PENYELESAIAN MASALAH LINTASAN TERPENDEK *FUZZY* DENGAN MENGGUNAKAN ALGORITMA CHUANG – KUNG DAN ALGORITMA FLOYD

[1]Anik Musfiroh, [2]Lucia Ratnasari, [3]Siti Khabibah

[1.2.3] Jurusan Matematika Universitas Diponegoro
Jl. Prof> Soedharto, SH, Tembalang Semarang 54275
Email : [1] musfiroh.anik@gmail.com, [2] lratnasari@yahoo.co.id

Abstract : *In the classic graph, shortest path problem is related problem with determination of the which are connected in a graph that form the shortest distance between source node and the destination node. This idea is extended to solve the fuzzy shortest path problem. In this paper, will be discussed about Chuang - Kung algorithm and Floyd algorithm to solve shortest path problem. Chuang - Kung algorithm's steps is to determine all pass possible path from source node to destination node, then compute value of similarity degree* $S(L_{min}, L_i)$ *with* L_{min} *is the fuzzy shortest length and* L_i *is length of possible path. While for Floyd algorithm, the first step is to determine the initial distance matrix* (D_0) *and the matrix of the initial order* (S_0)*, then check this elements. If in matrix* D_k *element is* $d_{ik} + d_{kj} < d_{ij}$ *so* d_{ij} *replaced with* $L_{min}\left(d_{ij}, d_{ik} + d_{kj}\right)$*. Then replace elements matrix* S_k *with* k*. Changes* s_{ij} *in matrix* S_k *following to changes* d_{ij} *on matrix* D_k*.*

Keywords : *Fuzzy* shortest path, Chuang – Kung algorithm, Floyd algorithm.

1. Pendahuluan

Masalah lintasan terpendek berkaitan dengan pencarian lintasan pada graf berbobot yang menghubungkan dua buah titik sedemikian hingga jumlah bobot sisi – sisi yang terpilih merupakan bobot minimum. Salah satu manfaat penggunaan graf adalah untuk memudahkan gambaran ilustrasi dan perhitungan dalam menyelesaikan masalah lintasan terpendek. Graf dapat didefinisikan sebagai kumpulan titik yang dihubungkan dengan garis. Graf biasanya digunakan untuk memodelkan obyek-obyek diskrit dan hubungan antar obyek-obyek tersebut. Ide dasar menentukan masalah lintasan terpendek pada graf klasik diperluas untuk penyelesaian masalah lintasan terpendek *fuzzy.*

Algoritma yang digunakan untuk menyelesaikan masalah lintasan terpendek *fuzzy* adalah dengan menggunakan algoritma Chuang – Kung[8] dan algoritma Floyd[7]. Langkah – langkah algoritma Chuang – Kung adalah dengan menentukan lintasan yang mungkin dilalui dari titik sumber ke titik tujuan dan menghitung panjang lintasan tersebut. Kemudian menentukan panjang terpendek *fuzzy* (L_{min}) dengan menggunakan metode panjang terpendek *fuzzy* serta menghitung ukuran kesamaannya untuk menghasilkan derajat kesamaan $S(L_{min}, L_i)$ antara L_{min} dan L_i. Lintasan terpendek pada algoritma Chuang – Kung adalah nilai tertinggi dari derjat kesamaan tersebut. Sedangkan untuk langkah - langkah algoritma Floyd adalah dengan menentukan matriks jarak awal (D_0) dan matriks urutan awal (S_0) kemudian memeriksa elemen – elemennya. Jika $d_{ik} + d_{kj} < d_{ij}$ maka d_{ij} pada matrik D_k diubah menjadi $L_{min}\left(d_{ij}, d_{ik} + d_{kj}\right)$. Elemen s_{ij} pada matriks S_k diubah mengikuti perubahan d_{ij} pada matrik D_k.

2. Dasar Teori

Definisi 2.1 [2]
Diberikan A dan B merupakan himpunan tidak kosong. *Cartesian product* dari dua buah himpunan A dan B dinotasikan dengan $A \times B$ adalah himpunan semua pasangan berurutan (a, b) dengan $a \in A$ dan $b \in B$. Secara matematis dituliskan $A \times B = \{(a, b) | a \in A, b \in B\}$.

Definisi 2.2 [6]
Fungsi karakteristik dari suatu himpunan tegas A dalam semesta X dapat dinyatakan dengan fungsi karakteristik $\chi_A : X \to \{0,1\}$ yang didefinisikan dengan aturan :

$$\chi_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$$

Dari definisi diatas, $\chi_A(x) = 1$ artinya derajat keanggotaan x pada A adalah 1 dan $\chi_A(x) = 0$ artinya derajat keanggotaan x pada A adalah 0.

Definisi 2.3
Fungsi keanggotaan dari suatu himpunan *fuzzy* A dalam semesta X adalah pemetaan μ_A dari X ke selang $[0, 1]$, yaitu $\mu_A: X \to [0, 1]$. Nilai fungsi $\mu_A(x)$ menyatakan derajat keanggotaan unsur $x \in X$.

