# Evolutionary Computation
## – Genetic Algorithms (GA)

Instructor: Dr. Muhammad Fahim

INNOPOLIS UNIVERSITY

# Contents

- What is evolution?
- Evolutionary Algorithm – GA
- Technical Terms
- Steps in Genetic Algorithms
- Running Example
- Drawbacks of GA
- When to use GA and NOT to use GA
- Application Areas

# What is Evolution?

- There is a population of creatures
- Creatures reproduce
  - Children are like parents, but different.
- Creatures die
  - Some animals are more likely to survive and reproduce.
  - They are considered "more fit"
  - **For example:** Dinosaurs are dead, cockroaches still around.

**Survival of the fittest**

# Paradigms of Evolutionary Algorithms

- **Genetic Algorithms (GA)**
  - This is the most popular paradigm of evolutionary computation which model genetic evolution.
  - In GA individuals were represented by binary strings or real valued strings.

- **Genetic Programming (GP)**
  - It is considered as a variation of the genetic algorithms.
  - Individuals are various kind of programs consisting data structures and functions represented by trees.

# Paradigms of Evolutionary Algorithms

- **Evolutionary Programming (EP)**
  - The paradigm is derived from the simulation of adaptive behavior in evolution and developed to evolve finite-state machines.
  - It usually use mutation operation and self-adaptation parameters of mutation.

- **Evolution Strategies (ES)**
  - In evolution strategies, each individual is represented by genetic building blocks and a set of strategy parameters that models the behavior of that individual in its environment.
  - It applies both reproduction operators but main focus on mutation and mutations are accepted only in successive cases.

# Paradigms of Evolutionary Algorithms

- **Differential Evolution**
  - It is similar to genetic algorithms, differing in the reproduction mechanism used.
  - It differs significantly in the sense that distance and direction information from the current population is used to guide the search process.
  - Differential Evolution operates better on fitness surfaces which are flat.

- **Cultural Evolution (CE)**
  - It is based on the principles of human social evolution.
  - The performance of the evolutionary algorithm can be improved if domain knowledge is used to bias the search space.
  - Domain knowledge serves as a mechanism to reduce the search space by pruning undesirable parts of the solution space, and by promoting desirable parts.

# Paradigms of Evolutionary Algorithms

- **Coevolution**
  - It is known as the competitive evolution to save an individuals.
  - In this process, win of one spices means the die of others.
  - In the next generation, each species changes in response to the actions of the other species during the previous generation.
  - Furthermore, it has two types of coevolution that is
    - Competitive and
    - Cooperative.

# Genetic Algorithms

# Evolutionary Algorithms - GA

- A biologically inspired model of intelligence and the principles of biological evolution are applied to find solutions to difficult problems.
- Uses concepts of "Natural Selection" and "Genetic Inheritance" (Darwin 1859).
- Based on the survival of the fittest among string creatures with a structured yet randomized exchange to form search algorithm.
- Formally introduced in the 1970's by John Henry Holland at University of Michigan (his PhD thesis).

# Genetic Algorithm – Applications

- Provide efficient, effective techniques for optimization and machine learning applications
- Widely-used today in business, scientific and engineering circles
- More detail about the applications at the end of the lecture

# Technical Terms

- Chromosome - string of genes
- Gene - a portion of a chromosome representing a parameter of the solution set
- Locus - a position on a chromosome
- Alleles - different values of a particular gene. Members of the domain of the gene's value
- Genotype - coding of the solution
- Phenotype - expression of the genotype
- Population - group of individuals capable of interbreeding
- Stochastic operators – Mutation and Crossover

# Genetic Algorithm: Definition

A genetic algorithm maintains a population of candidate solutions for the problem at hand, and makes it evolve by iteratively applying a set of stochastic operators.

# Simple Genetic Algorithm – a.k.a SGA

1. Select an encoding schema
2. Randomly initialize chromosome pool
3. Evaluate each individual fitness in the population
4. Select fit individuals to breed new population
5. Create new population from parents
6. Replace old population with new population
7. Repeat steps 3 - 6 for each generation

# 1. Select an encoding schema

- This can be thought of as the genetic code of a candidate – thus the term “genetic algorithm”!
  - The typical candidate representation is a binary string.
  - Other representations are possible, but they make crossover and mutation harder.

