VISUALISASI PENCARIAN LINTASAN TERPENDEK ALGORITMA FLOYD-WARSHALL DAN DIJKSTRA MENGGUNAKAN TEX

Imam Husni Al Amin[1], Veronica Lusiana[2], Budi Hartono[3]
[1,2,3]Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Stikubank
e-mail: [1]imam@edu.unisbank.ac.id, [2]vero@edu.unisbank.ac.id, [3]budihartono@edu.unisbank.ac.id

ABSTRAK

Dari beragam aplikasi graf, pencarian lintasan terpendek (shortest path) adalah salah satu yang memiliki terapan cukup banyak. Graf berbobot adalah graf yang setiap garis atau sisinya diberi sebuah harga (bobot). Bobot disini dapat menyatakan jarak antara dua buah kota, biaya perjalanan, waktu tempuh yang dibutuhkan, dan sebagainya. Penelitian ini akan melakukan visualisasi pencarian lintasan terpendek pada beberapa buah graph menggunakan algoritma Floyd-Warshall dan algoritma Dijkstra. Proses visualisasi lintasan terpendek menggunakan perintah atau kode sumber (cource code) dalam format Tex dengan perangkat lunak Texmaker versi 4.0.4 dan Beamer (document class) versi 3.24 untuk menghasilkan berkas presentasi dalam bentuk PDF. Melalui penelitian ini diharapkan dapat menghasilkan bahan ajar yang menarik untuk pokok bahasan pencarian lintasan terpendek yang diajarkan pada mata kuliah seperti Algoritma, Matematika Diskrit, dan Teori Graf.

***Kata Kunci**: lintasan terpendek (shortest path), algoritma Floyd-Warshall, algoritma Dijkstra*

1. PENDAHULUAN

Graf adalah suatu diagram yang memuat informasi tertentu jika diinterpretasikan secara tepat dapat menggambarkan berbagai macam struktur yang bertujuan untuk memvisualisasi obyek-obyek agar lebih mudah dimengerti. Graf memuat obyek titik dan obyek garis yang menghubungkan titik. Properti penting yang dimiliki oleh graf adalah arah dan bobot pada garis [1]. Garis dapat berarah atau tidak berarah. Pada garis tidak berarah digunakan untuk menyatakan hubungan antar obyek yang tidak mementingkan urutan. Graf berbobot adalah graf yang pada setiap garis diberi sebuah harga (bobot). Bobot dapat menyatakan jarak antara dua buah kota, biaya perjalanan, waktu tempuh yang dibutuhkan, dan sebagainya, menyesuaikan masalah yang dimodelkan oleh graf [2].

Penelitian ini akan melakukan visualisasi proses pencarian lintasan terpendek (Shortest Path). Jenis pencarian ini merupakan salah satu kasus yang memiliki banyak terapan dalam teori graf. Analisa terhadap proses pencarian lintasan terpendek menggunakan algoritma Floyd-Warshall dan Dijkstra untuk beberapa contoh kasus yang memiliki perbedaan tingkat kompleksitas. Visualisasi ditampilkan menggunakan perangkat lunak Texmaker 4.0.4. Berikut ini adalah beberapa penelitian sejenis yang pernah dilakukan.

Algoritma Floyd-Warshall merupakan salah satu varian dari pemrograman dinamis, yaitu suatu mode yang melakukan pemecahan masalah dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait. Pencarian suatu tempat merupakan salah satu permasalahan yang sering timbul pada setiap orang, dengan peta seseorang bisa melakukan pencarian tempat yang dituju namun terkadang banyak yang tidak tahu arah mana yang baik dan terdekat. Hal ini sama yang terjadi pada siswa-siswi SMA tingkat 3 yang berasal dari beberapa daerah mereka melakukan *survey* secara langsung untuk tiap falkutas, fasilitas dukungan mahasiswa, organisasi kegiatan mahasiswa, dan semua sarana prasarana yang ada di sekitar lingkungan ITB [3].

