ANALISIS ALGORITMA DIJKSTRA DAN GREEDY PADA PENCARIAN JALUR TERPENDEK DI GRAF BERBOBOT

Desri Alfiyunita Sinlae[1], Pingky Kimberli Ledoh[2], Diana Fallo[3]
desrisinlae8@gmail.com[1], pingkyledoh@gmail.com[2], dianayani25@gmail.com[3]
Universitas Citra Bangsa

ABSTRAK

Penentuan jalur terpendek pada graf berbobot merupakan masalah penting dalam ilmu komputer, khususnya pada bidang jaringan dan sistem navigasi. Dua algoritma yang sering digunakan untuk menyelesaikan masalah ini adalah algoritma Dijkstra dan algoritma Greedy. Penelitian ini bertujuan untuk menganalisis kinerja kedua algoritma dalam menentukan jalur terpendek, berdasarkan studi literatur. Metode yang digunakan adalah pendekatan kualitatif melalui tinjauan pustaka dari jurnal, artikel ilmiah, dan buku referensi yang relevan. Hasil analisis menunjukkan bahwa algoritma Dijkstra memberikan hasil yang lebih optimal pada graf berbobot positif, sedangkan algoritma Greedy memiliki keunggulan dalam efisiensi waktu, meskipun tidak selalu memberikan solusi terbaik. Perbandingan ini disajikan dalam bentuk tabel untuk menunjukkan kelebihan dan kekurangan masing-masing algoritma. Kesimpulan dari penelitian ini menegaskan pentingnya pemilihan algoritma berdasarkan konteks aplikasi dan karakteristik graf yang digunakan.
Kata Kunci: Algoritma Dijkstra, Algoritma Greedy, Graf Berbobot, Jalur Terpendek.

ABSTRACT

Finding the shortest path in a weighted graph is a fundamental problem in computer science, especially in fields such as networking and navigation systems. Two widely used algorithms for solving this problem are Dijkstra's algorithm and the Greedy algorithm. This study aims to analyze the performance of both algorithms in determining the shortest path, based on a literature review. The method used is a qualitative approach through the analysis of journals, scientific articles, and relevant reference books. The analysis shows that Dijkstra's algorithm provides a more optimal result in positively weighted graphs, while the Greedy algorithm excels in execution speed, although it does not always yield the best solution. A comparison is presented in a table to illustrate the strengths and weaknesses of each algorithm. The conclusion emphasizes the importance of algorithm selection based on application context and graph characteristics.
Keywords: *Dijkstra Algorithm, Greedy Algorithm, Weighted Graph, Shortest Path.*

PENDAHULUAN

Pencarian jalur terpendek dalam graf berbobot merupakan permasalahan klasik dalam teori graf dan algoritma yang memiliki banyak aplikasi praktis, seperti dalam sistem navigasi GPS, pengiriman paket logistik, routing dalam jaringan komputer, serta manajemen lalu lintas dan perencanaan rute kendaraan otonom (Cormen et al., 2009; Ahuja et al., 1993). Dalam representasi graf, simpul (nodes) menggambarkan titik atau lokasi, sedangkan sisi (edges) menggambarkan hubungan antar simpul, yang biasanya diberi bobot untuk merepresentasikan jarak, waktu tempuh, atau biaya tertentu.

Berbagai algoritma telah dikembangkan untuk menyelesaikan permasalahan jalur terpendek, dua di antaranya yang paling umum digunakan adalah algoritma Dijkstra dan algoritma Greedy. Algoritma Dijkstra, yang diperkenalkan oleh Edsger W. Dijkstra pada tahun 1959, dikenal karena kemampuannya menemukan jalur terpendek secara menyeluruh dan optimal pada graf dengan bobot positif. Algoritma ini menggunakan strategi pemrograman dinamis dengan struktur data seperti priority queue untuk mempercepat proses seleksi simpul berikutnya (Dijkstra, 1959; Sedgewick & Wayne, 2011).

