Institut Riset dan Publikasi Indonesia (IRPI)
MALCOM: Indonesian Journal of Machine Learning and Computer Science
Journal Homepage: https://journal.irpi.or.id/index.php/malcom
Vol. 4 Iss. 2 April 2024, pp: 607-613
ISSN(P): 2797-2313 | ISSN(E): 2775-8575

Implementation of the Best Route Search to Find Out the Location of Parking Places in the E-Parking System Using the Dijkstra Algorithm and Best First Search

Implementasi Pencarian Rute Terbaik untuk Mengetahui Lokasi Tempat Parkir pada Sistem E-Parking Menggunakan Algoritma Dijkstra dan Best First Search

Nava Gia Ginasta[1*], Supriady[2]

[1,2]Program Studi S1 Bisnis Digital, Fakultas Logistik Teknologi dan Bisnis,
Universitas Logistik dan Bisnis Internasional, Bandung, Indonesia

E-Mail: [1]navagia@ulbi.ac.id, [2]supriady@ulbi.ac.id

Received Dec 27th 2023; Revised Feb 07th 2024; Accepted Mar 10th 2024
Corresponding Author: Nava Gia Ginasta

Abstract

Searching for the best route is the problem of finding the best route from the starting point to the destination point of the parking lot. Using an algorithm that can be used to solve a problem finding the best route is the Dijkstra Algorithm. Dijkstra's algorithm is used to find the best route that will be taken by parking space seekers to store their vehicles. Selecting the best route with the Dijkstra algorithm and Best First Search (BFS), Best First Search (BFS) is allowed to search to visit a node at a lower level, if a node at a higher level has a bad value, there are 10 location object points parking block, from the entry location point to the destination parking block location. To speed up travel time and the direction of the destination that has been determined by the Dijkstra Algorithm, the parking space searcher optimizes the travel distance to the destination location so that the time needed is more efficient. Apart from that, storing vehicles in the parking lot will be faster because the destination path for the vehicle to be stored has been determined.

Keyword: Best First Search, Dijkstra Algorithm, Parking Lot, Route

Abstrak

Pencarian rute terbaik yaitu untuk permasalahan mencari sebuah rute terbaik dari titik awal ke titik tujuan tempat parkir. Dengan menggunakan algoritma yang dapat digunakan untuk menyelesaikan suatu masalah pencarian rute terbaik adalah Algoritma Dijkstra. Algoritma Dijkstra digunakan untuk mencari rute terbaik yang akan dilalui oleh pencari tempat parkir untuk menyimpan kendaraannya. Pemilihan rute terbaik dengan algoritma dijkstra dan Best First Search (BFS), Best First Search (BFS) diperbolehkan dalam mencari untuk mengunjungi suatu node pada levelnya yang lebih rendah, jika node pada levelnya lebih tinggi maka memiliki nilai tidak baik, terdapat 10 titik objek lokasi blok parkir, dari titik lokasi tempat masuk ke lokasi blok parkir tujuan. Untuk mempercepat waktu tempuh dan arah tujuan yang sudah ditentukan oleh Algoritma Dijkstra maka pencari tempat parkir untuk mengoptimalkan jarak tempuh menuju lokasi tujuan sehingga dapat mengefesiensi waktu yang dibutuhkan. Selain itu penyimpanan kendaraan pada tempat parkir akan lebih cepat karena sudah ditentukan jalur tujuan kendaraan yang akan disimpan.

Kata Kunci: Algoritma Dijkstra, Best First Search, Rute, Tempat Parkir

1. PENDAHULUAN

Parkir yaitu keadaan kendaraan tidak bergerak yang tidak bersifat sementara [1]. Pada saat parkir kendaraan yang berhenti ditempat tertentu yang sesuai dengan rambu lintas ataupun tidak, menyimpan kendaraan bukan untuk menaikkan dan atau menurunkan orang atau barang dapat bilang sebagai kegiatan parkir. Kegiatan parkir ini, sudah diatur dalam peraturan pemerintah atau undang-undang dasar (UUD) yang salah satunya yaitu pemanfaatan pada fasilitas parkir. Pada saat ini, parkir yang dipakai adalah manajemen konvensional. Manajemen konvesional yaitu pencatatan transaksi pada parkir dilakukan karcis retribusi manual tanpa tercatat pada sistem ataupun aplikasi.

