

Laporan Algoritma Optimisasi

Week VI

Penempatan Kepala Sekolah Dengan Simulated Annealing

Dosen Pengampu : Drs.Haris Sriwindono M.Kom, Ph.D.



Disusun oleh

Nama : Dyline Melynea Fernandez

NIM : 185314125

PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS & TEKNOLOGI
UNIVERSITAS SANATA DHARMA

1. Disain struktur data

Dominan pada program ini menggunakan struktur data list (di python dilambangkan dengan []). Selain list juga digunakan dict untuk memasukkan list-list yang ada bersesuaian dengan nama header kolom yang akan digunakan menjadi DataFrame.

Berikut digambarkan struktur data dari variabel/objek penting pada hasil akhir:

List singkatan (untuk konfigurasi)

List singkatan tersusun dari beberapa list 1D-berdimensi-2.

	S-T-RT-JF-P-RL	S-JF-RT-T-P-RL	JF-T-RT-S-P-RL	...
Index	0	1	2	...

List total_distance

	349	358	394	...
Index	0	1	2	...

List t

	331.55	331.55	340.10	...
Index	0	1	2	...

Ketiga variabel/objek list tersebut terpisah, namun dapat dihubungkan menggunakan posisi indexnya.

Dict bahan

Keys	Values
'Placement'	singkatan
'Total Distance'	Total_distance
'Temperature'	T

Values diambil dari 3 list di atas

2. Jalan algoritma secara manual

Inisiasi

Konfigurasi awal : S-T-RT-JF-P-RL

Total distance : 349

Temperature : $0.95 * 349 = 331.55$

Iterasi 1

Konfig : S-JF-RT-T-P-RL
Total distance : 358
Bil r : 0.5
p : $e^{\frac{349-358}{331.55}} = 0.97322$
Temperature : $0.95 \cdot 349 = 331.55$
 $\Rightarrow r < p$, konfigurasi iterasi 1 diterima

Iterasi 2

Konfig : JF-T-RT-S-P-RL
Total distance : 394
Bil r : 0.212
p : $e^{\frac{358-394}{331.55}} = 0.897106$
Temperature : $0.95 \cdot 358 = 340.1$
 $\Rightarrow r < p$, konfigurasi iterasi 2 diterima

Iterasi 3

Konfig : P-T-RT-JF-S-RL
Total distance : 273
Temperature : $0.95 \cdot 394 = 374.2$
 $\Rightarrow d_3 < d_2$, konfigurasi iterasi 3 diterima

Iterasi 4

Konfig : RL-T-RT-JF-P-S
Total distance : 401
Bil r : 0.296
p : $e^{\frac{273-401}{374.2}} = 0.710303$
Temperature : $0.95 \cdot 273 = 259.35$
 $\Rightarrow r < p$, konfigurasi iterasi 4 diterima

Iterasi 5

Konfig : T-S-RT-JF-P-RL
Total distance : 357
Temperature : $0.95 \cdot 401 = 380.95$
 $\Rightarrow d_5 < d_4$, konfigurasi iterasi 5 diterima

Iterasi 6

Konfig : RT-T-S-JF-P-RL
Total distance : 348
Temperature : $0.95 \times 357 = 339.15$
 $\Rightarrow d_6 < d_5$, konfigurasi iterasi 6 diterima

Iterasi 7

Konfig : S-T-RT-JF-RL-P
Total distance : 238
Temperature : $0.95 \times 348 = 330.6$
 $\Rightarrow d_7 < d_6$, konfigurasi iterasi 7 diterima

Iterasi 8

Konfig : S-RT-T-JF-P-RL
Total distance : 401
Bil r : 0.349
p : $e^{\frac{238-401}{330.6}} = 0.610765$
Temperature : $0.95 \times 238 = 226.1$
 $\Rightarrow r < p$, konfigurasi iterasi 8 diterima

Iterasi 9

Konfig : RL-T-RT-JF-P-S
Total distance : 401
Bil r : 0.457
p : $e^{\frac{401-401}{226.1}} = 1$
Temperature : $0.95 \times 401 = 380.95$
 $\Rightarrow r < p$, konfigurasi iterasi 9 diterima

Iterasi 10

Konfig : S-T-RL-JF-P-RT
Total distance : 345
Temperature : $0.95 \times 401 = 380.95$
 $\Rightarrow d_{10} < d_9$, konfigurasi iterasi 10 diterima