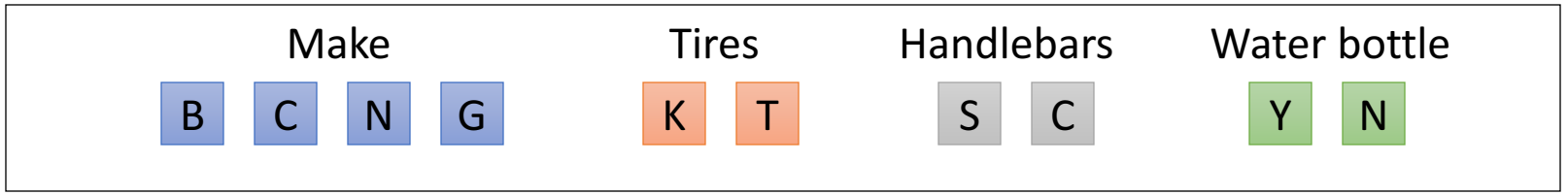
- We want to encode candidates in a way that makes mutation and crossover easy.

# 1. Select an encoding schema – Example

- Let's say we want to represent a rule for classifying bikes as mountain bikes or hybrid, based on these attributes:
  - **Make**
    - Bridgestone, Cannondale, Nishiki, or Gary Fisher
  - **Tire type**
    - Knobby, treads
  - **Handlebar type**
    - Straight, curved
  - **Water bottle holder**
    - *Boolean*

# 1. Select an encoding schema – Example

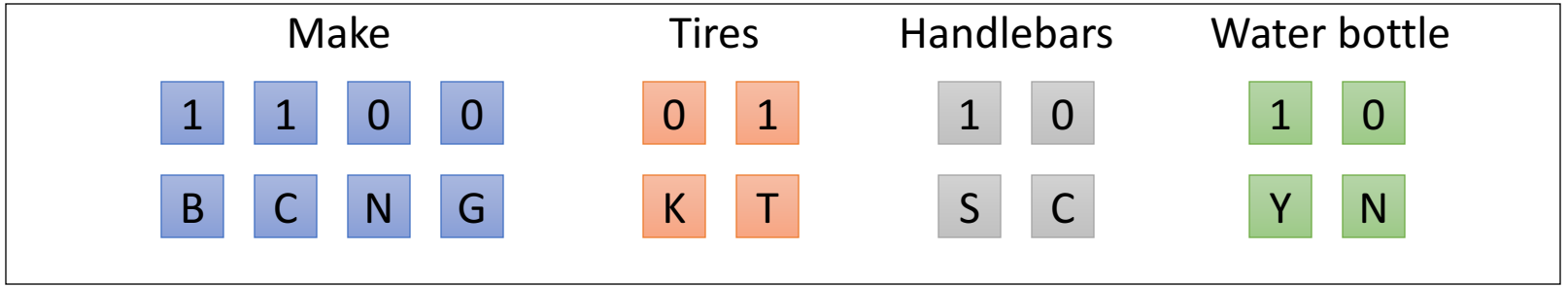
- Make (Bridgestone, Cannondale, Nishiki, or Gary Fisher)
- Tire type (knobby, treads)
- Handlebar type (straight, curved)
- Water bottle holder (*Boolean*)



