Jurnal Mobile and Forensics (MF)
Vol. 2, No. 2, September 2020, pp. 84-98
P-ISSN: 2656-6257, E-ISSN: 2714-6685, DOI: https://doi.org/10.12928/mf.v2i2.2767

Simulasi Aplikasi *Real Time Route Selection* berbasis *Wireless Sensor Network* di Kota Makassar menggunakan Algoritma Dijkstra dan Floyd-Warshall

[1]Erick Alfons Lisangan, [2]Sean Coonery Sumarta, [3]Levi Oktavian Tandungan
[1,2,3] Fakultas Teknologi Informasi, Teknik Informatika, Universitas Atma Jaya Makassar
e-mail: erick_lisangan@lecturer.uajm.ac.id

Abstrak

Seiring dengan pertumbuhan kota yang secara di dunia melebihi 50%, persoalan kota menjadi lebih rumit dan kompleks, salah satunya adalah kemacetan yang dapat disebabkan oleh banjir serta pertumbuhan kendaraan yang melebihi luas jalanan. Saat ini sering terjadi kemacetan lalu lintas di beberapa ruas jalan di Kota Makassar, terutama pada saat peak hours dimana terjadi kenaikan jumlah kendaraan lebih dari 100% dibandingkan tahun 2007. Pada penelitian ini akan dirancang sebuah aplikasi real time route selection dengan memanfaatkan Wireless Sensor Network sebagai penyedia data traffic condition pada ruas jalan. Algoritma pencarian rute terpendek yang digunakan adalah algoritma Dijkstra serta algoritma Floyd-Warshall yang dikombinasikan dengan fungsi skalar Chebycheff. Hasil penelitian menunjukkan bahwa rute yang dihasilkan oleh kedua algoritma sama tetapi algoritma Dijkstra memiliki waktu pemrosesan lebih cepat dengan rerata 7,45 ms. Kelemahan fungsi skalar Chebycheff adalah proses update jarak antar node yang dinamis bergantung pada perubahan kondisi lalu lintas. Hal ini dapat diatasi dengan menggunakan penggunaan metode inferensi lain untuk kriteria kondisi lalu lintas, seperti fuzzy logic maupun metode Multi Criteria Decision Making.

As cities grow more than 50 percent in the world, urban problems are becoming more complicated and complex, one of which is congestion which can be caused by flooding and the growth of vehicles that exceed the area of roads. Currently, there are frequent traffic jams on several roads in Makassar City, especially during peak hours when the number of vehicles increased by more than 100% compared to 2007. In this study, a real time route selection application will be designed by utilizing the Wireless Sensor Network as a data provider for traffic conditions on roads. The shortest route search algorithm used is the Dijkstra algorithm and the Floyd-Warshall algorithm combined with the Chebycheff scalar function. The results show that the route generated by the two algorithms is the same but Dijkstra's algorithm has a faster processing time with an average of 7,45 ms. The weakness of Chebycheff's scalar function is the dynamic updating of the distance between nodes depending on changing traffic conditions. This can be overcome by using other inference methods for traffic condition criteria, such as fuzzy logic and the Multi Criteria Decision Making method.

Kata Kunci: *Real Time Route Selection*, *Wireless Sensor* Network, Algoritma Dijkstra, Algoritma Floyd-Warshall, Fungsi Skalar Chebycheff

PENDAHULUAN

Kemacetan merupakan salah satu masalah lalu lintas yang sering dihadapi kota padat penduduk, baik pada negara maju maupun berkembang, seperti Indonesia. Kemacetan telah menjadi salah satu bagian dari ciri khas pusat perkotaan dikarenakan rutin terjadi pada periode waktu-waktu puncak seperti yang biasa dikenal dengan jam pergi kantor, jam pulang kantor, akhir pekan dan hari libur. Dampak yang dihasilkan oleh kemacetan menimbulkan kerugian baik dari segi materi, waktu dan tenaga. Ditinjau dari aspek ekonomi, kemacetan menghambat proses produksi dan distribusi sehingga laju perekonomian menjadi terganggu. Dari aspek kesehatan, kemacetan

Received August 20, 2020; Revised September 10, 2020; Accepted September 23, 2020

mempengaruhi kondisi fisik dan psikis para pengguna lalu lintas, terlebih lagi bagi mereka yang kemudian melakukan berbagai aktivitas seperti bekerja, belajar dan lain sebagainya [1].

Makassar merupakan salah satu kota yang mengalami kemajuan yang pesat. Sebagai kota yang mengalami kemajuan pesat pasti memiliki beberapa masalah perkotaan, salah satu diantaranya adalah masalah kemacetan lalu lintas di jalan raya. Kemacetan timbul karena semakin tingginya volume kendaraan pribadi yang tidak diimbangi dengan pembangunan infrastruktur yang cepat dan kurang disiplinnya para pengendara dalam menggunakan kendaraannya. Menurut data terakhir Warta Ekonomi pada Juni 2017, jumlah kendaraan di kota Makassar adalah sebanyak 1.425.635. Jika dibandingankan dengan satu dekade sebelumnya pertumbuhan jumlah kendaraan itu sudah meningkat lebih dari seratus persen yang mana pada 2007 jumlah kendaraan itu adalah 613.315.

Berdasarkan permasalahan tersebut maka dibutuhkan pengelolaan yang lebih baik sehingga masyarakat kota tetap merasa nyaman dan aman dalam berkendara, salah satunya dengan menerapkan konsep *smart city* [2]. Saat ini telah banyak kota-kota besar telah memulai pengembangan *smart city*, seperti Seoul, New York, Tokyo, Shanghai, Singapura, Amsterdam, Cairo, Dubai, Kochi, dan Malaga [3]. Kota Makassar juga telah mencanangkan konsep Makassar Smart City yang dipadukan dengan kearifan lokal (Sombere).

Smart city dapat dikatakan sebuah kota yang memonitor dan mengintegrasikan kondisi seluruh infrastruktur yang penting, seperti jalan raya, jembatan, terowongan, rel, kereta bawah tanah, bandar udara, pelabuhan, komunikasi, air, listrik, dan bangunan utama, sehingga dapat lebih mengoptimalkan sumber daya yang dimiliki, rencana aktifitas pemeliharaan yang preventif, dan memantau aspek keamanan sekaligus memaksimalkan layanan kepada warganya [4]. *Smart city* dikembangkan dengan memperkenalkan *smart system* yang dirancang untuk kepentingan penduduk dan lingkungan [5].