Proses yang dijalankan dalam mencari lokasi tempat parkir untuk menyimpan kendaraan yaitu memutari lokasi tempat parkir. Setiap melakukan parkir kendaraan, proses mencari lokasi parkir yang konvensional yaitu memutar-mutar untuk mencari tempat parkir, sehingga bisa mengganggu kendaraan lain yang akan parkir dan tidak efisien dalam segi waktu karena harus mencari tempat parkir yang kosong. Dalam sistem parkir dapat diusulkan sistem yang bisa menunjukan ke suatu tempat parkir yang dituju untuk dapat ditempuh melalui beberapa lintasan yang tidak efesien untuk menyimpan kendaraan. Dalam hal ini, akan menentukan fitur untuk mengklasifikasikan lokasi tempat parkir. Selain itu jalan manakah yang harus dilalui sehingga dapat mencari tempat tujuan untuk menyimpan kendaraan dengan jarak terbaik.

Algoritma Dijkstra yang dapat menyelesaikan masalah pencarian rute yang efesien dengan menggunakan *graf* pada simpul yang bernilai tidak negatif. Algoritma Dijkstra yaitu algoritma yang sering digunakan untuk menyelesaikan masalah yang berkaitan dengan suatu optimasi [2]. Dalam pencarian tempat parkir untuk menyimpan kendaraan dari mulai pertama kali masuk ke tempat parkir harus mencari tujuan untuk tempat kendaraan itu disimpan yang optimal untuk optimasi baik rute maupun efesiensi waktu yang ditempuh untuk sampai ke blok tempat parkir membutuhkan waktu (±) 5 menit. Memilih rute dengan algoritma Dijkstra digunakan algortima *Best First Search (BFS)*, *Best First Search (BFS)* dimana pencarian rute boleh mengunjungi *node* pada level rendah jika *node* pada level tinggi memiliki nilai tidak baik, dapat digunakan untuk menyelesaikan masalah pencarian rute terbaik untuk tujuan tempat parkir. Selain itu Algoritma Dijkstra dapat menentukan rute yang menghubungkan tempat tempat yang dituju. Maka akan memudahkan pencari tempat parkir untuk menyimpan kendaraannya dengan rute yang terbaik, pencari tempat parkir tidak perlu lagi mencari tempat untuk menyimpan kendaraannya.

Penelitian Yosdarso Afero (2021), berjudul *"*Algoritma Best First Search Menentukan Lintasan Jalur Terpendek Pada Kota Wisata Bukittinggi" menjelaskan tentang kelebihan dari Algoritma *Best First Search* yaitu untuk memudahkan dalam menemukan solusi mengukur suatu jarak dari titik yang akan dikunjungi, dengan menggunakan algoritma *Best First Search* ini akan menemukan solusi yang optimal. Penggunaan *Best First Search* pada Sistem Informasi geografis bisa dijadikan sebagai jalur untuk membantu dalam Pengambilan sistim keputusan [3]. Penelitian Syaiful Ahdana, Setiawansyah (2020), berjudul "Pengembangan Sistem Informasi Geografis Untuk Pendonor Darah dengan Algoritma Dijkstra berbasis Android" menjelaskan tentang kontribusi dalam penelitian ini yaitu untuk merancang sebuah sistem yang bisa menemukan pendonor di darah wilayah Bandar lampung menggunakan teknologi geolokasi dan algoritma dijkstra untuk menentukan rute terpendek [4]. Penelitian Zepri Pratama, Dedy Hartama (2020), berjudul "Penerapan Metode Dijkstra untuk Menentukan Jalur Lintasan Terpendek Kota Kisaran Menuju Objek Wisata Simalungun" menjelaskan tentang menentukan sebuah jalur pada lintasan terdekat dari kota Kisaran (Titik Awal) yang menuju salah satu Objek tempat Wisata di daerah Simalungun yang mengambil delapan belas sampel dari destinasi wisata. Metode yang digunakan pada makalah ini yaitu metode algoritma *Djikstra* yang digunakan para peneliti untuk menentukan rute jalur tercepat pada sebuah perjalanan [5]. Perbedaan dalam penelitian terdahulu dengan penelitian yang dibuat yaitu algoritma Dijkstra dalam untuk menerapkan pencarian rute terbaik dalam pemodelan *graph* jaringan jalan di lokasi blok tempat parkir. Pemilihan jalur dalam algoritma Dijkstra dilakukan menggunakan algoritma Best First Search (BFS), BFS digunakan untuk pencarian rute yang diperbolehkan mengunjungi *node* awal yang levelnya lebih rendah jika *node* yang levelnya lebih tinggi mempunyai nilai tidak baik.

