Jurnal CoreIT, Vol.7, No.1, Juni 2021
ISSN 2460-738X (Print)
ISSN 2599-3321 (Online)

OPTIMALISASI METODE *WEIGHTED-SUM DIJKSTRA ALGORITHM* UNTUK MENENTUKAN RUTE TERBAIK YANG SESUAI DENGAN KEBUTUHAN

Tetra Widianto[1], Windu Gata[2], Nurmalasari[3], Supriadi Panggabean[4], Siska Rahmadani[5]
[1,2,3,4,5]Program Studi Ilmu Komputer, STMIK Nusa Mandiri
Jl. Margonda Raya No. 545, Pondok Cina, Beji, Depok, Jawa Barat 16424
14002461@nusamandiri.ac.id[1], windu@nusamandiri.ac.id[2], nurmalasasi.nmr@nusamandiri.ac.id[3],
14002471@nusamandiri.ac.id[4], 14002456@nusamandiri.ac.id[5]

***Abstrak* – Di tengah kondisi jalan yang sulit untuk diprediksi, pencarian rute tercepat dapat mencegah hilangnya banyak waktu dan tenaga. Rute tercepat sering dipahami sebagai rute terpendek. Hal ini sangat beralasan karena semakin pendek jarak akan semakin cepat waktu tempuhnya. Dalam beberapa penelitian, pencarian rute terpendek dilakukan dengan menerapkan algoritma Dijkstra. Algoritma ini dijalankan dengan menghitung jarak terpendek di tiap-tiap titik guna mencari nilai terkecil dari titik awal ke titik tujuan. Namun, jika dalam perjalanan terdapat kriteria lain selain jarak maka rute terpendek yang didapatkan belum tentu merupakan rute terbaik. Dengan banyaknya kriteria seperti ini, penentuan rute terbaik memerlukan pendekatan yang disebut dengan MCDM (*Multi Criteria Decision Making*) supaya nilai jarak dapat disubstitusi dengan nilai agregat dari bobot seluruh kriteria. Dalam sebuah penelitian, terdapat sebuah metode yang menggabungkan salah satu metode MCDM, yakni WS (*Weighted-Sum*) dengan algoritma Dijkstra yang dinamakan WSDA (*Weighted-Sum Dijkstra Algorithm*). Meskipun metode ini dapat digunakan untuk mencari rute terbaik dengan banyak kriteria, tetapi berdasarkan penelitian lain teknik normalisasinya belum optimal karena tidak menggunakan teknik normalisasi vektor atau *linear-max*. Hasil penelitian ini menunjukkan bahwa optimalisasi metode WSDA dengan menerapkan teknik normalisasi *linear-max* dan mengubah *equal weight* menjadi bobot skala Likert sangat sesuai untuk menentukan rute terbaik yang sesuai kebutuhan.**

Kata Kunci – Dijkstra, rute terbaik, Weighted-Sum.

PENDAHULUAN

Dalam melakukan sebuah perjalanan, sering kali muncul pertanyaan tentang rute manakah yang tercepat. Dalam menjawab pertanyaan ini, kriteria yang sering digunakan adalah jarak karena semakin pendek jarak semakin cepat pula waktu tempuhnya. Dengan semakin kompleksnya kondisi jalan saat ini, jarak tidak lagi menjadi satu-satunya kriteria yang dibutuhkan dalam menentukan rute tercepat. Ada beberapa kondisi yang mempengaruhi kecepatan seperti kemacetan, adanya kerumunan massa dan kerusakan jalan yang semuanya perlu dipertimbangkan untuk dijadikan kriteria.

Dalam kehidupan sehari-hari, terkadang rute yang dicari bukanlah yang tercepat, tetapi yang lebih sesuai dengan kebutuhan. Sebagai contoh, seseorang yang perlu mengisi bahan bakar lebih memilih rute yang melewati SPBU atau contoh lainnya, seseorang yang ingin pulang sambil berbelanja lebih memilih rute yang melewati *mall/supermarket* atau *minimarket*. Beberapa contoh tersebut menjelaskan bahwa meskipun rute yang dilalui lebih jauh, tetapi karena lebih sesuai dengan kebutuhan maka rute tersebutlah yang terbaik.

