GENETIC ALGORITHMS: Evolution-search spaces and fitness landscapes

In evolutionary biology, fitness landscapes or adaptive landscapes (types of evolutionary landscapes) are used to visualize the relationship between genotypes and reproductive success. It is assumed that every genotype has a well-defined replication rate (often referred to as fitness). This fitness is the "height" of the landscape. Genotypes which are similar are said to be "close" to each other, while those that are very different are "far" from each other. The set of all possible genotypes, their degree of similarity, and their related fitness values is then called a fitness landscape.

The idea of a fitness landscape is a metaphor to help explain flawed forms in evolution by natural selection, including exploits and glitches in animals like their reactions to supernormal stimuli. The idea of studying evolution by visualizing the distribution of fitness values as a kind of landscape was first introduced by Sewall Wright in 1932.

In evolutionary optimization problems, fitness landscapes are evaluations of a fitness function for all candidate solution. In all fitness landscapes, height represents and is a visual metaphor for fitness. There are three distinct ways of characterizing the other dimensions, though in each case distance represents and is a metaphor for degree of dissimilarity.

Fitness landscapes are often conceived of as ranges of mountains. There exist local peaks (points from which all paths are downhill, i.e., to lower fitness) and valleys (regions from which many paths lead uphill). A fitness landscape with many local peaks surrounded by deep valleys is called rugged. If all genotypes have the same replication rate, on the other hand, a fitness landscape is said to be flat. An evolving population typically climbs uphill in the fitness landscape, by a series of small genetic changes, until – in the infinite time limit – a local optimum is reached.

Note that a local optimum cannot always be found even in evolutionary time: if the local optimum can be found in a reasonable amount of time then the fitness landscape is called "easy" and if the time required is exponential then the fitness landscape is called "hard". Hard landscapes are characterized by the maze-like property by which an allele that was once beneficial becomes deleterious, forcing evolution to backtrack. However, the presence of the maze-like property in biophysically inspired fitness landscapes may not be sufficient to generate a hard landscape.

Genotype to fitness landscapes

Wright visualized a genotype space as a hypercube. No continuous genotype "dimension" is defined. Instead, a network of genotypes is connected via mutational paths. Stuart Kauffman's NK model falls into this category of fitness landscape. Newer network analysis techniques such as selection-weighted attraction graphing (SWAG) also use a dimensionless genotype space.

Allele frequency to fitness landscapes

Wright's mathematical work described fitness as a function of allele frequencies. Here, each dimension describes an allele frequency at a different gene, and goes between 0 and 1.

Phenotype to fitness landscapes

In the third kind of fitness landscape, each dimension represents a different phenotypic trait. Under the assumptions of quantitative genetics, these phenotypic dimensions can be mapped onto genotypes. See the visualizations below for examples of phenotype to fitness landscapes.

In evolutionary optimization

Apart from the field of evolutionary biology, the concept of a fitness landscape has also gained importance in evolutionary optimization methods such as genetic algorithms or evolution strategies. In evolutionary optimization, one tries to solve real-world problems (e.g., engineering or logistics problems) by imitating the dynamics of biological evolution. For example, a delivery truck with a number of destination addresses can take a large variety of different routes, but only very few will result in a short driving time.

In order to use evolutionary optimization, one has to define for every possible solution *s* to the problem of interest (i.e., every possible route in the case of the delivery truck) how 'good' it is. This is done by introducing a scalar-valued function *f(s)* (scalar valued means that *f(s)* is a simple number, such as 0.3, while *s* can be a more complicated object, for example a list of destination addresses in the case of the delivery truck), which is called the fitness function.

A high *f(s)* implies that *s* is a good solution. In the case of the delivery truck, *f(s)* could be the number of deliveries per hour on route *s*. The best, or at least a very good, solution is then

found in the following way: initially, a population of random solutions is created. Then, the solutions are mutated and selected for those with higher fitness, until a satisfying solution has been found.

Evolutionary optimization techniques are particularly useful in situations in which it is easy to determine the quality of a single solution, but hard to go through all possible solutions one by one (it is easy to determine the driving time for a particular route of the delivery truck, but it is almost impossible to check all possible routes once the number of destinations grows to more than a handful).

The concept of a scalar valued fitness function *f(s)* also corresponds to the concept of a potential or energy function in physics. The two concepts only differ in that physicists traditionally think in terms of minimizing the potential function, while biologists prefer the notion that fitness is being maximized. Therefore, taking the inverse of a potential function turns it into a fitness function, and vice versa

Caveats and limitations

Several important caveats exist. Since the human mind struggles to think in greater than three dimensions, 3D topologies can mislead when discussing highly multi-dimensional fitness landscapes. In particular it is not clear whether peaks in natural biological fitness landscapes are ever truly separated by fitness valleys in such multidimensional landscapes, or whether they are connected by vastly long neutral ridges. Additionally, the fitness landscape is not static in time but dependent on the changing environment and evolution of other genes. It is hence more of a seascape, further affecting how separated adaptive peaks can actually be. Additionally, it is relevant to take into account that a landscape is in general not an absolute but a relative function. Finally, since it is common to use function as a proxy for fitness when discussing enzymes, any promiscuous activities exist as overlapping landscapes that together will determine the ultimate fitness of the organism, implying a gap between different coexisting relative landscapes.