2. METODOLOGI PENELITIAN

2.1. Algoritma Dijkstra

Algoritma Dijkstra digunakan untuk mencari jalur terpendek. Algoritma Dijkstra yaitu algoritma untuk pencarian jalur pendek berdasarkan *edge* yang terkecil dari lokasi ke lokasi wisata tempat kuliner yang dituju. Data wisata kuliner diperoleh jalan dari *Google Earth* atau *Google maps*. Penggunaan metode algoritma Dijkstra untuk memberikan solusi menyampaikan keluaran berupa jalur tercepat [6]. Algoritma Dijkstra digunakan untuk menentukan rute yang terpendek menggunakan *graph* berarah atau *graph* tak berarah untuk memiliki bobot secara eksplinsit dari semua rute alternatif yang mungkin menjadi solusi rute optimal. Prinsip dari algoritma dijkstra untuk menentukan suatu rute terpendek pada masalah jaringan yang dimodelkan dalam *graph* adalah pada waktu penentuan rute alternatif yang kemungkinan menjadi solusi, setiap bobot dari vertek yang belum dipilih akan dianalisis, lalu dipilih vertek dengan bobot yang paling kecil [7]. Algoritma dijkstra merupakan salah satu jenis algoritma yang sering dipakai dalam pemecahan persoalan yang terkait dengan masalah optimasi dan bersifat sederhana bila dibandingkan dengan algoritma lainnya [9].

Algoritma Dijkstra dapat memecahkan suatu masalah pencarian jalur dengan menggunakan suatu graf pada setiap simpul yang bernilai tidak negatif. Algoritma dijkstra untuk menyelesaikan persoalan pencarian rute terpendek ataupun lintasan terpendek dari satu *vertex* ke *vertex* lainnya pada suatu *graph* berbobot, jarak antara *vertex* adalah nilai bobot dari setiap *edge* pada *graph* [10].

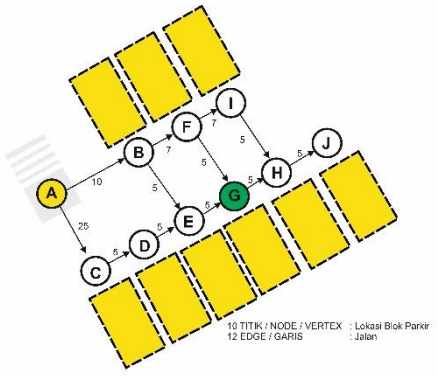
Dalam mencari jalur terpendek dengan algoritma dijkstra dengan mencari nilai yang minimal dari suatu *graf* berbobot, jarak terdekat akan diperoleh dari atau lebih dari suatu titik graf dan nilai total yang didapat adalah yang bernilai kecil.

Misalkan G pada graf berarah berlabel pada titik *V(G) = {v1,v2,...,vn}* dengan *path* terpendek yang dicari yaitu dari *v1* ke *vn*. Algoritma ini dimulai dengan titik *v1*. Algoritma akan mencari satu titik yang jumlah bobotnya dari titik 1 lebih kecil. Titik yang terpilih dipisahkan, titik tersebut diabaikan dalam iterasi selanjutnya[11]. Adapunt langkah-langkah dalam menentukan lintasan terdekat pada algoritma dijkstra yaitu:

1. Dengan awalnnya menentukan *node* yang bersumber sebagai *node* awal, diinisialkan dengan '1'.
2. Bentuk data pada tabel yang terdiri dari *node*, status, bobot, dan *predecessor*. Dengan melengkapi kolom bobot yang diperoleh dari jarak *node* ke semua *node* yang langsung terhubung dengan *node* sumbernya.
3. Jika *node* sumber ditemukan maka *node* akan dipilih.
4. Selanjutnya tetapkan *node* dipilih menggunakan label permanen dan perbaharui *node* yang langsung terhubung.
5. Selanjutnya tentukan *node* sementara untuk yang terhubung pada *node* yang sudah dipilih sebelumnya dan merupakan bobot yang terkecil dilihat dari data tabel dan dapat ditentukan sebagai *node* terpilih berikutnya.
6. Jika *node* yang dipilih merupakan suatu *node* tujuan, maka kumpulan *node* terpilih atau *predecessor* merupakan rangkaian lintasan terpendek.

