Implementasi Algoritma Greedy pada Pendataan Lokasi Suatu Kota

Jesika Putri-120140050 (*Author*)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Sumatera, Jl.Terusan Ryacudu Lampung
E-mail (gmail): jesika.120140050@student.itera.ac.id

***Abstract*—lokasi suatu daerah adalah hal yang snagat diperlukan. Dengan adanya lokasi yang jelas suatu daerah bisa ditemukan dengan cepat dan bisa di data dengan baik yang tujuan utamanya adalah untuk mempermudah akses suatu daerah.**

Keywords—greedy; lokasi ; Algoritma

I. PENDAHULUAN

Lokasi adalah tempat, laman dimana suatu benda, orang daerah berada. Lokasi dalam geografis dinyatakan dengan sebuah titik atau koordinat yang biasanya terdiri dari garis lintang dan garis bujur. Lokasi biasanya membahas letak suatu objek di muka bumi dengan memperhatikan jarak dan waktu. Lokasi biasanya bisa dilihat dengan menggunakan peta, satelit, Kompas dan lainya.

Algoritma merupakan suatu metode yang bertujuan untuk memcahkan suatu masalah dengan proses penyelesaiaan masalah dilakukan dengan cara penyusunan secara logis dan sistemastis sehingga bisa mendapatkan solusi yang efisien. Terdapat berbagai motode algoritma yang bisa digunakan dalam proses pemecahan masalah, salah satunya adalah algoritma greedy.

Algoritma greedy merupakan metode yang paling populer untuk memecahkan persoalan optimasi. Greedy sendiri diambil dari bahasa inggris yang artinya rakus, tamak atau serakah. Prinsip algoritma greedy adalah: "take what you can get now!"

Metode greedy yang digunakan pada pemecahan masalah lokasi ini didasari kepada banyaknya solusi dan kemungkinan yang bisa didapat dengan menggunakan algoritma greedy dan aka nada satu solusi paling efisien dan paling optimal, sehingga akan menghasilkan solusi paling cepat dan efisien untuk menentukan lokasi suatu daerah.

II. TEORI DASAR

A. Algoritma Greedy

Algoritma yang memecahkan persoalan
secara langkah per langkah (step by step) sedemikian sehingga, pada setiap langkah:
1. mengambil pilihan yang terbaik yang dapat diperoleh pada saat
itu tanpa memperhatikan konsekuensi ke depan
(prinsip "take what you can get now!")
2. dan "berharap" bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global.

Algoritma Bruce Force adalah sebuah pendekatan yang langsung (straightforward) untuk memecahkan sebuah topik masalah, Algoritma ini juga dapat memecahkan masalah dengan sangat sederhana dan langsung secara jelas dan mudah dipahami.

Kelebihan Algoritma Greedy

- Respon yang cepat dalam proses pencarian solusi.
- Membutuhkan waktu yang singkat dan solusi yang optimal
- Dengan algoritma greedy, meskipun tur dengan berbobot minimal tidak dapat ditemukan, namun solusi dengan algoritma greedy dianggap sebagai hampiran solusi optimal.

Kekurangan Algoritma Bruce Force

- Solusi yang sangat banyak sehingga memungkinkan solusi tersebut tidak bernilai positif.

Elemen-elemen algoritma greedy:

1. Himpunan kandidat, C : berisi kandidat yang akan dipilih pada setiap Langkah
(misal: simpul/sisi di dalam graf, job, task, koin, benda, karakter, dsb)
2. Himpunan solusi, S : berisi kandidat yang sudah dipilih
3. Fungsi solusi: menentukan apakah himpunan kandidat yang dipilih sudah
memberikan solusi
4. Fungsi seleksi (selection function): memilih kandidat berdasarkan strategi greedy
tertentu. Strategi greedy ini bersifat heuristik.
5. Fungsi kelayakan (feasible): memeriksa apakah kandidat yang dipilih dapat
dimasukkan ke dalam himpunan solusi (layak atau tidak)
6. Fungsi obyektif : memaksimumkan atau meminimumkan

