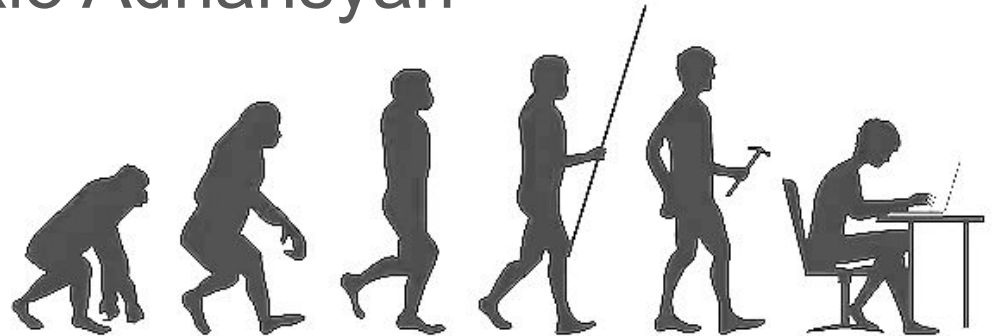


A stylized DNA double helix is positioned in the upper left area of the slide. The two strands are colored light blue and dark blue, and the base pairs are represented by red, orange, and teal rectangular blocks.

Evolutionary Algorithms

Ahmad Rio Adriansyah



Biomimicry

Apa itu?

Ada yang tahu?



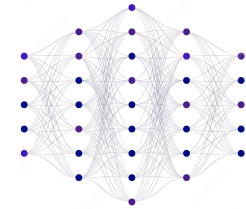
Biomimicry

Kita meniru sesuatu yang ada di alam untuk menyelesaikan masalah yang ada :

