An Introduction to Evolution for Computer Scientists

Kester Clegg

Tuesday 22$^{\text{nd}}$ January, 2008

Abstract

This report is intended as an introduction to evolution and evolutionary computation. Taking its lead from contemporary thinking in biology, it advocates that we have more to learn from the role that developmental processes play in natural evolutionary search than has previously been admitted by practitioners in evolutionary computation. A brief summary of evolutionary biology is given, from Darwin to Dawkins and beyond, alongside theoretical insights from authors such as Kauffman and Solé. The report then sketches the development of evolutionary computation as a field of computer science. It gives some background to how the "central dogma of biology" gained a canonical form as an algorithm and why decomposing tasks for evolutionary computation remains a difficult problem. It ends by indicating how gene re-use and context-dependent gene expression may provide one key to how evolution creates scalable solutions of such complexity.

Contents

1 Preface

Evolutionary computation uses the process of natural selection as a search algorithm. Like evolution, the algorithm works over successive generations, gradually moving the search closer to the objectives until no more improvement in the population is possible, or the objectives are satisfied.

Evolutionary computation has a long academic and industrial record (see § 8.1). Within that time it has branched into variants (discussed in § 8.1.5), developed a canonical form (§ 8.2) and been deployed in a wide range of industrial applications (§ 9.1). During the mid 1990s, evolutionary computation began to draw media attention with claims that human-competitive patents were being discovered through the use of evolutionary search techniques (Fonlupt, 2005; Koza et al., 2003). The techniques found industrial application wherever a design required taking a set of competing objectives into account. Examples from this period include Honda's "evolved" gas turbine fan blades (Jin, 2005), while NASA carried out experiments to evolve antennae and other aerospace hardware (Miller, 2000). More recently, claims in popular science journals, such as Peter Bentley's article in New Scientist, have fuelled expectations that evolutionary computation is poised to take over human design and that creativity can now be automated (Bentley, 2004). The reality is more mundane. The achievements of evolutionary computation remain relatively modest. No one has evolved the design for a car, a house, or anything where the number of parts exposed to random mutation significantly adds to the complexity of the objective.

The problem is one of scale. Evolutionary computation has successfully been used as a search-based optimisation technique. Such optimisation generally involves a handful of factors in the fitness assessment. But as the number of elements sought by evolutionary search goes up, the time required to find a solution increases dramatically (see § 8.1.1). Where elements can affect one another, the search space size increase is exponential.[1] To get round this, the process is generally "bootstrapped" to the point where evolutionary algorithms optimise just a few variables on an existing solution, poor though that initial solution might be. But bootstrapping incurs a penalty — it constricts the search start point and therefore its trajectory. While we can trace the search trajectory of an evolved artifact, it is harder to estimate the place we should have started from to get a better result, particularly when the nature of the search space is unknown. Tackling these issues requires being well-versed in the art of problem representation and decomposition, an area that continues to cause difficulties (see § 9.2).

Claims that evolutionary computation is inspired by biological evolution must be tempered with the understanding that the representation and process bear only a token correspondence to those in nature. Although evolutionary computation employs a genetic encoding from which a population is generated and upon which selection is made, the step between the genetic encoding and selection of the phenotype is very small. In contrast, natural evolution has brought about a complex developmental process that plays a vital role in the exploration of the functional search space, and by extension, on phenotype selection. It is worth noting the historical legacy at work here, both from the study of evolution by biologists and the "borrowing" done by practitioners of evolutionary computation. The theory of evolution and natural selection, as developed by Charles Darwin (see § 3), was taken up and revised after the 1960s by those who believed that genes alone were the driving forces behind evolutionary change (§ 3.2). This

[1] Natural evolution deals with just such multi-dimensional search spaces, but of mind-boggling proportions, see § 6.3.

was the point at which evolutionary computation first settled on an established representation of genetic encoding and mutation (§ 8.1.4). But the genetic "reductionism" of the late 1970s was seen by some in biology as omitting the bigger picture (§ 3.3). They argued that evolution did not have a free hand to exploit genetic mutations. Instead, developmental processes constrain the degree and type of changes permitted in order to maintain the viability of the organism (discussed in § 3.3 and § 5.1). These restrictions have important implications on how natural evolution handles the combinatorial explosion of scale that has so far defeated proponents of evolutionary computation. However, despite the emerging evidence from biology on gene re-use and expression, and criticisms of the field's own shortcomings over the last decade (Banzhaf and et al, 2006), evolutionary computation has failed to move away from evolutionary models founded in the 1960s and has largely ignored what phenotype development could bring to the process of evolutionary discovery.

During the 1990s, theoretical biologists began to investigate complexity and evolution by building models of how evolutionary paths were traversed in competitive ecologies (§ 6). The models indicated that rather than striving for continuous perfection, organisms are involved in an evolutionary arms race against other organisms. Those who stand still while the world evolves around them fail in the race for survival. Likewise those who stray too high up an evolutionary peak of specialist adaptation find their evolutionary paths are cut off when they need to adapt to new circumstances (§ 6.3). These theoretical models suggest that natural evolutionary systems are poised close to the edge of chaos, as small changes in one part frequently have knock on effects throughout the system (§ 6.3; page 31). This picture of evolution is closer to one where the system is held in balance and species expand into available niches as they appear, rather than one geared to isolated optimisations over an open landscape. Such theoretical evidence may give clues to how evolution works within highly connected systems and perhaps help evolutionary computation cope with the complexity generated when evolution can act on all parts of a system.

Although it is difficult to do justice to the breadth of developmental and evolutionary biology, the following sections attempt to show where the arguments in favour of a developmental approach to evolutionary computation come from, and in doing so try to give a flavour of the inspiration that natural evolution provides.

2 The Inspiration of Nature

Nature furnishes us with examples of organisms that have adapted to a bewildering variety of environments. Forms of life extend almost as far below the earth as they do above it. From bacteria thriving in complete darkness in hostile, boiling sulphur springs miles beneath the surface of the Pacific Ocean, to insects blown aloft in the oxygen-deprived, freezing temperatures of our upper atmosphere, there are few places that life has not managed to adapt to and survive. This general purpose, problem-solving ingenuity can be found almost anywhere life exists, but where there are abundant sources of food and water, such as equatorial rain forests, there is more opportunity for specialist adaptation. Camouflage is a vivid manifestation of adaptation to a particular habitat, and insects in particular draw inspiration for camouflage from almost anything in their environment. Examples can be found of beetles on the forest floor that have evolved to mimic fallen seeds pods (Fig. 1), while others mimic twigs or diseased leaves (Fig. 2). Caterpillars may mimic other, more poisonous caterpillars, or even bird excrement (Fig. 1). Birds also make use of their plumage and stance to render

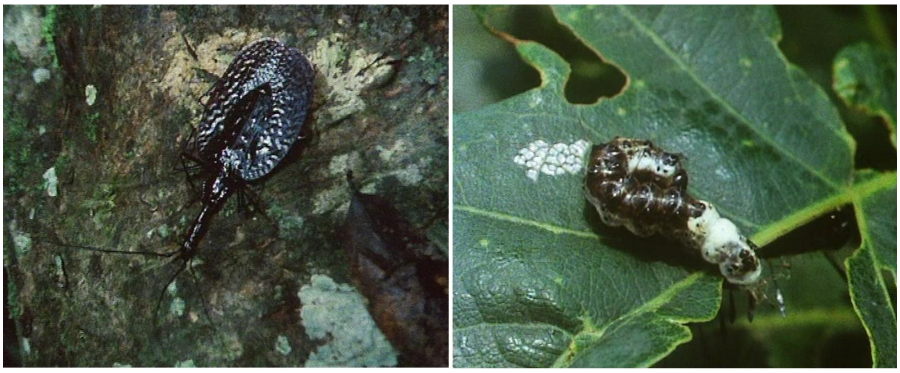


Figure 1: Insect camouflage: one mimics a shiny seed pod found on the forest floor (left), another mimics bird excrement while it feeds.

Images copyright of BBC Worldwide Ltd. (Attenborough, 1984)

themselves invisible (Fig. 3).

The variety of evolved forms is one factor that allows life to solve the problem of existence within such a wide range of habitat (see Fig. 4). But variety alone would be insufficient to allow phyla to survive indefinitely. If environments change, whether by movement of the organism or from external factors, organisms must change with them. If a species fails to change, it may quickly die out. No organism knows how it will need to adapt for the future and only reproduction permits a species to survive.

In Europe at least until the mid-nineteenth century, what had enabled a species to arise in the first place had traditionally been ascribed to the creative power of God. There was no recognition that things ever changed from the point of creation onward. But a famous debate by the British Association for the Advancement of Science in Oxford in 1860 (Howard, 2001), gave birth to a new vision of how life created and continued to create new forms. Theologians lost the right to impose a single understanding of how life had been created, and although that debate is still engaged in some quarters, the explanation of how life has evolved has been almost universally attributed to the theory of natural selection developed by Charles Darwin.

3 Darwin's Theory of Natural Selection

Jonathon Howard has commented in his book on Darwin that anyone attempting a biography of the man is "faced with an embarrassment of riches" (Howard, 2001). But while a very full account of Darwin's life would be possible given the notes and records that have come down to us, the following section attempts to summarise in brief terms who Darwin was, with respect to his time and place, and what enabled him to formulate the most famous theory in biology, in his book "On The Origin of Species" (Darwin, 1859).

Grandson of the doctor and "speculative evolutionist" (Howard, 2001) Erasmus Darwin, Charles Darwin was born into a wealthy family, allowing him the opportunity to study what he wished. After giving up a brief career as a medical student in Edinburgh, Darwin took the retrospectively ironic step of moving to Cambridge to become an Anglican priest. But his interest in science developed and led him at the age of

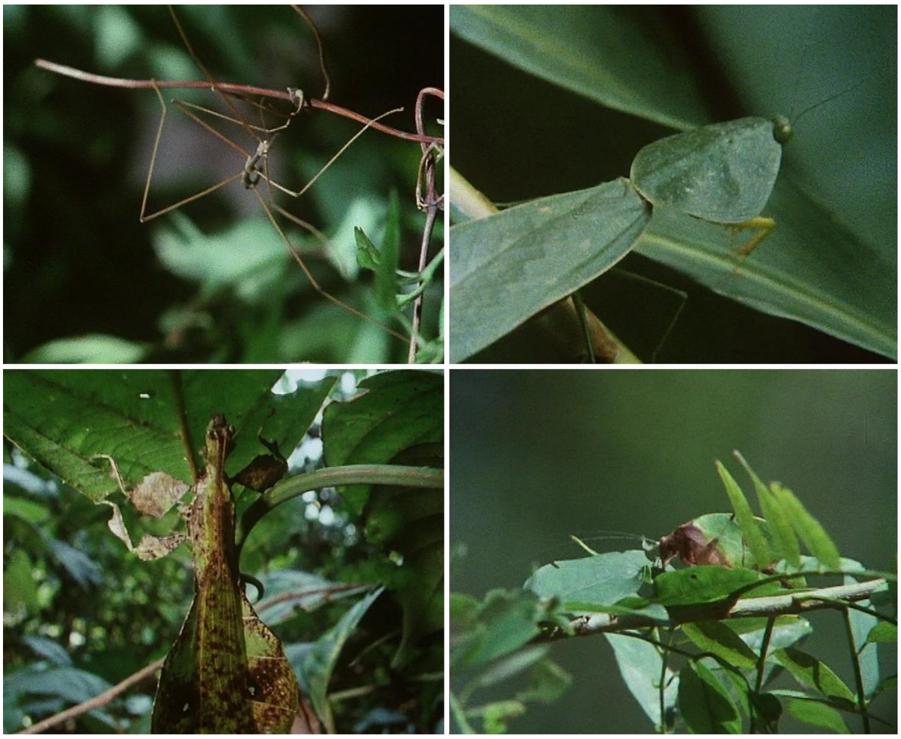


Figure 2: Stick insects come in huge variety of shapes, but each subspecies is closely adapted to a particular species of flora.

Images copyright of BBC Worldwide Ltd. (Attenborough, 1984)

22, thanks to his tutor's recommendation, to be accepted as the naturalist on board a five-year scientific voyage by the *HMS Beagle*. Darwin claimed that the "voyage of the Beagle has been by far the most important event in my life and has determined my whole career" (Darwin and Henry, 1974) and there is no doubt that the places that the Beagle visited, in particular some isolated islands, were to play a large part in shaping Darwin's thoughts about evolution.

Most biographers assert the principal influence on Darwin while aboard the Beagle was Charles Lyell's *Principles of Geology*. Prior to the theory of natural selection, the earliest battles fought by scientists against the scriptural dogma of the Church were fought by geologists:

> "It was inevitable that a geological science which looked at the surface of the earth as a mobile and changing structure and part of a mobile and changing cosmos should eventually come into direct conflict with theological limitations on the development of science. Historical geology, with its emphasis on slow and continued processes, introduced a new and almost limitless timescale for the past evolution of the earth which recognised none of the miraculous and instantaneous events of the Mosaic creation story." (Howard, 2001)

Alongside evolutionary geology were the natural sciences that represented the prevail-

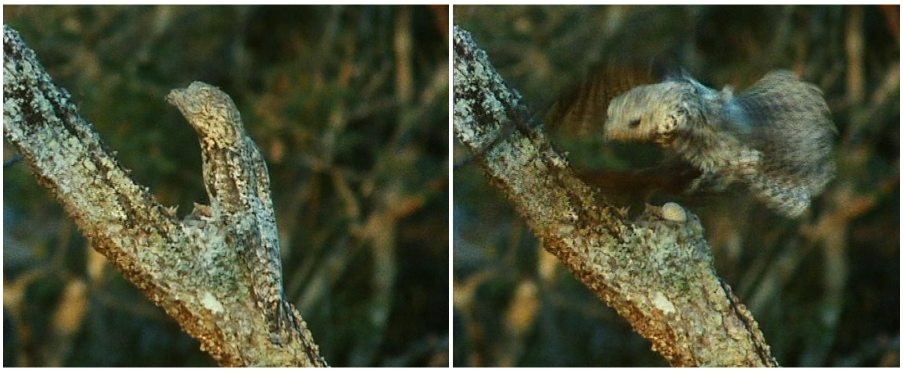


Figure 3: The feathers of Brazilian potoo closely resemble both the colour and textures of bark and lichen, allowing it to mimic a tree stump when sitting motionless on its nest.

Images copyright of BBC Worldwide Ltd. (Attenborough, 1984)

ing scientific beliefs of the time, some of which were intermingled with religious sentiments regarding man's "place" within nature, and the widespread belief that there was a well-defined hierarchy of species or types, and that such types persisted unchanged through time. For example, as Howard points out, the concept of permanence among species type was closely correlated to the story of Genesis, where God created all living things on a single day and their types continued unchanged to the present day via reproduction.

But our ability to classify distinct types or species had already been questioned. Linnaeus had begun his *System Naturae* in 1735 with the conviction that a distinct categorisation was possible for all species, but ended his life doubting it was achievable (Howard, 2001). This wasn't the only messy problem that failed to fit with the vision of a divinely created natural system. Others more firmly attached to "natural theology", such as the Anglican cleric Malthus, sought in 1809 to justify why an ideal creation countenanced a system that allowed a species population to exhaust its resources and suffer. Malthus argued that populations always increase geometrically where there is no competition for resources. What made Malthus different from previous justifications for nature's "unreasonableness" was that he declared that competition between species was a "law of nature" and that man was no different from any other creature with respect to competition. This was at least a shift from most classifications that painted man at the head of creation.

One can draw from early Victorian thinking to depict the background from which the theory of natural selection sprung, but whether such opinion would have had much sway on a passenger of the *Beagle* when it was several thousand miles from home is doubtful. What we do know is that Darwin took with him Lyell's work, and Lyell's second volume of the *Principles of Geology* reached Darwin midway through his voyage. In that volume, Lyell dealt with biological evolution and in particular, responded to a theory of evolution put forward by Jean Baptiste de Larmark in his book *Philosophie Zoologique* (1809). Lamark not only emphasised Linnaeus's doubts about the difficulties of classification, he noted the specialist adaptations of which nature was capable and presented a theory to explain such adaptations. Lamark claimed, without evidence, that the evolution of animals was propelled by their recognition of "new needs", which

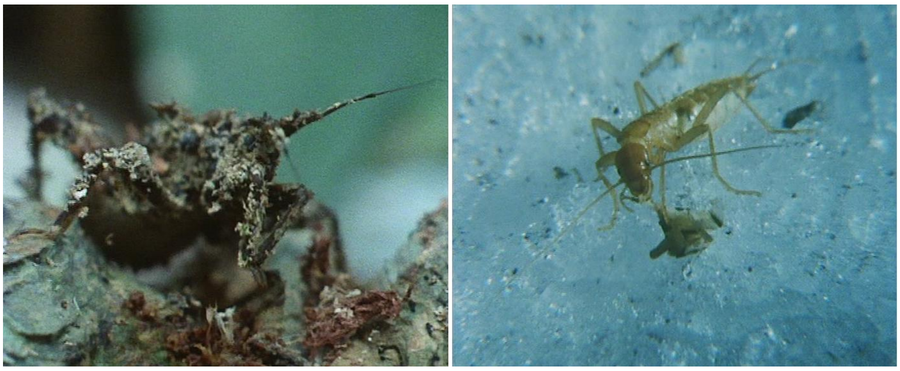


Figure 4: Two specialist adaptations: the lichen beetle (left) exudes a glue so that it can stick pieces of lichen to itself, while a relative of the cockroach, *grylloblattodea* (right), scavenges high up above the snowline on the Himalayan mountains. It is so closely adapted to temperatures below freezing that the heat of your hand would kill it in a few minutes.

Images copyright of BBC Worldwide Ltd. (1984)

in turn provoked behavioural change to satisfy those needs, and this in turn caused structural change to make the behavioural change more efficient (Howard, 2001). This neat circularity was complete when the structural changes made in the creature's lifetime were inherited by its offspring. But while Lyell came up with his own ideas about biological evolution (later rejected by Darwin), his importance to Darwin was that he effectively dismissed Lamark's ideas as speculative, i.e. that the *mechanism* of evolution had not been proven, even by argument (Howard, 2001). It was Lyell's insistence that the mechanism required a scientific explanation that led Darwin to concentrate on this first, rather than why variations occurred between species, or the more established problem of species classification.

3.1 The Three Generalisations

The mechanism involving the flawed replication of DNA code through which inheritable variation operates was not found in Darwin's lifetime. However, the attributes manifested by the process were deduced by Darwin through observation and argument. Darwin drew up three independent generalisations which allowed him to argue for the theory of natural selection:

Variation: no two individuals are identical within a population.

Hereditary characteristics: the variation expressed as individual characteristics is inheritable from parents to offspring.

Multiplication and competition: Malthus's observation — about population increase where no competition for resources exists — means that as resources are always finite, competition must act as a brake on population growth.

If these processes were acting, then hereditary variations within a population that allowed an organism to survive would stand a greater chance of being passed on to their offspring. This was the method that explained how characteristic adaptations to the environment evolved within species and why species diverged.

However, geographical divergence between species was not entirely explained by natural selection. A new species isolated on an island or separated by a mountain range from a similar species presented an easy case, but Darwin found it harder to provide convincing arguments for why speciation still occurred where there was no geographical barrier. This area would continue to cause difficulty for Darwin in his later years, even to the extent of damaging his reputation when he published an account of it (his hypothesis of *panegenesis*) which veered dangerously close to Lamark in explaining the process of heredity. Richard Dawkins points out that this may have had its roots in 19th century views that heredity was a blending process. He comments:

> ... if heredity is of this blending type, it is almost impossible for Darwinian natural selection to work because the available variation is halved in every generation. Darwin knew this, and it worried him enough to drive him in the direction of Lamarckism. (Dawkins, 1998)

Despite the difficulties Darwin faced trying to explain heredity and variation, his theory of natural selection went on to become the dominant explanation of evolution which the discovery of DNA, almost a century later, would do little to change. Indeed, some would say the discovery has strengthened the theory of natural selection, even though in the words of Richard Dawkins, biologists now had to turn "from the fact of evolution to the less secure theory of its mechanism" (Dawkins, 1998).

3.2 Neo-Darwinism

> "The definition that I want comes from G.C. Williams. A gene is defined as any portion of chromosomal material that potentially last [sic] for enough generations to serve as a unit of natural selection." (Dawkins, 1989)

The solution to Darwin's problem over the blending nature of hereditary was published, unknown to him, by the German Gregor Mendel in 1865. Mendel's theory was that heredity was particulate, rather than blending in nature, so that parents pass on to their offspring discrete hereditary particles. The mechanism underlying this was not provided until the twentieth century, with the discovery that how particular genes are inherited from a parent — and genes are particulate in nature, either they are inherited or they are not. There is no half-way, partial inheritance. It is claimed (by neo-Darwinists) that this makes all the difference to the mathematical plausibility of the theory of natural selection. Dawkins states that Hardy and Weinberg were the first to realise:

> "there is no inherent tendency for genes to disappear from the gene pool. If they do disappear, it will be because of bad luck, or because of natural selection — because something about those genes influences the probability that individuals possessing them will survive and reproduce. The modern version of Darwinism, often called Neodarwinism, is based upon this insight." (Dawkins, 1998)

Dawkins was one of those who championed neo-Darwinism, and he summarised the modern genetic theory of natural selection as follows.

