

LAPORAN TUGAS PROGRAM 1  
*ARTIFICIAL INTELLIGENCE*  
*SIMULATED ANNEALING*



Nama : I Putu Indra Aristya

NIM : 1301154219

Kelas : IF39-09

Mata Kuliah : Kecerdasan Buatan

Universitas Telkom

BANDUNG

2017

## I. Deskripsi Masalah

Dalam tugas program-1 ini, kasus yang diberikan adalah harus mencari minimum dari sebuah fungsi yang diberikan. Fungsinya adalah sebagai berikut.

$$f(x_1, x_2) = \left(4 - 2,1x_1^2 + \frac{x_1^4}{3}\right) x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$$

dengan batasan dimana  $x_1$  dan  $x_2$  harus bernilai diantara -10 dengan 10 (tidak boleh lebih besar dari 10 dan tidak boleh lebih kecil dari -10). Maka, kita harus dapat menentukan nilai  $x_1, x_2$  yang akan membuat fungsi tersebut bernilai minimal (paling kecil). Algoritma yang digunakan adalah *Simulated Annealing*.

## II. Landasan Teori

Algoritma *Simulated Annealing* adalah salah satu algoritma pencarian atau biasa disebut dalam ilmu kecerdasan buatan adalah algoritma *searching*. Dalam algoritma ini, perbedaannya dengan algoritma lainnya adalah terdapat probabilitas yang membantu pencarian agar tidak terjebak dengan hasil yang didapat namun tidak maksimal (biasa disebut dengan hasil *local minimum*). *Local minimum* yang dimaksud adalah suatu titik terendah yang ada di dalam grafik, namun sebenarnya terdapat titik yang lebih rendah lagi (biasa disebut dengan hasil *global minimum*) tetapi tidak terdeteksi karena setelah titik *local minimum*, terdapat tanjakan/kenaikan nilai yang terjadi. Dalam algoritma ini, probabilitas dapat digunakan. Rumus probabilitasnya adalah sebagai berikut.

$$p(\Delta E) = e^{-\frac{\Delta E}{T}} ; \text{dimana } \Delta E = E(\text{state baru}) - E(\text{state lama})$$

T = temperatur (mempengaruhi jumlah iterasi)  
E = nilai evaluasi dari state  $x_1$  dan  $x_2$

