AI VIET NAM
@aivietnam.edu.vn

Genetic Algorithms

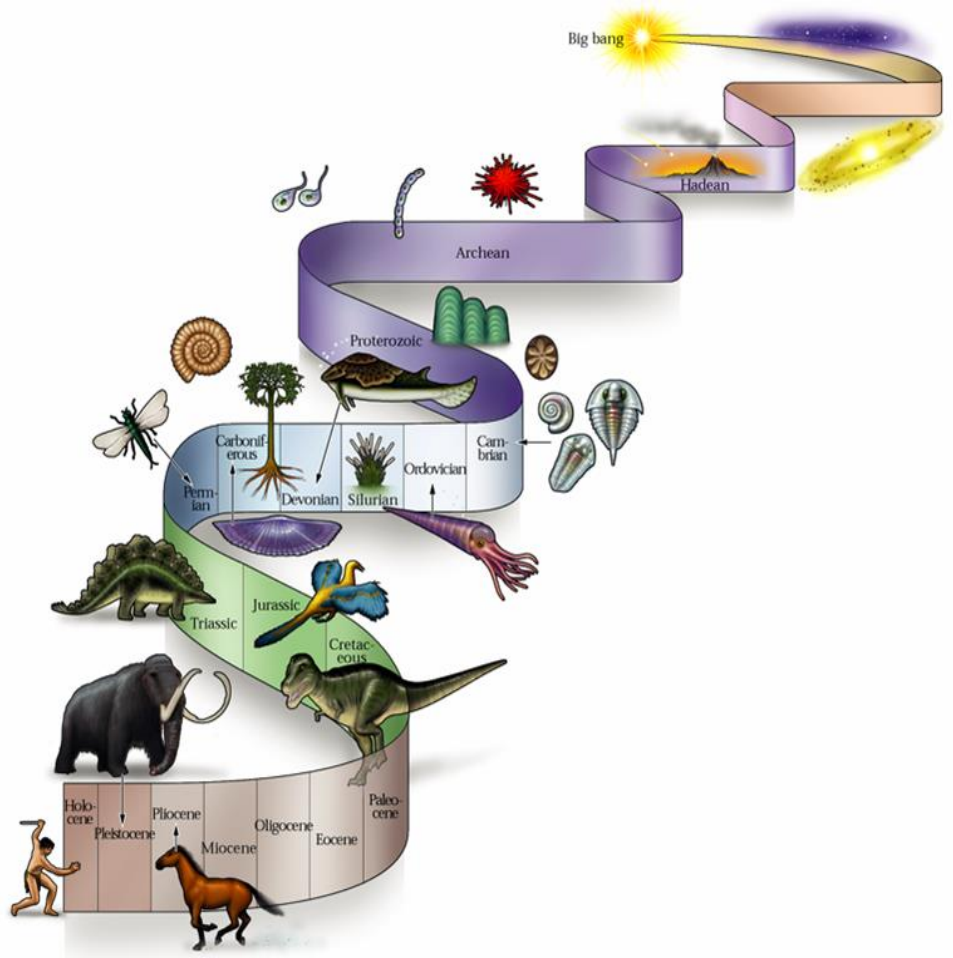
(Exercise)

Vinh Dinh Nguyen PhD in Computer Science

Outline

- GA Review
- How to Design Fitness Functions
- GA for Regression Problem
- GA for Classification Problem
- Assignment Discussion
- Summary

Principle of Evolutionary Algorithms

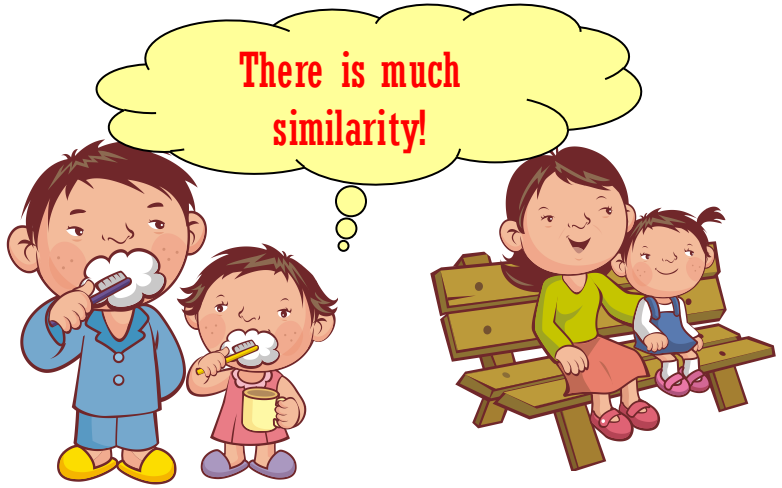


From Prof. Ahn

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



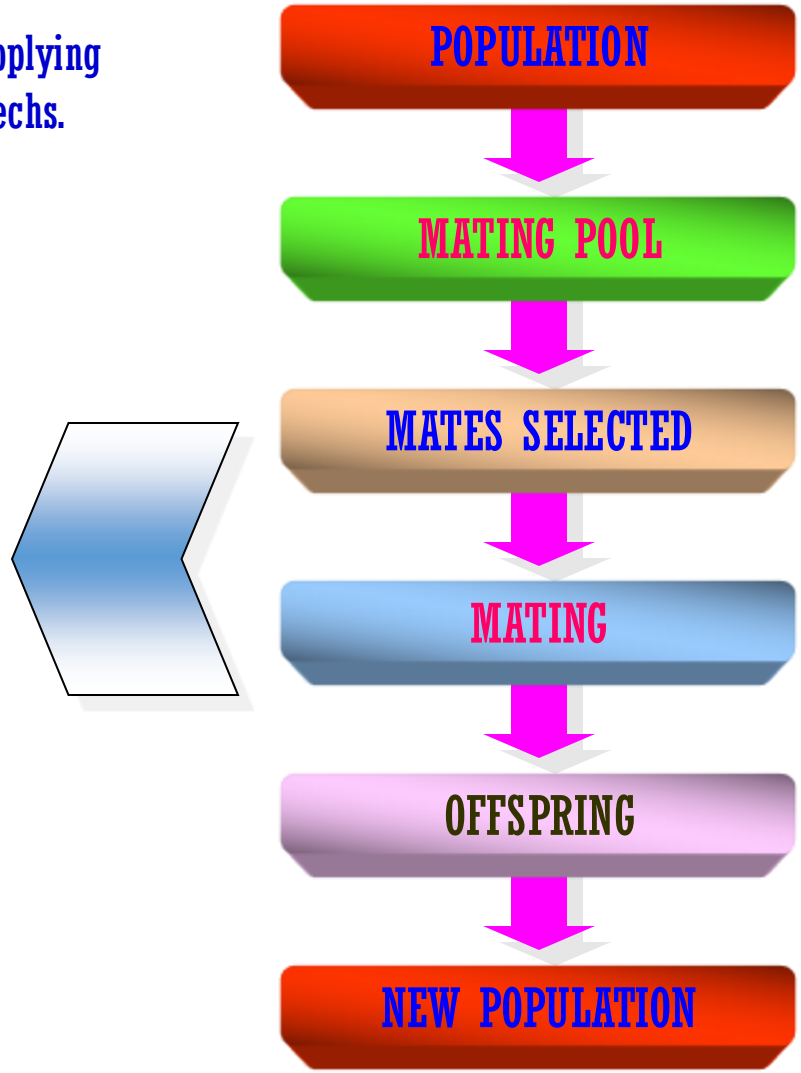
AI VIET NAM
@aivietnam.edu.vn

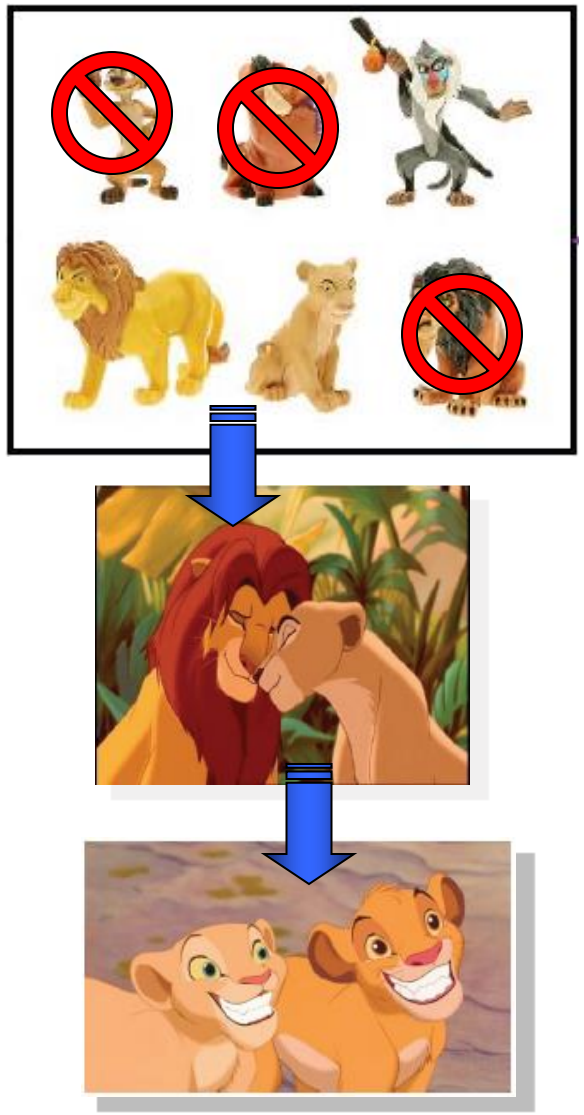
Evolutionary Algorithms: Principles

Lessons from Biological Evolution

Implications for applying to computing techs.

- Multiple
- Surviving
- Mixing
- Generation

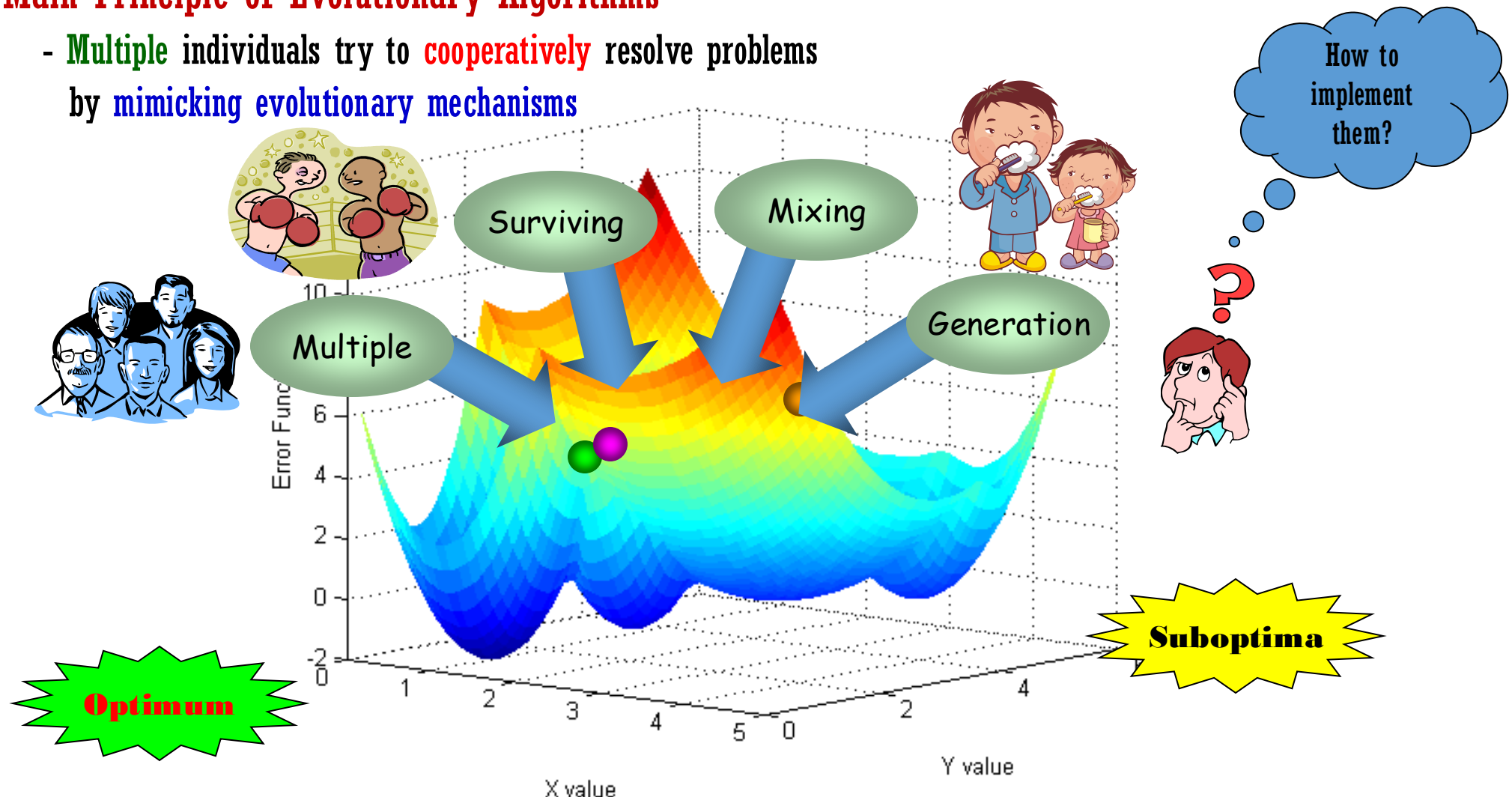




Evolutionary Algorithms: Principles

Main Principle of Evolutionary Algorithms

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



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)

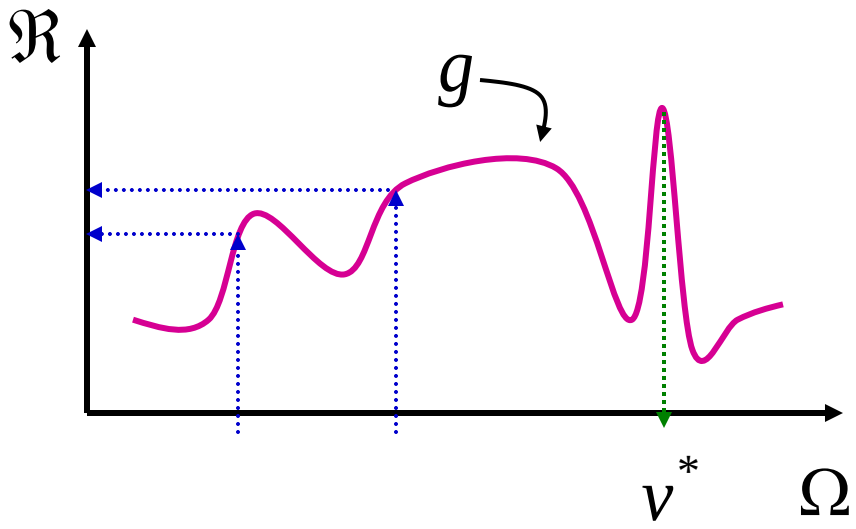
Formal Definition

For a search space Ω

There is a function $g : \Omega \mapsto \Re$

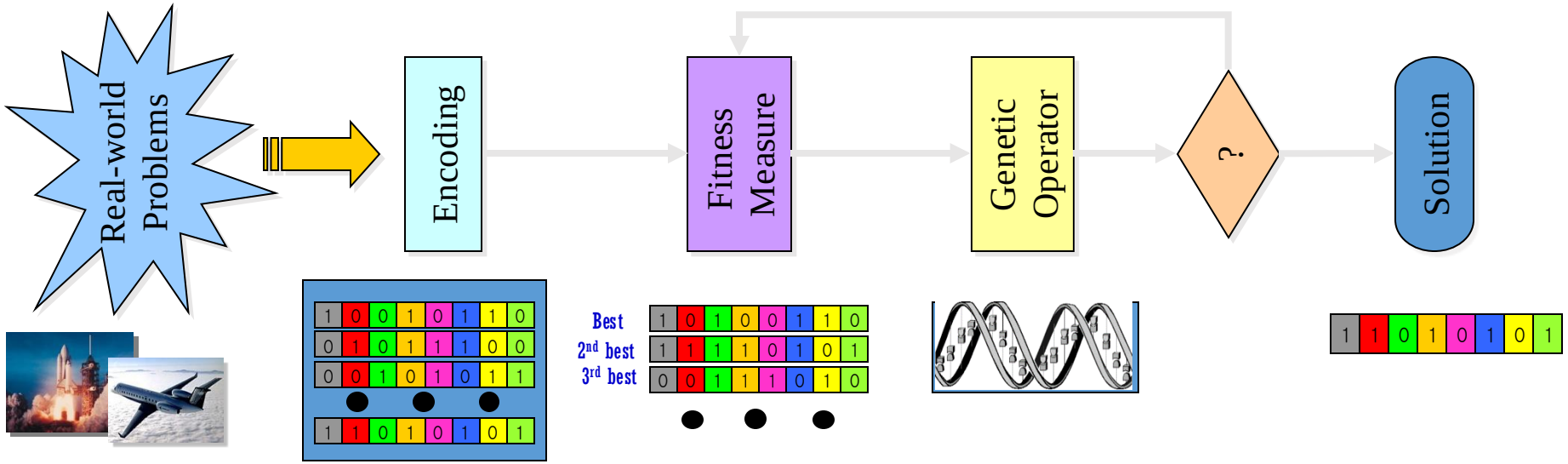
The task is to find $v^* = \arg\max_{v \in \Omega} g$

