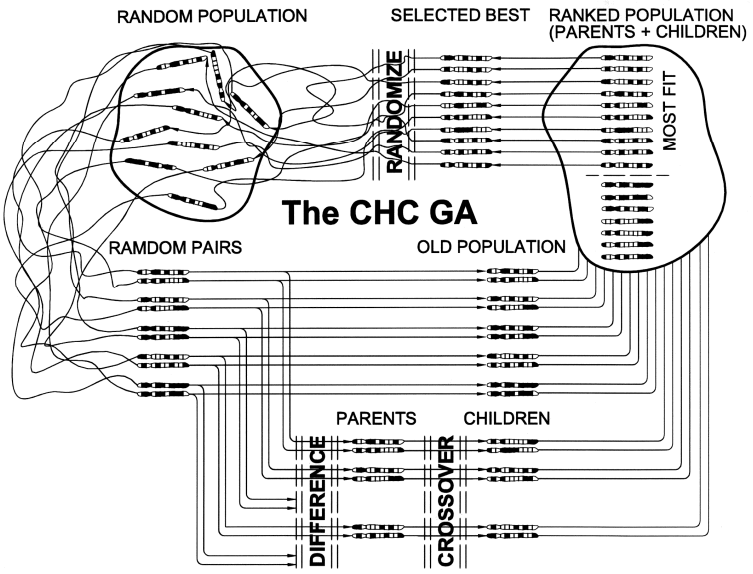
ILEK FORSCHUNGSBERICHT 01|07

A Survey and Comparison of Biological Genetics and Evolutionary Computation

Forschungsbericht erstellt von
Peter von Bülow
Juni 2007



Universität Stuttgart

Institut für Leichtbau Entwerfen und Konstruieren
Prof. Dr.-Ing. Werner Sobek
Prof. Dr.-Ing. Balthasar Novák

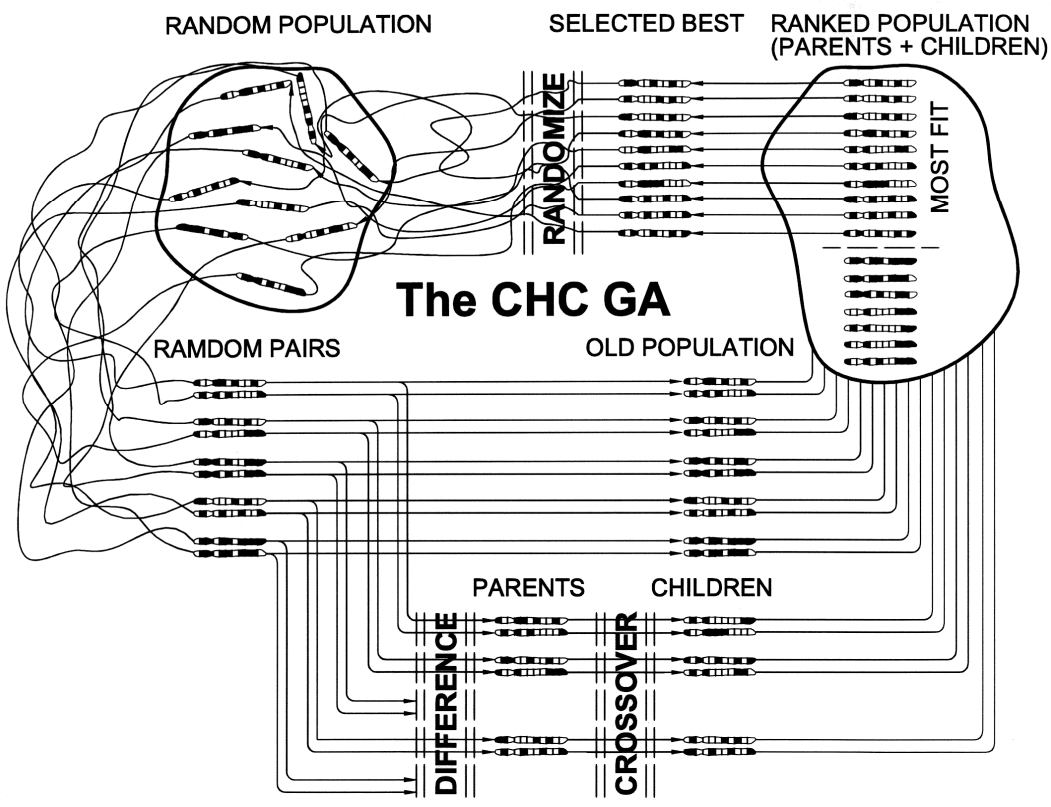
Table of Contents

A Comparison of Biological Genetics with Evolutionary Computation

Introduction

Evolutionary Computation (EC), in its various forms, is a stochastic search method based on an analogy with biological genetics. This report traces some of the parallel aspects between the two areas. An overview is given of classic methods applied in biological genetics, which might be useful to researchers in the area of EC. This includes the sub-disciplines of transmission genetics, molecular genetics, and population genetics.

In addition, an overview is given to general methods applied in EC, as well as specific procedures of Evolutionary Strategies (ES) and Genetic Algorithms (GA). Finally, the mechanics of a specific GA are outlined, the CHC-GA developed by Eshelman and Schaffer.



1 Analogy to Genetic Biology

Since all EC paradigms, and specifically GAs and ESs, draw upon an analogy to biological genetic evolution, it is worthwhile to outline the relevant aspects of biological genetics which may aid in understanding the workings and terminology used in GAs. It should go without saying, that no matter how appropriate or accurate any analogy may be devised, it remains only an analogy - a device to aid in the understanding or conceptualization of the matter at hand. The validity or invalidity of genetic evolution can in no way prove or disprove the effectiveness of EC. In the end, EC must stand or fall on its own merits as stochastic search method, regardless of the validity or lack of validity of analogous systems. None the less, it aids in understanding the terminology and concepts of EC to have a brief overview of the underlying analogy to biology.

1.1 Transmission Genetics

1.1.1 Mendelian Genetics

The modern understanding of genetics finds its roots in the painstaking experiments of the Augustinian monk Gregor Mendel, who in 1865 published the essay "Versuche über Pflanzen-Hybriden" (Experiments in Plant Hybridization). In this, at first neglected work, Mendel described the predictable patterns of inheritance resulting from diploid genetic transmission, which he observed over an eight year period while tracking the results of the careful self and cross pollination of 1000's of pea plants over several generations. By observing the patterns of inheritance of specific traits, Mendel postulated the existence of a "Vermittlungszelle" (transmission cell) which carries a record of the parent's traits and combines in pairs - pollen and ovules - yielding offspring in which the *observed* traits occur in the ratio of 3:1 for dominant (which includes mixed) : recessive traits (Mendel, 1865).

The same terminology used in Mendelian genetics has been adopted in EC. The initial generation which supplies the genetic material is the **parental generation**, P.
The subsequent generations of offspring are called **filial generations** designated F_1, F_2, F_3 ... The realized or apparent character of an individual organism is its **phenotype**. Its genetic character is its **genotype**. For instance, in one of Mendel's

experiments using the pea plant $Pisum\ quadratum$ (Linnaeus), the shape of the peas could be either round and smooth or angular and wrinkled. Mendel determined that the trait for smoothness (designated by A) was **dominant**, while the trait for wrinkledness (designated by a) was **recessive**. By this he meant that if two pure strains where crossed, genotypes which combined the traits (Aa or aA) would have the phenotype of the dominant trait, in this case smooth, A. Mendel's experiments observing different traits always yielded offspring in the ratio of 1:3 (recessive : dominant). In order for a recessive trait to appear as a phenotype, the genotype must be purebred (aa) or **homozygous**, and the individual is called a **homozygote**. A hybrid (Aa) is called a **heterozygote**. A phenotype having recessive traits must be homozygous (aa), but a phenotype with dominant traits might be either homozygous or heterozygous (AA or Aa). Only through further breeding can the genotype be determined in this case.

Mendel's "Vermittlungszelle" described above as AA, aa or Aa is now referred to as a combination of **gene alleles**. The traits which the gene carries, A or a, are **alleles**. Genes are positioned in a linear fashion on strings called **chromosomes**. A gene's position on the chromosome is its **locus**. In most plant and animal cells the

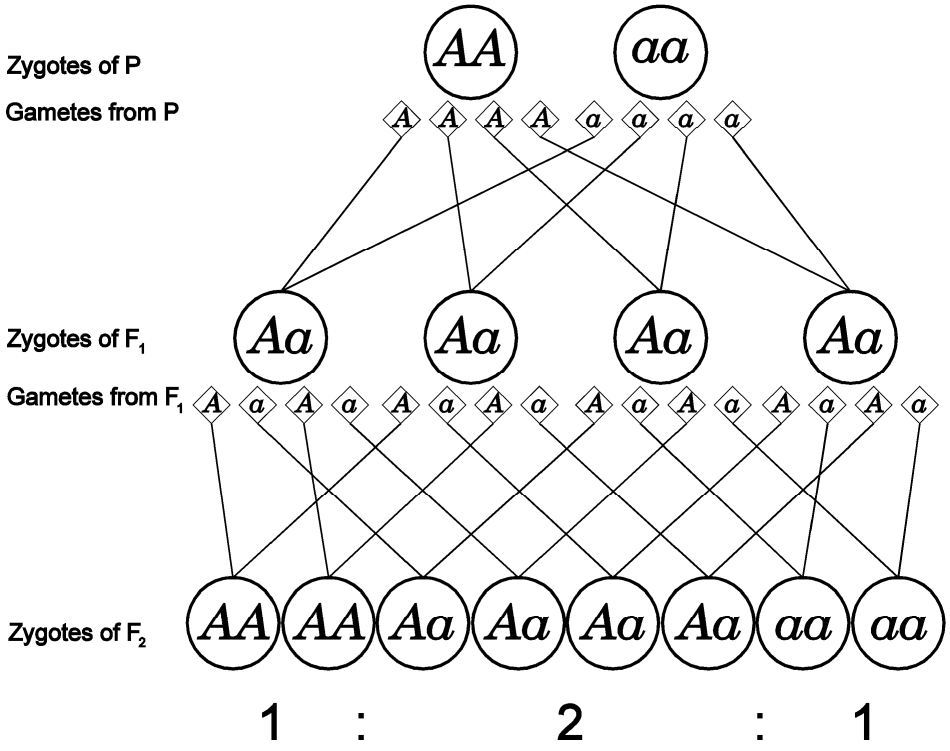


Figure 1. Diagram charting a single-gene Mendelian cross (monohybrid cross).

chromosomes occur in pairs or **diploid**. However, in **gametes**, which are the cells of reproduction (pollen and ovules for plants, egg and sperm for animals), the chromosomes are **haploid**, that is not paired. At fertilization the two parent gametes join to form a **zygote**, which is then again diploid.

Figure 1 depicts Mendel's experiment with the pea plant $Pisum\ quadratum$. In the parent generation, P, he crossed the homozygous plants AA with aa. Because the parents are homozygous, the gametes which they produce all carry the allele seen in the parent phenotype. Therefore, all zygotes formed in the F_1 generation are identical and heterozygous, Aa. They all have the same phenotype, i.e., smooth. The gametes produced by generation F_1 can, however, be either A or a, with about a 50% distribution either way. The two forms of the F_1 gametes can combine in $g^2=4$ ways to produce the zygotes of F_2 , viz. AA, Aa, aA and aa, with about equal chance for each to occur.

Three zygotes have the phenotype of smoothness, where as the homozygous aa will be wrinkled. Thus, the phenotypes exhibit the ratio 3:1 (dominant : recessive), while the genotypes are in the ratio 1:2:1 (AA:Aa:aa), where Aa and aA are the seen as the same genotype. Mendel observed seven other traits as well, and found the same ratios present in each case. The characteristic of genes to retain their traits is called the **principle of particulate inheritance** or Mendel's first law or sometimes the law of segregation (Minkoff, 1983, p. 24). Particulate inheritance means that the gene for smoothness and/or the gene for wrinkledness is passed on to subsequent generations unaltered. That is to say that A remains A, and a remains a, even if it is suppressed as recessive in the phenotype. There is no blending of the genes to form a new gene type. If a gene were to physically change it would represent a **mutation**.

Mendel also performed two experiments charting the inheritance of multiple traits. In one case he crossed a pea plant that had smooth peas with yellow meat ($albumen$) with a plant that had wrinkled peas with green meat. Designating smooth as A, wrinkled as a, yellow as B and green as b, the number zygotes resulting from the four different gametes would again be:

$$g^2 = 4^2 = 16$$

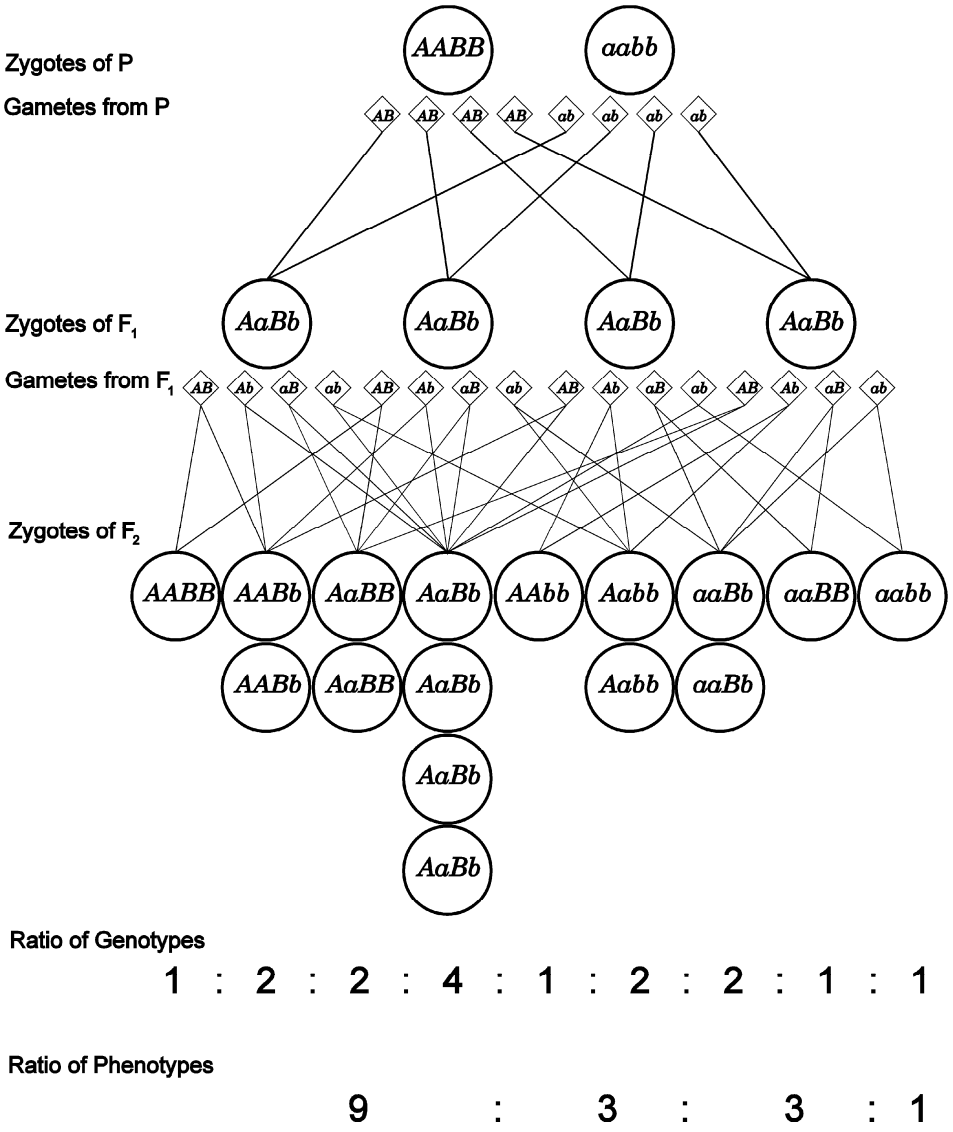


Figure 2. Results of Mendel's dihybrid cross.

Figure 2 shows the generations through F_2. The F_1 generation was all smooth and yellow, which correlated with the earlier findings which showed both smooth and yellow to be dominant. Mendel determined the genotypes of the F_2 generation by continued cross breeding. His results are given in Table 1.

Mendel's results showed that the phenotypes with the combined traits exhibit the individual traits in the same proportion, i.e., 3:1 as when the traits were isolated as single variables. This finding is called the **principle of independent assortment** or Mendel's second law and is formally stated as follows:

> A diploid organism heterozygous for two traits produces gametes in which all four combinations of these two traits are equally represented. (Minkoff, 1983, p. 24)

		FEMALE GAMETES			
		AB	Ab	aB	ab
MALE GAMETES	AB	$AABB$ smooth yellow	$AABb$ smooth yellow	$AaBB$ smooth yellow	$AaBb$ smooth yellow
	Ab	$AABb$ smooth yellow	$AAbb$ smooth green	$AaBb$ smooth yellow	$Aabb$ smooth green
	aB	$AaBB$ smooth yellow	$AaBb$ smooth yellow	$aaBB$ wrinkled yellow	$aaBb$ wrinkled yellow
	ab	$AaBb$ smooth yellow	$Aabb$ smooth green	$aaBb$ wrinkled yellow	$aabb$ wrinkled green

Table 1. Punnett square of Mendel's dihybrid cross.

It should be noted that the two genes governing the two traits mentioned in Mendel's second law must have their locus on different chromosomes for the law to hold. The reason for this is apparent when one considers the mechanics of cell division discussed in Section 1.2.1. It also points to the possible importance in the order of chromosome coding in a GA.

The F_2 generation is shown again in Table 1 as a Punnett Square. Table 2 shows the ratios of genotypes and phenotypes. These ratios compare well with the ones which Mendel observed in his experiments.

GENOTYPE RATIO	PHENOTYPE RATIO
$AABB$ 1 $AaBB$ 2 $AABb$ 2 $AaBb$ 4	smooth-yellow 9
$aaBB$ 1 $aaBb$ 2	wrinkled-yellow 3
$AAbb$ 1 $Aabb$ 2	smooth- green 3
$aabb$ 1	wrinkled-green 1

Table 2. Ratios of genotypes and phenotypes in Mendel's dihybrid cross.

1.1.2 Traits with Continuous Variation

Mendel carefully chose subjects which exhibit strongly differentiable characteristics. The peas were either wrinkled or smooth, green or yellow, but never yellow-green. One need not look far, however, to find hereditary traits that are not so clearly defined. Human height or strength offers a good example. Some further explanation is needed for such cases. One explanation is **polygenic inheritance** (Minkoff, 1983, p. 27). Polygenic traits are governed by alleles at more than one locus (Futuyma, 1979, p. 343). In the case of polygenic inheritance many genes combine together to produce the end effect. There may be dozens of genes which effect adult height. How they are combined has an additive effect on the phenotype. This is similar, but not quite the same, as **epistasis**. In epistasis, a gene at one locus, (the epistatic gene) interferes with the effect on the phenotype of a different gene (the hypostatic gene) at some other locus (Futuyma, 1979, p. 348). It must also be realized that for many traits the specific environment in which the organism matures will also affect the phenotype. The supply of food, water or sunlight, to mention a few, can have an obvious influence on growth and final size. Where as polygeny is a case of several genes affecting one trait, the case of one gene affecting several traits is called **pleiotropy** (Russell, 1992, p. 697).

GENOTYPE	PHENOTYPE
AA Ao	type A
BB Bo	type B
AB	type AB
oo	type O

Table 3. Genetic co-dominance found in human blood types.

Although most genes occur in one of two allelic states, either dominant or recessive, there are other situations which have been observed. **Co-dominance**, for example, is a condition where traits of *both* alleles appear in the phenotype. A well known example of co-dominance is found in the gene governing the type of antigens present in human blood. The gene has three alleles: A, which produces type A antigens, B, which produces type B antigens and o, which produces no antigens. A and B are co-dominant, while o is recessive. Because of the co-dominance of types A and B, a phenotype AB can occur. Table 3 lists the possible genotypes and corresponding phenotypes. A similar

condition is **incomplete dominance**, where the traits of two alleles combine to produce a phenotype showing intermediate traits. Plant flower coloration frequently exhibits incomplete dominance. For example a homozygous red snapdragon crossed with a homozygous white will always result in the intermediate color, the heterozygous pink (Russell, 1992, p. 98).

