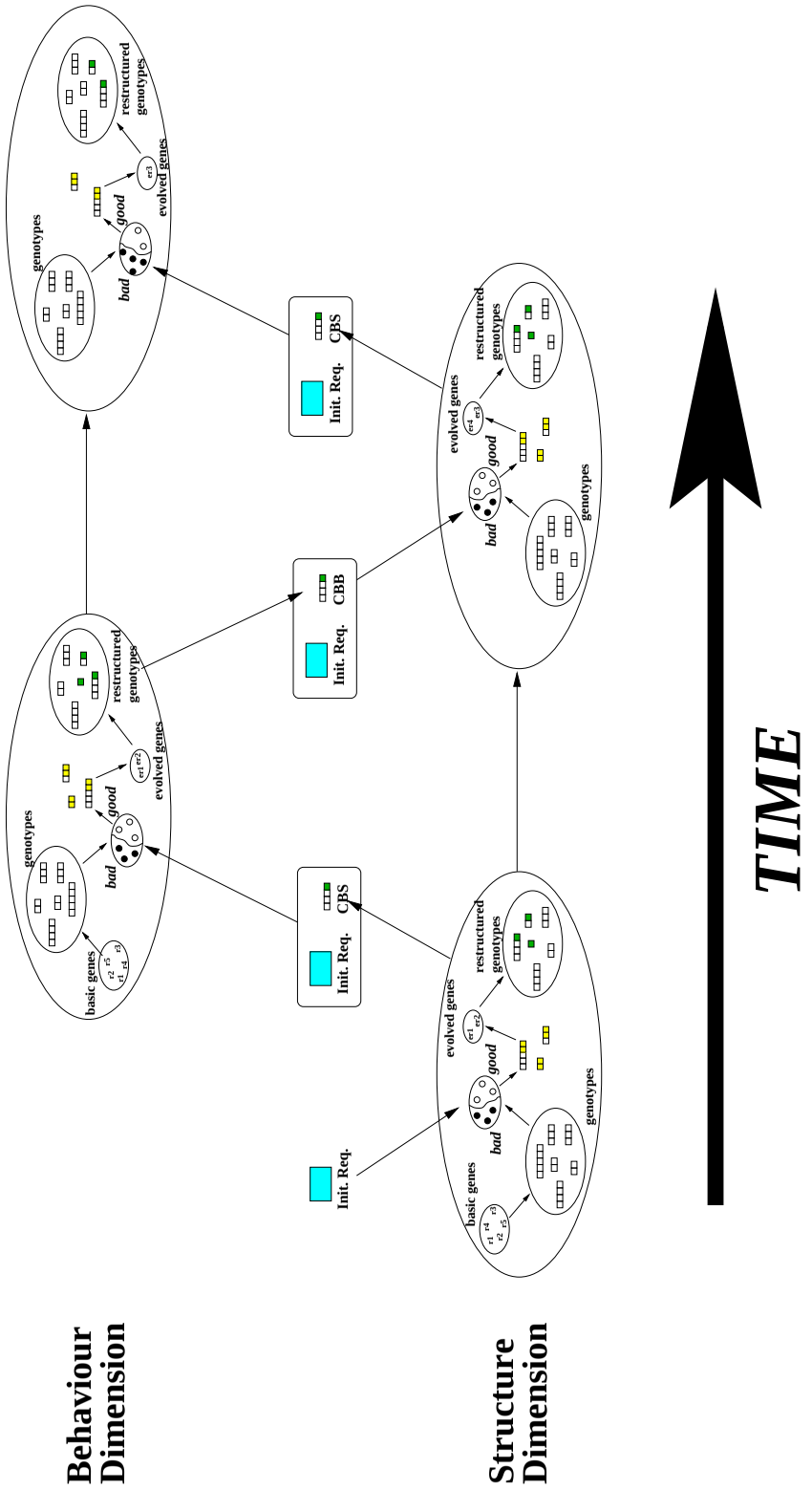


Fig. 3. Emergence in co-evolutionary design

4 Experiment

The design of a braced frame panel using CoGA1 is chosen to show co-evolution and the design of a floor plan layout using CoGA2 is used to illustrate emergence. These two domains are selected to demonstrate that the computational models of

co-evolution and emergence are applicable to quantitative design performance as well as qualitative.

