A Brief Overview of Genetic Optimization

Olfa Nasraoui
Department of Computer Engineering & Computer Science
University of Louisville,
olfa.nasraoui_AT_louisville.edu

The Genetic Algorithm

Evolutionary Algorithms started in the 1950's with [Fra57] and [Box57]. They form a powerful family of problem solving tools that attempt to mimic the natural selection process and evolution in order to find the optimal solution to a given problem. The most well known evolutionary methodologies are Evolutionary Programming [FOW66], Evolution Startegies [Rec73], and Genetic Algorithms [Hol75]. The first two methodologies emphasize mutation as the key search operator while the last one emphasizes crossover. In particular, the Genetic Algorithm (GA) has recently gained popularity due to its simplicity, close analogy with biological evolutionary systems, and its domain independent representation via bit-strings.

A GA [Hol75,Gol89,SJB$^+$93] works on a population of individuals representing possible solutions to a particular problem. Each individual is evaluated using a fitness value which is a measure of how well adapted this individual is to its environment. As in nature, individuals mate and reproduce in a GA with a reproduction rate that is proportional to their fitnesses. The individuals are represented by their genetic material or genotype. This is accomplished by choosing a suitable representation scheme to code the possible solutions into individuals. Commonly, a solution is coded into a genotype consisting of a binary chromosome string with a predetermined number of bits which determines the problem's size. This coding creates a quantization of the solution space. A GA starts with an initial population of candidate solutions or individuals, and tries to modify them until the population converges to a solution. A problem-dependent fitness function must be chosen to measure the goodness of an individual. The modification of the population is done using an iterative application of three genetic operators that provide general exploratory heuristics which evolve the population from one generation to the next. These operators are selection, crossover, and mutation.

Selection decides which individuals shall survive into the next generation. The Simple Genetic Algorithm (SGA) uses fitness-proportionate selection which is implemented using tournament or roulette wheel selection. In this method, members of the population are selected with a probablity that is proportional to their fitness score. This selection mechanism encourages the survival of the fittest individuals to the next generation, and causes them to eventually overtake the population.

Mating between two individuals is implemented using crossover which generally allows the creation of better children or offspring by combining the genetic materials of two parents with a large crossover probability P_c.

These offspring will replace their parents in the next generation. Typically, one-point crossover is performed, where a single position is randomly selected along the parent's bit strings. This position is referred to as the crossing site. Then the chromosome strings of two children are formed by copying the bits located to the left of the crossing point from one parent's string and the bits located to the right of the crossing point from the other parent's string. Crossover is generally considered to be the primary search mechanism in GA's.

Mutation is a totally random step, where each bit in the chromosome string of the offspring individuals is allowed to change value with a small mutation probability P_m. This operator adds diversity to the population, thereby encouraging the exploration of new areas of the solution space to globalize the search for the optimal solution.

GAs distinguish themeselves from other optimizing tools because of their implicit parallelism, diversity and intensification. Parallelism and diversity are achieved by using a population of solutions instead of a single solution, which makes the GA one of the best global optimization tools. Intensification consists in preserving good solutions and combining their good features to produce better solutions through selection and crossover. This desirable feature makes the GA a more efficient searching tool compared to other merely exploratory and costlier global optimization methods which are based on exhaustive-like and random mutation search.

Genetic Optimization of Multimodal Functions

Traditional Genetic Algorithms have proved effective in exploring complicated fitness landscapes and converging populations of candidate solutions to a single global optimum. Hence, they offer a powerful optimization tool for unimodal domains or multimodal domains with a single global optimum. However, some optimization problems require the identification of global as well as local optima in a multimodal domain. As a result, several population diversity mechanisms have been proposed to delay or counteract the convergence of the population to as single solution by maintaining a diverse population of members throughout its search. As for the case of the SGA, these diversity enhancing methods have turned to nature for ideas and analogies. An analogy to multimodal domains exists in nature in the form of ``niches'' which are subspaces that can support different types of species or organisms. The fertility of each niche as well as the efficiency of each organism at exploiting niche fertility affects the number of organisms within each niche. This has inspired the consideration of each peak in a multimodal domain as a niche in the framework of what has come to be called niche formation methods. The two most popular niche formation methods are sharing and crowding.