1.2 Molecular Genetics

1.2.1 Cell Division

The process by which diploid cells divide is called **mitosis**. It involves the duplication of the entire set of diploid chromosomes. **Meiosis** is the process by which haploid gamete cells are formed. In meiosis the diploid chromosomes are duplicated and then divided amongst four daughter cells - one half of each chromosome pair to each daughter cell. There is a good deal of specialized terminology used to refer to cellular anatomy and phases of division, and readers may wish to consult a textbook on Molecular Biology for a more detailed treatment of the subject. With regards to GAs, the processes found in Meiosis provide the metaphoric basis of the procedure.

As part of Meiosis, morphologically similar pairs of chromosomes come together to form a tetrad (see Figure 3). Because they are morphologically similar, i.e., having the same length and form, they can physically 'zipper up' side by side in a process called synapsis. Two chromosomes of this sort are called homologous partners.

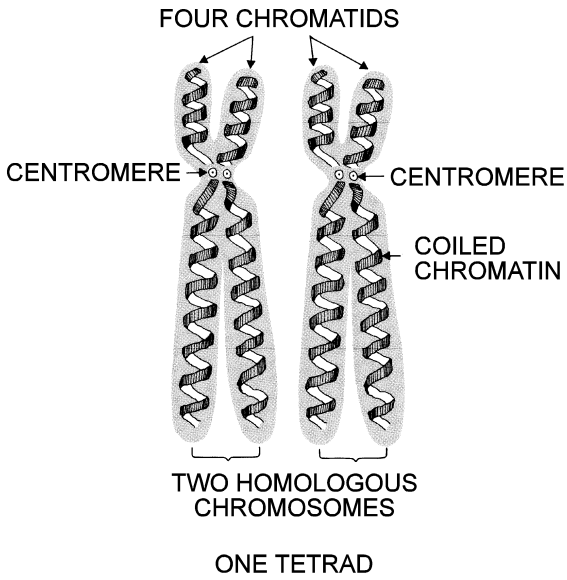


Figure 3. A tetrad as formed by two homologous chromosomes. (adapted from Minkoff, 1983, p. 23)

Now all of this is interesting to the coding of a GA in that early developers were careful to maintain constant length (homologous) binary strings to describe an individual solution. More recent implementations are able to breed unequal length chromosome strings as well.

During subsequent phases of meiosis, the closely aligned chromatids of the tetrad can reciprocally swap segments along their lengths in a process termed crossover. Crossover usually involves a direct exchange between chromatids resulting in a recombination of genetic material. Figure 4 gives a graphic depiction to the process. Crossover can take place between chromatids of one chromosome, or between the paired chromosomes of the tetrad. It can readily be observed that much more variation is in this way introduced into the progeny than could ever occur with simple division. A mimicry of this crossover process is a key technique employed by GAs in producing variation. However, not all biological species undergo crossover during meiosis. When it does occur the chromosomes present afterwards are genetically different from those before, and are called recombinant chromosomes.

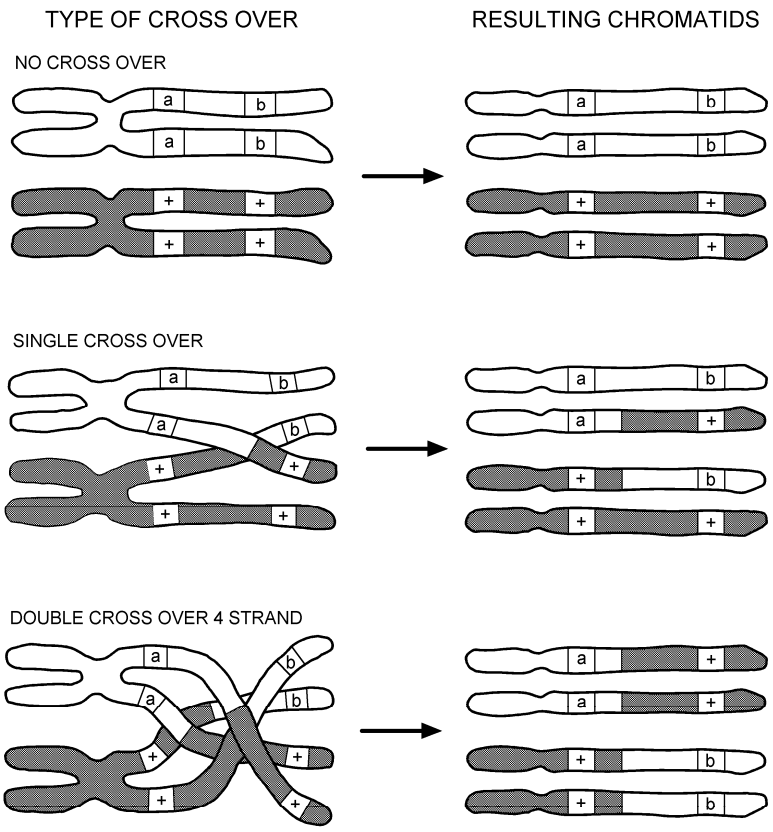


Figure 4. Three types of cross over of the homologous pairs. A, B and a, b represent genes located on the same chromosome. (adapted from Russell, 1992, p. 169)

Through the entire process of meiosis I and II a parent diploid cell it transformed into four haploid daughter cells. From the standpoint of application to GAs, it is important to note the degree of variation of progeny that can result from this process. The variation comes from two events.

- random mixing
- crossover

Random mixing of the chromosomes, which were originally supplied by the two parents in the previous generation, occurs in anaphase I, when dyads are randomly segregated one to each pole. The number of possible variations depends on the number of different bivalents present in the species. Taking n as the number of bivalents there would be 2^n number of possible different progeny based simply on combinations of parent chromosomes (i.e., the haploid number). For example, Mendel's $Pisum$ contains 14 diploid chromosomes which would give 7 haploid. This would allow $2^7 = 128$ possible variants. Although he had no clear concept of chromosomes, Mendel himself sights this number as the possible number variations of 7 traits (Mendel, p. 73). In humans with a haploid number of 23 chromosomes there would be $2^{23} = 8{,}400{,}000$. possible variations resulting from simple combinations alone.

Crossover adds vastly more variation to the formula. More complex chromosomes may have several crossover points (chiasmata), typically between one and eight per chromosome. Even more critical to variation, the chiasmata can form at random locations along the bivalent - either between sister chromatids or between the chromatids originating on different chromosomes (Gottschalk, 1984, p. 118). Potentially this allows the crossover of any genes between chromosomes. The amount of variation possible with such mechanisms is truly huge.

1.2.2 Genetic Coding

The key to the genetic coding of a chromosome lies in the structure of the molecule of which it is composed, deoxyribonucleic acid (DNA). A gene, the hereditary unit, is understood to be some length of the DNA molecule (ca. 1-2 nm). Chromosomes are formed by the bunching or winding of long lengths of DNA (Arms & Camp, p. 182, 1987). The 46 human chromosomes are composed of approximately two meters of DNA. Figure 5 shows the way DNA is packed into the form of chromosomes.

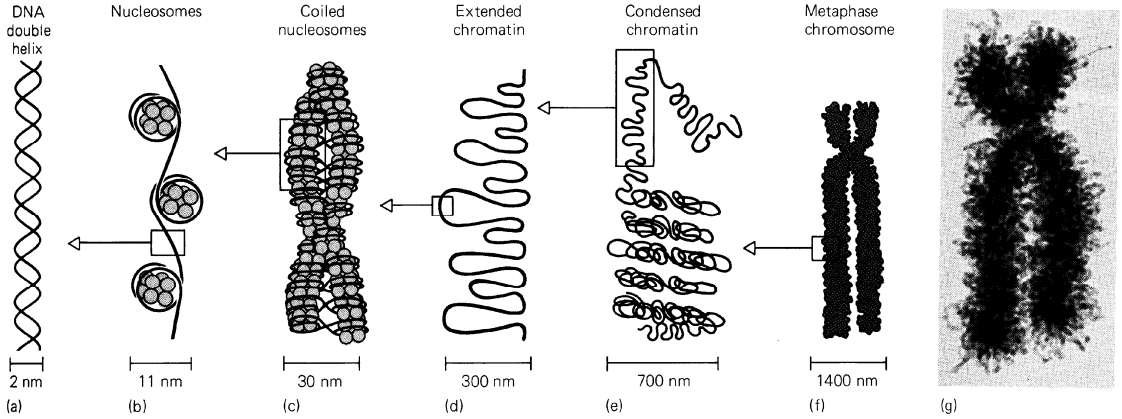


Figure 5. The way in which DNA is packed into chromatin. (from Arms & Camp, p. 183, 1987)

The well known structure of the DNA molecule was first published in *Nature* by James Watson and Francis Crick (Watson & Crick, 1953).

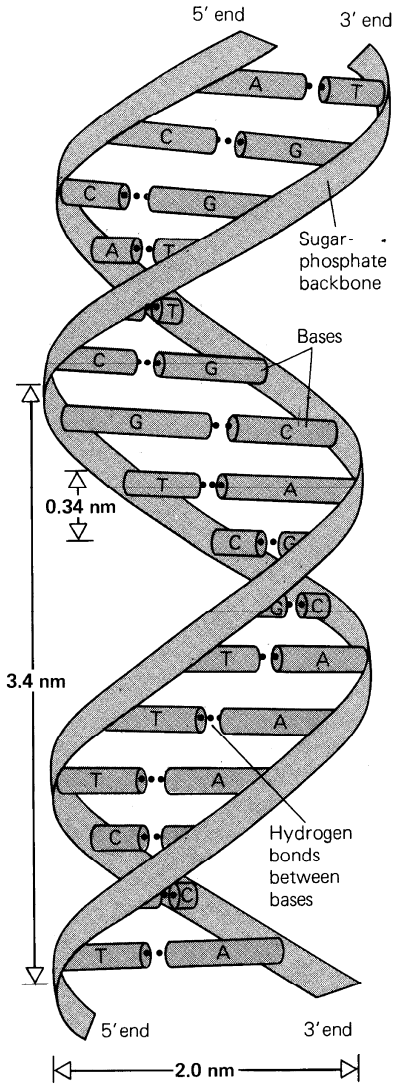


Figure 6. The DNA Double Helix Structure (from Arms & Camp, 1987, p. 174)

As shown in Figure 6 the DNA molecule is formed like a twisted ladder - a double helix. Rails of the ladder, called the backbone, are built of alternating molecules of phosphate and sugar. The rungs of the ladder attach to the sugar molecules, and are built of four possible nitrogenous bases - adenine (A), thymine (T), gaunine (G), cytosine (C). A single unit of the ladder is called a nucleotide, and consists of the nitrogenous base plus the associated length of backbone sugar and phosphate molecules. As can be observed in Figure 6 the four nitrogenous bases always combine in pairs to form a rung of the ladder, and always matched A with T and G with C. Thus, there are four possible nucleotides, viz. A-T, T-A, G-C and C-G, which in various combinations constitute all known DNA.

Because the nitrogenous base pairs are always matched by this rule, it is possible for the molecule to be replicated to its original structure after it has been split in half during cell division. Nucleotides work together in triplets called codons to define the genetic code. To describe a single gene takes several codons. The **genome** of an individual includes all of its DNA in all of its chromosomes. However it would seem that not all of the DNA is actually employed in genetic coding. About 30% of the DNA in eukaryotes does not seem to be involved in coding (Arms & Camp, p. 185, 1987).

Some sections of DNA simply repeat. For example in the fruit fly, Drosophila, the nucleotide sequence of A-G-A-A-G repeats about 100,000 times (Arms & Camp, p. 185, 1987). Other nucleotide sequences called satellite DNA have been found to repeat at intervals. The function of this noncoding DNA is uncertain. In any case, the simple amount of DNA does not correlate with the complexity of an organism in terms of proteins. For example, the largest known genome is that of the salamander which contains about 30 times the amount of DNA present in human cells (Arms & Camp, p. 185, 1987).

1.2.3 Genetic Code Disruption

The genome described above can, under some circumstances, become altered in ways other than the regular genetic recombination. Organizational changes may occur as routine, normal mechanisms for altering gene expression, or they may be the result of abnormal cell division. When the alteration results in a heritable change of the genome

it is termed a **mutation**. For purposes of study, mutations are usually divided into two categories:

- gene mutations
- chromosomal aberrations

Gene mutations occur at the level of individual genes or nucleotides, while chromosomal aberrations involve whole chromosomes.

Gene mutations occur as irregularities in the replication process of the DNA. Such irregularities take place with relative infrequency due to mechanisms that 'double check' the matching of the nucleotides as the DNA is replicated. There are also enzymes present in normal cells that seek and repair DNA damage due to heat and other natural sources. But, changes do, from time to time, take place in the genetic material which are not repaired. Arms & Camp list four categories of gene mutations (Arms & Camp, 1987, p. 180):

- **Substitution** of one nucleotide into another.
- **Insertion** of one or more nucleotides into a DNA sequence.
- **Deletion** of one or more nucleotides into a DNA sequence.
- **Inversion** of part of a nucleotide sequence (reversing part of the DNA sequence).

These mutations all seem to occur at a low rate in nature or can be induced unnaturally by exposure to higher than normal doses of radiation (such as ultra violet radiation or X-rays) or certain chemicals. Elements which induce mutation are called **mutagens**. The insertion or deletion of a nucleotide is generally more disruptive than either substitution or inversion. This is due to the way the code is read with each three adjacent nucleotides forming a codon. Insertion or deletion of one or two nucleotides alters the reading of all that follow on the chromosome. On the other hand, the substitution or inversion operations do not alter the total number of nucleotides, and therefore, the scope of influence is limited to a single gene. This is the type of gene mutation that is commonly modeled by a GA. Although gene mutations in GAs are usually located at random along the nucleotides, this is not always the case in nature. Certain combinations of nucleotides are more prone to alteration than others. For example, when 5-methylcytosine deaminates to form thymine, there are no mechanisms which can repair this error since thymine is also naturally present in the DNA. Therefore, because it is not

repairable, 5-methylcytosine is more prone to mutation than other bases found in the DNA, and locations where it occurs are mutation 'hot spots' (Russell, p. 564, 1992).

Chromosomal aberrations are alterations to the genome on the scale of complete chromosomes. Russell lists the five following categories of chromosomal aberrations (Russell, p. 588, 1992):

- **Deletion** or loss of a DNA segment or entire chromosome.
- **Duplication** or addition of a DNA segment or entire chromosome.
- **Inversion** or change in orientation of a DNA segment.
- **Translocation** or the movement of a DNA segment to another location in the genome.
- **Breakage** of chromosome, and attachment of part of one chromosome to another

Deletion and duplication of complete chromosomes is usually fatal in eukaryotes, and is also not typically used in GAs. Inversion and translocation, however produce mutations which more often survive, and have likewise been employed in GA design.

Inversion occurs when a segment of a chromosome is freed, and then rejoined at the same location but with a 180° rotation. Depending on where this takes place along the chromosome, and whether individual genes are thereby divided, the results on the phenotype can be more or less pronounced. Particularly critical are inversions that include the centromere. Unless the centromere happens to be perfectly centered in the freed segment, the end result will have an altered symmetry. This would cause abnormalities for bivalent pairing during cell division. In the GA model centromeres are not included, and so there is no the problem of symmetry change. When the GA chromosome is modeled as a binary string, breaks can be taken at any point. If, on the other hand, real numbers are used to model the genes, the real numbers are usually not split.

Translocation (or **transposition**) is the relocation of a segment of a chromosome to some other location on the genome. If the segment is relocated within the same chromosome the translocation is **intrachromosonal**. If the segment is relocated to a location on another chromosome, then the translocation is **interchromosonal**. Although some genes are not sensitive to location on the chromosome, some are, and some can exercise an affect on other genes in the proximity. An example of this if found

in **transposable elements** discovered by Barbara McClintock in her work with the Indian corn plant. McClintock found that some sections of the DNA can move about to different locations, and even different chromosomes resulting in suppression or expression of certain traits depending on the proximity of the transposable elements to certain genes. The instance that McClintock discovered had to do with the coloration of the corn kernels. If the transposable element is remote from the gene which controls coloration, the kernel will be purple; whereas if the transposable element is adjacent to the coloration gene on the DNA, the kernel will be white (Arms & Camp, p. 186, 1987).

1.3 Population Genetics

Population genetics studies the distribution and change over time of hereditary traits in a group of individuals. Whereas, transmission genetics studies the passing of hereditary traits from individuals to their immediate offspring, and molecular genetics studies the micro-mechanics of this transmission, population genetics is concerned with issues of change in the genetic pool of a group of interbreeding individuals. This is particularly instructive for the understanding of GAs. As a tool for searching populations of individuals by encouraging change in hereditary parameters, GAs draw heavily on the analogy to mechanisms recognized in population genetics.

1.3.1 Genotypic and Allelic Frequency

The genetic makeup of a population of individuals is called its **gene pool**. When discussing hereditary traits the terms genotypic frequency and allelic frequency are used as quantitative measurements of the composition of the gene pool. The genotypic frequency is usually limited to genes at a specific locus, and is simply the percentage of presence of a particular genotype in the population. It is defined as the number of individuals of the same genotype, divided by the total population.

$$\text{genotypic freq.} = \mathbf{P} = \frac{\text{number of individuals with the genotype}}{\text{number of individuals in the population}}$$

The allelic frequency is used to study the relative presence of a particular allele of a gene in a population.

$$\text{allelic freq.} = p = \frac{\text{number of copies of an allele of a gene in a population}}{\text{total number of alleles for a gene in the population}}$$

These two frequencies are used to describe a population with respect to specific traits of interest. Plotting the frequency values over time can give a picture of the genetic dynamics of a population. If the frequency values remain fairly constant, the population is in genetic equilibrium. The **Hardy-Weinberg principle** states that genetic equilibrium will be attained in one generation when the following criteria are met:

- the breeding population is infinitely large (at least relatively)
- the breeding parents are randomly paired
- no conditions which would cause a change to the gene pool (evolutionary forces) are present - viz.
  - mutation
  - genetic drift
  - migration
  - selection

This state, known as **Hardy-Weinberg equilibrium** is described for a case with two alleles by:

$$p^2\,AA + 2pq\,Aa + q^2\,aa = 1 \qquad \text{(Minkoff, 1983, p. 138)}$$

where, p and q are the allelic frequencies for, respectively, A and a, two alleles for a gene under study. As long as the conditions listed above remain true, the frequencies of the various alleles in the gene pool remain constant from one generation to the next. The Hardy-Weinberg equation can also be understood by considering a Punnett square showing the combination of the gametes A and a, with frequencies of p and q.

		FEMALE GAMETES	
		pA	qa
MALE GAMETES	pA	p^2AA	$pqAa$
	qa	$pqAa$	q^2aa

Table 4. Punnett square showing a cross with allelic frequencies.

In most cases in nature, as well as in GAs, the four restricting conditions: mutation, migration, selection, and genetic drift, are not met, and therefore, the gene pool of the population is always evolving. This aspect of evolution is what makes GAs useful as

search functions. Hence, it is worthwhile to consider each of these evolutionary forces in more detail.

1.3.2 Mutation

The mechanics of mutation were discussed in Section 1.2.3. Although in nature mutations occur with relative infrequency, this frequency can be increased in a GA in order to accelerate evolution in the population. Evolution of a gene pool can be seen as taking place in two phases. First, variation must be introduced into the gene pool, and second, evolutionary forces need to be present which will alter the genetic frequencies of the population. Mutation can play a roll in each of these phases.

Ultimately, any totally new allele can only arise through mutation. In nature; as in a GA, most mutations will be detrimental to the individual. In some percentage of the cases, however, the new allele will perform better in the current environment than other alleles already present in the gene pool. In this case the mutated allele can spread, and even replace the previous alleles through preferential selection. The frequency rate of mutation has a direct influence on the genetic variation of a population. This is particularly true of a relatively small, isolated population such as is usually the case in a GA.

The second way in which mutations affect evolution is by directly altering the genetic frequency of the gene pool. This effect of mutations is also most apparent in a smaller population. For example, if in a population of 50 individuals, two alleles A and a are present for a given gene with respective frequencies of

$$p = 0.50$$
$$q = 0.50$$

and one of the A alleles mutates to become an a, the new frequencies will be

$$p = 0.49$$
$$q = 0.51$$

Disregarding other evolutionary forces, if a few generations later the mutation again takes place, the frequencies would become

$$p = 0.48$$
$$q = 0.52$$

In this way, although gradual, genetic frequency is directly altered with the accumulation of periodic mutations. The situation just described is termed **forward mutation**. If the mutation occurs in reverse, i.e., a mutates to A, then the condition is termed a **reverse mutation** (Russell, p. 727, 1992).

