A comparison of *"On Genetic Algorithms and Their Application"* (Yasmin Said) p. 359-360
and *Sources on Genetic Algorithms (Ev. Comp FAQ, M Mitchell, J Holland)*

Regular font indicates substantially close wording between the two sources, *italic* represent paraphrased sections, **bold** represents significant departures of Said from sources, and **bold underline** represent points of outright contradiction. Paragraphs have been reformatted for easy comparison. Within sections of close wording, identical phrases (ID) are highlighted in cyan, trivial changes (TC) with yellow . Changes resulting in issues are underlined.

Yasmin Said, *On Genetic Algorithms and Their Application,* p. 359

Introduction – para 1
Under the umbrella of evolutionary computation (EC) also referred to as evolutionary algorithms (EAs)

are the areas of evolutionary programming (EPs) and evolution strategies (ESs).

Each of these methods of evolutionary computation simulate the process of evolution through the mutation, selection and/or reproduction processes and rely on perceived performance of individual structures assigned by the environment.

Evolutionary algorithms support population structures which progress to the rules of selection using genetic operators.

Genetic operators determine which structures will move on to the next level and which will not.

In essence, the individuals in the population obtain a degree of fitness from the environment. Reproduction concentrates on high fitness individuals.

[Two paragraphs omitted]

Evolutionary Computing FAQ (Intro) [N.B. Capitilized links rendered in lower-case for readability]

Paragraph 1
Evolutionary algorithm is an umbrella term used to describe computer-based problem solving systems which use computational models of some of the known mechanisms of evolution as key elements in their design and implementation. A variety of evolutionary algorithms have been proposed. The major ones are: genetic algorithms (see Q1.1), evolutionary programming (see Q1.2), evolution strategies (see Q1.3), classifier systems (see Q1.4), and genetic programming (see Q1.5).

They all share a common conceptual base of simulating the evolution of individual structures via processes of selection, mutation, and reproduction. The processes depend on ~~the~~ perceived performance of ~~the~~ individual structures as defined by an environment.

Paragraph 2
More precisely, EAs maintain a population of structures, that evolve according to rules of selection ~~and other operators, that are referred to as "search operators",~~ (or genetic operators), ~~such as recombination and mutation~~.

[Not found]

Each individual in the population receives a measure of its fitness in the environment. Reproduction focuses attention on high fitness individuals, ~~thus exploiting (cf. exploitation) the available fitness information.~~

Said, p. 360

History

Paragraph 1
In the middle of the twentieth century some computer scientists worked on evolutionary systems with the notion that this will yield to an optimization mechanism for an array of engineering queries (Mitchell. 1998).

GAs were invented and developed by John Holland, his students and his colleagues at the University of Michigan.

His team's original intentions were not to create algorithms, but instead to determine exactly how adaptation occurs in nature and then develop ways that natural adaptation might become a part of computer systems.

Holland's book Adaptation in Natural and Artificial Systems (1975) set forth the lexicon from which all further dialogue concerning GAs would be developed. In essence, this theoretical framework

provided the point of reference for all work on genetic algorithms up until recently whereupon it has taken on a new direction, given new technology.

GA, he stated, moves one population of bits (chromosomes and genes) to a new population using a type of "natural selection" along with genetic operators of crossover, mutation and inversion (all biological functions).

These operators determine which chromosomes are the fittest and thus able to move on. Although some less fit chromosomes do move forward, on average the most fit chromosomes produce more offspring than their less fit counterparts.

Melanie Mitchell, *An introduction to genetic algorithms*, p. 2.

