MATHunesa
Jurnal Ilmiah Matematika *Volume 14 No 02*
e-ISSN : 2716-506X | p-ISSN : 2301-9115 *Tahun 2026*

OPTIMASI RUTE SILATURAHMI DENGAN PEMODELAN *TRAVELING SALESMAN PROBLEM* (TSP) MENGGUNAKAN ALGORITMA PEMROGRAMAN DINAMIK

Elsya Febriani Rosada
Program Studi Matematika, FMIPA, Universitas Negeri Yogyakarta, Yogyakarta, Indonesia
e-mail: elsyafebrianir@gmail.com*, elsyafebriani.2022@student.uny.ac.id

Caturiyati
Program Studi Matematika, FMIPA, Universitas Negeri Yogyakarta, Yogyakarta, Indonesia
e-mail: caturiyati@uny.ac.id

Abstrak

Permasalahan dalam menentukan rute silaturahmi yang efisien dapat dimodelkan sebagai *Traveling Salesman Problem (*TSP), yaitu permasalahan optimasi rute dengan tujuan meminimalkan total jarak tempuh. Penelitian ini bertujuan memodelkan dan mengoptimalkan rute silaturahmi dari Randudongkal menuju empat desa tujuan, yaitu Mejagong, Moga, Warungpring, dan Sikasur, dengan menerapkan algoritma pemrograman dinamik deterministik. Data jarak antar lokasi diperoleh melalui *Google Maps* dan direpresentasikan dalam bentuk matriks jarak berbobot. Penyelesaian dilakukan menggunakan pendekatan rekursi mundur (*backward recursion*) berdasarkan prinsip optimalitas Bellman. Hasil penelitian menunjukkan bahwa total jarak minimum yang diperoleh adalah 29.200 meter, dengan dua rute optimal yang bersifat simetris, yaitu Randudongkal – Warungpring – Moga – Mejagong – Sikasur – Randudongkal dan Randudongkal – Sikasur – Mejagong – Moga – Warungpring – Randudongkal. Hasil tersebut menunjukkan efektivitas pemrograman dinamik dalam menghasilkan solusi eksak untuk permasalahan TSP berskala kecil dalam konteks non-komersial, serta dapat dijadikan dasar untuk pengembangan penelitian selanjutnya pada permasalahan rute dengan skala dan kompleksitas yang lebih tinggi.
Kata Kunci: Optimasi Rute, Pemrograman Dinamik, Rekursi Mundur, Rute Silaturahmi, *Traveling Salesman Problem* (TSP)

Abstract

The problem of determining an efficient social visit route can be modeled as the Traveling Salesman Problem (TSP), which is a route optimization problem aimed at minimizing the total travel distance. This study aims to model and optimize a social visit route starting from Randudongkal to four destination villages, namely Mejagong, Moga, Warungpring, and Sikasur, by applying a deterministic dynamic programming algorithm. Distance data between locations were obtained from Google Maps and represented in the form of weighted distance matrix. The solution was carried out using backward recursion approach based on Bellman's principle of optimality. The result show that the minimum total distance obtained is 29.200 meters, with two symmetric optimal routes, namely Randudongkal – Warungpring – Moga – Mejagong – Sikasur – Randudongkal and Randudongkal – Sikasur – Mejagong – Moga – Warungpring – Randudongkal. These results demonstrate the effectiveness of dynamic programming in producing exact solutions for small-scale TSP instances in a non-commercial context and provide a foundation for further research on route optimization problems with greater scale and complexity.
Keywords: *Route Optimization, Social Visit Route, Traveling Salesman Problem (TSP), Dynamic Programming, Backward Recursion*

PENDAHULUAN

Masalah optimasi merupakan salah satu permasalahan yang sering dijumpai dalam berbagai bidang, seperti logistik, transportasi, dan perencanaan perjalanan (Sitinjak *et al*., 2018). Secara umum, permasalahan optimasi bertujuan untuk menemukan solusi optimal, baik berupa nilai maksimum maupun minimum dari suatu fungsi tujuan, dengan mempertimbangkan serangkaian kendala tertentu (Taha, 2017). Pada konteks kehidupan sehari-hari, salah satu bentuk permasalahan optimasi yang relevan adalah penentuan rute perjalanan yang efektif, termasuk dalam aktivitas kunjungan silaturahmi ke beberapa lokasi dalam satu perjalanan. Penentuan urutan kunjungan yang tepat untuk meminimalkan total jarak atau waktu tempuh menjadi tantangan praktis,

terutama ketika lokasi-lokasi tujuan tersebar di wilayah geografis yang luas.

Secara konseptual, permasalahan penentuan rute silaturahmi dapat dimodelkan sebagai *Traveling Salesman Problem* (TSP), yaitu permasalahan optimasi kombinatorial klasik yang bertujuan menentukan rute terpendek untuk mengunjungi setiap lokasi tepat satu kali dan kembali ke titik awal (Singhal & Pandey, 2016). *Traveling Salesman Problem* (TSP) umumnya direpresentasikan dalam bentuk graf lengkap berbobot $G = (V, E)$, dengan himpunan simpul V merepresentasikan lokasi yang dikunjungi dan himpunan sisi E menggambarkan jalur antar lokasi dengan bobot c_{ij} sebagai jarak atau biaya perjalanan (Ugonna *et al*., 2024). Permasalahan TSP memiliki tingkat kompleksitas komputasi yang tinggi sehingga pendekatan penyelesaiannya bervariasi, mulai dari algoritma eksak yang menjamin optimalitas hingga metode heuristik dan metaheuristik untuk kasus berskala besar (Xu *et al*., 2018). Kajian terkini menunjukkan algoritma eksak, termasuk pemrograman dinamik (*dynamic programming*) dan pemodelan optimasi, menjadi pilihan yang tepat untuk permasalahan berskala kecil hingga menengah, sedangkan metode berbasis *machine learning* dikembangkan untuk menangani kasus berskala besar (Alanzi & Menai, 2025).

