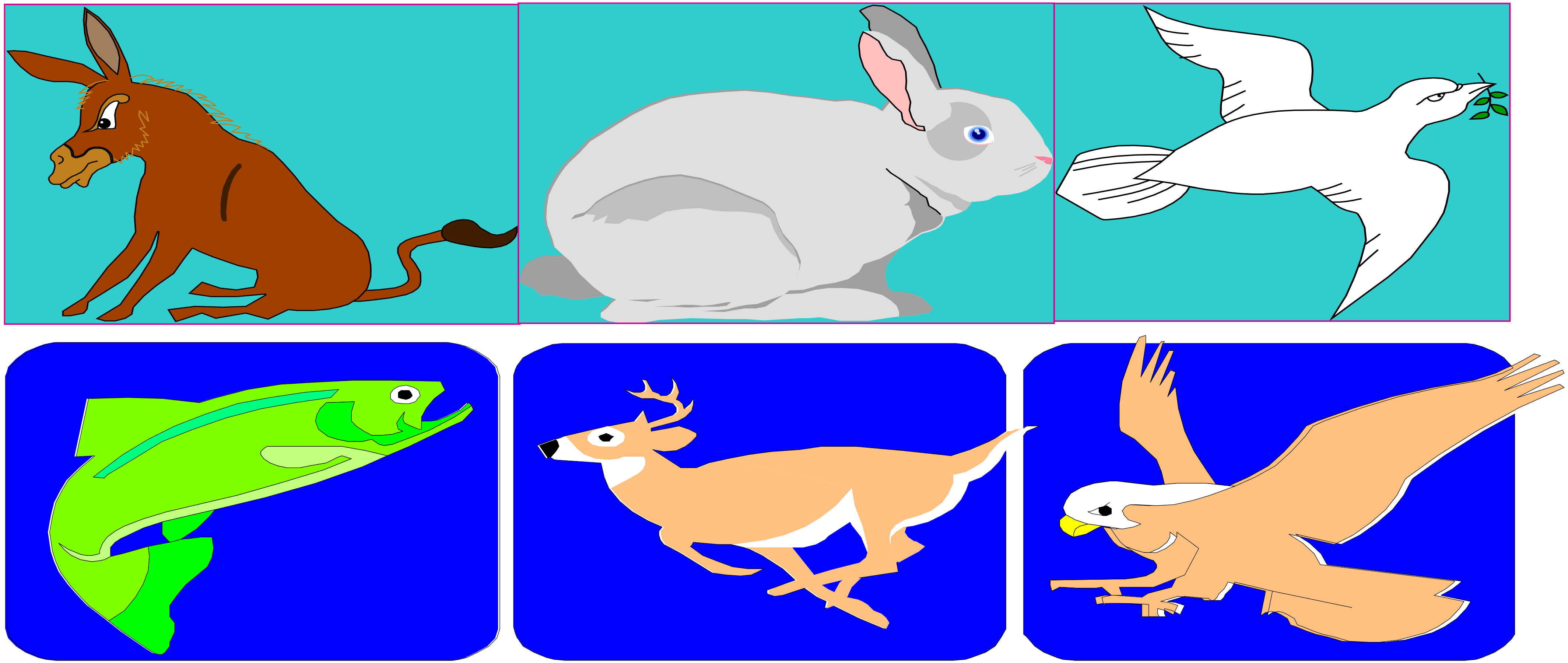
Local Search Algorithms and Optimization Problems in AI (Chapter 4)

Dr. Nasima Begum
Assistant Professor
Dept. of CSE, UAP

Genetic Algorithm



Survival of the fittest basis

Genetic Algorithm

Outline:

1. Introduction to GA
2. Main steps of GA
3. Genetic Operators
4. Examples
5. Summary

GA Introduction

- The Genetic algorithm, developed by **John Holland** [1962] is **an optimization (search) technique** that **operates over a population** of encoded candidate solutions to solve a given problem.
- GA belongs to a class of **stochastic search method based on biological evolution** (first observed by Charles Darwin). They represent a highly parallel adaptive search process.
- Genetic algorithms are commonly **used to generate high-quality solutions** to **optimization** and **search problems** by relying on biologically inspired operators such as **mutation, crossover** and **selection.**
- In natural (biological) evolution, **species search for increasingly beneficial adaptations** for survival within their complex environments.
- The **search takes place** in the species' **chromosomes** where changes and their effects are graded by **the survival** and **reproduction** of the species. This is the basis for **survival of the fittest.**

GA Introduction

- The central idea of GA is a **population** where individuals in the population represent possible solutions. An individual is called **chromosome**, in analogy with the genetic chromosome.

- The chromosome is usually represented by a **bit string** consisting of 0’s and 1’s.