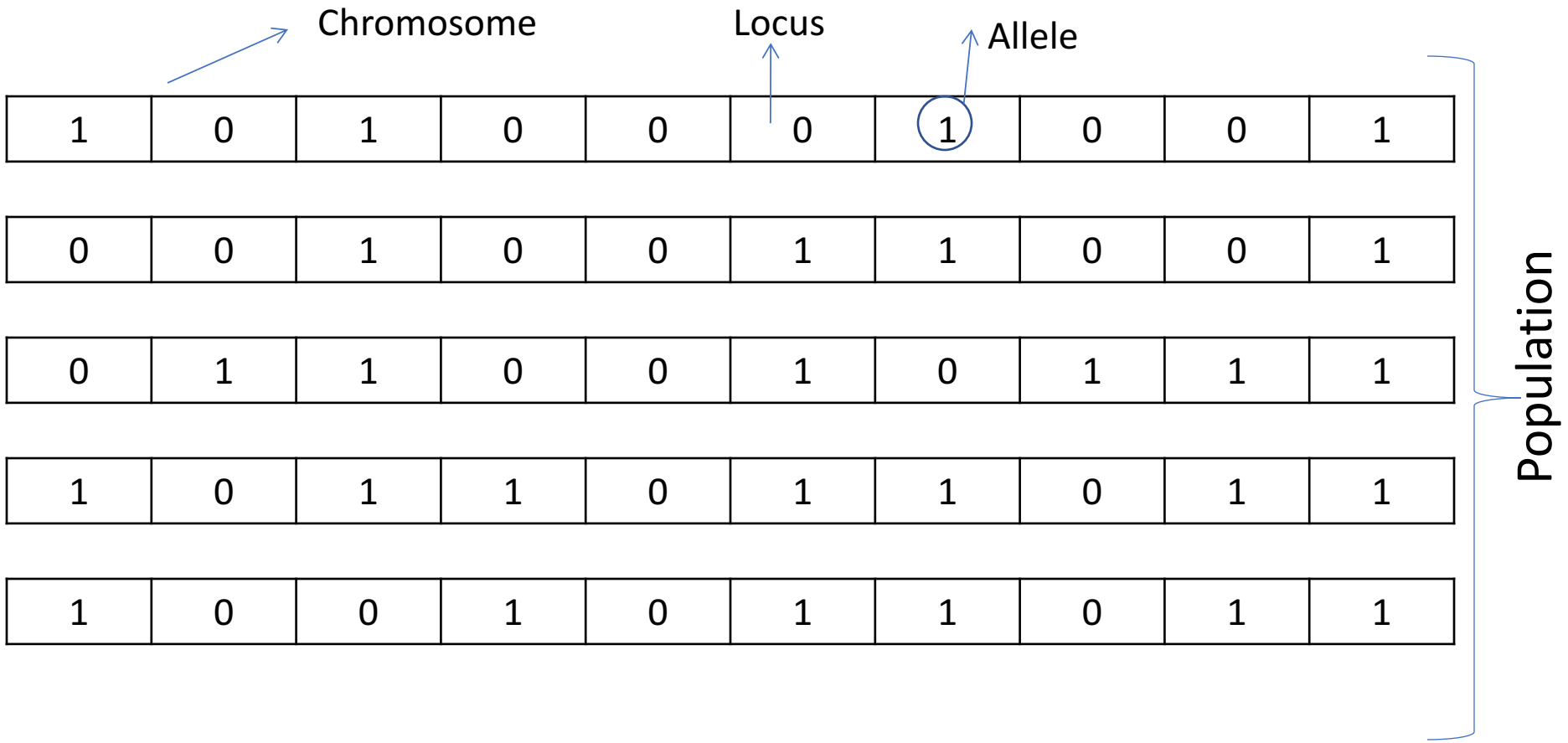
# 1. Select an encoding schema – Example

- Let's say we want a rule that will match any bike that is made by Bridgestone or Cannondale, has treaded tires, has straight handlebars and water bottle.

