CSE/ECE 848 Introduction to Evolutionary Computation

Module 2, Lecture 6, Part 2b

Principles of Evolutionary Computation—Generalizing the GA

Erik Goodman
Professor, ECE, ME, and CSE
Michigan State University

Generalizing the SGA– First Step toward More General Evolutionary Algorithms

- SGA -- Holland (60's, book in '75) -- binary chromosome, population, proportional selection, 1- or 2-pt crossover (recombination), low rate of mutation
- More general GA & Evolutionary Computation, typically:
  - Population of solutions—still a must
  - Many non-binary reps (ints, doubles, chars, trees, graphs, etc.)
  - Many forms of selection
  - Many types of recombination, different numbers of parents
  - Many types of mutation (real, correlated, adaptive, etc.)
  - May be hybridized with LOTS of other stuff

Representation Terminology across EC

- Individual (solution) called a chromosome
- What's on the chromosome is GENOTYPE
- What it *means* in the problem context is the PHENOTYPE: genotypes *MAP* to phenotypes
- Genotype may be function coefficients, determine order of execution, be inputs to a simulator, execute operations, etc.). Ints may map to reals, etc.
- What most directly *determines the fitness* is what we call the phenotype.
- Genotype determines phenotype, but phenotype may *look* very different

Representation Meets Mutation! Oops, Careful!

- **If problem is binary decisions, bit-flip mutation is fine**
- BUT if using binary numbers to encode integers, as in [0,15] <--> [0000, 1111], there is the problem of ***Hamming cliffs***:
  - One mutation can change 6 to 7: 0110 → 0111, BUT
  - Need 4 bit-flips to change 7 to 8: 0111 → 1000
  - That's called a "Hamming cliff"
- May use Gray (or other distance-one) codes to improve properties of operators: for example: 000, 001, 011, 010, 110, 111, 101, 100.
- But still, flip 1st bit of 000 --> 100, makes 0 into 8, so not ideal
- SO, may instead use ints and, for example, perform discretized Gaussian mutation, now small changes produce local effects
- Instead, today, MANY forms of EC use REAL number representations, not binary or ints, and some *started* that way (evolutionary strategies, differential evolution, etc.)

Recombination or Crossover

On "parameter encoded" representations, GA's often use:

- 1-pt
- 2-pt (circular)
- uniform crossover (but… CAREFUL, loses linkage!)

Different problem types require different recombination operators:

- for example, solving a Traveling Salesman problem:
  - chromosome might be order of visiting numbered cities
  - would want crossover to produce LEGAL chromosomes (each city appears ONCE)
  - none of the above crossovers would do that!

Useful Concepts When Choosing Representations & Recombination Operators

Linkage – interacting loci nearby on chromosome, not usually disrupted by a given crossover operator (cf. 1-pt, 2-pt, uniform re linkage…)

Epistasis – non-additive effect of non-adjacent loci on fitness (CAN be disrupted by crossover)

- In general, good if recombination tends to preserve linkages and even more distant epistatic connections
- That can help us in choosing/creating good representations and crossover operators
- “Works” is not as good as “works well” for a problem

Defining Objectives or Fitness Functions

- Problem-specific, of course
- Many involve using a simulator or other “big, slow” software
- EC doesn’t need to know (or even HAVE) derivatives
- Fitness may be stochastic—either parameters or values assigned
- Often need to evaluate thousands of times, so can’t be TOO COSTLY
- For real-world, fitness evaluation time is typical bottleneck
- Example: simple fitness criterion, but complex to calculate:
  - Minimize the weight of fuel a plane should carry for a flight from Chicago to New York, so has less than 1/1,000,000 chance of running out of fuel