Smart city tidak berhubungan secara langsung dengan salah satu teknologi atau sekumpulan teknologi tertentu. Sebaliknya, *smart city* lebih terkait dengan teknologi-teknologi yang dapat menghasilkan data mengenai kota dan juga memungkinkan interaksi pada setiap unsur-unsurnya [6]. Salah satu teknologi yang dapat diterapkan dalam infrastruktur *smart city* adalah *wireless sensor network* [5] [6]. Sebuah *wireless sensor network* adalah jaringan yang dibentuk oleh sejumlah besar *sensor node* dimana setiap node dilengkapi dengan sensor untuk mendeteksi fenomena fisik seperti cahaya, panas, tekanan, dan sebagainya [5]. Penerapan teknologi *wireless sensor network* di kota Makassar akan sangat membantu dalam merealisasi konsep *smart city* yang direncanakan. Salah satu contoh manfaat yang dapat diperoleh dengan penerapan *wireless sensor network* adalah *monitoring* informasi kemacetan dan banjir dengan memanfaatkan infrastruktur publik dan didistribusikan melalui internet dengan mengolah data dari sensor suhu, kandungan polusi udara, dan ketinggian air [7].

Penentuan rute terpendek tidak lepas dari penerapan algoritma dimana saat ini sudah digunakan dalam pembuatan suatu aplikasi atau sistem penentuan rute terpendek, seperti algoritma Dijkstra, Floyd-Warshall, dan algoritma lainnya [8]. Penerapan algoritma Dijkstra dan Floyd-Warshall terkait penentuan rute terpendek telah diteliti oleh banyak peneliti, seperti pada [9] dan [10].

Pada [9] menggunakan algoritma Dijkstra dalam penentuan rute terpendek dalam sistem informasi geografis di Timor Leste. Algoritma Dijkstra dalam sistem

tersebut dapat melakukan pencarian jalur terpendek dari titik awal ke titik tujuan dengan keakuratan nilai jarak rata-rata 0.03% dibandingkan hasil dari Google Earth. Penelitian [10] menggunakan algoritma Floyd-Warshall untuk mencari rute terpendek di kota Semarang. Algoritma Floyd-Warshall pada pembuatan program tersebut dapat menentukan rute terpendek dan dilakukan perbandingkan antara perhitungan manual dengan perhitungan yang dihasilkan oleh aplikasi. Hasil perbandingan tersebut menunjukkan bahwa terdapat hasil yang benar dan sama antara perhitungan manual dan aplikasi.

Pada penelitian ini akan dirancang sebuah simulasi aplikasi *real time route selection* yang merupakan sebuah aplikasi untuk memperoleh rute dari titik awal menuju titik tujuan yang diinginkan oleh pengguna dengan pemilihan rute berdasarkan kondisi lalu lintas secara *real time* [11],[12]. Aplikasi *real time route selection* dirancang dengan menggunakan algoritma Dijkstra dan Floyd-Warshall. Kedua algoritma akan dikolaborasikan dengan fungsi skalar Chebycheff untuk menentukan bobot keseluruhan jarak edge antar node dengan tidak hanya berdasarkan pada jarak tetapi juga pada kondisi lalu lintas yang diperoleh dari hasil pengumpulan data *wireless sensor network*. Penelitian ini disesuaikan dengan kondisi nyata suatu kemacetan pada beberapa titik di kota Makassar. Aplikasi navigasi yang terhubung dengan *wireless sensor network* pada jalur kemudian dikalkulasikan dengan bobot jarak setiap rute dengan fungsi skalar Chebysheff sehingga menghasilkan bobot jalur yang telah mempertimbangkan kondisi lalu lintas pada waktu tertentu.

Saat ini terdapat banyak aplikasi *real time route selection* yang banyak digunakan oleh masyarakat, salah satunya Google Maps. Google Maps memberikan informasi rute terpendek dari lokasi asal ke tujuan pengguna. Google Maps juga dapat menampilkan kondisi lalu lintas melalui dua jenis informasi. Pertama, *Historic Data* adalah data historis tentang waktu rata-rata yang diperlukan untuk melakukan perjalanan bagian jalan tertentu pada waktu tertentu pada hari-hari tertentu. Kedua, *Real Time Data* adalah data yang dikirimkan oleh banyak GPS *smartphone*, data ini melaporkan seberapa cepat suatu kendaraan melaju [13]. Kelemahan dari metode tersebut adalah sangat bergantung dengan lokasi GPS pengguna untuk kemudian diperhitungkan kecepatannya dan sangat bergantung pada data sebelumnya sehingga tidak sesuai dengan kondisi nyata.

Aplikasi pencari rute lain yang memanfaatkan data sensor yaitu TallyGo dimana memanfaatkan metal detector untuk mendeteksi kepadatan jalan di Amerika Serikat. Aplikasi pencarian rute yang akan dirancang dan disimulasikan pada penelitian ini berbeda dengan aplikasi lainnya dimana pemanfaatan *wireless sensor network* sebagai pengumpul data kondisi lalu lintas secara *real time* akan menjadi salah satu nilai masukan dalam menentukan rute terpendek yang akan dilalui oleh pengguna.

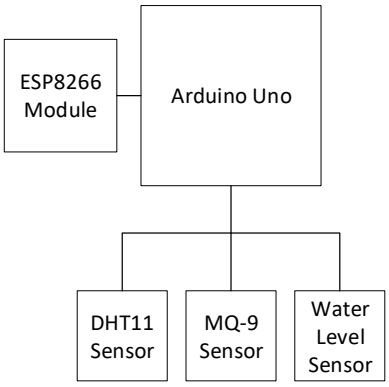
METODE PENELITIAN

Perancangan Wireless Sensor Network

Wireless Sensor Network (WSN) merupakan jaringan komputer terdistribusi (*Distributed Computer Network*) yang memanfaatkan sejumlah node sensor berukuran kecil, dikembangkan dan dikonfigurasikan dalam skala besar untuk membantu pemindaian terhadap lingkungan sekitar, memanfaatkan parameter pengukuran berupa temperatur, tekanan, suhu, gerakan, atau entitas lainnya yang diketahui oleh manusia [14]. Secara umum, *Wireless Sensor Network* (WSN) dapat didefinisikan sebagai salah satu jenis dari jaringan *wireless* (nirkabel) terdistribusi, yang

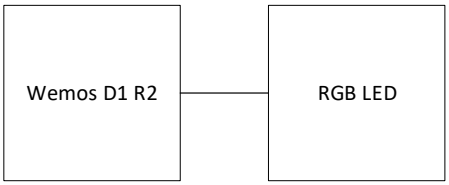
memanfaatkan teknologi *Embedded System* dan seperangkat node sensor, untuk melakukan proses sensor, *monitoring*, pengiriman data, dan penyajian informasi ke pengguna, melalui komunikasi di internet. Sensor meliputi banyak jenis, antara lain kelembaban, radiasi, temperatur, tekanan, mekanik, gerakan, getaran, posisi, dan lain-lain. Setiap jenis sensor memiliki perangkat lunak dan perangkat kerasnya masing-masing yang kemudian digabungkan dan dijalankan ke dalam sistem *Wireless Sensor Network* [15].

Simulasi pada penelitian ini melibatkan 2 (dua) *sensor node* dan 1 (satu) *gateway sensor node*. *Sensor node* merupakan *node* yang diletakkan pada beberapa wilayah dan memiliki tugas untuk menangkap keadaan wilayah tersebut melalui beberapa sensor, seperti sensor DHT11, MQ-9, dan water level Funduino. Setiap *sensor node* ketika terhubung ke *access point* melalui modul ESP8266 akan memperoleh alamat IP. Setiap *sensor node* akan tersimpan titik koordinat lokasinya ke dalam aplikasi di server [7]. Diagram blok sensor node dapat dilihat pada Gambar 1.



