AI VIETNAM
Free AI Course 2020

Genetic Algorithm

Nature-Inspired Optimization Method

Quang-Vinh Dinh
Hong-Phuc Nguyen

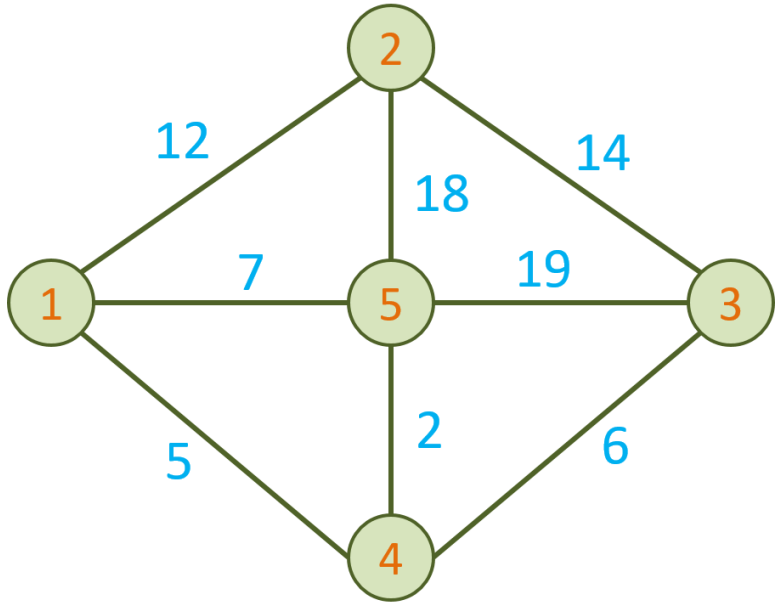
Year 2020

Outline

- NP-Hard Problem
- Random Search
- Genetic Algorithm
- For Sphere Function
- For House Price Prediction

NP-Hard Problems

Travelling Salesman



Phát biểu bài toán như sau:

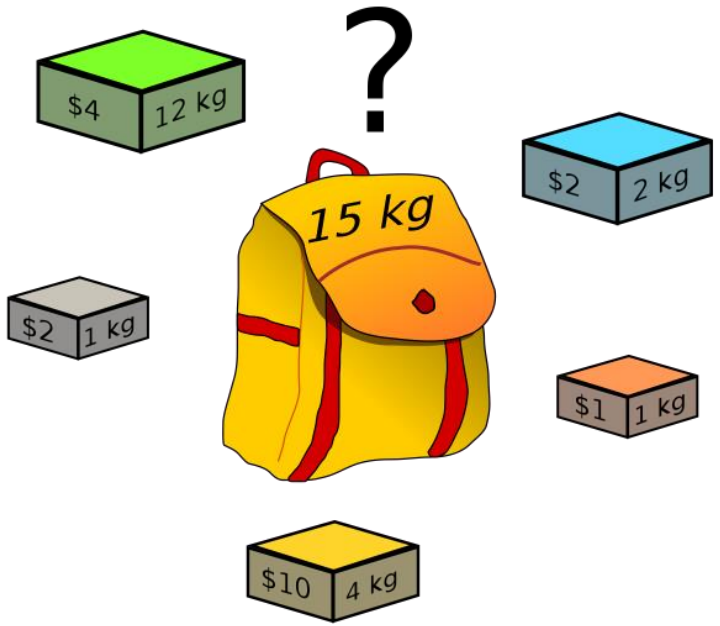
Có 5 thành phố được đánh số từ 1 đến 5, được nối với nhau như hình vẽ.

Một người muốn đi từ một thành phố, qua tất cả các thành phố khác và trở về thành phố ban đầu với chi phí nhỏ nhất.

Hãy thiết lập tuyến đường cho người này.

NP-Hard Problems

Knapsack problem



https://en.wikipedia.org/wiki/Knapsack_problem

Phát biểu bài toán như sau:

Có n=12 vật có giá trị và cân nặng cho trước.

Hãy để n vật này vào một cái túi có sức chứa tối đa max_weight=70 kg sao cho giá trị trong chiếc túi là lớn nhất.

