Applied Intelligence 16, 119–128, 2002

Evolutionary Design Systems and Generative Processes

PATRICK JANSSEN, JOHN FRAZER AND TANG MING-XI
School of Design, The Hong Kong Polytechnic University, Hung Hoon, Hong Kong
patrick.janssen@polyu.edu.hk
sdfrazer@polyu.edu.hk
sdtang@polyu.edu.hk

Abstract. Design tools that aim not only to analyse and evaluate, but also to generate and explore alternative design proposals are now under development. An evolutionary paradigm is presented as a basis for creating such tools. First, the evolutionary paradigm is shown to be the only successful design system on which this new phase of design tool could be based. Secondly, any characterisation of design as a search problem is argued to be a serious misconception. Instead it is proposed that evolutionary design systems should be seen as generative processes that are able to evaluate their own output. Thirdly, a generic framework for generative evolutionary design systems is presented. Fourth, the generative process is introduced as a key element within this generic framework. The role of the environment within this process is fundamental. Finally, the direction of future research within the evolutionary design paradigm is discussed with possible short and long term goals being presented.

Keywords: design, evolution, generative, environment, search

Tools for Design

A new phase of design tool is now under development. The use of the term 'tool' is not in the dismissive sense implied by the phrase 'a computer is just a tool' [1]. This dismissive attitude might have been justified when applied to the first phase of design tools that automate processes otherwise carried out by hand. The second phase, consisting of analysis tools that can simulate and measure the performance of design, made this attitude questionable. However, the third phase will make it totally inappropriate. These third phase tools are described as active as opposed to passive in that they will become an integral part not only of the manual design process but also of the cognitive design process. The task of these software tools is not only to analyse and evaluate, but also to generate and explore alternative design proposals. Such tools will free designers from 'design fixation' and the limitations of conventional wisdom, thereby allowing them to explore a huge number of possible proposals for a design problem [2].

Passive design tools play a relatively minor role in the cognitive design process. This minor role has allowed the traditional pre-computer design methodology to be adhered to. In this methodology, the design-makers are at the centre, controlling all aspects of the design process. The design-makers mediate any interaction between the design and the eventual design-users and between the design and the design tools. The proposed active tools on the other hand will initiate a new design methodology. This new methodology will relocate the tools at the centre of the design process. The design-users, the design-makers and also the tool-makers all participate in the creation of the design. However, in this new methodology, it is the active tools that mediate between these participants and the design. The tool-makers play a stronger role in defining the designs types towards which the tools are biased. The design-makers use these biased tools to explore possible designs. Finally, the design-users are empowered to alter and experiment within the boundaries set down by the design-makers.

These new tools will be contrivances that will enhance and extend the 'imaginative jump or creative leap' from one design proposal to the next. "Design as seen from the designer's perspective is a series of amazing imaginative jumps or creative leaps. But design as seen by the design historian is smooth progression or evolution of ideas that seem inevitable with hindsight. It is a characteristic of great ideas that they seem self evident and inevitable after the event. But the next step is anything but obvious for the artist/creator/inventor/designer stuck at that point just before the creative leap. They know where they have come from and have a general sense of where they are going, but often do not have a precise target or goal" [3].

Why Evolution?

It is often presumed that, in order to automate any part of the design process, one must start with a cognitive theory of how humans design. This is based on the assumption that humans offer the only example of a successful design system. However, alternative successful design systems do exist. One such system is biological evolution in nature, which has been evolving biological designs that far exceed any human designs in terms of complexity, performance and efficiency. Evolutionary programs use biological evolution in nature as a source of inspiration, rather than a phenomenon to be accurately modelled. There are a number of reasons why the choice of the evolutionary model is more appropriate.

Firstly, a cognitive theory of design presupposes an underlying general theory of cognition. Such a theory is, to say the least, an extremely distant goal. Conversely, the theory of evolution is well established and the field of evolutionary computation has benefited from a large amount of interest and research.

Secondly, the aim behind creating design tools is not to duplicate or mimic existing traditional design processes. Rather, the aim is to create innovative tools that challenge the design process, allowing designers to work in ways that were previously not possible. The tools to be developed reject the traditional architectural methodology "on the grounds that, first, the present architectural design process is fundamentally unsatisfactory in any known form and not worth imitating and, second, imitating the human process is unlikely in any case to represent the most imaginative use of a machine" [4].

Thirdly, the evolutionary process in nature is an extraordinarily impressive design system. "From the near-perfection of the streamlined shape of a shark to the extraordinary molecular structure of a virus, every living thing is a marvel of evolved design. Moreover, as biologists uncover more information about the workings of the creatures around us, it is becoming clear that many human design around us existed in nature long before they were thought of by any human, for example: pumps, valves, heat-exchange systems, optical lenses, sonar" [5].