Definisi 2.4 [6]
Komplemen dari suatu himpunan *fuzzy* A dalam semesta X adalah himpunan *fuzzy* $\bar{A}$, dengan fungsi keanggotaan :

$$\bar{A}(x) = 1 - A(x) \qquad \forall x \in X.$$

Definisi 2.5 [6]
Irisan dari himpunan *fuzzy* A dan B dengan notasi $A \cap B$, dengan fungsi keanggotaan :

$$(A \cap B)(x) = \inf\left[\left(A(x), B(x)\right)\right] \forall x \in X$$

Definisi 2.6 [6]
Gabungan dua buah himpunan *fuzzy* A dan B dengan notasi $A \cup B$, dengan fungsi keanggotaan :

$$(A \cup B)(x) = sup\ \left[\left(A(x), B(x)\right)\right] \forall x \in X.$$

Definisi 2.7 [4]
Penjumlahan dua buah himpunan *fuzzy* A dan B adalah himpunan *fuzzy* $A + B$, yang didefinisikan dengan fungsi keanggotaan :

$$(A + B)(z) = sup_{z=x+y}\ \min\left[\left(A(x), B(y)\right)\right],\ \forall x, y \in X \text{ dengan } z = x + y.$$

Definisi 2.8 [9]
Suatu graf G terdiri atas dua himpunan yaitu himpunan tidak kosong $V(G)$ yang elemen-elemennya disebut titik (*vertex*) dan himpunan (boleh kosong) $E(G)$ yang elemen-elemennya disebut sisi (*edge*) sedemikian sehingga setiap elemen e dalam $E(G)$ merupakan pasangan tak berurutan dari titik-titik di $V(G)$. Graf G yang didefinisikan sebagai pasangan himpunan (V, E) dapat ditulis dengan notasi $G = (V, E)$.

Definisi 2.9 [3]
Graf berbobot adalah graf yang setiap sisi diberikan sebuah harga (bobot). Bobot pada setiap sisinya dapat menyatakan jarak antara dua buah kota, biaya perjalanan, waktu tempuh, ongkos produksi, dan sebagainya.

Definisi 2.10 [9]
Suatu jalan (*walk*) yang panjangnya k dalam graf G_2 adalah urutan k sisi yang berbentuk

$$\underbrace{uv_1, v_1v_2, v_2v_3, \ldots, v_{k-1}v}_{k}$$

Walk ini dinotasikan dengan $uv_1v_2v_3\ldots v_{k-1}v$, dan disebut *walk* antara u dan v. Semua sisi dan titik dalam *walk* tidak perlu berbeda (boleh sama).

Definisi 2.11 [9]
Jika semua sisi (tetapi tidak perlu semua titik) dalam *walk* tersebut berbeda, maka *walk* itu disebut jejak (*trail)*. Jika semua titiknya juga berbeda, maka *trail* itu disebut lintasan *(path)*. Panjang lintasan adalah banyak sisi dalam lintasan tersebut.

Definisi 2.12 [9]
Suatu *walk* tertutup dalam graf G adalah *walk* yang titik sumbernya sama dengan titik akhirnya. Jika semua sisi berbeda, maka *walk* itu disebut *trail* tertutup (*closed trail*). Jika titik – titiknya juga berbeda kecuali pada titik sumber dan titik akhirnya maka *trail* itu disebut sikel (*cycle*).

Definisi 2.13 [9]
Jarak *(distance)* dari titik v ke titik u pada graf G, yang dinotasikan dengan $d(v,u)$ adalah panjang lintasan terpendek dari v ke u.

Definisi 2.14 [3]
Suatu graf berarah (*directed graph* atau digraf) adalah graf yang setiap sisinya memiliki orientasi arah. Pada graf berarah $(v_j, v_k) \neq (v_k, v_j)$.

Definisi 2.15 [10]
Dimisalkan V adalah himpunan titik, suatu graf *fuzzy* yang dinotasikan dengan $G: (\sigma, \mu)$ adalah sepasang fungsi dimana :

i. $\sigma: V \to [0,1]$
ii. $\mu: V \times V \to [0,1]$

Sedemikian hingga $\mu(xy) \leq \sigma(x) \wedge \sigma(y), \forall x, y \in V$, dengan $\sigma(x) \wedge \sigma(y)$ adalah $min\left(\sigma(x), \sigma(y)\right)$, dimana σ derajat keanggotaan titik dan μ adalah derajat keanggotaan sisi.

Algoritma Djikstra

Algoritma ini mencari panjang lintasan (*path)* terpendek dari suatu titik sumber ke titik yang lain dalam digraf berbobot.
Langkah – langkah algoritma :
Diberikan label permanen untuk titik sumber : $[0, -]$. Diambil $i = 1$.