3. Listing program

cell 1:

```
import pandas as pd
import numpy as np
import random
import math
import copy
```

cell 2:

```
data = pd.read_excel("kepsekfixgarandom.xls")
data.head()
```

cell 3:

```
s = data["Kode_Sekolah"].drop_duplicates()
s.reset_index(drop=True, inplace = True)
```

cell 4:

```
data_konfig = data.iloc[0:0,:].copy()
isi = data.iloc[0:0,:].copy()
guru = ''
for i in s:
    if(data_konfig.empty):
        isi = data[data["Kode_Sekolah"]==i].sample()
        guru = str(isi["Nama Guru"].any())
        data_konfig = data_konfig.append(isi)
    else:
        while(guru in str(data_konfig["Nama Guru"].values)):
            isi = data[data["Kode_Sekolah"]==i].sample()
            guru = str(isi["Nama Guru"].any())
            data_konfig = data_konfig.append(isi)
data_konfig
```

cell 5:

```
def new_konf_rand_check(konfig, placement):
    w = []
    w_copy = []
    kode = []
    idx_source = []

    cek = konfig["Nama Guru"].to_numpy()
    r_pick = random.sample(set(cek), 2)
    r_pick

    new_konf=konfig.copy()
    new_konf.reset_index(drop=True, inplace = True)

    for i in range(2):
        nilai= np.where(cek == r_pick[i])
```

```

        nilai = nilai[0][0]
        w.append(nilai)

w_copy = w.copy()
temp = w[0]
w[0] = w[1]
w[1] = temp

for i in range(2):
    if(w[i] == 0):
        kode.append('S1')
    elif(w[i]==1):
        kode.append('S2')
    elif(w[i]==2):
        kode.append('S3')
    elif(w[i]==3):
        kode.append('S4')
    elif(w[i]==4):
        kode.append('S5')
    elif(w[i]==5):
        kode.append('S6')

    condition = (data["Kode_Sekolah"]==kode[i]) & (data["Nama Guru"]==r
_pick[i])
    idx_source.append(data.index[condition].tolist()[0])
    new_konf.iloc[w_copy[i]] = data.loc[idx_source[i]]

new_konf = new_konf.sort_values(by=["Kode_Sekolah"],ascending = True)
new_konf.reset_index(drop=True, inplace = True)
#new_konf_name = new_konf["Nama Guru"].values.tolist()
if(new_konf["Nama Guru"].values.tolist() in placement):
    new_konf_rand_check(konfig,placement)
elif(new_konf["Nama Guru"].values.tolist() not in placement):
    #new_konf_namefix = new_konf_name
    new_konf = pd.DataFrame(new_konf, columns=['Nomor Urut', 'Kode_Seko
lah', 'Nama Sekolah',
                                                    'Nomor_guru', 'Nama Guru
', 'Jarak rmh-sek'])
    return new_konf

```

cell 6:

```

def simulated_annealing(iteration, a):
    konfig = data_konfig
    placement = []
    total_distance = []
    t = []
    cek_placement_awal = []
    new_konfig = konfig.iloc[0:0,:].copy()

```

```

placement.append(konfig["Nama Guru"].tolist())
total_distance.append(konfig["Jarak rmh-sek"].sum())
t.append(a*total_distance[0])

for i in range(iteration):
    new_konfig = new_konf_rand_check(konfig, placement)
    #atarashi.append(new_konfig)
    total_distance.append(new_konfig["Jarak rmh-sek"].sum())

    if(total_distance[i+1] < total_distance[i]):
        placement.append(new_konfig["Nama Guru"].tolist())
        t.append(a*total_distance[i])
    elif(total_distance[i+1] >= total_distance[i]):
        r = random.randint(0,100)/100
        p = math.exp(total_distance[i]-total_distance[i+1]/t[i])
        if(r<p):
            placement.append(new_konfig["Nama Guru"].tolist())
            t.append(a*total_distance[i])
        else:
            placement.append(placement[i])
            t.append(t[i])
            total_distance[-1] = total_distance[i]

print(t)
print(total_distance)
print(placement)
pl = copy.deepcopy(placement)
singkatan = []
for i in range(11):
    for j in range(6):
        if(pl[i][j]=='Sriningsih'):
            pl[i][j]='S'
        elif(pl[i][j]=='Rianny Linthin'):
            pl[i][j]='RL'
        elif(pl[i][j]=='Paijo'):
            pl[i][j]='P'
        elif(pl[i][j]=='Richard Tumewu'):
            pl[i][j]='RT'
        elif(pl[i][j]=='John Fonataba'):
            pl[i][j]='JF'
        elif(pl[i][j]=='Taufan'):
            pl[i][j]='T'
    singkatan.append(pl[i][0]+" — "+
                    pl[i][1]+" — "+
                    pl[i][2]+" — "+
                    pl[i][3]+" — "+
                    pl[i][4]+" — "+

