Kojonen's The Compatibility of Evolution and Design

with Zachary Ardern, "The Contentious Compatibility of Evolution and Design: Introduction to the Book Symposium"; David H. Glass, "An Evaluation of the Biological Case for Design"; Meghan D. Page, "Thomist or Tumblrist: Comments on The Compatibility of Evolution and Design *by E. V. R. Kojonen"; Peter Jeavons, "The Design of Evolutionary Algorithms: A Computer Science Perspective on the Compatibility of Evolution and Design"; Denis R. Alexander, "Evolution, Chance, Necessity, and Design"; Bethany N. Sollereder, "Response to* The Compatibility of Evolution and Design*"; Mats Wahlberg, "Divine Design and Evolutionary Evil"; and Erkki V. R. Kojonen, "Response: The Compatibility of Evolution and Design."*

THE DESIGN OF EVOLUTIONARY ALGORITHMS: A COMPUTER SCIENCE PERSPECTIVE ON THE COMPATIBILITY OF EVOLUTION AND DESIGN

by Peter Jeavons

Abstract. The effectiveness of evolutionary algorithms is one of the issues discussed in *The Compatibility of Evolution and Design*, where it is argued that such algorithms are only effective when stringent preconditions are met. This article considers this issue from the perspective of computer science. It explores the properties of problems that can be effectively solved by evolutionary algorithms, and the extent to which such algorithms need to be carefully adjusted. Although there are important differences between the study of evolutionary algorithms in computer science and the study of biological evolution, it is argued that the experience of computer science can help to strengthen the claim that evolutionary mechanisms generally require a considerable degree of fine-tuning.

Keywords: evolution; fine-tuning; fitness function; fitness landscape; teleology

---

Introduction: What Does the Study of Evolutionary Algorithms Contribute to the Design Debate?

The role of a computer scientist is to design and analyze computational and algorithmic approaches to problems. The idea of using *evolutionary*

Peter Jeavons is an Emeritus Professor of Computer Science, University of Oxford, Oxford, UK; e-mail: peter.jeavons@cs.ox.ac.uk.

 ISSN 0591-2385

algorithms to solve computational problems has now been explored by computer scientists for at least five decades (Holland 1975). The idea of a generic approach that can solve a wide variety of problems, a "universal acid" that can dissolve all difficulties, in the words of Daniel Dennett (1995), is very attractive. There is a wide literature on the subject and a number of regular conferences.

Using evolutionary algorithms has a particular appeal to students, as it offers the apparent prospect of solving problems that we do not know how to tackle any other way—problems that are poorly understood or appear to be difficult to solve with conventional approaches. It is a popular topic for student projects.

However, the sad fact of experience is that evolutionary approaches to computational problems, far from providing a convenient short-cut to success, generally only achieve satisfactory results after a great deal of adjusting, tuning, and development (Eiben and Smith 2015). This empirical finding is now becoming accepted amongst computer scientists, and the popularity of using evolutionary approaches to solve computational problems may be declining somewhat. It is becoming clear that there is no "universal acid"—to achieve significant results, an evolutionary algorithm must be carefully tailored to the problem in hand, and the problem itself must have appropriate properties.

The issue of the effectiveness of evolutionary algorithms is a key element of the discussion in Chapter 4 of *The Compatibility of Evolution and Design* (2021), which is called "Not by Selection Alone: Evolutionary Explanations and Their Requirements." This chapter considers carefully the assumption that is often made that "evolution was not difficult, or did not require design." The whole chapter argues, to the contrary, that "evolution … has very demanding preconditions." Although dealing primarily with evolution in the context of biology, it touches on wider issues of evolutionary algorithms in other areas as well. I think understanding the necessary conditions for an evolutionary algorithm to be effective is an important question, and some of my own research has attempted to explore and quantify these kinds of issues for evolutionary algorithms in general.

This argument is framed by Kojonen as a claim about the stringent necessary preconditions for (biological) evolution. I think this case is well made. I am going to argue here that the analysis and experience of evolutionary algorithms from computer science also allows us to make similar statements about the stringent preconditions for effective evolutionary approaches in other contexts beyond biology.

Moreover, I will argue that it is not just the preconditions that need to be carefully tuned to ensure success, but also the details of the algorithmic implementation itself. The idea that design may be seen in the process of going from a vague algorithmic strategy to a detailed implementation of that algorithm (as well as in the results of running the algorithm), may be

unfamiliar to the "theist on the street," who features prominently in Kojonen (2021), but this idea may be particularly congenial to a computer scientist. The word "design," as used in computer science or software engineering, generally refers to the act of transforming an idea for a program into a fully worked-out implementation, with all the details instantiated. This transformation is generally not easy, it can require a high degree of skill, and this skill is what professional software engineers are paid for. The idea that having a high-level concept such as "evolution by natural selection" is sufficient on its own, and that no further design work is necessary to build a process that works in practice, is out of line with the experience of anyone who has written a substantial program.