- **New population** is generated from **old population** with **two** basic **genetic operators** namely **cross-over** and **mutation**.

Main Steps of GA

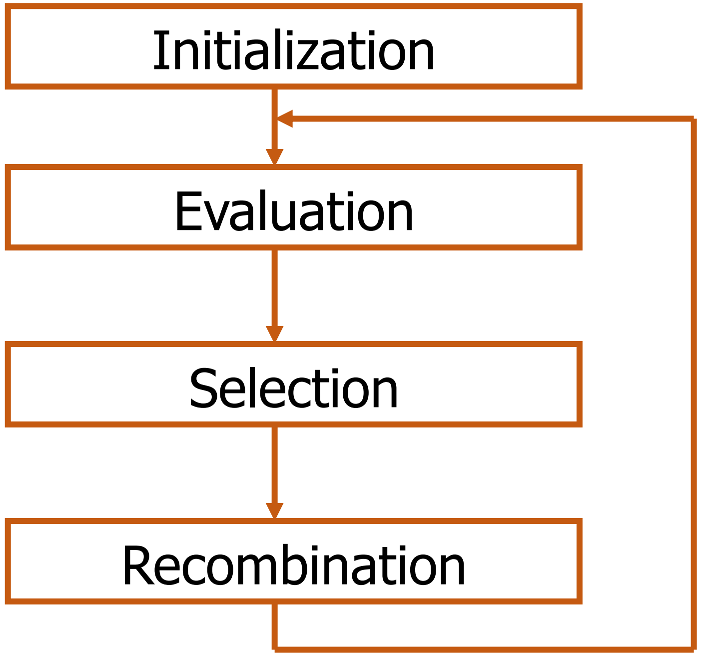
- A GA represents an **iterative process**. Each iteration is called a **generation**. The entire set of generations is called a **run**.
- At the end of a run, we expect to find **one or more highly fit chromosomes**.
- The GA consists of **three** fundamental **steps**, excluding the initialization. These are: **Evaluation, Selection,** and **Recombination.**

Initialization: initial creation of the population.

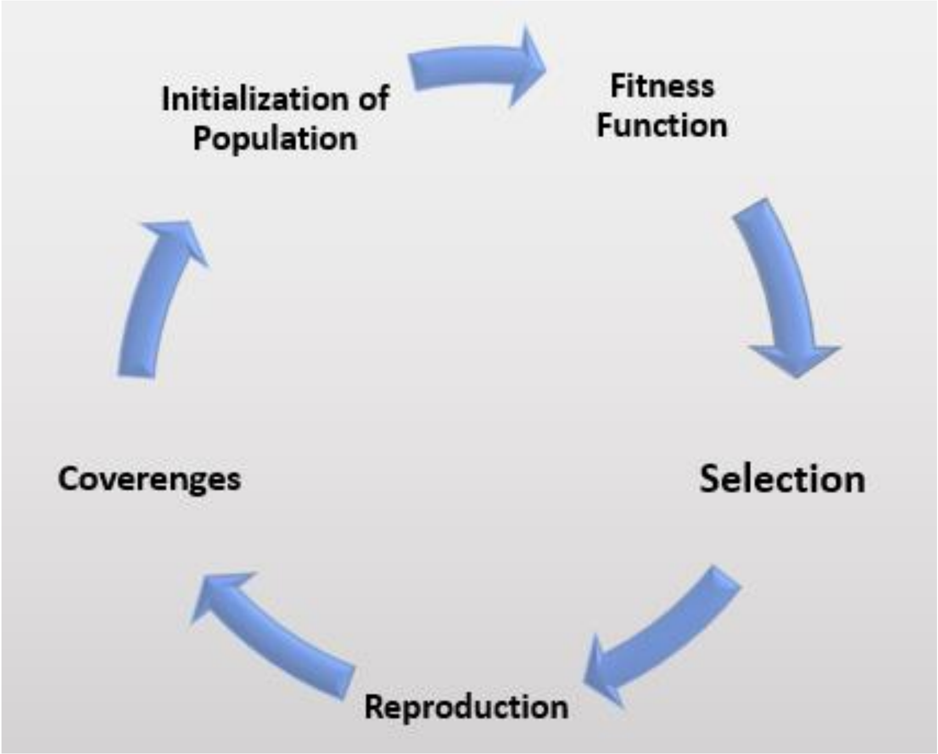
Evaluation: fitness of the population is calculated.