### III. Code Program

Dalam tugas program-1 ini, saya menggunakan bahasa pemrograman *Python* dengan IDE *PyCharm CE*. Dalam program ini, saya menentukan parameter temperatur adalah 500 dengan pengurangan tiap iterasi sejumlah 0.5. Pemilihan parameter tersebut karena dengan menggunakan parameter seperti yang disebutkan, hasil kemunculan nilai minimum (kurang dari 1) lebih sering daripada sebelumnya. Berikut adalah *code* program yang sudah saya buat.

```
import random
import math

def evaluateX(x,y):
    resultt = (((4 - (2.1*(x**2)) + ((x**4)/3)) * (x**2)) + (x*y) + ((-4 +
(4*(y**2)))*(y**2)))
    print ("Current X1: %f" %x)
    print ("f(x1,x2): %f" %resultt)
    return resultt

def evaluateNewXY(x,y):
    result = (((4 - (2.1*(x**2)) + ((x**4) / 3)) * (x ** 2)) + (x * y) + ((-4 +
(4 * (y ** 2))) * (y ** 2)))
    print ("New X1: %f" %x)
    print ("New X2: %f" %y)
    print ("f(x1,x2): %f" %result)
    return result

def generateNewXY(currentCoor):
    newstate = currentCoor - random.uniform(-2,2)
    while (-10>newstate or newstate>10):
        newstate = newstate + random.uniform(-2, 2)
    return newstate

def probability(delE,iter):
    return math.exp((-1*delE)/iter)

Cx = random.uniform(-10,10)
Cy = random.uniform(-10,10)

T = 500

bestX = Cx
bestY = Cy

i = 1;
```

```

while (T!=0):
    print ("LOOP KE-%d" %i)

    E1 = evaluateX(Cx, Cy) #evaluate current state to get the value of f(x,y)

    newStateX = generateNewXY(Cx) #generate new state
    newStateY = generateNewXY(Cy) #generate new state

    E2 = evaluateNewXY(newStateX,newStateY) #evaluate new state to get the value
    of f(x,y)

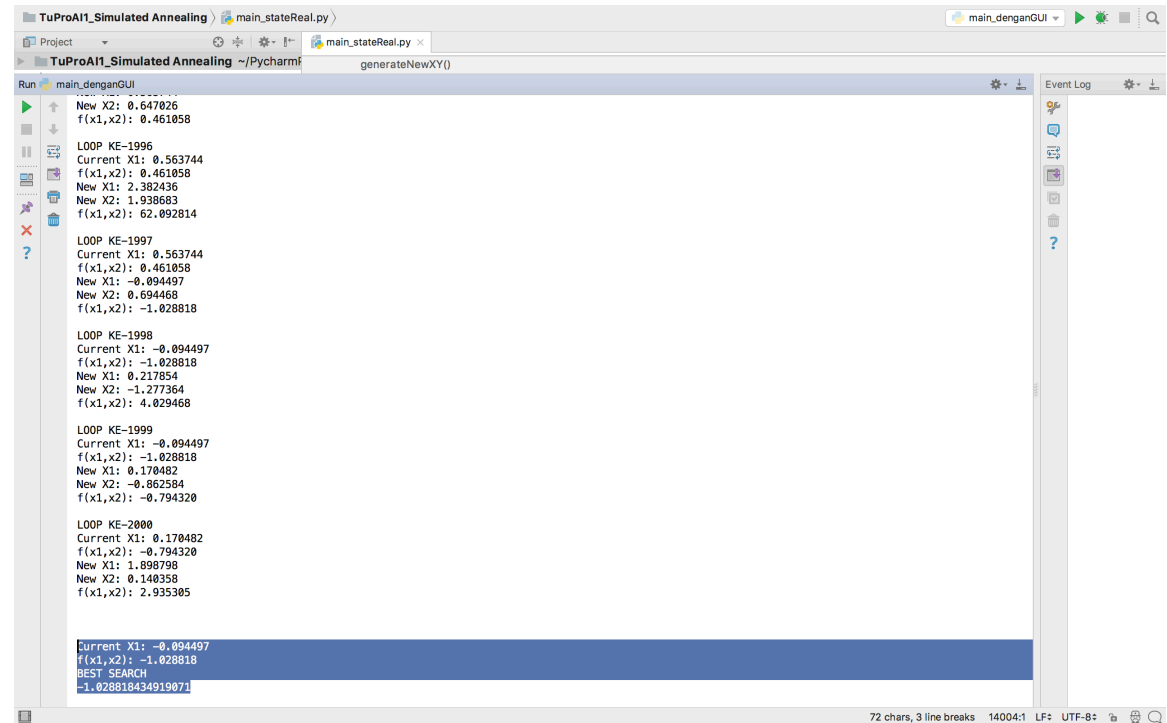
    if (E2 < E1): #compare; if new state better than current state (f(x,y)new <
    f(x,y)current
        Cx = newStateX
        Cy = newStateY
        bestX = newStateX
        bestY = newStateY
    else:
        deltaE = (E2 - E1)
        p = probability(deltaE,T)
        r = random.uniform(0,1)
        if (r < p):
            Cx = newStateX
            Cy = newStateY
    i = i+1
    T = T - 0.5
    print("")

print("")
print("")
bestF = evaluateX(bestX,bestY)
print("BEST SEARCH")
print(bestF)