1. Arsitektur
2. Engineering
3. Desain
4. Komputer
5. dll



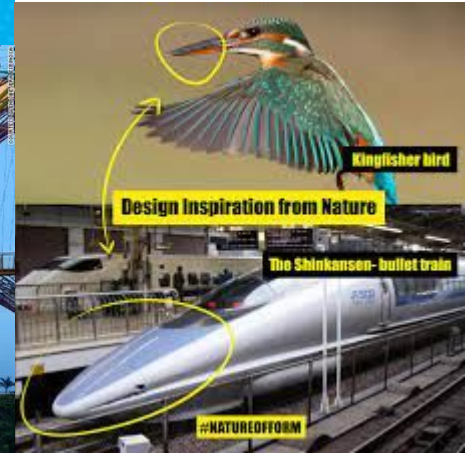
Artificial Neural Network



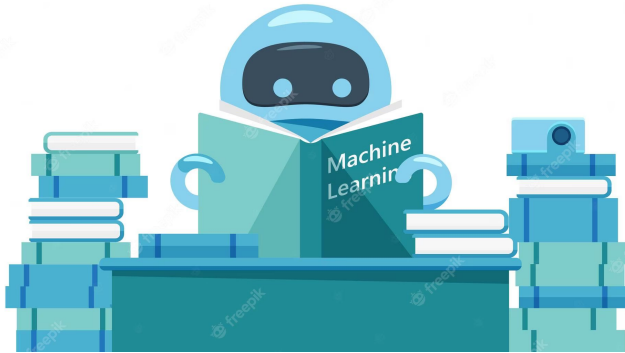
Stenocara beetle



Water collector



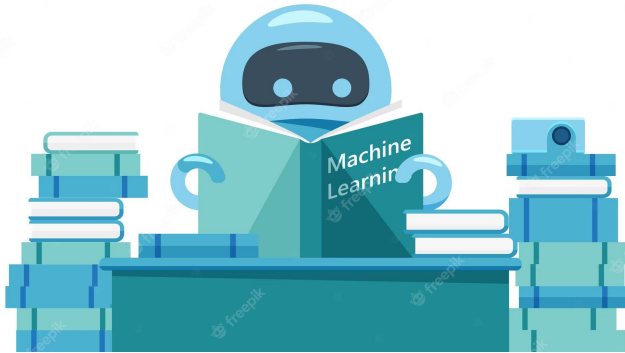
Machine Learning



Traditional Programming



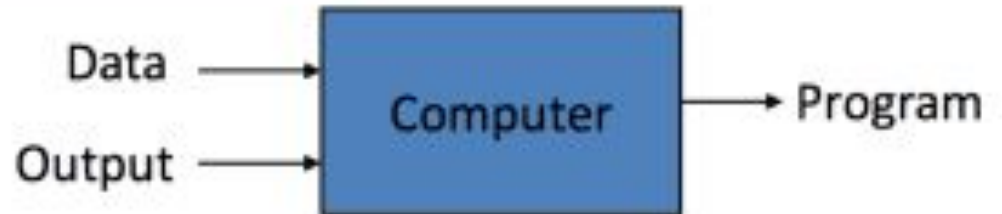
Machine Learning



Traditional Programming

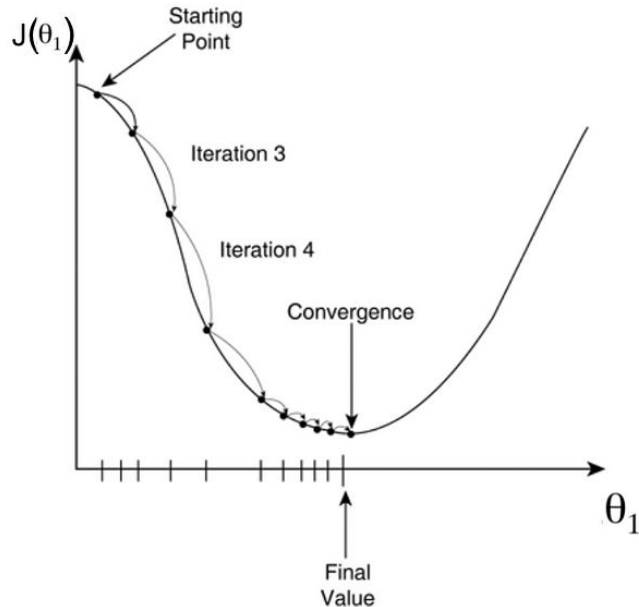


Machine Learning



Bagaimana Mesin Belajar?

Dengan dugaan yang terstruktur, dan sangat cepat. Tujuan dari model machine learning adalah **meminimalkan cost function / loss function**



Cost Function – “One Half Mean Squared Error”:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

Objective:

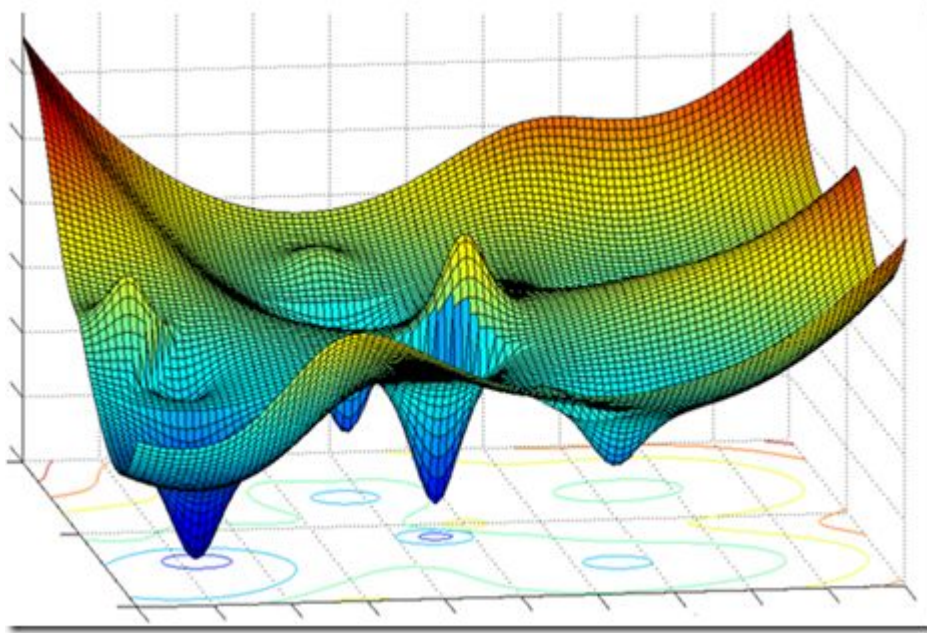
$$\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

Derivatives:

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$$

Bagaimana Mesin Belajar?