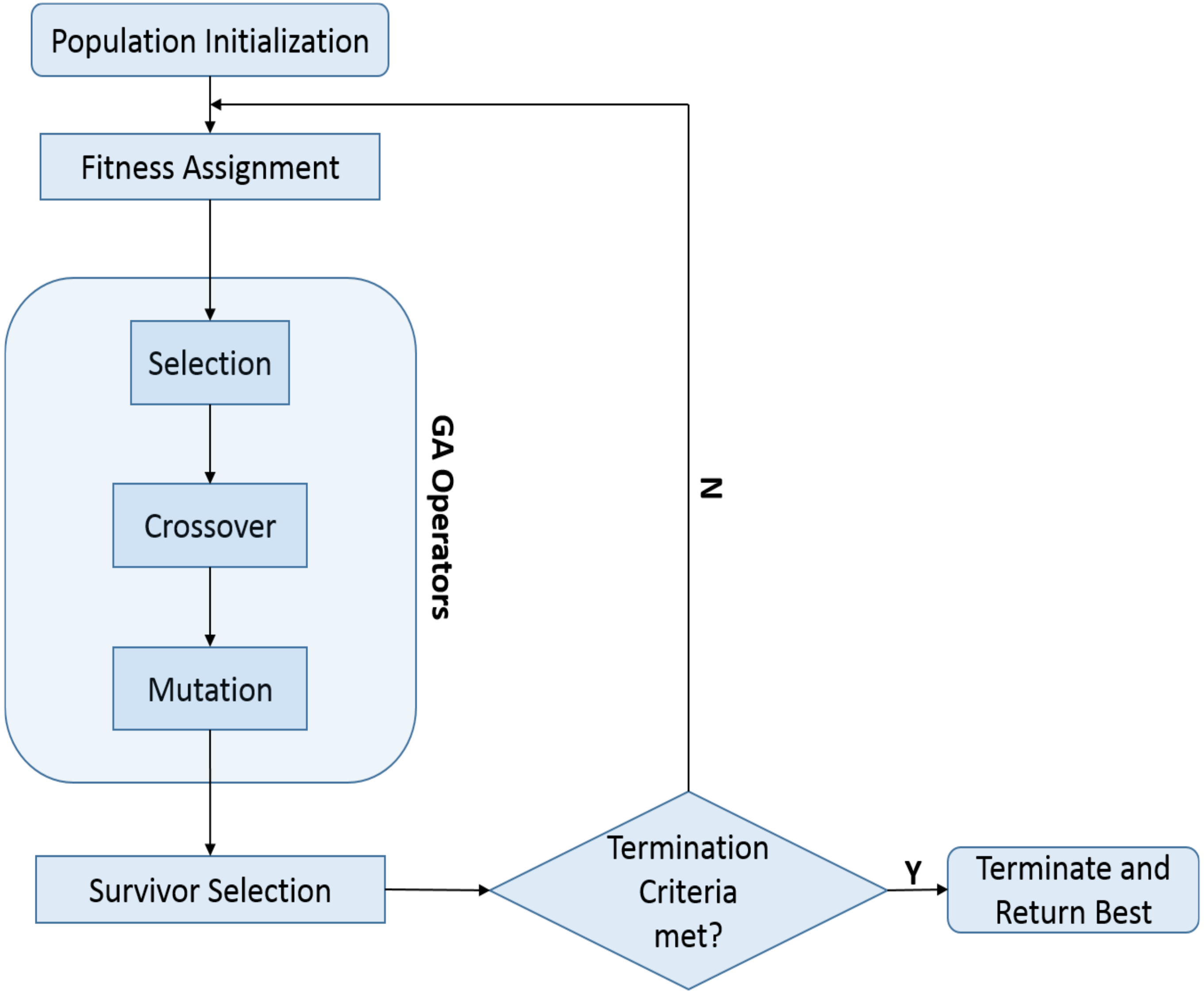
Selection: a subset of the population is selected based upon a predefined selection criterion.

Recombination: selected sub-population is recombined to result a new population.



Main Steps of GA





Flow chart of GA

Working Principal of GA:

Start: It generates a random population of ***n*** chromosomes.

Fitness: It calculates the ***fitness f(x)*** of each chromosome ***x*** in the population.

New Population: It generates a new population by **repeating** the following steps until the New population is finished.

Selection: It ***chooses two parent chromosomes*** from a population as per their fitness. The better the fitness, the **higher the probability** of getting selected.

Crossover: In crossover probability, ***cross over the parents*** to form new offspring (children). If **no crossover was performed**, the offspring is the exact copy of the parents.

Working Principal of GA:

Mutation: In mutation probability, ***mutate*** new offspring **at each locus**.

Accepting: It places ***new offspring*** in the new population.