With these limitations in mind, fitness landscapes can still be an instructive way of thinking about evolution. It is fundamentally possible to measure (even if not to visualise) some of the

parameters of landscape ruggedness and of peak number, height, separation, and clustering. Simplified 3D landscapes can then be used relative to each other to visually represent the relevant features. Additionally, fitness landscapes of small subsets of evolutionary pathways may be experimentally constructed and visualized, potentially revealing features such as fitness peaks and valleys. Fitness landscapes of evolutionary pathways indicate the probable evolutionary steps and endpoints among sets of individual mutations.

INTRODUCTION

Genetic Algorithms were invented to mimic some of the processes observed in natural evolution. The idea with GA is to use this power of evolution to solve optimization problems. The father of the original Genetic Algorithm was John Holland who invented it in the early 1970's. Genetic Algorithms (GAs) are adaptive heuristic search algorithm based on the evolutionary ideas of natural selection and genetics.

They are based on the Darwin's theory of evolution which stressed the fact that the existence of all living things is based on the rule of 'survival of the fittest.' Darwin also postulated that new breeds or classes living things come into existence through the process of reproduction, crossover and mutation among existing organisms.

The following table presents a list of different expressions, which are common in genetics, along with their equivalent in the framework of GAs:

Genetics	Genetic algorithm -
Genotype genotype phenotype	coded string uncoded point
chromosome gene allele fitness	string string position value at a certain position objective function value

The basic structure of a genetic algorithm

t := 0;

Compute initial population B0;

WHILE stopping condition not fulfilled **DO**

BEGIN

select individuals for
reproduction; create
offsprings by crossing
individuals; eventually
mutate some individuals;
compute new generation

END

As obvious from the above algorithm, the transition from one generation to the next consists of four basic components:

Selection: Mechanism for selecting individuals (strings) for reproduction according to their fitness (objective function value).

Crossover: Method of merging the genetic information of two individuals; if the coding is chosen properly, two good parents produce good children.

Mutation: In real evolution, the genetic material can by changed randomly by erroneous reproduction or other deformations of genes,

e.g. by gamma radiation. In genetic algorithms, mutation can be realized as a random deformation of the strings with a certain probability. The positive effect is preservation of genetic diversity and, as an effect, that local maxima can be avoided.

Sampling: Procedure which computes a new generation from the previous one and its offsprings.

The steps involved in GAs

The steps involved in creating and implementing a genetic algorithm are:

1. Generate an initial, random population of individuals for a fixed size.

2. Evaluate their fitness.

3. Select the fittest members of the population.

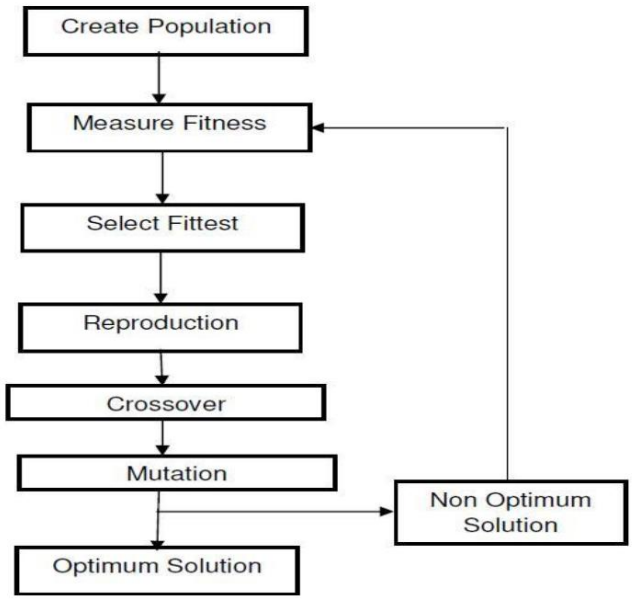
4. Reproduce using a probabilistic method (e.g., roulette wheel).

5. Implement crossover operation on the reproduced chromosomes

6. Execute mutation operation with low probability.

7. Repeat step 2-6 until a predefined convergence criterion is met.

Flow chart



fitness Function

A fitness function is a particular type of objective function that is used to summarize how close a given design solution is to achieving the set aims. After each round of testing, or simulation, the idea is to delete the 'n' worst design solutions, and to breed 'n' new ones from the best design solutions.

The fitness function must not only correlate closely with the designer's goal, it must also be computed quickly. Speed of execution is very important, as a typical genetic algorithm must be iterated many times in order to produce a usable result for a non-trivial problem.

Fitness approximation may be appropriate, especially in the following cases:

- Fitness computation time of a single solution is extremely high
- Precise model for fitness computation is missing
- The fitness function is uncertain or noisy.