The argument that Kojonen makes in Chapter 4, and that I will try to support here, is similar to the argument from fine-tuning in cosmology, discussed in Chapter 3 of Kojonen (2021). The fine-tuning argument in cosmology is based on the surprising discovery that the laws of physics and the fundamental constants of nature seem to have to be very precisely set in order for their outworking to yield a complex universe where it is possible for life to exist (Lewis and Barnes 2016). A similar fine-tuning argument for evolutionary algorithms would run as follows: the study of evolutionary algorithms in computer science has shown that the detailed features of such algorithms, and the environment that they work in, need to be carefully chosen and delicately balanced, in order for the outworking of such an algorithm to yield high quality outcomes in a feasible amount of time.

I will consider below some of the evidence for such claims. When exploring the properties of algorithms, it can be helpful to make a distinction between the properties of the problem to be solved, and the properties of the algorithm that is being used to solve it; I will consider each of these two aspects in turn in what follows. Of course in the context of biological evolution it may be misleading to think about evolution as "solving a problem," but it may still be helpful to conceptualize biological evolution as the outworking of an evolutionary algorithm in a particular problem space. I will discuss some specific features of biological evolution after considering general properties of computational problems and evolutionary algorithms.

Properties of Problems

What kinds of problems are suitable for evolutionary algorithms? Usually, problems where we are looking for a combination of choices that together achieve some desired quality. To model such problems, we typically represent the choices to be made by a list of variables, each of which can be set to a number of different values. Each possible *combination* of settings

for this list of variables is then assigned a numerical value, referred to as its "fitness."

This model of the problem we are trying to solve defines a "fitness landscape," which is an abstract high-dimensional space consisting of all the possible combinations of settings for the relevant variables,[1] where each combination has an associated numerical fitness value. Often the variables can only take a finite number of discrete values, so the fitness is defined by a numerical function of many discrete variables.

Functions of many discrete variables, are generally hard to visualize, and in most cases quite hard to specify in a concise way. Because of this difficulty, there is a tendency to impose our intuitions from the much lower-dimensional continuous spaces we are used to, such as the three-dimensional space that we inhabit, or even the two dimensions of a standard geographical map. Abstract fitness landscapes are sometimes visualized as similar to real landscapes in some rural idyll, with gently rolling hills and valleys. In such visualizations, the fitness value corresponds to the height at a given position; hence, the fitness function is simplified to a function of just two values, the x and y coordinates on the map. This form of visualization was introduced in the context of biological evolution by Sewall Wright (1932) and is still the default mental model for many people when considering the behavior of evolutionary algorithms.

However, such appealing visualizations can give a very misleading or oversimplified picture.

In a few cases, the picture of a gently sloping landscape, where a sequence of small moves uphill will take us in a direct path from wherever we happen to be to the highest point in the landscape (that is, the point of maximum fitness) is an appropriate picture, but this is only true for very special functions with very special properties. For example, if the fitness value depends on each of its input parameters *separately*, then we can indeed move easily to the point with the highest possible fitness value, by simply adjusting each individual variable to its best value and keeping it there. The fitness function in such cases is, in some sense, not really a function of many variables at all, it is a sum of functions of a single variable,[2] and each variable can be individually adjusted without worrying about the others. Such functions are sometimes called "separable" functions.

This very unusual special property of being separable is possessed by the fitness function in the classic example introduced by Dawkins in *The Blind Watchmaker* (Dawkins 1986) and discussed on page 107 of *The Compatibility of Evolution and Design*. In this example, it is shown that a random sequence of letters can be easily mutated step by step into the phrase "METHINKS IT IS LIKE A WEASEL." In order to achieve this, a fitness function is defined which depends on the closeness to the target phrase, and a simple evolutionary algorithm is then used to maximize this function by gradually modifying random sequences and selecting those which have a

higher value for the fitness function (that is, those that resemble the target phrase more closely when looking at individual letters).

This example has been criticized for building-in information about the target phrase into the fitness function, and hence not demonstrating the emergence of any new information. Kojonen considers these criticisms and agrees that they have some weight, but he concludes that "in Dawkins' WEASEL algorithm, for example, it is not inappropriate for there to be a selecting condition for the letters, because there are also selective conditions in natural environments. The point of the algorithms is to demonstrate the possibility of building complex order by a stepwise process involving differential survival, if the preconditions of such an evolutionary algorithm are met."

However, the point I am drawing attention to here is not that the target phrase is built-in to the definition of the fitness function, but that the fitness function has been chosen to be of a particularly simple and helpful kind to facilitate the evolutionary process. Because the fitness function has been chosen to be a sum of distances to the target function for each letter *separately*, it has a single peak value which is easily located incrementally: every possible sequence of letters can be easily adjusted to a sequence with a higher fitness value, by changing one letter (or more than one). This means that every possible sequence of 28 letters or spaces (all 1.19×10^{40} of them) has many, many sequences of small steps toward the target phrase along which the fitness gently rises, and each of these improving sequences has only 28 steps or fewer. These special properties of the fitness function make an evolutionary algorithm simple to write and virtually certain to succeed.