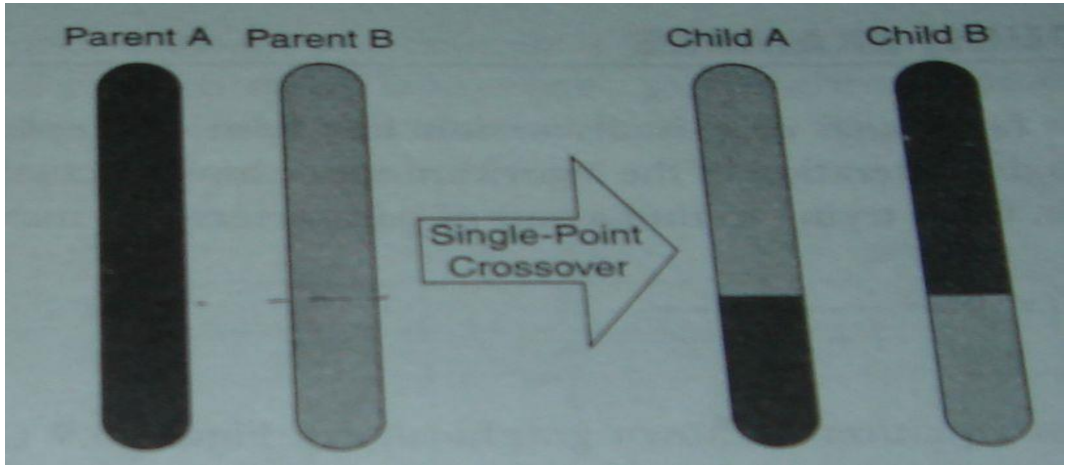
Replace: It ***uses the newly generated population*** for a ***further run*** of the algorithm.

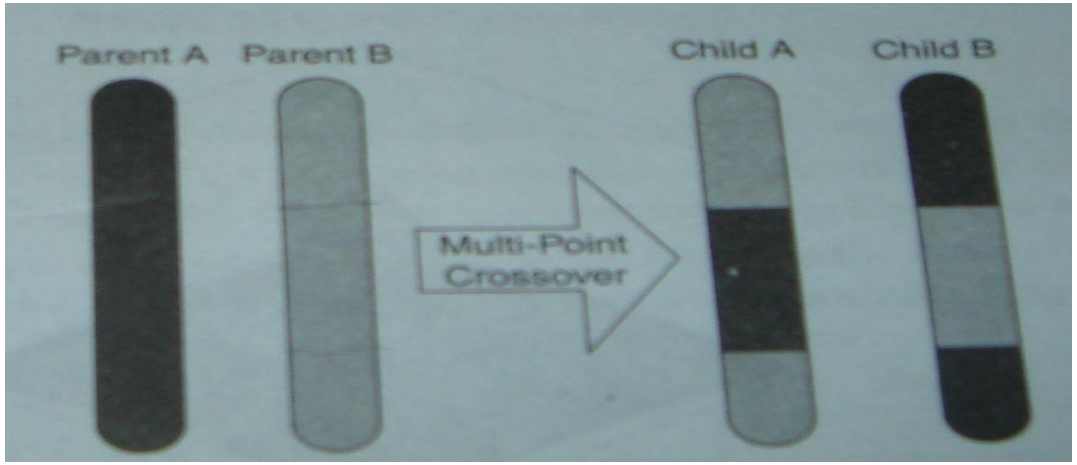
Test: If the end ***condition is satisfied***, then it ***stops*** and ***returns the best solution*** in the current population.

Loop: In this step, it need to go to the second step for fitness evaluation.

Genetic Operator: Cross-over

- **<u>Cross-over:</u>** takes two chromosomes (parents), separates them at a random site (in both chromosomes) and then swaps the tails of the two, resulting in two new chromosomes (children).