Dalam ilmu pengetahuan, penentuan rute atau lintasan terpendek (*shortest path*) merupakan salah satu aplikasi dari teori graf. Suatu graf $G = (V, E)$, didefinisikan sebagai suatu pasangan himpunan $(V(G), E(G), \psi G)$, dimana $V(G)$ adalah himpunan tak kosong dari elemen yang disebut titik (*vertex*), $E(G)$ adalah himpunan sisi (*edge*) dan ψG adalah suatu fungsi keterkaitan (*incidency function*) yang mengaitkan setiap sisi G dengan pasangan titik di G. Banyaknya titik di G disebut orde (*order*) dari G yang dinotasikan dengan $|V(G)| = P$, dan banyaknya sisi pada G disebut ukuran (*size*) dari G yang dinotasikan dengan $|E(G)| = Q$. Misal terdapat graf $G = (V, E)$ dengan $v_i , v_j \in V(G)$. Graf yang digunakan dalam penentuan rute terpendek adalah graf berbobot (*weighted graph*), yaitu graf yang untuk setiap sisinya memiliki nilai $w(e)$ sehingga panjang lintasan dari suatu titik ke titik lainnya $l(u, v)$ merupakan jumlah dari nilai-nilai pada setiap sisinya. Pengaitan suatu bilangan riil $w(e)$ untuk setiap sisi e di graf G disebut bobot (*weight*) [1].

Salah satu algoritma yang digunakan dalam penentuan lintasan terpendek adalah Dijkstra. Sesuai dengan namanya, algoritma ini dikembangkan oleh Edsger Wybe Dijkstra pada tahun 1959. Lintasan terpendek dengan titik awal v_0 dan titik akhir v_k didefinisikan sebagai lintasan yang memiliki panjang lintasan yang minimum dari v_0 ke v_k [1]. Dalam berbagai penelitian, algoritma ini cukup populer digunakan, di antaranya [2], [3], [4] dan [5].

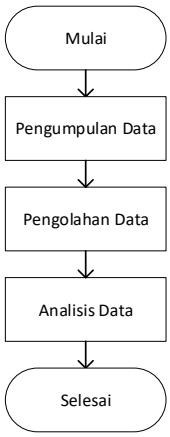
Seperti yang telah disinggung sebelumnya, rute terbaik bukanlah rute tercepat atau dalam hal ini terpendek, tetapi rute yang paling memenuhi seluruh kriteria. Kriteria adalah sekumpulan persyaratan atau atribut independent yang harus dipenuhi oleh beberapa alternatif. Setiap kriteria dapat diukur dalam satuan yang berbeda, tetapi semuanya harus dinormaliasi untuk mendapatkan klasifikasi tak berdimensi, yaitu rentang/skala numerik yang umum untuk memungkinkan agregasi menjadi skor akhir. Oleh karena itu, normalisasi data adalah bagian penting dari setiap proses pengambilan keputusan karena mengubah data masukan menjadi data numerik dan sebanding sehingga memungkinkan metode MCDM (*Multi Criteria Decission Making*) menilai dan memberi peringkat pada alternatif [6].

Beberapa metode MCDM yang dikenal seperti AHP (*Analitic Hierarchy Process*) dan TOPSIS (*Technique for Others Preference by Similarity to Ideal Solution*) [7] pernah digunakan dalam penelitian-penelitan lain untuk menentukan rute terbaik yang dikombinasikan dengan algoritma Dijkstra. Di antara penelitian-penelitian tersebut adalah [8] dan [9]. Demikian juga dengan metode WS (*Weighted Sum*) atau SAW (*Simple Addictive Weighting*) yang mungkin paling terkenal dan paling banyak digunakan dalam MCDM [10] juga pernah digunakan, yaitu pada penelitian [11] dan [12]. Meskipun peneltian [11] dan [12] menggunakan metode MCDM yang sama, tetapi teknik normalisasi yang digunakan berbeda. Penelitian [11] menggunakan normalisasi vektor (*vector normalization*) sedangkan penelitian [12] menggunakan normalisasi *linear-sum.*

Berdasarkan penelitian [10], teknik normalisasi yang lebih sesuai untuk metode WS adalah normalisasi vektor dan *linear-max*. Dengan demikian, penelitian [12] dengan metodenya yang dinamakan WSDA (*Weighted-Sum Dijkstra Algorithm*) dapat dikatakan belum optimal. Berangkat dari penelitian [10] dan [12], penelitian ini berupaya mengoptimalkan metode WSDA dengan mengubah teknik normalisasinya dari yang semula *linear-sum* menjadi *linear-max*. Di samping itu, dilakukan pengembangan pada teknik pembobotan kriterianya dari yang semula menggunakan *equal weight* menjadi bobot skala Likert. Dalam penelitian ini, subjektivitas lebih diunggulkan karena kriteria tambahan yang digunakan adalah fasilitas umum yang masing-masing orang dapat berbeda persepsinya. Kondisi inilah yang mendasari penggunaan skala Likert agar persepsi yang merupakan data kualitatif dapat dikuantifikasi menjadi skala numerik yang dapat diolah [13].

