PENGARUH MEKANISME *MULTI-PARENT* PADA *NEAREST NEIGHBOR CROSSOVER* DALAM ALGORITMA GENETIKA UNTUK PENYELESAIAN *TRAVELING SALESMAN PROBLEM*

SKRIPSI

OLEH
BADI'ATUL KHOIROH
NIM. 220601110009

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026

PENGARUH MEKANISME *MULTI-PARENT* PADA *NEAREST NEIGHBOR CROSSOVER* DALAM ALGORITMA GENETIKA UNTUK PENYELESAIAN *TRAVELING SALESMAN PROBLEM*

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)

Oleh
Badi'atul Khoiroh
NIM. 220601110009

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026

PENGARUH MEKANISME *MULTI-PARENT* PADA *NEAREST NEIGHBOR CROSSOVER* DALAM ALGORITMA GENETIKA UNTUK PENYELESAIAN *TRAVELING SALESMAN PROBLEM*

SKRIPSI

Oleh
Badi'atul Khoiroh
NIM. 220601110009

Telah Disetujui Untuk Diuji

Malang, 18 Mei 2026

Dosen Pembimbing I	Dosen Pembimbing II
Mohammad Nafie Jauhari, M.Si.	Erna Herawati, M.Pd.
NIPPPK. 19870218 202321 1 018	NIPPPK. 19760723 202321 2 006

Mengetahui,
Ketua Program Studi Matematika

Dr. Fachrur Rozi, M.Si.
NIP. 19800527 200801 1 012

PENGARUH MEKANISME *MULTI-PARENT* PADA *NEAREST NEIGHBOR CROSSOVER* DALAM ALGORITMA GENETIKA UNTUK PENYELESAIAN *TRAVELING SALESMAN PROBLEM*

SKRIPSI

Oleh
Badi'atul Khoiroh
NIM. 220601110009

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat)

Tanggal 17 Juni 2026

Ketua Penguji	: Ari Kusumastuti, M.Pd., M.Si.	.....
Anggota Penguji 1	: Muhammad Khudzaifah, M.Si.	.....
Anggota Penguji 2	: Mohammad Nafie Jauhari, M.Si.	.....
Anggota Penguji 3	: Erna Herawati, M.Pd.	.....

Mengetahui,
Ketua Program Studi Matematika

Dr. Fachrur Rozi, M.Si.
NIP. 19800527 200801 1 012

PERNYATAAN KEASLIAN TULISAN

Saya bertanda tangan di bawah ini

Nama	: Badi'atul Khoiroh
NIM	: 220601110009
Program Studi	: Matematika
Fakultas	: Sains dan Teknologi
Judul Skripsi	: Pengaruh Mekanisme *Multi-Parent* pada *Nearest Neighbor Crossover* dalam Algoritma Genetika untuk Penyelesaian *Traveling Salesman Problem*

Menyatakan bahwa skripsi yang saya tulis ini merupakan hasil karya sendiri, bukan mengambil alih tulisan atau pemikiran orang lain. Saya menyatakan skripsi ini sebagai pemikiran saya, kecuali dengan mencantumkan sumber kutipan pada daftar rujukan di halaman terakhir. Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini hasil plagiasi atau tiruan orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 17 Juni 2026

Yang membuat pernyataan

Badi'atul Khoiroh

NIM.220601110009

MOTO

“Setiap kesulitan membawa pelajaran, dan setiap pelajaran membawa kita lebih dekat pada tujuan.”

“Ilmu pengetahuan bagaikan sebuah pemikiran yang ingin hidup”
(Abu Bakar Ash-Shiddiq)

PERSEMBAHAN

Dengan memanjatkan puji syukur kepada Allah SWT. Skripsi ini penulis persembahkan sebagai ungkapan rasa syukur dan terima kasih kepada:

Kedua orang tua tercinta, Ayah Suwantah Ahmad Muhsin dan Ibu Shofiyatun yang tidak pernah lelah memberikan kasih sayang, doa, pengorbanan, dan dukungan tanpa syarat. Terima kasih atas setiap pelajaran hidup, nasihat, serta kepercayaan yang selalu diberikan. Semoga skripsi ini menjadi salah satu bentuk kebahagiaan dan kebanggaan untuk kalian.

Kakak-kakak tercinta, yang selalu hadir memberikan semangat, motivasi, dan dukungan dalam setiap proses yang penulis lalui. Terima kasih telah menjadi tempat berbagi cerita, memberikan kekuatan saat penulis merasa lelah, dan selalu meyakinkan bahwa setiap usaha akan menemukan hasil terbaik pada waktunya.

Sahabat dan teman-teman seperjuangan, yang telah mewarnai masa perkuliahan dengan cerita, tawa, pengalaman, serta bantuan yang tidak ternilai. Terima kasih telah menjadi bagian dari perjalanan yang penuh makna ini.

KATA PENGANTAR

Puji syukur kehadirat Allah SWT. atas segala limpahan rahmat, taufik, dan hidayah-Nya sehingga skripsi ini dapat diselesaikan dengan lancar. Shalawat dan salam tidak lupa senantiasa tercurah limpahkan kepada baginda Nabi Agung Muhammad Saw yang telah membawa kita dari zaman jahiliyyah ke zaman yang terang benderang yaitu addinul Islam.

Terselesaikannya skripsi ini tidak lepas dari doa, bantuan, bimbingan, serta arahan dari berbagai pihak, karena itu penulis mengucapkan terimakasih kepada:

1. Prof. Dr. Hj. Ilfi Nur Diana, M.Si., CAHRM., CRMP, selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. Agus Mulyono, M.Kes., selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Fachrur Rozi, M.Si., selaku Ketua Program Studi Matematika, Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Mohammad Nafie Jauhari, M.Si., selaku Dosen Pembimbing I yang senantiasa memberikan berbagai pengetahuan, nasehat, motivasi, dan arahan selama proses penyusunan skripsi ini.
5. Erna Herawati, M.Pd, selaku Dosen Pembimbing II yang telah memberikan arahan dan masukan dalam menyusun skripsi ini.
6. Ari Kusumastuti, M.Pd, M.Si., selaku Ketua Penguji yang telah meluangkan waktu, tenaga, dan perhatian dalam memberikan masukan, arahan, serta koreksi yang sangat berharga demi penyempurnaan skripsi ini. Saran dan pandangan beliau menjadi dorongan penting bagi penulis untuk meningkatkan kualitas penelitian ini.
7. Muhammad Khudzaifah, M.Si., selaku Penguji I yang dengan penuh kesabaran memberikan kritik, saran, serta bimbingan selama proses ujian dan perbaikan skripsi. Setiap masukan yang diberikan sangat membantu penulis dalam memperbaiki kekurangan dan menyempurnakan penulisan skripsi ini
8. Seluruh Dosen Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang
9. Ayah penulis, Suwantah Ahmad Muhsin, yang dengan caranya sendiri selalu menguatkan langkah penulis, memberikan dukungan tanpa banyak kata, serta

menghadirkan keteguhan yang menjadi sandaran dalam setiap proses yang tidak mudah. Dalam diamnya, beliau adalah alasan penulis tetap berdiri dan terus melangkah hingga mampu menyelesaikan skripsi ini.

10. Ibu penulis, Shofiyatun, yang dengan ketulusan tak pernah berhenti menyebut nama penulis dalam setiap doa, yang diam-diam menyimpan lelahnya sendiri demi memastikan penulis tetap kuat, serta yang selalu menjadi tempat pulang ketika dunia terasa terlalu berat. Kasih sayang beliau adalah kekuatan paling sunyi yang menjaga penulis untuk tidak menyerah hingga skripsi ini dapat diselesaikan.
11. Seluruh mahasiswa angkatan 2022 yang memberikan *support*.
12. Seluruh teman-teman yang senantiasa memberikan bantuan, dukungan dan semangat kepada penulis dalam berbagai kondisi.
13. Seluruh pihak yang sudah terlibat secara langsung maupun tidak langsung yang tidak bisa penulis tuliskan satu per satu.

Dengan seluruh kerendahan hati, penulis menyadari skripsi ini masih belum bisa dikatakan sempurna. Maka dari itu, penulis berharap kritik dan saran yang membangun untuk penyempurnaan skripsi ini. Semoga karya ini mampu memberikan manfaat bagi orang banyak. Aamin ya Rabbal ‘Alamin.

Malang, 6 Juni 2026

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SIMBOL

G	: Graf yang merepresentasikan jaringan kota dalam TSP
V	: Himpunan simpul (kota) dalam graf
E	: Himpunan sisi (jarak antar kota) dalam graf
n	: Jumlah kota atau jumlah individu dalam populasi
c_{ij}	: Jarak antara kota i dan kota j
x_{ij}	: Variabel keputusan bernilai 1 jika rute dari kota i ke j dipilih, dan 0 jika tidak
x_i	: Koordinat sumbu x pada kota ke-i
y_i	: Koordinat sumbu y pada kota ke-i
P_i	: Probabilitas seleksi individu ke-i
C_i	: Probabilitas kumulatif individu ke-i
r	: Bilangan acak pada *roulette wheel selection*
X	: Ruang solusi dalam permasalahan optimasi
$f(x)$	: Fungsi objektif yang akan diminimalkan atau dimaksimalkan
$f(i)$	: Nilai *fitness* individu ke-i
$Cost(i)$	: Total jarak tempuh dari rute solusi ke-i
$Cost_{max}$	: Nilai cost terbesar dalam populasi
$Cost_{min}$	: Nilai cost terkecil dalam populasi

DAFTAR LAMPIRAN

ABSTRAK

Khoiroh, Badi'atul. 2026. **Pengaruh Mekanisme *Multi-Parent* pada *Nearest Neighbor Crossover* dalam Algoritma Genetika untuk Penyelesaian *Traveling Salesman Problem***. Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Mohammad Nafie Jauhari, M.Si. (II) Erna Herawati, M.Pd.