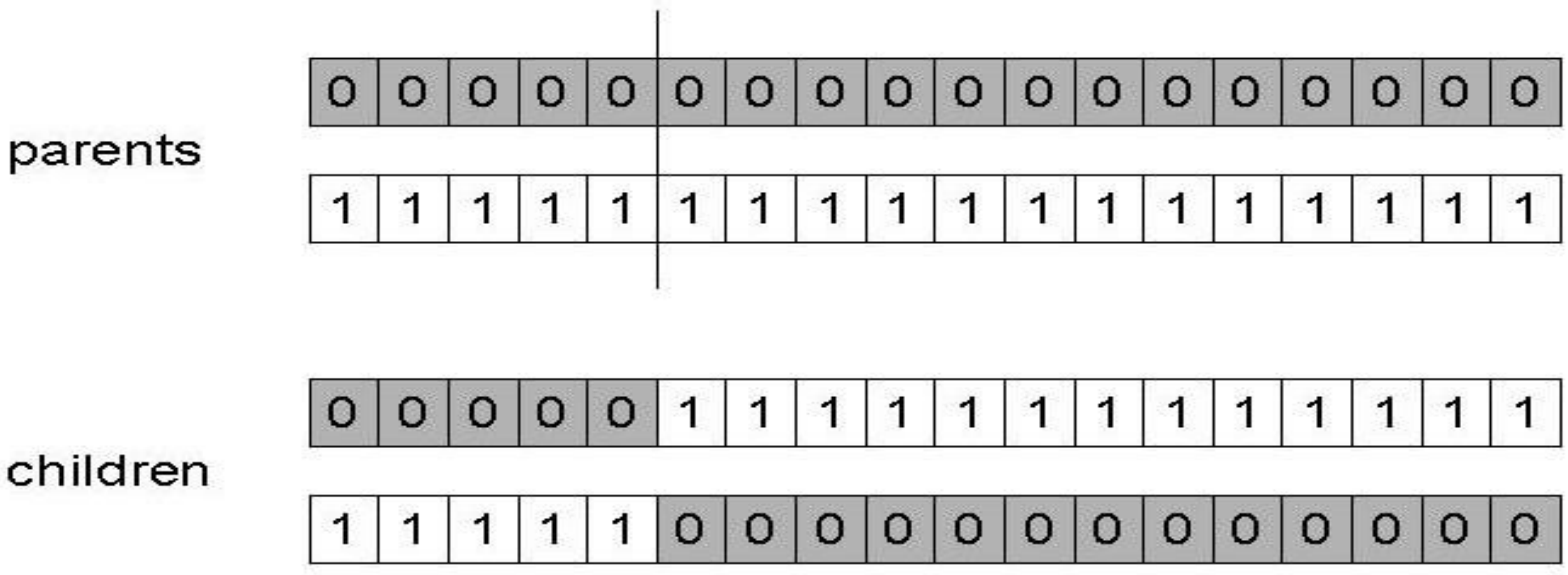


- Cross over operator randomly chooses a crossover point where two parent chromosomes ‘break’ and then exchanges the chromosome parts after that point.
- A value of 0.7 for the cross-over probability generally produces good results.

Genetic Operator: Cross-over

Single-point cross-over

- Choose a random point on the two parents
- Split parents at this crossover point
- Create children by exchanging tails



Genetic Operator: Mutation

- Mutation, which is rare in nature **represents a change in the gene**. It may lead to a significant improvement in fitness, but more often has rather harmful results.

- The mutation operator introduces a **random change into a gene** in the chromosome. It provides the ability to introduce new material into the chromosome.

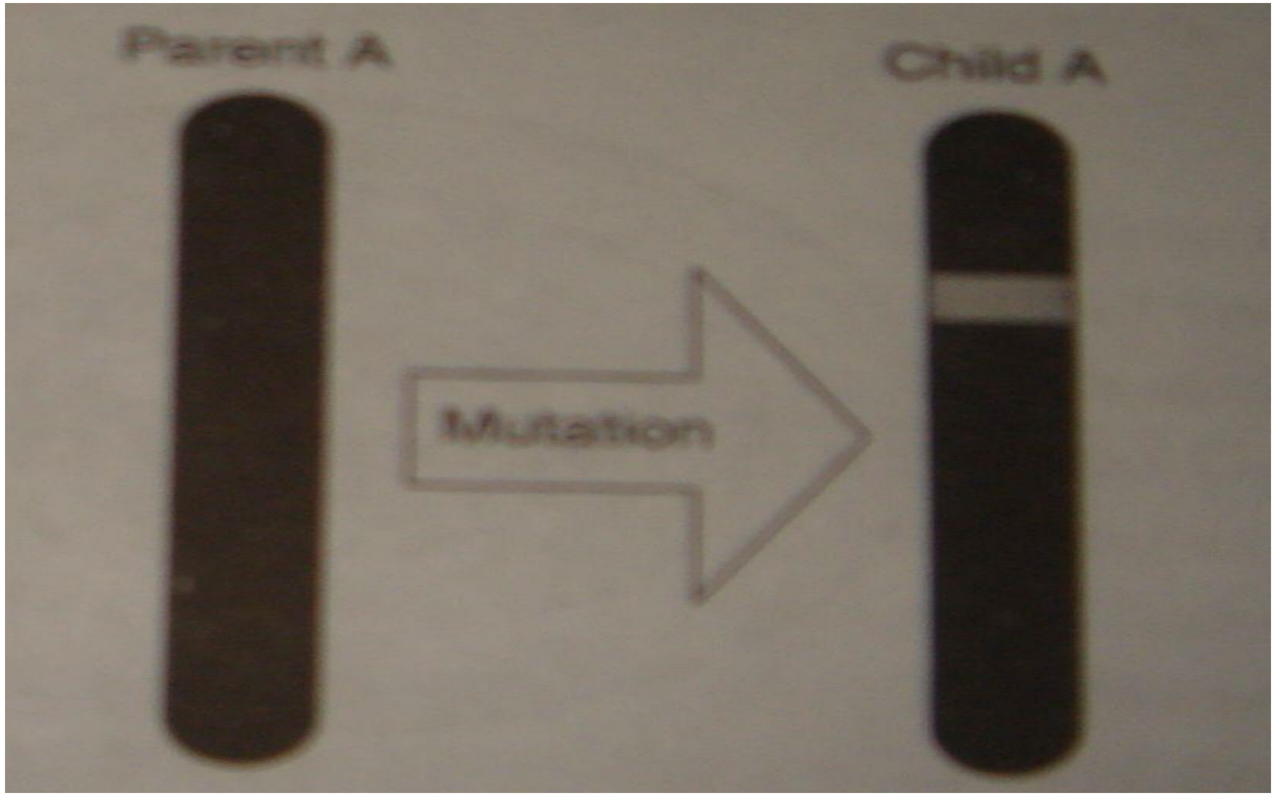
- The mutation probability is quite small in nature, in the range 0.0001 and 0.01.