BAHAN DAN METODE

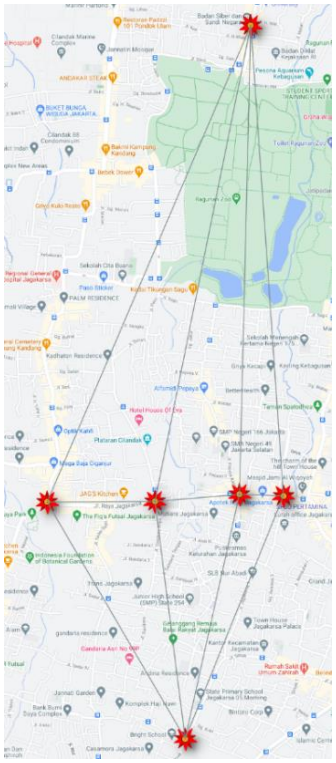
Penelitian ini dilakukan dengan tahapan seperti yang terlihat di Gambar 1.



Gambar 1. Tahapan penelitian

A. Pengumpulan Data

Pengumpulan data dilakukan dengan bantuan aplikasi QGIS supaya titik-titik yang ditentukan berada di lokasi yang sebenarnya. Adapun sisi yang menghubungkan antar titiknya tidak dibuat yang sebenarnya supaya dapat menyederhanakan bentuk graf yang akan dibuat.



Gambar 2. Ilustrasi lintasan

Gambar 2 mengilustrasikan *m* lintasan yang dapat dilalui dalam perjalanan pulang dari sebuah kantor di wilayah Pasar Minggu – Jakarta Selatan ke perumahan di wilayah Jagakarsa – Jakarta Selatan. Dengan menggunakan QGIS, tiap-tiap titik tersebut dapat diberi label nama dan diperoleh koordinatnya seperti yang terlihat di Gambar 3.

id	nama	xcoord	ycoord
1	KANTOR	106,820911847...	-6,3036091989...
2	HERMAN SUSILO	106,810899762...	-6,3269658255...
3	GANDARIA	106,816200277...	-6,3269300433...
4	KEL. LAMA	106,820354286...	-6,3265832317...
5	AL WIQOYAH	106,822514371...	-6,3266575485...
6	RUMAH	106,817687413...	-6,3385563541...

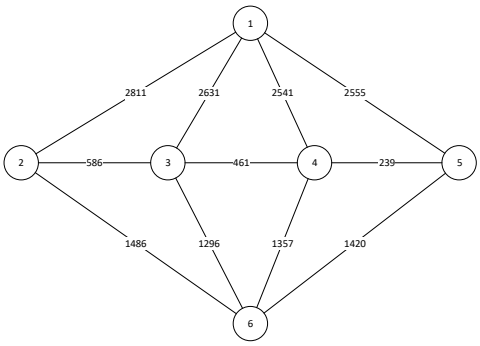
Gambar 3. Daftar nama titik dan koordinatnya

Masih menggunakan QGIS, bobot di tiap-tiap sisi dapat diperoleh dengan melihat matriks jarak. Jarak yang yang tertera di Gambar 4 merupakan jarak tegak lurus antar titik sehingga tidak mencerminkan jarak yang diukur dari panjang jalan.

InputID	TargetID	Distance
1	5	2555,0244998637113
1	2	2810,503486300014
1	4	2541,3946358785383
1	3	2631,1627983442754
2	3	586,4940035276687
2	6	1485,5855220048345
3	6	1296,2141169118838
3	4	461,2219078149738
4	5	239,1460670601621
4	6	1356,5619623641003
5	6	1420,1173307543029

Gambar 4. Matriks jarak

Berdasarkan ilustrasi lintasan, daftar nama titik dan koordinatnya serta matriks jarak dapat dibuat graf berbobot *G* seperti terlihat di Gambar 5.



Gambar 5. Graf berbobot *G*

Setelah jarak berhasil diperoleh, langkah berikutnya adalah mengumpulkan data tentang jumlah fasilitas umum yang tersedia di tiap-tiap sisi/jalan tersebut. Kecuali jumlah SPBU yang diperoleh dengan pengamatan di lapangan, jumlah bengkel dan masjid diperoleh dengan menggunakan fungsi RANDOM di Microsoft Excel sehingga tidak menggambarkan kondisi yang sebenarnya. Data yang telah diperoleh tersebut disajikan di Tabel 1.

