TUGAS 7
ALGORITMA GENETIKA

Mata Kuliah Kecerdasan Buatan

Dosen Pengampu : Yuyun Umaidah, M.Kom.

Disusun Oleh :

Nama : Eldoni Mosul

NPM : 2210631170065

Program Studi Sistem Informasi

Fakultas Ilmu Komputer

Universitas Singaperbangsa Karawang

Tahun 2024

SOAL

Misalkan terdapat sebuah persamaan, $f(x) = -(x^2) + 10$. Dibutuhkan sebuah solusi agar persamaan tersebut memiliki nilai maksimum dan batasannya adalah $0 \leq x \leq 31$. Temukan solusi terbaik menggunakan algoritma genetika!

JAWABAN

- Deskripsi Fungsi dan Inisiasi Populasi secarara acak serta melakukan persilangan dan mutasi untuk menghitung nilai *fitness*

```python
import numpy as np

def fungsi_kecocokan(x):
    return -(x**2) + 10

def inisialisasi_populasi(ukuran_populasi):
    return np.random.uniform(0, 31, ukuran_populasi)
```

```python
def crossover(induk1, induk2):
    induk1 = np.array([induk1]) if not isinstance(induk1, np.ndarray) else induk1
    induk2 = np.array([induk2]) if not isinstance(induk2, np.ndarray) else induk2
    panjang_induk = min(len(induk1), len(induk2))
    titik_crossover = np.random.randint(0, panjang_induk)
    anak1 = np.concatenate((induk1[:titik_crossover], induk2[titik_crossover:]))
    anak2 = np.concatenate((induk2[:titik_crossover], induk1[titik_crossover:]))
    return anak1, anak2


def mutasi(individu, tingkat_mutasi):
    for i in range(len(individu)):
        if np.random.random() < tingkat_mutasi:
            individu[i] = np.random.uniform(0, 31)
    return individu
```

- Fungsi Utama untuk menentukan populasi

```python
def algoritma_genetika(ukuran_populasi, tingkat_mutasi, maks_generasi):
    populasi = inisialisasi_populasi(ukuran_populasi)
    for generasi in range(maks_generasi):
        # Evaluasi kecocokan
        kecocokan = np.array([fungsi_kecocokan(x) for x in populasi])
        kecocokan = kecocokan - np.min(kecocokan) + 1  # Normalisasi nilai kecocokan
        # Seleksi
        probabilitas_seleksi = kecocokan / np.sum(kecocokan)
        probabilitas_seleksi = probabilitas_seleksi.flatten()  # Mengonversi ke satu dimensi
        indeks_terpilih = np.random.choice(np.arange(ukuran_populasi), size=ukuran_populasi, replace=True, p=probabilitas_seleks
        populasi_terpilih = populasi[indeks_terpilih]
        # Reproduksi
        populasi_baru = []
        for i in range(0, ukuran_populasi - 1, 2):  # Ubah indeks maksimum untuk menghindari indeks terakhir yang tidak memiliki
            anak1, anak2 = crossover(populasi_terpilih[i], populasi_terpilih[i+1])
            anak1 = mutasi(anak1, tingkat_mutasi)
            anak2 = mutasi(anak2, tingkat_mutasi)
            populasi_baru.extend([anak1, anak2])
        populasi = np.array(populasi_baru)
        # Cari solusi terbaik
        indeks_terbaik = np.argmax(kecocokan)
        solusi_terbaik = populasi[indeks_terbaik]
        kecocokan_terbaik = kecocokan[indeks_terbaik]
        print("Generasi:", generasi, "Solusi Terbaik:", solusi_terbaik, "Kecocokan:", kecocokan_terbaik)
    return solusi_terbaik
```