The genes of interbreeding animals constitute a gene pool. The genes compete, but in practice spend their time either sitting in individual bodies which they helped to build, or travelling from body to body via sperm or egg in the process of sexual reproduction. Sexual reproduction shuffles the genes, and it is in this sense that the long-term habitat of a gene is the gene pool. Any given gene originates in the gene pool as a result of a mutation, a random error in the gene-copying process. Once a mutation is formed, it can spread by means of sexual mixing provided that its carrier is able to sexually reproduce. Good carriers will contribute more to the gene pool than reproductively less successful ones. Any given gene in a gene pool is said to have a frequency, as it is likely to exist in the form of several copies, all descended from the original mutant. Some genes such as the albino gene are rare in the gene pool, others are common. At a genetic level, evolution may be defined as the process by which gene frequencies change in gene pools (adapted from (Dawkins, 1998)).

For Dawkins, although natural selection accounts for the "perfection of adaptation" in nature, it is of primary importance only because of its consequences for the survival of genes in the gene pool. If a gene is successful in creating a good body that reproduces successfully, then it ensures its own survival. Dawkins (1989) developed this idea in great detail in his first book, *The Selfish Gene*. In it he suggested that in the world of the selfish gene ("what is a single selfish gene trying to do? It is trying to get more numerous in the gene pool") there is no individual altruism in the carriers of genes other than kin selection, that is to say, "a gene might be able to assist replicas of itself that are sitting in other bodies". The logical extension to this, Dawkin argues, is that almost all behaviour is genetically predetermined to aid the survival of the gene and nothing else. This he argues, explains all manner of bizarre behaviours:

> Mantises are large carnivorous insects. . . If the female gets the chance she will eat [the male]. . . It might seem most sensible for her to wait until copulation is over before she starts to eat him. But the loss of the head does not seem to throw the rest of the male's body off its sexual stride. Indeed, since the insect head is the seat of some inhibitory nerve centers, it is possible that the female improves the male's sexual performance by eating his head. (Dawkins, 1989)

Dawkin's one caveat to the selfish gene is that humans may be the one organism capable of resisting genetically determined behaviour through cultural and moral values.

Dawkin's ideas helped swing popular opinion towards a belief that genes explain "life", and persuaded many that the notion of competitive genes fitted perfectly into the theory of natural selection. But such genetic reductionism began to look less convincing as more became known about the developmental process of molecular biology and the constraints of the environment. It became apparent that genes do not have a free hand when it comes to altering features that could affect the viability of the organism. Conflicting genetic evidence also started to muddy the waters concerning the separate evolution of similar organs across species, such as the eye (see §3.2). But even outside these discoveries, there were there those who remained sceptical about the presence of minute variations in the gene pool explaining the eventual appearance of new species.[2] Brian Goodwin, one of those who sought to bring the generic complexity across nature (including developmental biology) back to the forefront of the evolutionary debate, commented a few years after *The Selfish Gene* was published that

[2] When I refer to such scepticism, I am referring to those in the scientific community who wish to be convinced of better scientific arguments, rather than those who counter such arguments with religious bias.

neo-Darwinism failed to explain "large scale aspects of evolution, including the origin of species":

> "New types of organisms simply appear upon the evolutionary scene, persist for various periods of time, and then become extinct. So Darwin's assumption that the tree of life [i.e. speciation] is a consequence of the gradual accumulation of small hereditary differences appears to be without significant support. Some other process is responsible for the emergent properties of life, those distinctive features that separate one group of organisms from another. . . " [3] (Goodwin, 1994)

Goodwin questioned whether the mechanisms of evolution could be reduced to the action of genes alone. Genetic mutation painted a convincing picture for small scale changes — the "fine tuning" of varieties — but not differences of *type*: fish from amphibian, worms from insects, or horsetails from grasses. Darwin's original difficulty to explain speciation began to rear its head again, despite the best efforts of neo-Darwinists to put the argument to bed. This is a broad topic and one we can summarise only a part of. The next section tries to indicate the nature of current debates over speciation and functional features, and how these debates now include evidence from diverse areas, such as developmental biology and gene regulation, and the development of similar functional features across the species divide.

3.3 The Role of Complexity in Evolution

Fully developed forms of animals and plants, with their millions of eukaryotic cells acting in concert to provide much larger scale functionality, can seem a long way from the genetic code that translates a string of amino-acids to some proteins. The gap between the two is the part investigated by developmental biology.

Goodwin, in his book *How the Leopard Changed its Spots*, argued that genes are not the whole picture, and that complex organs have developed the way they are due to a robust, morphological process of development that was both helped and constrained by its environment (Goodwin, 1994). Goodwin's choice of the eye as one of his principal examples was deliberate. It was the organ that Darwin had gone to enormous effort to convince people that such "organs of extreme perfection and complication" could have developed by the infinitely gradual process of inheritable variation and natural selection, even though in his own words, it "seems I freely confess, absurd in the highest possible degree".[4] It was his defence of the evolution of the eye that provided one of the more famous quotes in *The Origins of Species*:

> "If it could be demonstrated that any complex organ existed, which could not possibly have been formed by numerous, successive, slight modifications, my theory would absolutely break down. But I can find out no such case." (Darwin, 1859, pg. chapter VI)

[3] Goodwin and others, such as Gould (1989), may not have had access to Darwin's notebooks, particularly some of his pencil sketches, which give a less rigid idea of his tree of descent than is often portrayed. Darwin was perfectly aware that his tree of descent could not simply be an ever-growing tree of diversification, but instead was somewhat fragmented, more "like a piece of coral, with some parts dead and missing at the root, some parts alive and growing at the tips." (Howard, 2001)

[4] To counter the scepticism he met, Darwin felt compelled to extend his arguments for the evolution of the eye between the first (1859) and sixth (1872) edition of the *The Origins of Species*.

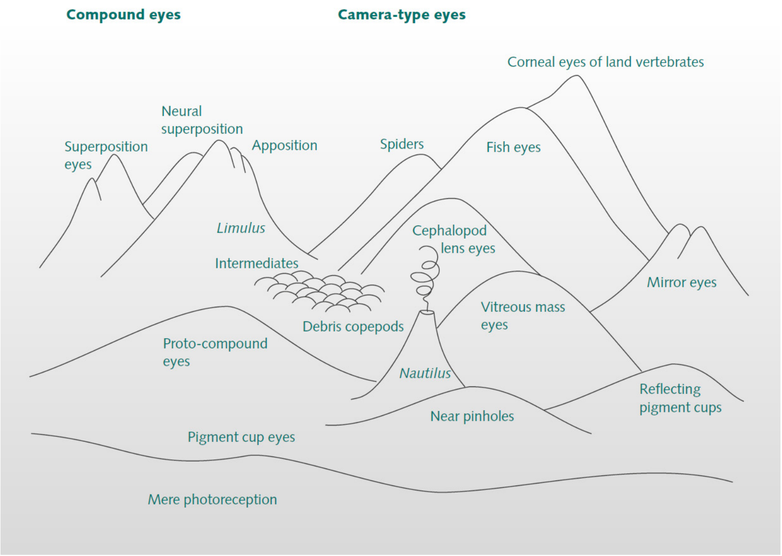


Figure 5: Possible landscape of eye evolution created by Mike Land. Height represents optical quality and the ground plane evolutionary distance. Land writes that "Climbing the hills is straightforward but going from one hilltop to another is near impossible" (Quoted by Ferald (2001)).

Image and text taken from Ferald (2001) (originally in Dawkins (1996))

Darwin marshaled all manner of examples to suggest that primitive eyes were present in many organisms in different stages of evolutionary development, that nerve endings in many creatures were sensitive to light and that the "unerring skill of natural selection" would forever refine the simplest form of eye towards greater perfection.

By re-examining the evolution of the eye, Goodwin created a long-running debate between molecular, developmental and evolutionary biologists. Dawkins wrote a direct response to Goodwin from the neo-Darwinist perspective in Chapter 5 of his book *Climbing Mount Improbable* (Dawkins, 1996). In it he gives a picture by Mike Land that tries to represent the evolutionary landscape of all known eyes (Fig. 5). Dawkins argued that climbing an evolutionary peak of optical sophistication was easy enough for a species, but no species could jump from one mountain top to another. Others have pointed to the fact that the simple compound image-forming eye appears to have been invented at least three times during the course of evolution (Fogel, 2000, pg. 35), and therefore types of eye couldn't be explained by a diagram such as Fig 5.

But it seems now as if neither Dawkins (for neo-Darwinists and gene reductionism) nor Goodwin (influence of developmental processes and morphological constraints) can claim an emphatic victory over the root causes of differences in species type and that the truth lies somewhere between the two. Russell Fernald, writing in (2001), sums up how the arguments swing this way, then that, almost as each new research paper is published:

> Have the structural similarities among eyes resulted from evolutionary convergence due to similar selective pressures (analogous) or from descent from a common ancestor (homologous)? This distinction is particularly

> hard to draw when comparing eyes because the physical laws governing light greatly restrict the construction of eyes. Similar eye structures may have arisen in unrelated animals simply because of constraints imposed by light. ... However... opsin has a significant DNA sequence homology across all phyla. Remarkably, recent work by Gehring and Ikeo (Gehring and Ikeo, 1999) has shown that features of ocular development in different phyla can be coordinated by a homologous "master" gene, Pax-6. That a single gene could trigger construction of an animal's eye in diverse species led to their proposal that eyes are monophyletic, i.e. evolved only once. (Ferald, 2001)

What can be said about the ongoing debate on the evolution of complex organs is that developmental processes and environmental constraints seem to be playing a larger part in the debate than in the early eighties, when they were largely discounted by those advocating gene reductionism. In the following sections, we examine some of the complexity that makes up the physical development of the phenotype and assess why such developmental processes are important in their own right to the creation of complex organisms.

4 Aspects of Development

If evolution has a guiding hand in creating successful designs, that hand is constrained by developmental processes. Development controls whether those designs can be built or not, and helps exploit the complexity of the physical world to allow life to be constructed. Genes play a vital role in development, but they are also restricted in what they can do or change. This section draws heavily from the introduction to Kumar and Bentley's book *On Growth, Form and Computers* (2003), which in turn draws almost exclusively from Wolpert's *The Principles of Development* (1998).

Wolpert describes development as "the emergence of organised structures from an initially very simple group of cells" (Wolpert, 1998). The process of moving from simple cells to more complex structures is almost entirely governed by proteins, but in order to understand how proteins influence development, we need to first become familiar with development at the level of the cell. This overview therefore starts with a high level view of cell development, then looks at the internals of cells, covering the synthesis of proteins within cells and cell signalling, before ending with a look at DNA and the role of proteins in gene regulation.

4.1 Cells

Cells are complicated. They have their own internal logic, they can act as sensors to respond to external signals and are able to emit signals that govern the behaviour of other cells. Cells and their proteins have resulted in a sophisticated control mechanism that not only dictates how development proceeds, but which also controls the running of bio-machinery after development.

Cells have two forms: *prokaryotic* (bacteria, including the important, large group of cyanobacteria, sometimes referred to as "blue green algae") and *eukaryotic* (everything else). The latter encases its DNA within a membrane, giving advantages in terms of control, defence and the ability to process information. Many regard the eukaryotic cell as the foremost achievement of evolution. Wolpert goes as far to claim that:

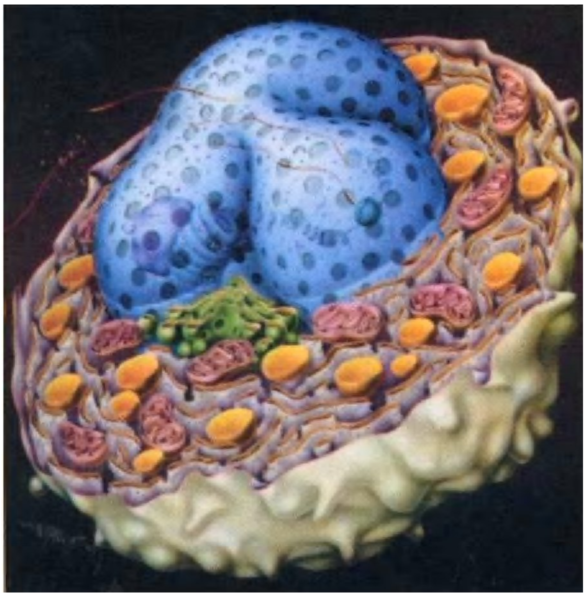


Figure 6: "Eukaryotic cells have an extensive array of membrane-bound compartments and organelles with up to 4 levels of nesting. The nucleus is a double membrane. The external membrane is less than 10% of the total."

Image and text taken from Cardelli (2005)

> "Once you had the eukaryotic cell, from the point of view of evolution and development, it was downhill all the way: very, very easy... Among the basic components required for development, I can think of virtually nothing that eukaryotic cells did not have which is required for the developmental process." (Wolpert, 2003, pg. 59)

Certainly when we look at the extraordinary information processing capabilities of the cell one can only marvel that the system scales so well.

4.1.1 Cell Signalling

The cell container is a membrane (Fig. 6), a subtle discriminator over what passes through it — some proteins may enter, others can leave, the membrane determines which. This *selective permeability* gives cells a universal filtering mechanism which can listen to broadcast messages as though tuned into a single hormonal frequency. Cells act as marvellously sensitive sensors and emitters of signals. They can operate in a wide range of media, from pheromone signalling in an air stream over miles of open space, to hormone signals carried in the blood stream in animals, to cell membrane surface proteins signalling to their immediate neighbours during development, to intra-cellular signalling for the presence of invading pathogens, cells can do it all. The mechanisms they use are equally complex and diverse (Hancock, 2003). Without cell signalling, it is hard to envisage how multicellular organisms, that are awash with information, could evolve, develop or exist at all.

The physical sequence of a cell signal generally follows the pattern of a molecule being released by one cell to be detected by the receptors on another cell, but there are variations on this, such as the detection of membrane proteins on one cell by receptors on another or the transfer of small molecules through “gap junctions”.

Cell signalling plays an important part in development, particularly mechanisms such as juxtacrine signalling in early development when the cells may be closely packed together (Wearing et al., 2000). But more generally, without cell signalling an organism could not even begin to differentiate positional information to set up the axes of the body plan (see Spemman’s ‘organiser experiment’ on amphibian embryos in Wolpert (1998)). Without cell signalling, only asymmetric distribution of transcription factors during cleavage could cause cells to become different from one another (Kumar and Bentley, 2003, pg. 5–6, but see also §5.2). Developmental processes, as we shall see in later sections, require context specific gene expression in order to achieve particular morphological functions.

Finally, binding a protein to a cell’s receptor can trigger internal cell reactions that relay information back to the genome via *signal transduction* pathways. Such pathways can be viewed as signal cascades, sometimes involving many events that can be used by the cell as an amplification mechanism. Pathways are complex, as during the “cascade” there are opportunities for increased interaction or influence from other pathways leading to a complex interplay of genes, proteins, even conflicting signals.

4.1.2 Cell Division

Cells multiply by duplicating their contents and splitting in two. The cycle of cell division contains several phases: *interphase*, where DNA replication and the production of proteins occurs, *mitosis* or nuclear division, and *cytokinesis*, which concerns the division of a cell’s cytoplasm after nuclear division (Kumar and Bentley, 2003).

Cell division is either symmetric or asymmetric. Symmetric division occurs when the plane of cleavage divides the cell into equal sizes with equal proportions of cytoplasmic proteins. Asymmetric division results in unequal sizes of daughter and parent cells containing different cytoplasmic factors. The different levels of cytoplasmic factors plays a large part in local regulation of gene expression in the embyro (§5.2).

4.2 Proteins

The broad behaviour range of eukaryotic cells is due to their interaction and production of proteins. With each cell containing several thousand proteins, the scope of potential functions a cell can achieve is huge. Proteins form not just the structural components of cells and tissues, but are involved in both signalling and the general “house keeping” of the cell, such as transporting or storing oxygen or haemoglobin molecules. Proteins are also involved in defensive mechanisms such as the production of antibodies.

In terms of their chemical structure, proteins are polymers comprised from twenty different amino acids. These amino acids are joined by peptide bonds giving long *polypeptide* chains, hundreds or thousands of amino acids in length. The chains “fold”, taking on distinctive three dimensional shapes which are critical to their function. Conceptually, proteins might be described as rather like a scrunched up ball of string, with the string itself being composed of up to 20 differently coloured segments, which can be repeated, and whose sequence order is specified by the order of nucleotides in the gene (see next section). The *spatial proximity* of the segments in their scrunched up ball largely determines the functionality of the protein. However understanding proteins in the real world is less easy than this simple, visual conceptualisation. 3D modelling software can help indicate the physical complexities of proteins by producing “ribbon”

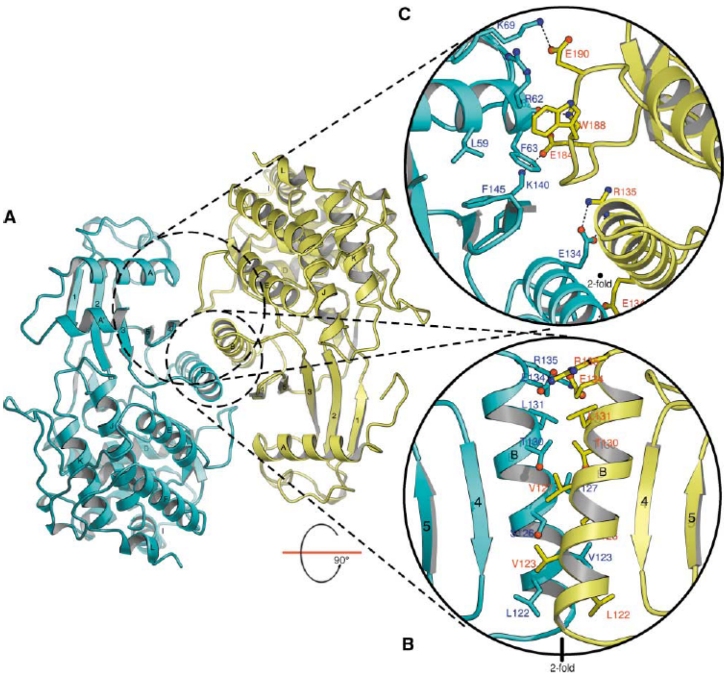


Figure 7: The structure of the Choline Kinase CKA-2 Dimer. (A) Shows a ribbon drawing of the CKA-2 dimer looking straight down the two-fold axis of symmetry. (B) and (C) show enlarged views of the residues involved in interactions between the two symmetry-related helices. The view in (B) is orthogonal to those in (A) and (C).

Image and text reproduced from Peisach et al. (2003)

drawings (see Fig. 7), but understanding the modelled structures remains difficult, particularly when trying to determine a functional role from protein folding.

4.3 Genes Code for Proteins

The genome specifies when and where proteins are synthesised, and there are those who argue that genes have no function other than to specify proteins (Wolpert, 2003). As hinted at in the introductory section on proteins (§4.2), complicated networks of interactions involving proteins and genes are built up within cells. Proteins can promote or inhibit other proteins, and the absence or presence of certain proteins can affect the expression of a gene, which would in turn affect the production of another protein. These forms of "cascading" control sequences in protein production are termed gene regulatory networks and can be extremely complex. A minor industry has built up trying to infer such networks by analysing the massive amounts of data produced during the study of gene expression. It is important to understand that it is via gene regulatory networks and protein signalling pathways that physical feedback loops are possible during the development of the organism. Feedback and control during development gives the organism a measure of flexibility in response to its environmental conditions.

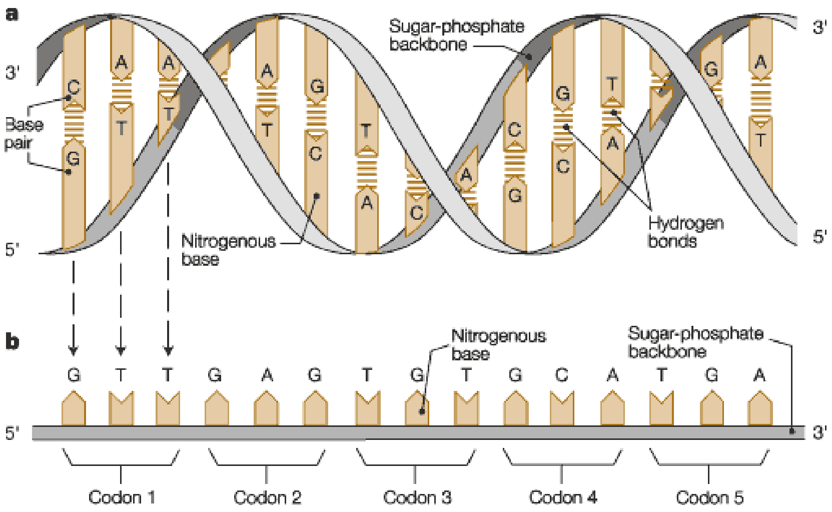


Figure 8: The double helix structure of DNA.
Image reproduced from Alberts et al. (2002)

4.3.1 Deoxyribonucleic Acid (DNA) and the translation process

In what would appear to be a case of massive redundancy, every cell in every living organism contains the unique instruction set for that organism's construction. That might seem a lot of information for each cell to contain, but the cell as it divides and multiplies will make use of just a tiny part of it. The cell must construct itself from proteins, and the rules for synthesising those proteins are contained in the DNA sequence that specifies the proteins for that cell.