Pemrograman dinamik, yang diperkenalkan oleh Richard Bellman, menyelesaikan permasalahan kompleks dengan memecahnya menjadi submasalah yang lebih sederhana dan saling berkaitan berdasarkan prinsip optimalitas (Bellman, 1957; Dreyfus, 2002). Karakteristik utama berupa adanya substruktur optimal dan submasalah yang tumpang tindih menjadikan pemrograman dinamik sangat sesuai untuk menyelesaikan TSP secara deterministik (Chandra & Naro, 2021). Meskipun demikian, peningkatan jumlah simpul menyebabkan kebutuhan komputasi pemrograman dinamik meningkat secara eksponensial sehingga penerapannya menjadi terbatas pada permasalahan berskala kecil hingga menengah (Anson, 2024).

Efektivitas pemrograman dinamik dalam menyelesaikan *Traveling Salesman Problem* (TSP) telah didukung oleh berbagai penelitian. Secara khusus, penelitian sebelumnya menunjukkan bahwa pemrograman dinamik mampu menghasilkan solusi optimal secara efektif pada permasalahan TSP berskala kecil (Chandra & Naro, 2021). Wahyudi dkk. (2024) menerapkan pemrograman dinamik untuk optimasi rute pengiriman paket dan berhasil memperoleh rute dengan jarak tempuh minimum. Sagala dkk. (2022) membuktikan keefektifan metode ini dalam menentukan rute terpendek pada perusahaan logistik. Selain itu, Ugonna *et al*. (2024) menunjukkan bahwa pemrograman dinamik menghasilkan penghematan biaya perjalanan yang signifikan pada kasus penjual yang harus mengunjungi beberapa lokasi. Namun demikian, sebagian besar penelitian tersebut masih berfokus pada konteks logistik, distribusi barang, dan aktivitas komersial yang menekankan efisiensi biaya operasional dan skala permasalahan yang relatif besar. Di sisi lain, kajian yang menerapkan pemrograman dinamik untuk menyelesaikan permasalahan TSP dalam konteks rute silaturahmi dengan tujuan non-komersial dan lingkup wilayah lokal masih relatif terbatas. Padahal, permasalahan ini memiliki karakteristik yang serupa dengan TSP klasik, tetapi diterapkan pada konteks sosial yang lebih sederhana dan dekat dengan kehidupan sehari-hari. Hal ini menunjukkan potensi untuk eksplorasi lebih lanjut.

Berdasarkan identifikasi tersebut, penelitian ini bertujuan untuk memodelkan permasalahan penentuan rute silaturahmi dari Randudongkal menuju empat desa tujuan, yaitu Mejagong, Moga, Warungpring, dan Sikasur, sebagai *Traveling Salesman Problem* (TSP), serta mengimplementasikan pemrograman dinamik deterministik dengan pendekatan rekursi mundur untuk memperoleh solusi rute optimal dengan jarak tempuh minimum. Hasil penelitian ini diharapkan dapat memberikan solusi praktis bagi perencanaan aktivitas sosial, sekaligus memperkaya kajian mengenai penerapan metode optimasi deterministik pada permasalahan rute berskala kecil dalam konteks non-komersial.

KAJIAN TEORI

TRAVELLING SALESMAN PROBLEM (TSP)

Traveling Salesman Problem (TSP) pertama kali diformulasikan pada tahun 1930 dan dikenal sebagai salah satu permasalahan klasik dalam bidang optimasi. Permasalahan ini mengilustrasikan situasi seorang penjual yang harus mengunjungi sejumlah lokasi dengan tujuan meminimalkan total biaya atau jarak tempuh perjalanan, dengan ketentuan bahwa setiap lokasi dikunjungi tepat satu kali dan

perjalanan harus berakhir kembali di lokasi asal. Secara matematis, TSP dirumuskan berdasarkan himpunan lokasi beserta jarak atau biaya antar setiap pasangan lokasi untuk menentukan lintasan terpendek yang mencakup seluruh lokasi. Karakteristik tersebut menjadikan TSP banyak digunakan sebagai model dalam berbagai aplikasi, seperti perencanaan rute distribusi, manajemen logistik, dan pengaturan jadwal perjalanan (Cormen *et al*., 2022; Hillier & Lieberman, 2015; Taha, 2017). Model TSP didefinisikan oleh jumlah lokasi n dan sebuah matriks biaya atau jarak c_{ij}. Pada definisi sebuah tur, tidak diperkenankan adanya lintasan yang menghubungkan suatu lokasi dengan dirinya sendiri. Oleh karena itu, elemen diagonal matriks biaya c_{ii} diasumsikan bernilai sangat besar sehingga lintasan tersebut tidak dipertimbangkan dalam solusi. Suatu TSP disebut simetris apabila biaya perjalanan memenuhi $c_{ij} = c_{ji}$, sedangkan jika kondisi tersebut tidak terpenuhi, maka TSP bersifat asimetris (Taha, 2017).

Salah satu pendekatan umum dalam memodelkan TSP adalah melalui pemrograman matematis dengan mendefinisikan variabel keputusan sebagai berikut (Hillier & Lieberman, 2015).

$$x_{ij} = \begin{cases} 1, & \text{jika lokasi } j \text{ dikunjungi langsung dari } i \\ 0, & \text{untuk lainnya} \end{cases} \quad (1)$$

Berdasarkan definisi tersebut, model matematis TSP dapat dirumuskan sebagai berikut.