1. Didefinisikan label sementara $[u_i + d_{ij}, i]$ untuk setiap titik j yang diperoleh langsung dari i dan belum mempunyai label permanen.
2. Dipilih titik j^* yang mempunyai $[u_i + d_{ij}, i]$ terpendek, beri label permanen.
3. Jika semua mempunyai label permanen, iterasi berhenti. Jika tidak, ambil $i = j^*$, kembali ke langkah 1.
   Dengan u_i = lintasan terpendek dari titik sumber ke titik i, d_{ij} = panjang busur (i,j), $[u_j, i]$ = lintasan terpendek dari titik sumber ke titik j melalui titik i.

Algoritma Floyd

Algoritma Floyd adalah algoritma yang digunakan untuk menentukan lintasan terpendek antara setiap pasang titik dalam graf.

Langkah – langkah algoritma :
Didefinisikan matriks D_0 dan S_0.
Jika i dan j tidak terhubung langsung, didefinisikan $d_{ij} = \infty$.

1. Diambil $k = 1$
   Didefinisikan baris k dan kolom k sebagai baris dan kolom pivot.
2. Untuk setiap d_{ij} dalam D_{k-1}, jika $d_{ik} + d_{kj} < d_{ij}$, $(i \neq k, j \neq k,$ dan $i \neq j)$:
   - D_k diubah dengan mengganti d_{ij} dalam D_{k-1} dengan $d_{ik} + d_{kj}$.
   - S_k diubah dengan mengganti s_{ij} dalam S_{k-1} dengan k.

Jika $k = n$, iterasi berhenti. Jika $k < n$, didefinisikan $k = k + 1$, kembali ke langkah 1.
Dengan d_{ij} = jarak dari titik i ke titik j, s_{ij} = suatu titik yang dilewati oleh suatu lintasan dari i ke j, D_0 = matriks jarak awal, D_k = matriks jarak pada iterasi k, S_0 = matriks urutan awal, dan S_k = matriks urutan pada iterasi k.

3. Hasil dan Pembahasan

Metode Panjang Terpendek *Fuzzy*

Dalam graf klasik, panjang lintasan terpendek *fuzzy* dari titik u ke titik v adalah lintasan terpendek dari titik u ke titik v.

Diberikan dua panjang lintasan dalam jaringan adalah $l_1 = 5$ dan $l_2 = 6$. Dalam himpunan *fuzzy*,$l_1 = 5$ dapat dituliskan dengan $L_1 = \{1.0/5\}$, dapat diartikan bahwa nilai keanggotaan dari 5 adalah 1.0. Demikian pula dengan $l_2 = 6$ dapat dituliskan dengan $L_2 = \{1.0/6\}$, yang artinya nilai keanggotaan dari 6 adalah 1.0.

Kemudian digeneralisasi untuk menghitung L_{min} pada m panjang lintasan *fuzzy* $L_1, L_2, \dots, L_m$. Notasi $L_i(x) / x$ menyatakan bahwa elemen x pada L_i yang memiliki nilai keanggotaan $L_i(x)$. Untuk menentukan L_{min} , langkah awal adalah menghitung $SL_i(x)$ dimana seperti panjang pada himpunan tegas, x adalah panjang terpendek jika tidak ada panjang lain yang lebih pendek atau sama dengan x tersebut. Oleh karena itu, $L_i(x) = L_i \wedge L_j'$ untuk $i, j = 1, 2, 3, \dots m$ dan $j \neq i$ atau dapat dinyatakan sebagai berikut :

$$\begin{aligned} SL_1(x) &= L_1(x) \wedge L_2'(x) \wedge L_3'(x) \wedge \dots \wedge L_m'(x) \\ &= min\ [L_1(x), L_2'(x), L_3'(x), \dots, L_m'(x)] \\ &= min\ [L_1(x), \min_{k\neq 1}\ [L_k'(x)]] \quad \dots\dots (1) \end{aligned}$$

$$L_k'(x) = 1 - \max_{\substack{y < x \\ y \in L_k}} L_k(y) \quad \dots\dots (2)$$

Substitusikan (2) ke pers (1) dihasilkan

$$SL_1(x) = min\left[L_1(x), \min_{k\neq 1} 1 - \max_{\substack{y < x \\ y \in L_k}} L_k(y)\right] \quad \dots\dots (3)$$

Dengan cara yang sama, dapat diperoleh nilai keanggotaan $SL_t(x)$ pada x dalam L_t adalah

$$SL_t(x) = min \left[L_t(x), \min_{k \neq t} 1 - \max_{\substack{y < x \\ y \in L_k}} L_k(y) \right] \ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

Sehingga:

$$L_{min}(x) = SL_1(x) \vee SL_2(x) \vee SL_3(x) \vee \ldots \vee SL_m(x) = \max_{t=1}^{m} SL_t(x)$$

$$= \max_{t=1}^{m} \left[min[L_t(x), min_{k \neq t} L_k'(x)] \right] = \max t$$

$$= 1^m \left[min \left[L_t(x), min_{k \neq t} \left[1 - \max_{\substack{y < x \\ y \in L_k}} L_k(y) \right] \right] \right] \ldots\ldots\ldots\ldots (5)$$

Definisi 3.1

Ukuran kesamaan berikut untuk mengukur derajat kesamaan antara dua panjang *fuzzy*:

$$S(A,B) = \sum_{k=1}^{m} [1 - |A(x_k) - B(x_k)|]/m \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

Ukuran kesamaan didefinisikan dalam pers(6) yang akan membantu mengambil keputusan untuk menentukan lintasan mana yang terpendek.