B. Penentuan lokasi/ rute efisien

Gambar 2.1 contoh gambar password

III. ANALISIS DAN PEMBAHASAN

Pada Tugas kali ini yaitu mata kuliah strategi algoritma saya mengambil topik permasalahan rute yang dan lokasi tempat yang kurang efisien dengan menggunakan konsep algoritma greedy. Percobaan untuk mengetahui sebuah daerah dengan greedy bisa dibilang akurat karena akan mencoba setiap kemungkinan yang ada secara terurut. Selain itu pada algoritma greedy juga akan lebih cepat dalam menemukan solusi sengingga waktu yang digunakan lebih efisien.

Pada tugas ini juga saya akan melakukan beberapa percobaan dengan memasukkan beberapa posisi kota yang berbeda hingga program akan menentukan rute dan letak paling cepat dan efisien.

Kemudian pada tugas ini saya juga menggunakan sebuah pemograman yaitu pemograman python dimana pemograman ini menurut saya adalah pemograman yang cukup mudah dipelajari oleh pemula. Python juga dikenal sebagai Bahasa pemrograman yang manusiawi dimana kebanyakan masih menggunakan Bahasa internasional yakni Bahasa

A. Greedy

Algoritma greedy merupakan metode yang paling populer untuk memecahkan persoalan optimasi. Greedy sendiri diambil dari bahasa inggris yang artinya rakus, tamak atau serakah. Prinsip algoritma greedy adalah: “take what you can get now!”

B. Python

Pemograman python merupakan Bahasa pemogramanyang mudah di mengerti. Python bisa melakukan eksekusisejumlah perintah secara langsung dengan metode OOP tetapi, python juga merupakan Bahasa pemograman yang memiliki level tinggi tapi pemograman ini dirancang sedemikian agar mudah dipahami dan dipelajari oleh pemula. python juga menampilkan fitur-fitur yang menarik dan mudah untuk dipelajari seperti tata Bahasa yang mudah sehingga mudah dipelajari. Python juga mempunyai sistem memori otomatis dan pengelola data.(KOMINFO, 2019)

IV. PENJELASAN PROGRAM

Penjelasan Source Code:

a) Program akan meminta inputan berupa nama file yang mana datanya berupa string tanpa ekstensi file.
b) Proses pemamnggilan inputan dari file sesuai dengan inputan pengguna dengan menggunakan *numpy array*.
c) Selanjutnya proses penyimpanan waktu mulai eksekusi program setelah file tersimpan kedalam variable *data* yang mana dalam hal ini menggunakan modul *time()* .
d) Berikutnya pembuatan *keys* yang yang diisi dengan list alphabet dari A – *len(A)* yang akan dijadikan sebagai variable kota.
e) Parsing data dari *numpy array* ke *list*
f) Berikutnya parsing dari *data* ke bentuk *dictionary* dengan *keys* yang telah dibuat, dan *values* di *round* menjadi 2 digit di belakang koma, dengan tujuan menghindari *bug* saat *parsing data* jika tipe *data* adalah *float*.
g) Berikutnya mencari jarak *maksimal* yang dapat *tercover* oleh pemadam kebakaran (fungi buatan yaitu *find_average())*
h) Pemanggilan Fungsi *find_average()* untuk melakukan penjumlahan jarak antar kota dari matriks segitiga bagian atas. Hal ini terjadi karena jarak matriks segitiga atas dan bawah bernilai sama jadi cukup menggunakan salah satu segitiga.
i) Proses penjumlahan dan dilanjutkan pembagian dengan banyaknya data yang sudah dijumlahkan (n), hingga didapatkan rata-rata jarak maksimal dengan selanjutnya disimpan kedalam variable *max_length*
j) Menggunakan konsep algoritma greedy, untuk membuat fungsi *greedy_scp()*yang melakukan pemetaan daerah-daerah yang tercover ketika pemadam kebakaran ditempatkan di suatu kota,
k) *max_length* melakukan perbandingan jarak dan dilanjutkan dengan pemilihan dengan kota lain lalu disimpan kedalam dictionary dengan nama *dict_cover_kota*
l) Simpan total maksimal kota yang tercover kedalam *max_covered*, dan hasil yang didapat menjadi acuan dalam mencari kota dengan coveran maksimum
m) Simpan jarak total antar kota yang tercover oleh kota tersebut kedalam *save_jarak*.