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



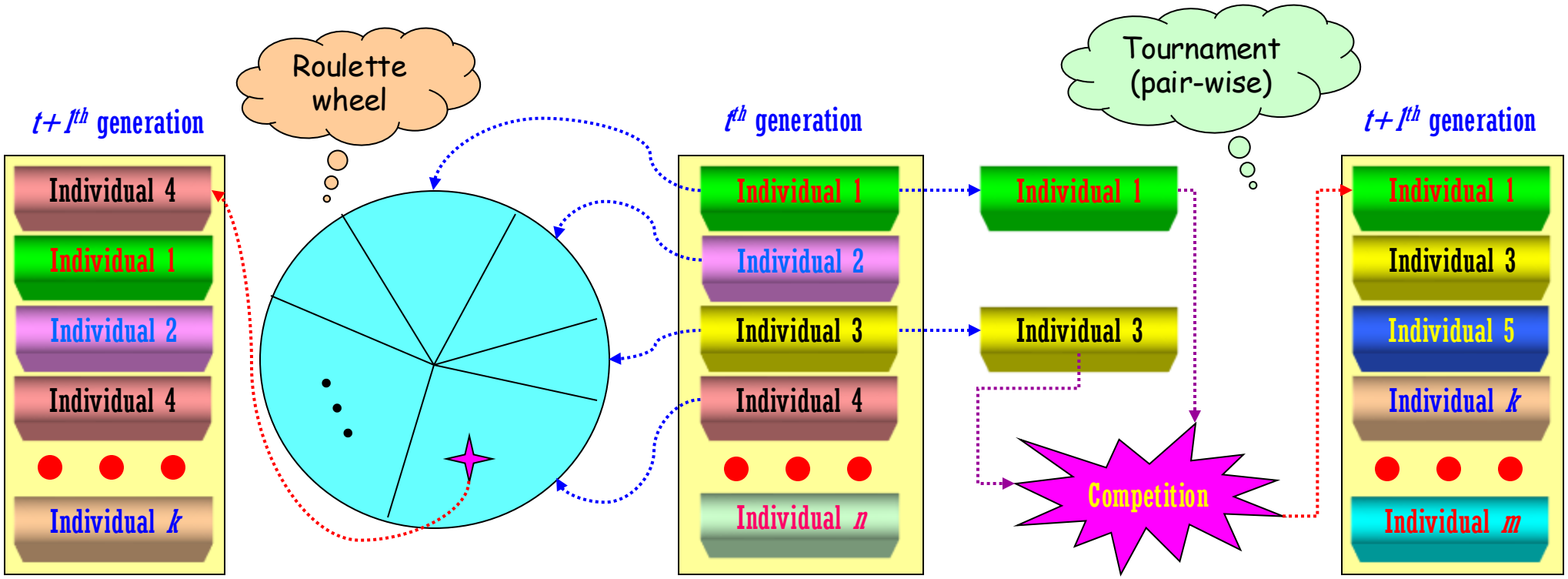
Genetic Algorithms

- **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

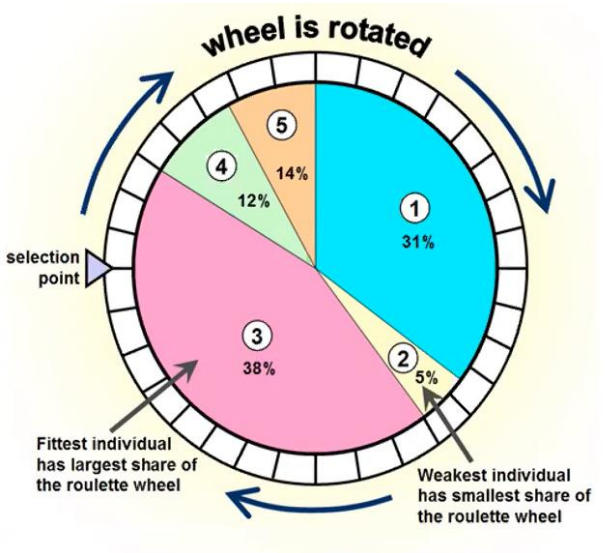
- **Roulette Wheel Selection**
  - The probability of selecting a given chromosome is proportional to its fitness
- **Tournament Selection**
  - Combine the fitness proportional concept with the random selection

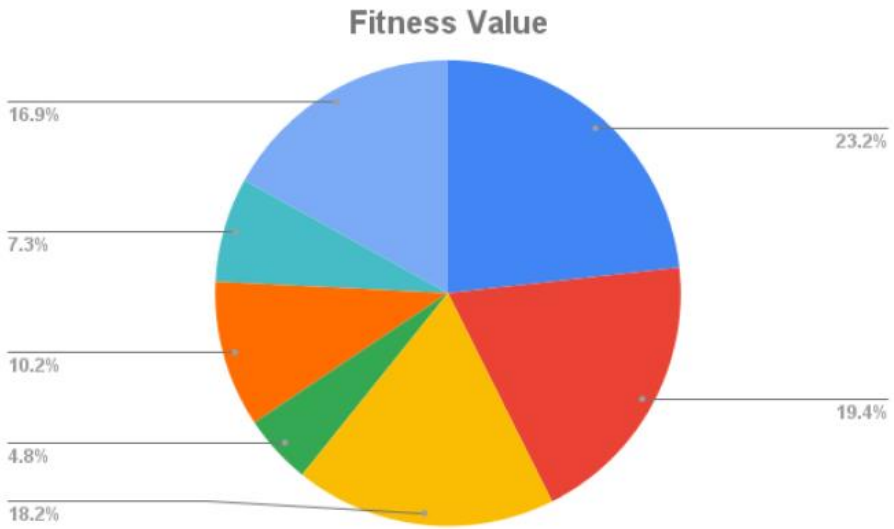


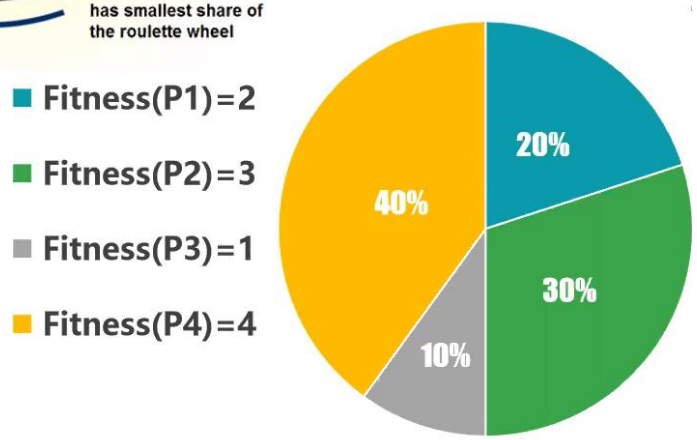
How to implement Roulette Wheel Selection

Roulette Wheel Selection

Chromosome	Fitness Value	Normalised fitness	Cumulative fitness
1	0.96	0.2324455206	0.2324455206
2	0.8	0.1937046005	0.4261501211
3	0.75	0.181598063	0.607748184
4	0.2	0.04842615012	0.6561743341
5	0.42	0.1016949153	0.7578692494
6	0.3	0.07263922518	0.8305084746
7	0.7	0.1694915254	1

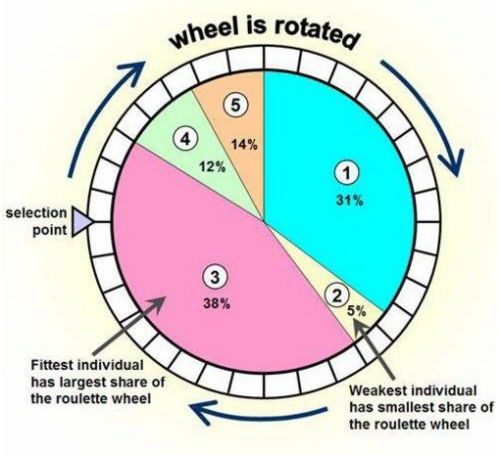






Roulette Wheel Selection

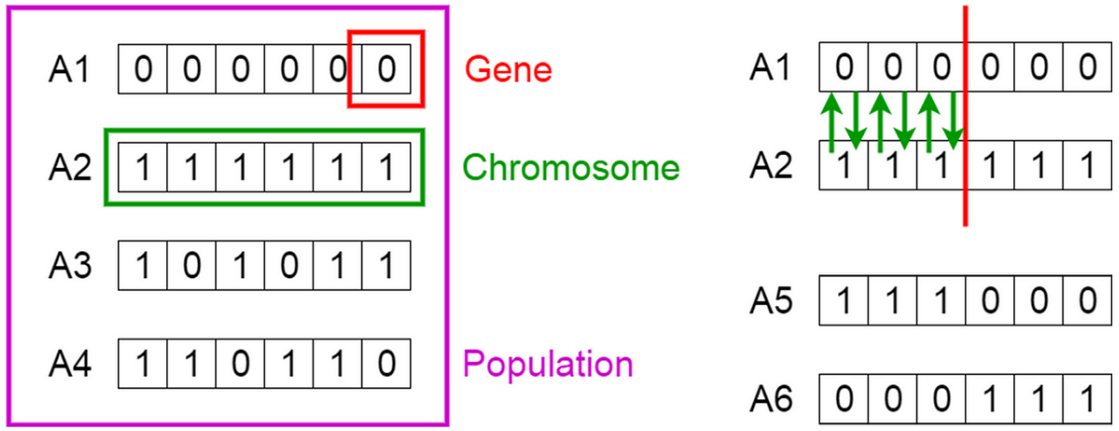
```python
import random

def roulette_wheel_selection(population, fitness_scores):
    total_fitness = sum(fitness_scores)
    relative_fitness = [f / total_fitness for f in fitness_scores]
    cumulative_probability = [sum(relative_fitness[:i+1]) for i in
range(len(relative_fitness))]

    rand = random.random()
    for i, cp in enumerate(cumulative_probability):
        if rand <= cp:
            return population[i]
```

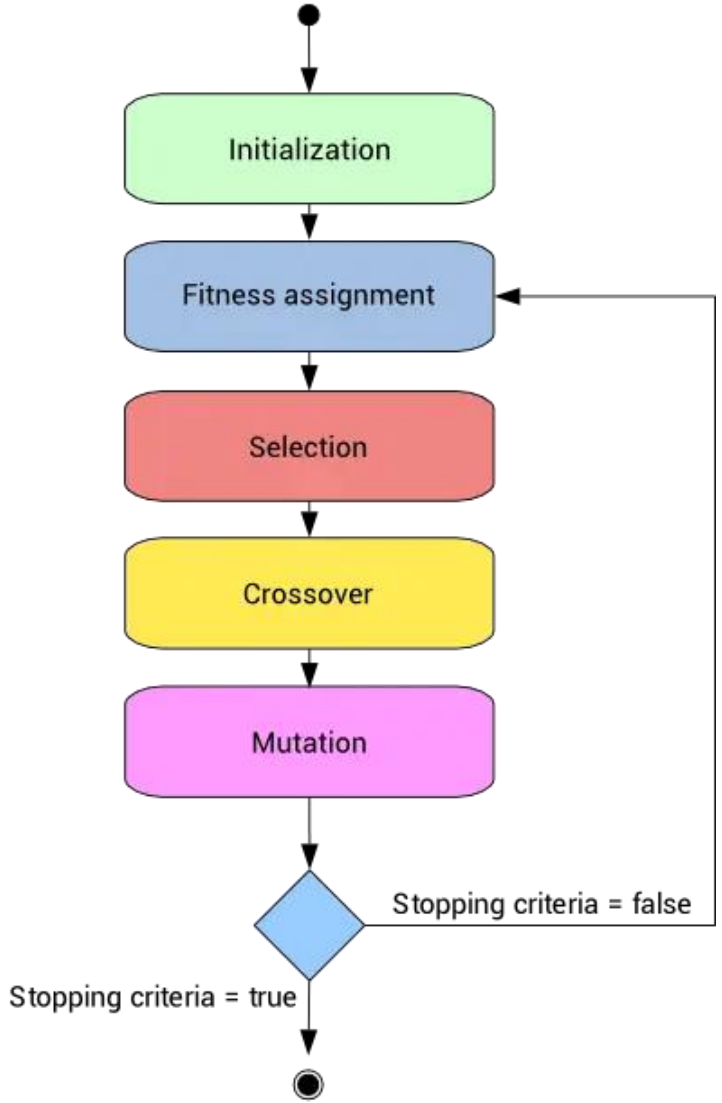


AI VIET NAM
@aivietnam.edu.vn

Genetic Algorithm

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)

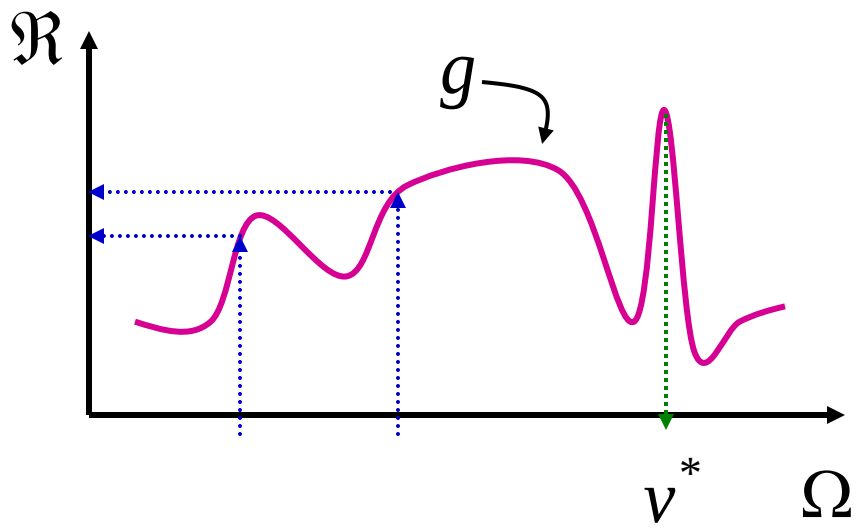
Formal Definition

For a search space Ω

There is a function $g : \Omega \mapsto \Re$

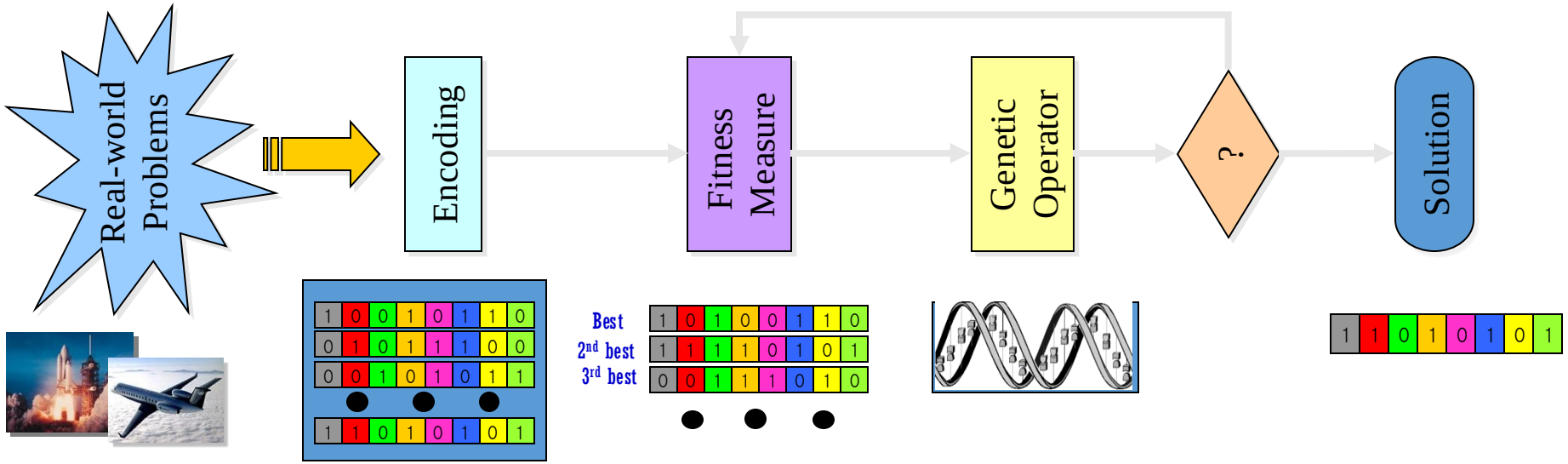
The task is to find $v^* = \arg\max_{v \in \Omega} g$