Tabel 1. Jumlah fasilitas umum

No	Sisi	Jarak (m)	SPBU	Bengkel	Masjid
1	1-2	2811	0	2	0
2	1-3	2631	0	1	1
3	1-4	2541	0	2	0
4	1-5	2555	0	1	1
5	2-3	586	0	2	0
6	3-4	461	0	1	0
7	4-5	239	0	0	1
8	2-6	1486	1	1	0
9	3-6	1296	0	2	1
10	4-6	1357	0	1	0
11	5-6	1420	1	0	1

B. Pengolahan Data

Dalam MCDM, biasanya melibatkan satu set m alternatif A_i ($i = 1, 2, \ldots, m$), yang dievaluasi berdasarkan satu set n atribut (kriteria evaluasi) C_j ($j = 1, 2, \ldots, n$). Penilaian dilakukan untuk menentukan vektor pembobotan $W = (w_1, w_2, \ldots, w_j, \ldots, w_n)$ dan matriks keputusan $X = \{x_{ij}, i = 1, 2, \ldots, m; j = 1, 2, \ldots, n\}$. Vektor pembobotan W merepresentasikan kepentingan relatif dari n atribut C_j ($j = 1, 2, \ldots, n$). Matriks keputusan X merepresentasikan peringkat kinerja x_{ij} dari alternatif A_i ($i = 1, 2, \ldots, m$) yang berhubungan dengan atribut C_j ($j = 1, 2, \ldots, n$) [10].

Pada umumnya, metode MCDM terdiri atas 2 proses, yaitu normalisasi dan agregasi. Normalisasi dilakukan untuk mengubah peringkat kinerja menjadi skala unit yang kompatibel. Selanjutnya, agregasi dilakukan untuk menggabungkan matriks keputusan yang dinormalisasi $R = \{r_{ij}, i = 1, 2, \ldots, m; j = 1, 2, \ldots, n\}$ dan bobot atribut W untuk mencapai nilai preferensi keseluruhan untuk setiap alternatif V_i ($i = 1, 2, \ldots, m$) [10]. Matriks keputusan yang dinormalisasi R merepresentasikan peringkat kinerja yang dinormalisasi r_{ij} dari alternatif A_i ($i = 1, 2, \ldots, m$) yang berhubungan dengan atribut C_j ($j = 1, 2, \ldots, n$) [10].

Normalisasi data dalam penelitian ini dilakukan dengan teknik *linear-max*. Formula dalam teknik normalisasi ini terbagi dalam 2 kategori, yaitu biaya (*cost*) dan manfaat (*benefit*) seperti yang terlihat di Tabel 2. Untuk kriteria yang termasuk dalam kategori biaya, nilai yang tinggi akan menghasilkan nilai normalisasi yang rendah (minimalisasi – biaya), sedangkan untuk kriteria yang termasuk dalam kategori manfaat, nilai yang tinggi akan menghasilkan nilai yang tinggi juga (maksimalisasi – manfaat) [6].

Tabel 2. Formula normalisasi *linear-max*

No	Kategori	Formula
1	Manfaat	$r_{ij} = \frac{x_{ij}}{x_j^{max}}$
2	Biaya	$r_{ij} = 1 - \frac{x_{ij}}{x_j^{max}}$

Catatan. x_j^{max} adalah peringkat kinerja maksimum di antara alternatif untuk atribut C_j ($j = 1, 2, \ldots, n$) [10].

Normalisasi

Dikarenakan algoritma Dijkstra memperhatikan bobot terkecil dari tiap-tiap sisi, maka formula normalisasi di Tabel 2 perlu disesuaikan supaya kriteria dengan kategori biaya dapat dimaksimalkan sedangkan kriteria dengan kategori manfaat dapat diminimalkan. Hasil penyesuaian ini dapat dilihat di Tabel 3.

Tabel 3. Penyesuaian formula normalisasi *linear-max*

No	Kriteria	Kategori	Formula
1	Jarak	Biaya	$r_{ij} = \frac{x_{ij}}{x_j^{max}}$
2	SPBU, Bengkel, Masjid	Manfaat	$r_{ij} = 1 - \frac{x_{ij}}{x_j^{max}}$

Catatan. x_j^{max} adalah peringkat kinerja maksimum di antara alternatif untuk atribut C_j *(j = 1, 2,…, n)* [10].

Pembobotan Atribut

Bobot atribut *W* merupakan persentase dari nilai persepsi $P = (p_1, p_2, \ldots, p_j, \ldots, p_n)$ yang menggambarkan tingkat kepentingan pada suatu atribut. Untuk mendapatkan persepsi pengguna, dilakukan pengukuran menggunakan skala Likert yang bernilai 1 sampai dengan 5 yang merepresentasikan tingkatan seperti yang terlihat di Tabel 4.