If, in the above example, only forward mutation persists, the population will gradually be converted to all a, and A will die out.

$p = 0.00$
$q = 1.00$

It can be useful to consider the rate at which mutation is taking place. In notation used in genetics, the rate at which A mutates to a ($A \rightarrow a$) is termed u. The reverse condition, $A \leftarrow a$, is termed v. In one generation, taken as Δt, the number of $A \rightarrow a$ is up and the reverse, $A \leftarrow a$, is vq. In most biological systems v is considerably less than u. The change in allelic frequency in one generation can be expressed as

$$\frac{\Delta p}{\Delta t} = vq - up$$

Since u and v tend to be very small in nature - on the order of 10^{-5} and 10^{-7} respectively (Futuyma, p. 241, 1979), the rate of change over one generation is rather slow. Using an initial value of 0.50 for p, and the values of 10^{-5} and 10^{-7} for u and v respectively, p in the next generation would only increase by 0.00000455. This indicates that mutation alone is a rather week evolutionary force. Theoretically, given enough generations without other interfering influences, equilibrium is reached when the rate of change converges to 0. Since $p+q=1$, one can substitute 1-p into the $\Delta p/\Delta t$ equation above to get

$$\frac{\Delta p}{\Delta t} = v(1-p) - up = 0$$

Solving for p in the above equation will give the equilibrium equation.

$$\hat{p} = \frac{v}{(u+v)}$$

Similarly, the equilibrium equation for q is

$$\hat{q} = \frac{u}{(u+v)}$$

Crow and Kimura (1970) estimate that considering mutations alone, it generally takes 70,000 generations for any allelic frequency to converge on half the distance toward the point of equilibrium. Russell describes the effect of mutation alone on changing gene frequencies as follows:

> In practice, mutation by itself changes the gene frequencies at such a slow rate that populations are rarely in mutational equilibrium. Other forces have more profound effects on gene frequencies, and mutation alone rarely determines the gene frequencies of a population. (Russell, p. 730, 1992)

The weak effect in nature of mutation alone is due to the low rates at which it takes place (i.e., 10^{-5} and 10^{-7}). Of course, this rate can be set to a much higher value in a GA.

1.3.3 Genetic Drift

The phenomena of genetic drift was first explained by Sewall Wright, and is defined as

> deviations from the Hardy-Weinberg equilibrium due to "chance" (sampling error) in small to moderate populations. (Minkoff, p.147, 1983)

Sampling error results from either bias or inadequate sampling of the population. Although the Hardy-Weinberg principle assumes an infinitely large breeding population, biological systems with at least 1000 individuals are usually seen as sufficiently large to give good results (Minkoff, p.146, 1983). Populations of less than 50, on the other hand, will show effects of sampling errors. The type of error that is encountered by too small a sample can easily be understood by considering the probability of landing heads or tails in a series of coin tosses. For a regular two sided coin, the expected frequency of landing either heads or tails is of course 0.50. Or in the terms of allelic frequencies used earlier

$p = 0.50$ (heads)
$q = 0.50$ (tails)

If 1000 tosses were made the prediction based on the allelic frequencies of 0.50 would predict 500 heads and 500 tails. If, as could easily be expected, the tosses actually produced 501 heads and 499 tails, the deviation of one toss would translate to actual frequencies of

$p = 0.501$ (heads)
$q = 0.499$ (tails)

that is, 0.1% difference to the predicted frequency. But, if a much smaller 'population' of tosses were made, say 4, the same discrepancy of one toss would translate into frequencies of

$p = 0.75$ (heads)
$q = 0.25$ (tails)

that is, 50% difference to the expected value of 0.50. And it would not be unimaginable that one might toss 4 heads in a row, which would yield frequencies of

$p = 1.00$ (heads)
$q = 0.00$ (tails)

At this point, if heads and tails were alleles of some gene, in the population of 4, tails would die out, not to return to the population again unless through mutation or migration from outside of the population. Although tossing four heads in a row would not seem so remarkable (a probability of 1 in 16) the same could not be said of tossing 1000 heads in a row (a probability of 1 in 10^{301}), although both cases have the same p. In the population of 4 tosses, tails (or heads too for that matter) had a reasonable chance of dying out. Whereas, in the population of 1000 the chance of one allele disappearing is extremely remote. From this it is easily realized how much more critical each individual is in a small population, and how major the impact of the random loss of one or two of those individuals might be.

Whereas, genetic drift is random in direction, its magnitude depends a great deal on the size of the breeding population. The amount of drift among populations is called the **variance of gene frequency** and is equal to,

$$\sigma_p^2 = \frac{pq}{2N_e}$$

where p and q are the allelic gene frequencies and N_e is the effective (breeding) population. The square root of variance is standard deviation, and in this instance is termed the **standard error of gene frequency**,

$$\sigma_p = \sqrt{\frac{pq}{2N_e}} = \sqrt{\frac{p(p-1)}{2N_e}}$$

A graphical depiction of σ is shown in Figure 7. This will be recognized as the well know Gaussian distribution curve. The total area under the curve represents in this case 100% of the populations considered in a theoretical sampling. As indicated, plus and minus one standard deviation encompasses about 65% of the sampled populations. Similarly the range contained between $\pm 2\ \sigma$ includes about 96% of the sampled populations.

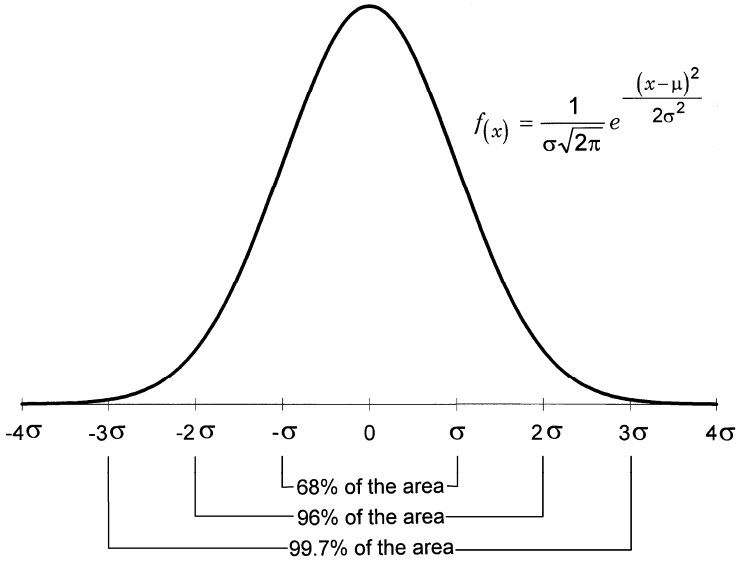


Figure 7. Graph of standard deviation from a mean value 0, relating the areas under the curve to bounds of $\pm\ \sigma$ values.

Being able to predict the standard error of gene frequency, σ_p, makes it possible to make decisions as to whether factors other than random drift are affecting the change of the gene pool. For instance, a common procedure is to use σ_p to calculate the 95% confidence limits of gene frequency.

From Figure 7 it can be seen that this is approximately the range encompassed by $\pm 2\ \sigma$. The meaning of the 95% confidence limits is that range by which p might be expected to deviate from its theoretical value based on the Hardy-Weinberg principle in 95 out of 100 instances. This is the deviation due to drift and occurs over the range

$$p \pm \sigma_p$$

where p is some theoretical mean value. For example, in a population of $N_e = 50$, with a gene frequency of $p = 0.80$, the standard error of gene frequency is calculated as

$$s_p = \sqrt{\frac{p(1-p)}{2N_e}} = \sqrt{\frac{0.80(1-0.80)}{2(50)}} = 0.04$$

from which the 95% confidence limits of gene frequency is

$$0.72 \le p \le 0.88$$

Continuing with this example, if $p = 0.80$ in generation **P**, and if Hardy-Weinberg equilibrium is in effect, then in the next generation **F_1**, the value of p will lie between 0.72 and 0.88 with a probability of 95%. If, for example, the value of p in **F_1** is found to be outside of this range, (e.g., 0.60), then in all likelihood some factors other than genetic drift are providing an evolutionary force to the population.

Genetic drift has two major effects on a population over time.

- Change of gene frequency.
- Reduction of genetic variation.

Figure 8 shows a computer simulation of how five populations can vary over 30 generations. The population size used was 20, and each initially contained a gene frequency of $p=q=0.50$. Were it not for sampling errors associated with the restricted size of the population, the Hardy-Weinberg equation would predict that the values of q would remain stable after one generation. What the graph shows, however, is that with the population of 20, the values 'drift' randomly both up and down. If a value should drift so far as $q=0.0$ or 1.0, then one or the other allele would be lost from the gene pool and the population would become **fixed** for the remaining allele. From that point onward it could only change through mutation or migration of the lost allele from another population.

The second effect of genetic drift is the segregation of alleles to form a majority of homozygous individuals. When through random drift an allele becomes fixed, as noted above, it remains that way. When due to smallness of size drift is significant, over time the population will become fixed for more and more genes, and thus lose genetic variation. Figure 9 shows the results of tests with 400 populations of 8 diploid individuals. Each of the populations began with an initial frequency of $p=0.5$ for the allele under observation. Within 32 generations most of the populations had become fixed at either 100% or 0%. As the population size increases this effect is diminished.

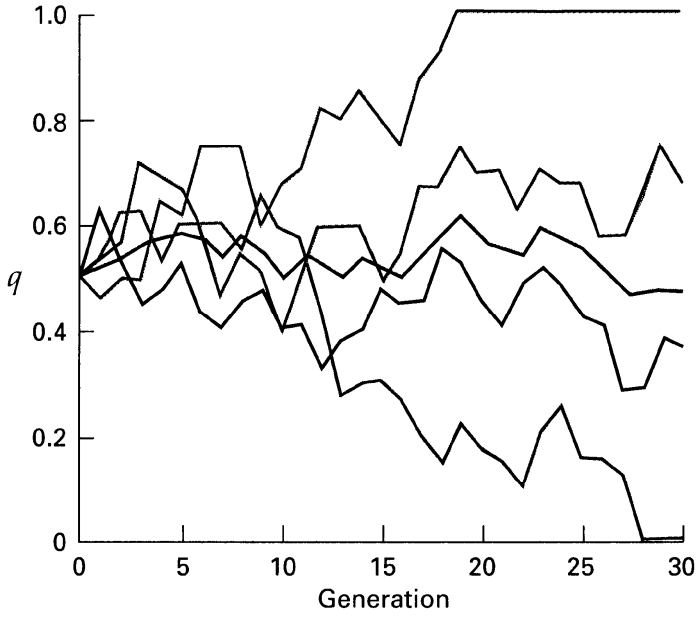


Figure 8. Simulation of genetic drift in 5 populations, each of size 20, over 30 generations. (Russell, p. 734, 1992)

Therefore, the population size must be carefully considered when designing a GA to avoid false convergence due to drift.

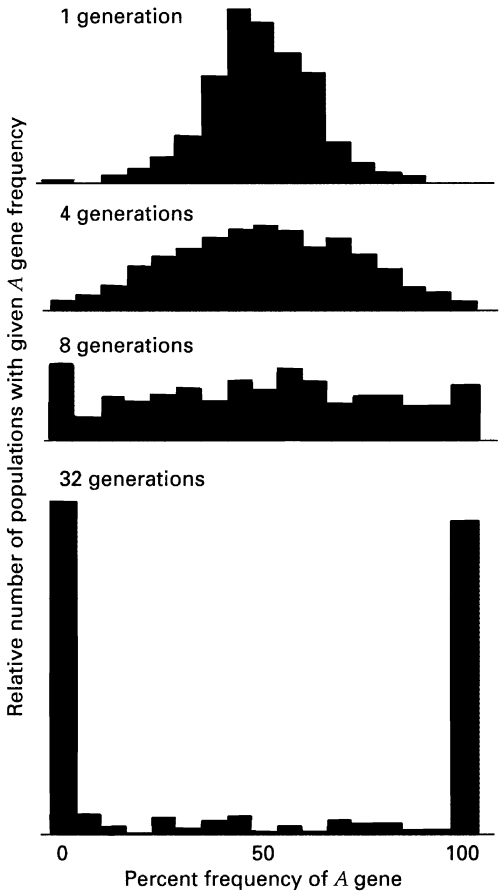


Figure 9. The effect of genetic drift on 400 populations with 8 individuals each. (from Mettler, Gregg and Shaffer, *Population Genetics and Evolution*, 1988)

1.3.4 Migration

Hardy-Weinberg equilibrium is also disturbed by migration. Migration in the sense of genetics refers to the movement or introduction of new alleles to a gene pool from which they were earlier isolated. This normally takes place by the introduction of an individual to the breeding population from a source outside that population. The result is called **gene flow** (Russell, p. 736, 1992). Gene flow can alter a gene pool in two ways:

- Introduction of new alleles.
- Alteration of allelic frequencies.

As discussed in Section 1.3.2., mutation occurs in nature at a very low rate. When a mutation does occur in a form which spreads within a population, it is a rare event, and would more likely be introduced to another population by the migration of some individual carrying the mutated allele, than by the occurrence of the same mutation a second time in the other population. Thus, advantageous mutations that have a low probability of occurring more than once, can still be spread throughout various populations as migration takes place. Even a small migration rate can have a significant effect by introducing new alleles to a population in which they can over time gain in frequency.

Even when no new alleles are introduced by migration, allelic frequencies will be altered when there exists some difference in the frequency of the migrant, and that to the population into which it migrates. Russell develops the equation

$$\Delta p = m\left(p_I - p_{II}\right) \qquad \text{(Russell, p.736, 1992)}$$

in which m is the portion in a final population of migrants with allelic frequencies of p_I which have been introduced into a population with an initial allelic frequency of p_{II}. As long as there is a difference between the frequencies of the two groups (i.e., p_I-$p_{II} \neq 0$), then there will be some change in the allelic frequency of the final combined group which equals Δp.

Both of these effects of migration tend to increase allelic variation within a gene pool, and, therefore, reduce the effect of genetic drift. Russell states,

> Even a small amount of gene flow can reduce the effect of genetic drift. Calculations have shown that a single migrant moving between two populations each generation will prevent the two

> populations from becoming fixed for different alleles. (Russell, p. 737, 1992)

In addition, by increasing the size of the breeding population, genetic drift is also reduced.

1.3.5 Natural Selection

The last of the four evolutionary forces listed in relation to the Hardy-Weinberg principle in Section 1.3.1., is natural selection. Natural selection was proposed as an evolutionary force by both Charles Darwin and Alfred Russel Wallace in 1858 to the Linnaean Society of London (Russell, p. 737, 1992). Whereas, mutation, genetic drift and migration all influence gene frequency within a population, selection has the additional effect of adaptation of a population to pressures of the environment. Natural selection is simply the non random reproduction of genotypes in the population. Phenotypes which breed more often, because they survive to breeding age or are for any reason more prone to successfully produce progeny, transmit their genotype in a higher frequency to the next generation. Thus, over many generations, the alleles of these phenotypes become more prevalent in the gene pool. This is the mechanism which allows long time adaptation of a species to a certain environment.

Charles Darwin sites numerous examples of species adaptation in his book, *On the Origin of Species* (1859). He shows how through the preferential selection of genotypes with a better ability to survive and breed, a gene pool will adapt to its environment. Being well adapted to the environment is usually synonymous with producing offspring. In genetics the term **fitness** refers to the relative success of an individual organism to reproduce itself. Fitness, designated as W, is a scale which relates the different genotypes in the gene pool, to the genotype most successful at reproducing itself. The genotype which produces the most offspring is the most fit, and is represented by W=1.0. Other genotypes in the population are represented as a proportional ranking based on how many offspring they produce in relation to the fittest genotype. For example, three genotypes of AA, Aa, and aa in a population might produce respectively, 12, 9 and 3 offspring. AA would be the most fit with W=12/12 or 1.0. Aa would have a fitness of W=9/12 or 0.75, and aa, the least fit, would have W=3/12 or

0.25. Another measurement used in relation to fitness is the **selection coefficient**, symbolized as s. The selection coefficient is defined as

$$s = 1\text{-}W$$

In the above example, AA, Aa, and aa would have respective selection coefficients of 0.0, 0.25 and 0.75.

The effect of natural selection on genetic frequencies is more difficult to quantify than the effects discussed in relation to mutation, drift and migration. Russell describes the situation as follows:

> Natural selection produces a number of different effects. At times, natural selection eliminates genetic variation, and at other times it maintains variation; it can change gene frequencies or prevent gene frequencies from changing; it can produce genetic divergence among populations or maintain genetic uniformity. Which of these effects occurs depends primarily on the relative fitness of the genotypes and on the frequencies of the alleles in the population. (Russell, p. 340, 1992).

Russell goes on to derive equations which can be used to calculate the change in genetic frequency with different fitness values and different forms of allelic dominance. Table 5 summarizes these equations.

Type of Selection	Fitness of Genotypes A^1A^1	A^1A^2	A^2A^2	Equations for Change in Gene Frequency
Selection against recessive homozygote	1	1	1-s	$\Delta q = \frac{-spq^2}{1-sq^2}$
Selection against a dominant allele	1-s	1-s	1	$\Delta q = \frac{-spq^2}{1-s+sq^2}$
Selection with no dominance	1	1-s/2	1-s	$\Delta q = \frac{-spq/2}{1-sq}$
Selection which favors the heterozygote (overdominance)	1-s	1	1-t	$\Delta q = \frac{pq(sp-tq)}{1-sp^2-tq^2}$
Selection which favors the heterozygote	1	1-s	1	$\Delta q = \frac{spq(q-p)}{1-2spq}$
General	W_{11}	W_{12}	W_{22}	$\Delta q = \frac{pq\left[p(W_{12}-W_{11})+q(W_{22}-W_{12})\right]}{p^2W_{11}+2pqW_{12}+q^2W_{22}}$

Table 5. Equations Used to Calculate Gene Frequency (after Russell, p. 743, 1992)

2 EC Mechanics

Evolutionary Computation (EC) or sometimes also called Evolutionary Algorithms (EAs) are class of stochastic numerical methods based on analogies with biological genetics. EC paradigms have been developed by different researchers starting in the late 1950's and early 1960's (Mitchell, 1996. p. 2). It is useful to recognize three categories of EC.

- Genetic Algorithms (GAs)
  - Genetic Programming (GP)
  - Classifier Systems (CFSs)
- Evolution Strategies (ESs)
  - Evolutionary Programming (EP)
- Interactive Evolutionary Computation (IEC)

Genetic Programming and Classifier Systems are actually subdivisions of GAs. Also, Evolution Strategies and Evolutionary Programming are very similar. All EC paradigms draw in some way upon an analogy to evolutionary genetics with distinctions between groups having in some instances more to do with the history of their isolated development than any major conceptual differences.

Genetic Algorithms (GAs) are credited to John Holland who worked on them at The University of Michigan and collected his findings in the book *Adaptation in Natural and Artificial Systems*, 1975. As originally conceived, they are based on a fairly close analogy to biologic genetic evolution. David Goldberg, a student of Holland coming from a background in civil engineering, did much to introduce the engineering community to the potential of GAs with his book *Genetic Algorithms in Search, Optimization, and Machine Learning*, 1989, and more recently *The Design of Innovation : Lessons from and for Competent Genetic Algorithms*, 2002. A GA typically operates on a population of binary strings using mechanisms of selective reproduction, crossover, and mutation. The specific mechanics of a GA are described in more detail in Section 2.1.

Genetic Programming (GP) uses GA techniques to evolve program structure. In this idiom, the population is composed of individual program routines which are usually represented in the form of parse trees. Crossover mechanisms are used to combine branches from the more fit structures forming new programs. The goal is to auto-

generate programs capable of solving problems as they occur (e.g., in response to a changing environment). J. R. Koza is a leading researcher in the area of GP.

Classifier Systems (CFSs), originally designated as CS and then later re-termed Learning Classifier Systems (LCS), were originally developed by John Holland as an application which incorporated GAs in a system that was able to evolve in, adapt to, or alter an environment in which it was placed. The 'environment' could be physical and perceived by sensors, but is more typically digital, and may be represented as text strings, numbers or equations. A classifier is an if-then rule which describes a relation of the system to its environment. In GA fashion, the classifying rules are generated (initially at random) in a population which then evolves to satisfy a fitness to the environment. In this way the best classifying rules which relate the system to its environment evolve without the need to actually program the rules.

