

Genetic Algorithms

Nature-Inspired Optimization Methods

{Vinh Dinh Nguyen, Vinh Dinh-Quang}
PhD in Computer Science

Outline



➤ **Introduction to Evolutionary Algorithms**

➤ **Introduction to Genetic Algorithms**

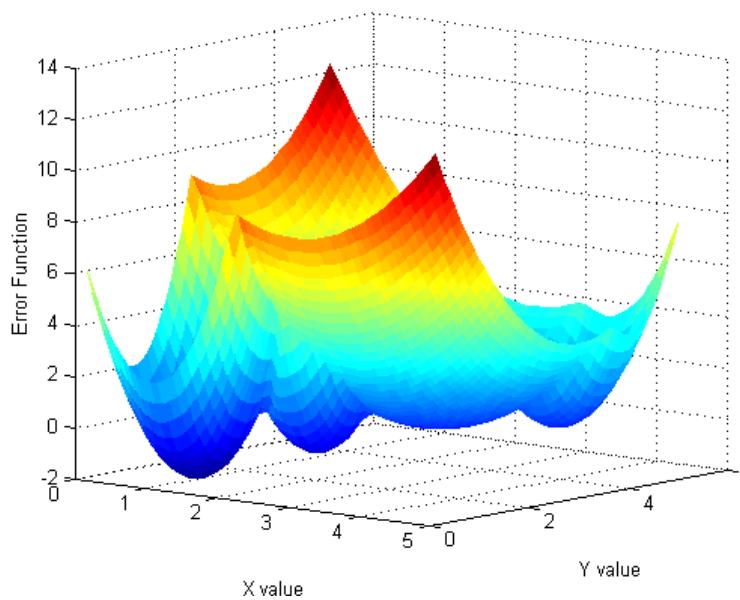
➤ **One-Max Problem**

➤ **Sphere Function**

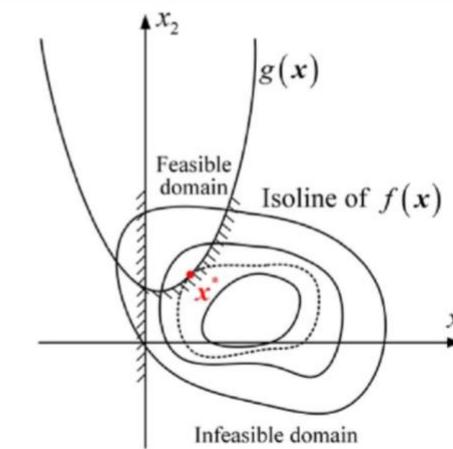
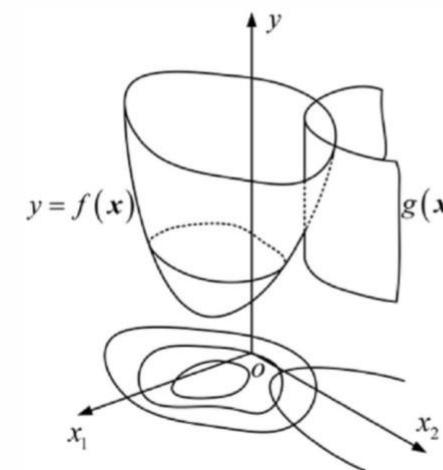
➤ **Regression Problem**

➤ **GA Applications**

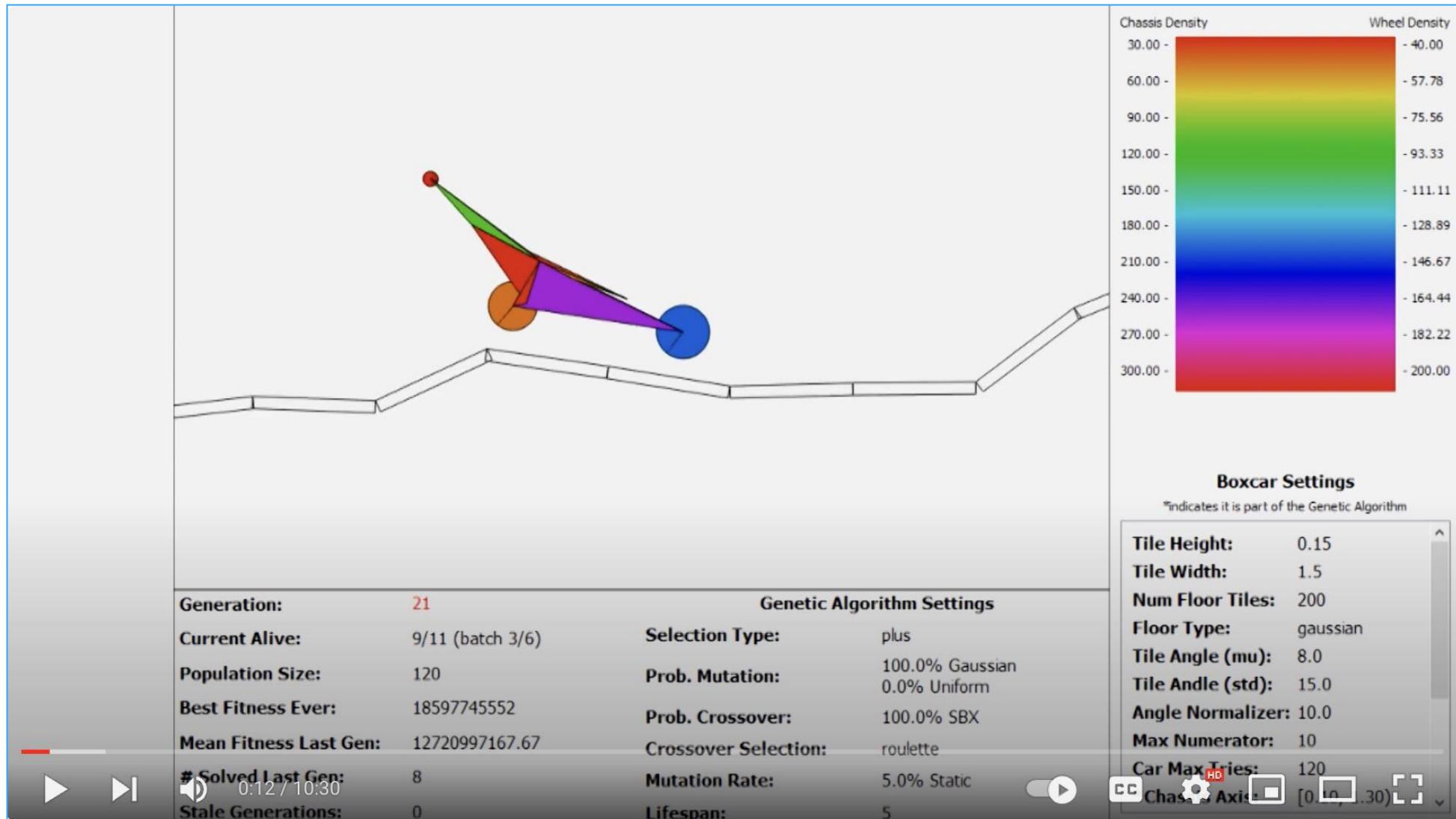
Genetic Algorithms are **heuristic search** algorithms that solve **constrained and unconstrained optimization** problems using the concepts of natural selection — a famous discipline in biological evolution.



What is Constrained Optimization

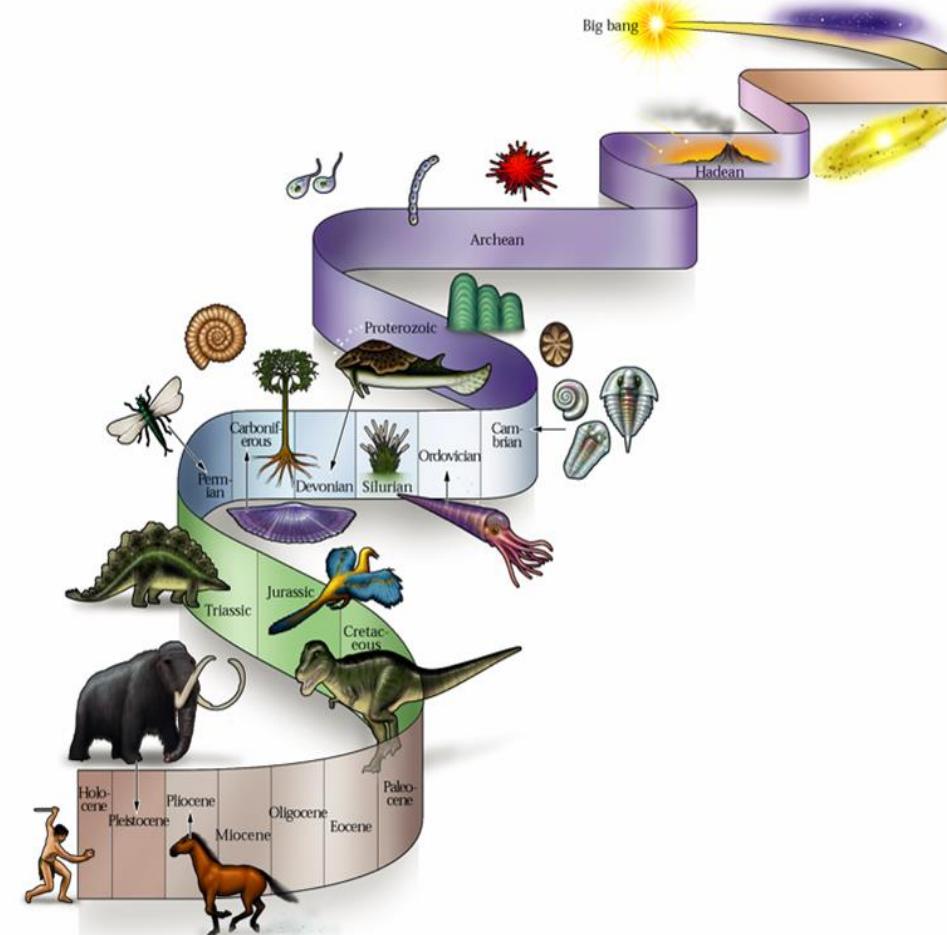


AI Learns to be a Car using a Genetic Algorithm



<https://www.youtube.com/watch?v=EI2g8XoCkdg>

Principle of Evolutionary Algorithms

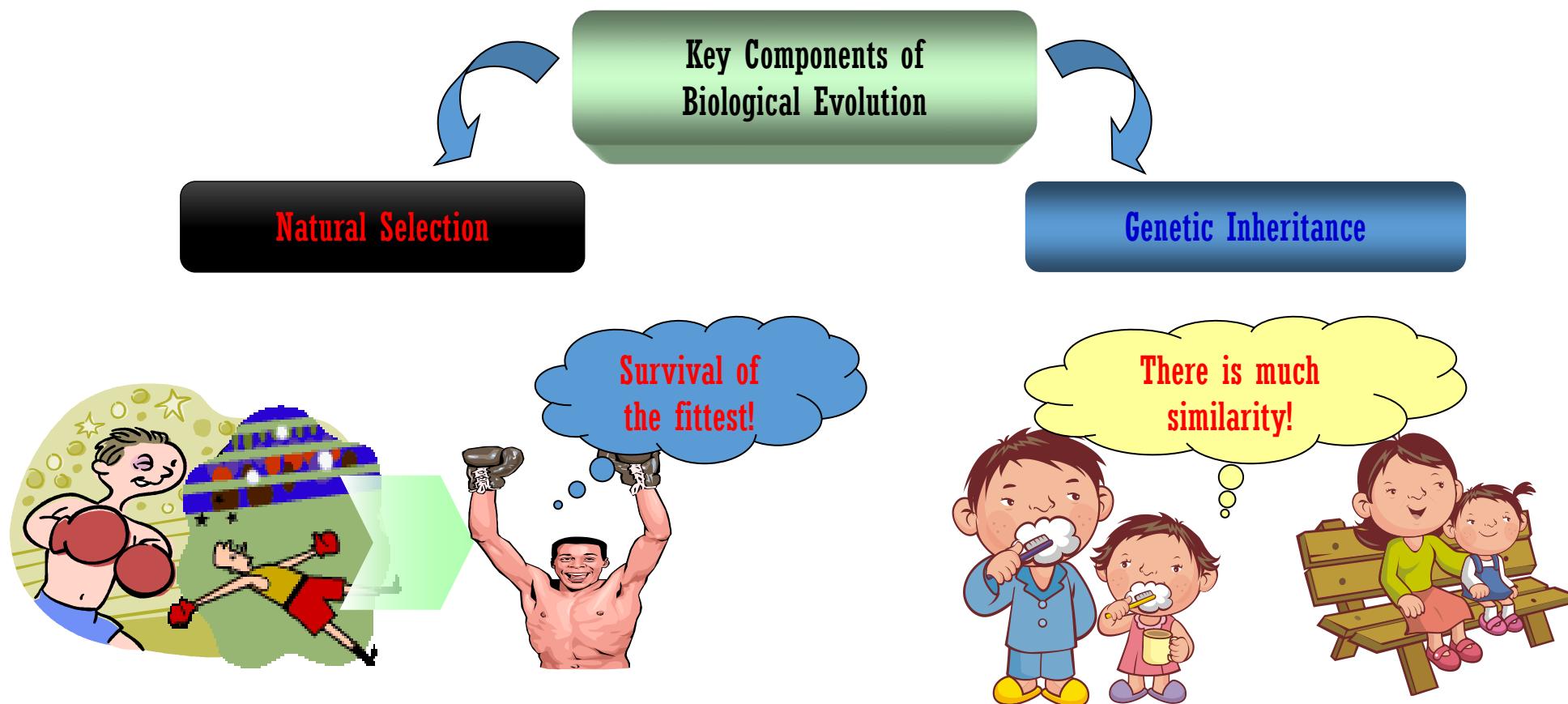


From Prof. Ahn

Evolutionary Algorithms: Principles

Evolutionary Algorithms (EAs)

- Any problem-solving method inspired from the theory of biological evolution, usually implemented on computers, which is employed for resolving problems



Evolutionary Algorithms: Principles

Lessons from Biological Evolution

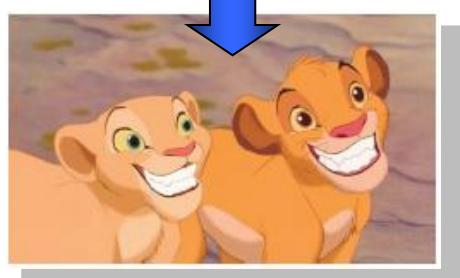
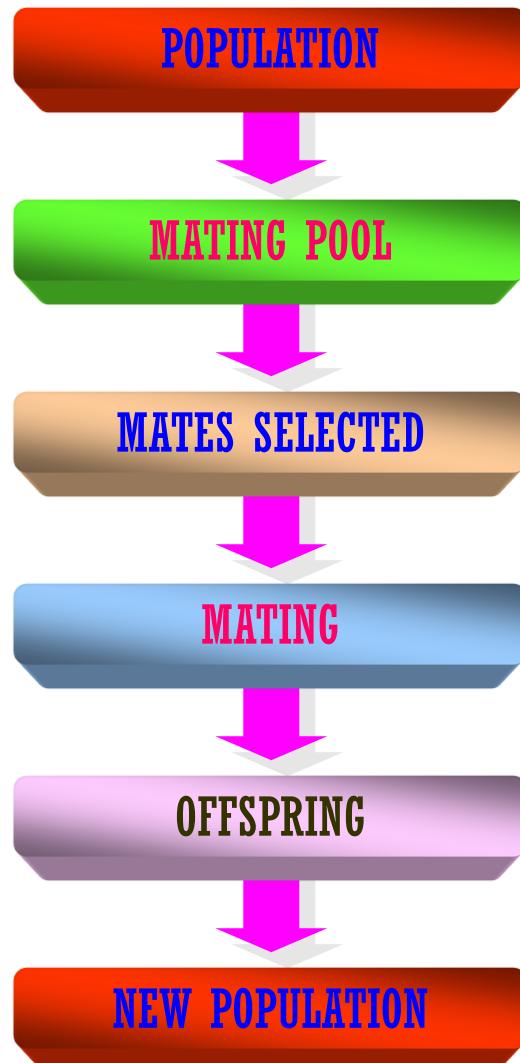
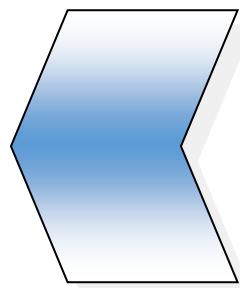
Implications for applying
to computing techs.