Terakhir cari kota yang jarak antar kota nya paling dekat dengan acuan nilai minimum dari *save_jarak* yang disimpan kedalam *best_result*, kota dengan jarak yang sama (*best_result)* menjadi kota yang paling cocok untuk penyelesaian solusi *set covered problem.*

V . KODE PROGRAM

```
import numpy as np
from string import ascii_uppercase
import time
nama_file = input("Masukkan nama file tanpa : ")
#pemanggilan nama file dan harus ada dalam satu folder (input) jika tidak maka tidak dapat memanggil
      input
folder = "input/"
data = np.loadtxt(folder + nama_file + ".txt", dtype="f", delimiter=" ")
waktu_mulai = time.time()
keys = list(ascii_uppercase)
keys = keys[:len(data)]
#parsing array ke list
data=data.tolist()
#parsing list ke dictionary
dict_data = {keys[j]:{keys[i]:round(data[j][i],2) for i in range(len(data))}  for j in range(len(data))}
#mencari jarak tempuh rata-rata
def find_average(data):
    total = 0
    n = 0
    # ambil matriks segitiga atas untuk mencari jarak tempuh rata2
    for kolom in data :
        for baris in range(data.index(kolom),len(kolom)):
            total+=kolom[baris]
            n+=1
    rata2 = total/n
    return round(rata2,2)
#pencarian lokasi yang cocok untuk membangun gedung damkar menggunakan algoritma greedy
def greedy_scp(max_length):
    # buat dictionary 2dimensi dengan keys alphabet dan value empty dictionary
dict_cover_kota : dict = {i:{} for i in keys}
    # lgoritma greedy. mencari lokasi yang tercover jika penempatan gedung di key1
    for key1 in keys:
        print(f"Lokasi {keys.index(key1)+1}: Kota {key1}")
        print("Lokasi yang di cover: ",end="")
        for key2 in keys:
            if dict_data[key1][key2] <= max_length :
                # simpan data area yang tecover rata2
                dict_cover_kota[key1].update({ key2:dict_data[key1][key2]})
        print(list(dict_cover_kota[key1].keys()))
        print()
    #mencari lokasi yang dapat mengcover area terbanyak dengan algoritma greedy
    max_covered = (max(dict_cover_kota.values(), key = len))
    list_max_covered : dict = {}
    save_jarak : dict = {}
    for key in keys :
        if len(dict_cover_kota[key]) == len(max_covered) :
            list_max_covered[key] = {}
            list_max_covered[key]=dict_cover_kota[key]
            save_jarak[key] = round(sum(list_max_covered[key].values()),2)
    #menentukan jalan terpendek
    best_result = min(save_jarak.values())
    for kota, jarak in save_jarak.items():
        if jarak == best_result:
            best_result = kota
    print()
    print("Kota terbanyak mengcover area:",list(list_max_covered.keys()))
    print()
    print("Kota terbagus untuk membangun gedung damkar:",best_result)
def main():
    max_length = find_average(data)
    print("Area Covered by Damkar :",max_length)
    greedy_scp(max_length)
    print()
    waktu_akhir = time.time() - waktu_mulai
    print("waktu eksekusi:",waktu_akhir)
print()
    print("Spek laptop : AMD Ryzen 3, RAM 8GB, SSD 512GB")
    print("Jesika Putri_13520050_RA")
```