- This rule could be represented as 1100011010



# 2. Randomly initialize chromosome pool

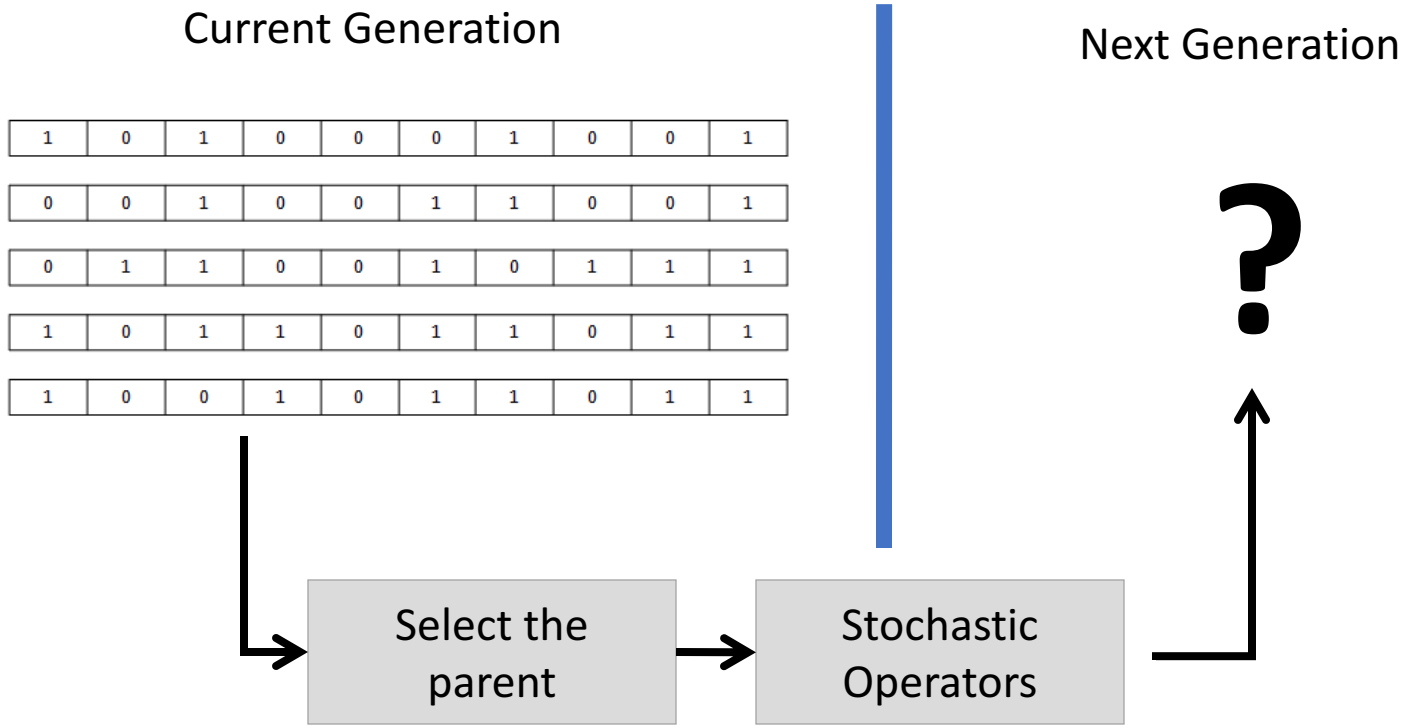


# 3. Evaluate each individual fitness

- Method dependent upon problem
- Involves quantifying performance of the phenotypes
- In our classification rule example, one possible fitness function would be information gain over training data.

# 4. Select fit individuals to breed new population

- Involves selecting the parents of the next generation
- All based upon the fitness of the individual