Tabel 4. Skala Likert

No	Nilai	Tingkat Kepentingan
1	5	Sangat Tinggi
2	4	Tinggi
3	3	Sedang
4	2	Rendah
5	1	Sangat Rendah

Nilai persepsi yang telah didapatkan kemudian dihitung persentasenya untuk menghasilkan bobot atribut dengan rumus di Persamaan 1.

$$w_j(\%) = \frac{p_j}{\sum_{j=1}^{n} p_j} \times 100 \qquad (1)$$

Dimana p_j adalah nilai persepsi pada atribut ke-*j*, *j=1, 2, …, n.*

Agregasi

Agregasi data dilakukan dengan metode WS dimana logika dasarnya adalah mendapatkan jumlah bobot peringkat kinerja setiap alternatif dari seluruh atribut. Nilai preferensi keseluruhan dari setiap alternatif diperoleh dengan rumus di Persamaan 2.

$$V_i = \sum_{j=1}^{n} w_j r_{ij} \; ; i = 1, 2, \ldots m. \qquad (2)$$

Dimana $V(A_i)$ adalah nilai fungsi dari alternatif A_i, w_j adalah bobot atribut C_j dan r_{ij} adalah peringkat kinerja yang dinormalisasi [10].

Pencarian Rute Terpendek

Rute terpendek ditentukan berdasarkan bobot terkecil nilai preferensi keseluruhan untuk setiap alternatif V_i ($i = 1, 2, \ldots, m$) dengan menjalankan algoritma Dijkstra sebagai berikut.

1. Tandai semua titik sebagai "belum dikunjungi" dan buat satu himpunan yang terdiri atas semua titik "belum dikunjungi" yang dinamakan himpunan titik "belum dikunjungi"
2. Beri nilai bobot tentatif ke semua titik. Setel nilai 0 (nol) untuk "titik awal" dan ~ (tak terhingga) untuk titik lainnya. Tetapkan "titik awal" sebagai "titik saat ini".
3. Dari "titik saat ini", periksa semua "titik terdekat" "belum dikunjungi" dan hitung nilai bobot tentatifnya dari "titik saat ini". Bandingkan nilai bobot tentatif yang baru dihitung dengan nilai yang ditetapkan saat ini dan tetapkan yang lebih kecil.
4. Ketika selesai mempertimbangkan semua "titik terdekat" "belum dikunjungi" dari "titik saat ini" tandai "titik saat ini" sebagai "sudah dikunjungi" dan hapus dari himpunan titik "belum dikunjungi". Titik "sudah dikunjungi" tidak akan diperiksa lagi.
5. Jika titik tujuan ditandai "sudah dikunjungi" atau jika nilai bobot tentatif terkecil di antara titik dalam himpunan titik "belum dikunjungi" adalah tak terhingga maka berhenti. Algoritma selesai.
6. Jika tidak maka pilih titik "belum dikunjungi" yang ditandai dengan nilai bobot terkecil dan tetapkan sebagai "titik saat ini" kemudian kembali ke langkah 3.

HASIL DAN PEMBAHASAN

A. Normalisasi/Matriks *R*

Nilai r_{ij} ($i = 1, 2, \ldots, m; j = 1, 2, \ldots, n$) diperoleh dengan mengaplikasikan formula normalisasi *linear-max* sesuai dengan kondisi yang telah ditetapkan. Hasil penghitungan matriks *R* ini dapat dilihat pada Tabel 5.

Tabel 5. Matriks *R*

No	Sisi	Jarak	SPBU	Bengkel	Masjid
1	1-2	1	1	0	1
2	1-3	0,936	1	0,5	0
3	1-4	0,904	1	0	1
4	1-5	0,909	1	0,5	0
5	2-3	0,208	1	0	1
6	3-4	0,164	1	0,5	1
7	4-5	0,085	1	1	0
8	2-6	0,529	0	0,5	1
9	3-6	0,461	1	0	0
10	4-6	0,483	1	0,5	1
11	5-6	0,505	0	1	0

Sebagai contoh, pada sisi 1-2, nilai jarak dan SPBU diperoleh dengan cara sebagai berikut.

$$r_{jarak} = {}^{x_{jarak}}/_{x_{jarak}^{max}} = {}^{2811}/_{2811} = 1$$
$$r_{SPBU} = 1 - {}^{x_{SPBU}}/_{x_{SPBU}^{max}} = 1 - {}^{0}/_{1} = 1$$

B. Simulasi Pencarian Rute Terpendek

Dalam penelitian ini disimulasikan 2 skenario yang sama-sama mencari rute terbaik dari titik 1 ke titik 6, tetapi dengan tingkat kepentingan yang berbeda untuk melihat apakah metode WSDA yang telah dioptimalkan mampu merespon perubahan bobot atribut *W* dan memberikan rute terbaik yang sesuai dengan kebutuhan.