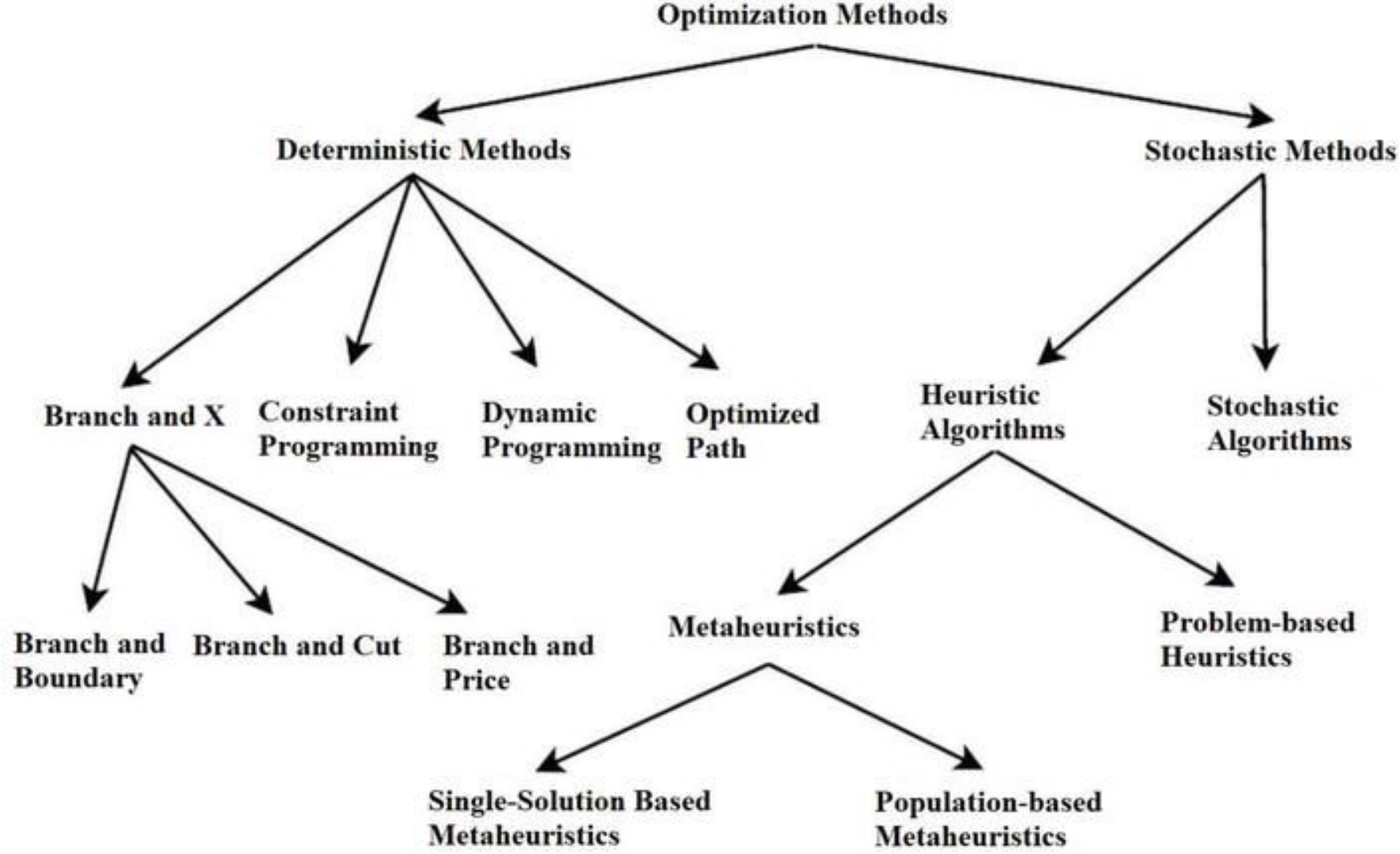


Makin banyak variabel terlibat, cost function menjadi tidak sederhana (high dimensional problem).

Bagaimana memastikan bahwa kita mendapatkan suatu nilai yang optimal (cost function paling kecil)?

Teknik Optimisasi

- Hill Climbing / Gradient Descent
- Stochastic Gradient Descent
- Particle Swarm Optimization
- Simulated Annealing
- Evolutionary Algorithm
- Convex Optimization
- dll

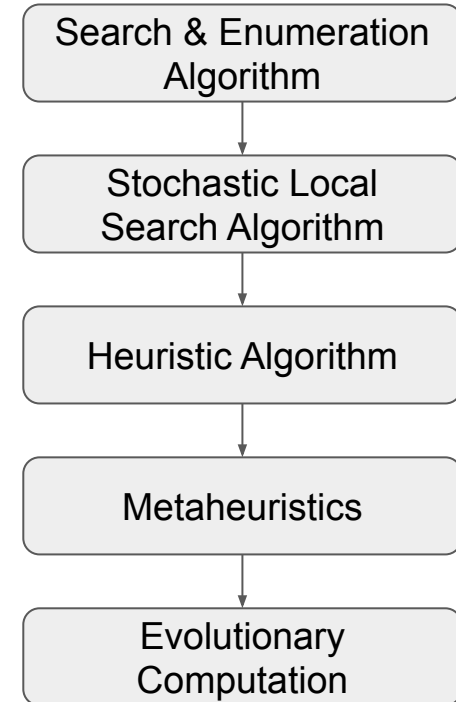
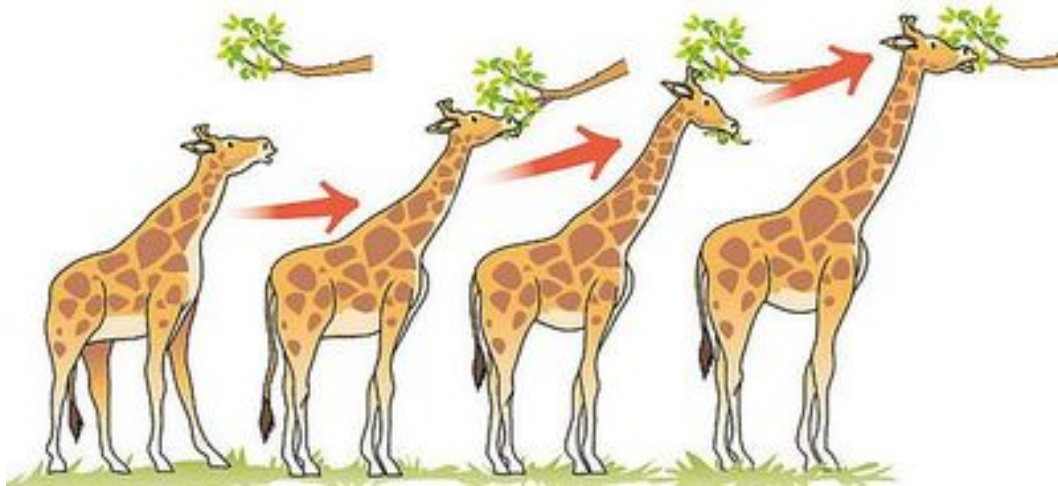