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



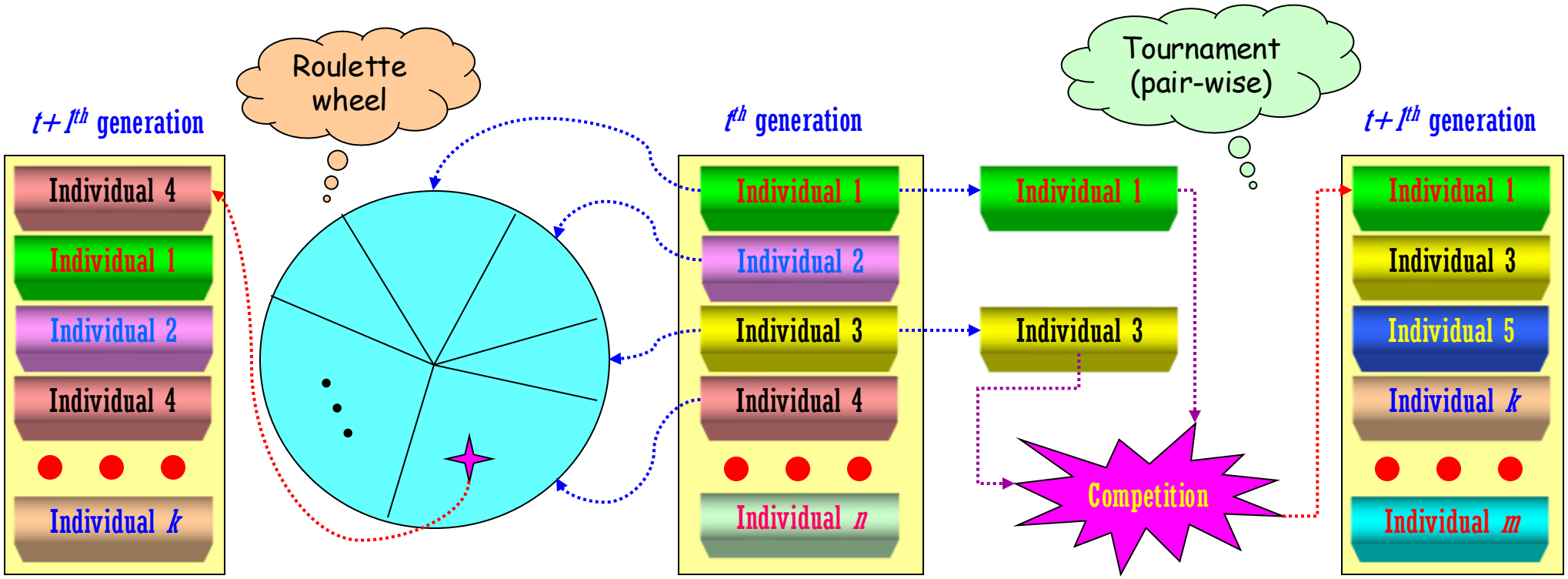
Genetic Algorithms

- **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

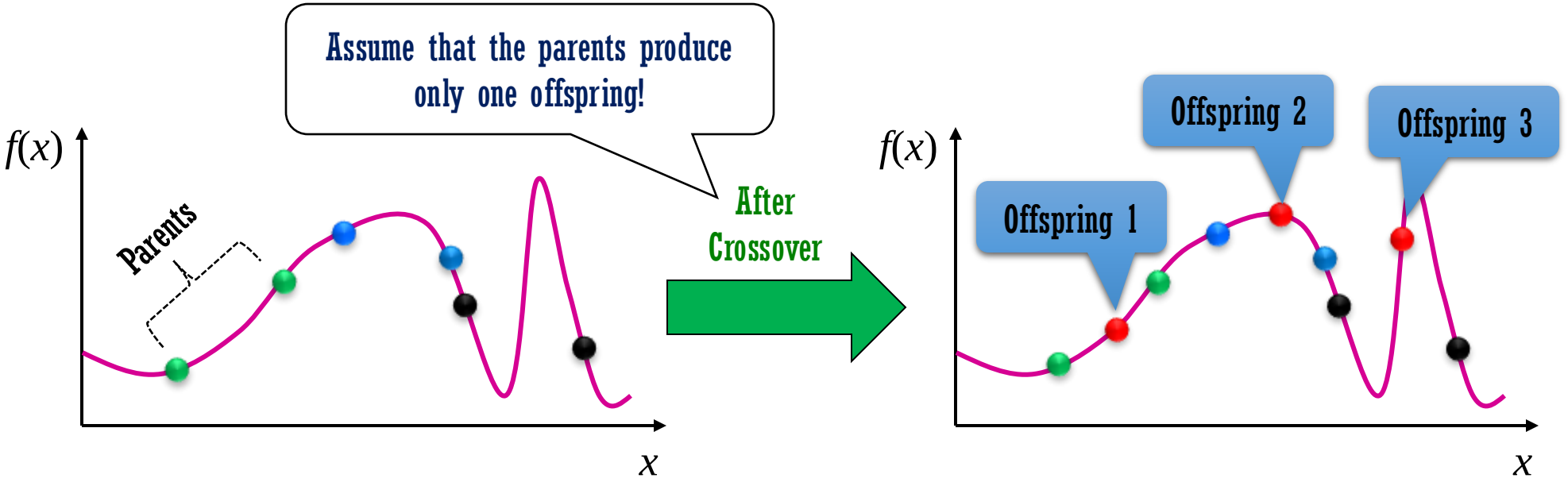
- **Roulette Wheel Selection**
  - The probability of selecting a given chromosome is proportional to its fitness
- **Tournament Selection**
  - Combine the fitness proportional concept with the random selection



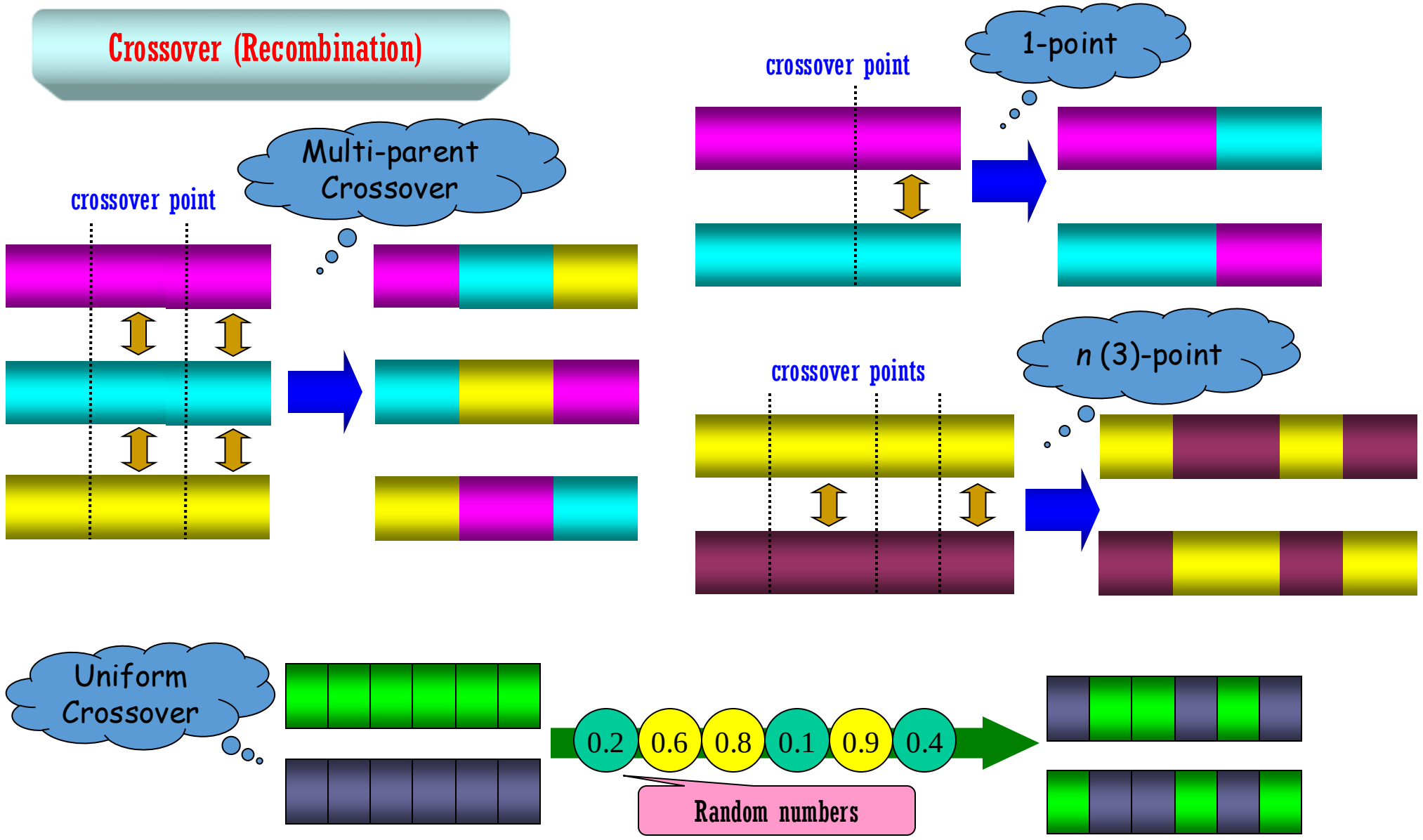
Genetic Algorithms

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.