For the domain of steel braced frame design, the representation of the problem-design spaces provides a range of design solutions that can be described geometrically and more than one design foci. The design of braced frames for buildings is done by a structural engineer with some constraints/requirements imposed by the architect's layout of the building. The design foci considered here are: design for architectural compatibility and design for structural resistance. These alternatives are represented as the following four design performance criteria:

f0: measure of closeness to initial requirements
f2: measure of conformance to bay layout
f3: measure of structural efficiency
f4: measure of structural integrity

In our combined gene approach, a genotype consists of two parts: the problem part and the solution part (figure 4). Including the focus of attention, the problem part has five parameters, while the solution has seven feature elements which are necessary to solve the problem.

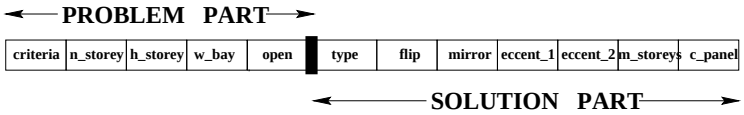
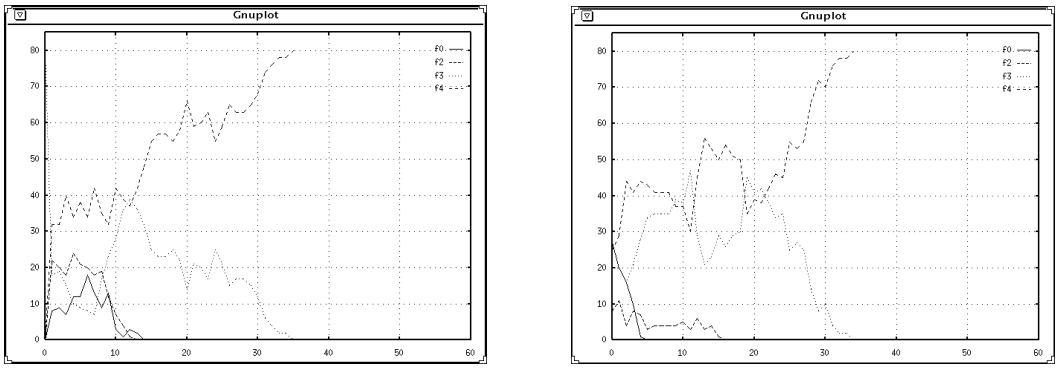


Fig. 4. Genotype template of a braced frame

Figure 5 shows the distribution of foci of two of the runs. Each graph carries four pieces of information, i.e. f0, f2, f3 and f4. The y-axis is the number of individuals which uses a particular focus in one generation, while the x-axis is the generation number. The two runs have remarkably different distribution. Run A is dominated by f4 throughout all generations. However, there is competition between f3 and f4 for Run B. Though f3 has gained momentum in the middle of the run, it lost ground to f4 finally. This diagram shows how the problem part responds to the current solution. Figure 5 shows how the fitness function changes over time, which is in effect an evolving fitness function. The graphs show the variation in focus for the braced frame solution. This demonstrates that our goal of exploration as a change in focus during the design process occurred through the rise and fall of the proportion of the population that used each design focus. More detailed discussion of this experiment can be found in (Maher & Poon 1995).