Sebaliknya, algoritma Greedy mengambil pendekatan yang lebih sederhana dengan memilih lintasan lokal terbaik di setiap langkah tanpa mempertimbangkan keseluruhan graf. Meskipun pendekatan ini secara komputasi lebih ringan dan cepat, namun sering kali tidak menghasilkan solusi optimal secara global (Kiran & Raju, 2018). Oleh karena itu, algoritma Greedy lebih cocok untuk aplikasi real-time atau sistem dengan keterbatasan sumber daya, meskipun harus dikompromikan dalam hal akurasi.

Penelitian ini bertujuan untuk menganalisis perbedaan performa antara algoritma Dijkstra dan Greedy dalam menentukan jalur terpendek pada graf berbobot berdasarkan kajian literatur. Fokus utama penelitian adalah membandingkan efektivitas algoritma berdasarkan beberapa aspek seperti kompleksitas waktu, kebutuhan memori, akurasi solusi, dan efisiensi komputasi. Penelitian sebelumnya oleh Singh dan Sharma (2021) menunjukkan bahwa meskipun algoritma Dijkstra lebih akurat dalam menemukan jalur terpendek, algoritma Greedy menawarkan keunggulan dalam kecepatan dan kesederhanaan implementasi.

Untuk mendukung analisis ini, landasan teori yang digunakan meliputi konsep dasar graf (berarah, tidak berarah, berbobot), teori jalur terpendek, serta penjelasan rinci tentang prinsip kerja dan kelebihan-kekurangan dari algoritma Dijkstra dan Greedy. Evaluasi performa algoritma juga akan dikaji berdasarkan literatur terkini dan dibantu dengan representasi tabel perbandingan untuk mempermudah pemahaman.

METODE PENELITIAN

Jenis penelitian ini adalah studi literatur dengan pendekatan kualitatif deskriptif. Objek penelitian adalah algoritma Dijkstra dan algoritma Greedy dalam konteks pencarian jalur terpendek pada graf berbobot.

Data dikumpulkan dari berbagai sumber sekunder, termasuk jurnal internasional, prosiding konferensi, buku teks algoritma, dan artikel ilmiah lain yang relevan dalam 10 tahun terakhir. Teknik analisis data dilakukan dengan mengklasifikasikan hasil penelitian sebelumnya, membandingkan kelebihan dan kekurangan kedua algoritma, serta menyusun tabel komparatif untuk memperjelas perbandingan performa algoritma.

HASIL DAN PEMBAHASAN

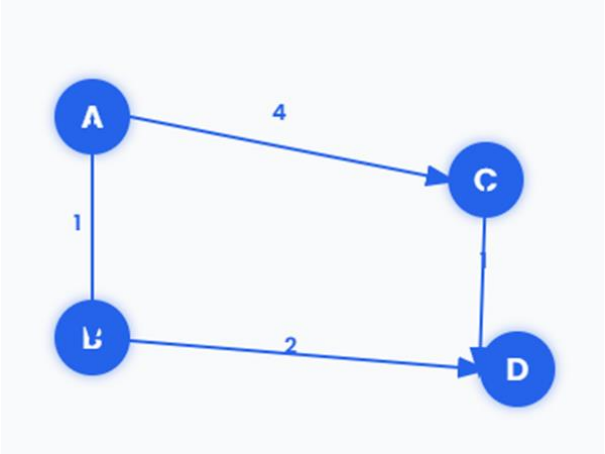
Hasil studi literatur menunjukkan bahwa algoritma Dijkstra mampu memberikan solusi jalur terpendek secara optimal dengan kompleksitas waktu O(V^2) atau O((V+E) log V), tergantung pada struktur data yang digunakan. Sementara itu, algoritma Greedy cenderung lebih cepat dalam proses komputasi, namun tidak selalu menghasilkan solusi optimal karena hanya mempertimbangkan jarak lokal terbaik.

Berikut adalah tabel perbandingan:

Kriteria	**Dijkstra**	**Greedy**
Optimalitas	Optimal	Tidak selalu optimal
Kompleksitas Waktu	$O(V^2)$ / O((V+E) log V)	Lebih rendah (tergantung implementasi)
Kebutuhan Memori	Sedang–Tinggi	Rendah
Jenis Graf	Berbobot positif	Beragam, hasil bervariasi
Kecepatan	Relatif lambat	Cepat

Graf Contoh Dijkstra dan Greedy

Graf dengan node A, B, C, D yang dihubungkan oleh sisi dengan bobot. Ilustrasi ini digunakan untuk menjelaskan jalur terpendek dan perbedaan algoritma.