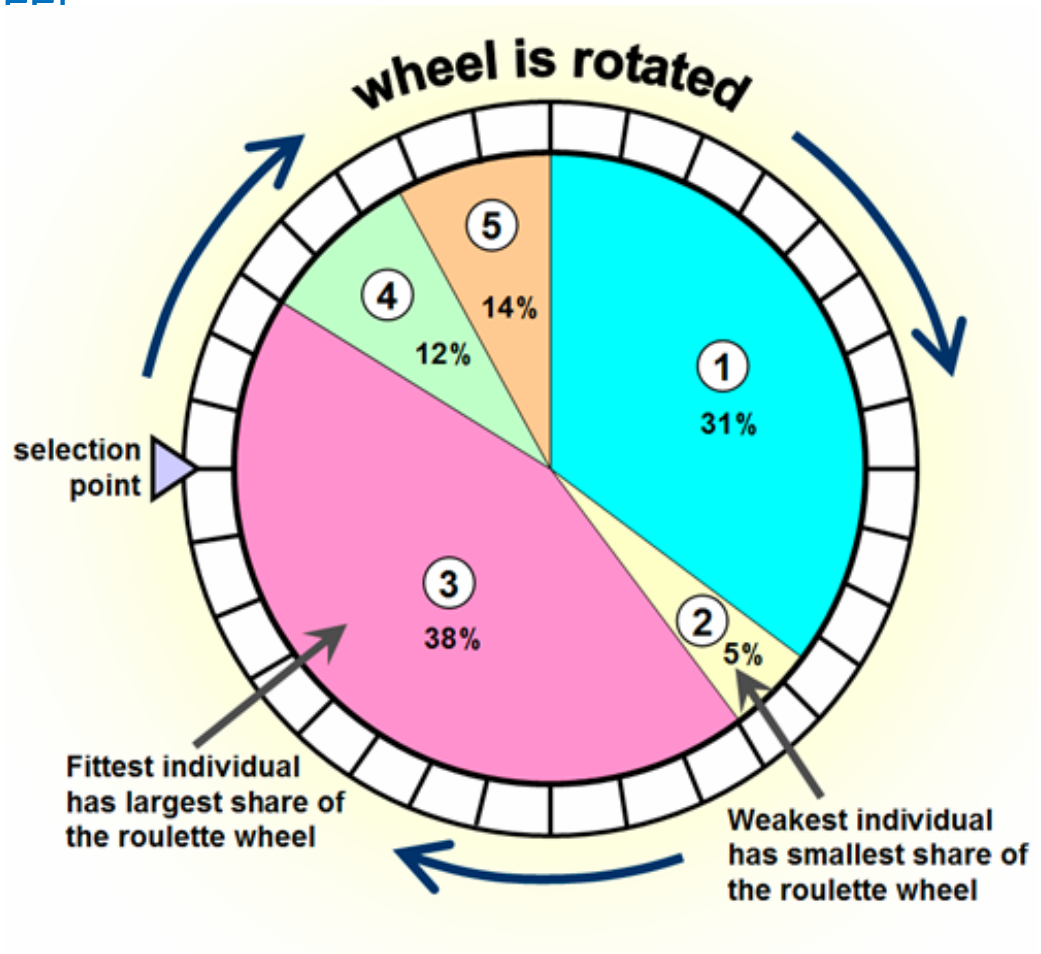


# 4. Select fit individuals to breed new population

- Selection schemes
  - **Roulette wheel selection**
    - Probabilistic selection based on fitness
  - **Rank selection**
    - Pick the best individual each time
  - **Tournament selection**
    - Select K individuals, and keep best for reproduction

# Roulette Wheel Selection

- Each individual is given a slice of a virtual roulette wheel
- Size of each slice is proportional to fitness
- Spin the wheel once to select each individual

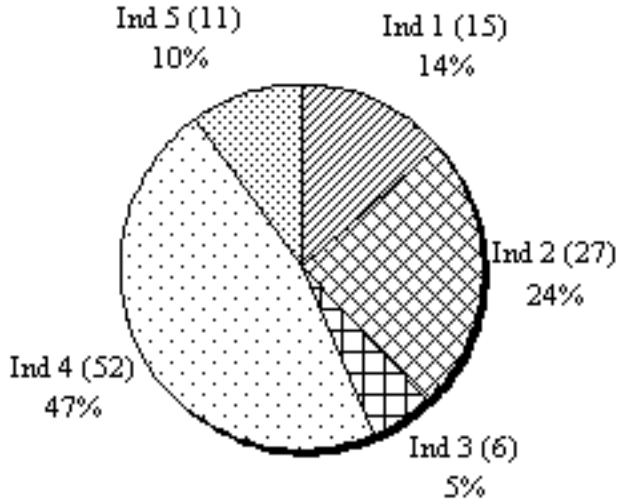



http://www.edc.ncl.ac.uk/highlight/rhjanuary2007g02.php/

# Roulette Wheel Selection

- Each individual is given a slice of a virtual roulette wheel
- Size of each slice is proportional to fitness
- Spin the wheel once to select each individual

| *Population* | *Fitness* |
|---|---|
| Individual 1 | 15 |
| Individual 2 | 27 |
| Individual 3 | 6 |
| Individual 4 | 52 |
| Individual 5 | 11 |