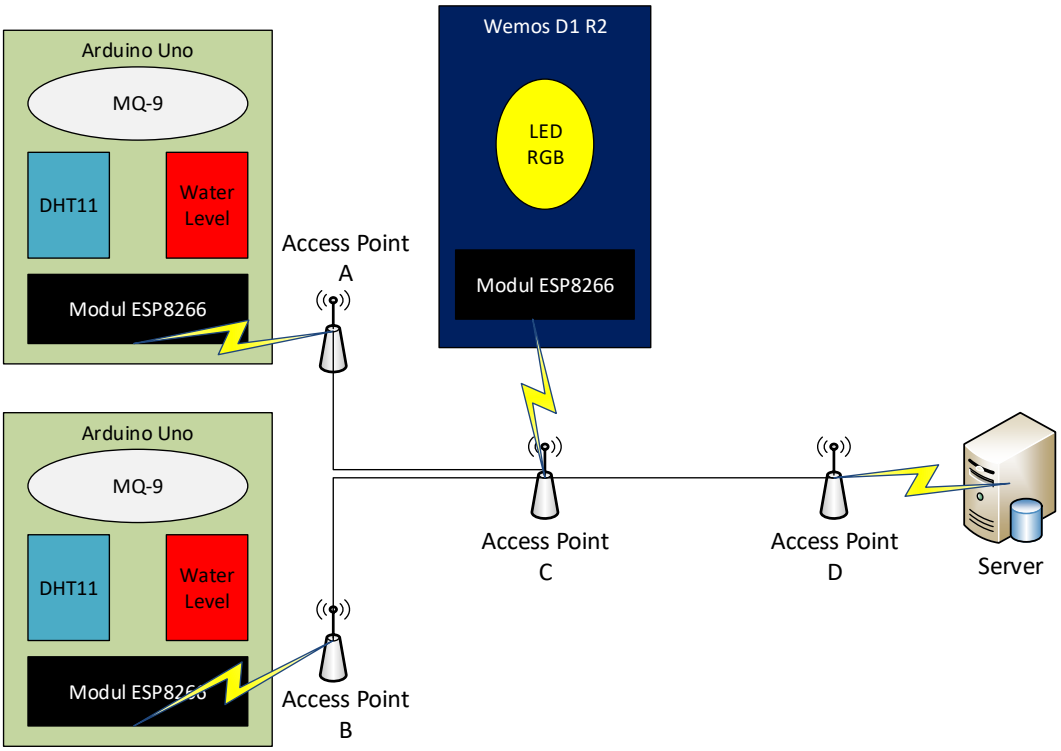
Gambar 1. Diagram Blok *Sensor Node*

Selain *sensor node*, terdapat juga *gateway sensor node* atau *sink node*. *Gateway sensor node* bertujuan sebagai pengumpul informasi dari beberapa *sensor node* dari suatu wilayah tertentu. Setelah mengumpulkan informasi, *gateway sensor node* kemudian menyediakan informasi kepada *server*. Pada *gateway sensor node* terdapat LED untuk menunjukkan apakah sedang terhubung dengan *server* atau tidak [7]. Diagram blok *gateway node sensor* dapat dilihat pada Gambar 2.



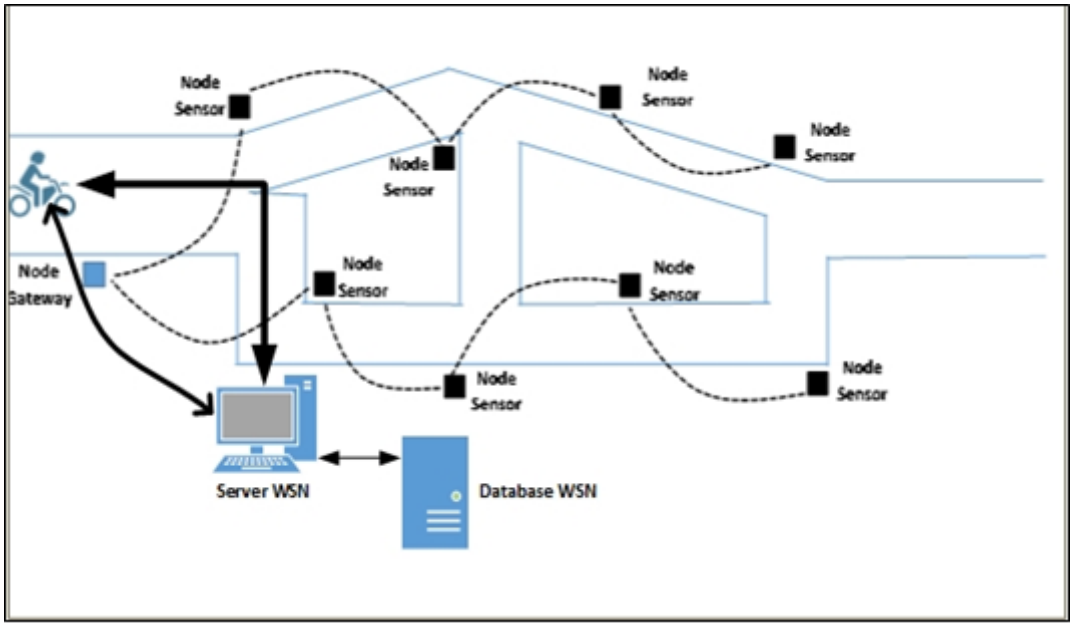
Gambar 2. Diagram Blok *Gateway Sensor Node*

Alur kerja dari *wireless sensor network* yang didesain melibatkan 2 (dua) *sensor node*, 1 (satu) *gateway sensor node*, 1 (satu) *server*, dan beberapa *access point*. *Access point* ini mensimulasikan pengiriman data *sensor* seluruh kota memanfaatkan layanan infrastruktur publik [7]. Alur kerja dari *wireless sensor network* dapat dilihat pada Gambar 3.



Gambar 3. Alur Kerja *Wireless Sensor Network*

Pada Gambar 3 dapat dilihat *sensor node* (Arduino Uno) akan terhubung ke *public access point* terdekat. Setiap *sensor node* akan memperoleh IP statis dari *access point* yang terhubung (Access Point A dan B). *Gateway sensor node* (Wemos D1 R2) kemudian terkoneksi ke *access point* yang lain (Access Point C). Asumsi yang dapat diperoleh berdasarkan hasil pengumpulan data adalah jaringan internet yang disediakan saling terhubung satu sama lain. *Sensor node* yang terkoneksi ke Access Point A dapat berkomunikasi dengan *sensor node* di Access Point B selama koneksi internet yang digunakan disediakan oleh *provider* internet yang sama. *Gateway sensor node* (Wemos D1 R2) bertugas untuk mengkoordinasikan data sensor dari *sensor node* terdekatnya. Pada *gateway sensor node* terdapat LED RGB yang bertujuan untuk memberikan petunjuk apakah sedang terjadi permintaan data atau tidak. Apabila server sedang melakukan permintaan data maka lampu akan berwarna hijau, jika tidak maka lampu akan berwarna merah. Ketika terjadi permintaan data dari server, *gateway sensor node* secara otomatis akan memanggil *web server* dari masing-masing *sensor node* yang dikoordinirnya.

