there is no scaling constant at the beginning—the scaling constant is simply one.

Therefore, active information fulfills the requirements for a canonical specified complexity model.

Generalized information (GI) measures the amount of generalization that a model achieves for a dataset. GI allows for judging machine learning models in the face of model errors by using active information as a discounting mechanism (Bartlett and Holloway, 2019). The simplified form of GI is simply the difference between the active information and the program length in bits (L).

$$I_G = I_+ - L. \tag{3}$$

We can rewrite this as follows:

$$\begin{aligned} I_G &= -\log_2\left(\frac{p(x)}{q(x)}\right) - \log_2(2^L), & (4)\\ &= -\log_2\left(\frac{p(x)}{q(x)}\right) - \log_2(2^L), & (5)\\ &= -\log_2\left(2^L\frac{p(x)}{q(x)}\right). & (6) \end{aligned}$$

As you can see, 2^L serves as the scaling constant r. As mentioned before, $\int_X \mathrm{d}q(x) = 1$. Since any program will have a length $L \geq 0$, this means that $2^L \geq 1$. This means that GI also serves as a specified complexity model.

Bartlett, J and E Holloway (2019). “Generalized Information: A Straightforward Method for Judging Machine Learning Models”. In: *Communications of the Blyth Institute* 1.2, pp. 13–21. DOI: `10.33014/issn.2640-5652.1.2.bartlett.1`.

Dembski, W A and R J Marks II (2009). “Conservation of Information in Search: Measuring the Cost of Success”. In: *IEEE Transactions on Systems, Man and Cybernetics A, Systems & Humans* 5.5, pp. 1051–1061. DOI: `10.1109/TSMCA.2009.2025027`.

Montañez, G D (2018). “A Unified Model of Complex Specified Information”. In: *BIO-Complexity* 2018 (4), pp. 1–26. DOI: `10.5048/BIO-C.2018.4`.

Evolution in the Valley of Illusions

Eric Holloway

DOI: 10.33014/issn.2640-5652.2.2.holloway.3

Evolutionary algorithms are inspired by the biological theory of evolution. These kinds of algorithms are called optimization algorithms. They are meant to find ’good enough’ solutions to problems by searching through a combination of possible solutions and evaluating each solution by some objective function.

The two tenets of evolutionary theory is that change to the genome are performed without information about what makes a fit organism. The adjustment for fitness is made by the environment selecting organisms that manage to survive and reproduce. This simple procedure is assumed to be responsible for the extraordinarily complex biological organisms we see all around us (including ourselves) that greatly exceed anything humans can engineer. In fact, many cutting edge human inventions, such a sonar, electricity and motors have already been invented many millions of years ago by evolution. Additionally, when consideration of time and oportunities to evolve have been taken into account, the equivalent computation is not too far out of reach of our modern super computers. Thus, it would seem that we almost automatically create phenomenal inventions automatically just by copying the simple evolutionary process.

With Mendel’s discovery of genes, this simplified the situation even more. Instead of having to deal with messy analogue systems, evolution was reduced to variations on discrete modules. These modules were simplified even more through the discovery of DNA, and the institution of the fundamental dogma, which is that information only flows from the DNA sequence to the organism. The organism is said to have no ability to reverse the flow of information to modify its DNA in anticipation of environmental changes. This is a restatement of Darwin’s notion that variation is random in that it occurs without any foresight.

We now know the fundamental dogma is not quite airtight, and is in fact fairly leaky. The discovery of horizontal gene transfer shows that the DNA can be edited directly in a variety of different ways. So, there are potential ways for the flow of information to be reversed back into the genome. For example, at the RNA level, viruses frequently modify their own genetic code, and this mechanism allows them to adapt to new species through a process known as zoonosis.

A Simple Evolutionary Algorithm

But, let's return back to evolutionary algorithms. These algorithms are based on the original Darwinistic formulation that mutation occurs randomly. The potential solutions are usually represented as binary strings, but sometimes the encoding is much more sophisticated. The algorithm usually use some mechanism that is considered to be evolutionary, such as randomly mutating bits in the binary string, combining multiple strings with crossover, or even some variants of horizontal gene transfer. Then, once a new set of candidate solutions are created, their 'fitness' is measured with the objective function, and then a subset of the solutions are selected based on fitness for the next round.