Algoritma Dijkstra digunakan untuk menentukan rute dan lokasi perpindahan koridor atau *transfer point* pada aplikasi digital untuk pencarian rute Bus Trans Semarang. Akan tetapi, calon penumpang menghadapi kesulitan untuk mendapatkan informasi rute bus yang akan diambil untuk menuju lokasi yang ditentukan, karena informasi yang tersedia masih berupa informasi statis berupa poster tempel. Dalam aplikasi ini, Google Maps API digunakan sebagai data spasial, sedangkan data non spasial berupa informasi detail *shelter* dan koridor. Calon penumpang Bus Trans Semarang dapat memanfaatkan aplikasi ini dengan memasukkan informasi lokasi yang ingin dituju untuk mendapatkan rute, kemudian aplikasi akan menampilkan peta rute dari titik awal menuju lokasi tujuan [4].

2. METODE PENELITIAN

Metode penelitian yang digunakan yaitu studi pustaka, menampilkan lintasan terpendek melalui contoh kasus beberapa buah graph menggunakan algoritma Floyd-Warshall dan algoritma Dijkstra. Visualisasi hasil penyelesaian pencarian lintasan terpendek dengan cara menyusun skrip menggunakan Texmaker versi 4.0.4, Beamer (*document class*) 3.24 untuk menghasilkan berkas PDF, dan Tikz (*package library*) 2.10.

2.1 Algoritma Floyd-Warshall

Cara kerja algoritma Floyd-Warshall yaitu dengan masukan berupa matriks hubung graf berarah berlabel dan keluaran berupa lintasan terpendek dari semua titik ke semua titik yang lain. Algoritma ini memulai iterasi

dari titik awalnya kemudian memperpanjang lintasan dengan mengevaluasi titik demi titik hingga mencapai titik tujuan dengan jumlah bobot yang seminim mungkin [5]. Contoh W_0 adalah matriks hubung graf berarah berlabel mula-mula. W*adalah matriks hubung minimal dengan w_{ij}* = lintasan terpendek dari titik V_i ke J_i. Algoritma ini mencari lintasan terpendek dengan cara sebagai berikut:

```
1. W=W0
2. Untuk k=1 hingga n, lakukanlah:
      Untuk i=1 hingga n, lakukanlah:
            Untuk j=1 hingga n, lakukanlah:
                  Jika W[i,j] > (W[i,k] + W[k,j]) maka tukarlah
                  W[i,j] dengan (W[i,k] + W[k,j])
3. W* = W
```

Dalam iterasinya, untuk mencari lintasan terpendek maka akan membentuk n matriks sesuai dengan iterasi-k. Hal ini menyebabkan kebutuhan waktu proses yang cukup lama terutama untuk n yang besar [1].

2.2 Algoritma Dijkstra

Algoritma ini mencari lintasan terpendek dalam sejumlah langkah menggunakan prinsip greedy. Prinsip ini pada algoritma Dijkstra menyatakan bahwa pada setiap pemilihan langkah kita memilih sisi yang berbobot minimum dan memasukkannya ke dalam himpunan solusi [2]. Misalkan G adalah graf berarah berlabel dengan titik-titik V(G) = {v_1,v_2, v_3, … v_n} dan lintasan terpendek yang dicari adalah dari v_1 ke v_n. Proses pencarian algoritma ini dimulai dari titik v_1. Dalam iterasinya akan mencari satu titik yang jumlah bobotnya dari titik 1 terkecil. Titik-titik yang terpilih dipisahkan dan titik-titik tersebut tidak akan diperhatikan lagi pada iterasi berikutnya [1][5][6]. Algoritma selengkapnya adalah sebagai berikut:

```
1. L = { }
   V = {v2,v3, v4, … vn}
2. Untuk I = 2, … n, lakukan D(i) = W(1,i)
3. Selama vn∉ L lakukan:
      a. Pilih titik vk ∈ V-L dengan D(k) terkecil
         L= L ∪{vk}
      b. Untuk setiap vj ∈V-L lakukan:
         Jika D(j) > D(k) + W(k,j) maka ganti D(j) dengan(D(k) + W(k,j)
4. Untuk setiap vj ∈ V, w * (1,j) = D (j)
```

