Chapter 15
Evolutionary Processes as Models for Exploratory Design

Long Nguyen*, Daniel Lang*, Nico van Gessel, Anna K. Beike, Achim Menges, Ralf Reski, and Anita Roth-Nebelsick

Abstract Biological evolution drives morphological diversity via genetic variation and results in a high level of adaptation, performance and resource efficiency. However, "biological design" arising from evolution is often counterintuitive and unexpected in a non-linear way. Evolutionary processes are undirected and very good at exploring novel design possibilities in an open-ended manner. Biological evolution thus differs profoundly from the gradualistic and constantly converging character of technical optimization with defined and static fitness functions. Evolutionary algorithms based on Darwinian principles are mainly developed for solving multi-criteria problems in technology. Technical goals are defined as fitness functions and the evolutionary mechanisms of selection, heredity, reproduction and mutation are employed as stochastic optimization processes. These metaheuristic algorithms do not include recent insights into micro- and macroevolutionary mechanisms derived from genomics, phylogenomics and population genomics. Similar to natural evolution, the architectural design process is an open-ended process exploring possible solutions. However, in order to navigate this vast and dynamic design space, most design methodologies in architecture are based on a typological approach. The designers, based on their knowledge and understanding of the problems, usually limit the solution space to a particular structural, constructional, spatial or programmatic type that is iteratively adapted to the particular design requirements. The constraints inherent in typology-based

* Both authors contribute equally for this chapter.

L. Nguyen (✉) • A. Menges
Institute for Computational Design (ICD), University of Stuttgart, Keplerstraße 11, 70174 Stuttgart, Germany
e-mail: long.nguyen@icd.uni-stuttgart.de

D. Lang (✉) • N. van Gessel • R. Reski
Plant Biotechnology, University of Freiburg, Schänzlestraße 1, D-79104 Freiburg, Germany
e-mail: daniel.lang@biologie.uni-freiburg.de

A.K. Beike • A. Roth-Nebelsick
Stuttgart State Museum of Natural History, Rosensteintraße 1, D-70191 Stuttgart, Germany

J. Knippers et al. (eds.), *Biomimetic Research for Architecture and Building Construction*, Biologically-Inspired Systems 9, DOI 10.1007/978-3-319-46374-2_15

design methodologies exclude a vast range of potentially more effective and better design variants. In contrast, the dynamics of biological evolution suggest ways of continuously expanding the design space towards new and unexplored possibilities, that can potentially in a new set of typologies that still satisfy the constraints. Thus, in architecture, evolutionary processes are more relevant as exploratory processes than as optimization tools.

15.1 Introduction

Biological innovation and morphological diversity are the product of evolutionary processes that are ultimately driven by genetic variation and diversification resulting in a high level of adaptation, performance and resource efficiency. Early on in the history of computer science, ideas arose for the adoption of evolutionary principles for the implementation of digital optimization tools that gradually refine a "population" of solutions according to certain fitness criteria. In his essay "Computing machinery and intelligence", Alan Turing (Turing 1950) compared the programming and education of a "child" learning machine to the process of natural evolution. The subsequent transition from theory to practice was accomplished by the digital simulation of evolutionary processes in the mid-1950s to early 1960s of the twentieth century (Fraser 1958; Barricelli 1962). The field gained further momentum with the work carried out on evolutionary strategies (Rechenberg 1971; Schwefel 1974). Genetic algorithms (GA) became popular with the book *Adaptation in Natural and Artificial Systems* (Holland 1975). This research gave birth to a new field in computer science and artificial intelligence known as evolutionary computing. The algorithms emerging from this field are collectively known as "evolutionary algorithms" (EA) and usually maintain a population of potential solutions that are subjected to recombination, mutation and selection for retention in the population by using predefined "fitness" criteria.

The underlying view of natural evolution dates back to the beginning of the twentieth century when the founders of population genetics reconciled Darwin's ideas concerning natural selection with Mendelian genetics into what is now known as the Modern Synthesis or Neo-Darwinism. Although the work on GAs largely remained theoretical until the 1980s, once the first GA conferences were held and General Electric had sold the first software including a GA, EAs can be considered as very early examples of biomimetics (mimicking a process found nature). Since its origins, the field of evolutionary computing has substantially diversified and EAs have been applied to many problems, including computational design. Surprisingly, except for a few instances of adaptations of additional genetic or evolutionary concepts such as transposable elements (Simoes and Costa 2000), the field largely remained uncoupled from biological research. Furthermore, if we consider, for example, the insights gained from evolutionary developmental biology or genome sequencing, biological research undoubtedly has progressed beyond the view of the Modern Synthesis. However, limited to no exchange has occurred between the disciplines of biology, computer science and design over the last decades. EAs and

evolutionary simulation have also not attracted much attention in the biological mainstream. EAs depend on the exact quantitative knowledge of evolutionary rates to implement these principles. In EAs, the employed rates are usually tuned based on experience and adjusted to the problem. To our knowledge, *in silico* and *in vivo* evolutionary rates have not been consolidated so far. As we will see in the later sections of this chapter, although the terminology often is adopted from biological concept generators (indicated by italics in the subsequent text), in many cases, the meaning, i.e. the implementation, differs considerably (and sometimes fundamentally) from the biological source.

Many of the "designs" shaped by biological evolutionary processes often appear counterintuitive or unexpected. On a global scale, evolutionary processes are undirected and excel in the open-ended exploration of novel design possibilities. Thus, natural evolution differs profoundly from the gradualist and constantly converging character of technical evolutionary optimization with fitness functions. Technical goals are defined and static fitness functions and the evolutionary mechanisms of selection, heredity, reproduction and mutation are employed as stochastic optimization processes. Implementations of these mechanisms reconstitute microevolutionary changes that largely correspond to the biological variability observed at the population level. This resembles an optimization process analogous to the dominant genotype of a population that is optimally adapted to the respective ecological niche. However, as we will see later, true biological innovation and diversity are observed at the species level or above and is the product of additional macroevolutionary mechanisms usually spanning larger time intervals. Mostly still relying on the concepts of the Modern Synthesis, the metaheuristic EAs do not include recent insights into micro- and macroevolutionary mechanisms derived from genomics, phylogenomics and population genomics.

Although a certain renaissance of EAs occurred when coupled with Deep Learning (Tirumala 2014; David and Greental 2014), many computational scientists and machine learners consider EAs and GAs as no longer relevant. Many perceive them as too slow, too inefficient and prone to being stuck at local maxima, suggesting that the approach of transferring evolutionary principles to optimization and design problems has failed. This raises the obvious questions as to whether this view is correct or whether just the chosen abstractions of evolutionary processes are too simple to mimic evolutionary processes effectively. Furthermore, we need to ask whether and in what way biology can benefit from a realignment of the two fields and whether we can use EAs in a reverse biomimetic approach to gain novel biological insights.

Similar to biological evolution, the architectural design process is an open-ended process exploring possible solutions that satisfy the design requirements, which quite often include aspects that are extremely difficult to quantify. In order to navigate and search in the vast, dynamic, highly varying and multi-dimensional solution space, most design methodologies in architecture are based on a typological approach. This limits the solution space to a particular structural, constructional, spatial or programmatic type, which is iteratively adapted to the particular design requirements. The constraints inherent in typology-based design methodologies exclude a vast range of potentially more effective and better design variants. In

contrast, the dynamics of biological evolution suggest ways of continuously expanding the solution space towards new and unexplored regions, as in macroevolution. Thus, in architecture, the exploratory nature of macroevolutionary processes is more relevant than the microevolutionary processes implemented in current EA optimization tools.

This chapter reviews the current state of EAs with relevance to computational design, aligns them with the current state of biological research, gives an overview of the computational approach to architectural design and highlights some of the relevant biological insights bridging the gap between micro- and macroevolution. Not all of these questions can be addressed at the global scale; any algorithmic instantiation of these principles aiming at avoiding being too simplistic or too divergent from the biological concept generator needs detailed and quantitative data. Finally, the chapter offers a perspective with regard to the way that these principles can inform the computational approach to biomimetic architectural design. The achievement of this goal requires interdisciplinary collaboration between the fields of biology, architecture, engineering and computer science.

