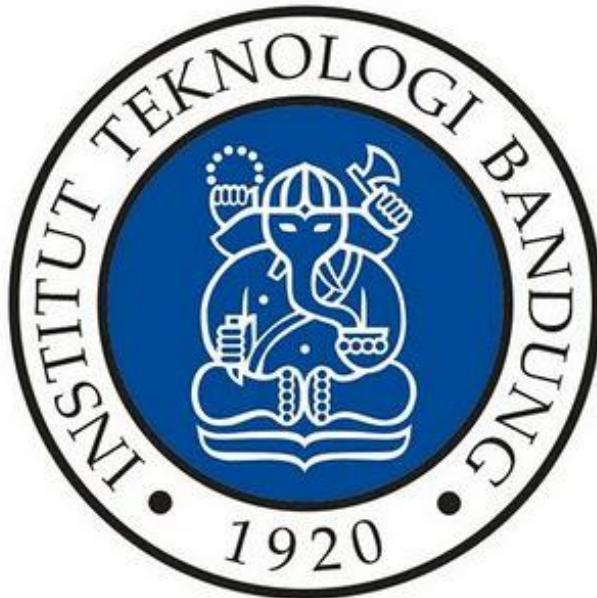


Laporan Tugas Kecil 3 IF2211 Strategi Algoritma
Semester II tahun 2022/2023

Pencarian Pasangan Titik Terdekat di Bidang 3D
dengan Algoritma Divide and Conquer



Disusun oleh:

Rachel Gabriela Chen **13521044**

Hobert Anthony Jonatan **13521079**

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2023

DAFTAR ISI

BAB I : DESKRIPSI MASALAH.....	2
BAB II : ALGORITMA UCS DAN A*	4
2.1 Algoritma Uniform Cost Search (UCS).....	4
2.2 Algoritma A*.....	4
BAB III : IMPLEMENTASI PROGRAM	6
BAB IV : EKSPERIMEN.....	12
BAB V : KESIMPULAN DAN KOMENTAR.....	48
5.1 Kesimpulan.....	48
5.2 Komentar Terkait Tugas.....	48
LAMPIRAN.....	49

BAB I

DESKRIPSI MASALAH

Algoritma UCS (Uniform cost search) dan A* (atau A star) dapat digunakan untuk menentukan lintasan terpendek dari suatu titik ke titik lain. Hal ini dapat dimanfaatkan untuk mencari lintasan terpendek antara dua titik lokasi di jalan-jalan Kota Bandung. Jalanan di kota Bandung dapat direpresentasikan sebagai graf berbobot dimana persimpangan jalan menjadi simpul. Persimpangan jalan yang saling berhubungan kemudian digambarkan sebagai sisi graf dengan jaraknya sebagai bobot dalam km). Permasalahan ini diselesaikan dengan membuat sebuah program sederhana yang berisi file .txt yang memuat titik-titik persimpangan di jalan-jalan di Kota Bandung.

Program dapat menerima file .txt dengan format sebagai berikut ini:

1. Baris pertama berisi n , yaitu jumlah lokasi (simpul).
2. Baris ke- $(2i)$ berisi nama lokasi (simpul) dan baris ke- $(2i+1)$ berisi koordinat (garis lintang, garis bujur) yang dipisahkan dengan koma (tanpa spasi di antaranya). Dengan $0 < i \leq n$.
3. n baris selanjutnya berupa matriks ketetanggan berbobot yang menunjukkan simpul-simpul yang saling bertetangga.

Pengguna kemudian dapat memilih algoritma yang akan digunakan untuk menyelesaikan persoalan.

Contoh input :

```
9
Metro Indah Mall
-6.942120717034387,107.65874719557186
Jl Soekarno Hatta
-6.940290411439278,107.65824797384171
Jembatan Sungai Cicadas
-6.942040661767213,107.65277526794446
Yamaha Service Center
-6.943154221380146,107.65990485388869
```

Jl Jupiter Barat 26

-6.943648609245794,107.65761911899006

Taman S2

-6.945294809444772,107.65724156585514

Taman Jupiter

-6.9440854448675,107.66042481163986

Jl Jupiter Barat Utama

-6.944109165356709,107.66130417588958

Jl Raya Cirebon Bandung

-6.939199773655335,107.66391628016386

0 0.210848599761068 0 0.1718572494631123 0.21063785177058295 0 0 0

0.6565320975146275

0.210848599761068 0 0.6346539115360498 0.36722212954705163 0 0 0 0 0

0 0.6346539115360498 0 0 0 0 0 0

0.1718572494631123 0.36722212954705163 0 0 0.2582177666444844 0 0.11838902414100805

0 0

0.21063785177058295 0 0 0.2582177666444844 0 0.18773305374822957 0 0 0

0 0 0 0 0.18773305374822957 0 0 0 0

0 0 0 0 0.11838902414100805 0 0 0 0.09709941054551648 0

0 0 0 0 0 0.09709941054551648 0 0.6173627264659105

0.6565320975146275 0 0 0 0 0 0.6173627264659105 0

BAB II

ALGORITMA UCS DAN A*

UNTUK PENCARIAN LINTAS TERDEKAT

2.1 Algoritma Uniform Cost Search (UCS)

Algoritma Uniform Cost Search (UCS) adalah algoritma pencarian lintasan terpendek di antara dua simpul pada sebuah graf atau peta dengan menghitung biaya(*cost*) terkecil untuk mencapai simpul tujuan dari simpul awal. Dalam masalah yang telah dijelaskan pada Bab I, langkah-langkah algoritma UCS adalah sebagai berikut:

1. Tentukan simpul awal dan simpul tujuan.
2. Buat sebuah *priority queue* yang akan digunakan untuk menyimpan simpul-simpul yang akan dikunjungi beserta dengan lintasan yang digunakan untuk mencapai simpul tersebut dan biayanya.
3. Selama *priority queue* tidak kosong, kunjungi simpul yang memiliki lintasan dengan biaya terendah.
4. Periksa apakah simpul yang sedang dikunjungi adalah simpul tujuan atau bukan. Jika ya, kembalikan lintasan tersebut.
5. Jika simpul yang sedang dikunjungi (v) bukan simpul tujuan dan belum pernah dikunjungi sebelumnya, tambahkan tetangga-tetangganya ke dalam *queue* yang telah dibuat. Lintasan dari masing-masing tetangga adalah lintasan yang ditempuh untuk mencapai simpul v ditambahkan dengan dirinya sendiri. *Cost* dihitung dari *cost* untuk mencapai simpul v ditambah dengan *cost* dari v ke tetangga tersebut.
6. Ulangi langkah 3-5 hingga *priority queue* kosong atau ditemukan solusi. Jika *priority queue* kosong, artinya tidak ada lintasan dari simpul awal ke simpul tujuan.

2.2 Algoritma A*

Algoritma A* adalah algoritma pencarian jalur atau *pathfinding* yang digunakan untuk menemukan jalur terpendek antara dua simpul atau titik pada sebuah graf atau peta. Dalam masalah yang telah dijelaskan pada Bab I, langkah-langkah algoritma A* adalah sebagai berikut:

1. Tentukan simpul asal dan simpul tujuan.

2. Cari terlebih dahulu nilai heuristik tiap simpul untuk algoritma A*, yang dalam persoalan ini berupa jarak tegak lurus dari simpul ke simpul tujuan, simpan jarak tersebut sebagai atribut dalam simpul.
3. Buat sebuah *priority queue* yang akan digunakan untuk menyimpan simpul-simpul yang akan dikunjungi beserta lintasan yang dilalui untuk mencapai simpul tersebut dan biayanya (dalam hal ini biaya adalah jarak), serta prediksi biaya untuk mencapai simpul tujuan dari simpul saat ini. *Priority queue* akan terurut menaik berdasarkan prediksi biaya yang diperlukan untuk mencapai simpul tujuan dari simpul saat ini, prediksi biaya diperoleh dari biaya sebenarnya yang telah terakumulasi hingga mencapai simpul saat ini ditambah dengan nilai heuristik simpul tersebut.
4. Tambahkan simpul asal ke dalam *priority queue*.
5. Selama *priority queue* tidak kosong, kunjungi simpul pertama pada *priority queue* untuk diperiksa.
6. Periksa apakah simpul yang sedang dikunjungi adalah simpul tujuan atau bukan. Jika, ya kembalikan biaya tempuh sebenarnya dari simpul tersebut sebagai jawaban.
7. Jika simpul yang sedang dikunjungi (v) bukan simpul tujuan, periksa apakah tetangganya sudah pernah dikunjungi atau dilewati dalam lintasan untuk mencapai v , jika belum maka tambahkan tetangganya ke dalam *priority queue* yang telah dibuat. Lintasan dari masing-masing tetangga adalah lintasan yang ditempuh untuk mencapai simpul v ditambahkan dengan dirinya sendiri (tetangga itu sendiri). *Cost* tetangga adalah *cost* untuk mencapai simpul v ditambah dengan *cost* dari v ke tetangga tersebut. Prediksi biaya dihitung dari *cost* dirinya sendiri ditambah dengan nilai heuristik dirinya sendiri.
8. Ulangi langkah 5-7 hingga *priority queue* kosong atau ditemukan solusi. Jika *priority queue* kosong, artinya tidak ada lintasan dari simpul awal ke simpul tujuan.

BAB III

IMPLEMENTASI PROGRAM

DENGAN BAHASA PYTHON

Algoritma pada Bab II diimplementasikan dengan menggunakan bahasa Python. *Source code* dari program dapat diakses pada https://github.com/chaerla/Tucil3_13521044_13521079.

Source code diimplementasikan secara modular dengan beberapa modul yang terdapat pada lib dan juga main.py yang berperan sebagai program utama.

1. node.py

File ini berisi *class* Node yang merepresentasikan sebuah lokasi pada graf (peta). Atribut pada *class* ini antara lain:

- name : Nama dari simpul (lokasi)
- latitude: Garis lintang dari lokasi
- longitude: Garis bujur dari lokasi
- index: Index dari simpul (berperan sebagai id untuk memudahkan pemrosesan graf)
- heuristic: Jarak tegak lurus dari sebuah simpul ke node tujuan (digunakan pada algoritma A*)

Class ini juga dilengkapi dengan *getter* dan *setter* yang dibutuhkan.

2. graph.py

File ini berisi *class* Node yang merepresentasikan sebuah lokasi pada graf (peta). Atribut pada *class* ini antara lain:

- nodes: List dari Node yang terdapat pada graf
- adjmatrix: Matriks ketetanggaan yang menunjukkan hubungan antar simpul. adjmatrix[i][j] bernilai -1 jika tidak ada sisi yang menghubungkan simpul ke-i dengan simpul ke-j dan bernilai jarak Haversine antar simpul ke-i dengan simpul ke-j apabila ada sisi yang menghubungkan keduanya.

Class ini juga dilengkapi dengan *getter* dan *setter* yang dibutuhkan.

3. util.py

File ini berisi fungsi-fungsi yang dibutuhkan untuk menunjang program utama:

- calc_haversine_dist(pos1, pos2): menghitung jarak Haversine antara dua lokasi

- `read_graph_from_file(file_name)`: membaca dan memuat graf dari file .txt dimana `file_name` merupakan nama file pada folder test.

4. ucs.py

File ini berisi fungsi `ucs` yang merupakan implementasi dari algoritma UCS untuk mencari lintasan terpendek.

```

● ● ●
1  from lib.graph import *
2  from lib.node import *
3
4  def ucs(start: Node, goal: Node, graph: Graph):
5      """
6          A function to find the shortest path between start and goal
7          returns a dictionary:
8          {
9              "cost" : {the cost of the path}
10             "path" : {the shortest path between the nodes}
11             "success" : {whether or not there's a path connecting the start and goal}
12         }
13     """
14
15     queue = [] #priority queue that stores the nodes to be visited
16
17     # the queue element has node(the node to be visited), cost(the cost to visit the node), path(the path to reach the node)
18     queue.append({
19         "node": start,
20         "cost": 0,
21         "path": [start]
22     })
23
24     # initialize the answer to the starting node
25     answer = {
26         "cost": start,
27         "path": 0,
28         "success": False,
29     }
30
31     # list to store the index of the visited nodes
32     visited = []
33     while (len(queue) > 0):
34         queue = sorted(queue, key=lambda x: x["cost"])
35
36         # the current node to check
37         curr = {
38             "node": queue[0]["node"],
39             "cost": queue[0]["cost"],
40             "path": queue[0]["path"]
41         }
42
43         # checks if it hasn't been visited yet
44         if (curr["node"].index() not in visited):
45             # if the current node is the goal, return
46             if (curr["node"].index() == goal.index()):
47                 answer = {
48                     "cost": curr["cost"],
49                     "path": curr["path"],
50                     "success": True,
51                 }
52                 return answer
53             # expands the node, push all its neighbours to the priority queue
54             for i in range (len(graph.adjmatrix()[curr["node"].index()])):
55                 if(graph.adjmatrix()[curr["node"].index()][i] != -1):
56                     temp = {
57                         "node": graph.nodes()[i],
58                         "cost": curr["cost"] + graph.adjmatrix()[curr["node"].index()][i],
59                         "path": curr["path"] + [graph.nodes()[i]]
60                     }
61                     queue.append(temp)
62             queue = sorted(queue, key=lambda x: x["cost"])
63
64             # add the node to the visited list
65             visited.append(curr["node"].index())
66
67             # pop the node from the queue
68             del queue[0]
69
70     return answer

```

Pada implementasi di atas, fungsi menerima parameter *start* yaitu simpul awal, *goal* yaitu simpul tujuan, dan *graph*. Digunakan list sebagai *priority queue* yang selalu di-*sort* berdasarkan *cost* pada setiap pemrosesan elemen *queue*. Elemen *queue* berupa *dictionary* yang menyimpan informasi *node* yang akan dikunjungi, *cost* untuk mengunjungi *node* tersebut melalui lintasan *path*. *Return value* dari fungsi tersebut juga berupa *dictionary* yang terdiri atas *cost* minimum dengan lintasan *path* dan juga boolean *success* yang bernilai True jika berhasil ditemukan lintasan dari *start* dan *goal*.