Randomly generated number = 21

Individual 2 is selected

# 5. Create new population from parents

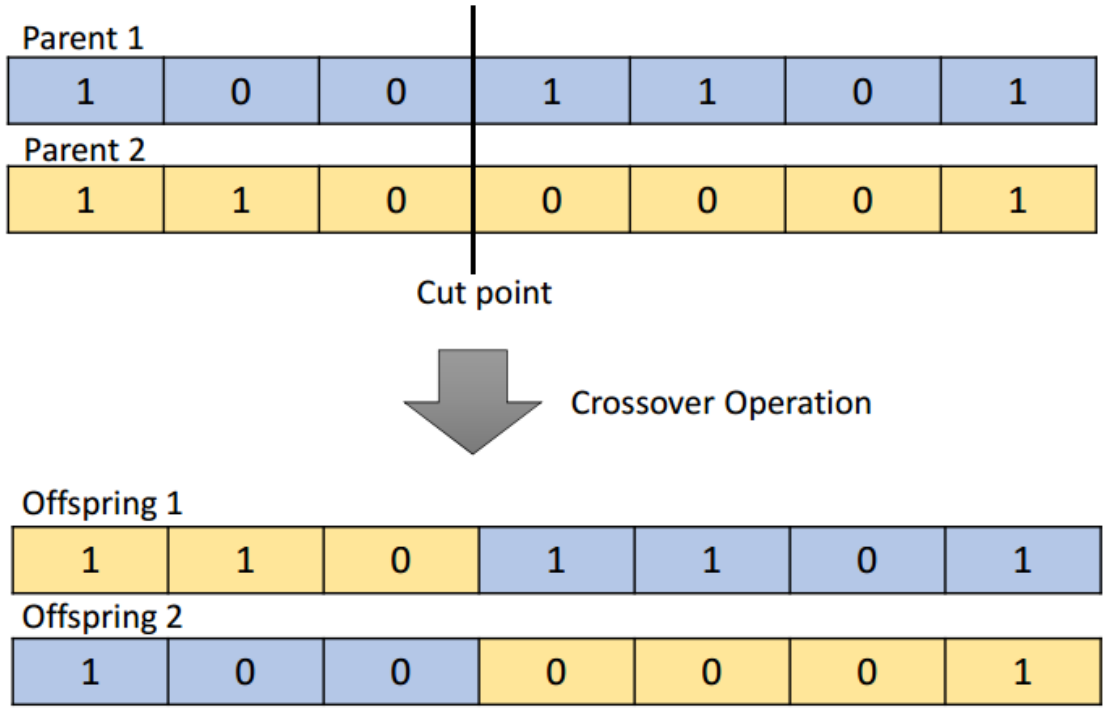
- Creates new individuals from selected parents
- Stochastic/Reproduction operators come into play
  - Crossover
  - Mutation

# Crossover

- Two chromosomes join at one or more points and exchange genes
- Generating offspring from two selected parents
- Types of crossover
  - Single point crossover
  - Two point/Multi point crossover
  - Uniform crossover