15.2 Evolutionary Computation

15.2.1 Overview

Evolutionary computation (EC), or *evolutionary computing* is a subfield of *artificial intelligence* that comprises a group of algorithms known as *evolutionary algorithms* (EA) (Bentley and Corne 2001; Bentley et al. 2008). These algorithms are based on principles that are inspired by Darwin's theory of evolution and that act on a "population" of potential solutions to the problem at hand. The three most fundamental principles are:

1. *Recombination*: Individuals from the current *population* are combined to produce new individuals. The main idea is to allow "good" features from two (or sometime more) individuals to be (re)combined and carried on by the resulting offspring. Although the algorithmic details vary between the different versions, implementations or applications of the algorithms, in most cases, the recombination method involves some means of randomization.
2. *Mutation*: Each individual in the *population* has a certain probability of be *mutated* (i.e. randomly modified). This allows the production individuals that are possibly divergent from those in the current population and introduces a further source of variability.
3. *Selection*: Only the "best" individuals are retained in the *population*; the rest will be discarded. This ensures that, over time, the fitness of the individuals will increase. The selection procedure can be implemented explicitly by the computation of a fitness value for each individual and the selection of those with the highest values or by implicitly letting the individuals compete with each other.

Because the principles above are applied to the *population* of solutions over many iterations, they result in the gradual maximization of the fitness of the solutions. The converging character of this class of algorithms obviates their applicability to optimization problems.

EAs are generally grouped into four categories (Bentley 1999): genetic algorithms, genetic programming, evolutionary strategies and evolutionary programming. Among these, only genetic algorithms and genetic programming clearly distinguish between genotype and phenotype representations. As this distinction is important for the translation of macroevolutionary principles, the subsequent sections will focus solely on these two categories.

15.2.2 Genetic Algorithms

Genetic algorithms (GA) were introduced in the 1970s by John. H. Holland (Holland 1975) as an attempt to demonstrate and explain the adaptive processes found in natural systems and as an approach to produce artificial systems that exhibit adaptive characteristics similar to their natural counterparts. However, the same algorithm can be used for optimization (Bentley 1999). The object to be optimized is termed the *phenotype*, which is controlled by an underlying presentation known as the *genotype*. Both the terms *phenotype* and *genotype* are adopted from genetics. In architectural design, the *phenotype* is usually some type of three-dimensional (3D) shape. For example, a *phenotype* that is a 3D box shape can be controlled by an underlying *genotype* of three numerical parameters for width, length and height.

A typical GA (Fig. 15.1) comprises the iterative execution of many generations (cycles), which usually consists of seven stages:

Initialization (executed once): A *population* of solutions together with their corresponding *genotypic* representations is generated. The ideal *population* size (the number of individuals in the population) varies between applications and is constrained by the computational resources. The most straightforward way to generate these initial *ancestral* solutions is by assigning random parameter values to the genotype. Alternatively, they can be pre-generated by other optimization algorithms in order to ensure a minimum level of quality. Careful pre-generation of *ancestral populations* can help to improve the speed of finding an optimal solution (Rahnamayan et al. 2007). A comprehensive review of population initialization techniques can be found in (Kazimipour et al. 2014).

Recombination: *Genotypes* for new solutions are generated by combining *genotypes* from existing solutions. Prior to the actual recombination is a preparation step called *mating* whereby the algorithm decides which of the existing *genotypes* will be paired with each other to produce the *offspring genotypes*. The default and simplest approach is random *mating*, i.e. random selection of *mating* partners. Restricted mating procedures represent algorithmic means to implement non-random or assortative *mating* patterns found in nature, i.e. consider *genotypic* or

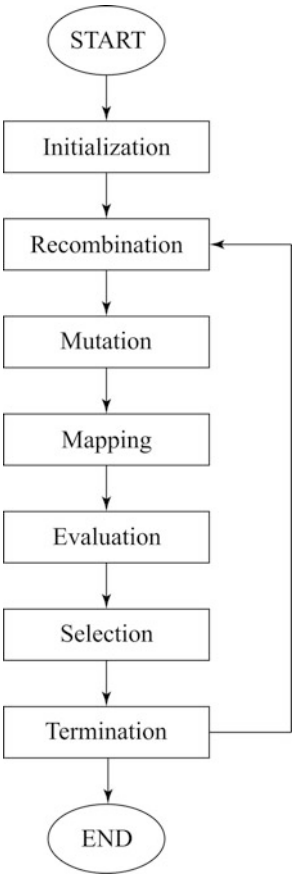

Fig. 15.1 Typical data flow of genetic algorithm

phenotypic compatibility or the similarity of the *mating* partners. For example, the two solutions in the same *mating* pair might be required to have a sufficiently similar *genotype* to avoid *offspring genotypes* that map to *phenotypes* with low *fitness* values (Deb and Goldberg 1989). Then again, the *mated genotypes* should not be too similar in order to prevent pre-mature convergence on a local maximum that might overlook a potentially better solution in another area of the search space; this is tackled by a procedure called incest prevention (Eshelman and Schaffer 1991). One popular version of *mating* allows the *fittest* solution (i.e. the one with the highest quality) to *mate* with every single other solution to allow its best features to combine with those from other solutions (Bentley 1999). Galán et al. (2013) generalized the traditional uniform *mating* method by attaching two extra parameters called the *mating* size and the *mating* index to each individual. The values of these parameters will decide whether an individual will *mate* with the most or least similar candidates in the *mating* pool or any in between. This allows the *mating* behaviour of each individual to vary between explorative (i.e. favouring variety) and exploitative (i.e. favouring convergence).

Furthermore, the values of the *mating* size and *mating* index do not have to be manually set by the user but are allowed to evolve together with other parameters in the *genotype*, resulting in a self-adaptive *mating* behaviour. The *genotype* of the *child* solution can be obtained in various ways, such as by using *crossover*, analogous to the way that homologous chromosomes recombine and exchange genetic material in meiosis during sexual reproduction. Whereas the biological recombination process involves a complex multi-step process most likely involving the tight regulation of the DNA damage machinery, artificial *crossover* in GA often represents a rather simple abstraction. In the original version described by Holland (1975), the parameters in a *genotype* are encoded as binary strings (i.e. genes). These gene strings are concatenated to form the *chromosome* binary string of that particular *genotype*. The *crossover* method will swap a region of the *father chromosome* string with the corresponding region of the *mother chromosome* string, resulting in two *children chromosome* strings. Alternatively to using binary representation, the real-value parameters can be combined into a *chromosome* vector and the *crossover* will act on two *parental chromosome* vectors (Wright 1991). In practice, the *crossover* operation can be applied many times to the same pair of *parental chromosomes* (but each time with a different *crossover* point) resulting in multiple distinct child solutions. Apart from the standard *crossover* procedure, real-value *chromosome* vectors can also be recombined by taking the weighted average of the parental parameter values by using Gaussian distribution as the basis for the weights (Ladkany and Trabia 2012). Apart from the oversimplification of the process of recombination, an obvious deviation from biological reality is present in the size and content of the *genome*. Eukaryotic genomes usually comprise multiple, independently inherited chromosomes that represent long-chained DNA polymers (i.e. vectors).

Mutation: Probabilistically, each individual solution in the current *population* can have their *genotype mutated* according to a certain probability value, often called the *mutation rate*. When *mutation* occurs, the *genotype* real-value parameters will be modified by adding a random offset to the current values based on some probability distribution function (Wright 1991) or, in the case of binary reorientation, a randomly chosen bit in the *chromosome* string will be altered (i.e. 0 becomes 1 or vice versa). *Mutation* allows unexplored regions of the search space in which potentially *fit* solutions can be found to be reached. Mühlenbein and Schlierkamp-Voosen (1993) have reported a *mutation rate* set inversely proportional to the number of *genes* (parameters) in the *genotypes* tends to give good results. GAs are usually tested in the range between 1E-03 and 0.5 (e.g. Kühn et al. 2013). *In vivo* mutation rates seem to be much lower and to depend on the outcome of the mutation. The spontaneous occurrence of indels (insertion or deletion of DNA stretches) has been recorded at a rate of 5E-12 per base per generation in yeasts (Zhu et al. 2014). Whereas selectively neutral changes, such as synonymous substitutions that affect the second and third codon position and that do not result in a change of the encoded amino acid, have been observed to occur at a rate of 1.9E-08 per site and year in mosses (Rensing et al. 2007).