- Output

```
solusi_terbaik = algoritma_genetika(100,0.1,100)
print("Solusi Terbaik yang Ditemukan:", solusi_terbaik)
print("Nilai Maksimum dari Fungsi:", fungsi_kecocokan(solusi_terbaik))
```

```
Generasi: 0 Solusi Terbaik: [9.99788944] Kecocokan: 952.1703216948223
Generasi: 1 Solusi Terbaik: [19.15558747] Kecocokan: [913.4440347]
Generasi: 2 Solusi Terbaik: [4.04294062] Kecocokan: [775.43089077]
Generasi: 3 Solusi Terbaik: [5.92985088] Kecocokan: [633.32578159]
Generasi: 4 Solusi Terbaik: [9.35349001] Kecocokan: [872.64257127]
Generasi: 5 Solusi Terbaik: [23.17274211] Kecocokan: [922.94383978]
Generasi: 6 Solusi Terbaik: [2.31573469] Kecocokan: [873.58961641]
Generasi: 7 Solusi Terbaik: [29.68626238] Kecocokan: [937.33871219]
Generasi: 8 Solusi Terbaik: [2.78575838] Kecocokan: [929.83388993]
Generasi: 9 Solusi Terbaik: [1.99653292] Kecocokan: [467.99772829]
Generasi: 10 Solusi Terbaik: [9.78119971] Kecocokan: [640.70265646]
Generasi: 11 Solusi Terbaik: [2.31722137] Kecocokan: [898.16825514]
Generasi: 12 Solusi Terbaik: [2.31573469] Kecocokan: [796.88603262]
Generasi: 13 Solusi Terbaik: [2.53983511] Kecocokan: [954.84219578]
Generasi: 14 Solusi Terbaik: [2.68247124] Kecocokan: [913.45215528]
Generasi: 15 Solusi Terbaik: [4.70688108] Kecocokan: [847.62272589]
Generasi: 16 Solusi Terbaik: [2.53983511] Kecocokan: [905.94438285]
Generasi: 17 Solusi Terbaik: [2.31573469] Kecocokan: [937.01315511]
Generasi: 18 Solusi Terbaik: [8.31464044] Kecocokan: [711.62030718]
Generasi: 19 Solusi Terbaik: [2.53983511] Kecocokan: [741.53442925]
Generasi: 20 Solusi Terbaik: [2.31722137] Kecocokan: [850.58780748]
Generasi: 21 Solusi Terbaik: [10.20942613] Kecocokan: [851.87325335]
Generasi: 22 Solusi Terbaik: [2.31573469] Kecocokan: [949.28204404]
Generasi: 23 Solusi Terbaik: [2.53902278] Kecocokan: [943.02977301]
Generasi: 24 Solusi Terbaik: [0.6360283] Kecocokan: [689.12287206]
Generasi: 25 Solusi Terbaik: [2.31573469] Kecocokan: [946.17224345]
Generasi: 26 Solusi Terbaik: [0.87445607] Kecocokan: [633.56243137]
Generasi: 27 Solusi Terbaik: [0.87445607] Kecocokan: [392.82358419]
Generasi: 28 Solusi Terbaik: [0.87445607] Kecocokan: [887.56086164]
Generasi: 29 Solusi Terbaik: [2.31573469] Kecocokan: [934.58028006]
Generasi: 30 Solusi Terbaik: [0.87445607] Kecocokan: [811.48165342]
Generasi: 31 Solusi Terbaik: [0.87445607] Kecocokan: [801.06950131]
Generasi: 32 Solusi Terbaik: [0.87445607] Kecocokan: [945.7400756]
...
Generasi: 73 Solusi Terbaik: [20.33765886] Kecocokan: [598.74755446]
Generasi: 74 Solusi Terbaik: [1.15940223] Kecocokan: [588.59861499]
Generasi: 75 Solusi Terbaik: [20.36566543] Kecocokan: [905.61059934]
Generasi: 76 Solusi Terbaik: [1.1790212] Kecocokan: [940.11870961]
Generasi: 77 Solusi Terbaik: [9.84010103] Kecocokan: [777.21771594]
Generasi: 78 Solusi Terbaik: [1.15940223] Kecocokan: [940.5662037]
Generasi: 79 Solusi Terbaik: [4.08917739] Kecocokan: [864.21136137]
Generasi: 80 Solusi Terbaik: [1.15940223] Kecocokan: [896.31388861]
Generasi: 81 Solusi Terbaik: [5.40925182] Kecocokan: [944.71594429]
Generasi: 82 Solusi Terbaik: [27.75309872] Kecocokan: [934.74447149]
Generasi: 83 Solusi Terbaik: [0.00913579] Kecocokan: [934.56740745]
Generasi: 84 Solusi Terbaik: [1.15940223] Kecocokan: [810.94925411]
Generasi: 85 Solusi Terbaik: [1.15940223] Kecocokan: [725.78934082]
Generasi: 86 Solusi Terbaik: [1.1790212] Kecocokan: [710.3460153]
Generasi: 87 Solusi Terbaik: [1.15940223] Kecocokan: [792.23392653]
Generasi: 88 Solusi Terbaik: [16.08791748] Kecocokan: [947.51982731]
Generasi: 89 Solusi Terbaik: [4.70688108] Kecocokan: [683.96143272]
Generasi: 90 Solusi Terbaik: [9.84010103] Kecocokan: [840.70328115]
Generasi: 91 Solusi Terbaik: [4.96532683] Kecocokan: [889.79737581]
Generasi: 92 Solusi Terbaik: [14.29262636] Kecocokan: [785.74778139]
Generasi: 93 Solusi Terbaik: [1.15940223] Kecocokan: [849.77199541]
Generasi: 94 Solusi Terbaik: [1.15940223] Kecocokan: [757.90111282]
Generasi: 95 Solusi Terbaik: [1.15940223] Kecocokan: [784.23494747]
Generasi: 96 Solusi Terbaik: [30.91017549] Kecocokan: [860.81577788]
Generasi: 97 Solusi Terbaik: [15.72682074] Kecocokan: [955.69046393]
Generasi: 98 Solusi Terbaik: [1.15940223] Kecocokan: [861.79527421]
Generasi: 99 Solusi Terbaik: [9.21657961] Kecocokan: [834.87987406]
Solusi Terbaik yang Ditemukan: [9.21657961]
Nilai Maksimum dari Fungsi: [-74.94533971]
```