Kata Kunci: *Traveling Salesman Problem*, Algoritma Genetika, *Nearest Neighbor Crossover*, *Multi-Parent Crossover*, Optimasi Kombinatorial

Traveling Salesman Problem (TSP) merupakan permasalahan optimasi kombinatorial yang tergolong NP-hard, di mana tujuannya adalah menemukan lintasan terpendek yang mengunjungi setiap kota tepat satu kali dan kembali ke kota asal. Salah satu pendekatan yang digunakan untuk menyelesaikan TSP adalah algoritma genetika (AG), yang efektivitasnya sangat dipengaruhi oleh operator *crossover*. Penelitian ini mengembangkan operator *Nearest Neighbor Crossover* (NNX) dengan menerapkan mekanisme *multi-parent* sehingga menghasilkan *Multi-Parent Nearest Neighbor Crossover* (MPNNX) yang melibatkan tiga induk dalam proses *crossover*. Pengujian dilakukan menggunakan dataset pr1002 dari TSPLIB yang memiliki 1.002 titik kota, dengan 35 kali percobaan untuk masing-masing metode menggunakan bahasa pemrograman Python di Google Colaboratory. Hasil penelitian menunjukkan bahwa MPNNX menghasilkan nilai rata-rata total jarak sebesar 1.955.529,184 dan nilai minimum sebesar 1.480.099,216, keduanya lebih kecil dibandingkan NNX yang menghasilkan rata-rata 2.392.412,232 dan minimum 1.611.346,468. Perbedaan performa kedua metode terbukti signifikan secara statistik berdasarkan uji Mann-Whitney U dengan p-value sebesar 0,0000046281. Secara keseluruhan, MPNNX lebih unggul dalam menghasilkan rute yang lebih pendek, sedangkan NNX menunjukkan konsistensi hasil yang lebih baik antarpercobaan.

ABSTRACT

Khoiroh, Badi'atul. 2026. **The Effect of Multi-parent Mechanism on Nearest Neighbor Crossover in Genetic Algorithm for Solving the Traveling Salesman Problem**. Undergraduate Thesis. Mathematics Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisors: (I) Mohammad Nafie Jauhari, M.Si. (II) Erna Herawati, M.Pd.

Keywords: Traveling Salesman Problem, Genetic Algorithm, Nearest Neighbor Crossover, Multi-Parent Crossover, Combinatorial Optimization

The Traveling Salesman Problem (TSP) is an NP-hard combinatorial optimization problem that aims to find the shortest route visiting each city exactly once before returning to the origin. One approach used to solve TSP is the Genetic Algorithm (GA), whose effectiveness is greatly influenced by the *crossover* operator applied. This study develops the Nearest Neighbor Crossover (NNX) operator by incorporating a multi-parent mechanism, producing Multi-Parent Nearest Neighbor Crossover (MPNNX) which involves three parent chromosomes in the crossover process. Testing was conducted using the pr1002 dataset from TSPLIB consisting of 1,002 city nodes, with 35 runs for each method implemented in Python via Google Colaboratory. The results show that MPNNX produces a mean total distance of 1,955,529.184 and a minimum of 1,480,099.216, both lower than NNX which yields a mean of 2,392,412.232 and a minimum of 1,611,346.468. The performance difference between both methods is statistically significant based on the Mann-Whitney U test with a p-value of 0.0000046281. Overall, MPNNX is superior in generating shorter routes, while NNX demonstrates greater consistency across runs.

مستخلص البحث

الخيرة، بديعة. ٢٠٢٦. **تأثير آلية "متعدد الآباء" على تقاطع "الأقرب" في الخوارزمية الجينية لحل مشكلة البائع المتجول**. أطروحة البكالوريوس. قسم الرياضيات، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية في مالانج.
المشرف: (١) محمد نافع جوهري،الماجستير في العلوم. (٢) ايرنا هيراواتى، الماجستيرة في التربية.

الكلمات الاساسية: مشكلة البائع المتجول، الخوارزمية الجينية، تقاطع أقرب الجيران، تقاطع متعدد الآباء، التحسين التوافقي

مشكلة البائع المتجول (*TSP*) هي مشكلة تحسين تركيبية تُصنف ضمن فئة *NP-hard* ، حيث يتمثل الهدف منها في إيجاد أقصر مسار يزور كل مدينة مرة واحدة فقط ثم يعود إلى المدينة الأصلية. ومن بين الأساليب المستخدمة لحل مشكلة البائع المتجول (*TSP*) الخوارزمية الجينية (*AG*) ، التي تتأثر فعاليتها بشكل كبير بمشغل التهجين. تطور هذه الدراسة عامل التهجين (*NNX*) *Nearest Neighbor Crossover* من خلال تطبيق آلية متعددة الأبوين، مما ينتج عنه (*MPNNX*) *Multi-Parent Nearest Neighbor Crossover* الذي يشمل ثلاثة آباء في عملية التهجين. تم إجراء الاختبار باستخدام مجموعة البياناتpr ١٠٠٢ من *TSPLIB* التي تحتوي على ١,٠٠٢ نقطة مدينة، مع ٣٥ تجربة لكل طريقة باستخدام لغة البرمجة *Python* في *Google Colaboratory.* أظهرت نتائج البحث أن *MPNNX* أنتجت متوسطًا إجماليًا للمسافة يبلغ ١,٩٥٥,٥٢٩,١٨٤ وقيمة دنيا تبلغ ١,٤٨٠,٠٩٩,٢١٦، وكلاهما أقل مقارنةً بـ *NNX* التي أنتجت متوسطًا يبلغ ٢,٣٩٢,٤١٢,٢٣٢ وقيمة دنيا تبلغ ١,٦١١,٣٤٦,٤٦٨. وقد ثبت أن الفرق في أداء الطريقتين ذو دلالة إحصائية بناءً على اختبار *Mann-Whitney U* بقيمة p تبلغ ٠,٠٠٠٠٠٤٦٢٨١. وبشكل عام، يتفوق *MPNNX* في إنتاج مسارات أقصر، بينما يُظهر *NNX* اتساقًا أفضل في النتائج بين التجارب.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Permasalahan optimasi merupakan salah satu kajian utama dalam matematika terapan yang berfokus pada pencarian solusi terbaik dari berbagai alternatif yang memenuhi kendala tertentu agar hasilnya efisien dan optimal (Boyd & Vandenberghe, 2004). Dalam konteks kombinatorial, optimasi banyak diterapkan untuk menyelesaikan persoalan penentuan rute, penjadwalan, dan alokasi sumber daya yang melibatkan struktur-struktur matematis seperti graf dan himpunan (Hillier & Lieberman, 2015). Salah satu permasalahan yang paling dikenal dalam bidang ini adalah *Traveling Salesman Problem* (TSP), yang menjadi model dasar untuk berbagai persoalan optimasi jalur dan distribusi (Applegate et al., 2011).

TSP menggambarkan permasalahan pencarian lintasan terpendek yang melewati setiap kota tepat satu kali dan kembali ke titik awal, sehingga dapat dimanfaatkan sebagai model perencanaan distribusi barang, pengiriman surat, maupun penjadwalan transportasi (Kharisma et al., 2024). TSP dikategorikan sebagai *NP-hard problem*, karena kompleksitas komputasinya meningkat eksponensial seiring bertambahnya jumlah kota atau titik yang dikunjungi (Alkafaween et al., 2024; Applegate et al., 2011). Dalam konteks dunia nyata, TSP semakin rumit ketika mempertimbangkan faktor-faktor seperti kemacetan lalu lintas, kapasitas kendaraan, waktu layanan pelanggan, dan kondisi geografis yang dinamis (Fontaine et al., 2023). Oleh karena itu, dibutuhkan pendekatan komputasional cerdas yang mampu menghasilkan solusi mendekati optimal secara efisien dalam waktu terbatas.

Pengumpulan data dalam penelitian ini dilakukan dengan menggunakan data benchmark *Traveling Salesman Problem* (TSP). Data diperoleh melalui repositori GitHub yang bersumber dari TSPLIB. Dataset yang digunakan yaitu pr1002 yang merepresentasikan permasalahan TSP dengan jumlah titik sebanyak 1002. Dataset berupa koordinat kota yang kemudian diolah dengan menghitung jarak Euclid antar titik dan disusun menjadi matriks jarak. Data tersebut dikonversi serta disimpan dalam format Microsoft Excel untuk memudahkan proses pengolahan di Google Colaboratory menggunakan bahasa pemrograman Python. Sumber dataset dapat diakses melalui kata "TSPLIB". Matriks jarak yang dihasilkan selanjutnya digunakan sebagai input pada algoritma genetika dalam proses optimasi rute (Reinelt, 1991).