A BRIEF HISTORY OF EVOLUTIONARY COMPUTATION

Paragraph 1
In the 1950s and the 1960s several computer scientists ~~independently~~ studied evolutionary systems with the idea that evolution could be used as an optimization tool for engineering problems. …

~~Genetic algorithms~~ (GAs) were invented by John Holland ~~in the 1960s~~ and ~~were~~ developed ~~by Holland and~~ his students and colleagues at the University of Michigan ~~in the 1960s and the 1970s~~. ~~In contrast with evolution strategies and evolutionary programming,~~ ~~Holland's~~ original goal was not to design algorithms ~~to solve specific problems~~, but rather to formally study the ~~phenomenon of~~ adaptation as it occurs in nature and to develop ways in which ~~the mechanisms~~ of natural adaptation might be imported into computer systems.

Holland's 1975 book Adaptation in Natural and Artificial Systems presented the genetic algorithm as an abstraction of biological evolution and gave a theoretical framework for adaptation under the GA.

[Not found]

Holland's GA is a method for moving from one population of "chromosomes" (~~e.g., strings of ones and zeros, or~~ "bits") to a new population by using a kind of "natural selection" together with the genetics−inspired operators of crossover, mutation, and inversion. …

The selection operator chooses those chromosomes in the population that will be allowed to reproduce, and on average the fitter chromosomes produce more offspring than the less fit ones.

Biological recombination occurs between these chromosomes and chromosomal inversion further completes the process of providing as many type possible of recombination or "crossover".

This remarkable quality that genetic algorithms have of focusing their attention on the "fittest" parts of a solution set in a population is directly related to their ability to combine strings which contain partial solutions. First, each string in the population is ranked to determine the execution of the tactic that it predetermines. Second, the higher-ranking strings mate in couplets. When the two strings assemble, a random point along the strings is selected and the portions adjacent to that point are swapped to produce two offspring: one which contains the encoding of the first string up to the crossover point, those of the second beyond it, and the other containing the opposite cross.

Biological chromosomes perform the function of crossover when zygotes and gametes meet and so the process of crossover in genetic algorithms is designed to mimic its biological nominative. Successive offspring do not replace the parent strings; rather they replace low-fitness ones, which are discarded information at each generation in order that the population size is maintained.

Crossover exchanges subparts of two chromosomes, roughly mimicking biological recombination between two single−chromosome ("haploid") organisms; mutation randomly changes the allele values of some locations in the chromosome; and inversion reverses the order of a contiguous section of the chromosome, thus rearranging the order in which genes are arrayed.

John H. Holland

This remarkable ability of genetic algorithms to focus their attention on the most promising parts of a solution space is a direct outcome of their ability to combine strings containing partial solutions. First, each string in the population is evaluated to determine the performance of the strategy that it encodes. Second, the higher-ranking strings mate. Two strings line up, a point along the strings is selected at random and the portions to the left of that point are exchanged to produce two offspring: one containing the symbols of the first string up to the crossover point ~~and~~ those of the second beyond it, and the other containing the complementary cross.

Biological chromosomes cross over ~~one another~~ when ~~two~~ gametes meet ~~to form~~ a zygote, and so the process of crossover in genetic algorithms does in fact closely mimic its biological model. The offspring do not replace the parent strings; instead they replace low-fitness strings, which are discarded at each generation so that the ~~total~~ population remains the same size.

References

1. Yasmin H. Said, *Data mining and data visualization*, ed. Calyampudi Radhakrishna Rao, Edward J. Wegman, Jeffrey L. Solka, Elesevier, 2005. Online at: http://books.google.ca/books?id=fEgUjUPCtsEC
2. Evolutionary Computing FAQ (from "An Overview of Evolutionary Computation", *ECML: European Conference on Machine Learning Machine Learning [ECML93]*, 442-459. Online at: http://www.faqs.org/faqs/ai-faq/genetic/part2/section-1.html
3. Melanie Mitchell, *An introduction to genetic algorithms*, MIT Press, 1998. Online at: http://books.google.ca/books?id=0eznlz0TF-IC
4. John H. Holland, "Genetic Algorithms", *Scientific American,* July 1992.. Online at: http://www.fortunecity.com/emachines/e11/86/algo.html