The other experiment is on emergence when a floor plan is designed. To design a floor plan encompasses two tasks: to draw a configuration of lines with spaces to represent walls and rooms, and to specify the usage of each of the spaces. When a design commences, the requirement, R_{init}, is usually given as a preliminary list of desired room-types and room-adjacency. Room adjacency is defined to be rooms which have a doorway which can lead a person from one room to another. The topology of a floor plan is drawn with turtle graphics. The drawing commands include up, down, left and right, with the addition of whether a door is to be introduced. When the drawing process completes, it is an unlabelled floor plan. An unlabelled plan is a plan where the spaces are not marked with their behaviour, i.e. room-types. Hence, an unlabelled floor plan is assigned with the required room



Run A ***Run B***

Fig. 5. Comparison of distribution of foci

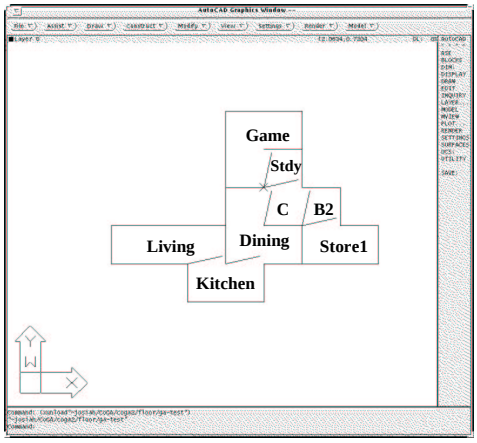
types as much as possible. The room-type assignment is under a set of constraints, e.g. the invalid adjacency of room-types, minimum and maximum size of a room-type, etc.

This experiment is carried out in CoGA2, hence, a current best (CB) is identified at the end of each generation. The CBB or CBS is incorporated with the R_{init} to form the fitness function for the next generation. CBB is the best performing phenotype of room-types and adjacencies. CBS is the best performing geometric layout of rooms and door locations. The run is set to terminate after 200 generations. Figure 6(a) is the best floor plan from the interaction between the problem and solution space. This design is far from perfect because the aim of this example is to demonstrate co-evolution and emergence; it is not targeted to generate a perfect floor plan, hence, the algorithm to assign room types to spaces is very primitive.

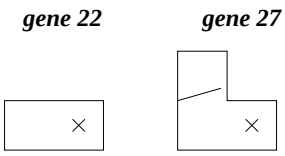
During this 200 generations, there are 8 "good" gene-pairs added to the basic structure genes (*eS*), of which 7 of them came from generation 62, and the last one appeared at generation 170. This is not an isolated case where it is observed that *eS* always come in a sudden *"burst"*. Figure 6(b) shows two of the *eS* during the evolution. In fact, gene-pair #27 is composed of another evolved-pair, and these two genes contribute to the best solution. Figure 6(c) is the emerging behaviour (room-adjacency) due to the interactions. There are 3 "good" gene-pairs added to the basic behaviour genes (*eB*) during the run. They occurred at generation 69, 147 and 171 respectively. The time to form *eB*s is more scattered and they do not come in bulk. These three *eB*s are also used by the best solution. There is no strong indication in this run (or with any other runs) that an *eB* has any relationship with any *eS*, the two phenomena happen independently. However, there is a stronger evidence that the best solution usually encapsulates some of the *eS*s and/or *eB*s. Emerging genes are obviously reused as building components as soon as they are identified.

5 Conclusion

This paper focuses on the characteristics of co-evolution in design and highlights the ecological relationship between the problem (expected behaviour) space and solution (structure) space. Computational approaches are proposed and difficult



(a) Best floor plan layout



(b) "Good" gene-pairs for structure

adjacency(corridor, bedroom2)
adjacency(game room, study)
adjacency(dining, kitchen)

(c) "Good" gene-pairs for behaviour

Fig. 6. Results of a floor plan layout design

implementation issues are addressed: the *evaluation* of individuals under a moving target, and the problem of *termination* when it is implemented in a computational paradigm. The concept of co-evolutionary design also leads to the possibility of *emergence*: specifically the emergence of behaviours and structures. The definition of emergence is cross-posted from ALife community and further argues that an evolving representation can be a means to implement emergence. The success of an algorithm to find interesting behaviour depends on the crafting of the credit assignment process. Co-evolution and emergence were illustrated in two different design domains. The results demonstrate these two phenomena can be observed in a computational approach to design.