NP-Hard Problems

❖ One-max problem

Each vector v has the length of n

$n = 10$

Secret information

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

vector v1

1	1	0	1	0	1	1	0	0	1

secret(v1) → 6

vector v2

1	0	0	0	1	0	0	1	0	0

secret(v2) → 3

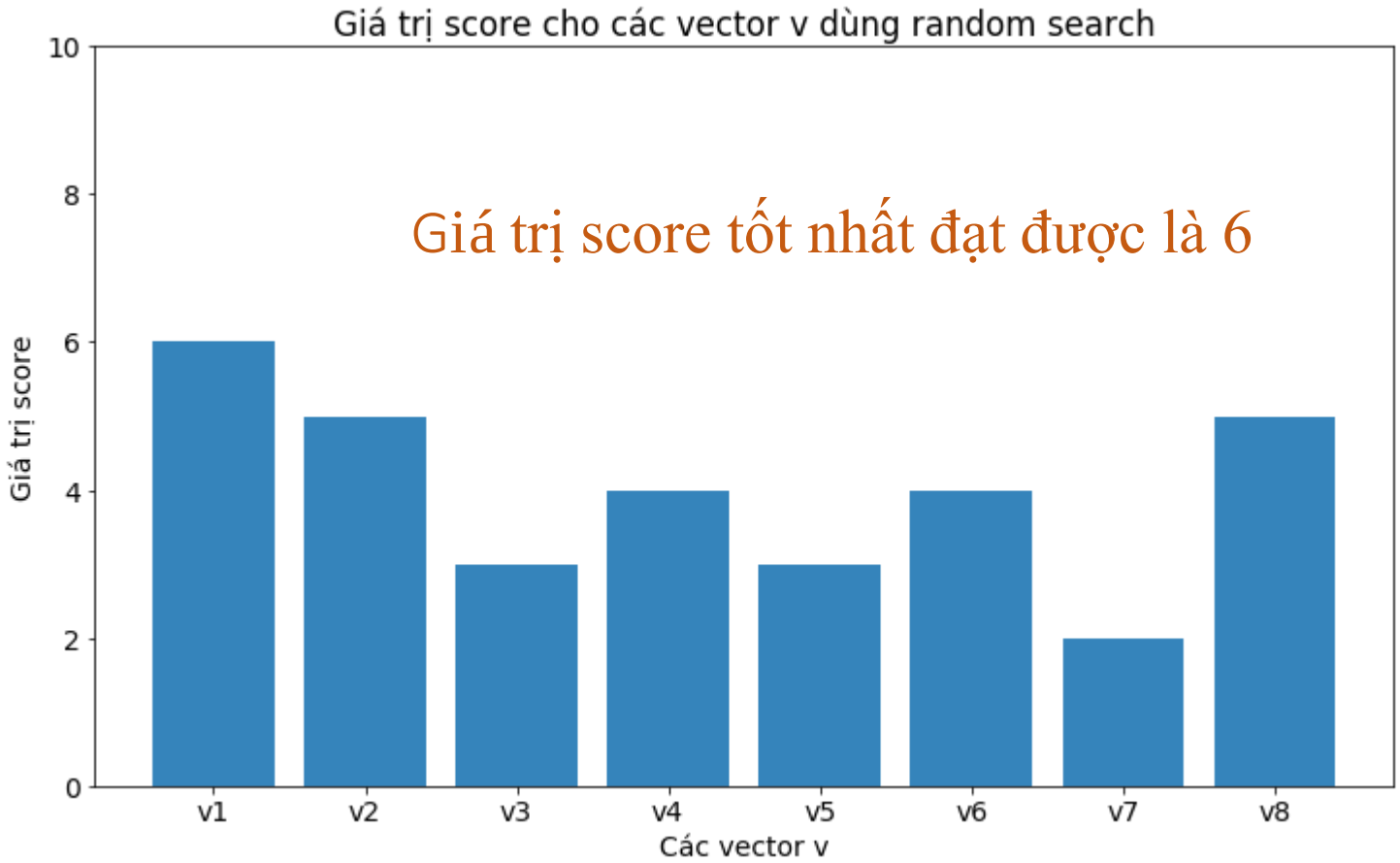
Random Search

❖ How to obtain vectors with good scores

Generate m vectors randomly

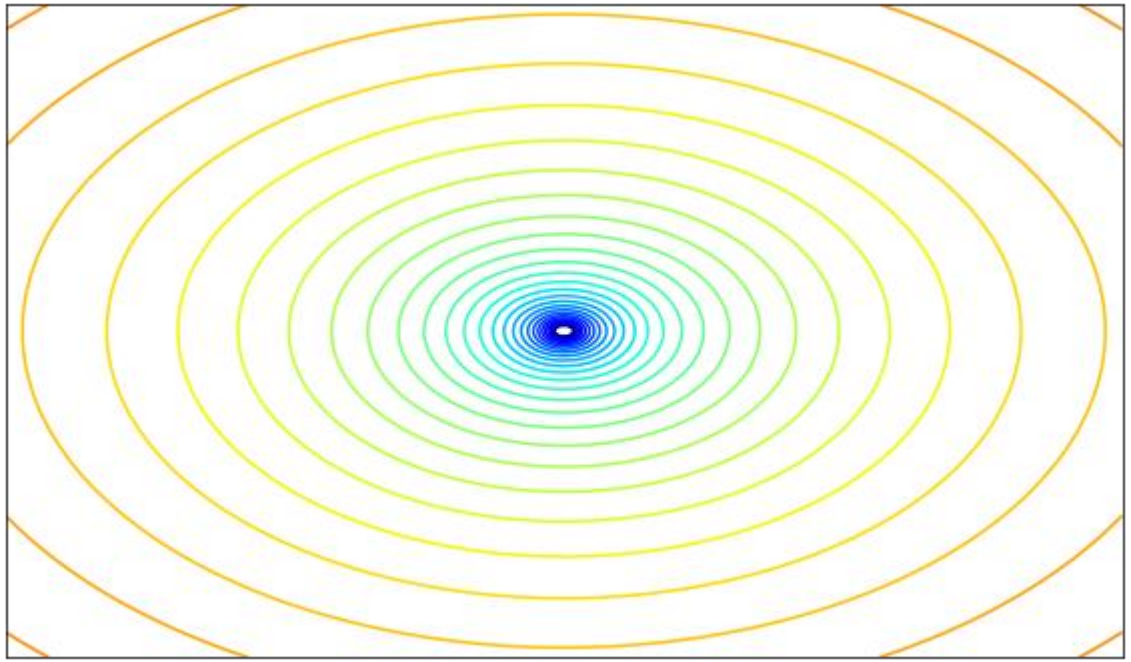
v_i receives value 0 or 1 randomly

```python
def generate_vectors(n=10, m=8):
    vectors = [[0]*n for _ in range(m)]

    for i in range(m):
        for j in range(n):
            if random.random() >= 0.5:
                vectors[i][j] = 1

    return vectors
```