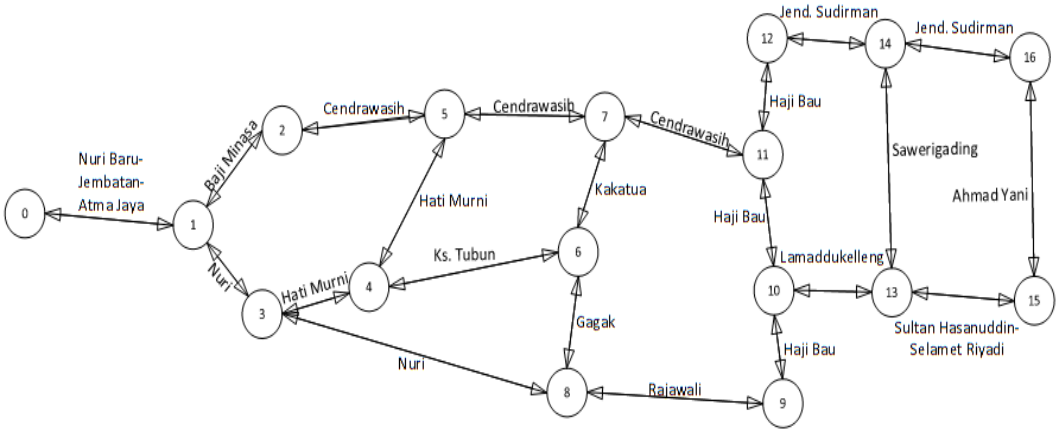


Gambar 4. Desain Pengambilan Data Kondisi Lalu Lintas

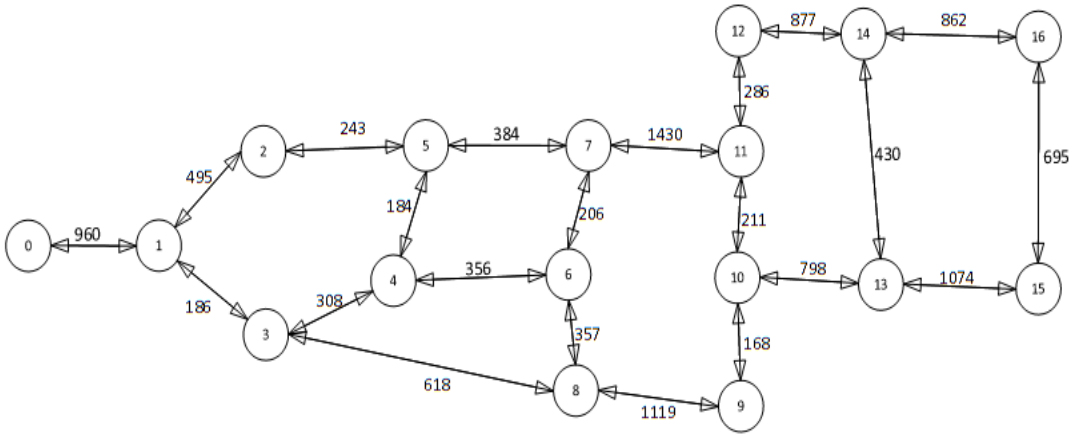
Pada Gambar 4 dapat dilihat gambaran peletakan titik *sensor node* yang berfungsi untuk menangkap kondisi kemacetan dalam variabel kadar CO_2, suhu udara, dan ketinggian air. Titik *sensor node* tersebut diletakkan pada beberapa ruas jalan dengan menggunakan tiang setinggi 1,5 meter. Seluruh *node sensor* dan *gateway sensor node* saling terhubung dengan menggunakan modul Wi-Fi ESP8266. ESP8266 merupakan modul wifi yang berfungsi sebagai perangkat tambahan mikrokontroler seperti Arduino agar dapat terhubung langsung dengan wifi dan membuat koneksi TCP/IP. Data dari *gateway sensor node* kemudian dikirimkan ke server untuk disimpan dalam *database*. Server juga bertugas untuk menghasilkan rute bagi pengguna berdasarkan titik awal dan titik akhir yang telah ditentukan dengan mempertimbangkan kondisi lalu lintas pada waktu tertentu.

Pembentukan Graf

Representasi dari titik lokasi yang akan disimulasikan diinterpretasikan dalam graf berarah dan berbobot yang dapat dilihat pada Gambar 5. Setiap *node* atau titik menunjukkan persimpangan jalan dan *edge* atau sisi menunjukkan ruas jalan yang menghubungkan antar persimpangan tersebut. *Edge* berarah dan ganda pada graf menunjukkan bahwa terdapat ruas jalan yang memiliki dua arah ruas jalan. Bobot *edge* pada graf menginterpretasikan jarak antara ruas jalan tersebut. Bobot jarak antar setiap node dalam graf diperoleh melalui Google Maps. Proses penentuan dilakukan dengan meletakkan titik node awal pada Google Maps serta titik node tujuan kemudian dilakukan pengambilan nilai jarak dalam satuan meter (m). Bobot jarak pada graf dapat dilihat pada Gambar 6.



Gambar 5. Pembentukan Graf Titik Lokasi



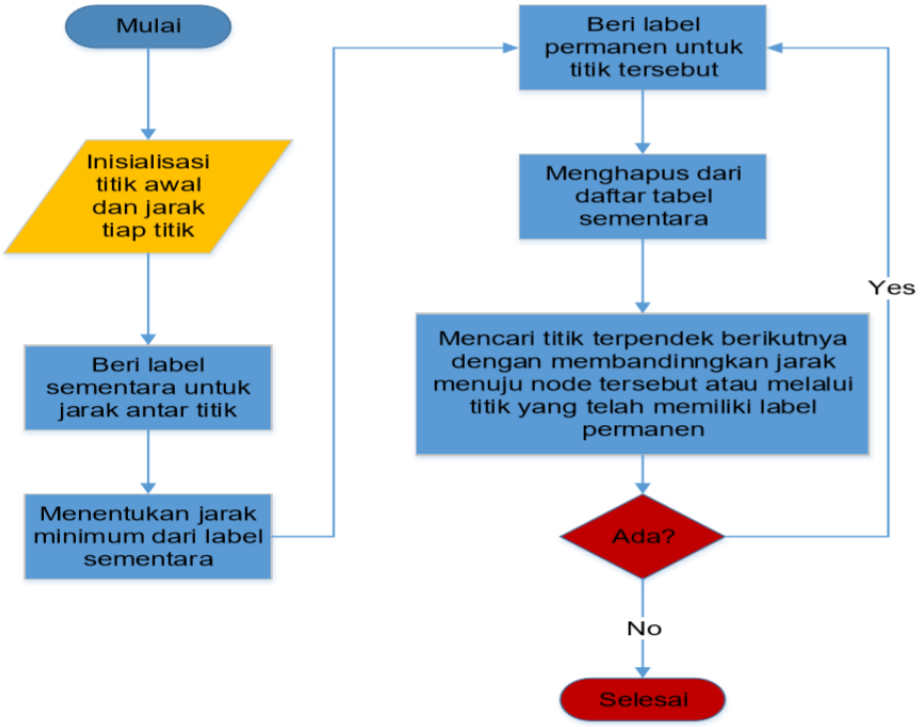
Gambar 6. Bobot Edge Setiap Node Titik Lokasi