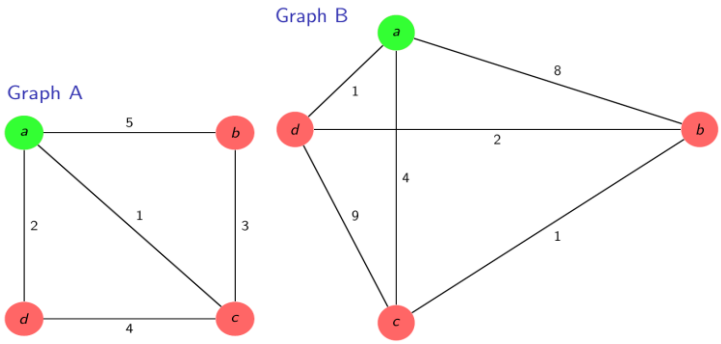
Lintasan terpendek dari titik v_1 ke v_n adalah melalui titik-titik dalam L secara v=berurutan, dengan jumlah bobot lintasan terkecilnya adalah D(n). Keterangan,

V(G) = {v_1,v_2, v_3, … v_n}
L = himpunan titik-titik ∈V(G) yang sudah terpilih dalam lintasan *path* terpendek
D(j) = jumlah bobot lintasan terkecil dari v_1 ke v_j
W(i,j) = bobot garis dari titik v_i ke titik v_j
W*(1,j) = jumlah bobot path terkecil dari v_1 ke v_j

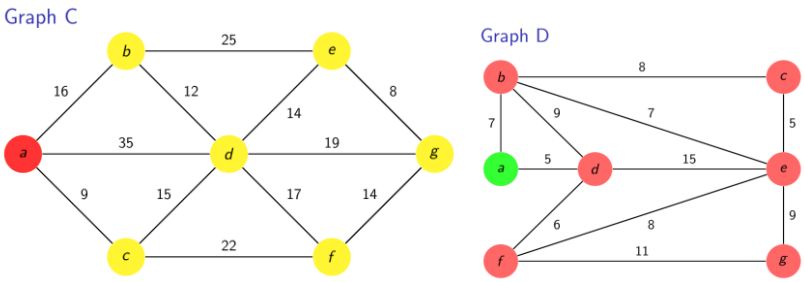
3. HASIL DAN PEMBAHASAN

3.1 Bahan Penelitian

Bahan penelitian menggunakan empat buah graf seperti dapat dilihat pada Gambar 1 dan Gambar 2. Pada Gambar 1, Graph A memiliki 4 buah titik dan 5 buah sisi atau garis, sedangkan Graph B memiliki 4 buah titik dan 6 buah garis. Pada Gambar 2, Graph C memiliki 7 buah titik dan 12 buah garis, sedangkan Graph D memiliki 7 buah titik dan 11 buah garis. Garis atau sisi keempat buah graph tersebut adalah tidak berarah.



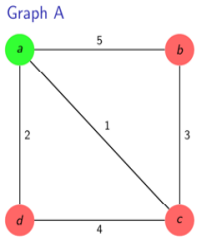
Gambar 1. Graf A dan graf B



Gambar 2. Graf C dan Graf D

Pada Tabel 1 adalah data seluruh lintasan terpendek yang dimiliki oleh Graph A. Terdapat enam buah kombinasi lintasan terpendek. Tabel 2 berisi data matriks koneksi antar titik (adjacency matrix) dan matriks jarak pada graph A. Disini untuk menghubungkan antara titik b dan d dapat dilakukan melalui titik a atau titik c, karena titik b dan d tidak memiliki garis yang menghubungkan keduanya secara langsung.