Skenario I

Pada skenario I, pengguna menetapkan bengkel sebagai "sangat penting", SPBU sebagai "sedang" dan kriteria lainnya sebagai "rendah". Tingkat kepentingan tersebut kemudian dikonversi ke nilai persepsi dan bobot atribut sebagaimana yang terlihat di Tabel 6.

Tabel 6. Bobot atribut *W* pada skenario I

No	Kriteria	Nilai	*W*
1	Jarak	1	0,1
2	SPBU	3	0,3
3	Bengkel	5	0,5
4	Masjid	1	0,1
	Jumlah	10	1

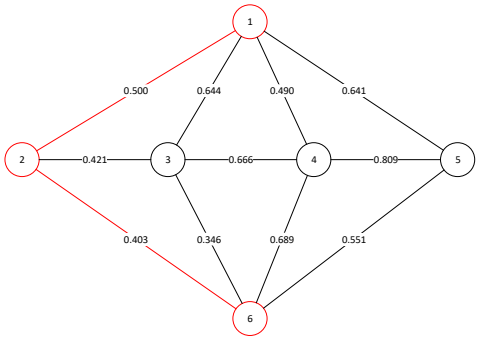
Hasil perkalian antara matriks *R* dan bobot atribut *W* di Tabel 6 disajikan dalam Tabel 7. Disertakan pula nilai preferensi keseluruhan untuk setiap alternatif V_i (i = *1, 2, …, m*). Sebagai contoh, pada sisi 1-2, nilai *V* diperoleh dengan cara sebagai berikut.

$V_{1\text{-}2} = (w_{jarak} \times r_{jarak}) + (w_{SPBU} \times r_{SPBU}) + (w_{bengkel} \times r_{bengkel}) + (w_{masjid} \times r_{masjid}) = (0.1 \times 1) + (0.3 \times 1) + (0.5 \times 0) + (0.1 \times 1) = 0.1 + 0.3 + 0 + 0.1 = 0.500$.

Tabel 7. Nilai preferensi keseluruhan *V* pada skenario I

No	*W*	0,1	0,3	0,5	0,1	*V*
	Sisi	**Jarak**	**SPBU**	**Bengkel**	**Masjid**	
1	1-2	0,100	0,3	0	0,1	0,500
2	1-3	0,094	0,3	0,25	0	0,644
3	1-4	0,090	0,3	0	0,1	0,490
4	1-5	0,091	0,3	0,25	0	0,641
5	2-3	0,021	0,3	0	0,1	0,421
6	3-4	0,016	0,3	0,25	0,1	0,666
7	4-5	0,009	0,3	0,5	0	0,809
8	2-6	0,053	0	0,25	0,1	0,403
9	3-6	0,046	0,3	0	0	0,346
10	4-6	0,048	0,3	0,25	0,1	0,698
11	5-6	0,051	0	0,5	0	0,551

Bobot *w(e)* di Gambar 5 kemudian diperbarui nilainya dengan nilai *V* di Tabel 7. Gambar 6 menunjukkan graf berbobot *G* yang telah diperbarui yang menjadi dasar dalam pencarian rute terpendek.



Gambar 6. Graf berbobot *G* pada skenario I

Tabel 8. Penyelesaian algoritma Dijkstra pada skenario I

No	*i*	1	2	3	4	5	6
1	1	0_1	$0{,}500_1$	$0{,}644_1$	$0{,}490_1$	$0{,}641_1$	$\sim_1$
2	4	0_1	$0{,}500_1$	$0{,}644_1$	$0{,}490_1$	$0{,}641_1$	$1{,}188_4$
3	2	0_1	$0{,}500_1$	$0{,}644_1$	$0{,}490_1$	$0{,}641_1$	$0{,}903_2$
4	5	0_1	$0{,}500_1$	$0{,}644_1$	$0{,}490_1$	$0{,}641_1$	$0{,}903_2$
5	3	0_1	$0{,}500_1$	$0{,}644_1$	$0{,}490_1$	$0{,}641_1$	$0{,}903_2$
6	6	0_1	$0{,}500_1$	$0{,}644_1$	$0{,}490_1$	$0{,}641_1$	$0{,}903_2$

Tabel 8 menunjukkan penyelesaian algoritma Dijkstra dalam mencari rute terpendek dari titik 1 ke titik 6. Berdasarkan tabel tersebut, rute yang dihasilkan adalah **1 – 2 – 6.**

Dengan skenario bengkel "sangat penting", SPBU "sedang" dan kriteria lainnya "rendah", rute yang direkomendasikan adalah 1 – 2 – 6. Rute tersebut melewati dua sisi yaitu 1-2 dan 2-6 dimana pada sisi 1-2 terdapat dua bengkel dan di sisi 2-6 terdapat satu SPBU. Dengan demikian, rute yang direkomendasikan sudah sesuai dengan kebutuhan.