Mapping: The *offspring genotypes* are mapped to their respective *phenotype* representation. For example, the simple 3D box *genotype* will map the parameters controlling the width, length and height to the actual geometry of the box. Not every region of the biological chromosomes directly encodes an attribute of the phenotype (i.e. gene) that is under selection. This phenomenon has resulted in the debate about junk DNA (Palazzo and Gregory 2014) that is tightly connected to the biological paradox that genome size is not correlated with organismic or morphological complexity (Gregory 2005; Greilhuber et al. 2005). Nevertheless, biological genomes are far more complex in gene content than their usual GA counterparts: unicellular bacteria usually harbour 1000–5000 genes, whereas multicellular eukaryotic genomes encode between 5000 and 10,0000 genes. Humans have about 20,000 genes and plants usually have around 30,000 genes (Sterck et al. 2007). Viruses can have similarly low gene numbers as those in common GA genomes but these strongly rely on their hosts for almost every aspect of their existence and are usually not considered as "alive" and coevolving with their respective host genomes. Only a few genes directly affect the morphology or behaviour of an organism. Furthermore, genes can be involved in multiple functions or affect multiple phenotypic traits (pleiotropy). In most cases, the products of multiple genes need to interact or act in sequence to implement a phenotype (polygenic inheritance). Thus, even if a gene has a direct phenotype, its activity might depend on the action of other genes (epistasis).

Evaluation: The *fitness* of each *phenotype* is evaluated via user-defined criteria. For example, the criterion can be the ratio of volume to surface area; this might serve as a *fitness* criterion, referring to the evaluation of a geometrical body. Alternatively, in case of a roof surface for a building, *fitness* might depend on the amount of solar energy that it receives, taking into account the occlusion caused by neighbouring buildings. The evaluation of the *phenotypical* consequences of a given *genotype in vivo* occurs on multiple levels and timescales. Not every change is evaluated or consequential directly after its occurrence. Organisms live in a changing environment that dynamically "evaluates" distinct aspects of the genotype.

Selection: A solution will be either retained or discarded based on its *fitness* value. Obviously, solutions with high *fitness* values need to be retained, whereas those with a lower *fitness* are discarded to ensure that the overall *fitness* of the solutions increases over time. The selection procedure can be deterministic, such that only a certain number of solutions with the highest *fitness* values are kept. This procedure is known as "truncation selection" (Mühlenbein and Schlierkamp-Voosen 1993). Stochastic methods provide an alternative to deterministic selection procedures. For example, in the "*fitness proportionate selection*" scheme, the probability that a solution will be retained is directly proportional to its *fitness* value. This means that solutions with low *fitness* scores still have a small non-zero chance of *survival* (rather than being certainly discarded as in the "*truncation selection*" scheme). In natural populations, not every trait is under strong purifying selection at all times. Thus, these methods introduce some aspects of balancing or relaxed selection and neutral evolution, which are seen

as important drivers of diversification and biological innovation. The relaxed *selective pressure* on these suboptimal solutions in the current *generation* might eventually lead to even better *adapted* or more generalist solutions in later *generations*. It also tends to be computationally more efficient as it does not require the sorting of all individuals based on their *fitness* values. Another well-known stochastic selection scheme is the *tournament* scheme (Miller and Goldberg 1995), in which individuals are grouped randomly and only the *fittest individual* of each group will be retained in the *population*. In addition, *parents* from the current *generation* can also proceed to the next generation if they have the highest *fitness* values. This strategy is known as "*elitism*" and can help preserve solutions with very high *fitness*.

As we have seen above, mutational rates vary across sites of a genome. This is linked to distinctive and varying selective pressures associated with structural and functional constraints. Consequently, we can observe that the mutational rates of biological genomes and genes vary at the level of time and structure. Molecular phylogenetics document a high degree of heterogeneity of evolutionary rates among distinct sites of the genome or even within the same gene (Jia et al. 2014). These rates constantly change as a function of time. Furthermore, we can distinguish different directionalities of selection: (1) negative selective forces (purifying selection) result in a level of high sequence conservation of functionally or structurally important regions based on the effective elimination of altered alleles; (2) positive or "Darwinian" selection in contrast favours certain alleles as they positively influence fitness and is measurable by increased rates of change and diversification; (3) as we have seen earlier in the discussion of silent synonymous mutations, not every change is under selection. The relaxation of selection towards neutral evolution is of particular importance for the so-called junk DNA, such as transposable elements or integrated viroids.

Termination: The algorithm is stopped if the termination criterion is met. Otherwise, it will return to the recombination stage and continue iteratively until the termination criterion is met. Typically, the algorithm is terminated when a sufficiently good solution has been produced or the overall running time or number of *generations* has reached a certain threshold defined by the user.

15.2.3 Genetic Programming

Genetic programming (GP) can be regarded as a special case of GAs, whereby the objects being evolved are computer programs. This was originally developed by Koza (1992) as a strategy to create computer programs that can program themselves. The *genotype* of a program is typically represented as a tree structure and the *phenotype* is the program or function itself. In this regard, the *gene* representations of GP more closely resemble their biological counterparts, as they encode evolvable functions that can be cascaded. An example of such a scheme is shown in Fig. 15.2. The tree structure serves as a basis on which *recombination*

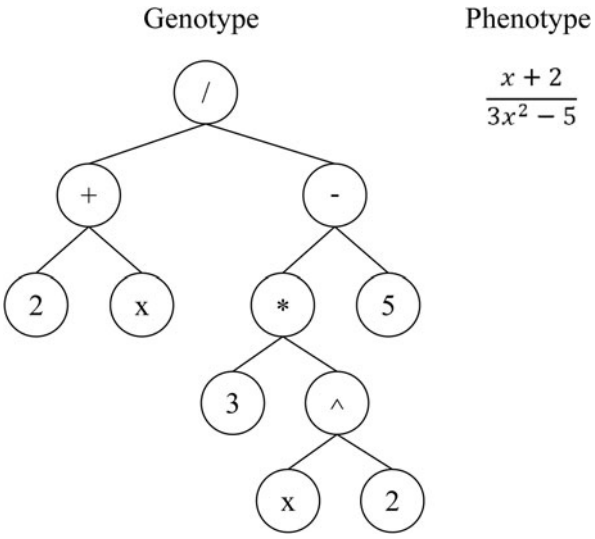

Fig. 15.2 Genotype and phenotype of a mathematical formula being evolved using genetic programming

and *mutation* mechanisms such as *crossover* can act. As an example, *crossover* is achieved via the exchange of branches between two parental trees, resulting in two child trees that represent novel *genotypes* that subsequently can be mapped to their respective *phenotype* (Fig. 15.3).

15.3 Evolution and Computational Design in Architectural Exploration

15.3.1 Computational Approach to Architectural Design

The introduction of computation and programming as a design tool in architecture implies that a profound shift in attitude and thinking with regard to the very nature of the architectural design process itself is necessary in order to fully embrace the potential that computation can offer. This approach to design is referred to as *computational design* and is well-discussed in Menges and Ahlquist (2011).

Computational design entails new design methods that profoundly extend the range of possible architectural outcomes. This is because, rather than directly modelling the design by using standard geometric constructions such as points, lines, curves or flat and curved surfaces, a designer who adopts the computational approach models the steps that lead to the generation of the design solutions and lets them be executed by the computers. Typically, the steps here are founded upon many algorithms descending from various fields within computer science, mathematics, biology and physics, such as computer graphics, computational geometry, artificial intelligence, numerical computation, analysis and simulation. These algorithms are used to process large amounts of design and feedback data, transforming them into