Ukuran kesamaan $S: F^2 \rightarrow [0, \infty)$ harus memenuhi beberapa sifat dibawah ini :

(P1) $S(A,B) = S(B,A)$ untuk semua $A, B \in F$;

(P2) $S(E, \bar{E}) = 0$, jika E adalah himpunan *crisp* dan $\bar{E}$ adalah *complement* dari E;

(P3) $S(H,H) \geq S(A,B)$ untuk semua $A, B, H \in F$;

(P4) Jika $A \subseteq B \subseteq C$ untuk semua $A, B, C \in F$ maka $S(A,C) \leq S(A,B)$ dan $S(A,C) \leq S(B,C)$.

Algoritma Chuang – Kung

Langkah 1: Menentukan lintasan yang mungkin dari titik sumber(s) ke titik tujuan(d) dan menghitung panjang lintasan (L_i) yang sesuai. Untuk $i = 1,2, \ldots, m$ pada m lintasan yang mungkin.

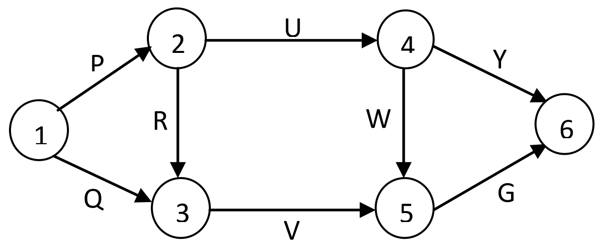
Langkah 2: Menentukan panjang terpendek *fuzzy* (L_{min}) dengan menggunakan metode panjang terpendek *fuzzy*.

Langkah 3: Menghitung Ukuran kesamaan *fuzzy* yang didefinisikan pada persamaan (8) untuk menghasilkan derajat kesamaan $S(L_{min}, L_i)$ antara L_{min} dan L_i untuk $i = 1,2, \ldots, m$.

Langkah 4: Memilih lintasan terpendek dengan derajat kesamaan $S(L_{min}, L_i)$ yang tertinggi.

Contoh 1:

Jaringan sederhana dengan panjang busur *fuzzy* ditunjukkan pada Gambar 3.1 dimana panjang busur *fuzzy* di $P = \{0.5/2, 1.0/3\}$, $Q = \{0.4/1, 1.0/3\}$,
$R = \{1.0/2, 0.2/3, 0.1/4\}$, $U = \{1.0/2, 0.6/4\}$, $V = \{1.0/3, 0.5/4\}$,
$Y = \{1.0/4, 0.7/5\}$, $W = \{0.6/4, 1.0/5, 0.5/6\}$,dan $G = \{1.0/3, 0.4/5\}$, berturut – turut.



Graf Berarah Tidak Siklik

Langkah 1 : Lintasan yang mungkin yaitu :

Lintasan (1) : 1 + 2 + 4 + 5 + 6 = $\{0.5/11, 0.6/12, 1.0/13, 0.6/14, 0.6/15, 0.5/16, 0.4/17, 0.4/18\}$

Lintasan (2) : 1 + 2 + 4 + 6 = $\{0.5/8, 1.0/9, 0.7/10, 0.6/11, 0.6/12\}$

Lintasan (3) : 1 + 2 + 3 + 5 + 6 = $\{0.5/10, 1.0/11, 0.5/12, 0.4/13, 0.4/14, 0.2/15, 0.1/16\}$

Lintasan (4) : 1 + 3 + 5 + 6 = $\{0.4/7, 0.4/8, 1.0/9, 0.5/10, 0.4/11, 0.4/12\}$

Langkah2 : Didapat nilai panjang terpendeknya adalah $\{0.4/7, 0.5/8, 0.6/9\}$ atau $L_{min} = \{0.4/7, 0.5/8, 0.6/9\}$.

Langkah 3 : Dengan ukuran kesamaan, didapat nilai $S(L_{min}, L_i)$ antara L_{min} dan L_i.

$S(L_{min}, L_1) = 0.49$
$S(L_{min}, L_2) = 0.77$
$S(L_{min}, L_3) = 0.62$
$S(L_{min}, L_4) = 0.85$

Langkah 4 : Dipilih lintasan 1-3-5-6 sebagai lintasan terpendek dengan panjang *fuzzy* L_4 merupakan derajat kesamaan tertinggi (=0.85) untuk L_{min} panjang terpendek *fuzzy*.