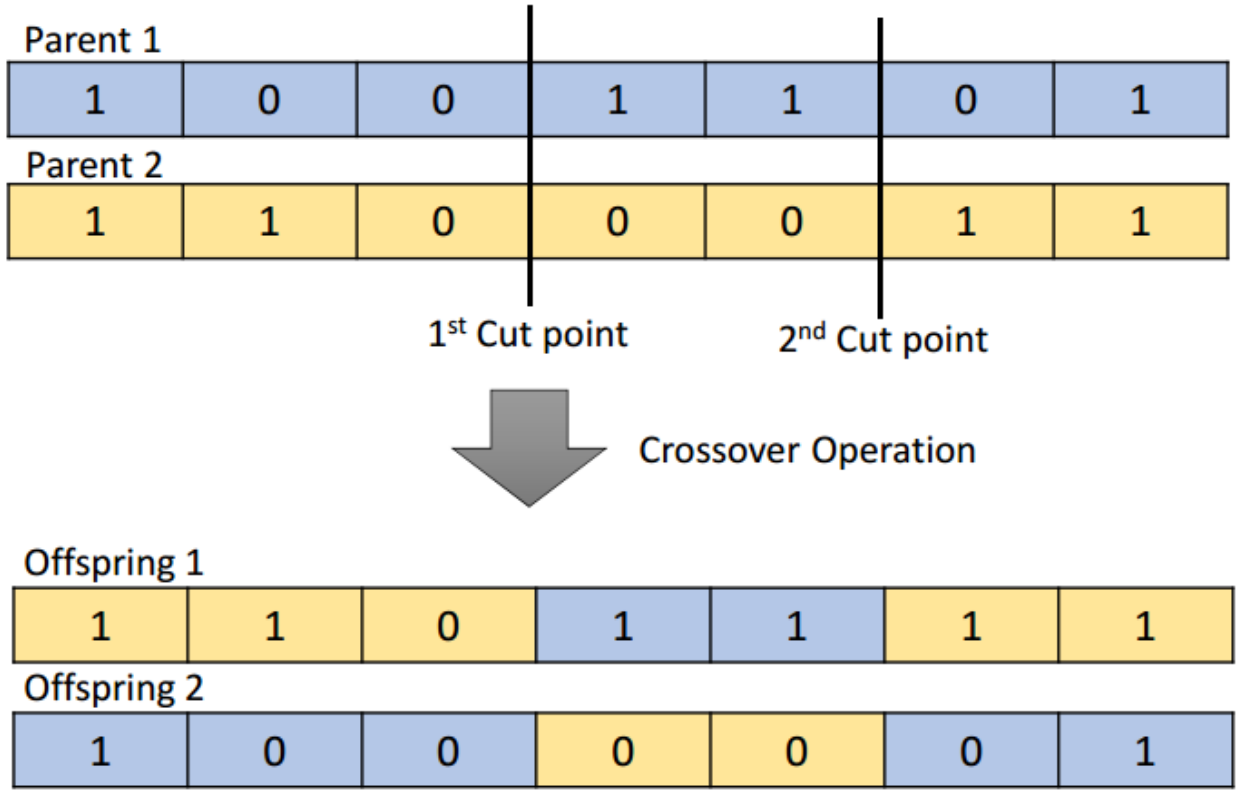
# Single Point Crossover

- It exchange the information between the parents and transform into offspring by selecting the random crossover point known as cut-point



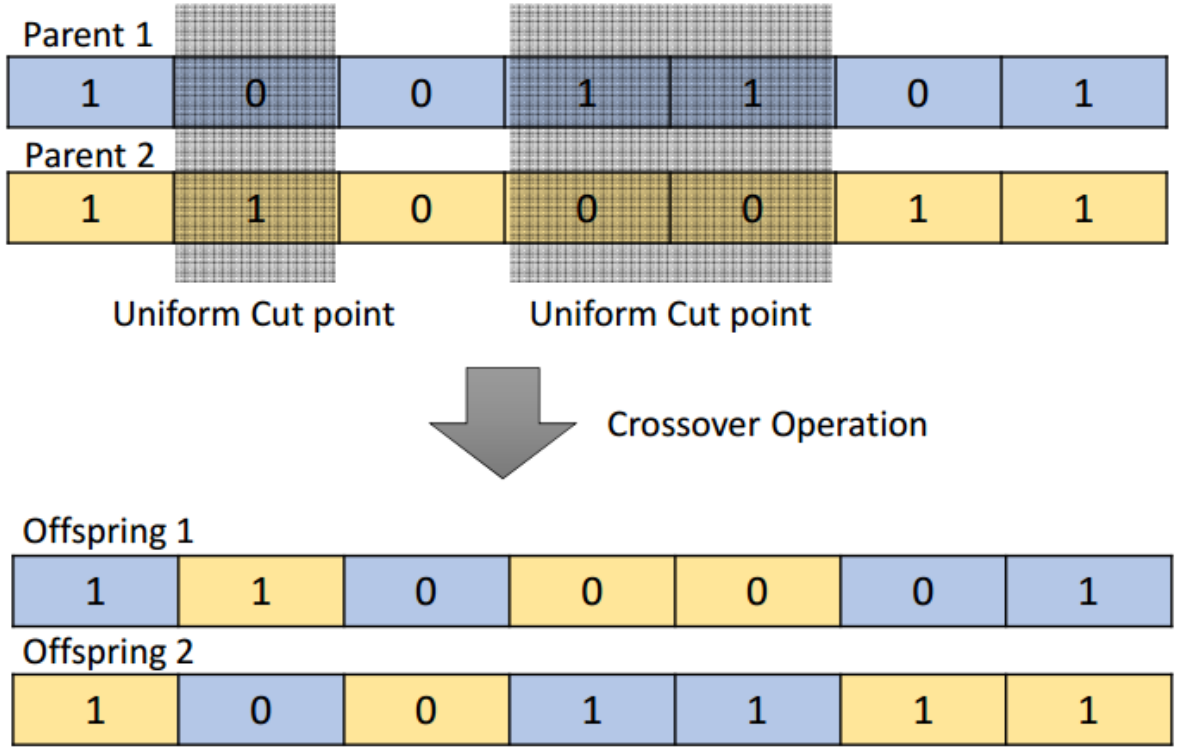
# Two-point crossover (Multipoint crossover)

- In two-point crossover, randomly two cut-points are selected



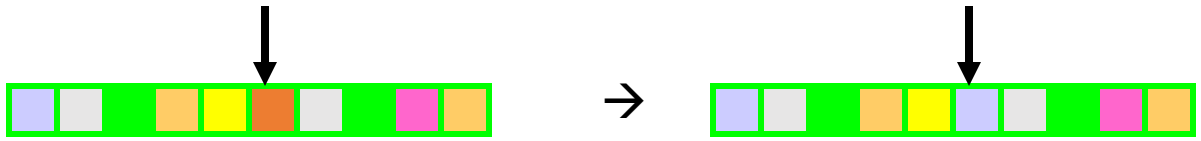
# Uniform crossover

- Crossover at each locus determined by a "coin toss"