Algoritma Dijkstra

Algoritma Dijkstra merupakan sebuah algoritma yang digunakan untuk menyelesaikan masalah pencarian jalur atau lintasan terpendek pada sebuah graf. Algoritma Dijkstra bekerja dengan mencari jalur atau lintasan antara dua *node* dalam sebuah graf berbobot dengan mencari jarak terpendek antara *node* awal dengan *node* lainnya sehingga lintasan yang terbentuk memiliki jumlah bobot terkecil [16]. Langkah-langkah algoritma Dijkstra adalah sebagai berikut (Gambar 7) [17], [18]:

1. Menetapkan jarak pada seluruh *node*. Jarak *node* awal dibuat nol sedangkan jarak *node* lainnya diberikan nilai yang cukup besar atau tidak terhingga.
2. Menandakan semua *node* dengan label belum terkunjungi. *Node-node* ini dikelompokkan menjadi satu. *Node* awal ditetapkan sebagai *node* sekarang.
3. Untuk *node* sekarang, hitung jarak dari semua *node* tetangga yang terhubung. Simpan jarak ini sebagai jarak sementara. Bandingkan jarak sementara ini dengan jarak yang tersimpan sebelumnya. Jika jarak sementara lebih kecil maka jarak sementara ini yang disimpan.
4. Setelah setelah menghitung jarak dari *node* sumber dengan *node* tetangga selanjutnya mengubah label *node-node* tersebut menjadi sudah terkunjungi. *Node* tersebut telah dihapus dari kumpulan *node-node* yang belum terkunjungi. *Node* yang berlabel sudah dikunjungi tidak akan dikunjungi atau dicek lagi.

5. Jika semua *node* telah berlabel sudah dikunjungi atau jarak sementara terkecil bernilai tak terhingga maka pencarian *node* lain dihentikan atau proses telah selesai. Jika tidak maka masih ada *node* yang berlabel belum terkunjungi atau jarak sementara terkecil berhingga, pilih *node* dengan jarak atau bobot terkecil sebagai *node* sekarang dan lanjutkan proses dari langkah ketiga.

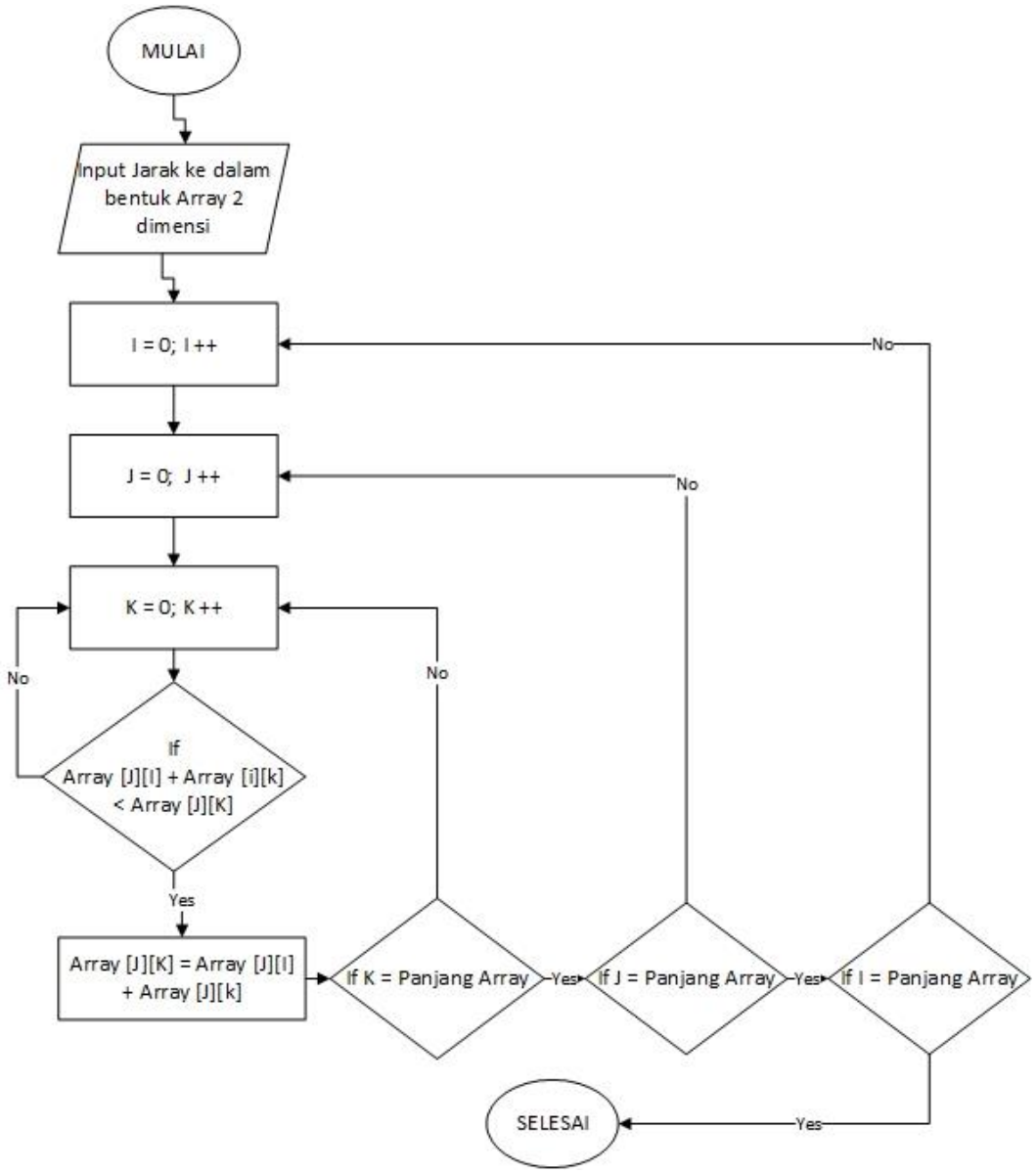


Gambar 6. *Flowchart* Algoritma Dijkstra

Algoritma Floyd-Warshall

Algoritma Floyd Warshall merupakan salah satu algoritma pencarian rute yang termasuk dalam kategori pemrograman dinamis, yaitu suatu metode yang melakukan pemecahan masalah dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait. Solusi-solusi yang terbentuk merupakan solusi dari tahapan sebelumnya dan memungkinkan terdapat lebih dari satu solusi [19].