Berbagai metode metaheuristik telah dikembangkan untuk menyelesaikan TSP, seperti *Simulated Annealing*, *Ant Colony Optimization* (ACO), dan algoritma genetika (Kirkpatrick et al., 2023). Algoritma genetika adalah metode pencarian solusi yang bekerja berdasarkan prinsip evolusi makhluk hidup dengan memperbaiki sekumpulan solusi awal (populasi) secara bertahap melalui proses seleksi, *crossover*, dan mutasi (Goldberg, 1989). GA menonjol karena kemampuannya menyeimbangkan antara eksplorasi dan eksploitasi ruang solusi melalui mekanisme seleksi, *crossover*, dan mutasi (Goldberg, 1989; Holland, 1975; Isa et al., 2024). Namun, efektivitas GA sangat bergantung pada desain *crossover* yang digunakan untuk menggabungkan gen antar individu (Pretorius & Pillay, 2024; Srinivas & Patnaik, 1994). Operator klasik seperti *Partially Mapped Crossover* (PMX), *Order Crossover* (OX), dan *Cycle Crossover* (CX), yang hanya

melibatkan dua induk sering mengalami konvergensi prematur, sehingga mengurangi keragaman genetik (Dalkilic et al., 2025; Potvin, n.d.).

Salah satu operator *crossover* yang cukup populer untuk penyelesaian TSP adalah *Nearest Neighbor Crossover* (NNX), yang memilih kota berikutnya berdasarkan jarak terdekat dari dua *parent* (Onder, 2007; Sharma et al., 2024). NNX bekerja dengan memilih kota berikutnya berdasarkan jarak terdekat dari dua *parent*, sehingga menghasilkan rute yang lebih efisien dibandingkan operator urutan tradisional seperti OX dan PMX. Meski demikian, penelitian sebelumnya hanya membahas NNX dengan dua *parent*, sementara potensi penggunaan *multi-parent* NNX (melibatkan lebih dari dua induk) belum dieksplorasi secara mendalam (Mahmoudinazlou & Kwon, 2024; Onder, 2007). Sementara itu, penggunaan lebih dari dua induk berpotensi meningkatkan keberagaman kromosom dan memperkaya kombinasi genetik yang dihasilkan selama proses evolusi (Dalkilic et al., 2025).

Penelitian ini mengusulkan pengembangan operator *crossover* pada GA dengan menerapkan mekanisme *multi-parent* pada *Nearest Neighbor Crossover* (NNX), yaitu *crossover* tiga induk (*three-parent crossover*). Pengembangan operator ini diharapkan dapat meningkatkan keanekaragaman genetik populasi, memperkecil kemungkinan konvergensi dini, dan menghasilkan rute yang lebih efisien. Dengan demikian, penelitian ini memberikan kontribusi terhadap pengembangan algoritma genetika adaptif berbasis *multi-parent crossover* yang lebih efektif untuk penyelesaian masalah TSP.

Selain memiliki dasar ilmiah, penelitian ini juga mencerminkan nilai-nilai Islam yang menekankan pentingnya usaha, perubahan, dan perbaikan berkelanjutan. Hal ini sejalan dengan firman Allah dalam Q.S. Ar-Ra’d ayat 11:

لَهٗ مُعَقِّبٰتٌ مِّنْۢ بَيْنِ يَدَيْهِ وَمِنْ خَلْفِهٖ يَحْفَظُوْنَهٗ مِنْ اَمْرِ اللّٰهِ ۗاِنَّ اللّٰهَ لَا يُغَيِّرُ مَا بِقَوْمٍ حَتّٰى يُغَيِّرُوْا مَا بِاَنْفُسِهِمْ ۗوَاِذَآ اَرَادَ اللّٰهُ بِقَوْمٍ سُوْۤءًا فَلَا مَرَدَّ لَهٗ ۚوَمَا لَهُمْ مِّنْ دُوْنِهٖ مِنْ وَّالٍ ﴿١١﴾

(Kementerian Agama Republik Indonesia, 2022)

"Baginya (manusia) ada (malaikat-malaikat) yang menyertainya secara bergilirian dari depan dan belakangnya yang menjaganya atas perintah Allah. Sesungguhnya Allah tidak mengubah keadaan suatu kaum hingga mereka mengubah apa yang ada pada diri mereka. Apabila Allah menghendaki keburukan terhadap suatu kaum, tidak ada yang dapat menolaknya, dan sekali-kali tidak ada pelindung bagi mereka selain dia." (Q.S. Ar-Ra'd: 11)

Menurut (Katsir & Umar, 2005) pada buku "*Tafsir Ibnu Katsir*", ayat ini dijelaskan bahwa setiap manusia senantiasa dijaga oleh para malaikat yang bergantian siang dan malam atas perintah Allah SWT. Meskipun demikian, manusia tetap memiliki tanggung jawab untuk berikhtiar dan memperbaiki dirinya. Allah SWT tidak akan mengubah keadaan suatu kaum menjadi keadaan yang lebih baik tanpa adanya usaha dari mereka sendiri. Apabila manusia berpaling dari ketaatan, maka kenikmatan dapat berubah menjadi kesulitan, dan sebaliknya, jika mereka berusaha memperbaiki diri, maka Allah SWT akan memperbaiki keadaan mereka.

Pesan tersebut memiliki keterkaitan yang erat dengan penelitian ini, yang berupaya melakukan pengembangan terhadap sistem yang telah ada. Penerapan mekanisme *multi-parent* pada operator *Nearest Neighbor Crossover* (NNX) menjadi *Multi-Parent Nearest Neighbor Crossover* (MPNNX) mencerminkan bentuk ikhtiar manusia untuk terus berinovasi dan meningkatkan hasil agar lebih baik. Dengan demikian, penelitian ini tidak hanya berorientasi pada peningkatan performa algoritma secara teknis, tetapi juga mencerminkan nilai-nilai Al-Qur'an yang menekankan pentingnya perubahan, usaha, dan inovasi dalam mencapai hasil yang lebih optimal.

1.2 Rumusan Masalah

Berdasarkan paparan latar belakang di atas, maka penelitian ini akan membahas beberapa rumusan masalah sebagai berikut:

1. Bagaimana penerapan penyelesaian mekanisme *multi-parent* pada *Nearest Neighbor Crossover* dalam algoritma genetika untuk penyelesaian *Traveling Salesman Problem*?
2. Bagaimana perbandingan kinerja *Nearest Neighbor Crossover* dan *Nearest Neighbor Crossover* dengan mekanisme *multi-parent* dalam algoritma genetika untuk penyelesaian *Traveling Salesman Problem* berdasarkan total jarak rute, pola konvergensi, dan variasi hasil percobaan?
3. Apakah mekanisme *multi-parent* memberikan pengaruh yang signifikan terhadap kinerja *Nearest Neighbor Crossover* dalam algoritma genetika untuk penyelesaian *Traveling Salesman Problem*?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah disampaikan, penelitian ini bertujuan untuk:

1. Menjelaskan penerapan mekanisme *multi-parent* pada *Nearest Neighbor Crossover* dalam algoritma genetika untuk penyelesaian *Traveling Salesman Problem*.
2. Membandingkan kinerja *Nearest Neighbor Crossover* dan *Nearest Neighbor Crossover* dengan mekanisme *multi-parent* dalam algoritma genetika untuk penyelesaian *Traveling Salesman Problem* berdasarkan total jarak rute, pola konvergensi, dan variasi hasil percobaan.

3. Mengetahui pengaruh signifikan mekanisme *multi-parent* terhadap kinerja *Nearest Neighbor Crossover* dalam algoritma genetika untuk penyelesaian *Traveling Salesman Problem*.

1.4 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan beberapa manfaat sebagai berikut:

1. Manfaat Teoritis

   Penelitian ini diharapkan dapat memberikan kontribusi terhadap kajian algoritma genetika, khususnya mengenai pengaruh mekanisme *multi-parent* pada operator *Nearest Neighbor Crossover* dalam penyelesaian *Traveling Salesman Problem*. Hasil penelitian ini dapat memperkaya pemahaman mengenai bagaimana penggunaan lebih dari dua *parent* dalam proses *crossover* dapat memengaruhi kualitas solusi, pola konvergensi, dan variasi hasil pada algoritma genetika.

2. Manfaat Praktis

   Penelitian ini diharapkan dapat menjadi referensi bagi peneliti atau pengguna algoritma genetika dalam mempertimbangkan penggunaan mekanisme *multi-parent* pada *Nearest Neighbor Crossover* untuk penyelesaian *Traveling Salesman Problem*. Hasil penelitian ini memberikan gambaran empiris mengenai potensi dan keterbatasan mekanisme *multi-parent*, terutama dalam menghasilkan total jarak rute yang lebih kecil serta dalam menjaga konsistensi hasil antarpercobaan.