Penjelasan

- Node A terhubung ke Node B dengan bobot 1 dan ke Node C dengan bobot 4.
- Node B terhubung ke Node D dengan bobot 2.
- Node C terhubung ke Node D dengan bobot 1.

Algoritma Dijkstra

1. Inisialisasi:
   - Jarak dari A ke A: 0
   - Jarak dari A ke B: 1
   - Jarak dari A ke C: 4
   - Jarak dari A ke D: ∞ (tak terhingga)
2. Proses:
   - Mulai dari A, proses node A.
   - Dari A, kita dapat mencapai B (1) dan C (4).
   - Proses node B selanjutnya (karena jaraknya lebih kecil).
   - Dari B, kita dapat mencapai D dengan bobot 2, sehingga jarak ke D melalui B adalah 1 + 2 = 3.
   - Proses node C, dari C ke D dengan bobot 1, jarak ke D melalui C adalah 4 + 1 = 5.
   - Jarak terpendek ke D adalah 3 (A → B → D).

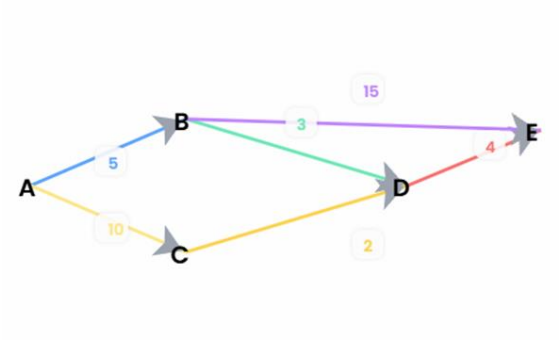
Algoritma Greedy

1. Mulai dari A:
   - Dari A, pilih jalur dengan bobot terkecil, yaitu A → B (1).
2. Dari B:
   - Dari B, kita hanya memiliki satu pilihan, yaitu B → D (2).
   - Total bobot dari A ke D melalui jalur ini adalah 1 + 2 = 3.

Graf terdiri dari empat titik (A, B, C, D) yang dihubungkan oleh sisi dengan bobot masing-masing. Warna berbeda membantu membedakan tiap titik dan sisi terkait. Pada graf ini, algoritma Dijkstra akan mencari jalur terpendek secara optimal dengan mempertimbangkan semua kemungkinan jalur, sementara algoritma Greedy memilih sisi dengan bobot terkecil secara lokal yang bisa saja tidak menghasilkan jalur terpendek secara keseluruhan.

Contoh Studi Kasus Pemetaan Rute Transportasi Kota

Menemukan Jalur Tercepat dari Terminal Utama (A) ke Halte Universitas (E)



Hasil Studi Kasus

Jalur Optimal:

A → B → D → E

Waktu Tempuh: 12 menit

Penjelasan: Grafik di atas menunjukkan jaringan halte bus di sebuah kota kecil dengan bobot berupa waktu tempuh antar halte dalam menit.

Setiap lingkaran berwarna mewakili titik halte, sedangkan garis berwarna menghubungkan halte-halte tersebut dengan angka di tengah menunjukkan waktu tempuh.

Jalur tercepat dari Terminal Utama (A) ke Halte Universitas (E) adalah melalui B dan D dengan total waktu tempuh 12 menit seperti yang dicapai oleh algoritma Dijkstra.

Jika bobot jalur B → E diubah lebih kecil, algoritma Greedy mungkin memilih jalur yang lebih cepat secara lokal namun tidak optimal secara global.

Titik A - Terminal Utama

Titik B - Halte Pasar

Titik C - Halte Sekolah

Titik D - Halte Rumah Sakit

Titik E - Halte Universitas

Hasil ini selaras dengan penelitian Singh & Sharma (2021) yang menyimpulkan bahwa Dijkstra lebih akurat sedangkan Greedy lebih efisien secara waktu. Namun, dalam aplikasi real-time seperti game atau sistem navigasi kendaraan dengan sumber daya terbatas, algoritma Greedy masih relevan digunakan.