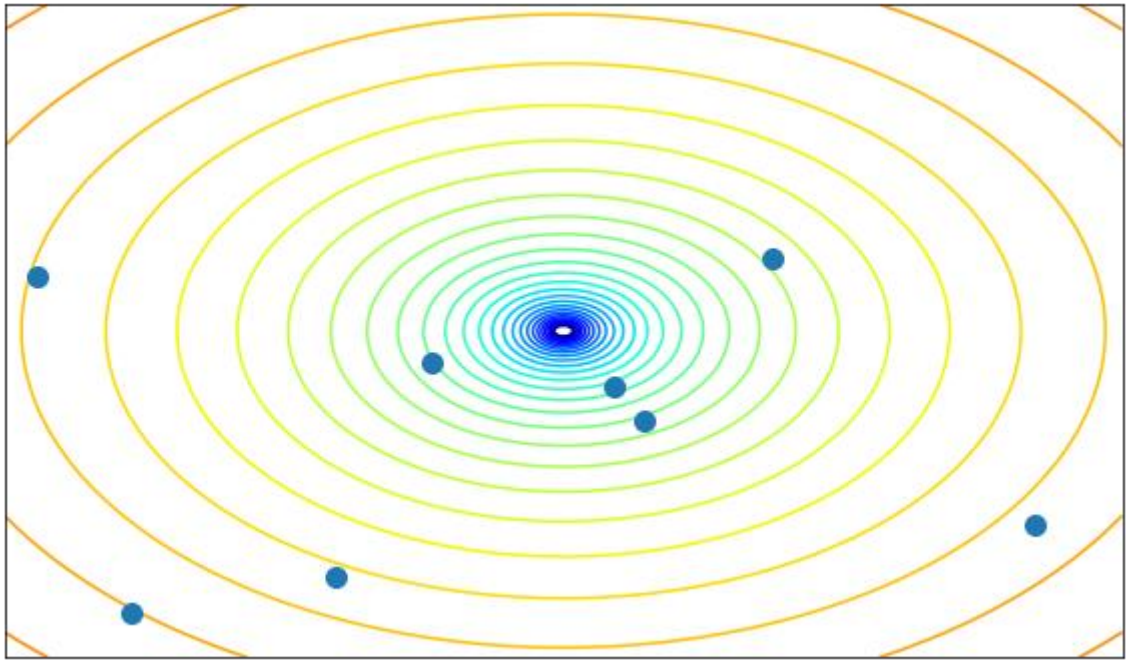


Random Search

Searching space



M randomly generated vectors

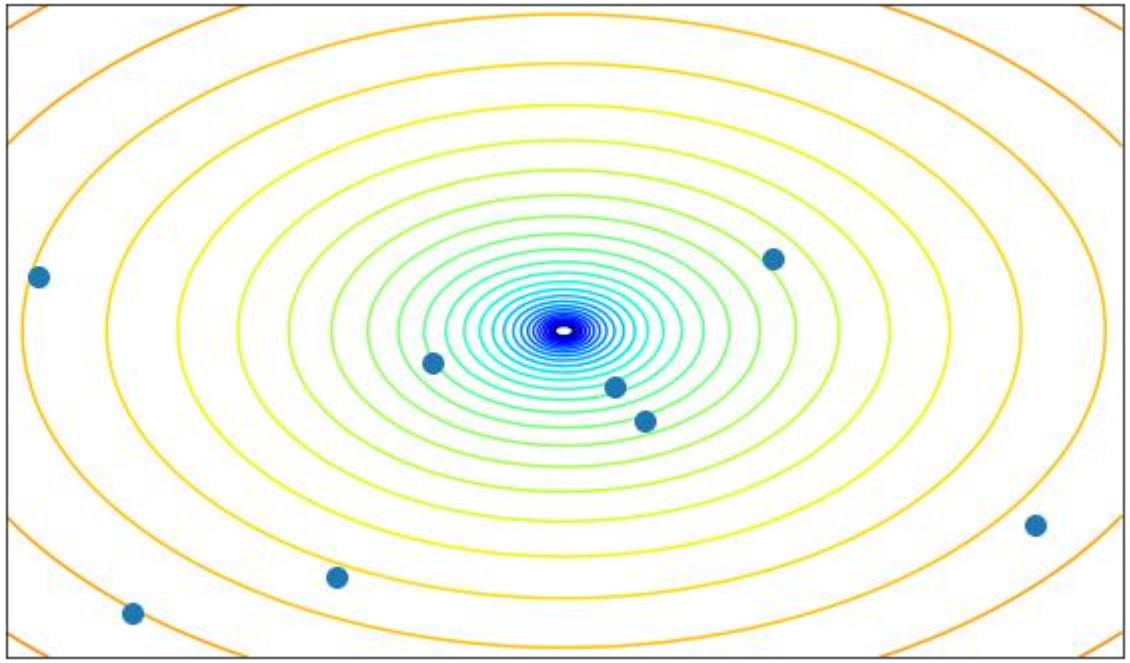


Random Search

❖ **Increase m to infinity**

Impractical because of the limitation of resources

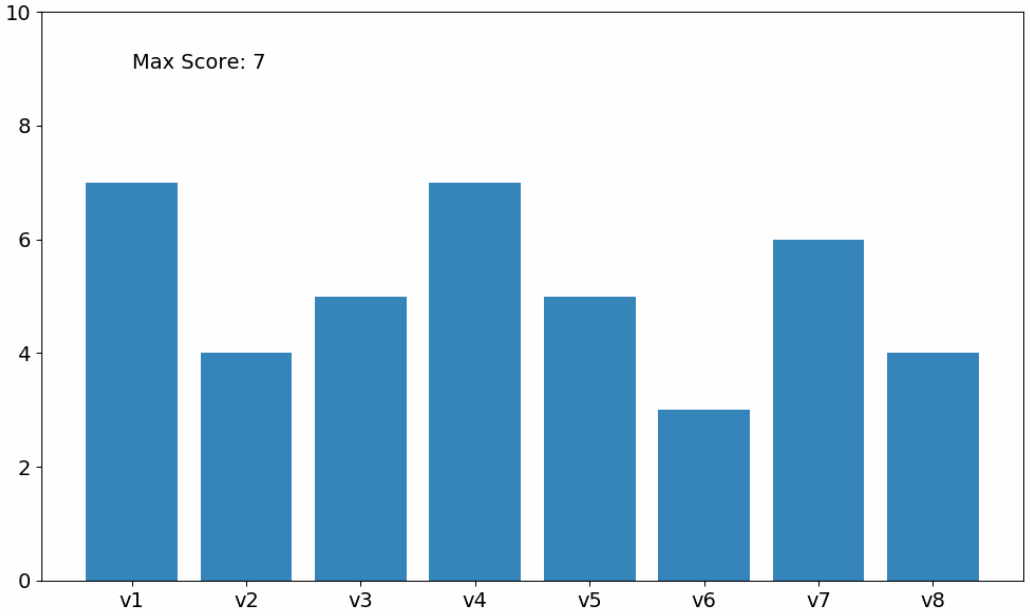
m randomly generated vectors

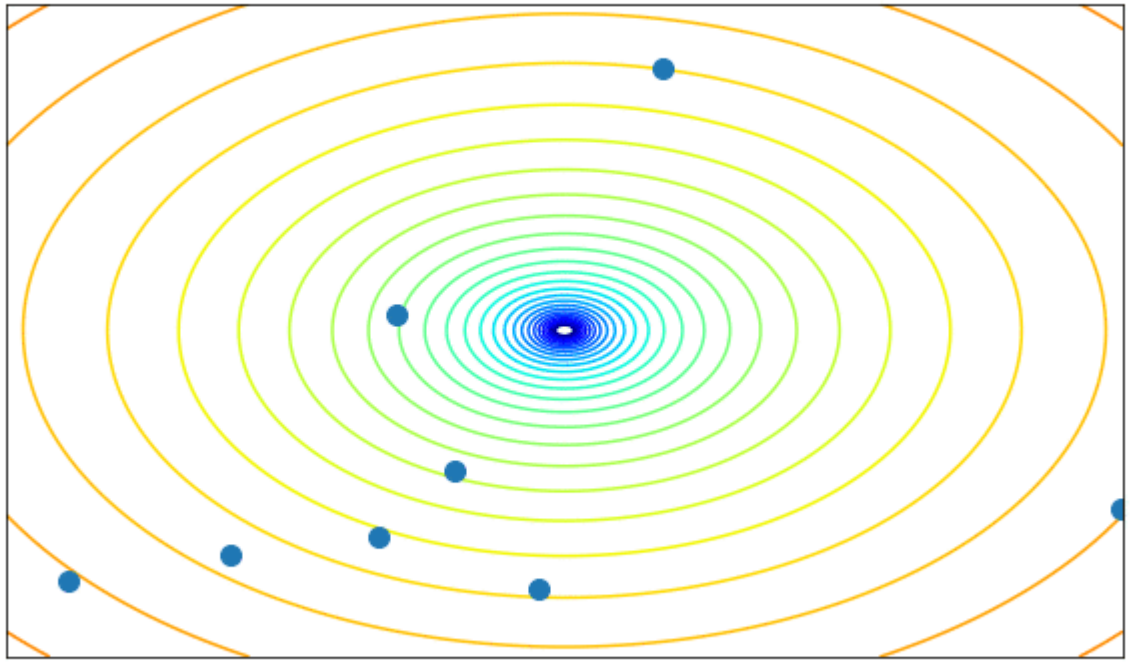


Need to use  rather than 

Random Search

- Generate different groups of m vectors



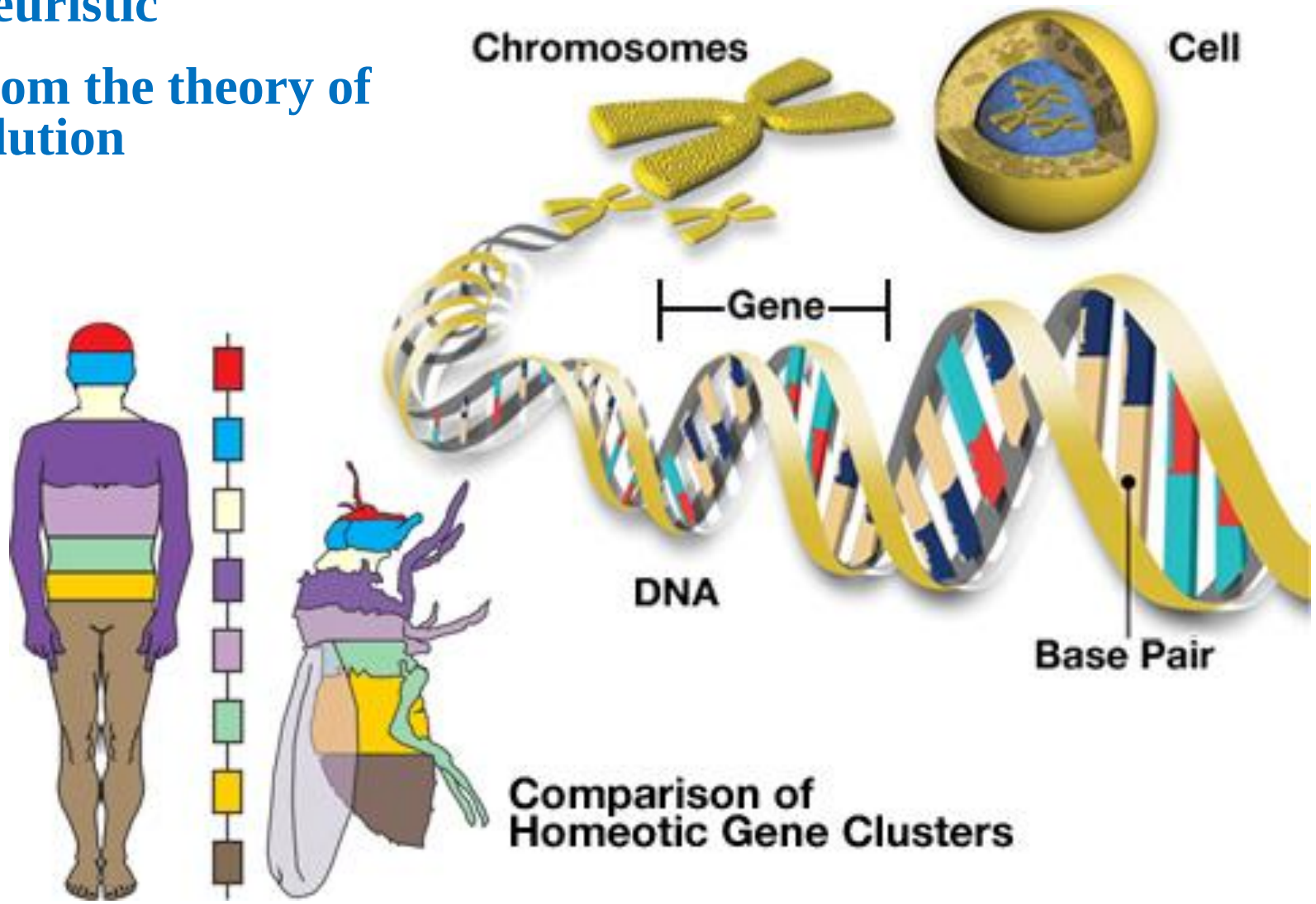


Outline

- NP-Hard Problem
- Random Search
- Genetic Algorithm
- For Sphere Function
- For House Price Prediction

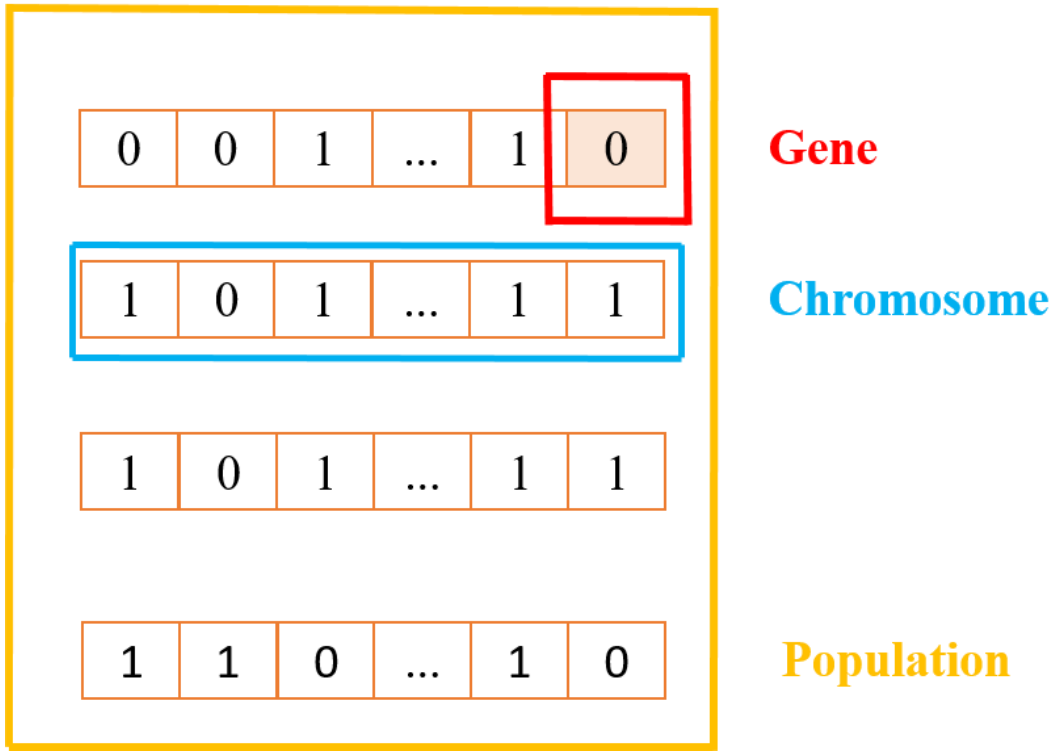
Genetic Algorithm

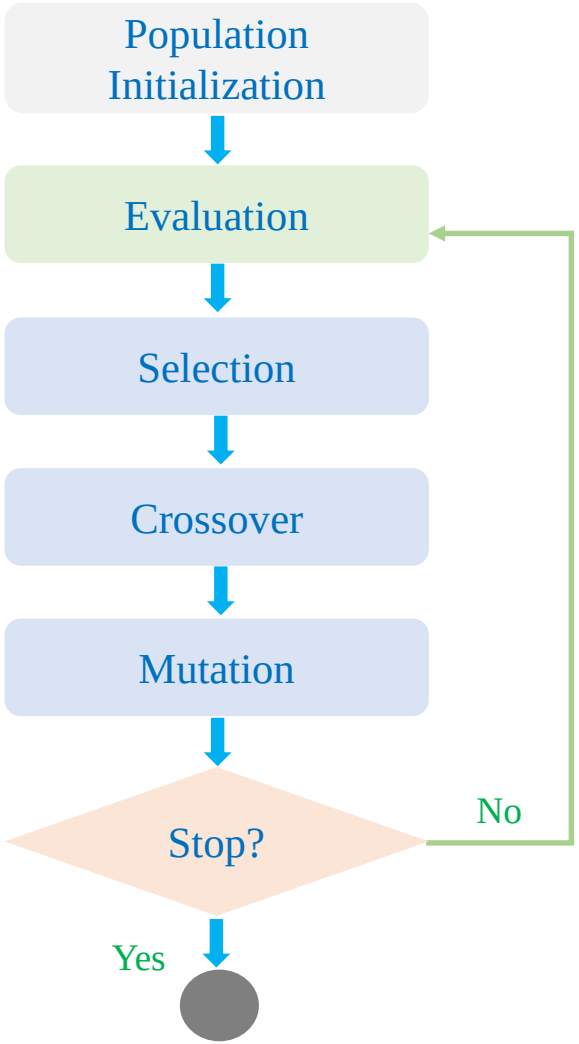
- A search heuristic
- Inspired from the theory of natural evolution