Fungsi tujuan:

$$\text{Meminimumkan } Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \quad (2)$$

dengan batasan:

$$\sum_{i=1}^{n} x_{ij} = 1, \quad \text{untuk setiap } i = 1, 2, \dots, n \quad (3)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \text{untuk setiap } j = 1, 2, \dots, n \quad (4)$$

$$x_{ij} \in \{0,1\}, \forall i, j$$

Keterangan:

Z = Nilai fungsi tujuan yang akan dioptimalkan
n = Jumlah lokasi yang harus dikunjungi
c_{ij} = Biaya atau jarak perjalanan dari lokasi i ke j
x_{ij} = Variabel keputusan

Batasan (3) dan (4) memastikan bahwa setiap lokasi dikunjungi tepat satu kali dan setiap lokasi ditinggalkan tepat satu kali, sehingga membentuk lintasan tertutup yang mencakup seluruh lokasi.

PEMROGRAMAN DINAMIK

Pemrograman dinamik merupakan metode optimasi untuk menyelesaikan permasalahan kompleks dengan membagi masalah utama menjadi submasalah yang lebih kecil, saling tumpang tindih, dan memiliki struktur submasalah optimal, sehingga solusi optimal dapat dibangun dari solusi optimal submasalahnya (Bellman, 1957; Cormen *et al*., 2022). Pemrograman dinamik memanfaatkan strukur rekursif, prinsip optimalitas, serta formulasi matematis berbentuk persamaan fungsi yang diselesaikan secara bertahap untuk memperoleh solusi optimal secara numerik, dan telah terbukti efektif dalam berbagai permasalahan seperti jaringan kerja, pengendalian persediaan, penggantian alat, penjadwalan, dan alokasi sumber daya (Gillett, 1979; Sargent & Stachurski, 2024). Dasar utama metode ini adalah prinsip optimalitas Bellman, yang menyatakan bahwa kebijakan optimal untuk keseluruhan proses terdiri atas kebijakan optimal untuk setiap *stage* (tahap), tanpa bergantung pada kebijakan sebelumnya (Bellman, 1957; Hillier & Lieberman, 2015). Dengan kata lain, apapun *state* (keadaan) awal dan keputusan yang diambil pada suatu *stage*, rangkaian keputusan yang tersisa akan tetap optimal jika hanya mempertimbangkan *state* yang dihasilkan pada *stage* berikutnya (Bensoussan, 2011; Sargent & Stachurski, 2024).

Pemrograman dinamik diterapkan pada masalah keputusan berurutan yang dimodelkan sebagai serangkaian *stage*, yaitu tahapan atau langkah dalam proses pengambilan keputusan yang saling berurutan (Hillier & Lieberman, 2015; Taha, 2017). Struktur *stage* disusun sedemikian rupa sehingga ketika hanya tersisa satu *stage*, masalah menjadi lebih mudah diselesaikan, serta dalam berbagai aplikasi, *stage* dapat merepresentasikan periode waktu, langkah investasi, titik jaringan, atau tahapan keputusan lainnya (Gillett, 1979; Winston & Goldberg, 2004). Pada setiap *stage*, pengambil keputusan mengamati *state* saat ini, memilih keputusan, dan menghasilkan transisi ke *state* berikutnya (Sargent & Stachurski, 2024). *State* merepresentasikan kondisi sistem pada suatu *stage* tertentu yang memuat informasi yang cukup untuk menentukan keputusan selanjutnya tanpa bergantung pada riwayat keputusan sebelumnya, sehingga fungsi nilai dan keputusan hanya

bergantung pada *state* dan keputusan saat ini (Denardo, 1982).

Berdasarkan pendapat beberapa ahli, karakteristik pemrograman dinamik adalah sebagai berikut (Denardo, 1982; Hillier & Lieberman, 2015; Sargent & Stachurski, 2024; Winston & Goldberg, 2004).

1. Permasalahan dapat dibagi menjadi beberapa *stage* dengan keputusan yang harus diambil pada setiap *stage*.
2. Masing-masing *stage* terdiri dari sejumlah *state* yang berhubungan dengan *stage* tersebut.
3. Hasil dari keputusan yang diambil pada setiap *stage* ditransformasikan dari *state* yang bersangkutan ke *state* berikutnya pada *stage* berikutnya.
4. Biaya (*cost*) pada suatu *stage* meningkat secara teratur (*steadily*) dengan bertambahnya jumlah *stage*.
5. Biaya pada suatu *stage* bergantung pada biaya *stage* yang sudah berjalan dan biaya dari *stage* tersebut ke *stage* berikutnya.
6. Keputusan terbaik pada suatu *stage* bersifat independen terhadap keputusan yang dilakukan pada *stage* sebelumnya.
7. Adanya hubungan rekursif yang mengidentifikasikan keputusan terbaik untuk setiap *state* pada *stage* saat ini memberikan keputusan terbaik untuk setiap *state* pada *stage* selanjutnya.
8. Prinsip optimalitas berlaku pada persoalan tersebut.
9. Memiliki struktur submasalah yang saling tumpang tindih dan struktur submasalah optimal.
10. Solusi dinyatakan dalam bentuk fungsi nilai (*value function*) yang merepresentasikan nilai optimal yang dapat dicapai dari suatu *state*.
11. Pemrograman dinamik dapat diterapkan pada masalah deterministik maupun probabilistik (*stochastic*), serta pada *finite horizon* (jumlah *stage* terbatas) maupun *infinite horizon* (jumlah *stage* tak terbatas).
12. Formulasinya sangat bergantung pada perumusan persamaan fungsional (*functional equation*) dan pemilihan *state* yang tepat.
13. Kurang efisien jika dimensi *state* terlalu besar, karena dapat menyebabkan *curse of dimensionality* (lonjakan eksponensial dalam kebutuhan komputasi).