Penejelasan :

1. Import Library:
   - Menggunakan library NumPy untuk operasi matematika dan array.

2. Fungsi-fungsi Utama:

   - fungsi_kecocokan(x): Menghitung nilai kecocokan (fitness) dari suatu individu (x) berdasarkan fungsi ( )=−(2)+10f(x)=−(x2)+10.
   - inisialisasi_populasi(ukuran_populasi): Menginisialisasi populasi awal dengan nilai-nilai acak dalam rentang 00 hingga 3131.
   - crossover(induk1, induk2): Melakukan operasi crossover antara dua individu untuk menghasilkan dua anak baru.
   - mutasi(individu, tingkat_mutasi): Melakukan operasi mutasi pada individu dengan tingkat mutasi tertentu.
   - algoritma_genetika(ukuran_populasi, tingkat_mutasi, maks_generasi): Fungsi utama yang mengimplementasikan algoritma genetika.

3. Proses Algoritma Genetika:
   - Inisialisasi populasi awal.
   - Evaluasi kecocokan (fitness) dari setiap individu dalam populasi.
   - Seleksi individu untuk reproduksi berdasarkan nilai kecocokannya.
   - Reproduksi dengan melakukan operasi crossover dan mutasi.
   - Iterasi proses seleksi, reproduksi, dan evaluasi untuk sejumlah maksimum generasi.
   - Mengambil solusi terbaik dari individu dengan kecocokan terbesar dalam populasi.

4. Output
   - Mencetak solusi terbaik yang ditemukan.
   - Mencetak nilai maksimum dari fungsi tersebut pada solusi terbaik.