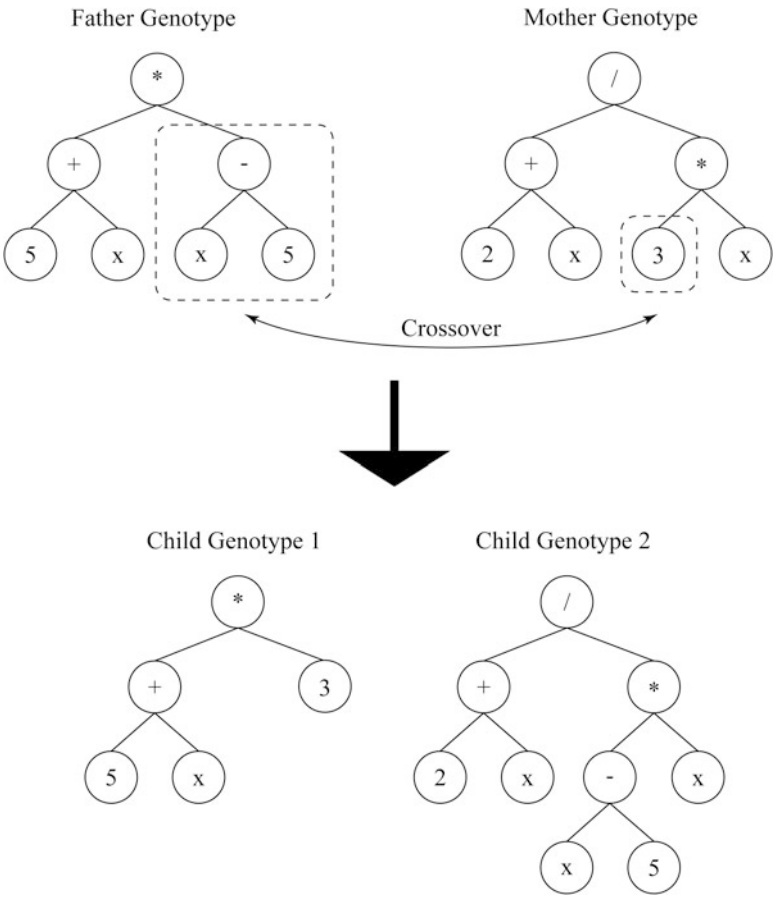

Fig. 15.3 Example of a crossover in the tree-structure representations of genotypes

design solutions that include both geometric and non-geometric information. Such features lead to the assumption that computational design will potentially lead to novel architectural outcomes that are simply not possible to achieve by using either manual techniques or their computerized version, which is often the found in current Computer-Aided Design workflows.

15.3.2 Evolutionary Computation in the Current Practice of Computational Design

The technology-affine part of the design community expressed interest in utilizing evolutionary computation very early (Frazer 1995). Among the broad range of approaches to evolutionary computation, GAs are the most widely used. On the

one hand, this might be attributable to the prevalence of the concepts of Darwin and the Modern Synthesis in our society and, thus, ultimately might be linked to the "biomimetic promise" in which we perceive the outcome of natural evolution as optimal, efficient, sustainable and elegant (see Chap. 18). On the other hand, GAs are generic and flexible enough to be applied to a wide range of design problems. Most importantly, the genotype-phenotype mapping innate to GAs makes them particularly attractive for designers.

Overall, GAs are particularly suitable for architectural design problems in which various performance aspects and design criteria can be quantified and used as *fitness* criteria in order to guide the evolution of design solutions. Examples include lighting (Caldas and Norford 2002), acoustics (Sato et al. 2004; Spaeth and Menges 2011), view exposure (Menges 2012) or structural aspects (Dimčić 2011; Coelho et al. 2014). One advantage of GAs is their conceptual ease of use in a design environment and the lack of requirement for gradient information upon which many other optimization techniques depend, e.g. gradient descent.

The general approach to using GAs in architectural and structural design involves four major steps:

1. Defining a parametric model that captures the design solution space. The number of parameters is carefully picked in advance and, usually, their value ranges are also determined.
2. Defining the *fitness* criteria for the design.
3. Defining the way to compute the *fitness* value for a design generated by the parametric model in step 1.
4. Using GAs to explore the parameter space and the design solution space, so that eventually design solutions with high *fitness* values will be generated without having exhaustively to search the entire space.

Evolutionary computation in general and GAs in particular have been recurring features in the context of computational design. In many cases, GAs have been used as optimization tools. One of the most important and earliest pioneers in evolutionary architectural design is John Frazer. His work with evolutionary computation is documented in Frazer (1995). Examples of design problems that have been tackled by using evolutionary computation include, but are not limited to, free-form curved surface design (Hemberg et al. 2001, 2008), spatial layout (Jo and Gero 1998; Michalek et al. 2002), building envelope design (Tuhus-Dubrow and Krarti 2010), thermal and lighting performance (Caldas and Norford 2002; Wright et al. 2002) and urban design (Finucane et al. 2006). Evolutionary computation has also been applied in the related field of structural engineering; for example, Dimčić (2011) and Coelho et al. (2014) have shown the way that GAs can be used to optimize the design of shell structures.

Menges (2012) presented two case studies in which design solutions were generated by using an evolutionary approach. In the first case study, the form of a building envelope was evolved by using a GA guided by multiple *fitness* criteria, including floor area, envelope height, envelope slope, unobstructed view exposure and environmental criteria such as incident solar radiation in interior thermal loading (Fig. 15.4).

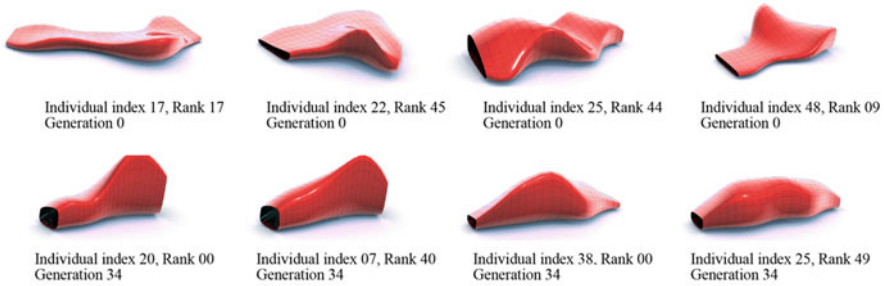

Fig. 15.4 Spatial envelope design of a structure on the Island of Boracay (Philippines) using GA, guided by multiple fitness criteria such as floor areas, height, slopes, incident solar radiation and interior thermal loading

The second case study dealt with the design of an urban block (Fig. 15.5). In this particular study, the designer attempted to overcome the typological design approach by devising a genotypic structure that actually consisted of five sets of genes, coupled with a so-called "*embryology* operator" (an advanced form of the genotype-phenotype mapping operator) that took each set of *genes* into account at various phases during the mapping (Fig. 15.6). In this regard, the *embryology* operator is a rough abstraction of a simple developmental program in biology. Similar to a developmental pathway in multicellular organisms, it establishes an expression hierarchy and implements a decision tree guiding the *developmental program*. In contrast to most eukaryotic developmental programs, the order of execution is encoded in the order of the *genes* on the *chromosomes*. An analogous counterpart of this can be seen in the operons found in unicellular bacteria in which genes encoding a specific pathway are co-localized on the bacterial genome and controlled by a single promoter. As we will see in the following section, developmental programs of multicellular eukaryotes are not coordinated via gene ordering and are much more complex.

15.3.3 Limitations of Current EAs for Computational Design

As we compare and contrast current approaches in evolutionary computation with advances in evolutionary biology, the great potential for enhancement seems to be worth investigating. In a biological context, evolutionary processes are undirected and open-ended and natural evolution has constantly produced novel designs that have not previously been seen. In this sense, the architectural design process itself is very similar to evolution. Thus, evolutionary principles seem to be more relevant to architectural exploration than design optimization with respect to a clearly defined set of goals. Indeed, as optimization tools, EAs are arguably falling out of favour as they tend to be quite inefficient compared with other superior techniques. This inefficiency is attributable to the rather "blind" searching mechanism of EAs,