As an example, Figure 1 shows a very simple evolutionary algorithm written in Python, used to solve a bin packing problem. This particular problem has useful applications. For instance, if you want to optimize what TODO tasks to do, where each task has a time cost and a value associated, and there is a limited amount of time to do the tasks. We can also imagine the evolutionary utility of solving this problem. When being chased by a predator, an animal needs to correctly rank its various actions appropriately to maximize survival probability. If the animal decides to perform an elaborate mating dance while being chased, it is a goner. So, this simple problem actually turns out to be key for the reproductive survival that drives evolution.

Building Block Hypothesis

John Holland, the inventor of the "genetic algorithm," came up with a theory as to why these algorithms work. He thought that good solutions could be built up from worse solutions through the composition of "building blocks." The thought was that the reliance on building blocks reduced the search space and thus allowed the algorithm to find good solutions more quickly. However, as the no free lunch theorem proves, while this hypothesis is valid for some scenarios, it is not valid in general, and building blocks in fact can bias an algorithm towards bad solutions.

Deceptive Landscapes

This problem of bias towards bad solutions is known as the "deceptive landscape." In this scenario, the evolutionary algorithm is lead along a "road" of incrementally better and better solutions by minor modifications to previous solutions. However, eventually the road leads to a cliff, where any modification to the solution drastically lowers the solution fitness, resulting in a halt to any further searching.

This is related to the problem of local optima, where if we envision the search landscape as a real, physical landscape, with hills and valleys, the search process can be a ball we are trying to roll to the lowest part on the landscape, but it can get stuck in dips that are still fairly high up on the hills. The usual approach to solve this problem is to jostle the ball a bit to get it out of the dip. The deceptive road is a variant where the ball falls into a really deep pit such that no amount of jostling can get it out, and the pit is still really high up on the hill.

Here is one example of a deceptive landscape that we can plug into our evolutionary algorithm. 99 options receive a reward of 1, but the last option gets a reward of 10,000,000. The last option is the only option needed, but we will see evolution drives the solution to ignore the enormous reward in favor of a more piddly reward. You can see the result in Figure 2.

```
durations = [1] * 99 + [timeframe]
values = [1] * 99 + [10000000]
```

This landscape makes it almost impossible for the evolutionary algorithm to find the optimal solution since the structure of the landscape biases the algorithm to add more elements to the collection when what it should be doing is removing all elements except the element with the highest score.

What Does This Mean?

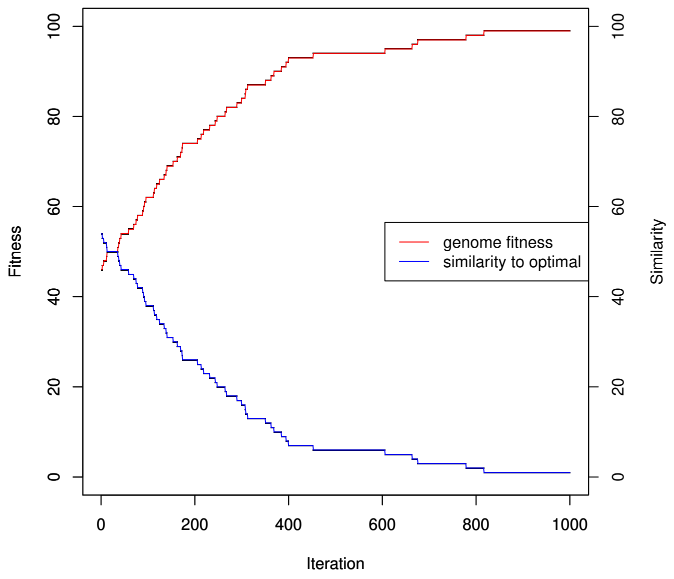
In arguments for evolution, it is often assumed that as long as some genetic code generates a functional organ important for survival, the environment will select towards that code, and with enough opportunities the organism's genome will make all the necessary random baby steps to get to the target. Richard Dawkins makes this argument in *The Blind Watchmaker*.