Multiple

Surviving

Mixing

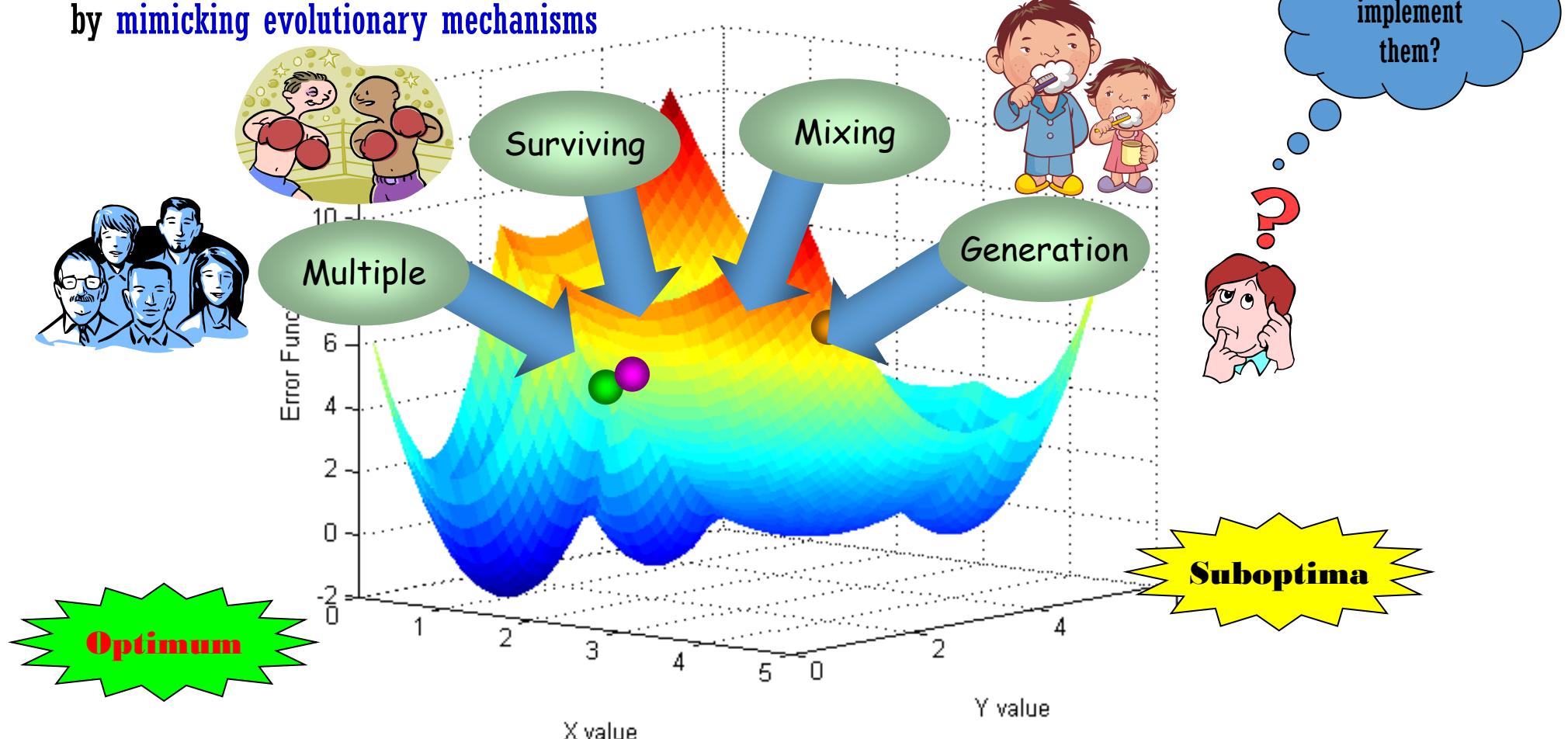
Generation



Evolutionary Algorithms: Principles

Main Principle of Evolutionary Algorithms

- Multiple individuals try to **cooperatively** resolve problems by **mimicking** evolutionary mechanisms



Outline



➤ **Introduction to Evolutionary Algorithms**

➤ **Introduction to Genetic Algorithms**

➤ **One-Max Problem**

➤ **Sphere Function**

➤ **Regression Problem**

➤ **GA Applications**

Genetic Algorithms



Genetic Algorithms

● What's the Target of Interest?

➤ Optimization Problems

- ✓ Can be defined by specifying the set of all feasible candidates
- ✓ The goal is to find the best solution(s)

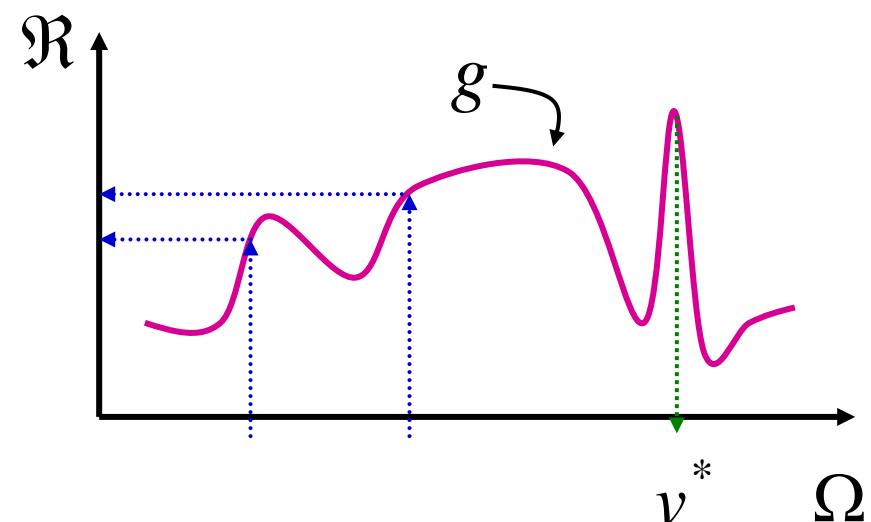
Formal Definition

For a search space Ω

There is a function $g : \Omega \mapsto \mathcal{R}$

The task is to find $v^* = \arg \max_{v \in \Omega} g$

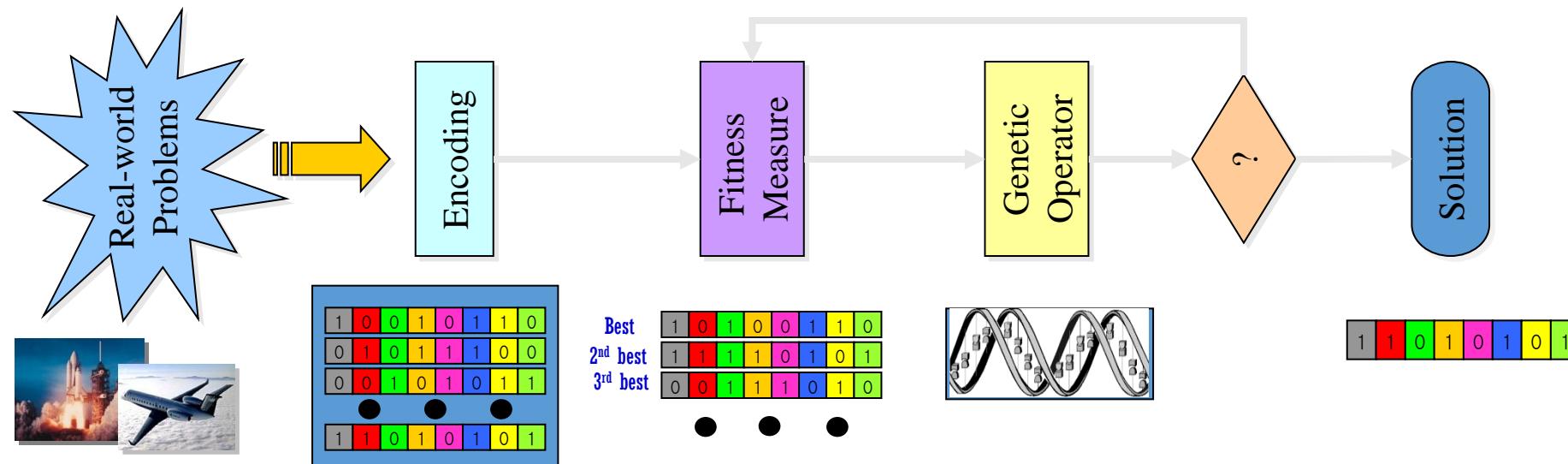
Here, v is a vector of decision variables,
and g is the objective function



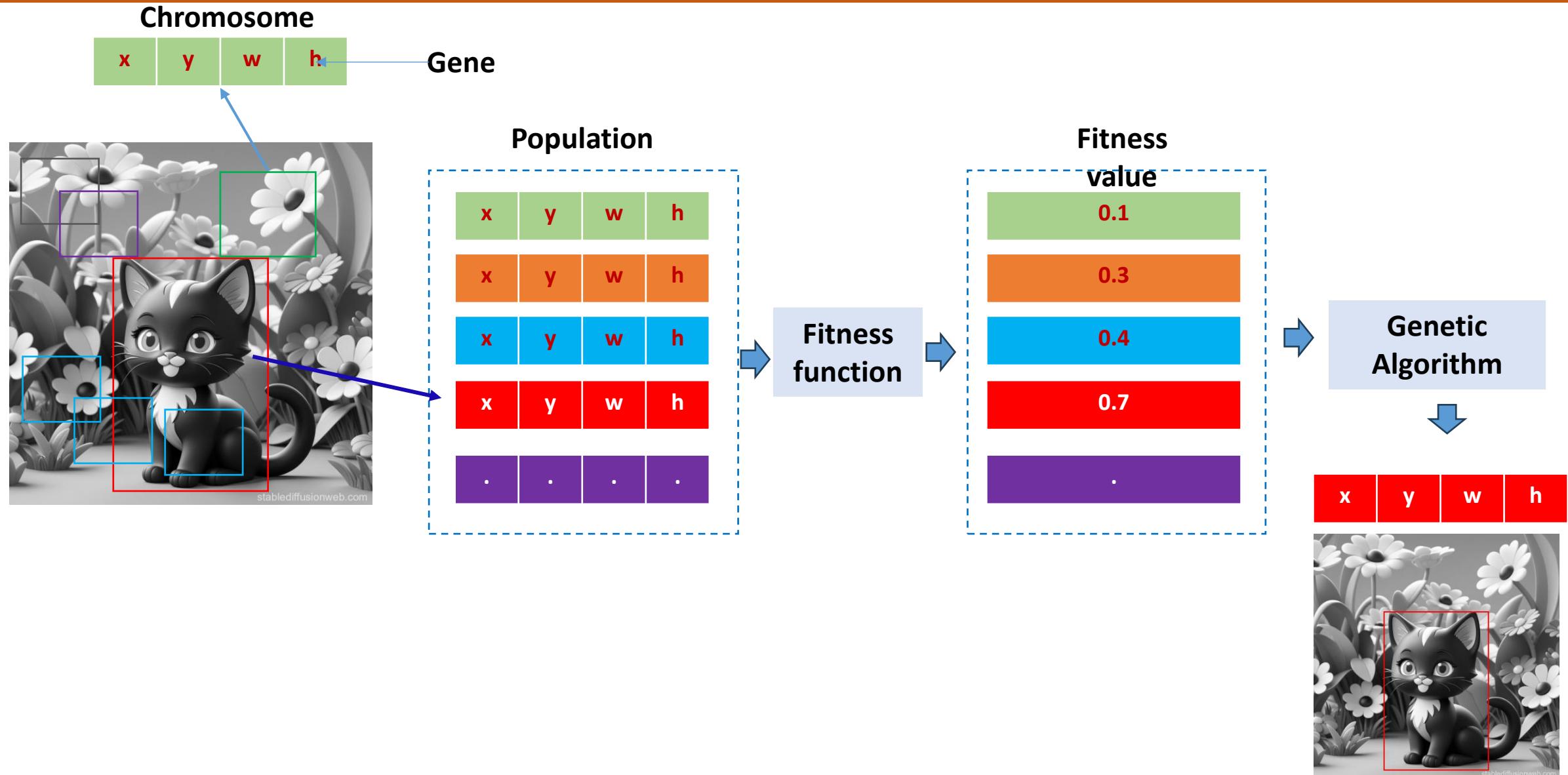
Genetic Algorithms

● Key Components & Terminology

- **Encoding:** variables (phenotype) are encoded into a chromosome (genotype)
- **Population:** a set of chromosomes (i.e., individuals or candidate solutions)
- **Fitness function:** measure the goodness of each candidate solution:
it can be mathematical terms, computer simulation, human evaluation
- **Genetic operators:** boosting chromosomes up towards the optimum
 - ✓ **Selection:** realize the survival of the fittest
 - ✓ **Crossover:** realize the genetic inheritance
 - ✓ **Mutation:** realize the genetic mutation

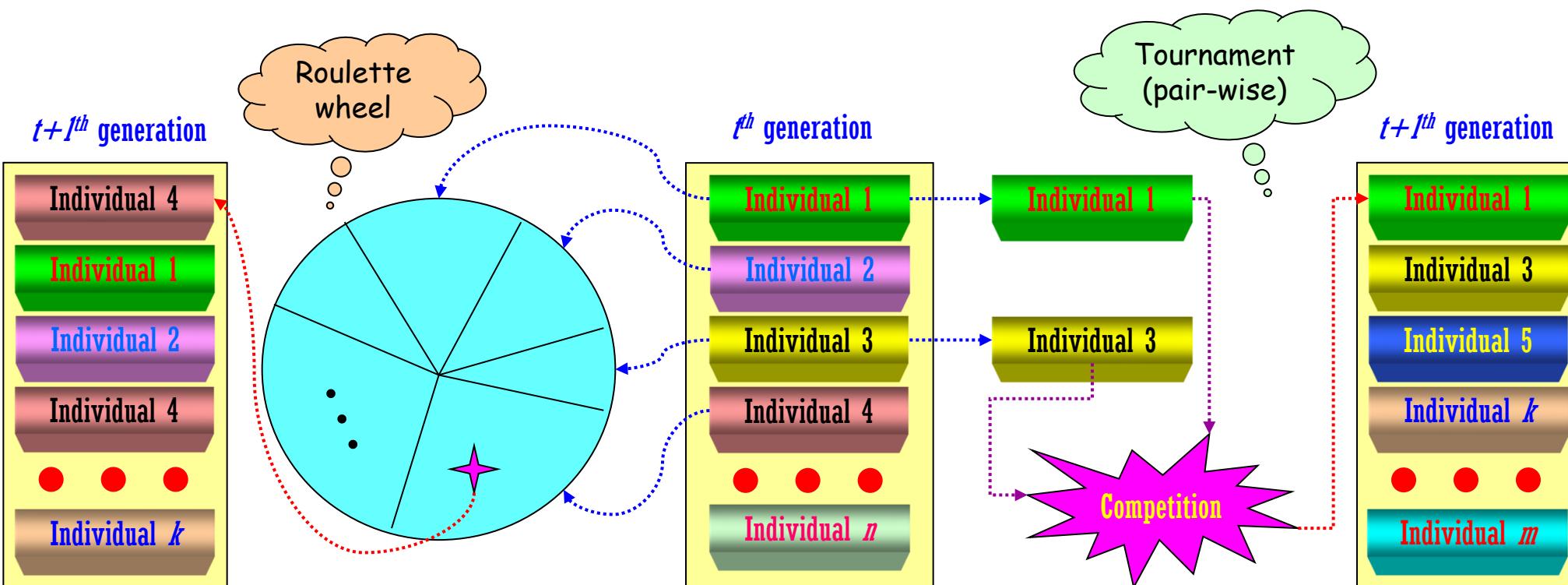


Genetic Algorithms



Genetic Algorithms

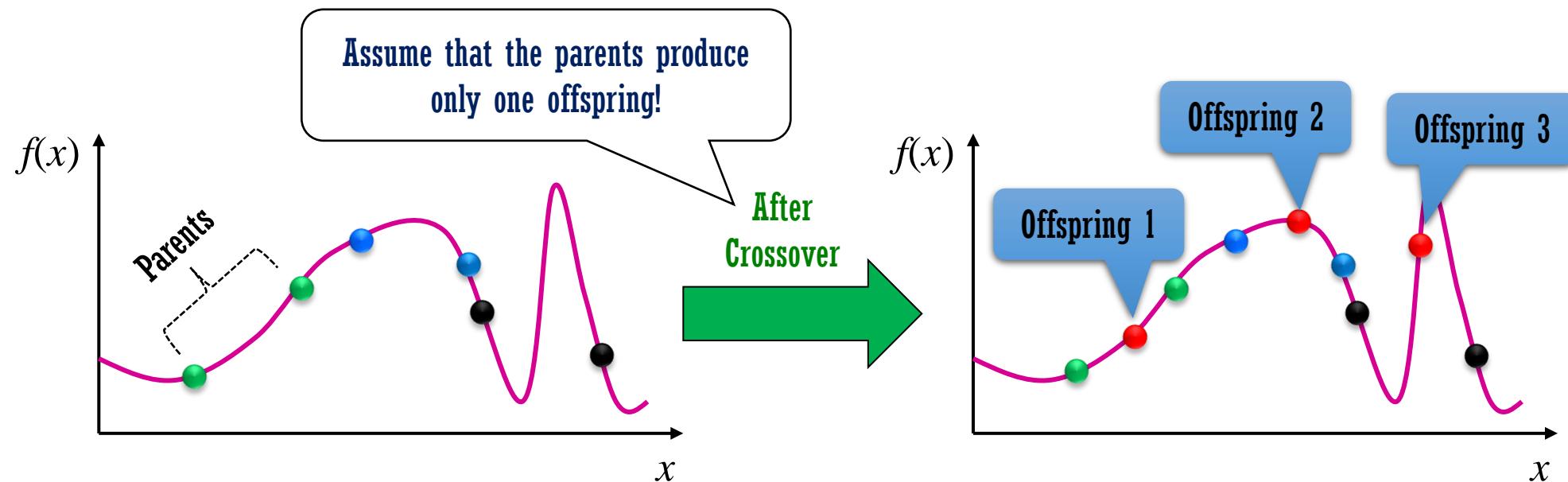
- **Roulette Wheel Selection**
 - ❖ The probability of selecting a given chromosome is proportional to its fitness
- **Tournament Selection**
 - ❖ Combine the fitness proportional concept with the random selection