Sharing methods [Hol75,GR87] attempt to maintain a diverse population with members distributed among all the niches corresponding to the peaks in a multimodal fitness landscape. They achieve this diversity by reducing the fitness of individuals that have highly similar members within the population. This in turn discourages redundent solutions from overtaking the entire population, while rewarding individuals that uniquely exploit areas of the domain. As a result, diversity is enhanced and popultion members are kept at local optima. The shared fitness of the i^{th} individual is defined as $f_{sh,i} = \frac{f_i}{m_i}$, where f_i is the raw fitness and m_i is the niche count, given by $m_i = \sum_{j=1}^{N_P} sh(d_{ij})$, where N_P is the number of individuals in the population, and d_{ij} is the distance between the i^{th} and j^{th} individuals. It is often recommended that d_{ij} be based on the individuals' phenotypes, i. e., it should be based on domain knowledge as opposed to being based solely on their genotype. The sharing function $sh(.)$ reaches a maximum of 1 at zero, decreases monotonically with distance, and falls to zero for distances that are greater than σ_{sh}. For example, the triangular sharing function is given by

$$sh(d_{ij}) = \begin{cases} 1 - \frac{d_{ij}}{\sigma_{sh}} & \text{if } d_{ij} < \sigma_{sh} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Hence, $sh(d_{ij})$ measures the amount of sharing or similarity between two individuals. The parameter σ_{sh} is vital to the performance of sharing methods, and ideally, it should approximate the peaks' widths. Deb and Goldberg [DG89] suggested ways for determining the appropriate value for σ_{sh} based on the expected number of peaks and the hypervolume of the entire domain space. Unfortunately, in many real applications the number of peaks may not be known. Moreover, one value of σ_{sh} may not be sufficient when peaks differ vastly in their widths. An inaccurate value of σ_{sh} can lead to erroneous shared fitness values which affects the diversity of the population. Even when σ_{sh} is accurately estimated, members of denser niches tend to have their fitnesses overly suppressed because of the nature of the sharing function. As a result, they are disadvantaged in competing with members of sparser niches. Sharing also encourages a diversification of the population that extends to the niche level when σ_{sh} is less than a peak's width. This within-niche diversification causes members of a niche to remain evenly dispersed throughout the niche instead of converging towards the peak. This means that the exploration of the peak areas is slowed and the identification of the peak of each niche will be hard. This problem exists even when σ_{sh} is correctly approximated because members of the same niche have different neighbors and hence different niche counts. Another problem that sharing methods have inherited from the SGA is the possible creation of mediocre or lethal offspring by mating two good individuals from different niches. This can cause a migration of good individuals from the niche areas to non-peak areas of the search space. In addition to all the above drawbacks, we should note that sharing adds an $\mathcal{O}(N_P^2)$ complexity because of all the pairwise distance calculations.

Recently, Miller and Shaw [MS95] proposed a ``dynamic sharing'' scheme which attempts to alleviate some of the drawbacks of standard sharing. In their scheme, the *dynamic* niche count is estimated more accurately leading to more accurate shared fitness values. This is accomplished by identifying the peaks in the population and assigning each member to the closest peak. However their method still assumes that the number of peaks is known and that all the peaks are a minimum distance of $2\sigma_{sh}$ apart, where the niche radius σ_{sh} is also assumed to be known and equal for all niches. In their scheme, they also used restricted mating to prevent crossover between members of different niches. In their mating restriction method, called dynamic inbreeding, the first parent is selected through tournament selection based upon shared fitness values. Then a second parent is determined by examining a pool of MF (Mating Factor) members selected randomly without replacement from the population, and selecting the fittest individual coming from the same niche as the first parent. If no such individual is found, then the parent is mated with the most similar individual in the mating pool, which can create lethal offspring. Dynamic inbreeding certainly offers an improvement over unrestricted mating. However, it relies on a correct identification of all the niche peaks which is based on two critical assumptions that require prior knowledge about the fitness landscape as discussed above.

Both standard and dynamic sharing methods have the disadvantage of giving preference to higher peaks because they use tournament selection based on fitness which can cause the extinction of relatively low peaks. One of the

early attempts at maintaining a diverse population without the use of fitness based tournament selection or shared fitnesses was made by De Jong [Jon75] who proposed Crowding methods which try to form and maintain niches by replacing population members preferrably with the most similar individuals. Unfortunately, stochastic ``replacement errors'' prevented this method from maintaining more than two peaks in a multimodal fitness landscape. Mahfoud [Mah92] proposed an improved crowding mechanism, called ``deterministic crowding'' (DC), which nearly eliminated replacement errors and proved more effective in maintaining multiple niches. DC is presented below:

height 0.02 true in width 5.0 true in

Deterministic Crowding (DC)

Repeat for G ***generations*** *{*

Repeat $\frac{N_P}{2}$ ***times*** *{*

Select two parents p_1 *and* p_2 *randomly without*

replacement;