Reproduction

During the reproduction phase, the fitness value of each chromosome is assessed. This value is used in the selection process to provide bias towards fitter individuals. Just like in natural evolution, a fit chromosome has a higher probability of being selected for reproduction.

There are four common methods for selection are:

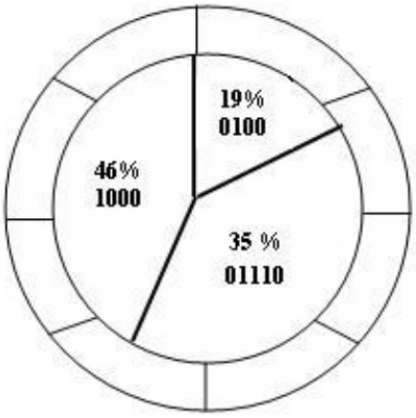
1. Roulette Wheel selection

2. Stochastic Universal sampling

3. Normalized geometric selection

4. Tournament selection

An example of a common selection technique is the ‘Roulette Wheel’ selection method, as shown in Figure. Each individual in the population is allocated a section of a roulette wheel; the size of the section is proportional to the fitness of the individual. A pointer is spun and the individual to whom it points is selected. This continues until the selection criterion has been met. The probability of an individual being selected is thus related to its fitness, ensuring that fitter individuals are more likely to leave offspring. Multiple copies of the same string may be selected for reproduction and the fitter strings should begin to dominate. However, for the situation illustrated in Figure it is not implausible for the weakest string (01001) to dominate the selection process.



REFERENCE

Eiben, A. E. et al (1994). "Genetic algorithms with multi-parent recombination". PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature: 78–87. ISBN 3-540-58484-6.

Ting, Chuan-Kang (2005). "On the Mean Convergence Time of Multi-parent Genetic Algorithms Without Selection". Advances in Artificial Life: 403–412. ISBN 978-3-540-28848-0.

Harik, Georges R.; Lobo, Fernando G.; Sastry, Kumara (1 January 2006). Linkage Learning via Probabilistic Modeling in the Extended Compact Genetic Algorithm (ECGA). Scalable Optimization Via Probabilistic Modeling. Studies in

Janikow, C. Z.; Michalewicz, Z. (1991). "An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms" (PDF). Proceedings of the Fourth International Conference on Genetic Algorithms: 31–36. Retrieved 2 July 2013.

Baluja, Shumeet; Caruana, Rich (1995). Removing the genetics from the standard genetic algorithm (PDF). ICML.

Srinivas, M.; Patnaik, L. (1994). "Adaptive probabilities of crossover and mutation in genetic algorithms" (PDF). IEEE Transactions on System, Man and Cybernetics.

Noetel M, Ciarrochi J, Sahdra B, Lonsdale C (2019). "Using genetic algorithms to abbreviate the Mindfulness Inventory for Sport: A substantive-methodological synthesis". Psychology of Sport and Exercise. 45 (101545): 101545.

Ferreira, C. "Gene Expression Programming: A New Adaptive Algorithm for Solving Problems" (PDF). Complex Systems, Vol. 13, issue 2: 87-129.

Falkenauer, Emanuel (1997). Genetic Algorithms and Grouping Problems. Chichester, England: John Wiley & Sons Ltd. ISBN 978-0-471-97150-4.

Banzhaf, Wolfgang; Nordin, Peter; Keller, Robert; Francone, Frank (1998). Genetic Programming – An Introduction. San Francisco, CA: Morgan Kaufmann. ISBN 978-1558605107.

Bies, Robert R.; Muldoon, Matthew F.; Pollock, Bruce G.; Manuck, Steven; Smith, Gwenn; Sale, Mark E. (2006). "A Genetic Algorithm-Based, Hybrid Machine Learning Approach to Model Selection". Journal of Pharmacokinetics and Pharmacodynamics.

Cha, Sung-Hyuk; Tappert, Charles C. (2009). "A Genetic Algorithm for Constructing Compact Binary Decision Trees". Journal of Pattern Recognition Research.

Eiben, Agoston; Smith, James (2003). Introduction to Evolutionary Computing. Springer. ISBN 978-3540401841.

Fraser, Alex S. (1957). "Simulation of Genetic Systems by Automatic Digital Computers. I. Introduction". Australian Journal of Biological Sciences.

Goldberg, David (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley Professional. ISBN 978-0201157673.

Michalewicz, Zbigniew (1996). Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag. ISBN 978-3540606765.

Mitchell, Melanie (1996). An Introduction to Genetic Algorithms. Cambridge, MA: MIT Press. ISBN 9780585030944.

Schmitt, Lothar M. (2001). "Theory of Genetic Algorithms". Theoretical Computer Science. 259 (1–2): 1–61.

Schmitt, Lothar M. (2004). "Theory of Genetic Algorithms II: models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling".

Whitley, Darrell (1994). "A genetic algorithm tutorial" (PDF). Statistics and Computing.