Jika dibandingkan algoritma lain misalnya Dijkstra, hasil penelusuran jalur terpendek dari algoritma Floyd-Warshall lebih terjamin optimum. Hal tersebut dikarenakan algoritma Dijkstra tergolong jenis *greedy* yang hanya menentukan jarak terpendek dengan mengambil nilai optimum yang bersifat lokal dan tidak secara *global* sehingga nilai akhir yang ditemukan belum tentu merupakan nilai optimum atau jarak yang terpendek [20]. Dalam pencarian jalur menggunakan algoritma Floyd-Warshall tujuan akhir yang dicari adalah menemukan jalur terpendek dari semua kemungkinan pasangan titik selama masih terdapat jalur yang menghubungkan titik-titik tersebut. Algoritma Floyd-Warshall dapat dilihat pada Gambar 8.



Gambar 8. *Flowchart* Algoritma Floyd-Warshall

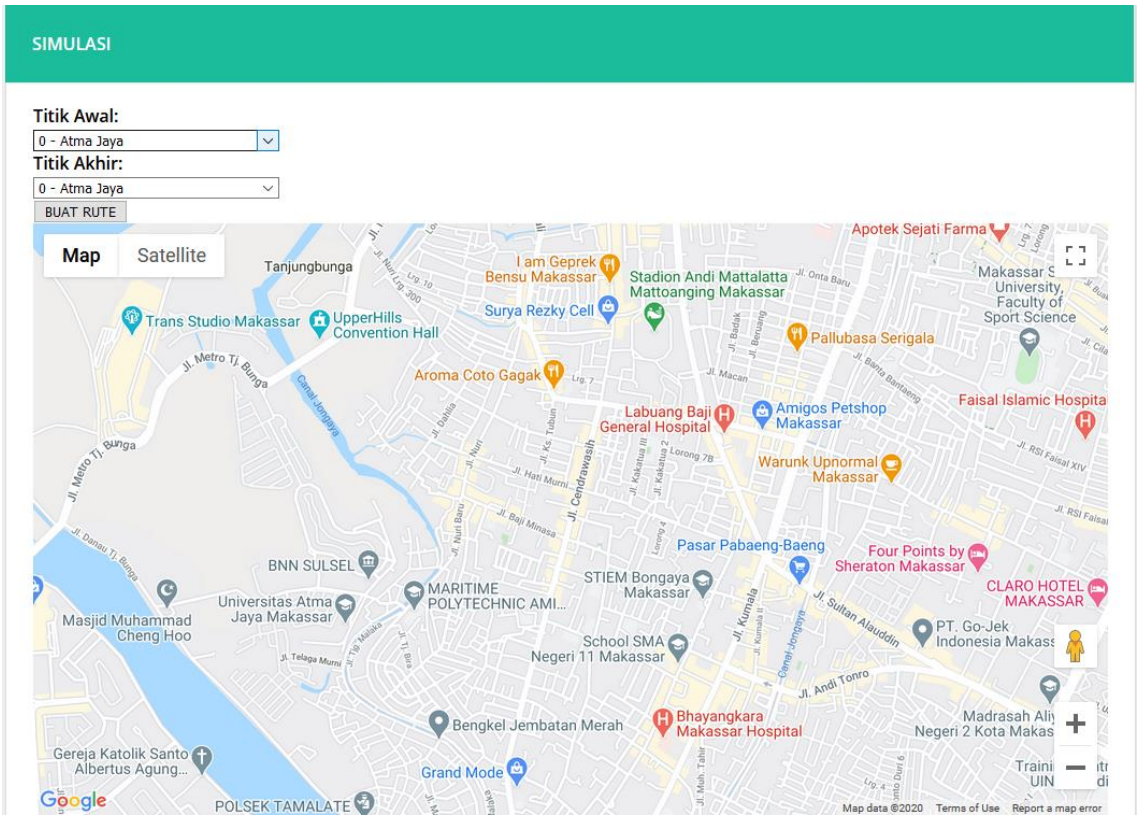
Fungsi Skalar *Chebycheff*

Fungsi skalar Chebycheff digunakan untuk mengatasi keterbatasan metode pemilihan rute untuk beberapa kriteria. Metode pemilihan rute pada umumnya hanya menggunakan satu kriteria nilai, sebagian besar jaraknya, sebagai fungsi objektif dalam menentukan rute yang optimal. Sedangkan penentuan rute berdasarkan kondisi lalu lintas membutuhkan lebih dari satu kriteria, seperti jarak, jumlah lalu lintas lampu, dan kriteria lainnya [12]. Fungsi skalar Chebycheff adalah fungsi untuk meringkas semua nilai obyektif kemudian menjadi nilai jarak antar lokasi [21].

Desain Aplikasi *Real Time Route Selection*

Aplikasi *real time route selection* dirancang dengan menggunakan bahasa pemrograman PHP serta tampilan peta dan rute memanfaatkan API Google Maps.

Pengguna dapat memilih titik awal serta titik akhir yang menjadi tujuan dari pengguna. Output dari aplikasi akan menampilkan rute yang dapat dilalui oleh pengguna untuk menuju lokasi titik tujuan. Tampilan aplikasi dapat dilihat pada Gambar 9. Pada Gambar 9 dapat dilihat bahwa pengguna terlebih dahulu memilih titik awal lokasi atau lokasi pengguna saat ini. Selain itu, pengguna juga perlu untuk memilih titik akhir atau lokasi tujuan yang hendak dituju. Setelah pengguna menginput titik awal dan akhir, sistem akan memberikan rute berdasarkan kondisi lalu lintas dari rute yang akan dilalui secara *real time*.



Gambar 9. Tampilan Aplikasi *Real Time Route Selection*

HASIL DAN PEMBAHASAN

Berdasarkan hasil rancangan aplikasi *real time route selection* berbasis *wireless sensor network* kemudian dilakukan simulasi dengan menggunakan algoritma Dijkstra maupun Floyd-Warshall. Simulasi dilakukan dengan terlebih dahulu mengumpulkan data kondisi lalu lintas pada 2 (dua) ruas jalan yang diwakili oleh ruas jalan antara node 1 dan 2 (1-2) serta node 1 dan 3 (1-3). Pengambilan data dilakukan dengan menggunakan 2 (dua) periode waktu yang berbeda, yaitu *peak hour* (12:24) dan *off-peak hour* (05:27). Pengambilan data untuk setiap *sensor node* dilakukan dengan interval setiap 20 detik. Representasi data kondisi lalu lintas dapat dilihat pada Tabel 1 dan Tabel 2.

Tabel 1. Representasi data kondisi lalu lintas node 1-2 (11 September 2019)

Waktu	Data Sensor Ruas Jalan Node 1 - 2		
	Kadar CO_2 (ppm)	***Suhu (^oC)***	***Ketinggian Air (mm)***
05:27:15	232	30	7
05:27:34	217	30	5
05:27:54	225	30	8
....	.....	....	....
12:24:30	878	32	5
12:24:50	897	33	2
12:25:09	892	32	0
....	....	....	....