2.2. *Graph*

Graph yaitu model-model yang berguna untuk diimplementasikan sangat yang luas. Walaupun teori *graph* dari bidang ilmu matematika, tapi pada penerapannya, teori *graph* bisa digunakan diberbagai bidang ilmu dan juga kehidupan sehari-hari [12]. Graf untuk merepresentasikan suatu objek-objek yang diskrit dan hubungan dari objek-objek tersebut. Dengan diberikannya peta, maka dapat mengetahui apakah ada lintasan antara jalan dua buah kota. Selain itu, bila jalan panjang kereta api antara dua buah kota yang bertetangga diketahui, maka dapat ditentukan rute perjalanan ditiadakan dari kota A ke kota B [13]. Graf yang digunakan untuk merepresentasikan suatu objek-objek diskrit. Banyak persoalan pada kehidupan sehari-hari yang sebenarnya merupakan representasi visual dari suatu graf [14]. Contoh salah satu representasi dari visual dari graf yaitu peta. Banyak hal yang bisa didapat dari representasi tersebut, diantaranya yaitu menentukan jalur terpendek dari satu tempat ke tempat lainnya, dengan menggambarkan dua tempat yang bertetangga menggunakan warna yang berbeda pada peta, dengan menentukan tata letak dari jalur transportasi, pengaturan jaringan komunikasi [15].



Gambar 1. Graf beberapa blok parkiran

Secara ini graf dapat didefinisikan sebagai berikut:

Graf G yaitu dapat didefinisikan sebagai pasangan dari suatu himpunan (V, E) yang dalam hal ini:

V = dijelaskan suatu himpunan tidak kosong dari suatu simpul (*vertices* atau *node*):
{v1, v2,…, vn }

E = dijelaskan suatu himpunan sisi (*edges* atau *arcs*) yang akan menghubungkan sepasang simpul:
{e 1 ,e 2 ,…,e n } atau dapat dinotasi G = (V,E).

2.3. Best First Search

Best First Search adalah metode yang menghasilkan *node* dari node sebelumnya. Best First Search memilih node baru yang mempunyai *cost* paling rendah atau kecil diantara semua node leaf yang telah dimunculkan. Penentuan *node* terbaik bisa dilakukan menggunakan fungsi yang disebut sebagai fungsi evaluasi f(n), fungsi dari evaluasi pencarian terbaik pertama dapat berupa perkiraan biaya dari suatu node ke suatu tujuan atau kombinasi antara biaya aktual dan perkiraan biaya [16]. Best First Search bekerja dengan cara memperluas *node* atau simpul yang mendekati kecocokan yang sesuai dengan aturan [17]. Pencarian rute terbaik dengan *(Best First Search)* merupakan cara yang menggabungkan keuntungan untuk kelebihan dari

pencarian *Breadth First* dan *Depth First* [18][19]. Cara kerja Algoritma Best First Search yaitu dengan menggunakan nilai-nilai heuristic dalam mencari solusi permasalahannya [20]. Pada setiap langkah proses pencarian terbaik pertama, memilih simpul-simpul (*node*) yang paling menjanjikan sesuatu. Kemudian, dikembangkan simpul yang sudah terpilih itu langsung menggunakan aturan-aturan untuk menghasilkan penggantinya [21]. Jadi salah satu pada simpul merupakan solusi, maka pencarian akan berhenti. Jika suatu simpul bukan solusi, maka semua pada suatu simpul baru itu ditambahkan pada suatu himpunan simpul yang telah dibuat. Setelah itu, simpul kembali akan memilih yang paling diharapkan atau sesuai dan melanjutkan prosesnya. Namun, jika suatu proses tidak bisa mendapatkan solusi, selanjutnya pencabangan akan mulai mencari titik simpul yang tidak menjanjikan sesuatu pada cabang di tingkat puncak yang telah diabaikan. Di titik ini, cabang yang lebih menjanjikan sesuatu dan telah terabaikan.