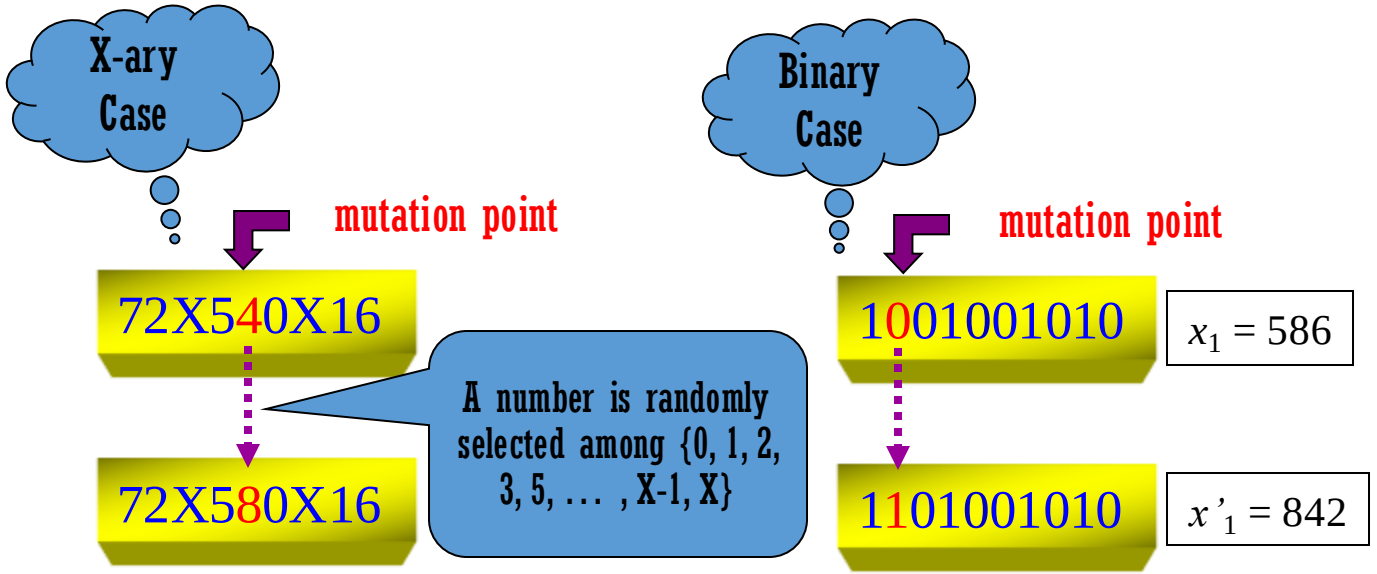
Genetic Algorithms

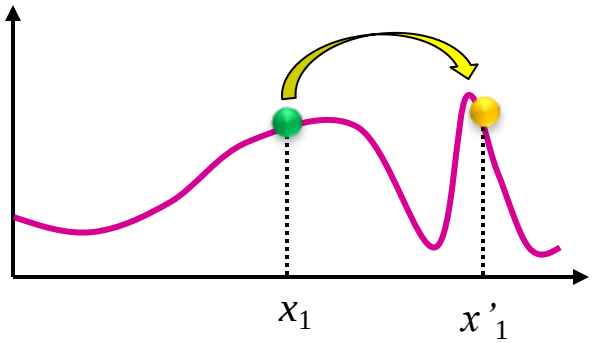


Genetic Algorithms

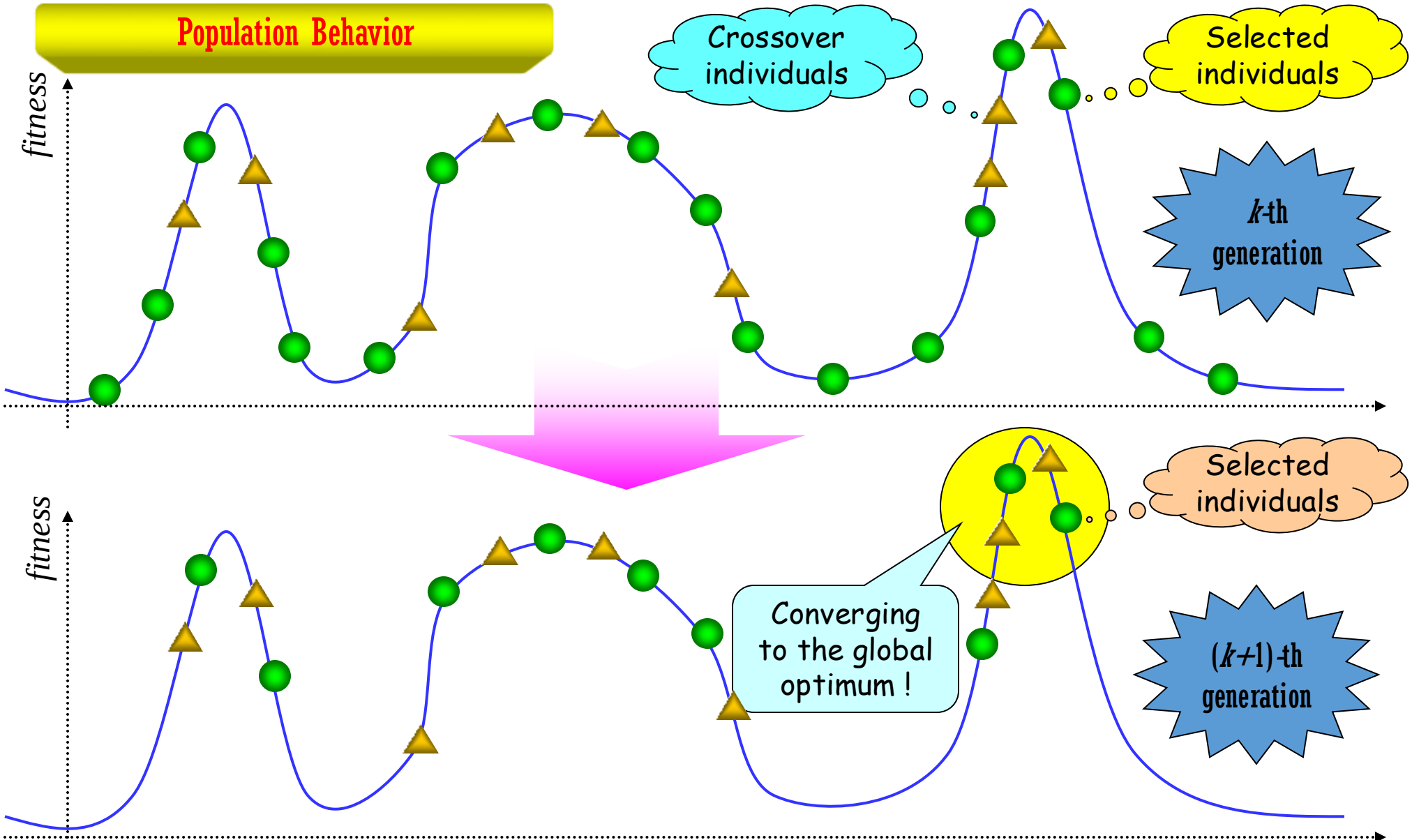
Mutation

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



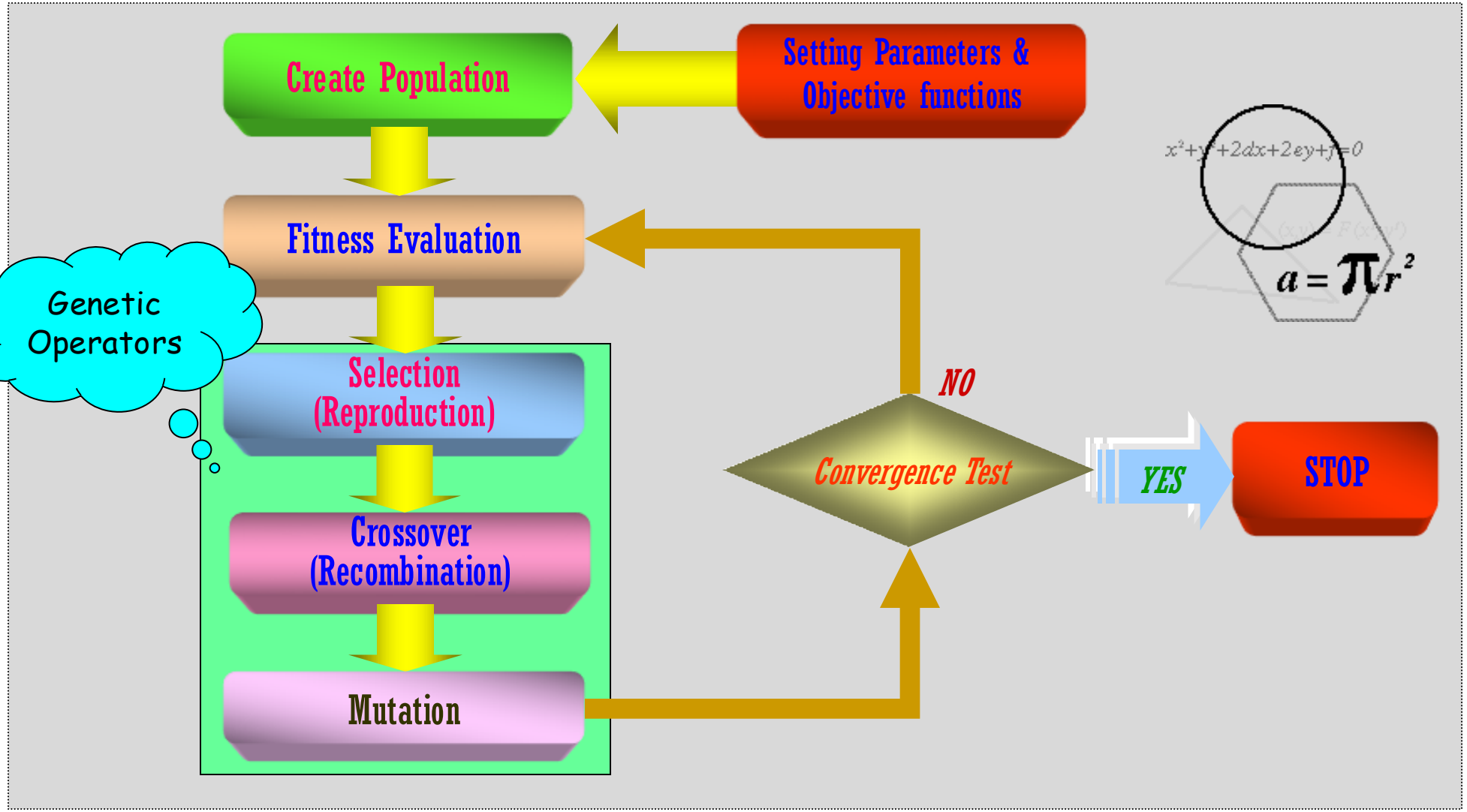


Genetic Algorithms

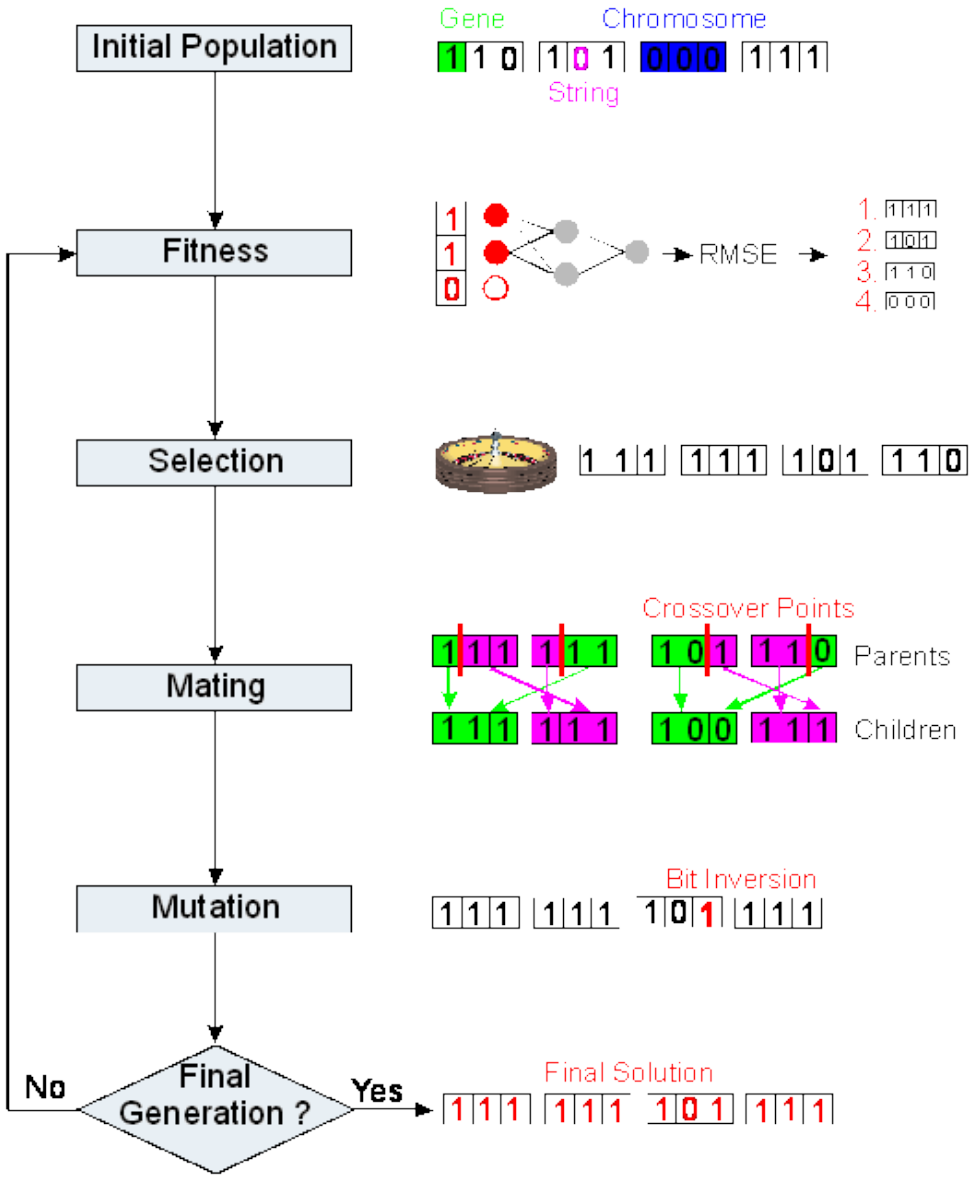


Genetic Algorithms

- Overall Procedures of GAs



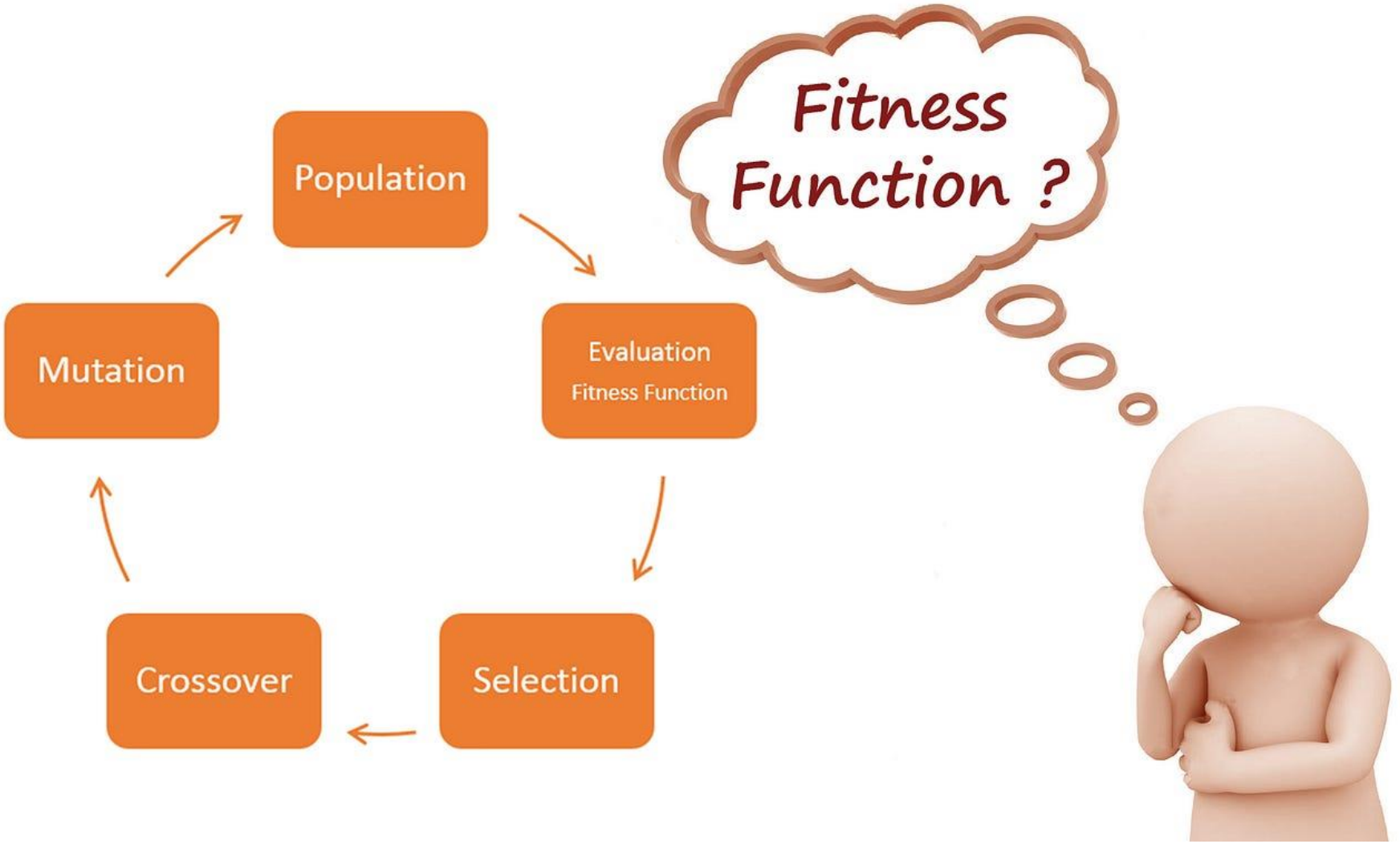
Genetic Algorithm



Outline

- GA Review
- How to Design Fitness Functions
- GA for Regression Problem
- GA for Classification Problem
- Assignment Discussion
- Summary

How to Design Fitness Function



Example 1— Generating Sequences

Given a set of 5 genes, which can hold one of the binary values 0 and 1, we have to come up with the sequence having all 1s. So we have to maximize the number of 1s as much as possible. This can be considered as an optimization problem.

The fitness function is considered as the number of 1s pres
five 1s, then it is having maximum fitness and solves our pr
if has the minimum fitness.

```
//Calculate fitness
public void calculateFitness() {
    fitness = 0;
    for (int i = 0; i < geneLength; i++) {
        if (genes[i] == 1) {
            ++fitness;
        }
    }
}
```

Example 2— Timetable Scheduling

Weekly schedule

Name:

Time / period	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday

Consider you are trying to come up with a weekly timetable for classes in a college for a particular batch. We have to arrange classes and come up with a timetable so that there are no clashes between classes. Here, our task is to search for the optimum timetable schedule.

Example 2— Timetable Scheduling

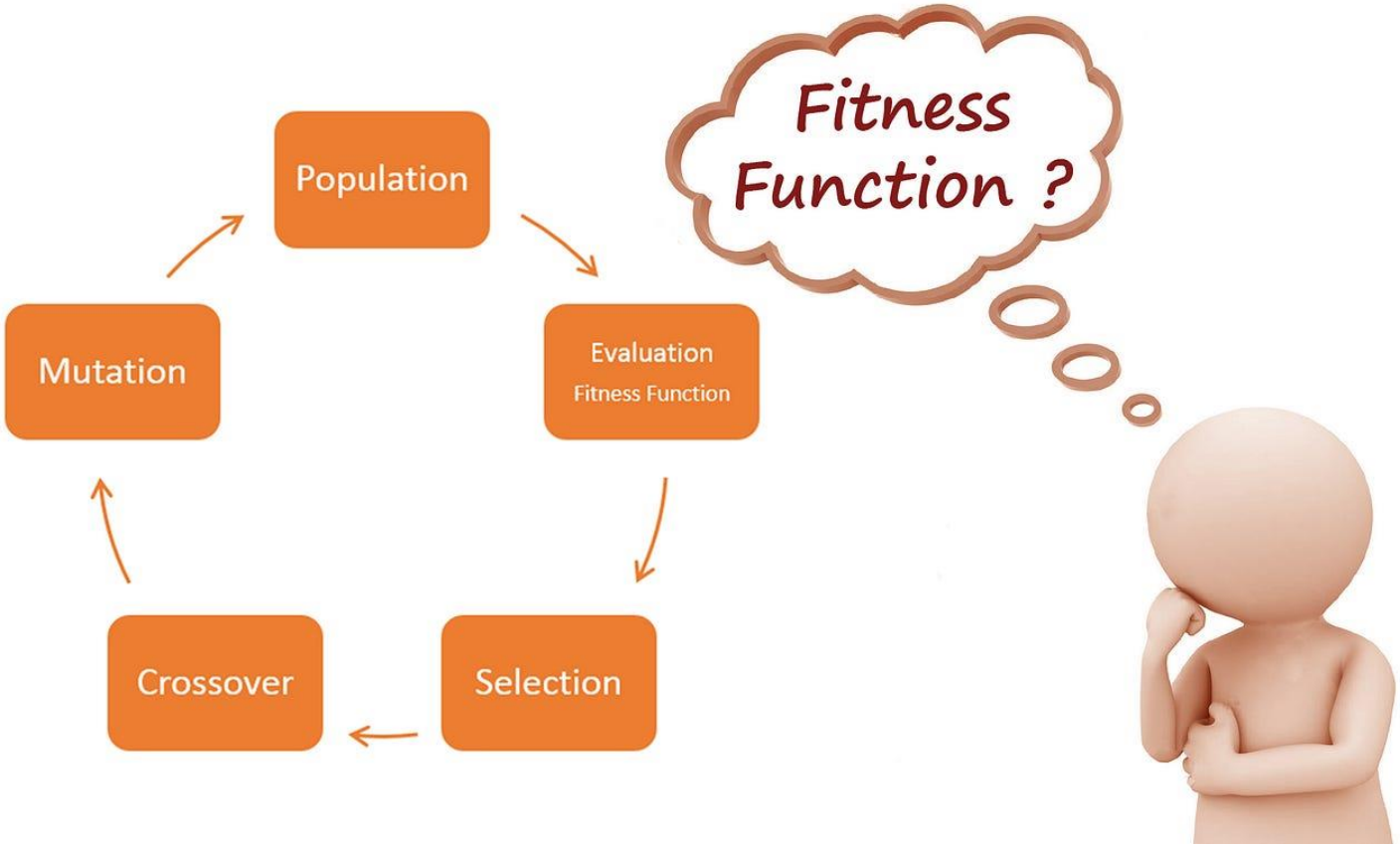
Weekly schedule

Name:

Time / period	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday

Since there should be no collisions among classes we should minimize the number of students having class conflicts. You can formulate the fitness function as the inverse of the number of students with class conflicts. Lesser the number of students with class conflicts, more fit the class is.

You have to play around with the problem, look in different ways and think about what kind of function you can use to check how good your



Example 3

❖ **Finding values for a set of variables which satisfy a given constraint**

Consider three variables x, y and z. The problem is to find the best set of values for x, y and z so that their total value is equal to a value t.

We have to reduce the sum x+y+z from deviating from t, i.e. |x + y + z — t| should be zero. Hence the fitness function can be considered as the inverse of |x + y + z - t|.