But what if we select a more realistic fitness function, where the fitness values represent the quality of the sequence of letters *as a whole*, in some way, rather than the individual letters? For example, the fitness could be defined by how plausible the entire sequence of letters is as a statement by Hamlet about clouds and weasels? Or how well the sequence works as a whole as a phrase about weasels in iambic pentameter? A fitness function that depends on the overall sequence, rather than the individual letters, is arguably closer to the "selective conditions in natural environments" which do, after all, concern the viability and likely reproductive success of the organism as a whole. Fitness functions of these kinds arguably still have the problem that they build-in knowledge about the desired outcome, but with this kind of fitness function simple evolutionary algorithms are much less likely to succeed in reaching any phrase about weasels in a reasonable amount of time. In many areas of the fitness landscape, such a fitness function would provide no guidance as to how to modify the current sequence, since the current sequence and all its neighbors would have a zero fitness value. In such regions, even making the first steps of an evolutionary algorithm would be difficult. In other places, the possible modifications to

improve fitness might take the sequence in different, incompatible, directions. There would be many cases where all small changes would be individually deleterious, reducing the fitness value, even if jointly advantageous. For example, from the sequence "YONDER CLOUD ALMOST IN CAMEL," what small modifications would increase its fitness?

It is a simple mathematical fact that the vast majority of fitness functions are not separable: they cannot be expressed as sums of functions of a single variable. In general, the value for any particular variable that results in the highest fitness value will depend on the value of the other variables and cannot be simply fixed once and for all. For example, in the biological context, it clearly makes no sense to talk about the correct or optimal value for one single base-pair in a genome—it obviously cannot be defined without knowing the context of the other base-pairs in the rest of the genome. Hence Dawkins' example, and the intuition behind it, of gently rising short paths toward a unique optimal fitness peak is misleading for this application.

Once we move away from the rather trivial case of separable fitness functions, can evolutionary algorithms still achieve success? That depends of course on how success is defined. One measure of success may be the highest fitness value that is found by the algorithm. Another may be the number of generations or steps needed to achieve a high fitness value. We now consider how the nature of the fitness function may impact on each of these.

For almost all fitness functions, there will be more than one combination of the input values that constitute a *local peak*, in the sense that changing any of the variables individually will only reduce the fitness value. These local peaks may have values that are much lower than the overall best possible fitness value. This arises in all cases where changes to different variables are separately deleterious but jointly advantageous—a very common feature of fitness landscapes for interesting problems.

Hence, if success means finding a combination of values for the variables that has a high fitness value, then this may be difficult to achieve by a sequence of small modifications. Simple evolutionary algorithms, which make small modifications to individual variables, may not be able to find any variants with improved fitness, and the population may become stuck at a local peak with a rather small fitness value (Watson 2006). For fitness functions with multiple local peaks, which are sometimes called "rugged," we have to design more sophisticated algorithms, as we will discuss in the next section.

For example, if the problem we are trying to solve is the design of a lock-and-key mechanism, where some of our variables describe the structure of the lock, and others describe the shape of the key, then once we have a basic system that works (i.e., the key fits in the lock) it will be very hard to improve: changing any of the individual variables is likely to make the

mechanism worse, since the key will no longer fit the lock. Only by making coordinated changes to whole sets of variables at once can we move to an improved system overall, and this requires a more sophisticated approach than simply making changes at random to individual variables and selecting for increased fitness. Every functioning lock-and-key mechanism, however basic or unsuitable, is likely to be a local peak of a fitness function that measures overall usefulness, and improving such systems with an evolutionary algorithm will require a careful restructuring of either the fitness function or the algorithmic approach.

The problem of moving away from one local peak to another, potentially better, point in the fitness landscape is widely recognized. Already in 1935, Wright wrote that "It has seemed to me … that the central problem of evolution … is that of a trial and error mechanism by which the locus of a population may be carried across a saddle from one peak to another and perhaps a higher one" (Wright 1935, 264). For a detailed discussion of this issue, and the various responses that have been made to it, with a substantial list of references, see Chapter 1 of Richter and Engelbrecht (2013). As a rough summary, Watson (2006) notes that, on the whole, biologists have tended to assume that the relevant biological fitness landscapes will somehow avoid the problem of local peaks by always providing incrementally improving paths from points with low fitness to points with high fitness; computer scientists, in contrast, have generally sought to refine the basic evolutionary algorithm with extra mechanisms to escape from local peaks, as discussed in the next section. Either approach can be seen as a case of fine-tuning the basic evolutionary process to avoid getting stuck at local peaks.

In addition to the problem of local peaks, there is another issue which may limit the ability of an evolutionary algorithm to achieve even a local peak of fitness. One important feature of any algorithm is the time that it takes to achieve its goal. For computer scientists, an algorithm for which the number of steps required to carry it out increases exponentially with the size of the input is generally considered to be infeasible, except for very small problems. This is the fundamental reason why blind search over all possible combinations of variables is not considered a useful algorithm for maximizing a fitness function—the number of combinations that need to be explored increases exponentially with the number of variables, and soon becomes totally impractical, even on the fastest computers.