```

#### IV. Hasil Keluaran Program dan Nilai Minimal

Berikut adalah hasil keluaran dari program yang dijalankan dan nilai minimal yang pernah di dapat.



```
TuProAI1_Simulated Annealing > main_stateReal.py >
main_stateReal.py x
TuProAI1_Simulated Annealing ~\Pycharm generateNewXY()
Run main_denganGUI
New X2: 0.647826
f(x1,x2): 0.461858
LOOP KE-1996
Current X1: 0.563744
f(x1,x2): 0.461858
New X1: 2.382436
New X2: 1.938683
f(x1,x2): 62.092814
LOOP KE-1997
Current X1: 0.563744
f(x1,x2): 0.461858
New X1: -0.094497
New X2: 0.694468
f(x1,x2): -1.028818
LOOP KE-1998
Current X1: -0.094497
f(x1,x2): -1.028818
New X1: 0.217854
New X2: -1.277364
f(x1,x2): 4.029468
LOOP KE-1999
Current X1: -0.094497
f(x1,x2): -1.028818
New X1: 0.178482
New X2: -0.862584
f(x1,x2): -0.794320
LOOP KE-2000
Current X1: 0.178482
f(x1,x2): -0.794320
New X1: 1.898798
New X2: 0.148358
f(x1,x2): 2.935305
Current X1: -0.094497
f(x1,x2): -1.028818
BEST SEARCH
-1.028818434919071
72 chars, 3 line breaks 14004:1 LF: UTF-8
```

Gambar 1 Hasil keluaran adalah -1.028818434919071

```
31 return newstate
32
33
34 def probability(deLE, iter):
35     return math.exp((-1*deLE)/iter)
36
37
38 Cx = random.uniform(-10,10)
39 Cy = random.uniform(-10,10)
40
41 T = 500
42
43 bestX = Cx
44 bestY = Cy
45
46 i = 1
47
48 while (T!=0):
49     print ("LOOP KE-~d" %i)
50
51     E1 = evaluateX(Cx, Cy) #evaluate current state to get the value of f(x,y)
52
53     newStateX = generateNewXY(Cx) #generate new state
54     newStateY = generateNewXY(Cy) #generate new state
55
56     E2 = evaluateNewXY(newStateX, newStateY) #evaluate new state to get the value of f(x,y)
57
58     if (E2 < E1): #compare; if new state better than current state (f(x,y)new < f(x,y)current)
59         Cx = newStateX
60         Cy = newStateY
61         bestX = newStateX
62         bestY = newStateY
```

Run main\_denganGUI

New X2: -0.040895  
f(x1,x2): 2.161328

Current X1: 0.023346  
Current X2: 0.842333  
f(x1,x2): -0.802553  
BEST SEARCH  
-0.8025531634915103

Process finished with exit code 0

**Gambar 2 Hasil Keluaran adalah -0.8025531634915103**

```
32 return newstate
33
34 def probability(deLE, iter):
35     return math.exp((-1*deLE)/iter)
36
37
38 Cx = random.uniform(-10,10)
39 Cy = random.uniform(-10,10)
40
41 T = 500
42
43 bestX = Cx
44 bestY = Cy
45
46 i = 1
47
48 while (T!=0):
49     print ("LOOP KE-~d" %i)
50
51     E1 = evaluateX(Cx, Cy) #evaluate current state to get the value of f(x,y)
52
53     newStateX = generateNewXY(Cx) #generate new state
54     newStateY = generateNewXY(Cy) #generate new state
55
56     E2 = evaluateNewXY(newStateX, newStateY) #evaluate new state to get the value of f(x,y)
57
58     if (E2 < E1): #compare; if new state better than current state (f(x,y)new < f(x,y)current)
59         Cx = newStateX
60         Cy = newStateY
61         bestX = newStateX
62         bestY = newStateY
```

Run main\_denganGUI

New X2: 0.515506  
f(x1,x2): -0.721718

Current X1: -0.285043  
Current X2: 0.515506  
f(x1,x2): -0.721718  
BEST SEARCH  
-0.7217184348523007

Process finished with exit code 0

**Gambar 3 Hasil Keluaran adalah -0.7217184348523007**

## V. Kesimpulan

Dari hasil yang didapat, saya simpulkan bahwa untuk menemukan hasil *global minimum* dengan algoritma *Simulated Annealing* dipengaruhi oleh besar parameter yang digunakan, yaitu temperatur dan operator untuk mendapat state baru. Selain itu, probabilitas yang ada juga berguna agar membuat hasil yang didapat adalah hasil *global minimum*.

# Daftar Pustaka

Suyanto. (2014). *Artificial Intelligence* (Vol. 2). Bandung: Informatika Bandung.