Evolution Strategies (ESs) were developed during the 1960's by Ingo Rechenberg with the help of Hans-Paul Schwefel and Peter Bienert working at the Technical University of Berlin. Rechenberg published the findings in *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution* (1973). However, this work at the TU Berlin with ESs remained relatively unknown outside of the German engineering community, and developed independently from the work going on in the United States with GAs at that time. Although ESs and GAs are both based on genetic evolution, they vary in their approach in several details. For example ES problems are coded with real numbers were as GAs typically use binary strings. Mutation tends to play a more principle role in ES than in GA. Also, the use of populations was not initially part of ES. For example there is no use of populations in the simple (1+1)-ES, which has one parent and one offspring. A more detailed description of ES is given in Section 2.2.

Evolutionary Programming (EP) was developed primarily by Lawrence Fogel in the early 1960's. It was fully outlined in the book, *Artificial Intelligence Through Simulated Evolution*, 1966, by Fogel, Owens and Walsh. EP is very similar to ES, but there was virtually no contact between the two groups until the First Annual Conference on Evolutionary Programming in La Jolla, California in 1992 (Heitkötter & Beasley, 1998). Both ES and EP differ from GAs in their less literal translation of the mechanics of genetics. EP, like ES, typically uses real numbers in coding problem parameters where

GAs use binary strings in analogy to gene coding of chromosomes. GAs also make use of other genetic mechanisms such as crossover that are not used in EP.

Interactive Evolutionary Computation is the youngest of the divisions shown here. Although published applications of IEC did not begin to appear until the early 1990's, the concept was certainly well known well before that. Richard Dawkins picturesque description of his Biomorphs in *The Blind Watchmaker* (1986) no doubt inspired many. Art oriented graphics were an early use for IEC. Artists such as Karl Sims and William Latham are well known for their genetically derived graphics. Outside of graphics, acoustics (music) and morphology are typical areas of application.

In this study, different EC paradigms were investigated to determine which could be used most effectively with the truss structures being designed. For geometry optimization, a variation of a GA proposed by Larry Eshelman, et al. (1991) was found to be well suited. Eshelman's GA incorporates some typical ES features including real numbers and segregated use of mutation. Section 2.3 describes Eshelman's CHC-GA in detail. For topology search and optimization, a modified ES-($\mu+\lambda$) is proposed. The ($\mu+\lambda$) ES is described in Section 2.2.

2.1 GA Mechanics

As mentioned earlier, the analogy between biological genetics and GAs is fairly close. One use of GAs is, in fact, to model biological systems, a point which Holland recognized in his book *Adaptation in Natural and Artificial Systems* (1975). Bäck (1996, p. 16) notes that most GAs model haploid organisms with single chromosomes. Nonetheless, some GAs have also been successfully written based on multi-chromosomal, diploid organisms (Hillis, 1991). Also, the nomenclature used in GAs varies in part from the biological terminology. Table 6 compares the nomenclature used in the two fields.

Genetics	GAs	Meaning
phenotype	solution decoded structure	the manifestation of the individual (solution) in its environment
genotype	structure	the coded description of the individual
karyotype	coding format	the format of an organism's genetic material, number and form of chromosomes.
	schema schemata (plural)	Schemata are patterns of genes or building blocks.
chromosome	string	the element which contains the coded description of the individual
gene	feature, character, building block	the element which contains the coded description of one characteristic of the individual
allele	feature value	a specific value of a characteristic of the individual
locus	string position, address	the location of the gene on the chromosome (character in the string)
epistasis	nonlinearity	interference or masking of the affect of one gene (feature) on another
gene pool	population	a collection of interbreeding genotypes (structures)
crossing-over, cross over, crossover	crossover	reciprocal exchange of some portion(s) of the chromosome (string)

Table 6. Comparison of Nomenclature between the Fields of Genetics and GAs.

GAs have been applied to a wide range of search and optimization problems in fields of engineering and architecture, as well as areas such as drug design, scheduling and routing, financial prediction, data mining, system control, and even the composition of poetry, music and art (Goldberg, 1998b, p. 1; Lund, 1995; Beasley, 1997). This broad scope of application is part of the attraction of GAs. Goldberg describes GAs as "nonlinear, stochastic algorithms operating on combinatorial, possibly nondeterministic, problems of infinite variety." (Goldberg, 1992, p.1) The quality of applicability to "problems of infinite variety" is referred to as **robustness**, and is one of the most attractive attributes of GA's and EC in general. The no-free-lunch theorem (NFL) advanced by Wolpert and Macready (Schwefel, 1997), states that no generalistic type algorithm can ever be expected to surpass on the average the performance of other competing algorithms in solving all optimization type problems. At best GA's can be expected to excel over a certain range of problems. Hans-Paul Schwefel describes this range as, "discontinuous, nondifferentiable, multimodal, noisy, and otherwise

unconventional" (Schwefel, 1997). In other words, if the only design criteria was weight, and if the design space was fully defined, then the solution might better be sought using a traditional optimization technique such as linear programming. However, if nondifferentiable criteria such as aesthetics, historic allusion, contextual form, etc. are to be taken into account, then GA's can certainly offer an advantage. Schwefel concludes his remarks regarding the advantages and disadvantages of evolutionary computation over other approaches in the *Handbook of Evolutionary Computation* with the statement:

> A warning should be given about a common practice - the linearization or other decomplexification of the situation in order to make a traditional method applicable. Even a guaranteed globally optimal solution for the simplified task may be a long way off and thus largely inferior to an approximate solution to the real problem. (Schwefel, 1997, p. A1.3:2)

To solve a problem using a GA only a very few criteria must be satisfied. One, the parameters that describe a solution must be **coded** as a 'genetic' string or chromosome. Two, the solutions must be rankable by a single **fitness** value. Very little need be known about the problem or solutions otherwise. The process continues by cyclically applying certain genetic operators to generations of populations of solutions until an acceptably good solution evolves. The steps can be summarized as follows:

1. coding the problem
2. generating an initial population
3. applying genetic operators
4. determining fitness
5. selecting individuals
6. filling the population of the next generation

The procedure then cycles through steps 3. to 6. until a satisfactory solution is reached or the rate of improvement is diminished below a productive level.

2.1.1 Coding the Problem

A GA does not actually search for a solution. Instead, a GA searches for the **image** of a coding of a solution. By image of the solution what is meant is how the code looks, not what the code means. Most authors beginning with Holland (1975) onward have coded the parameters that describe a solution as binary strings. A string is usually composed of several binary numbers which represent the least number of parameters needed to describe the solution. The binary bits are strung together like the nitrogenous bases are

strung together in DNA. This binary string is the GA chromosome. Using genetic operators, described below, the GA searches for images of this chromosome that exhibit high fitness values. Usually, searching for the image of a number or the meaning of the number are about the same thing, but not always.

		IMAGE		MEANING
		Binary	Gray	Base 10
SOLUTIONS	A	0111	0100	7
	B	1000	1100	8
	C	1001	1101	9

Table 7. Comparison of image and meaning of the numbers 7, 8 and 9.

Table 7 shows how the small difference in meaning between the numbers 7 and 8, translates into the very different binary images of 0111 and 1000 respectively. The binary images could not be more different. Each of the 4 bits is reversed. But the meaning is very close. In an effort to provide better correlation between the binary image and the decoded meaning, a transposition of the binary digits developed by Gray is often employed (Bäck, 1996, p. 110). For the three numbers shown in Table 7 for example, although the binary strings look very different from one another, the Gray numbers change by only one bit between each consecutive number.

It is possible as well to write GAs using integer or real numbers. This is a technique which is more common in ES but has been applied to GAs as well (Eshelman & Schaffer, 1992; Powell & Skolnick, 1993).

In coding the problem it is important not to code too much information. Only enough information is needed to point to a unique solution. For example, when coding an architectural structure only information which will describe a unique geometry or physical material is needed. Parameters which describe the structure's behavior within some environment, such as stress levels, deflections, stability, etc., may be needed to determine the fitness of the solution, but are not needed to describe the structure as a unique solution. The parameters which are encoded make up the 'genotype'. Decisions regarding fitness are made based on the 'phenotype', which has fitness with respect to a specific environment (e.g., loading, time, exposure, etc.). This dichotomy of the problem

into solution and fitness gives the GA a great deal of flexibility. In application the fitness may be determined in any of a variety of ways as discussed below.

From the description of cellular reproduction in Section 1.2.1, the necessity for the genetic material of breeding individuals to be in the same format (i.e., be of the same karyotype - the same number and length of chromosomes) is apparent. In the same way, the format for the encoding of the GA strings must be consistent in order to allow successful application of the various genetic operators (viz. crossover and recombination). In fact formatting on a GA string is even more strictly followed than in nature. Genes on a GA string are usually assigned a particular position, where as biological genes are not in every instance limited to a single locus (e.g., as in translocation - Section 1.2.3).

The digits which make up the encoded strings are analogous to genes on a chromosome. They are the **building blocks** (BBs) which are manipulated by the genetic operators. Just as genes are composed of several nitrogenous base pairs on the DNA, a BB is composed of several digits of an encoded string. The patterns into which these building blocks can be combined are called **schemata** (singular schema). For a GA to be successful it must seek out the better schemata, and the associated better building blocks.

Schemata and BBs can perhaps be better understood using an example coded with English alphabet characters rather than binary digits. Although English words are coded with 26 letters rather than the 2 bit binary possibilities of 1 and 0, concepts of schemata and BBs are identical for the two coding techniques. As explained in Section 1.2.2, DNA is actually coded in a base 4 system using the nucleotides A-T, T-A, G-C, and C-G. In that sense, there is nothing too special about binary strings. The reason English words are chosen for this example is simply that most people can more easily recognize patterns in words than in numbers. The example will be to search for a string of length l=6; the 6 letter word "`flower`". All of the possible schemata would include all possible letter patterns that could be found in 6 characters. The different schemata would include variations of the 26 alphabet characters plus a 'wild card' character, here designated as "□", which stands for any of the 26 regular characters. Examples of possible schemata are shown in Table 8.

SCHEMA	FITNESS
□□owe□	3
f□□□□r	2
□□□□er	2
□□□□e□	1
□lowes	4
□□kwee	2
xyz□□□	0

Table 8. Examples of schemata with associated fitness values. The example is coded using 6 alpha characters, and searches for the word "flower".

Altogether there are 27^6 = 387,420,489 different six place schemata (assuming 26 alphabet characters, plus the wildcard □). However, they do not all have equal value when searching for "flower". The schemata which are more valuable are those which come closer to the goal. The value of a particular schema is expressed as its fitness. In the example with the word "flower", the fitness could simply be the number of correct places in the 6 letter word. The fitness values listed with the schemata above were determined in this way. In the GA population, fitness is used to decide the breeding rates at which individuals reproduce, and which die out in subsequent generations.

In Holland's original schema analysis (1975), the approach was to calculate the expected growth of better than average schemata within a population over time measured in generations. Using the notation of Goldberg (1989, p. 30), it can be shown that for any schema the rate at which it will increase or decrease in number within the population over time is proportional to the ratio of the average fitness of individuals possessing that schema to the average fitness of all individuals in the population, or;

$$m(H,t+1) = m(H,t)\frac{f(H)}{\bar{f}}$$

where;

H	is a particular schema
$f(H)$	is the average fitness of individuals containing schema H
t	is time, measured in generations
$m(H,t)$	is m number of instances of individuals with schema H at time t
$m(H,t+1)$	is m in the generation immediately following t
$\bar{f}$	is the average fitness of all individuals in the population

This means that if individuals containing a certain schema have an above average fitness, they will be favored in the next generation, while those with below average fitness

will be lessened in the next generation and gradually die out. It is important to notice that each individual contains many schemata simultaneously. That is, the individual xyzher represents at the same time the schemata □□□her and xyz□□□ as well as x□□□□□ and xy□□□□ and x□□□□r and □y□□□r and so forth. If the individual is selected and bred, then all of its component schemata are simultaneously brought forward to the next generation. Holland refers to this as **intrinsic parallelism** (Holland, 1975, p. 71).

Using Goldberg's above equation it is easy to derive the rate at which good schema should increase and bad schema decrease. If one assumes that a schema H will remain above average by a constant amount c, then substituting $\bar{f} + c\bar{f}$ for $f(H)$ in the schema growth equation yields;

$$m(H,t+1) = m(H,t)\frac{(\bar{f} + c\bar{f})}{\bar{f}}$$

or,

$$m(H,t+1) = (1+c)\cdot m(H,t)$$

and taking t=0 and maintaining a stationary c;

$$m(H,t) = m(H,t)\cdot(1+c)^t$$ (Goldberg, 1989, p. 30.)

This shows that under selective reproduction alone the growth rate for good schemata is geometric. In Section 2.1.3. the schemata growth rate equation is expanded to include the effects of the genetic operators, crossover and mutation.

The other aspect of schemata worth consideration deals with building blocks. Schemata are composed of building blocks. In the schema example using alpha characters, it is easy to see how building blocks combine to form schema and how some have a higher probability of success than others. Building blocks can be of different lengths - one bit or several. Expanding the example presented in Table 8, Table 9 shows a possible building block for each schema, and the frequency with which each occurs in Ogden's Basic English.

Schemata	Possible Building Blocks	Frequency found Basic English
□□owe□	ow	23
f□□□□r	f	79
□□□□er	er	69
□□□□e□	e	547
□lowes	low	7
□□kwee	w	84
xyz□□□	xy	0

Table 9. Alpha coded schemata from Table 8, showing examples of possible building blocks and the frequency with which each occurs in Ogden's Basic English.

The frequency of occurrence in Ogden's Basic English of each of the chosen building blocks, gives a quantitative measure to what anyone familiar with English would know intuitively. If one were trying to solve a word puzzle, the combination "xy" is not likely to occur, and would be a bad guess. On the other hand, the combination "er" is fairly common. Not only that, "er" is more likely to occur at the end of a word (40 out of 69 occurrences). Sherlock Holmes broke the code in *The Adventure of the Dancing Men*, by observing that "e" is the letter occurring most frequently in the English vocabulary (Doyle, 1997). It would seem to be an advantage for a GA to somehow make use of this extra knowledge regarding the relative values of building blocks. Unfortunately, the 'generic' GA has no way of recognizing building blocks at all, and randomly destroys them during crossover by selecting crossover points which may divide some advantageous building block. This does not often occur in natural DNA because the genes have end markers, which mark locations for crossover which would not split (and thereby destroy) genes. David Goldberg has put much effort into enhancing GAs with the ability to preserve good building blocks. His fast messy Genetic Algorithm (fmGA) is an attempt in this direction (Goldberg, Deb, Kargupta & Harik, 1993). The fmGA has been shown to offer superior performance particularly on problems traditionally hard for GAs to solve. In recent years several researchers have worked to solve the problem of GA building block loss. Efforts in this direction include Kargupta's work (1996) with "gene expression messy genetic algorithms" (gemGA), and Harik's linkage learning genetic algorithm (LLGA) (Harik & Goldberg, 1996). Goldberg sees many parallels between the operations of GAs and the way innovation and creativity seem to function in human problem solving. In his paper "The Race, the Hurdle, and the Sweet Spot: Lessons

from Genetic Algorithms for the Automation of Design Innovation and Creativity" (1998b), Goldberg proposes that the metaphor between the GA and creative human design processes can be used by GA designers to overcome the problem of indiscriminate building block disruption. Goldberg's observations also offer much to the design community at large as a means of formulating long disputed general theories regarding human innovation and creative design.

2.1.2 Generating a Population

Genetic Algorithms work not with single solutions in iteration, but with a population of solutions in iterative parallel. It is in this way that the implicit parallelism of schema optimization discussed in Section 2.1.1 is possible. Most GA implementations begin with the random generation of a population of individuals which can then be subjected to genetic operators, such as crossover, mutation and selection, to produce subsequent generations. Just as too small a population size can affect the gene distribution in a biological population (see Section 1.3.3), genetic drift can also lead to the loss of possibly beneficial building blocks in GAs. Therefore, it is important that a population be large enough to prevent premature convergence. On the other hand, the larger the population is, the more time cycles will be required to process it through the GA operations. To ensure a search which is thorough, and yet efficient, it is necessary to carefully consider the population size.

Goldberg has contributed significantly to the solution of this problem (Goldberg, 1989, 1991), and has derived equations for population sizing based on the signal difference and mean variance of competing building blocks. Goldberg notes that for good building blocks to be preserved the background noise level must be low in relation to the signal level of the building blocks. The noise present in the population can be lowered by improved (i.e., more frequent) sampling, which in turn requires a larger population. Considering sampling frequency alone, Goldberg found that:

> For a given pairwise competition of schemata, the population size varies inversely with the square of the signal that must be detected and proportionally to the product of the number of competitors in the competition partition, the total building-block error, and a constant that increases with decreasing permissible error. (Goldberg, 1991, p. 6)

This he expressed in the equation:

$$n = 2c\kappa \frac{\delta_M^2}{d^2}$$

where;

n	is the population size (number of individuals)
c	is a coefficient of discrimination with a given error rate
κ	is the number of competing schemata
d	is the signal difference in two competing building blocks
δ_M^2	is the mean variance of the two building blocks

Goldberg also notes that signal noise may come from a variety of other sources as well, including: "inherently noisy problems, noisy selection algorithms, and the variance of other genetic operators." By defining multiplier factors for each of these other noise sources, Goldberg further qualifies the above equation for some of these additional noise sources (Goldberg, 1991, pp. 6-8). Goldberg tests the population sizing equations on five example functions. Although the equations proved to make good, conservative estimates of population size for the five test-bed equations, not knowing the defining equation in advance (the condition assumed in this work) would make it difficult to supply quantitative measurements for d and δ_M^2. In such a case, Goldberg's following qualitative summary offers a direction which can be explored with problem specific trials to find the boundary defining 'large' and 'small' populations.

> At low population sizes we see GAs buffeted by the vagaries of chance, converging only through the good graces of random changes that are lucky enough to survive to a time when they may be properly judged. At high population sizes we see GAs that promote only the best among competing building blocks, and when and if these are global, with high probability we can expect convergence to global solutions after sufficient recombination. (Goldberg, 1991, p. 23)

One further qualitative guide offered by Goldberg; in the end, population size is simply a function of the string length (Goldberg, 1991, p. 8). For this reason, careful attention to problem definition and encoding to use a minimum of descriptors, enhances the ratio of optimal convergence to population size.

2.1.3 Applying Genetic Operators

With the problem genetically encoded and a population of individuals in place, the genetic operators can next be a applied. By the application of the genetic operators, the individual structures within the population will adapt, under the pressure exerted by the fitness function, to a form most suited for survival in the defined environment of the problem. In this procedure, many of the same pitfalls are present which hinder adapting biological populations (viz. genetic drift, inbreeding, too low or too high environmental pressure). Through attention to the parameters associated with the genetic operators, these pitfalls can be somewhat avoided. At the same time, in the interest of maintaining robustness, the genetic operators used in this study were designed to fit a wide class of structural design problems.

The genetic operators used in a GA can be described under three headings:

- crossover
- mutation
- selection

Of these three, crossover and selection are the most important, with mutation playing a secondary roll.

Crossover is the operator associated with breeding. It is also referred to as recombination, because it recombines the schemata of parent individuals to form new daughter individuals. As discussed in Section 1.2.1, during meiosis the haploid chromosomes are randomly assorted to the daughter cells. But during prophase I of meiosis, crossover provides a much greater degree of mixing of the genetic material. As shown in Figure 4 the individual chromatids exchange segments of genetic material between randomly spaced chiasmata. This is mimicked in GAs by the crossover operator.

Although biological systems tend to use something more like n-point crossover, described below, Holland's initial work (1975) favored one-point crossover (n=1). Since that time, several styles of crossover have been proposed with the general intent of removing specific biases, and thus producing a more robust operator. The most successful methods can be described in four categories:

- one-point
- n-point

- uniform
- parameterized uniform

Of these four types, one-point is actually a special case of n-point, and uniform is a special case of parameterized uniform.

One-point crossover, as defined by Holland, divides each of the two parent GA strings into two parts at corresponding, random chosen, points. One set of corresponding segments is then switched between parents. Figure 10 shows three possible outcomes of the application of the one-point operator on two example strings.

Before Crossover:			
$\mathbf{P}_1$	b|utter	bu|tter	but|ter
$\mathbf{P}_2$	g|lower	gl|ower	glo|wer
After Crossover:	⇓	⇓	⇓
$\mathbf{F}_1$	g|utter	bl|tter	glo|ter
$\mathbf{F}_2$	b|lower	bu|ower	but|wer

Figure 10. Three possible applications of one-point crossover on the same two structures. P_1 and P_2 represent parents of the children F_1 and F_2.