```

```

        pl[i][5])

    bahan = {'Placement': singkatan, 'Total Distance': total_distance, 'Temperature': t}
    SA = pd.DataFrame(bahan, columns= ['Placement', 'Total Distance', 'Temperature'])
    return SA, cek_placement_awal, placement

```

cell 7:

```

hasil, awalmula, pnmpn = simulated_annealing(10,0.95)
hasil

```

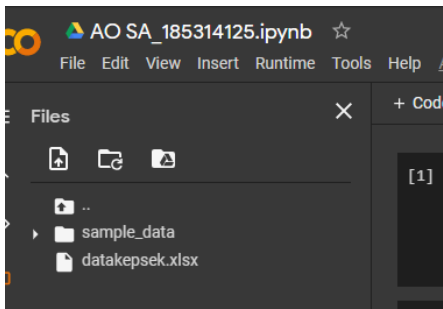
cell 8:

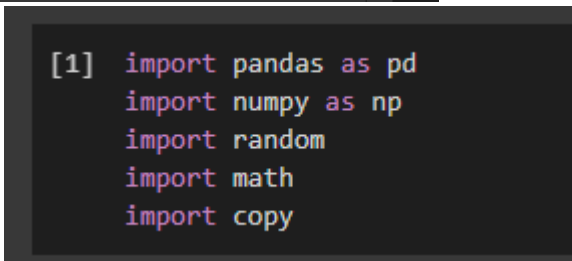
```

hasil[hasil["Total Distance"]==hasil["Total Distance"].min()]

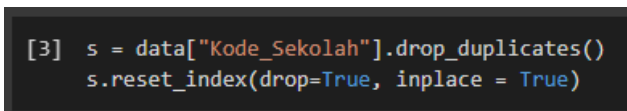
```

4. Screenshot running program

1.  ⇒ Upload file dataset ke folder google colab

2.  ⇒ Impor library yang diperlukan

3.  ⇒ Baca dataset dengan pandas, dan melihat beberapa data di atasnya untuk memastikan dataset yang dibaca benar yang dimaksud.

4.  ⇒ Preprocessing (mengambil data kode sekolah)

5.

```
[4] data_konfig = data.iloc[0:0,:].copy()
isi = data.iloc[0:0,:].copy()
guru = ''
for i in s:
    if(data_konfig.empty):
        isi = data[data["Kode_Sekolah"]==i].sample()
        guru = str(isi["Nama Guru"].any())
        data_konfig = data_konfig.append(isi)
    else:
        while(guru in str(data_konfig["Nama Guru"].values)):
            isi = data[data["Kode_Sekolah"]==i].sample()
            guru = str(isi["Nama Guru"].any())
            data_konfig = data_konfig.append(isi)
data_konfig
```

	Nomor Urut	Kode_Sekolah	Nama Sekolah	Nomor_guru	Nama Guru	Jarak rmh-sek
4	5	S1	SD PL1	G0005	Sriningsih	57
11	12	S2	SD KNS 1	G0006	Taufan	41
14	15	S3	SD PL2	G0003	Richard Tumewu	32
18	19	S4	SD KNS2	G0001	John Fonataba	46
27	28	S5	SD PL3	G0004	Paijo	96
31	32	S6	SD KLSN1	G0002	Rianny Linthin	77

⇒ Mengambil secara acak kandidat-kandidat kepek berdasarkan tiap kode sekolah. Ini merupakan konfigurasi awal yang menjadi bahan untuk dirandom ke depannya

6.

```
[5] def new_konf_rand_check(konfig, placement):
    w = []
    w_copy = []
    kode = []
    idx_source = []

    cek = konfig["Nama Guru"].to_numpy()
    r_pick = random.sample(set(cek), 2)
    r_pick

    new_konf=konfig.copy()
    new_konf.reset_index(drop=True, inplace = True)

    for i in range(2):
        nilai= np.where(cek == r_pick[i])
        nilai = nilai[0][0]
        w.append(nilai)

    w_copy = w.copy()
    temp = w[0]
    w[0] = w[1]
    w[1] = temp
```

⇒ Bagian atas method/function **new_konf_rand_check** dengan parameter yang diharapkan adalah konfigurasi awal dan list placement-placement yang terbentuk sebagai alat pengecekan. Bagian atas method ini bekerja untuk mengambil 2 sample acak kepala sekolah yang ada, kemudian diswap posisi sekolahnya.

