Genetic Algorithms Implementation

- Advanced Topics
  - Niching Genetic Algorithms
  - Multi-Objective Optimization
- Applications
Advanced Topics

Niching Genetic Algorithms

• Motivation
• The Idea
• Ecological Meaning
• Niching Techniques
Niching Genetic Algorithms

- Multimodal Optimization

- Traditional genetic algorithms with elitist selection are suitable to locate the optimum of unimodal functions as they converge to a single solution of the search space.

- Real problem, however, often require the identification of optima along with some local optima.

- For this purpose, niching methods extend the simple genetic algorithms by promoting the formation of subpopulations in the neighborhood of the local optimal solutions.
Canonical GA

 Niching GA

Canonical GA

 Niching GA
Niching Genetic Algorithms: The Idea

• Niching methods have been developed to reduce the effect of *genetic drift* resulting from the selection operator in the simple genetic algorithms.

• They maintain population diversity and permit genetic algorithms to explore more search space so as to identify multiple peaks, whether optimal or otherwise.

• The *fitness sharing method* is probably the best known and best used among the niching techniques.
Ecological Meaning (1)

- In natural ecosystem, a niche can be viewed as an organisms task, which permits species to survive in their environment.
- Species are defined as a collection of similar organisms with similar features.
- The subdivision of environment on the basis of an organisms role reduces inter-species competition for environmental resources.
- This reduction in competition helps stable sub-populations to form around different niches in the environment.
Analogy

• By analogy, in multimodal GAs, a niche is commonly referred to as the location of each optimum in the search space and the fitness representing the resource of that niche.
Genetic Niching: Niching Techniques

• The Fitness Sharing Method
• Crowding Method
• Clearing Method
Genetic Niching: The Sharing Method


The Fitness Sharing Method: Essence

- The sharing method essentially modifies the search landscape by reducing the payoff in densely populated regions.

- This method rewards individuals that uniquely exploit areas of the domain, while discouraging highly similar individuals in a domain.

- This causes population diversity pressure, which helps maintain population members at local optima.
Genetic Niching: The Crowding Method

- Standard Crowding Method (DeJong, 1975):
- Deterministic Crowding Method (Mahfoud, 1995):
Genetic Niching: Deterministic Crowding Method

• Mahfoud (1995) improved DeJong's standard crowding and proposed his deterministic crowding method.

• Deterministic crowding method introduces competition between children and parents of identical niches.

• This method tends to identify solutions with higher fitness and lose solutions with lower fitness.
Genetic Niching: Clearing Method

• Clearing method was proposed by Petrowski (1996, 1997) based on limited resources of environment.
• Instead of evenly sharing the available resources among the individuals of a subpopulation, the clearing procedure supplies these resources only to the best individuals of each subpopulation.
• Clearing method is found to be most suitable to deal with the extremely complicated search space of the portfolio optimization problem.
Clearing Method: Technical Details

- Notations
- The Three Functions
- Algorithm
Clearing Method: Some Notations

Suppose that \( P \) and \( n \) denote the whole population and the population size, respectively. "Sigma" is the clearing radius and "Kappa" is the capacity of each niche. "nbWinners" indicates the number of winners of the subpopulation associated with the current niche. Population \( P \) is an array of \( n \) individuals. The clearing procedure uses three functions, which are:
Clearing Method: The Three Functions

- The clearing procedure uses three functions:
  - $\text{SortFitness}(P)$ sorts the population $P$ according to the fitness of the individuals by decreasing order.
  - $\text{Fitness}(P[i])$ returns the fitness of the $i$-th individual of population $P$.
  - $\text{Distance}(P[i], P[j])$ returns the distance between two individuals $i$ and $j$ of population $P$. This distance indicates how different individuals $i$ and $j$ are from each other, the larger it is, the more different the two corresponding individuals are.
Clearing Method: The Algorithm

function clearing (Sigma, Kappa)

begin

SortFitness (P)
for i = 0 to n-1
    if Fitness (P[i]) > 0
        nbWinners := 1
        for j = i+1 to n-1
            if Fitness (P[j]) > 0 and Distance (P[i], P[j]) < Sigma
                if nbWinners < Kappa
                    nbWinners := nbWinners + 1
                else
                    Fitness(P[j]) := 0
                endif
            endif
        endfor
    endif
endfor

end
Summary of Niching Genetic Algorithms

• In order to locate multiple solutions, Aichholzer et al. developed a multi-population ES, where proposed the number of subpopulations, specifies the number of generations that each individual is allowed to survive.

