

2020/2021 Second Semester

Course Code	DS630
Course Name	Artificial Intelligence for Data Science
CRN	24543
Assignment type	Critical Thinking
Module	02
Assignment Points	125

Student ID	G200007615
Student Name	Abdulaziz Alqumayzi

Critical Thinking Assignment 1

Introduction

In this activity, critical thinking assignment 1, we will construct a map of air and land distances with direct routes between the cities listed in table 1. we will create two graphs (maps) that connect the cities to each other, one to measure travel time between routes in land, the other one to measure travel time between cities in the air with the help of many tools that we will explain in this critical thinking. Then apply Dijkstra's algorithm to find the shortest path from a source node to a destination node. The source will be always my city Riyadh and other cities will be the destinations.

Table 1

Routes between cities

From	To
Dammam	Riyadh
Riyadh	Makkah
Makkah	Jeddah
Jeddah	Madinah
Al Bahah	Abha
Abha	Najran
Hayil	Buraydah
Buraydah	Riyadh
Tabuk	Sakaka
Sakaka	Al Ar Ar
Sakaka	Hayil
Hayil	Madinah
Makkah	Jizan
Jeddah	Jizan
Jizan	Najran
Dammam	Al Ar Ar
Jizan	Abha
Hayil	Riyadh
Dammam	Hafoof
Hafoof	Riyadh
Jeddah	Yanbu
Madinah	Yanbu

Goal of this Task

Our task is to apply the Dijkstra's algorithm to find the shortest distance from my home city Riyadh to all other cities. The travel time in the air and land from Riyadh to all cities. Lastly, the path that taken from Riyadh to the destination city.

Preparation

To accomplish this task, we will use many tools and external data. First, collecting the longitude and latitude of the cities from this website (distance.to) shown in figure 1.

Figure 1

longitude and latitude of Riyadh city

#1 Riyadh, SAU

Riyadh Region, Saudi Arabia

منطقة الرياض، السعودية

Latitude: 24.68773 24° 41' 15.828" N

Longitude: 46.72185 46° 43' 18.660" E

Secondly, calculating the distance travel time between cities in the air and land. Land travel time was calculated from the same website (distance.to) that shown in figure 2 and the air travel time was calculated from the website (flighttime-calculator.com) shown in figure 3.

Figure 2

travel time in the land between two Riyadh and Dammam (4h 43min)

#1 Riyadh, SAU 24.687730,46.721850

منطقة الرياض، السعودية

Riyadh Region, Saudi Arabia

Air line: 242.93 mi (390.96 km)

Driving route: 254.49 mi (409.57 km) (4h 43min)

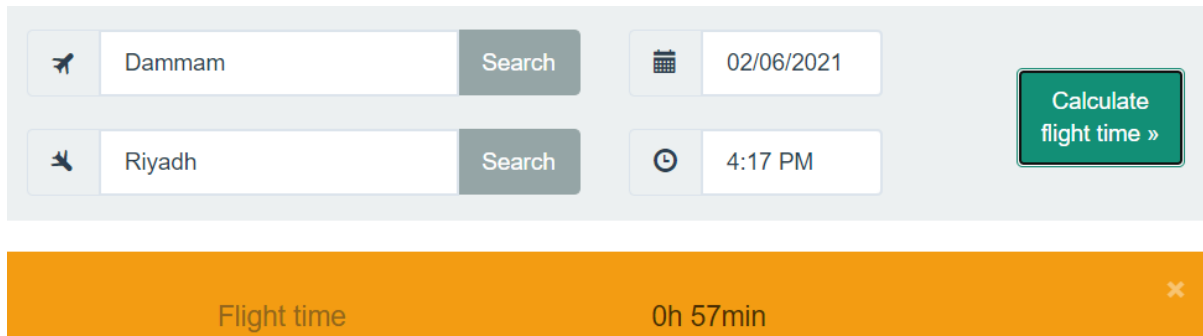
#2 Dammam, SAU 26.434420,50.103260

المنطقة الشرقية، السعودية

Eastern Region, Saudi Arabia

Figure 3

travel time in the air between two Dammam and Riyadh (0h 57min)



Jupyter notebook IDE to create the graph and apply Dijkstra's algorithm using the following packages:

- 1- Networkx: to create nodes, edges and positions.
- 2- Matplotlib: to make visualization.
- 3- Datetime: used to convert integers to time.

Implantation

First thing is importing the packages in Jupyter notebook as shown in figure 4.

Figure 4

importing necessary packages

```
import networkx as nx
import matplotlib.pyplot as plt
import datetime as dt
```

Then, creating an empty structure for the land called land_G using Graph() function shown in figure 5.

Figure 4

create an empty structure using Graph() function

```
land_G = nx.Graph()
```

Next, creating nodes, edges and travel time between cities using the function add_edge(). The function takes two main arguments, and one argument is optional as shown in

figure 5. This function will create the two nodes and the edge between them. Plus create an optional argument that can be used for weights of edges. w argument stands for weight and used in the code to be the travel time. The value inside it refers to the time between the two nodes, the number before the point is the hours; and the number after the point are the minutes.

Figure 5

add_edge() function to create nodes, edges and weight

```
land_G.add_edge('Dammam','Riyadh', w= 4.43)
```

Next is the location of cities, the location of cities assigned the longitude and latitude to all cities using a dictionary data type as shown in figure 6. The first number is longitude, and the second number is latitude. It must be like that because drawing functions that we will see later read longitude first.

