Optimization of Digital Modulation Schemes using Evolutionary Algorithms

Derek Kozel
Introduction

• Derek Kozel
  – Masters in ECE from Carnegie Mellon
  – SpaceX, Range Networks, currently Ettus Research
  – Past research includes SDR and SDN based cellular systems, low power system on chip architecture, and evolutionary algorithms
Recognition and Thanks

- Original work done during my masters
- “Evolution of digital modulation schemes for radio systems” at GECCO 2014
- Co-Authors
  - Ervin Teng, Bob Iannucci, and Jason Lohn
What is Evolutionary Computing?

• Subfield of artificial intelligence
• Continuous and combinatorial optimization
• Stochastic, optimal result not guaranteed
• Good for large, complex, search spaces
Basic Flow

- Initialization
- Population
- Parent selection
- Parents
- Mutation
- Recombination
- Offspring
- Survivor Selection
- Termination
BEGIN

INITIALIZE population with random candidate solutions
EVALUATE each candidate
REPEAT UNTIL (TERMINATION CONDITION is satisfied)
  1 SELECT parents
  2 RECOMBINE pairs of parents
  3 MUTATE the resulting offspring
  4 EVALUATE new candidates
  5 SELECT individuals for the next generation
Components

- Representation
- Fitness Function
- Population
- Parent selection mechanism
- Variation operators
  - Mutation, Recombination, others
- Survivor selection mechanism
Representation

• Genotype
  – Physical representation
    • i.e. bit strings

• Phenotype
  – Logical meaning
    • i.e. ints, floats, coordinates on a chess board
Fitness Functions

• Defining how good a solution is
  – AKA Objective Function in optimization

• Can be a simple formula or a statistic from a complex simulation

Example 2D fitness landscape
Population

• A pool of individual potential solutions (inputs)
  – Collection of phenotypes
• Can be divided up in different ways
  – Island
  – Age
  – Fitness
• Total of $\mu$ individuals
Parent Selection

- Parenthood not guaranteed
- Selection can be random, fitness based, deterministic, etc
- \( \lambda \) offspring are created for each generation
Fitness Proportional Selection

• Individuals are chosen to be parents with a probability proportional to their absolute fitness

\[ p(\text{individual } i \text{ chosen}) = \frac{f_i}{\sum_{j=1}^{\mu} f_j} \]

• This means that individuals with high fitness dominate the population quickly
  – Premature convergence
Ranking Selection

- Individuals chosen as parents with a probability proportional to their relative fitnesses

\[ P_{\text{lin-rank}}(i) = \frac{(2 - s)}{\mu} + \frac{2i(s - 1)}{\mu(\mu - 1)} \]

<table>
<thead>
<tr>
<th>Fitness</th>
<th>Rank</th>
<th>(P_{\text{sel}FP})</th>
<th>(P_{\text{sel}LR}) (s=2)</th>
<th>(P_{\text{sel}LR}) (s=1.5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>0.1</td>
<td>0</td>
<td>0.167</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>0.5</td>
<td>0.67</td>
<td>0.5</td>
</tr>
<tr>
<td>C</td>
<td>4</td>
<td>0.4</td>
<td>0.33</td>
<td>0.33</td>
</tr>
<tr>
<td>Sum</td>
<td>10</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Variation Operators

• Used to create children different but related to the parents

• Operators apply to a certain number of parents (Arity)
  – Mutation is done to one individual at a time
  – Recombination takes two or more parents to make an child
Mutation

• Changes a single individual

• Bit flipping for bitstrings
  – For each bit, flip with a probability $p_m$

• For numbers
  – For each value, add a random value with probability $p_m$
  – Reset to a random value with probability $p_m$
Mutation continued

- Swapping

1 2 3 4 5 6 7 8

- Scramble, Insert, Inversion

1 2 8 4 5 6 7 3
Recombination

- Crossover is the main operation
  - One-point
    - Uniform (Pick a specific gene with probability $p_c$)
Survivor Selection

• Population is now $\mu$ parents and $\lambda$ children
• Usually population remains constant over generations
• Must choose $\mu$ individuals from $\mu + \lambda$
  – Age based
  – Fitness based
• Sometimes desirable to preserve elites
  – Best $n$ individuals guaranteed survival
Initialization

- Usually completely random generation
- Can incorporate domain specific knowledge
  - Choose known good answers
  - Introduces some bias to the search
Termination Condition

- Prevent infinite or exhaustive searching
- N generations
- Absolute fitness threshold
- Minimum amount of change in fitness
So it begins, now what?

• Exploration
• Exploitation
• Premature Convergence
  – Local Optimum
PSK Constellations

- A mapping of input bits to physically transmittable signals
- Can be represented as a string of integers

<table>
<thead>
<tr>
<th>Input Bits</th>
<th>I</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>1.414</td>
<td>0</td>
</tr>
<tr>
<td>001</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
<td>1.414</td>
</tr>
<tr>
<td>011</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Setup

- Python implementations of algorithms
- Call GNU Radio simulation for fitness testing
Basic Evolutionary Strategy

• Create 20 random constellations (individuals)
• Mutate and combine to create 140 constellations (children)
• Evaluate their fitness (BER + fudge factors)
• Select 20 survivors
Parameters

• 150 Generations
• Parent pool of 20 individuals
• Child pool of 140
  – 100 the product of mutation
  – 40 from recombination
Basic ES Results

• Best Fitness: 0.6422
One Run
Island Model

• Galapagos Islands
  – Population is divided with occasional migrations
• Run multiple populations in parallel
• Transfer small number of individuals when diversity converges on an island
• Use a fitness based survivor selection
• Mean best fitness = 0.6369
Age Layered Population Structure

- Gregory Hornby (UC Santa Cruz)
- Divide population by age cohorts
- Move population onwards and re
- Restart lowest level each epoch
ALPS Results

- Best fitness: 0.6140
Future Extensions

• Additional fitness tests
  – Variety of SNR and channel condition tests
• Longer tests, more bits, over the air
• Add additional transmission parameters to individuals
• Improve simulation
  – Framing, sync, complex channels
Thanks

https://github.com/dkozel/Evolving-Constellations

derek@derekkozel.com
www.derekkozel.com/talks
@derekkozel