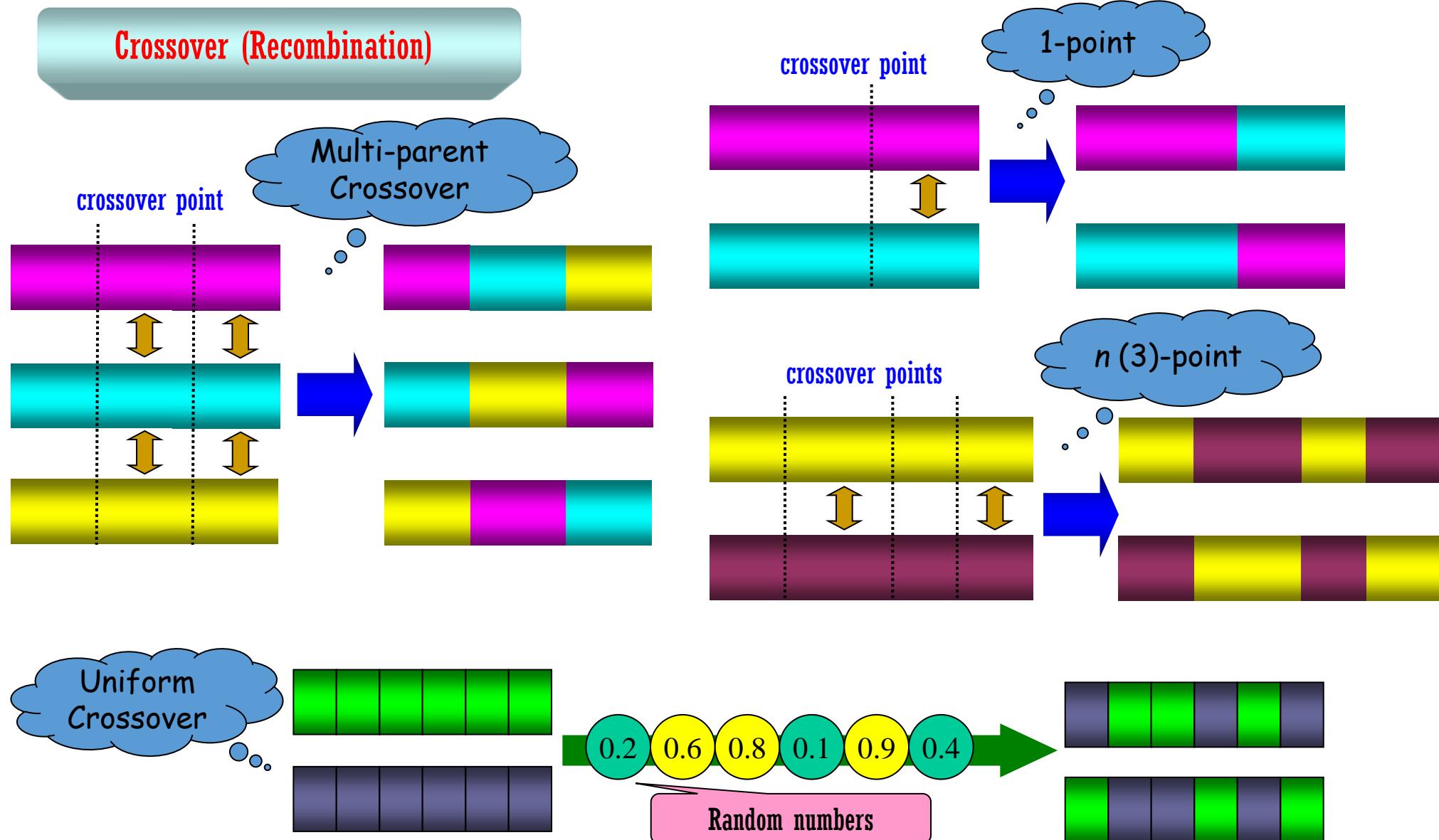
Genetic Algorithms

Crossover (Recombination)

1. Imitating the genetic inheritance (by recombining segments belonging to the individuals corresponding to parents)
2. One-point crossover, n -point crossover, Uniform crossover, etc.



Genetic Algorithms



Genetic Algorithms

Mutation

1. Realize the **self-variation** (mutation) of genetics
(by changing the value of the considered gene into a different value)
2. The **second way** of exploring search space
 - ❖ Its portion must be **very small**.
 - ❖ But, very crucial for **possibly escaping from local optima**

X-ary Case

mutation point

72X540X16

A number is randomly selected among {0, 1, 2, 3, 5, ..., X-1, X}

72X580X16

Binary Case

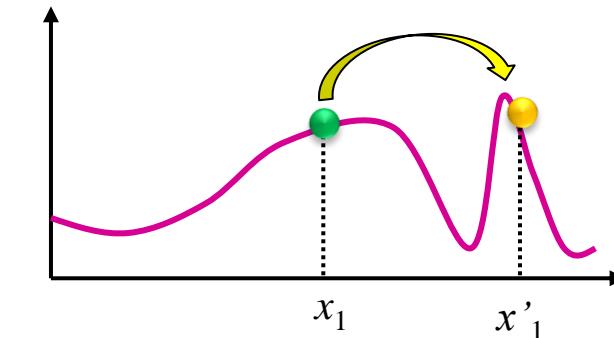
mutation point

1001001010

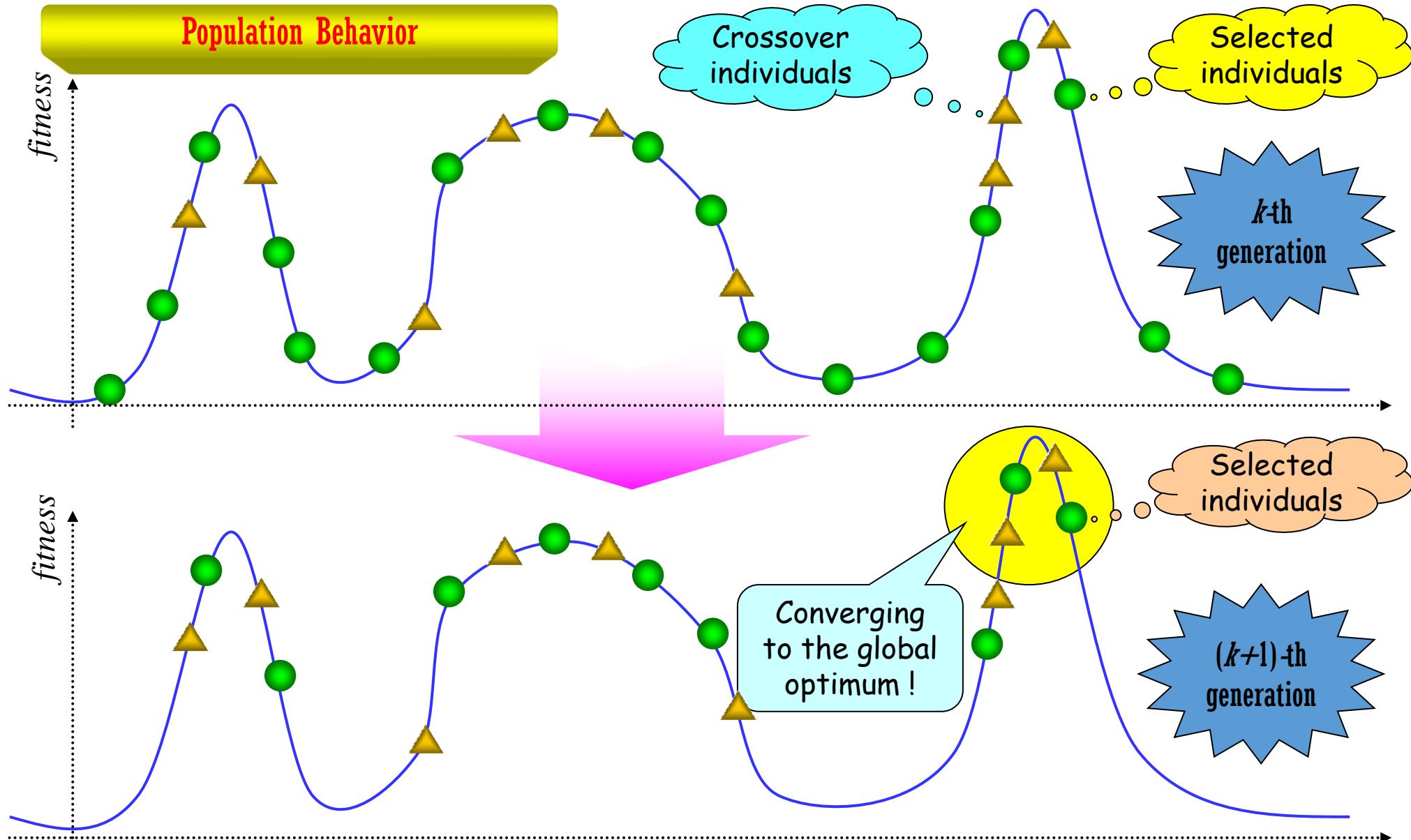
$x_1 = 586$

1101001010

$x'_1 = 842$

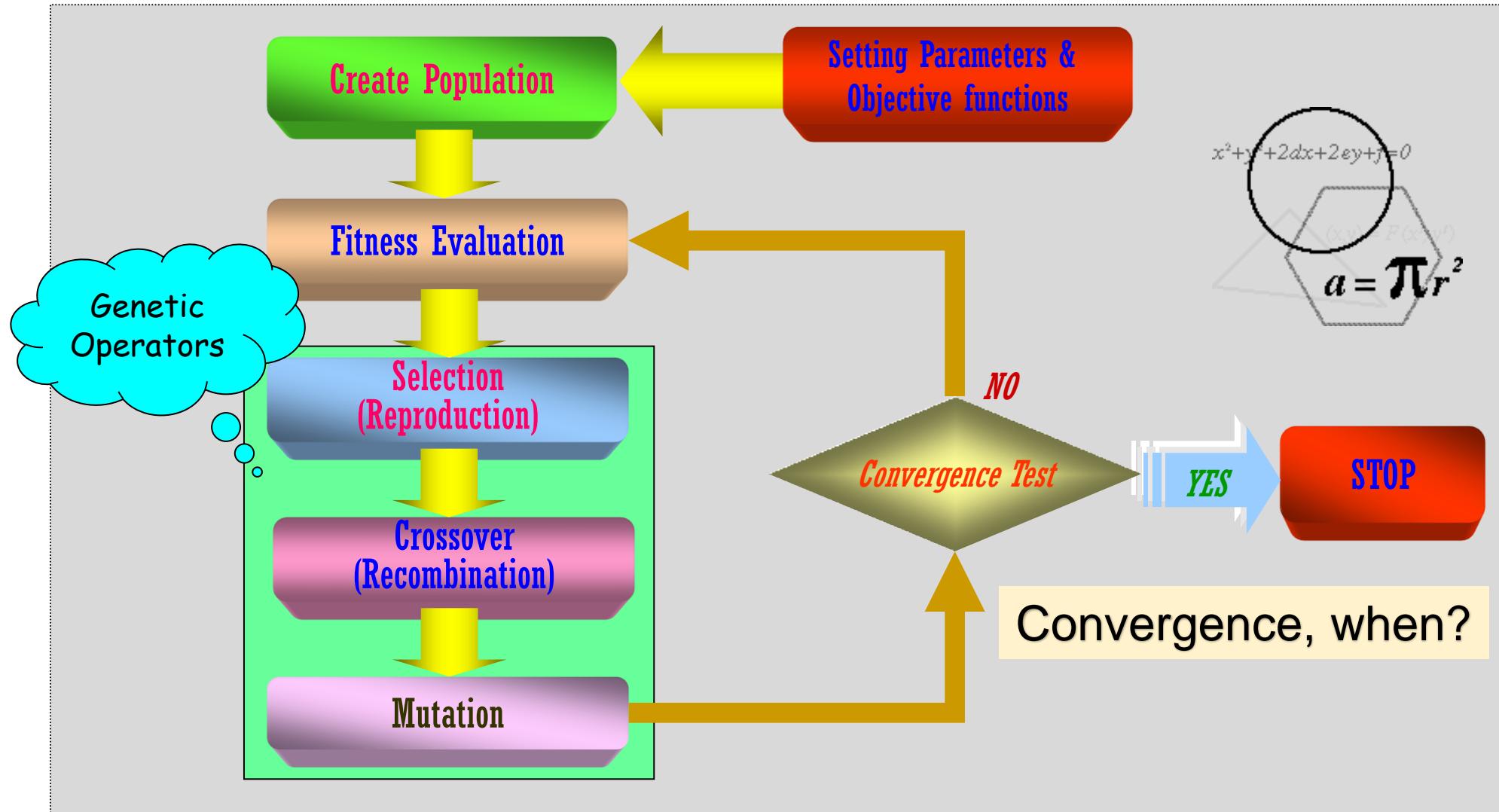


Genetic Algorithms



Genetic Algorithms

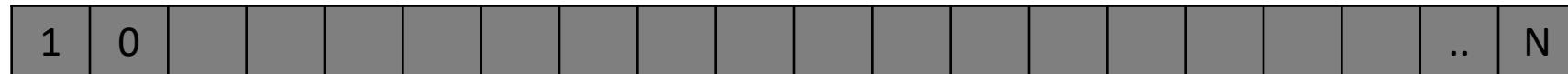
- Overall Procedures of GAs



Genetic Algorithm

❖ Introduction to one-max problem

What is the maximum sum of a bitstring (a string consisting of only 1s and 0s) of length N?



Very simple, the maximum sum of a bitstring of length N is N.

However, if you wanted to prove this using a brute-force search, you'd end up needing to search through ...?

Outline



➤ **Introduction to Evolutionary Algorithms**

➤ **Introduction to Genetic Algorithms**

➤ **One-Max Problem**

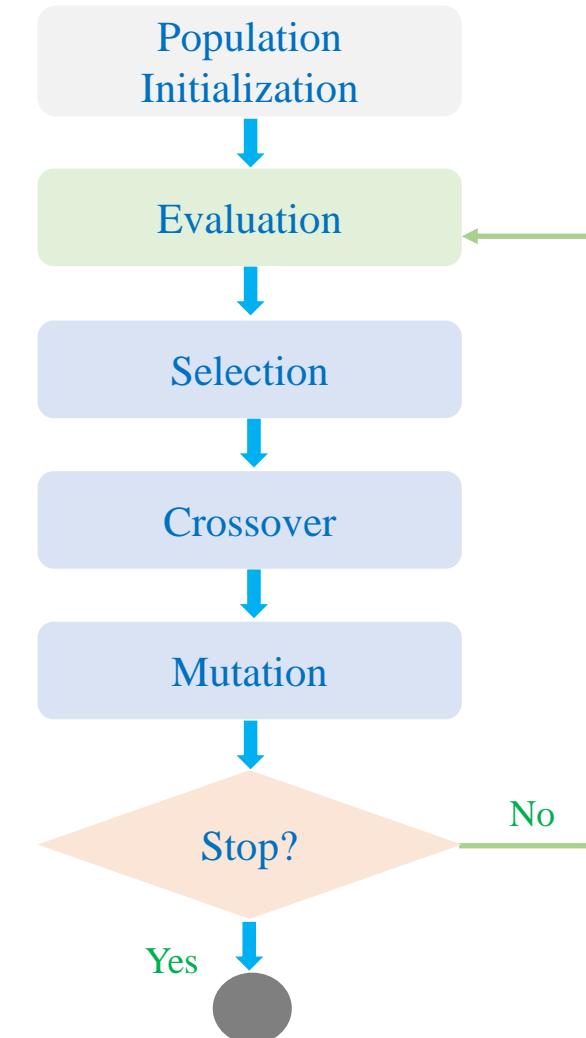
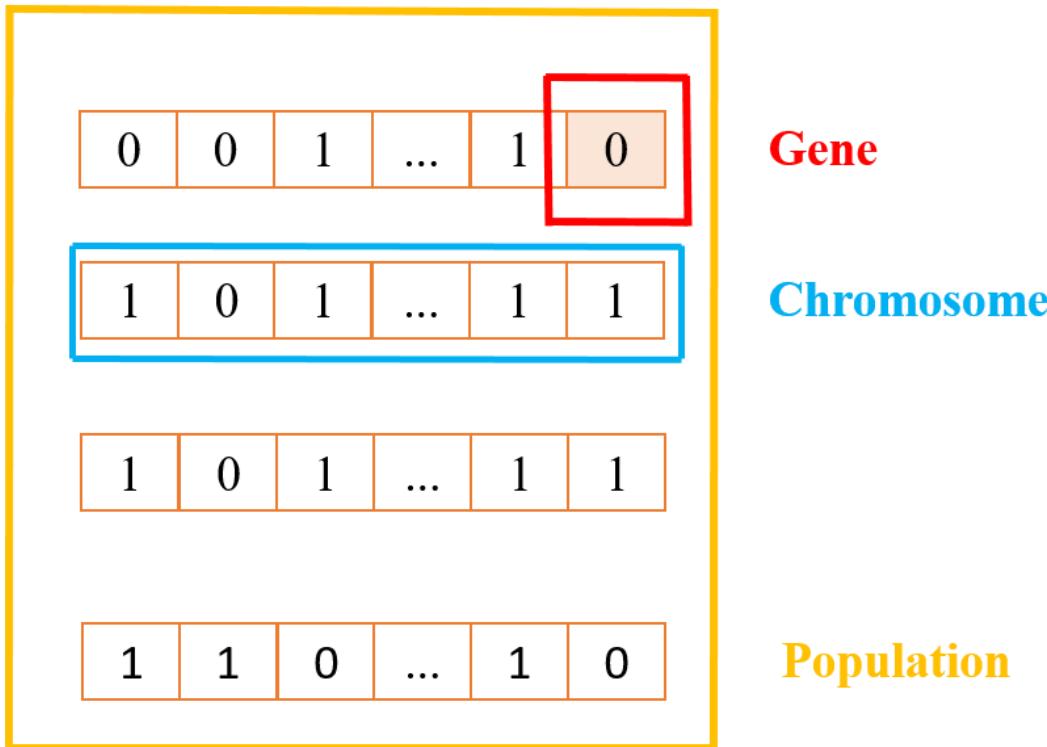
➤ **Sphere Function**

➤ **Regression Problem**

➤ **GA Applications**

Genetic Algorithm

❖ To solve the one-max problem



Genetic Algorithm

❖ Steps in Genetic Algorithm

Population Initialization

Evaluation

Selection

Crossover

Mutation

Stop?