The discovery of the structure of DNA by Crick and Watson in 1953 helped understand how genetic information copied itself. Rather like the polypeptide chains of amino acids that make up proteins, DNA contains two polynucleotide chains of nucleic acids, linked to form the two strands of a double helix. The nucleic acids are simpler than their amino acid counterparts in proteins and contain just four bases (usually represented by their first letters), two purines: adenine (A) and guanine (G), and two pyrimidines: thymine (T) and cytosine (C) (see Fig. 8). Along the two polynucleotide chains of the helix, purines and pyrimidines face inward and pair up in what are termed the complementary base pairings: G-C and A-T. [5] One consequence of complementary base pairings is that a single strand of DNA or RNA can act as a template for the synthesis of its complementary strand, allowing nucleic acids the capability of directing their own replication (Kumar and Bentley, 2003, pg. 8).

In a section of the DNA, the order of the nucleic acid sequences governs the creation of amino acids that make up the protein. However, DNA is passive. It does not directly control the protein synthesis generated in the cell cytoplasm. That process is controlled by synthesised RNA, known as messenger RNA (mRNA), itself synthesised from the DNA template. In order to construct a protein, the nucleotide sequence in the mRNA is read three bases at a time, in what are termed nucleic triplets, or *codons*, with each codon corresponding to a single amino acid (Kumar and Bentley, 2003, pg. 8). This allows some redundancy, with some of the amino acids being encoded by more than

[5] In RNA, thymine is replaced with uracil (U), giving a base pairing of A-U.

one codon.

The translation of the codon into an amino acid is carried out by transfer RNA (tRNA) molecules, with at least one tRNA molecule being specific for an amino acid and a particular codon (although some amino acids may require the services of two or three different tRNAs). The amino acid is attached to the tRNA by an enzyme (aminoacyt-tRNA synthetase) which is again specific to that amino acid and its corresponding tRNA molecule. In a process similar to DNA replication, each kind of tRNA has a sequence of 3 unpaired nucleotides known as the *anticodon*, which bind into complementary base pairs in exactly the same way as the double helix strands in DNA, except this pairing is to the codon in the mRNA molecule.[6]

Having covered the basic building blocks of protein transcription, we can now look at what restrictions development places on evolution, how developmental processes differentiate cells and how evolution is able to re-use functional genes in new locations.

5 Evolutionary Developmental Biology

Although the supporting evidence changes, evolutionary developmental biology is still tussling with the same question that Darwin hesitatingly put to his readers: how did organs "of extreme perfection and complication" such as the eye evolve?

> "One of the most important concepts in evolutionary developmental biology is that any developmental model for a structure must be able to account for the development of earlier forms in the ancestors." Wolpert (2003, pg. 47), citing Carroll et al. (2001).

Over a hundred years after Darwin first posed the question, and even with our increased knowledge about the developmental process, being able to account for earlier ancestral forms presents a fascinating challenge and as we shall see in later sections, its answers may help computer scientists as much as biologists.

Evolution and development are closely interwoven. In §3.3 it was argued that the importance of developmental processes has begun to be given greater credence in evolution. Wolpert is not merely being provocative when he states that "DNA is rather boring and passive" (Wolpert, 2003, pg. 47), he is also promoting the extent to which proteins are responsible for the complexity we see around us. It is true that all changes of form and function are down to changes in DNA, as it is DNA that determines which proteins are made, when and where. But this implies that there is a "one-way flow of information, from DNA to proteins" (Banzhaf and et al, 2006) that forms the basis of functional exploration by evolutionary search, and this is clearly not the case. Organisms evolve *into* environments, and this would be impossible unless the flow of information was two-way. As the environment into which an organism develops isn't a given constant, the developmental process needs constant feedback and gene regulatory networks cannot obtain information about their environment except via protein interaction. The presence of certain proteins (transcription factors) inhibits or promotes gene expression, so that genes encode not only for the proteins that build the biological infrastructure, but also for the proteins that control their own self-expression.

The ability of genes to encode for proteins that allow them to self-construct their own rules of operation is one of the more remarkable features of development. But

[6] Although I can't include it here, I can heartily recommend John Kyrk's animation of how the translation process constructs proteins, which is viewable online from http://www.johnkyrk.com/DNAtranslat.swf.

there are restrictions on how those rules evolve to explore functional space, and the evidence for that comes from the field of evolutionary developmental biology.

5.1 The Development of Complexity

For evolutionary developmental biologists,[7] every structure has two histories that relate to how it developed: *ontogeny* (its complete development to maturity) and *phylogeny* (its evolutionary history). Wolpert states that, "ontogeny does not recapitulate phylogeny", as many embryos pass through common phases that their ancestors passed through (Wolpert, 2003, pg. 47).[8] For example, all vertebrates pass through a similar phylotypic stage involving the development of the nervous system (neurolation) and the formation of somites (body segments). This suggests that a distant ancestor of all vertebrates passed through this stage, and despite the stages before and after the phylotypic stage diverging in many species, neurolation has persisted to become a feature of all vertebrate development.

Sharing an embryonic stage provides evidence of common ancestors. An alternative source of evidence is to trace the alteration of structures present in ancestral forms in early embryonic stages. An example is the evolution of the branchial arches and clefts that are present in all vertebrate embryos, including humans. During evolution the branchial arches have produced both gills in primitive jawless fishes, and in a later modification, given rise to jaws. But Wolpert makes an important point:

> "These are not the relics of the gill arches and the gill slits of an adult fish-like ancestor, but of structures that would have been present *in the embryo* of the fish-like ancestor." Wolpert (2003, pg. 55) (my emphasis)

This has profound implications. It suggests that while all changes to form result from changes in the DNA, the changes are limited to *where* they can occur. Early embryonic stages appear to be robustly protected against change, perhaps because change here would be dangerous to the organism, but also because change appears to be easier once critical stages of development have passed.

Striking evidence of the restriction imposed on gene mutations that control early development is the almost universal conservation of a group of genes called *Hox* gene clusters. Hox gene clusters control a wide range of developmental processes, such as limb bud or body plan development, and range across species from fruit flies to elephants. An important function of some Hox genes is to specify positional identity in the embroyo. These positional values are interpreted differently in different embryos, so that cells develop into, for example, segments and appendages (Wolpert, 2003). This means that the same genes expressed in a different location, time or context, may give rise to a different morphological form (Carroll et al., 2001). This aspect of re-use is examined in more detail in the following section, as it demonstrates how nature has invented a few, very useful genes, and re-used them widely across species. If we want evolutionary computation to mimic this trick, we need to discover what allows context-specific expression of a gene, and allow that context to in-part define the gene's functional role.

But the universal presence of Hox gene complexes should not be used as blanket evidence that such genes cannot mutate. Quite the reverse is true, but the manner they

[7] This subsection draws heavily from a single source (Wolpert, 2003).

[8] This is a common rebuttal of Ernst Haeckel's theory of recapitulation put forward in 1866, which claims that embryos pass through all their evolutionary stages. A full discussion of the debate (with references) can be found on Wikipedia, `http://en.wikipedia.org/wiki/Ontogeny_and_phylogeny`.

have mutated gives an insight into a key evolutionary mechanism. Gene duplication can occur in a variety of ways during DNA replication and doing so it provides the embryo with an additional copy of the gene. The beauty of this is that:

> "... this copy can diverge in its nucleotide sequence and acquire a new function and regulatory region, so changing its pattern of expression and downstream targets without depriving the organism of the function of the original gene." (Wolpert, 2003, pg. 57)

Haemoglobins (an oxygen carrier in red blood cells) in humans are an example of the evolution of new proteins and patterns of gene expression that have occurred by gene duplication. The duplication that gave rise to Hox genes means that they can be compared across a variety of species, allowing one to reconstruct how they are likely to have evolved from "a simple set of six genes in a common ancestor of all species" (Wolpert, 2003, pg. 57). Thus Hox genes are evidence on the one hand of a certain 'conservatism' in development due to their widespread deployment, and on the other hand they indicate how successful genes that provide functional features can be both kept *and* changed at the same time. Both these aspects are pertinent to evolutionary computation, where there is a need to protect a good solution as part of solving a larger problem.

If the gene mutation is limited to where it can occur, it suggests that functional complexity is hierarchical in nature with changes only possible at the 'leaves' of the tree. The supporting evidence from evolutionary biology is two-fold. Firstly, entirely new structures are rare, evolution tends to "tinker with existing structures":

> "New anatomical features usually arise from modification of an existing structure. A nice example is provided by the evolution of the mammalian middle ear. This is made up of three bones that transmit sound from the eardrum (the tympanic membrane) to the inner ear. In the reptilian ancestors of mammals, the joint between the skull and the lower jaw was between the quadrate bone of the skull and the articular bone of the lower jaw, which were also involved in transmitting sound. During mammalian evolution, the lower jaw became just one bone, the dentary, with the articular no longer attached to the lower jaw. By changes in their development, the articular and the quadrate bones in mammals were modified into two bones, the malleus and incus, whose function was now to transmit sound from the tympanic membrane to the inner ear." (Wolpert, 2003, pg. 55–56).

Although mammals and reptiles appear to have evolved separate mechanisms for hearing, the mechanism actually stems from a common structure. Secondly, comparisons of embryos suggests that those characteristics that are shared by a group of animals appear earliest in their evolution:

> "In the vertebrates, a good example of such a general characteristic would be the notochord (a skeletal rod of tissue enclosed by a firm sheath), which is common to all vertebrates and is also found in other chordate embryos. Paired appendages, such as limbs, which develop later, are special characters that are not found in other chordates and which differ in form among different vertebrates." (Wolpert, 2003, pg. 54)

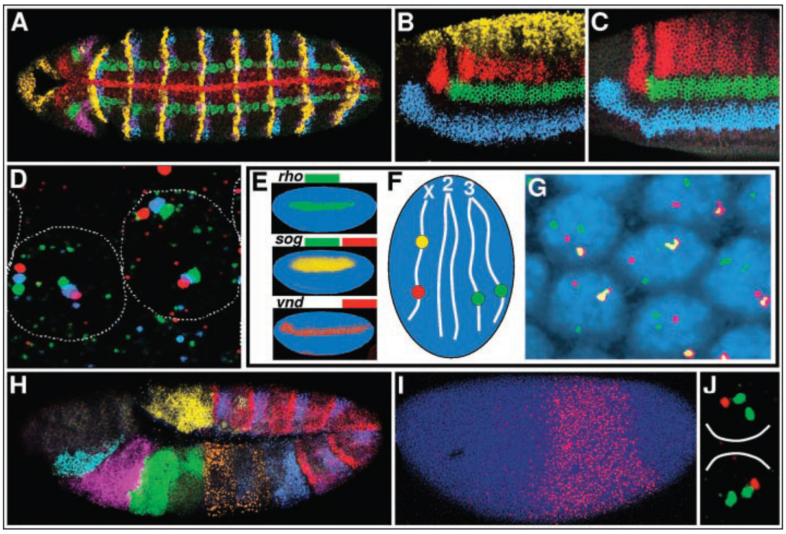


Figure 9: Drosophila embryos showing pattern and stripe formation.
Image reproduced from Kosman et al. (2004)

The evolutionary record therefore provides us with some interesting evidence: shared characteristics occur earlier in evolution, and entirely new structures are, if not infeasible, at least rare. While the fossil record has long suggested these observations, fossil evidence is often patchy and its discovery down to chance. It is the more recent genetic evidence from embryology and developmental biology that has thrown light on the restrictions that evolution operates under. Although mutations may occur at random, developmental processes place restrictions on changes to the DNA, as any change must leave the organism viable. The scope of potential change is therefore narrowed to produce the "tinkering" effect described by Wolpert.

Evidence provided by Sean Carroll and colleagues suggests that most changes are in the *cis*-regulatory region of the genes, rather than in the nature of the protein for which the genes encode (Carroll et al., 2001). The reason for this is that the *cis*-regulatory region is where protein factors — the transcription factors — bind and determine whether or not the gene will be transcribed. The subtle variations in morphological form, or "tinkering", is a by-product of this process, and is explained in the following section.

5.2 Evolutionary Development

Work on developmental processes has uncovered much of what drives morphological variation in organisms. The literature is full of examples of drosophila embryos showing early body plan layout or stripe formation. These pictures are created by attaching fluorescent proteins to certain transcription factors (see Fig. 9) which enables biologists to see the distribution of such proteins across the embryo. As we will see, it is the individual contexts provided by the varying distribution of transcription factors that allows the expression of repetitive morphological structures, such as backbone vertebrae or body segments. Developmental processes are governed by complex networks of gene regulation. In the following sections we outline how regulation works at the level of a single cell nucleus, rather than tackle the complexity caused by cascading networks of

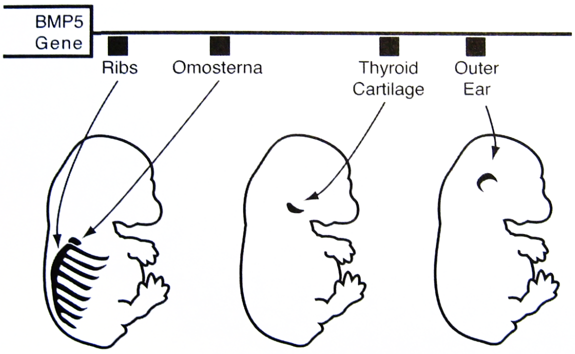


Figure 10: Different switches cause the gene for bone material protein 5 (BMP5) to be expressed in different locations in a mouse embryo.

Image taken (with permission) from Carroll (2006)

control, which is beyond a short introduction such as this.

As mentioned briefly in §5.1, the *cis*-regulatory regions or "switches" employed by gene regulatory networks (GRNs) determine the contexts in which a particular gene is expressed or inhibited from transcribing proteins in the nucleus of a cell. These "switches" allow a gene to be re-used in a variety of contexts, which means that the protein a gene encodes will have the opportunity of interacting with different sets of proteins according to the cell's location in the organism. For example, the different bones in our body are not created by different genes encoding separate proteins for particular bones, but by the same gene being used in different contexts to create the bone material protein for a rib, a sinus, an outer ear and so on (see Fig. 10).

Gene switches work by certain proteins being able to *bind* to small sections of DNA material upstream of where the gene is located. These transcription factors or "binding proteins" act on DNA to inhibit or promote gene expression, and whether a transcription factor is present or not in a particular cell type is determined by that cell's location in the embryo. For example, in Fig. 11, a promoter for a gene is distributed in vertical stripes that extend to the horizontal axis of an embryo. However, the presence of inhibiting transcription factors for the same gene in the lower third and back half of the embryo results in a net expression of the gene as a series of dots along the horizontal axis.

5.3 Gene Expression and Re-use

Pattern formation is the basis of all gene re-use: the same gene is turned on or off according to the presence of the transcription factors that occur in cells distributed throughout the embryo. Research suggests that hardly any morphological features are created *de novo* from new genes (Carroll, 2006; Wolpert et al., 2002), instead different morphological features between species are the result of the same genes (usually one of the four Hox clusters) being employed in different contexts. The Distal-less gene, essential for the formation of appendages, such as limbs or wings, is one example. In butterflies, this gene has evolved an additional "switch". The switch provides a new context for the Distal-less gene to be expressed — in this particular case, that location

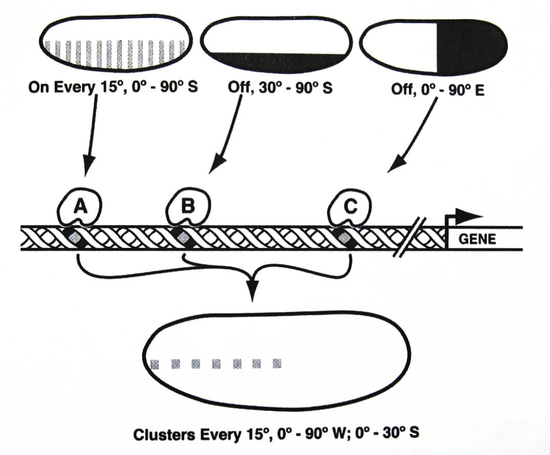


Figure 11: Gene switches acting on a drosophila embryo results in the gene being expressed as a series of dots extending halfway along the horizontal axis.
Image taken (with permission) from Carroll (2006)

is on the wing. In the new context, rather than forming a limb bud, the gene results in an entirely different morphological feature: a spot of colour (see Fig. 12).

The gene switch mechanism allows re-usable, configurable instances of a gene to be expressed in the different contexts of embryo development. Repeated use of a gene in different contexts is called *modularity* by biologists, and gives rise to repetitive morphological structures such as vertebrate backbones, thorax segmentation, rib cages, leaf and wing venation, limbs, etc. Such structures are common in nature, but it has taken researchers a long time to understand the link between switches and gene expression leading to gene re-use. An important part of this interaction is how the switches work to allow the binding process some degree of flexibility.

5.4 Binding Signatures

Transcription factors attach to stretches of DNA by recognising signature sequences of base pairs. For example, a single switch for a gene may consist of several hundred base pairs (bp), lying perhaps several thousand bp upstream of the gene. Within the gene switch, there are usually 6–20 signature sequences (each ∼6–9 bp in length) that affect the expression of the gene concerned (a gene contains ∼1000 or so bp, and a chromosome contains thousands of genes, so millions of bp). Even a short signature length has a huge number of possible combinations (Carroll, 2006).

Signature sequences sometimes require exact matches for every position, sometimes they contain wildcards. Wildcard positions can be filled by all four nucleic acids (Cytosine, Thymine, Adenosine, and Guanine) but are more often limited to pairs of alternatives (e.g T or A, C or G, etc.). For example, Tinman, a gene related to heart development in most species, is highly specific. However, Pax-6 (the gene supposedly controlling the development of sight across species) and the gene Dorsal use wildcards in their binding signatures, represented by **K** (G or T), **Y** (C or T), **M** (C or A), **W**(. . .), etc. (example below is from Carroll (2006)):

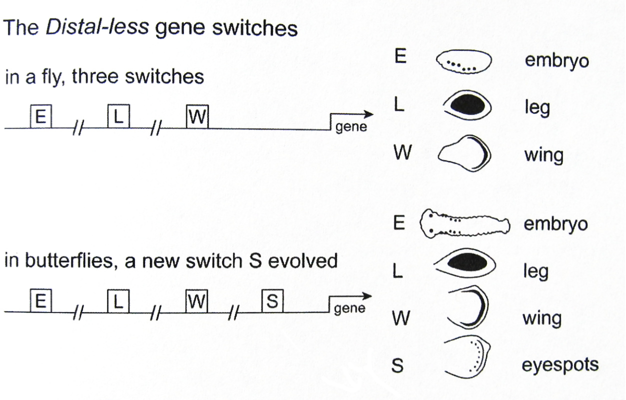


Figure 12: Switches in the Distal-less gene control expression in the embryo, larval legs and wing in both flies and butterflies, but butterflies evolved an additional switch providing a new context for Distal-less (giving wing eyespots — a different morphological feature).

Image and text taken (with permission) from Carroll (2006)

Tinman	TCAAGTG
Pax-6(eyeless)	**KKYM**CGC**W**T**S**AT**KMNY**
Dorsal	GGG**WWWW**CC**M**

Thus Pax-6 has a signature with only 6 specific sites out of 16 possible bp combinations, indicating that it could bind at a variety of locations. This is borne out by experimental evidence that shows eyes can be "grown" in other contexts — such as on wings or legs — by altering the transcription factors present at those locations (Carroll, 2006).

Binding signatures and proteins permit the genome to maintain a *set* of solutions from which it selects how to explore its functional domain. The action of "binding" is one of feedback: the information fed back to the DNA determines which genes will be expressed in that context. By exploring the functional search space in this way, developmental processes have a fundamental impact on which genes are conserved. But their continued presence means that they are also more likely to be re-used by evolution during later developmental processes.