5. astar.py

File ini berisi fungsi *astar* yang merupakan implementasi dari algoritma A* untuk mencari lintasan terpendek.

```

● ● ●
1  from lib.graph import *
2  from lib.node import *
3  from lib.util import *
4
5  def astar(start: Node, goal: Node, graph: Graph):
6      """
7          A function to find the shortest path between start and goal using A* algorithm
8          returns a dictionary:
9      {
10          "cost"      : {the cost of the path}
11          "path"      : {the shortest path between the nodes}
12          "success"   : {whether or not there's a path connecting the start and goal}
13      }
14  """
15  # add heuristic distance for every node in graph
16  for i in range(len(graph.nodes())):
17      node = graph.nodes()[i]
18      heuristic = calc_haversine_dist(node, goal)
19      graph.nodes()[i].set_heuristic(heuristic)
20
21  queue = [] #priority queue that stores the nodes to be visited
22
23  """
24  queue element is a dictionary consist of:
25  {
26      "node"        : {the node to be visited}
27      "cost_so_far" : {the cost to visit the node}
28      "path"        : {the path to reach the node}
29      "prediction_cost" : {cost prediction from the start node to the goal node with heuristics}
30  }
31  prediction_cost = cost_so_far + heuristic
32  """
33  first_entry = {
34      "node"        : start,
35      "cost_so_far" : 0,
36      "path"        : [start],
37      "prediction_cost" : start.heuristic()
38  }
39  queue.append(first_entry)
40
41  # initialize the answer to the starting node
42  answer = {
43      "cost"      : 0,
44      "path"      : 0,
45      "success"   : False
46  }
47
48  while(len(queue) > 0):
49      queue = sorted(queue, key=lambda x: x["prediction_cost"])
50
51      # the current node to check
52      curr = {
53          "node"        : queue[0]["node"],
54          "cost_so_far" : queue[0]["cost_so_far"],
55          "path"        : queue[0]["path"],
56          "prediction_cost" : queue[0]["prediction_cost"]
57      }
58
59      # check if current node is goal, the first found is guaranteed to be the minimum
60      if(curr["node"].index() == goal.index()):
61          answer = {
62              "cost": curr["cost_so_far"],
63              "path": curr["path"],
64              "success": True
65          }
66          return answer
67
68      # expands the node, push all its neighbours to the priority queue
69      for i in range(len(graph.adjmatrix())[curr["node"].index()]):
70          if(graph.adjmatrix()[curr["node"].index()][i] != -1):
71
72              # if node that want to be visited already in curr path, don't add it to queue (to limit the search)
73              if(graph.nodes()[i] not in curr["path"]):
74                  cost_so_far = curr["cost_so_far"] + graph.adjmatrix()[curr["node"].index()][i]
75                  temp = {
76                      "node"        : graph.nodes()[i],
77                      "path"        : curr["path"] + [graph.nodes()[i]],
78                      "cost_so_far" : cost_so_far,
79                      "prediction_cost" : cost_so_far + graph.nodes()[i].heuristic()
80                  }
81                  queue.append(temp)
82
83          # remove the first element from the queue
84          del queue[0]
85
86      # if reach this point, then the answer must be unsuccess
87      return answer

```

Implementasi algoritma A* serupa dengan implementasi pada algoritma UCS, perbedaannya hanya terletak pada elemen queue yang ditambahkan satu komponen lagi yaitu *prediction_cost* yang merupakan prediksi *cost* (jarak) dari simpul ke simpul tujuan, *prediction_cost* dihitung dari *cost* untuk mencapai simpul (*cost_so_far*) ditambah dengan nilai heuristik dari simpul itu sendiri, nilai heuristik dari simpul merupakan jarak tegak lurus antara simpul dengan simpul tujuan. *Priority queue* pada algoritma ini diurutkan berdasarkan *prediction_cost* dari tiap elemen. *Return value* dari fungsi ini sama dengan *return value* pada algoritma UCS.

6. app.py

File ini berisi main program jika pengguna ingin menunjukkan hasil visualisasi map. Dibedakan dari main.py karena app.py tidak menangani kasus perulangan dengan *test case* berbeda. Kami memanfaatkan Flask dan Folium untuk menampilkan peta.



```

1  from main import *
2  from flask import Flask
3  import folium
4  from folium.plugins import MarkerCluster
5  from folium.vector_layers import PolyLine
6
7  app = Flask(__name__)
8  result = None
9
10 @app.route('/')
11 def index():
12     graph = result["graph"]
13     path = result["path"]
14
15     m = folium.Map(location=[graph.nodes()[0].latitude(), graph.nodes()[0].longitude()], zoom_start=16)
16
17     # Add markers to the map
18     marker_cluster = MarkerCluster().add_to(m)
19     for node in graph.nodes():
20         folium.Marker(location=[node.latitude(), node.longitude()], popup=str(node.index() + 1)).add_to(marker_cluster)
21
22     # Add polylines to the map based on the adjacency matrix
23     for i in range(len(graph.admatrix())):
24         for j in range(i+1, len(graph.admatrix())):
25             if graph.admatrix()[i][j] != 0:
26                 line = PolyLine(locations=[[graph.nodes()[i].latitude(), graph.nodes()[i].longitude()],
27                                             (graph.nodes()[j].latitude(), graph.nodes()[j].longitude())], color='red', weight=5, opacity=0.7).add_to(m)
27
28     # Add polylines to the map based on the result path
29     for i in range(len(path)-1):
30         line = PolyLine(locations=[[path[i].latitude(), path[i].longitude()],
31                                 (path[i+1].latitude(), path[i+1].longitude())], color='green', weight=5, opacity=0.7).add_to(m)
32
33     # Return the map
34     return m._repr_html_()
35
36 if __name__ == '__main__':
37     print(welcome_screen())
38     graph = create_graph_from_input_file()
39     start_index = ask_start_node(graph)
40     goal_index = ask_goal_node(graph)
41
42     choice = ask_algorithm()
43     if choice == 1:
44         ans = ucs(graph.nodes()[start_index], graph.nodes()[goal_index], graph)
45     else:
46        ans = astar(graph.nodes()[start_index], graph.nodes()[goal_index], graph)
47
48     print()
49     print(answer(ans))
50     print("Silahkan visualisasi graf dapat dilihat pada localhost:5000")
51     ans["graph"] = graph
52     result = ans
53
54 app.run(debug=False)

```

7. IO.py

File ini memuat fungsi dan prosedur yang mendukung proses meminta input dan proses menampilkan output pada program utama.

8. main.py

File ini memuat program utama yang memanfaatkan fungsi-fungsi pada modul-modul di atas.

BAB IV

EKSPERIMEN

File *test* yang digunakan untuk menguji program pada bab ini ada dalam *folder test* pada repository kami. Untuk setiap *input file* akan diujikan dua algoritma (UCS dan A*) untuk titik awal dan titik tujuan yang sama. Pada hasil visualisasi jalur terpendek akan direpresentasikan dengan lintasan berwarna hijau dan keterhubungan antar simpul akan direpresentasikan dengan garis berwarna merah antar simpulnya, simpul asal akan berwarna biru, sedangkan simpul tujuan akan berwarna merah, serta simpul lainnya dalam peta akan berwarna orange. Berikut adalah hasil pengujian program:

1. Uji coba dengan peta jalan sekitar kampus ITB

file input : “Map ITB2.txt”

```
10
Plaza Widya Nusantara
-6.890895299442325,107.61034662460492
Jl. I ITB
-6.891044787183361,107.6110413429995
Kelas Kuliah 9009
-6.892005264397014,107.61104403541327
Laboratorium Pengujian Doping
-6.890855470915589,107.61188707117256
Jl. A ITB
-6.891948643023322,107.61039257954421
Fakultas Seni Rupa dan Desain
-6.8927194214051415,107.61067366146906
Aula Timur ITB
-6.892499983616666,107.61104355690973
Galeri Soemardja
-6.8923527458026275,107.61176284906381
Ikatan Mahasiswa Geodesi ITB
-6.891333180853256,107.61234263141938
Laboratorium Penelitian Konversi Energi Elektrik
```

```
-6.8908975578749585,107.6090223083595
0 0.07847184254150753 0 0 0 0 0 0 0.14619373859012696
0.07847184254150753 0 0.10680060688911823 0.0957052176332474 0 0 0 0 0
0 0.10680060688911823 0 0 0.07219023217274877 0 0 0 0 0
0 0.0957052176332474 0 0 0 0 0 0.07314853997584349 0
0 0 0.07219023217274877 0 0 0.09115058443575602 0 0 0 0
0 0 0 0 0.09115058443575602 0 0.047568176625711986 0 0 0
0 0 0 0 0 0.047568176625711986 0 0.0810739276461696 0 0
0 0 0 0 0 0.0810739276461696 0 0.1301892739854349 0
0 0 0 0.07314853997584349 0 0 0 0.1301892739854349 0 0
0.14619373859012696 0 0 0 0 0 0 0 0
```

Pengujian pertama :

Mencari lintasan terpendek dari simpul ke-1 (Plaza Widya Nusantara) ke simpul ke-8 (Galeri Soemardja)

```
Masukkan nama file (dalam folder test): Map ITB2.txt
Map ITB2.txt
Graph berhasil dibaca dari file input.
[1]Plaza Widya Nusantara (107.61034662460492, -6.890895299442325)
[2]Jl. I ITB (107.6110413429995, -6.891044787183361)
[3]Kelas Kuliah 9009 (107.61104403541327, -6.892005264397014)
[4]Laboratorium Pengujian Doping (107.61188707117256, -6.890855470915589)
[5]Jl. A ITB (107.61039257954421, -6.891948643023322)
[6]Fakultas Seni Rupa dan Desain (107.61067366146906, -6.8927194214051415)
[7]Aula Timur ITB (107.61104355690973, -6.892499983616666)
[8]Galeri Soemardja (107.61176284906381, -6.8923527458026275)
[9]Ikatan Mahasiswa Geodesi ITB (107.61234263141938, -6.891333180853256)
[10]Laboratorium Penelitian Konversi Energi Elektrik (107.6090223083595, -6.8908975578749585)

Pilih titik asal [1-10]: 1
Pilih titik tujuan [1-10]: 8
```