Kojonen discusses this issue on page 102, and points out that, because of the need to consider the time required for the process, "evolutionary theory is built on the assumption that chance alone is insufficient as an explanation." Clearly some element of chance, in the form of random variation, is an important component of evolutionary algorithms, but such algorithms can only be considered as more effective than chance alone when they achieve their goals in a more feasible amount of time. In other words,

an evolutionary algorithm that requires an exponential number of steps to make significant progress may well be no more practical or effective than simply trying combinations at random.

In the early days of evolutionary computing, it was sometimes naively assumed that evolutionary algorithms would be more effective than blind search across the whole range of possible problems (see Goldberg 1989, fig. 1.4, as cited in Eiben and Smith 2015, 44). However, this view was shown to be incorrect by the "No Free Lunch" theorem (Wolpert and Macready 1997), which is mentioned briefly by Kojonen (2021, 111). This theorem demonstrated that, when averaged across all possible fitness functions, the performance of any search algorithm that relies on sampling the problem space, without knowing its structure, would be the same as blind search. In other words, an evolutionary search algorithm that is more effective than blind search on some fitness functions must pay the price by being worse on others. The relevance of considering all possible fitness functions (including extremely bizarre and unlikely ones) may be disputed, but the No Free Lunch theorem has been extended and sharpened in a number of ways, including to the "Almost No Free Lunch Theorem" (ANFL) (Droste, Jansen, and Wegener 2002), and is still a helpful corrective to overoptimistic views. Jansen summarizes the situation as follows:

> We know from the ANFL … that good performance on some problems implies bad performance on many others. The best we can hope for is to tune the search heuristic we work on in a way that it performs well on the problems we happen to be interested in. (Jansen 2013, 62)

In trying to understand better the time required for an evolutionary process to find a local peak, Kaznatcheev (2019) introduced a scheme for categorizing fitness landscapes as "easy" or "hard." He notes that there are families of fitness functions where, from some starting points, following improving paths can trace out exponentially many steps, before reaching a local peak. How common such landscapes are is not yet clear, and it has been shown (Kaznatcheev, Cohen, and Jeavons 2020) that restricting the nature of the fitness function to some simple forms can ensure that all improving paths are of a more practical length (i.e., a polynomial function of the number of variables). However, the mere existence of landscapes with exponentially long paths to the nearest peak serves to counteract the oversimplifying intuition that fitness landscapes all resemble gently rolling hills. Kaznatcheev proposes that the existence of hard landscapes:

> suggests an alternative metaphor for fitness landscapes: fitness landscapes as mazes with the local fitness optima as exits. Natural selection cannot see far in the maze, and must rely only on local information from the limited genetic variation of nearby mutants. In hard mazes, we can end up following exponentially long winding paths to the exit because we cannot spot the shortcuts. (Kaznatcheev 2019, 2)

He concludes that, in such cases, finding a combination of variables with a locally optimal fitness value by a simple evolutionary algorithm will require an infeasible amount of time, and conjectures that this may account for various instances of open-ended and ongoing evolution in biology.

Kojonen concludes in Chapter 4 that many of the questions that have been raised about the effectiveness of evolutionary mechanisms in biology, including notions such as "irreducible complexity," can helpfully be viewed as a discussion of the properties of a fitness function that are required in order for an evolutionary algorithm to obtain certain kinds of results within a feasible amount of time. Rather than viewing such arguments as attempts to show the impossibility of biological evolution, they can be viewed as showing "how demanding the conditions for evolvability are, and how much fine-tuning evolution actually requires" (Kojonen 2021, 119). I think this is a helpful and productive approach.

Current research on the structural properties of fitness functions may shed more light on such questions, which are crucial to understanding the power of evolutionary algorithms in general, as well as understanding more about the preconditions needed for biological evolution in particular.

Properties of Evolutionary Algorithms

We now turn from the properties of the fitness function that are required for an evolutionary algorithm to work effectively, to the properties of such algorithms themselves. Although evolutionary algorithms are discussed in Kojonen (2021), especially in Chapter 4, no detailed description is given for such an algorithm. This is surely appropriate in a book on philosophical aspects of design arguments, but it may leave the reader unclear as to what is actually involved in going from the basic idea of evolution to a fully implemented specific algorithm. What is it about such an algorithm that needs to be designed?

A helpful breakdown of the high-level design decisions that need to be made to implement any evolutionary algorithm is given in Jansen (2013). He lists five modules that must be implemented and combined to construct a typical evolutionary algorithm: initialization, selection, mutation, crossover, and termination. For each of these modules, the algorithm designer must make a number of design decisions. I will give a few more details of these design decisions below, to indicate what is involved.

Initialization: Any evolutionary algorithm works with a collection of individuals, each of which exemplifies some setting of the chosen variables for the problem, and hence is associated with a numerical value by the fitness function. At the start of any run of the algorithm, this population needs to be initialized, by choosing the individuals that make up the initial population. One decision that has to be made is the size of the

population, and there is very little guidance in the literature on how to select this value. In many implementations, the initial individuals are then chosen at random, often with a uniform probability distribution across the entire search space. This may appear to be making few assumptions and injecting the minimum amount of information at the start, but in fact it is already giving the algorithm a significant advantage by sampling from widely different regions of the search space. Jansen notes that "many evolutionary algorithms suffer if the members of a population are too similar to each other" (Jansen 2013). In other words, the algorithm is less likely to work if it is started without a great deal of variety already present in the initial population.