Tabel 2. Representasi data kondisi lalu lintas node 1-3 (11 September 2019)

Waktu	Data Sensor Ruas Jalan Node 1 - 3		
	Kadar CO_2 (ppm)	***Suhu (^oC)***	***Ketinggian Air (mm)***
05:27:15	98	30	10
05:27:34	91	30	5
05:27:54	86	30	5
....	....	....	....
12:24:30	476	31	2
12:24:50	489	31	3
12:25:09	462	31	5
....	....	....	....

Setelah dilakukan pengambilan data, kemudian dilakukan simulasi *real time route selection* dengan menggunakan algoritma Dijkstra dan Floyd-Warshall. Seluruh data kondisi lalu lintas dan jarak antar node kemudian dijadikan sebagai input pada fungsi skalar Chebycheff atau dilakukan penjumlahan keseluruhan data kondisi lalu lintas dan jarak antar node yang kemudian menjadi nilai bobot jarak antara setiap node (Tabel 3). Pada Tabel 3 dapat dilihat bahwa hanya terdapat sebuah nilai yang konstan, yaitu nilai jarak. Nilai bobot baru akan senantiasa berubah sesuai dengan perubahan kondisi lalu lintas yang diperoleh dari *wireless sensor network* sehingga bobot jarak antar *node* akan senantiasa berubah seiring dengan perubahan yang terjadi pada kondisi lalu lintas. Hasil penentuan rute dengan menggunakan algoritma Dijkstra dan Floyd-Warshall berdasarkan kondisi lalu lintas secara *real time* dapat dilihat pada Tabel 4.

Tabel 3. Penerapan Fungsi Skalar Chebycheff pada Node 1-3

Waktu	Bobot Baru Ruas Jalan Node 1 - 3				
	Bobot Lama Jarak (m)	***Kadar CO_2 (ppm)***	***Suhu (^oC)***	***Ketinggian Air (mm)***	***Nilai Bobot Baru (fungsi Chebycheff)***
05:27:15	186	98	30	10	324
05:27:34	186	91	30	5	312
05:27:54	186	86	30	5	307
....	....	....	....	....	....
12:24:30	186	476	31	2	695
12:24:50	186	489	31	3	709
12:25:09	186	462	31	5	684
....	....	....	....	....	....

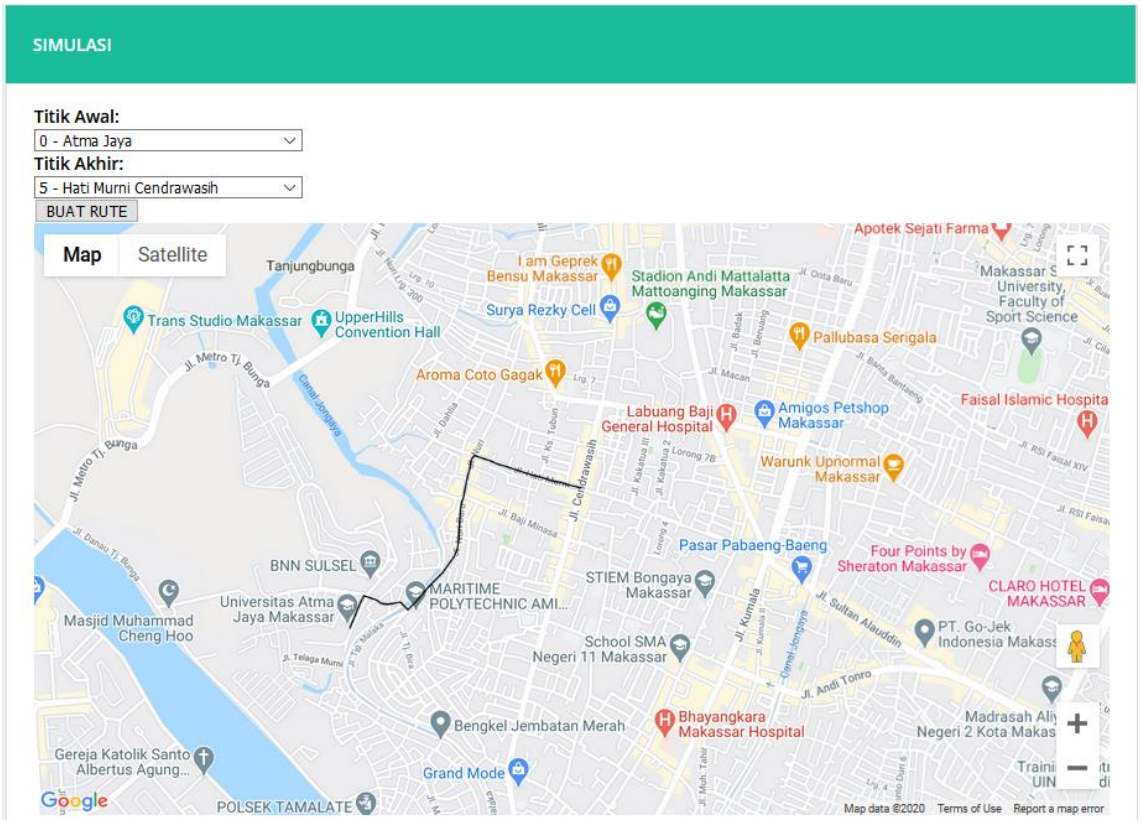
Tabel 4. Hasil Penentuan Rute

Origin/ Destination (O/D)	Algoritma Dijkstra		Algoritma Floyd-Warshall	
	Jalur	**Kecepatan Proses (ms)**	**Jalur**	**Kecepatan Proses (ms)**
0/5	0-1-3-4-5	6.77	0-1-3-4-5	7.04
0/16	0-1-3-8-9-10-13-14-16	8.27	0-1-3-8-9-10-13-14-16	8.92
1/11	1-3-8-9-10-11	7.42	1-3-8-9-10-11	9.26
3/14	3-8-9-10-13-14	7.34	3-8-9-10-13-14	8.51
Rerata		**7.45**		**8.43**

Tampilan hasil rute dari O/D = 0/5 dapat dilihat pada Gambar 10. Apabila melihat pada graf (Gambar 6) maka terdapat 2 (dua) kemungkinan rute yang dapat dilalui untuk menuju titik 5 dari titik 0, yaitu 0-1-2-5 dan 0-1-3-4-5. Hasil dari algoritma Dijkstra dan Floyd-Warshall menunjukkan bahwa rute 0-1-3-4-5 merupakan rute yang terpendek dengan mempertimbangkan jarak serta kondisi lalu lintas pada saat dilakukan pencarian rute. Hal ini sesuai dengan prinsip kerja kedua algoritma yang mencari jalur dengan bobot terpendek atau terkecil dibandingkan jalur yang lain.