Hasil Algoritma UCS

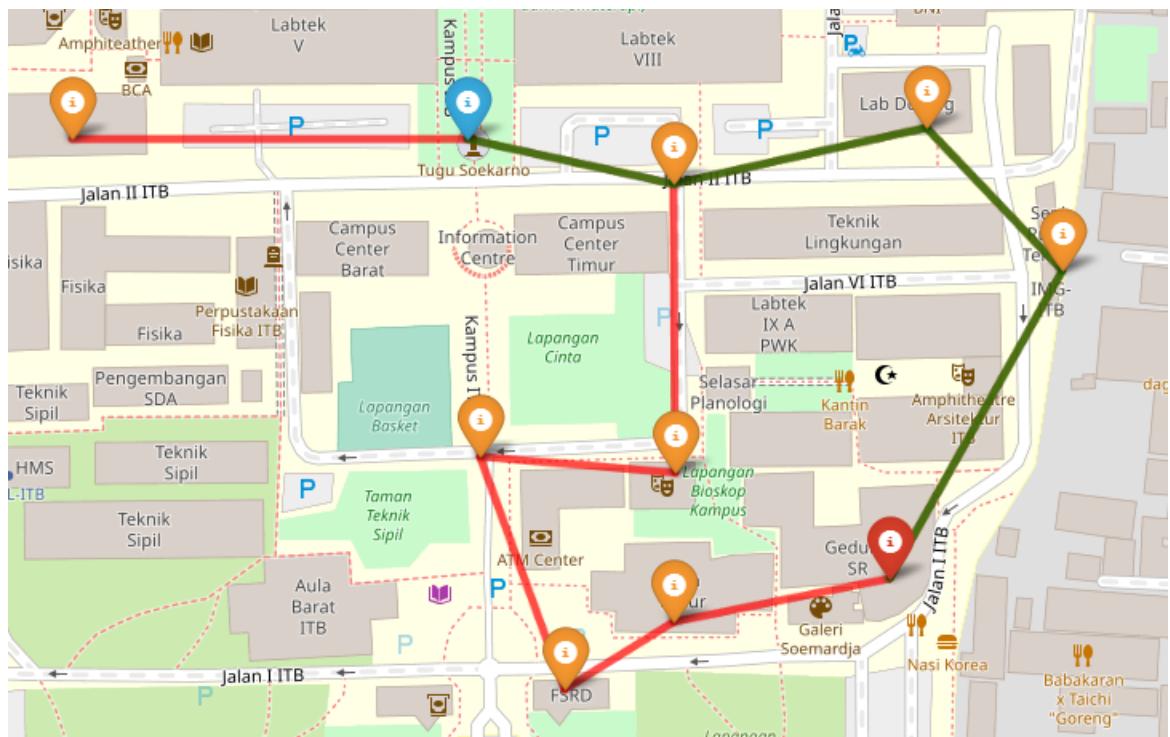
```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 1
#####
Lintasan : Plaza Widya Nusantara - Jl. I ITB - Laboratorium Pengujian Doping - Ikatan Mahasiswa Geodesi ITB - Galeri Soemardja
Jarak(km) : 0.3775148741360333
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Plaza Widya Nusantara - Jl. I ITB - Laboratorium Pengujian Doping - Ikatan

Jarak (km) : 0.3775148741360333

Visualisasi :



Hasil Algoritma A*

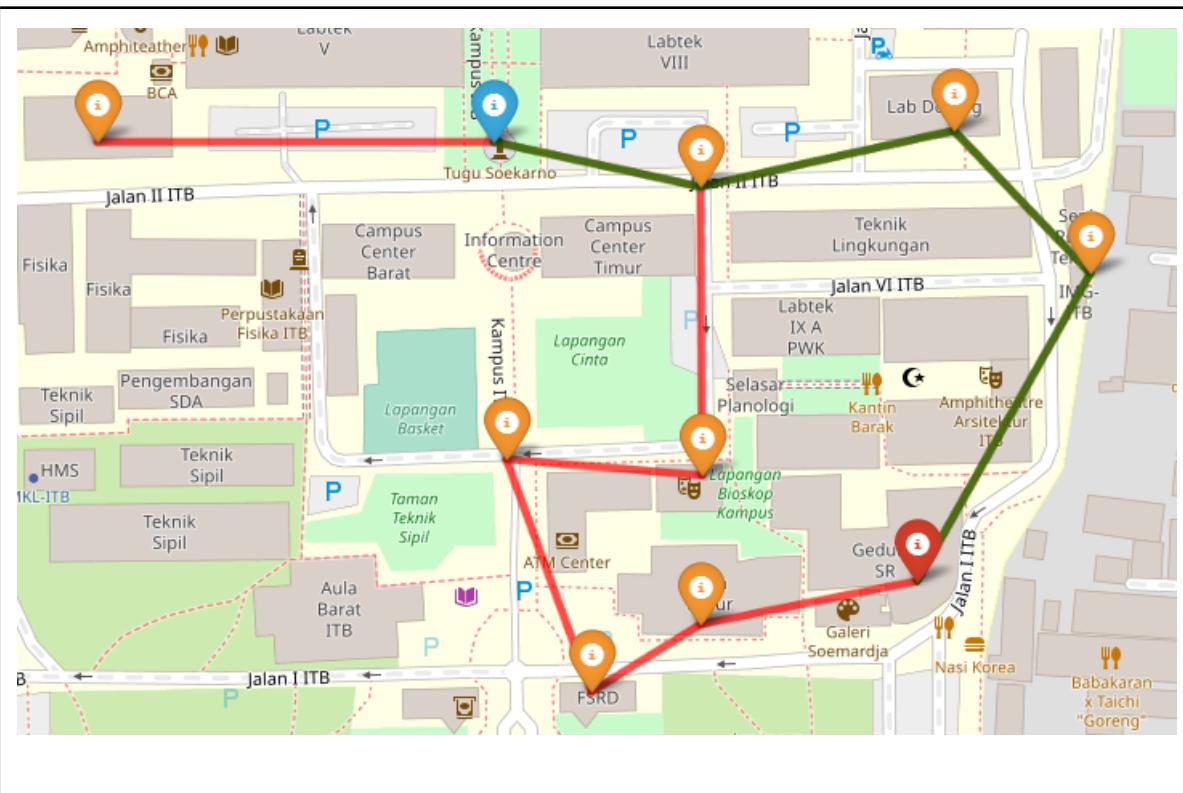
```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 2
#####
Lintasan  : Plaza Widya Nusantara - Jl. I ITB - Laboratorium Pengujian Doping - Ikatan Mahasiswa Geodesi ITB - Galeri Soemardja
Jarak(km) : 0.3775148741360333
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Plaza Widya Nusantara - Jl. I ITB - Laboratorium Pengujian Doping - Ikatan Mahasiswa Geodesi ITB - Galeri Soemardja

Jarak (km) : 0.3775148741360333

Visualisasi :



Pengujian kedua :

Mencari lintasan terpendek dari simpul ke-9 (Ikatan Mahasiswa Geodesi) ke simpul ke-5 (Jl. A ITB)

```

Masukkan nama file (dalam folder test): Map ITB2.txt
Map ITB2.txt
Graph berhasil dibaca dari file input.
[1]Plaza Widya Nusantara (107.61034662460492, -6.890895299442325)
[2]Jl. I ITB (107.6110413429995, -6.891044787183361)
[3]Kelas Kuliah 9009 (107.61104403541327, -6.892005264397014)
[4]Laboratorium Pengujian Doping (107.61188707117256, -6.890855470915589)
[5]Jl. A ITB (107.61039257954421, -6.891948643023322)
[6]Fakultas Seni Rupa dan Desain (107.61067366146906, -6.8927194214051415)
[7]Aula Timur ITB (107.61104355690973, -6.892499983616666)
[8]Galeri Soemardja (107.61176284906381, -6.8923527458026275)
[9]Ikatan Mahasiswa Geodesi ITB (107.61234263141938, -6.891333180853256)
[10]Laboratorium Penelitian Konversi Energi Elektrik (107.6090223083595, -6.8908975578749585)

Pilih titik asal [1-10]: 9
Pilih titik tujuan [1-10]: 5

```

Hasil Algoritma UCS

Pilih algoritma yang ingin digunakan

```
#####
1. Uniform Cost Search
2. A*
#####
```

Pilihan(1-2): 1

```
#####

```

Lintasan : Ikatan Mahasiswa Geodesi ITB - Laboratorium Pengujian Doping - Jl. I ITB - Kelas Kuliah 9009 - Jl. A ITB
Jarak(km) : 0.3478445966709579

```
#####

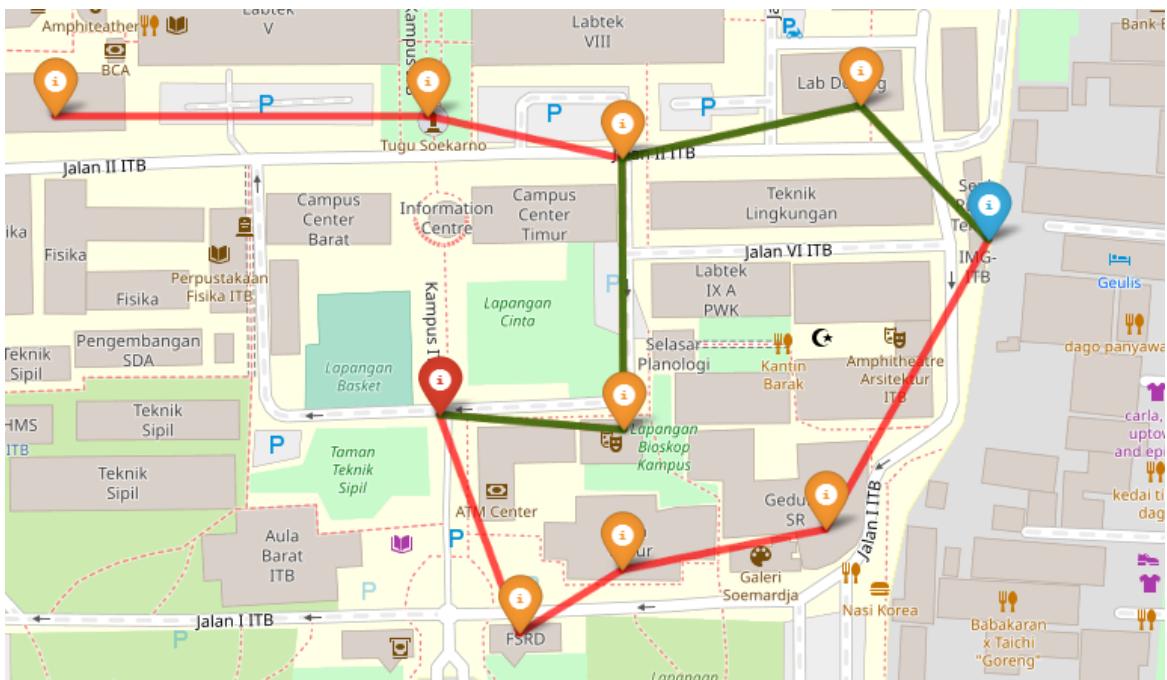
```

Map hasil visualisasi graf dapat dilihat pada localhost:5000

Lintasan : Ikatan Mahasiswa Geodesi ITB - Laboratorium Pengujian Doping - Jl. I ITB - Kelas Kuliah 9009 - Jl. A ITB

Jarak (km) : 0.3478445966709579

Visualisasi :



Hasil Algoritma A*

Pilih algoritma yang ingin digunakan

```
#####
1. Uniform Cost Search
2. A*
#####

```

Pilihan(1-2): 2

```
#####

```

Lintasan : Ikatan Mahasiswa Geodesi ITB - Laboratorium Pengujian Doping - Jl. I ITB - Kelas Kuliah 9009 - Jl. A ITB
Jarak(km) : 0.3478445966709579

```
#####

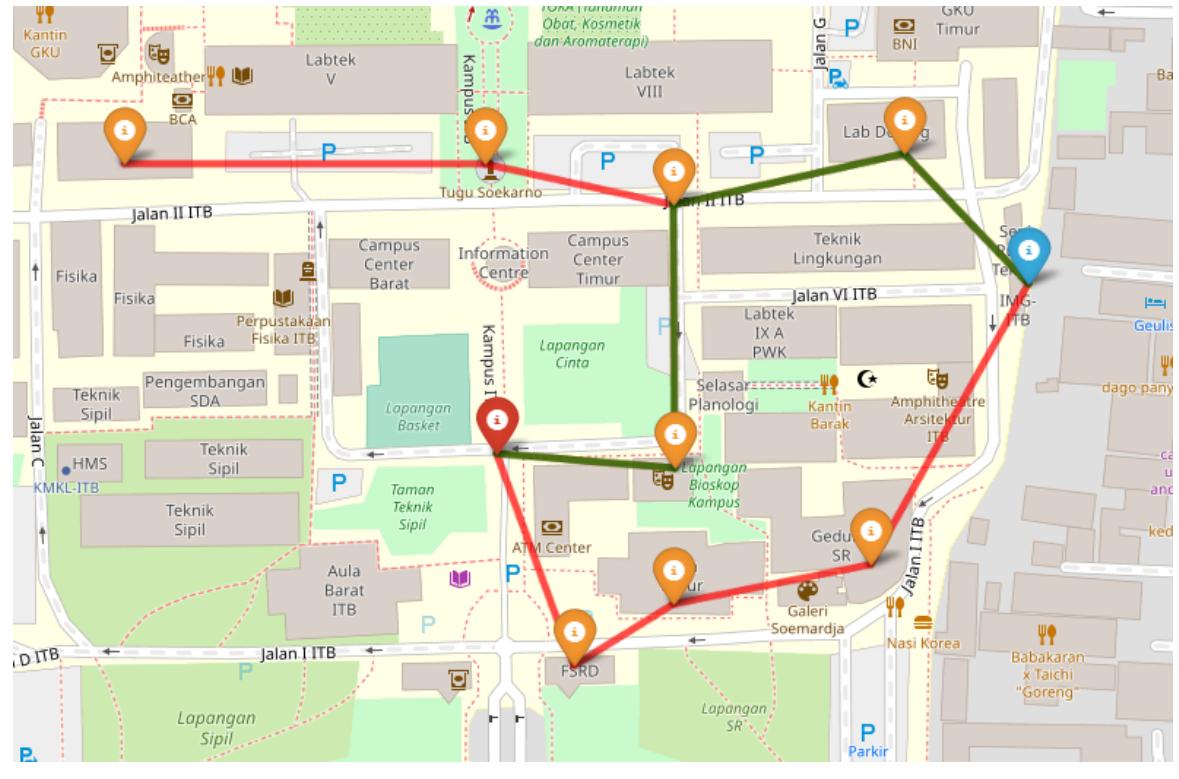
```

Map hasil visualisasi graf dapat dilihat pada localhost:5000

Lintasan : Ikatan Mahasiswa Geodesi ITB - Laboratorium Pengujian Doping - Jl. I ITB - Kelas Kuliah 9009 - Jl. A ITB