Secara operasional, penerapan prinsip optimalitas direpresentasikan dalam persamaan rekursif. Menurut Hillier & Lieberman (2015), notasi yang digunakan dalam perumusan rekursi adalah sebagai berikut.

N	=	Jumlah *stage*
n	=	Indeks *stage* ke-n, dengan $n = 1, 2, \dots, N$
s_n	=	*State* sekarang untuk *stage* n
x_n	=	Variabel keputusan untuk *stage* n
x_n^*	=	Nilai optimal x_n
$f_n(s_n, x_n)$	=	Nilai total fungsi tujuan dari *stage* ke-n hingga *stage* ke-N, jika sistem dimulai dari *state* s_n, keputusan sekarang adalah x_n, dan keputusan optimal dibuat untuk *stage* berikutnya
$f_n^*(s_n)$	=	Nilai optimal dari $f_n(s_n, x_n)$, yaitu $f_n(s_n, x_n^*)$

Terdapat dua pendekatan utama dalam melakukan perhitungan berdasarkan persamaan rekursi, yaitu rekursi mundur (*backward recursion*) dan rekursi maju (*forward recursion*). Rekursi mundur merupakan proses perhitungan yang dimulai dari *stage* akhir dan bergerak mundur ke *stage* awal, sementara rekursi maju dimulai dari *stage* awal dan bergerak maju ke *stage* akhir (Hillier & Lieberman, 2015; Winston & Goldberg, 2004). Hubungan rekursif tidak hanya terbatas pada penjumlahan, tetapi dapat berupa perkalian atau operasi perbandingan, tergantung pada struktur masalah dan tujuan masalah optimasi yang dimodelkan. Dalam persamaan berikut, g_n merepresentasikan kontribusi langsung pada *stage* n, yang nilainya bergantung pada *state* s_n dan keputusan x_n yang dipilih. Variasi bentuk hubungan rekursif untuk kedua arah tersebut dirangkum dalam Tabel 1.

Tabel 1. Hubungan Rekursi Maju dan Rekursi Mundur

⊗	Rekursi Maju	Rekursi Mundur
+	$f_n(s_n, x_n) = g_n + f_{n-1}^*(s_{n-1})$	$f_n(s_n, x_n) = g_n + f_{n+1}^*(s_{n+1})$
×	$f_n(s_n, x_n) = g_n \times f_{n-1}^*(s_{n-1})$	$f_n(s_n, x_n) = g_n \times f_{n+1}^*(s_{n+1})$
max	$f_n(s_n, x_n) = \max\{g_n, f_{n-1}^*(s_{n-1})\}$	$f_n(s_n, x_n) = \max\{g_n, f_{n+1}^*(s_{n+1})\}$
min	$f_n(s_n, x_n) = \min\{g_n, f_{n-1}^*(s_{n-1})\}$	$f_n(s_n, x_n) = \min\{g_n, f_{n+1}^*(s_{n+1})\}$

Selain dua arah perhitungan tersebut, dalam implementasi komputasi pemrograman dinamik terdapat dua strategi implementasi, yaitu *top-down* dan *bottom-up*. Metode *top-down* menyelesaikan permasalahan dengan memecah masalah menjadi submasalah yang lebih kecil menggunakan teknik rekursi, dan menyimpan hasil setiap submasalah

untuk menghindari perhitungan berulang. Teknik ini dikenal sebagai teknik memoisasi (Cormen *et a*l., 2022). Sebaliknya, metode *bottom-up* memulai penyelesaian dari submasalah terkecil, kemudian membangun solusi untuk permasalahan yang lebih besar berdasarkan hasil submasalah tersebut menggunakan teknik tabulasi (Bellman, 1957; Cormen *et al*., 2022).

Dengan demikian, dapat dikatakan bahwa rekursi maju dan mundur berkaitan dengan arah atau urutan logika perhitungan dalam formulasi model, sedangkan *top-down* dan *bottom-up* merupakan strategi implementasi komputasi. Pemilihan pendekatan yang sesuai bergantung pada karakteristik masalah yang dihadapi, ketersediaan kondisi awal atau akhir, serta efisiensi yang ingin dicapai dalam proses komputasi.

Berdasarkan sifat parameternya, pemrograman dinamik dibedakan menjadi deterministik dan probabilistik. Pemrograman dinamik deterministik digunakan ketika transisi antar *state* sepenuhnya ditentukan oleh *state* dan keputusan sebelumnya, tanpa melibatkan unsur ketidakpastian atau probabilitas (Denardo, 1982). Sebaliknya, pemrograman dinamik probabilistik (*stochastic*) diterapkan ketika transisi ke *state* berikutnya mengikuti suatu distribusi probabilitas, sehingga fungsi nilainya dinyatakan dalam bentuk nilai harapan (*expected value*) (Denardo, 1982). Oleh karena itu, penelitian ini menggunakan pendekatan deterministik karena semua parameter bersifat pasti.