Review Questions

Khi nào thuật toán di truyền sẽ kết thúc?
(A) Số thế hệ tối đa đã được tạo ra
(B) Mức độ thích nghi thỏa đáng đã đạt được cho quần thể.
(C) Cả A & B
(D) Không cái nào trong số này

Review Questions

Let the population of chromosomes in genetic algorithm is represented in terms of binary number. The strength of fitness of a chromosome in decimal form, x, is given by

$$Sf(x) = \frac{f(x)}{\sum f(x)} \; where \;\; f(x) = x^2$$

The population is given by P where:

P = {(01101), (11000), (01000), (10011)}

The strength of fitness of chromosome (11000) is _____

Review Questions

SOLUTION

Outline

- GA Review
- How to Design Fitness Functions
- GA for Regression Problem
- GA for Classification Problem
- Assignment Discussion
- Summary

AI VIET NAM
@aivietnam.edu.vn

Genetic Algorithms for Regression

Date	USDJPY
5/26/17	111.33
5/25/17	111.84
5/24/17	111.49
5/23/17	111.78
5/22/17	111.3
5/19/17	111.26
5/18/17	111.49
5/17/17	110.83
5/16/17	113.12
5/15/17	113.79

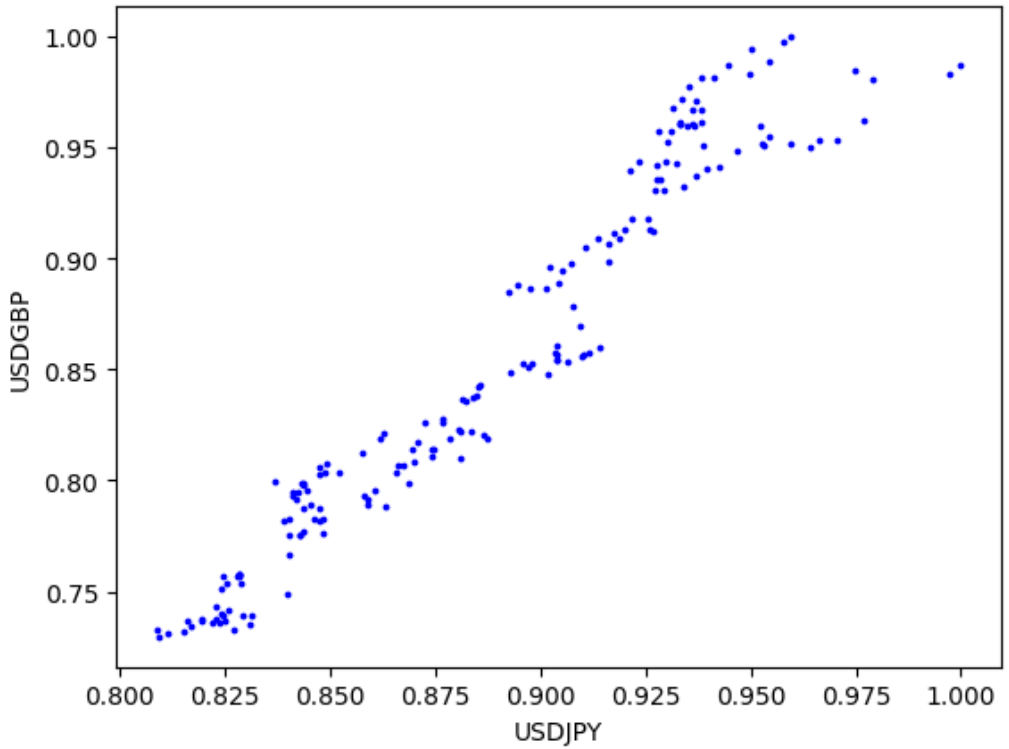
Exchange rate of USD and JPY

*We need to determine the relationship between **JPY** and **GBP***

Exchange rate of USD and GBP

Date	USDGBP
5/26/17	0.7809
5/25/17	0.7727
5/24/17	0.7708
5/23/17	0.7716
5/22/17	0.7693
5/19/17	0.7671
5/18/17	0.7726
5/17/17	0.7709
5/16/17	0.7741
5/15/17	0.7754

Genetic Algorithms for Regression



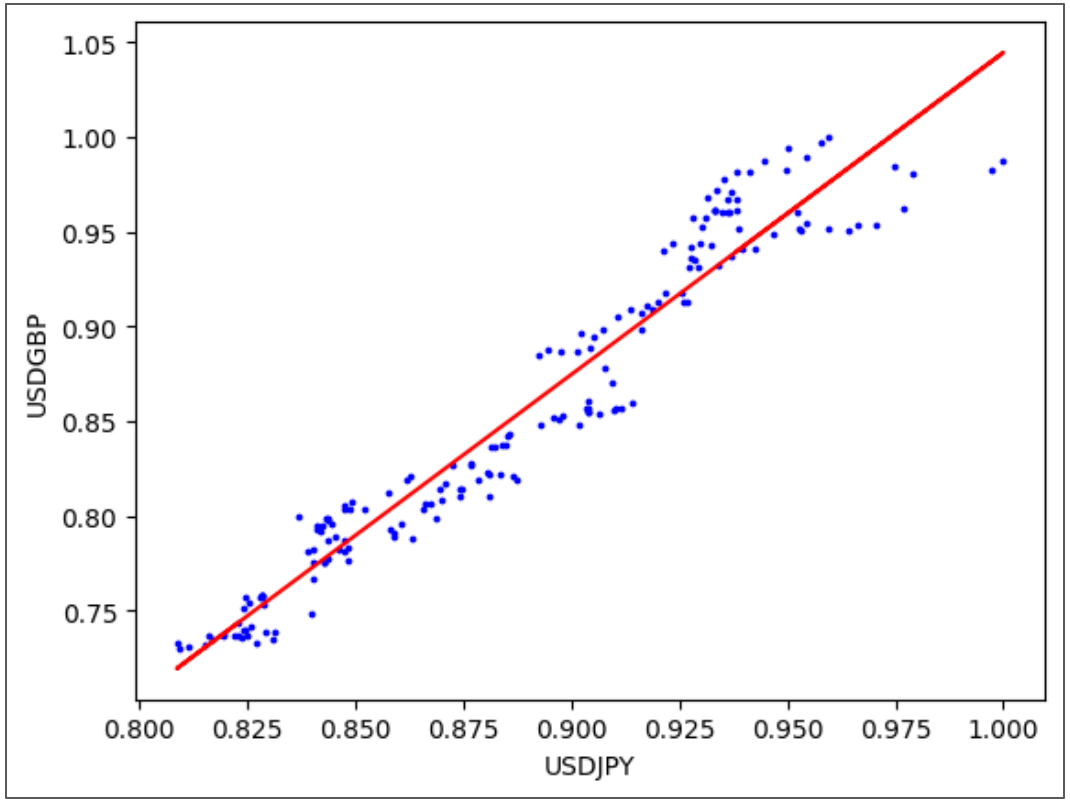

```
usd_gbp = pd.read_csv('USDGBP.csv', parse_dates=[0])
usd_jpy = pd.read_csv('USDJPY.csv', parse_dates=[0])
usd_dataset = usd_jpy.merge(right=usd_gbp, how='left', on='Date')
usd_dataset = usd_dataset.dropna(axis=0, how='any')
days = 180
X = preprocessing.normalize(usd_dataset['USDJPY'].values.reshape(-1, 1)[-days:], axis=0, norm='max')
y = preprocessing.normalize(usd_dataset['USDGBP'].values.reshape(-1, 1)[-days:], axis=0, norm='max')
plt.scatter(X, y, marker='o', color='blue', s=3)
plt.xlabel('USDJPY')
plt.ylabel('USDGBP')
```

AI VIET NAM
@aivietnam.edu.vn

Linear Regression for Regression

```
# Sử dụng thư viện LinearRegression từ Sklearn
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X, y)
print('Coefficient :', np.round(model.coef_[0][0],decimals=4),
      '\nIntercept :', np.round(model.intercept_[0], decimals=4))
```

```
# Vẽ phương trình tuyến tính với các hệ số tìm được từ SKlearn
line = model.predict(X)
plt.scatter(X, y, marker='o', color='blue', s=3)
plt.plot(X, line, color='red')
plt.xlabel('USDJPY')
plt.ylabel('USDGBP')
```

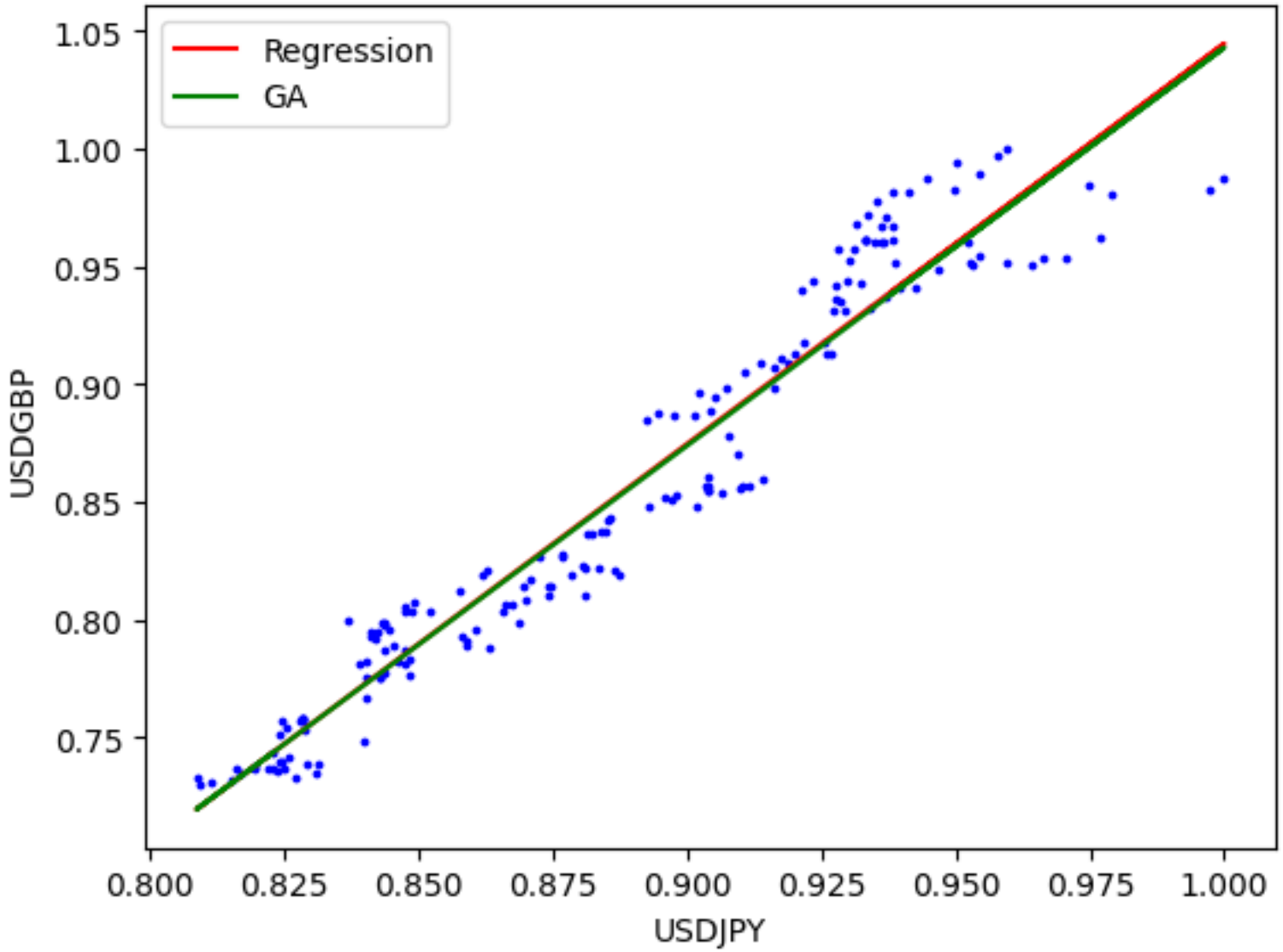


Genetic Algorithms for Regression

```python
# GA functions
def fitness(solution):
    '''Fitness function is the Ordinary Least Squares'''
    return 1/len(X)*np.sum(np.power(y-np.concatenate((X,np.ones((X.shape[0],1)))),axis=1).dot(solution.reshape(-1,1))

def genPopulation(size=400):
    '''Generate population of size individuals'''
    # Coefficient and intercept in the range [-1, 1] since all data has been normalized with maximum value
    solutions = ((np.random.rand(size, 2) * 2) - 1)
    return solutions

def selectBest(solutions):
    '''Return the best 20 solution for current population'''
    fitnesses = np.apply_along_axis(fitness, 1, solutions)
    return solutions[np.argsort(fitnesses, axis=0)[:20],:]
```

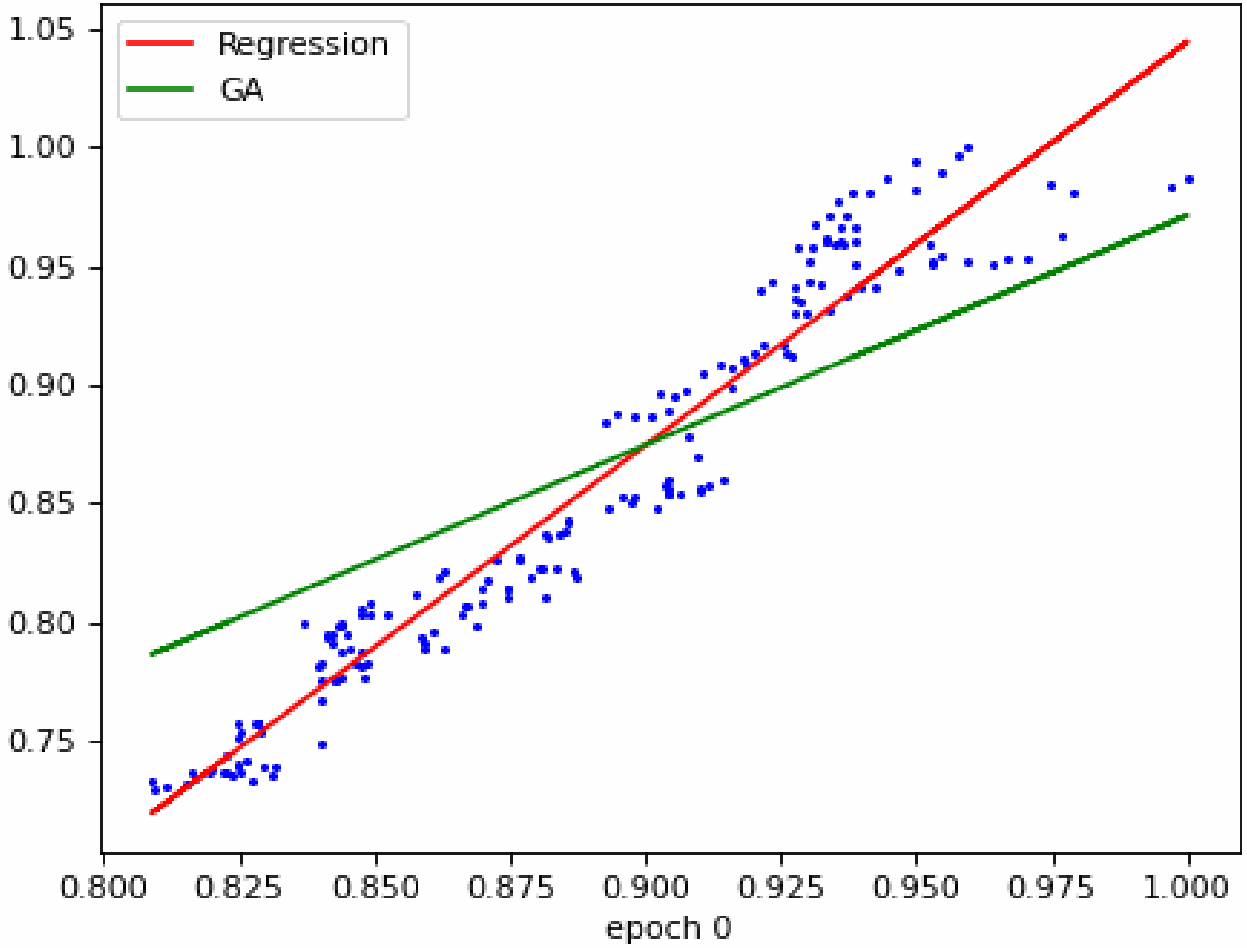
```python
def crossover(sol1, sol2):
    '''Perform crossover between two solutions by exchanging the intercepts and maintaining the coefficient'''
    offsprings = np.array([[sol1[0], sol2[1]]])
    offsprings = np.vstack((offsprings, [[sol2[0], sol1[1]]]))
    offsprings = np.vstack((offsprings, [sol2]))
    return offsprings

def mutation(sol):
    '''Perform mutation on the solution'''
    mutationProb = 0.15
    if np.random.rand() < mutationProb:
        sol = sol + (((np.random.rand(2) * 2) - 1) * 0.1)
    return sol
```

GA vs Linear Regression



AI VIET NAM
@aivietnam.edu.vn

Genetic Algorithms for Linear Regression



https://github.com/edervishaj/genetic-linear-regression

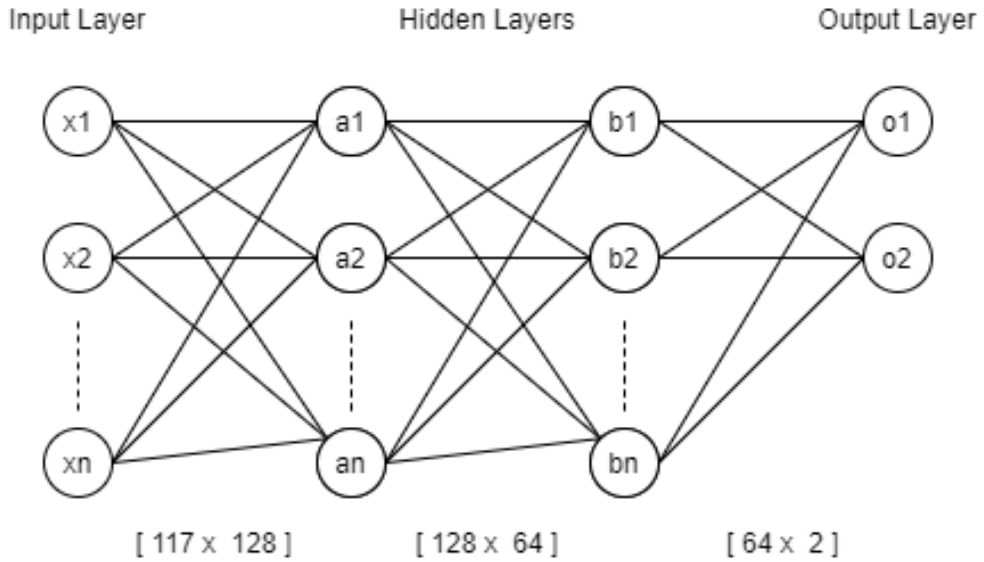
Outline

- GA Review
- How to Design Fitness Functions
- GA for Regression Problem
- GA for Classification Problem
- Assignment Discussion
- Summary