Depending on where the cut point takes place, certain schema will be destroyed, and others preserved. Holland favored a form of crossover that would be less disruptive to schemata, which is why he proposed the use of the one-point type. Crossover operators which use more splicing points, increase the probability of destroying schema by splitting.

Before Crossover:			
$\mathbf{P}_1$	□□□□|er	□□|□□er	□□□□e|r
$\mathbf{P}_2$	□low|es	□l|owes	□lowe|s
After Crossover:	⇓	⇓	⇓
$\mathbf{F}_1$	□□□□es	□□owes	□□□□es
$\mathbf{F}_2$	□lower	□l□□er	□lower
Effect on Schemata:	⇓	⇓	⇓
er	preserved	preserved	preserved
lowes	destroyed	destroyed	destroyed

Figure 11. The effect of one-point crossover on the schemata of two structures. Longer schemata have a greater probability of being destroyed, which results in a bias towards shorter schemata.

Since the success of the GA depends on the retention, and inheritance of better schemata by subsequent generations, the issue of schemata destruction by crossover is of particular interest. Holland defined several terms which help describe what happens with schemata during crossover. Schema **length** is defined as the difference between the first and last *defining* bit positions (Holland, 1975, p. 72). That is the first and last non-wildcard bits in the schema. For example if the schema starts at bit position 3, and ends at bit position 8, then 8-3 gives the length 5. Longer schemata are more likely to be disrupted by crossover simply because there is a greater probability that a crossover point will occur within their length. Further, the **order** of a particular schema refers to the hyperplane order which is the number of set places (i.e., not counting wildcards) in the schema.

SCHEMA	LENGTH	ORDER	FITNESS
□□owe□	2	3	3
f□□□□r	5	2	2
□□□□er	1	2	2
□□□□e□	0	1	1
□lowes	4	5	4
□□kwee	3	4	2
xyz□□□	2	3	0

Table 10. Schema data from an example using alphabetic coding to search for the word `flower`.

Table 10 shows the same alphabet coded example used above with length and order tabulated. The longer the schema length, the more likely it is to be destroyed in crossover. This can readily be observed in the examples of one-point crossover shown in Figure 11.

Building on Goldberg's version of the probability equation for schema survival, given in Section 2.1.1, the probable loss of schema due to disruption by one-point crossover can be factored in as the final term in the statement:

$$m(H,t+1) \geq m(H,t) \cdot \frac{f(H)}{\bar{f}} \left[1 - p_c \cdot \frac{\delta(H)}{l-1} \right]$$

where;

$1 - p_c \cdot \frac{\delta(H)}{l-1}$ is the probability that schema H will survive one-point crossover

where;

p_c	is the probability of crossover occurring
$\delta(H)$	is the spanning length of the schema H
l	is the length of the string

As illustrated in Figure 11, and represented by the placement of $\delta(H)$ in the above statement, one-point crossover has a bias against the survival of longer schemata.

One-point crossover exhibits other anomalies associated with bit position. Firstly, not all bit positions have the same probability of being exchanged. For instance, the end positions are always exchanged, while those near the center less often. Secondly, because larger sections of the strings are exchanged, mixing of smaller schemata is impaired. Poorly performing schemata can sometimes survive if they are clustered near good performers. This results in a phenomena called hitchhiking.

Finally, one-point crossover is limited in the possible combinations attainable in a single step. This is illustrated in Figure 12. It can be seen that even though all the desired schemata were present in the two initial parents, a minimum of two steps were needed to reach a combination of the better schemata. The result is that one-point crossover explores less of the problem search space in a given number of cycles.

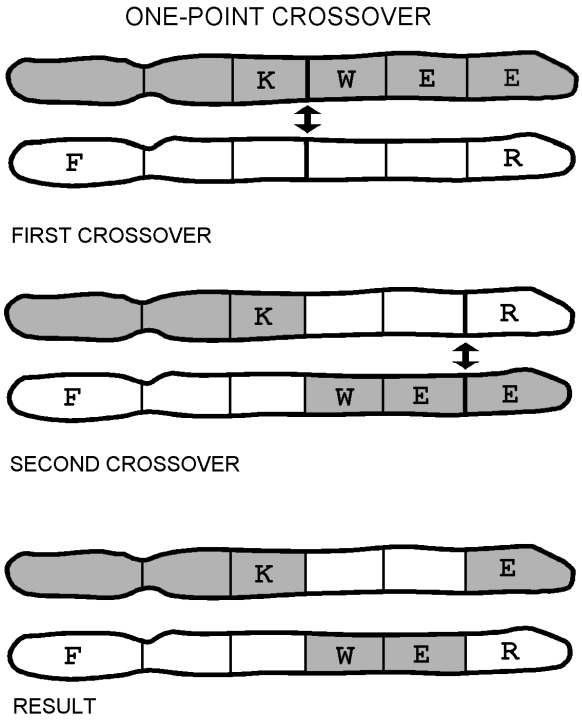


Figure 12. An example of possible steps in one-point crossover to produce a given result.

***n*-point crossover** was proposed by Kenneth De Jong (1975) as a means of diminishing the length and positional biases of the one-point crossover operator. In n-point crossover the number of cut points can vary from one to several. The value of n can be allowed to vary within a GA or given some set value. The variable n is most commonly taken as 2, and called two-point crossover (Mitchell, 1996, p. 172). Figure 13 shows an example of $n=2$ point crossover.

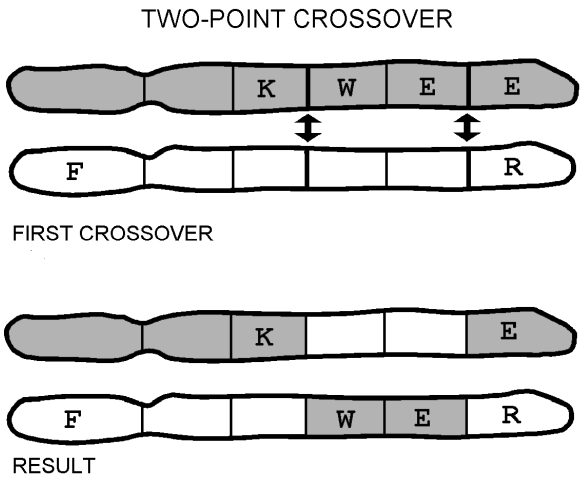


Figure 13. An example of possible steps in $n=2$ point crossover to produce a given result.

Because more sections of the strings are mixed as n increases, it is apparent that n-point crossover will be progressively more disruptive for $n>1$. Also, the denominator in the term $\frac{\delta(H)}{l-1}$ becomes progressively smaller, indicating that the probability of disrupting long schemata is lessened. Because n-point crossover is more disruptive the effect of hitchhiking is reduced. However, it is still not possible to reach all combinations in a single crossing.

Uniform crossover was suggested by Syswerda (1989). As shown in Figure 14, uniform crossover makes no use of cut points, but instead proceeds bit-by-bit down the string, making a decision to swap or to not swap the individual bit positions between parents (50% probability either way). This removes all positional and length biases as all bit positions are handled uniformly. The process is the most disruptive of the crossover operators. This has lead to criticism that uniform crossover prevents co-adapted alleles from ever grouping into larger schemata (Mitchell, 1996, p. 172). Syswerda contends that uniform crossover is more likely to construct new schemata, and is more explorative

than one- or two-point crossover. It can reach any possible point describable by the parents' alleles in one step. This is not possible with either of the operators discussed above.

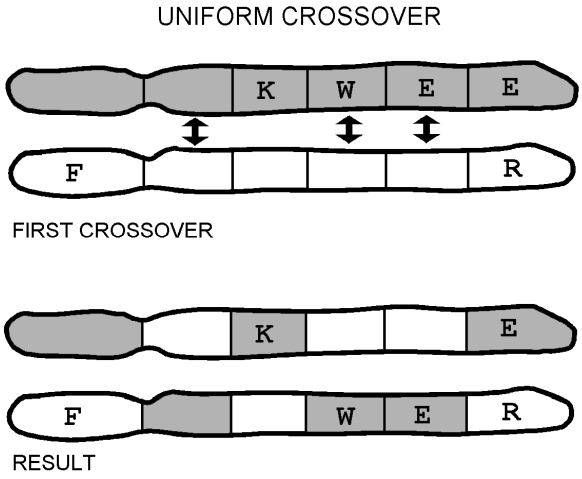


Figure 14. An example of a possible uniform crossover which attains the same result found in Figures 12 and 13.

Parameterized uniform crossover is a modification of uniform crossover proposed by Spears and De Jong (1991), which simply allows for bit swapping probabilities other than 50%. By introducing a swap probability factor, the disruptiveness of the operator can be regulated to give any desired degree of disruption, while maintaining the lack of bias toward length and position. Mitchell cites a range of 0.7 to 0.8 as being generally effective for the swap probability (Mitchell, 1996, p. 173). It is also possible to gradate the swap probability to allow, for instance, greater disruption at the beginning of a run. De Jong and Spears also introduced two measures useful in qualifying crossover operators - **productivity** and **exploration power**. Productivity is the degree to which an operator is capable of generating offspring which differ from their parents. More disruptive operators are then more productive. Exploration power is a gage of how much of the search space can possibly be reached in any one step. For example, if the number of differing bit locations between two binary strings is h (Hamming distance), uniform crossover can reach any of h^2 solutions in one step. Assuming h is equal to the total length l (all bits locations with different values), then one-point crossover at best can only reach $2h$ solutions in one step. With more crossover points, the exploration power increases, as does disruption.

Mutation has traditionally been considered by most GA theorists to be a secondary or "background" operator (Holland, 1975; Goldberg, 1989; Mitchell, 1996). Although ESs use mutation as a primary operator (see Section 2.2), Holland describes the use of mutation alone as "little more than an enumerative plan" (Holland, 1975, p. 110). Holland does note, however, that mutation plays an important roll in replacing alleles lost through genetic drift in limited populations, and insuring at least potential access to all allelic combinations. This can help avoid entrapment at some local optimum.

Mutation operators are generally based on a random change occurring at a determined time interval (measured in breeding instances). The interval is set by a probability of mutation, p_m, which is used to regulate a random bit change per number of bit operations. The value of p_m is usually held constant throughout a run, although Fogarty (1989) has found through testing that varying the mutation rate through progressive generations significantly improves performance. Figure 15 shows a graph of one of Fogarty's variable rate functions.

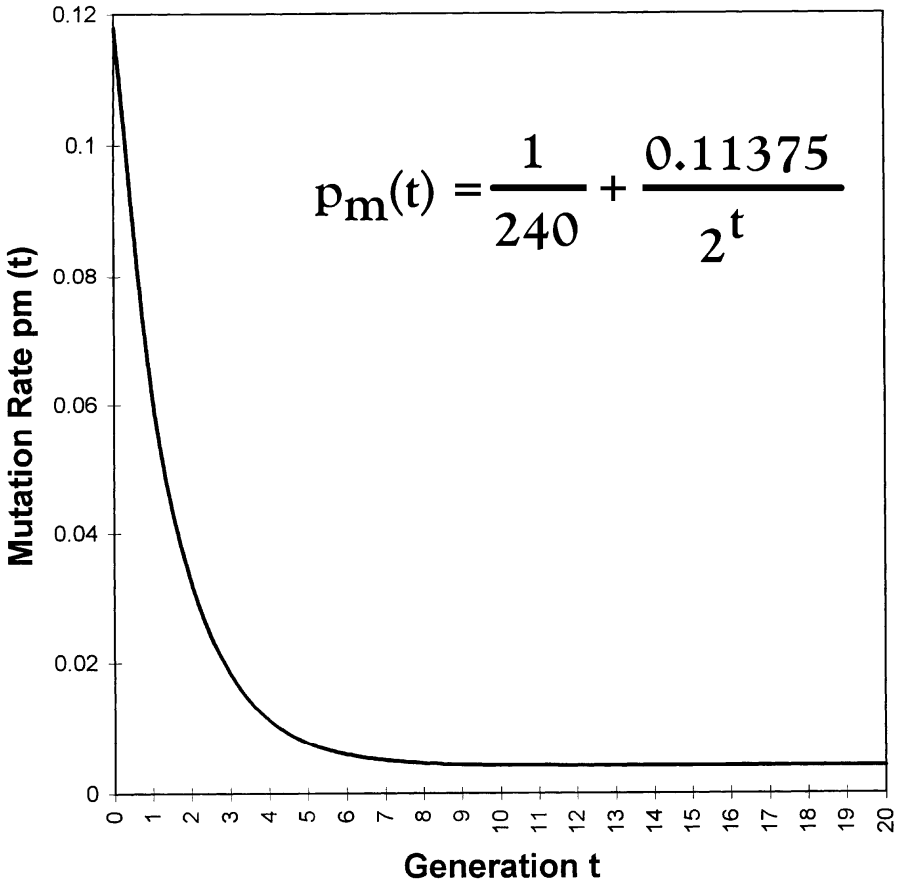


Figure 15. A graph of a variable mutation rate developed by Fogarty.

Consistent with the understanding of mutation as a secondary operator, constant mutation rates are set low by most authors. The range of recommended values for p_m is given in Table 11. All of these values are empirical. Goldberg comments that low mutation rates are in keeping with biological systems. This appears to be true, although observational data is limited. Futuyma cites 10^{-5} per locus, per generation, as a likely rate of mutation for higher animals or plants, and 10^{-9} to 10^{-8} for microorganisms such as bacteria (Futuyma, 1979, p. 244). Of course, mutation rates in biology are no proof of correctness for GAs, and Bäck suggests that low rates found in nature merely belies the conservatism of biological systems, which one would hope to alter in artificial techniques.

p_m	Source
0.001	De Jong (1975, pp. 67-71)
0.01	Grefenstette (1986)
0.005-0.01	Schaffer, et al. (1989)

Table 11. A survey of proposed constant values for the rate of mutation, p_m.

Like crossover, mutation also causes a degree of disruption to schemata. Returning to the equation describing schemata growth and decay from Section 2.1.1, the roll of mutation may now be added.

$$m(H,t+1) \geq m(H,t) \cdot \frac{f(H)}{\bar{f}} \left[1 - p_c \cdot \frac{\delta(H)}{l-1} - o(H) p_m \right]$$

where;

$1 - p_c \cdot \frac{\delta(H)}{l-1}$ is the probability that schema H will survive crossover

where;

p_c is the probability of crossover occurring

$\delta(H)$ is the spanning length of the schema H

l is the length of the string

and,

$o(H)p_m$ is the probability that schema H will survive mutation

where;

p_m is the probability of mutation occurring

$o(H)$ is the order of schema H

The above equation is labeled the Schema Theorem by Goldberg (1989, p. 33) or the Fundamental Theorem of Genetic Algorithms.

Holland also describes **inversion** as a GA operator (1975, p. 106). Inversion mimics a chromosomal aberration of the same name described in Section 1.2.3, and as such it can be classified as a form of mutation. In inversion a section of the bit string is selected between two cut points and reversed. Figure 16 shows the result of an inversion operator.

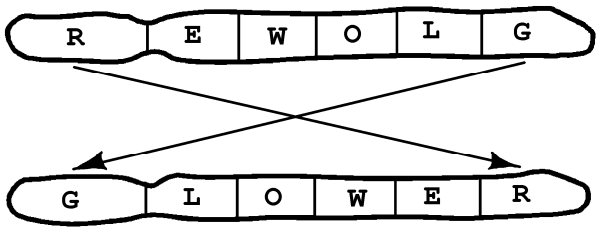


Figure 16. The result of an inversion operator.

2.1.3.1 Determining Fitness

The fitness of an individual is a measure of how well it satisfies a given objective or set of objectives. Fitness is used to rank an individual with respect to other individuals in the population. Fitness can be either **objective** or **relative**.

Objective fitness is a scalar value, determined by some type of analysis of the design defined by the individual. It is an objective, qualitative measure of the degree of success the design achieves in satisfying the goal. Objective fitness values rank the individual in relation to all possible solutions. It is not necessarily a unique value, but the ranking (outside of tied values) is unique.

One problem with objective fitness values is that when coupled with some selection methods (such as roulette-wheel described in Section 2.1.5) the large difference in a particularly high fitness value and the mean fitness value for the population, i.e., the high variance, can cause too many copies of the individual with the high fitness value to be propagated in the next generation, to the deletion of most of the less fit individuals. This can easily happen at the beginning of a run when a random population is generated. The result is that the population quickly converges on the individual with the high fitness found in the initial generation, without adequate exploration of the search space. This

situation is termed **premature convergence**. On the other hand, as the range of fitness differences diminishes toward the end of a run, i.e., low variance, better individuals are lost in the population , and not given sufficient preference during breeding for evolution to continue. The result in this case can be termed a **random walk** through the insufficiently differentiated population.

To address this problem, **fitness scaling** is used in an effort to maintain a more constant level of variance. Goldberg (1989, p. 76) describes this condition, and recommends a linear scaling technique:

$$f' = af + b$$

where:

f'	is the scaled fitness
f	is the raw fitness
a, b	are linear scaling coefficients

Goldberg sets the average fitness values f_{avg} and f'_{avg} equal, and then uses another factor, C_{mult} , to scale the variance of the population by:

$$f'_{max} = C_{mult} \bullet f'_{avg}$$

C_{mult} becomes the number of copies expected in the next generation of the individual with the best fitness. This number will depend on the size of the population, but Goldberg recommends values between 1.2 and 2.0 for small populations between 50 and 100. Figure 17 shows a graph of Goldberg's scaling function. Care must be taken that the lower fitness values do not scale below zero, which can happen as the population variance decreases. Similar scaling methods are sometimes referred to as sigma scaling (Mitchell, 1996, p. 167).

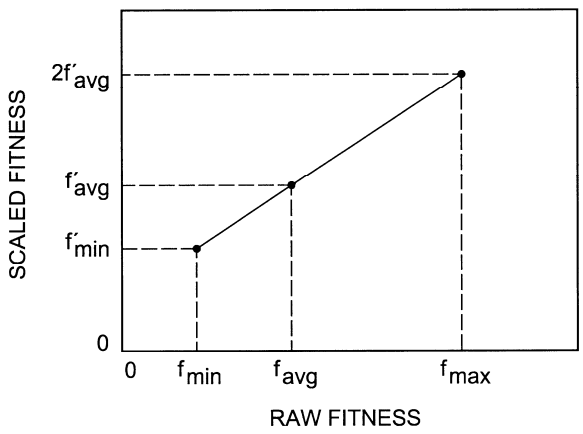


Figure 17. Linear scaling under normal conditions (from Goldberg, 1989, p. 77).

Relative fitness is a gage of how an individual performs in relation to another selected or given benchmark individual. This is particularly useful in cases where conflicts between incompatible objectives make a unified ranking impossible. In this case individuals must be ranked relatively. As a simple example consider the well know child's game of Paper-Scissors-Rock. The game is typical of the type where a winning strategy depends on what is played against it. Table 12 lists the three possible strategy combinations and the outcome of each.

Strategy One	Strategy Two	Result
Scissors	Paper	"scissors cuts paper" Scissors wins
Paper	Rock	"paper covers rock" Paper wins
Rock	Scissors	"rock breaks scissors" Rock wins

Table 12. Strategy combinations and result in the game Paper-Scissors-Rock.

From the results in Table 12 it can be seen than an absolute ranking in this case is not possible. Scissors has better performance relative to paper, and rock ranks better relative to scissors, but paper ranks better relative to rock. So in determining the absolute ranking of the three, there is a problem. In this type of problem, the value of the fitness function can only be given relative to a specific comparison between two specific individuals in the population.

One advantage of a relative fitness function is that it is self scaling. Early in the run an individual need not be one of the absolute best in order to be favored in reproduction. But none the less, better solutions (at least the better of two) are always given preference. Toward the end of a run when the fitness of the population would be both higher and have less variance, finding the relative fitness by direct comparison still favors the better solutions without the need for scaling.

There are generally three types of competitive fitness functions currently in use (Angeline, 1997):

- full competition
- bipartite competition
- tournament competition

Full competition describes a situation where each individual in the population competes against every other individual. The number of evaluations necessary for full competition is:

$$\frac{N(N-1)}{2}$$

where:

N is the population size

This is more evaluations than are necessary in the other schemes, but it can allow a full ranking of the population in cases where that is possible. A reference to this type of fitness function can be found in the work of Axelrod (1987).