Genetic Algorithm

❖ To solve the one-max problem

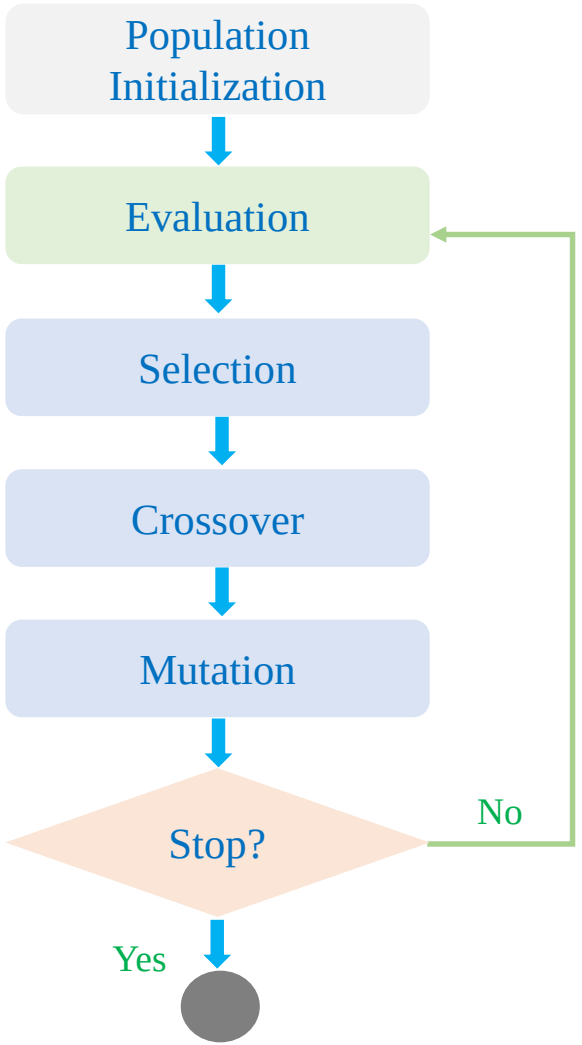




AI VIETNAM
Free AI Course

Genetic Algorithm

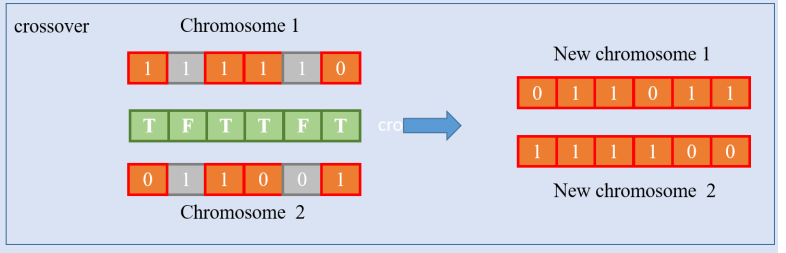
❖ Steps in Genetic Algorithm



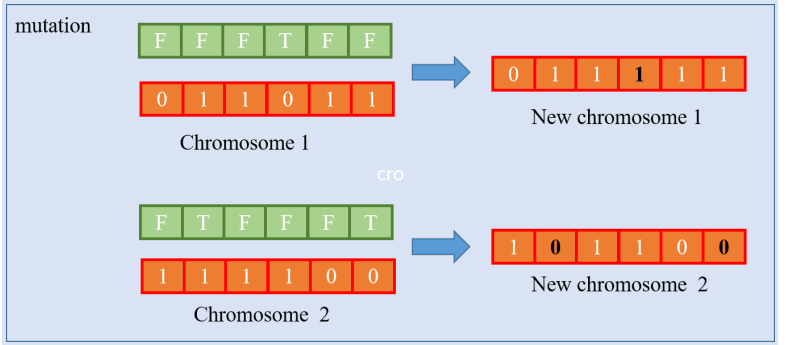
Khởi tạo quần thể: Gán giá trị ngẫu nhiên cho các gen của các cá thể

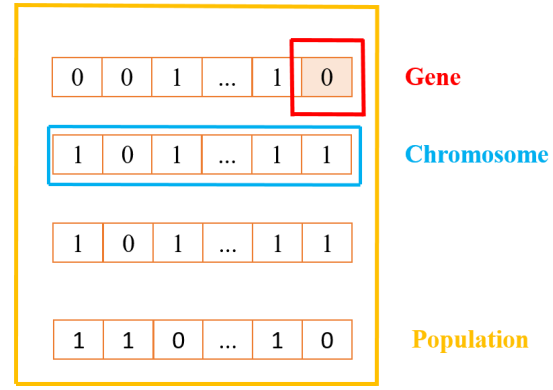
Cá thể 1	0	0	1	1	1	0
Cá thể 2	1	0	1	0	1	0
Cá thể 3	0	0	1	0	0	1
Cá thể 4	0	1	1	0	0	1
Cá thể 5	0	1	1	1	1	0
Cá thể 6	1	0	0	0	1	1
Cá thể 7	0	0	0	0	0	1
Cá thể 8	1	0	0	0	0	0
Cá thể 9	0	1	0	0	0	1
Cá thể 10	1	1	1	1	1	0

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

Genetic Algorithm

❖ Population initialization

Population size m = 10

Vector length is with n = 5

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

```python
# aivietnam.ai
import random

n = 6   # size of individual (chromosome)
m = 10  # size of population

def generate_random_value():
    return random.randint(0, 1)

def create_individual():
    return [generate_random_value() for _ in range(n)]

population = [create_individual() for _ in range(m)]

# print population
for ind in population:
    print(ind)
```

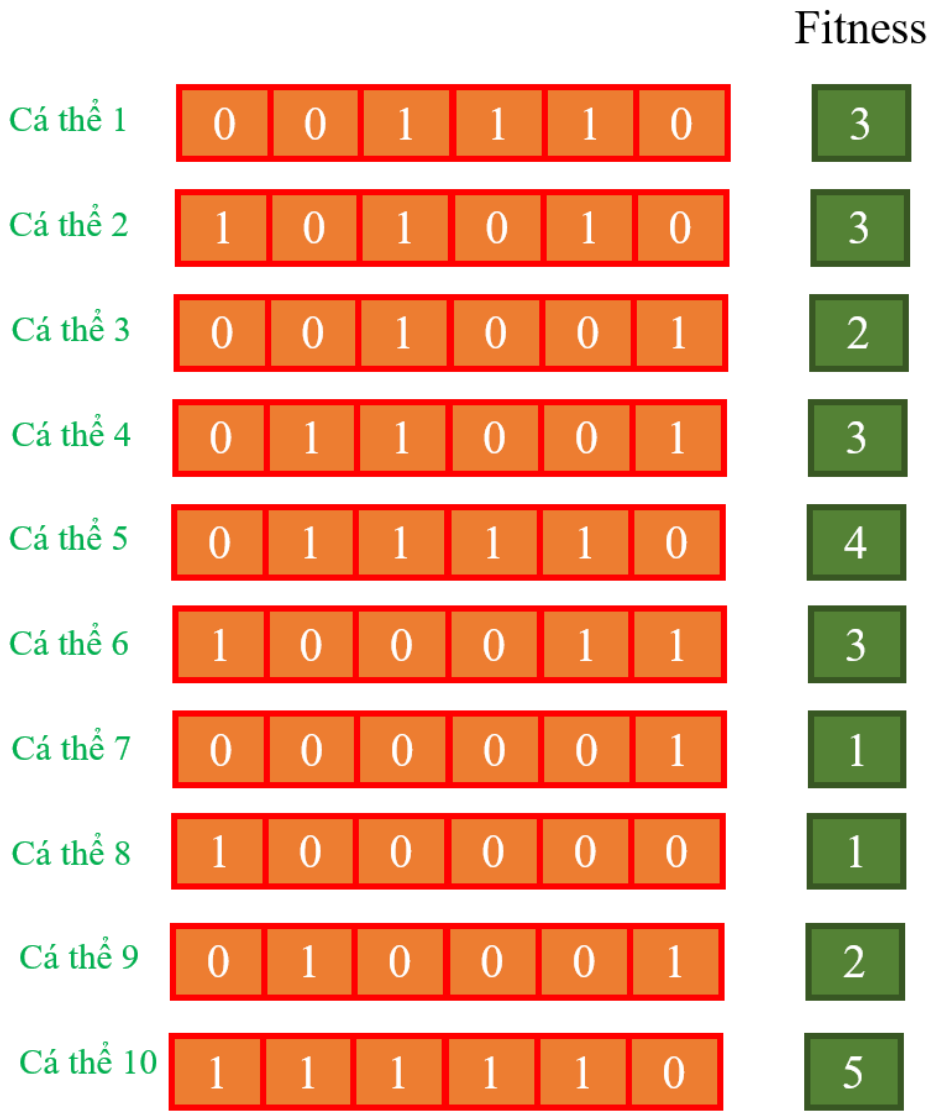
```
[0, 0, 0, 1, 0, 1]
[0, 0, 0, 0, 1, 1]
[0, 1, 0, 1, 0, 1]
[1, 0, 1, 1, 0, 1]
[1, 0, 0, 1, 0, 1]
[1, 1, 0, 1, 0, 0]
[0, 0, 1, 1, 0, 0]
[1, 1, 1, 1, 1, 1]
[1, 0, 0, 0, 0, 1]
[0, 0, 1, 1, 0, 1]
```