1.5 Batasan Masalah

Agar penelitian ini lebih terarah dan sesuai dengan tujuan yang telah ditetapkan, maka ruang lingkup penelitian dibatasi sebagai berikut:

1. Data yang digunakan dalam penelitian ini berupa dataset *Traveling Salesman Problem* dari TSPLIB yaitu pr1002 yang memiliki jumlah simpul 1002.
2. Permasalahan dimodelkan dalam bentuk graf berbobot lengkap (*fully connected graph*) dengan jarak antar kota dihitung menggunakan jarak Euclidean berdasarkan koordinat yang tersedia pada dataset.
3. Operator *crossover* yang digunakan terbatas pada dua jenis, yaitu:
   a. *Nearest Neighbor Crossover* (NNX) yang melibatkan dua induk (*two-parent crossover*)
   b. *Multi-parent Nearest Neighbor Crossover* (MPNNX) yang melibatkan tiga induk (*three-parent crossover*).
4. Parameter algoritma genetika meliputi ukuran populasi, jumlah generasi, elitisme, serta jumlah percobaan (*run*). Nilai parameter ditentukan pada tahap desain eksperimen dan disesuaikan dengan karakteristik dataset penelitian.
5. Operator mutasi yang digunakan adalah *swap mutation*.
6. Kinerja algoritma dievaluasi berdasarkan panjang total rute hasil optimasi dari sejumlah percobaan.

1.6 Definisi Istilah

Untuk memahami konteks dan istilah yang digunakan dalam penelitian ini, berikut adalah definisi dari istilah-istilah kunci yang relevan:

1. Algoritma Genetika

   Algoritma genetika adalah metode optimasi berbasis metaheuristik yang terinspirasi oleh proses evolusi biologis seperti seleksi alam, mutasi, dan rekombinasi. Algoritma ini digunakan untuk menyelesaikan berbagai masalah kompleks, termasuk optimasi, pencarian solusi, dan pembelajaran mesin (Holland, 1975).

2. *Traveling Salesman Problem* (TSP)

   TSP adalah masalah optimasi dalam matematika dan ilmu komputer yang bertujuan menentukan lintasan terpendek untuk mengunjungi setiap kota tepat satu kali dan kembali ke kota asal, sehingga total jarak tempuh menjadi minimum (Alkafaween et al., 2024).

3. Optimasi

   Optimasi adalah proses untuk menemukan solusi terbaik dari berbagai kemungkinan solusi dengan mempertimbangkan kriteria tertentu, seperti efisiensi waktu, biaya, atau sumber daya (Goldberg, 1989).

4. Populasi

   Sekumpulan solusi awal yang dihasilkan secara acak atau ditentukan sebelumnya. Populasi digunakan sebagai basis untuk proses evolusi dalam algoritma genetika (Holland, 1975).

5. Individu

   Setiap solusi potensial dalam populasi disebut individu. Individu biasanya direpresentasikan dalam bentuk kromosom (Soni & Kumar, 2007).

6. Kromosom

   Representasi dari suatu solusi dalam algoritma genetika yang tersusun dari beberapa gen. Bentuk kromosom menyesuaikan jenis masalah, seperti urutan kota pada permasalahan *Traveling Salesman Problem* (TSP) (Goldberg, 1989).

7. Gen

   Bagian terkecil dari kromosom yang merepresentasikan atribut atau elemen tertentu dari solusi. Misalnya, dalam TSP, gen dapat mewakili kota tertentu (Holland, 1975).

8. Fungsi *Fitness*

   Fungsi yang digunakan untuk mengevaluasi kualitas setiap individu dalam populasi. Semakin tinggi nilai *fitness*, semakin baik solusi tersebut. Dalam TSP, nilai *fitness* biasanya terkait dengan panjang total rute (semakin pendek, semakin baik) (Holland, 1975; Zibaei & Mesgari, 2023).

9. Seleksi

   Proses pemilihan solusi terbaik dari populasi untuk dijadikan induk dalam pembuatan solusi baru. Misalnya, rute dengan jarak lebih pendek (lebih baik)

10. Populasi

    Sekumpulan solusi awal yang dihasilkan secara acak atau ditentukan sebelumnya. Populasi digunakan sebagai basis untuk proses evolusi dalam algoritma genetika (Holland, 1975).

11. Individu

    Setiap solusi potensial dalam populasi disebut individu. Individu biasanya direpresentasikan dalam bentuk kromosom (Soni & Kumar, 2007).

12. Kromosom

    Representasi dari suatu solusi dalam algoritma genetika yang tersusun dari beberapa gen. Bentuk kromosom menyesuaikan jenis masalah, seperti urutan kota pada permasalahan *Traveling Salesman Problem* (TSP) (Goldberg, 1989).

13. Gen

    Bagian terkecil dari kromosom yang merepresentasikan atribut atau elemen tertentu dari solusi. Misalnya, dalam TSP, gen dapat mewakili kota tertentu (Holland, 1975).

14. Fungsi *Fitness*

    Fungsi yang digunakan untuk mengevaluasi kualitas setiap individu dalam populasi. Semakin tinggi nilai *fitness*, semakin baik solusi tersebut. Dalam TSP, nilai *fitness* biasanya terkait dengan panjang total rute (semakin pendek, semakin baik) (Holland, 1975; Zibaei & Mesgari, 2023).

15. Seleksi

    Proses pemilihan solusi terbaik dari populasi untuk dijadikan induk dalam pembuatan solusi baru. Misalnya, rute dengan jarak lebih pendek (lebih baik) memiliki peluang lebih besar untuk dipilih dibandingkan rute yang panjang (Katoch et al., 2021).

16. *Crossover*

    Proses kombinasi antara dua kromosom (individu induk) untuk menghasilkan satu atau lebih individu baru (anak) dengan harapan memperoleh solusi yang lebih baik (Soni & Kumar, 2007).

17. *Nearest Neighbor Crossover* (NNX)

    Salah satu jenis operator *crossover* pada algoritma genetika yang memilih kota berikutnya berdasarkan jarak terdekat dari dua induk (*parent*), sehingga menghasilkan rute yang lebih efisien pada permasalahan *Traveling Salesman Problem* (Onder, 2007).

18. Mutasi

    Proses perubahan kecil yang diterapkan pada kromosom untuk menjaga keberagaman dalam populasi dan mencegah algoritma dari terjebak dalam solusi lokal. Contoh mutasi dalam TSP adalah pertukaran posisi dua kota dalam rute (Srinath Murthy & P Mankame, 2024).

19. Generasi

    Satu iterasi lengkap dalam algoritma genetika, yang mencakup evaluasi fungsi *fitness*, seleksi, *crossover*, dan mutasi untuk menghasilkan populasi baru (Goldberg, 1989).

BAB II

KAJIAN TEORI

2.1 Optimasi

Optimasi merupakan cabang ilmu yang berfokus pada pencarian solusi terbaik dari sejumlah alternatif berdasarkan suatu fungsi objektif tertentu. Dalam konteks riset operasi dan komputasi modern, optimasi dimanfaatkan untuk meminimalkan atau memaksimalkan nilai dari suatu fungsi dengan tetap memperhatikan kendala-kendala yang ditetapkan (Zibaei & Mesgari, 2023).

Permasalahan optimasi secara umum dapat dituliskan sebagai:

$$\min_{x \in X} f(x) \quad \text{atau} \quad \max_{x \in X} f(x)$$

di mana X merupakan ruang solusi dan $f(x)$ adalah fungsi objektif. Secara garis besar, optimasi dibagi menjadi optimasi kontinu dan optimasi kombinatorik. Optimasi kontinu memiliki ruang solusi berupa domain kontinu dan biasanya diselesaikan menggunakan metode analitik atau numerik seperti *linear programming*.

Sementara itu, optimasi kombinatorik memiliki ruang solusi berupa himpunan diskrit yang sangat besar sehingga metode eksak tidak lagi efisien untuk ukuran masalah besar. Contohnya adalah *Traveling Salesman Problem* (TSP), *Vehicle Routing Problem*, dan berbagai kasus penjadwalan (Baty et al., 2024). Kerumitan ruang solusi yang meningkat secara eksponensial menyebabkan metode eksak seperti *Branch and Bound* atau *Integer Programming* tidak mampu menangani ukuran besar. Oleh karena itu, metaheuristik seperti *Genetic Algorithm* (GA), *Simulated Annealing*, dan *Ant Colony Optimization* banyak digunakan karena mampu menemukan solusi mendekati optimal secara efisien (Pop et al., 2024).

2.2 *Traveling Salesman Problem*

Traveling Salesman Problem (TSP) merupakan permasalahan optimasi kombinatorial yang bertujuan menentukan lintasan terpendek untuk mengunjungi setiap kota tepat satu kali dan kembali ke kota asal, sehingga total jarak tempuh menjadi minimum (Alkafaween et al., 2024). Masalah TSP dapat dimodelkan menggunakan graf berbobot lengkap:

$$G = (V, E)$$

di mana V adalah himpunan kota dan E adalah himpunan jarak antar kota. Tujuan dari TSP adalah mencari siklus hamilton yang mengunjungi setiap kota tepat satu kali dan kembali ke kota asal dengan total jarak minimum. Fungsi objektifnya dapat dinyatakan sebagai:

$$Min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{2.1}$$

dimana c_{ij} adalah jarak antara kota i dan j, x_{ij} bernilai 1 jika rute dari kota i ke kota j dipilih, dan 0 jika tidak dipilih.

