

Praktikum Kecerdasan Buatan

Algoritma Genetika

Dosen Pengampu Entin Martiana Kusumaningtyas S.Kom,
M.Kom



Wahyu Ikbal Maulana

3323600056

D3 SDT B

Politeknik Elektronika Negeri Surabaya

| Tugas 1

1. Buatlah program python algoritma genetika untuk word matching yang ada pada link ini:

https://colab.research.google.com/drive/1k4r9NY55iUqsm219ScNvxR_G07a2gKVe?usp=sharing


2. Ubahlah Target menjadi nama masing-masing!
3. Jalankan Program! Butuh berapa generasi agar menemukan nama kalian?
4. Ubah banyak Populasi menjadi:
 - jauh lebih kecil seperti 10 dan 100.
 - Lebih besar seperti 10000 dan 100000, Jelaskan yang terjadi?

```
In [ ]: # Python3 program to create target string, starting from
# random string using Genetic Algorithm
# https://www.geeksforgeeks.org/genetic-algorithms/
import random

# Number of individuals in each generation
POPULATION_SIZE_SMALL = 100
POPULATION_SIZE_BIG = 10000

# Valid genes
GENES = '''abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890, .-:;_!"#%&/()=?@${[]}' ''

# Target string to be generated
TARGET = "WAHYU IKBAL MAULANA"
```

 **Analisis :** Ketika populasi menjadi jauh lebih kecil menjadi 100, proses generation text untuk word matching akan berjalan lebih cepat karena hanya sedikit tiap huruf yang dievaluasi dan diproses. Sementara itu, ketika populasi menjadi lebih besar seperti 10000, proses evolusi akan memakan waktu lebih lama karena ada lebih banyak individu yang dievaluasi dan diproses. Namun, ini dapat meningkatkan keberagaman gen dalam populasi dan kemungkinan menemukan solusi yang lebih baik.

| Tugas 2

5. Pada proses Reproduksi (mate) ubahlah nilai probabilitas dari proses cross over dan mutasi, dimana fungsi awalnya: prob = random

Jika $\text{prob} < 0.45 \rightarrow$ gen parent 1 dijadikan gen child
Jika $0.45 < \text{prob} < 0.90 \rightarrow$ gen parent 2 dijadikan gen child
Jika $\text{prob} \geq 0.90 \rightarrow$ gen child merupakan hasil mutasi
Jelaskan hasil yang terjadi?

6. Sama dengan nomor 5 ubahlah nilai probabilitas untuk mutasi sehingga kemungkinan bermutasi jauh lebih besar dari sebelumnya. Contoh: $\text{prob} \geq 0.90$ ini berarti kemungkinan mutasi $\rightarrow 0.1$

```

In [ ]: class Individual(object):
    '''
    Class representing individual in population
    '''

    def __init__(self, chromosome):
        self.chromosome = chromosome
        self.fitness = self.cal_fitness()

    @classmethod
    def mutated_genes(self):
        '''
        create random genes for mutation
        '''

        global GENES
        gene = random.choice(GENES)
        return gene

    @classmethod
    def create_gnome(self):
        '''
        create chromosome or string of genes
        '''

        global TARGET
        gnome_len = len(TARGET)
        return [self.mutated_genes() for _ in range(gnome_len)]

    def mate(self, par2):
        '''
        Perform mating and produce new offspring
        belum ada Cross-over or
        just get the parent gene to become offspring
        '''

        #chromosome for offspring
        child_chromosome = []
        for gp1, gp2 in zip(self.chromosome, par2.chromosome):

            #random probability
            prob = random.random()

            #if prob is less than 0.45, insert gene
            #from parent 1
            if prob < 0.45:
                child_chromosome.append(gp1)

            #if prob is between 0.45 and 0.90, insert
            #gene from parent 2
            elif (prob < 0.90):
                child_chromosome.append(gp2)

            #otherwise insert random gene(mutate)
            #for maintaining diversity
            else:
                child_chromosome.append(self.mutated_genes())

        return Individual(child_chromosome)

    def cal_fitness(self):
        '''
        Calculate fitness score, it is the number of


```

```

characters in string which differ from target
string
'''

global TARGET
fitness = 0
for gs, gt in zip(self.chromosome, TARGET):
    if gs != gt: fitness +=1
return fitness