Finally, evolutionary systems have proved to be extremely well suited to design problems. Rather than analysing one or even several proposals for a given design problem, the evolutionary system is, at any one time, considering whole populations of proposals. This parallel approach allows the system to legitimately arbitrate its own generative process.

Search, Search Spaces and Fitness Landscapes

Initially, evolutionary systems in design focused on optimisation of problems for which near optimal solutions were already well known. The question of improving the design is then characterised as a search problem. This idea of searching among a selection of candidate solutions gives rise to the search space concept [6]. This concept encompasses some notion of distance between candidate solutions. An algorithm for searching this space is a method for choosing which candidate solutions to test at each stage of the search. In most cases the next candidate solution(s) to be tested will depend on the results of testing previous sequences; most useful algorithms assume that there will be some meaningful relationships between 'neighbouring' candidate solutions—those close together in the space. As a consequence of the search space concept, the idea of a fitness landscape arises naturally. A fitness landscape is a representation of the space of all possible solutions along with their fitnesses. It is referred to as a landscape because the fitness values can form 'hills', 'peaks', 'valleys', and other features analogous to those found in physical landscapes. The task of the evolutionary system is to home in on the highest peaks in this landscape.

A large number of search algorithms have been developed in order to try and search as efficiently and thoroughly as possible. Hill-climbing is one of the best known. This algorithm utilises the iterative improvement technique whereby, each iteration, a new solution is searched for in the neighbourhood of the current solution. This is known as a point-to-point search since, at any one time, only a single point is being

processed. However, such point-to-point search algorithms are susceptible to stagnation at local peaks and have difficulty searching rugged fitness landscapes. In order to overcome these drawbacks, numerous alternative algorithms were introduced. The most successful alternatives discarded the point-to-point search strategy in favour of a parallel strategy, whereby whole populations of solutions are considered at any one time. Evolutionary algorithms, being based on the intrinsically parallel process of natural evolution, have consequently gained the reputation as being a very efficient and robust type of parallel search algorithm.

Design is Not a Search Problem

The idea that computers might become active intelligent tools is not new. As a result of the growing computer capabilities since the 1960's automated design engendered a great number of expectations. Unfortunately, most of these expectations were not met. With hindsight, it seems clear that machine intelligence had been overestimated and the complexities of the design processes had been underestimated. In many cases, designing was characterised as a problem of searching for an optimal design solution in a space of all possible designs.

The search process requires four preparatory steps to be taken prior to the actual searching of the space. First the problem must be clearly specified. A precise problem specification will delineate the boundaries of the search space. Second, the search space must be meaningfully structured. Structure ensures that neighbouring solutions are related to each other in some meaningful way, which in turn will allow the search procedure to make rational 'guesses' as to where the next best solution might be. Third, a fitness function must be defined. The fitness function must be able to compare all solutions found in any region of the search space. Fourth, a search procedure must be defined to search the space. Once these four preparatory steps have been taken, then the search can begin. However, the design process is incompatible with the first three steps.

First, during the design process, the problem specification does not remain static. Design problems are well known to be under-defined in that the problem specification provides only a very small part of the information required for a solution to be found. As a result, design will typically be a cyclical process whereby the search for solutions will continuously redefine the problem specification. Second, the structuring of the space of possible designs will also not remain static during the design process. The structuring of any search space will invariably impose a strong bias on the types of solutions that can be found. Design processes continually explore and manipulate such biases in conjunction with the gradual cyclical development of a design proposal. Third, the direct comparison of all possible design proposals is impossible. Although alternatives in the same region of the space of possible designs can be compared to one another, distant alternatives will stem from radically different roots and will therefore not be comparable in any straightforward manner.

A typical design problem creates an ill-defined, unstructured and vast multidimensional design space. Metaphors such as search and search spaces aim to elucidate the design process. However, they do not accurately reflect the reality of the design process and thereby actually result in further confounding the issue. "This is why it is misleading to talk of design as a problem solving activity—it is better defined as a problem finding activity. This has been very frustrating for those trying to assist the design process with computer based, problem solving techniques. By the time the problem has been defined it has been solved. Indeed the solution is often the very definition of the problem" [7].

The Library of Babel, described by Jorge Luis Borges [8] allows one to conceptualise such vast search spaces. This is an imaginary library of all possible books that are 410 pages long. There are $25^{1312000}$ books in this library, a figure which is astronomically larger than the number of particles in the visible universe. The philosopher Daniel C. Dennett exploits the concept of the Library of Babel in order to mentally picture the vastness of such search spaces. "Imagine travelling in a spaceship through the *Moby Dick* galaxy of the Library of Babel. This galaxy itself is vastly larger than the whole universe itself, so no matter what direction you go in, for centuries on end, even if you travel at the speed of light, all you see are virtually indistinguishable copies of *Moby Dick . . . David Coperfield* is unimaginably distant in this space" [9]. Through this visualisation, he brings into question the applicability of many of our usual ideas about location, about searching and finding and other such mundane and practical activities.

