Nama: Syahrul Akbar Ramdhani

Nim: 11230940000027

Mata Kuliah: Algoritma dan Struktur data

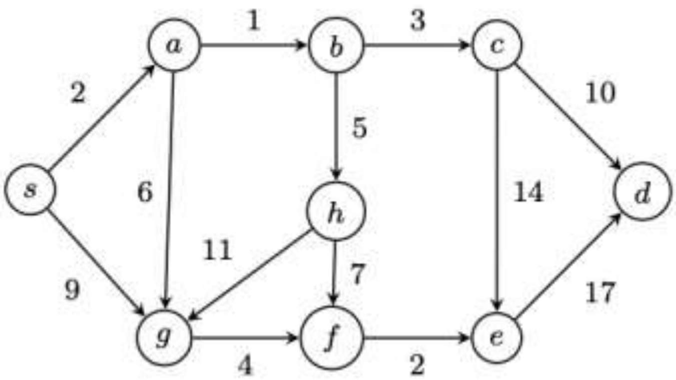
Soal 1

Use Dijkstra's algorithm to compute all shortest paths starting at node s. Show the values of the program variables B, R, U, p, D after each iteration of the main while-loop of Dijkstra's algorithm.

Jawaban

1. Menggunakan Algoritma Djikstra untuk menghitung jalur terpendek dari node s. dan menunjukkan nilai dari variabel-variabel program B, R, U, p, D setelah setiap iterasi dari perulangan utama dari algoritma Dijkstra.

- **B (Tree Nodes / Finalized)** : Semua node yang sudah "settled".
- **R (Boundary Nodes / Candidates)** : Node yang bertetangga langsung dengan B
- **U (Unknown Nodes / Unreached)** : Node yang belum bisa dijangkau sama sekali dari node sumber s
- **D (Distance Map)** : Menyimpan jarak minimum sementara dari s ke semua node.
- **p (Predecessor Map)** : Menyimpan node pendahulu dari setiap simpul di jalur terpendek.



Iterasi 0

B (Visited): ['s']
R (Queue): ['a', 'g']
U (Unvisited): ['a', 'g', 'b', 'c', 'h', 'd', 'e', 'f']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	∞	∞	∞	∞	∞	**9**	∞
Predecessor(p)	**-1**	**s**	**-1**	**-1**	**-1**	**-1**	**-1**	**s**	**-1**

Iterasi 1

B (Visited): [‘s’, ‘a’]
R (Queue): [‘b’, ‘g’, ‘g’]
U (Unvisited): [‘g’, 'b', 'c', 'h', 'd', 'e', 'f']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	∞	∞	∞	∞	**8**	∞
Predecessor(p)	**-1**	**s**	**a**	**-1**	**-1**	**-1**	**-1**	**a**	**-1**

Iterasi 2

B (Visited): [‘s’, ‘a’, ‘b’]
R (Queue): [‘c’, ‘h’, ‘g’, ‘g’]
U (Unvisited): [‘g’, 'c', 'h', 'd', 'e', 'f']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	**6**	∞	∞	∞	**8**	**8**
Predecessor(p)	**-1**	**s**	**a**	**b**	**-1**	**-1**	**-1**	**a**	**b**

Iterasi 3

B (Visited): [‘s’, ‘a’, ‘b’, ‘c’]
R (Queue): ['g', 'h', 'g', 'd', 'e']
U (Unvisited): [‘g’, 'h', 'd', 'e', 'f']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	**6**	**16**	**20**	∞	**8**	**8**
Predecessor(p)	**-1**	**s**	**a**	**b**	**c**	**c**	**-1**	**a**	**b**

Iterasi 4

B (Visited): [‘s’, ‘a’, ‘b’, ‘c’, ‘g’]
R (Queue): ['h', 'f', 'e', 'd']
U (Unvisited): ['h', 'd', 'e', 'f']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	**6**	**16**	**20**	**12**	**8**	**8**
Predecessor(p)	**-1**	**s**	**a**	**b**	**c**	**c**	**g**	**a**	**b**

Iterasi 5

B (Visited): [‘s’, ‘a’, ‘b’, ‘c’, ‘g’, ‘h’]
R (Queue): ['f', 'd', 'e']
U (Unvisited): ['d', 'e', 'f']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	**6**	**16**	**20**	**12**	**8**	**8**
Predecessor(p)	**-1**	**s**	**a**	**b**	**c**	**c**	**g**	**a**	**b**