Selection: The evolutionary algorithm will repeatedly select individuals to reproduce, and select individuals to replace, and hence the population will change over a series of repeated cycles or generations. Selection is usually based on the fitness of the individuals (although Jansen notes that some evolutionary algorithms "do additionally take other properties of the individuals or even the population into account"). However, the way in which fitness values are used to select individuals has to be specified. One way is to use the values of the fitness function to build a probability distribution, and then select individuals at random based on this distribution. For example, we can arrange that an individual is selected with probability proportional to its fitness value.[3] One problem with this approach is that it depends very strongly on the numerical values of the fitness function—for example, if we add a constant to all the fitness values (which is not a significant change in terms of finding the optimum), then the probability values change. Hence, it is common to use more sophisticated selection mechanisms, such as ranking the entire population, and then selecting a fixed number of them. Often specific tweaks are also introduced to the selection mechanism to try to increase the diversity in the resulting population and prevent "premature convergence." In summary, the algorithm designer must arrange for individuals to be selected to reproduce and individuals to be selected to be replaced, in such a way that the population maintains a suitable size and a good spread of values for the variables. For challenging problems, only carefully chosen selection schemes are effective (Watson 2006).

Mutation: Once individuals have been selected for reproduction, the algorithm may introduce the possibility of small changes occurring; these are usually implemented as random changes to individual variables selected at random. However, a key parameter to be specified is the mutation rate, or the expected number of changes in each generation. If this is too high or too low the evolutionary algorithm will not perform well.

Crossover: The idea of crossover is to generate one or more offspring that are in some way similar to their parents. If the parents are represented by a list of variable values, then these lists of values have to be combined in

some way to generate offspring individuals that can be inserted into the population. There are many ways of combining two lists of values, but a technique that is commonly used in evolutionary algorithms, based on the analogy with sexual reproduction in biology, is to switch between the values of two parents at a small number of crossover points. (Note that this requires choosing a specific ordering for the variables, which may not be part of the problem specification, and can be done in many different ways, some of which will work better than others (Watson 2006).) The number of crossover points, the number of offspring, and even the number of parents to combine, are all parameters that must be chosen by the algorithm designer. Combining elements from individuals at different points in the fitness landscape is one way that an evolutionary algorithm can move individuals in the population away from a local peak to another position with a higher fitness value, which is why such a complex mechanism of deriving offspring is included.

Termination: Some evolutionary algorithms may be designed to run indefinitely, but in most cases there is a need to halt execution at some point, and return an answer. This may be as simple as running for a fixed number of generations, or it may be that the algorithm runs until some property of the population is achieved. If the aim is to find a combination of values for the variables that maximizes the fitness function, then the algorithm will return an individual that has the maximum fitness value out of all the individuals considered during the run. However, it should be noted that this individual may well not be present in the final population—evolutionary algorithms often select the highest-fitness individuals from the population to reproduce and replace them with their offspring, but the offspring may be less fit than their parents, and such high-fitness individuals may not arise again.

The above description should give some indication of the decisions that need to be made in order to implement an evolutionary algorithm. There are many parameters to be chosen, including the population size, the selection rule, the mutation rate, the number of crossover points, and so on.

> The principal challenge for evolutionary algorithm designers is that the design details, i.e., parameter values, have such a large influence on the performance of the algorithm. Hence, the design of algorithms in general, and Evolutionary Algorithms in particular, is an optimization problem itself. (Eiben and Smith 2015, 121, acronym expanded)

But it is not enough to solve that optimization problem and find good initial values for the parameters. Jansen comments that:

> There are different ways of setting the parameters in evolutionary algorithms. the simplest and most common way is to set all parameters to some fixed values in advance and keep the parameters unchanged during the run.

> Usually, one experiments a little to find settings that lead to acceptable performance. *It is not clear however that such static parameter settings are able to deliver satisfactory results at all.* (Jansen 2013, 18, italics added)

In other words, some additional feedback mechanisms must generally be added to modify the properties of the evolutionary algorithm in a goal-directed way during a run, based on some information about its current performance. Jansen notes that typical examples of such information include "the current population, the population's distribution in the search space, the current fitness values, or measures based on a history of the population."

I hope this brief excursion into the design of evolutionary algorithms makes clear that, once we get into fleshing out the necessary details, we find ourselves a long way from a self-evident, automatic process. There is now a very large and growing literature within computer science on the design of evolutionary algorithms, and this seems to indicate that successful algorithms, even evolutionary ones, need to be carefully designed. Presumably, all the authors of such articles believe that they have contributed something, and that their designs were not the inevitable outworkings of some automatic mechanism.

In summary, I claim that the experience of developing evolutionary algorithms in computer science provides a new perspective on the existence of a well-functioning evolutionary process in nature.