Lintasan Terpendek *Fuzzy* dengan Algoritma Floyd

Berikut ini Algoritma Floyd akan digunakan untuk menghitung lintasan terpendek *fuzzy*. Langkah awal dengan menentukan matriks jarak awal (D_o) dan matriks urutan awal (S_o). Jika $d_{ik} + d_{kj} < d_{ij}$ maka d_{ij} pada matriks D_o diganti dengan $L_{min}\left(d_{ij}, d_{ik} + d_{kj}\right)$ pada matriks D_k. Untuk matriks S_o atau S_k, perubahan elemen matriksnya mengikuti perubahan elemen pada matriks D_o atau D_k.
Langkah – langkah algoritma :
Didefinisikan matriks D_0 dan S_0.
Jika i dan j tidak terhubung langsung, definisikan $d_{ij} = \infty$.

3. Diambil $k = 1$.
4. Didefinisikan baris k dan kolom k sebagai baris dan kolom pivot.
5. Untuk setiap d_{ij} dalam D_{k-1}, jika $d_{ik} + d_{kj} < d_{ij}$, $(i \neq k, j \neq k$, dan $i \neq j)$.
   - D_k diubah dengan mengganti d_{ij} dalam D_{k-1} dengan $d_{ik} + d_{kj}$.
   - S_k diubah dengan mengganti s_{ij} dalam S_{k-1} dengan k.
6. Jika $k = n$, iterasi berhenti. Jika $k < n$, definisikan $k = k + 1$, kembali ke langkah 1.

Dengan d_{ij} = jarak dari titik i ke titik j, s_{ij} = suatu titik yang dilewati oleh suatu lintasan dari i ke j yang mempunyai jarak d_{ij} , D_0 = matriks jarak awal, D_k = matriks L_{min} pada iterasi k, S_0 = matriks urutan awal, dan S_k = matriks urutan pada iterasi k.

Contoh 3.6

Jaringan sederhana dengan panjang busur *fuzzy* ditunjukkan pada Gambar 3.1 dimana panjang busur *fuzzy* di $P = \{0.5/2\ , 1.0/3\}$, $Q = \{0.4/1, 1.0/3\}$,
$R = \{1.0/2, 0.2/3, 0.1/4\}$, $U = \{1.0/2, 0.6/4\}$, $V = \{1.0/3, 0.5/4\}$,
$Y = \{1.0/4, 0.7/5\}$, $W = \{0.6/4, 1.0/5, 0.5/6\}$,dan $G = \{1.0/3, 0.4/5\}$, berturut – turut.

Didefinisikan matriks jarak awal (D_0) dan matriks urutan awal (S_0), yaitu

	1	**2**	**3**	**4**	**5**	**6**
1	_	$\{0.5/2, 1.0/3\}$	$\{0.4/1, 1.0/3\}$	∞	∞	∞
2	∞	_	$\{1.0/2, 0.2/3, 0.1/4\}$	$\{1.0/2, 0.6/4\}$	∞	∞
3	∞	∞	_	∞	$\{1.0/3, 0.5/4\}$	∞
4	∞	∞	∞	_	$\{0.6/4, 1.0/5, 0.5/6\}$	$\{1.0/4, 0.7/5\}$
5	∞	∞	∞	∞	_	$\{1.0/3, 0.4/5\}$
6	∞	∞	∞	∞	∞	_

Matriks D_0

	1	**2**	**3**	**4**	**5**	**6**
1	_	2	3	4	5	6
2	1	_	3	4	5	6
3	1	2	_	4	5	6
4	1	2	3	_	5	6
5	1	2	3	4	_	6
6	1	2	3	4	5	_

Matriks S_0

Iterasi 1 :

1) Diambil $k = 1$.
   Menentukan baris 1 dan kolom 1 dari matriks D_0 sebagai baris dan kolom pivot.

	1	**2**	**3**	**4**	**5**	**6**
1	_	$\{0.5/2, 1.0/3\}$	$\{0.4/1, 1.0/3\}$	∞	∞	∞
2	∞	_	$\{1.0/2, 0.2/3, 0.1/4\}$	$\{1.0/2, 0.6/4\}$	∞	∞
3	∞	∞	_	∞	$\{1.0/3, 0.5/4\}$	∞
4	∞	∞	∞	_	$\{0.6/4, 1.0/5, 0.5/6\}$	$\{1.0/4, 0.7/5\}$
5	∞	∞	∞	∞	_	$\{1.0/3, 0.4/5\}$
6	∞	∞	∞	∞	∞	_

Matriks D_0 dengan baris 1 dan kolom 1 sebagai baris dan kolom pivot

	1	**2**	**3**	**4**	**5**	**6**
1	_	2	3	4	5	6
2	1	_	3	4	5	6
3	1	2	_	4	5	6
4	1	2	3	_	5	6
5	1	2	3	4	_	6
6	1	2	3	4	5	_