Jarak (km) : 0.3478445966709579

Visualisasi :



2. Uji coba dengan peta jalan sekitar Alun-alun Bandung

file input :

10
Kantor Pos Bandung
-6.921059074042508,107.60648416401784
Monumen Asia Afrika
-6.921175462231705,107.60756695706431
Jl Alun-Alun Timur
-6.922524224973004,107.60754971801818
Gerbang Alun-Alun
-6.922555034702758,107.60731177866587
Jl Alun Dalam Tenggara

-6.9223462162952885,107.60728763938474
Jl Alun Dalam Timur Laut
-6.921411667504991,107.60745316046116
Jl Alun Dalam Barat Laut
-6.921302124814193,107.60688072810532
Jl Alun Dalam Barat Daya
-6.922281175417579,107.60646347403119
Masjid Raya Bandung
-6.921949120113879,107.60625657080152
Parahyangan Plaza
-6.922404412455186,107.60615656749039
0 0.1202223425059817 0 0 0 0 0.051445304568855835 0 0 0
0.1202223425059817 0 0.1499876459473663 0 0 0 0 0 0
0 0.1499876459473663 0 0.026487260644031528 0 0 0 0 0
0 0 0.026487260644031528 0 0.023371937693006172 0 0 0 0 0.12861239907593103
0 0 0 0.023371937693006172 0 0.10551108109078657 0 0.09126198857984519 0 0
0 0 0 0 0.10551108109078657 0 0.06435102032012648 0 0 0
0.051445304568855835 0 0 0 0 0.06435102032012648 0 0.11820770967848501
0.09961212959282403 0
0 0 0 0 0.09961212959282403 0 0 0.051815704211098136
0 0 0 0.12861239907593103 0 0 0 0 0.051815704211098136 0

Pengujian pertama :

Mencari lintasan terpendek dari simpul ke-9 (Masjid Raya Bandung) ke simpul ke-1 (Kantor Pos Bandung)

```
Masukkan nama file (dalam folder test): Alun Alun Kota Bandung.txt  
Alun Alun Kota Bandung.txt  
Graph berhasil dibaca dari file input.  
[1]Kantor Pos Bandung (107.60648416401784, -6.921059074042508)  
[2]Monumen Asia Afrika (107.60756695706431, -6.921175462231705)  
[3]Jl Alun-Alun Timur (107.60754971801818, -6.922524224973004)  
[4]Gerbang Alun-Alun (107.60731177866587, -6.922555034702758)  
[5]Jl Alun Dalam Tenggara (107.60728763938474, -6.9223462162952885)  
[6]Jl Alun Dalam Timur Laut (107.60745316046116, -6.921411667504991)  
[7]Jl Alun Dalam Barat Laut (107.60688072810532, -6.921302124814193)  
[8]Jl Alun Dalam Barat Daya (107.60646347403119, -6.922281175417579)  
[9]Masjid Raya Bandung (107.60625657080152, -6.921949120113879)  
[10]Parahyangan Plaza (107.60615656749039, -6.922404412455186)
```

```
Pilih titik asal [1-10]: 9
```

```
Pilih titik tujuan [1-10]: 1
```

Hasil Algoritma UCS

```
Pilih algoritma yang ingin digunakan
```

```
#####
1. Uniform Cost Search
2. A*
#####
```

```
Pilihan(1-2): 1
```

```
#####
```

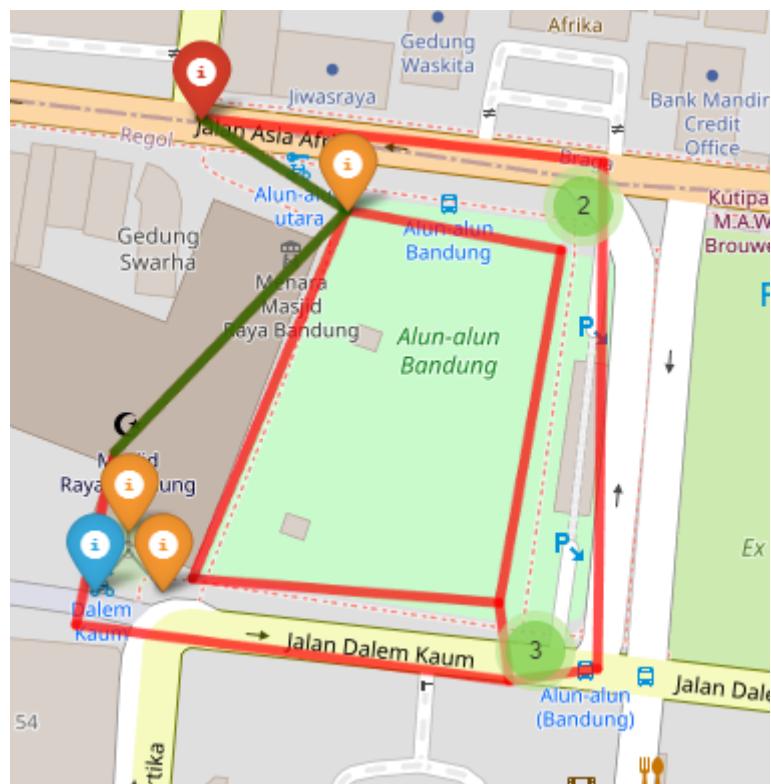
```
Lintasan : Masjid Raya Bandung - Jl Alun Dalam Barat Laut - Kantor Pos Bandung
Jarak(km) : 0.15105743416167988
```

```
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Masjid Raya Bandung - Jl Alun Dalam Barat Laut - Kantor Pos Bandung

Jarak (km) : 0.15105743416167988

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

Hasil Algoritma A*

```
Pilih algoritma yang ingin digunakan
```

```
#####
1. Uniform Cost Search
2. A*
#####
```

```
Pilihan(1-2): 2
```

```
#####

```

```
Lintasan : Masjid Raya Bandung - Jl Alun Dalam Barat Laut - Kantor Pos Bandung
Jarak(km) : 0.15105743416167988
```

```
#####

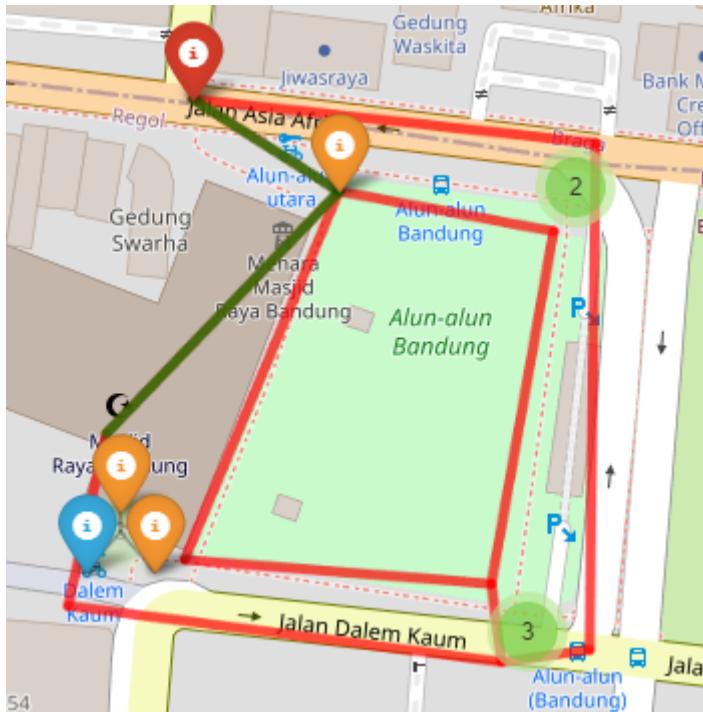
```

```
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Masjid Raya Bandung - Jl Alun Dalam Barat Laut - Kantor Pos Bandung

Jarak (km) : 0.15105743416167988

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

Pengujian kedua :

Mencari lintasan terpendek dari simpul ke-7 (Masjid Raya Bandung) ke simpul ke-5 (Kantor Pos Bandung)

```
Masukkan nama file (dalam folder test): Alun Alun Kota Bandung.txt  
Alun Alun Kota Bandung.txt  
Graph berhasil dibaca dari file input.  
[1]Kantor Pos Bandung (107.60648416401784, -6.921059074042508)  
[2]Monumen Asia Afrika (107.60756695706431, -6.921175462231705)  
[3]Jl Alun-Alun Timur (107.60754971801818, -6.922524224973004)  
[4]Gerbang Alun-Alun (107.60731177866587, -6.922555034702758)  
[5]Jl Alun Dalam Tenggara (107.60728763938474, -6.9223462162952885)  
[6]Jl Alun Dalam Timur Laut (107.60745316046116, -6.921411667504991)  
[7]Jl Alun Dalam Barat Laut (107.60688072810532, -6.921302124814193)  
[8]Jl Alun Dalam Barat Daya (107.60646347403119, -6.922281175417579)  
[9]Masjid Raya Bandung (107.60625657080152, -6.921949120113879)  
[10]Parahyangan Plaza (107.60615656749039, -6.922404412455186)  
  
Pilih titik asal [1-10]: 7  
  
Pilih titik tujuan [1-10]: 5
```

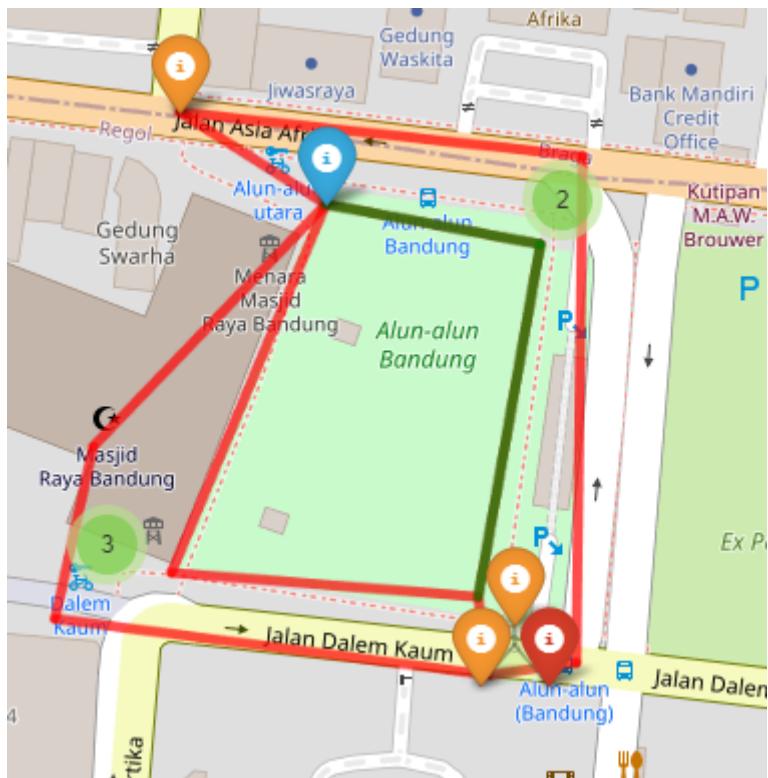
Hasil Algoritma UCS

```
Pilih algoritma yang ingin digunakan  
#####  
1. Uniform Cost Search  
2. A*  
#####  
  
Pilihan(1-2): 1  
  
#####  
Lintasan : Jl Alun Dalam Barat Laut - Jl Alun Dalam Timur Laut - Jl Alun Dalam Tenggara  
Jarak(km) : 0.16986210141091304  
  
#####  
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Jl Alun Dalam Barat Laut - Jl Alun Dalam Timur Laut - Jl Alun Dalam Tenggara

Jarak (km) : 0.16986210141091304

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

Hasil Algoritma A*

Pilih algoritma yang ingin digunakan

```
#####
1. Uniform Cost Search
2. A*
#####
```

Pilihan(1-2): 2

```
#####

```

Lintasan : Jl Alun Dalam Barat Laut - Jl Alun Dalam Timur Laut - Jl Alun Dalam Tenggara
Jarak(km) : 0.16986210141091304