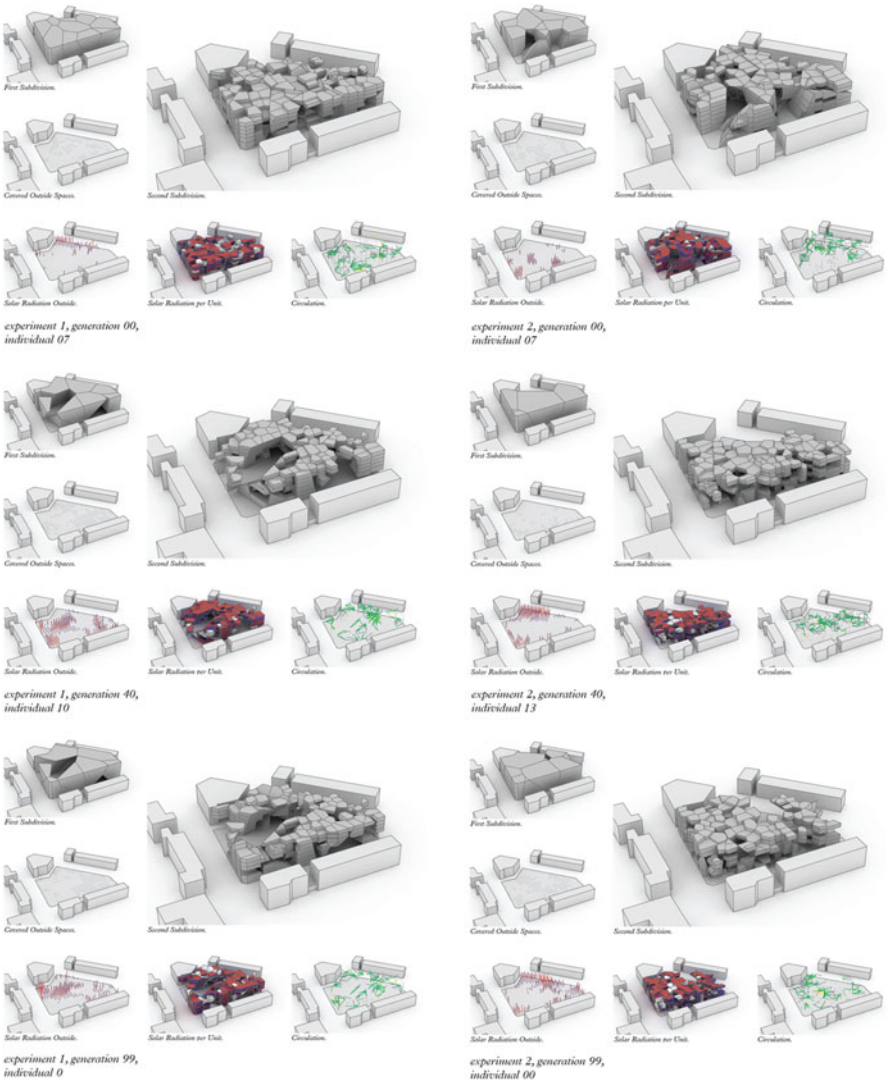

Fig. 15.5 Evolution of an urban block morphology driven by a GA, guided by five climatic fitness criteria (the amount of covered outside space, the cross-ventilation and solar radiation on both the overall morphology level and on the unit level) and two spatial fitness criteria (the number of units and the connectivity between them)

which generate hundreds or thousands of possible solutions in a manner of trial and error and do not take into account additional problem-specific information. However, a recent revival of interest has occurred in evolutionary computation, not for pure optimization purposes, but for more creative application domains, such as architecture, art and music (Bentley and Corne 2001).

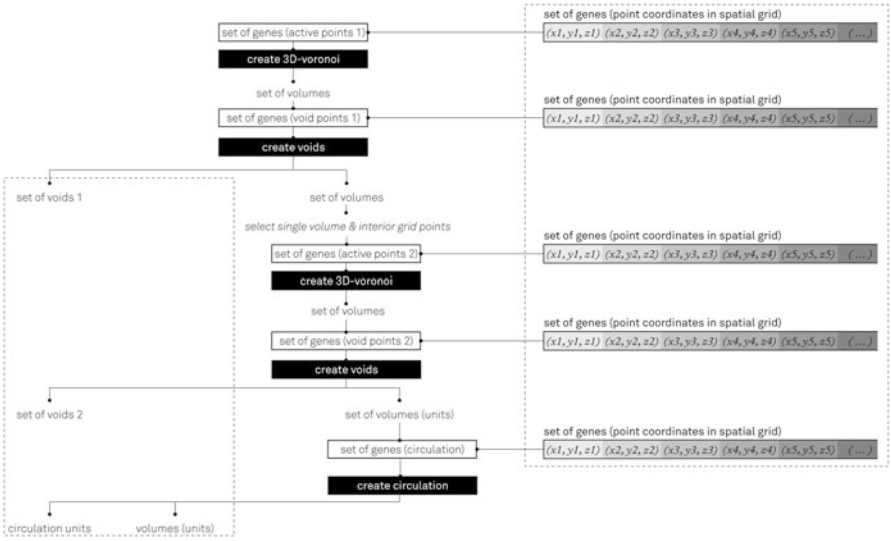

Fig. 15.6 The embryology operator that maps the genotype to phenotype via a multi-step hierarchical spatial subdivision scheme that drives the design shown in Fig. 15.5

To improve our understanding of an important limitation of the current approach in evolutionary computation, we should discuss the general workflow that involves four main steps as described in Sect. 18.3.2. The level of success in achieving good designs from the evolution process heavily depends on step 1. This is the point at which the designer must set up some sort of generative process that will transform a set of input parameters into a particular design solution. The designer must decide what aspects of the design can be changed and explored in order to maximize fitness criteria and then capture these aspects in these parameters. If the designer is overly ambitious and aims to capture as many aspects as possible, not only will he/she have to invest more time and effort in setting up the necessary parameters and the way that they drive the geometric relationships in the models, but this may also result in a very heavy parametric model that is slow to execute. This can seriously affect the optimization process as GAs work by generating hundreds or thousands of potential design solutions per generation, over many generations. Furthermore, the higher number of parameters results in a larger search space, i.e. an increased effort in order to find a good solution. This phenomenon is informally referred to as "the curse of dimensionality" and directly affects the algorithmic complexity of the problem (Bellman 1966). At the opposite extreme, if the designer only parametrizes a few aspects of the design, the search space will be very limited and the GA might not be able to find an acceptable solution at all.

The problem with the above approach is that, irrespective of the way that the designer sets up the parametric frameworks, the search space will be fixed and bounded. EAs will navigate the search space to find optimal designs but will

never extend the search space beyond its current status. In contrast, biological evolution does indeed extend the search space on its own, as long as it does not violate any fundamental physical or biological laws. The key to this is probably the fact that not only the biological entities (e.g. organisms) evolve, but the biological evolutionary processes themselves are also evolving! In contrast, the current approach to evolutionary computation mostly concerns the evolution of objects (the digital genotypes and phenotypes) but not the evolutionary processes and mechanisms.

This limitation can be illustrated by using the two case studies in Menges (2012). In the first case study (Fig. 15.4), a GA operated on *genotypes* with the same fixed structure, by using a linear non-evolving *genotype-phenotype* mapping process. Although it resulted in a nice variety of design solutions with good performances, all these solutions shared similar characteristic appearances. This is because they were all obtained from a pre-defined and fixed search space. No matter how many *generations* the algorithm was allowed to compute, it would never produce any design that was outside of the predefined typology described by the search space.

In the second case study (Fig. 15.5), the designer employed a multi-step *embryology* mapping operator in order to diversify the search space (Fig. 15.6). This helped the design process to start breaking away from the current limitation. However, as the *embryology* operator itself was not evolving, ultimately the search space could not expand beyond that allowed by the *embryology* operator.

15.4 Molecular Motors of Micro- and Macroevolution

15.4.1 Introduction

Although its principle action and importance for the generation of biological diversity is scientifically undisputed, our understanding of the evolutionary mechanisms driving species diversification and organismic complexity has evolved substantially since the initial conception of EAs and the divergence of the field of evolutionary computation. The availability of entire genomes from an increasing number of model organisms and the accessibility of population dynamics at the genomic scale enabled by next-generation sequencing (NGS) technologies continuously inform comparative and evolutionary biology (Lang et al. 2008).

So far, out review of the role of evolutionary computation in architectural design, illustrated its current limitations and provocatively insinuated that these limitations are attributable to a combination of the level of abstraction (i.e. oversimplification) and a certain detachment of the evolutionary computation field from biological research beyond the Modern Synthesis. Furthermore, if aligned with the biological concept generators, EAs mostly correspond to a crude simulation of microevolutionary processes in viruses or very simple unicellular bacteria. This is certainly not the goal that engineers, designers or architects have in mind when they employ EAs as biomimetic products to utilize the exploratory powers of evolution. We think instead

of the processes that have shaped the organismic and morphological complexity that is found in extant multicellular eukaryotes, such as that in plants ranging from unicellular photosynthetic eukaryotes (e.g. picoplankton *Ostreococcus*; Fig. 15.8c), multicellular algae (e.g. *Volvox*; Fig. 15.8b), mosses (e.g. *Physcomitrella* Fig. 15.8d) to, grasses (e.g. maize; Fig. 15.8e), shrubs (e.g. grape vine; Fig. 15.8f) and trees (e.g. poplar; Fig. 15.8g). Thus, as potential designers of evolution-inspired exploratory design tools, we are more interested in macro- than in microevolution.