Proses penyelesaian masalah dengan pemrograman dinamik dapat dipecah menjadi empat tahapan yang berurutan. Pertama, karakterisasi struktur dari solusi optimal, meliputi pembagian masalah menjadi beberapa submasalah. Kedua, mendefinisikan fungsi rekursif yang memberikan nilai pada solusi optimal. Ketiga, menghitung nilai dari solusi optimal secara maju atau mundur menggunakan fungsi rekursif yang telah dibuat. Keempat, menyusun solusi optimal berdasarkan informasi hasil komputasi. Langkah 1-3 menjadi landasan solusi algoritma pemrograman dinamik terhadap masalah dan langkah 4 dapat dilewati apabila hanya nilai dari solusi optimal yang ingin diperoleh (Cormen *et al*., 2022).

METODE

RANCANGAN PENELITIAN

Penelitian ini merupakan penelitian terapan dengan pendekatan kuantitatif. Kerangka pemecahan masalah dilakukan melalui beberapa tahapan sistematis. Tahapan penelitian di awali dengan identifikasi permasalahan rute silaturahmi dari Randudongkal menuju empat desa tujuan, yaitu Mejagong, Moga, Warungpring, dan Sikasur. Permasalahan tersebut kemudian dimodelkan dalam bentuk *Traveling Salesman Problem* (TSP). Setiap lokasi direpresentasikan sebagai simpul dalam graf lengkap berbobot, dengan Randudongkal sebagai simpul awal dan akhir. Tahap selanjutnya adalah penerapan algoritma pemrograman dinamik deterministik untuk memperoleh rute optimal dengan jarak tempuh minimum.

POPULASI DAN SAMPEL PENELITIAN

Populasi dalam penelitian ini adalah seluruh lokasi yang terlibat dalam permasalahan rute silaturahmi. Sampel penelitian mencakup lima lokasi, yaitu Randudongkal sebagai titik awal dan akhir perjalanan serta empat desa tujuan, yaitu Mejagong, Moga, Warungpring, dan Sikasur. Seluruh populasi digunakan sebagai sampel penelitian karena jumlah lokasi yang terbatas dan seluruhnya relevan terhadap permasalahan yang dikaji.

TEKNIK PENGUMPULAN DATA DAN PENGEMBANGAN INSTRUMEN

Teknik pengumpulan data dalam penelitian ini dilakukan melalui studi dokumentasi dengan memanfaatkan layanan peta digital *Google Maps*. Data yang dikumpulkan berupa jarak aktual antar lokasi yang menjadi objek penelitian. Jarak antar lokasi diperoleh dengan memilih rute tercepat dan dapat dilalui kendaraan yang direkomendasikan oleh *Google Maps*.

Intrumen penelitian yang digunakan adalah aplikasi *Google Maps* sebagai sumber data jarak antar lokasi. Data jarak yang diperoleh kemudian disusun dan direpresentasikan dalam bentuk matriks jarak berbobot, yang selanjutnya digunakan sebagai input dalam pemodelan *Traveling Salesman Problem* (TSP).

TEKNIK ANALISIS DATA

Teknik analisis data dilakukan dengan menerapkan algoritma pemrograman dinamik

deterministik menggunakan pendekatan rekursi mundur (*backward recursion*) berdasarkan prinsip optimalitas Bellman. Analisis diawali dengan formulasi model *Traveling Salesman Problem* (TSP) dalam bentuk graf lengkap berbobot, dengan setiap simpul merepresentasikan lokasi dan setiap sisi memiliki bobot berupa jarak antar lokasi.

Algoritma pemrograman dinamik digunakan untuk menghitung solusi optimal dengan memecah permasalahan utama menjadi submasalah yang lebih kecil dan saling berkaitan. Proses analisis dilakukan secara bertahap hingga memperoleh dua keluaran utama, yaitu urutan kunjungan yang membentuk rute optimal serta total jarak minimum yang harus ditempuh.

HASIL DAN PEMBAHASAN

DESKRIPSI DATA

Penelitian ini menggunakan data jarak antar lokasi yang diperoleh dari lima lokasi yang menjadi objek kajian. Lokasi tersebut terdiri dari Randudongkal sebagai titik awal dan akhir perjalanan, serta empat desa tujuan, yaitu Mejagong, Moga, Warungpring, dan Sikasur. Data jarak diperoleh dari *Google Maps* dengan mempertimbangkan rute tercepat yang dapat dilalui oleh kendaraan. Kelima lokasi tersebut kemudian direpresentasikan dalam bentuk titik atau simpul untuk memudahkan pemodelan. Daftar lokasi beserta penamaan titik yang digunakan dalam penelitian ini disajikan pada Tabel 2.

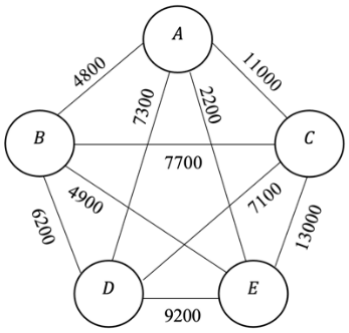
Tabel 2. Lokasi Kunjungan Silaturahmi

Lokasi	Titik
Randudongkal	A
Mejagong	B
Moga	C
Warungpring	D
Sikasur	E

Secara geografis, kelima lokasi tersebut berada dalam satu wilayah administratif yang saling terhubung melalui jaringan jalan darat. Persebaran lokasi kunjungan silaturahmi tersebut ditampilkan pada Gambar 1.

Gambar 1. Peta Lokasi Kunjungan Silaturahmi

Berdasarkan persebaran lokasi pada Gambar 1, diagram jaringan antar lokasi ditunjukkan oleh Gambar 2.