> Over recent decades the Evolutionary Computation community shifted from believing that Evolutionary Algorithm performance is to a large extent independent from the given problem instance to realising that it is [dependent]. In other words*, it is now acknowledged that Evolutionary Algorithms need more or less fine-tuning to specific problems and problem instances*. (Eiben and Smith 2015, 146, acronyms expanded, italics added)

The Case of Biological Evolution

Biological evolution is not generally viewed as an example of an algorithm solving a predefined problem, but it may be helpful to ask whether the experience of developing and using evolutionary algorithms for solving problems in other settings can help us to appreciate important aspects of biological evolution. Kojonen notes that there have been many attempts to obtain insights about biological evolution from studies of evolutionary algorithms in general (Kojonen 2021) and I think this approach can be valuable.

Certainly biological evolution is often thought of as exploring a problem space, where the individual points are represented by *genotypes*, which are combinations of values for variables—either combinations of allele values for a list of genes, or combinations of base pairs in a DNA sequence.

Of course there are a number of significant contrasts between the standard evolutionary algorithms of computer science, as described earlier, and the case of biological evolution (Eiben and Smith 2015, 45). The biological notion of "fitness" is not some predefined function of the genotype, but is an *a posteriori* measure of differential survival and reproduction rates associated with different organismal *phenotypes*. These phenotypes result from a highly complex biochemical process of development, influenced by both the genotype and the environment. The differential survival and reproduction rates for different phenotypes then arise from complex interactions between organisms and their environments, including other organisms. Hence the fitness values in biological evolution are generally stochastic quantities that may vary with time, and with changes in the environment (Nichol et al. 2016).

However, the notion of a population of organisms occupying various positions within a fitness landscape is still a helpful model, at least for studies of evolutionary processes over short time periods in reasonably stable environments. In such cases, it still makes sense to ask what properties does such a fitness landscape need to have in order for an evolutionary mechanism to result in the arrival of more complex or well-adapted organisms over time.

Theoretical studies have shown that the potential effectiveness of simple evolutionary mechanisms depends critically on the properties of such fitness landscapes. For example, Wilf and Ewens consider fitness landscapes associated with sequences of alleles for different genes and assume that the optimal allele can be set for each gene separately, as in Dawkins' "WEASEL" example; they conclude that in such cases a simple evolutionary algorithm can find an optimal combination of alleles in a reasonable amount of time (Wilf and Ewens 2010). In contrast, Chatterjee, Pavlogiannis, Adlam and Nowak (2014) consider fitness landscapes associated with sequences of base-pairs for a single gene and assume that the fitness of a sequence only begins to increase when a significant fraction of the base-pairs in the sequence are correctly chosen; they show that in such cases simple evolutionary algorithms cannot find high-fitness sequences in a feasible amount of time.

Questions about necessary properties of biological fitness landscapes are discussed in some detail in Kojonen (2021). For example, the question of whether functional protein sequences are sufficiently prevalent and sufficiently clustered to enable new ones to be successively discovered by an evolutionary search process within a reasonable timescale. Such questions are likely to become increasingly prominent now that increasing attempts are being made to measure biological fitness landscapes empirically. Initial results of such studies seem to suggest that many empirical fitness landscapes are highly rugged, that is, they have many local peaks (Obolski, Ram, and Hadany 2017). In fact, it has been suggested that such

properties, and the tight constraints they impose on possible evolutionary trajectories, can be used to develop ways of steering evolution to particular outcomes, such as preventing the emergence of antibiotic resistance (Nichol et al. 2015).

Another contrast between biological evolution and evolutionary algorithms in other contexts is the astonishing scale of the biological problem space. Most organisms have very large numbers of genes (many thousands) and extraordinarily large numbers of base-pairs in the genotype (many millions or billions). This is way beyond the scope of evolutionary algorithms applied to other problems, which typically have at most a few tens of variables. No-one would contemplate attempting to use an evolutionary algorithm in any other context for problems of the scale of genotypes or even the coding sequences for individual genes. Even with the ability to simulate large populations over billions of generations, current evolutionary algorithms are only applied to problems with a few tens of variables, and we have seen that even then it is acknowledged that they generally need considerable tailoring and adjustment to make significant progress. The enormous gulf in scale between what is achievable by evolutionary algorithms in computer science and in biology suggests there must be something rather unusual about biological systems, to allow an evolutionary algorithm to function effectively so far beyond the usual range of applicability.

Some have suggested that the fact that biological fitness arises indirectly from the outworkings of complex developmental processes—the so-called *genotype–phenotype map*—may structure the resulting fitness landscapes in ways that make evolutionary mechanisms unusually effective. For example, Kirschner and Gerhart in *The Plausibility of Life*, argue for a notion they call "facilitated variation" where small genotypic changes can lead to significant variation in the resulting phenotypes, while preserving functionality (Kirschner and Gerhart 2005). Others have suggested that the nature of the genotype–phenotype map may allow exploration of larger regions of the genotype space without affecting fitness, through the presence of large "neutral networks," and hence facilitate the discovery of novel phenotypes (Wagner 2007). Discovering the properties of genotype–phenotype maps is an important research question, not just for understanding the historical evolution of the biosphere, but also for understanding the processes of mutation and selection within individual organisms that drive diseases such as cancer (Nichol et al. 2019).