Fungsi *heuristic h(n)* yaitu estimasi *cost* dari *n* pada simpul tujuan. Dalam hal ini sangat penting untuk memilih fungsi *heuristic* yang lebih baik. Misalkan *h*(n)* yatu *cost* sebenarnya dari suatu simpul *n* ke simpul tujuan [22][23], maka pada algoritma dijkstra dapat beberapa kemungkinan yang bisa terjadi pada pemilihan fungsi *heuristic* yang sedang digunakan yaitu:

1. Jika h (n) = 0, maka hanya g(n) yang terbilata yaitu A* dengan bekerja seperti hanya algoritma dijkstra.
2. Jika h(n) < h*(n), maka algoritma dijkstra akan mengembangkan pada titik dengan nilai paling terendah dan algoritma *dijkstra* akan menjamin mendapatkan lintasan terpendek. Nilai h(n) terendah akan membuat algoritma untuk mengembangkan lebih banyak simpul. Jika h(n) < h*(n), maka h(n) dapat dikatakan *heuristic* yang *admissible*.
3. Jika h(n) = h*(n), dijkstra akan mengikuti pada lintasan. Tetapi hal ini tidak akan terjadi pada semua kasus. Informasi yang sesuai akan mempercepat kinerja dijkstra.
4. Jika h(n) > h*(n), dijkstra tidak menjamin untuk pencarian rute terpendek, tetapi akan berjalan dengan lebih cepat.
5. Jika h(n) lebih tinggi relative dengan *g(n)* sehingga akan hanya *h(n)* yang bekerja, maka dijkstra berubah jadi Greedy Best First Search.

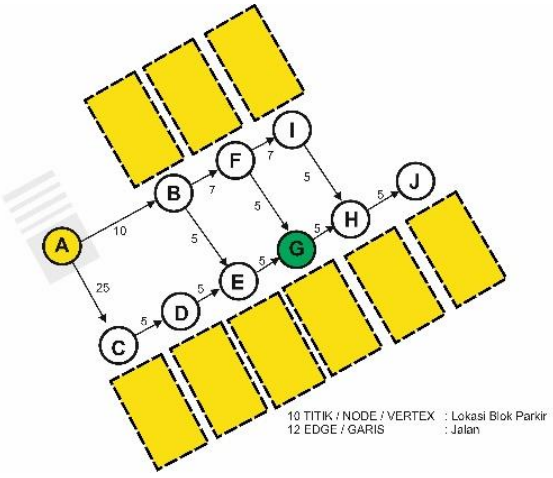
3. HASIL DAN DISKUSI

3.1. Pengukuran

Terdapat 10 titik objek lokasi blok parkir, dari titik lokasi tempat masuk tempat pakrir ke tempat lokasi blok parkir yang dituju dengan pemodelan *graph* untuk menghitung algoritma dijkstra untuk mendapatkan rute terbaik dari titik awal blok tempat parkir menuju titik tujuan lokasi blok tempat parkir. Dalam *graph* ini, titik pada objek lokasi tempat parkir yaitu *node*, sedangkan pada *vertex* dijalan merupakan suatu garis atau *edge* yang terhubung antara titik objek lokasi tempat parkir, pada bobot dari jarak terdapat titik yang dihitung dengan jarak garis lurus antara dua titik *node*. Untuk menerapkan algoritma dijkstra pada pencarian rute terbaik dibutuhkan pemodelan dengan *graph* pada jaringan jalan di lokasi blok tempat parkir.

Pemilihan *rute* menggunakan algoritma dijkstra dilakukan menggunakan metode Best First Search *(BFS)*, metode best first search akan diperbolehkan untuk mengunjungi *node* yang mempunyai level lebih rendah jika *node* yang mempunyai level lebih tinggi memiliki nilai heuristik yang tidak baik. Dari penjelasan algoritma dijkstra dapat dilihat pada Gambar 1.

Pada *graph* yang berarah mempunyai G dan sumber *vertexs* didalam G dan V merupakan suatu himpunan semua *vertices* dalam *graph* G, pada hal ini *graph* berarah yang berbobot dan sumber dari *vertexs* dapat disimpulkan sebagai *inputan* dari algoritma dijkstra. Setiap sisi dari *graph* adalah pasangan *vertices* (u,v) yang melambangkan hubungan dari *vertex* u ke *vertex* v.