Genetic Operator: Mutation

Why use Mutation:

- Mutation's role is to provide a **guarantee** that the search algorithm is **not trapped on a local optimum**. The sequence of selection and crossover operations may **stagnate** at any **homogeneous set of solutions**. Under such conditions, all **chromosomes are identical**, and thus the average fitness of the population cannot be improved. The search algorithm is not able to proceed further. **Mutation aids us in avoiding loss of genetic diversity.**

Genetic Operator: Mutation



Genetic Algorithms

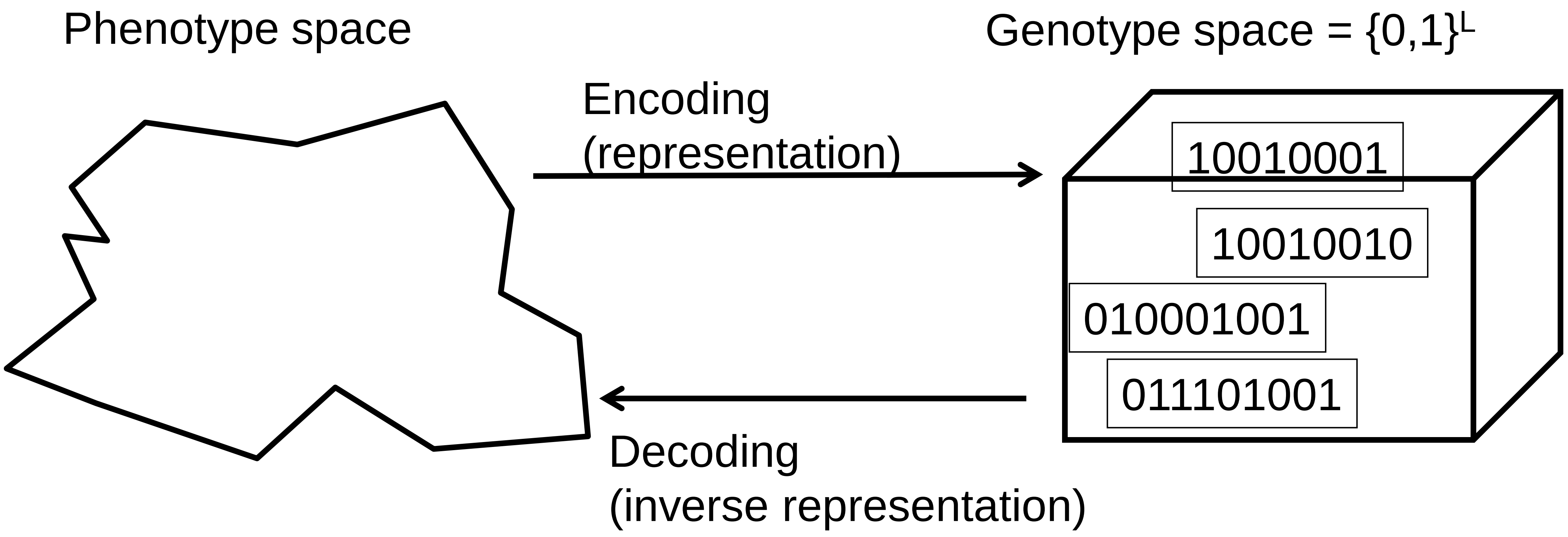
- Holland’s original GA is now known as the simple genetic algorithm (SGA)