Pada kasus *Euclidean Traveling Salesman Problem* (ETSP), jarak antar kota dihitung berdasarkan koordinat dua dimensi menggunakan rumus *Euclidean distance*. Jika kota ke-i memiliki koordinat (x_i, y_i) dan kota ke-$\boldsymbol{j}$ memiliki koordinat (x_j, y_j), maka jarak antar keduanya dirumuskan sebagai:

$$C_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{2.2}$$

C_{ij} menyatakan jarak antara kota ke-i dan kota ke-j. Nilai x_i dan y_i masing-masing merupakan koordinat sumbu x dan y dari kota ke-i, sedangkan x_j dan y_j merupakan

koordinat sumbu x dan y dari kota ke-j. Persamaan ini digunakan untuk menghitung jarak antar dua titik dalam bidang dua dimensi berdasarkan jarak Euclidean.

Permasalahan ini bersifat *NP-hard*, sehingga waktu komputasi meningkat secara eksponensial terhadap jumlah kota (Pop et al., 2024; Shenmaier, 2022). Variasi TSP bergantung pada karakteristik jarak antar kota, seperti *Symmetric* TSP (STSP), *Asymmetric* TSP (ATSP), atau *Euclidean* TSP (ETSP) (Applegate et al., 2011; Shenmaier, 2022). Pendekatan metaheuristik banyak diterapkan untuk menyelesaikan TSP karena mampu mengeksplorasi ruang solusi yang besar tanpa mengevaluasi semua kemungkinan rute (Pop et al., 2024).

2.3 Algoritma Genetika

Algoritma Genetika (AG) merupakan salah satu metode optimasi berbasis populasi yang meniru prinsip genetika dan evolusi alam. Konsep ini pertama kali diperkenalkan oleh (Holland, 1975) dan kemudian dikembangkan lebih lanjut oleh (Goldberg, 1989) dengan memperkenalkan struktur operator evolusi untuk meningkatkan kualitas solusi. Seiring berjalannya waktu, penelitian mengenai GA terus berkembang untuk mencakup aspek diversitas populasi, hybridisasi metaheuristik, dan aplikasi skala besar (Srinath Murthy & P Mankame, 2024). Dalam algoritma ini, solusi-solusi kandidat disebut kromosom, dan kumpulan kromosom tersebut membentuk suatu populasi yang berevolusi untuk memperoleh solusi terbaik.

Proses dasar algoritma genetika dimulai dengan pembangkitan populasi awal secara acak. Setiap individu dievaluasi menggunakan fungsi *fitness* untuk mengukur kualitasnya. Individu dengan nilai *fitness* tertinggi memiliki peluang

lebih besar untuk terpilih sebagai induk melalui proses seleksi. Selanjutnya, dilakukan proses *crossover* yang bertujuan menghasilkan ketu*run*an baru dengan kombinasi gen yang diharapkan lebih baik, diikuti dengan mutasi untuk menjaga keragaman genetik dan mencegah konvergensi dini (Soni & Kumar, 2007; Srinath Murthy & P Mankame, 2024). Setelah setiap generasi terbentuk, individu dalam populasi dievaluasi ulang hingga memenuhi kriteria penghentian tertentu seperti jumlah generasi maksimum atau konvergensi nilai *fitness* (Soni & Kumar, 2007; Srinath Murthy & P Mankame, 2024).

Komponen utama dalam algoritma genetika terdiri atas:

1. Representasi Kromosom yaitu cara menyusun sebuah solusi ke dalam bentuk struktural yang dapat diproses oleh algoritma genetika. Dalam konteks komputasi evolusioner, kromosom terdiri dari serangkaian gen yang masing-masing memuat nilai atau atribut tertentu yang membentuk keseluruhan solusi (Goldberg, 1989).
2. Fungsi *Fitness* digunakan untuk menilai kualitas setiap solusi, di mana individu dengan nilai *fitness* lebih tinggi memiliki peluang seleksi yang lebih besar (Goldberg, 1989; Melanie, 1998). Pada permasalahan minimasi seperti TSP, solusi dengan jarak lebih kecil dianggap lebih baik. Oleh karena itu, nilai cost perlu diubah agar berbanding lurus dengan kualitas solusi. Salah satu bentuk yang umum digunakan adalah:

$$f(i) = \frac{1}{Cost(i)} \tag{2.3}$$

di mana $Cost(i)$ merupakan total jarak rute ke-i. Melalui persamaan tersebut, semakin kecil nilai cost maka semakin besar nilai *fitness* yang dihasilkan (Onder, 2007; Zibaei & Mesgari, 2023).

3. Seleksi yaitu proses pemilihan individu yang akan menjadi induk berdasarkan nilai *fitness*. Individu dengan nilai *fitness* lebih tinggi memiliki peluang lebih besar untuk terpilih dan menghasilkan ketu*run*an pada generasi berikutnya (Goldberg, 1989; Melanie, 1998). Salah satu metode seleksi yang umum digunakan adalah *Roulette wheel Selection*, yaitu metode pemilihan induk berdasarkan probabilitas yang sebanding dengan nilai *fitness*. Semakin besar nilai *fitness* suatu individu, maka semakin besar pula peluangnya untuk terpilih. Metode ini digunakan karena mampu menjaga keseimbangan antara peluang individu terbaik dan keberagaman populasi (Katoch et al., 2021).

   Probabilitas Seleksi setiap individu dihitung menggunakan persamaan berikut:

$$P_i = \frac{f(i)}{\sum_{j=1}^{n} f(i)} \tag{2.4}$$

   P_i merupakan probabilitas seleksi individu ke-i, sedangkan $f(i)$ merupakan nilai *fitness* individu ke-i. Notasi $\sum_{j=1}^{n} f(j)$ menyatakan total *fitness* seluruh individu dalam populasi yang diperoleh dengan menjumlahkan nilai *fitness* dari individu ke-1 hingga ke-n, dimana n adalah jumlah inividu dalam populasi. Simbol j digunakan sebagai penanda urutan dalam proses penjumlahan tersebut, sehingga $f(j)$ dapat dipahami sebagai nilai *fitness* dari setiap individu yang dijumlahkan. Selanjutnya, probabilitas kumulatif dihitung untuk mempermudah proses pemilihan individu, yaitu sebagai berikut:

$$C_i = \sum_{j=1}^{i} Pj \tag{2.5}$$

C_i merupakan probabilitas kumulatif hingga individu ke-i. P_j menyatakan probabilitas seleksi dari setiap individu yang dijumlahkan secara bertahap dari individu pertama sampai individu ke-i. Nilai probabilitas kumulatif ini digunakan dalam mekanisme *roulette wheel* dengan membangkitkan bilangan acak r pada interval $(0,1]$, kemudian individu pertama yang memenuhi kondisi $C_i \geq r$ akan dipilih sebagai induk.

Selain *roulette wheel*, metode seleksi lain yang sering digunakan antara lain *Tournament Selection* dan *Rank Selection* (Katoch et al., 2021).

4. *Crossover* yaitu proses pertukaran gen antar induk untuk membentuk individu baru. Tahapan ini merupakan inti dari algoritma genetika karena menentukan variasi solusi yang dihasilkan (Soni & Kumar, 2007).
5. Mutasi yaitu proses perubahan susunan gen pada kromosom untuk menghasilkan variasi solusi baru. Pada TSP, mutasi dapat dilakukan dengan menukar posisi dua kota secara acak untuk menjaga keberagaman populasi (Srinath Murthy & P Mankame, 2024).

$$Jumlah\ Gen\ Dimutasi = Presentasi\ Mutasi \times Total\ Gen \qquad (2.6)$$

2.4 Kajian Integrasi Topik dengan Al-Quran dan Al-Hadist

Konsep optimasi dalam matematika berfokus pada upaya mencari solusi terbaik di antara berbagai alternatif yang memenuhi kendala tertentu agar hasil yang diperoleh efisien dan seimbang. Prinsip ini memiliki kecocokan dengan nilai-nilai yang diajarkan dalam Al-Qur'an, terutama dalam hal menjaga keseimbangan dan tidak berlebihan dalam setiap urusan. Salah satu ayat yang mencerminkan nilai keseimbangan tersebut terdapat dalam Q.S. Al-Furqan ayat 67 sebagai berikut:

وَالَّذِيْنَ اِذَآ اَنْفَقُوْا لَمْ يُسْرِفُوْا وَلَمْ يَقْتُرُوْا وَكَانَ بَيْنَ ذٰلِكَ قَوَامًا ﴿٦٧﴾

(Kementerian Agama Republik Indonesia, 2022)

"Dan, orang-orang yang apabila berinfak tidak berlebihan dan tidak (pula) kikir. (Infak mereka) adalah pertengahan antara keduanya." (Q.S. Al-Furqan: 67)

Menurut (Katsir & Umar, 2005) pada buku "*Tafsir Ibnu Katsir"*, ayat ini termasuk dalam rangkaian ayat yang menggambarkan sifat-sifat "ʿIbad ar-Raḥman" (hamba-hamba Allah SWT Yang Maha Pengasih). Ibnu Katsir menjelaskan bahwa ayat tersebut mengandung makna sikap adil dan proporsional dalam menggunakan sumber daya, baik dalam hal harta maupun dalam aspek kehidupan lainnya. Hamba Allah SWT yang sejati tidak berlebihan dalam membelanjakan hartanya, dan tidak pula terlalu kikir, tetapi menempuh jalan tengah yang penuh keseimbangan. Beliau juga mengutip pernyataan ulama seperti Al-Hasan Al-Bashri yang menafsirkan bahwa "sebaik-baik perkara adalah yang pertengahan," yaitu tidak condong kepada pemborosan maupun kekikiran. Ayat ini menjadi peringatan agar manusia menghindari perilaku ekstrem dalam segala bentuk, dan menjadikan keseimbangan sebagai prinsip dalam bertindak.