Analisis lebih lanjut menunjukkan bahwa algoritma Dijkstra, meskipun lebih lambat dalam hal kecepatan komputasi, tetap unggul dalam hal akurasi dan keoptimalan hasil. Kompleksitas waktu algoritma Dijkstra yang dapat mencapai O((V+E)log⁡V)O((V+E) \log V)O((V+E)logV) jika menggunakan struktur data heap, membuatnya cocok untuk graf besar dan aplikasi yang membutuhkan akurasi tinggi, seperti perencanaan transportasi cerdas dan manajemen jaringan (Cormen et al., 2009).

Di sisi lain, algoritma Greedy lebih efisien secara waktu dan lebih hemat memori, sehingga sangat cocok untuk aplikasi dengan keterbatasan sumber daya seperti sistem real-time, game, atau perangkat IoT. Namun, kelemahan utama dari pendekatan Greedy adalah kemampuannya yang terbatas dalam menghasilkan solusi optimal, karena pengambilan keputusan berdasarkan keuntungan lokal tanpa mempertimbangkan dampak global (Kiran & Raju, 2018).

Grafik di atas mengilustrasikan bahwa tidak ada algoritma yang secara mutlak lebih baik; pemilihan algoritma sangat bergantung pada konteks dan kebutuhan aplikasi.

KESIMPULAN

Penelitian ini menyimpulkan bahwa algoritma Dijkstra lebih unggul dalam hal akurasi jalur terpendek pada graf berbobot positif, sedangkan algoritma Greedy lebih cepat namun

berisiko menghasilkan solusi yang tidak optimal. Pemilihan algoritma sebaiknya mempertimbangkan konteks penggunaan, seperti keterbatasan waktu proses atau kebutuhan akurasi.

Saran untuk penelitian selanjutnya adalah melakukan implementasi langsung pada kasus nyata, seperti pemetaan rute transportasi publik atau routing jaringan internet, serta menggabungkan kedua algoritma dalam pendekatan hybrid untuk mendapatkan solusi yang lebih adaptif.

DAFTAR PUSTAKA

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). Network flows: Theory, algorithms, and applications. Prentice Hall.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to algorithms (3rd ed.). MIT Press.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. Numerische Mathematik, 1(1), 269–271.

Kiran, S., & Raju, K. R. (2018). Comparative study of shortest path finding algorithms. International Journal of Computer Applications, 181(25), 21–24.

Singh, P., & Sharma, R. (2021). Performance analysis of shortest path algorithms: Dijkstra vs. Greedy. International Journal of Computer Science and Engineering, 9(3), 45–50.

Sedgewick, R., & Wayne, K. (2011). Algorithms (4th ed.). Addison-Wesley.

Mohri, M. (2002). Semiring frameworks and algorithms for shortest-distance problems. Journal of Automata, Languages and Combinatorics, 7(3), 321–350.

Zhan, F. B., & Noon, C. E. (1998). Shortest path algorithms: An evaluation using real road networks. Transportation Science, 32(1), 65–73.

Yen, J. Y. (1971). Finding the K shortest loopless paths in a network. Management Science, 17(11), 712–716.

Das, A., & Mohapatra, D. P. (2015). A comparative study of Dijkstra's and A* algorithms in AI pathfinding. Procedia Computer Science, 54, 220–229.

Goldberg, A. V., & Harrelson, C. (2005). Computing the shortest path: A* search meets graph theory. In Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (pp. 156–165).

Papadimitriou, C. H., & Steiglitz, K. (1998). Combinatorial optimization: Algorithms and complexity. Dover Publications.

Fredman, M. L., & Tarjan, R. E. (1987). Fibonacci heaps and their uses in improved network optimization algorithms. Journal of the ACM, 34(3), 596–615.

Eppstein, D. (1998). Finding the k shortest paths. SIAM Journal on Computing, 28(2), 652–673.

Cherkassky, B. V., Goldberg, A. V., & Radzik, T. (1996). Shortest paths algorithms: Theory and experimental evaluation. Mathematical Programming, 73(2), 129–174.