6 Models of Evolution and Complexity

So far we have taken examples from nature, looked at how theories of evolution have developed, and tried to investigate how the evidence provided by our greater understanding of genetics has influenced those theories. We have also looked at the process of construction, including the basic building blocks of life and combined them with a brief excursion into evolutionary developmental biology to see if they could shed some light on the evolution of complex features. However, it should be noted that not all the progress in this area has been done by practitioners gathering samples and amassing data from real life examples. Some important concepts have been developed by theoretical biologists, particularly those such as Kauffman (1995) and Solé and Goodwin

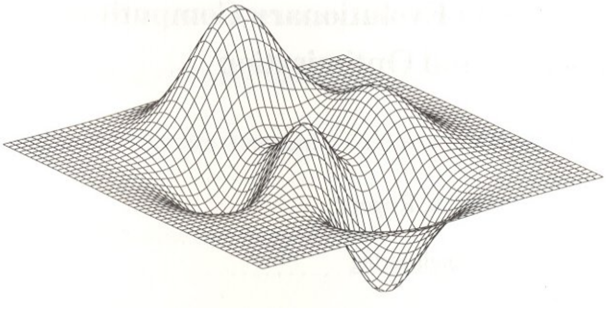


Figure 13: Schematic "adaptive" or "fitness" landscape.

(2000) working on abstract models of evolution and complexity.

One consequence of the restrictions that developmental processes impose on evolution's design is a "smoothing out" effect. Large jumps, as we saw in the theoretical landscape of all possible eyes, are not possible; small, incremental changes of direction are.

6.1 Fitness Landscapes

The analogy of the landscape that represents peaks and troughs of evolutionary success within a population was first devised by the American geneticist Sewall Wright, who coined the phrase "adaptive landscape" to describe the shifting balance of population genetics (Wright, 1932). The adaptive landscape and its associated heuristics is an influential model in evolutionary biology, but its use is not universally accepted. It was strongly criticised by Wright's own biographer, Provine, who declared the heuristic was mathematically uninterpretable (as there appears to be no way of generating the continuous "surface" of the landscape) (Provine, 1986). But it has also been defended by Ruse (1996) and most recently by Skipper (2002) who, in a short survey on the influence of Wright's work, claims that the adaptive landscape diagram remains of use in the study of dynamic behaviours. Despite apparent weaknesses with the model, Wright's adaptive landscape has gone on to be extensively developed by Kauffman, Levin, Johnsen and others, in the investigation of what they term "adaptive walks" by organisms (Kauffman and Levin, 1987). They term their models "fitness landscapes".

The model is a simple one. An individual within a species is represented as a string of genes that defines its genotype. The string itself has a real number associated with it. This number defines the *fitness* of the string in terms of the phenotype it produces. The assessment of fitness as a single or two dimensional trait is one aspect of the model that has been criticised (see following paragraph). The distribution of fitness values over the space of all genotypes gives the *fitness landscape*, and all members of the population map onto that landscape according to their fitness value. If an individual has a high fitness value, it falls somewhere near a peak on the landscape; if it has a low fitness value, it is in a trough. A schematic fitness adaptive landscape is shown in Fig. 13, although most landscapes are considerably more rugged (and higher dimensional, see § 6.3) than this, something that has important implications. Initially (at least conceptually), the population of phenotypes falls over the landscape with a random distribution, according to the fitness values given to them by their genes. However, after undergoing selection and mutation (more details of this are given in § 8.1.4), individuals start to gain higher fitness values and start to “walk” towards the nearest peak. For the purpose of the model, the process of adaptation or improving a phenotype’s fitness values is equivalent to walking up a peak.[9]

A crucial part of the model is the definition of the fitness landscape, as it is this which measures how the fitness of the population moves towards some optimal configuration. Although the model is intuitive to visualise for two fitness traits that can be isolated and easily quantified, the situation becomes more difficult as we try to represent complex, interrelated factors. But let us first examine the case of single fitness traits. In an imaginary world, I might have a creature that is predated on by an animal that can run fast. The environment therefore selects individuals who can run fast as these individuals have more chances of escape, by reproducing those individuals who are located higher up on the fitness peak that represents the ability to run fast and discarding others. Over time, the individuals who survive are all fast runners. At least that is the theory of the model. In reality, there is no evidence that evolution works on single traits. It may select for longer legs, faster muscles, and so on, but it could equally evolve the strategy of growing sharp horns, becoming poisonous or evolving a thick skin. Even in the former case, there might be many different ways of achieving the same fitness value as it is measured solely in terms of running speed. For the purpose of modelling selection based on a single trait the model is adequate, and some authors even go so far to claim that the model reflects reality:

> “Selection acts on collections of interactive phenotypic traits, not on singular traits in isolation. The appropriateness of an organism’s holistic functional behaviour in light of the physics of its environment is the sole quality that is optimised through selection”. (Fogel, 2000, pg. 35)

Fogel thus suggests that providing one selects (or assesses) on the basis of an aggregated, holistic functional behaviour, the complex genetic relationships that cause the behaviour can safely be ignored. This perspective is one that is interested in the perceived “optimisation” of that behavioural trait.

The problem for fitness-landscape models is that we are forced to assess all fitness in terms of such amalgamated functional features, which in the real world are the complex expressions of genes and proteins. But as this would lead to a very difficult to visualise model, involving thousands of dimensions, fitness landscapes are instead generally plotted against just two dimensions representing two traits in the phenotype. Often, there is a sleight of hand at work here, in that we need to forget the multidimensional nature of the genotype space. Stuart Kauffman gives a flavour of this easy-to-imagine model:

> “Consider a set of all possible frogs, each with a different genotype. Locate each frog in a high-dimensional “genotype space”, each next to all genotypes that differ from it by a single mutation. Imagine that you can measure the fitness of each frog. Graph the fitness of each frog as a height above that position in genotype space. The resulting heights form a fitness landscape over the genotype space, much as the Alps form a mountainous landscape over part of Europe”. (Kauffman, 2000, pg. 18)

The appealing nature of such illustrations is shown in Fig. 14 from Solé and Goodwin (2000, pg. 257), where a fictitious landscape has been plotted based on fossil tribolites

[9] One could select for negative values and this might be more apt, particularly for aspects of development as one could model “basins of attraction” (see Fogel (2000, pg. 37) and also § 6.4).

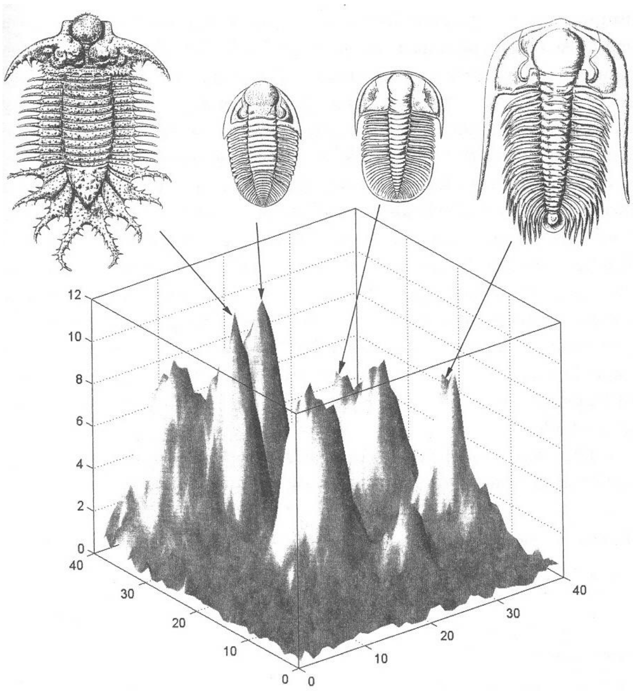


Figure 14: Imaginary rugged fitness landscape, showing optimal shape configurations of fossil tribolites.

Image reproduced from (Solé and Goodwin, 2000, pg. 257)

and fitness assessed in terms of shape. In any one set of conditions, there are some optimal configurations for the organism, represented by the peaks in the fitness landscape. However, the extent to which this has any real meaning in nature is doubtful. Even Wright (1932, pg. 161-163) realised that such a representation hugely over-simplified the case:

> "... *accurately representing the population genetics of the evolutionary process requires thousands of dimensions. This is because the field of possible gene combinations in the field of gene frequencies of a population is vast.* ... Wright used the two dimensional graphical depiction of an adaptive landscape ... as a way of intuitively conveying what can only be realistically represented in thousands of dimensions. The surface of the landscape is typically understood as representing the joint gene frequencies of all genes in a population graded for adaptive value." (Wright's words italicised by me) (Skipper, 2002)

But Provine (1986, pg. 308–316) argues that Wright's original illustrations have no gradation along the axis or even any indication of what the units are (as they represent "genotype interpretation") and neither are there points along them to indicate where a gene combination is placed. Provine therefore claimed that there is no way of generating the continuous surface of an adaptive landscape. A second, more serious criticism by Gavrilets (1997), is that many gene combinations are incompatible, the number of which rises with the number of genes under consideration. Therefore the idea of representing gene combinations by a smooth continuous surface is itself specious; reality more closely resembles a landscape pock-marked with variously sized holes, where the holes indicate unachievable gene combinations. Others, notably Stadler (2002), have extended fitness landscapes with views similar to Gavrilets based on the impossibility of certain genotypes realising phenotypes due to developmental processes (in essence the same restrictions placed on evolution that were noted in §5.1). As a result of these issues, most biologists have abandoned Wright's original genotype interpretation in favour of one that assumes a population-based interpretation: "joint frequencies of all genes in a population graded for adaptive value" (Skipper, 2002). But even if one takes this model as realistic, the problems are not over for fitness landscapes.

6.2 The NK Model

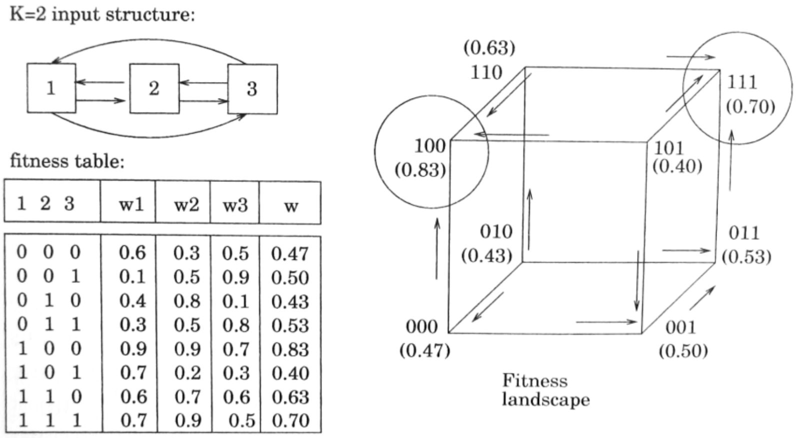
The difficulties with the genotype interpretation led to the development in the late 1980s of a fitness landscape that tried to tackle the interdependency between genes, known as *epistasis*. In Kauffman and Levin's well-known NK model (Kauffman and Levin, 1987), N represents the number of genes (and therefore the dimension), while K indicates how many other genes influence any given gene, i.e. the K other genes are epistatic inputs to the fitness of the considered gene. If $K = 0$, so that no gene influences any other gene, then it results in a fitness landscape of only one peak with smooth sides (known as the Fujiyama landscape). But as K increases, the number of peaks on the landscape increases and the mean fitness of the nearest peak decreases toward that of an entirely random genotype (Skipper, 2002). Typically, interconnection results in a rugged fitness landscape. Genes (or traits) are represented as binary alleles, so that they are either expressed (1), or not (0). For computer scientists, the model starts to sound familiar:

> "the 2 to the N combinations of alleles of the N genes are therefore located on the vertices of the N-dimensional Boolean hypercube. The fitness of each type of organism, or vertex, is written on that vertex and can be thought of as a height. Hence the NK model creates a fitness landscape over the N-dimensional Boolean hypercube" (Kauffman, 2000, pg. 198).

The NK model and its statistical properties (i.e. the effects of changing values of N and K) have been widely explored (Solé and Goodwin, 2000, pg. 258).

As before, a species evolves by "adaptive walks". Essentially this means that we can choose a given trait, mutate the bit and then examine the fitness table. If the average fitness of the resulting configuration is higher, an adaptive walk has occurred and the species moves in the landscape (it starts to climb). As already mentioned, when $K = 0$ the system is disconnected and there is a single global optimum. But when K is the theoretical maximum, $N - 1$, the system is entirely interconnected and every gene influences every other gene. Kauffman explored the generic properties of interconnected landscapes by assigning random fitness values across each of the allele states affecting a given gene (i.e. alleles of other genes whose expression affects the gene you're looking at). The fitness value of a specific allele at each of the N genes is then the average of the fitness contributions of the other N genes, yielding a random fitness landscape over the N dimensional hypercube (see Fig. 15). These random, highly interconnected landscapes yield interesting properties:

> "A main feature of random landscapes is that there are nearly exponentially many local peaks, indeed the number of local peaks is 2 to the



1 2 3	w1	w2	w3	w
0 0 0	0.6	0.3	0.5	0.47
0 0 1	0.1	0.5	0.9	0.50
0 1 0	0.4	0.8	0.1	0.43
0 1 1	0.3	0.5	0.8	0.53
1 0 0	0.9	0.9	0.7	0.83
1 0 1	0.7	0.2	0.3	0.40
1 1 0	0.6	0.7	0.6	0.63
1 1 1	0.7	0.9	0.5	0.70

Figure 15: Building up fitness landscape. Each gene receives inputs from two other genes ($K = 2$) that affect the fitness contribution of the gene. Each gene in each of the $23 = 8$ possible genomes is randomly assigned a fitness contribution between 0 and 1. The fitness value of each genome is then computed as the mean value of the fitness contributions of the three genes. A fitness landscape is constructed as a boolean hypercube. Circled vertices on the cube represent local optima, arrows represent "uphill" directions *(text from Kauffman (2000, pg. 199)).*

Image reproduced from Solé and Goodwin (2000, pg. 258)

> $N/(N+1)$. For $N = 1000$, there are 10^{297} local peaks on the landscape. Finding the global peak by hill climbing is improbable, and the system becomes trapped on a local peak. Other features include the lengths of walks via fitter neighbours to nearby peaks, which scales as the logarithm of N, and the way directions uphill dwindle on walks uphill. At each step uphill, the fraction of directions uphill is cut in half, yielding exponential slowing in the rate of finding fitter variants." (Kauffman, 2000, pg. 200)

These features of highly interconnected, rugged landscapes are crucial to understanding the nature of a genotype search space. Because the landscape is big and interconnected, finding a global optimum becomes not just improbable, but of dubious value even as a strategy. The odds are stacked against the organism. Furthermore, the fact that the rate of improving fitness slows exponentially with each uphill step, and that the system gets trapped on local optima, correlates to the earlier suggestion by Wolpert that evolution merely "tinkers" with existing structures. Big jumps are not possible. The reason you are forced to tinker with the edges, making only incremental movements in any direction, is because each step uphill seriously restricts the other directions you can move in. If an organism wants to stay flexible in a dynamic environment, it can't afford to get trapped on a local peak of specialist perfection having abandoned its options for adaptive movement. Unfortunately organisms have no way of knowing whether their adaptive movements may strand them or keep them in the race of poor, but flexible competitors.

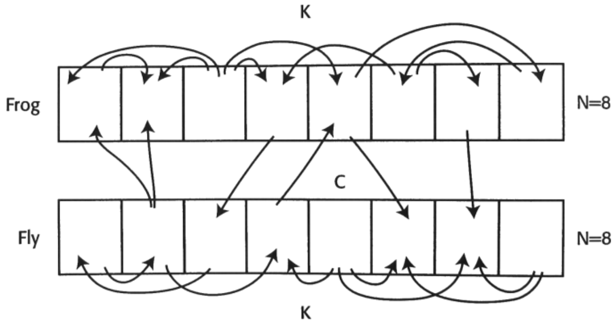


Figure 16: Interconnected genes affecting two coevolving species where K represents epistatic influence, C represents the degree of epistatic coupling between species.

Image reproduced from Kauffman (2000, pg. 200)

6.3 Coevolution

Natural systems are much more complicated than single genotype populations. In a Malthusian world of limited resources, everything is fighting for survival. For cheetahs to stay alive and reproduce, the species must keep up with the gazelles who are constantly evolving to outrun the fastest cheetah. Organisms not only affect each other's environment, they compete in an evolutionary race against other species. Losing means extinction. The idea of organisms competing merely to "stay in the race" was first put forward by Van Valen (1973) and is known as the *Red Queen hypothesis* [10]

Solé and Goodwin (2000) explain the hypothesis thus: based on the fossil record, Van Valen observed that a species may become extinct at any time, regardless of how long it had previously existed. But if evolution is a process of constant improvement, why are modern species as equally likely to disappear as their ancestors were? Van Valen's hypothesis suggests if that continual improvement were the case, we would expect to see a decreasing probability of extinction the longer a species had existed. Instead, the fossil record shows the probability remains constant. That constant probability can only mean that continuous improvement is not possible for any species.[11] Van Valen claims that this means species are compelled instead to continuously adapt to each other's changes. Rather than continuous improvement, we have continuous re-adjustment. And despite natural selection doing its best to improve your chances of genetic survival, you might find you can no longer hill-climb as well as you could because someone else is affecting your ability to do that. Thus you can drop out of the race at any moment, and according to Van Valen, that would probably be the moment you failed to adapt to someone else's advantageous change. So we have yet another route to extinction. But this time, rather than getting trapped on a local peak due to your own adaptive movements, you get trapped because *the landscape moves faster than you do*.

Modelling adaptive landscapes takes on a whole new level of complexity when competing species are able to affect the fitness criteria of the genotype search space.

[10] The name of the hypothesis comes from the Red Queen in Lewis Carroll's *Alice Through the Looking Glass*, in which she explains to Alice "Here, you see, it takes all the running *you* can do, to keep in the same place" Solé and Goodwin (2000, pg. 254).

[11] As pointed out in the previous section, the evidence suggests that continuous improvement is actually dangerous to the continued existence of a species.

Kauffman uses a simple, fictitious model of a frog and fly in evolutionary competition (see Fig. 16)

> "Each of the N genes in the frog receives inputs from K genes in the frog and C genes in the fly, and vice-versa. Thus, the sticky tongue of the frog affects the fitness of the fly via the presence or absence in the fly of slippery feet, sticky stuff dissolver, or a strong sense of smell for sticky frog tongues . . . Now when the frog population moves by mutation and selection uphill on the frog landscape, those moves distort the fly's landscape and vice-versa. Coevolution is a game of coupled, deforming landscapes." (Kauffman, 2000, pg. 201)

The NKC model developed by Kauffman and Johnsen (1991) introduces the new parameter C to represent the coupling between species. Kauffman and Johnsen claim that these models "generally behave in two regimes: an ordered regime and a chaotic regime, separated by phase transition" (Kauffman, 2000, pg. 201). Solé and Goodwin describe these regimes as a) low-K, ordered or frozen, where species settle on local optima, and b) high-K, chaotic or Red Queen, where the ecosystem is in constant flux. They also comment that the system appears finely, if not critically balanced:

> "At the boundary between these regimes, species in a finite system reach local peaks, but any small perturbation generates a coevolutionary avalanche of changes through the system. The distribution of these avalanches follows a power law, as expected for a critical state." (Solé and Goodwin, 2000, pg. 259)

Such changes are usually interpreted as extinction events. Kauffman and Johnsen (1991) mapped these avalanches to extinction events in the fossil record and although initially unsuccessful in finding a correlation, once the model was adapted to allow connections between species themselves to co-evolve, a correlation was found (Kauffman, 1995). Kauffman concluded that as avalanches of extinction events can propagate across species, it appears that species survive by niching on local optima and thus protecting themselves against too much "evolutionary competition" from other species. This niching, in a highly coupled, adaptive landscape is akin to each species "tuning" the ruggedness of its landscape (i.e. managing its interconnectedness) so that it retains both a degree of independence from the adaptive movements of other species and the ability to make its own adaptive moves. Both Solé and Kauffman further claim that by tuning their own landscapes, species poise the entire system as close to the critical boundary as possible. That boundary line is *the edge of chaos* between the two system states, low-K and high-K, as described above. The knack of maintaining the system near that edge of chaos is termed "self-organised criticality".[12] This is an interesting property of interconnected evolutionary systems, and perhaps one that those in evolutionary computation should bear in mind as they attempt to scale their models.

6.4 Deforming Landscapes of Development

The NK and NKC models of adaptive landscapes are not the only theoretical models that use the metaphor of deforming landscapes. Waddington developed a model in the 1950s of an epigenetic landscape that uses a slightly different metaphor to explain stability during development. In Waddington's model, rather than adaptive walks over

[12] Per Bak (1996) was largely responsible for developing ideas around self-organised criticality.

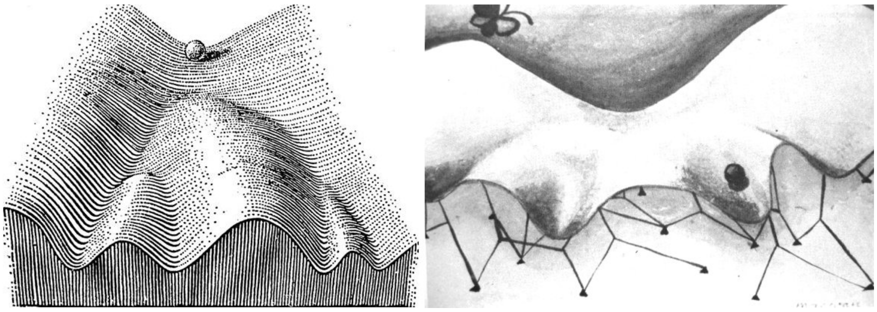


Figure 17: Waddington's epigenetic landscapes: his original drawing on the left showing developmental pathways (Waddington, 1957, pg. 11–58) and a later alternative showing epigenetic influences on the same landscape. Genes pull guy ropes attached to the landscape, deforming it and fixing the path of the ball, but a slight alteration in the genotype will not significantly change the final state, due to the stability (*homeostasis*) of development.