```
#####

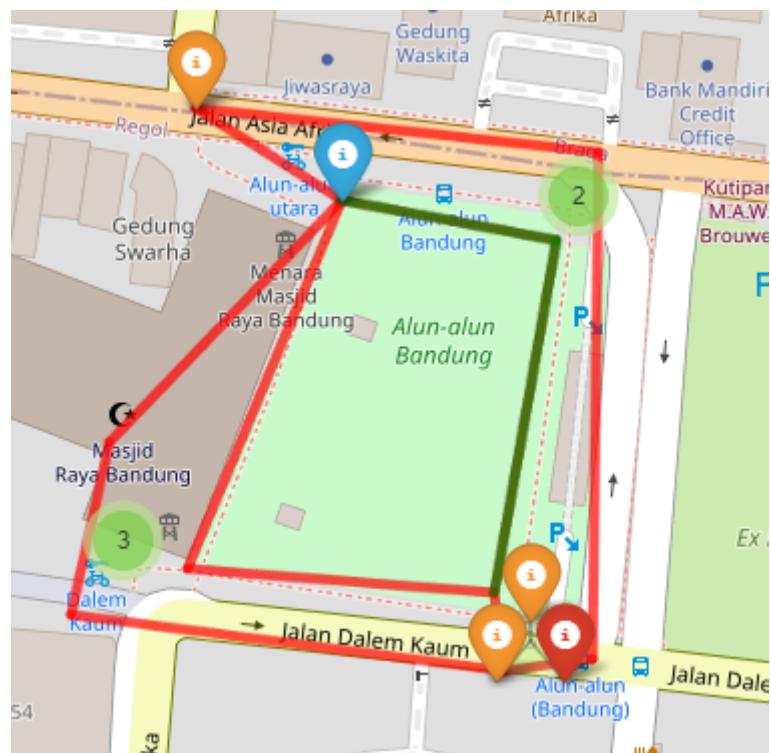
```

Map hasil visualisasi graf dapat dilihat pada localhost:5000

Lintasan : Jl Alun Dalam Barat Laut - Jl Alun Dalam Timur Laut - Jl Alun Dalam Tenggara

Jarak (km) : 0.16986210141091304

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

3. Uji coba dengan peta jalan sekitar daerah Kopo

file input : "Map Daerah Kopo.txt"

9
Rumah Sakit Immanuel
-6.935315651653132, 107.59618451751646
Hotel Grand Tulip
-6.93005103032233, 107.59748899998054
Toko TTH
-6.930249878187188, 107.5997071513455
Gapensa Grosir
-6.93207302805938, 107.59959649507888
Bengkel Nanang Dolenk
-6.932045590091672, 107.59752179702505
Kelapa Bakar Kuningan
-6.93134922938818, 107.59708524857666

Toko Perhiasan Perak Matahari Silver
-6.926660779276434, 107.59820692906548
Nasi Kuning Ibu Elly
-6.926912470904012, 107.59941986522041
Bubur Ayam Mentari
-6.929964885390252, 107.59940348922812
0 0 0 0 0.4521136978004759 0 0 0
0 0 0.2458415103925541 0 0 0.15107631833236373 0.38521819897040105 0 0
0 0 0.2458415103925541 0 0.20309265391328843 0 0 0 0 0.0461276883961913
0 0 0.20309265391328843 0 0.22902982505540104 0 0 0 0
0 0 0 0.22902982505540104 0 0.09120133292561669 0 0 0
0.4521136978004759 0.15107631833236373 0 0 0.09120133292561669 0 0 0
0 0.38521819897040105 0 0 0 0 0.13678171880184864 0
0 0 0 0 0 0.13678171880184864 0 0.33941781834849893
0 0 0.0461276883961913 0 0 0 0 0.33941781834849893 0

Pengujian pertama :

Mencari lintasan terpendek dari simpul ke-2 (Hotel Grand Tulip) ke simpul ke-8 (Nasi Kuning Ibu Elly)

```
Masukkan nama file (dalam folder test): Map Daerah Kopo.txt
Map Daerah Kopo.txt
Graph berhasil dibaca dari file input.
[1]Rumah Sakit Immanuel (107.59618451751646, -6.935315651653132)
[2]Hotel Grand Tulip (107.5974889998054, -6.93005103032233)
[3]Toko TTH (107.5997071513455, -6.930249878187188)
[4]Gapensa Grosir (107.59959649507888, -6.93207302805938)
[5]Bengkel Nanang Dolenk (107.59752179702505, -6.932045590091672)
[6]Kelapa Bakar Kuningan (107.59708524857666, -6.93134922938818)
[7]Toko Perhiasan Perak Matahari Silver (107.59820692906548, -6.926660779276434)
[8]Nasi Kuning Ibu Elly (107.59941986522041, -6.926912470904012)
[9]Bubur Ayam Mentari (107.59940348922812, -6.929964885390252)

Pilih titik asal [1-9]: 2
Pilih titik tujuan [1-9]: 8
```

Hasil Algoritma UCS

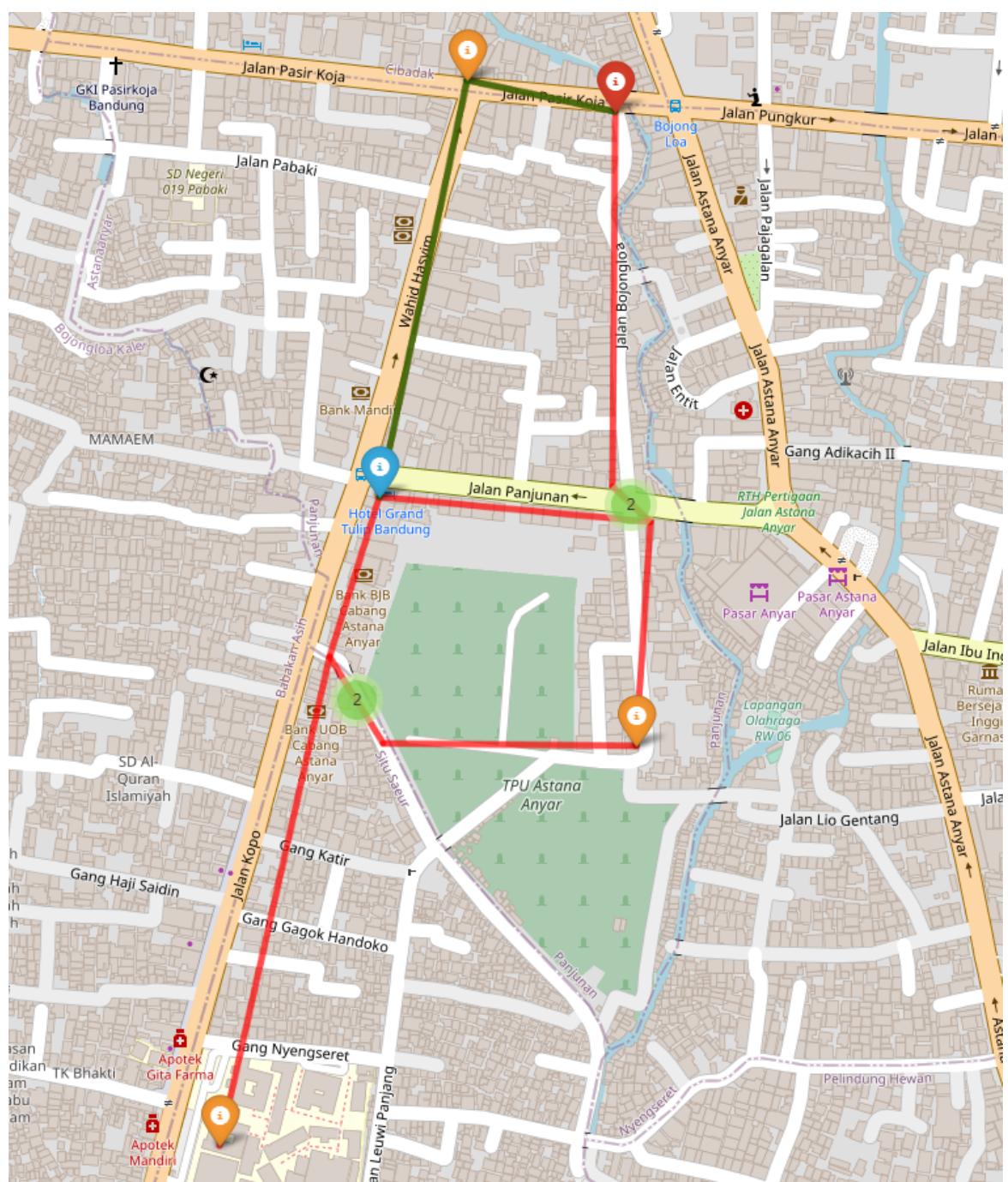
```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 1
#####
Lintasan : Hotel Grand Tulip - Toko Perhiasan Perak Matahari Silver - Nasi Kuning Ibu Elly
Jarak(km) : 0.5219999177722496
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Hotel Grand Tulip - Toko Perhiasan Perak Matahari Silver - Nasi Kuning Ibu Elly

Jarak (km) : 0.5219999177722496

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

Hasil Algoritma A*

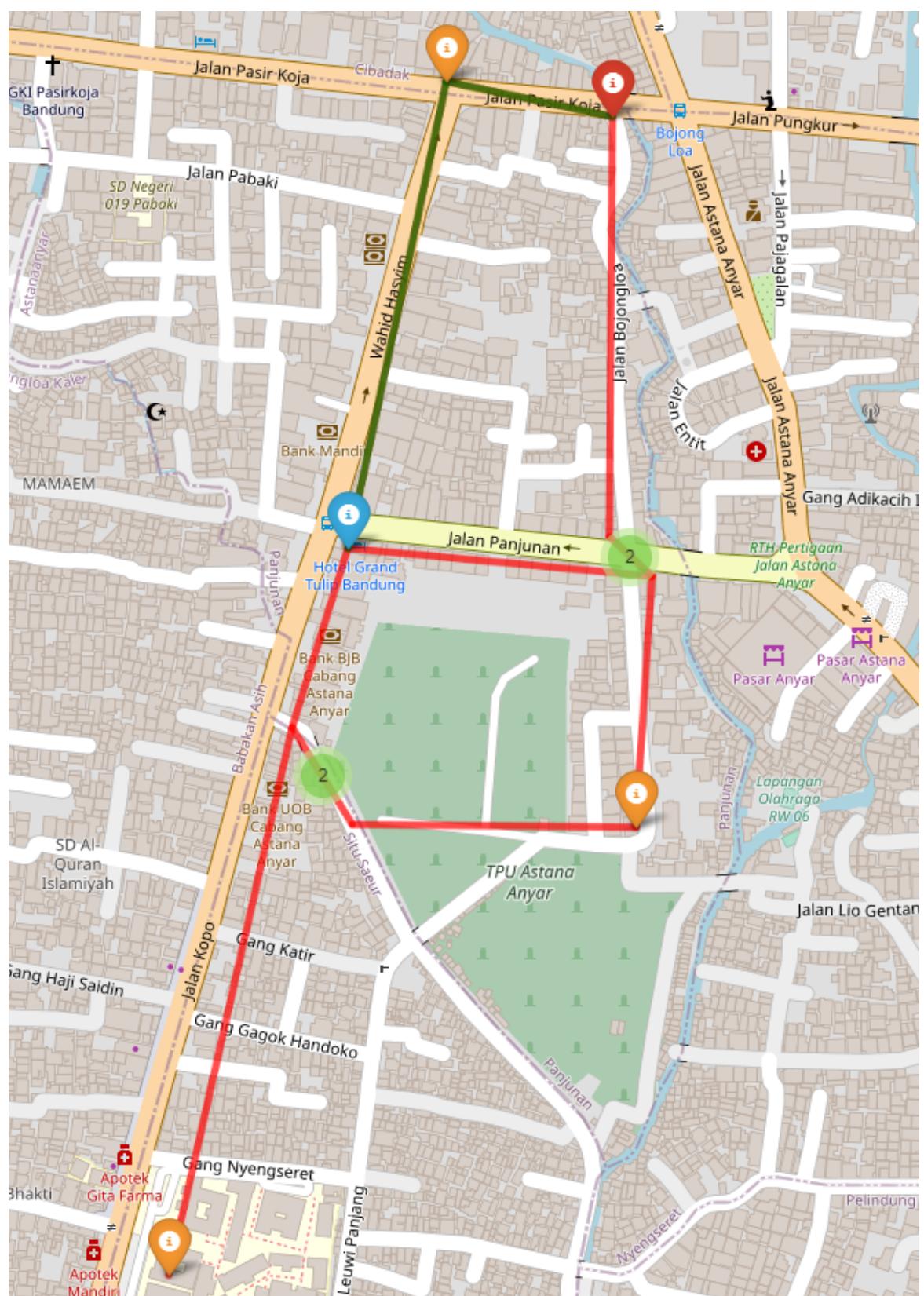
```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 2
#####
Lintasan : Hotel Grand Tulip - Toko Perhiasan Perak Matahari Silver - Nasi Kuning Ibu Elly
Jarak(km) : 0.5219999177722496
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Hotel Grand Tulip - Toko Perhiasan Perak Matahari Silver - Nasi Kuning Ibu Elly

Jarak (km) : 0.5219999177722496

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

Pengujian kedua :

Mencari lintasan terpendek dari simpul ke-4 (Gapensa Grosir) ke simpul ke-7 (Toko Perhiasan Perak Matahari Silver)

```
Masukkan nama file (dalam folder test): Map Daerah Kopo.txt
Map Daerah Kopo.txt
Graph berhasil dibaca dari file input.
[1]Rumah Sakit Immanuel (107.59618451751646, -6.935315651653132)
[2]Hotel Grand Tulip (107.59748899998054, -6.93005103032233)
[3]Toko TTH (107.5997071513455, -6.930249878187188)
[4]Gapensa Grosir (107.59959649507888, -6.93207302805938)
[5]Bengkel Nanang Dolenk (107.59752179702505, -6.932045590091672)
[6]Kelapa Bakar Kuningan (107.59708524857666, -6.93134922938818)
[7]Toko Perhiasan Perak Matahari Silver (107.59820692906548, -6.926660779276434)
[8]Nasi Kuning Ibu Elly (107.59941986522041, -6.926912470904012)
[9]Bubur Ayam Mentari (107.59940348922812, -6.929964885390252)