Genetic Algorithm

❖ Evaluation

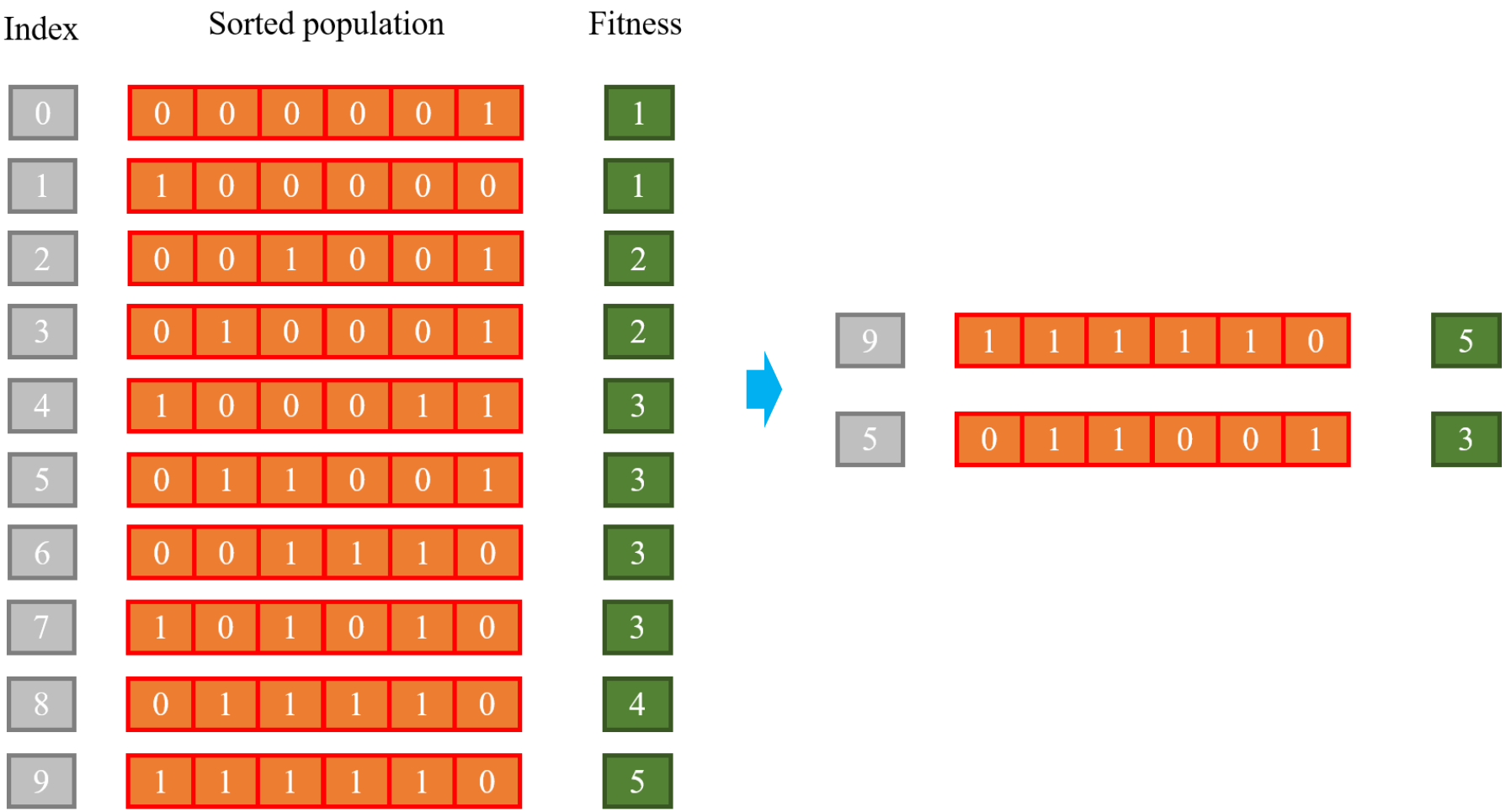
Secret information

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



Genetic Algorithm

❖ Selection



Genetic Algorithm

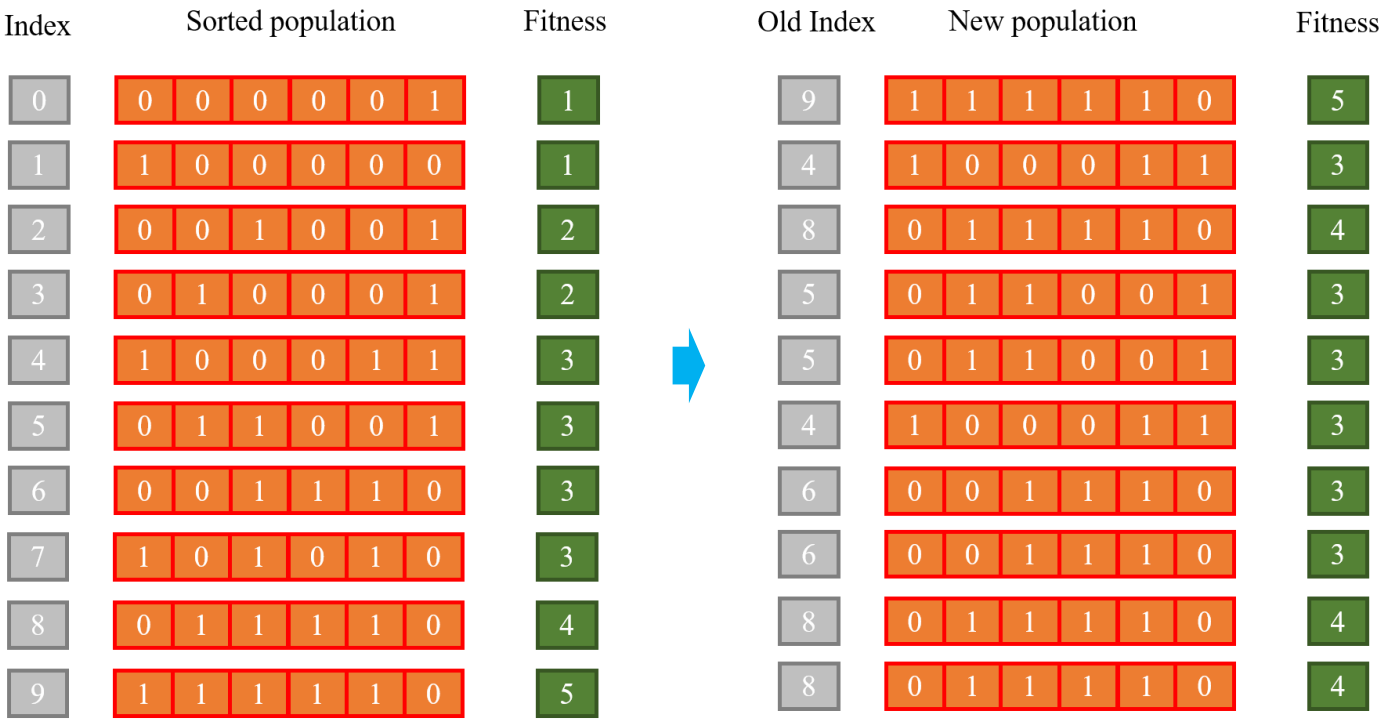
❖ Selection

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

Old Index	New population	Fitness
9	1 1 1 1 1 0	5
4	1 0 0 0 1 1	3
8	0 1 1 1 1 0	4
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
8	0 1 1 1 1 0	4
8	0 1 1 1 1 0	4