# Mutation

- This operator is applied to add the diversity in the genetic characteristics of population
- Generating new offspring from single parent



- Crossover can only explore the combinations of the current gene pool
- Mutation can “generate” new genes

# Word Matching Example

- Problem is to ‘guess’ a word

- Difficult to solve with a brute force approach
  - assuming case sensitivity, to investigate every possible combination of letters in a seven letter word would require 8031810176 attempts

# Word Matching Example

- Assume trying to guess the word ‘genetic’
  - **Fitness Function:** Assign fitness based on number of correct letters in the correct place

- **Step one:** Select an encoding schema
  - Use characters

# Word Matching Example

- **Step two:** initialize the population randomly
  1. kdjirid
  2. ginddcc
  3. nmugjyb
  4. zezezez
  5. uhjklyt
  6. wojikli
  7. kladonn
  8. flortik

# Word Matching Example

- **Step three:** evaluate the population

| Individual | Genotype | Fitness |
|---|---|---|
| 1 | kdjirid | |
| 2 | ginddcc | |
| 3 | nmugjyb | |
| 4 | zezezez | |
| 5 | uhjklyt | |
| 6 | wojikli | |
| 7 | kladonn | |
| 8 | flortik | |

# Word Matching Example

- **Step three:** evaluate the population

| Individual | Genotype | Fitness |
|---|---|---|
| 1 | kdjirid | 1 |
| 2 | ginddcc | 3 |
| 3 | nmugjyb | 0 |
| 4 | zezezez | 2 |
| 5 | uhjklyt | 0 |
| 6 | wojikli | 0 |
| 7 | kladonn | 0 |
| 8 | flortik | 2 |

**Average fitness = 1**

# Word Matching Example

- **Step four:** select breeding population
  - Selection based on fitness, so breeding population is

| Individual | Genotype | Fitness |
|---|---|---|
| 2 | ginddcc | 3 |
| 2 | ginddcc | 3 |
| 2 | ginddcc | 3 |
| 4 | zezezez | 2 |
| 4 | zezezez | 2 |
| 8 | flortik | 2 |
| 8 | flortik | 2 |
| 7 | kdjirid | 1 |

# Word Matching Example

- **Step five:** create new population
  - Crossover comes into play
  - No mutation used here to keep things simple
  - For Example:  cross individual 2 with individual 4
    - individual 2 genotype is: ginddcc
    - individual 4 genotype is: zezezez

- Crossing over produces : genedec

# Word Matching Example

- New population is:

| Individual | Genotype | Fitness |
|---|---|---|
| 1 | genedec | 5 |
| 2 | glnrdic | 4 |
| 3 | fdoitik | 2 |
| 4 | zdziziz | 1 |
| 5 | gdnidic | 4 |
| 6 | feoetek | 3 |
| 7 | kijdrcd | 0 |
| 8 | zlzrez | 0 |

**average fitness = 2.375**

# Drawbacks of GA

- Difficult to find an encoding for a problem
- Difficult to define a valid fitness function
- May not return the global maximum

# Why use a GA?

- The problem has a very large solution space
- The problem is non-convex
- Does not require derivatives
- When the goal is a *good* solution

# When NOT to use a GA?

- If *global* optimality is required
- If the problem is smooth and convex
  - use a gradient-based optimizer

# Some Applications Areas

| Domain | Application Types |
|---|---|
| **Control** | gas pipeline, pole balancing, missile evasion |
| **Design** | semiconductor layout, aircraft design, keyboard configuration, communication networks |
| **Scheduling** | manufacturing, facility scheduling, resource allocation |
| **Robotics** | trajectory planning |
| **Machine Learning** | improving classification algorithms, classifier systems |
| **Signal Processing** | filter design |
| **Game Playing** | poker, checkers, prisoner's dilemma |
| **Combinatorial Optimization** | set covering, travelling salesman, routing, bin packing, graph coloring and partitioning |

# Summary

- Genetic algorithms are a class of evolutionary algorithm
- They are stochastic and population-based algorithms
- GA is able to efficiently investigate large search spaces

- Have two major requirements
  - Representation
  - Evaluation

# References


- Machine Learning Book by Tom M. Mitchell
- Slides from Richard Frankel at Stanford University
- Slides from Michael J. Watts

# Thank You ☺