Pilih titik asal [1-9]: 4

Pilih titik tujuan [1-9]: 7
```

Hasil Algoritma UCS

```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

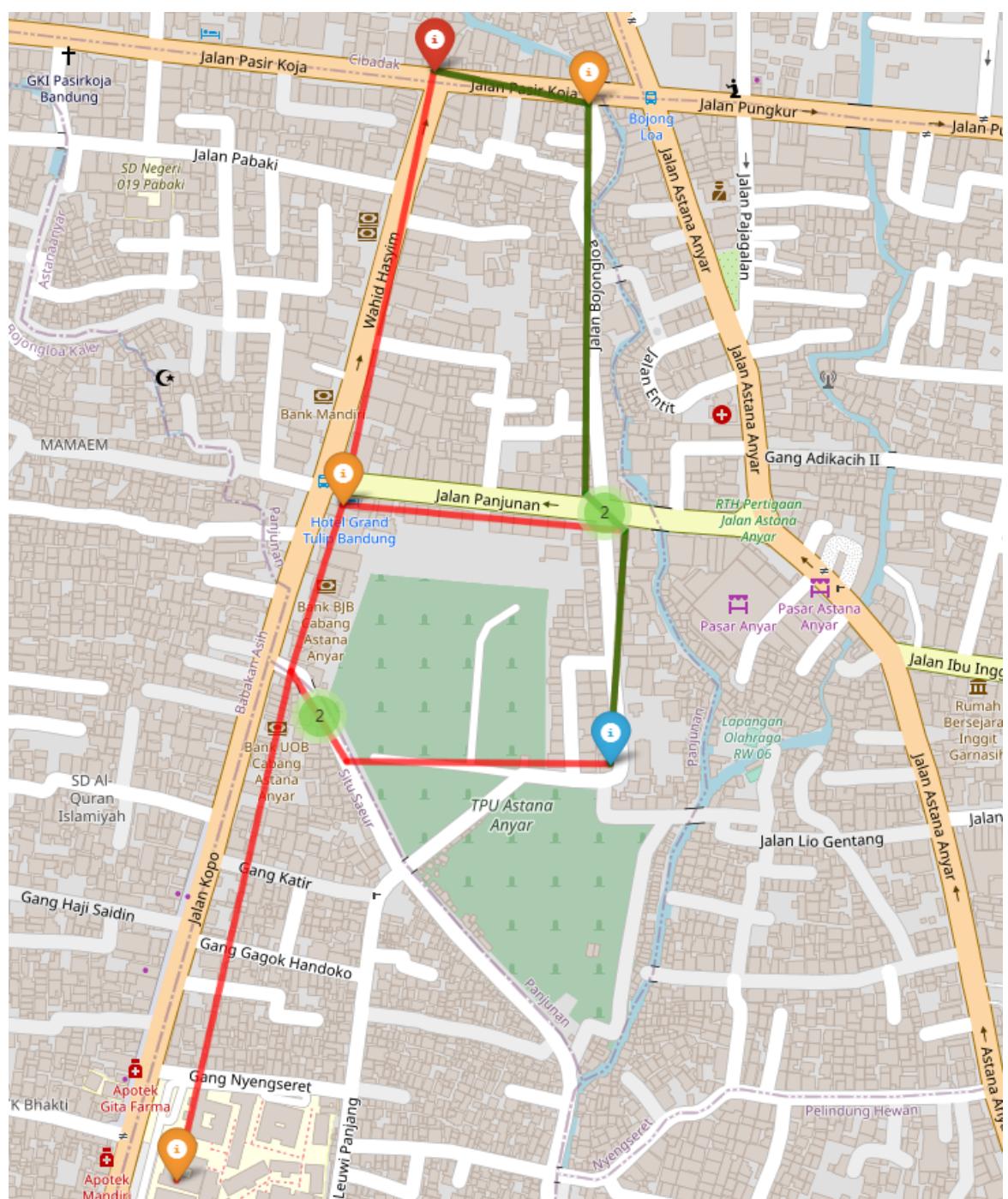
Pilihan(1-2): 1

#####
Lintasan : Gapensa Grosir - Toko TTH - Bubur Ayam Mentari - Nasi Kuning Ibu Elly - Toko Perhiasan Perak Matahari Silver
Jarak(km) : 0.7254198794598273
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Gapensa Grosir - Toko TTH - Bubur Ayam Mentari - Nasi Kuning Ibu Elly - Toko Perhiasan Perak Matahari Silver

Jarak (km) : 0.7254198794598273

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

Hasil Algoritma A*

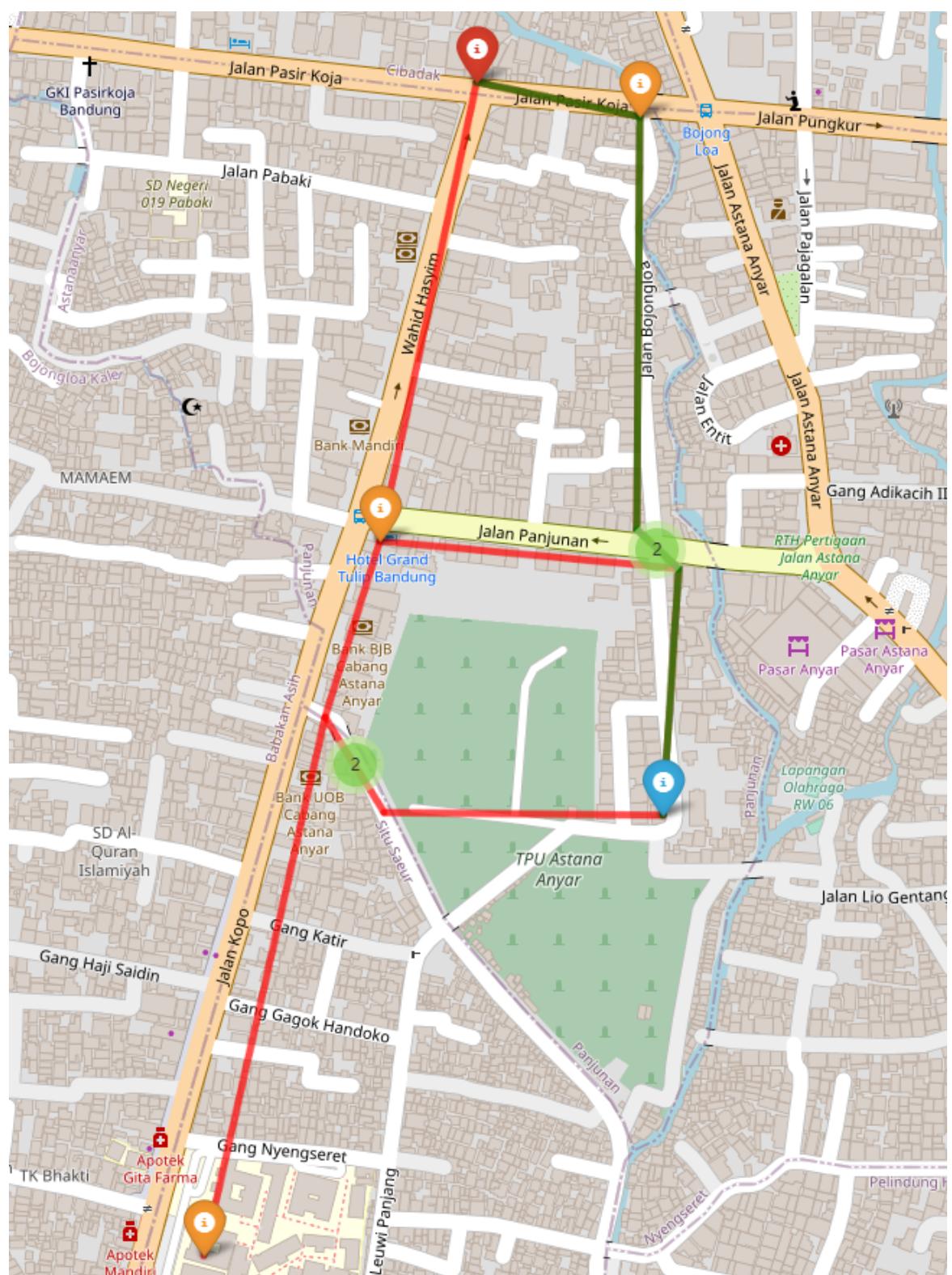
```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 2
#####
Lintasan : Gapensa Grosir - Toko TTH - Bubur Ayam Mentari - Nasi Kuning Ibu Elly - Toko Perhiasan Perak Matahari Silver
Jarak(km) : 0.7254198794598273
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Gapensa Grosir - Toko TTH - Bubur Ayam Mentari - Nasi Kuning Ibu Elly - Toko Perhiasan Perak Matahari Silver

Jarak (km) : 0.7254198794598273

Visualisasi :



Note : warna bulat hijau bertuliskan angka (n), menandakan ada simpul sebanyak n yang berdekatan di daerah tersebut, tetapi tidak dapat divisualisasikan karena akan bertabrakan dalam visualisasinya.

4. Uji coba dengan peta jalan di Kota Ambon

file input : "Map Kota Ambon.txt"

11
Gereja Silo
-3.698444396272507, 128.17986996343117
CV Immanuel
-3.699521598341391, 128.18143689113748
Fotocopy Setiabudi
-3.697199517307983, 128.18366462439317
Agen Sewa Mobil Ambon
-3.696245501482123, 128.18318706873524
SD Kristen Urimessing
-3.700474046287815, 128.1822736471614
Nasi Kuning Ibu Stien
-3.699510992765935, 128.18318358780436
Toko Biasa Ambon
-3.697512149280845, 128.18128250081327
Drg. Adi Setiawan
-3.6987007746809617, 128.18203016812922
Jl. Diponegoro 63
-3.702274413798278, 128.18411311622503
RRI Ambon
-3.700938478898734, 128.184797119721
Bank Mandiri Taspen
-3.6992956880320342, 128.1854724676014
0 0.21113593601471917 0 0 0 0 0.18791765836859844 0 0 0 0
0.21113593601471917 0 0 0 0.14084504299110626 0 0 0.11253585768902233 0 0 0
0 0 0 0.1185808028627856 0 0 0 0.24649455634054587 0 0 0.30752245264264527
0 0 0 0.1185808028627856 0 0 0 0.2539701649495007 0 0 0 0
0 0.14084504299110626 0 0 0 0.14718176366462904 0 0 0.2859000446286319 0 0
0 0 0 0 0.14718176366462904 0 0 0.15651619970380895 0 0.23927222343053245 0
0.18791765836859844 0 0 0.2539701649495007 0 0 0 0.1560501534278865 0 0 0
0 0.11253585768902233 0.24649455634054587 0 0 0.15651619970380895
0.1560501534278865 0 0 0 0
0 0 0 0 0.2859000446286319 0 0 0 0 0.16681584229950785 0
0 0 0 0 0 0.23927222343053245 0 0 0.16681584229950785 0 0.19744402856755092
0 0 0.30752245264264527 0 0 0 0 0 0.19744402856755092 0

Pengujian pertama :

Mencari lintasan terpendek dari simpul ke-1 (Gereja Silo) ke simpul ke-8 (Drg. Adi Setiawan)

```
Masukkan nama file (dalam folder test): Map Kota Ambon.txt
Map Kota Ambon.txt
Graph berhasil dibaca dari file input.
[1]Gereja Silo (128.17986996343117, -3.698444396272507)
[2]CV Immanuel (128.18143689113748, -3.699521598341391)
[3]Fotocopy Setiabudi (128.18366462439317, -3.697199517307983)
[4]Agen Sewa Mobil Ambon (128.18318706873524, -3.696245501482123)
[5]SD Kristen Urimessing (128.1822736471614, -3.700474046287815)
[6]Nasi Kuning Ibu Stien (128.18318358780436, -3.699510992765935)
[7]Toko Biasa Ambon (128.18128250081327, -3.697512149280845)
[8]Drg. Adi Setiawan (128.18203016812922, -3.6987007746809617)
[9]Jl. Diponegoro 63 (128.18411311622503, -3.702274413798278)
[10]RRI Ambon (128.184797119721, -3.700938478898734)
[11]Bank Mandiri Taspen (128.1854724676014, -3.6992956880320342)
```

Pilih titik asal [1-11]: 1

Pilih titik tujuan [1-11]: 8

Hasil Algoritma UCS