Yes

No

Khởi tạo quần thể: Gán giá trị ngẫu nhiên cho các gen của các cá thể

Cá thể 1 0 0 1 1 1 0

Cá thể 2 1 0 1 0 1 0

Cá thể 3 0 0 1 0 0 1

Cá thể 4 0 1 1 0 0 1

Cá thể 5 0 1 1 1 1 0

Cá thể 6 1 0 0 0 1 1

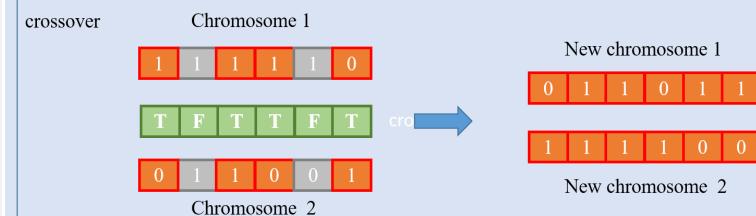
Cá thể 7 0 0 0 0 0 1

Cá thể 8 1 0 0 0 0 0

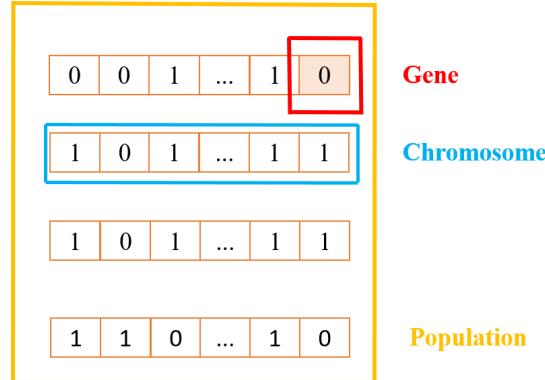
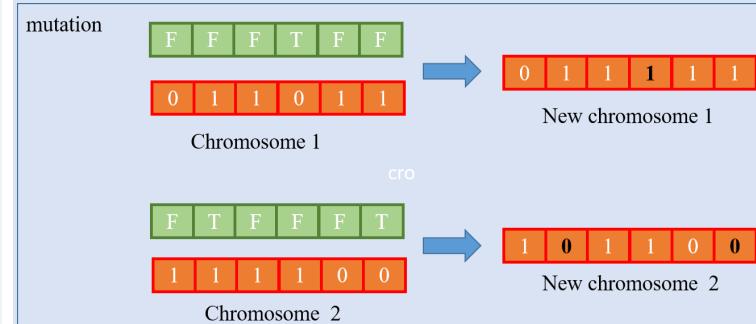
Cá thể 9 0 1 0 0 0 1

Cá thể 10 1 1 1 1 1 0

Trao đổi gen (giá trị) giữa các cá thể với nhau



Đột biến gen cho từng cá thể



Tính fitness cho các cá thể

	Fitness
Cá thể 1	0 0 1 1 1 0
Cá thể 2	1 0 1 0 1 0
Cá thể 3	0 0 1 0 0 1
Cá thể 4	0 1 1 0 0 1
Cá thể 5	0 1 1 1 1 0
Cá thể 6	1 0 0 0 1 1
Cá thể 7	0 0 0 0 0 1
Cá thể 8	1 0 0 0 0 0
Cá thể 9	0 1 0 0 0 1
Cá thể 10	1 1 1 1 1 0

Genetic Algorithm

❖ Population initialization

Cá thể 1	0	0	1	1	1	0
Cá thể 2	1	0	1	0	1	0
Cá thể 3	0	0	1	0	0	1
Cá thể 4	0	1	1	0	0	1
Cá thể 5	0	1	1	1	1	0
Cá thể 6	1	0	0	0	1	1
Cá thể 7	0	0	0	0	0	1
Cá thể 8	1	0	0	0	0	0
Cá thể 9	0	1	0	0	0	1
Cá thể 10	1	1	1	1	1	0

Population size m = 10

Vector length is with n = 6

```

1 # aivietnam.ai
2 import random
3
4 n = 6 # size of individual (chromosome)
5 m = 10 # size of population
6
7 def generate_random_value():
8     return random.randint(0, 1)
9
10 def create_individual():
11     return [generate_random_value() for _ in range(n)]
12
13 population = [create_individual() for _ in range(m)]
14
15 # print population
16 for ind in population:
17     print(ind)

```

```

[0, 0, 0, 1, 0, 1]
[0, 0, 0, 0, 1, 1]
[0, 1, 0, 1, 0, 1]
[1, 0, 1, 1, 0, 1]
[1, 0, 0, 1, 0, 1]
[1, 1, 0, 1, 0, 0]
[0, 0, 1, 1, 0, 0]
[1, 1, 1, 1, 1, 1]
[1, 0, 0, 0, 0, 1]
[0, 0, 1, 1, 0, 1]

```

Genetic Algorithm

❖ Evaluation

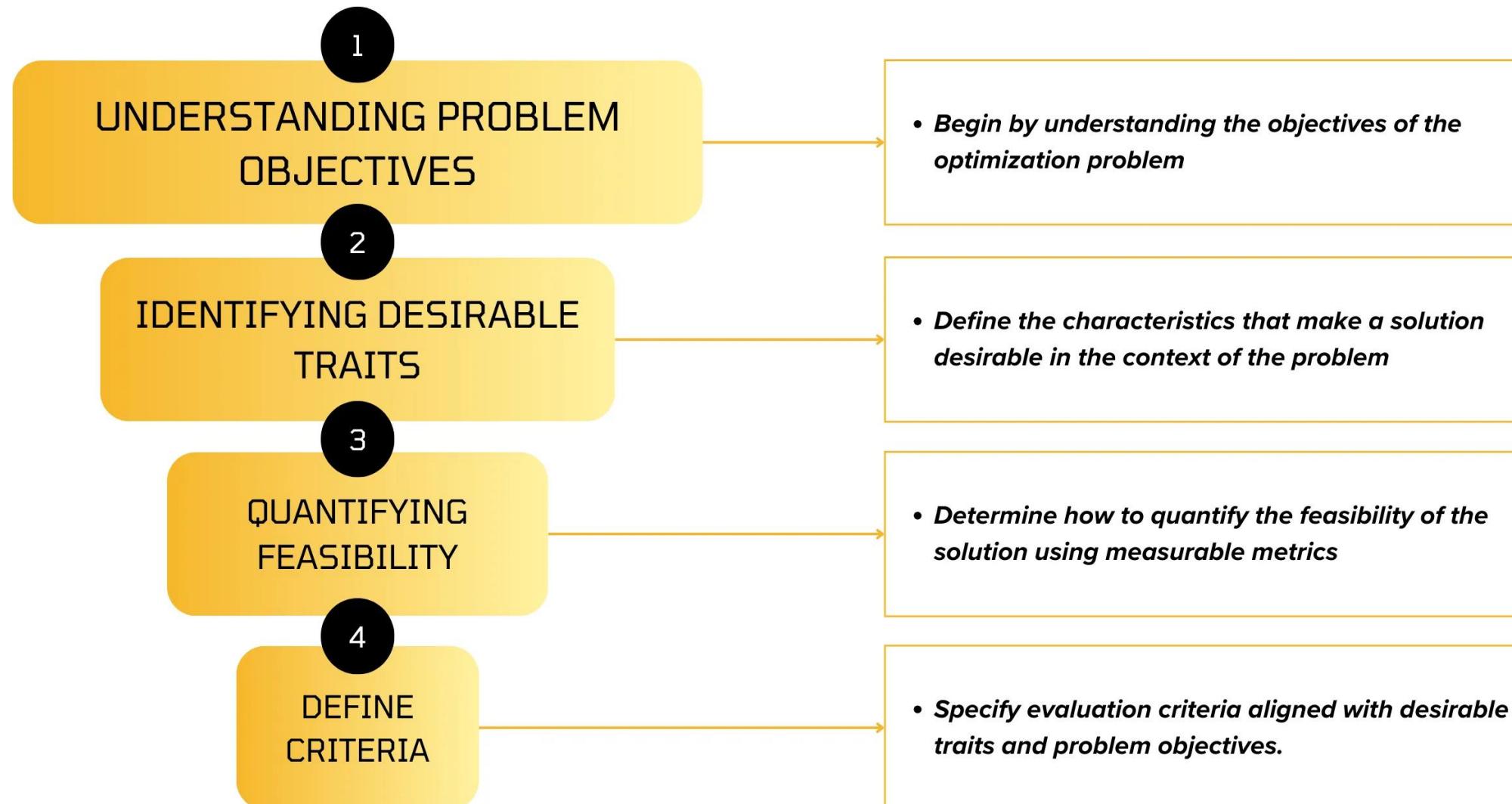
- ❖ This function is given

Secret information

$$\text{secret}(v) = \sum_i v_i$$

	Fitness
Cá thể 1	3
Cá thể 2	3
Cá thể 3	2
Cá thể 4	3
Cá thể 5	4
Cá thể 6	3
Cá thể 7	1
Cá thể 8	1
Cá thể 9	2
Cá thể 10	5

How to design Fitness Function



Genetic Algorithm

❖ Selection

Index	Sorted population	Fitness		
0	0 0 0 0 0 1	1		
1	1 0 0 0 0 0	1		
2	0 0 1 0 0 1	2		
3	0 1 0 0 0 1	2		
4	1 0 0 0 1 1	3		
5	0 1 1 0 0 1	3		
6	0 0 1 1 1 0	3		
7	1 0 1 0 1 0	3		
8	0 1 1 1 1 0	4		
9	1 1 1 1 1 0	5		
			9	1 1 1 1 1 0 5
			5	0 1 1 0 0 1 3



Genetic Algorithm

❖ Selection

Index	Sorted population	Fitness	Old Index	New population	Fitness
0	0 0 0 0 0 1	1	9	1 1 1 1 1 0	5
1	1 0 0 0 0 0	1	4	1 0 0 0 1 1	3
2	0 0 1 0 0 1	2	8	0 1 1 1 1 0	4
3	0 1 0 0 0 1	2	5	0 1 1 0 0 1	3
4	1 0 0 0 1 1	3	5	0 1 1 0 0 1	3
5	0 1 1 0 0 1	3	4	1 0 0 0 1 1	3
6	0 0 1 1 1 0	3	6	0 0 1 1 1 0	3
7	1 0 1 0 1 0	3	6	0 0 1 1 1 0	3
8	0 1 1 1 1 0	4	8	0 1 1 1 1 0	4
9	1 1 1 1 1 0	5	8	0 1 1 1 1 0	4

Genetic Algorithm

❖ Selection

Index	Sorted population	Fitness	Old Index	New population	Fitness
0	0 0 0 0 0 0 1	1	9	1 1 1 1 1 0	5
1	1 0 0 0 0 0 0	1	4	1 0 0 0 1 1	3
2	0 0 1 0 0 0 1	2	8	0 1 1 1 1 0	4
3	0 1 0 0 0 0 1	2	5	0 1 1 0 0 1	3
4	1 0 0 0 1 1	3	5	0 1 1 0 0 1	3
5	0 1 1 0 0 0 1	3	4	1 0 0 0 1 1	3
6	0 0 1 1 1 0	3	6	0 0 1 1 1 0	3
7	1 0 1 0 1 0	3	6	0 0 1 1 1 0	3
8	0 1 1 1 1 0	4	8	0 1 1 1 1 0	4
9	1 1 1 1 1 0	5	8	0 1 1 1 1 0	4



```

48 def selection(sorted_old_population):
49     index1 = random.randint(0, m-1)
50     while True:
51         index2 = random.randint(0, m-1)
52         if (index2 != index1):
53             break
54
55     individual_s = sorted_old_population[index1]
56     if index2 > index1:
57         individual_s = sorted_old_population[index2]
58
59     return individual_s

```

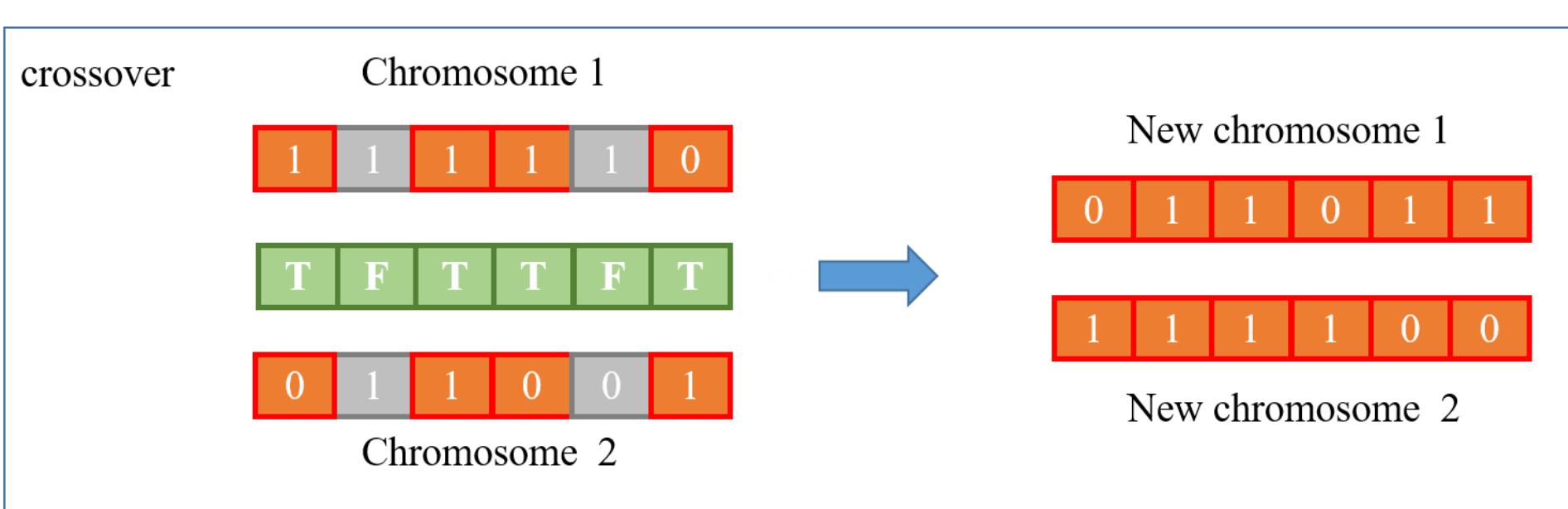
Genetic Algorithm

- ❖ Crossover
- ❖ Binary crossover

```

28 def crossover(individual1, individual2, crossover_rate = 0.9):
29     individual1_new = individual1.copy()
30     individual2_new = individual2.copy()
31
32     for i in range(n):
33         if random.random() < crossover_rate:
34             individual1_new[i] = individual2[i]
35             individual2_new[i] = individual1[i]
36
37     return individual1_new, individual2_new
38