Algoritma Evolusi (Evolutionary Algorithm / EA)

Bagian dari algoritma optimasi metaheuristik

Metaheuristik = prosedur tingkat tinggi untuk mencari, membuat, atau memilih prosedur tingkat rendah (heuristik)

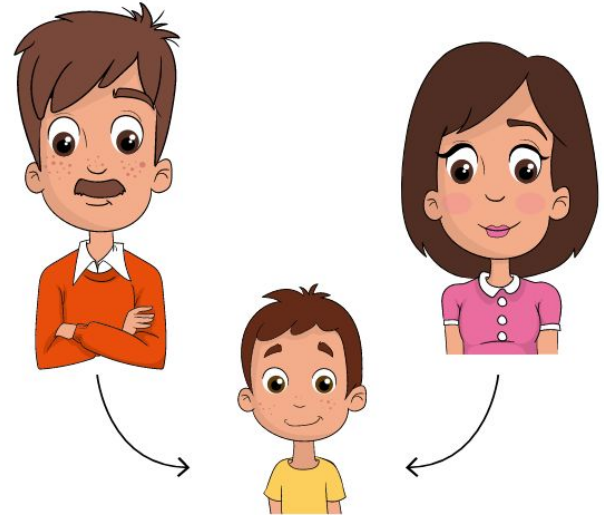
Meniru proses evolusi yang terjadi di alam



Bagaimana Evolusi Bekerja

Evolusi = perubahan pada karakteristik individu dalam populasi yang diwariskan secara turun temurun dari generasi ke generasi

- Karakteristik yang bisa diwariskan diturunkan dari satu generasi ke generasi berikutnya melalui DNA
- DNA (Deoxyribonucleic Acid) = molekul yang mengkodekan informasi genetik
- Perubahan dapat terjadi karena :
 - Mutasi = perubahan pada rangkaian DNA
 - Crossover = pencampuran gen karena reproduksi



Konsep Biologi

Evolusi digerakkan oleh seleksi alam (**natural selection**), dimana individu yang paling bisa beradaptasi yang bisa bertahan hidup (**survival of the fittest**)

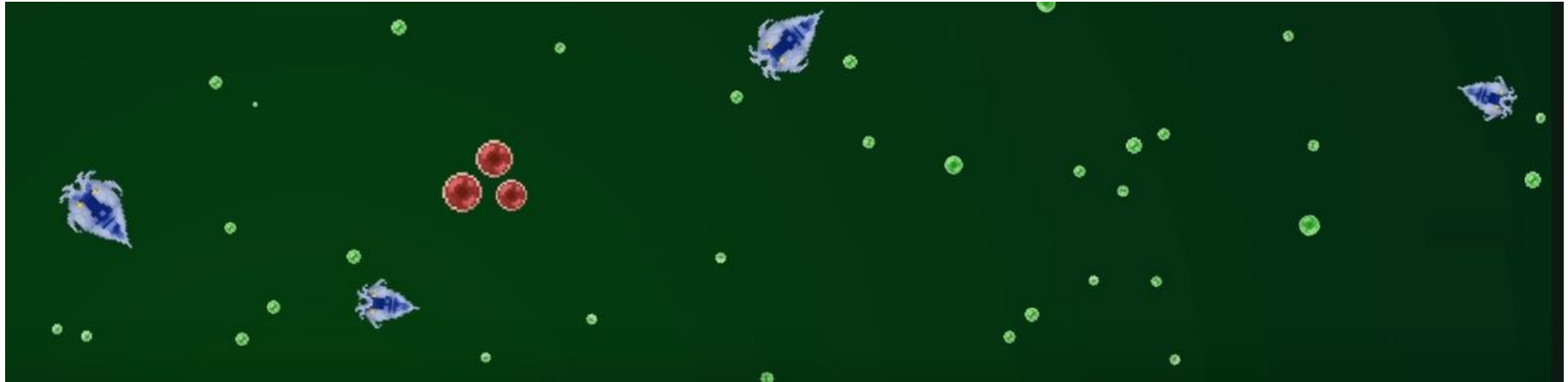
Variasi genetik yang mendukung kemampuan bertahan hidup dan reproduksi akan semakin terlihat dalam tiap generasinya