```
Pilih algoritma yang ingin digunakan
```

```
#####
1. Uniform Cost Search
2. A*
#####
```

Pilihan(1-2): 1

```
#####
```

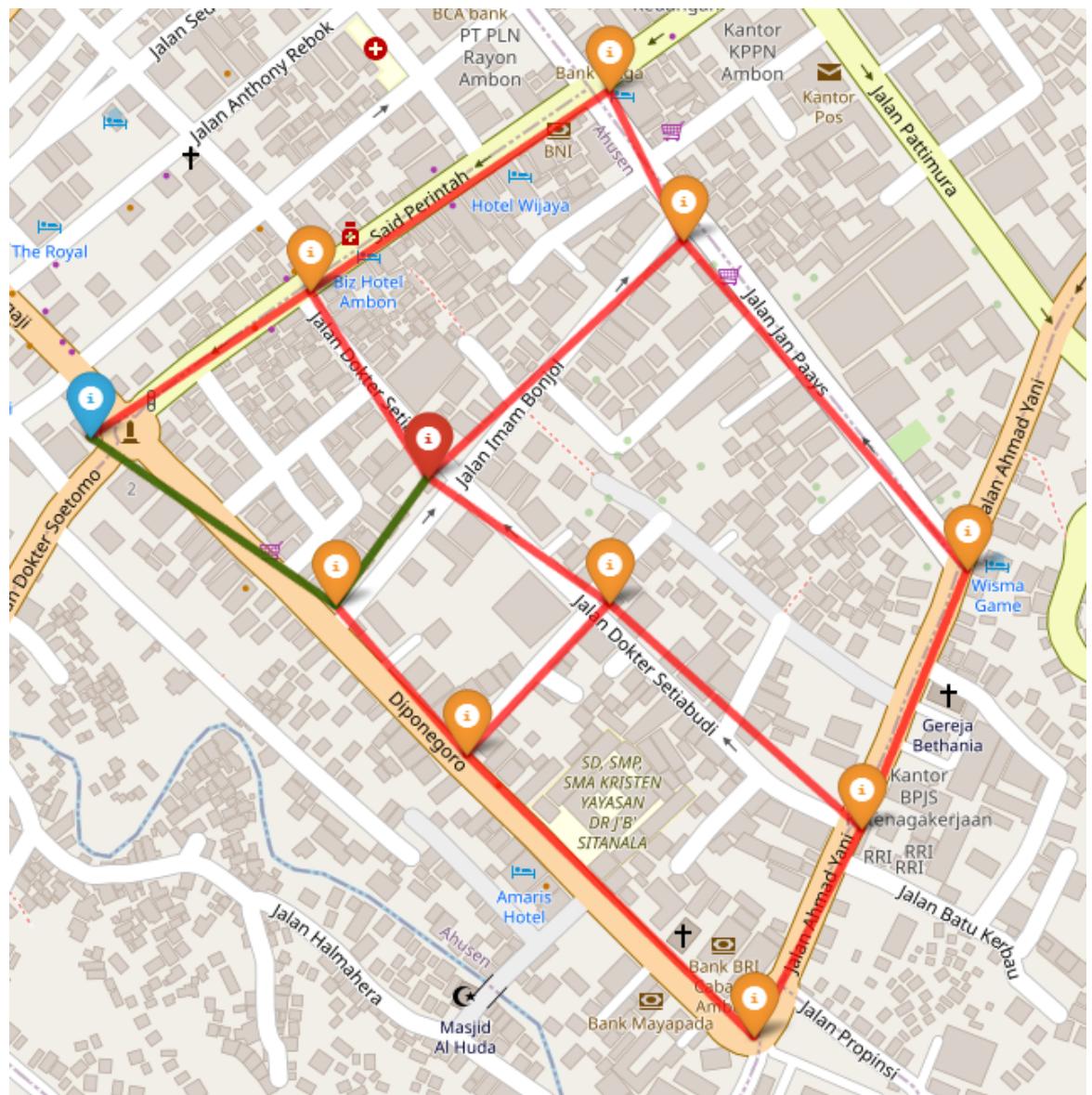
```
Lintasan : Gereja Silo - CV Immanuel - Drg. Adi Setiawan
Jarak(km) : 0.3236717937037415
```

```
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Gereja Silo - CV Immanuel - Drg. Adi Setiawan

Jarak (km) : 0.3236717937037415

Visualisasi :



Hasil Algoritma A*

```
Pilih algoritma yang ingin digunakan
```

```
#####
1. Uniform Cost Search
2. A*
#####
```

```
Pilihan(1-2): 2
```

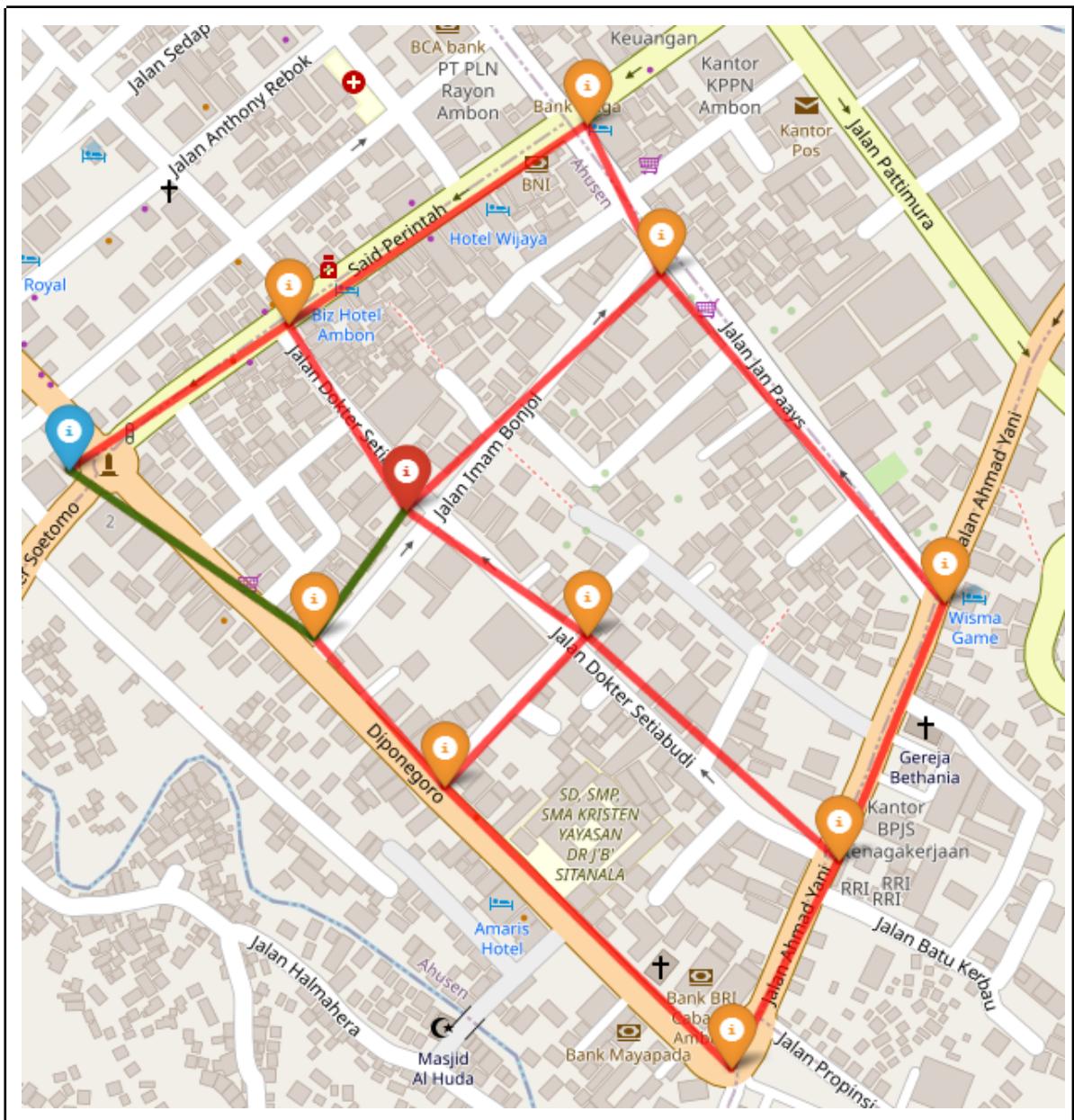
```
#####
Lintasan : Gereja Silo - CV Immanuel - Drg. Adi Setiawan
Jarak(km) : 0.3236717937037415
#####
```

```
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
#####
```

Lintasan : Gereja Silo - CV Immanuel - Drg. Adi Setiawan

Jarak (km) : 0.3236717937037415

Visualisasi :



Pengujian kedua :

Mencari lintasan terpendek dari simpul ke-4 (Agen Sewa Mobil Ambon) ke simpul ke-9 (Jl. Diponegoro 63)

```
Masukkan nama file (dalam folder test): Map Kota Ambon.txt
Map Kota Ambon.txt
Graph berhasil dibaca dari file input.
[1]Gereja Silo (128.17986996343117, -3.698444396272507)
[2]CV Immanuel (128.18143689113748, -3.699521598341391)
[3]Fotocopy Setiabudi (128.18366462439317, -3.697199517307983)
[4]Agen Sewa Mobil Ambon (128.18318706873524, -3.696245501482123)
[5]SD Kristen Urimessing (128.1822736471614, -3.700474046287815)
[6]Nasi Kuning Ibu Stien (128.18318358780436, -3.699510992765935)
[7]Toko Biasa Ambon (128.18128250081327, -3.697512149280845)
[8]Drg. Adi Setiawan (128.18203016812922, -3.6987007746809617)
[9]Jl. Diponegoro 63 (128.18411311622503, -3.702274413798278)
[10]RRI Ambon (128.184797119721, -3.700938478898734)
[11]Bank Mandiri Taspen (128.1854724676014, -3.6992956880320342)
```

Pilih titik asal [1-11]: 4

Pilih titik tujuan [1-11]: 9

Hasil Algoritma UCS

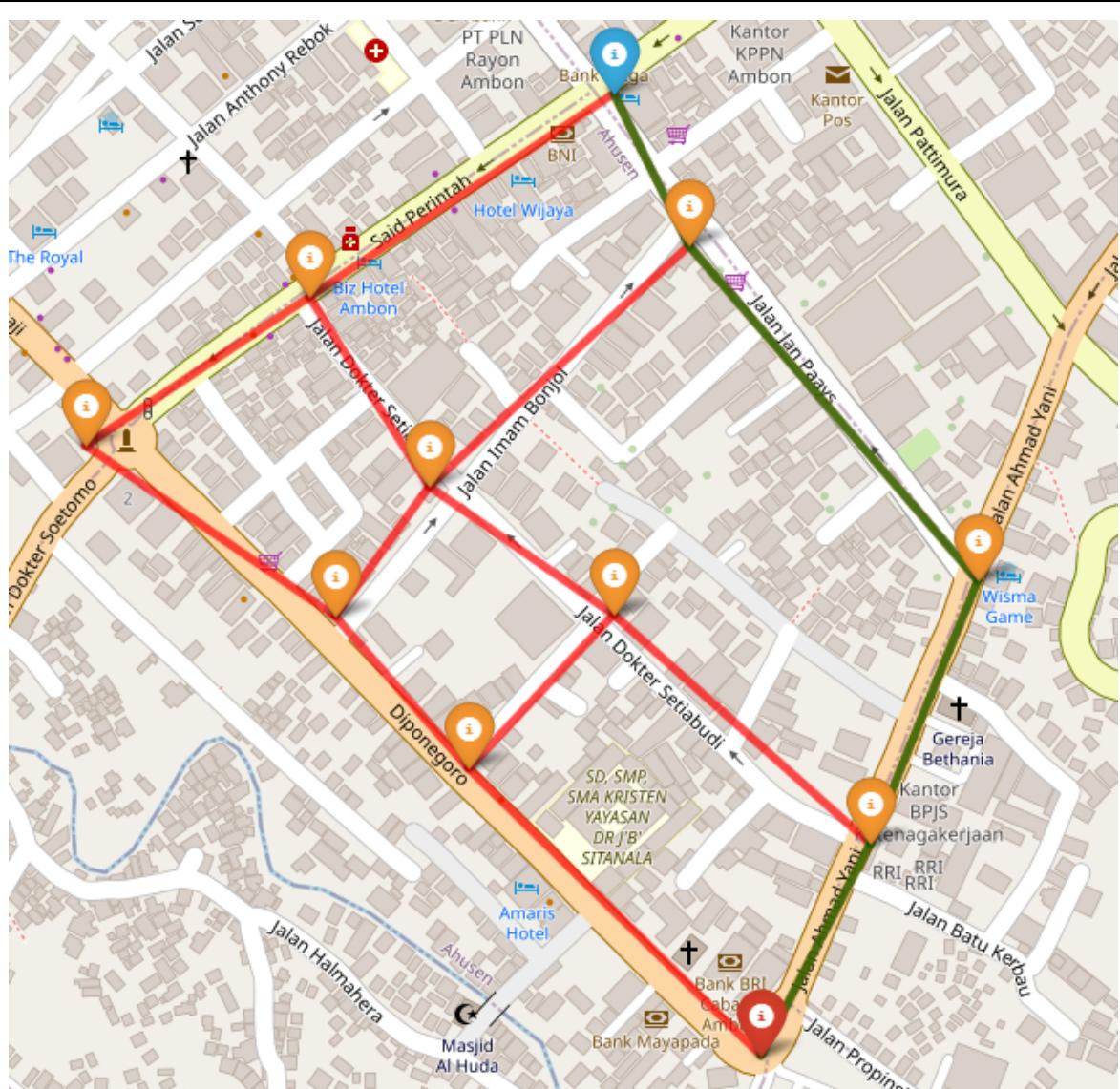
```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 1
#####
Lintasan : Agen Sewa Mobil Ambon - Fotocopy Setiabudi - Bank Mandiri Taspen - RRI Ambon - Jl. Diponegoro 63
Jarak(km) : 0.7903631263724896
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000
```