Nilai keseimbangan (*tawazun*) dan prinsip jalan tengah (*tawassut*) yang diajarkan dalam ayat ini memiliki relevansi dengan konsep optimasi dalam ilmu pengetahuan modern. Dalam konteks matematika dan komputasi, optimasi tidak hanya bertujuan memperoleh hasil yang paling ekstrem, tetapi mencari kondisi yang paling tepat dengan mempertimbangkan berbagai batasan, sehingga penggunaan sumber daya tidak berlebihan dan tidak pula kurang dari kebutuhan. Dengan demikian, prinsip ini mencerminkan upaya menghindari kondisi ekstrem untuk mencapai keadaan yang proporsional dan sesuai.

Berdasarkan prinsip ini, berbagai metode komputasi modern dikembangkan untuk menemukan solusi optimal secara sistematis. Salah satu contohnya adalah

algoritma genetika (GA), yang terinspirasi dari proses seleksi alam. Dalam GA, individu terbaik dari suatu populasi bertahan dan bereproduksi untuk menghasilkan generasi berikutnya yang lebih baik, sehingga proses ini berlangsung bertahap hingga ditemukan solusi optimal. Prinsip ini sejalan dengan nilai-nilai yang terkandung dalam Al-Qur'an, khususnya dalam Q.S. At-Tin ayat 4 yang berbunyi:

لَقَدْ خَلَقْنَا الْاِنْسَانَ فِيْٓ اَحْسَنِ تَقْوِيْمٍۖ ۝٤

(Kementerian Agama Republik Indonesia, 2022)

"Sesungguhnya kami telah menciptakan manusia dalam bentuk yang sebaik-baiknya." (Q.S. At-Tin: 4)

Menurut (Katsir & Umar, 2005) pada "*Tafsir Ibnu Katsir*", ayat ini dijelaskan sebagai penegasan bahwa Allah menciptakan manusia dalam bentuk yang paling sempurna, baik dari segi fisik, akal, maupun potensi rohaninya. Manusia diberikan struktur tubuh yang ideal, kemampuan berpikir, serta kebebasan memilih antara kebaikan dan keburukan. Namun, jika potensi itu disia-siakan, manusia dapat jatuh ke derajat yang paling rendah. Sebaliknya, mereka yang menjaga fitrah dan beramal saleh akan mencapai derajat kemuliaan yang kekal.

Makna ayat ini memiliki kesesuaian dengan prinsip dasar algoritma genetika, di mana setiap individu (solusi) memiliki potensi untuk berkembang menuju kondisi terbaik melalui proses seleksi dan penyempurnaan bertahap. Dalam konteks ilmiah, hal ini menggambarkan sunnatullah dalam sistem kehidupan bahwa setiap proses penciptaan dan perkembangan selalu mengarah pada kesempurnaan dan efisiensi. Dengan demikian, penelitian berbasis algoritma genetika dapat dipandang sebagai bentuk tadabbur terhadap keteraturan ciptaan Allah SWT, yang menjadi sumber inspirasi bagi manusia dalam merancang metode komputasi yang adaptif dan optimal.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

Penelitian ini menggunakan metode kuantitatif dengan pendekatan eksperimen komputasional. Metode kuantitatif dipilih karena seluruh proses penelitian menggunakan data berbentuk angka yang dapat diukur, seperti panjang total rute, nilai *fitness*, serta ringkasan hasil dari beberapa percobaan. Sementara itu, pendekatan eksperimen komputasional digunakan karena penelitian dilakukan melalui simulasi algoritma pada data *Traveling Salesman Problem* (TSP) untuk menilai kinerja metode yang diuji.

Secara khusus, penelitian ini bersifat komparatif, karena membandingkan performa beberapa skenario algoritma genetika pada permasalahan yang sama. Perbandingan ini dilakukan untuk mengetahui pengaruh mekanisme *multi-parent* pada *Nearest Neighbor Crossover* terhadap kinerja algoritma genetika. Dengan demikian, penelitian ini tidak hanya berfokus pada implementasi algoritma, tetapi juga pada evaluasi hasil secara terukur dan sistematis.

Agar proses penelitian lebih mudah dipahami, penelitian ini disajikan dalam dua tahap. Pertama, dilakukan demonstrasi mekanisme algoritma menggunakan data berukuran kecil untuk memperjelas proses seleksi, *crossover*, dan mutasi. Kedua, dilakukan pengujian utama menggunakan dataset berukuran besar, yaitu dataset pr1002, untuk memperoleh hasil performa metode secara empiris. Pemisahan ini dilakukan agar pembahasan penelitian lebih kuat, baik dari sisi proses maupun dari sisi hasil.

3.2 Data dan Sumber Data

Data yang digunakan dalam penelitian ini merupakan data benchmark *Traveling Salesman Problem* (TSP) yang berasal dari TSPLIB (*Traveling Salesman Problem Library*). TSPLIB merupakan kumpulan dataset standar yang digunakan untuk penelitian mengenai permasalahan optimasi TSP. Dataset pada TSPLIB tidak berupa data perjalanan nyata, melainkan representasi masalah TSP dalam bentuk sekumpulan titik yang merepresentasikan kota atau lokasi.

Dataset utama yang digunakan adalah pr1002. Nama dataset tersebut mengikuti penamaan yang digunakan pada TSPLIB, di mana bagian angka menunjukkan jumlah node atau kota yang terdapat pada permasalahan TSP. Dataset pr1002 terdiri dari 1002 titik yang masing-masing memiliki identitas node dan pasangan nilai koordinat dua dimensi.

Koordinat pada dataset TSPLIB digunakan untuk merepresentasikan posisi relatif setiap kota pada bidang dua dimensi. Data koordinat tersebut bukan merupakan koordinat geografis bumi seperti lintang dan bujur (latitude dan longitude), melainkan koordinat numerik yang dibuat sebagai representasi posisi antar kota dalam ruang dua dimensi untuk kebutuhan pengujian algoritma optimasi. Oleh karena itu, nilai koordinat pada dataset tidak memiliki satuan fisik tertentu seperti kilometer atau meter.

Berdasarkan koordinat tersebut, jarak antar node dihitung menggunakan persamaan jarak Euclid. Perhitungan ini menghasilkan nilai jarak numerik antar titik yang digunakan sebagai bobot perjalanan dalam proses optimasi. Karena koordinat awal tidak memiliki satuan fisik, maka nilai jarak yang dihasilkan juga berupa satuan relatif atau satuan jarak dataset.

Matriks jarak yang diperoleh dari proses perhitungan tersebut kemudian digunakan sebagai input utama dalam algoritma genetika. Setiap individu pada populasi merepresentasikan urutan kunjungan node, kemudian nilai fitness dihitung berdasarkan total jarak perjalanan yang diperoleh dari penjumlahan jarak antar node pada rute tersebut. Pemilihan dataset benchmark bertujuan agar pengujian memiliki data yang terstandar sehingga performa operator *Nearest Neighbor Crossover* (NNX) dan *Multi-parent Nearest Neighbor Crossover* (MPNNX) dapat dibandingkan secara objektif. Selain dataset utama, digunakan pula data berukuran kecil pada tahap ilustrasi proses algoritma. Data kecil tersebut hanya digunakan untuk menjelaskan mekanisme kerja algoritma pada Bab IV dan tidak digunakan dalam analisis perbandingan hasil eksperimen.

3.3 Instrumen Penelitian

Instrumen utama dalam penelitian ini adalah perangkat lunak Python yang dijalankan melalui Google Colab atau lingkungan komputasi lain yang setara. Python dipilih karena mampu mendukung pengolahan data numerik, implementasi algoritma optimasi, visualisasi hasil, serta penyimpanan keluaran dalam format yang mudah digunakan kembali dalam penyusunan laporan penelitian.

Untuk mendukung implementasi tersebut, digunakan beberapa pustaka pemrograman, yaitu:

1. *Pandas*, digunakan untuk membaca, mengelola, dan menyajikan data dalam bentuk tabel.
2. *Numpy*, digunakan untuk mendukung komputasi numerik dan manipulasi array.