Skenario II

Pada skenario II, pengguna menetapkan SPBU sebagai "sangat penting", masjid sebagai "sedang" dan kriteria lainnya sebagai "rendah". Tingkat kepentingan tersebut kemudian dikonversi ke nilai persepsi dan bobot atribut sebagaimana yang terlihat di Tabel 9.

Tabel 9. Bobot atribut *W* skenario II

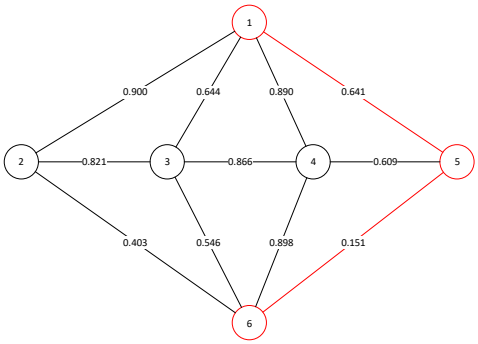
No	Kriteria	Nilai	*W*
1	Jarak	1	0,1
2	SPBU	5	0,5
3	Bengkel	1	0,1
4	Masjid	3	0,3
	Jumlah	10	1

Tabel 10 menunjukkan hasil perkalian antara matriks *R* dan bobot atribut *W* di Tabel 9 serta nilai preferensi keseluruhan untuk setiap alternatif V_i (i = *1, 2, …, m*).

Gambar 7 menunjukkan graf berbobot *G* yang telah diperbarui nilai di tiap sisinya *w(e)* menggunakan nilai *V* di Tabel 10.

Tabel 10. Nilai preferensi keseluruhan *V* pada skenario II

No	*W*	0,1	0,5	0,1	0,3	*V*
	Sisi	**Jarak**	**SPBU**	**Bengkel**	**Masjid**	
1	1-2	0,100	0,5	0	0,3	0,900
2	1-3	0,094	0,5	0,05	0	0,644
3	1-4	0,090	0,5	0	0,3	0,890
4	1-5	0,091	0,5	0,05	0	0,641
5	2-3	0,021	0,5	0	0,3	0,821
6	3-4	0,016	0,5	0,05	0,3	0,866
7	4-5	0,009	0,5	0,1	0	0,609
8	2-6	0,053	0	0,05	0,3	0,403
9	3-6	0,046	0,5	0	0	0,546
10	4-6	0,048	0,5	0,05	0,3	0,898
11	5-6	0,051	0	0,1	0	0,151



Gambar 7. Graf berbobot *G* pada skenario II

Tabel 11. Penyelesaian algoritma Dijkstra pada skenario II

No	*i*	1	2	3	4	5	6
1	1	0_1	$0{,}900_1$	$0{,}644_1$	$0{,}890_1$	$0{,}641_1$	$\sim_1$
2	5	0_1	$0{,}900_1$	$0{,}644_1$	$0{,}890_1$	$0{,}641_1$	$0{,}792_5$
3	3	0_1	$0{,}900_1$	$0{,}644_1$	$0{,}890_1$	$0{,}641_1$	$0{,}792_5$
4	6	0_1	$0{,}900_1$	$0{,}644_1$	$0{,}890_1$	$0{,}641_1$	$0{,}792_1$

Tabel 11 menunjukkan penyelesaian algoritma Dijkstra dalam mencari rute terpendek dari titik 1 ke titik 6 yang menghasilkan rute **1 – 5 – 6**.

Dengan skenario SPBU "sangat penting", masjid "sedang" dan kriteria lainnya "rendah", rute yang direkomendasikan adalah 1 – 5 – 6. Rute tersebut melewati dua sisi yaitu 1-5 dan 5-6 dimana pada sisi 1-5 terdapat satu masjid dan di sisi 5-6 terdapat satu SPBU. Dengan demikian, rute yang direkomendasikan sudah sesuai dengan kebutuhan.