Bipartite competition was proposed by Hillis (1991). In bipartite competition the total population is split into two teams from which pairs of individuals compete against each other. In this scheme $N/2$ evaluations take place. This is considerably less than the required evaluations in full competition, but the entire population is not rank ordered.

Tournament competition is a well known scheme from sports events. Different variations exist - e.g., single elimination, double elimination, etc. An example of single elimination can be found in Angeline and Pollack (1993). Tournament competition always proceeds in a series of rounds. In single elimination, the first round begins with all individuals being randomly paired and evaluated. The 'winners' are advanced to the next round and again randomly paired with other winners from the previous round. The process continues until only one individual is left. This individual, having never lost in competition is the best-of-run individual. The rest of the population is ranked by how many wins each had. This is a stepped ranking with progressively fewer individuals in each advancing rank, but never fewer than one per rank. The number of competitions necessary is N-1. Angeline and Pollack (1993) concede that there is a high level of noise in this type of evaluation, but claim that it fosters a more diverse breeding population which is ultimately more explorative and better suited to evolving good solutions.

2.1.3.2 Selecting Individuals

The mechanism which differentiates GA search from random search is the selection mechanism. Although the genetic operators of crossover and mutation discussed in Section 2.1.3, are capable of generating very large numbers of individuals to be

assessed by the fitness function, it is the selection operator which is responsible for evolution toward optima. The selection operator can be described as a dualistic system which must balance the mechanisms of **exploitation** and **exploration**.

Exploitation is the pressure to seek a solution that responds well to the fitness function. The greater the exploitation pressure, the faster the population will converge. As a system converges, the scope of the search narrows, and less of the problem space is sampled for solutions. Also, if a system converges quickly, there will be less sampling of the search space, assuming the search terminates once the system has converged.

Exploration is the counter mechanism to exploitation. Exploration is high when a large area of the problem space is sampled. Exploration is increased by higher rates of mutation or more disruptive crossover operators. Very high exploration, combined with low exploitation, will result in a random walk behavior. In a large design space unguided, random sampling is seldom effective. Without the pressure of exploitation to guide the direction of search, there is a very low probability that the better solutions will be stumbled upon.

The counter condition of high exploitation pressure combined with low exploration, tends to produce a what is termed a hill climber. The system quickly climbs the first 'hill' that it finds, but having reached the top, it gets stuck, and misses the neighboring mountain. A successful selection algorithm is able to balance exploitation and exploration to find the better solutions in a limited time.

Selection methods are closely linked to the type of fitness determination used. Different problem types have shown success with different methods of fitness/selection. The following have come to be recognized as well tested methods for use with GAs:

- Proportional Selection – 'Roulette Wheel'
- Tournament Selection
- Rank-based Selection
- Boltzmann Selection
- Steady-state Selection – 'Generation Gap Method'

In most GA applications, selection actually precedes the genetic operators of crossover and mutation. This avoids needlessly processing individuals that will be later deleted from the population.

Proportional Selection, often called 'Roulette Wheel', was the selection method originally proposed by Holland (1975) based on a probabilistic survival rule analogous to the multi-armed bandit problem found in game theory, and has been widely used since (Goldberg, 1989; Grefenstette, 1997, p. C2.2:1). Greffenstette comments that,

> proportional selection provides a natural counterpart in artificial evolutionary systems to the usual practice in population genetics of defining an individual's fitness in terms of offspring. (Grefenstette, 1997,p. C2.2:1)

The method selects individuals for breeding at a probability proportional to the fitness of the individual. The allusion to a roulette wheel is that slots on the wheel would be proportioned in size according to the individual's fitness. Then the wheel would be spun to select individuals for breeding. In this scheme, individuals with good fitness, receiving bigger slots, have a high probability of being selected multiple times. Conversely the less fit individuals with smaller slots may not get selected at all. The entire population is replaced by newly bred children in each generation. This requires sampling the parent population for each child that is to be produced. Most implementations breed two children from a pair of parents. A problem can occur particularly in smaller populations, that when making a series of stochastic selections (spins of the wheel) the fitter parent individuals might repeatedly be missed, resulting in a regressive generation of offspring. Or seen another way, there is often considerable variation in the expected number of times a certain individual should be selected based on that individual's fitness, versus the actual number of times that individual is actually selected. In an effort to lessen this selection variation, Baker (1987) developed a technique he called **stochastic universal sampling** (SUS). In a SUS algorithm, rather than spinning a roulette wheel with one pointer again and again for each selection, a modified roulette wheel with evenly spaced pointers is spun once. With the number of pointers equal to the number of individuals in the population, the entire set of breeding parents is selected in one spin. This overcomes the fluke chance that poorly fit individuals might be repeatedly selected.

Like other proportionate methods, SUS is usually combined with fitness scaling to prevent early takeover of the population by the initial fit individuals, or late stagnation due to waning selection pressure (see Section 2.1.4).

Tournament selection was alluded to in Section 2.1.4 in reference to relative fitness determination. In addition to being well suited for relative fitness schemes, it is in

general, computationally more efficient as it does not need to sort the population or make global calculations of population mean fitness, as would be necessary to provide fitness scaling. It also lends itself well to parallel implementations.

The basic procedure in tournament selection is to randomly choose two individuals at a time from the population, compare the two and choose the better as a parent. Several variations have been explored by different authors. Two significant parameters which can be varied are:

- tournament size
- unfit pass rate
- return or discard

Tournament size is the number of individuals chosen to compete. As the method was described above, the tournament size, q, was taken as 2. Higher values are of course also possible. Blickle recommends choosing

$$q \in \{6,7,8,9,10\}$$ (Blickle, 1997, C2.3:2)

By raising the value q the selection pressure is increased, i.e. the selection is more exploitative. Taking $q=1$ would essentially be no selection at all, as each individual would be randomly selected, but not compete with any other to breed.

Unfit pass rate refers to the probability that the lesser fit member in the competition might be selected. A pass rate percentage is chosen at the start of a run (e.g., $k=0.75$), and then a comparative value, $0<r<1$, is randomly chosen for each selection tournament. If r>k then the less fit individual in the tournament will be selected. This mechanism reduces the selection pressure (Mitchell, 1996, p. 171).

Return or discard are the two possible actions taken with the selected individuals after a tournament. Either the individuals are returned to the sampled population where they might be chosen again, or they are discarded and bared from further selection.

In addition to the above parameters, Blickle (1997, p. C2.3:2) suggests that SUS can be used in tournament selection to reduce variance, however this complicates the algorithm, and removes most of the advantages of processing speed.

Rank-based selection is a method developed by Baker (1985) which bases selection on a number derived from a ranking of the population by fitness, rather than the actual fitness value. As originally described by Baker the ranking followed a linear order, but other ranking schemes have since been tried using nonlinear, exponential rankings (Michalewicz, 1996, pp. 60-61). Rank-based selection offers the following advantages:

- It eliminates the need for scaling. By removing the reference to actual fitness values it avoids premature convergence due to a 'super individual', as well as stagnation due to lack of fitness differentiation.
- The selective pressure (exploitation vs. exploration) can be controlled by the slope of the linear rank function or by the degree of the exponent.
- In practice ranking is usually a faster, computationally more efficient method.

Ranking schemes are commonly used in Evolutionary Strategies (ES) and Evolutionary Programming (EP), but were slower to gain acceptance in the GA community as they did not seem to follow the Schema Theorem (Bäck, 1996, p. 170). Goldberg cites a source of criticism in the GA community as being the disassociation of the fitness function from the underlying objective function. But he concludes that, "the direct link assumed between fitness and objective function is not grounded in theory and the ranking procedure does provide a consistent means of controlling offspring allocation" (Goldberg, 1989, pp. 124-125).

The method of linear rank-based selection starts with the sorting of all individuals in the population based on fitness. Next each individual is assigned a rank number from 1 for the least fit to N for the most fit, where N is the total population size. The parameter $1 \leq Max \leq 2$ must be selected by the user. Baker recommends a value of Max=1.1 to give good results in most cases. The expected number of copies selected of individual i is given by:

$$ExpNo(i) = Min + (Max - Min)\frac{rank(i) - 1}{N - 1} \qquad \text{(Mitchell, 1996, p. 170)}$$

where;

$ExpNo(i)$	is the expected number of copies of the i^{th} individual
Min	is the minimum number of copies desired
Max	is the maximum number of copies desired
$rank(i)$	is the rank of the i^{th} individual
N	is the population size

As with proportional selection methods, variance in selection spread can be reduced by incorporating a SUS sampling procedure.

From the above description it can be seen that rank-base selection is effectively independent of the actual fitness values. This has a further advantage in cases where the fitness is a particularly noisy function, i.e. not always yielding exactly the same value for the same individual. This might be the case when the fitness value is derived from instrumentation (applications such as smart materials) or when the fitness is determined by direct interaction with a human user (personal evaluation). In these examples rank-based selection removes the need for a quantifiable fitness value. All that is required is the relative ranking of individuals in relation to others in the population.

Boltzmann Selection is derived from Simulated Annealing (SA). SA is conceptually built on an analogy with annealing techniques used to relieve (converge to a single value) the internal stresses bound in the material sample. Annealing works by first raising the temperature of the sample which frees the stresses, and then slowly, under control, lowering the temperature in stages, allowing the stresses to stabilize throughout the material at each stage. In Botzman selection, the selection pressure can be treated analogously to temperature. This offers the advantage that rather than producing constant selection pressure, as is the case with fitness scaling and rank-based selection, the selection pressure can be controlled. At the beginning of the run, the temperature can be more slowly decreased (i.e. low selection pressure). This offers initially a more explorative mode. Then, later in the run, after a better area of the search space has been determined, the temperature may be dropped more rapidly (i.e. higher selection pressure) in order to zero in more rapidly on the optimum. For a more detailed treatment of Boltzmann selection see Mahfoud & Goldberg (1995) and de la Maza & Tidor (1993).

Steady-state selection refers to the state of the population in the course of the GA. The methods discussed thus far have all been **generational**. Successive generations of n (the total population size) children where produced by breeding selected parents, thereby forming the next generation. In a generational method, a totally new population is created in each generation (although some children may be identical to their parents).

In a steady-state method, the population does not have distinct generations. Steady-state is referred to as an **overlapping** method (Sarma & De Jong, 1997, p. C2.7:2) in that there is always some portion of the old population retained in the new population. The term **generation gap** refers to the overlap of members of one population to the next. Holland (1975, p. 91) described this method as adaptive plan R_1. In Holland's plan, one individual is chosen probabilistically, altered by genetic operators, and substituted back into the population in the place of a second, randomly selected individual. The R_1 plan as described by Holland was generally not implemented because of its slow speed and problems with genetic drift in population sizes less than 100. Instead, Holland's alternative adaptive plan, R_d , which is a generational type replacing all parents with offspring, became the preferred method used in GAs (Sarma & De Jong, 1997, p. C2.7:2). In Evolutionary Strategies (ESs) overlapping populations are used in the $(\mu+\lambda)$ scheme (see Section 2.2.2). Evolutionary Programming (EP) as well as Classifier Systems both use overlapping populations.

When the best individual or a percentile of the best individuals are maintained in the population, the scheme is termed **elitist**. Since the section of the population which is overlapped in steady-state methods is typically determined by superior fitness, steady-state methods are usually elitist. Elitist strategies are often employed in combination with other selection methods as well, to either increase selection pressure, or simply prevent the chance of losing the best individual. Particularly in breeding schemes which delete the parents, there is the chance that a superior parent will be degraded by breeding, producing less fit children. If the parents are then deleted from the population, their superior schemata may be lost as well. To prevent this, elitism always maintains the best individual (or top n best) to carry over to the next generation.

2.1.3.3 A Flowchart of GA Procedure

From the preceding Sections it is clear that there is not a standard procedure that can be said to describe a proto-typical GA. Instead, a GA is characterized by containing certain programmatic elements and techniques. These elements are shown below in the form of flowcharts in order to make clear some of the more common GA procedures. Section 2.3 gives more specific detail of the CHC GA which was used in this work.

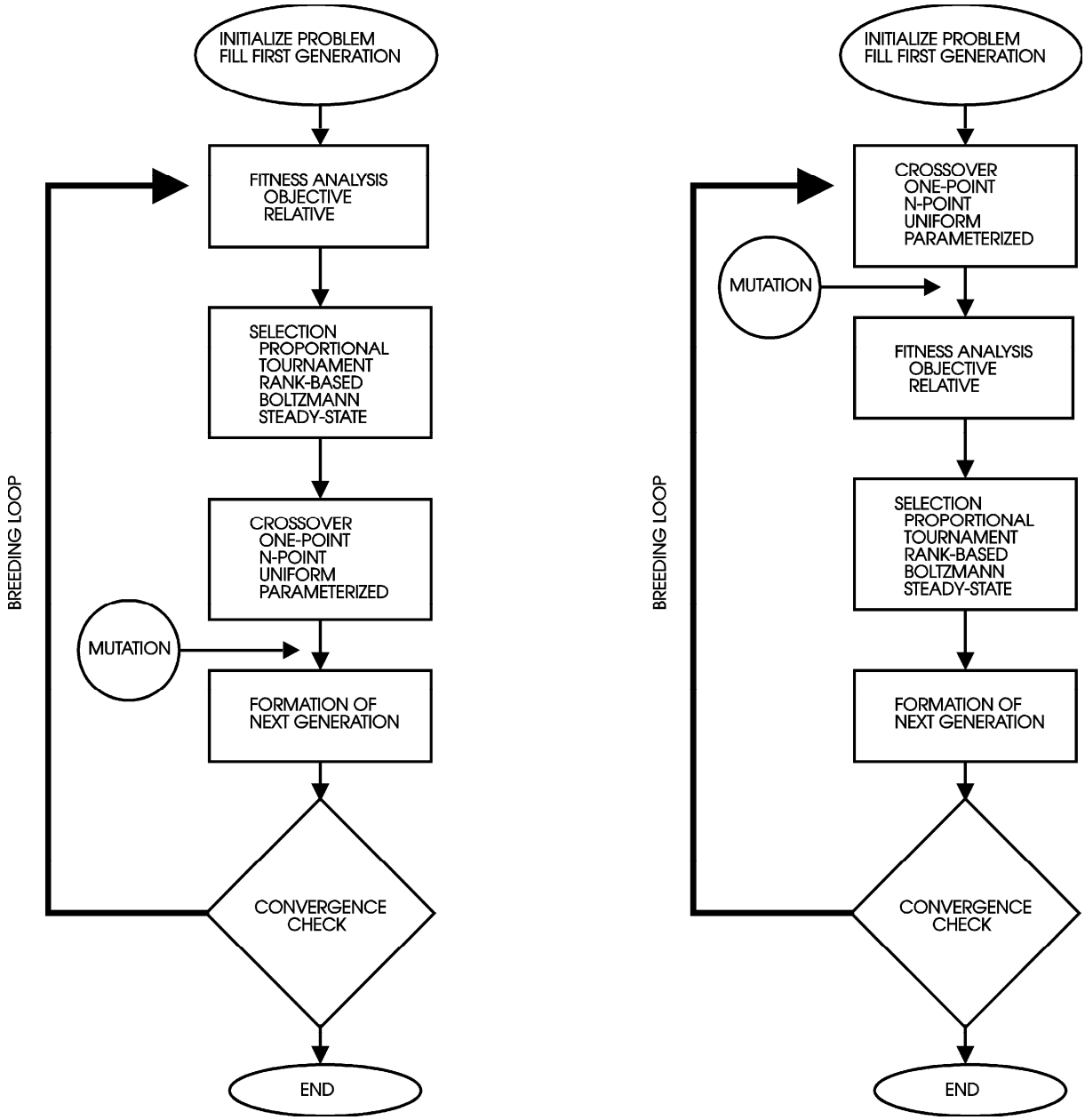


Figure 18. Flowcharts of two possible GA procedures.

2.2 ES Mechanics

Evolutionary Strategies underwent an evolution of their own during their first decade of use. ESs were originally devised during the early 1960's by Ingo Rechenberg, working with fellow engineering students Hans-Paul Schwefel and Peter Biener at the Technical University of Berlin (TUB), to search for shapes of bodies with optimal flow characteristics in wind tunnel tests. Although the development took place concurrently with the work John Holland was doing at The University of Michigan with GAs, the two groups had no contact for several years, and the two methods developed independently on either side of

the Atlantic (Heitkötter & Beasley, 1998). Encouraged by the success of the method, The Evolutionary Engineering Group was founded at TUB, and continued development of ESs. Rechenberg's dissertation, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution* (1973) followed by Schwefel's dissertation, *Evolutionsstrategie und numerische Optimierung* (1977), laid the theoretic foundation for ES, and introduced the method to the engineering community.

2.2.1 ES variants

During the developmental period of ES, a series of strategies were explored which can be listed as follows:

(1 , 1)	random walk
(1+1)	one parent → one child - select from all
(μ+1)	multi-parent → one child - select from all
(μ+λ)	multi-parent → multi-child - select from all
(μ , λ)	multi-parent → multi-child - select from children

(Bäck, 1992)

In the shorthand ES nomenclature used above, the bracketed pair represents two successive generations. The separating *comma* or *plus*, describes the selection method - *comma* being selection from only children, and *plus* being selection from a pool of both children and parents. The number 1, signifies one individual (i.e. not a population). The letters μ and λ refer to populations of parents and children respectively.

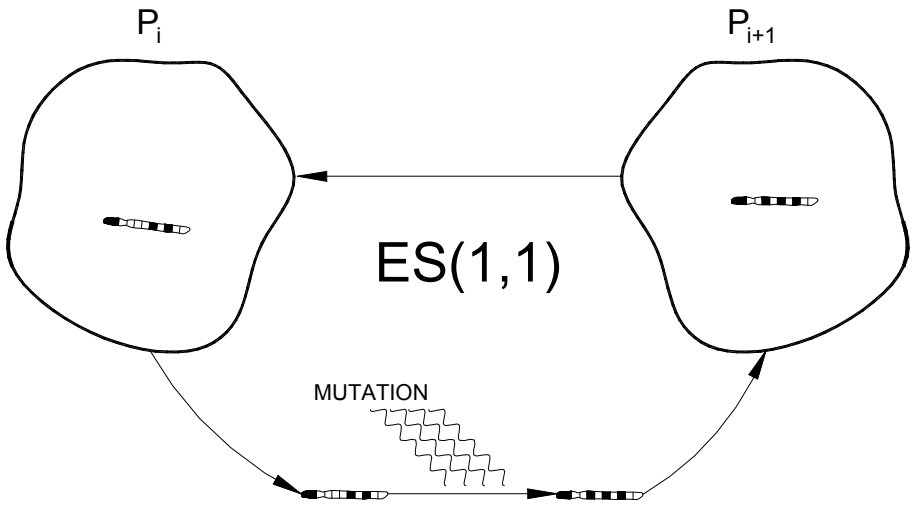


Figure 19. Graphic depiction of a ES-(1,1)

(1 , 1) can be regarded as a point of departure. It is not an ES but merely a random selection method. One parent is randomly mutated to form one child which is then

selected, without regard for fitness, to be the parent in the subsequent generation. With no fitness based selection this it simply a 'random walk' strategy.

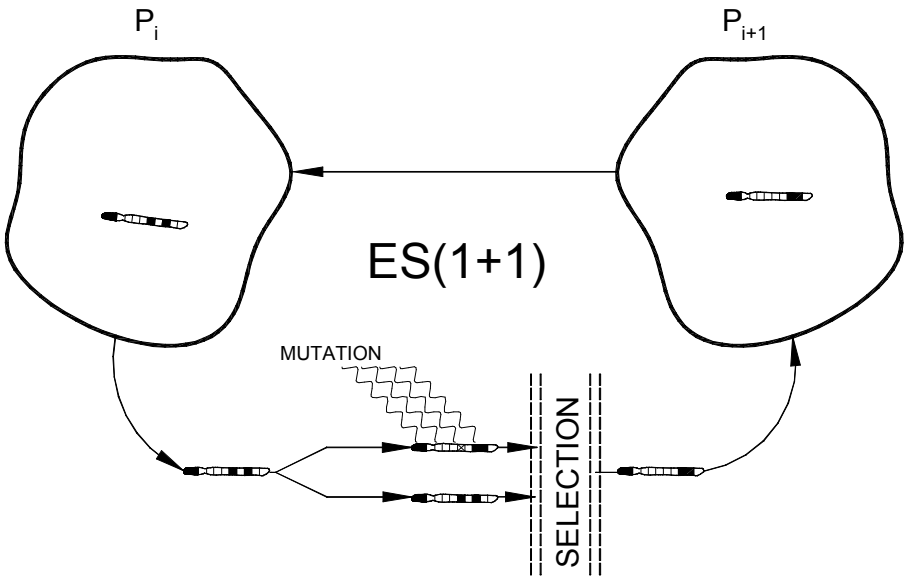


Figure 20. Graphic depiction of a ES-(1+1)

(1+1) was the first ES tried. In the (1+1) strategy populations are not used, but rather one parent is mutated to produce one child. Then a fitness criteria is used to select the better of the two, to be the parent of the subsequent generation. Even this simple ES showed success when tried on the non-linear flow optimization problems at the Technical University of Berlin, TUB. But without the use of populations there is no possibility for parallelism.