Genetic Algorithm

❖ Selection

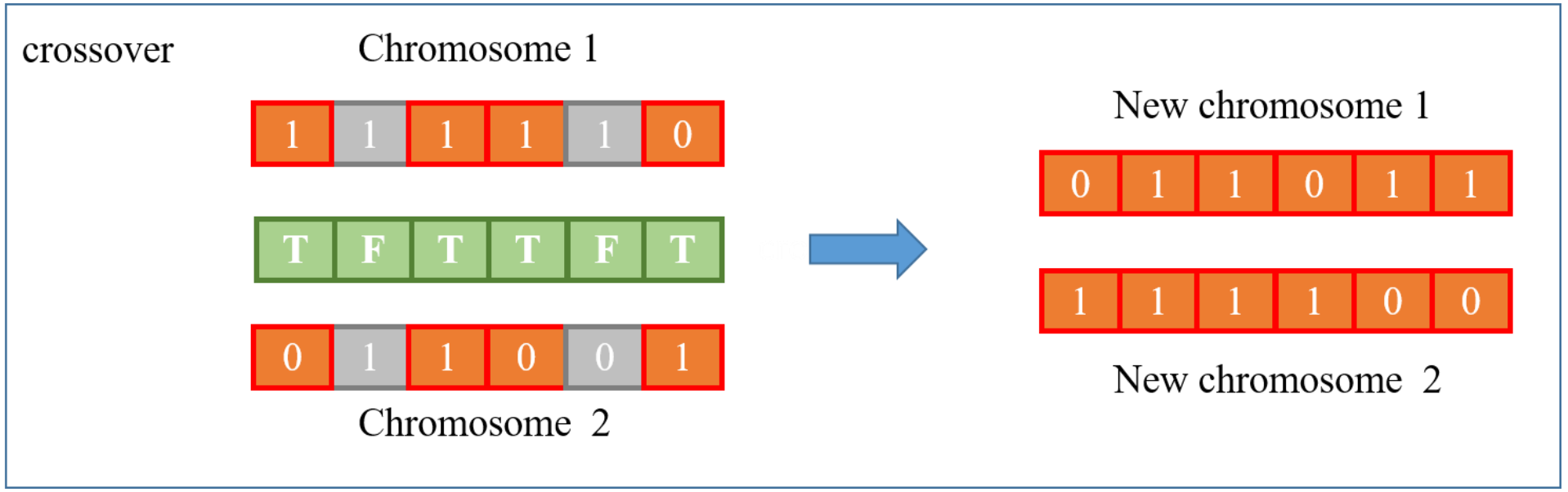



```python
def selection(sorted_old_population):
    index1 = random.randint(0, m-1)
    while True:
        index2 = random.randint(0, m-1)
        if (index2 != index1):
            break

    individual_s = sorted_old_population[index1]
    if index2 > index1:
        individual_s = sorted_old_population[index2]

    return individual_s
```

Genetic Algorithm

- **Crossover**
  - **Binary crossover**

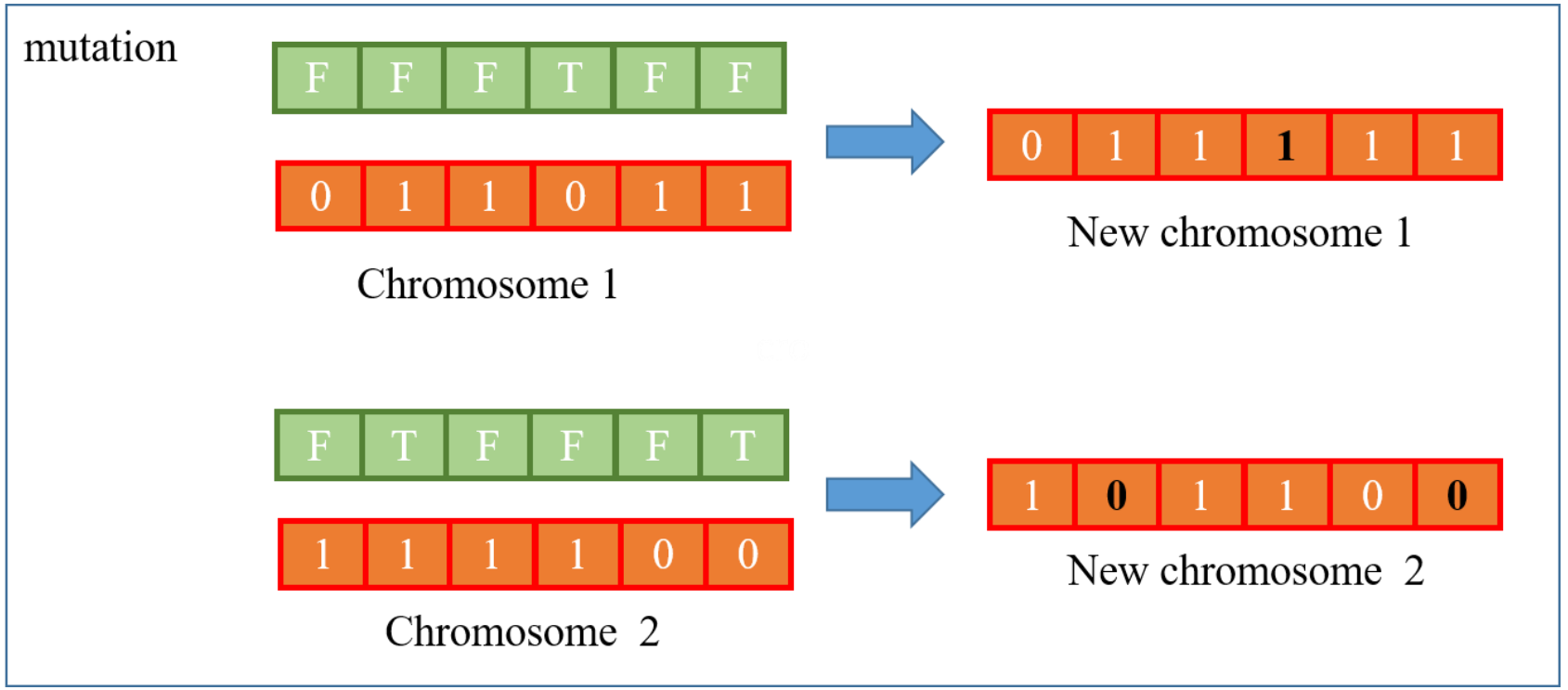
```python
def crossover(individual1, individual2, crossover_rate = 0.9):
    individual1_new = individual1.copy()
    individual2_new = individual2.copy()

    for i in range(n):
        if random.random() < crossover_rate:
            individual1_new[i] = individual2[i]
            individual2_new[i] = individual1[i]

    return individual1_new, individual2_new
```



Genetic Algorithm

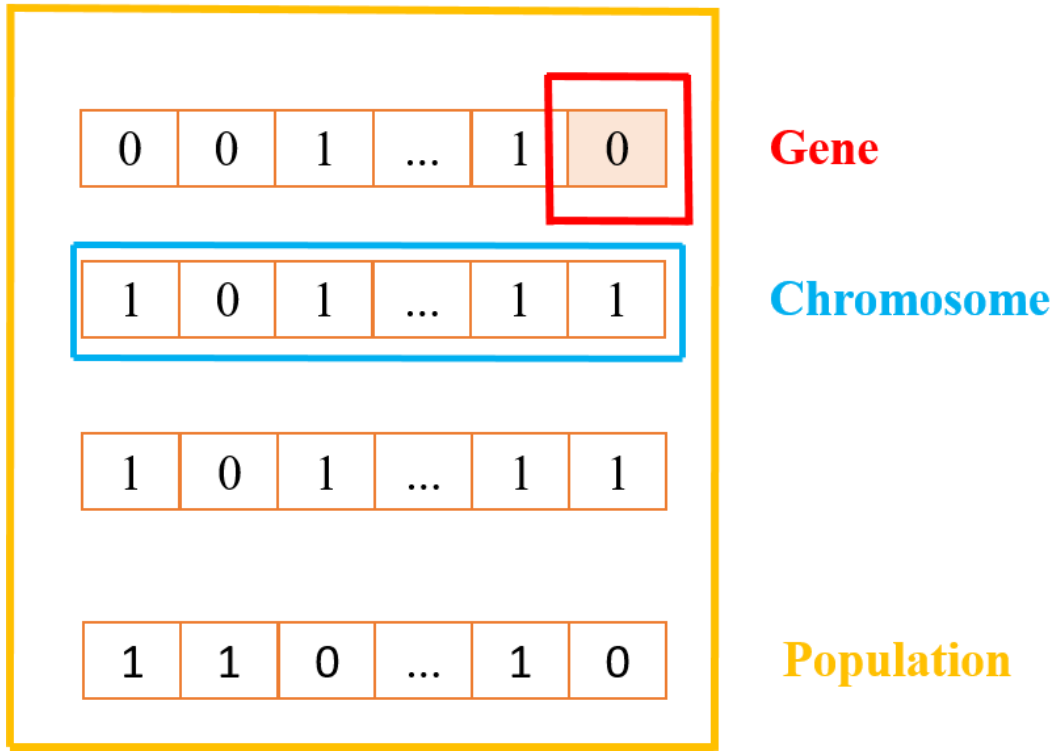
❖ Mutation

```python
def mutate(individual, mutation_rate = 0.05):
    individual_m = individual.copy()

    for i in range(n):
        if random.random() < mutation_rate:
            individual_m[i] = generate_random_value()

    return individual_m
```