- Other GAs use different:
  - Representations
  - Mutations
  - Crossovers
  - Selection mechanisms

SGA technical summary table

Representation	Binary strings
Recombination	N-point or uniform
Mutation	Bitwise bit-flipping with fixed probability
Parent selection	Fitness-Proportionate
Survivor selection	All children replace parents
Speciality	Emphasis on crossover

Representation

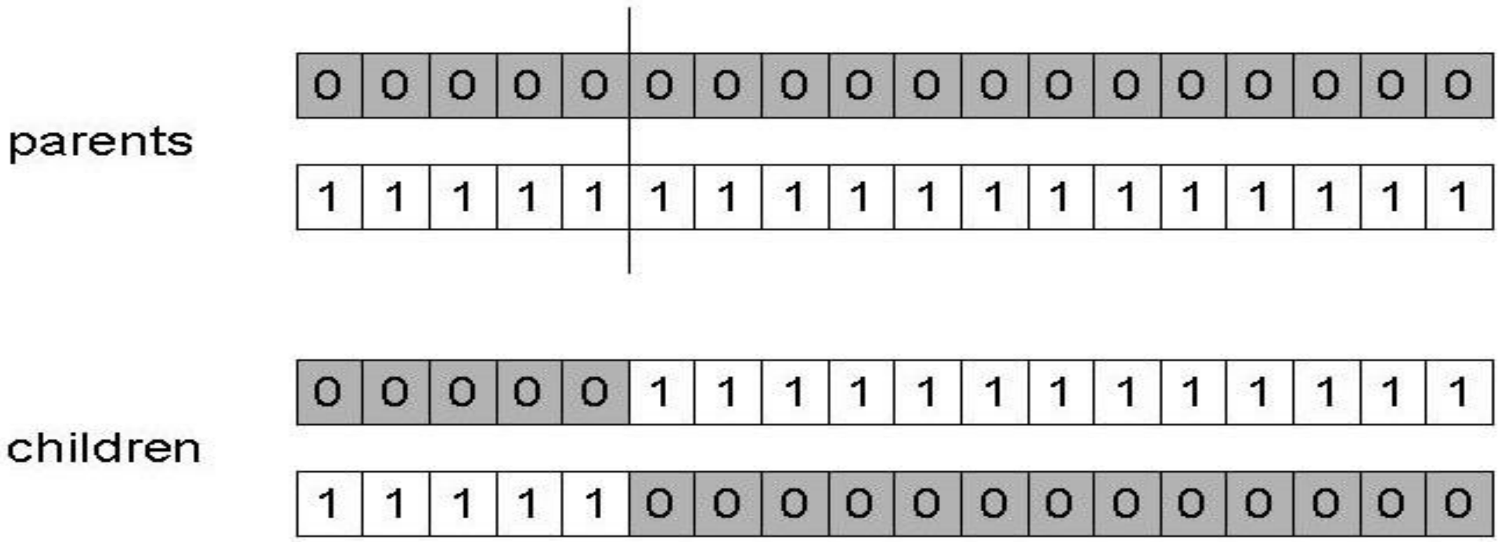


SGA Reproduction Cycle

1. Select parents for the mating pool (size of mating pool = population size)
2. Shuffle the mating pool
3. For each consecutive pair, apply crossover with probability p_c , otherwise copy parents
4. For each offspring apply mutation (bit-flip with probability p_m independently for each bit)
5. Replace the whole population with the resulting offspring

SGA operators: 1-point crossover

- Choose a random point on the two parents
- Split parents at this crossover point
- Create children by exchanging tails
- P_c typically in range (0.6, 0.9)