(μ+1) was the first ES to make use of populations. In this strategy a population of parents of size $\mu>1$, combine to produce one child. The fittest individuals of the combined parent + child group are carried forward. In this way the new child competes against the established members of population, for a place in μ. With the advent of multiple parents, genetic recombination operators first became possible. ES recombination operators are discussed in Section 2.2.3.

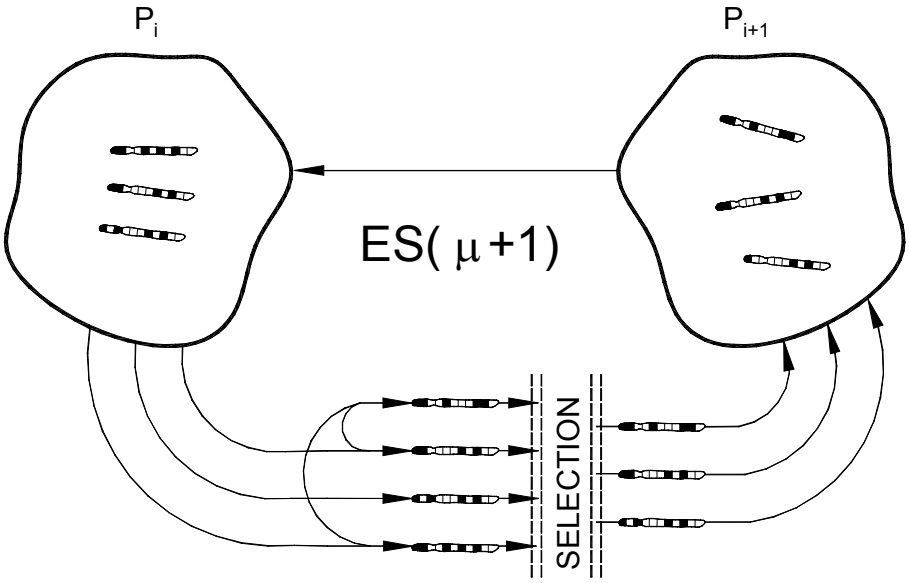


Figure 21. Graphic depiction of a ES-(μ+1)

(μ+λ) extended the population structure to the generation of children. In this strategy, multiple parents produced multiple children which all competed for a place in the next μ generation. This approach can make better use of parallel processing machines, and allows the adaptive evolution of parameters which guide the ES (viz. the σ' parameter which controls the standard deviation of mutation).

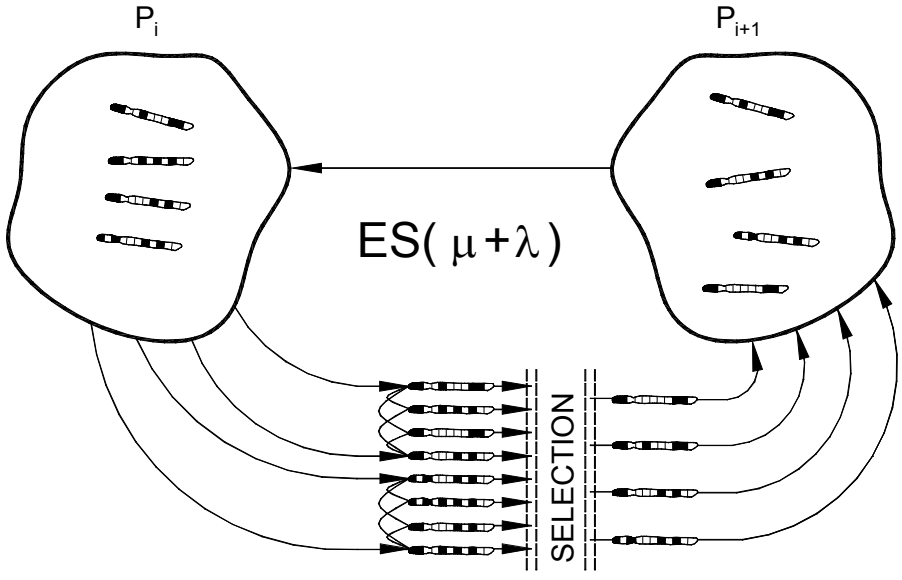


Figure 22. Graphic depiction of a ES-(μ+λ)

(μ , λ) was introduced to avoid stagnation at outdated optima in situations where the fitness function varies over time. The (μ , λ) strategy uses only the child population for selection of the next parent population. In this way the parents all die out with each new generation, allowing the ES to move off of local or previous optima. This may result in the temporary loss of the best fit individuals, but avoids long periods of stagnation, and ultimately allows the population to find new or better regions of the search space (Bäck, et al., 1992).

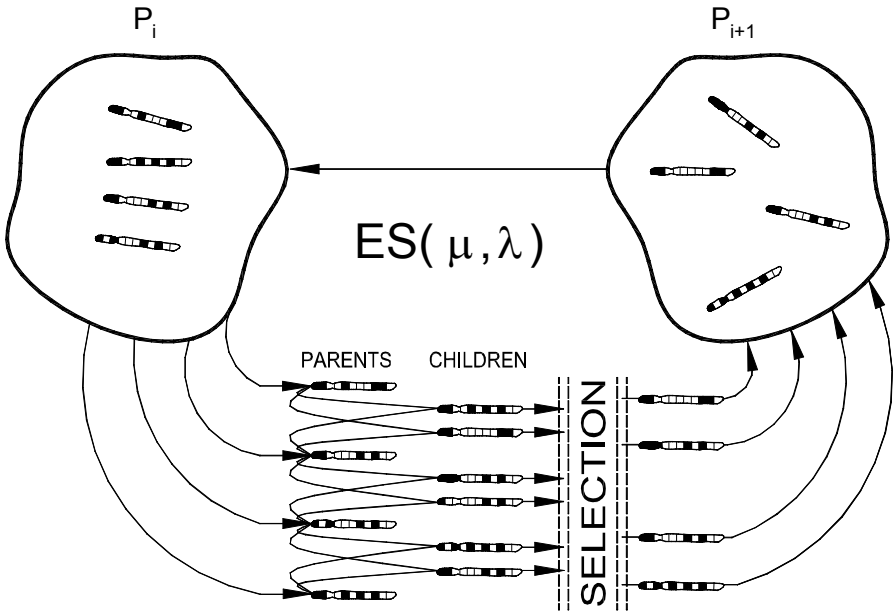


Figure 23. Graphic depiction of a ES-(μ ,λ)

The different ESs described above represent a development in which the last two strategies (viz. (μ+λ) and (μ , λ)), are considered the state of the art. The following three sections describe the basic coding and operators used in (μ+λ) and (μ , λ) ESs.

2.2.2 Coding and Fitness

Having been originally developed for engineering analysis, ESs use real number vectors rather than binary strings to code the parameters. In the (μ+λ) and (μ , λ) strategies, each individual is coded in the form of a vector, $\vec{a}$, which has three component parts:

$$\vec{a} = \left(\vec{x}, \vec{\sigma}, \vec{\alpha}\right)$$

where;

$\vec{x}$ contains the object parameters $\left(x_1,...,x_n\right)$

$\vec{\sigma}$ contains the standard deviations for each x_i, $\left(\sigma_1,...,\sigma_{n_\sigma}\right)$

$\vec{\alpha}$ contains the rotation angles: either set to 0 or calculated

There can be, of course, n number of object parameters needed to describe an individual. Each of these object parameters can be mutated individually by an amount based on the associated standard deviation recorded in $\vec{\sigma}$. The case of $n=2$ is easily depicted in two dimensions, and is therefore used as an illustration in Figures 24 and 25. The ellipsoidal rings shown in Figures 24 and 25 represent contours of equal probability for mutations of the two parameters, x_i and x_j.

The strategy is further enhanced by the addition of rotation angles, α , which orient the direction of most extreme mutation based on the variances and covariance of the expected sets of mutations using:

$$\tan(2\alpha_{ij}) = \frac{2c_{ij}}{\sigma_i^2 - \sigma_j^2}$$ (Bäck, 1996, p. 70)

where;

c_{ij} is the covariant associated with σ_i and σ_j

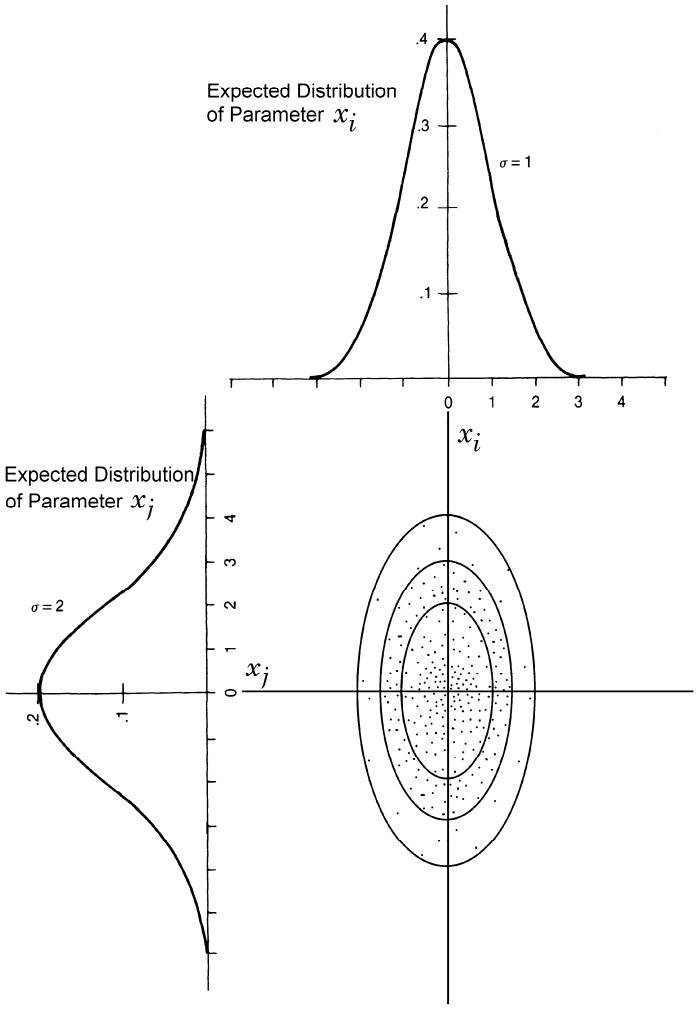


Figure 24. An example of normally distributed expected mutations for two variables (x_i: σ=1.0 and x_j: σ=2.0). The σ value associated with a variable evolves along with the variable itself as part of the strategy.

Figure 25 shows the affect of using a preset rotation angle α=0, compared with using a calculated by the above equation. By orienting the mutation ellipsoids toward a more preferential direction, the step size is better tuned to the local contours of the search space.

It is a significant feature that the ($\mu+\lambda$) and (μ , λ) ESs include the σ values in $\vec{a}$. In this way, the σ values are able to optimally adapt to the specific contours of the search space. In ES this is referred to as self-adaptation (Schwefel 1995).

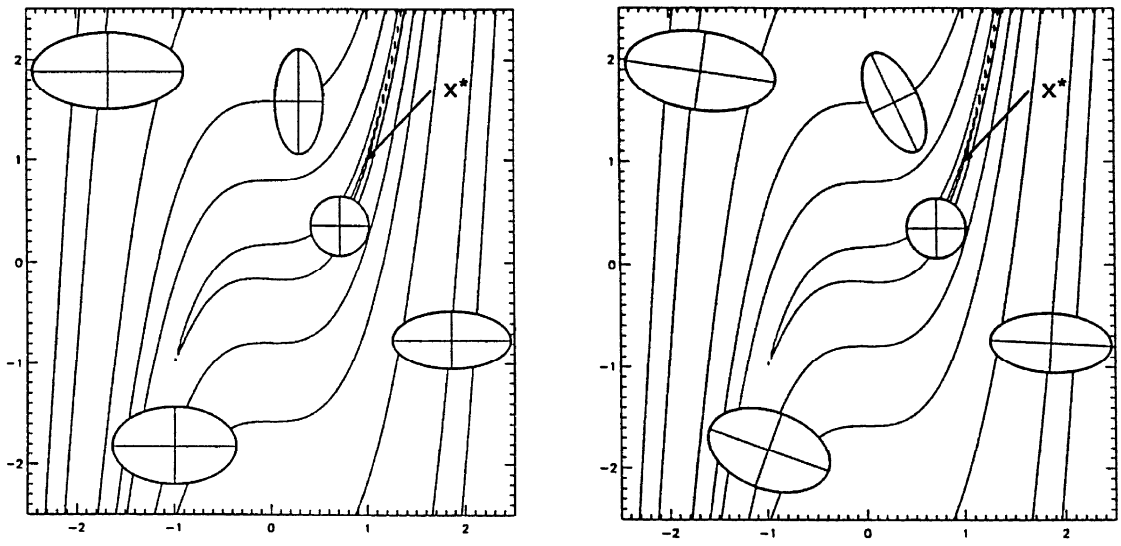


Figure 25. Left: Mutational Ellipsoids Oriented along the Coordinate Axes.
Right: Use of a Rotation in Orienting the Mutational Ellipsoids. (from Bäck, 1996)

2.2.3 Mutation

Unlike mutation in GAs which plays a secondary roll as a 'background' operator, in ES mutation is a primary operator. It was, in fact, originally the only operator used. From the discussion of coding in Section 2.2.1, the description of the σ and α elements makes the importance of mutation in ES clear. The mutation operator, which is denoted by $m_{\{\tau,\tau',\beta\}}$, acts on each of the three parts of the coded individual $\vec{a}=(\vec{x},\vec{\sigma},\vec{\alpha})$ to produce the mutated individual $\vec{a}'=(\vec{x}',\vec{\sigma}',\vec{\alpha}')$ (Bäck, 1996, p. 71). Similar to GAs, mutation in ES is an asexual operator. However, whereas mutation only occurs in GAs with a small probability (usually pm=0.005 to 0.01), in ES it is carried out on each individual, but to a degree that varies based on a Gaussian distribution. The difference in the treatment of mutations between GA and ES has to do with the difference in the way individuals are coded in the two. Since GAs are usually coded as binary strings, a mutation results in a

bit being flipped, from 0 to 1 or 1 to 0. Therefore, in GAs the mutation must either occur or not. The use of the binary bits precludes any variation in degree to which a bit will mutate. ES, on the other hand, uses real number vectors to code the individuals. In ES each real number in the coded vector can be mutated to varying degrees. The real numbers can be multiplied by a random factor which has a normal (Gaussian) distribution given by $N(m,\sigma)$, where m is the mean of the curve or expectation, and σ is the standard deviation. In ES formulation $N(0,1)$ signifies a normalized random factor, with expectation of 0 and standard deviation of 1. $N_i(0,1)$ indicates that a new random variable is sampled for each i element. Mutations are applied to the three components of individuals as follows:

$$\sigma_i' = \sigma_i \cdot e^{\left(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1)\right)}$$
$$\alpha_j' = \alpha_j + \beta \cdot N_j(0,1)$$
$$\vec{x}' = \vec{x} + \vec{N}\left(\vec{0}, \mathbf{C}\left(\vec{\sigma}', \vec{\alpha}'\right)\right) \qquad \text{(Bäck, 1996, p. 71)}$$

where;

$\tau \propto \left(\sqrt{2\sqrt{n}}\right)^{-1}$

$\tau' \propto \left(\sqrt{2n}\right)^{-1}$

$\beta \approx 0.0873$ in radians (about 5°)

$\vec{N}\left(\vec{0}, \mathbf{C}\right)$ denotes a normalized random vector with m= $\vec{0}$ and $\mathbf{C}$ the covariance matrix $\mathbf{C}^{-1} = \mathbf{C}^{-1}\left(\vec{\sigma}', \vec{\alpha}'\right)$

The constants τ, τ' and β , Bäck describes as "learning rates", and are similar to parameters used in neural networks. The proportionalities given above come from Schwefel (1977, pp. 167-168), and Bäck comments further that the values for τ and τ' are usually taken as unity (Bäck, 1996, p. 72). In application, mutation is first used to find new values for each σ_i' and α_j', which are then used in turn to mutate the values of $\vec{x}'$. As mentioned at the end of Section 2.2.1, it is significant that the mutation parameters σ and α are incorporated into the evolving structure. By doing this, the ES is able to optimize the mutation parameters to fit the solution landscape of the particular problem. Schwefel and Bäck refer to this as "self-adaptation" (Schwefel, 1995; Bäck, 1996). The σ and α terms provide a key to the local topology of the search space. This information is then exploited to moderate the speed of convergence through the step

size, and this moderation is tailored for each x_i rather than being set deterministically as a single value for all x_i parameters (as is ordinarily the case in a GA).

2.2.4 Recombination

Although recombination was not used at all in the initial (1+1) ES, there are several recombination techniques which are currently used in ES when $\mu>1$, for example in the $(\mu+\lambda)$ and (μ,λ) strategies. In these latter strategies, a recombination operator is applied λ times to the population of μ parents, P_μ, with each application resulting in one offspring. The offspring together form the population of λ children, P_λ. The techniques used by recombination operators divide into two basic forms: sexual (based on two parents) and panmictic (based on the entire P_μ). For each of these forms of recombination there are two methods of execution which can be applied: discrete and intermediate.

- Sexual Recombination
  - discrete
  - intermediate
- Panmictic Recombination
  - discrete
  - intermediate

In addition, it will be remembered that each individual is represented by a vector comprised of object parameters, standard deviations and rotations in the form $\vec{a}=(\vec{x},\vec{\sigma},\vec{\alpha})$. Each of these three component vectors can be recombined independently by a different operator. Thus, it is possible, for example, to have the object parameters be recombined using an intermediate sexual method, following the use of a panmictic discrete recombination of the standard deviations.

Discrete recombination randomly selects individual (discrete) values from the pool of parents involved in the recombination (two or more). This is similar to the uniform crossover operator used in GAs, as discussed in Section 2.1.3.

Intermediate recombination delivers some form of averaging (usually the mean) of the values of two or more selected individuals. This is of course an option with real coded traits that is not possible with binary alleles used by GAs. A *generalized* intermediate case has also been suggested by Schwefel (Bäck, 1996, p. 74), in which a

weight factor, $0<\chi<1$, is chosen at random, and used in a weighted averaging of the parent traits. When $\chi=0.5$, the resulting average is the mean.

Sexual recombination begins by selecting at random two individuals from P_μ. A child is then derived using either discrete or intermediate recombination. There is usually no filter used that would prevent a parent from being selected again in subsequent breeding. To select the same individual for both parents will of course produce a child clone of the parents. Weighted probabilistic methods (such as 'roulette wheel' in GAs) for the selection of parents are not used. Parents are always chosen at random from P_μ with equal probability.

Panmictic recombination begins with the random selection of one parent from P_μ. This first parent is then maintained, while new mates are chosen from P_μ to derive each trait of the child individual. The individual traits may result from either discrete or intermediate recombination, or selected combinations (e.g., intermediate to derive $\vec{x}$, and discrete to derive $\vec{\sigma}$ or $\vec{\alpha}$).

Using these basic recombination operators, a variety of combinations can be achieved. However, Schwefel recommends that discrete methods be used when recombining object variables, and (panmictic) intermediate operators when recombining the strategy parameters $\vec{\sigma}$ or $\vec{\alpha}$.

2.2.5 Selection

The selection operator for ES is defined by the strategy variant, viz. $(1+1)$, $(\mu+1)$, $(\mu+\lambda)$, (μ,λ) as described in Section 2.2. All of these strategies are strictly elitist with the exception of (μ,λ). In the case of (μ,λ), only individuals from the child generation, P_λ, are chosen to proceed. This means that for any selection to take place in (μ,λ), $\lambda>\mu$. Being non-elitist, it is expected that the fittest individual will from time to time be lost. But, as with non-elitist GAs, it is expected that in the long run better alleles will be preserved, and by limiting the life span of any individual, the population as a whole will be better able to move off of local optima. Schwefel also cites cases where the optima change over time, as benefiting from a (μ,λ) strategy.