Figure 6

A dictionary called pos to assign longitude and latitude for cities

```
pos= {'Riyadh':(46.72185, 24.68773), 'Dammam':(50.10326, 26.43442), 'Makkah':(39.82563, 21.42664),
      'Jeddah':(39.19797, 21.54238), 'Madinah':(39.61417, 24.46861), 'Al Bahah':(41.46767, 20.01288),
      'Abha':(42.50528, 18.21639), 'Najran':(44.12766, 17.49326), 'Hayil':(41.690731, 27.521879),
      'Buraydah':(43.97497, 26.32599), 'Tabuk':(36.57151, 28.3998), 'Sakaka':(40.20641, 29.96974),
      'Al Ar Ar':(41.03808, 30.97531), 'Jizan':(42.55111, 16.88917), 'Hafoof':(49.56532, 25.36457),
      'Yanbu':(38.0618, 24.08954)}
```

Next, drawing the map using draw() and draw_network_edge_labels() functions to draw the nodes and edges as shown in figure 7. Both functions use the structure that we created before with the positions (locations) of the dictionary. The map that constructed is shown in figure 8.

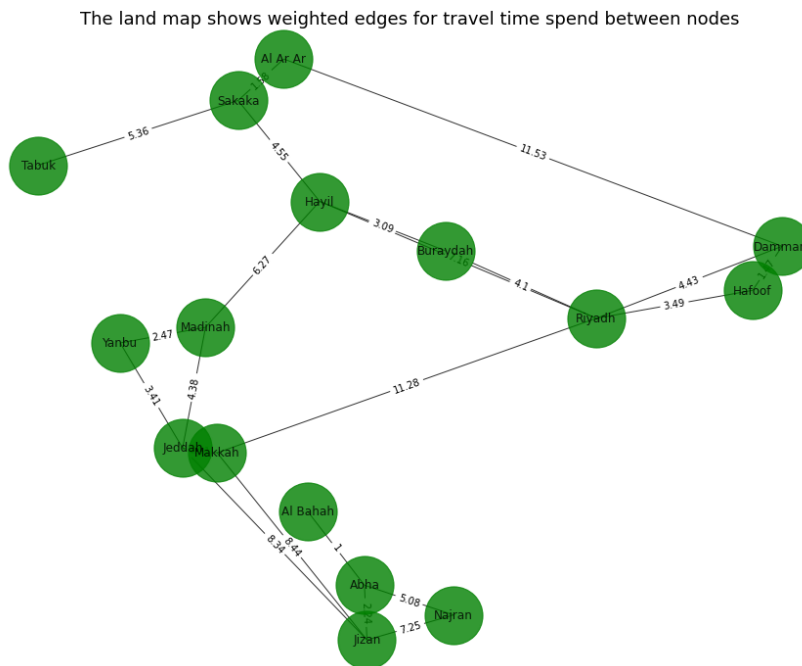
Figure 7

Functions to draw the map

```
plt.figure(figsize=(15,12))
plt.title('The land map shows weighted edges for travel time spend between nodes',fontsize=18);
nx.draw(land_G,pos, with_labels=True, node_size=3500,alpha=0.8, node_color='g')
nx.draw_networkx_edge_labels(land_G,pos, edge_labels=nx.get_edge_attributes(land_G,'w'));
```

Figure 8

map of the land graph



Finally, applying Dijkstra's algorithm using two functions. The first function to find the weight which is the travel time between cities as shown in figure 9 in the first line and the code is in figure 10 in the first print function. With the use of timedelta function to convert the integer into time data. The second function is to find the path that taken from Riyadh to the destination city as shown in figure 9 in the second line and the code is in figure 10 in the second print function.

Figure 9

results of Dijkstra's algorithm

The travel time from Riyadh to Al Bahah is 22:57:36 hours
The shortest way from the cities to reach Al Bahah is ['Riyadh', 'Makkah', 'Jizan', 'Abha', 'Al Bahah']

Figure 10

Code of Dijkstra's algorithm

```

print('The travel time from Riyadh to Al Bahah is',
      dt.timedelta(hours=nx.dijkstra_path_length(land_G, 'Riyadh', 'Al Bahah', weight='w')), 'hours')
print('The shortest way from the cities to reach Al Bahah is', nx.dijkstra_path(land_G, 'Riyadh', 'Al Bahah'), '\n')

```

The construction of the air map and travel time had the same procedure as the previous process of the constructed land map and travel time.

References:

Russell, S. J., & Norvig, P. (2021). *Artificial intelligence: A modern approach*. Hoboken: Pearson.

Distance calculator - calculate the distance online! (n.d.). Retrieved February 06, 2021, from <https://www.distance.to/>

Shortest paths¶. (2020, August 22). Retrieved February 06, 2021, from https://networkx.org/documentation/stable//reference/algorithms/shortest_paths.html

Flighttime-calculator.com. (n.d.). Retrieved February 06, 2021, from <https://flighttime-calculator.com/>

Create nodes/edges from Csv (latitude and longitude) for graphs. Retrieved February 06, 2021, from <https://datascience.stackexchange.com/questions/61248/create-nodes-edges-from-csv-latitude-and-longitude-for-graphs>

Datetime - basic date and time types¶. (n.d.). Retrieved February 06, 2021, from <https://docs.python.org/3/library/datetime.html>