Right-hand image reproduced from Saunders (1993, pg. 200)

a fitness landscape, the image is one of a ball rolling down the hills and valleys of a landscape of potential developmental paths, as shown in Fig. 17. The hills and valleys are created by the competing influence of genes (the genes were later shown as pulling on guy ropes that are attached both to each other *and* to the surface of the landscape with different degrees of force). Waddington's model does not suffer the problems of some gene combinations being impossible, as we are looking at the developmental process. Instead unrealisable developmental paths (due to the viability of the organism) are represented by peaks in the landscape that deflect the path the ball can take. The smooth continuous surface is the result of gene expression during development, itself an interconnected system of guy ropes where many genes can affect the influence of a specific gene's tension on the landscape. Waddington was fascinated by the ability of the system to return to a stable state even after being perturbed by environmental or genetic effects, a property he termed *homeostasis*. He also used the word *canalization* to describe the property that development can typically proceed to one or more of a restricted number of alternative end states, rather than to a broad spectrum (Saunders, 1993). Waddington's point was that a system, especially a dynamic, non-linear system such as an organism, is unlikely to have stability in the traditional sense of a single point equilibrium. Waddington sought to emphasise that dynamic, nonlinear systems had a richer notion of stability, one closer to a *path* or *trajectory*, which could be returned to if travel along it was deflected at some point.

The importance of stability is crucial in determining the viability of the organism and although this acts as a restriction on evolutionary change, it also brings benefits. For example, it is through developmental stability that great genetic variation can be supported in a population of nearly identical phenotypes (Saunders, 1993). The model also has explanatory value when we want to view how large evolutionary changes might affect the developmental process. From Waddington's model, we can see that, even if the influence (i.e. the tension on the guy rope) of a gene is increased dramatically, change is mitigated by the opposing tensions from other genes' guy ropes that are attached to it. Thus for one change to have a large impact, it would have to affect many other genes with a similar degree of force, and make them somehow conspire to work together to allow a large alteration to affect the shape of the landscape. Large changes to the outcome of developmental processes are thus both difficult and unlikely by random mutation.

7 Concluding Remarks on Biological Evolution

The invention of nature appears almost limitless. Nevertheless, evolutionary adaptation is tightly constrained by natural selection and the viability of the organism throughout both its evolutionary and developmental history. The interplay of so many aspects in evolution makes the process difficult to model. From complex developmental processes that show little evidence of tampering with the early embryonic stages, to theoretical models showing the interplay of epigenetic forces through to evidence from the avalanches of extinction events in the fossil records, no single model can capture it all. The best we can hope for, it seems, is to take as much evidence as we can from natural and theoretical biology when hoping to understand evolution in its broadest sense, and investigate how the mechanics of the process tune and govern themselves.

If there were a single criticism of the field of evolutionary computing, it would be that too little evidence from biology has been used during adoption of the evolutionary paradigm. While no one would wish to try to replicate the intricacy of biology, evolutionary computation has historically taken a very narrow interpretation of evolution, one that is predominantly based on models of fitness landscapes. These models are highly abstract and perhaps rather sterile as a consequence. There is little opportunity for the rich, complex interactions we see in real biological processes to take place. For example, hardly any work has included the role developmental processes play in evolutionary search, despite evidence from biology suggesting its fundamental importance.

In the following sections we briefly examine the history of evolutionary computation, taking in the major trends in the field and examining where the current research effort is focused. We see how some of the early successes of evolutionary computation were in part responsible for the direction of later research. We look at interesting results from research into the evolution of logic circuits in hardware, which provide evidence that "richness" is a quality we need more of in our models and our approach to evolution, and end by considering a few developmental approach.

8 Evolutionary Computation

Towards the end of the twentieth century, biotechnology increasingly made headline news. There was a growing awareness, even hysteria, about the extent to which genes determined many aspects of our lives. Genes were discovered, it was disturbingly claimed, for homosexuality, schizophrenia, even criminality (Hutcheon, 1996). Other spinoffs from biotechnology, such DNA "fingerprinting" have become commonplace and large scale. Publicly funded research such as the Human Genome Project[13] kept biotechnology in the public eye. The famous double helix even features on a British sterling two pound coin minted specially for the fiftieth anniversary of the discovery of DNA.

Biotechnology's rise to fame and its increasing influence on people's lives is perhaps matched only by that of computer science over the same period. As biologists

[13] http://www.genome.gov/ and http://www.ornl.gov/sci/techresources/Human_Genome/home.shtml

have become increasingly reliant on computers, it was inevitable that the latest findings and theories in biology would feed back into computer science research. Just a decade after Crick and Watson's discovery of the structure of DNA, computer scientists were already investigating what ideas could be taken from biology to use in their own field (Fogel, 1963). But although the borrowing has been somewhat piecemeal to date, the trend to adopt ideas from biology shows signs of becoming even stronger in the coming years. For some, such as Peter Bentley writing the New Scientist in 2004, the future of computer science will be inextricably linked to paradigms of biological processes:

> "You could say we are going back to nature. I am convinced that in the future, software will evolve and grow instead of being designed and built. In place of programmers there will be digital horticulturalists who plant, prune and grow software from seeds that they have cultured. Not a single line of code will ever be typed into a computer again." (Bentley, 2004)

Whatever the truth to such grandiose claims, links between computer science and biology are gaining strength. One could point to areas such as neural networks and immune systems as examples of this, but perhaps the greatest recipient of ideas in computer science is the area covered by evolutionary computation.

8.1 A Brief History of Evolutionary Computation

The earliest attempts at simulating evolution were linked to machine learning. Turing (1950) suggested how an evolutionary or genetic search might be used in general machine learning, while Friedman (1956) speculated on the use of feedback, selection and mutation to design "thinking machines" (an idea which has raised its head again recently, with the work of Bongard and Lipson (2004)). There were others too, perhaps less closely tied to academia, such Friedburg (1958) (who suggested "a population-based hill climbing search") and Box (1957). Box's work is interesting in that it dates the involvement of industrial systems control engineering to the earliest days of evolutionary computation. While never purely an engineering discipline, the practical side of evolutionary computation has remained influential and been partly responsible for the direction and narrowness of later research (see §9.1). However, despite these early pioneers, evolutionary computation as a field didn't really develop until the early 1960s, when several branches appeared independently.

8.1.1 Evolutionary Programming

Evolutionary programming was part of the attempt to create artificial intelligence. L. Fogel (1962) used finite state machines and simulated evolution on a population of contending algorithms to demonstrate intelligent behaviour. The machine in question had to predict an input symbol and its prediction was an output based on previous experienced input symbols. Fogel exposed a population of machines to the learning environment of input symbols and selected machines on fitness criteria. The selected machines were then randomly mutated and the process repeated with their offspring.

It is worth noting that Fogel presented his work initially in an industrial research journal and it would be fair to say received mixed reviews. While some were positive, others such as Solomonoff (1966) were critical of the inefficiency of random, hill-climbing searches. But the criticisms stemmed from comparisons with other artificial intelligence research, rather than an objective look at the potential of the method for its own sake. More recent commentary on the early work in automated programming by Lenat (1983) is equally critical, but perhaps unfairly so given the considerable benefit of hindsight:

> ... early (1958–1970) researchers in automatic programming were confident that they could succeed by having programs randomly mutate into desired new ones. This hypothesis was simple, elegant, aesthetic and incorrect. The amount of time necessary to synthesise or modify a program was seen to increase exponentially with its length. Switching to a higher level language ... merely chipped away somewhat at the exponent, without muffling the combinatorial nature of the process. All the attempts to get programs to "evolve" failed miserably, casualties of the combinatorial explosion. (cited in Fogel, 2000, pg. 67–68)

Fogel argues that claims about the amount of time to evolve a solution increasing exponentially with its length are unsubstantiated in the literature. However, the "combinatorial explosion" of the total search space is a well-documented issue with other branches of evolutionary computation, particularly genetic programming, but also more generally when the representation length is increased or given greater complexity. Practitioners still advocate throwing more computing power at this problem, much as they did in the 1960s (discussed in § 9).

8.1.2 Evolutionary Strategies

Evolutionary strategies developed as general function optimisation algorithms to solve difficult real-valued parameter optimisation problems. The work was started in the mid-1960s at the Technical University of Berlin by Rechenburg (1963; 1964; 1973) and Schwefel (1975; 1977; 1981). Evolutionary strategies had some noteworthy features. For example, "the components of a trial solution are viewed as behavioural traits of an individual, not as genes along a chromosome" (Fogel, 2000, pg. 69). Although a genetic source for phenotypic traits is assumed, the nature of that linkage is not made explicit. The genetic transformations result in behaviour changes that follow a Gaussian distribution, allowing many phenotypic characteristics to change following a genetic alteration.

Another interesting feature was the self-adapting strategy parameters, enabling the degree of mutation of a parent to change dynamically and for the parameter to be mutated and undergo evolution itself. This work bears comparison with the more recent dynamic parameter encoding in genetic algorithms (Schraudolph and Belew, 1992). Fogel (2000, pg. 72) claims that "strong similarities exist between evolution strategies and evolutionary programming ... In many cases, the procedures are virtually equivalent even though they developed independently". More recent work on evolutionary strategies can be found in Voigt *et al* (1996).

8.1.3 Genetic Programming

Genetic programming was extensively developed by Koza as a means to automate programming, but some of its greatest successes have been in the field of machine generated analogue circuit designs (Fonlupt, 2005; Koza, 1992, 1994; Koza et al., 1999, 2003). An individual in genetic programming is a computer program rather than a chromosome. Each program is evaluated by being run and a fitness is then assigned to it (although this may be over multiple runs with different inputs).

Programs themselves are usually represented as parse tree structures, with subtree nodes acting as the points on which mutation or recombination occurs. For example, in recombination, two different subtrees in the same node position might be swapped between parent trees, or if using a mutation operator, a node might be selected and the subtree replaced by a randomly generated subtree. Other variants of mutation exist.

Like other branches of evolutionary computation, genetic programming has been successful in industrial applications (Koza et al., 2004), particularly with respect to human-competitive solutions in analogue electrical circuit design, some of which have been patented (Streeter et al., 2003).

However, a big problem with genetic programming has historically been the issue of "bloat". Operators can grow large structures that have no effect or that are wasteful. This feature of genetic programming has been likened to "junk DNA", in that while it doesn't alter the semantics of the program it represents junk code that is either unused or wasteful. There are many papers suggesting ways to tackle bloat (Brameier and Banzhaf, 2003; Fernández et al., 2004; Langdon, 2000; Langdon and Banzhaf, 2000; Langdon and Poli, 1997; Tomassini et al., 2004; Vanneschi, 2004). Some good papers theorising about the shapes of parse trees and the causes of bloat have been recently been published by Diada *et al* (2004; 2003; 2005), in particular an analysis of the visual form of evolved tree structures which has lead to the hypothesis that evolved trees are inherently "deep and narrow rather than wide" due to the numbers of nodes on deep subtrees leading to them being more likely to be selected.

8.1.4 Genetic Algorithms

Genetic algorithms (GA) were largely developed by John Holland and his students from the 1960s onwards (1962; 1992),[14] with theoretical work being added by Goldberg (1989; 2002) and Vose (1999a; 1999b). Holland's original motivation was to "understand the principles of adaptive systems" (Dimutrescu et al., 2000, pg. 7) and in common with other branches of evolutionary computation, the early papers presented the process of evolution in a highly abstract form, so that key elements of the simplified process could be identified and understood in terms of what made the process effective as a search algorithm.

GAs are generally comprised of a population of candidate solutions encoded as chromosomes in a binary string representation.[15] The process is simple: take the best members from the candidate population of solutions and use those to form your next generation of solutions by combining them with randomly chosen individuals or each other. Assess your new population and repeat. Natural selection ensures successive generations move the population towards your fitness objectives. The canonical form of GAs is given in the following section.

Exactly how each successive generation should be formed soon became a major topic of debate, with many forms of crossover between individuals being tried. A large part of the debate focused on improving the performance of GAs. Running a simulation over many generations containing large populations was computationally expensive and early workers in the field frequently struggled with limited computing power (Fogel, 1998). This, combined with the pressure to achieve practical results on engineering problems, meant it became the dominant area of research.

[14] Fogel (2000) also cites Bremermann (1962; 1966) and Fraser (1957; 1968) as among the early developers.

[15] Binary representation has declined in recent years, but the canonical form is generally given as a fixed length binary string representation. See § 8.2.

The GA is considered to be the main paradigm of evolutionary computation (Dimutrescu et al., 2000, pg. 21) and as so much work has been done on aspects of the model, we will look in greater detail at its main components and theories relating to their influence in § 8.2.

8.1.5 Evolutionary Computation

The previous four sections have outlined the historical branches of evolutionary computation. Although each of the variants presents slightly different models of evolution, the representation of individuals and the mechanism of exploring the population search space, the differences are not sufficient to consider any of the variants unique. Fogel (2000) notes that since 1993 and the formation of the journal *Evolutionary Computation*, "evolutionary computation" (or "evolutionary computing") has become an accepted umbrella term for all the variants. Fogel even doubts whether any value can continue to be gained by using specialist terminology from one of the branches:

> "It is no longer possible to identify a particular effort in evolutionary computation as a genetic algorithm, an evolution strategy, or an evolutionary program, simply by examining the representation chosen, the selection method, the use of self-adaptation, re-combination or any other factor. In fact, the practical utility of each of these terms has evolved to be essentially useless: Little or no information is conveyed by identifying a particular effort as a genetic algorithm, evolution strategy or evolutionary program." (Fogel, 2000, pg. 85)

In agreement with this sentiment and notwithstanding the historical importance of the variant branches, the term *evolutionary computation* will be used for the remainder of this section.

8.2 The Canonical Genetic Algorithm

The basic framework of evolutionary computation is one based on population convergence over optimal peaks in a fitness landscape of the genotype's population. In the breeding of successive generations, selection occurs according to individuals being assessed against some fitness criteria, thus some of their "good" genes are carried over into the next generation. The canonical process uses fixed-length binary strings to represent chromosomes. In terms of actual algorithms, the genetic operators are procedures that modify the individuals represented as chromosomes by mutation (or inversion) or by combining them (crossover). As individuals usually map to a represented solution, it is common in evolutionary computation to refer to the population as containing *candidate solutions*.

The canonical or simple GA is as follows (from Dimutrescu et al. (2000, pg. 29–31)), where t means time step:

1. Set $t = 0$
2. Initialise chromosome population $P(t)$.
3. Evaluate $P(t)$ using fitness criteria.
4. **while** termination condition not satisfied **do**
   begin

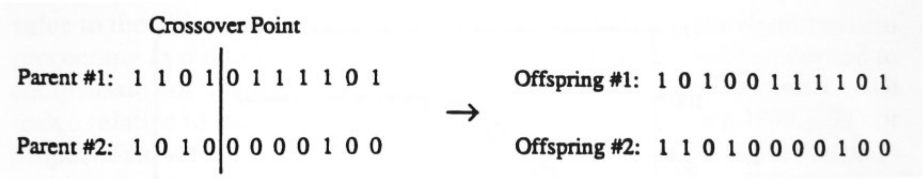
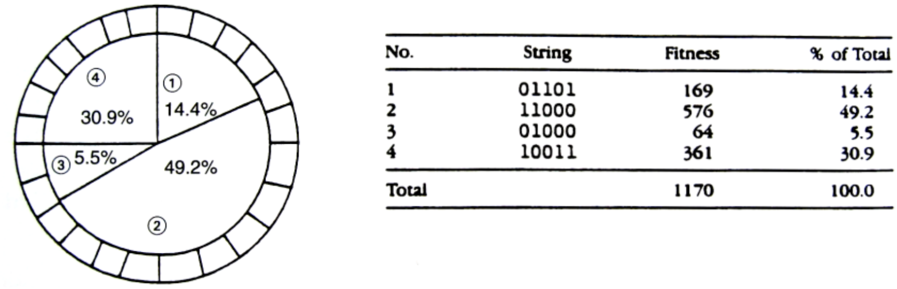

Crossover Point
Parent #1: 1 1 0 1 0 1 1 1 1 0 1
Parent #2: 1 0 1 0 0 0 0 0 1 0 0
→
Offspring #1: 1 0 1 0 0 1 1 1 1 0 1
Offspring #2: 1 1 0 1 0 0 0 0 1 0 0

Figure 18: The one-point crossover operator applied to two parents.
Image and text reproduced from (Fogel, 2000, pg. 75)



No.	String	Fitness	% of Total
1	01101	169	14.4
2	11000	576	49.2
3	01000	64	5.5
4	10011	361	30.9
Total		1170	100.0

Figure 19: Roulette wheel selection.
Image and text reproduced from (Fogel, 2000, pg. 74)

```
    (a) Select best individuals from $P(t)$. Let $P_1(t)$ be the set of selected chromosomes. Choose individuals from $P_1(t)$ to enter mating pool ($MP$).
    (b) Recombine chromosomes in $MP$ forming populations $P_2$. Mutate chromosomes in $P_2$ forming $P_3$.
    (c) Select replacements from $P_3$ and $P(t)$ forming $P(t+1)$.
    (d) Set $t = t + 1$
end
```

This simple process provides certain key elements to the model that have been investigated in great detail. For example, there are many forms of crossover. Holland created a straightforward crossover between two parent chromosomes to get two offspring by "selecting a random position along the coding and splicing the section that appears before the selected position in the first string with the section that appears after the selected position in the second string, and vice versa" (see Fogel (2000, pg. 74) and Fig. 18). Other types exist that use multiple crossover points.

Generally each chromosome is assigned a probability of reproduction so that its chances of being selected are proportional to its fitness. One method of doing this is the *roulette wheel*, which divides up the population such that all chromosomes receive a probability in relation to their fitness (Fig. 19). Another popular method is tournament selection, and there are several others.

The use of binary encoding for fixed-length chromosomes has been criticised since it was first proposed as a universal encoding by Holland (1975). Binary encoding has its historical roots in the introduction of the schema theorem and building blocks, also by Holland. Fogel (2000, pg. 75) states that "schemas allow a way of determining the usefulness of finding out fitness values for strings that match your schema, as a partial match should also mean a partial fitness." His example explains how using a wild card [*] in a schema where the evaluation of the string [0000] has some fitness, the schema would suggest that partial information is also received about the worth of sampling the variations in [****], [0***], [*0**], [0*0*], [*00*] and so on (Fogel, 2000, pg. 75). This characteristic is called *implicit parallelism* and indicates that a single sample can provide information with respect to many schemas. It is claimed that certain representations and problem spaces are more amenable to implicit parallelism in schema design, particularly those where individual genes are not epistatic (MacKay, 2003).

But although Holland claims to have proved maximum implicit parallelism (i.e. the effectiveness of using schemas) occurs when the encoding is binary (1975, pg. 71), others have found no practical advantage. Michalewicz (1992, pg. 82) finds that real-valued numerical optimisation problems are best encoded in floating-point representations (faster, more precise), and others had similar experiences after practical experimentation (Koza, 1989; Syswerda, 1991; Wright, 1991). Nowadays, binary representations are rarely used except when the representation can be easily mapped to a series of Boolean decisions or a bit mask. Whatever representation is chosen for the chromosome encoding, it would be well to remember that Fogel and Ghozeil (1997) "proved that there are essential equivalencies between any bijective representations, regardless of cardinality . . . Thus, no intrinsic advantage accrues to any particular representation" (cited in Fogel, 2000, pg. 76).

According to the *building block hypothesis* (Goldberg, 1989; Holland, 1975), genetic algorithms work by locating and maintaining "good" building blocks. Building blocks are defined as "low order, low defining-length schemata with above average fitness".[16] Good building blocks are joined to other building blocks to create sequences that are associated with above average fitness. The hypothesis rests on the assumption that combinations of good schemata are likely to result in higher fitness more quickly than could be achieved if every possible combination of bits in a string were tried. Goldberg states "instead of building high-performance strings by trying every conceivable combination, we construct better and better strings from the best partial solutions of past samplings" (Goldberg, 1989). The building block hypothesis has been criticised as having no theoretical basis (Wright et al., 2003) and experimental evidence has shown that single point crossover does not result in identifiably better solutions (Syswerda, 1989). Despite the uncertainty around the building block hypothesis, it is notable that there is no other well developed philosophy about *how* genetic algorithms work and such a theory (or practical understanding) is needed if evolutionary computation is ever to scale to tackling large, complex problems (see §9.2).