As Dennett points out, if one is to work within such vast spaces it is essential to realise the difference between what is possible in principle and what is

possible in practice. Thus, although it is in principle possible to exhaustively search such spaces, practically it is impossible. A key concept in what is practical is the idea of distance from the region that has already been conquered within the vast universe of possible designs. This region represents what actually exists and is in itself huge but is nevertheless vanishingly small in relation to the whole design universe. What counts as possible will be anything that can be obtained by travelling from this conquered region outwards. The resulting notion of possibility will have an important property: some designs will be more possible than others—that is nearer in the multidimensional universe, and more accessible, easier to get to.

A more accurate metaphor for elucidating the design process might be evolution in nature. It is clear that natural evolution has not been a search process that found the optimal organisms, species and ecosystems that we find today in a vast space of possible designs. This is reflected by the fact that if the process were repeated, the chances of it discovering the same organisms, species and ecosystems would be infinitesimally small [10]. Instead, the course of natural evolution has been the result of combination of history and accident. When John Holland first conceived of un-natural evolution, it was seen as a direct analogy with natural evolution [11]. The evolutionary design paradigm presented reinstates the direct analogy between natural evolution and un-natural evolution.

Natural Evolution

The evolutionary process acts through selection, transmission and variation (Universal Darwinism) [12]. In nature, large populations of organisms individually reproduce to generate new offspring. Natural selection ensures that more successful creatures are more likely to reproduce. Transmission ensures that the offspring inherit some features of their parents. Variation ensures that offspring also have some entirely new features. Thus, although the evolutionary process is not explicitly specified anywhere, when these three ingredients are present the process of evolution can emerge. Furthermore, this process can emerge regardless of the medium, be it biological, computational, cognitive, etc.

Within a biological medium, each organism is grown from an encoded set of instructions, known as the genotype, into a fully developed organism, known as the phenotype. An organism's genotype consists of a set of genes, where a gene can be thought of as a group of manufacturing instructions. This process of growing the phenotype from the genotype is known as an embryogeny and is a key element in understanding the interplay between selection, transmission and variation.

Transmission and variation both act at the level of the genotype. Transmission occurs through sexual reproduction, whereby a 'copy and paste' mechanism combines genetic material from both parents and transfers it to the offspring. Variation occurs through errors in this copying process. Natural selection, however, acts at the level of the phenotype. Only those organisms whose phenotype is well suited to the current environment will survive long enough to be able to reproduce.

Thus, those organisms that have phenotypes that are unfit will not reproduce and therefore their genes will not survive. On the other hand, organisms that are fit will reproduce thereby allowing their genes to survive, possibly long after the organism has already died. The key point is that selection, transmission and variation are able to edit the genes of a whole species to produce a revised set of genes specifically suited to the current environment. In the resulting population, the frequency of genes that are able to construct fit individuals will have increased.

Un-natural Evolution

The evolutionary systems mirrors this process. A population of alternative designs is maintained. Natural selection is simulated with more successful designs having a higher chance of selection for reproduction. The process of reproduction will then generate new designs that inherit some features from their parent designs and have some entirely new features.

A wide range of evolutionary algorithms exist. The four main types are genetic algorithms [13], evolutionary programming [14], evolution strategies [15] and genetic programming [16]. Of these, genetic algorithms are the best known and probably the most widely used. Genetic algorithms were developed by Holland and his students as a way of formally studying the phenomena of adaptation as it occurs in nature. He presented the genetic algorithm as an abstraction of biological evolution [17]. The algorithm maintains a population of individuals where each individual consists of a genotype and a phenotype. By using a kind of natural selection together with genetics-inspired operators of crossover, mutation and inversion the algorithm is able to move from one generation of to the next. Each generation some individuals will die off and others will be born.

Those individuals that die off will, on average, have phenotypes that find themselves in the lower levels of the fitness landscape, whereas those that are born will, on average, have phenotypes that find themselves up in the higher levels, towards the peaks. Thus the population is continuously edited ensuring that each new generation will have a slightly higher average fitness than the previous one.