If the special features of biological fitness landscapes do turn out to be particularly congenial to evolutionary mechanisms, then I suggest that should be seen as a further example of remarkable fine-tuning in the context of biology, and hence strengthen the argument being made by Kojonen.

Conclusion

Kojonen argues, in Chapter 4 of *The Compatibility of Evolution and Design*, that evolution and design do not need to be seen as competing explanations, but can be seen as compatible. I have tried to argue here that the perspective of computer science, in particular the accumulated experience with analyzing, developing and using evolutionary algorithms over the past 50 years, can add additional elements to Kojonen's argument.

Much of the long-running discussion about the compatibility of evolution and design in the context of biology has considered the process of biological evolution "from the outside," without focusing too closely on the details of the evolutionary mechanism. Some see these details as irrelevant to the question of whether organisms give evidence of design. For example, in a detailed review of Darwin's *Origin of Species*, the botanist and theist Asa Gray (1861) writes that "the adoption of … Darwin's particular hypothesis, if we understand it, would leave the doctrines of final causes, utility, and special design just where they were before." Kojonen summarizes the arguments put forward by Gray in the following terms:

> According to Gray, it makes no difference whether the ordered complexity of life is produced through a secondary cause, such as evolution, or through a miraculous act of creation that exceeds the normal operation of nature. In either case, he claims, *design can only be inferred from the purposeful arrangement of the end result*. (Kojonen 2021, 100, italics added)

Such arguments "from the outside," based on the observed features of the biosphere, are important, and are still being ably and persuasively made, for example by Alexander (2018). However, it may now be possible to add another strand to the argument, by considering the details of the evolutionary mechanism "from the inside."

Such arguments have also been attempted for a very long time. Indeed, Kojonen points out that Gray also raised questions of this kind, by questioning the source of positive variation in organisms, and the need for suitable laws and environment within which evolution could operate (Kojonen 2021, 100). However, I think that the study of general evolutionary algorithms in computer science, over the past 50 years or so, now allows such arguments to be made in a more informed and detailed way.

As one example of the way that computer science may bring a distinctive perspective, consider the following quotation from Daniel Dennett, that is cited by Kojonen as an example of the belief that an evolutionary explanation renders design unnecessary:

> Paley was right in saying not just that Design was a wonderful thing to explain, but also that Design took Intelligence. All he missed – and Darwin provided – was the idea that this Intelligence could be broken up into bits so tiny and stupid that they didn't count as intelligence at all, and then

> distributed through space and time in a gigantic connected network of algorithmic process. (Dennett 1995, 133) cited in Kojonen (2021, 103)

As a computer scientist, I see the art of breaking up a task into "bits so tiny and stupid that they didn't count as intelligence" as the fundamental design activity in my professional arena—the art of designing and implementing an algorithm. To orchestrate such "tiny bits" into an overall process that achieves remarkably impressive ends is the core skill of the software engineer.

As I hope I have indicated in this article, this process of algorithm design is not trivial, or automatic, in the case of evolutionary algorithms. I think this adds to the case that simply pointing in the general direction of natural selection is not a sufficient explanation of biological or any other phenomena, unless the algorithmic details can be fleshed out, and it seems to be becoming clearer that fleshing out those details requires a great many sophisticated refinements and careful adjustments. Kojonen notes the trend in evolutionary biology to seek "substantial expansion and modification of evolutionary theory," and we have seen a similar recognition in computer science of the need for substantial refinement to the basic components of evolutionary algorithms.

Moreover, it is becoming clear that even our most carefully crafted evolutionary algorithms only work well in very special circumstances, when the fitness function satisfies very special properties. Kojonen sums up these requirements at a high level as follows:

> In order for evolution to be possible, viable forms must be close enough to each other in the space of possible form, and must form a network that can be navigated by evolutionary search. (Kojonen 2021, 132)

If we discover that the fitness function associated with genome sequences, via the exquisite transformations of each organism's genotype–phenotype map, happens to satisfy these special properties, then that seems to constitute another remarkable example of fine-tuning, which is highly compatible with the idea of overall design. As Kojonen remarks:

> Instead of explaining the appearance of purpose in biology by reference to non-teleological factors, the attempt at explaining design by evolution has succeeded in finding new layers of teleology. The further we study, the more the universe seems to be filled with teleology (Kojonen 2021, 133)

Notes

1. Technically, the set of points in the fitness landscape is the Cartesian product of the domains of the variables.

2. That is, we assume that the fitness function $f(x_1, x_2, \ldots .x_n) = \sum_{i=1}^{n} g_i(x_i)$ for some unary functions g_i. Such functions are sometimes initially expressed as a product of unary functions, but such expressions are easily transformed into a sum by taking logarithms.

3. That is, set the probability of selecting an individual s to $f(s)/\sum_{x \in P} f(x)$, where f is the fitness function, and P is the population.

References

Alexander, Denis. 2018. *Is There Purpose in Biology? The Cost of Existence and the God of Love*. New edition. Oxford: Monarch Books.