Lintasan : Agen Sewa Mobil Ambon - Fotocopy Setiabudi - Bank Mandiri Taspen - RRI Ambon - Jl. Diponegoro 63

Jarak (km) : 0.7903631263724896

Visualisasi :



Hasil Algoritma A*

Pilih algoritma yang ingin digunakan

```
#####
1. Uniform Cost Search
2. A*
#####
```

Pilihan(1-2): 2

```
#####

```

Lintasan : Agen Sewa Mobil Ambon - Fotocopy Setiabudi - Bank Mandiri Taspen - RRI Ambon - Jl. Diponegoro 63
Jarak(km) : 0.7903631263724896

```
#####
Map hasil visualisasi graf dapat dilihat pada localhost:5000

```

Lintasan : Agen Sewa Mobil Ambon - Fotocopy Setiabudi - Bank Mandiri Taspen - RRI Ambon - Jl. Diponegoro 63

Jarak (km) : 0.7903631263724896

Visualisasi :



5. Uji coba dengan peta jalan di Kota Makassar

file input : "Map Makassar.txt"

8

Pantai Losari

-5.1435791480543145,119.40807772925862

Aryaduta Makassar

-5.146392606156453,119.40873131954885

Jl. Sumba Opu

```
-5.147120793285612, 119.40891964217484
Centre Point of Indonesia
-5.148488900182874, 119.40775647301429
Jl. Haji Bau
-5.148632330574229, 119.4095621546635
Rumah Sakit Siloam
-5.149393129076859, 119.40695273340393
Universitas Ciputra Makassar
-5.14872196905484, 119.39529554240654
The Rinra
-5.152686051545299, 119.40525537315486
0 0.32110683404027024 0 0 0 0 0
0.32110683404027024 0 0.08361359448248039 0 0 0 0 0
0 0.08361359448248039 0 0.19933956321013896 0.18251703738414457 0 0 0
0 0 0.19933956321013896 0.0 0 0.1342848180922526 0 0
0 0 0.18251703738414457 0.20060753830711373 0 0.30111148046648845 0 0
0 0 0 0.1342848180922526 0.30111148046648845 0 0 0.4115886929889947
0 0 0 0 1.380245137004157 0 0 0 0
0 0 0 0 0.4115886929889947 0 0
```

Pengujian pertama :

Mencari lintasan terpendek dari simpul ke-1 (Pantai Losari) ke simpul ke-8 (The Rinra)

```
#####
Masukkan nama file (dalam folder test): Map Makassar.txt
Map Makassar.txt
Graph berhasil dibaca dari file input.
[1]Pantai Losari (119.4080772925862, -5.1435791480543145)
[2]Aryaduta Makassar (119.40873131954885, -5.146392606156453)
[3]Jl. Sumba Opu (119.40891964217484, -5.147120793285612)
[4]Centre Point of Indonesia (119.40775647301429, -5.148488900182874)
[5]Jl. Haji Bau (119.4095621546635, -5.148632330574229)
[6]Rumah Sakit Siloam (119.40695273340393, -5.149393129076859)
[7]Universitas Ciputra Makassar (119.39529554240654, -5.14872196905484)
[8]The Rinra (119.40525537315486, -5.152686051545299)
```

```
Pilih titik asal [1-8]: 1
```

```
Pilih titik tujuan [1-8]: 8
```

Hasil Algoritma UCS

```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 1
#####
Lintasan : Pantai Losari - Aryaduta Makassar - Jl. Sumba Opu - Centre Point of Indonesia - Rumah Sakit Siloam - The Rinra
Jarak(km) : 1.1499335028141369
#####
```

Lintasan : Pantai Losari - Aryaduta Makassar - Jl. Sumba Opu - Centre Point of Indonesia - Rumah Sakit Siloam - The Rinra

Jarak (km) : 1.1499335028141369

Visualisasi :



Hasil Algoritma A*

```
Pilih algoritma yang ingin digunakan
#####
1. Uniform Cost Search
2. A*
#####

Pilihan(1-2): 2
#####
Lintasan   : Pantai Losari - Aryaduta Makassar - Jl. Sumba Opu - Centre Point of Indonesia - Rumah Sakit Siloam - The Rinra
Jarak(km)  : 1.1499335028141369
#####
```

Lintasan : Pantai Losari - Aryaduta Makassar - Jl. Sumba Opu - Centre Point of Indonesia - Rumah Sakit Siloam - The Rinra

Jarak (km) : 1.1499335028141369

Visualisasi :



Pengujian kedua :

Mencari lintasan terpendek dari simpul ke-2 (Aryaduta Makassar) ke simpul ke-7 (Universitas Ciputra Makassar)

Hasil Algoritma UCS

Pilih algoritma yang ingin digunakan

```
#####
1. Uniform Cost Search
2. A*
#####
```

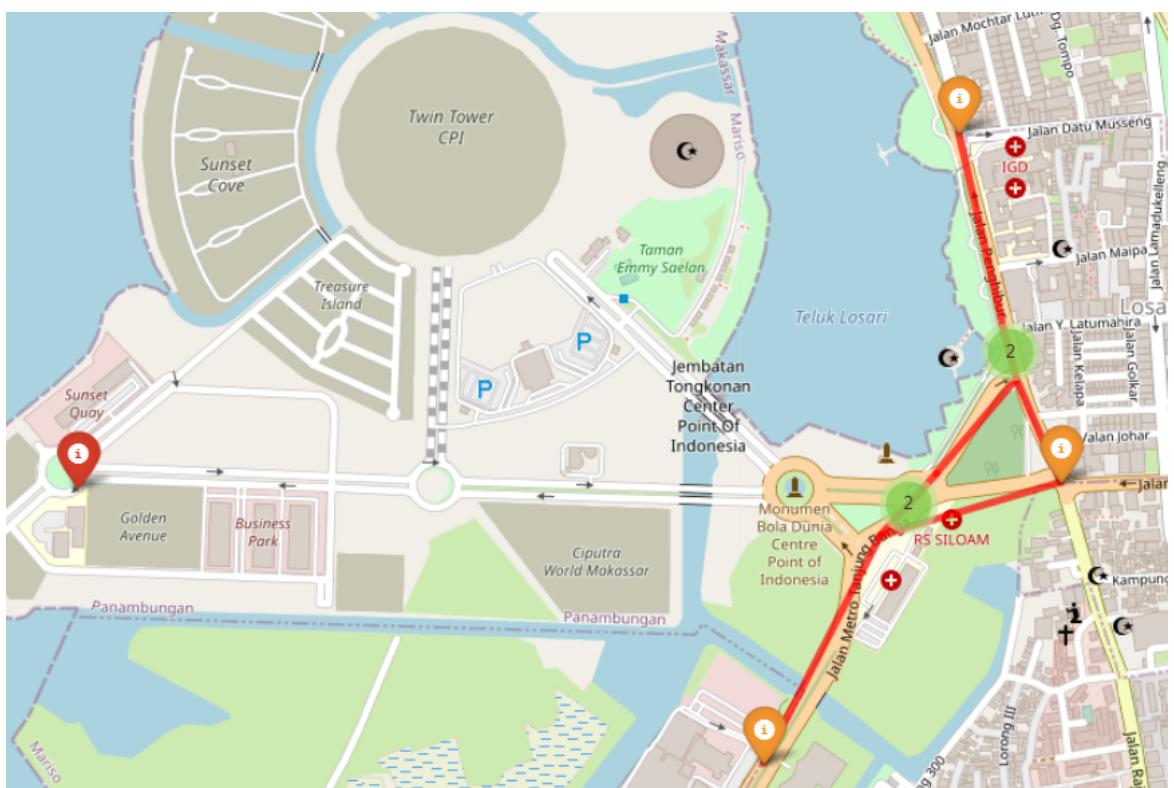
Pilihan(1-2): 1

```
#####
Tidak ada lintasan yang ditemukan!
#####
```

Lintasan : -

Jarak (km) : -

Visualisasi :



Hasil Algoritma A*

Pilih algoritma yang ingin digunakan

```
#####
1. Uniform Cost Search
2. A*
#####
```

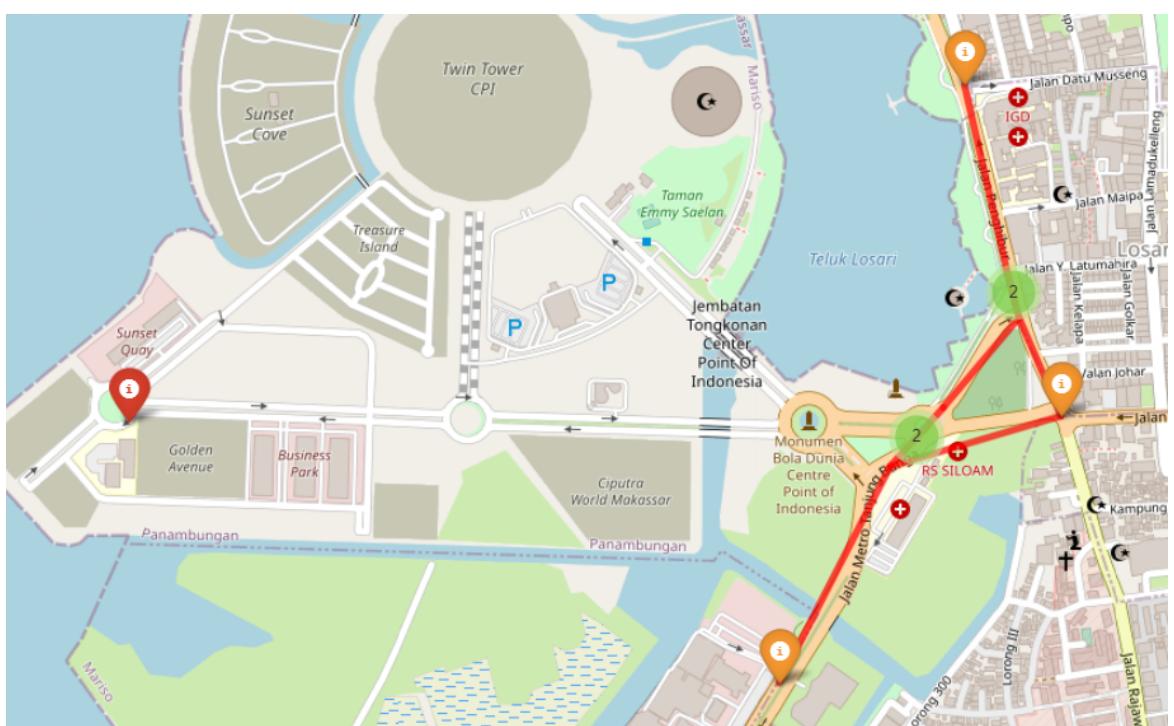
Pilihan(1-2): 2

```
#####
Tidak ada lintasan yang ditemukan!
#####
```

Lintasan : -

Jarak (km) : -

Visualisasi :



BAB V

KESIMPULAN DAN KOMENTAR

5.1 Kesimpulan

Algoritma *Uniform Cost Search (UCS)* dan *A** keduanya dapat digunakan untuk mencari jalur terpendek dari suatu titik asal ke titik tujuan dan dijamin akan selalu menemukan hasil yang optimal, tetapi akan memerlukan ruang pencarian yang luas dan juga waktu yang lebih lama dibanding algoritma *traversal* yang telah dipelajari sebelumnya, seperti DFS dan BFS.

5.2 Komentar Terkait Tugas

- Spesifikasi Tugas Kecil ini seharusnya dibuat lebih jelas dan serinci mungkin untuk menghindari kebingungan saat penggerjaan, terdapat hal-hal kecil yang tidak dirincikan dalam spesifikasi sehingga membuat kebingungan saat mengerjakan, contohnya seperti format file input yang tidak jelas.
- Tugas ini mendorong mahasiswa untuk mengeksplor teknologi-teknologi baru mengingat bahwa spesifikasi memberikan kebebasan yang cukup leluasa dalam pemilihan bahasa pemrograman untuk pengembangan serta platform apa yang akan digunakan dalam *delivery* program akhir (bisa berupa *mobile app*, *website*, *desktop app*, dll).

LAMPIRAN

Github repository: https://github.com/chaerla/Tucil3_13521044_13521079

REFERENSI

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Route-Planning-Bagian1-2021.pdf>

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Route-Planning-Bagian2-2021.pdf>

<https://www.youtube.com/watch?v=novFniWDQbQ>

Berikut adalah tabel keberhasilan tugas kecil ini :

1.	Program dapat menerima input graf.	✓
2.	Program dapat menghitung lintasan terpendek dengan UCS.	✓
3.	Program dapat menghitung lintasan terpendek dengan A*.	✓
4.	Program dapat menampilkan lintasan terpendek serta jaraknya.	✓
5.	Bonus: Program dapat menerima input peta dengan Google Map API dan menampilkan peta serta lintasan terpendek pada peta.	✓