```



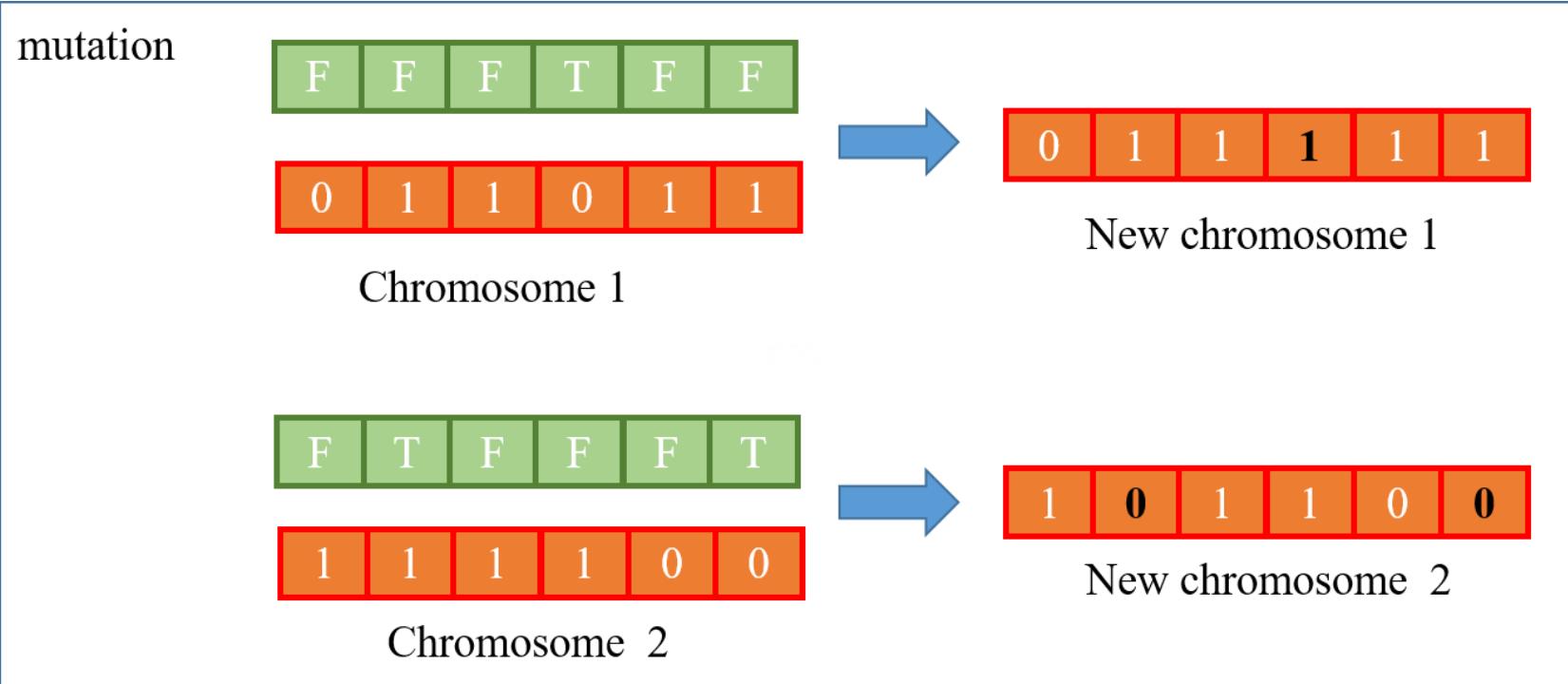
Genetic Algorithm

❖ Mutation

```

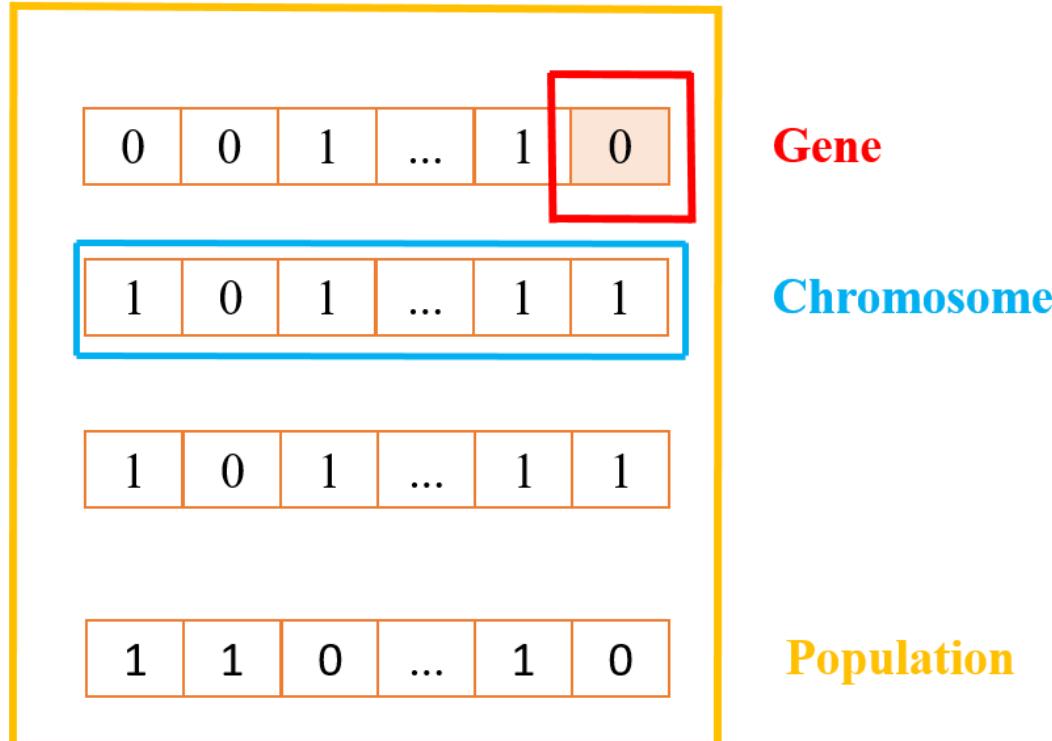
39 def mutate(individual, mutation_rate = 0.05):
40     individual_m = individual.copy()
41
42     for i in range(n):
43         if random.random() < mutation_rate:
44             individual_m[i] = generate_random_value()
45
46     return individual_m

```

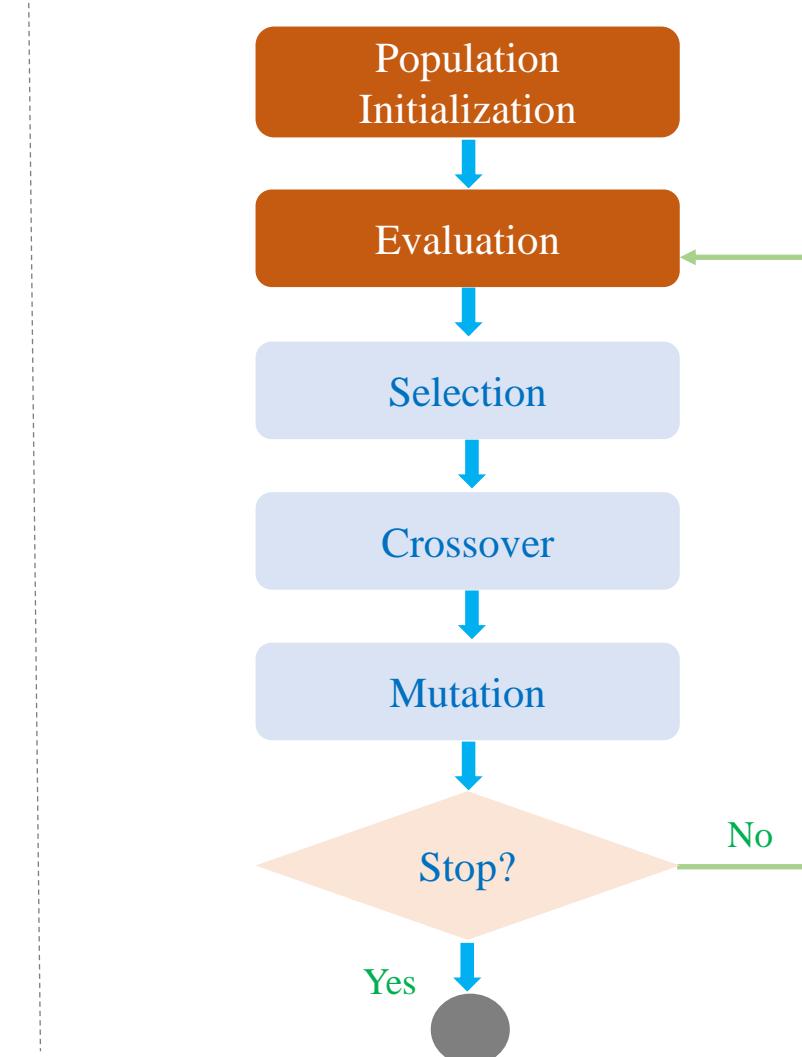


Genetic Algorithm

❖ To solve the one-max problem



Can you find any limitation?



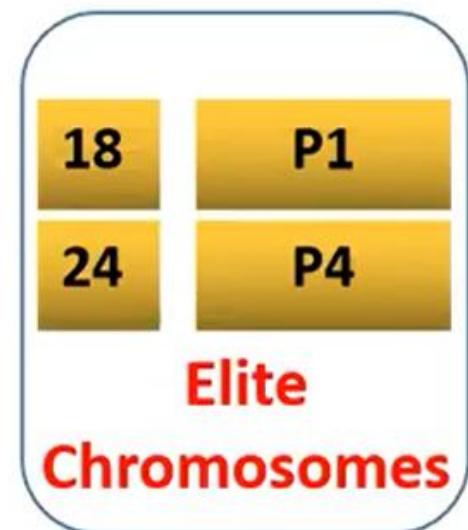
Example of Elitism

Fitness Chromosome

18	P1
06	P2
10	P3
24	P4
05	P5
12	P6

Fitness Chromosome

18	P1'
24	P2'
15	P3'
22	P4'
10	P5'
08	P6'



Current Generation

New Generation

Outline

➤ **Introduction to Evolutionary Algorithms**

➤ **Introduction to Genetic Algorithms**

➤ **One-Max Problem**

➤ **Sphere Function**

➤ **Regression Problem**

➤ **GA Applications**



Genetic Algorithm

❖ Sphere function

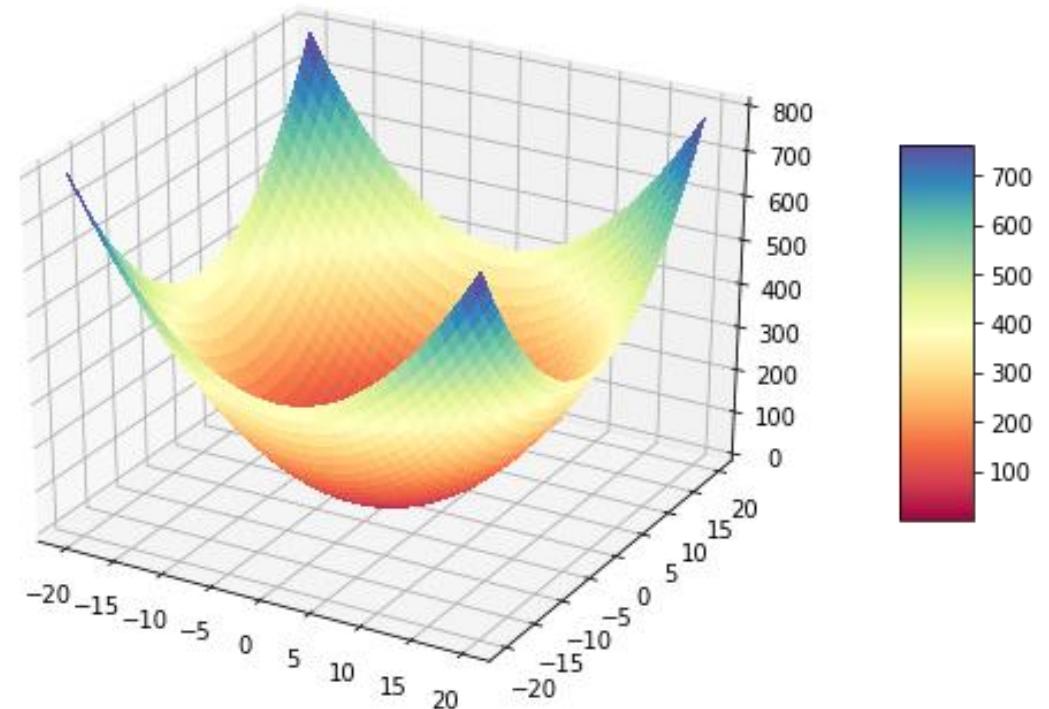
$$f(x) = x_1^2 + x_2^2$$

Các biến x_i có kiểu số tự nhiên và $x_i \in \mathbb{N}$.

Gen của chromosome được biểu diễn bằng kiểu integer.

Hàm $f(x)$ có 2 biến. Do đó, độ dài của chromosome là 2 ($n = 2$).

fitness và accuracy
loss, error, và cost

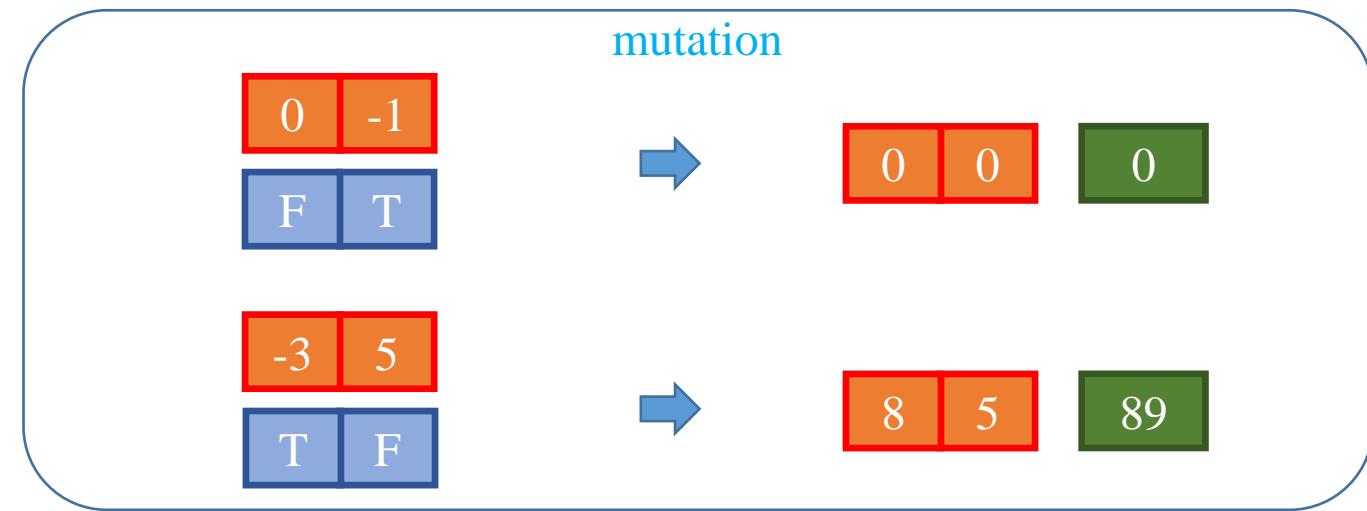
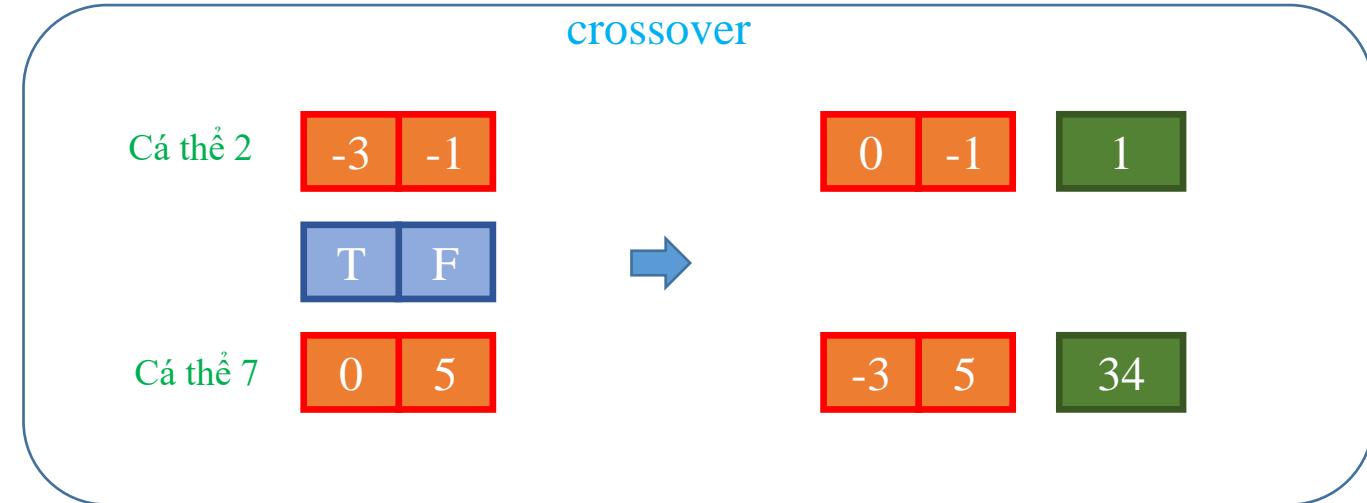


Genetic Algorithm

	Cost			Cost			Cost		
Cá thể	1	2	Cá thể	1	2	Cá thể	1	2	
Cá thể 1	5	-2	29	1	-3	10	5	-2	29
Cá thể 2	-6	4	50	-3	-1	10	-3	-1	10
Cá thể 3	1	6	37	2	3	13	1	6	37
Cá thể 4	-3	-1	10	0	5	25	-3	-1	10
Cá thể 5	8	6	100	5	-2	29	1	-3	10
Cá thể 6	7	-2	53	1	6	37	-6	4	50
Cá thể 7	0	5	25	-6	4	50	0	5	25
Cá thể 8	9	-1	82	7	-2	53	0	5	25
Cá thể 9	2	3	13	9	-1	82	2	3	13
Cá thể 10	1	-3	10	8	6	100	1	-3	10

Genetic Algorithm

	Cost	
Cá thể 1	5	-2
Cá thể 2	-3	-1
Cá thể 3	1	6
Cá thể 4	-3	-1
Cá thể 5	1	-3
Cá thể 6	-6	4
Cá thể 7	0	5
Cá thể 8	0	5
Cá thể 9	2	3
Cá thể 10	1	-3
	29	10
	37	10
	10	34
	50	0
	25	0
	25	0
	13	89
	10	0



Genetic Algorithm

❖ Sphere function

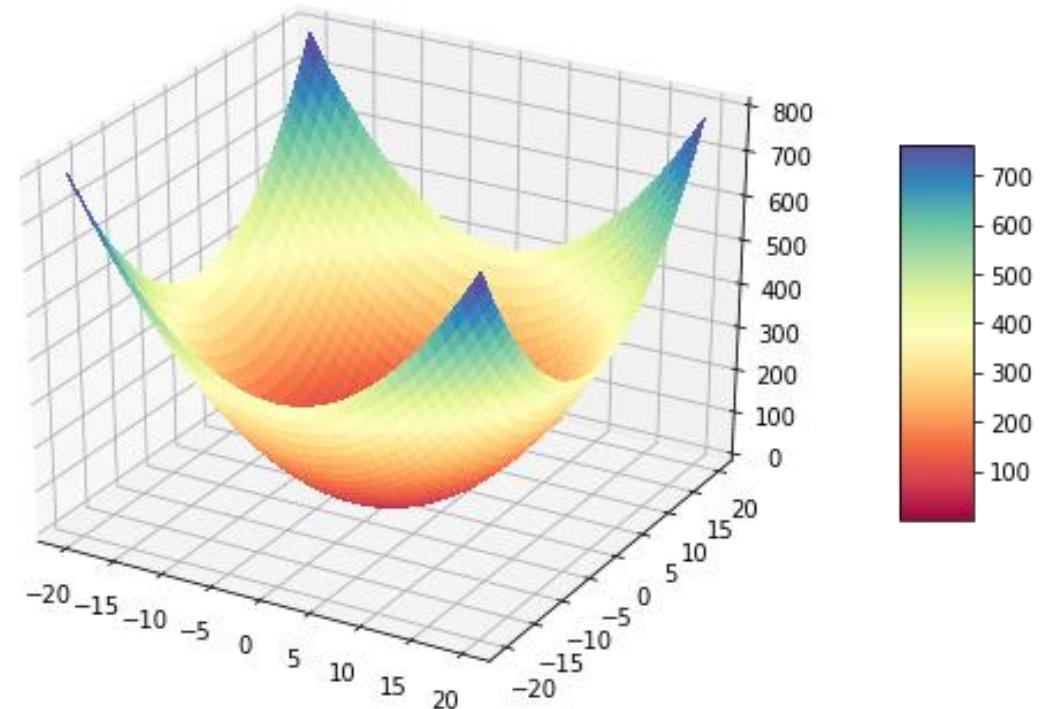
$$f(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$$

Do các biến x_i có kiểu số thực, nên số điểm trong không gian tìm kiếm là vô hạn.