Gambar 2. Diagram Jaringan Antar Lokasi

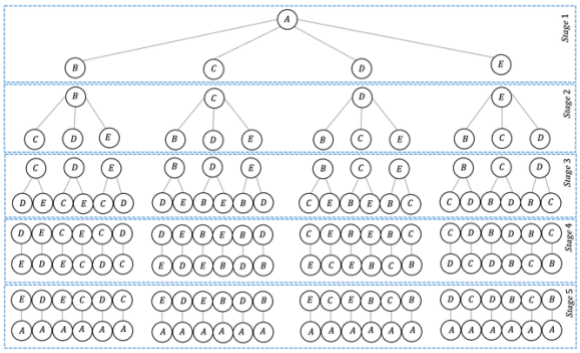
Berdasarkan Gambar 2, selanjutnya disusun matriks jarak antar lokasi yang ditampilkan pada Tabel 3. Matriks jarak tersebut memuat jarak tempuh antara setiap pasangan lokasi dalam satuan meter (m). Nilai pada diagonal utama bernilai nol yang menunjukkan jarak suatu lokasi ke lokasi itu sendiri, sedangkan nilai di luar diagonal merepresentasikan jarak antar lokasi yang berbeda.

Tabel 3. Jarak Antar Lokasi

Lokasi	A	B	C	D	E
A	0	4800	11000	7300	2200
B	4800	0	7700	6200	4900
C	11000	7700	0	7100	13000
D	7300	6200	7100	0	9200
E	2200	4900	13000	9200	0

PENYELESAIAN DENGAN PEMROGRAMAN DINAMIK

Berdasarkan data jarak antar lokasi pada Tabel 3, pembagian *stage* dalam penyelesaian *Traveling Salesman Problem* (TSP) pada penentuan rute silaturahmi yang optimal dapat diilustrasikan secara lebih rinci sebagaimana ditunjukkan pada Gambar 3.



Gambar 3. Pembagian *Stage*

Penyelesaian permasalahan ini dilakukan dengan menggunakan pendekatan rekursi mundur, dengan persamaan rekursi yang dirumuskan sebagai berikut.

$$f_n^*(i,S) = \min_{j \in S}\{c_{ij} + f_{n+1}^*(j, S-\{j\})\} \qquad (5)$$

Keterangan:

n = *Stage*
i = Lokasi saat ini
j = Lokasi tujuan berikutnya
c_{ij} = Jarak lokasi i ke lokasi j
S = Himpunan lokasi yang belum dikunjungi

Kondisi batas:

$$f(i,\emptyset) = c_{i1} \qquad (6)$$

Setelah pembagian *stage* dan perumusan persamaan rekursi dilakukan, langkah selanjutnya adalah melakukan perhitungan nilai fungsi biaya optimal pada setiap *stage* secara bertahap. Perhitungan dimulai dari *stage* terakhir dan dilanjutkan secara bertahap ke *stage* sebelumnya sesuai dengan pendekatan rekursi mundur yang digunakan.

Perhitungan *Stage* 5

Berdasarkan persamaan (5) dan (6), perhitungan pada *stage* 5 dilakukan secara sistematis, dengan hasil perhitungan disajikan pada Tabel 4.

Tabel 4. Perhitungan *Stage* 5

(i,S)	c_{i1}	$f_5^*(i,\emptyset)$	j^*
	A		
B	4800	4800	A
C	11000	11000	A
D	7300	7300	A
E	2200	2200	A

Nilai fungsi biaya optimal yang diperoleh pada *stage* 5, selanjutnya digunakan sebagai biaya masa depan dalam perhitungan *stage* 4.

Perhitungan *Stage* 4

Berdasarkan persamaan (5), perhitungan pada *stage* 4 dilakukan secara sistematis, dengan hasil perhitungan disajikan pada Tabel 5.

Tabel 5. Perhitungan *Stage* 4

(i,S)	$c_{ij} + f_5^*(j,S-\{j\})$				$f_4^*(i,S)$	j^*
	B	C	D	E		
$(B,\{C\})$	–	18700	–	–	18700	C
$(B,\{D\})$	–	–	13500	–	13500	D
$(B,\{E\})$	–	–	–	7100	7100	E
$(C,\{B\})$	12500	–	–	–	12500	B
$(C,\{D\})$	–	–	14400	–	14400	D
$(C,\{E\})$	–	–	–	15200	15200	E
$(D,\{B\})$	11000	–	–	–	11000	B
$(D,\{C\})$	–	18100	–	–	18100	C
$(D,\{E\})$	–	–	–	11400	11400	E
$(E,\{B\})$	9700	–	–	–	9700	B
$(E,\{C\})$	–	24000	–	–	24000	C
$(E,\{D\})$	–	–	16500	–	16500	D

Nilai fungsi biaya optimal yang diperoleh pada *stage* 4, selanjutnya digunakan sebagai biaya masa depan dalam perhitungan *stage* 3.

Perhitungan *Stage* 3

Berdasarkan persamaan (5), perhitungan pada *stage* 3 dilakukan secara sistematis, dengan hasil perhitungan disajikan pada Tabel 6.

Tabel 6. Perhitungan *Stage* 3

(i,S)	$c_{ij} + f_4^*(j,S-\{j\})$				$f_3^*(i,S)$	j^*
	B	C	D	E		
$(B,\{C,D\})$	-	22100	24300	-	22100	C
$(B,\{C,E\})$	-	22900	-	28900	22900	C
$(B,\{D,E\})$	-	-	17600	21400	17600	D
$(C,\{B,D\})$	21200	-	18100	-	18100	D
$(C,\{B,E\})$	14800	-	-	22700	14800	B
$(C,\{D,E\})$	-	-	18500	29500	18500	D
$(D,\{B,C\})$	24900	19600	-	-	19600	C
$(D,\{B,E\})$	13300	-	-	18900	13300	B
$(D,\{C,E\})$	-	22300	-	33200	22300	C
$(E,\{B,C\})$	23600	25500	-	-	23600	B
$(E,\{B,D\})$	18400	-	20200	-	18400	B
$(E,\{C,D\})$	-	27400	27300	-	27300	D

Nilai fungsi biaya optimal yang diperoleh pada *stage* 3, selanjutnya digunakan sebagai biaya masa depan dalam perhitungan *stage* 2.