Pada Tabel 4 dapat dilihat bahwa kedua algoritma menghasilkan jalur yang sama tetapi kecepatan proses algoritma Dijkstra lebih cepat dibandingkan algoritma Floyd-Warshall. Hal ini dapat disebabkan karena pada algoritma Floyd-Warshall terdapat proses penelusuran jarak terpendek dengan menghitung semua pasangan titik yang mungkin dan kemudian menyimpan jalur yang dilewati dari semua pasangan titik tersebut. Proses tersebut dapat menyebabkan waktu pemrosesan yang lebih lama dibandingkan algoritma Dijkstra. Kekurangan yang dimiliki oleh penggunaan fungsi skalar Chebycheff adalah nilai bobot jarak antar node yang senantiasa berubah-ubah bergantung pada nilai sensor yang menjadi inputan dari fungsi. Ketika terjadi perubahan nilai sensor maka nilai bobot jarak antar node akan senantiasa di-*update* untuk menyesuaikan dengan nilai keadaan kondisi lalu lintas pada waktu tertentu. Hal ini dapat diatasi dengan mengkolaborasikan algoritma penentuan rute dengan metode

inferensi lain untuk kriteria kondisi lalu lintas, seperti *fuzzy logic* maupun metode *Multi Criteria Decision Making*.



Gambar 10. Tampilan Rute Dari Titik O/D = 0/5

KESIMPULAN

Kesimpulan yang dapat diperoleh dari penelitian ini adalah algoritma Dijkstra dan algoritma Floyd-Warshall mampu menghasilkan rute yang sama tetapi algoritma Dijkstra memiliki kecepatan pemrosesan yang lebih cepat dengan nilai rata-rata sebesar 7,45 ms. Fungsi skalar Chebycheff dapat menjadi salah satu alternatif kolaborasi dengan algoritma penentuan rute terpendek untuk studi kasus multi kriteria tetapi sangat bergantung dengan nilai inputan data sensor dan sangat mempengaruhi penentuan rute. Hal ini dapat diatasi dengan menggunakan penggunaan metode inferensi lain untuk kriteria kondisi lalu lintas, seperti *fuzzy logic* maupun metode *Multi Criteria Decision Making*.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Direktorat Riset dan Pengabdian Masyarakat (DRPM) Kementerian Riset, Teknologi, dan Pendidikan Tinggi atas dukungan yang diberikan kepada penulis berupa bantuan dana penelitian dalam skema Penelitian Dosen Pemula.

DAFTAR PUSTAKA

[1] L. Malihah, R. Marawati, dan F. Agustina, "Aplikasi Algoritma Ant Dispersion Routing (DR) Untuk Penyelesaian Masalah Penyebaran Rute Lalu Lintas Sebagai Upaya Untuk Mengurangi Kemacetan," *Jurnal EurekaMatika*, vol. 2,

no. 1, pp. 79–97, 2014.

[2] S. Supangkat, "Smart City Indonesia," *Smart City ID*, vol. 1, no. 1, pp. 1–5, 2015.

[3] J. S., Hwang, dan Y. H., Choe, "Smart Cities Seoul: A Case Study", *ITU-T Technology Watch*, 2013.

[4] R. E. Hall, B. Bowerman, J. Braverman, J. Taylor, dan H. Todosow, "The vision of a Smart City," *2nd International Life Extension Technology Workshop*, 2000.

[5] International Electrotechnical Commission, "*Internet of Things: Wireless Sensor Networks*", Geneva: White Paper, 2014.

[6] T. Gea, dan J. Paradells, "Smart cities as an application of Internet of Things: Experiences and lessons learnt in Barcelona," *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2013, pp. 552-557.

[7] E. A. Lisangan, dan S. C. Sumarta, "Proposed Prototype and Simulation of Wireless Smart City: Wireless Sensor Network for Congestion and Flood Detection in Makassar," *2018 2nd East Indonesia Conference on Computer and Information Technology (EIConCIT)*, pp. 72-77, 2018.

[8] X. Xu, "Speedy algorithm of public taffic route selection based on adaptive backbone network", *Computer and Information Science*, vol. 1, no. 1, pp. 12-16, 2008.

[9] A. Gusmão, dan S. H. Pramono, "Sistem Informasi Geografis Pariwisata Berbasis Web Dan Pencarian Jalur Terpendek Dengan Algoritma Dijkstra," *Jurnal EECCIS (Electrics, Electronics, Communications, Controls, Informatics, Systems)*, vol. 7, no. 2, pp. 125–130, 2013.

[10] A. Khamami, dan R. Saputra, "The shortest path search application based on the city transport route in Semarang using the Floyd-warshall algorithm," *Journal of Physics: Conference Series*, 2019, pp. 1-7, 2019.

[11] E. A. Lisangan, "Route Selection Berdasarkan Hasil Profiling Real Time Traffic Condition," Seminar Nasional Sistem Informasi dan Teknologi Informasi 2016, Makassar, 2016.

[12] E. A. Lisangan, dan S. C. Sumarta, "Route Selection based on Real Time Traffic Condition using Ant Colony System and Fuzzy Inference System," *2017 3rd International Conference on Science in Information Technology (ICSITech)*, pp. 66-71, 2017.

[13] M. D. Al-Amin, dan M. Uddin, "*Real Time Traffic Monitoring System Using Crowd Sourced GPS Data*," Thesis Report BRAC University, 2015.

[14] M. K. Jha, dan T.P. Sharma, "Secure Data aggregation in Wireless Sensor Network: A Survey," *International Journal of Engineering Science and Technology,* vol. 3, no. 3, pp. 2013-2019, 2011.

[15] I. P. A. E. Pratama, *Wireless Sensor Network*. Bandung: Informatika, 2015.

[16] E. Ismantohadi, dan I. Iryanto, "Penerapan Algoritma Dijkstra untuk Penentuan Jalur Terbaik Evakuasi Tsunami – Studi Kasus: Kelurahan Sanur Bali," *Jurnal Teknologi Terapan*, vol. 4, no. 2, pp. 72-78, 2018.

[17] T. H. Cormen, C. E. Leiserson, dan R. L. Rivest, *Introduction to Algorithms , Second Edition*. MIT Press, 2001.

[18] Y. Deng, Y. Chen, Y. Zhang, dan S. Mahadevan, "Fuzzy Dijkstra algorithm for

shortest path problem under uncertain environment," *Applied Soft Computing*, vol. 12, no. 3, pp. 1231-1237, 2012.

[19] Y. Darnita, R. Toyib, dan R. Rinaldi, "Implementasi Algoritma Floyd Warshall untuk Menentukan Letak dan Lokasi Perusahaan Travel/Rental Mobil di kota Bengkulu," *Jurnal Pseudocode*, vol. 4, no. 2, pp. 144-156, 2017.

[20] R. A. D. Novandi, "P*erbandingan Algoritma Dijkstra dan Algoritma Floyd-Warshall dalam Penentuan Lintasan Terpendek (Single Pair Shortest Path)*," Makalah Strategi Algoritmik, 2013.

[21] T. Lust and J. Teghem, "The multiobjective traveling salesman problem: a survey and a new approach", *Advances in Multi-Objective Nature Inspired Computing Studies in Computational Intelligence*, vol. 272, pp. 119-141, 2010.