Gen của chromosome được biểu diễn bằng kiểu floating-point.

Hàm $f(x)$ có 6 biến. Do đó, độ dài của chromosome là 6 ($n = 6$).

fitness và accuracy
loss, error, và cost



Genetic Algorithm

❖ Sphere function

generate_random_value()

 feasible values in [-20, 20]

```
def generate_random_value(bound = 20):
    return (random.random()*2 - 1)*bound
```

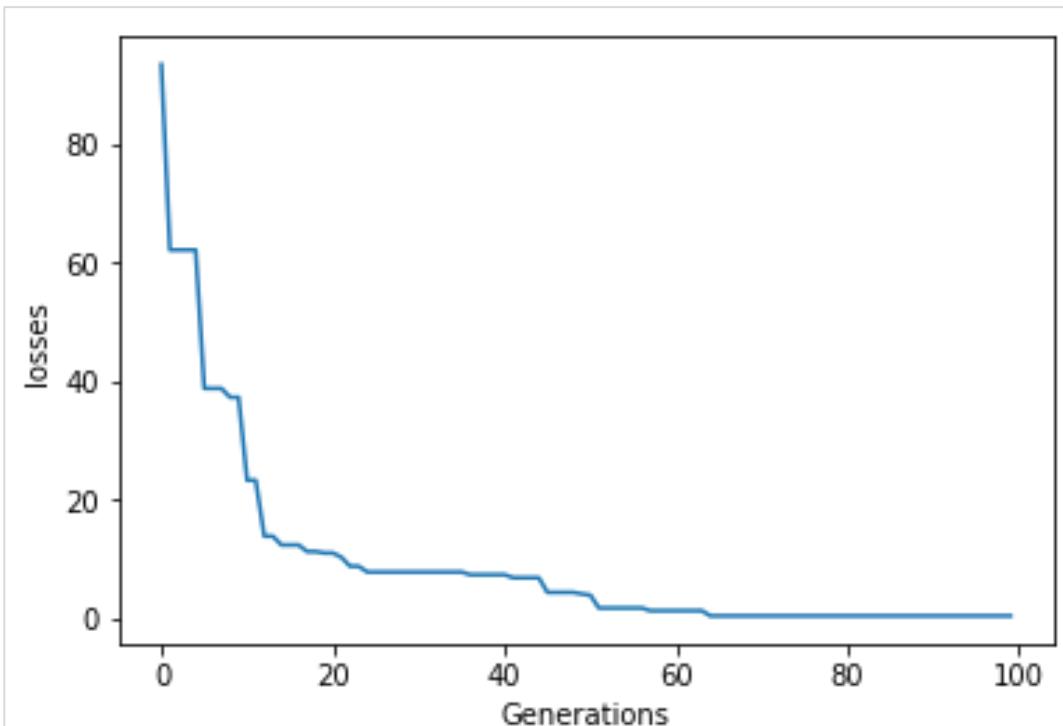
compute_fitness()

```
def compute_loss(individual):
    return sum(gen*gen for gen in individual)

def compute_fitness(individual):
    loss = compute_loss(individual)
    fitness = 1 / (loss + 1)
    return fitness
```

```
def compute_loss(individual):
    return sum(gen*gen for gen in individual)

def compute_fitness(individual):
    loss = compute_loss(individual)
    fitness = 1 / (loss + 1)
    return fitness
```



Outline

➤ **Introduction to Evolutionary Algorithms**

➤ **Introduction to Genetic Algorithms**

➤ **One-Max Problem**

➤ **Sphere Function**

➤ **Regression Problem**

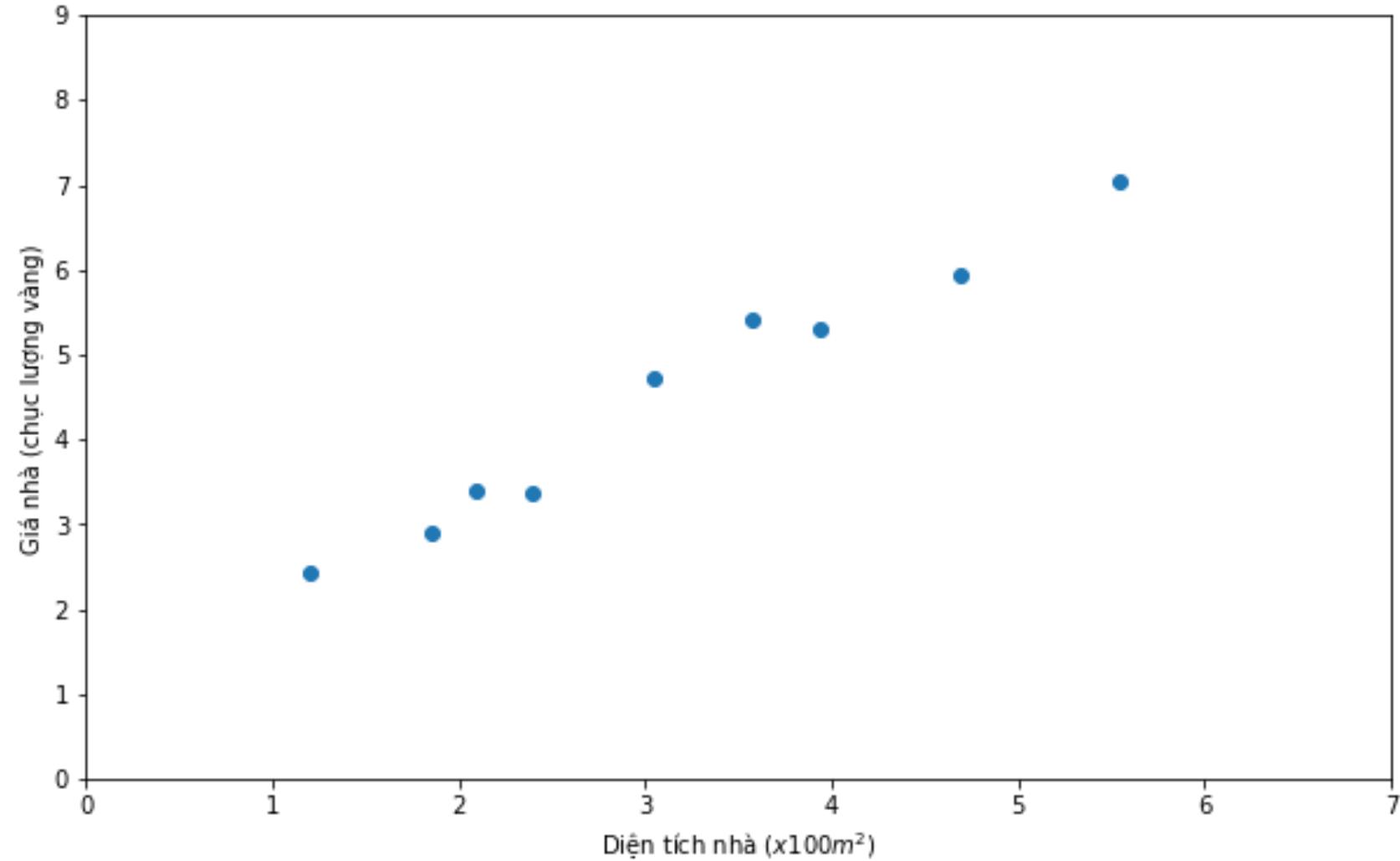
➤ **GA Applications**



Genetic Algorithm

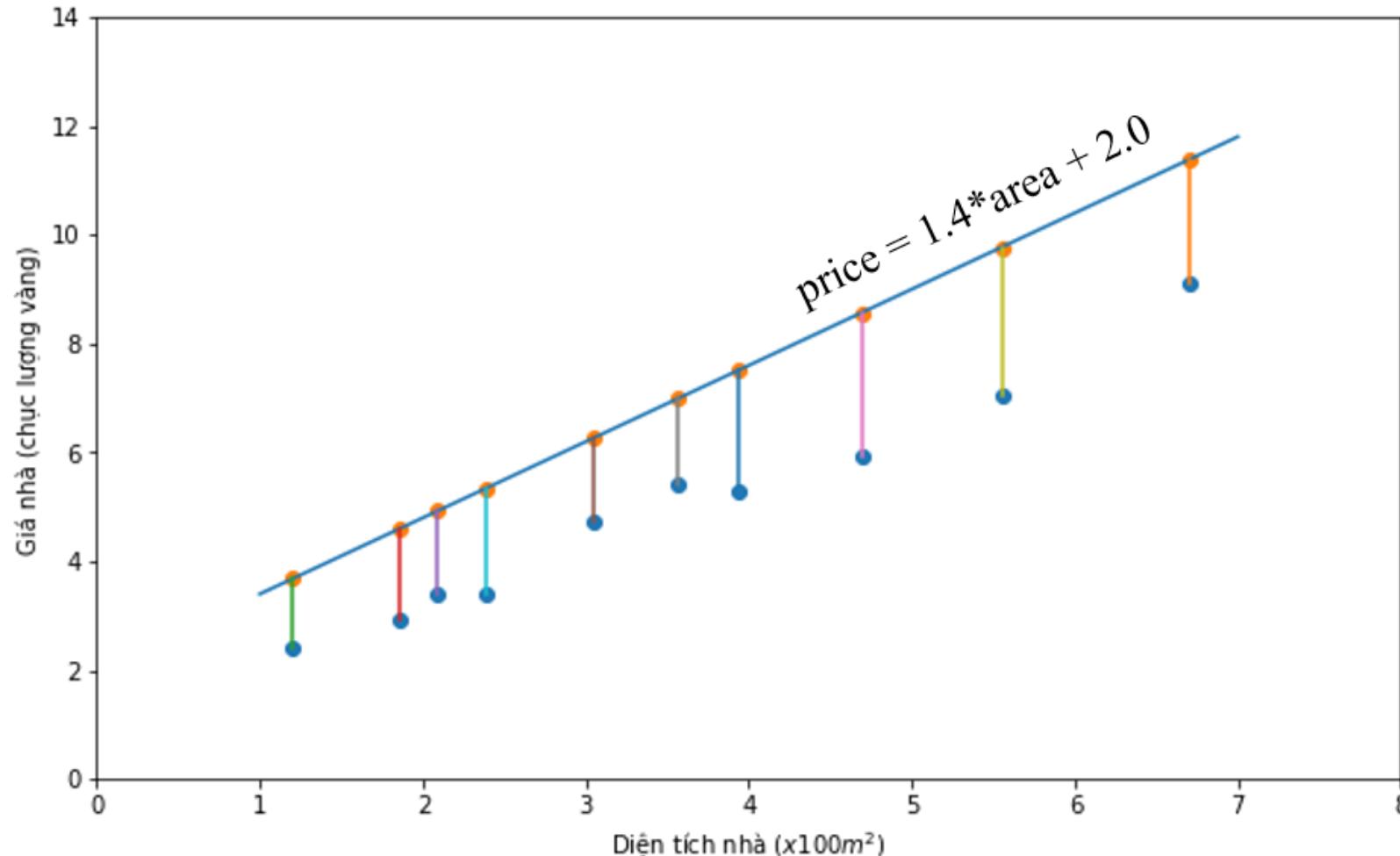
❖ House price prediction

area	price
6.71	9.12
1.2	2.43
1.86	2.91
2.09	3.41
3.05	4.71
4.69	5.94
3.57	5.4
5.55	7.04
2.39	3.38
3.94	5.29



Genetic Algorithm

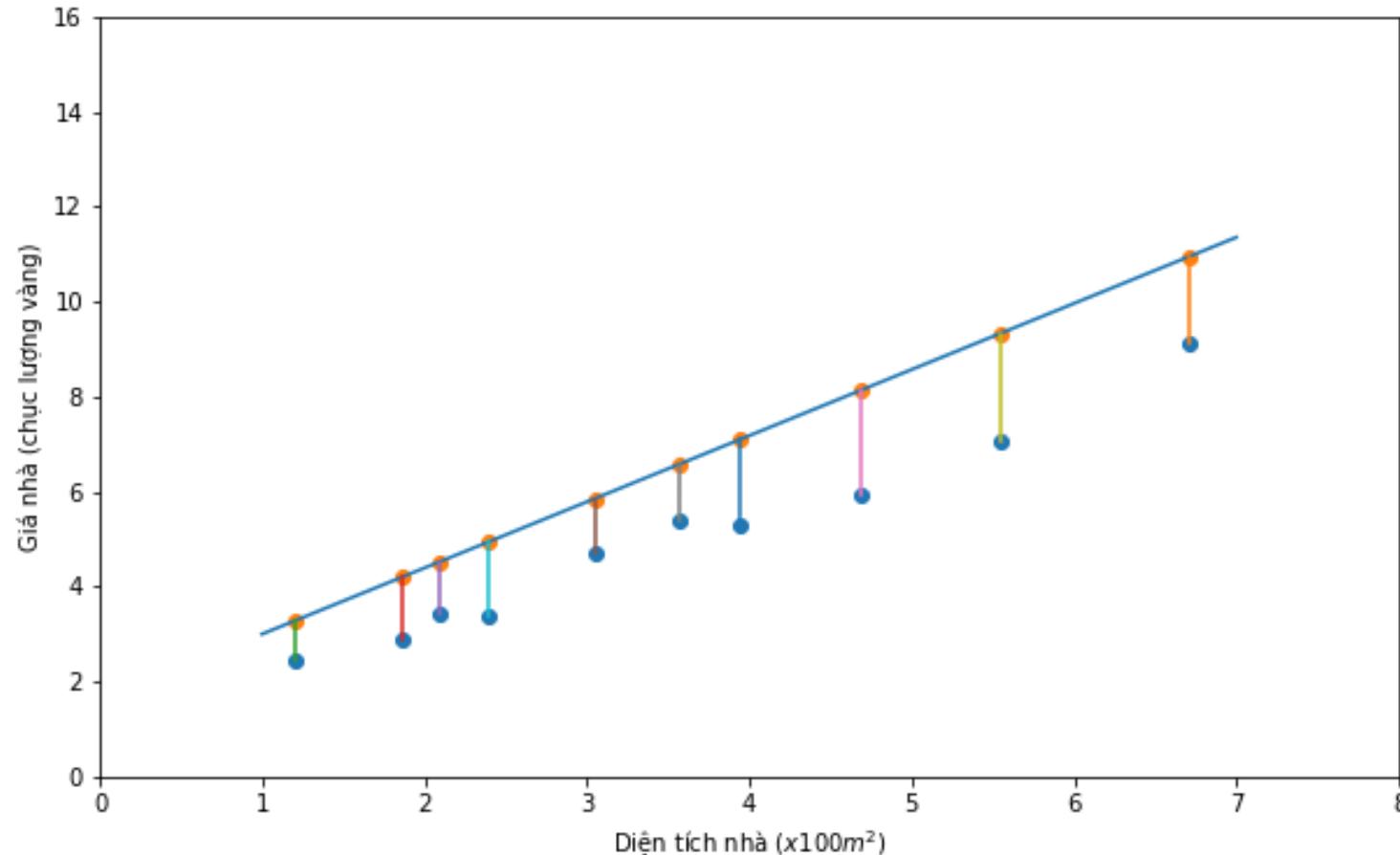
❖ House price prediction



Genetic Algorithm

❖ House price prediction

$$price = a * area + b$$



Genetic Algorithm

❖ House price prediction

area	price
6.71	9.12
1.2	2.43
1.86	2.91
2.09	3.41
3.05	4.71
4.69	5.94
3.57	5.4
5.55	7.04
2.39	3.38
3.94	5.29

```
# Hàm load data
def load_data():
    # kết nối với file
    file = open('data.csv', 'r')

    # readlines giúp việc đọc file theo từng dòng , mỗi dòng là 1 chuỗi
    lines = file.readlines()

    areas  = []
    prices = []
    for i in range(10):
        string = lines[i].split(',')
        areas.append(float(string[0]))
        prices.append(float(string[1]))

    # Đóng kết nối với file
    file.close()

    return areas, prices
```

Genetic Algorithm

❖ House price prediction

```
def compute_loss(individual):
    result = 65534

    a = individual[0]
    b = individual[1]
    estimated_prices = [a*x + b for x in areas]

    # all prices should be positive numbers
    num_negetive_prices = sum(p < 0 for p in estimated_prices)
    if num_negetive_prices == 0:
        losses = [abs(y_est-y_gt) for y_est, y_gt in zip(estimated_prices, prices)]
        result = sum(losses)

    return result
```