Matriks S_0 dengan baris 1 dan kolom 1 sebagai baris dan kolom pivot

2) Pada matriks D_0, elemen $d_{ij} < d_{ik} + d_{kj}$ maka tidak ada elemen yang diubah untuk matriks D_0. Matriks $D_1 = D_0$ pada iterasi berikutnya.
3) Tidak ada elemen s_{ij} yang diubah dari matriks S_0. Karena perubahan S_k mengikuti perubahan pada D_k, maka matriks $S_1 = S_0$ pada iterasi berikutnya.
4) Karena $k < 6$ maka untuk iterasi berikutnya didefinisikan $k = 2$ dan kembali ke langkah 1.

Iterasi 2

1) Diambil $k = 2$.
Menentukan baris 2 dan kolom 2 dari matriks D_0 sebagai baris dan kolom pivot.

	1	**2**	**3**	**4**	**5**	**6**
1	_	{0.5 /2, 1.0/3}	{0.4 /1, 1.0/3}	∞	∞	∞
2	∞	_	{1.0 /2, 0.2 /3, 0.1/4}	{1.0 /2, 0.6/4}	∞	∞
3	∞	∞	_	∞	{1.0 /3, 0.5/4}	∞
4	∞	∞	∞	_	{0.6 /4, 1.0 /5, 0.5/6}	{1.0 /4, 0.7/5}
5	∞	∞	∞	∞	_	{1.0 /3, 0.4/5}
6	∞	∞	∞	∞	∞	_

Matriks D_1

	1	2	3	4	5	6
1	_	2	3	4	5	6
2	1	_	3	4	5	6
3	1	2	_	4	5	6
4	1	2	3	_	5	6
5	1	2	3	4	_	6
6	1	2	3	4	5	_

Matriks S_1

2) Pada matriks D_1, elemen $d_{14} > d_{12} + d_{24}$ maka elemen d_{14} pada matriks D_1 diubah menjadi $\{0.4/3, 1.0/5, 0.6/7\}$ sehingga diperoleh matriks D_2 pada iterasi berikutnya.
3) Untuk elemen s_{14} matriks S_1 diubah menjadi 2 sehingga diperoleh matriks S_2 pada iterasi berikutnya.
4) Karena $k < 6$ maka untuk iterasi berikutnya didefinisikan $k = 3$ dan kembali ke langkah 1.

Iterasi 3

1) Diambil $k = 3$.
Menentukan baris 3 dan kolom 3 dari matriks D_1 sebagai baris dan kolom pivot.

	1	2	3	4	5	6
1	_	$\{0.5/2, 1.0/3\}$	$\{0.4/1, 1.0/3\}$	$\{0.4/3, 1.0/5, 0.6/7\}$	∞	∞
2	∞	_	$\{1.0/2, 0.2/3, 0.1/4\}$	$\{1.0/2, 0.6/4\}$	∞	∞
3	∞	∞	_	∞	$\{1.0/3, 0.5/4\}$	∞
4	∞	∞	∞	_	$\{0.6/4, 1.0/5, 0.5/6\}$	$\{1.0/4, 0.7/5\}$
5	∞	∞	∞	∞	_	$\{1.0/3, 0.4/5\}$

6	∞	∞	∞	∞	∞	_

Matriks D_2

	1	**2**	**3**	**4**	**5**	**6**
1	_	2	3	1	5	6
2	1	_	3	4	5	6
3	1	2	_	4	5	6
4	1	2	3	_	5	6
5	1	2	3	4	_	6
6	1	2	3	4	5	_

Matriks S_2

2) Pada matriks D_2, elemen $d_{15} > d_{13} + d_{35}$ maka elemen d_{15} pada matriks D_2 diubah menjadi $\{0.4/4, 0.4/5, 1.0/6, 0.5/7\}$ dan elemen $d_{25} > d_{23} + d_{35}$ maka elemen d_{25} pada matriks D_2 diubah menjadi $\{1.0/5, 0.5/6,\ 0.2/7, 0.1/8\}$ sehingga diperoleh matriks D_3 pada iterasi berikutnya.
3) Untuk elemen s_{15} dan s_{25} pada matriks S_2 diubah menjadi 3 sehingga diperoleh matriks S_3 pada iterasi berikutnya.
4) Karena $k < 6$ maka untuk iterasi berikutnya didefinisikan $k = 4$ dan kembali ke langkah 1.

Iterasi 4

1) Diambil $k = 4$.
   Menentukan baris 4 dan kolom 4 dari matriks D_2 sebagai baris dan kolom pivot.