For any given problem, many evolution programs can be defined. Such programs will range from being very domain independent, referred to as 'weak' methods, to very domain specific, referred to as 'strong' methods. Classical genetic algorithms fall into the category of weak methods. Although they must have a domain specific mapping from the genotype to the phenotype, a domain specific phenotype representation and domain specific fitness criteria, other parts of the algorithm are totally domain independent. Three fundamental elements of this domain independent core of the genetic algorithm are the selection process, the genetic operators and the genotype representation. Weak evolution programs, such as the classical genetic algorithm, have certain advantages over other stronger alternatives. Since they make few assumptions about the problem domain, they enjoy wide applicability. The ruthless abstraction and simplification of these algorithms has also allowed a strong theoretical foundation to be laid down [18, 19] .

However, a number of researchers in various specialised fields, including design, have found that most real-world problems can not be handled with the classical genotype representation and the corresponding genetic operators. The representation consists of fixed length binary strings and the genetic operators consist of binary crossover and binary mutation. When applied to complex problems, these representations and operators do not allow the knowledge within the domain to be succinctly described. In order to overcome this hurdle, stronger assumptions must be made about the problem domain [20]. Thus, whereas genetic algorithms require key elements to remain domain independent, evolutionary systems in fields such as design will typically allow them to become more complex specialised domain specific components.

One area that has lately been becoming gradually more complex is the mapping process from genotype to phenotype. The use of genetic algorithms in optimisation problems required only a straightforward direct mapping from the binary string to the parameter being optimised. However, in attempting to capture a wide range of alternatives, evolutionary systems are now employing highly complex problem specific developmental processes. In evolutionary design systems, this process takes the form of a generative process that starts with encoded design code scripts and decodes them into fully developed design proposals [21, 22].

The Generative Evolutionary Paradigm in Design

The cognitive process of design can be understood as a wetware contrivance created by natural evolution that increases travel distance in design universe. Thus, individuals whom are in possession of this cognitive wetware contrivance are able to explore regions of the design universe otherwise unreachable. These wetware contrivances have created numerous hardware contrivances such as the pencil and the drawing board in order to make travel in design universe even more daring. Furthermore, software contrivances such as CAD programs or even evolutionary design systems will further enhance and extend the regions of the design universe that can be reached. Thus these software systems take their place alongside other wetware and hardware contrivances as instruments for undertaking vanishingly diminutive assaults on the vast design universe [23].

The proposed generic framework for a evolutionary design system is shown in Fig. 1 and consists of four

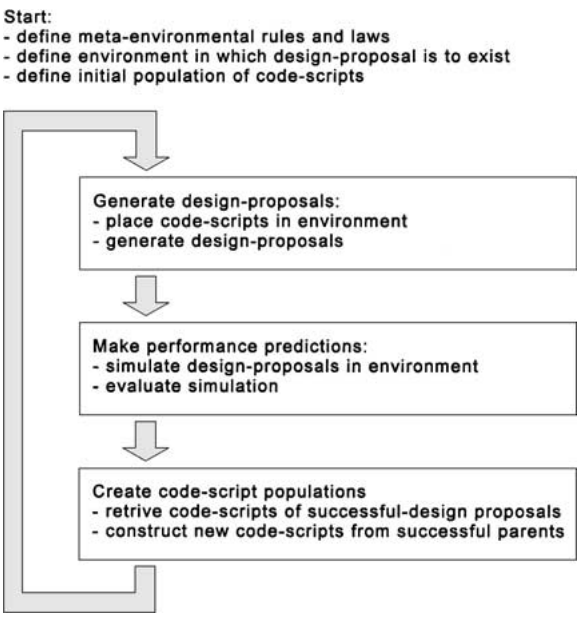

Figure 1. Diagram showing four main phases of the evolutionary cycle.

main phases; start, generate design proposals, make performance predictions and create populations. The framework drops the biological references and instead refers to the genotype as the code script and the phenotype as the design proposal. In the start phase, various representations are defined and populations are initialised. The generate phase creates a population of design proposals from a population of code scripts through a form of development or growth within an environment. The prediction phase simulates each design proposal in the environment and then evaluates its performance. In the create population phase a new code script population is created by copying and transforming the most successful code scripts from the previous population. This new code script population then becomes the start of the next cycle.

Environments

Any evolutionary system must operate within some sort of pre-existing universe. The laws and meta-representations of this universe remain unaffected throughout the evolutionary process. In the natural world the laws are analogous to fundamental physical laws and the meta-representations are analogous to molecular chemistry. They define a kind of meta-environment, within whose constraints the evolutionary process must proceed. Stuart Kauffman describes this as a collaboration. "The actual morphologies of organisms must also be viewed as a collaboration between the self-ordered properties of physicochemical systems together with the action of selection. Oil droplets are spherical in water because that is the lowest energy state. . . . The genomes capacity to generate a form must depend on very many physicochemical processes constituting a panoply of developmental mechanisms beyond the sheer capacity of the genome to co-ordinate the synthesis of specific RNA and protein molecules in time and space. Morphology is a marriage of underlying laws of form and the agency of selection" [24].