AI VIET NAM
@aivietnam.edu.vn

Mushroom Classification

Neural Network Solution

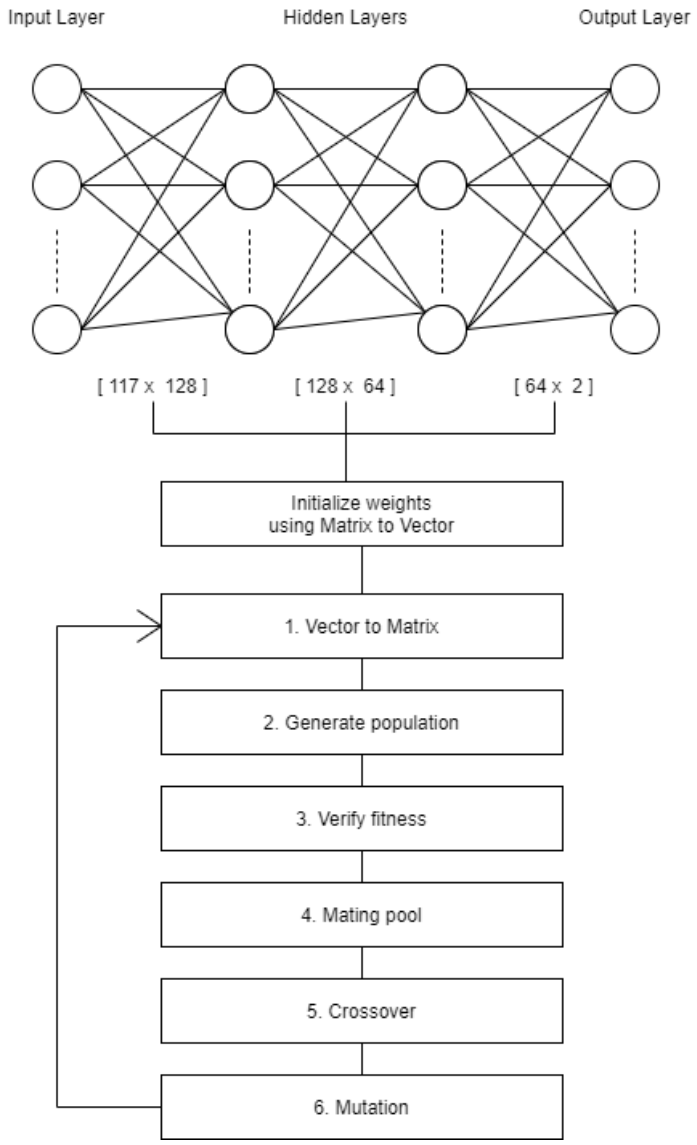



```
hidden_layer_1 = random.int( shape=(117, 128) )
hidden_layer_2 = random.int( shape=(128, 64) )
hidden_layer_output = random.int( shape=(64, 2) )

a = sigmoid ( np.dot( input, hidden_layer_1 ) )
b = sigmoid ( np.dot( a, hidden_layer_2 ) )
o = sigmoid ( np.dot( b, hidden_layer_output) )
```

AI VIET NAM
@aivietnam.edu.vn

Mushroom Classification

Genetic Algorithms



Outline

- GA Review
- How to Design Fitness Functions
- GA for Regression Problem
- GA for Classification Problem
- Assignment Discussion
- Summary

AI VIET NAM
@aivietnam.edu.vn

Assignment Discussion

Giới thiệu về bài tập :

Bài toán dự đoán Doanh thu bán hàng (sale) dựa vào các kênh Marketing khác nhau là một dạng bài toán kinh điển thường được dùng trong giảng dạy Machine Learning. Bộ dữ liệu có thể download tại LINK

Bộ dữ liệu bao gồm 200 mẫu, mỗi mẫu có 3 đặc trưng, là số tiền quảng cáo trên TV, Radio và Newspaper. Giá trị sale (doanh thu) được cung cấp tương ứng với chi phí của 3 kênh marketing được minh họa qua hình 1.

TV	Radio	Newspaper	Sales
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	12
151.5	41.3	58.5	16.5
180.8	10.8	58.4	17.9
8.7	48.9	75	7.2
57.5	32.8	23.5	11.8
120.2	19.6	11.6	13.2
8.6	2.1	1	4.8
199.8	2.6	21.2	15.6

Hình 1: Một vài sample data từ dữ liệu quảng cáo advertising.csv

AI VIET NAM
@aivietnam.edu.vn

Assignment Discussion

Để thuận tiện cho việc cài đặt, phương trình (1) có thể viết lại như sau:

$$sale = \theta_1 * TV + \theta_2 * Radio + \theta_3 * Newspaper + \theta_4 * 1.0 \quad (2)$$

Question 1: Hãy cho biết chiều dài của chromosome trong trường hợp này là bao nhiêu:

(A) 4

(B) 5

(C) 6

(D) 7

AI VIET NAM
@aivietnam.edu.vn

Assignment Discussion

Question 2: Hãy cho biết kết quả của đoạn chương trình sau:

```
features_X, _ = load_data_from_file()
print(features_X[:5,:])
```

```
a) [[  1.   230.1   37.8   69.2]
 [  1.    44.5   39.3   45.1]
 [  1.    17.2   45.9   69.3]
 [  1.   151.5   41.3   58.5]
 [  1.   180.8   10.8   58.4]]

b) [[  0.   230.1   37.8   69.2]
 [  0.    44.5   39.3   45.1]
 [  0.    17.2   45.9   69.3]
 [  0.   151.5   41.3   58.5]
 [  0.   180.8   10.8   58.4]]

c) [[  -1.   230.1   37.8   69.2]
 [  -1.    44.5   39.3   45.1]
 [  -1.    17.2   45.9   69.3]
 [  -1.   151.5   41.3   58.5]
 [  -1.   180.8   10.8   58.4]]

d) [[  2.   230.1   37.8   69.2]
 [ 2.    44.5   39.3   45.1]
 [  2.    17.2   45.9   69.3]
 [  2.   151.5   41.3   58.5]
 [  2.   180.8   10.8   58.4]]
```

AI VIET NAM
@aivietnam.edu.vn

Assignment Discussion

Question 3: Hãy cho biết kết quả của đoạn chương trình sau:

```
_, sales_Y = load_data_from_file()
print(sales_Y.shape)
```

```
_, sales_Y = load_data_from_file()
print(sales_Y.shape)

a) (200,1)
b) (200,)
c) (1,200)
d) (0,200)
```

AI VIET NAM
@aivietnam.edu.vn

Assignment Discussion

Bài tập 3 (kỹ thuật xây dựng fitness function để đánh giá fitness score cho từng chromosome): hãy hoàn thiện function **compute_fitness(individual)** để tính fitness values của một chromosome bằng cách ngịch đảo giá trị của hàm loss theo công thức *fitness = 1 / (loss + 1)*. Hàm **compute_fitness(individual)** đánh giá độ tốt của một cá thể. Giá trị càng lớn thì cá thể càng tốt.

```
features_X, sales_Y = load_data_from_file()

def compute_loss(individual):
    theta = np.array(individual)
    y_hat = features_X.dot(theta)
    loss  = np.multiply((y_hat-sales_Y), (y_hat-sales_Y)).mean()
    return loss

def compute_fitness(individual):
    loss = compute_loss(individual, features_X, sales_Y)
    fitness_value =  0

    # ************ your code here ************

    return. fitness_value
```

Question 4: Hãy cho biết kết quả của đoạn chương trình sau:

```
features_X, sales_Y = load_data_from_file()
individual = [4.09, 4.82, 3.10, 4.02]
fitness_score = compute_fitness(individual)
print(fitness_score)
```

(A) 2.018e-06

(B) 3.018e-06

(C) 1.018e-06

(D) 4.018e-06

AI VIET NAM
@aivietnam.edu.vn

Assignment Discussion

Bài tập 4 (kỹ thuật thực hiện crossover giữa 2 chromosomes): hãy hoàn thiện function **crossover(individual1, individual2, crossover_rate = 0.9)** để thực hiện crossover giữa 2 individual (chromosome). Bước này, thực hiện việc lai tạo (trao đổi gen) giữa 2 individual với tỉ lệ *crossover_rate*

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

    ############### Your code here ##############

    return individual1_new, individual2_new
```

Question 5: Hãy cho biết kết quả của đoạn chương trình sau:

```
#question 5
individual1 = [4.09, 4.82, 3.10, 4.02]
individual2 = [3.44, 2.57, -0.79, -2.41]

individual1, individual2 = crossover(individual1, individual2, 2.0)
print("individual1: ", individual1)
print("individual2: ", individual2)

(A)
individual1 = [4.09, 4.82, 3.10, 4.02]
individual2 = [3.44, 2.57, -0.79, -2.41]

(B)
individual1:  [7.44, 2.57, -0.79, -2.41]
individual2:  [4.09, 4.82, 3.1, 4.02]

(C)
individual1:  [5.44, 2.57, -0.79, -2.41]
individual2:  [4.09, 4.82, 3.1, 4.02]

(D)
individual1:  [3.44, 2.57, -0.79, -2.41]
individual2:  [4.09, 4.82, 3.1, 4.02]
```

AI VIET NAM
@aivietnam.edu.vn

Assignment Problem

Bài tập 5 (kỹ thuật mutation với chromosome): hãy hoàn thiện function **mutate(individual, mutation_rate = 0.05)** để thực hiện việc đột biến cho một cá thể với tỉ lệ đột biến là mutation_rate.

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

    # ******** Your code here ******************

    return individual_m
```

Question 6: Hãy cho biết kết quả của đoạn chương trình sau:

```
before_individual = [4.09, 4.82, 3.10, 4.02]
after_individual = mutate(individual, mutation_rate = 2.0)
print(before_individual == after_individual)

(A) False
(B) True
```

Assignment Problem

Bài tập 8 (Kỹ thuật tạo ra quần thể (population) mới: Hãy hoàn thiện function **create_new_population()** để tạo ra quần thể mới đựa trên các bước selection, crossover và mutation. Lưu ý rằng chúng ta sẽ sử dụng giải thuật Elitist algorithms để đảm bảo rằng **elitism** cá thể tốt nhất không bị loại bỏ, bằng cách chuyển chúng trực tiếp sang thế hệ tiếp theo.

```
def create_new_population(old_population, features_X, sales_Y, elitism=2, gen=1):
    m = len(old_population)
    sorted_population = sorted(old_population, key=compute_fitness)


    if gen%1 == 0:
        print("Best loss:", compute_loss(sorted_population[m-1]), "with chromsome: ",
    sorted_population[m-1])

    new_population = []


   # **************** your code here ****************


    return new_population, compute_loss(sorted_population[m-1])
```

Question 7: Hãy cho biết kết quả của đoạn chương trình sau:

```
individual1 = [4.09, 4.82, 3.10, 4.02]
individual2 = [3.44, 2.57, -0.79, -2.41]
old_population = [individual1, individual2]
new_population, _ = create_new_population(old_population, elitism=2, gen=1)



(A) Best loss: 123415.051528805 with chromsome:  [3.44, 2.57, -0.79, -2.41]
(B) Best loss: 123415.051528805 with chromsome:  [4.09, 4.82, 3.10, 4.02]
```

Outline

- Genetic Algorithm Review
- How to design Fitness Function
- Genetic Algorithm for Regression
- Genetic Algorithm for Classification
- Assignment Problem and Solution
- Case study: Vehicle Detection using GA
- Latest Trends in GA

AI VIET NAM
@aivietnam.edu.vn

GA for Vehicle Detection

A Fast Evolutionary Algorithm for Real-Time Vehicle Detection

Publisher: IEEE | Cite This | PDF

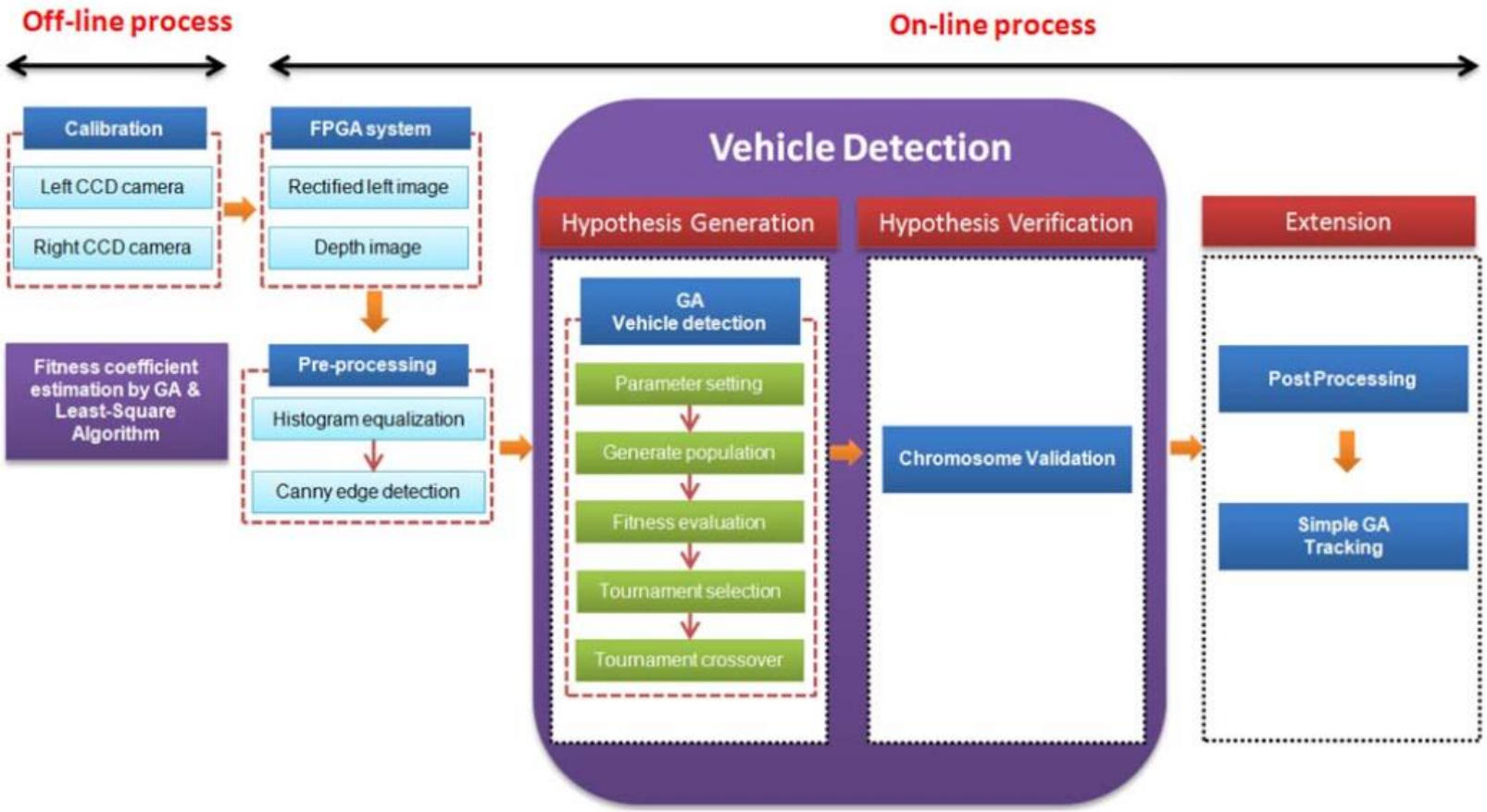
Vinh Dinh Nguyen ; Thuy Tuong Nguyen ; Dung Duc Nguyen ; Sang Jun Lee ; Jae Wook Jeon All Authors

Abstract:
The evolutionary algorithm (EA) is an effective method for solving various problems because it can search through very large search spaces and can quickly come to nearly optimal solutions. However, existing EA-based methods for vehicle detection cannot achieve high performance because their fitness functions depend on sensitive information, such as edge or color information on the preceding vehicle. This paper focuses on improving the performance of existing evolutionary-based methods for vehicle detection by introducing an effective fitness function that can more accurately capture a vehicle's information by combining a disparity map, edge information, and the position and motion of the preceding vehicle. The proposed method can detect multiple vehicles by using a turn-back genetic algorithm (GA) and can prevent false detection by using motion detection. Our fitness function is designed in a typical manner along with the fitness parameters. These parameters are usually selected using heuristic methods, making the choice of optimal parameters difficult. Therefore, this paper proposes a new approach to estimating optimal fitness parameters using EA and the least squares method. Robustness testing showed that the proposed method provides detection rate (DR) results close to those obtained using a state-of-the-art system and outperforms other dominant vehicle-detection-based EAs.

V. D. Nguyen, T. T. Nguyen, D. D. Nguyen, S. J. Lee and J. W. Jeon, "A Fast Evolutionary Algorithm for Real-Time Vehicle Detection," in IEEE Transactions on Vehicular Technology, vol. 62, no. 6, pp. 2453-2468, July 2013, doi: 10.1109/TVT.2013.2242910.

AI VIET NAM
@aivietnam.edu.vn

GA for Vehicle Detection



V. D. Nguyen, T. T. Nguyen, D. D. Nguyen, S. J. Lee and J. W. Jeon, "A Fast Evolutionary Algorithm for Real-Time Vehicle Detection," in IEEE Transactions on Vehicular Technology, vol. 62, no. 6, pp. 2453-2468, July 2013, doi: 10.1109/TVT.2013.2242910.