Gambar 2. Pemodelan *Graph* Jaringan Lokasi Blok Parkir

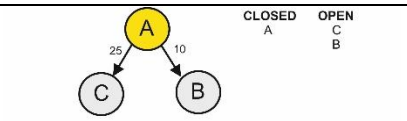
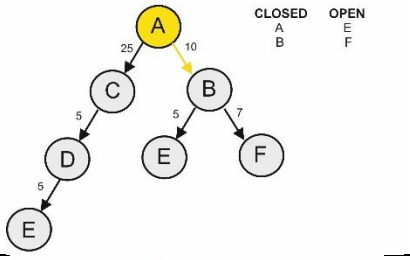
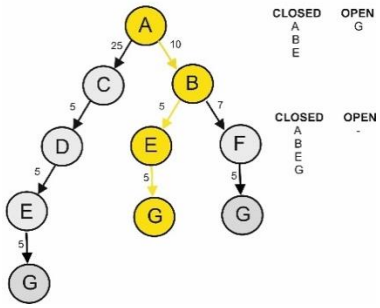
Tabel 1. Keterangan dari Pemodelan Graf

Blok	Keterangan
A	Blok A
B	Blok B
C	Blok C
D	Blok D
E	Blok E
F	Blok F
G	Blok G
H	Blok H
I	Blok I
J	Blok J

3.2. Hasil

Algoritma Dijkstra untuk mencari lintasan terdekat dari *node a* ke *node g* menggunakan *graph* berbobot yang terhubung. Dengan hasil *graph* pada jaringan dijalan dari blok lokasi tempat parkir, maka akan dilakukan pencarian dengan Best First Search untuk mencari lintasan terpendek untuk dibandingkan dari titik awal ke titik terdekat. Best First Search yaitu membandingan terlebih dahulu jarak antara node awal ke *node* yang terdekat yang mempunyai nilai terkecil maka akan dilalui. Dari hasil pencarian dari titik awal yaitu Blok A ke Blok G dengan rute algoritma dijkstra yang dilakukan menggunakan algorima Best First Search.

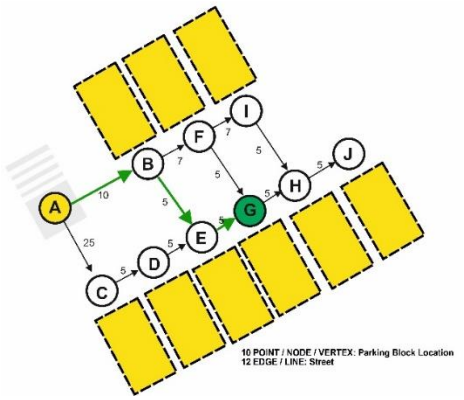
Tabel 2. Hasil Best First Search dari simpul *a* ke simpul yang berhubungan.

Langkah Pertama		Pada node ini dicari jalur terpendek dari *vertex* sumber A ke *vertex* lain.
Langkah Kedua		Dari hasil pencarian jalur terpendek pada langkah pertama antara *vertex* C dan *vertex* B maka dihasilkan *vertex* B adalah jalur terpendek yang mempunyai jarak 10 m.
Langkah Ketiga		Dari hasil pencarian jalur terpendek pada langkah kedua antara *vertex* E dan *vertex* F maka dihasilkan *vertex* E adalah jalur terpendek yang mempunyai jarak 5 m, *vertex* E yang berhubungan langsung dengan *vertex* G yaitu *vertex* tujuan.

Hasil dari algoritma dijkstra yaitu jalur terbaik mulai dari simpul asal *a* = Blok A menuju simpul tujuan *g* = Blok G. Dapat dilihat pada tabel 3.

Tabel 3. Hasil dijkstra dari Blok A ke Blok G

Asal ke Tujuan	Rute	Keterangan	Jarak	Total
A-G	(A-B-F-G)	Blok A – Blok B – Blok F – Blok G	(10+7+5+5)	27
	(A-B-F-G)	Blok A – Blok B – Blok F – Blok G	(10+7+5+5)	27
	(A-B-E-G)	Blok A - Blok B - Blok E - Blok G	(10+5+5)	20



Gambar 3. Hasil dijkstra dari Blok A ke Blok G

Dalam hasil pencarian rute dengan algoritma dijkstra yang dilakukan menggunakan Best First Search dengan sistem yang telah dibuat dari titik A (Blok A) hingga titik G (Blok G). Dengan hasil Best First Search yaitu membandingan terlebih dahulu jarak antara *node* awal ke *node* yang terdekat yaitu A (Blok A) hingga titik C (Blok C) dengan jarak 25 meter, sedangkan A (Blok A) ke titik B (Blok B) dengan jarak 10 meter, maka yang mempunyai nilai terkecil akan dilalui. Dari hasil pencarian dari titik awal yaitu A (Blok A) hingga titik G (Blok G) dihasilkan rute tebaik dengan rute yang dilalui yaitu Blok A - Blok B - Blok E - Blok G dengan total jarak 20 meter.