In un-natural evolutionary systems this non-evolvable meta-environment is traditionally much more specialised that in its natural counterpart. For example, the meta-environments for optimisation systems include a complete (usually implicit) specification of the overall structure of the design. In such simplified systems there is no potential for the evolution of the design; only certain parameter values can evolve. Evolutionary design systems, on the other hand, aim to be more generic in that they employ complex representational schemas that allow a much wider variety of designs to be evolved. Such systems will commonly rely on a generative process that takes an encoded code script and expands this to a decoded design. Thus, compared with optimisation systems the meta-environments of evolutionary design systems need to be more abstract and generalised.

The generative process is in essence a simulation of an invented meta-environment. This meta-environment has been designed such that, when a code script of a certain representation is placed within it, it will 'take root' and grow into a design proposal. The importance of the environment within this process must be stressed. Consider the example of the acorn and the oak tree. The environment can affect the generative process in three main ways. Firstly, the environment can affect the way that the code script grows into the design. For example, a tree might grow towards the area with most daylight. Secondly, the environment can affect the designs chances of reproducing. For example, in a forest were most trees are short, a tall tree might have greater chances of survival and therefore also have greater chances of 'reproducing'. Lastly, the relative success of a particular species within a particular environment can change that environment. For example, in a forest of short trees, a particular species of tall tree might do rather well and might eventually come to dominate the forest, thereby drastically changing the environment.

Those aspects of developmental processes that are due to external influences rather than internal genetic influences, are referred to as epigenetic factors. Evolutionary design systems tend to disregard epigenetic factors, instead focusing purely on the genetic factors. In such a scenario, the environment within which the development takes place is totally independent from the environment within which the proposed design must compete [25]. For example, when a hen lays an egg, the interior of the egg can be described as the environment in which embryogenesis occurs. Subsequently, once the egg hatches, it is the outside world that becomes the environment in which the chick must live. However, within the domain of design, the environment of the outside world plays a fundamental part in assessing the appropriateness of the proposed design. To disassociate the developmental process from the environmental influences is to deny the evolutionary process a valuable symbiotic interaction between the environment and design proposals.

The Generative Process

In nature a one-dimensional genetic code is able to specify a three dimensional animal. An organism's shape and that of its tissues derive from the shapes of collections of cells of a variety of types. The 'genetic program' controls cell differentiation during the development of the adult from the fertilised ovum. Different cell types arise and differentiate and, ultimately, in a human, form several hundred cell types. Each cell in the human body contains essentially the same genetic instructions. These instructions include the structural genes coding for about 100,000 different proteins. The shapes of these proteins confers properties and functions on them; for example certain shapes might allow them to fit together with other proteins to form cell structures, while others might allow it to bind to chemicals and change the speed with which they react [26].

The question that remains is what controls the expression and repression of genes? Kauffman describes this as follows "Cell types differ because different subsets of genes are 'active' in different cell types. . . . The expression of gene activity is controlled at a variety of levels, ranging from the gene itself to the ultimate protein product. It is this web of regulatory circuitry which orchestrates the genetic system into coherent order. That circuitry may comprise thousands of molecularly distinct interconnections" [27].

Computational models of generative processes face similar questions. How can complex structures emerge using only relatively small quantities of initial data? Such models typically attempt to develop rising levels of organisation through the creation of self organising complex systems. These systems encode rules that will self-organise to produce a morphology. Furthermore, they often employed fractal like iterative growth processes in order to attempt to replicate the types of forms found in nature. Two types of model will be compared; accretive growth models and Lindenmayer systems. The accretive growth models simulate exogenous mechanisms of growth in that components of the growing structure communicate through the surrounding space. Thus the rules that control the generative process are assumed to reside within the meta-environment. In contrast L-systems simulate endogenous control mechanisms, which rely on information flow within the developing structure. In this case, the rules are assumed to reside in the structure itself. Nature employs both these strategies. "In nature, endogenous and exogenous control mechanisms are often combined. For example, the development of a tree is affected by the genetically controlled formation of apices, the flow of water, nutrients, and phytohormones through the branching structure, and the plant response to the environmental factors, such as the shading and crowding of branches" [28].