Evolusi Sebagai Optimasi

- Fitness ~ optimisation objective function
- Individual of species ~ solution to an optimisation problem
- Genetic variation (diversity) ~ exploration
- Nature selection ~ exploitation

Finding global optimum = balancing exploration and exploitation



Algoritma Evolusi (Umum)

1. Buat populasi awal (\mathbf{X}_0) secara random
2. Set kriteria pemberhentian, **termination_flag** = false
3. Set $t = 0$
4. Evaluasi nilai fitness masing masing individu dalam populasi \mathbf{X}_0
5. Selama **termination_flag** != true :
 - a. Pilih individu dari \mathbf{X}_t sebagai parent berdasarkan nilai fitnessnya
 - b. Buat beberapa individu baru dengan menerapkan mutasi dan crossover dari parent
 - c. Evaluasi nilai fitness dari individu-individu barunya
 - d. Buat populasi \mathbf{X}_{t+1} dengan mengganti individu yang nilai fitnessnya rendah
 - e. $t = t+1$
 - f. Jika kriteria terminasinya terpenuhi, **termination_flag** = true
6. Output = $\mathbf{x}_{\text{best}} \in \mathbf{X}_t$

1	0	1	1	1	1
1	0	1	1	1	1
0	1	1	0	1	0
1	0	0	0	0	1
1	0	1	0	1	0
0	1	0	1	0	1

Initialize
population

$F(x)$

Fitness
calculation



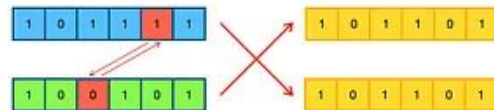
Final result



Selection

1	0	1	1	1	1
1	0	1	1	1	1
0	1	1	0	1	0
1	0	0	0	0	1
1	0	1	0	1	0
0	1	0	1	0	1

Survivor
selection



Crossover



Mutation

Untuk dapat mengaplikasikan Algoritma Evolusi ke suatu masalah tertentu, diperlukan :

1. Representasi dari solusi masalah yang ingin diselesaikan. Tiap solusi kita sebut sebagai **individu / kromosom**
2. Cara untuk menilai apakah solusi yang ada bagus atau tidak : **fitness (objective) function**
3. Cara untuk mengeksplorasi search space : **variation operators / operasi genetik**
4. Cara untuk mengarahkan algoritma agar menemukan solusi yang lebih baik (exploitation) : **selection** dan **reproduction**

Beberapa Jenis EA

1. Genetic Algorithm (GA) :
 - a. Representasi individu = strings, biasanya dalam biner
 - b. Variation operators = crossover (recombination) dan/atau mutasi
2. Evolution Strategy (ES)
 - a. Representasi individu = vektor bilangan riil
 - b. Variation operators = mutation, self-adaptive
 - c. Populasi tidak konstan
3. Evolutionary Programming (EP)
 - a. Mirip dengan ES($\mu+\lambda$)
 - b. μ dan λ menentukan banyak individu yang dipilih dan dijadikan parent
4. Genetic Programming (GP)
 - a. Mirip dengan GA
 - b. Representasi individu = graf pohon (kromosom tidak linier)
5. Differential Evolution (DE)
 - a. Digunakan untuk optimasi fungsi di search space yang kontinu
 - b. Lebih efisien dari GA dari sisi memori dan kompleksitas komputasi

Contoh Algoritma Genetika (Toy Example)

Problem : minimalkan fungsi $f(x) = 2x^4 - 5x^3 + x^2 - 3x + 1$
dimana $\{ x \mid 10x \in \mathbb{Z}, 0 \leq x \leq 3.1 \}$

Representasi : string biner , contoh '00101' = 0.5

Variation Operators :

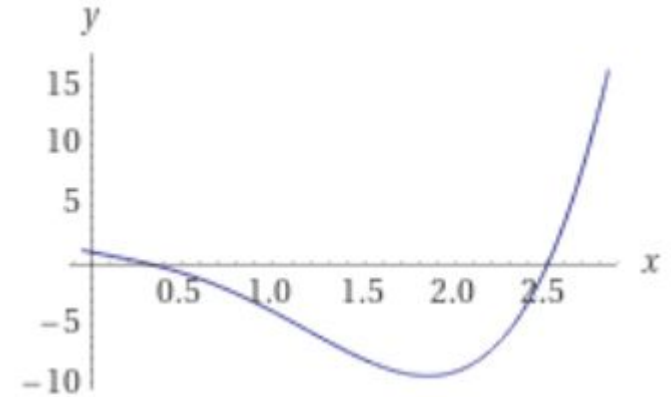
- Mutation : pilih bit secara random dan dibalik nilainya (flip)
- Crossover : pilih satu titik (swapping point) pada dua buah string, tukar data kedua string setelah titik tersebut

Quiz : bagaimana cara menentukan panjang string representasinya?