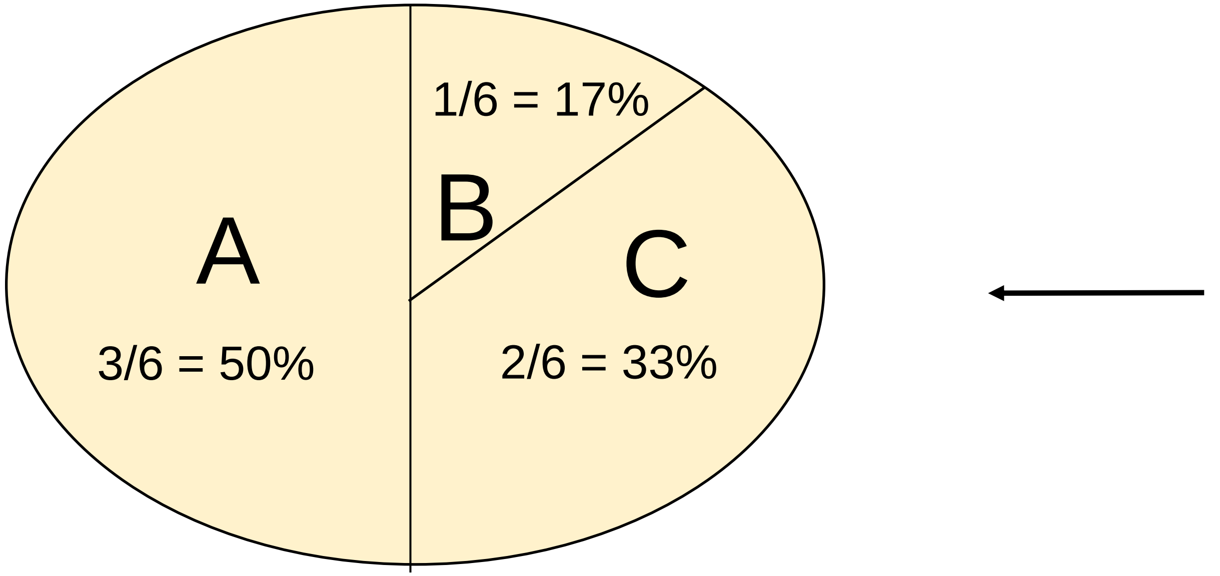
SGA Operators: Mutation

- Alter each gene independently with a probability p_m
- p_m is called the mutation rate
  - Typically between 1/pop_size and 1/ chromosome_length

parent	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
child	0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1

SGA operators: Selection

- Main idea: better individuals get higher chance
  - **Chances proportional to fitness**
  - Implementation: **roulette wheel** technique
    - Assign to each individual a part of the roulette wheel
    - Spin the wheel n times to select n individuals



fitness(A) = 3

fitness(B) = 1

fitness(C) = 2

An example after Goldberg '89 (1)

- Simple problem: f(x) = {MAX (x^2): 0 <= x <= 32 } over {0,1,…,31}
- GA approach:
  - Encode Solution: Just use 5 bits (1 or 0).
  - Representation: binary code, e.g. 01101 $\leftrightarrow$ 13
  - Generate initial population.
  - Population size: 4
  - 1-point xover, bitwise mutation
  - Roulette wheel selection
  - Random initialization
- Lets se one generational cycle done by hand

x^2 example: selection

String no.	Initial population	x Value	Fitness $f(x) = x^2$	$Prob_i$	Expected count	Actual count
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1
Sum			1170	1.00	4.00	4
Average			293	0.25	1.00	1
Max			576	0.49	1.97	2

i

Calculation Draft:

i = No. of population = 4

Probability: 169/1170 = 0.144, 576/1170 = 0.492, …….

293/1170 = 0.250, ….

Expected Count = 0.144x4 = 0.576, 0.492x4 = 1.968, 0.25x4 = 1,

x^2 example: crossover

String no.	Mating pool	Crossover point	Offspring after xover	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 | 1	4	0 1 1 0 0	12	144
2	1 1 0 0 | 0	4	1 1 0 0 1	25	625
2	1 1 | 0 0 0	2	1 1 0 1 1	27	729
4	1 0 | 0 1 1	2	1 0 0 0 0	16	256
Sum					1754
Average					439
Max					729

x^2 example: mutation

String no.	Offspring after xover	Offspring after mutation	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 0	$\boxed{1}$ 1 1 0 0	26	676
2	1 1 0 0 1	1 1 0 0 1	25	625
2	1 1 0 1 1	1 1 0 1 1	27	729
4	1 0 0 0 0	1 0 $\boxed{1}$ 0 0	18	324
Sum				2354
Average				588.5
Max				729

A Simple GA Example (cont.)

- Create next generation of solutions
  - Probability of “being a parent” depends on the fitness.
- Ways for parents to create next generation
  - Crossover
    - Cut and paste portions of one string to another.
  - Mutation
    - Randomly flip a bit.
  - COMBINATION of all of the above.

GA Summary

- GA is a heuristic method based on ' **survival of the fittest**.
- Useful when search space is very large or too complex for analytic treatment.
- GA has been employed in a wide variety of practical problems related to pattern recognition and image processing, computer-aided design, scheduling, economics and game theory, and so on.
- In computer science, there is a large set of problems, which are **NP-Hard**. What this essentially means is that, even the most powerful computing systems take a very long time (even years!) to solve that problem. In such a scenario, GAs prove to be an efficient tool to provide **usable near-optimal solutions** in a short amount of time.

Acknowledgement

- AIMA = Artificial Intelligence: A Modern Approach by Stuart Russell and Peter Norving (3rd edition)
- UC Berkeley (Some slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley)
- U of toronto
- Other online resources

Thank You