3. *Math*, digunakan untuk membantu perhitungan matematis dasar, terutama perhitungan jarak.
4. *Random*, digunakan untuk membangkitkan solusi awal dan proses acak dalam algoritma genetika.
5. *Matplotlib*, digunakan untuk membuat grafik dan visualisasi hasil.
6. *Statistics* atau pustaka statistik lain yang relevan, digunakan untuk membuat ringkasan statistik hasil eksperimen.
7. Pustaka statistik tambahan, bila diperlukan, digunakan untuk melakukan uji perbandingan hasil antar metode.

Selain perangkat lunak, penelitian ini juga menggunakan perangkat keras berupa komputer atau laptop yang memadai untuk menjalankan simulasi komputasi pada dataset TSP berukuran besar. Perangkat keras ini diperlukan untuk memastikan proses komputasi dapat berjalan dengan stabil dan efisien, terutama pada tahap eksperimen utama yang melibatkan jumlah titik yang besar.

Instrumen penelitian ini tidak hanya digunakan untuk menghasilkan solusi optimasi, tetapi juga untuk menghasilkan keluaran yang terstruktur, seperti tabel, grafik, dan ringkasan hasil eksperimen. Keluaran tersebut selanjutnya digunakan sebagai bahan analisis dan pembahasan pada Bab IV.

3.4 Desain Penelitian

Desain penelitian ini dibuat agar seluruh langkah yang telah direncanakan pada Bab III dapat diterapkan secara jelas pada Bab IV. Oleh karena itu, penelitian ini dibagi menjadi dua tahap pelaksanaan, yaitu tahap demonstrasi mekanisme algoritma dan tahap pengujian utama.

3.4.1 Demonstrasi Mekanisme Algoritma

Tahap ini bertujuan untuk menjelaskan secara rinci bagaimana metode yang digunakan dalam penelitian bekerja. Pada tahap ini digunakan data berukuran kecil agar setiap proses algoritma dapat ditampilkan langkah demi langkah dan lebih mudah dipahami. Demonstrasi ini meliputi representasi solusi dalam bentuk kromosom, perhitungan total jarak rute, perhitungan nilai *fitness*, proses seleksi *parent*, pembentukan *offspring* menggunakan NNX, pembentukan *offspring* menggunakan MPNNX, serta proses mutasi dengan *swap mutation*.

Tahap demonstrasi ini tidak digunakan sebagai dasar untuk menarik kesimpulan utama, tetapi hanya sebagai sarana untuk memperjelas prosedur metode yang digunakan. Dengan demikian, Bab IV tidak hanya menampilkan hasil akhir, tetapi juga memperlihatkan proses terbentuknya hasil tersebut.

3.4.2 Pengujian Utama Pada Dataset Utama

Tahap ini merupakan bagian utama dari penelitian. Pada tahap ini, seluruh metode diterapkan pada dataset pr1002 untuk memperoleh hasil performa yang sebenarnya. Hasil pengujian kemudian disajikan dalam bentuk tabel hasil tiap percobaan, ringkasan statistik, grafik konvergensi, dan analisis perbandingan.

Melalui tahap ini, penelitian dapat menunjukkan apakah mekanisme *multi-parent* pada *Nearest Neighbor Crossover* memberikan pengaruh terhadap kualitas solusi yang diperoleh.

3.4.3 Skenario Pengujian

Agar pengaruh setiap metode dapat diamati dengan lebih jelas, penelitian ini menggunakan beberapa skenario pengujian pada dataset yang sama. Skenario yang digunakan adalah sebagai berikut:

1. GA + NNX, sebagai skenario pembanding dasar
2. GA + MPNNX, untuk melihat pengaruh penggunaan *multi-parent crossover*

Dengan desain ini, pengaruh penggunaan *multi-parent crossover* dapat diamati secara lebih jelas.

3.4.4 Variabel Yang Diamati

Variabel yang diamati dalam penelitian ini meliputi:

1. Panjang total rute terbaik yang diperoleh pada setiap percobaan
2. Rata-rata hasil optimasi dari sejumlah percobaan
3. Variasi hasil antarpercobaan berdasarkan simpangan baku
4. Pola konvergensi melalui grafik nilai terbaik pergenerasi.

Variabel-variabel tersebut diamati pada setiap skenario pengujian, kemudian dibandingkan untuk menilai performa masing-masing metode.

3.5 Tahapan Penelitian

Tahapan penelitian disusun secara sistematis agar dapat langsung diterapkan dalam implementasi komputasional serta mudah dijelaskan pada Bab IV. Adapun tahapan penelitian yang dilakukan adalah sebagai berikut:

1. Pemaparan dan Validasi Data

   Tahap pertama adalah membaca dataset pr1002 dari file sumber, kemudian memeriksa struktur datanya. Pemeriksaan dilakukan untuk memastikan bahwa data berisi identitas titik dan koordinat yang diperlukan, tidak memiliki nilai kosong, serta siap digunakan pada tahap berikutnya. Pada tahap ini juga dilakukan deskripsi awal data, seperti jumlah titik, nama kolom,

serta ringkasan sederhana koordinat. Tahap ini penting karena kualitas data input akan memengaruhi keseluruhan proses optimasi.

2. Pembentukan Matriks Jarak

   Setelah data dinyatakan valid, dilakukan pembentukan matriks jarak antartitik menggunakan jarak Euclidean. Perhitungan jarak antartitik mengacu pada persamaan (2.2). Matriks jarak yang dihasilkan menjadi representasi numerik dari permasalahan TSP dan digunakan sebagai input utama dalam proses pencarian rute. Pada tahap pembahasan pembentukan matriks jarak dapat ditunjukkan melalui beberapa contoh perhitungan sebelum digunakan pada dataset utama.

3. Representasi Solusi

   Pada penelitian ini, setiap solusi direpresentasikan sebagai urutan kunjungan titik dalam bentuk kromosom. Satu kromosom menunjukkan satu rute lengkap yang mengunjungi seluruh titik tepat sati kali dan Kembali ke titik awal. Dengan representasi ini, algoritma genetika dapat mengolah solusi secara konsisten dan mudah dievaluasi.

4. Penetapan Parameter Eksperimen

   Sebelum algoritma dijalankan, terlebih dahulu menentukan parameter eksperimen yang digunakan. Parameter tersebut meliputi ukuran populasi, jumlah generasi maksimum, probabilitas mutasi, jumlah elitisme, dan jumlah percobaan atau *run*. Semua skenario pengujian menggunakan parameter yang sama agar perbandingan hasil berlangsung secara adil. Nilai parameter ditentukan pada tahap implementasi dan dilaporkan secara jelas pada Bab IV.

5. Pembangkitan Populasi Awal

   Setelah parameter ditetapkan, dibangkitkan populasi awal secara acak. Setiap individu dalam populasi merupakan satu kromosom yang berisi urutan titik yang valid. Proses ini dilakukan hingga jumlah individu sesuai dengan ukuran populasi yang telah ditentukan. Pada tahap ini juga dilakukan pemeriksaan validitas untuk memastikan tidak ada titik yang berulang atau hilang dalam satu kromosom.

6. Perhitungan Fungsi *Fitness*

   Setiap individu dievaluasi berdasarkan total jarak rutenya. Semakin kecil total jarak yang diperoleh, semakin baik kualitas solusi tersebut. Berdasarkan total jarak ini, dihitung nilai *fitness* yang digunakan dalam proses seleksi. Dalam penelitian ini, fungsi *fitness* digunakan secara konsisten agar fokus penelitian tetap pada perbandingan operator *crossover*, bukan pada perbandingan formula *fitness*. Oleh karena itu, satu bentuk *fitness* digunakan pada seluruh skenario eksperimen.

7. Seleksi

   Tahap berikutnya adalah memilih individu yang akan menjadi *parent* dalam proses *crossover*. Seleksi dilakukan berdasarkan nilai *fitness*, sehingga individu dengan kualitas lebih baik memiliki peluang lebih besar untuk terpilih. Metode seleksi yang digunakan adalah *roulette wheel selection*. Melalui tahap ini, individu dengan kualitas lebih baik memiliki peluang lebih besar untuk menghasilkan ketu*run*an, namun keragaman populasi tetap terjaga.

8. Pembentukan *Offspring* dengan Operator *Crossover*

Tahap ini merupakan bagian utama penelitian. Pada tahap ini dilakukan pembentukan *offspring* menggunakan operator *crossover* sesuai skenario pengujian. Pada skenario pembanding digunakan *Nearest Neighbor Crossover* (NNX) yang melibatkan dua *parent*. Pada skenario usulan digunakan *Multi-parent Nearest Neighbor Crossover* (MPNNX) yang melibatkan tiga *parent*. Proses *crossover* dilakukan dengan mempertimbangkan kandidat kota berikutnya dari beberapa *parent* tersebut, kemudian dipilih kandidat terbaik sesuai aturan operator yang digunakan. Tahap ini bertujuan untuk melihat apakah penggunaan lebih dari dua *parent* dapat meningkatkan kualitas *offspring* yang dihasilkan.

9. Mutasi

Setelah proses *crossover*, dilakukan mutasi terhadap *offspring* menggunakan *swap mutation*. Mutasi dilakukan dengan menukar posisi dua titik dalam satu kromosom. Tujuan mutasi adalah menjaga keberagaman populasi dan mengurangi kemungkinan algoritma terjebak pada solusi lokal.

