

# **CSE/ECE 848**

## **Introduction to**

# **Evolutionary Computation**

**Module 2, Lecture 6, Part 1**

## **The Simple Genetic Algorithm**

**Erik Goodman, Executive Director**

**BEACON Center for the Study of Evolution in Action**

**Professor, ECE, ME, and CSE**

# Genetic Algorithms

- Invented in mid-late 1960's by John H. Holland, University of Michigan, but not named GAs until about 1975.
- Are a method of search, often applied to optimization or machine learning
- Are stochastic – but are *not* random search
- GA is one of earliest types of evolutionary computation-- uses an evolutionary analogy, “survival of fittest”
- Not *fast* in some sense; but sometimes more robust; scale relatively well, so can be useful
- Have extensions including Genetic Programming (GP) (LISP-like function trees), learning classifier systems (evolving rules), linear GP (evolving “ordinary” programs), many others

# The Canonical or Classical GA – “The Simple Genetic Algorithm”

- Maintains a set or “population” of strings at each stage
- Each string is called a chromosome, and encodes a “candidate solution” – in the SGA, encodes as a *binary string* (and later, in almost any conceivable representation)

# Criterion for Search

- Goodness (“fitness”) or optimality of a string’s solution determines its FUTURE influence on search process -- survival of the fittest
- Good solutions are used to generate other, similar solutions which may also be good (even better)
- The POPULATION at any time stores ALL we have learned about the solution, at any point
- Robustness (efficiency in finding good solutions in difficult searches) is key to GA success

# Simple GA:

## The Representation

1011101010 – a possible 10-bit string (“CHROMOSOME”) representing a possible solution to a problem

Bits or subsets of bits might represent choice of some feature, for example. Let's represent choice of shipping container for some object:

<u>bit position</u>	<u>meaning</u>
1-2	steel, aluminum, wood or cardboard
3-5	thickness (1mm-8mm)
6-7	fastening (tape, glue, rope, plastic wrap)
8	stuffing (paper or plastic “peanuts”)
9	corner reinforcement (yes, no)
10	handles (yes, no)

# Terminology Refresher— evolutionary biology provides hints

Each position (or each *set of positions* that encodes some feature) is called a LOCUS (plural LOCI)

Each possible value at a locus is called an ALLELE

For THIS problem, we need a simulator, or evaluator program, that can tell us the likelihood of damage of shipping a given object in any particular type of container—we want to MINIMIZE that objective

- ◆ Objective may be a function of COST (including losses from damage) (for example, maybe 1.4 means very low cost, 8.3 is very bad on a scale of 0-10.0), which we try to minimize, or
- ◆ may be a FITNESS, or a number that is larger if the result is BETTER

# How Does an SGA Operate?

- ◆ For ANY chromosome, must be able to determine a FITNESS (measure performance toward an objective)
- ◆ Fitness is always something we seek to MAXIMIZE
- ◆ An “objective” may be maximized or minimized; but *fitness* is to be maximized, and if objective is to be minimized, define fitness from it as something to maximize (- objective, or  $1/\text{objective}$ , etc.)

# GA Operators:

## Classical Mutation

- Operates on ONE “parent” chromosome
- Produces an “offspring” with changes
- Classically, toggles each bit of a binary representation with some small probability
- So, for example: 1101000110 could mutate to:  
1111000110
- Each bit has same probability of mutating
- (Easy to make this calculation more efficient than calling a random variable for each bit position)



# Classical Crossover

- ◆ Operates on two parent chromosomes
- ◆ Produces one or two children or offspring
- ◆ Classical crossover occurs **at 1 or 2 points**:

- ◆ 1-point crossover

- ◆
 

1111111111
X <u>0000000000</u>

# Classical Crossover

- ◆ Operates on two parent chromosomes
- ◆ Produces one or two children or offspring
- ◆ Classical crossover occurs at 1 or 2 points:

- ◆ 1-point crossover

◆

	1111111111
X	<u>0000000000</u>
	1110000000
and	0001111111

# Classical Crossover

- ◆ Operates on two parent chromosomes
- ◆ Produces one or two children or offspring
- ◆ Classical crossover occurs at 1 or 2 points:
  - ◆ 2-point crossover

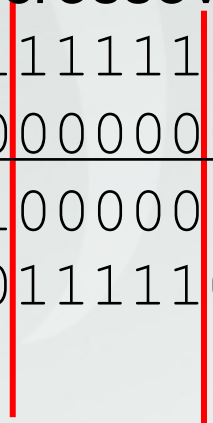
	1	1	1	1	1	1	1	1	1
X	0	0	0	0	0	0	0	0	0

Diagram illustrating 2-point crossover. Two parent chromosomes are shown: one with 10 ones (1111111111) and one with 10 zeros (0000000000). Two vertical red lines indicate the crossover points between the 3rd and 4th bits, and between the 7th and 8th bits. A horizontal line is drawn under the zero chromosome.

# Classical Crossover

- ◆ Operates on two parent chromosomes
- ◆ Produces one or two children or offspring
- ◆ Classical crossover occurs at 1 or 2 points:
  - ◆ 2-point crossover

	1111111111
X	<u>0000000000</u>
	1110000011
and	0001111100



## Selection Operation— *Fitness Proportional Selection*

- *Traditionally* (SGA), parents are chosen to mate with probability proportional to their fitness: *proportional selection*
- Traditionally (SGA), children replace their parents (what works like this in biology?)
- Many other variations now more commonly used (we'll come back to this)
- Overall principle: survival of the fittest

# Synergy – the KEY

Clearly, selection alone is no good ...

Clearly, mutation alone is no good ...

Clearly, crossover alone is no good ...

Fortunately, using all three simultaneously is  
sometimes spectacular!