Tabel 1. Seluruh lintasan terpendek pada graph A



Nomor	Verteks asal	Verteks tujuan	Jarak	Lintasan terpendek
1	a	b	4	a - c - b
2	a	c	1	a - c
3	a	d	2	a - d
4	b	c	3	b - c
5	b	d	6	b - c - a - d
6	c	d	3	c - a - d

Tabel 2. Matriks koneksi antar titik dan matriks jarak pada graph A

Matriks koneksi antar titik (Adjacency matrix)

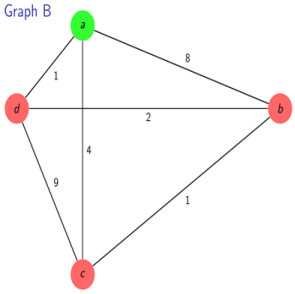
	a	b	c	d
a	-	5	1	2
b	5	-	3	-
c	1	3	-	4
d	2	-	4	-

Matriks Jarak

	a	b	c	d
a	-	4	1	2
b	4	-	3	6
c	1	3	-	3
d	2	6	3	-

Pada Tabel 3 adalah data seluruh lintasan terpendek yang dimiliki oleh Graph B. Terdapat tujuh buah kombinasi lintasan terpendek. Pada titik asal a menuju titik tujuan c terdapat dua buah pilihan lintasan terpendek yang memiliki jarak sama yaitu 4. Tabel 4 berisi data matriks koneksi antar titik (adjacency matrix) dan matriks jarak pada graph B. Disini