From the perspective of the Modern Synthesis and in congruence with Darwin's model of the gradual acquisition and selection of changes, the distinction between micro- and macroevolution is not a fundamental one but is solely dependent on time and scale. Accordingly, the accumulation of small-scale random genetic changes over time (favouring changes that are advantageous for the survival of the organisms) eventually results in the rise of new species and higher-order taxa. Over decades, the Modern Synthesis has been constantly aligned to scientific progress by, for example, introducing single factors not necessarily qualifying as small changes, such as whole genome duplication events, and the various mechanisms of speciation, each in their own way resulting in reproductive isolation. However, the relative importance of these mechanisms in the light of species evolution largely remains unclear. In 1972, the gradualist model was challenged by palaeontologists Eldredge and Gould and their theory of punctuated equilibrium (Eldredge and Gould 1972). Derived from morphological observations in the fossil record, they proposed rapid bursts of evolutionary changes during relatively short timeframes as being the origin of new species punctuating longer periods of morphological stasis. Today, both models co-exist in the scientific community. Instead of two opposing views, they are rather seen as two distinct modes of evolution dynamically acting on multiple taxonomic levels and time scales.

As in biology, the variability-generating mechanisms in EAs ultimately need to be implemented and resolved down to the genotype level. Thus, any work on the improvement of EAs in this respect needs to target molecular processes. Nevertheless, what are the candidate molecular processes and mechanisms that affect macroevolution? What are the concepts determining morphological or organismic complexity? What are the molecular motors of diversification?

Comprehensively answering these questions would certainly require a whole series of books. Although evolutionary research has made tremendous progress since the dawn of the Modern Synthesis, it certainly still cannot provide the definitive answers. Indeed, because of the time scales and complexity of macroevolutionary processes, definite experimental proof for the causality of certain phenomena is extremely difficult to come by within a human lifetime. Darwin himself (1859) was very aware of this limitation: "We see nothing of these slow changes in progress, until the hand of time has marked the long lapses of ages, and then so imperfect is our view into long past geological ages, that we only see that the forms of life are now different from what they formerly were." This is the point at which biological research can benefit from a realignment with evolutionary computation and can use it in a reverse biomimetic approach in order to assess the macroevolutionary potential of candidate mechanisms and concepts.

For the remainder of this chapter, we will use plants as concept generators to highlight some of the candidate mechanisms and concepts driving macroevolution. Genomes of multicellular plants are complex being huge with so-called junk DNA and inherited on multiple chromosomes with usually around 30,000–40,000 genes (Sterck et al. 2007). In most instances, no simple 1:1 mapping occurs between genotype and phenotype. Biological processes are more like acyclic graphs than decision trees, and polygenetic inheritance, epistasis, multiple alleles and pleiotropy are common phenomena. Redundancy is the rule, not the exception. Genes are expressed and the central dogma "from gene to mRNA to protein" is being continuously revised and expanded (every step is regulated, i.e. dynamic). The dosage of gene products shapes the phenotype either via gene regulation (expression level, alternative splicing and degradation) and/or gene copy number (gene duplication) (Rensing 2014). From an engineering standpoint, all this seems counter-intuitive and it is difficult to believe that anyone would buy a machine or a building designed in this way. However, biological organisms are not intelligently designed; they are the result of a multi-scale, dynamic, stochastic and chaotic process. This is most evident in their genomes and encoded molecular processes.

Key concepts for our understanding of the way in which this apparent chaos is orchestrated and has evolved to form trees or humans from unicellular organisms are illustrated in Figs. 15.7 and 15.8. Gene products (proteins, rRNAs, tRNA, miRNAs and other non-protein-coding RNAs) interact in a spatio-temporal fashion to form a complex regulatory and enzymatic network. Multi-layered interaction networks and gene regulation are fundamental to an understanding of morphological evolution at the molecular level. Figure 15.7a depicts a randomly chosen subset of a protein-protein interaction network of a flowering plant. This interaction can be physical in the form of a multi-protein complex such as the dynamic protein complexes that control and initiate transcription (i.e. gene expression; Figs. 15.7c) or it can be indirect and causal (in trans) as in the regulatory network between a transcription factor and its target genes and their gene products (Fig. 15.7b).

Obvious differences can be found in the morphological complexity in the different plants listed earlier (Figs. 15.8b–g). One determinant feature of this complexity is the number of distinct cell types or tissues that these plants express in their life cycle.

In particular, gene regulatory networks and, consequently, the complements of gene regulators (GR: transcription factors [TF], transcriptional regulators [TR] and miRNAs) seem to be tightly linked to the evolution of organismic complexity (Levine and Tjian 2003; De Mendoza 2013). If we consider the phylogenetically inferred gains, losses and expansions of GR gene families along the lineages leading to plants (Fig. 15.8a), a significant correlation can be seen in the complements of GRs with complexity (Lang et al. 2010). The expansion of GR complements, and thus complexity, was achieved via multiple rounds of whole genome or large-scale duplication events (e.g. Lang et al. 2010; De Smet and Van de Peer 2012).

By sequencing the biological diversity ranging from the taxon to the population level, we can construct phylogenomics and population genomics frameworks to

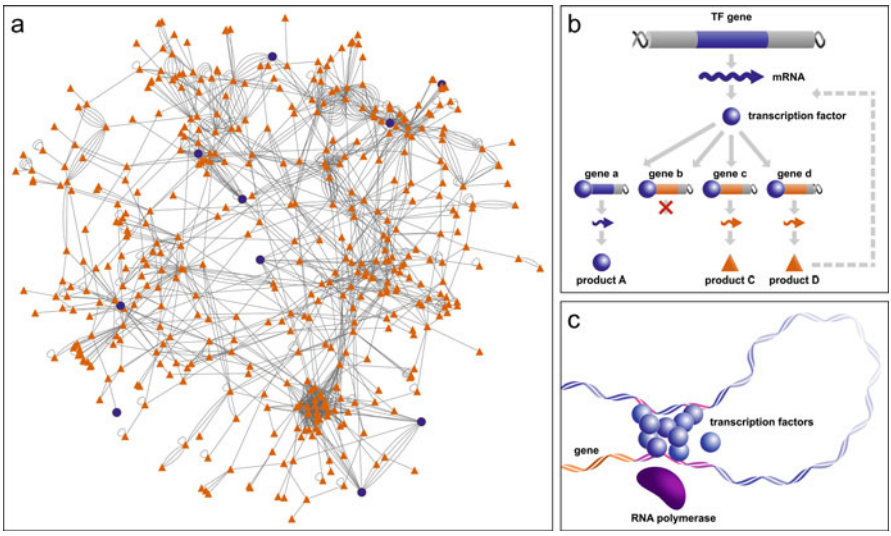

Fig. 15.7 Gene regulatory complexity by means of transcriptional regulation. (**a**) Depicts a complex network formed by a randomly chosen subset of interacting *Arabidopsis thaliana* proteins (Brandão et al. 2009; comprising 6905 nodes and 19938 edges in total). The nodes (451 of 6905) in this network represent gene products whose physical interactions, depicted as edges (1593 of 19938), were experimentally demonstrated. Transcription factors (TF) are highlighted as *blue circles*. Multiple edges between the same products represent the formation of complexes with more than one instance of each product and looped edges on single products represent multimers. A simple gene regulatory network is shown in (**b**). A TF itself is the product of its corresponding gene and physically binds to the promoter region of specific target genes. In that way the TF causally yet indirectly influences the expression of gene products, by either promoting (gene a, c, d) or suppressing (gene b) target gene expression. Such a target gene in turn might either encode for another TF (product A) or for any other gene product that either functions independently (product C) or forms a feedback loop (product D) that controls the expression of the original transcription factor, e.g. a microRNA. (**c**) illustrates a case of target gene regulation via complex formation of several transcription factors. The binding of transcription factors at the promoter region (*pink*) of the gene (*orange*) leads to a conformational change of the DNA structure and enables the binding of additional factors. Only by forming this complex, the RNA polymerase is recruited to the transcription start site and transcription is initiated

identify evolutionary processes (e.g. Wright and Andolfatto 2008). Combined analyses with further experimental data allows the bridging of the gap from genotype to phenotype and helps to explain the evolution of complex traits and organismic diversity (e.g. Freeling and Thomas 2006; Rensing et al. 2008; Lang et al. 2010; De Smet and Van de Peer 2012).