V. UJI COBA PROGRAM DAN HASIL

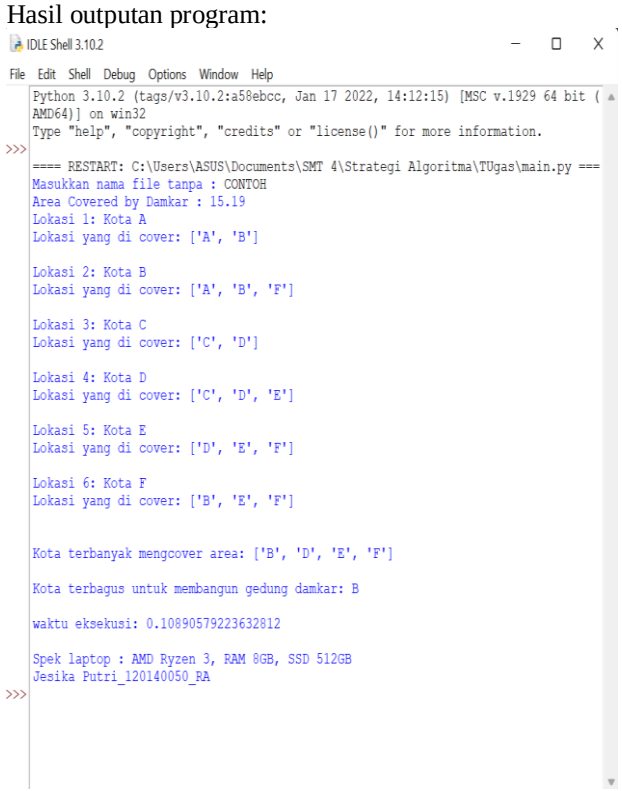
Saya akan menguji program yang saya buat beberapa kali dengan daerah-daerah yang berbeda sehingga dari program akan mengeluarkan lokasi terdekat agar bisa mengakses derah yang dituju.

1. Uji Coba Pertama

pada percobaan kita akan menginputkan berbagai lokasi yang mana program akan mengeluarkan luaran berupa lokasi terefisien, seperti pada gambar berikut:

0	10	20	30	30	20
10	0	25	35	20	10
20	25	0	15	30	20
30	35	15	0	15	25
30	20	30	15	0	14
20	10	20	25	14	0

Hasil outputan program:



Data percobaan ke 2:

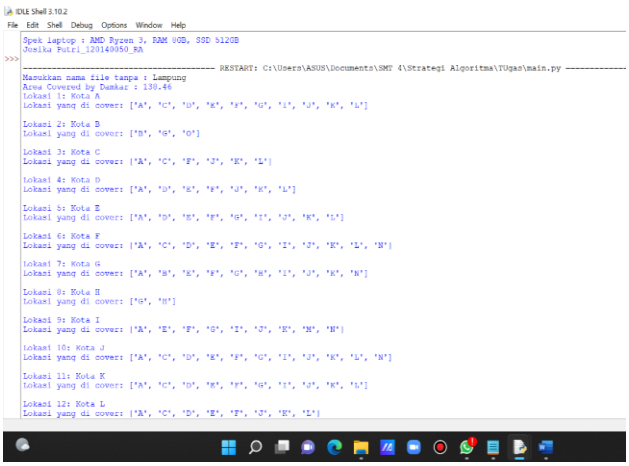
0	1645	3818	2852	2988	5194
1645	0	3540	3442	4005	5274
3818	3540	0	1833	2914	1913
2852	3442	1833	0	1047	2486
2988	4005	2914	1047	0	3138
5194	5274	1913	2486	3138	0

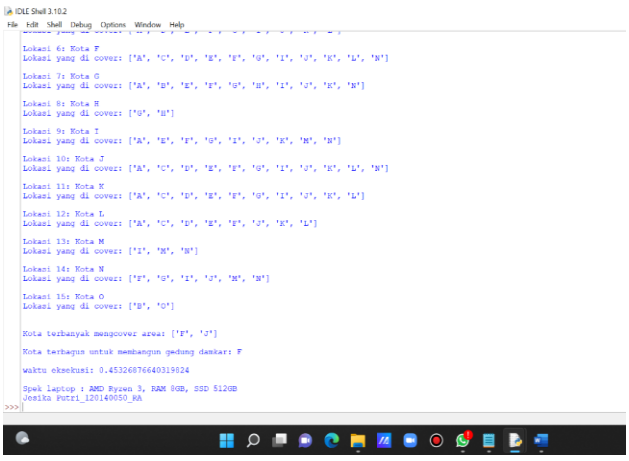
Hasil outputan:

Data percobaan ke 3:

0.00	241.42	80.78	59.22	80.24	57.85	115.19	209.32	119.36	51.81	25.98	41.40	204.34	139.60	231.00
241.42	0.00	183.66	296.98	256.23	183.57	133.08	139.73	221.60	227.80	238.46	223.04	306.59	241.85	28.56
80.78	183.66	0.00	140.00	151.89	129.50	186.84	268.83	191.00	123.46	54.80	39.38	275.99	211.25	150.82
59.22	296.98	140.00	0.00	135.80	113.41	169.41	264.88	174.92	107.38	85.20	100.62	259.90	195.16	325.54
80.24	256.23	151.89	135.80	0.00	84.40	125.78	224.13	134.17	29.50	97.09	112.51	219.15	154.41	284.79
57.85	183.57	129.50	113.41	84.40	0.00	53.12	151.47	61.51	48.69	74.70	90.12	146.49	81.75	212.13
115.19	133.08	186.84	169.41	125.78	53.12	0.00	100.98	91.16	101.81	132.04	147.46	176.14	111.40	161.64
209.32	139.73	268.88	264.88	224.13	151.47	100.98	0.00	189.51	200.16	214.03	229.45	274.49	209.75	168.29
119.36	221.60	174.92	174.92	134.17	61.51	91.16	189.51	0.00	110.20	136.20	151.62	86.68	21.94	250.16
51.81	227.80	107.38	107.38	29.50	48.69	101.81	200.16	110.20	0.00	68.66	84.08	195.19	130.45	256.36
25.98	238.46	54.80	85.20	97.09	74.70	132.04	214.03	136.20	68.66	0.00	15.42	221.19	156.45	267.02
41.40	223.04	39.38	100.62	112.51	90.12	147.46	229.45	151.62	84.08	15.42	0.00	236.61	171.87	251.60
204.34	306.59	275.99	259.90	219.15	146.49	176.14	274.49	86.68	195.19	221.19	236.61	0.00	108.62	335.15
139.60	241.85	211.25	195.16	154.41	81.75	111.40	209.75	21.94	130.45	156.45	171.87	108.62	0.00	270.41
231.00	28.56	150.82	325.54	284.79	212.13	161.64	168.29	250.16	256.36	267.02	251.60	335.15	270.41	0.00

Hasil outputan program:





KESIMPULAN

Jadi dengan menggunakan algoritma greedy kita dapat membuat sebuah program untuk menentukan letak daerah yang paling mudah diakses sehingga efisien dalam penggunaan waktu. Karena algoritma greedy adalah sebuah pendekatanyang langsung (straightforward) untuk memecahkan sebuah topik masalah, Algoritma ini juga dapat memecahkan masalah dengan sangat sederhana dan langsung secara jelas dan mudah dipahami dan mengecek mulai dari awal hingga akhir.

UCAPAN TERIMAKASIH

Begitulah hasil dari implementasi tugas kali ini dimana saya membuat dengan metode algoritma brute force untuk mencari password. Semoga dengan penjelasan yang saya buat mudah dimengerti oleh pembaca atau peneliti makalah ini semoga bermanfaat jika ada kesalahan pada penulisan atau kurang nya dalam penjelesan saya minta maaf, sekian terimakasih.

REFERENCES

[1] KOMINFO, B. (2019, september 19). *baktikominfo.id*. Retrieved from https://www.baktikominfo.id/en/informasi/pengetahuan/bahasa_pemrograman_python_pengertian_sejarah_kelebihan_dan_kekurangannya-954

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag1.pdf Di akses 28 maret 2022

[3] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Greedy-(2018).pdf

[4] Link video : https://drive.google.com/drive/folders/14IJIUEl2s0Xjsv8I2NCEJaEunzwHS8QM?usp=sharing

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Lampung, 29 Maret 2022

Jesika Putri
120140050