seluruh titik (a, b, c, dan d) dapat terhubung secara langsung ke titik yang lain, karena terdapat garis yang menghubungkan secara langsung untuk setiap titik menuju seluruh titik yang lainnya.

Tabel 3. Seluruh lintasan terpendek pada graph B



Nomor	Verteks asal	Verteks tujuan	Jarak	Lintasan terpendek
1	a	b	3	a - d - b
2	a	c	4	a - c a - d - b - c
3	a	d	1	a - d
4	b	c	1	b - c
5	b	d	2	b - d
6	c	d	3	c - b - d

Tabel 4. Matriks koneksi antar titik dan matriks jarak pada graph B

Matriks koneksi antar titik (Adjacency matrix)

	a	b	c	d
a	-	8	4	1
b	8	-	1	2
c	4	1	-	9
d	1	2	9	-

Matriks Jarak

	a	b	c	d
a	-	3	4	1
b	3	-	1	2
c	4	1	-	3
d	1	2	3	-

3.2 Visualisasi Lintasan

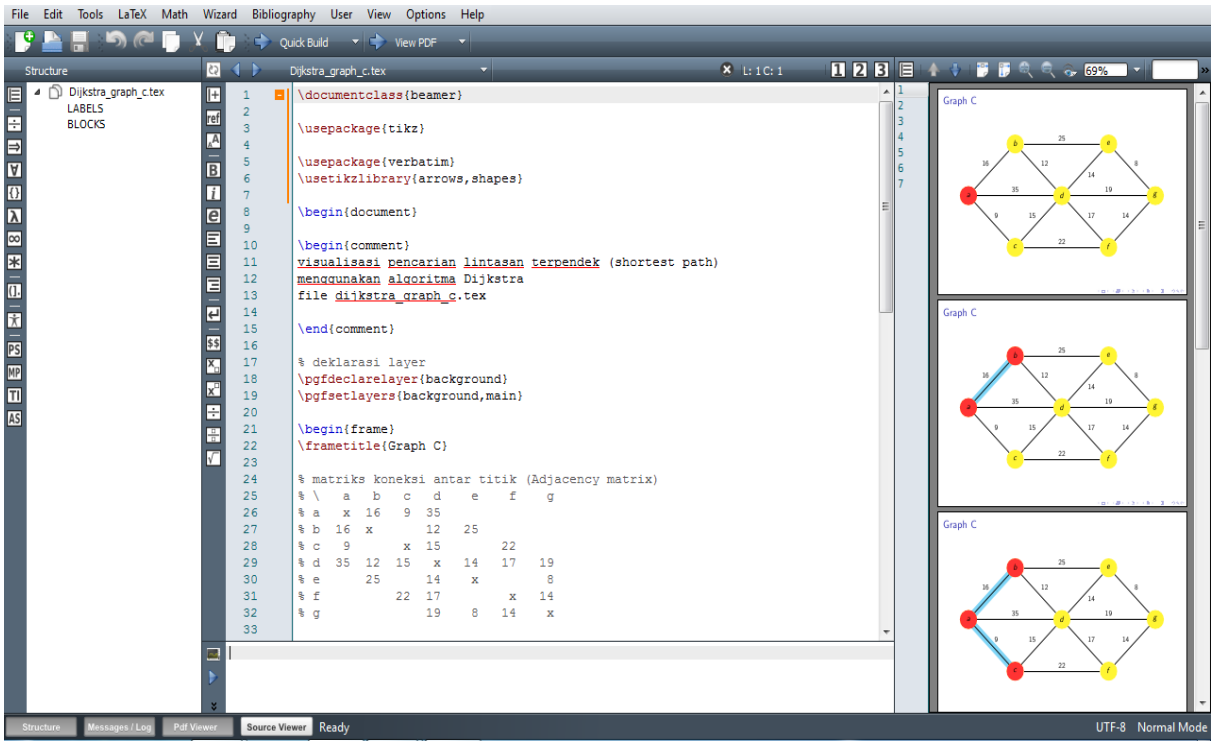
Secara umum, penulisan perintah pada Beamer dan Tikz untuk proses visualisasi lintasan terpendek adalah sebagai berikut:

a. Menentukan posisi relatif antara satu titik dengan titik lainnya. Format penulisan perintah untuk menggambar sebuah titik yaitu {(posisi_kolom,posisi_baris)/nama_titik}.
   Posisi kolom paling kiri diberi nilai 0 dan akan bertambah 1 untuk kolom yang disebelah kanannya. Pada posisi baris, maka baris tengah bernilai 0, akan bertambah 1 untuk baris diatasnya dan berkurang 1 untuk baris dibawahnya.
b. Menghubungkan titik dengan garis dan menuliskan bobot setiap garis. Format penulisan perintah menggambar sebuah garis atau sisi adalah {titik_pertama/titik_kedua/bobot}.
   Pada graf tidak berarah maka penulisan titik pertama dan titik kedua bisa dipertukarkan.
c. Menggambar urutan titik yang membentuk pohon rentang minimum. Format penulisan perintahnya adalah {nama_titik/nomor_slide}.
d. Menggambar urutan garis yang membentuk pohon rentang minimum. Format penulisan perintahnya adalah {titik_pertama/titik_kedua}.

Pada Gambar 3 dapat dilihat tampilan antar muka TexMaker yang digunakan untuk menulis skrip perintah atau kode sumber (*cource code*) dalam format Tex. TexMaker memiliki fasilitas untuk melihat hasil (*output*) melalui *quick build* dan *view* PDF, dan memberitahu seandainya terjadi kesalahan (*error* atau *warning*) apabila penulisan perintah masih belum benar. Penulisan perintah selengkapnya dapat dilihat pada contoh berkas Tex berikut ini.

```
\documentclass{beamer}
\usepackage{tikz}
\usepackage{verbatim}
\usetikzlibrary{arrows,shapes}
\begin{document}
\begin{comment}
visualisasi pencarian lintasan terpendek (shortest path) menggunakan
algoritma Dijkstra
file dijkstra_graph_c.tex
pustaka:
http://www.cs.uiuc.edu/~jeffe/teaching/algorithms
http://www.texample.net/tikz/examples/
\end{comment}
% deklarasi layer
\pgfdeclarelayer{background}
\pgfsetlayers{background,main}
\begin{frame}
\frametitle{Graph C}
% menentukan atribut titik (vertex) dan garis (edge)
% menentukan diameter-warna titik, ketebalan-warna garis
\tikzstyle{vertex}=[circle,fill=yellow!80,minimum size=25pt,inner sep=0pt]
\tikzstyle{selected vertex} = [vertex, fill=red!80]
\tikzstyle{edge} = [draw,thick,-]
\tikzstyle{weight} = [font=\small]
\tikzstyle{selected edge} = [draw,line width=10pt,-,cyan!50]
```

```
\tikzstyle{ignored edge} = [draw,line width=4pt,-,black!10]
\begin{figure}
\begin{tikzpicture}[scale=2.4, auto,swap]
% menggambar graph berisi 7 titik dan 12 garis
% menentukan posisi relatif satu titik dengan titik lainnya
\foreach  \pos/\name  in  {{(0,0)/a},  {(1,1)/b},  {(1,-1)/c},  {(2,0)/d},
{(3,1)/e}, {(3,-1)/f}, {(4,0)/g}}
\node[vertex] (\name) at \pos {$\name$};
% menghubungkan titik-garis, menuliskan bobot tiap garis
\foreach \source/ \dest /\weight in {b/a/16, e/b/25, g/e/8, g/f/14, g/d/19,
f/d/17, f/c/22, c/a/9, d/c/15, d/a/35, d/b/12, d/e/14}
\path[edge] (\source) -- node[weight] {$\weight$} (\dest);
% pembentukan lintasan terpendek
% menggambar urutan titik
\foreach \vertex / \fr in {a/1, b/2, c/3, d/4, e/5, f/6, g/7}
\path<\fr-> node[selected vertex] at (\vertex) {$\vertex$};
\begin{pgfonlayer}{background}
% menggambar urutan garis
\pause
\foreach \source / \dest in {a/b, a/c, c/d, d/e, c/f, d/g}
\path<+->[selected edge] (\source.center) -- (\dest.center);
% menentukan garis yang boleh dihilangkan
\foreach \source / \dest / \fr in {b/b/5}
\path<\fr->[ignored edge] (\source.center) -- (\dest.center);
\end{pgfonlayer}
\end{tikzpicture}
\end{figure}
\end{frame}
\end{document}
```



Gambar 3 Tampilan antar muka TexMaker

Hasil visualisasi graf C dapat dilihat pada Gambar 4 dan visualisasi graf D dapat dilihat pada gambar 5. Lintasan terpendek graph C dengan titik awal a menuju titik b, c, d, e, f, dan g yaitu masing-masing diperoleh jarak 16, 9, 24, 38, 31, dan 43. Lintasan terpendek graph D dengan titik awal a menuju titik b, c, d, e, f, dan g yaitu masing-masing diperoleh jarak 7, 15, 5, 14, 11, dan 22. Matriks koneksi antar titik untuk graph c dapat dilihat pada Tabel 5. Sedangkan matriks koneksi antar titik untuk graph d dapat dilihat pada Tabel 6.