The use of the moss *Physcomitrella patens* and related species as evolutionary models and representatives of an early diverging lineage of land plants has provided important insights into land plant evolution including molecular adaptations related to the conquest of land (Rensing et al. 2008) and the evolution of organismic complexity (Lang et al. 2010).

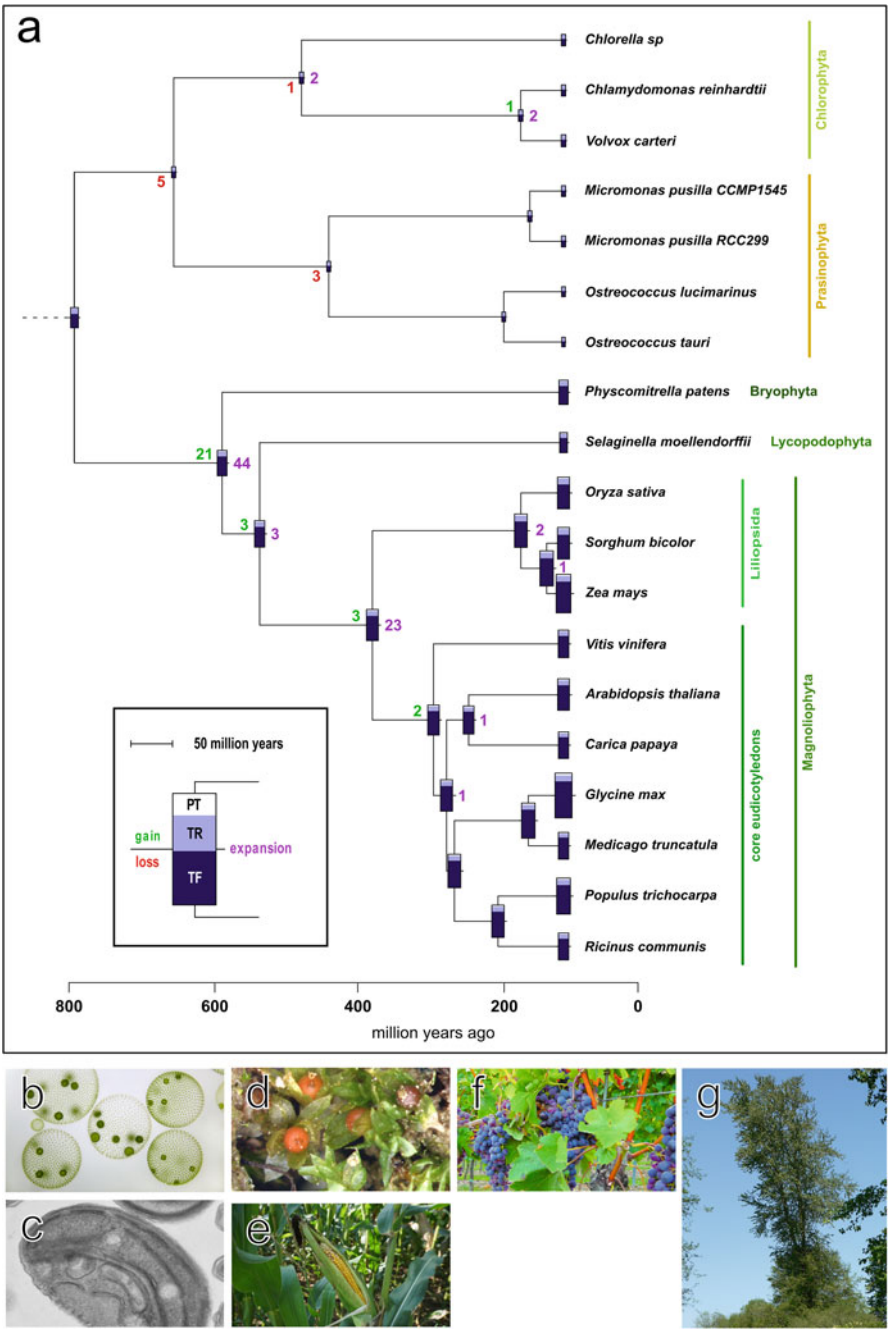

Fig. 15.8 Organismal complexity via expansion of gene regulator families in the evolution of plants. (**a**) shows the extant and reconstructed ancestral numbers of gene regulators (GR), namely transcription factors (TF, *dark blue*), transcriptional regulators (TR, *light blue*) and putative transcription-associated proteins (PT, *white*) along the phylogenetic tree of plant model species

The evolution and radiation of *P. patens* and its close relatives is thought to be shaped by whole-genome duplication events, i.e. polyploidization, and subsequent aneuploidization (Rensing et al. 2007; Beike et al. 2014).

The reconstruction of the polyploidization history of *P. patens* and related taxa enable tracing micro- and macro-evolutionary mechanisms behind speciation and morphological diversification. Furthermore, genomic analysis of these mosses provides an ideal opportunity to study the gene-phene evolution in the scope to serve as a concept generator that can inform the development of exploratory design tools.

15.5 Towards an Exploratory Design Framework Based on Evolutionary Principles

Concluding this chapter, we propose a re-evaluation of the abstractions and heuristics implemented in current EA approaches in order to achieve the transition from micro- to macro-evolutionary processes. We need to increase the complexity of our digital *organisms* and their *environment*. To reach this goal, we suggest the establishment of a flexible algorithmic framework that can be dynamically coupled to the results and hypotheses from current biological research encompassing a greater extent of the complexity of multicellular life. Because of its very nature, evolutionary biology does not usually deliver absolute ground truths, but instead candidate mechanisms and processes. Thus, the proposed framework needs to be dynamic and modular, allowing for experimental execution at low developmental cost to allow the direct integration of quantitative biological data such as evolutionary rates from multiple species or distinct scenarios.

From the perspective of computational design, this framework should enable the dynamic alteration of the *genotype-phenotype* mapping that accommodates both the evolution of 3D structures and more operational outputs, such as the behaviour of agent systems. Given this capacity, a truly explorative design tool that expands current design methods should be achievable. However, we also envisage obvious computational challenges in this approach. If traditional EAs with limited organismic complexity are currently computationally expensive, the challenge will be to find suitable abstractions and heuristics that increase organismic and operational complexity but nevertheless keep algorithmic costs in check.

Fig. 15.8 (continued) (Modified from Lang et al. 2010). At each node in the tree, i.e. for each common ancestor and extant species, the numbers represent the gain (*green*), loss (*red*) and expansion (*pink*) of GRs. Respective boxes depict the individual composition of GRs while their size corresponds to their absolute numbers. (**b**)–(**g**) illustrate the increase in organismic complexity coinciding with the expansion of GRs via a selection of extant species from (**a**), ranging from the multicellular and unicellular green algae *Volvox carteri* (**b**) and *Ostreococcus tauri* (**c**), respectively, over the moss *Physcomitrella patens* (**d**), the grass *Zea mays* (**e**) and the shrub *Vitis vinifera* (**f**) to the tree *Populus trichocarpa* (**g**)

Acknowledgements This work has been funded by the German Research Foundation (DFG) as part of the Transregional Collaborative Research Centre (SFB/Transregio) 141 'Biological Design and Integrative Structures'/project B02.

References

Barricelli NA (1962) Numerical testing of evolution theories. Acta Biotheor 16:69–98. doi:10.1007/BF01556771

Beike AK, von Stackelberg M, Schallenberg-Rüdinger M et al (2014) Molecular evidence for convergent evolution and allopolyploid speciation within the Physcomitrium-Physcomitrella species complex. BMC Evol Biol 14:158. doi:10.1186/1471-2148-14-158

Bellman R (1966) Dynamic programming. Science (80-) 153:34–37. doi: 10.1126/science.153.3731.34

Bentley PJ (1999) Evolutionary design by computers. Morgan Kaufmann Publishers, San Francisco

Bentley PJ, Corne DW (eds) (2001) Introduction to creative evolutionary systems. Morgan Kaufmann Publishers, San Francisco

Bentley DR, Balasubramanian S, Swerdlow HP et al (2008) Accurate whole human genome sequencing using reversible terminator chemistry. Nature 456:53–59. doi:10.1038/nature07517

Brandão MM, Dantas LL, Silva-Filho MC (2009) AtPIN: *Arabidopsis thaliana* protein interaction network. BMC Bioinf 10:454. doi:10.1186/1471-2105-10-454

Caldas LG, Norford LK (2002) A design optimization tool based on a genetic algorithm. In: Automation in construction, pp 173–184