$10x \in \mathbf{Z}$, berarti x kelipatan 0.1

$0 \leq x \leq 3.1$, berarti ada 32 kemungkinan solusi (individu) berbeda yang perlu dicoba

32 individu dapat direpresentasikan dengan 5 bit string biner



Step 1 : Generate Populasi Awal (X_0)

10110	-> individu 1, mewakili nilai $x = 2.2$
01011	-> individu 2, mewakili nilai $x = 1.1$
01010	-> individu 3, mewakili nilai $x = 1.0$
11111	-> individu 4, mewakili nilai $x = 3.1$
00100	-> individu 5, mewakili nilai $x = 0.4$

Populasi

$t = 0$
termination_flag = false

Step 2 : Menghitung Nilai Fitness

Kita ingin meminimalkan fungsi $f(x) = 2x^4 - 5x^3 + x^2 - 3x + 1$, berarti makin kecil nilai $f(x)$, individunya dianggap makin fit.

Kita buat fitness function $fit(x) = -f(x)$, jadi makin kecil nilai $f(x)$, nilai $fit(x)$ makin besar.


10110 -> individu 1, mewakili nilai $x = 2.2$, **$fit(x) = 7.1488$**

01011 -> individu 2, mewakili nilai $x = 1.1$, **$fit(x) = 4.8170$**

01010 -> individu 3, mewakili nilai $x = 1.0$, **$fit(x) = 4.0000$**

11111 -> individu 4, mewakili nilai $x = 3.1$, $fit(x) = -37.0592$

00100 -> individu 5, mewakili nilai $x = 0.4$, $fit(x) = 0.3090$



Pilih beberapa individu terbaik untuk reproduksi

Step 3 : Crossover

10110

-> individu 1, mewakili nilai $x = 2.2$, **fit(x) = 7.1488**

01011

-> individu 2, mewakili nilai $x = 1.1$, **fit(x) = 4.8170**

01010

-> individu 3, mewakili nilai $x = 1.0$, **fit(x) = 4.0000**

Individu terpilih
(selected_chromosomes)

Random parent dari individu terpilih

10110 + 01010

10010

01110

Step 4 : Mutasi

Populasi baru (parent + child)

10110
01011
01000

-> individu 3 termutasi di bit keempat

10010
01110

Dari generasi sebelumnya yang selamat

01010

01000

Individu baru hasil crossover

Step 5 : Ulangi Loop dari Step 2

Populasi baru di generasi selanjutnya (\mathbf{X}_1)

10110 -> individu 1, mewakili nilai $x = 2.2$, **fit(x) = 7.1488**

01011 -> individu 2, mewakili nilai $x = 1.1$, $\text{fit}(x) = 4.8170$

01000 -> individu 3, mewakili nilai $x = 0.8$, $\text{fit}(x) = 2.5010$

10010 -> individu 4, mewakili nilai $x = 1.8$, **fit(x) = 9.3248**

01110 -> individu 5, mewakili nilai $x = 1.4$, **fit(x) = 7.2768**

Individu terbaik
untuk reproduksi

$t = 1$
termination_flag = false

Cek Implementasinya dalam Python

<buka file **ga_sample.py**>

[https://github.com/Arasy/Statistik-dan-Data-Science/
tree/main/Genetic%20Algorithm](https://github.com/Arasy/Statistik-dan-Data-Science/tree/main/Genetic%20Algorithm)

Kelebihan dan Kekurangan

- + Berguna untuk masalah yang memiliki kemungkinan solusi sangat besar dimana exhaustive search tidak memungkinkan
- + Versatile, dapat diterapkan ke berbagai macam kasus optimasi dalam berbagai bidang
- + Berpotensi untuk menemukan global optima atau solusi yang mendekati optimal, terutama untuk masalah kompleks dan multimodal
- + Kecil kemungkinan untuk terjebak dalam local optima seperti metode optimasi lainnya
- + Dapat diparalelkan dengan baik
- + Bisa mendapat solusi yang tidak terduga dari kombinasi berbagai elemen dalam populasi, yang berujung pada pemecahan masalah yang inovatif