Tabel 5. Matriks koneksi antar titik pada graph C

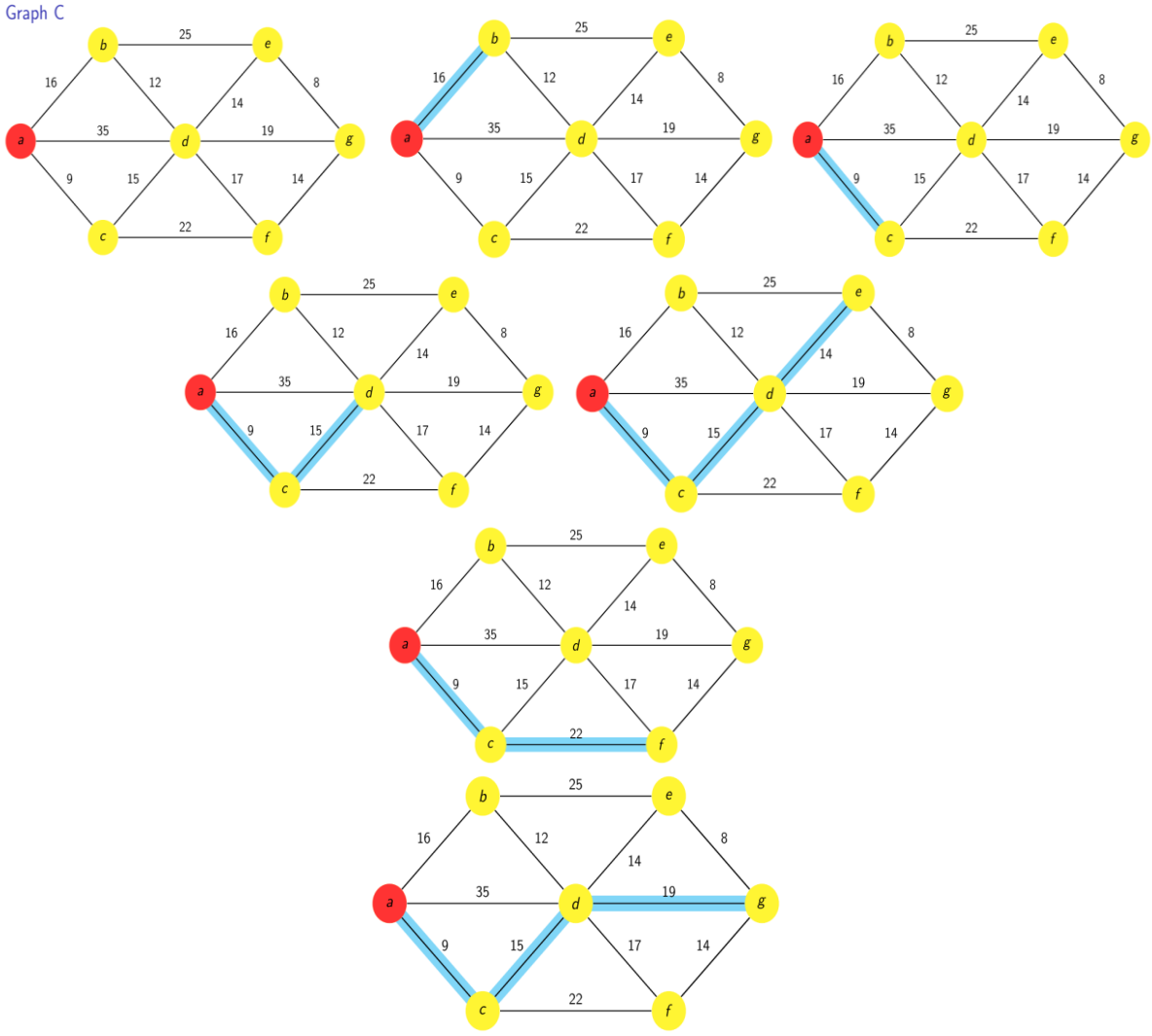
Matriks koneksi antar titik (Adjacency matrix)

	a	b	c	d	e	f	g
a	-	16	9	35	-	-	-
b	16	-	-	12	25	-	-
c	9	-	-	15	-	22	-
d	35	12	15	-	14	17	19
e	-	25	-	14	-	-	8
f	-	-	22	17	-	-	14
g	-	-	-	19	8	14	-