Perhitungan *Stage* 2

Berdasarkan persamaan (5), perhitungan pada *stage* 2 dilakukan secara sistematis, dengan hasil perhitungan disajikan pada Tabel 7.

Tabel 7. Perhitungan *Stage* 2

(i,S)	$c_{ij} + f_3^*(j,S-\{j\})$				$f_2^*(i,S)$	j^*
	B	C	D	E		
$(B,\{C,D,E\})$	-	26200	28500	32200	26200	C
$(C,\{B,D,E\})$	25300	-	20400	31400	20400	D
$(D,\{B,C,E\})$	29100	21900	-	32800	21900	C
$(E,\{B,C,D\})$	27000	31100	28800	-	27000	B

Nilai fungsi biaya optimal yang diperoleh pada *stage* 2, selanjutnya digunakan sebagai biaya masa depan dalam perhitungan *stage* 1.

Perhitungan *Stage* 1

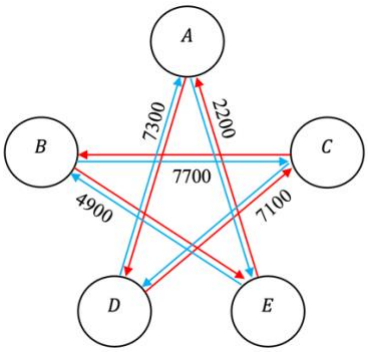
Berdasarkan persamaan (5), perhitungan pada *stage* 1 dilakukan secara sistematis, dengan hasil perhitungan disajikan pada Tabel 8.

Tabel 8. Perhitungan *Stage* 1

(i,S)	$c_{ij}+f_2^*(j,S-\{j\})$				$f_1^*(i,S)$	j^*
	B	C	D	E		
$(A,\{B,C,D,E\})$	31000	31400	29200	29200	29200	D, E

Hasil optimasi menunjukkan bahwa total jarak minimum perjalanan yang diperoleh adalah sebesar 29.200 meter dengan dua rute optimal yang bersifat simetris, yaitu:

1. $A \rightarrow D \rightarrow C \rightarrow B \rightarrow E \rightarrow A =$ Randudongkal ke Warungpring ke Moga ke Mejagong ke Sikasur ke Randudongkal.
2. $A \rightarrow E \rightarrow B \rightarrow C \rightarrow D \rightarrow A =$ Randudongkal ke Sikasur ke Mejagong ke Moga ke Warungpring ke Randudongkal.



Gambar 4. Solusi Optimal

Munculnya lebih dari satu rute optimal dengan nilai biaya yang sama merupakan karakteristik umum dari *Traveling Salesman Problem* (TSP) simetris, yaitu matriks jarak memenuhi sifat sehingga lintasan yang ditempuh secara berlawanan arah menghasilkan total biaya yang identik. Selain itu, hasil perhitungan pada setiap *stage* menunjukkan bahwa keputusan optimal pada suatu *stage* tidak dipengaruhi oleh urutan kunjungan sebelumnya, melainkan hanya bergantung pada *state* saat ini dan himpunan lokasi yang belum dikunjungi. Penelitian ini secara langsung memvalidasi penerapan prinsip optimalitas Bellman pada permasalahan rute silaturahmi yang dimodelkan sebagai *Traveling Salesman Problem* (TSP). Meskipun hasil penelitian menunjukkan bahwa pemrograman dinamik mampu memberikan solusi optimal untuk permasalahan rute silaturahmi yang dimodelkan sebagai *Traveling Salesman Problem* (TSP), penelitian ini memiliki keterbatasan, antara lain jumlah lokasi yang dianalisis masih terbatas pada satu titik awal dan empat desa tujuan sehingga kompleksitas permasalahan relatif kecil dan belum merepresentasikan kasus berskala besar, serta penelitian ini hanya berfokus pada pendekatan pemrograman dinamik deterministik tanpa melakukan perbandingan dengan metode lainnya.

PENUTUP

SIMPULAN

Berdasarkan hasil dan pembahasan penelitian, permasalahan penentuan rute silaturahmi dengan titik awal dan akhir di Randudongkal yang meliputi empat desa tujuan yaitu Mejagong, Moga, Warungpring, dan Sikasur, berhasil dimodelkan sebagai *Traveling Salesman Problem* (TSP) dengan merepresentasikan setiap lokasi sebagai simpul dan jarak antar lokasi sebagai bobot sisi pada sebuah graf lengkap berbobot. Penyelesaian model tersebut menggunakan algoritma pemrograman dinamik deterministik dengan pendekatan rekursi mundur terbukti mampu menghasilkan solusi optimal secara eksak, dengan total jarak minimum 29.200 m, serta menghasilkan dua rute optimal yang bersifat simetris, yaitu Randudongkal – Warungpring – Moga – Mejagong – Sikasur – Randudongkal dan Randudongkal – Sikasur – Mejagong – Moga – Warungpring – Randudongkal. Hasil ini menunjukkan bahwa pemrograman dinamik efektif diterapkan pada permasalahan TSP berskala kecil hingga menengah dengan parameter yang bersifat deterministik dan statis. Oleh karena itu, penelitian ini tidak hanya memberikan solusi optimal untuk kasus rute silaturahmi yang dikaji, tetapi juga dapat menjadi dasar konseptual bagi pengembangan penelitian selanjutnya.