Historically, the first model of morphogenesis was proposed by Alan Turing in 1952. He presented a reaction-diffusion model for the chemical basis of morphogenesis. The model is based on substances called morphogens, that are made responsible for pattern formation. A system of partial differential equations describes changes in morphogene concentrations while the substances diffuse and react in space and over time. The reaction-diffusion model assumes that the medium on which the reaction diffusion originally takes place, be it a surface or a line, does not grow. One of the first computer models of growing biological structures was proposed by Eden [29]. The simulation takes place in a square grid. A single initial particle is placed in the centre of this grid. The subsequent particles are attached, one by one, to randomly chosen points to the border of the structure formed in the previous steps. Ulam [30] proposed extension of Eden's model, using a formalism known as the cellular automaton. A cellular automaton consists of cells, arranged in a (usually) square grid, and communicating with each other. Ulam assumed that new cells can be added on the border of the structure formed so far only if these cells do not collide with each other or with the previously added cells. The development of the resulting branching structure, he called *Maltese crosses*.

In 1968, Lindenmayer invented a formalism that yields a mathematical description of plant growth known as an L-system. L-systems are remarkably compact. An L-system consists of a specialised cell and a description of how new cell types can be generated from old cell types. The seed cell is known as an axiom, and for a particular L-system, all growth starts from the same axiom. The description of how to grow new cell types from old cell types are known as production rules. The rules specify a simple rewriting scheme that involves taking the axiom and substituting as many of the symbols as we can, as specified by the rules. After performing the substitution on the axiom, we will have another string of symbols. We can apply the substitution rules to this result to get a third string of symbols, and so on. By itself the string means absolutely nothing, but in the context of a device that can interpret each symbol as a simple instruction, the string can represent the building plan for a fractal structure.

The two types of growth model represent two possible methods of interaction between the generative process, the environment and the meta-environment. The accretive growth models rely on the meta-environment to store the growth rules and rely on the environment for exogenous communication. The Lindenmayer system also relies on the meta-environment for storing the production rules. However, the environment is not utilised for communication, instead relying on an endogenous mechanism.

Evolving Evolvability

In nearly all cases, evolutionary systems do not allow the rules and representations themselves to evolve. However, in nature, the DNA framework, within which the evolutionary process is currently operating, was not created prior to starting the evolutionary process. The rules and representations of the DNA framework themselves evolved from the RNA framework. The reasons for its preferability are clear: by being double stranded, the DNA rules and representations permitted a system of error correcting enzymes, which could repair copying errors in one strand by reference to its mate. This made the creation of longer more complicated genomes possible [31].

Computational systems that allow for the evolution of the rules and representations that define the individual designs thus aim to become even more generic. These systems allow evolution to directly influence both the generative and transformative processes. The predictive process, on the other hand, can only ever be indirectly influenced by evolution. Within the generative process, the evolvable elements might include the representation of the code script. Within the transformative process, the code script transformation operators might be evolved. However, the predictive process is only affected by changes in the evaluation environment. Thus, although the evolutionary process can not influence this environment directly, it can affect the evaluation environment for any one species by influencing the evolution of other, competing, species.

The key step in attempts in allowing evolutionary systems to take control of these rules and representations is to include their definition in the code script. For example, the Lindenmayer generative process could be used within an evolutionary system. The code script of such a system could then either be restricted to defining the initial starting seed or it could additionally define the production rules by which that seed is going to grow. The latter system would obviously allow a much wider range of designs to be generated. However, in order for the evolutionary system to attain the level of control found in nature, not only the production rules but also the language within which these rules are written needs to be stored within the code script. But this implies that one of the representations to be defined in the code script is the code script representation itself! In order to unravel this seeming contradiction, the evolutionary process must be seen as a sequence of historical events. In Nature, for evolution to be able to influence the 'language' in which a code script is written, two consecutive steps must take place. First, a species must evolve with the propensity to be able to interpret some new, hitherto non-existent, type of code script 'language'. Second, that same species must further evolve so that, rather than passing down its code script to the next generation in the old 'language', it is now able to produce and pass down code written in the new 'language'. Only in this way can the new code script get a hold since the creation of any new code script 'language' that precedes the mechanisms to interpret that 'language' is sure to never proceed past the embryonic stage.

For evolutionary systems to be able to come anywhere near the awesome creative power of evolution in nature, similar meta-representational schemas would have to be created. In theory, such systems might then, given enough time, generate unfathomable designs incomprehensibly well adapted to the most complex environments. However, the creation of such generic systems is extremely complex and fraught with problems. At present, evolutionary systems are relatively simple and attempts at including the simplest representational evolvability is proving to be a serious challenge. Furthermore, even if the creation of such a generic system was feasible, time is a critical factor. Nature has had billions of years to create the complexity of organisms, species and ecosystems. Within an evolutionary system, billions of years must be compressed to hours. Despite the seeming unfeasibility of these grandiose goals, the creation of practical evolutionary software systems that are able to exploit a highly constrained version of the creativity displayed by natural evolution are becoming feasible. Within this context, two avenues of future development seem to present themselves; one short term and one long term.