Discussion

- ❖ Mutation
- ❖ Crossover
- ❖ Gene

Outline

➤ **Introduction to Evolutionary Algorithms**

➤ **Introduction to Genetic Algorithms**

➤ **One-Max Problem**

➤ **Sphere Function**

➤ **Regression Problem**

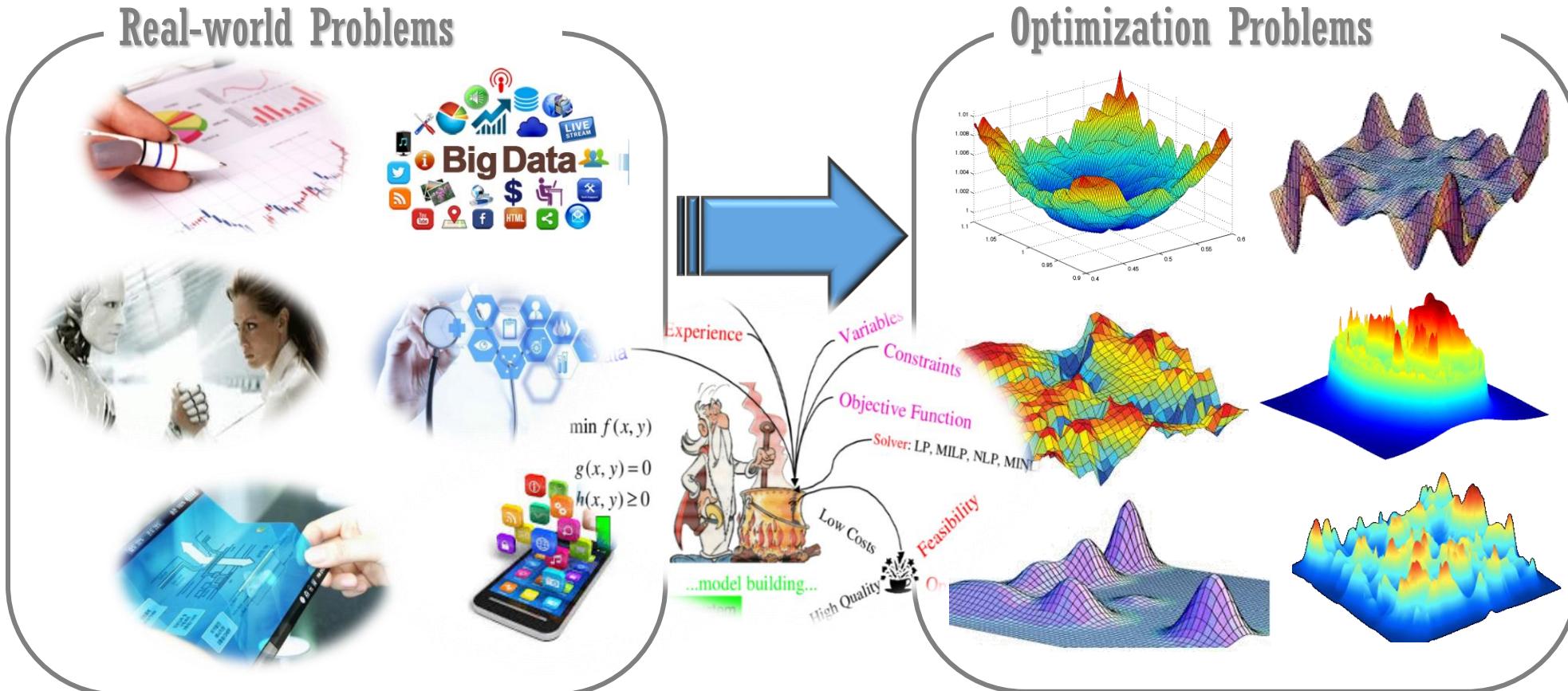
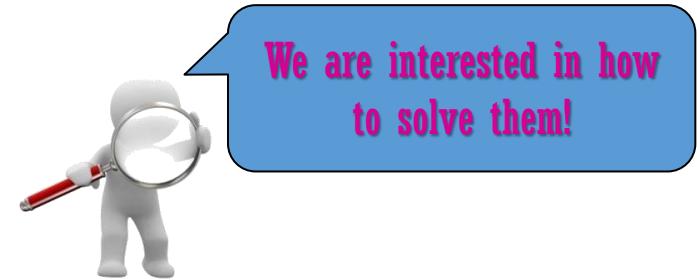
➤ **GA Applications**



Applications

What is the Target of Interest?

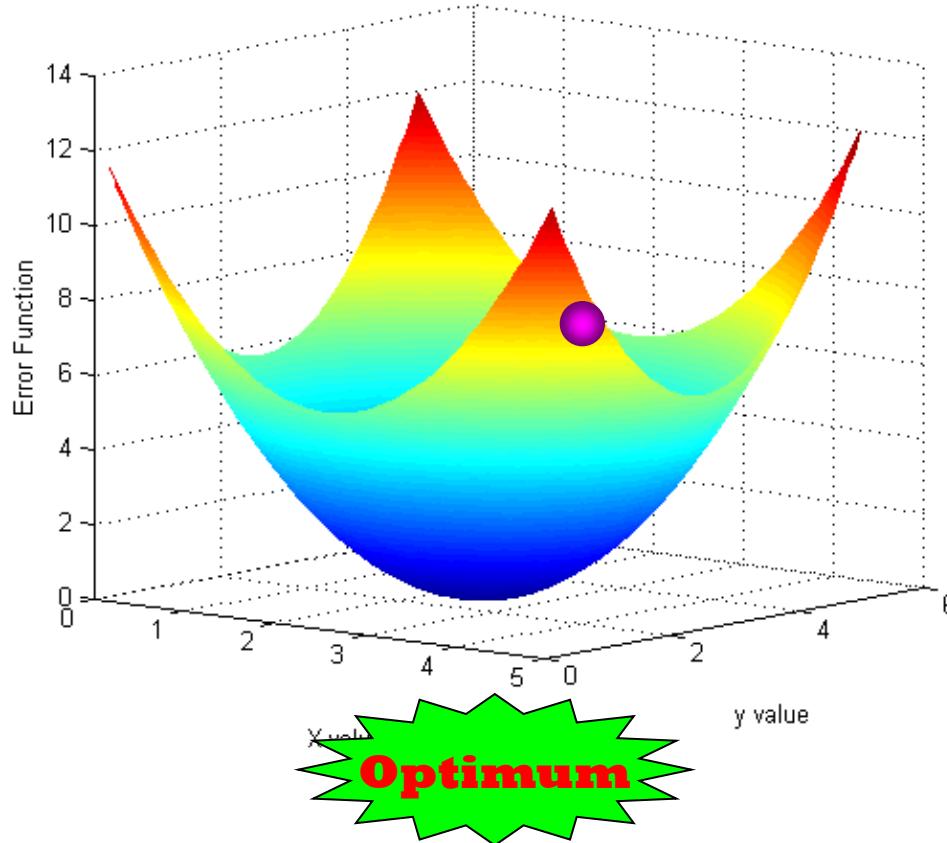
- All the Problems in Any Field!
- That is, a sort of Optimization Problem!



Applications

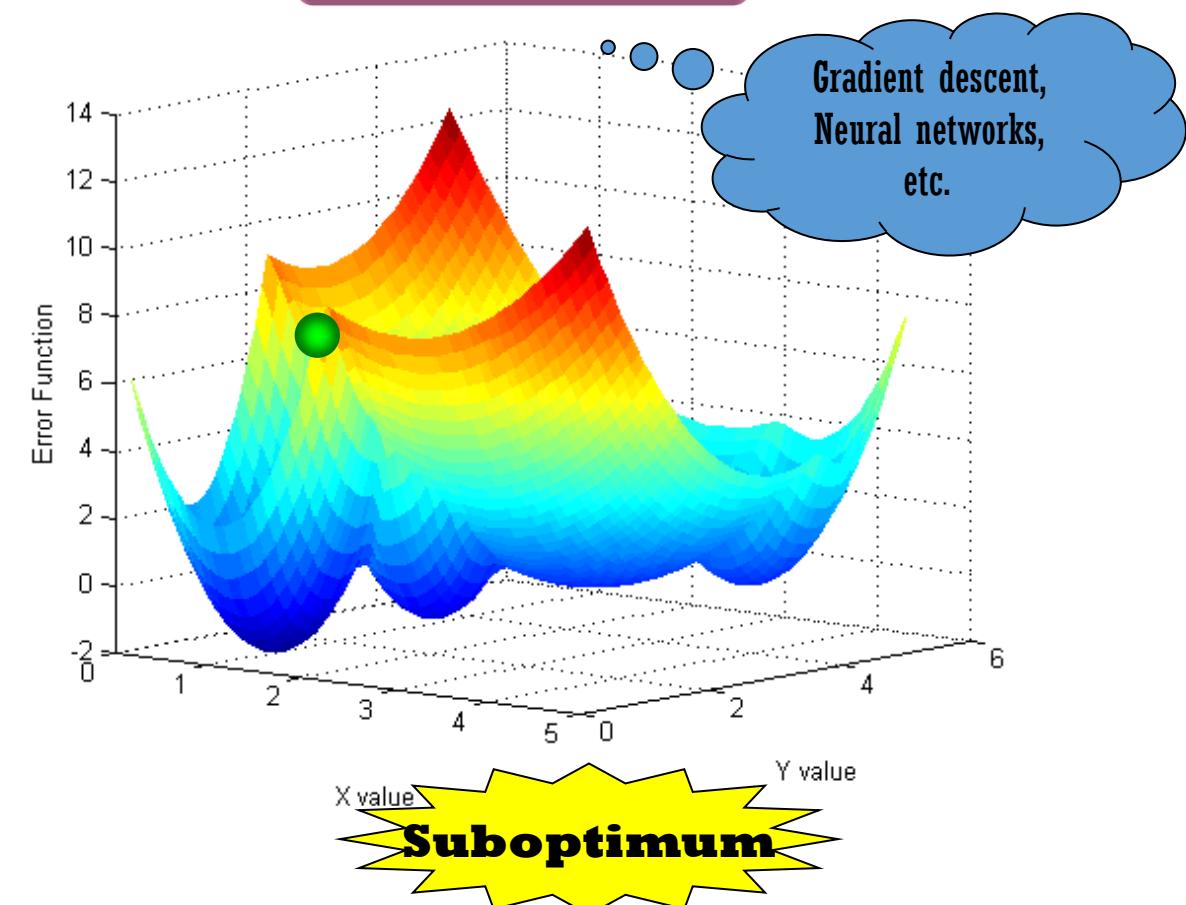
What's the Problem of Existing (Deterministic) Methods?

Single modal case



Optimum

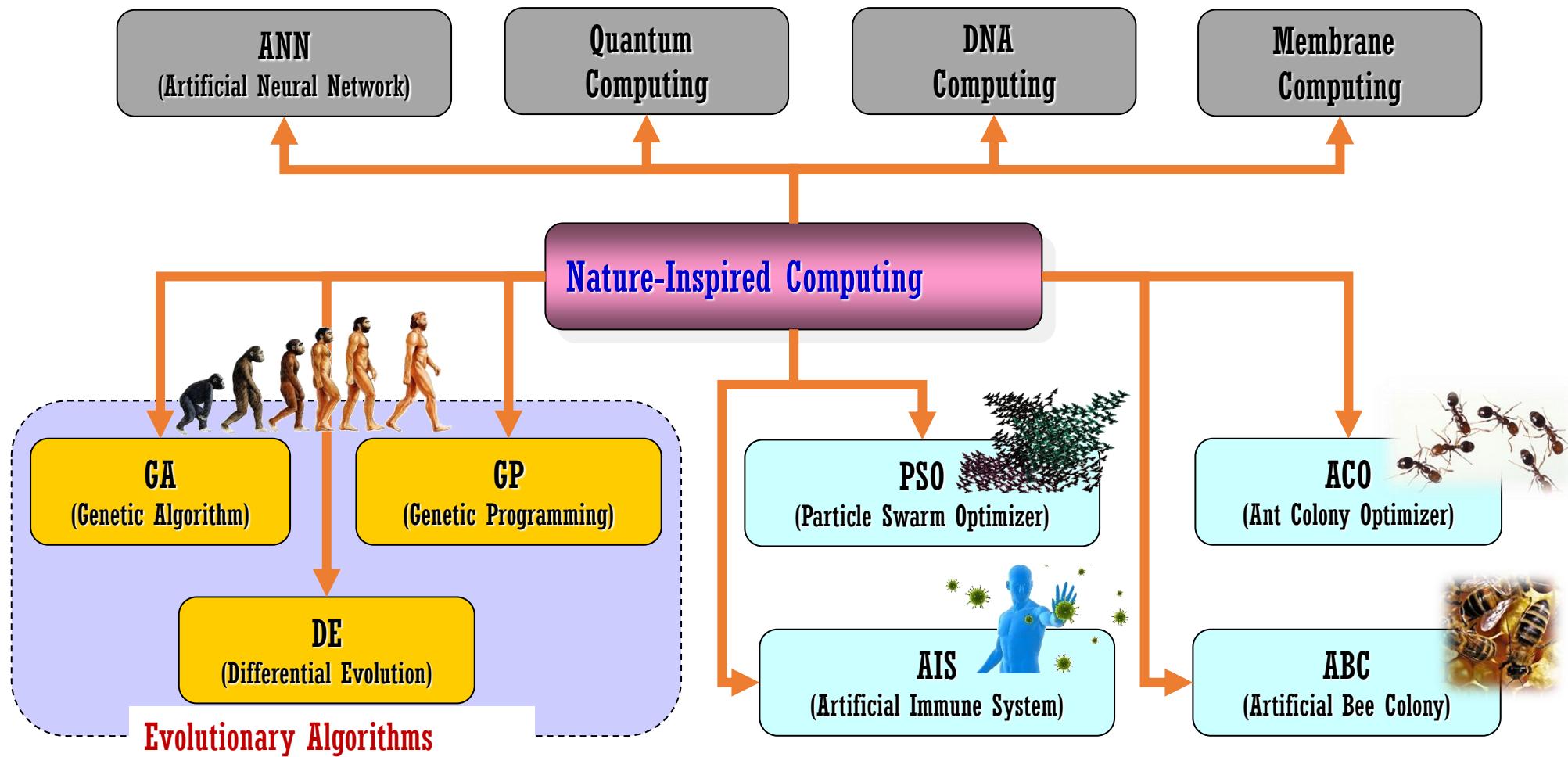
Multiple modal case



Suboptimum

Applications

Nature-Inspired Computing (Algorithms): Examples

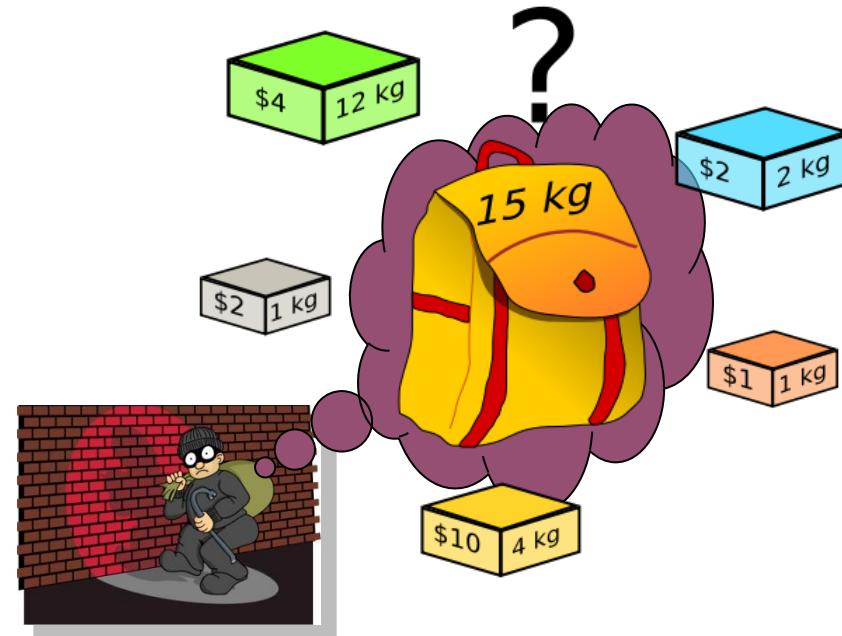


Applications

Traveling Salesman Problem



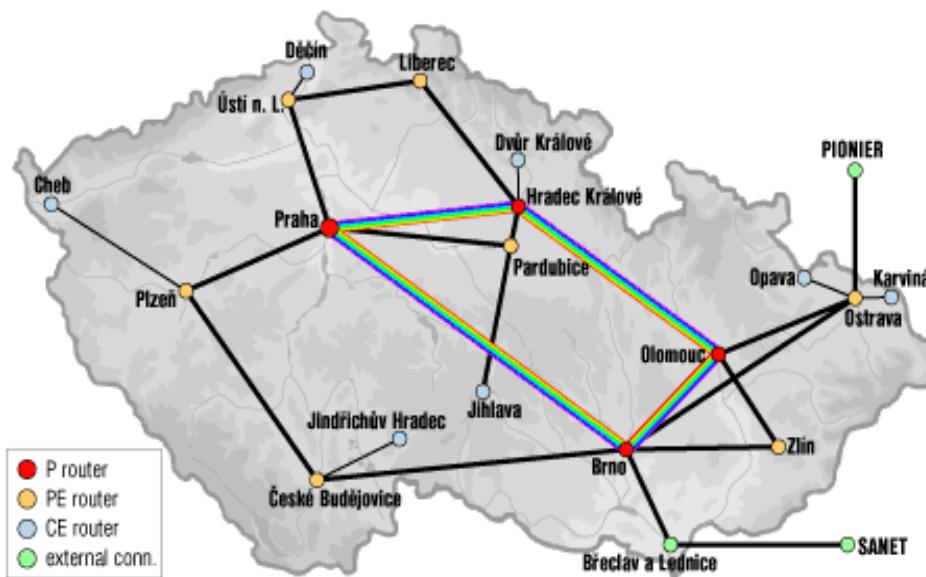
Knapsack Problem



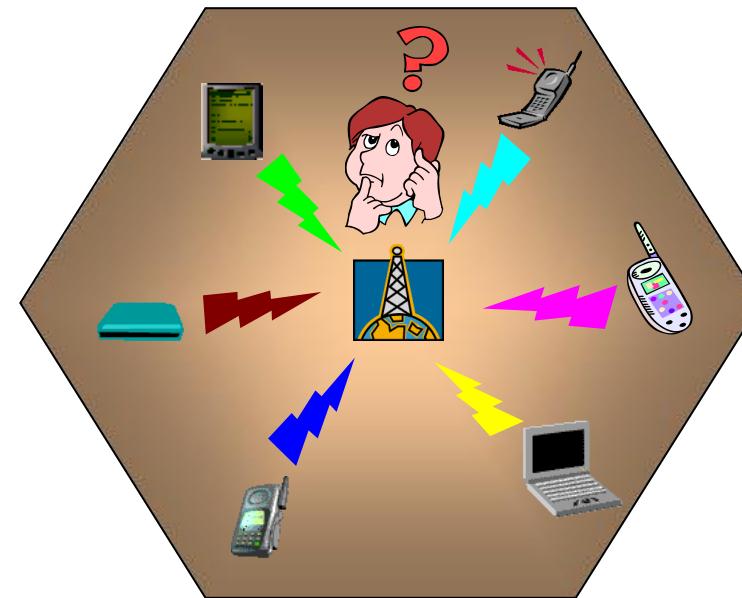
- Maximize the amount of profits (e.g., money) while still keeping the overall weight under or equal to a given limit!