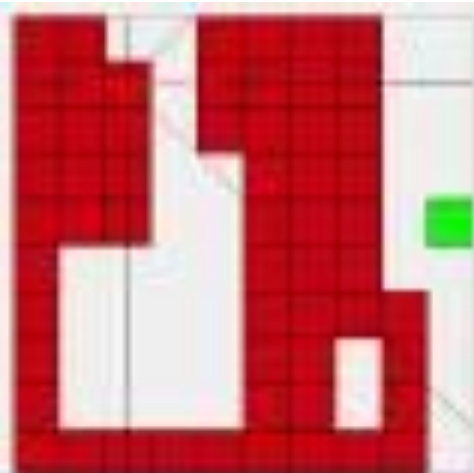
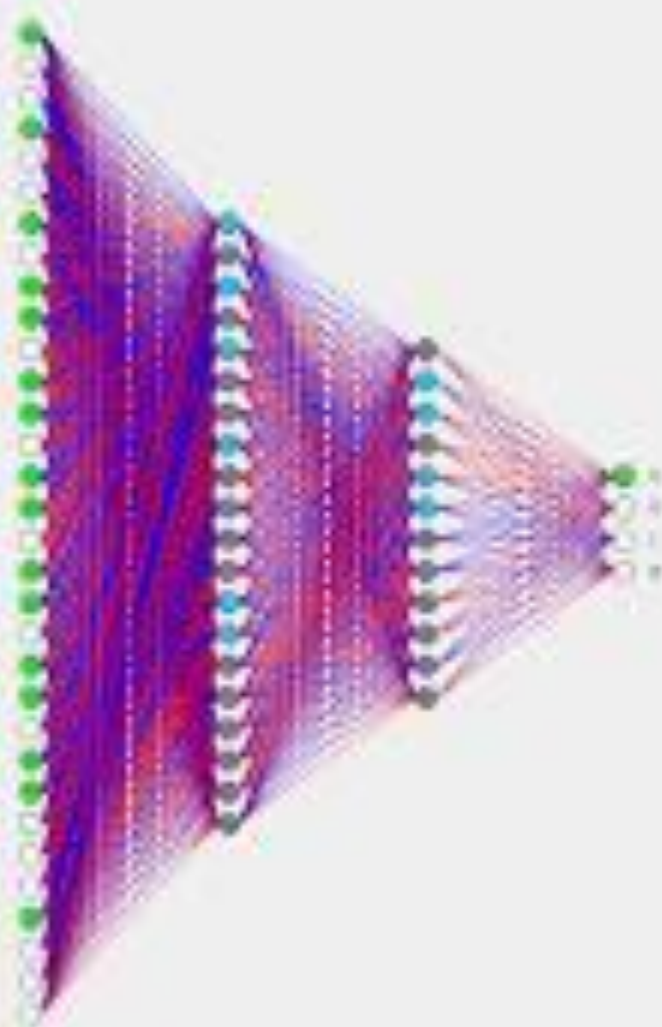
Kelebihan dan Kekurangan

- Komputasi tinggi, membutuhkan sumber daya yang besar (computationally expensive)
- Tidak banyak menggunakan problem-specific knowledge, hanya bergantung pada evaluasi nilai fitness function. Biasanya dengan problem-specific insight dapat meningkatkan efisiensi dan efektivitas
- Membutuhkan desain representasi dan operator yang tepat
- Sensitif terhadap pengaturan parameter seperti ukuran populasi, faktor mutasi, dan peluang crossover
- Tidak menjamin adanya solusi optimal, tapi bisa mencapai solusi yang near-optimal
- Tidak menjamin konvergensi solusi, kadang konvergensi terjadi terlalu cepat (prematur) sehingga algoritmanya terjebak dalam solusi suboptimal, terutama disebabkan oleh divergensi yang rendah

What Else?

EA dapat digunakan juga untuk mengoptimasi hyperparameter tuning pada machine learning dan teknik optimasi lainnya sehingga bisa mengurangi campur tangan manusia dalam memperoleh solusi yang lebih baik.

EA juga dapat digunakan untuk menyederhanakan struktur dalam neural network sehingga mengurangi waktu training dan kompleksitas model (sparse neural network).



Convolution: 1x1

Individual: 1/100

Batch Normal: 1%

Batch Inference: 1/10 x 1/1

EA Settings

Activation Type: ReLU

Convolution Type: 100%, 100%

Mutation Type: 100% Success

0% Crossover

Mutation Rate: 1% x 100%

Language: python

NN Settings

Hidden Activation: ReLU

Output Activation: sigmoid

NN Architecture: [10, 10, 10, 1]

Number of Layers: 4 layers

Apply L2/L1 Norm: distance

Dengan representasi and variasi operasi yang tepat, Evolutionary Algorithm dapat digunakan untuk memecahkan beragam masalah

Contohnya seperti hal-hal berikut :

Generation: 71





Game 1
Carri 280
Amp 37
Saw 1000 0
Unsp 177
Hayer OFF
PMS 30

- Red - Change Mutation
- Blue - Belov/Lowest
- Black - Online
- Green - Global Side Lines
- B. Brant
- B. Brent
- C. Clon
- N. New Food
- A. Toggle Player
- On Toggle Side