In the short term various non-evolvable rules and representations can be explored. These rules and representations are analogous to the 'design schemas' used by

most designers. “Most designers employ a methodology highly personalised yet can often be generic when the designer’s body of work is taken as a whole. It is part of their working method and hence characterises their ‘style’ by which they are known. With an artist this style maybe a clearly recognisable graphic technique, as with say the drawings of Beardsley, a painting technique, such as the impressionists, or perhaps a distinctive choice of palette, say Titian. With architects and designers, the same is true. In some cases the style is also immediately recognisable from visual clues such as the work of Gaudi or Mackintosh. But often the style is more organisational or procedural or concerned with more abstract space and form such as with the work of Frank Lloyd Wright, although even with Wright the large roofs and low eaves of his early houses are an immediate stylistic give away. This personalised but generic methodology can be described as a *design schema* in that it is an abstract conception of what is common to all designs” [32]. These design schemas can be relatively domain specific, thereby allowing for certain concrete and immediately useful results to be achieved.

Within the vast universe of possible designs, design schemas are abstract formulations that attempt to cover certain regions of the already conquered region but also to influence the assaults on the as yet unconquered regions. This balance between covering what exists today and influencing what might exist tomorrow can be clarified by considering the two extremes that tend to be adhered to today. On the one hand, the representational schemas commonly utilised in optimisation algorithms warily advance the frontier of the conquered region by the smallest possible amount. On the other hand, the so called generic representational schemas utilised in some exploration algorithms foolishly attempt to advance the frontier far into the unknown. Design schemas propose a different strategy that is neither totally rigid nor totally flexible.

In the long term, these new insights gained from work on specialised schema can be utilised to start experimenting with the implementation of highly controlled meta-representational schemas. These meta-representations will not discard the design schema, rather they will allow for the evolution of the schemas themselves. Such experimentation may some day become the basis for the development of evolutionary systems that are able to go beyond the limitations of human imagination.

Conclusion

An evolutionary paradigm has been presented as a basis for creating active design tools that are able to generate and explore alternative design proposals. The generative process within an environment has been argued to be a key component of any such tools. Intrinsic properties of the generative processes are believed to lead to much more complex morphologies than those achievable with direct mapping.

However, a number of constraints upon the evolvability have been discovered. The Darwinian process of mutation, recombination and selection is not universally effective in improving complex systems like computer programs or chip designs. For adaptation to occur, these systems must posses “evolvability”. “Among the earliest experiments in evolutionary computation, Friedberg [33] attempted to evolve functioning computer programs by mutating and selecting the code but found that mutations effectively randomised the behaviour of the programs, and adaptive evolution was impossible. . . . It became understood that the mutation/selection process is not universally effective in producing adaptation if favourable mutations can not be produced. . . . In contrast to Friedberg’s results, Koza [34] succeeded in designing computer programs that perform well on complex tasks by recombining branches of parse trees for the programs. . . . . The difference between Friedbergs and Koza’s systems was in the representation of the computer programs and the way genetic operators act on them” [35].

In Nature, organisms are built up by the process of growth. A mutation, if it has to change the shape of an organism, will normally do it by adjusting the process of embryonic growth. The kind of mutations that are available for natural selection to work on will depend upon the kind of embryological process that the species possesses. Dawkins describes how one embryology might be better than another. “Some types of embryology may, in some sense, be ‘better’ at evolving than others. The kinds of variations thrown up by some types of embryology may be more evolutionarily promising than the kinds of variations thrown up by other embryologies. Certain kinds of embryology may be prone to vary in certain ways; other kinds of embryology tend to vary in other ways. And some of these ways may be, in some sense, more evolutionarily fruitful than others” [36]. The success of any evolutionary design system will depend on how readily the generative process allows evolution to emerge.

The factors that influence this dependence are yet to be unravelled.

Acknowledgments

Our research project is supported by a UGC PhD project grant from the Hong Kong Polytechnic University.