GAs have been applied to a wide variety of real-world tasks. As that experience has been gained, practitioners discovered there were issues that reduce the technique's attractiveness as a search-based optimisation algorithm. Premature convergence is a common problem that occurs when the population of chromosomes reaches a generation where crossover no longer provides offspring that are capable of out-performing their parents. Although one might suspect this is the natural fate of any hill-climbing search, premature convergence is peculiar in that the means to avoid it often seem landscape dependent. An example was the attempt to introduce *dynamic parameter encoding* (DPE) by Schraudolph and Belew (1992). The technique seems to offer promise on quadratic bowl shaped landscapes; however, it actually fares worse than a simple GA on multimodal type landscapes (such as Shekel's foxholes, see Fig. 20).

[16] Definition taken from Wikipedia: `http://en.wikipedia.org/wiki/Genetic_algorithm`.

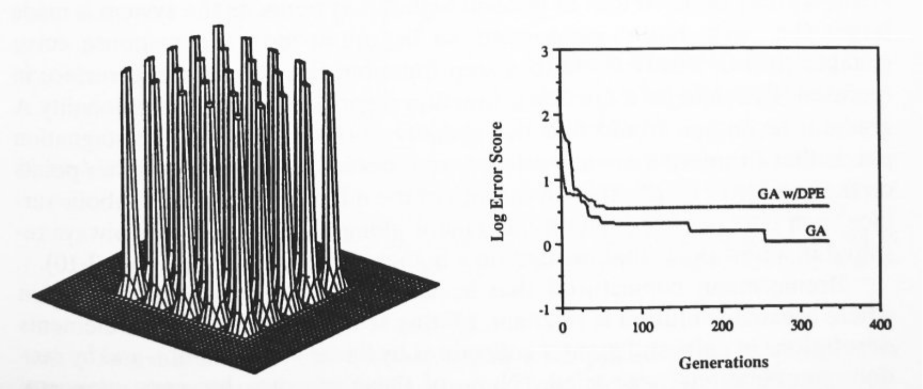


Figure 20: DPE performance on a testing landscape (inverted Shekel's foxholes). *Image and text reproduced from* (Fogel, 2000, pg. 79)

Further doubt was cast on the wisdom of pursuing solutions to premature convergence by coming up with landscape-specific algorithms. The "No free lunch theorem" by Wolpert and Macready (1995; 1997) states that averaged over all landscapes, no search algorithm performs better than any other. The same may be true for landscape-specific solutions to premature convergence. In addition to which, landscape-specific solutions require prior knowledge of the search space — something which it may be impossible to ascertain.

Despite these shortcomings, there is no doubting the popularity of GAs, either in academia or industry. They are now part of the standard toolbox of search algorithms where the search space is large and unpredictable, and have become the default method for tackling traditional, NP-hard problems, such as the travelling salesman. Unsurprisingly, their ability to tackle multiple objectives and to find optimal (or "good enough") solutions grouping those objectives finds many applications in control systems and other industrial applications.

However, the range of problems that evolutionary computation has tackled outside those relating to optimisation is not as wide as one might imagine. Perhaps a victim of its own (industry-based?) success, evolutionary computation and in particular, GAs, have themselves evolved little beyond their basic operational framework that was first described in the 1960s.

9 Weaknesses in EC models

No one has yet evolved a design for a car, a house, or anything that has a high number of parts, each of which can be exposed to evolutionary change. In the current model, as complexity grows, the length of the chromosome string representation grows, and large numbers of generations start to be required to reach good solutions. This increasingly hampers the effectiveness of an evolutionary search and during the 1980–1990s researchers attempted to address the issue. Initially, in an echo of the 1960s, researchers tackled the problem by simply throwing more computing power at it (Koza et al., 2003). Parallel GAs running on parallel machines were also tried (Cantu-Paz, 1998; Cantu-Paz and Goldberg, 1997). But the promised breakthrough hasn't happened. To date, no one has cracked the problem of scale when it comes to complexity, and evolutionary computation remains tied to addressing the same problems of multi-objective optimisation that it first started investigating over twenty five years ago.

9.1 Obsessed by Optimisation

By tracing the historical successes in evolutionary computation we can understand better the influence of those successes on the direction of subsequent research. Industrial success is often a good thing, but there is no doubt that evolutionary computation as a field has been heavily influenced by the need to fulfill its practical promise. Success in industrial applications, such as circuit design (Koza et al., 2004) and control systems (Robinson and McIlroy, 1995; Sharman et al., 1995), has meant that evolutionary computation was always being pushed towards making the evolutionary process more efficient, more practical. Such implementation concerns are not usually the domain of academia, but Chris Stephens being interviewed in 2003 for the EvoNet website, admits that research in the field is driven by those who want to use evolutionary computation for practical design problems:

> "Evolutionary computation, at least in terms of the fraction of papers dedicated to it, is mainly driven by the practitioners. … there is a big gap between the mathematical perspective and the engineering perspective." (Stephens, 2003)

While research programs can be forgiven for focusing on ways to improve the performance of genetic algorithms, that same focus has produced a rather blinkered view of evolutionary computation, one that sees nothing more in evolution than a set of optimising search algorithms. For example, David Fogel, in his introduction to *Evolutionary Computation: the fossil record* firmly states that "natural evolution is a population-based optimization process" (Fogel, 1998). Martin Keane, in a similar introductory chapter, describes evolutionary computation as "design search and optimization" (Keane, 2000). Both of these views stem from a practitioner's perspective, a perspective which has built up authority after the success GAs had in particular with multi-objective optimisation problems. So the last two decades have seen a continuous stream of papers published on the performance of GAs and optimisation, perhaps to the detriment of work that could have explored other features of the evolutionary process, such as greater exploration, better bootstrapping to deal with complexity, alternative mechanisms for encoding or problem representation and so on.[17] For example, a survey by Coello (2000) for the IEEE on multiple objective GAs (MOGAs) managed to list almost fifty separate applications and variants of MOGAs, and one wonders whether even industrial applicants would wish to wade through them all to find one appropriate to their needs (see Table. 1).

9.2 The Black Art of Decomposition

Despite the attention to optimisation issues, to say that nothing had been reported about other interesting aspects of the evolutionary process would be wrong. Indeed the ability of evolution to "invent" things was widely publicised in popular journals like *Scientific American* (Koza et al., 2003). As noted, human-competitive, even patented designs have been produced by evolutionary computation, and practitioners such as

[17] This remains the case. A count of papers submitted to EvoWorkshop 2003 and 2004 shows the majority (over 60%) in areas related to optimisation.

Multi-objective GA	**Year**
Schaffer's Vector Evaluated GA	1985
Syswerda & Palmucci GA with weighted sums	1991
Fonseca and Fleming propose MOGAs	1992
Wilson & MacLeod goal based GA	1993
Goldberg's Fast Messy GA	1993
Srinivas & Deb Nondominated sorting GA	1993
Horn, Nafliotis and Goldberg's Niched Pareto GA — co-operative sharing	1993
Coello Min-Max Optimisation — ideal non-Pareto feasibility vetting	1996
Priaux et al. GA-based approach with game theory	1997
Gary Lamont & David Van Veldhuizen's survey of MOGAs	2002
Tan, Khor, Lee &. Yang Tabu-based exploratory GA	2003

Table 1: List of multi-objective GAs (MOGAs), adapted and much reduced from Coello (2000).

David Goldberg claim that what was going on in these processes was more than mere optimisation:

> "… the design of effective GAs [is] ultimately helping us create first-order *computational models of innovation*." Goldberg (2002) (my italics)

However, a genuine computational model of innovation is something we are far from having. Innovation is hard to quantify or model in any sense, and working out *how* people (or GAs) invent things has proved equally difficult (although Thompson (2002) has some interesting comments on how evolution does this).

Goldberg, following on from Holland, believes that the knack of getting your problem effectively solved by GAs lies in the correct representation of the problem, and that representation itself relies on the problem being broken down in the correct "chunks", so that the right building blocks can be chosen. This may seem something of a black art to the uninitiated, an impression unlikely to be diminished by Goldberg's interesting, if unconvincing, description of the invention of human flight by the Wright Brothers in 1903. According to Goldberg, the Wright brother's success was down to how they decomposed the problem. While the evidence for this is sketchy and based on Goldberg's analysis of events, Goldberg nevertheless makes some interesting observations about the nature of invention and human design.

Goldberg demonstrates that human design isn't always a rational process of problem decomposition. When the problem domain is poorly understood, people will apparently try anything, no matter how deeply it may run against the grain of common sense. Fig. 21 gives an idea of just *how wild* human invention can be when unconstrained by design principles.[18] But it equally demonstrates just how dangerous a little knowledge can be, as an incomplete understanding of aerofoils led to people misapplying what little knowledge they had and trying designs that were doomed to fail. Goldberg described these early pioneers of aviation as appearing to "flail about in design space, hoping for good luck" (Goldberg, 2002, pg. 7).

[18] I can find no authoritative source for the photographs in Fig. 21, which appear on many websites. A good source for explanations about the machines can be found at http://www.ctie.monash.edu.au.

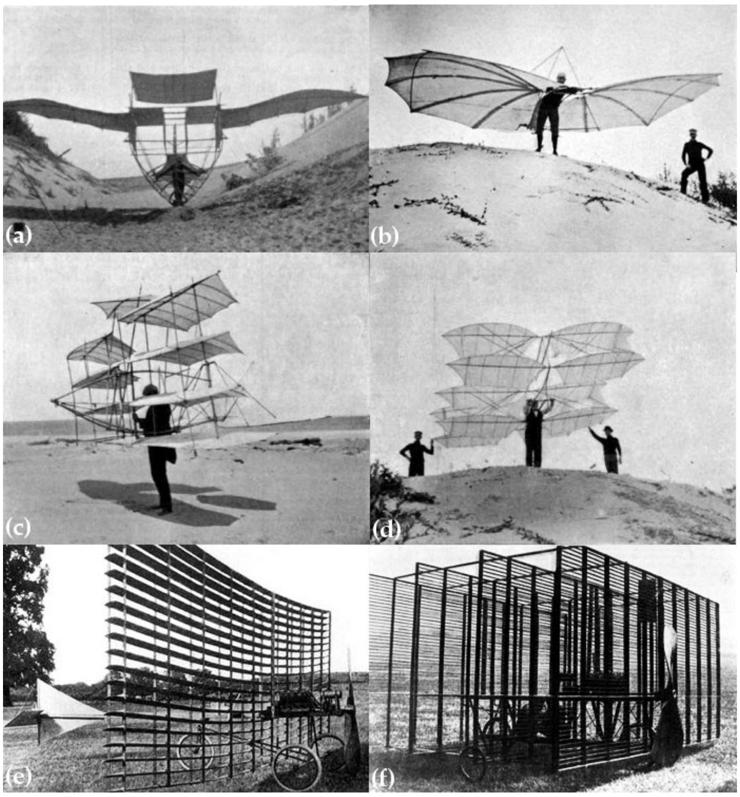


Figure 21: Bats and bikes as flying machines from early aviation pioneers: (a) Le Bris, The Albatross, 1868; (b)–(d) Adapted Lilienthal c.1890; (e) Phillips Multiplane of 1904; (f) Phillips Multiplane of 1907.

But even if Goldberg were correct, and the reason for the Wright brothers' success was the correct decomposition of their problem, there is still no convincing rationale behind *why* they broke it down the way the did — was it fluke, intuition, or did they use a set of rules that could be applied to unknown problem domains everywhere with the same degree of success? Goldberg claims the latter, but offers only a vague method, while the record of those aviation pioneers shows that while many tried, hardly any had success. One argument that supports finding the "lucky" decomposition of the problem is that *over a population*, the success rate of so many failures is consistent with an evolutionary process exploring a large search space. However, it is one thing for human engineers to have discovered "good building blocks" that could then be combined to solve a bigger problem, it quite another to suggest that evolutionary computation can do the same. The problem again comes back to scale.

Practitioners, such as Goldberg, advocate "careful" decomposition for complex problems: one should decompose the problem into small chunks, then run an evolutionary computation process over them. Leaving aside whether your problem decomposition is correct, for larger, more complex problems, decomposed solutions must "bolt back together" so that the whole thing works as a single solution. But although problem decomposition may be a typically human approach to finding a solution, there is no evidence that natural evolution tackles large scale problems this way. In fact the evidence is to the contrary. We have seen that epistatic fitness landscapes in co-evolution appear to be self-tuned close to the critical point of interconnectedness (see page 31), meaning that although species appear to be evolving in isolation, they are in fact responding to a wider ecosystem that cannot easily be broken into parts. Despite ideas about species niching on local optima, there is no evidence that natural evolution allows a chunk of the system to evolve in isolation, with the aim of making it fit into the wider whole at a later point. Such a proposal seems straight-forwardly counter-intuitive, but this is exactly what has been proposed as a way to scale evolutionary computation (Goldberg, 2002). However, work by Torrensen (2000) again reinforces the fact that evolutionary algorithms exploit the particular characteristics of their search space to find a solution, and this results in highly localised solutions for local problems.

Torrensen (2000) investigated issues of scalability and complexity while trying to evolve a signal filter over a range of inputs for a road image recognition exercise. Torrenson first broke the problem into a series of subproblems (each evolving a logic circuit on part of an FPGA (Field Programmable Gate Array)). For each subproblem, he then subdivided his inputs. Although the circuits evolved individually to high fitness, Torrensen found that when they were reassembled, the circuits failed to work due to noise from other inputs. This is a typical characteristic of evolved solutions in that evolution is "lazy"; it does the minimum possible to achieve satisfaction. It is also highly environment sensitive, solutions are not generally portable (see also Thompson (1997), discussed in more detail in §9.3). Torrensen's solution was to evolve his decomposed filters by exposing them to the full range of inputs. This worked when the decomposed elements were reassembled, but meant that the evolutionary process was now much slower for each subunit than before, resulting in a less than ideal solution to the problem of complexity and scale (see also investigations by Vassilev et al. (2000).

9.3 Towards Richer Invention

Humans "flailing about in the design space" of early aviation is an example of highly unconstrained invention, but it is not a realistic example of evolutionary invention. We know from evidence in evolutionary developmental biology, that evolution tends to tinker with successful structures rather than create entirely new structures out of the blue. A better example of evolutionary design by humans is the evolution of golf balls in the latter part of the twentieth century (Thompson, 2002). In contrast to early aviation, where a little knowledge led to many misguided designs, the evolution of golf balls was carried out in ignorance of why the changes led to improvements.

The earliest balls were called featheries and were made of hide case densely packed with feathers. Around 1850, a new type of ball appeared, a guttie, of solid *gutta percha* (a sort of rubber). They were cheap and smooth, but didn't fly as far as the older featheries. Gradually it was noticed that used featheries travelled further than brand new ones, so people experimented adding nicks and cuts to make their balls fly further. Over time, the manufacturers started to produce balls that had similar textures on them. Modern balls prefer a variety of dimples. However, it is only recently that the aerodynamics causing a rough ball to travel further have been understood. Two things stand out in this example of design evolution; i) changes were made at random and in ignorance of why they were good changes to make, ii) the changes were incremental.

Thompson wanted to see if a blind, incremental evolutionary search could still be encouraged to generate truly innovative designs. In 1996, he set up a ground-breaking experiment designed to promote design innovation through the relaxation of constraints (Thompson, 1996). His experiment evolved a circuit to distinguish between two frequencies on an FPGA, at the lowest level of abstraction possible — that of the physical behaviour of the platform. A 10x10 area of Xilinx 6126 bitstream was evolved (i.e. all the bits in this area were evolved directly as chromosome bits in a GA). The evolved circuit had to discriminate between 1kHz and 10kHz bursts of signals, no other input was given. Even the clock on the chip was turned off (Gordon and Bentley, 2002).

The experiment was a success. However, when Thompson tried to copy the evolved circuit onto another FPGA chip, he found that the circuit wasn't portable. He then tried to move the circuit onto another part of the original chip used in the experiment. Again the circuit failed to work. Thompson discovered that the circuit made use the *physical properties* of the silicon on the FPGA chip. It was extremely sensitive to any alteration in its environment — temperature, electricity supply, even the silicon of the chip — a change in any of them could stop the circuit working. Again, the result demonstrates the laziness of evolution.

Thompson's circuits lacked both robustness and portability.[19] But there was another feature of the evolved circuit that caused puzzlement. A lengthy analysis of the circuit concluded that its functionality was "bizarre" and some parts of it are still not understood (Thompson, 1997; Thompson et al., 1999). By utilising physical characteristics of the platform, the algorithm made some very unusual and complex circuitry. Commentators on Thompson's work have suggested that the reason for the innovative nature of his evolved circuit was not because evolution had searched a bigger design space than human designers, but that evolution had navigated through that search space differently (Gordon and Bentley, 2002, pg. 14). It seems inescapable that the reason for the search trajectory is due in part to evolution making use of the physical properties of its environment.

Miller and Downing (2002) have investigated what it would mean for complexity and innovation if evolved solutions were given a free reign to make use of the physics embedded in a rich medium. Rather than being surprised at evolution exploiting the complex physical properties of silicon, Miller states we should be impressed that it was able to do anything at all, given that silicon as a material was chosen expressly for its stability in electronics. He suggests the time is ripe to abandon media traditionally chosen for its physical stability and even the conventional components of electronic circuit boards: "... artificial intrinsic evolution may be best attempted in physical substrates that are rich and complex, rather than conventional transistor based technology" (Miller and Downing, 2002). To this end, Miller and Harding recently investigated evolving robot controllers in media such as liquid crystal (Harding and Miller, 2003).

9.4 The Gap Between Genotype and Phenotype

Environments select for fitness based on the capabilities of the phenotype. In the natural world, the viability of the phenotype not only acts as a brake on the random mutation of genetic material, it also acts to link evolutionary search to the process of construction. In most models of evolutionary computation, the development of the phenotype prior to selection is conspicuously absent. But as greater importance began to be attached to developmental processes in biology, so researchers in evolutionary computation began to question why it was missing from their models (Shipman et al., 2000). The standard representation for genetic algorithms, for example, is that shown in Fig. 22 (Lewontin, 1974). While such diagrams show the mappings between genotype and phenotype populations, and even some cursory epistasis, they stem from the period of

[19] Thompson later evolved more robust circuits by varying the environmental conditions.

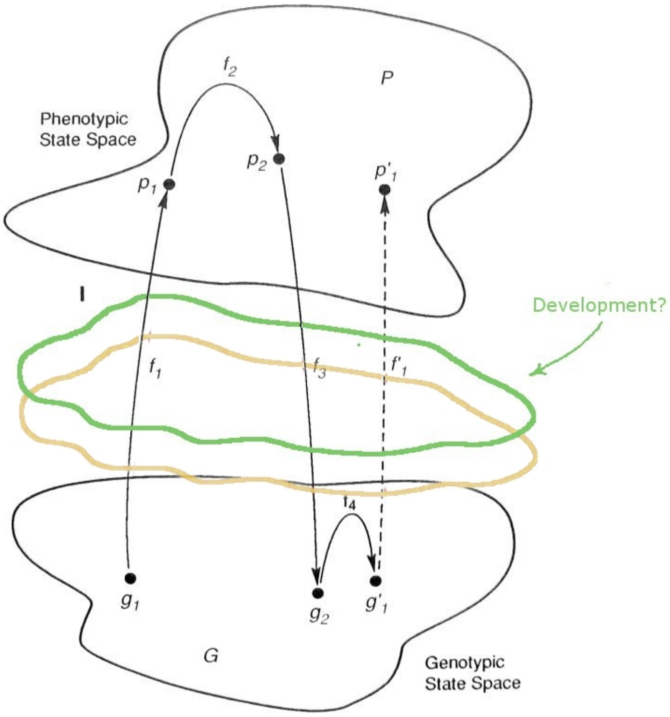


Figure 22: Lewontin's (1974) distinction between the two state spaces of genotype population space (informational/encoding) and phenotype population space (behavioural/performance). The middle spaces have been added to indicate the missing stages of development in the model.

Image and text reproduced from (Fogel, 2000, pg. 33)

neo-Darwinism that saw genes as the source of all phenotypic features and behaviours, and the role of development was credited with less influence than it has today. But these diagrams still constitute the majority of models employed in evolutionary computation. Genotype to phenotype mapping remains insignificant. The developmental process of the organism interacting with its environment and the restrictions imposed on evolution by development — such as the viability of the organism and the dependence of complex features on phylogenic predecessors — is ignored. But if these things help nature handle the combinatorial explosion of complexity when all parts of the system are potentially exposed to mutation and selection, then why wouldn't they also be of advantage to evolutionary computation?