AI VIET NAM
@aivietnam.edu.vn

GA for Vehicle Detection

Fig. 3. Proposed system with two charge-coupled device cameras, a chess-board, an FPGA board, and a PC for stereo camera calibration.

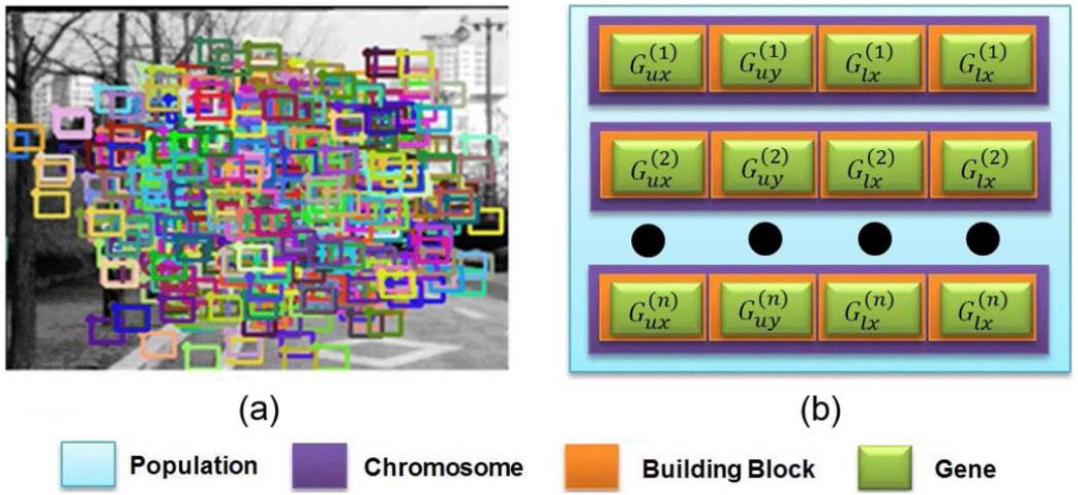


Fig. 4. (a) Initial population based on a Gaussian process. (b) Chromosome structure.

V. D. Nguyen, T. T. Nguyen, D. D. Nguyen, S. J. Lee and J. W. Jeon, "A Fast Evolutionary Algorithm for Real-Time Vehicle Detection," in IEEE Transactions on Vehicular Technology, vol. 62, no. 6, pp. 2453-2468, July 2013, doi: 10.1109/TVT.2013.2242910.

AI VIET NAM
@aivietnam.edu.vn

GA for Vehicle Detection

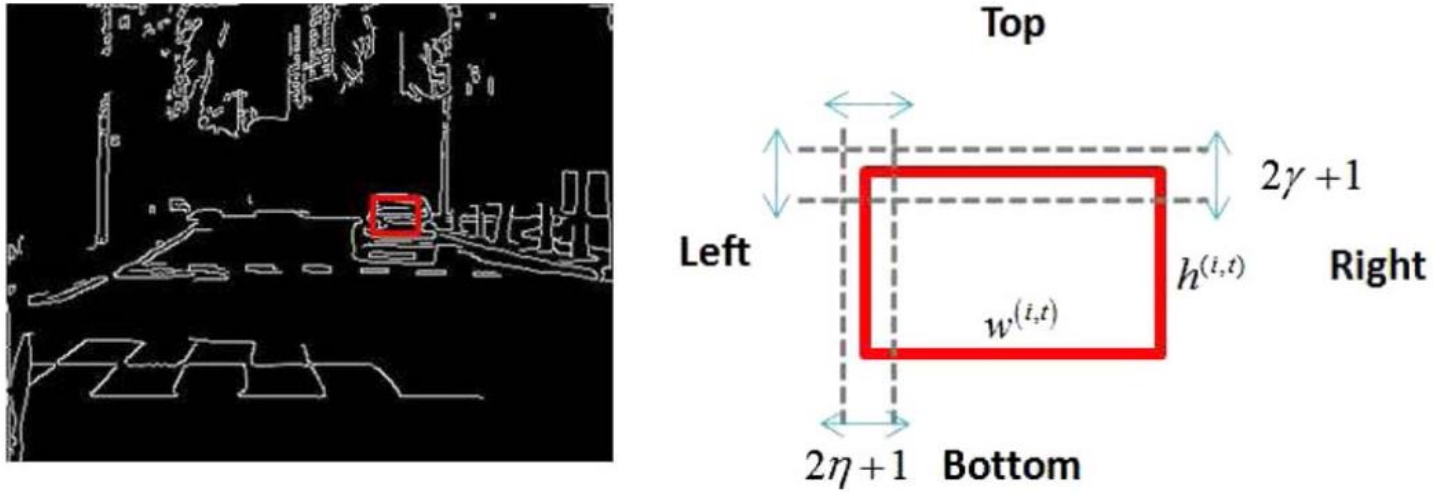


Fig. 5. (Left) Edge view of a sample chromosome. (Right) Chromosome features based on edge information.

V. D. Nguyen, T. T. Nguyen, D. D. Nguyen, S. J. Lee and J. W. Jeon, "A Fast Evolutionary Algorithm for Real-Time Vehicle Detection," in IEEE Transactions on Vehicular Technology, vol. 62, no. 6, pp. 2453-2468, July 2013, doi: 10.1109/TVT.2013.2242910.

AI VIET NAM
@aivietnam.edu.vn

GA for Vehicle Detection

$$F^{(i,t)} = -C_{\text{de}}G_{\text{de}}^{(i,t)} + C_l G_l^{(i,t)} + C_r G_r^{(i,t)} + C_{\text{to}}G_{\text{to}}^{(i,t)} + C_b G_b^{(i,t)} + C_d G_d^{(i,t)} + C_{\text{rd}}G_{\text{rd}}^{(i,t)} + C_m G_m^{(i,t)} \quad (10)$$

$$G_{\text{de}}^{(i,t)} = \frac{1}{w^{(i,t)}h^{(i,t)}}\left(w^{(i,t)}h^{(i,t)} - \sum_{j=0}^{h^{(i,t)}}\sum_{k=0}^{w^{(i,t)}} d(j,k,t)\right)$$

$$G_{\text{rd}}^{(i,t)} = \frac{1 - \sqrt{\left(G_{ux}^{(i,t)} - x\right)^2 + \left(G_{uy}^{(i,t)} - y\right)^2}}{\sqrt{x^2+y^2}}$$

$$G_m^{(i,t)} = \frac{1}{w^{(i,t)}h^{(i,t)}}\sum_{j=0}^{h^{(i)}}\sum_{k=0}^{w^{(i)}} B\left(\left|I^{(t)}(j,k) - I^{(t-1)}(j,k)\right|\right)$$

$$B(I) = \begin{cases} 1, & \text{if } I \geq \omega \\ 0, & \text{if } I < \omega \end{cases} \quad (9)$$

$$G_l^{(i,t)} = \frac{1}{(2\eta+1)h^{(i,t)}}\sum_{j=0}^{h^{(i,t)}}\sum_{k=-\eta}^{\eta} I_{\text{eg}}\left(G_{uy}^{(i,t)}+j, G_{ux}^{(i,t)}+k\right)$$

$$G_r^{(i,t)} = \frac{1}{(2\eta+1)h^{(i,t)}}\sum_{j=0}^{h^{(i,t)}}\sum_{k=-\eta}^{\eta} I_{\text{eg}}\left(G_{uy}^{(i,t)}+j, G_{lx}^{(i,t)}+k\right)$$

$$G_{\text{to}}^{(i,t)} = \frac{1}{(2\gamma+1)w^{(i,t)}}\sum_{j=-\gamma}^{\gamma}\sum_{k=0}^{w^{(i,t)}} I_{\text{eg}}\left(G_{uy}^{(i,t)}+j, G_{ux}^{(i,t)}+k\right)$$

$$G_b^{(i,t)} = \frac{1}{(2\gamma+1)w^{(i,t)}}\sum_{j=-\gamma}^{\gamma}\sum_{k=0}^{w^{(i,t)}} I_{\text{eg}}\left(G_{ly}^{(i,t)}+j, G_{ux}^{(i,t)}+k\right)$$

V. D. Nguyen, T. T. Nguyen, D. D. Nguyen, S. J. Lee and J. W. Jeon, "A Fast Evolutionary Algorithm for Real-Time Vehicle Detection," in IEEE Transactions on Vehicular Technology, vol. 62, no. 6, pp. 2453-2468, July 2013, doi: 10.1109/TVT.2013.2242910.

AI VIET NAM
@aivietnam.edu.vn

GA for Vehicle Detection

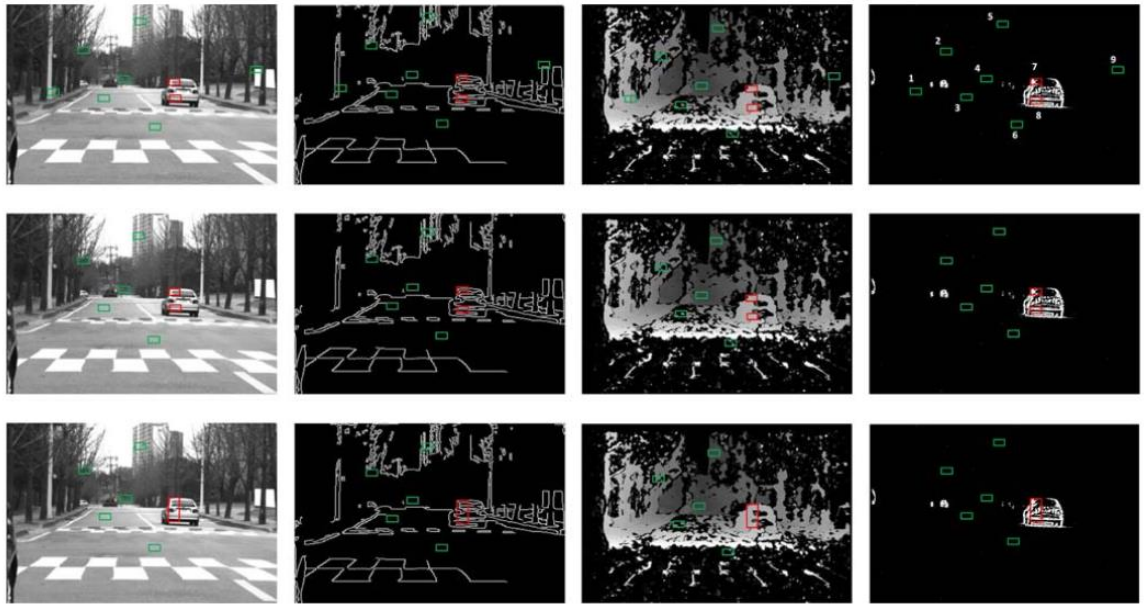


Fig. 7. Example of the proposed method in the first generation. The two red chromosomes have a higher fitness value than the other chromosomes. Therefore, they can survive into the next generation throughout the selection and crossover stages. First row: initialize population. Second row: tournament selection result. Third row: crossover result. First column: left image. Second column: edge image. Third column: disparity image. Fourth column: motion detection.

V. D. Nguyen, T. T. Nguyen, D. D. Nguyen, S. J. Lee and J. W. Jeon, "A Fast Evolutionary Algorithm for Real-Time Vehicle Detection," in IEEE Transactions on Vehicular Technology, vol. 62, no. 6, pp. 2453-2468, July 2013, doi: 10.1109/TVT.2013.2242910.

Outline

- Genetic Algorithm Review
- How to design Fitness Function
- Genetic Algorithm for Regression
- Genetic Algorithm for Classification
- Assignment Problem and Solution
- Case study: Vehicle Detection using GA
- Latest Trends in GA

IEEE Transactions on Evolutionary Computation

Submit Manuscript | Add Title To My Alerts | Ac My

Home | Popular | Early Access | Current Issue | All Issues | A

14.3	0.0129	4.031	25.5
Impact Factor	Eigenfactor	Article Influence Score	CiteScore Powered by Scopus

The articles in this journal are peer reviewed in accordance with the requirements set forth in the IEEE PSPB Operations Manual (sections 8.2.1.C & 8.2.2.A). Each published article was reviewed by a minimum of two independent reviewers using a single-blind peer review process, where the identities of the reviewers are not known to the authors, but the reviewers know the identities of the authors. Articles will be screened for plagiarism before acceptance.
Corresponding authors from low-income countries are eligible for waived or reduced open access APCs.

Read Full Aims & Scope

Journals & Magazines > IEEE Transactions on Evolutio... > Early Access

A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization

Publisher: IEEE

Cite This

PDF

Songbai Liu ; Qiuzhen Lin ; Jianqiang Li ; Kay Chen Tan **All Authors**

Abstract:

Recent decades have witnessed great advancements in multiobjective evolutionary algorithms (MOEAs) for multiobjective optimization problems (MOPs). However, these progressively improved MOEAs have not necessarily been equipped with scalable and learnable problem-solving strategies for new and grand challenges brought by the scaling-up MOPs with continuously increasing complexity from diverse aspects, mainly including expensive cost of function evaluations, many objectives, large-scale search space, time-varying environments, and multi-task. Under different scenarios, divergent thinking is required in designing new powerful MOEAs for solving them effectively. In this context, research studies on learnable MOEAs with machine learning techniques have received extensive

A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization

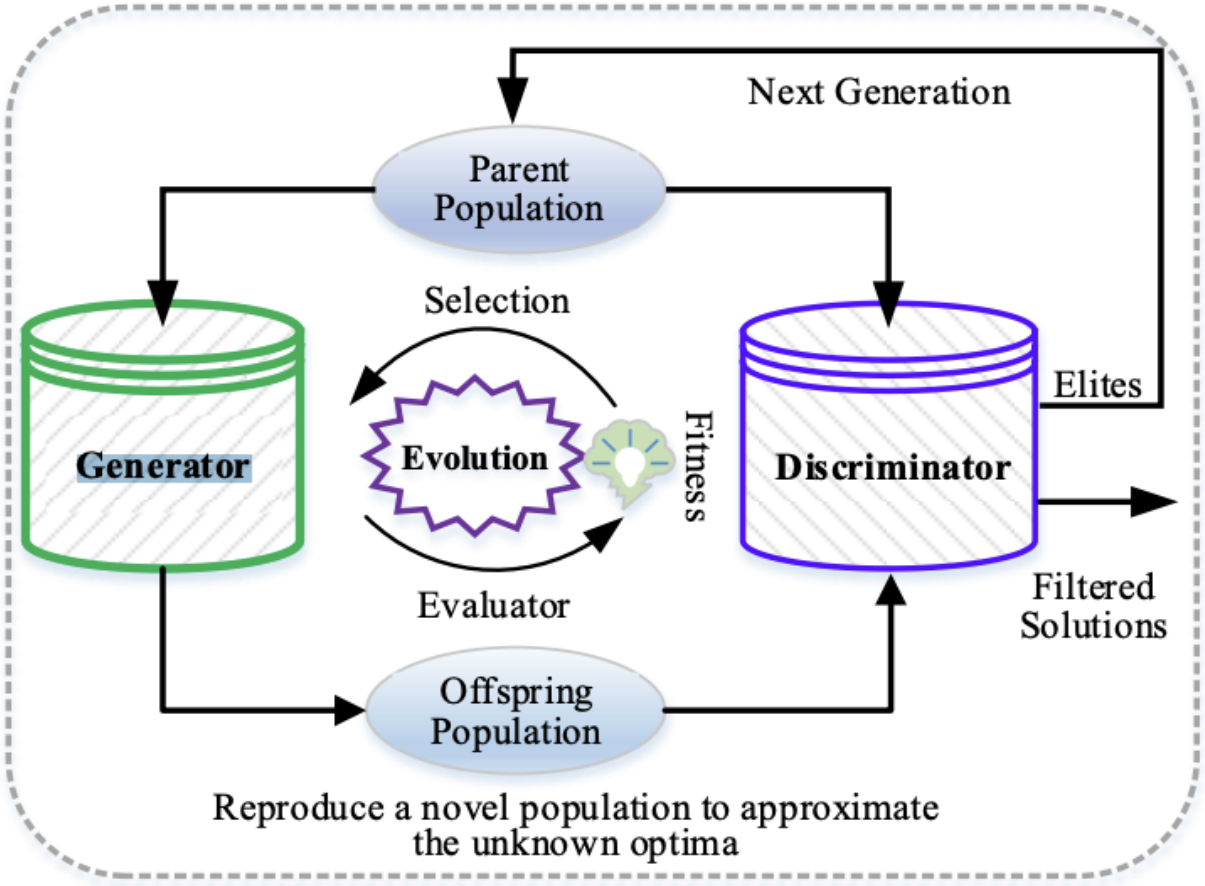


Fig. 1. The general flow of an MOEA from the perspective of ML.