Coelho RF, Echenagucia TM, Pugnale A, Richardson JN (2014) Genetic algorithms for structural design. In: Adriaenssens S, Block P, Veenendaal D, Williams C (eds) Shell structures for architecture: form finding and optimization. Routledge, London, pp 290–294

Darwin C (1859) On the origin of species. John Murray, London

David OE, Greental I (2014) Genetic algorithms for evolving deep neural networks. In: Proceedings of the 2014 conference companion on genetic and evolutionary computation. ACM Press, New York, pp 1451–1452

De Mendoza A (2013) Transcription factor evolution in eukaryotes and the assembly of the regulatory toolkit in multicellular lineages. PNAS 110(50):E4858–E4866

De Smet R, Van de Peer Y (2012) Redundancy and rewiring of genetic networks following genome-wide duplication events. Curr Opin Plant Biol 15:168–176

Deb K, Goldberg D (1989) An investigation of niche and species formation in genetic function optimization. In: Proceedings of the 3rd international conference on genetic algorithms, pp 42–50

Dimčić M (2011) Structural optimisation of grid shells based on genetic algorithms. University of Stuttgart, Stuttgart

Eldredge N, Gould SJ (1972) Punctuated equilibria: an alternative to phyletic gradualism. In: Schopf TJM (ed) Models in paleobiology. Freeman Cooper, San Francisco, pp 82–115

Eshelman L, Schaffer J (1991) Preventing premature convergence in genetic algorithms by preventing incest. In: Proceedings of the 3rd international conference on genetic algorithms, pp 115–122

Finucane E, Derix C, Coates P (2006) Evolving urban structures using computer optimisation techniques. In: Generative Art 2006 GA2006, IX Generative Art conference, 13–15 December 2006

Fraser AS (1958) Monte Carlo analyses of genetic models. Nature 181:208–209

Frazer J (1995) An evolutionary architecture. Architectural Association, London

Freeling M, Thomas BC (2006) Gene-balanced duplications, like tetraploidy, provide predictable drive to increase morphological complexity. Genome Res 16:805–814. doi:10.1101/gr.3681406

Galán S, Mengshoel O, Pinter R (2013) A novel mating approach for genetic algorithms. Evol Comput 21:197–229

Gregory TR (2005) The C-value enigma in plants and animals: a review of parallels and an appeal for partnership. Ann Bot 95:133–146. doi:10.1093/aob/mci009

Greilhuber J, Dolezel J, Lysák MA, Bennett MD (2005) The origin, evolution and proposed stabilization of the terms "genome size" and "C-value" to describe nuclear DNA contents. Ann Bot 95:255–260. doi:10.1093/aob/mci019

Hemberg M, O'Reilly U-M, Nordin P (2001) GENR8 – a design tool for surface generation. In: Goodman ED (ed) Genetic and evolutionary computation conference late breaking papers, pp 413–416

Hemberg M, O'Reilly U-M, Menges A et al (2008) Genr8: architects' experience with an emergent design tool. In: The art of artificial evolution. Springer, Berlin, pp 167–188

Holland JH (1975) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. University of Michigan Press, Ann Arbor

Jia F, Lo N, Ho SYW (2014) The impact of modelling rate heterogeneity among sites on phylogenetic estimates of intraspecific evolutionary rates and timescales. PLoS One 9, e95722. doi:10.1371/journal.pone.0095722

Jo J, Gero J (1998) Space layout planning using an evolutionary approach. Artif Intell Eng 12:149–162

Kazimipour B, Li X, Qin A (2014) A review of population initialization techniques for evolutionary algorithms. In: Proceedings of the IEEE congress on evolutionary computation, pp 2585–2592

Koza JR (1992) Genetic programming as a means for programming computers by natural selection. Stat Comput 4:87–112

Kühn M, Severin T, Salzwedel H (2013) Variable mutation rate at genetic algorithms: introduction of chromosome fitness in connection with multi-chromosome representation. Int J Comput Appl 72:31–38

Ladkany G, Trabia M (2012) A genetic algorithm with weighted average normally-distributed arithmetic crossover and twinkling. Appl Math 3:1220–1235

Lang D, Zimmer AD, Rensing SA, Reski R (2008) Exploring plant biodiversity: the Physcomitrella genome and beyond. Trends Plant Sci 13:542–549. doi:10.1016/j.tplants.2008.07.002

Lang D, Weiche B, Timmerhaus G et al (2010) Genome-wide phylogenetic comparative analysis of plant transcriptional regulation: a timeline of loss, gain, expansion, and correlation with complexity. Genome Biol Evol 2:488–503. doi:10.1093/gbe/evq032

Levine M, Tjian R (2003) Transcription regulation and animal diversity. Nature 424:147–151

Menges A (2012) Biomimetic design processes in architecture: morphogenetic and evolutionary computational design. Bioinspir Biomim 7:015003. doi:10.1088/1748-3182/7/1/015003

Menges A, Ahlquist S (eds) (2011) Computational design thinking. Wiley, Chichester

Michalek J, Choudhary R, Papalambros P (2002) Architectural layout design optimization. Eng Optim 34:461–484

Miller B, Goldberg DE (1995) Genetic algorithms, tournament selection, and the effects of noise. Complex Syst 9(3):193–212

Mühlenbein H, Schlierkamp-Voosen D (1993) Predictive models for the breeder genetic algorithm I. Continuous parameter optimization. Evol Comput 1:25–49

Palazzo AF, Gregory TR (2014) The case for junk DNA. PLoS Genet 10, e1004351. doi:10.1371/journal.pgen.1004351

Rahnamayan S, Tizhoosh H, Salama M (2007) A novel population initialization method for accelerating evolutionary algorithms. Comput Math Appl 53:1605–1614. doi:10.1016/j.camwa.2006.07.013

Rechenberg I (1971) Evolutionsstrategie – Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. TU, Berlin

Rensing SA (2014) Gene duplication as a driver of plant morphogenetic evolution. Curr Opin Plant Biol 17:43–48. doi:10.1016/j.pbi.2013.11.002

Rensing SA, Ick J, Fawcett JA et al (2007) An ancient genome duplication contributed to the abundance of metabolic genes in the moss *Physcomitrella patens*. BMC Evol Biol 7:130. doi:10.1186/1471-2148-7-130

Rensing SA, Lang D, Zimmer AD et al (2008) The *Physcomitrella* genome reveals evolutionary insights into the conquest of land by plants. Science 319:64–69. doi:10.1126/science.1150646

Sato S, Hayashi T, Takizawa A et al (2004) Acoustic design of theatres applying genetic algorithms. J Temporal Des Archit Environ 4:41–51

Schwefel H-P (1974) Numerische Optimierung von Computer-Modellen. TU, Berlin

Simoes A, Costa E (2000) Using genetic algorithms with asexual transposition. In: Genetic and evolutionary computation conference

Spaeth AB, Menges A (2011) Performative design for spatial acoustics: concept for an evolutionary design algorithm based on acoustics as design driver. In: Respecting fragile places, pp 461–468

Sterck L, Rombauts S, Vandepoele K et al (2007) How many genes are there in plants (. . . why are they there)? Curr Opin Plant Biol 10:199–203. doi:10.1016/j.pbi.2007.01.004

Tirumala SS (2014) Implementation of evolutionary algorithms for deep architectures. In: Proceedings of the second international workshop on artificial intelligence and cognition, pp 164–171

Tuhus-Dubrow D, Krarti M (2010) Genetic-algorithm based approach to optimize building envelope design for residential buildings. Build Environ 45:1574–1581. doi:10.1016/j.buildenv.2010.01.005

Turing A (1950) Computing machinery and intelligence. Mind LIX:433–460

Wright A (1991) Genetic algorithms for real parameter optimization. In: Foundations of genetic algorithms, pp 205–220

Wright S, Andolfatto P (2008) The impact of natural selection on the genome: emerging patterns in *Drosophila* and *Arabidopsis*. Annu Rev Ecol Evol Syst 39:199–213

Wright JA, Loosemore HA, Farmani R (2002) Optimization of building thermal design and control by multi-criterion genetic algorithm. Energy Build 34:959–972. doi:10.1016/S0378-7788(02)00071-3

Zhu YO, Siegal ML, Hall DW, Petrov DA (2014) Precise estimates of mutation rate and spectrum in yeast. Proc Natl Acad Sci U S A 111:E2310–E2318. doi:10.1073/pnas.1323011111