Genetic evidence from evolutionary developmental biology has provided us with evidence of why evolution has to tinker with existing structures. Embryology has given us clues about the hierarchical nature of developmental structures and how evolution is constrained to act at the later stages of development. Evolutionary computation has yet to take account of such evidence. Instead, hypotheses such as building blocks and schema theorems that have no basis in biology have been allowed to dominate the research agenda, while failing to tackle either complexity or scalability. Thompson's work opened researchers' eyes to a whole new world of search potential. To explore it we need to allow our search algorithms to exploit physical resources. If we want to evolve solutions beyond human design space, we must move evolutionary computation out of sterile software abstractions into a much richer environment. However, providing access to richer resources does not guarantee they will be used. We know that biological development feeds back information about the search environment to the genome. The genome in its turn, dictates how to respond to that environment. To attempt a similar exploratory system in evolutionary computation, we need a mechanism capable of dynamic gene expression that can control the developmental process.

9.5 Models of Development

Development in evolutionary computation is a still nascent subject area. Other than an interesting collection of essays edited by Kumar and Bentley (2003) and some isolated submissions in the field of evolvable hardware (Gordon and Bentley, 2002; Lones, 2003; Tufte and Haddow, 2003), there has been little work done on modelling development within the evolutionary computation community.[20] Some early papers that are often cited, such as Fleischer and Barr (1994); Hogeweg (2000a,b) have presented models that, while impressive, have not been further developed by other authors. However, the field is growing and we highlight below some of the better known frameworks.

One relatively successful framework with a considerable body of research behind it is Lindenmayer, or L-system, grammars (Lindenmayer and Prusinkiewicz, 1989). The approach is capable of modelling the growth of plants and simple cell development. The use of generative grammars such as L-systems provides a means of modelling structure, and in particular, the growth of that structure. Structural elements represented in L-systems may not have or need a close mapping to the microscopic units that comprise real biological structures — in fact, successful models have been built using macro-level abstractions of plant parts, such as petals and leaves.

A powerful feature of L-systems is their brevity of expression. A relatively small rule set can generate surprisingly complex structures. Another is that generative grammars lend themselves to repetitive modular structures, so that structural elements such as branches or hair can be elegantly represented. One side effect of the "abstraction" of macro-level units is that some irregularity has to be introduced into the models so that forms acquire "roughness" (see Fig. 23). This can be done using context-sensitive or stochastic means. Some success has also been achieved using fractals for this purpose (Ferraro et al., 2005) and when rendered with turtle graphics, such models can be startlingly realistic in both behaviour and appearance (see Jacob (1999); Prusinkiewicz (2000) and Fig. 24).

For those looking at particular influences on development, feedback points can be introduced into L-system structures in conjunction with turtle graphics to produce models that respond to changes in their environment. Work by Měch and Prusninkiewicz (1996) has shown how feedback can be incorporated into open L-systems to show variance in the developmental outcome according to environmental conditions (see Fig. 25). Thus the potential to have open systems that develop in a natural way by interaction becomes a possibility.

To date, such systems have been implemented in software-based virtual environments (as opposed to embedded solutions using real sensor data). Jacob was one of the first to use evolutionary computation with L-systems to explore the evolution of plants and branching structures under light deprivation (Jacob, 1999). Following on

[20] Until recently there were few places to get such work published. For example GECCO, the main conference for evolutionary computation, featured its first track in development in 2007.

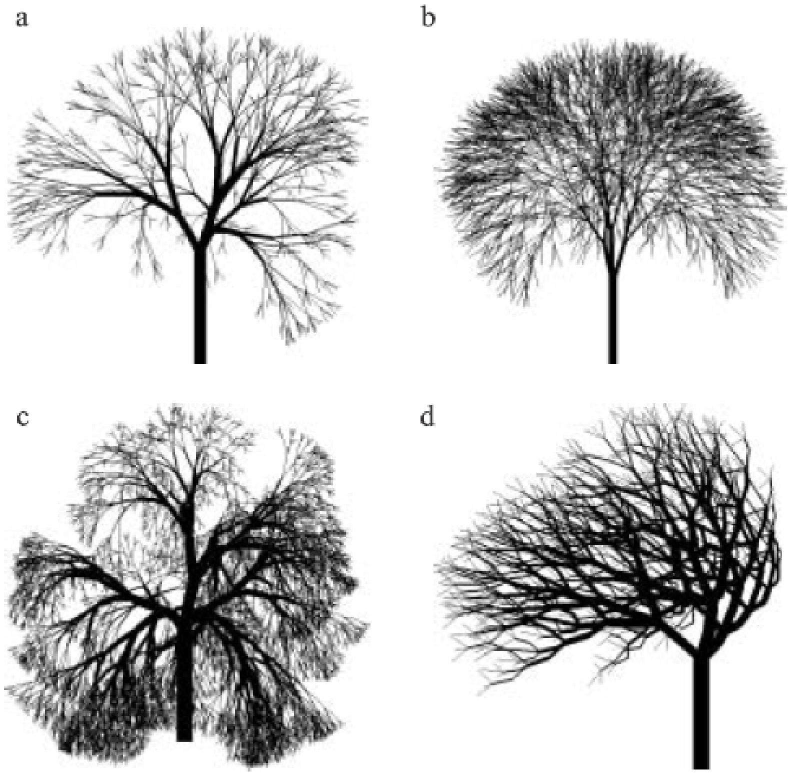


Figure 23: Some trees with different branching structures produced by L-systems. *Image reproduced from* Prusinkiewicz and Lindenmayer (1990, pg. 60, Fig. 2.8)

from this, work by Hornby has shown that evolution is able to quickly make use of the structural forms that L-systems can describe, and that the use of a generative encoding to produce such structures is advantageous in that "good" structures can be built more quickly (Hornby et al., 1999, 2001; Hornby and Pollack, 2001a,b). His work suggests that there may be a link between these sorts of highly compressed, generative descriptions and the complex, cascading control of genetic regulatory expression, and this would be interesting to explore. Hornby and Pollack have also evolved controllers based on L-systems (in conjunction with neural networks) to produce realistic gaits in simulated walking robots (Hornby and Pollack, 2002). The use of L-systems to model developmental processes is attractive to computer scientists as generative grammars are easy to represent and much of the work to render the structures graphically has already been done.

Developmental characteristics, such as canalization, have been viewed as a useful attribute for systems seeking fault recovery or robustness. Such approaches generally take the view of development as a robust construction process, rather than an adaptive control response to exploration. One example is Miller's French flag "multicellular organism" (Miller, 2004), which formed the starting point for several other pieces of work. Miller evolved two solutions demonstrating interesting capacities for self-repair based on cell growth (i.e. cell replication). The cell behaviour was based on chemical input bits and determined whether the cell would live, die or differentiate as it

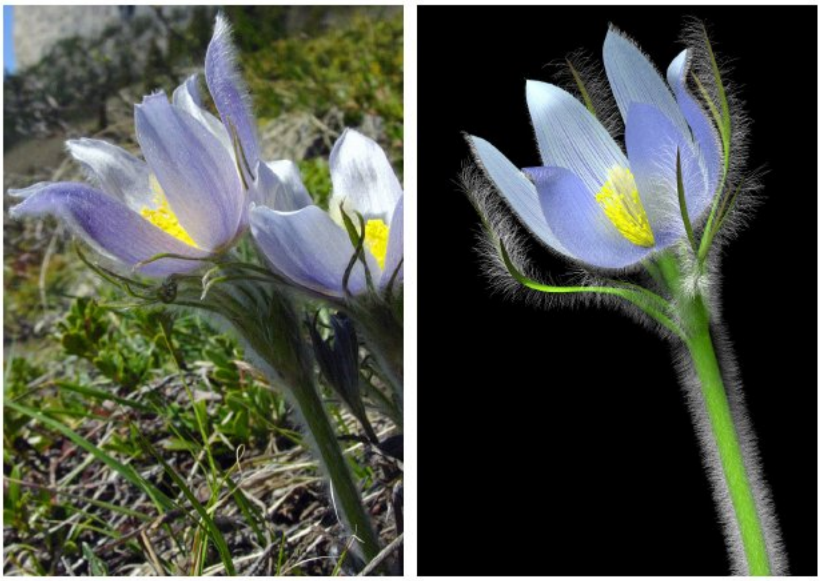


Figure 24: Photograph of wild crocus (left) and rendered image produced by a "hairy" L-system (right).

Image and text reproduced from Fuhrer et al. (2006)

grew in the Moore neighbourhood (the 8 cells surrounding a 2-D cell). The number of chemicals varied but all chemicals followed a diffusion rule as they spread to new neighbourhoods. Miller's choice of a flag as the task map was based on Wolpert's description of positional information in early embryo development (Wolpert, 1998) and his experiments demonstrated that pattern recovery, even after damage, was possible following a simple developmental model. In his models, the presence of more chemicals had a positive effect on fitness. Miller proposed to take the work forward and implement the system as a control mechanism giving cell growth a function, however to my knowledge this has not yet been done.

Pauline Haddow's group in Norway have also emphasised the importance of a developmental approach, initially using FPGAs (Haddow and Tufte, 2001) and examining the benefits of an extended genotype-phenotype mapping and redundancy / self-repair, and more recently extending Miller's work and looking at cell development in three dimensions (Haddow and Hoye, 2007). In the latter, they asked whether the presence of chemicals was helpful or a hindrance to the developmental process, but rather than having cell behaviour solely determined by chemicals, proteins requests are used. The chemicals form part of a precondition to protein production, and in contrast to Miller's results it was found that too many chemicals hindered phenotype fitness. Haddow's work is possibly the closest models we have to investigating development in the light of gene expression control. However, even here development is essentially modelled at the level of the single cell (albeit with some parallelism). No gene regulatory networks are modelled, nor are attempts made to see if intermediate fitness can be assigned to developmental stages. Such cascading regulatory control is common in natural systems, but it isn't clear what role evolution plays in all aspects of regulatory control. Like many areas of biological development, this could be a rich vein for evolutionary

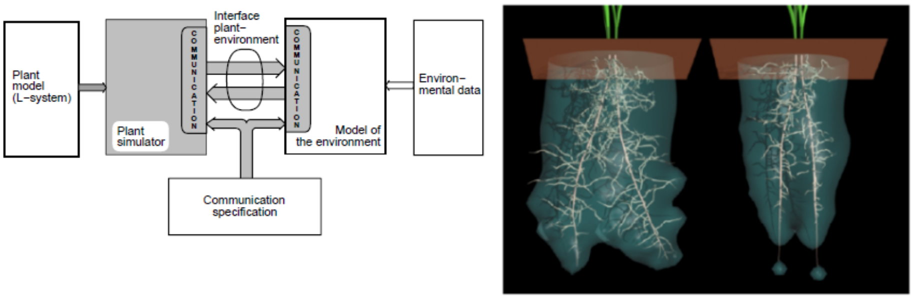


Figure 25: Model for an open L-system allowing environmental feedback, in this case modelling root exploration.

Image and text reproduced from Měch and Prusinkiewicz (1996)

computation to explore.

As a final note, a taxonomy has been proposed by Stanley and Mikkulainen (2003) for artificial developmental systems, or what the authors call "artificial embryogeny". Under their classification, models of development fall into two camps: L-systems (or similar generative grammars) and cell chemistry approaches. The latter is inspired by the early models byTuring (1952), who defined mathematical models of diffusion and reactions in physical substrates. Stanley and Mikkulainen claim that the division between grammatical systems and cell chemistry approaches is the easiest to make in what is admittedly a fledgling field. However, they go on to say that differences between the two are "largely superficial and [do] not reflect how phenotypes can develop" (page 106). Instead, their taxonomy draws up five dimensions of development, which help position an artificial developmental system in terms of what it is trying to achieve:

Cell Fate The eventual role of a cell during development;

Targeting Connections made by cells to target locations;

Heterchrony Timing and ordering of events in the phylogeny of an organism;

Canalization Stable development despite genetic perturbation;

Complexification The addition of new genes.

These dimensions are sliding scales with respect to nature. Stanley and Mikkulainen point out that being closer to natural systems isn't necessarily a measure of whether the system is "better", and that the dimensions instead inform you of the broad characteristics and capabilities of the system. For example, not being faithful to nature could give an artificial development model considerable advantage in terms of its computing speed. However, Stanley and Mikkulainen's dimensions contain some bias towards neural networks in particular and machine intelligence more generally. The dimension of *targeting* seems focused on the quality (or ability) of cells to form extensions such as dendrites and axons used in neural connections and nervous systems. But such qualities relate more to animal cytology than a general measurement of biological development. This criticism aside, their system of classification is at least a useful reminder of some of the qualities that artificial development models should emulate in order to get closer to nature and it contains a useful summary of work in this area.

10 Summary

The previous section has taken an overview of the field of evolutionary computation. Some omissions were necessary in both this overview and that of biological evolution and development. The purpose of covering what has already become well-trodden ground to some in computer science is to try and emphasise the importance of a developmental perspective on evolutionary processes. The genetic reductionism of the 1970s in biology left a lasting impression on evolutionary computation, and it is one that has become the *de facto* viewpoint.

However, there is evidence that calls for more inclusive models of evolutionary search are gaining ground. A research agenda calling for evolutionary computation to abandon its "restricted and dated understanding of natural evolution" has recently appeared (Banzhaf and et al, 2006). That article asks the field to challenge its long held assumption that there is a "one-way flow of information, from DNA to proteins" that forms the basis of solution discovery by evolutionary search algorithms. The view prevalent among practitioners of evolutionary computation is that genetic material is essentially symbolic rather than physical. But ignoring the physical aspects of gene translation may have led the field to underestimate the importance of developmental processes on issues like scalability and re-use. Advances in developmental biology have given us fresh insights into how evolution explores a functional domain and the constraints it operates under. Criticisms of this nature have appeared elsewhere (Kumar and Bentley, 2003) but have had little impact on the field, which continues to be dominated by efforts to optimise evolutionary search.

One agenda, to investigate *how* evolutionary algorithms find solutions and *what* they are capable of finding, was set in motion by Adrian Thompson in the mid 1990s (Thompson, 1996). His *in silico* experiments were designed to encourage as much innovation from the evolutionary process as possible. By allowing free access to the physical nature of the search domain, Thompson discovered that evolution was capable of finding solutions in areas that humans would find difficult or impossible to operate (see discussions in Gordon (2001); Harding and Miller (2004); Miller and Downing (2002)). An outstanding task for evolutionary computation — for those who want to pursue Thompson's aims — is to find ways of introducing the equivalent richness of real world physics into virtual environments.

However, the introduction of richer resources does not guarantee their accessibility. In order to access interesting physical properties in evolved solutions, we may need a physical embodiment of the developmental mechanisms employed by nature. This requires a two-way flow of information that allows a phenotype to explore a functional domain in a manner controlled by the genome. A crucial ability of the developmental process is to sense environmental inputs and respond. Things grow in accordance with their surroundings, using a feedback mechanism that tells cells when to start producing certain proteins or inhibit the production of others. Evidence from the study of gene regulatory networks suggests that evolution has exploited developmental mechanisms to allow the re-use of "useful" genes in different contexts (Carroll, 2006; Carroll et al., 2001).

The issue of re-use prompts another criticism of current models of evolutionary computation, namely the "single solution genome". This artifact results from the proximity between genotype and phenotype, the translation process between them being so direct a mapping as to make them often indistinguishable. In nature, a genome controlling the developmental process selects from many potential responses, according to the developmental context. A gene used in one place will have *a different role* somewhere else, roles that are separated by time and space. The repeated morphological features we witness throughout nature are the product of developmental processes. By contrast, in evolutionary computation, selection is carried out on "instant" phenotypic solutions, randomly mutating from one generation to the next, in a process that has no natural mechanism for exploring the functional search space, or conserving and re-using useful genes (see discussion in §5.4). Such evolved solutions only work as single, fixed answers to static environments. There is no redundancy to draw on.

Biological evolution has come up with a neat trick: DNA encodes for proteins, and those proteins can govern the production of other proteins. Thus it encodes for the rules that dictate how it explores a particular functional domain. Not only that, but a tiny fraction of what could be expressed is ever realised in a phenotype. A genome contains solutions for countless *sets* of contexts. Change the contexts and the genome still has room for developmental exploration. This flexibility and redundancy of solutions has an impact on the re-use and conservation of genes during developmental processes. Without similar mechanisms of interaction and feedback, digital genomes cannot guide themselves across functional search spaces in a way that fully exploits a domain's resources, and this is particularly true where that domain includes the complexity provided by real-world physics.

While the wet manufacture of life is hardly a practical ambition for computer scientists, it contains clues, patterns if you like, of how subtle, scalable structures can be built that allow evolution to explore and interact with the world about it. DNA alone can't do that: it's a passive instruction set. To paraphrase Lewis Wolpert, it's *proteins* that do all the work (Wolpert, 2003).

References

Alberts, B., A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter (2002, March). *Molecular Biology of the Cell: Fourth Edition*. Garland. 17

Attenborough, D. (1984). *The Living Planet*. BBC Worldwide Ltd.: BBC TV in association with Time Life Films. 5, 6, 7, 8

Bak, P. (1996). *How Nature Works: The Science of Self-Organised Criticality*. Copernicus Press. 31

Banzhaf, W. and et al (2006, September). Guidelines: From artificial evolution to computational evolution: a research agenda. *Nature Reviews Genetics 7*(9), 729–735. 4, 18, 51

Bentley, P. (2004). The Garden Where Software Grows. *New Scientist 2437*. 3, 34

Bongard, J. C. and H. Lipson (2004). Automating genetic network inference with minimal physical experimentation using coevolution. See Deb et al. (2004), pp. 333–345. 34

Box, G. E. P. (1957). Evolutionary operation: A method for increasing industrial productivity. *Applied Statistics 6*(2), 81–101. 34

Brameier, M. and W. Banzhaf (2003). Neutral variations cause bloat in linear GP. In C. Ryan, T. Soule, M. Keijzer, E. Tsang, R. Poli, and E. Costa (Eds.), *Genetic Programming, Proceedings of EuroGP 2003*, Volume 2610 of *LNCS*, pp. 290–299. Springer-Verlag. 36

Bremermann, H. J. (1962). Optimization through evolution and recombination. In M. C. Yovitis and G. T. Jacobi (Eds.), *Self-Organizing Systems*, pp. 93–106. Washington, D.C.: Spartan Books. 36

Bremermann, H. J., M. Rogson, and S. Salaff (1966). Global properties of evolution processes. In H. H. Pattee, E. A. Edlsack, L. Fein, and A. B. Callahan (Eds.), *Natural Automata and Useful Simulations*, pp. 3–41. Washington D.C.: Spartan Books. 36

Cantu-Paz, E. (1998). Designing efficient master-slave parallel genetic algorithms. In J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. Riolo (Eds.), *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pp. 455. Morgan Kaufmann. 40

Cantu-Paz, E. and D. E. Goldberg (1997). Predicting speedups of ideal bounding cases of parallel genetic algorithms. In T. Bäck (Ed.), *Proc. of the Seventh Int. Conf. on Genetic Algorithms*, San Francisco, CA, pp. 113–120. Morgan Kaufmann. 40

Cardelli, L. (2005). Abstract machines of systems biology. *T. Comp. Sys. Biology 3737*, 145–168. 14

Carroll, S. (2006). *Endless Forms Most Beautiful: The New Science of Evo Devo and the Making of the Animal Kingdom*. Weidenfeld & Nicolson. 22, 23, 24, 51

Carroll, S. B., J. K. Grenier, and S. D. Weatherbee (2001). *From DNA to Diversity*. Blackwell. 18, 19, 21, 51

Coello, C. A. (2000). An updated survey of GA-based multiobjective optimization techniques. *ACM Comput. Surv. 32*(2), 109–143. 41, 42

Daida, J. (2004). Considering the roles of structure in problem solving by a computer. In U.-M. O'Reilly, T. Yu, R. L. Riolo, and B. Worzel (Eds.), *Genetic Programming Theory and Practice II*, Chapter 5, pp. 67–71. Kluwer. 36

Daida, J. and A. M. Hilss (2003). Identifying structural mechanisms in standard genetic programming. In E. e. a. Cantú-Paz (Ed.), *Genetic and Evolutionary Computation – GECCO-2003*, Volume 2724 of *LNCS*, Chicago, pp. 1639–1651. Springer-Verlag. 36

Daida, J., A. M. Hilss, D. J. Ward, and S. L. Long (2005). Visualizing tree structures in genetic programming. *Genetic Programming and Evolvable Machines 6*(1), 79–110. 36

Darwin, C. (1859). *On the Origin of Species. A facsimile of the First Edition*. Oxford University Press. 5, 11

Darwin, C. and T. Henry (1974). *Charles Darwin, Thomas Henry Huxley. Autobiographies*. Oxford University Press. (edited by Gavin de Beer). 6

Dawkins, R. (1989). *The Selfish Gene* (2nd ed.). Oxford University Press. 9, 10

Dawkins, R. (1996). *Climbing Mount Improbable*. Norton. 12

Dawkins, R. (1998). Darwin and Darwinism. In *Microsoft Encarta Encyclopedia 98*. Microsoft Corporation. 9, 10

Deb, K., R. Poli, W. Banzhaf, H.-G. Beyer, E. K. Burke, P. J. Darwen, D. Dasgupta, D. Floreano, J. A. Foster, M. Harman, O. Holland, P. L. Lanzi, L. Spector, A. Tettamanzi, D. Thierens, and A. M. Tyrrell (Eds.) (2004). *Genetic and Evolutionary Computation - GECCO 2004, Genetic and Evolutionary Computation Conference, Seattle, June 26-30, 2004, Proceedings, Part I*, Volume 3102 of *Lecture Notes in Computer Science*. Springer. 52, 58