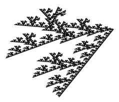
On the other hand, this article shows that "survival benefit = generated by evolution" is not a forgone conclusion. One might think that since animals evolved then what we observe is merely the successful paths. But, this makes the fallacy of affirming the consequent, that because A implies B, then observing B implies A. We can only draw this conclusion if evolution is the only possible explanation; essentially a tautology. However, if we start off with the *a priori* assumption that evolution is the only possible explanation, then we have crossed over from the realm of science to dogmatism.

Figure 1: Python Evolutionary Algorithm

```
from random import randint

cnt = 100
durations = [randint(1, 10) for _ in range(cnt)]
values = [randint(1, 10) for _ in range(cnt)]

timeframe = 400
def fitness(solution):
    total_duration = 0
    for i, d in zip(solution, durations):
        if i == 1:
            total_duration += d
    if total_duration > timeframe: return -1
    total_value = 0
    for i, v in zip(solution, values):
        if i == 1:
            total_value += v
    return total_value

def mutate(individual):
    pos = randint(0, len(individual)-1)
    new_ind = [v for v in individual]
    new_ind[pos] = randint(0, 1)
    return new_ind

genome = [randint(0, 1) for _ in range(cnt)]

print('start fitness: ' + str(fitness(genome)))

iterations = 300
for _ in range(iterations):
    new_genome = mutate(genome)
    if fitness(new_genome) > fitness(genome):
        genome = new_genome

print('ending fitness: ' + str(fitness(genome)))
```

Figure 2: A Deceitful Landscape





Is Active Information Applicable to Biology?

Jonathan Bartlett

DOI: 10.33014/issn.2640-5652.2.2.bartlett.2

Active information was originally introduced in 2009 by William Dembski and Robert Marks II (Dembski and Marks II, 2009). Active information identifies how much information that a search has compared to a "random search." Introduced in the context of information science, it was originally utilized towards identifying information sources in various computerized forms and simulations of evolution.

The reason why active information works is because there is no general "best" algorithm for searching. A search that is good in one context will be terrible in another. There might be a best search *for a particular situation*, but not one that serves all situations equally. In fact, it turns out that, for any particular search situation, a random search has *average* performance characteristics compared to any other search algorithm.

Therefore, for any search situation, we have the capability of determining what the average success rate for a search should be (note that the success rate is how many times the search algorith has to "look" before finding a successful hit). If we simply perform a random search and measure successes, we can determine the average value.

In terms of statistics, this average value is the expected value for the success rate for a search strategy chosen arbitrarily. That is, if the search strategy is chosen arbitrarily, we would expect that the success rate should be roughly equivalent to that of a random search.

Active information measures the distance between the success rate that we actually observe and the success rate that we would expect from an arbitrarily chosen search strategy. If this is measured prior to selection affecting the success rate, we can then measure the distance between the success rate that the cell's own mutational machinery is having and the success rate that we would expect from arbitrary mutation strategies. This will tell us the amount of information that the cell's mutational machinery has for finding a solution in a given selective process.

Recently, I demonstrated how this could be measured in biological systems, giving examples for how different types of systems might be measured (Bartlett, 2020). Since this is a fairly new approach for thinking about mutations in the genome, there are many confusions about what is actually being claimed and proposed. This note intends to clarify, explain, and defend the notions presented in the paper.

Addressing Misconceptions

I want to start by clarifying that active information does not (a) hold that mutations form a uniform random distribution, (b) hold that mutations *should* form a uniform random distribution, or (c) hold that standard evolutionary theory holds that mutations should form a uniform random distribution. Instead, active information attempts to simulate a uniform random distribution of mutations *in order to get an expected value* for the success rate of other mutational strategies. This follows not from evolutionary theory but rather from information theory, which states that such a search *will give you the expected value* for the success rate of *other* searches. This distinction is critical and forms the basis of the logic of applying active information to biology.

Another important clarification is that, as stated in the paper, it does not matter if evolution is *ontologically* a search.