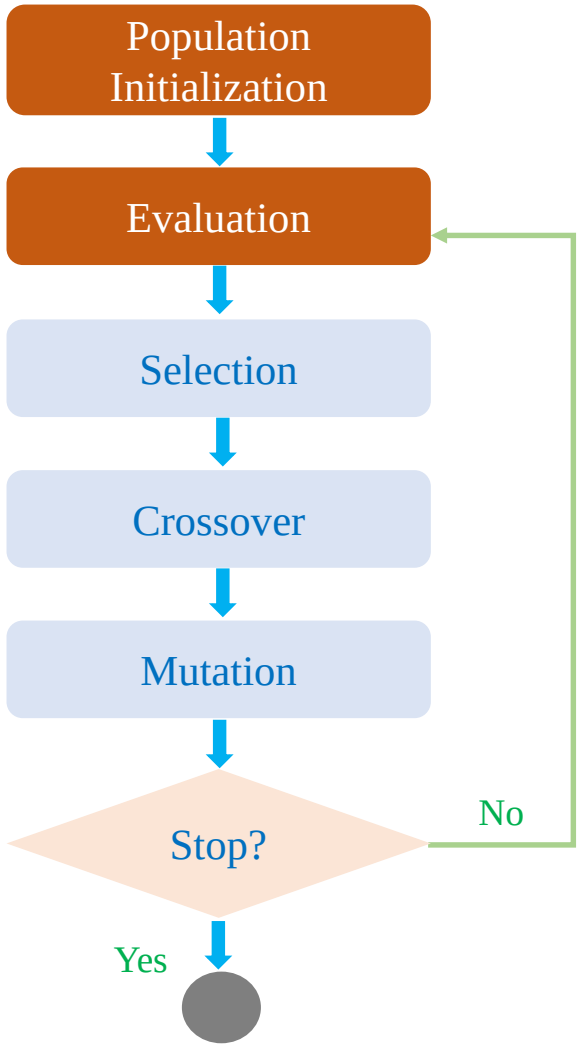


Genetic Algorithm

❖ To solve the one-max problem



Elitism



Outline

- NP-Hard Problem
- Random Search
- Genetic Algorithm
- For Sphere Function
- For House Price Prediction

Genetic Algorithm

❖ Sphere function

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

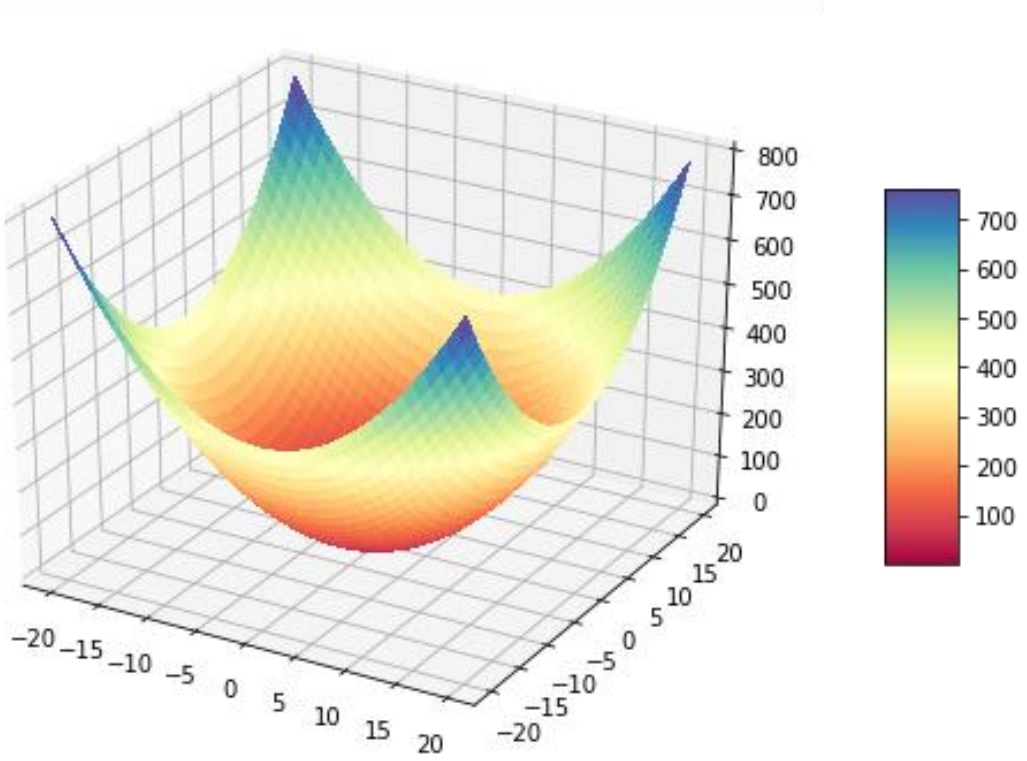
Do các biến x_i có kiểu số thực, nên số điểm trong không gian tìm kiếm là vô hạn.

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()

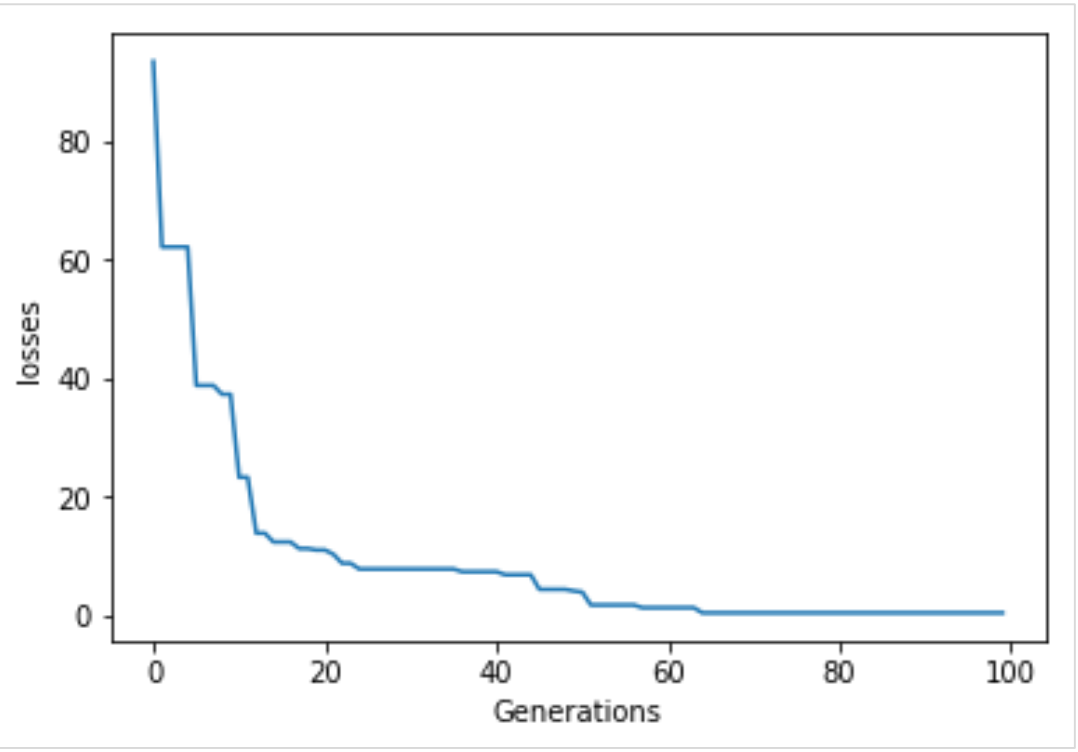
[-20, 20]

```
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
```