4. KESIMPULAN

Permasalahan pencarian lokasi parkir terbaik, pemodelan menggunakan *graph* untuk menghitung algoritma dijkstra untuk mendapatkan rute terbaik dari titik awal ke titik akhir dari lokasi blok parkir. Pada *graph* ini titik objek pada lokasi parkir merupakan *node* awal, sedangkan jalan merupakan garis atau *edge* yang terhubung antara titik lokasi tempat parkir yang mempunyai bobot jarak antara titik didapat menggunakan perhitungan dari jarak garis lurus antara dua titik *node*. Untuk menerapkan pada algoritma dijkstra untuk pencarian rute terbaik diperlukan pemodelan *graph* pada jaringan jalan di lokasi blok tempat parkir. Pemilihan untuk rute dalam algoritma dijkstra dilakukan dengan menggunakan algoritma Best First Search (BFS), best first search diperbolehkan untuk mengunjungi *node* pada level lebih kecil jika *node* pada level lebih tinggi memiliki nilai tidak baik.

Dari hasil pembahasan dan pengujian yang telah dilakukan, dapat disimpulkan bahwa algoritma Dijkstra telah berhasil diterapkan untuk menyelesaikan permasalahan jalur terpendek dari rute asli menuju objek-objek blok tempat parkir sehingga menghasilkan efisiensi waktu. Pada sistem ini menghasilkan solusi rute dengan total jarak minimal dari titik awal Blok A menuju titik akhir Blok G dengan total jarak 20 meter.

REFERENSI

[1] Angriani, H., & Saharaeni, Y. (2020). Implementasi Algoritma Best First Search Dalam Sistem Pakar Pertolongan Pertama Pada Bayi dan Anak. Inspiration: Jurnal Teknologi Informasi Dan Komunikasi, 10(2). https://doi.org/10.35585/inspir.v10i2.2575

[2] Bunaen, M. C., Pratiwi, H., & Riti, Y. F. (2022). PENERAPAN ALGORITMA DIJKSTRA UNTUK MENENTUKAN RUTE TERPENDEK DARI PUSAT KOTA SURABAYA KE TEMPAT BERSEJARAH. Jurnal Teknologi Dan Sistem Informasi Bisnis, 4(1).

[3] Bertarina, & Arianto, W. (2021). Analisis Kebutuhan Ruang Parkir (Studi Kasus pada Area Parkir ICT Universitas Teknokrat Indonesia). Jurnal SENDI, 02(02).

[4] Saputra, I., & Ahmad, D. (2020). Algoritma Genetika Untuk Menentukan Jalur Terpendek Wisata Kota Bukittinggi. Journal of Mathematics UNP, 3(1).

[5] Ahdan, S., & Setiawansyah. (2020). Pengembangan sistem informasi geografis untuk pendonor darah dengan algoritma dijakartaa berbasis android. Jurnal Sains Dan Informatika, 6(2).

[6] Pratama, Z., Hartama, D., Ridwan Lubis, M., Retno Andani, S., & Okta Kirana, I. (2020). Penerapan Metode Dijkstra untuk Menentukan Jalur Lintasan Terpendek Kota Kisaran Menuju Objek Wisata Simalungun. Rekayasa Teknik Informatika Dan Informasi, 1(2).

[7] A Pahlevi, M. R., & Komalasari, R. T. (2022). Implementasi Algoritma Dijkstra Rute Terpendek pada Aplikasi WisKul PasMing. Jurnal JTIK (Jurnal Teknologi Informasi Dan Komunikasi), 6(4). https://doi.org/10.35870/jtik.v6i4.554.

[8] Gautama, I. P. W., & Hermanto, K. (2020). Penentuan Rute Terpendek dengan Menggunakan Algoritma Dijkstra pada Jalur Bus Sekolah. Jurnal Matematika, 10(2). https://doi.org/10.24843/jmat.2020.v10.i02.p128