Iterasi 6

B (Visited): [‘s’, ‘a’, ‘b’, ‘c’, ‘g’, ‘h’, ‘f’]
R (Queue): ['e’, 'e', ‘d’]
U (Unvisited): ['d', 'e']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	**6**	**16**	**14**	**12**	**8**	**8**
Predecessor(p)	**-1**	**s**	**a**	**b**	**c**	**f**	**g**	**a**	**b**

Iterasi 7

B (Visited): [‘s’, ‘a’, ‘b’, ‘c’, ‘g’, ‘h’, ‘f’, ‘e’]
R (Queue): [‘d’]
U (Unvisited): ['d']

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	**6**	**16**	**14**	**12**	**8**	**8**
Predecessor(p)	**-1**	**s**	**a**	**b**	**c**	**f**	**g**	**a**	**b**

Iterasi 8

B (Visited): [‘s’, ‘a’, ‘b’, ‘c’, ‘g’, ‘h’, ‘f’, ‘e’, ‘d’]
R (Queue): [ ]
U (Unvisited): [ ]

	s	a	b	c	d	e	f	g	h
Distance(D)	**0**	**2**	**3**	**6**	**16**	**14**	**12**	**8**	**8**
Predecessor(p)	**-1**	**s**	**a**	**b**	**c**	**f**	**g**	**a**	**b**

Soal 2

Suppose we define a different kind of graph where we have weights on the vertices and not the edges. Does the shortest-paths problem make sense for this kind of graph? If so, give a precise and formal description of the problem. If not, explain why not. Note we are not asking for an algorithm, just what the problem is or that it makes no sense.

Jawaban

Ya, permasalahan *shortest-path* (lintasan terpendek) tetap masuk akal meskipun bobot diberikan pada **simpul** alih-alih sisi. Secara formal, misalkan $G = (V, E)$ adalah sebuah graf, dan $w: V \rightarrow R_{\geq \mathbb{0}}$ adalah fungsi bobot yang memberikan nilai bobot tak negatif pada setiap simpul. Diberikan dua simpul $s, t \in V$, maka *vertex-weighted shortest path problem* adalah mencari lintasan ($P = (s = v_0, v_1, \dots, v_k = t)$) dari *s* ke *t* sedemikian sehingga total biaya

$$\sum_{i=0}^{k} w(v_i)$$

adalah minimum. Tergantung definisi yang digunakan, bobot simpul awal *s* dan simpul akhir *t* dapat dimasukkan atau dikecualikan dari jumlah tersebut. Permasalahan ini tetap bermakna karena kita masih dapat mendefinisikan total biaya lintasan berdasarkan jumlah bobot simpul yang dilalui.

Soal 3

A university campus has 6 main buildings connected by walkways. The distances between buildings are given in the table below (in meters):

From Building	To Building	Distance (m)
A	B	300
A	C	200
B	C	100
B	D	400
C	D	600
C	E	800
D	E	300
E	F	500
D	F	700

Tasks:

- Model the system as a weighted directed graph using the given building and distance data.
- Apply Dijkstra's algorithm to determine the shortest path from building A to building F.
- If the university decides to build a shuttle bus route only along the shortest path, list all the buildings that will be connected by the shuttle.