	1	2	3	4	5	6
1	_	{0.5/2, 1.0 /3}	{0.4/1, 1.0 /3}	{0.4/3, 1.0 /5, 0.6/7}	{0.4/4, 0.4 /5, 1.0 /6, 0.5/7}	∞
2	∞	_	{1.0/2, 0.2 /3, 0.1/4}	{1.0/2, 0.6 /4}	{1.0/5, 0.5 /6, 0.2 /7, 0.1/8}	∞
3	∞	∞	_	∞	{1.0/3, 0.5 /4}	∞
4	∞	∞	∞	_	{0.6/4, 1.0 /5, 0.5/6}	{1.0/4, 0.7 /5}
5	∞	∞	∞	∞	_	{1.0/3, 0.4 /5}
6	∞	∞	∞	∞	∞	_

Matriks D_3

	1	2	3	4	5	6
1	_	2	3	1	3	6
2	1	_	3	4	3	6
3	1	2	_	4	5	6
4	1	2	3	_	5	6
5	1	2	3	4	_	6
6	1	2	3	4	5	_

Matriks S_3

2) Pada matriks D_3, elemen $d_{16} > d_{14} + d_{46}$ maka elemen d_{16} pada matriks D_3 diubah menjadi $\{0.4/7, 0.4/8, 1.0/9, 0.7/10, 0.6/11, 0.6/12\}$, elemen $d_{25} > d_{24} + d_{45}$ maka elemen d_{25} pada matriks D_3 diubah menjadi $\{1.0/5, 0.5/6, 0.2/7\}$, dan elemen $d_{26} > d_{24} + d_{46}$ maka elemen d_{26} pada matriks D_3 diubah menjadi $\{1.0/6, 0.7/7, 0.6/8, 0.6/9\}$ sehingga diperoleh matriks D_4 pada iterasi selanjutnya.
3) Untuk elemen s_{16}, s_{25}, dan s_{26} pada matriks S_3 diubah menjadi 4 pada matriks S_4.
4) Karena $k < 6$ maka untuk iterasi berikutnya didefinisikan $k = 5$ dan kembali ke langkah 1.

Iterasi 5

1) Diambil $k = 5$.
Menentukan baris 5 dan kolom 5 dari matriks D_3 sebagai baris dan kolom pivot.

	1	**2**	**3**	**4**	**5**	**6**
1	$-$	$\{0.5/2, 1.0/3\}$	$\{0.4/1, 1.0/3\}$	$\{0.4/3, 1.0/5, 0.6/7\}$	$\{0.4/4, 0.4/5, 1.0/6, 0.5/7\}$	$\{0.4/7, 0.4/8, 1.0/9, 0.7/10, 0.6/11, 0.6/12\}$
2	∞	$-$	$\{1.0/2, 0.2/3, 0.1/4\}$	$\{1.0/2, 0.6/4\}$	$\{1.0/5, 0.5/6, 0.2/7\}$	$\{1.0/6, 0.7/7, 0.6/8, 0.6/9\}$
3	∞	∞	$-$	∞	$\{1.0/3, 0.5/4\}$	∞
4	∞	∞	∞	$-$	$\{0.6/4, 1.0/5, 0.5/6\}$	$\{1.0/4, 0.7/5\}$
5	∞	∞	∞	∞	$-$	$\{1.0/3, 0.4/5\}$
6	∞	∞	∞	∞	∞	$-$

Matriks D_4

	1	2	3	4	5	6
1	_	2	3	1	3	4
2	1	_	3	4	4	4
3	1	2	_	4	5	6
4	1	2	3	_	5	6
5	1	2	3	4	_	6
6	1	2	3	4	5	_

Matriks S_4

2) Pada matriks D_4, elemen $d_{16} > d_{15} + d_{56}$ maka elemen d_{16} pada matriks D_4 diubah menjadi $\{0.4/7, 0.4/8, 0.6/9\}$, elemen $d_{26} > d_{25} + d_{56}$ maka elemen d_{26} pada matriks D_4 diubah menjadi $\{1.0/6, 0.7/7, 0.6/8\}$, dan elemen $d_{36} > d_{35} + d_{56}$ maka elemen d_{36} pada matriks D_4 diubah menjadi $\{1.0/6, 0.5/7, 0.4/8, 0.4/9\}$ sehingga diperoleh matriks D_5 pada iterasi berikutnya.
3) Untuk elemen s_{16}, s_{26}, dan s_{36} pada matriks S_4 diubah menjadi 5 sehingga diperoleh matriks S_5 pada iterasi berikutnya.
4) Karena $k < 6$ maka untuk iterasi berikutnya didefinisikan $k = 6$ dan kembali ke langkah 1.

Iterasi 6

1) Diambil $k = 6$.
Menentukan baris 6 dan kolom 6 dari matriks D_5 sebagai baris dan kolom pivot.