[9] Indrayanti, I., Risqiati, R., & Setianto, W. (2020). Penentuan Rute Terpendek Perjalanan Promosi Marketing Menggunakan Algoritma Dijkstra. IC-Tech.
[10] Hidayah, A. A. (2022). PENERAPAN ALGORITMA DIJKSTRA PADA APLIKASI JASA TRANSPORTASI ONLINE DI KOTA MEDAN. AL-ULUM: JURNAL SAINS DAN TEKNOLOGI, 7(1). https://doi.org/10.31602/ajst.v7i1.5710
[11] Muharrom, M. (2020). IMPLEMENTASI ALGORITMA DIJKSTRA DALAM PENENTUAN JALUR TERPENDEK STUDI KASUS JARAK TEMPAT KULIAH TERDEKAT. Indonesian Journal of Business Intelligence (IJUBI), 3(1). https://doi.org/10.21927/ijubi.v3i1.1229
[12] Supiyandi, & Eka, M. (2018). Penerapan Teknik Pewarnaan Graph Pada Penjadwalan Ujian Dengan Algoritma Welch-Powell. ALGORITMA: Jurnal Ilmu Matemattika Dan Komputer, 3(1).
[13] Baharudin, I., Purwanto, A. J., Budiman, T. R., & Fauzi, M. (2021). IMPLEMENTASI ALGORITMA DIJKSTRA UNTUK MENENTUKAN JALUR TERPENDEK DALAM DISTRIBUSI BARANG. Jurnal Lebesgue : Jurnal Ilmiah Pendidikan Matematika, Matematika Dan Statistika, 2(2). https://doi.org/10.46306/lb.v2i2.74
[14] Mahardika, F. (2019). Penerapan Teori Graf Pada Jaringan Komputer Dengan Algoritma Kruskal. Jurnal Informatika: Jurnal Pengembangan IT, 4(1). https://doi.org/10.30591/jpit.v4i1.1032
[15] Arif, A., & Sasmita. (2021). IMPLEMENTASI HANNAFIN & PECK MODEL PADA APLIKASI ANIMASI PEMBELAJARAN TEORI GRAPH BERBASIS ANDROID. Jurnal Teknologi Informasi Mura, 13(1).
[16] Sulistiani, H., Wardani, F., & Sulistyawati, A. (2019). Application of Best First Search Method to Search Nearest Business Partner Location (Case Study: PT Coca Cola Amatil Indonesia, Bandar Lampung). Proceedings - 2019 International Conference on Computer Science, Information Technology, and Electrical Engineering, ICOMITEE 2019. https://doi.org/10.1109/ICOMITEE.2019.8920905
[17] Liana, L. I., & Nudin, S. R. (2020). Implementasi Algoritma Best-First Search untuk Aplikasi Mesin Pencari Handphone pada E-commerce (Apenphone). Journal of Informatics and Computer Science (JINACS), 2(01). https://doi.org/10.26740/jinacs.v2n01.p67-73
[18] Sujaini, H., Perwitasari, A., & Januardi, T. (2023). Sistem Pembelajaran Algoritma Best First Search, Breadth First Search & Depth First Search. Jurnal Teknik Indonesia, 2(2). https://doi.org/10.58860/jti.v2i2.15
[19] Muhardono, A. (2023). Penerapan Algoritma Breadth First Search dan Depth First Search pada Game Angka. Jurnal Minfo Polgan, 12(1). https://doi.org/10.33395/jmp.v12i1.12340
[20] Herfandi, H., Soleha, U., Susilo Yuda Irawan, A., Ahmad Baihaqi, K., & Maulana, R. (2022). Implementasi Algoritma Best First Search untuk Pencarian Rute Terpendek pada Aplikasi Cerdas Pendaftaran Santri Baru. Syntax : Jurnal Informatika, 11(01). https://doi.org/10.35706/syji.v11i01.6398
[21] Mardiana, M., Despa, D., Ardhi Muhammad, M., Septiana, T., & Lorenza, T. A. (2022). SISTEM NAVIGASI AUGMENTED REALITY DENGAN PENCARIAN JALUR TERBAIK MENUJU LOKASI PUSTAKA (STUDI KASUS PADA UPT PERPUSTAKAAN UNILA). Jurnal Profesi Insinyur Universitas Lampung, 3(2). https://doi.org/10.23960/jpi.v3n2.78
[22] Purwoko Aji, D. K., Ririd, A. R. T. H., & Setiyawan, A. (2019). Pencarian Jalur Terpendek Untuk Penjemputan Barang Kiriman Pelanggan Mitra (Studi Kasus Pada Kantor Pos Malang). In Jurnal Informatika Polinema (Vol. 5, Issue 2).
[23] Angriani, H., & Saharaeni, Y. (2020). Implementasi Algoritma Best First Search Dalam Sistem Pakar Pertolongan Pertama Pada Bayi dan Anak. Inspiration: Jurnal Teknologi Informasi Dan Komunikasi, 10(2). https://doi.org/10.35585/inspir.v10i2.2575