References

1. J.H. Frazer, "Can computers be just a tool?," in *Systemica: Mutual Uses of Cybernetics and Science*: Amsterdam, vol. 8, pp. 27–36, 1991.
2. P.J. Bentley, "An introduction to evolutionary design by computers," in *Evolutionary Design by Computers*, edited by P.J. Bentley, Morgan Kaufman: San Francisco, CA, pp. 1–73, 1999.
3. J.H. Frazer, "Creative design and the generative evolutionary paradigm," in *Creative Evolutionary Systems*, edited by P.J. Bentley and D.W. Corne, Morgan Kaufmann: San Francisco, CA, to appear.
4. J.H. Frazer and J. Connor, "A conceptual seeding technique for architectural design," in *PArC79, proceedings of International Conference on the Application of Computers in Architectural Design*, Online Conferences with AMK: Berlin, 1979, pp 425–434.
5. P.J. Bentley, "An introduction to evolutionary design by computers," in *Evolutionary Design by Computers*, edited by P.J. Bentley, Morgan Kaufman: San Francisco, CA, pp. 1–73, 1999.
6. L. Kanal and V. Cumar (Eds.), Search in Artificial Intelligence, Springer-Verlag: Berlin, 1988.
7. J.H. Frazer, "Creative design and the generative evolutionary paradigm," in *Creative Evolutionary Systems*, edited by P.J. Bentley and D.W. Corne, Morgan Kaufmann: San Francisco, CA, to appear.
8. J.L. Borges, "The library of Babel," in *Labyrinths: Selected Stories & Other Writings*, translated by James E. Irby, New Directions: New York, pp. 51–58, 1964.
9. D.C. Dennett, Darwin's Dangerous Idea, Evolution and the Meanings of Life, Simon & Schuster: New York, 1995.
10. S.J. Gould, Wonderful Life: The Burgess Shale and the Nature of History, Norton: New York, 1989.
11. J.H. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press: Ann Arbor., 1975.
12. R. Dawkins, "Universal Darwinism," in *Evolution from Molecules to Men*, edited by D.S. Bendall, Cambridge University Press: Cambridge, ch. 20, pp. 403–425, 1983.
13. J.H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press: Ann Arbor., 1975.
14. L.J. Fogel, *Biotechnology: Concepts and Applications*, Englewood Cliffs, NJ: Prentice Hall, 1963.
15. I. Rechenberg, Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien der Biologischen Evolution, Frommann-Holzboog Verlag: Stuttgart, 1973.
16. J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press: Cambridge, MA, 1992.
17. J.H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press: Ann Arbor., 1975.
18. K De Jong, "An analysis of behaviour of a class of genetic adaptive systems," Doctoral Dissertation, University of Michigan, Dissertation Abstract International, 36(10), 5140B. (University Microfilms No 76-9381).
19. J.H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press: Ann Arbor., 1975.
20. Z. Michalewicz, *Genetic Algorithns + Data Structures = Evolution Programs*, Sringer-Verlag: Berlin, 1996.
21. J.H. Frazer and J. Connor, "A conceptual seeding technique for architectural design," in *PArC79, Proceedings of International Conference on the Application of Computers in Architectural Design*, Online Conferences with AMK: Berlin, pp. 425–434, 1979.
22. P.J. Bentley, "An introduction to evolutionary design by computers," in *Evolutionary Design by Computers*, edited by P.J. Bentley, Morgan Kaufman: San Francisco, CA, pp. 1–73, 1999.
23. D.C. Dennett, *Darwin's Dangerous Idea, Evolution and the Meanings of Life*, Simon & Schuster: New York, 1995.
24. S. Kauffman, *The Origins of Order: Self Organisation and Selection In Evolution*, Oxford University Press: New York, 1993.
25. P.J. Bentley, "An introduction to evolutionary design by computers," in *Evolutionary Design by Computers*, edited by P.J. Bentley, Morgan Kaufman: San Francisco, CA, pp. 1–73, 1999.
26. G. Edelman, Brilliant Air, Bright Fire: on the Matter of the Mind, Penguin Group, 1992.
27. S. Kauffman, *The Origins of Order: Self Organisation and Selection In Evolution*, Oxford University Press: New York, 1993.
28. P. Prusinkiewicz, "Visual models of morphogenesis," *Artificial Life*, vol. 1, no. 1/2, pp. 67–74, 1994.
29. M. Eden, "A two-dimensional growth process," in *Proceedings of Fourth Berkeley Symposium on Mathematics, Statistics, and Probability*, University of California Press: Berkeley, vol. 4, pp. 223–239, 1960.
30. S. Ulam, "On some mathematical properties connected with patterns of growth of figures," in *Proceedings of Symposia on Applied Mathematics, American Mathematical Society*, vol. 14, pp. 215–224, 1962.
31. D.C. Dennett, *Darwin's Dangerous Idea, Evolution and the Meanings of Life*, Simon & Schuster: New York, 1995.
32. J.H. Frazer, "Creative design and the generative evolutionary paradigm," in *Creative Evolutionary Systems*, edited by P.J. Bentley and D.W. Corne, Morgan Kaufmann: San Francisco, CA, to appear.
33. R.M. Friedberg, "A learning machine," *IBM Journal, Research and Development*, vol. 3, Part II, pp. 183–191, 1959.
34. J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, 1992.
35. G. Wagner and L. Altenberg, "Complex adaptations and the evolution of evolvability," *Evolution*, vol. 50, no. 3, pp. 967–976, 1996.
36. R. Dawkins, *Climbing Mount Improbable*, Norton: New York, 1996.