7.

```
for i in range(2):
    if(w[i] == 0):
        kode.append('S1')
    elif(w[i]==1):
        kode.append('S2')
    elif(w[i]==2):
        kode.append('S3')
    elif(w[i]==3):
        kode.append('S4')
    elif(w[i]==4):
        kode.append('S5')
    elif(w[i]==5):
        kode.append('S6')

    condition = (data["Kode_Sekolah"]==kode[i]) & (data["Nama Guru"]==r_pick[i])
    idx_source.append(data.index[condition].tolist()[0])
    new_konf.iloc[w_copy[i]] = data.loc[idx_source[i]]

new_konf = new_konf.sort_values(by=["Kode_Sekolah"],ascending = True)
new_konf.reset_index(drop=True, inplace = True)
#new_konf_name = new_konf["Nama Guru"].values.tolist()
if(new_konf["Nama Guru"].values.tolist() in placement):
    new_konf_rand_check(konfig,placement)
elif(new_konf["Nama Guru"].values.tolist() not in placement):
    #new_konf_namefix = new_konf_name
    new_konf = pd.DataFrame(new_konf, columns=['Nomor Urut', 'Kode_Sekolah','Nama Sekolah',
                                                'Nomor_guru', 'Nama Guru', 'Jarak rmh-sek'])

return new_konf
```

- ⇒ Bagian tengah ke bawah adalah mengambil entri data dari dataframe sebelum dipreprocessing, dan hasil tersebut menggantikan 2 baris yang terpilih sebelumnya (posisi sekolah ditukar, sehingga perlu menyesuaikan value-value kolom lain selain posisi sekolah dan nama guru).
- ⇒ Kemudian juga terdapat pengecekan apakah konfigurasi yang terbentuk setelah menswap sudah ada di list **placement** atau belum. Jika sudah maka akan rekursif mengerjakan method ini lagi sampai akhirnya masuk ke kondisi elif yang menyimpan konfigurasi baru tersebut untuk diproses di method khusus proses simulated annealing.

8.

```
def simulated_annealing(iteration, a):
    konfig = data_konfig
    placement = []
    total_distance = []
    t = []
    cek_placement_awal = []
    new_konfig = konfig.iloc[0:0,:].copy()

    placement.append(konfig["Nama Guru"].tolist())
    total_distance.append(konfig["Jarak rmh-sek"].sum())
    t.append(a*total_distance[0])
```

- ⇒ Method/function **simulated_annealing** bagian atas dengan parameter yang diharapkan yaitu jumlah iterasi, dan nilai alpha. Pada bagian atas dari method/function ini, dilakukan persiapan perhitungan untuk konfigurasi awal (total_distance, temperature, placement).

9.

```
for i in range(iteration):
    new_konfig = new_konf_rand_check(konfig, placement)
    #atarashi.append(new_konfig)
    total_distance.append(new_konfig["Jarak rmh-sek"].sum())

    if(total_distance[i+1] < total_distance[i]):
        placement.append(new_konfig["Nama Guru"].tolist())
        t.append(a*total_distance[i])
    elif(total_distance[i+1] >= total_distance[i]):
        r = random.randint(0,100)/100
        p = math.exp(total_distance[i]-total_distance[i+1]/t[i])
        if(r<p):
            placement.append(new_konfig["Nama Guru"].tolist())
            t.append(a*total_distance[i])
        else:
            placement.append(placement[i])
            t.append(t[i])
            total_distance[-1] = total_distance[i]
```

⇒ Pada bagian menengah daerah atas dari method **simulated_annealing** berisi perhitungan total_distance, temperature dan storing konfig nama baru ke list placement untuk ke-10 iterasi (konfig awal tidak termasuk, karena sudah didefinisikan duluan di luar loop). Kemudian juga pada bagian ini berisi rule-rule suatu konfigurasi diterima/tidak. Jika $d_2 < d_1$ maka konfigurasi baru langsung diterima, jika $d_2 \geq d_1$ maka konfigurasi baru memiliki potensi diterima dengan syarat nilai probabilitasnya lebih besar dari bilangan desimal random dengan range 0-1 yang digenerate otomatis ($r < p$). Rule ini beracu pada skripsi dari repositori usu yang akan disertakan linknya di bagian referensi.