Dimutrescu, D., B. Lazzerine, l. Jain, and A. Dumitrescu (2000). *Evolutionary Computing*. CRC Press LLC. 36, 37

Ferald, R. D. (2001). The eye in focus. *Karger Gazette 64*, 2–4. Available online: `http://www.karger.com/gazette/index.htm`. 12, 13

Fernández, F., G. Galeano, J. A. Gómez, and J. L. Guisado (2004). Control of bloat in genetic programming by means of the island model. In X. Yao, E. Burke, J. A. Lozano, J. Smith, J. J. Merelo-Guervós, J. A. Bullinaria, J. Rowe, P. T. A. Kabán, and H.-P. Schwefel (Eds.), *Parallel Problem Solving from Nature - PPSN VIII*, Volume 3242 of *LNCS*, pp. 260–269. Springer-Verlag. 36

Ferraro, P., C. Godin, and P. Prusinkiewicz (2005). Toward a quantification of self-similarity in plants. *Fractals 13(2)*, 91–109. 47

Fleischer, K. W. and A. H. Barr (1994). A simulation testbed for the study of multicellular development: The multiple mechanisms of morphogenesis. *Artificial Life III*, 389–416. 47

Fogel, D. B. (Ed.) (1998). *Evolutionary Computation: the fossil record*. Piscataway, NJ: IEEE Press. 36, 41

Fogel, D. B. (2000). *Evolutionary Computing*. IEEE. 12, 26, 35, 36, 37, 38, 39, 40, 46

Fogel, D. B. and A. Ghozeil (1997). A note on representations and variation operators. *IEEE Trans. on Evolutionary Computation 1*(2), 159–161. 39

Fogel, L. (1962). Autonomous automata. *Industrial Research 4*, 14–19. 34

Fogel, L. (1963). *Biotechnology: Concepts and Applications*. Prentice Hall. 34

Fonlupt, C. (2005). Book review: Genetic programming IV: Routine human-competitive machine intelligence. *Genetic Programming and Evolvable Machines 6*, 231–233. 3, 35

Fraser, A. S. (1957). Simulation of genetic systems by automtic digital computers I.–Introduction. *Australian Journal of Biological Sciences 10*, 484–491. 36

Fraser, A. S. (1968). The evolution of purposive behaviour. In H. von Foerster, J. D. White, L. J. Peterson, and J. K. Russell (Eds.), *Purposive Systems*, pp. 15–23. Spartan Books. 36

Friedburg, R. (1958). A learning machine. *IBM Journal of Research and Development 2*, 2–13. 34

Friedman, G. J. (1956). Selective feedback computers for engineering synthesis and nervous system analogy. Master's thesis, University of California, Los Angeles. 34

Fuhrer, M., H. W. Jensen, and P. Prusinkiewicz (2006). Modeling hairy plants. *Proceedings of Pacific Graphics 68*(4), 333–342. 49

Gavrilets, S. (1997). Evolution and speciation on holey adaptive landscapes. *Trends in Ecology and Evolution 12*, 307–312. 27

Gehring, W. and K. Ikeo (1999). Pax 6: Mastering eye morphogenesis and eye evolution. *Trends Genet 15*, 371–377. 13

Goldberg, D. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley. 36, 39

Goldberg, D. (2002). *The Design of Innovation: Lessons from and for competant Genetic Algorithms*. Kluwer Academic Publishers. 36, 42, 44

Goodwin, B. (1994). *How the Leopard Changed Its Spots: The Evolution of Complexity*. Charles Scribner's Sons. 11

Gordon, T. and P. Bentley (2002). On evolvable hardware. 45, 47

Gordon, T. G. (2001). Book review of Hardware Evolution by Adrian Thompson. In *Genetic Programming and Evolvable Machines*, 2(4). Kluwer Academic Publishers. 51

Gould, S. J. (1989). *Wonderful Life: The Burgess Shale and the Nature of History*. Norton. 11

Haddow, P. C. and J. Hoye (2007). Achieving a simple development model for 3D shapes: are chemicals necessary? In H. Lipson (Ed.), *GECCO '07*, pp. 1013–1020. ACM. 49

Haddow, P. C. and G. Tufte (2001). Bridging The Genotype-Phenotype Mapping For Digital FPGAs. In *Evolvable Hardware*, pp. 109–115. IEEE Computer Society. 49

Hancock, J. T. (2003). The principles of cell signalling. See Kumar and Bentley (2003). 14

Harding, S. and J. Miller (2004). Evolution in materio: Initial experiments with liquid crystal. In *Evolvable Hardware*, pp. 298–. IEEE Computer Society. 51

Harding, S. and J. F. Miller (2003). A scalable platform for intrinsic hardware and in materio evolution. In *Evolvable Hardware*, pp. 231–234. IEEE Computer Society. 45

Hogeweg, P. (2000a). Evolving mechanisms of morphogenesis: on the interplay between differential adhesion and cell differentiation. *Journal of Theoretical Biology 203.4*, 317–33. 47

Hogeweg, P. (2000b). Shapes in the shadow: Evolutionary dynamics of morphogensis. *Artificial Life 6*(1), 85–101. 47

Holland, J. (1962). Outline for a logical theory of adaptive systems. *Journal of the ACM 3*, 297–314. 36

Holland, J. (1975). *Adaptation in Natural and Artifical Systems* (1st ed.). University of Michigan Press. 38, 39

Holland, J. (1992). *Adaptation in Natural and Artifical Systems* (Fifth printing, 1st MIT ed.). MIT Press. 36

Hornby, G. S., M. Fujita, S. Takamura, T. Yamamoto, and O. Hanagata (1999). Autonomous Evolution of Gaits with the Sony Quadruped Robot. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference*, Volume 2, pp. 1297–1304. Morgan Kaufmann. 48

Hornby, G. S., H. Lipson, and J. B. Pollack (2001). Evolution of generative design systems for modular physical robots. In *IEEE International Conference on Robotics and Automation*. 48

Hornby, G. S. and J. B. Pollack (2001a). The advantages of generative grammatical encodings for physical design. In *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*, pp. 600–607. IEEE Press. 48

Hornby, G. S. and J. B. Pollack (2001b). Evolving L-systems to generate virtual creatures. *Computers and Graphics 25*(6), 1041–1048. 48

Hornby, G. S. and J. B. Pollack (2002). Creating high-level components with a generative representation for body-brain evolution. *Artif. Life 8*(3), 223–246. 48

Howard, J. (2001). *Darwin. A Very Short Introduction*. Oxford University Press. 5, 6, 7, 8, 11

Hutcheon, P. D. (1996). Fear ignorance – not sociobiology! *Humanist in Canada 9*, 12–14. 33

Jacob, C. (1999). Lindenmayer systems and growth program evolution. In T. S. Hussain (Ed.), *Advanced Grammar Techniques Within Genetic Programming and Evolutionary Computation*, Orlando, Florida, USA, pp. 76–79. 47

Jin, Y. (2005). A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing 9*(1), 3–12. 3

Kauffman, S. (1995). *At Home in the Universe: The search of the laws of self-organization and complexity*. Oxford University Press. 24, 31

Kauffman, S. (2000). *Investigations*. Oxford University Press. 26, 28, 29, 30, 31

Kauffman, S. and S. Johnsen (1991). Coevolution at the edge of chaos: coupled fitness landscapes, poised states and coevolutionary avalanches. *Journal of Theoretical Biology 149*, 467–482. 31

Kauffman, S. and S. Levin (1987). Towards a general theory of adaptive walks on rugged landscapes. *Journal of Theoretical Biology 128*, 11–45. 25, 28

Keane, A. M. (2000). An Intoduction to Evolutionary Computing in Design Search and Optimisation. In A. M. Keane (Ed.), *Theoretical Aspects of Evolutionary Computing*. Kluwer Academic. 41

Kosman, D., C. M. Mizutani, D. Lemons, W. G. Cox, W. McGinnis, and E. Bier (2004). Multiplex detection of rna expression in drosophila embryos. *Science 305*(5685), 846. 21

Koza, J. R. (1989). Hierarchical genetic algorithms operating on populations of computer programs. In N. S. Sridharan (Ed.), *Proc. of the 11th Joint Conf. on Genetic Algorithms*, San Francisco, CA, pp. 786–774. Morgan Kaufmann Publishers. 39

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press. 35

Koza, J. R. (1994). *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press. 35

Koza, J. R., D. Andre, F. H. Bennett III, and M. Keane (1999). *Genetic Programming 3: Darwinian Invention and Problem Solving*. Morgan Kaufmann. 35

Koza, J. R., L. Jones, M. Keane, and M. Streeter (2004). Towards industrial strength automated design of analog electrical circuits by means of genetic programming. In U.-M. O'Reilly, T. Yu, R. L. Riolo, and B. Worzel (Eds.), *Genetic Programming Theory and Practice II*, Chapter 8, pp. 121–138. Kluwer. 36, 41

Koza, J. R., M. Keane, and M. Streeter (2003, Feb). Evolving inventions. *Scientific American*, 52–59. 40, 41

Koza, J. R., M. A. Keane, M. J. Streeter, W. Mydlowec, J. Yu, and G. Lanza (2003). *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. Kluwer Academic Publishers. 3, 35

Kumar, S. and P. Bentley (Eds.) (2003). *On Growth, Form and Computers*. Elsevier. 13, 15, 17, 47, 51, 55, 61

Langdon, W. B. (2000). Quadratic bloat in genetic programming. In D. Whitley, D. E. Goldberg, E. Cantu-Paz, L. Spector, I. Parmee, and H.-G. Beyer (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000)*, Las Vegas, Nevada, USA, pp. 451–458. Morgan Kaufmann. 36

Langdon, W. B. and W. Banzhaf (2000). Genetic programming bloat without semantics. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel (Eds.), *Parallel Problem Solving from Nature - PPSN VI 6th International Conference*, Volume 1917 of *LNCS*, Paris, France, pp. 201–210. Springer Verlag. 36

Langdon, W. B. and R. Poli (1997). Genetic programming bloat with dynamic fitness. Technical Report CSRP-97-29, University of Birmingham, School of Computer Science. 36

Lenat, D. (1983). The role of heuristic in learning by discovery: Three case studies. In R. Michalski, J. Carbonell, and T. Michell (Eds.), *Machine Learning*, pp. 243–306. Tioga Publishing. 35

Lewontin, R. (1974). *The Genetic Basis of Evolutionary Change*. Columbia University Press. 45, 46

Lindenmayer, A. and P. Prusinkiewicz (1989). Developmental models of multicellular organisms: A computer graphics perspective. In C. G. Langton (Ed.), *Artificial Life*, pp. 221–249. Addison-Wesley. 47

Lohn, J., A. Stoica, and D. Keymeulen (Eds.) (2000). *Proc. 2000 NASA/DoD workshop on Evolvable Hardware*. IEEE Computer Society. 60

Lones, M. A. (2003). *Enzyme Genetic Programming: Modelling Biological Evolvability in Genetic Programming*. Ph. D. thesis, The University of York, Heslington, York, YO10 5DD, UK. 47

MacKay, D. (2003, Sep). *Information Theory, Inference and Learning Algorithms*. Cambridge University Press. 39

Michalewicz, Z. (1992). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer. 39

Miller, J. (2000). Review: First NASA DOD Workshop on Evolvable Hardware 1999. *Genetic Programming and Evolvable Machines 1*, 171–174. 3

Miller, J. F. (2004). Evolving a Self-Repairing, Self-Regulating, French Flag Organism. See Deb et al. (2004), pp. 129–139. 48

Miller, J. F. and K. Downing (2002). Evolution in materio: Looking beyond the silicon box. In A. Stoica, J. Lohn, R. Katz, D. Keymeulen, and R. S. Zebulum (Eds.), *Proc. 2002 NASA/DoD Conference on Evolvable Hardware*, pp. 167–176. IEEE Computer Society. 45, 51

Měch, R. and P. Prusinkiewicz (1996). Visual models of plants interacting with their environment. In *SIGGRAPH*, pp. 397–410. 47, 50

Peisach, D., P. Gee, C. Kent, and Z. Xu (2003). The crystal structure of choline kinase reveals a eukaryotic protein kinase fold. *Structure 11*, 703–713. 16

Provine, W. B. (1986). *Sewall Wright and Evolutionary Biology*. University of Chicago Press. 25, 27

Prusinkiewicz, P. (2000). Simulation modeling of plants and plant ecosystems. *Communications of the ACM 43*(7), 84–93. 47

Prusinkiewicz, P. and A. Lindenmayer (1990). *The Algorithmic Beauty of Plants*. Springer-Verlag. 48

Rechenberg, I. (1963). *Evolutionsstrategie: Optimierung Technisher Systeme nach Prinspipien der Biologischen Evolution*. Frommann-Holzboog. 35

Rechenberg, I. (1964). Kybernetische Lösungsansteuerung einer experimentellen Forschungsaufgabe. Presented at the Annual Conference of the WGLR at Berlin in September 1964. 35

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzbog. 35

Robinson, G. and P. McIlroy (1995). Exploring some commercial applications of genetic programming. In T. C. Fogarty (Ed.), *Evolutionary Computing*, Number 993. Springer-Verlag. 41

Ruse, M. (1996). Are Pictures Really Necessary? The Case of Sewall Wright's Adaptive Landscapes. In B. Baigre (Ed.), *Picturing Knowledge: Historical and Philosphical Problems Concerning the Use of Art in Science*. University of Toronto. 25

Saunders, P. (1993). The organism as a dynamical system. In W. Stein and F. Varela (Eds.), *Thinking about Biology*, Volume III, pp. 41–63. Addison-Wesley. 32

Schraudolph, N. N. and R. K. Belew (1992). Dynamic parameter encoding for genetic algorithms. *Machine Learning 9*, 9–21. 35, 39

Schwefel, H.-P. (1975). *Evolutionsstrategie und numerische Optimierung*. Ph. D. thesis, Technische Universität Berlin, Berlin, Germany. 35

Schwefel, H.-P. (1977). *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*. Basel, Stuttgart: Birkhäuser. Volume 26 of Interdisciplinary Systems Research. 35

Schwefel, H.-P. (1981). *Numerical optimization of Computer models*. John Wiley & Sons, Ltd. 35

Sharman, K. C., A. I. Esparcia Alcazar, and Y. Li (1995). Evolving signal processing algorithms by genetic programming. In A. M. S. Zalzala (Ed.), *First International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, GALESIA*, Volume 414, Sheffield, UK, pp. 473–480. IEE. 41

Shipman, R., M. Shackleton, and I. Harvey (2000). The use of neutral genotype-phenotype mappings for improved evolutionary search. *BT Technology Journal 18*(4), 103–111. 45

Skipper, R. (2002). The Heuristic Role of Sewall Wright's 1932 Adaptive Landscape Diagram. In *Proceedings Philosophy of Science Assoc. 18th Biennial Mtg - PSA 2002: PSA 2002 Symposia*. 25, 27, 28

Solé, R. and B. Goodwin (2000). *Signs of Life: how complexity pervades biology*. Basic Books. 24, 26, 27, 28, 29, 30, 31

Solomonoff, R. (1966). Some recent work in artificial intelligence. *Proc. of the IEEE 54:12*, 1687–1697. 34

Stadler, P. (2002). Fitness landscapes. In M. Lässig and A. Valleriani (Eds.), *Biological Evolution and Statistical Physics*, pp. 187–207. Springer-Verlag. 28

Stanley, K. and R. Mikkulainen (2003). A taxonomy for artificial embryogeny. *Artificial Life 9*(2), 93–130. 50

Stephens, C. (2003). interviewed by EvoNet, `http://www.evonet.org`. 41

Streeter, M., M. Keane, and J. Koza (2003). Routine human-competitive automatic synthesis using genetic programming of both the topology and sizing for five post-2000 patented analog and mixed analog-digital circuits. In *2003 Southwest Symposium on Mixed-Signal Design*, pp. 5–10. IEEE Circuits and Systems Society. 36

Syswerda, G. (1989). Uniform crossover in genetic algorithms. In J. D. Schaffer (Ed.), *Proc. of the Third Int. Conf. on Genetic Algorithms*, San Mateo, CA, pp. 2–9. Morgan Kaufmann. 39

Syswerda, G. (1991). Schedule optimization using genetic algorithms. In L. Davis (Ed.), *Handbook of Genetic Algorithms*, pp. 332–349. Van Nostrand Reinhold. 39

Thompson, A. (1996). Silicon evolution. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo (Eds.), *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 444–452. MIT Press. 44, 51

Thompson, A. (1997). An Evolved Circuit, Intrinsic in Silicon, Entwined with Physics. In T. Higuchi, M. Iwata, and L. Weixin (Eds.), *Proc. 1st Int. Conf. on Evolvable Systems (ICES'96)*, pp. 390–405. Springer-Verlag. 44, 45

Thompson, A. (2002). Notes on design through artificial evolution: Opportunities and algorithms. In I. C. Parmee (Ed.), *Adaptive computing in design and manufacture V*, pp. 17–26. Springer-Verlag. 42, 44

Thompson, A., P. Layzell, and R. S. Zebulum (1999). Explorations in Design Space: Unconventional Electronics Design Through Artificial Evolution. *IEEE Transactions on Evolutionary Computation 3*(3), 167–196. 45

Tomassini, M., L. Vanneschi, J. Cuendet, and F. Fernandez (2004). A new technique for dynamic size populations in genetic programming. In *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*, Portland, Oregon, pp. 486–493. IEEE Press. 36

Torrensen, J. (2000). Scalable Evolvable Hardware Applied to Road Image Recognition. See Lohn et al. (2000), pp. 151–160. 44

Tufte, G. and P. Haddow (2003). Building knowledge into developmental rules for circuit design. *Proc. of the 5th Int. Conf. on Evolvable Systems (ICES 2003)*, 69–80. 47

Turing, A. (1950). Computing machinery and intelligence. *Mind 49*, 433–460. 34

Turing, A. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society 237*(641), 37–72. 50

Van Valen, L. (1973). A new evolutionary law. *Evolutionary Theory 1*, 1–30. 30

Vanneschi, L. (2004). *Theory and Practice for Efficient Genetic Programming*. Ph. D. thesis, Faculty of Sciences, University of Lausanne, Switzerland. 36

Vassilev, V. K., D. Job, and J. F. Miller (2000). Towards the automatic design of more efficient digital circuits. See Lohn et al. (2000), pp. 151–160. 44

Voigt, H.-M., W. Ebeling, I. Rechenberg, and H.-P. Schwefel (Eds.) (1996). *Parallel Problem Solving from Nature – PPSN IV*, Berlin. Springer. 35

Vose, M. D. (1999a). *The simple genetic algorithm: foundations and theory*. Cambridge, MA: MIT Press. 36

Vose, M. D. (1999b). What are genetic algorithms? a mathematical perspective. In L. D. Davis, K. De Jong, M. D. Vose, and L. D. Whitley (Eds.), *Evolutionary Algorithms*, pp. 251–276. Springer. 36

Waddington, C. (1957). *The Strategy Of The Genes*. George Allen & Unwin. 32

Wearing, H., M. Owen, and J. Sherratt (2000). Mathematical modelling of juxtacrine patterning. *Bulletin of Mathematical Biology 62*, 293–320. 15

Wolpert, D. and W. Macready (1995). No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM. 40

Wolpert, D. and W. Macready (1997). No free lunch theorems for optimization. *IEEE Trans. on Evolutionary Computation 1*(1), 67–82. 40

Wolpert, L. (1998). *The Principles of Development*. Oxford University Press. 13, 15, 49

Wolpert, L. (2003). Relationships Between Development And Evolution. See Kumar and Bentley (2003), Chapter 2, pp. 47–62. 14, 16, 18, 19, 20, 52

Wolpert, L., R. Beddington, and T. Jessell (2002). *Principles of Development*. Oxford University Press. 22

Wright, A. H. (1991). Genetic algorithms for real parameter optimization. In G. J. Rawlins (Ed.), *Foundations of genetic algorithms*, pp. 205–218. Morgan Kaufmann. 39

Wright, A. H., M. D. Vose, and J. E. Rowe (2003). Implicit parallelism. In E. Cantú-Paz, J. A. Foster, K. Deb, L. Davis, R. Roy, U.-M. O'Reilly, H.-G. Beyer, R. K. Standish, G. Kendall, S. W. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. A. Potter, A. C. Schultz, K. A. Dowsland, N. Jonoska, and J. F. Miller (Eds.), *GECCO*, Volume 2724 of *Lecture Notes in Computer Science*, pp. 1505–1517. Springer. 39

Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding and selection in evolution. In *Proceedings of the Sixth Annual Congress of Genetics*, Volume 1, pp. 356–366. 25, 27

NB. Numbers in red after the source show the page numbers in this document where the source is cited (hyperlinked).