```

 **Analisis :** Hasil yang terjadi adalah terjadinya perubahan kemampuan yang lebih cepat untuk emncapai string untuk solusi dari target. Jadi hal ini menunjukkan bahwa penyesuaian probabilitas dapat mempengaruhi algoritma genetika

| Tugas 3

7. Analisa lah semua percobaan yang dilakukan dan jelaskan apa itu:

- Generasi, Populasi, Individu, Kromosom dan Gen yang ada pada program pencarian nama masing-masing!
- Apa hubungan nilai probabilitas cross over dan probabilitas mutasi terhadap output yang didapatkan?
- Apa itu elitism?

```

In [ ]: #Driver code
def main():
    global POPULATION_SIZE_BIG

    #current generation
    generation = 1

    found = False
    population = []

    #create initial population (populasi awal)
    for _ in range(POPULATION_SIZE_BIG):
        gnome = Individual.create_gnome()
        population.append(Individual(gnome))

    while not found:

        #sort the population in increasing order of fitness score
        #(Evaluasi Nilai Fitness)
        population = sorted(population, key = lambda x:x.fitness)

        # if the individual having lowest fitness score ie.
        # 0 then we know that we have reached to the target
        # and break the loop
        if population[0].fitness <= 0:
            found = True
            break

```

```

#otherwise generate new offsprings for new generation
new_generation = []

#perform elitism, that mean 10% of fittest population
#goes to the next generation
s=int((10*POPULATION_SIZE_BIG)/100)
new_generation.extend(population[:s])


#from 50% of fittest population (Seleksi), Individuals
#will mate (Reproduksi) to produce offspring
s = int((90*POPULATION_SIZE_BIG)/100)
for _ in range(s):
    parent1 = random.choice(population[:50])
    parent2 = random.choice(population[:50])
    child = parent1.mate(parent2)
    new_generation.append(child)

population = new_generation

print("Generation: {}\\tString: {}\\tFitness: {}".\\
      format(generation,
              "".join(population[0].chromosome),
              population[0].fitness))
generation +=1

print("Generation: {}\\tString: {}\\tFitness: {}".\\
      format(generation,
              "".join(population[0].chromosome),
              population[0].fitness))

```

 **Analisis :** Generasi yaitu iterasi dalam algoritma genetika disini word matching mengenerate karakter dan melakukan pencocokan hingga karakter yang dijadikan target berhasil tercapai, Populasi adalah kumpulan individu, Individu adalah representasi satu entitas dalam populasi, Kromosom adalah kumpulan gen dengan gen dan representasi dari solusi dalam searching, mirip dengan kromosom biologis yang membawa informasi genetik dalam organisme hidup, sementara gen adalah unit dasar yang membentuk kromosom, mewakili atribut atau fitur solusi. Nilai probabilitas cross over mempengaruhi seberapa sering gen dari dua individu/huruf yang berbeda dipertukarkan, sedangkan probabilitas mutasi menentukan seberapa sering gen individu berubah secara acak, dengan fitness yang semakin kecil semakin dekat. Probabilitas ini memengaruhi variasi gen dalam populasi dan kemungkinan menemukan solusi yang lebih baik.

Elitism adalah strategi/algoritma yang mempertahankan sebagian dari individu terbaik, dalam hal ini yaitu yang individu dengan fitness terbaik dari iterasi generasi sebelumnya untuk langsung dimasukkan ke generasi berikutnya tanpa melalui proses seleksi atau reproduksi. Generasi disini dalam word matching adalah generasi text.

```

In [ ]: if __name__ == '__main__':
        main()

```

Generation: 1	String: Tzt10pI4BXnxMV"LEv{	Fitness: 15
Generation: 2	String: TmtY=1I4B{LjMV"LE8{	Fitness: 13
Generation: 3	String: Y.-,U IKBPL MAUGYzp	Fitness: 9
Generation: 4	String: Y.zYU IKBPL MADLAzA	Fitness: 6
Generation: 5	String: WAHYUVIKBrL MAULA7A	Fitness: 3
Generation: 6	String: WAHYU IKBAL MAULANA	Fitness: 0