Next Read

1. Neuroevolution of Augmenting Topologies (NEAT)
2. Swarm Intelligence
3. Simulated Annealing
4. Non Convex Optimization
5. Sparse Neural Network
6. dst

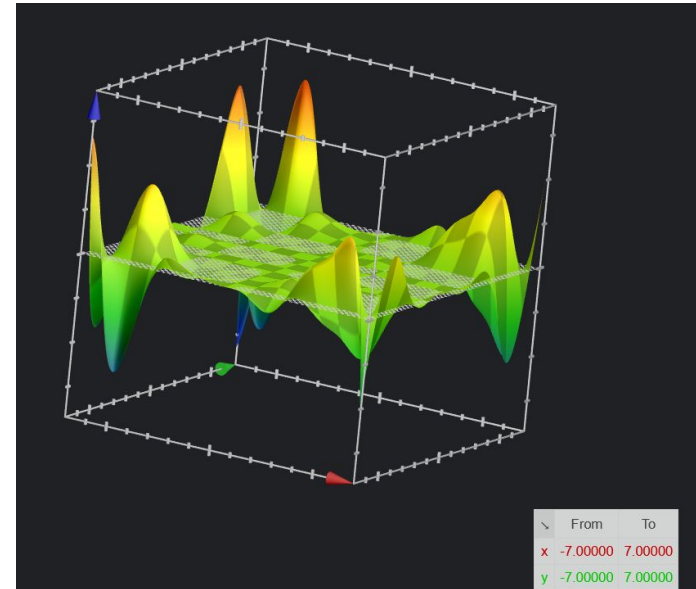
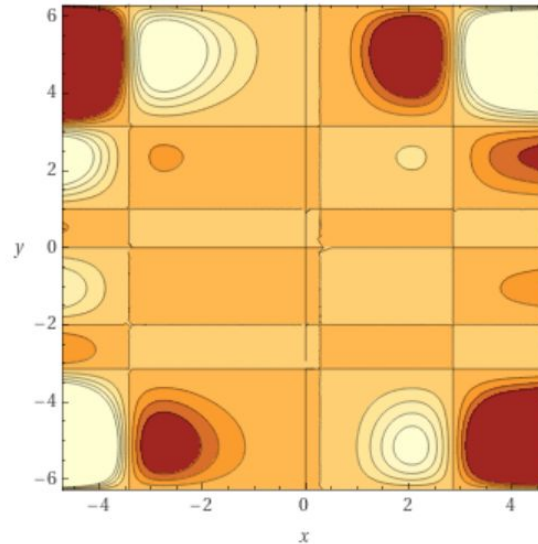
Latihan

Diberikan sebuah fungsi :

$$(3*\sin(x)-\cos(2x))*(x^2)*\sin(y)*(y-1)*(y+2)$$

dengan constrain

$$-7 \leq x, y \leq 7$$



Latihan

Buatlah program berbasis Genetic Algorithm (GA) untuk mencari nilai maksimum dari fungsi tersebut. Lakukan analisis dan desain program GA, lalu implementasikan dalam python.

Yang harus didesain :

1. Ukuran populasi, rancangan kromosom, dan cara decodenya
2. Metode pemilihan parent
3. Operasi variasi genetik yang ada (crossover, mutation)
4. Peluang operasi variasi genetik (P_c , P_m)
5. Metode pergantian generasi (seleksi alam)
6. Kriteria penghentian evolusi

Referensi

1. Shan He's Lecture Notes (University of Birmingham)
<https://www.cs.bham.ac.uk/~szh/teaching/ec/>
2. Wayan Firdaus Mahmudi, Modul Kuliah Dasar-dasar Algoritma Evolusi (Universitas Brawijaya)
<http://wayanfm.lecture.ub.ac.id/category/materi-kuliah/>
3. Emerging Trends in Mechatronics
<https://www.intechopen.com/chapters/66272>
4. Adam Slowik & Halina Kwasnicka, Evolutionary algorithms and their applications to engineering problems
<https://link.springer.com/article/10.1007/s00521-020-04832-8>
5. Kenneth O. Stanley & Risto Miikkulainen, Evolving Neural Networks through Augmenting Topologies
<https://ieeexplore.ieee.org/document/6790655>

Video Used

1. AI Learns to Play Snake!
<https://www.youtube.com/watch?v=vhiO4WsHA6c>
2. Genetic Algorithm. Learning to Jump Over Ball
https://www.youtube.com/watch?v=Gl3EjiVlz_4
3. Neural Network Cars and Genetic Algorithms
<https://www.youtube.com/watch?v=-sg-GgoFCP0>
4. The Evolution of Predation in a Simulated Ecosystem
<https://www.youtube.com/watch?v=rPkMoFJNcLA>