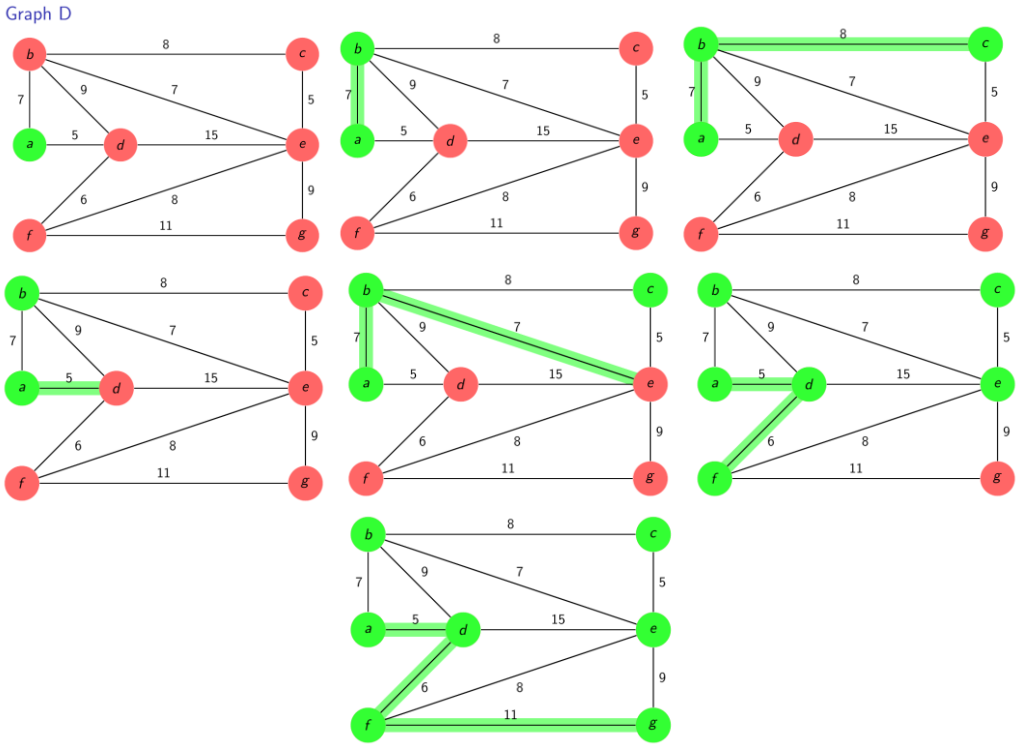
Tabel 6. Matriks koneksi antar titik pada graph D

Matriks koneksi antar titik (Adjacency matrix)

	a	b	c	d	e	f	g
a	-	7	-	5	-	-	-
b	7	-	8	9	7	-	-
c	-	8	-	.-	5	-	-
d	5	9	-	-	15	6	-
e	-	7	5	15	-	8	9
f	-	-	-	6	8	-	11
g	-	-	-	-	9	11	-



Gambar 4. Lintasan graph C dengan titik awal a



Gambar 5. Lintasan graph D dengan titik awal a

4. KESIMPULAN

Proses visualisasi pencarian lintasan terpendek menggunakan algoritma Floyd-Warshall dan algoritma Dijkstra dapat disusun menggunakan perangkat lunak pengolah dokumen LaTex. Berkas hasil visualisasi disimpan menggunakan format PDF yang dapat digunakan sebagai modul ajar yang menarik. Untuk melengkapi apa yang telah diperoleh dari penelitian ini maka dapat ditambahkan algoritma lain yang bisa digunakan untuk mencari lintasan terpendek pada sebuah graph. Dapat dipertimbangkan pemilihan perangkat lunak yang mampu membuat visualisasi dengan adanya unsur animasi.

UCAPAN TERIMA KASIH

Tim Peneliti mengucapkan terima kasih kepada Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) Universitas Stikubank (Unisbank) untuk dukungannya terhadap penelitian ini.

DAFTAR PUSTAKA

[1] Siang, J.J., 2009, *Matematika Diskrit dan Aplikasinya pada Ilmu Komputer edisi keempat*, Penerbit Andi, Yogyakarta.
[2] Munir, R., 2010, *Matematika Diskrit edisi ketiga*, Informatika, Bandung.
[3] Anggoro, A.A., 2015, Pencarian Titik Lokasi dengan Pemanfaatan Algoritma Floyd-Warshall Sebagai Perhitungan Jarak Terdekat di Institut Teknologi Bandung, *Jurnal LPKIA*, Vol.1, No.1, Hal. 1-5, Januari 2015.
[4] Ardana, D., Saputra, R., 2016, Penerapan Algoritma Dijkstra pada Aplikasi Pencarian Rute Bus Trans Semarang, *Prosiding Seminar Nasional Ilmu Komputer (SNIK)*, Hal. 299-306, Semarang, 10 Oktober 2016.
[5] Sedgewick, R., and Wayne, K., 2011, *Algorithms 4th edition*, Adisson-Wesley USA.
[6] Erickson, J., 2013, *Algorithms*, Department of Computer Science University of Illinois Urbana-Champaign, USA.