Applications

Multicast Routing



Resource Allocation

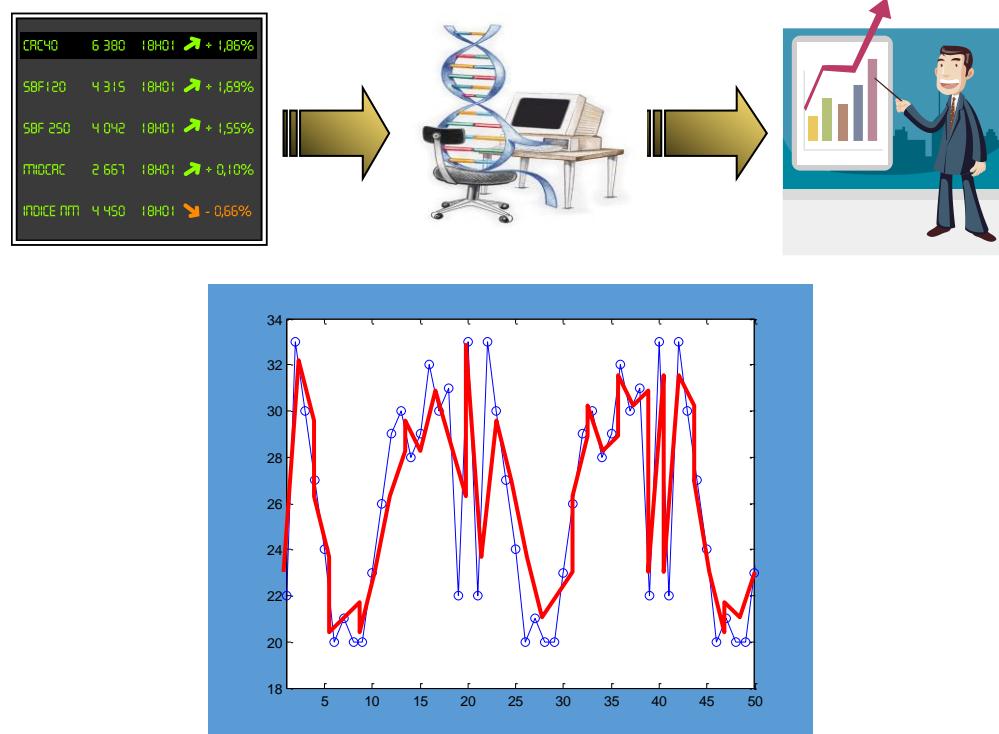


- Minimize the cost of multicast tree while satisfying delay and bandwidth constraints

- Maximize resource utilization by fairly distributing wireless resources among the connections

Applications

Time-Series Forecasting



- Predicting some future outcomes from a set of historical events
- Stock prediction, Weather forecasting, etc.

Decision in Dilemma

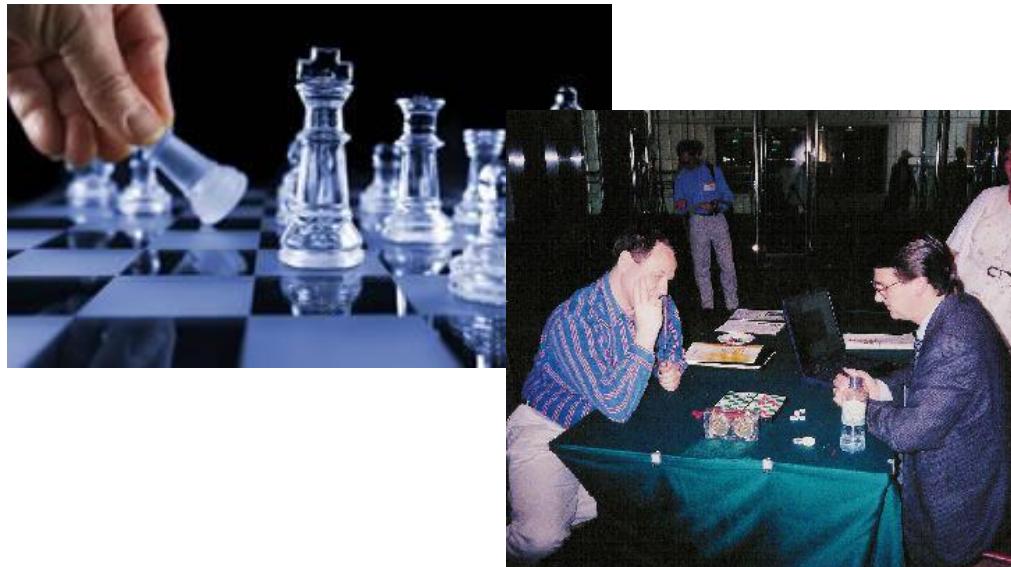


- Choosing a decision in conflict objectives
- Prisoner's dilemma, Game theory, etc.

Applications

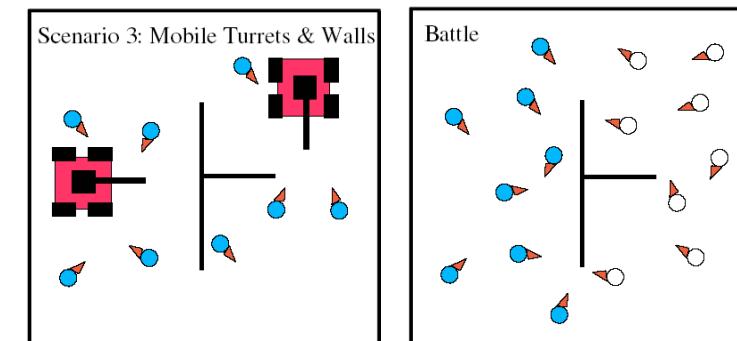
● Evolutionary Checker

- 8X8 board, 12 checkers for each player
 - Diagonal moves, Jumps are forced, etc.
- Neural Networks + Evolutionary Prog.
 - Checkerboards are evaluated by NNs
 - NNs and King value are evolved with EP
- Almost the expert level without knowledge



● Video Game: NERO

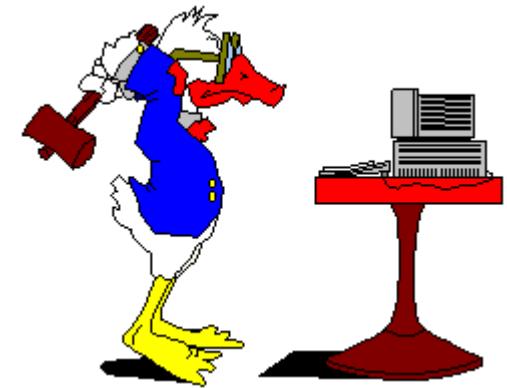
- Univ. of Texas at Austin
- Player's role
 - Train agents for competition
- No prepackaged or scripted agents
- Evolve in real-time



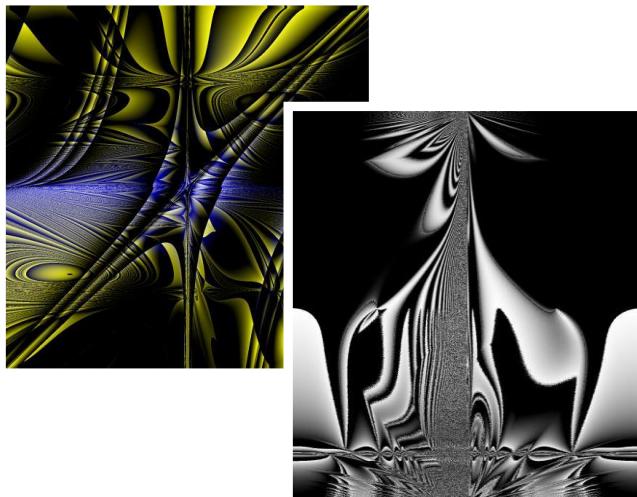
Applications

● What's Evolutionary Art?

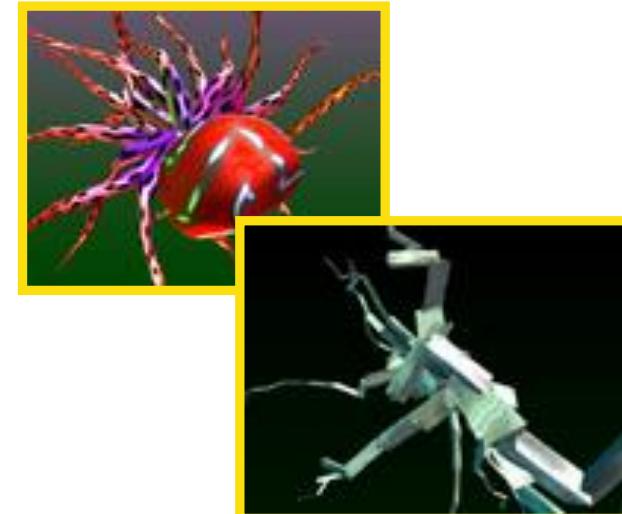
- Technically, it is creating pieces of art through human-computer interaction
- Computer runs evolutionary algorithms and human applies subjective selection
 - Role of computers: offer choices and create diversity
 - Role of human: make (subjective) choices and reduce diversity
- Selection (aesthetic/subjective) steers towards implicit user preferences



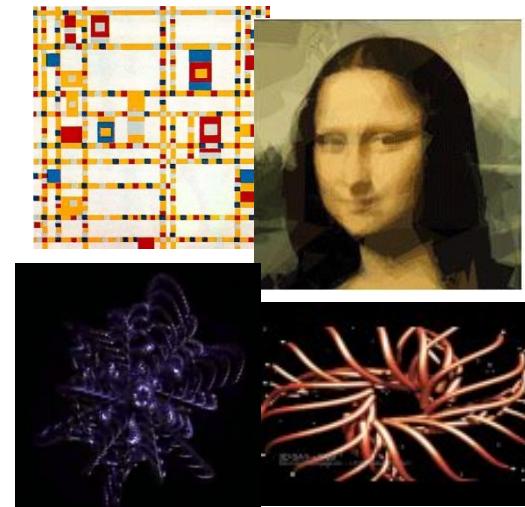
Evol. Art by Kleiweg



Galapagos by Karl Sims



Other Examples



Applications

● GenJam (Genetic Jammer)

- Developed in 1993~94 by Prof. John Al Biles
- Interactive GA that learns to play jazz solos
- GenJam's repertoire: Over 250 jazz-style tunes
- Evolving by special fitness operators;
e.g., rhythm conformity
- What can it be done?
 - ✓ Playing full-chorus improvised solos
 - ✓ Listening to trumpet and responds interactively when we trade fours
 - ✓ Engaging in collective improvisation;
we both solo simultaneously and GenJam performs a smart echo of improvisation
 - ✓ Listening to me and play the head of a tune and breeds my measures



Source:
<http://www.it.rit.edu/~jab/GenJam.html>



Source: <http://phoenix.inf.upol.cz/~dostal/evm.html>



Virtual quintet

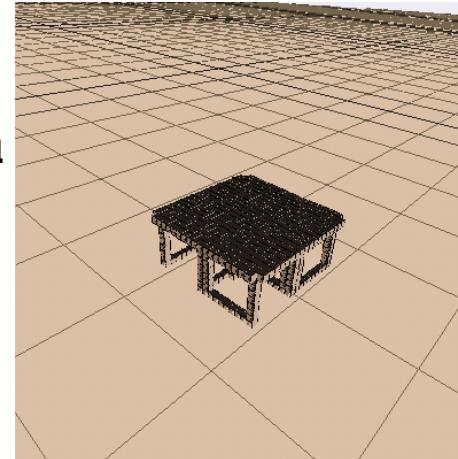
MusiGenesis



Applications

● Structure Design

- Bridge structure optimization
- Building structure design



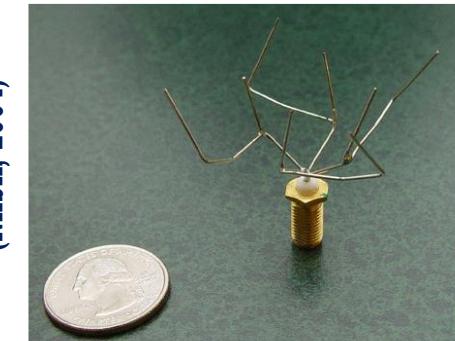
● Aviation System Design

- Airfoil, wing, and antenna designs
- Space platform structure optimization
- Jet aircraft model optimization

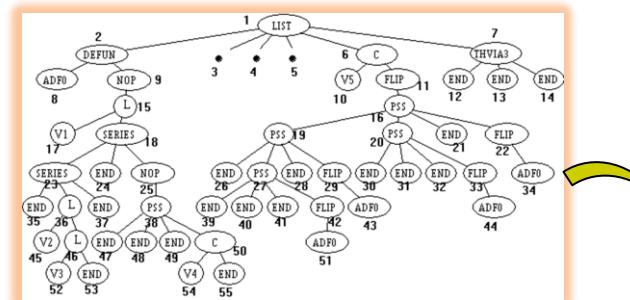


● Circuit Design

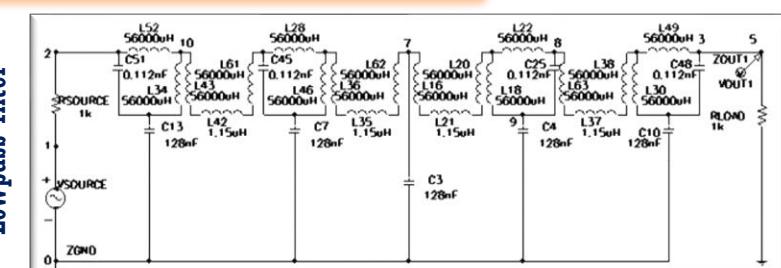
- Automatic synthesis of topology & sizing of analog electrical circuits



Evolved antenna
(NASA, 2004)



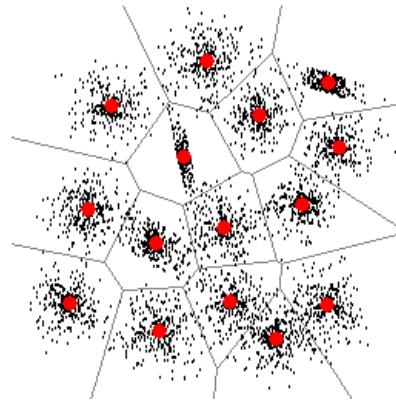
Lowpass filter



Applications

● Clustering

- Data clustering
- Text mining
- Web search

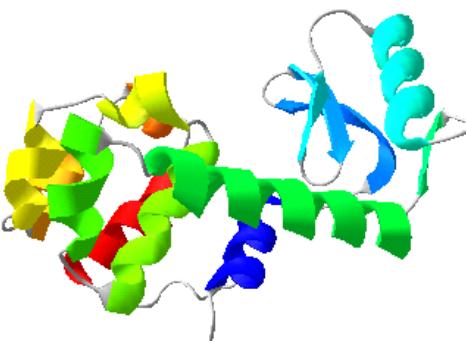


enterprise infrastructure
technology operations
information objectives
scorecards
analyze text mining
metrics
applications connection
solution stakeholder



● Bioinformatics

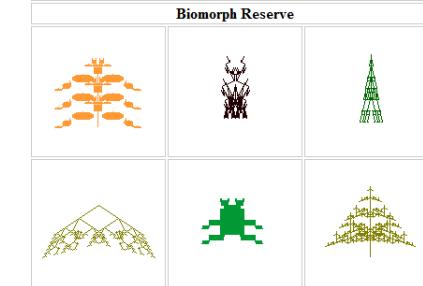
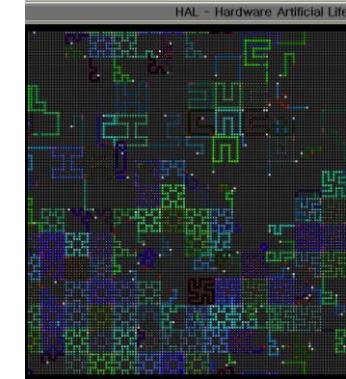
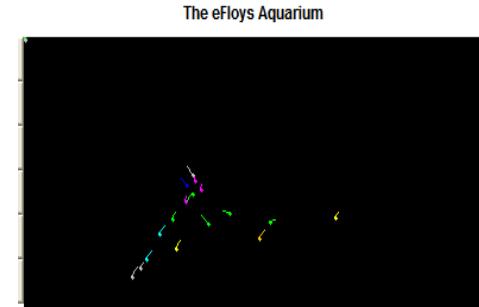
- Drug discovery
- Protein folding
- Cancer diagnosis



Applications

● Artificial Creatures

- [eFly](#), [Biomorph](#), [HAL](#),
- [Self-replicating Worms](#)
- [Gozilla](#), [Solitaire](#)



● Robotics

- Humanoid Robots; e.g., e.g., [ASIMO](#)
- Genetic Robots; e.g., [Gene](#)
- Others; e.g., [Six-Legged Robot](#)
[Robot Snake](#)