KESIMPULAN DAN SARAN

Dalam penelitian ini, rute terbaik merupakan rute yang mempertimbangkan jarak terpendek dan ketersediaan fasilitas umum. Adapun kebutuhan yang dimaksud adalah preferensi pengguna terhadap fasilitas umum yang tersedia. Berdasarkan dua skenario yang telah dijalankan, rute-rute yang dihasilkan merupakan rute terbaik di masing-masing skenario yang dihitung berdasarkan algoritma Dijkstra dengan input jumlah tertimbang/*weighted sum*. Perbedaan rute di tiap-tiap skenario menunjukkan efektivitas bobot skala Likert dalam memberikan pengaruh dalam penghitungannya sehingga rute yang dihasilkan lebih mencerminkan preferensi pengguna.

Berdasarkan referensi [10], teknik normalisasi yang sesuai untuk metode pembobotan WS adalah normalisasi vektor dan *linear-max*. Untuk itu, guna melengkapi penelitian ini, disarankan untuk dilakukan penelitian lanjutan yang memvariasikan teknik normalisasinya dengan normalisasi vektor.

UCAPAN TERIMA KASIH

Terima kasih kepada seluruh pihak yang telah berperan mengembangkan aplikasi QGIS sehingga dapat dimanfaatkan dalam penelitian ini.

REFERENSI

[1] A. Zaki, "Algoritma Dijkstra : Teori dan Aplikasinya," *J. Mat. UNAND*, 2017, doi: 10.25077/jmu.6.4.1-8.2017.

[2] A. Noviriandini and M. Safitri, "Implementasi Algoritma Dijkstra Untuk Menentukan Jalur Terpendek Wilayah Pisangan Dan Kampus Nusa Mandiri Tangerang," *J. Pilar Nusa Mandiri*, vol. 13, no. 2, pp. 181–186, 2017, Accessed: Nov. 27, 2020. [Online]. Available: http://ejournal.nusamandiri.ac.id/index.php/pilar/article/view/235.

[3] S. Lestari, Ardiansyah, A. P. Giovani, and D. Dwijayanti, "A Dijkstra Algorithm Implementation in Determining Shortest Route to Mosque in Residential Citra Indah City," *J. Pilar Nusa Mandiri*, vol. 16, no. 1, pp. 65–70, Mar. 2020, doi: 10.33480/pilar.v16i1.1199.

[4] D. Wahyuningsih and E. Syahreza, "Shortest Path Search Futsal Field Location With Dijkstra Algorithm," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 12, no. 2, p. 161, Jul. 2018, doi: 10.22146/ijccs.34513.

[5] S. Sunardi, A. Yudhana, and A. A. Kadim, "Implementasi Algoritma Dijkstra Untuk Analisis Rute Transportasi Umum Transjogja Berbasis Android," *J. Sist. Inf. BISNIS*, vol. 9, no. 1, p. 32, May 2019, doi: 10.21456/vol9iss1pp32-38.

[6] N. Vafaei, R. A. Ribeiro, and L. M. Camarinha-Matos, "Normalization techniques for multi-criteria decision making: Analytical hierarchy process case

study," in *IFIP Advances in Information and Communication Technology*, 2016, vol. 470, pp. 261–268, doi: 10.1007/978-3-319-31165-4_26.

[7] G. O. Odu, "Weighting methods for multi-criteria decision making technique," *J. Appl. Sci. Environ. Manag.*, vol. 23, no. 8, p. 1449, Sep. 2019, doi: 10.4314/jasem.v23i8.7.

[8] B. Şahin, "Route Prioritization by using Fuzzy Analytic Hierarchy Process extended Dijkstra Algorithm," *J. ETA Marit. Sci.*, vol. 7, no. 1, pp. 3–15, 2019, doi: 10.5505/jems.2019.39306.

[9] E. Roghanian and Z. S. Kebria, "The combination of TOPSIS method and Dijkstra's algorithm in multi-attribute routing," *Sci. Iran.*, vol. 24, no. 5, pp. 2540–2549, Sep. 2017, doi: 10.24200/sci.2017.4390.

[10] S. Chakraborty, "A Simulation Based Comparative Study of Normalization Procedures in Multiattribute Decision Making," *6th WSEAS Int. Conf. Artif. Intell. Knowl. Eng. Data Bases*, vol. Proceeding, pp. 102–109, 2007.

[11] Y. D. Rosita, E. E. Rosyida, and M. A. Rudiyanto, "Implementation of dijkstra algorithm and multi-criteria decision-making for optimal route distribution," in *Procedia Computer Science*, Jan. 2019, vol. 161, pp. 378–385, doi: 10.1016/j.procs.2019.11.136.

[12] T. Kien Hua and N. Abdullah, "Weighted Sum-Dijkstra's Algorithm in Best Path Identification based on Multiple Criteria," doi: 10.20967/jcscm.2018.04.008.

[13] P. D. Sugiyono, *metode penelitian kuantitatif, kualitatif,dan R&D*. 2016.