Cross them to produce children c_1 *and* c_2 *;*

Optionally apply mutation to produce children c_1' *and* c_2' *;*

IF $[d(p_1, c_1') + d(p_2, c_2')] \leq [d(p_1, c_2') + d(p_2, c_1')]$ *THEN {*

IF $f(c_1') > f(p_1)$ *THEN replace* p_1 *with* c_1'

IF $f(c_2') > f(p_2)$ *THEN replace* p_2 *with* c_2'

}

ELSE{

IF $f(c_2') > f(p_1)$ *THEN replace* p_1 *with* c_2'

IF $f(c_1') > f(p_2)$ *THEN replace* p_2 *with* c_1'

}

}

}

height 0.02 true in width 5.0 true in

Unlike sharing methods, DC is free of any parameters relying on assumptions about the number of peaks or their widths. Also, unlike sharing, the expected distribution of the population at convergence is independent of fitness. Instead, the cardinality of each niche is expected to be proportional to the fraction of the population from the search space that falls within this niche, or in other words the prior probability of the niche. This is because the distribution of the converged population is expected to be similar to that of the initial population which is selected randomly. Hence, competition and improvement occur only within the niches leading to a diverse population with members that are closer to the actual peaks. Despite its considerable improvements, DC, like sharing methods, suffered from crossover interactions between different niches. This problem occurs when the fitness landscape contains dominated and dominant niches. A dominated niche can, with another niche's assistance, cross to form members from a fitter (dominant) niche. This causes a migration of members from the dominated peaks to the dominant peaks that will only come to a halt when one of the dominated niches is depleted of its members. Obviously, the effect of this migration on diversity can be pernicious because it eventually leads to the extinction of certain dominated peaks. Despite the crossover interaction problem inherent in all the discussed niching schemes, DC appears to be the most efficient scheme for multimodal domain optimization.

Acknowledgment

This work is supported by the National Science Foundation (CAREER Award IIS-0133948 to O. Nasraoui). Partial support of earlier stages of this work by the Office of Naval Research grant (N000014-96-1-0439) and by the National Science Foundation Grant IIS 9800899 is also gratefully acknowledged.

Bibliography

Box57

G. E. P. Box.
Evolutionary operation: a method of increasing industrial productivity.
Applied Statistics, 6:81-101, May 1957.

DG89

K. Deb and D. E. Goldberg.
An investigation of niche and species formation in genetic function optimization.
In *3rd Intl. Conf. Genetic Algsorithms*, pages 42-50, San Mateo, CA, 1989.

FOW66

L. J. Fogel, A. J. Owens, and M. J. Walsh.
Artificial Intelligence Through Simulated evolution.
Wiley Publishing, New York, 1966.

Fra57

A. S. Fraser.
Simulation of genetic systems by automatic digital computers.
Australian Journal of Biological Science, 10:484-491, May 1957.

Gol89
D. E. Goldberg.
Genetic algorithms in search, optimization, and machine learning.
Addison-Wesley, New York, 1989.

GR87
D. E. Goldberg and J. J. Richardson.
Genetic algorithms with sharing for multimodal function optimization.
In *2nd Intl. Conf. Genetic Algsorithms*, pages 41-49, Cambridge, MA, Jul. 1987.

Hol75
J. H. Holland.
Adaptation in natural and artificial systems.
MIT Press, 1975.

Jon75
K. A. De Jong.
An analysis of the behavior of a class of genetic adaptive systems.
Doct. Diss., U. of Michigan., 36(10-5140B):29-60, 1975.

Mah92
S. W. Mahfoud.
Crowding and preselection revisited.
In *2nd Conf. Parallel problem Solving from Nature, PPSN '92*, Brussels, Belgium, Sep. 1992.

MS95
B. L. Miller and M. J. Shaw.
Genetic algorithms with dynamic niche sharing for multimodal function optimization.
Technical Report 95010, IlliGAL, Dept. of general engineering, University of Illinois at Urbana-Champaign, Dec. 1995.

Rec73
I. Rechenberg.
Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution.
Frommann-Holzboog, Stuttgart, 1973.

SJB$^+$93
W. M. Spears, K. A. De Jong, T. Back, D. B. Fogel, and H. De Garis.
An overview of evolutionary computation.
In *1993 European Conference on Machine Learning*, 1993.

About this document ...

A Brief Overview of Genetic Optimization

This document was generated using the **LaTeX2HTML** translator Version 2002 (1.62)

The command line arguments were:
latex2html `-split 0 -image_type gif GeneticAlgorithms.tex`