A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization

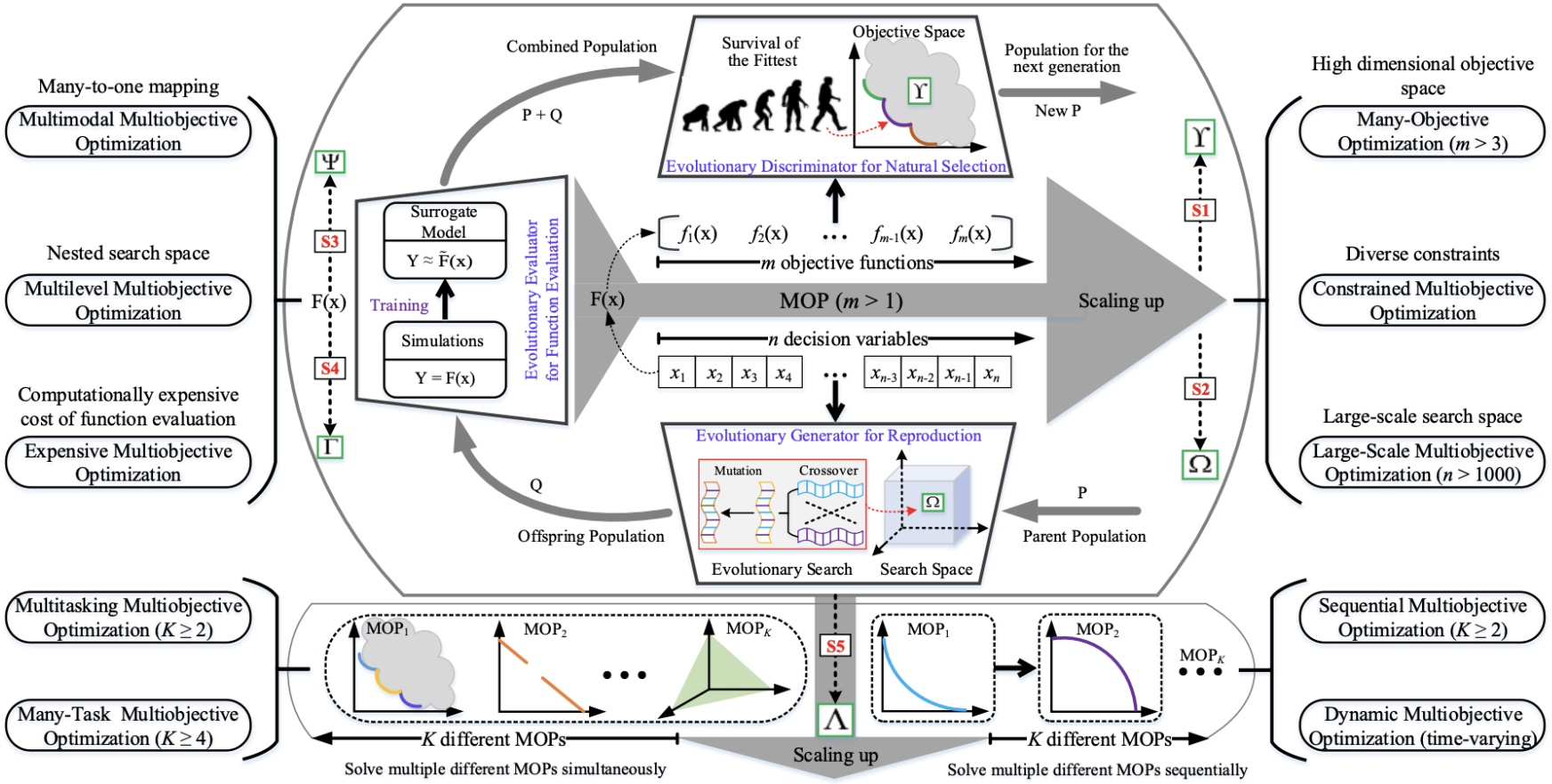


Fig. 2. Taxonomy of scaling-up MOPs according to the struct S_{MOP} from its five scalable components, including S1: objective space Υ, S2: search space Ω, S3: mapping diagram Ψ, S4: computational cost Γ for function evaluation, and S5: problem domain Λ.

A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization

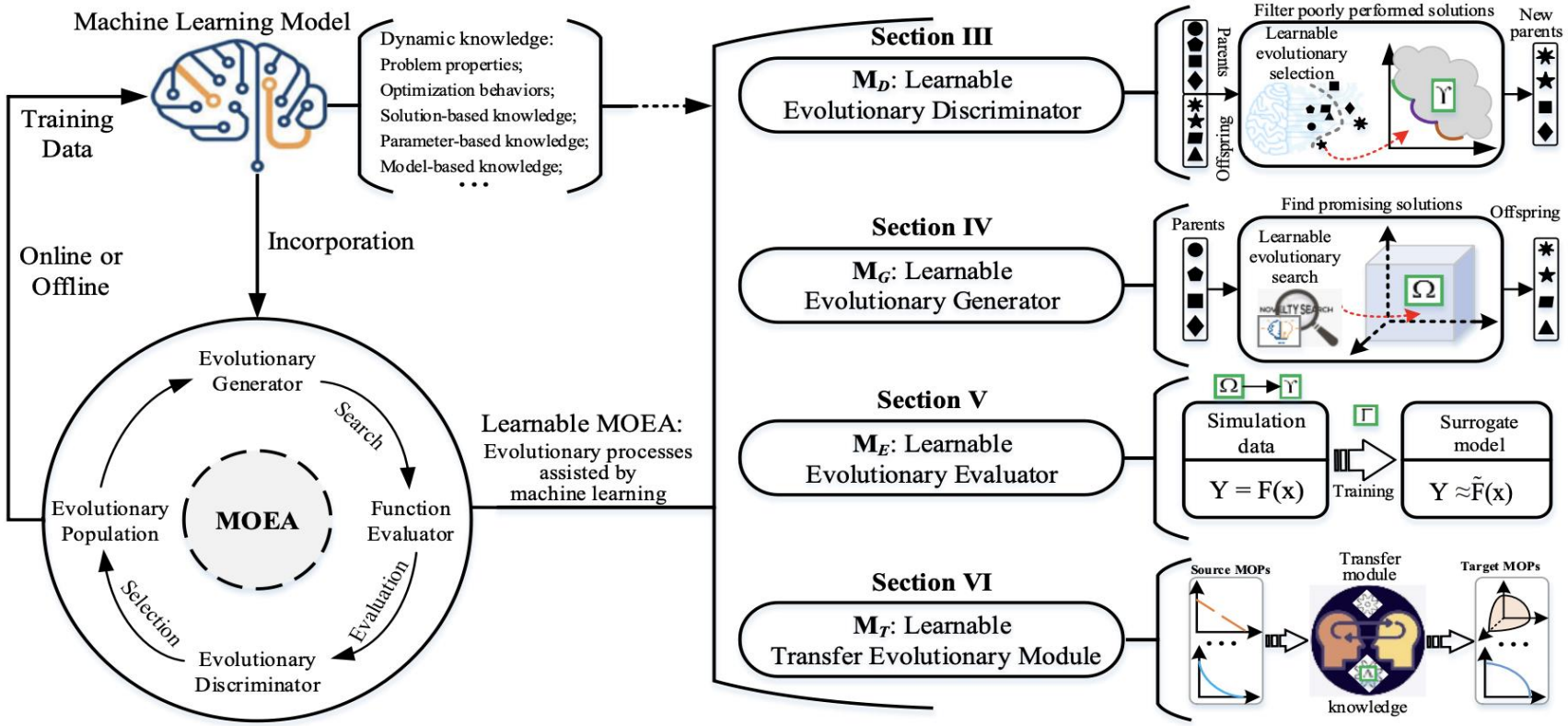


Fig. 3. Taxonomy of learnable MOEAs according to L_{MOEA} from its four learnable components, including M_D: evolutionary discriminator, M_G: evolutionary generator, M_E: evolutionary evaluator, and M_T: evolutionary transfer modules.

A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization

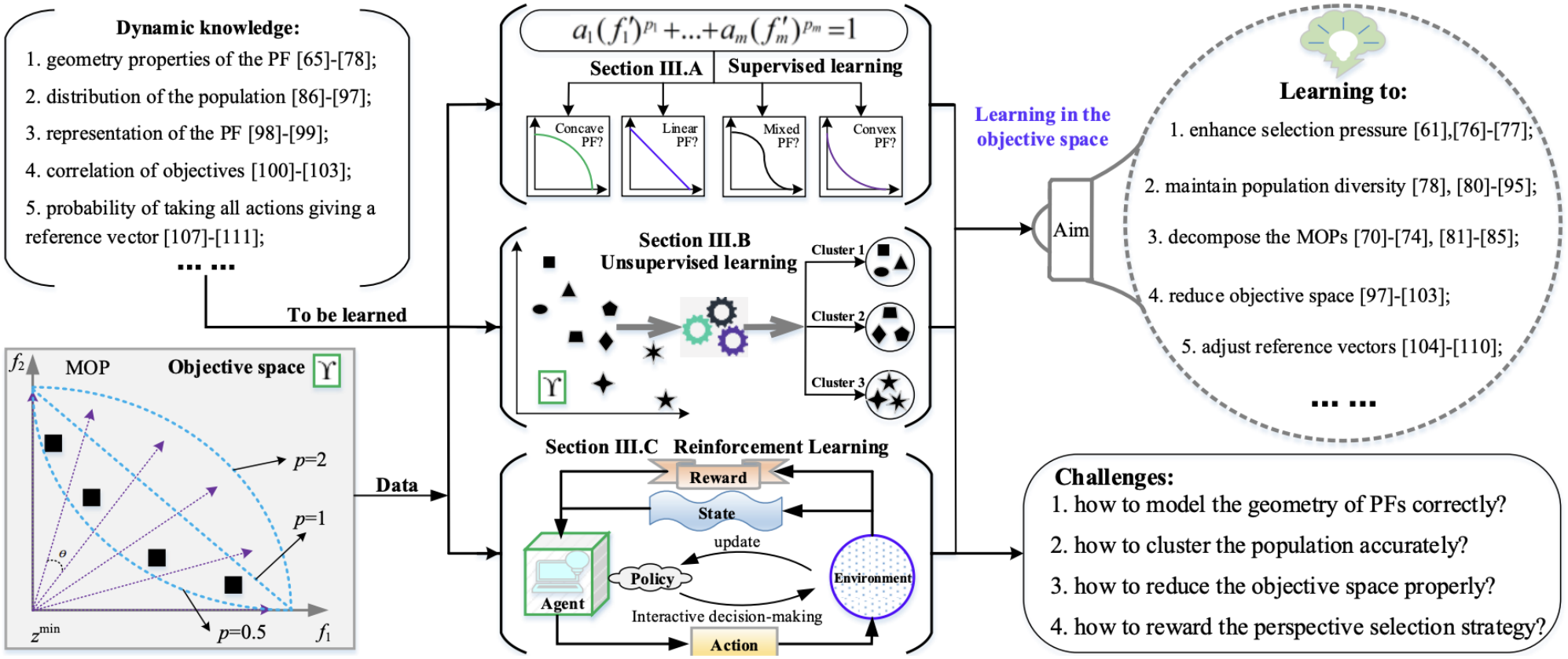


Fig. 4. Illustrating what knowledge can be learned in the objective space via various ML models and the aims of learning this dynamic knowledge.

A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization

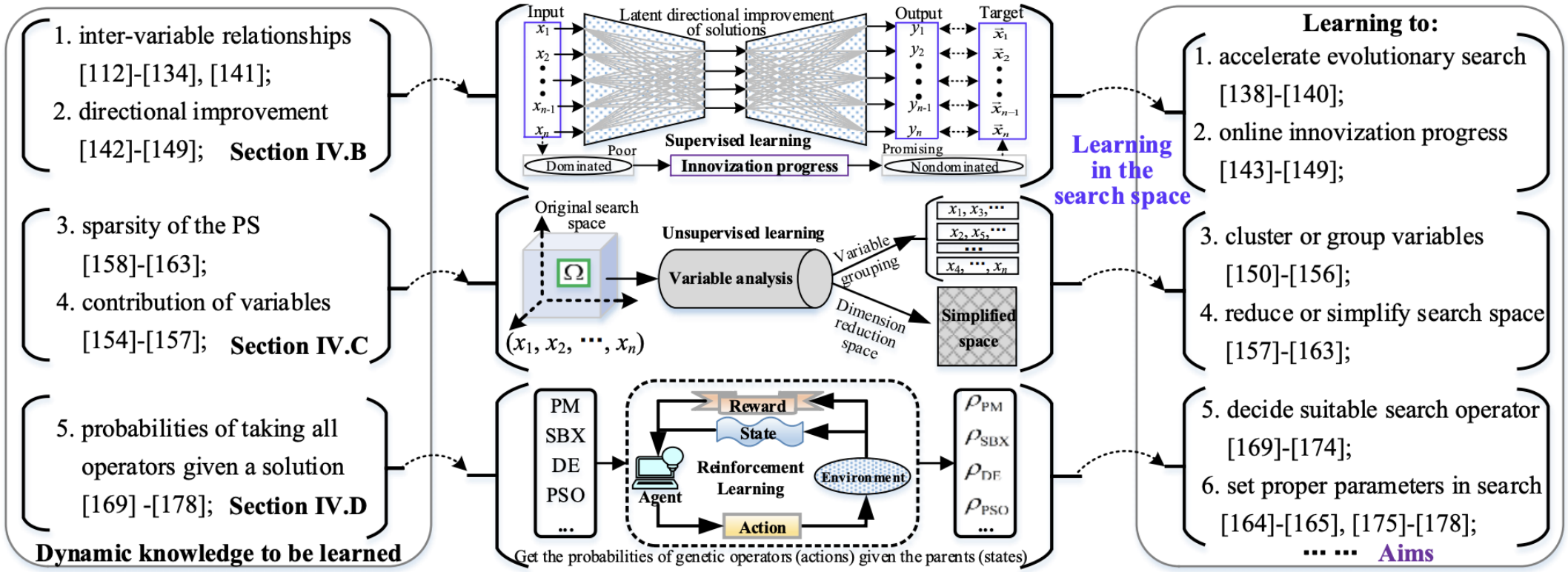


Fig. 5. Illustrating what knowledge can be learned in the search space via various ML models and the aims of learning this dynamic knowledge.

A Survey on Learnable Evolutionary Algorithms for Scalable Multiobjective Optimization

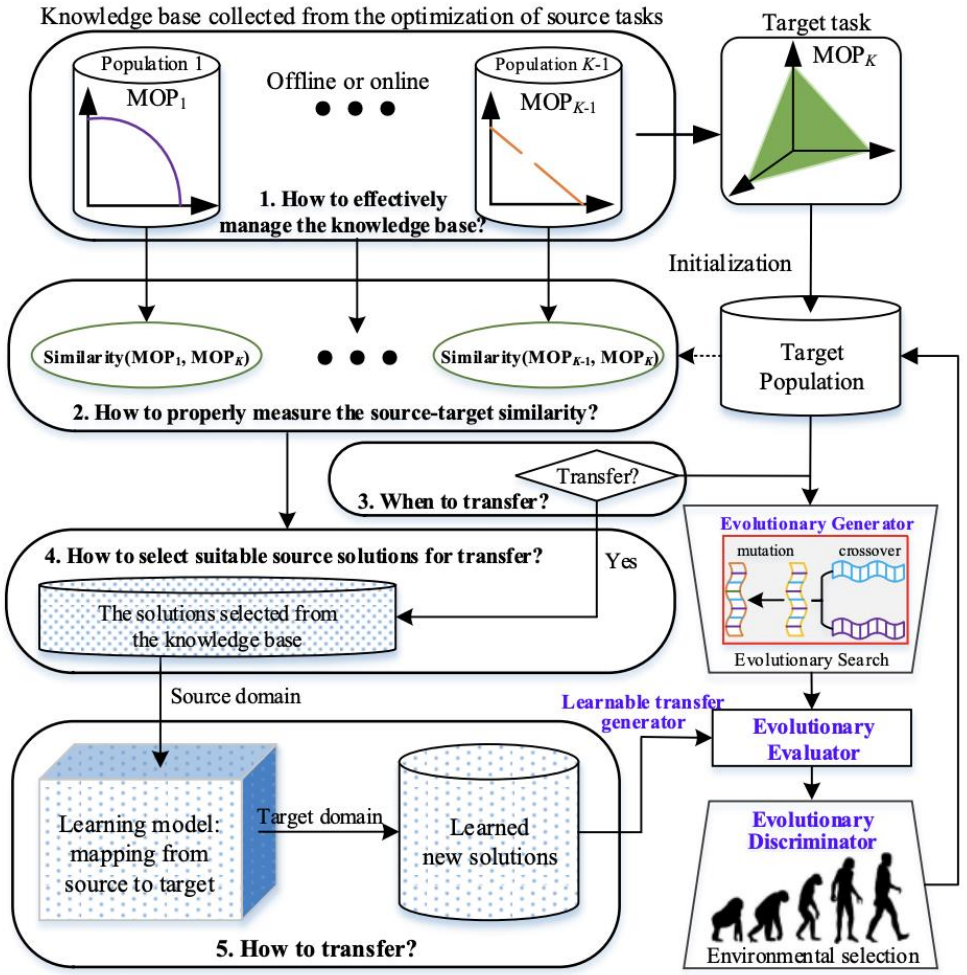


Fig. 6. The basic framework of solution-based transfer generators and the main challenges when designing this kind of transfer generators.

Outline

- GA Review
- How to Design Fitness Functions
- GA for Regression Problem
- GA for Classification Problem
- Assignment Discussion
- Summary