10.

```
print(t)
print(total_distance)
print(placement)
pl = copy.deepcopy(placement)
singkatan = []
for i in range(11):
    for j in range(6):
        if(pl[i][j]=='Sriningsih'):
            pl[i][j]='S'
        elif(pl[i][j]=='Rianny Linthin'):
            pl[i][j]='RL'
        elif(pl[i][j]=='Paijo'):
            pl[i][j]='P'
        elif(pl[i][j]=='Richard Tumewu'):
            pl[i][j]='RT'
        elif(pl[i][j]=='John Fonataba'):
            pl[i][j]='JF'
        elif(pl[i][j]=='Taufan'):
            pl[i][j]='T'
    singkatan.append(pl[i][0]+" — "+
                    pl[i][1]+" — "+
                    pl[i][2]+" — "+
                    pl[i][3]+" — "+
                    pl[i][4]+" — "+
                    pl[i][5])
```

⇒ Bagian tengah area bawah method/function **simulated_annealing** dimaksudkan untuk merapihkan format list placement agar tidak terlalu panjang tulisan rutenya nanti. Di bagian ini ada juga mencetak beberapa variabel untuk sebagai informasi saja

11.

```
bahan = {'Placement': singkatan, 'Total Distance': total_distance, 'Temperature': t}
SA = pd.DataFrame(bahan, columns= ['Placement', 'Total Distance', 'Temperature'])
return SA, cek_placement_awal, placement
```

⇒ Bagian bawah method/fuction **simulated_annealing** dimaksudkan untuk memasukkan informasi-informasi seperti **placement**, **total_distance**, dan **t** ke suatu dataframe. Dan hasil return method/function ini yaitu berupa DataFrame dari 3 informasi yang baru saja disebutkan, konfigurasi sebelum terkena rule, dan list placement sebelum dirapihkan.

12.

```
[7] hasil, awalmula, pnmpn = simulated_annealing(10,0.95)
hasil

[331.55, 331.55, 340.09999999999997, 374.29999999999995, 259.34999999999997, 380.95, 339.15, 330.59999999999997, 226.1, 349, 358, 394, 273, 401, 357, 348, 238, 401, 345]
[['Sriningsih', 'Taufan', 'Richard Tumewu', 'John Fonataba', 'Paijo', 'Rianny Linthin'], ['Sriningsih', 'John Fonataba']]

Placement Total Distance Temperature
0 S — T — RT — JF — P — RL 349 331.55
1 S — JF — RT — T — P — RL 358 331.55
2 JF — T — RT — S — P — RL 394 340.10
3 P — T — RT — JF — S — RL 273 374.30
4 RL — T — RT — JF — P — S 401 259.35
5 T — S — RT — JF — P — RL 357 380.95
6 RT — T — S — JF — P — RL 348 339.15
7 S — T — RT — JF — RL — P 238 330.60
8 S — RT — T — JF — P — RL 401 226.10
9 RL — T — RT — JF — P — S 401 380.95
10 S — T — RL — JF — P — RT 345 380.95
```

⇒ Pemanggilan method/function **simulated_annealing** dengan masukan parameter 10 untuk jumlah iterasi dan 0.95 untuk nilai alpha. Sebelum DataFrame tertampil, terdapat informasi semi-raw dicetak.

13.

```
hasil[hasil["Total Distance"]==hasil["Total Distance"].min()]

Placement Total Distance Temperature
7 S — T — RT — JF — RL — P 238 330.6
```

⇒ Menampilkan entri data berdasarkan **Total Distance** terkecil dari sejumlah 11 konfigurasi (10 iterasi konfigurasi+konfigurasi awal).

Hasil sama dengan pengerjaan manual (tidak mencantumkan r dan p karena hanya bermaksud untuk mengecek program untuk skenario-skenario placement, penghitungan distance dan temperature sudah sesuai atau belum. Nilai r dan p di program tidak dimasukkan di list sehingga tidak terlihat tracenya, dan sepertinya informasi yang diperlukan adalah 3 informasi seperti yang tertampil). Pengerjaan manual disesuaikan dengan yang terjadi di program (mencocokkan placement list dan data raw) agar proses dan hasil berjalan ke arah yang sama, karena di program menggunakan sample random, jadi jika pengerjaan manual tidak menyesuaikan dengan program, maka tentu saja tidak dapat menentukan program sudah berjalan sesuai harapan atau belum.

<https://drive.google.com/file/d/1GJgx4agxp4USZi5PQ7RL0QIU83vhNEkG/view?usp=sharing> (Hal 38)