Acknowledgments

This work is supported by an Australian Postgraduate Award.

References

Corne, D., Smithers, T. & Ross, P. (1994). Solving design problems by computational exploration, *in* J. S. Gero & E. Tyugu (eds), *Formal Design Methods for CAD*, North-Holland, Amsterdam.

Edmonds, E. & Soufi, B. (1992). The computational modelling of emergent shapes in design, *in* J. S. Gero & F. Sudweeks (eds), *Preprints of the Second International Conference on Computational Models of Creative Design*, Key Centre of Design Computing, University of Sydney, pp. 173–189.

Fogel, D. B. (1995). *Evolutionary computation: Toward a new philosophy of machine intelligence*, IEEE Press.

Fogel, L. J., Owens, A. J. & Walsh, Michael, J. (1966). *Artificial intelligence through simulated evolution*, John Wiley and Sons.

Gero, J. (1992). Creativity, emergence and evolution in design, *in* J. S. Gero & F. Sudweeks (eds), *Preprints of the Second International Round-Table Conference on Computational Models of Creative Design*, Key Centre of Design Computing, University of Sydney, pp. 1–28.

Gero, J., Damski, J. & Jun, H. (1995). Emergence in caad systems, *in* M. Tan & R. Teh (eds), *The Global Design Studio*, Centre for Advanced Studies of Architecture, National University of Singapore, pp. 423–438.

Gero, J. S. (1994). Towards a model of exploration in computer-aided design, *in* J. S. Gero & E.Tyugu (eds), *Formal Design Methods for CAD*, North-Holland, Amsterdam, pp. 315–336.

Gero, J. & Schnier, T. (1995). Evolving representation of design cases and their use in creative design, *in* J. Gero & F. Sudweeks (eds), *Preprints of the Third International Conference on Computational Models of Creative Design*, Key Centre of Design Computing, University of Sydney, pp. 343–368.

Gero, J. & Yan, M. (1994). Shape emergence by symbolic reasoning, *Environment and Planning B: Planning and Design* **21**: 191–218.

Goldberg, D. E. (1989). *Genetic algorithms: In search, optimization and machine learning*, Addison-Wesley.

Hybs, I. & Gero, J. S. (1992). An evolutionary process model of design, *Design Studies* **13**(3): 273–290.

Maes, P. (1990). Situated agents can have goals, *in* P. Maes (ed.), *Designing Autonomous Agents*, MIT/Elsevier, pp. 49–70.

Maher, M. L. & Poon, J. (1995). Co-evolution of the fitness function and design solution for design exploration, *Proceedings of IEEE International Conference on Evolutionary Computing*, IEEE, pp. 240–244.

Maher, M. L. & Poon, J. (1996). Modelling design exploration as co-evolution, *Microcomputers in Civil Engineering* **11**(3): 195–209.

Mitchell, W, J. (1993). A computational view of design creativity, *in* J. S. Gero & M. L. Maher (eds), *Modelling creativity and knowledge-based creative design*, Lawrence Erlbaum Associates, pp. 25–42.

Poon, J. & Maher, M. L. (1996). Emergent behaviour in co-evolutionary design, *in* J. S. Gero & F. Sudweeks (eds), *Artificial Intelligence in Design AID'96*, Kluwer Academic Publishers, pp. 703–722.

Simon, H. A. (1981). *The sciences of the artificial*, 2nd edn, MIT Press.

Steels, L. (1991). Towards a theory of emergent functionality, *in* J.-A. Meyer & S. W. Wilson (eds), *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, MIT Press, pp. 451–461.

Taylor, C. E. (1990). "fleshing out" artificial life ii, *in* J. D. F. Christopher G. Langton, Charles Taylor & S. Rasmussen (eds), *Artificial Life II: Proceedings of the Workshop on Artificial Life*, Addison-Wesley, pp. 25–38.