Chatterjee, Krishnendu, Andreas Pavlogiannis, Ben Adlam, and Martin A. Nowak. 2014. "The Time Scale of Evolutionary Innovation." *PLOS Computational Biology* 10 (9):e1003818.

Dawkins, Richard. 1986. *The Blind Watchmaker*. Harlow: Longman Scientific & Technical.

Dennett, Daniel C. 1995. *Darwin's Dangerous Idea: Evolution and the Meanings of Life*. First Edition. New York: Simon & Schuster.

Droste, Stefan, Thomas Jansen, and Ingo Wegener. 2002. "Optimization with Randomized Search Heuristics: The (A)NFL Theorem, Realistic Scenarios, and Difficult Functions." *Theoretical Computer Science* 287 (1): 131–44. https://doi.org/10.1016/S0304-3975(02)00094-4.

Eiben, A. E., and J. E. Smith. 2015. *Introduction to Evolutionary Computing*. 2nd ed. 2015 edition. Berlin Heidelberg New York Dordrecht London: Springer.

Goldberg, David E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. 1st ed. USA: Addison-Wesley Longman Publishing Co., Inc.

Gray, Asa. 1861. *Natural Selection Not Inconsistent with Natural Theology: A Free Examination of Darwin's Treatise on the Origin of Species, and of Its American Reviewers*. Reprinted from the Atlantic Monthly for July, August, and October, 1860. Trübner & Company.

Holland, John H. 1975. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI: University of Michigan Press.

Jansen, Thomas. 2013. *Analyzing Evolutionary Algorithms: The Computer Science Perspective*. Springer Science & Business Media.

Kaznatcheev, Artem. 2019. "Computational Complexity as an Ultimate Constraint on Evolution." *Genetics* 212 (1): 245–65. https://doi.org/10.1534/genetics.119.302000.

Kaznatcheev, Artem, David Cohen, and Peter Jeavons. 2020. "Representing Fitness Landscapes by Valued Constraints to Understand the Complexity of Local Search." *Journal of Artificial Intelligence Research* 69: 1077–102.

Kirschner, Marc W., and John C. Gerhart. 2005. *The Plausibility of Life: Resolving Darwin's Dilemma*. New Haven, CT: Yale University Press.

Kojonen, E. V. R. 2021. *The Compatibility of Evolution and Design*. Cham, Switzerl: Palgrave Macmillan.

Lewis, Geraint F., and Luke A. Barnes. 2016. *A Fortunate Universe: Life in a Finely Tuned Cosmos*. Cambridge: Cambridge University Press.

Nichol, Daniel, Peter Jeavons, Alexander G. Fletcher, Robert A. Bonomo, Philip K. Maini, Jerome L. Paul, Robert A. Gatenby, Alexander R. A. Anderson, and Jacob G. Scott. 2015. "Steering Evolution with Sequential Therapy to Prevent the Emergence of Bacterial Antibiotic Resistance." *PLOS Computational Biology* 11 (9): e1004493.

Nichol, Daniel, Mark Robertson-Tessi, Alexander R. A. Anderson, and Peter Jeavons. 2019. "Model Genotype–Phenotype Mappings and the Algorithmic Structure of Evolution." *Journal of The Royal Society Interface* 16 (160): 20190332.

Nichol, Daniel, Mark Robertson-Tessi, Peter Jeavons, and Alexander R.A. Anderson. 2016. "Stochasticity in the Genotype-Phenotype Map: Implications for the Robustness and Persistence of Bet-Hedging." *Genetics* 204 (4): 1523–39.

Obolski, Uri, Yoav Ram, and Lilach Hadany. 2017. "Key Issues Review: Evolution on Rugged Adaptive Landscapes." *Reports on Progress in Physics* 81 (1): 012602. https://doi.org/10.1088/1361-6633/aa94d4.

Richter, Hendrik, and Andries Engelbrecht, eds. 2013. *Recent Advances in the Theory and Application of Fitness Landscapes*. Springer.

Wagner, Andreas. 2007. Robustness and Evolvability in Living Systems. https://press.princeton.edu/books/paperback/9780691134048/robustness-and-evolvability-in-living-systems.

Watson, Richard A. 2006. "Compositional Evolution: The Impact of Sex, Symbiosis, and Modularity on the Gradualist Framework of Evolution." *Vienna Series in Theoretical Biology* Cambridge, MA: MIT Press.
Wilf, Herbert S., and Warren J. Ewens. 2010. "There's Plenty of Time for Evolution." *Proceedings of the National Academy of Sciences* 107 (52): 22454–56.
Wolpert, D.H., and W.G. Macready. 1997. "No Free Lunch Theorems for Optimization." *IEEE Transactions on Evolutionary Computation* 1 (1): 67–82.
Wright, Sewall. 1932. "The Roles of Mutation, Inbreeding, Crossbreeding and Selection in Evolution." *Proceedings of the Sixth International Congress of Genetics* 1:356–66.
———. 1935. "Evolution in Populations in Approximate Equilibrium." *Journal of Genetics* 30 (2): 257.