Finally, Bäck points out that the character of the search is governed to an extent by the size of μ, as well as the ratio μ/λ. Strategies using smaller values of μ will converge more quickly, but tend to sacrifice the breadth of the search. Also, using small μ values, can loose good object parameters due to genetic drift described in Section 1.3.3. The lower the μ/λ ratio, the less selective pressure will be present. For a (μ,λ) strategy with $\mu/\lambda=1$ there is actually no selection because every λ is needed to fill the next μ. This results in a random walk.

2.3 CHC Mechanics

The search and exploration engine used in this study has been patterned after the CHC Genetic Algorithm, developed by Larry J. Eshelman and J. D. Schaffer of Philips Laboratories (Eshelman, 1991). CHC represents one of the more recent directions in GA development. It combines a highly disruptive recombination operator, which allows for thorough exploration, with an elitist selection operator for good convergence velocity. CHC is successful with small populations (ca. 50). In addition, it makes use of a breeding filter that allows only the fraction of the parent population which promises more productive pairings to produce children. This makes the process of breeding more efficient. These operators combine to give the CHC good exploration qualities, and a high rate of convergence. Good exploration qualities are desirable to provide the designer with a broad view of the solution space. Also, for the process to function interactively, speed is needed in finding good solutions. Since the CHC offers advantages in thorough exploration and rapid convergence, it was chosen as the basis for the geometry optimization preformed in this study.

2.3.1 CHC Cycle

A CHC can be seen as running in two, nested cycles. The inner cycle uses an elitist GA with half uniform crossover. The outer cycle begins with a population of mutated individuals based on the best result of the previous cycle. Figure 26 shows a diagram of the CHC outer cycle (labeled cycle). The inner cycle is contained in the "run GA" circle.

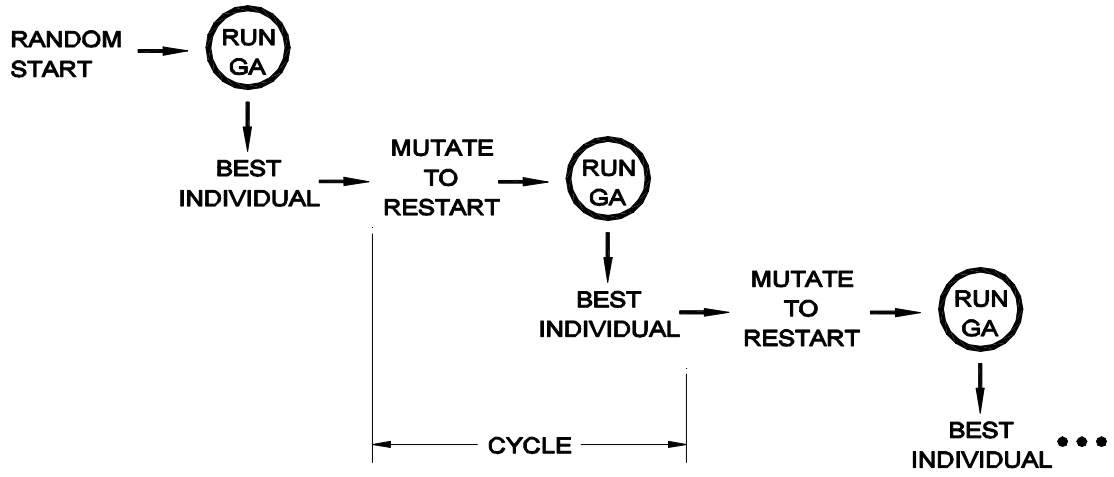


Figure 26. The nested cycles of the CHC-GA.

The outer cycle runs a GA a series of times. The very first cycle generally begins with a randomly generated population of parents. A GA, as described in Section 2.3.2, is run until the population converges. At the end of each GA run, the best individual is selected, and mutated repeatedly to form a new *restart* population. In contrast to traditional GAs, the CHC does not make use of mutation during the breeding cycle. Instead, mutation is applied only during the restart phase. During restart, the best individual from the converged GA is mutated by randomly flipping a percentage (ca. 35%) of the binary bits in the string. The mutation process is repeated until the new population is filled. One copy of the unmutated source individual is retained in the next cycle as well. The CHC cycle is terminated after a chosen number of restarts are made without improvement to the best individual.

There are advantages in both thoroughness of search as well as computational efficiency which are realized by the cyclic CHC structure. Because the GA used by the CHC incorporates highly disruptive recombination, mutation is ineffective (actually unneeded) in providing diversity to the breeding population. By positioning mutation as a concentrated event at the start of the CHC cycle, new areas of the search space are more likely to be encountered, and exploited by the ensuing GA portion of the cycle.

Also the cyclic CHC structure provides a mechanism for tailoring the search to the degree of difficulty present in a particular problem. In easy problems where the optimum solution is rapidly found in the first few cycles, the CHC will terminate after a few cycles without improvement. In more difficult problems, the cycles will continue to show

improvement, and the CHC will continue to explore new areas of the search space by repeated restarts, until a good solution is found. In this way the CHC adjusts the run time to the degree of difficulty of a given problem.

2.3.2 The GA Run

The GA used in the CHC is nontraditional in several aspects. Most significant is the highly disruptive recombination operator used, half uniform crossover (HUX). Uniform crossover, as developed by Syswerda, is discussed in Section 2.1.3. CHC amplifies the disruptiveness further by providing a filter which allows only parents whose Hamming distance is above a certain threshold to breed. The Hamming distance is a direct

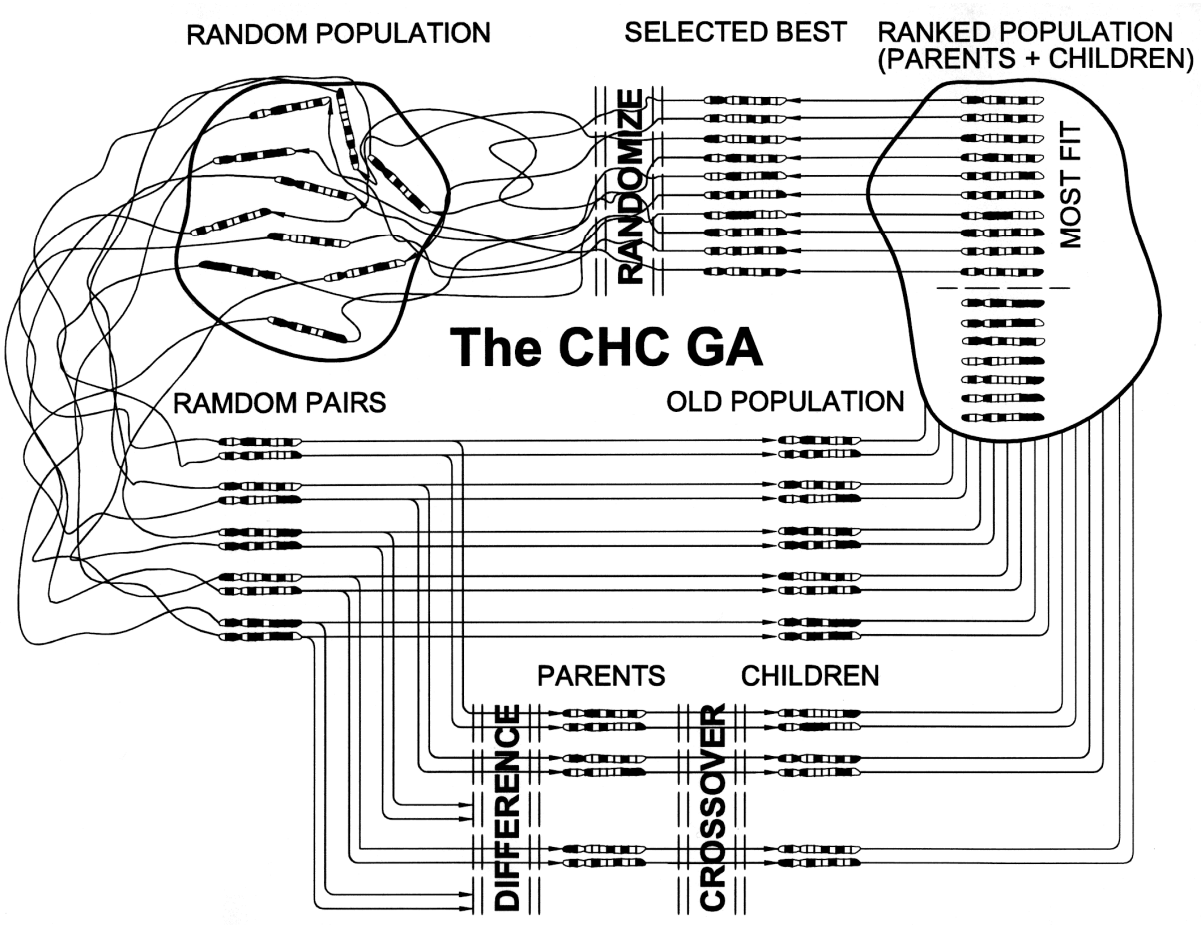


Figure 27. Graphic depiction of the CHC-GA.

measurement of the number of differing bit positions of two binary strings. The difference threshold is set so that only the more different (larger Hamming distance) fraction of the parent population is bred. By promoting the recombination of more different individuals, better exploration is insured. By discouraging the recombination of

more similar individuals, convergence is retarded by slowing the takeover rate of a group of similar, better performing individuals. At the same time, because only some fraction of the entire parent population is actually breeding, the amount of calculation needed for the generation, analysis and sorting of the resulting smaller child population, is less. Eshelman refers to this filtering mechanism as "avoiding incest" (Eshelman, 1991, p. 273). The difference threshold is initially set at one forth of the binary string length (L/4), which is half the expected Hamming distance of two randomly generated strings. As the population begins to converge the difference threshold is decremented each time a generation occurs which produces no children. In this way breeding is maintained only amongst the most differing individuals in the population at any given time.

The recombination operator HUX is a form of uniform crossover in which one half of the differing bits are exchanged between two parents. This insures a high level of disruption. The resulting two children are both added to the new population along with the two parents, and all subsequently undergo an elitist 'survival' selection (Eshelman, 1991, p. 266). This type of selection, although not typical for GAs, is actually the same as that used by $(\mu+\lambda)$-ES, as described in Section 2.2. During selection, the parent and child generations are combined, and the best individuals are selected to fill the next generation. The process is strictly elitist. In addition one copy of the best individual is always maintained in the restart population.

3 Reference List

Angeline, Peter J. "Competitive fitness evaluation", in: Bäck, Fogel and Michalewicz (ed.) *Handbook of Evolutionary Computation*, Oxford University Press, Oxford, 1997.

Angeline; P. J. & Pollack, J. B. "Competitive environments evolve better solutions for complex tasks", in: S. Forrest (ed.) *Proc. 5th Int. Conf. on Genetic Algorithms (Urbana-Champaign, IL, July 1993)*. Morgan Kaufmann, San Mateo, CA, 1993. pp. 264-270.

Arms, Karen and Camp, Pamela. *Biology*. Saunders College Publishing, New York, 1987.

Axelrod, R. "Evolution of strategies in the iterated prisoner's dilemma", in: L. Davis (ed.) *Genetic Algorithms and Simulated Annealing*. Pitman, Boston, 1987. pp. 32-41.

Bäck; Hoffmeister and Schwefel. "A Survey of Evolution Strategies", in: *Proceedings of the Fourth International Conference on Genetic Algorithms*. Morgan Kaufmann Publ., San Mateo, California, 1992.

Bäck, T. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, Oxford, England, 1996.

Baker, Ellie. "Adaptive selection methods for genetic algorithms", in: Grefenstette, J. (ed.) *Proceedings of the Second International Conference on Genetic Algorithms*. Erlbaum, Hillsdale, NJ, 1985.

Baker, Ellie. "Reducing bias and inefficiency in the selection algorithm", in: Grefenstette, J. (ed.) *Proceedings of the Second International Conference on Genetic Algorithms*. Erlbaum, Hillsdale, NJ, 1987.

Beasley David. "Possible applications of evolutionary computation", in: Bäck, T., Fogel, D. B. and Michalewicz, Z. (eds.) *Handbook of Evolutionary Computation*. Oxford University Press, Oxford, New York, 1997.

Blickle, T. "Tournament selection", in: Bäck, T., Fogel, D. B. and Michalewicz, Z. (eds.) *Handbook of Evolutionary Computation*. Oxford University Press, Oxford, New York, 1997. p. C2.3:2

Crow, J. & Kimura, M. *An Introduction to Population Genetics Theory*. Harper & Row, New York, 1970.

Darwin, Charles. *On the origin of species by means of natural selection, or, The preservation of favoured races in the struggle for life*. J. Murray, London, 1859.

Dawkins, Richard. *The Blind Watchmaker*. Norton, W.W. & Co., London, 1986.

De Jong, K. A. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. University of Michigan, Diss, 1975.

de la Maza, M. and Tidor, B. "An analysis of selection procedures with particular attention paid to proportional and Boltzmann selection", in: Forrest, S. (ed.), *Proceedings on the Fifth International Conference on Genetic Algorithms*. Morgan Kaufmann Publ., San Mateo, California. 1993.

Doyle, Arthur Conan, Sir. *The Adventures of the Dancing Men and Other Sherlock Holmes Stories*. Dover Publ., New York, 1997.

Eshelman, Larry. "The CHC Adaptive Search Algorithm: How to Have Safe Search When Engaging in Nontraditional Genetic Recombination", in: *Foundations of Genetic Algorithms*. Morgan Kaufmann Publ., San Mateo, California. 1991.

Eshelman, Larry. and Schaffer, J. "Real-coded Genetic Algorithms and Interval-schemata ", 1992.

Fogarty, T. C. "Varying the probability of mutation in the genetic algorithm", in: Schaffer, J. D. *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kaufmann Publ., San Mateo, California, 1989.

Fogel, L. J., Owens, A. J. and Walsh, M. J. "On the evolution of artificial intelligence", in: *Fifth National Symp. on Human Factors in Electronics - Proceedings*. IEEE, San Diego, CA, 1964.

Futuyma, Douglas J. *Evolutionary Biology*. Sinauer Associates, Sunderland, Massachusetts, 1979.Goldberg, David E. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, Reading, Mass, 1989.

Goldberg, D. E.; Deb, K.; and Clark, J. "Genetic Algorithms, Noise, and the Sizing of Populations", IlliGAL Report No. 91010. Univ. of Illinois, Urbana-Champaign, IL. 1991.

Goldberg, D. E.; Deb, K.; and Thierens, D. "Toward a Better Understanding of Mixing in Genetic Algorithms", IlliGAL Report No. 92009. Univ. of Illinois, Urbana-Champaign, IL. 1992.

Goldberg, D. E.; Deb, K.; Kargupta, H.; and Harik, G. "Rapid, Accurate Optimization of Difficult Problems Using Fast Messy Genetic Algorithms", IlliGAL Report No. 93004. Univ. of Illinois, Urbana-Champaign, IL. 1993.

Goldberg, David E. "The Design of Innovation: Lessons form Genetic Algorithms", IlliGAL Report No. 98004. Univ. of Illinois, Urbana-Champaign, Ill. 1998a.

Goldberg, David E. "The Race, the Hurdle, and the Sweet Spot: Lessons from Genetic Algorithms for the Automation of Design Innovation and Creativity", IlliGAL Report No. 98007. Univ. of Illinois, Urbana-Champaign, IL. 1998b.

Goldberg, David E. *The Design of Innovation : Lessons from and for Competent Genetic Algorithms*. Kluwer Academic Publishers, Boston, 2002.

Gottschalk, Werner. *Allgemeine Genetik*. Georg Thieme Verlag, Stuttgart, 1984.

Grefenstette, J. "Optimization of Control Parameters for Genetic Algorithms", in: *IEEE Transactions on Systems, Man and Cybernetics*, v SMC-16, n 1, Jan-Feb 1986. pp. 122-128.

Grefenstette, J. "Proportional selection and sampling algorithms", in: Bäck, T., Fogel, D. B. and Michalewicz, Z. (eds.) *Handbook of Evolutionary Computation*. Oxford University Press, Oxford, New York, 1997. p. C2.2:1

Harik, G & Goldberg, D. E. "Learning Linkage", IlliGAL Report No. 96006. Univ. of Illinois, Urbana-Champaign, Ill. 1996.

Heitkötter, J. & Beasley, D. (eds.) *The Hitch-Hiker's Guide to Evolutionary computation*. FAQ in: comp.ai.genetic. Issue 6.1, 20 March 1998.

Hillis, W. D. "Co-evolving parasites improve simulated evolution as an optimization procedure", in: S. Forrest (ed.) *Emergent Computation*, MIT Press, Cambridge, MA, 1991. pp. 228-235.

Holland, John H. *Adaptation in Natural and Artificial Systems*. The Univ. of Mich. Press, 1975.

Kargupta, H. *SEARCH, Evolution, and the Gene Expression Messy Genetic Algorithms*, Los Alamos National Laboratory; Los Alamos, NM, 1996.

Lund, H. H.; Pagliarini, L., Miglino, O. "Artistic Design with Genetic Algorithms and Neural Networks". in : *Proceedings of 1NWGA*, Alexander, J. T. (ed.), Univ. of Vaasa, Vaasa, Finland, 1995.

Mahfoud, S. W. and Goldberg, D. E. "Parallel recombinative simulated annealing: a genetic algorithm", in: *Parallel Computing*, Vol. 21, 1995.

Mendel, Gregor: "Versuche über Pflanzen-Hybriden", in: Krizenecky (ed.). *Fundamenta Genetica*. Anthropological Publications. Oosterhout, The Netherlands, 1965.

Mettler, Greg and Shaffer. *Population Genetics and Evolution*. Prentice Hall, 1988.

Michalewicz, Z. *Genetic Algorithms + Data Sources = Evolution Programs*. 3rd ed., Springer, Berlin, 1996.

Minkoff, Eli C. *Evolutionary Biology*. Addison-Wesley, Reading, Mass., 1983.

Mitchell, M. *An introduction to genetic algorithms*. The MIT Press, Cambridge, Mass., 1996

Powell, D., Skolnick, M. "Using Genetic Algorithms in Engineering Design Optimization with Non-linear Constraints", in: *Proceedings on the Fifth International Conference on Genetic Algorithms*. Morgan Kaufmann Publ., San Mateo, California. 1993.

Rechenberg, Ingo. *Evolutionsstrategie Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Friedrich Frommann Verlag. Stuttgart, 1973.

Russell, Peter J. *Genetics*. Harper Collins, New York, 1992.

Sarma, J. and De Jong, K. "Generation gap methods", in: Bäck, T., Fogel, D. B. and Michalewicz, Z. (eds.) *Handbook of Evolutionary Computation*. Oxford University Press, Oxford, New York, 1997. p. C2.7:2

Schaffer, J. David (ed.). *Proceedings of the Third International Conference on Genetic Algorithms: George Mason University, June 4 - 7, 1989*. Morgan Kaufmann Publ., San Mateo, Calif., 1989.

Schwefel, H-P. *Numerische Optimierung von Computor-Modellen mittels der Evolutionsstrategie : mit einer vergleichenden Einführung in die Hill-Climbing- und Zufallsstrategie*. Birkhauser, Stuttgart, 1977.

Schwefel, H-P. *Evolution and Optimum Seeking*. John Wiley & Sons, New York, 1995.

Schwefel, Hans-Paul. "Advantages of evolutionary computation over other approaches", in: Bäck, T., Fogel, D. B. and Michalewicz, Z. (eds.) *Handbook of Evolutionary Computation*. Oxford University Press, Oxford, New York, 1997.

Spears, W. M., and De Jong, K. A. "On the virtues of parameterized uniform crossover", in: Belew, R. K., and Booker, L. B. (eds.), *Proceedings on the Fourth International Conference on Genetic Algorithms*. Morgan Kaufmann Publ., 1991.

Syswerda, G. "Uniform crossover in genetic algorithms", in: *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kaufmann Publ., 1989.

Watson, J. D. and Crick, F. H. C, et al. "Molecular Structure of Nucleic Acids", in: *Nature,* Vol. 171, No. 4356, MacMillan & Co., London, 1953.