Jawaban

Model system graph berarah dan berbobot

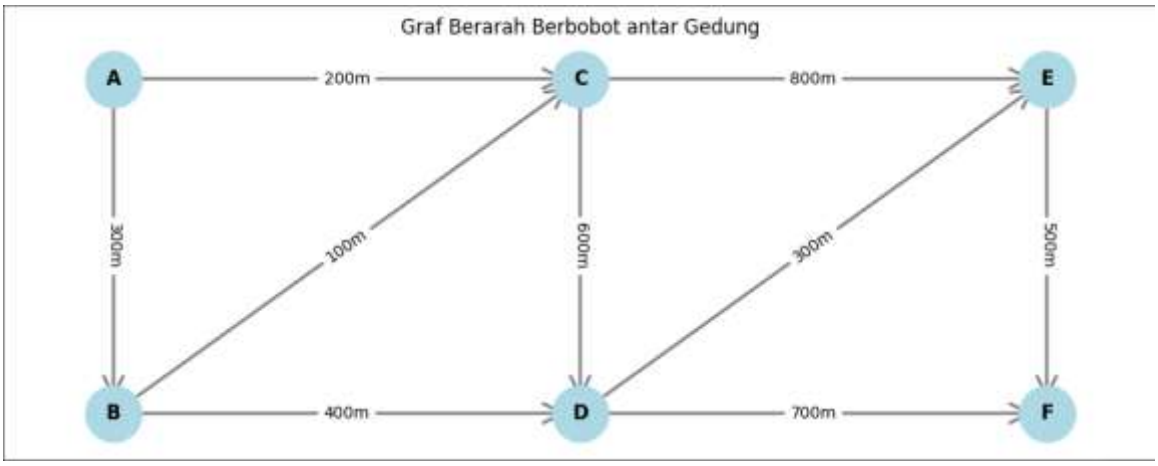
$$\text{Graph } G = (V, E)$$

Dengan:

- Himpunan simpul (Vertex)

$$V = \{A, B, C, D, E, F\}$$

- Himpunan sisi berarah dan berbobot
  E = {
  (A, B, 300), (A, C, 200),
  (B, C, 100), (B, D, 400),
  (C, D, 600), (C, E, 800),
  (D, E, 300), (D, F, 700),
  (E, F, 500)
  }



Penerapan Algoritma Djikstra untuk menentukan jalur terpendek dari Gedung A ke Gedung F

Iterasi 0

B (Visited): [‘A’]
R (Queue): [‘C’, ‘B’]
U (Unvisited): ['B', 'C', 'D', 'E', 'F’]

	A	B	C	D	E	F
Distance(D)	**0**	**300**	**200**	∞	∞	∞
Predecessor(p)	**-1**	**A**	**A**	**-1**	**-1**	**-1**

Iterasi 1

B (Visited): [‘A’, ‘C’]
R (Queue): [‘B’, ‘D’, ‘E’]
U (Unvisited): ['B', 'D', 'E', 'F’]

	A	B	C	D	E	F
Distance(D)	**0**	**300**	**200**	**800**	**1000**	∞
Predecessor(p)	**-1**	**A**	**A**	**C**	**C**	**-1**

Iterasi 2

B (Visited): [‘A’, ‘C’, ‘B’]
R (Queue): [‘D’, ‘E’, ‘D’]
U (Unvisited): ['D', 'E', 'F’]

	A	B	C	D	E	F
Distance(D)	**0**	**300**	**200**	**700**	**1000**	∞
Predecessor(p)	**-1**	**A**	**A**	**B**	**C**	**-1**

Iterasi 3

B (Visited): [‘A’, ‘C’, ‘B’, ‘D’]
R (Queue): [‘E’, ‘F’]
U (Unvisited): ['E', 'F’]

	A	B	C	D	E	F
Distance(D)	**0**	**300**	**200**	**700**	**1000**	**1400**
Predecessor(p)	**-1**	**A**	**A**	**B**	**C**	**D**

Iterasi 4

B (Visited): [‘A’, ‘C’, ‘B’, ‘D’, ‘E’]
R (Queue): [‘F’]
U (Unvisited): ['F’]

	A	B	C	D	E	F
Distance(D)	**0**	**300**	**200**	**700**	**1000**	**1400**
Predecessor(p)	**-1**	**A**	**A**	**B**	**C**	**D**

Iterasi 4

B (Visited): [‘A’, ‘C’, ‘B’, ‘D’, ‘E’, ‘F’]
R (Queue): [ ]
U (Unvisited): [ ]

	A	B	C	D	E	F
Distance(D)	**0**	**300**	**200**	**700**	**1000**	**1400**
Predecessor(p)	**-1**	**A**	**A**	**B**	**C**	**D**

Bisa dilihat visual untuk jalur terpendek yaitu $A \rightarrow B \rightarrow D \rightarrow F$

Dari table juga bisa, jika ditelusuri hasilnya $F \leftarrow D \leftarrow B \leftarrow A$, Jadi jalur terpendek dari $A \rightarrow F$ Adalah

$A \rightarrow B \rightarrow D \rightarrow F$ dengan total jarak 300 + 400 + 700 = 1400

Maka Berdasarkan jalur terpendek yang ditemukan oleh Dijkstra, maka bus antar-jemput akan melewati Gedung A, Gedung B, Gedung D dan Gedung F