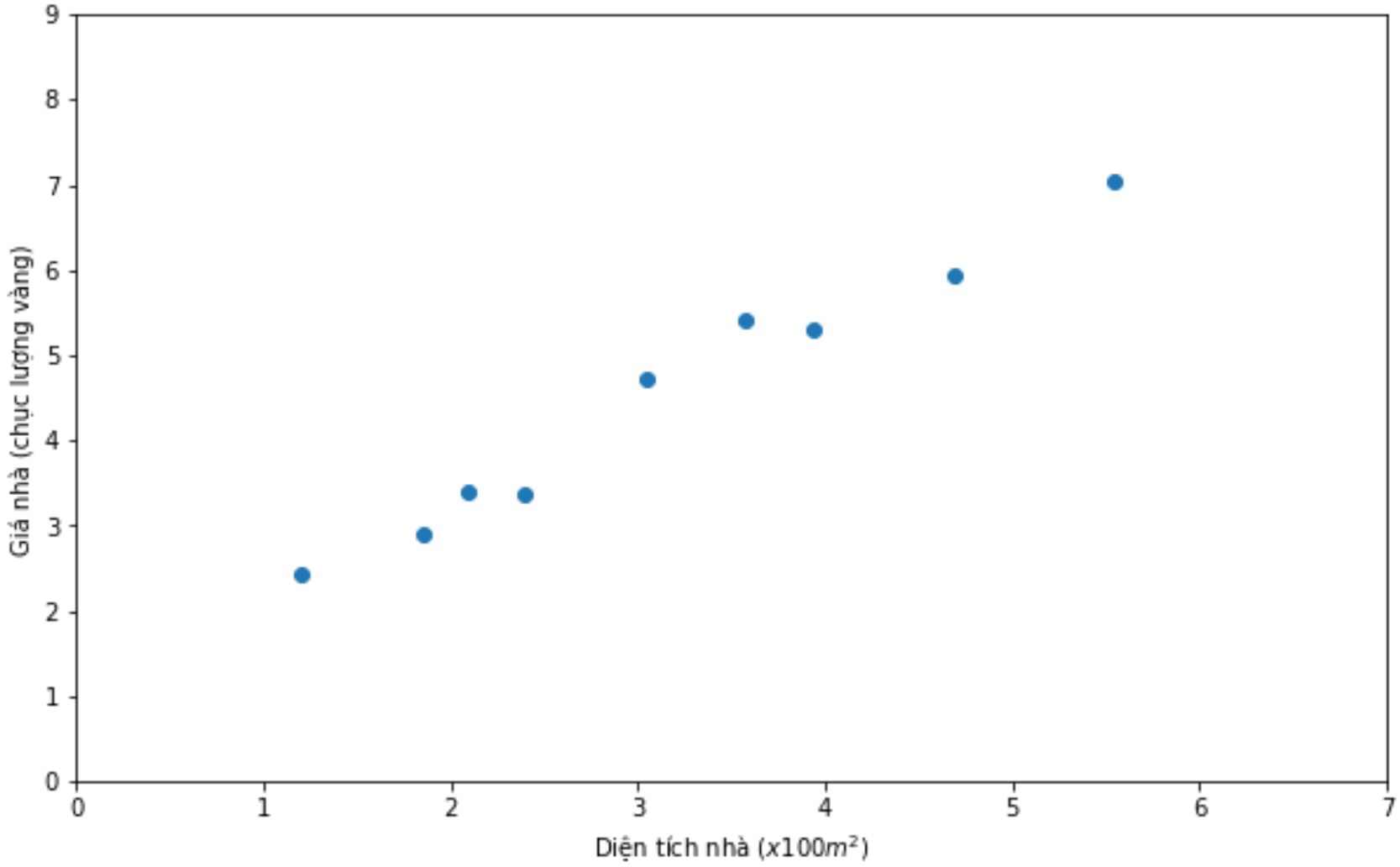
Outline

- NP-Hard Problem
- Random Search
- Genetic Algorithm
- For Sphere Function
- For House Price Prediction

Genetic Algorithm

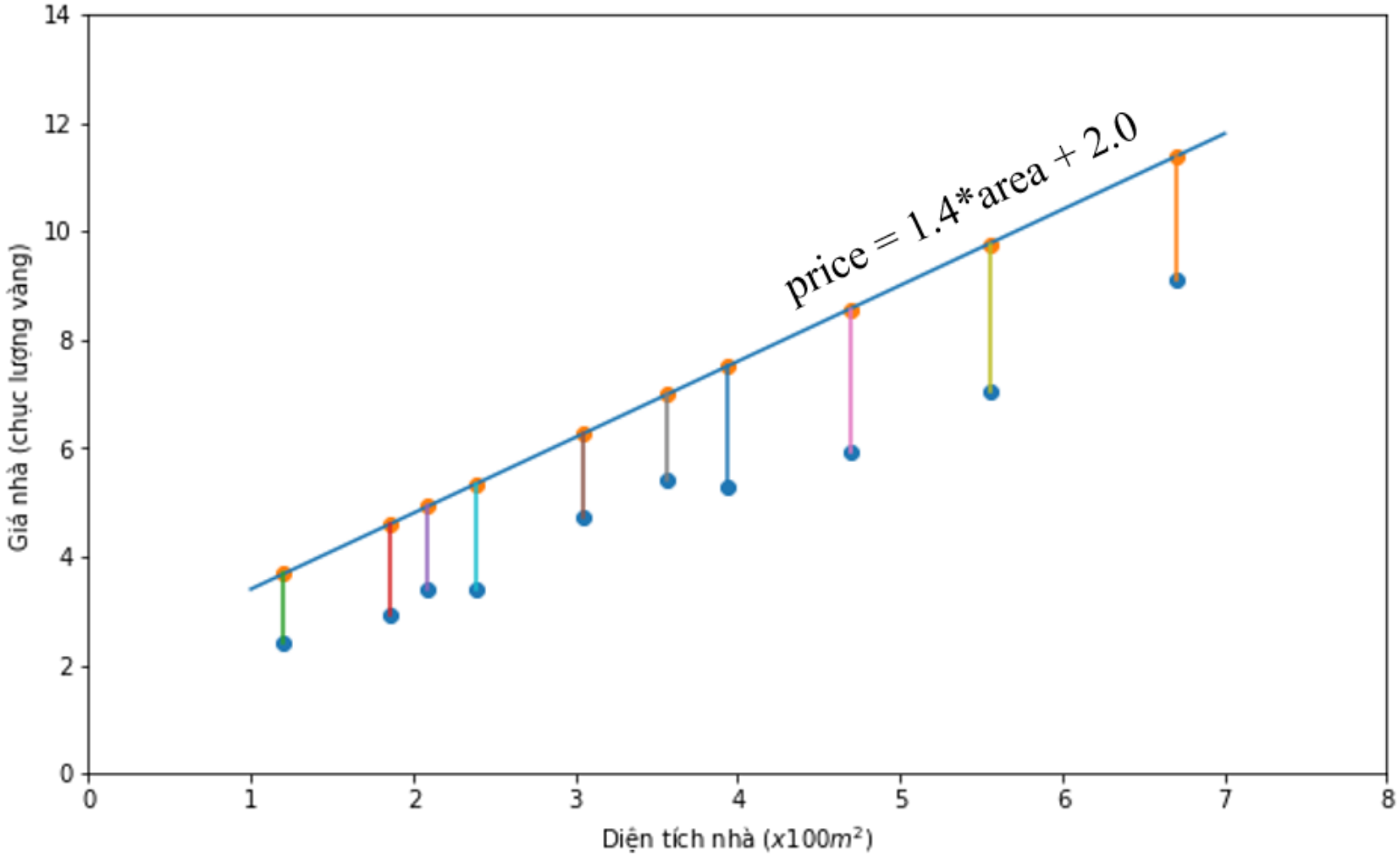
❖ House price prediction

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

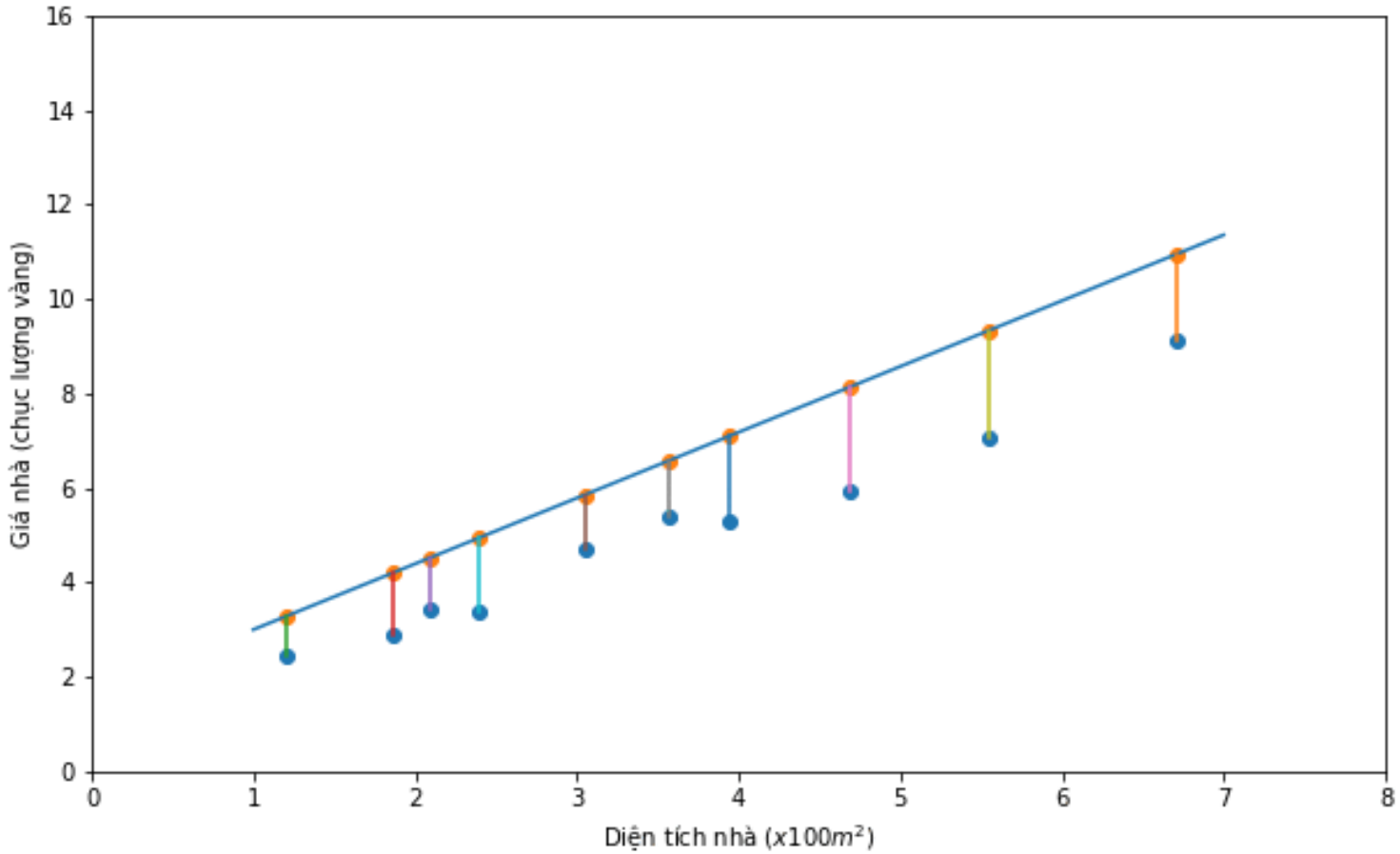
❖ House price prediction



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

```python
# 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
```


Thank you!