SARAN

Berdasarkan hasil penelitian, beberapa saran dapat dipertimbangkan untuk memperdalam kajian di masa mendatang. Pertama, penelitian lanjutan dapat memperluas cakupan lokasi yang dianalisis dalam permasalahan *Traveling Salesman Problem* (TSP), sehingga performa algoritma pemrograman dinamik dapat dievaluasi pada kondisi yang lebih kompleks. Kedua, penelitian lanjutan dapat mempertimbangkan penggunaan parameter lain, seperti waktu tempuh dan biaya perjalanan yang dipengaruhi oleh kondisi lalu lintas, agar model yang dihasilkan lebih merepresentasikan kondisi perjalanan yang sebenarnya. Ketiga, disarankan untuk melakukan komparasi antara pendekatan pemrograman dinamik dan metode heuristik atau

metaheuristik lainnya untuk menilai keseimbangan antara optimalitas solusi dan efisiensi komputasi. Selain itu, penerapan model optimasi rute ini dapat dikembangkan lebih lanjut pada bidang perencanaan perjalanan sosial atau layanan publik lainnya, sehingga kontribusi metode ini tidak hanya bersifat teoretis, tetapi juga memberikan kontribusi praktis yang lebih luas.

DAFTAR PUSTAKA

Alanzi, E., & Menai, M. E. B. (2025). Solving the traveling salesman problem with machine learning: a review of recent advances and challenges. *Artificial Intelligence Review*, *58*(9), 1–48. https://doi.org/10.1007/s10462-025-11267-x

Anson, A. M. (2024). Enhanced Dynamic Programming Approaches for Efficient Solutions to the Traveling Salesman Problem. *JOURNAL OF COMPUTER SCIENCE APPLICATION AND ENGINEERING*, *2*(2), 24–28. https://doi.org/https://doi.org/10.70356/josapen.v2i2.32

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press.

Bensoussan, A. (2011). *DYNAMIC PROGRAMMING AND INVENTORY CONTROL*. IOS Press BV.

Chandra, A., & Naro, A. (2021). A Comparative Study between Dynamic Programming and Artificial Atom Algorithm for Traveling Salesman Problem. *International Journal of Engineering and Science Applications IJEScA*, *8*(1).

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to Algorithms. In *The MIT Press Cambridge, Massachusetts London, England* (4th ed.). The MIT Press.

Denardo, E. V. (1982). *DYNAMIC PROGRAMMING Models and Applications*. Dover Publications.

Dreyfus, S. (2002). RICHARD BELLMAN ON THE BIRTH OF DYNAMIC PROGRAMMING. *Institute for Operations Research and the Management Sciences (INFORMS)*, *50*(1), 48–51. https://doi.org/https://doi.org/10.1287/opre.50.1.48.17791

Gillett, B. E. (1979). *Introduction to Operations Research: A Computer-Oriented Algorithmic Approach*. Tata McGraw-Hill Publishing Company Ltd.

Hillier, F. S., & Lieberman, G. J. (2015). *Introduction to Operations Research* (10th ed.). McGraw-Hill Education.

Sagala, J. P., Sinaga, R. F., & Simbolon, L. D. (2022). TRAVELLING SALESMAN PROBLEM UNTUK OPTIMASI RUTE TERPENDEK MENGGUNAKAN PROGRAM DINAMIK. *Jurnal Pembelajaran Dan Matematika Sigma (JPMS)*, *8*(2), 438–444. https://doi.org/https://doi.org/10.36987/jpms.v8i2.3399

Sargent, T. J., & Stachurski, J. (2024). *Dynamic Programming: I*.

Singhal, A., & Pandey, P. (2016). TRAVELLING SALESMAN PROBLEMS BY DYNAMIC PROGRAMMING ALGORITHM. *International Journal of Scientific Engineering and Applied Science (IJSEAS)*, *2*(1), 263–267.

Sitinjak, A. A., Pasaribu, E., Simarmata, J. E., Putra, T., & Mawengkang, H. (2018). The Analysis of Forward and Backward Dynamic Programming for Multistage Graph. *IOP Conference Series: Materials Science and Engineering*, *300*(1). https://doi.org/10.1088/1757-899X/300/1/012010

Taha, H. A. (2017). *Operations Research An Introduction* (10th ed.).

Ugonna, A. J., Nkechi, A. M., & Cynthia, O. U. (2024). Travel Salesman Problem Using Dynamic Programming. *Asian Journal of Probability and Statistics*, *26*(6), 63–72. https://doi.org/10.9734/ajpas/2024/v26i6625

Wahyudi, R., Sanjaya, A., & Mahdiyah, U. (2024). Implementasi Dynamic Programming Dalam Menentukan Rute Pengiriman Paket. *SEMNAS INOTEK (Seminar Nasional Inovasi Teknologi)*, *8*, 960–967.

Winston, W. L., & Goldberg, J. B. (2004). Operations Research: Applications and Algorithms. In *Knowledge Creation Diffusion Utilization*. Thomson/Brooks/Cole.

Xu, X., Yuan, H., Liptrott, M., & Trovati, M. (2018). Two phase heuristic algorithm for the multiple-travelling salesman problem. *Soft Computing*, *22*(1), 6567–6581. https://doi.org/10.1007/s00500-017-2705-5