• Given clusters, recombination proceeds as follows: One of the clusters is randomly selected, and roulette wheel selection is used to select one of the individuals from this cluster as one of the parents.

• Fitness sharing is one of the earliest GA niching techniques, originally introduced as a population diversity maintenance technique.
Advanced Topics

Multi-Objective Optimization

• Overview
• Example
• Pareto-Optimal Solutions
Multi-Objective Optimization

• Multi-objective optimization (MOO) is the optimization of conflicting objectives.

• We were actually doing MOO in the Introduction chapter when we optimized the insulation thickness – we balanced two opposing objectives.

• We were able to use one objective function by linking the two objectives with a common feature – cost.

• In some optimization problems, it is not possible to find a way to link competing objectives.

• Sometimes the differences are qualitative and the relative importance of these objectives can’t be numerically quantified.
Conceptual Example

• Suppose you need to fly on a long trip: Should you choose the cheapest ticket (more connections) or shortest flying time (more expensive)?

• It is impossible to put a value on time, so these two objectives can’t be linked.

• Also, the relative importance will vary.
  – There may be a business emergency you need to go fix quickly.
  – Or, maybe you are on a very tight budget.
Pareto-Optimal Solutions

• A MOO problem with constraints will have many solutions in the feasible region.
• Even though we may not be able to assign numerical relative importance to the multiple objectives, we can still classify some possible solutions as better than others.
• We will see this in the following example.
**Pareto-Optimal Solutions Example**

- Suppose in our airplane-trip example from earlier, we find the following tickets:

<table>
<thead>
<tr>
<th>Ticket</th>
<th>Travel Time (hrs)</th>
<th>Ticket Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>1700</td>
</tr>
<tr>
<td>B</td>
<td>9</td>
<td>2000</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>1800</td>
</tr>
<tr>
<td>D</td>
<td>7.5</td>
<td>2300</td>
</tr>
<tr>
<td>E</td>
<td>6</td>
<td>2200</td>
</tr>
</tbody>
</table>
Comparison of Solutions

- If we compare tickets A & B, we can’t say that either is superior without knowing the relative importance of Travel Time vs. Price.

- However, comparing tickets B & C shows that C is better than B in both objectives, so we can say that C “dominates” B.

- So, as long as C is a feasible option, there is no reason we would choose B.
Comparison of Solutions

<table>
<thead>
<tr>
<th>Ticket</th>
<th>Travel Time (hrs)</th>
<th>Ticket Price ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>1700</td>
</tr>
<tr>
<td>B</td>
<td>9</td>
<td>2000</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>1800</td>
</tr>
<tr>
<td>D</td>
<td>7.5</td>
<td>2300</td>
</tr>
<tr>
<td>E</td>
<td>6</td>
<td>2200</td>
</tr>
</tbody>
</table>

• If we finish the comparisons, we also see that D is dominated by E.

• The rest of the options (A, C, & E) have a trade-off associated with Time vs. Price, so none is clearly superior to the others.

• We call this the “non-dominated” set of solutions become none of the solutions are dominated.
Graph of Solutions

Usually, solutions of this type form a typical shape, shown in the chart below:
Types of Solutions

- Solutions that lie along the line are non-dominated solutions while those that lie inside the line are dominated because there is always another solution on the line that has at least one objective that is better.
Pareto-optimal Solutions

• The line is called the **Pareto front** and solutions on it are called Pareto-optimal.
• All Pareto-optimal solutions are non-dominated.
• Thus, it is important in MOO to find the solutions as close as possible to the Pareto front & as far along it as possible.
Graphical Example

For the following feasible region with objectives $f_1$ & $f_2$ where both $f_1$ & $f_2$ are minimized:

![Diagram of feasible region and Pareto front]
Finding the Pareto Front

If this is done for a 90° span of lines, all the points on the Pareto front will be found.
Practicality of this Procedure

- Actually, this is not the procedure that is used in practice, but it is a good illustration of the concept.
- This procedure would require finding all possible points in the feasible region and then using many combinations of weights.
- For more than two objectives, the complexities and the number of combinations make this impractical.
Realistic Procedures

• There are different methods used in practice, but one is to use a genetic algorithm to enumerate points along the Pareto front over several iterations, then use some method to rank the quality of the trade-offs based on the particular application being modeled.
Optimization

• It should be remembered that each point on the Pareto front is found by solving an optimization problem.

• multi-objective optimization methods: for problems with more than one objective to optimize.
Advanced Topics

Summary of multi-objective optimization

- Many real-world problems require the simultaneous optimization of a number of objective functions. Some of these objectives may be in conflict with one another.
- The set of solutions is referred to as the non-dominated set, or the Pareto-optimal set.

\[
\begin{align*}
\text{Minimize} & \quad f(x) \\
\text{Subject to} & \quad g_m(x) \leq 0, \quad m = 1, \ldots, n_g \\
& \quad h_m(x) = 0, \quad m = n_g + 1, \ldots, n_g + n_h \\
& \quad x \in [x_{min}, x_{max}]^{n_x}
\end{align*}
\]
Applications

• Homogeneous vs. heterogeneous representation
• Online adaptation vs. offline adaptation
• Static adaptation versus adaptive adaptation
• Implementation
Homogeneous vs. Heterogeneous Representation

- Homogeneous representation
  Used traditionally
  Simple; can use existing EC operators
  Binary is traditional coding for GAs; it’s simple and general
  Use integer representation for discrete valued parameters
  Use real values to represent real valued parameters if possible

- Heterogeneous representation
  Most natural way to represent problem
  Real values represent real parameters, integers or binary strings
    represent discrete parameters
  Complexity of evolutionary operators increases
  Representation-specific operators needed
Binary Representations

• Advantages
  Simple and popular
  Use standard operators

• Disadvantages
  Can result in long chromosomes
  Can introduce inaccuracies
Final Thoughts on Representation

• The best representation is usually problem-dependent.

• Representation is often a major part of solving a problem.

• In general, represent a problem the way it appears in the system implementation.
Population Adaptation Versus Individual Adaptation

• Individual: Most commonly used. Pittsburgh approach; each chromosome represents the entire problem. Performance of each candidate solution is proportional to the fitness of its representation.

• Population: Used when system can’t be evaluated offline. Michigan approach: entire population represents one solution. (Only one system evaluated each generation.) Cooperation and competition among all components of the system.
Static Adaptation Versus Dynamic Adaptation

• Static: Most commonly used. Algorithms have fixed (or pre-determined) values.

• Adaptive: Can be done at
  - Environment level
  - Population level (most common, if done)
  - Individual level
  - Component level
Genetic Algorithm Implementation

• Essentially a canonical GA that utilizes crossover and mutation

• Uses binary representation

• Searches for optima with real value parameters

• Several benchmark functions are included
Data Types

Enumeration data type used for selection types, crossover types, and to select the test function.
GA Selection Mechanisms

• All use elitism

• Proportional selection – roulette wheel that uses fitness shifting and keeps fitnesses positive

• Binary tournament selection – better of two randomly-selected individuals

• Ranking selection – evenly-spaced fitness values; then like roulette wheel
Mutate According to Bit Position Flag

When 0, bit-by-bit consideration
When 1, mutation done that is \textit{approximation} of Gaussian

Probability of mutation $m_b$ varies with bit position:

$$m_b = m_0 \frac{1}{\sqrt{2\pi}} e^{-\frac{b}{2}}$$

where $b=0$ for the least significant bit, 1 for the next, etc. and $m_0$ is the value in the run file.

Bit position is calculated for each variable. The mutation rate for the first bit is thus about .4 times the value in the run file.

(This mutation is similar to that carried out in EP and ES (Gaussian).)
Crossover Flag

0: One-point crossover

1: Uniform crossover

2: Two-point crossover