	1	2	3	4	5	6
1	_	$\{0.5/2, 1.0/3\}$	$\{0.4/1, 1.0/3\}$	$\{0.4/3, 1.0/5, 0.6/7\}$	$\{0.4/4, 0.4/5, 1.0/6, 0.5/7\}$	$\{0.4/7, 0.4/8, 0.6/9\}$
2	∞	_	$\{1.0/2, 0.2/3, 0.1/4\}$	$\{1.0/2, 0.6/4\}$	$\{1.0/5, 0.5/6, 0.2/7\}$	$\{1.0/6, 0.7/7, 0.6/8\}$
3	∞	∞	_	∞	$\{1.0/3, 0.5/4\}$	$\{1.0/6, 0.5/7, 0.4/8, 0.4/9\}$

4	∞	∞	∞	_	$\{0.6/4, 1.0/5, 0.5/6\}$	$\{1.0/4, 0.7/5\}$
5	∞	∞	∞	∞	_	$\{1.0/3, 0.4/5\}$
6	∞	∞	∞	∞	∞	_

Matriks D_5

	1	**2**	**3**	**4**	**5**	**6**
1	_	2	3	1	3	5
2	1	_	3	4	4	5
3	1	2	_	4	5	5
4	1	2	3	_	5	6
5	1	2	3	4	_	6
6	1	2	3	4	5	_

Matriks S_5

1) Pada matriks D_5, elemen $d_{ij} < d_{ik} + d_{kj}$ maka tidak ada elemen pada matriks D_4 dan S_4 yang diubah. Karena $k = n$ maka iterasi berhenti.

Dari iterasi 6 didapat hasil akhir untuk lintasan terpendek dari titik 1 ke titik 6 adalah

$s_{16} = 5$ —— $s_{15} = 3$ —— $s_{13} = 3$

$s_{56} = 6$ $s_{35} = 5$

Jadi, lintasan terpendeknya adalah 1 – 3 – 5 – 6 dengan menggunakan metode Floyd.

4. Kesimpulan

Dari pembahasan yang telah diuraikan sebelumnya, untuk menentukan lintasan terpendek *fuzzy* dapat diselesaikan dengan menggunakan algoritma Chuang – Kung dan algoritma Floyd. Dalam algoritma Chuang – Kung, dapat diselesaikan dengan menentukan semua lintasan yang mungkin dilalui dari titik awal ke titik tujuan. Kemudian digunakan metode panjang terpendek *fuzzy* dan ukuran kesamaan *fuzzy* untuk menyelesaikannya. Dengan metode panjang terpendek *fuzzy* didapat nilai dari L_{min}. Kemudian ukuran kesamaan digunakan untuk menentukan derajat kesamaan $S(L_{min}, L_i)$ antara dua panjang *fuzzy*. Nilai L_i yang terbesar merupakan panjang terpendek lintasan tersebut. Dalam algoritma Floyd, langkah awal adalah dengan menentukan matriks jarak awal (D_0) dan matriks urutan awal (S_0) kemudian memeriksa elemen – elemennya. Jika $d_{ik} + d_{kj} < d_{ij}$ maka d_{ij} pada matrik D_k diubah menjadi $L_{min}(d_{ij}, d_{ik} + d_{kj})$. Elemen s_{ij} pada matriks S_k diubah mengikuti perubahan d_{ij} pada matrik D_k. Dari hasil simulasi pada graf (Gambar 3.1) dengan menggunakan kedua algoritma tersebut didapat lintasan terpendeknya adalah 1 – 3 – 5 – 6.

5. Referensi

[1] Arifin, Achmad. 2000. *Aljabar*. ITB

[2] Bartle, Robert G. dan Donald R. Sherbert. 2000. *Introduction to Real Analysis Third Edition.* New York : John Willey and Sons.

[3] Aini, Yusra Dewi. 2010. *Analisis Algoritma A Star* (A^*) *dan Implementasinya dalam Pencarian Jalur Terpendek pada Jalur Lintas Sumatera dari Provinsi Sumatera Utara*. Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Sumatera Utara.

[4] Seblyanry, Roland. 2012. *Kajian Tentang Ruang Tapologi Fuzzy*. Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Sumatera Utara.

[5] Sunitha, M.S. and Vijaya Kumar, A. 2001. *Complement of A Fuzzy Graphs.* Indian J. pure applied Mathematical, Vol. 33, No. 9.

[6] Susilo, Frans. 2006. *Himpunan dan Logika Kabur serta Aplikasinya.* Yogyakarta : Graha Ilmu.

[7] Taha, Hamdy A. 1997. *Operation Research ; An Introduction third edition.* New York.

[8] Tzuang-Nan Chuang and Jung-Yuan Kung. 2006. *A New Algorithm for the discrete fuzzy shortest path problem in a network,* Applied Matematics and Computation 174.

[9] Wilson, J. Robin and John J. Watskin. 1990. *Graphs An Introductory Approach*. New York : University Course Graphs, Network, and Design.

[10] Zadeh, A. Lotfi, dkk. 1974. *Fuzzy Sets and Their Applications to Cognitive and Decision Process.* California : The University of California.