10. Pembentukan Generasi Baru

Offspring hasil *crossover* dan mutasi kemudian dievaluasi kembali. Berdasarkan hasil evaluasi tersebut, dibentuk generasi baru dengan menerapkan mekanisme elitisme yang mempertahankan sejumlah individu terbaik. Proses ini diulang hingga mencapai jumlah generasi maksimum atau memenuhi kriteria penghentian yang telah ditentukan.

11. Pengulangan Percobaan

    Karena algoritma genetika mengandung unsur acak, satu kali percobaan belum cukup untuk mewakili performa metode. Oleh karena itu, setiap skenario diuji melalui beberapa *run* dengan parameter yang sama. Hasil dari beberapa *run* tersebut kemudian diringkas dalam bentuk statistik deskriptif. Tahap ini penting agar hasil penelitian lebih stabil dan tidak bergantung pada satu percobaan saja.

12. Rekapitulasi dan Analisis Hasil

    Setelah seluruh percobaan selesai, hasil dari setiap skenario direkapitulasi. Rekapitulasi dilakukan dalam bentuk tabel hasil per *run*, nilai minimum, rata-rata, dan simpangan baku, serta grafik konvergensi atau grafik perbandingan apabila diperlukan. Selanjutnya, dilakukan uji statistik untuk mengetahui apakah perbedaan hasil antarmetode bersifat signifikan. Tahap ini menjadi dasar dalam penarikan kesimpulan penelitian.

3.6 Parameter Pengujian

Parameter pengujian dalam penelitian ini terdiri dari beberapa komponen utama, yaitu ukuran populasi, jumlah generasi, probabilitas mutasi, elitisme, dan jumlah *run*. Parameter-parameter tersebut ditentukan sebelum eksperimen dilakukan dan digunakan secara konsisten pada seluruh skenario pengujian.

Penggunaan parameter yang sama pada setiap skenario bertujuan agar perbandingan hasil dapat dilakukan secara adil. Dengan demikian, perbedaan hasil yang diperoleh lebih mencerminkan pengaruh operator *crossover* yang digunakan, bukan disebabkan oleh perbedaan pengaturan parameter.

Nilai rinci dari setiap parameter disajikan pada Bab IV sesuai dengan hasil implementasi akhir. Penyajian pada Bab IV dilakukan agar parameter yang dilaporkan benar-benar sesuai dengan parameter yang digunakan dalam program.

3.7 Teknik Analisis Data

Data hasil penelitian dianalisis menggunakan dua pendekatan, yaitu statistik deskriptif dan uji statistik. Kedua pendekatan ini digunakan untuk memberikan gambaran hasil sekaligus memastikan apakah perbedaan performa antarmetode memiliki makna yang signifikan.

3.7.1 Statistik Deskriptif

Statistik deskriptif digunakan untuk merangkum hasil percobaan pada setiap skenario pengujian. Ringkasan yang dihitung meliputi nilai minimum, nilai maksimum, nilai rata-rata, median jika diperlukan, serta simpangan baku. Langkah ini dilakukan untuk memberikan gambaran umum mengenai kualitas hasil yang diperoleh dari masing-masing metode sehingga memudahkan proses perbandingan.

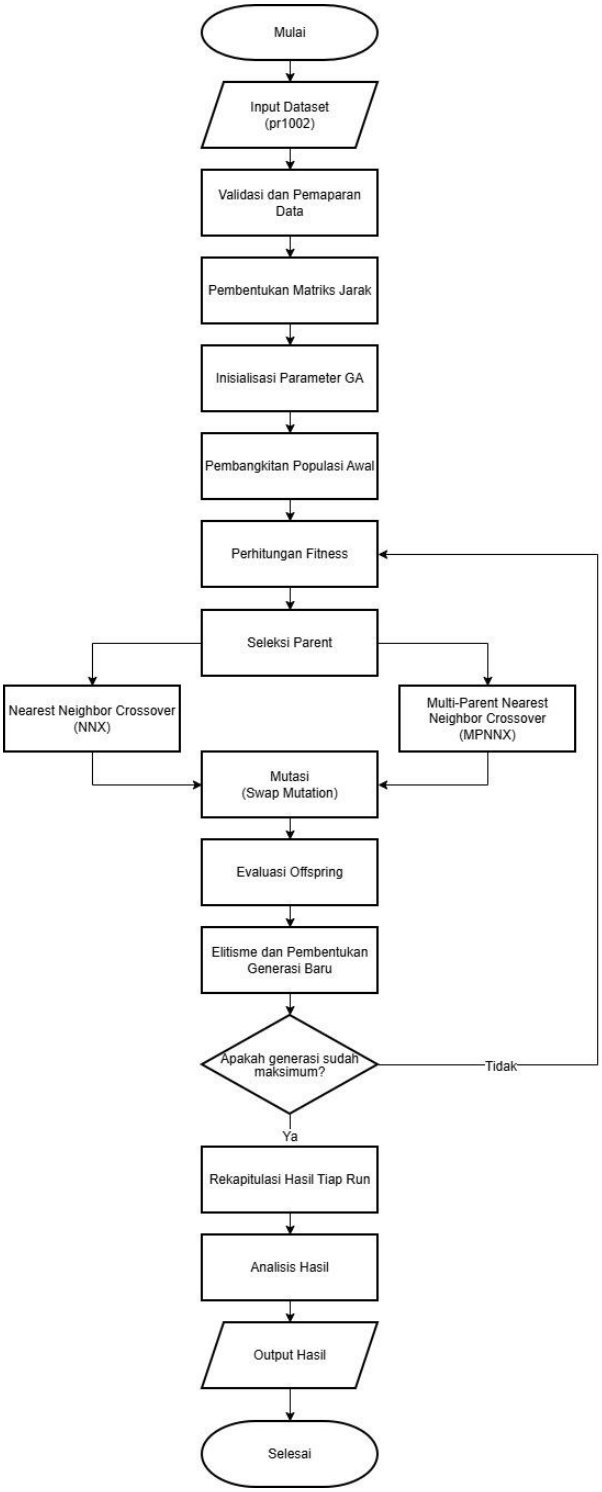
3.7.2 Uji Statistik

Untuk mengetahui apakah perbedaan performa antar skenario benar-benar signifikan, dilakukan uji statistik terhadap hasil percobaan. Data yang digunakan dalam pengujian adalah hasil utama dari setiap *run*, khususnya total jarak terbaik yang diperoleh oleh masing-masing metode.

Jenis uji statistik yang digunakan ditentukan berdasarkan karakteristik data hasil eksperimen. Jika data memenuhi asumsi tertentu, digunakan uji parametrik. Namun, jika asumsi tersebut tidak terpenuhi, digunakan uji nonparametrik yang sesuai. Dengan cara ini, kesimpulan mengenai keunggulan metode dapat

diperoleh secara lebih objektif dan tidak hanya berdasarkan perbedaan nilai rata-rata.

3.8 *Flowchart* Penelitian



Gambar 3.1 Flowchart Penelitian

BAB IV

HASIL DAN PEMBAHASAN

4.1 Deskripsi Dataset Penelitian

Pada penelitian ini digunakan dua jenis dataset, yaitu dataset contoh dan dataset utama. Dataset contoh digunakan untuk menjelaskan mekanisme algoritma secara manual, sedangkan dataset utama digunakan untuk melakukan pengujian kinerja metode yang diusulkan. Penggunaan dua dataset ini bertujuan agar tahapan algoritma dapat dijelaskan secara sistematis serta mempermudah proses analisis hasil penelitian.

Dataset contoh yang digunakan dalam penelitian ini diambil dari delapan titik pertama pada dataset pr1002. Dataset ini digunakan untuk menjelaskan mekanisme algoritma genetika secara manual, mulai dari pembentukan populasi awal, proses seleksi, *crossover*, dan mutasi. Setiap titik direpresentasikan dalam bentuk koordinat dua dimensi yang terdiri dari sumbu x dan y. Koordinat tersebut digunakan untuk menghitung jarak antar titik menggunakan rumus *Euclidean Distance*.

Tabel 4.1 Dataset Contoh 8 Titik dari pr1002

Titik	**Koordinat x**	**Koordinat y**
1	1150	4000
2	1050	2750
3	1150	2250
4	1250	2050
5	1350	2350
6	1050	1550
7	3350	1700
8	3450	1450

Dataset utama yang digunakan dalam penelitian ini adalah dataset pr1002 yang diperoleh dari TSPLIB (*Traveling Salesman Problem Library*), yaitu sebuah

repositori benchmark standar yang digunakan secara luas dalam penelitian optimasi kombinatorial, khususnya untuk permasalahan *Traveling Salesman Problem* (TSP). Penamaan "pr" pada dataset pr1002 merupakan singkatan dari nama peneliti yang memperkenalkannya, yaitu Padberg dan Rinaldi (1991), sedangkan angka "1002" menunjukkan jumlah titik kota yang terdapat dalam dataset tersebut. Dataset pr1002 merupakan salah satu instansi benchmark dalam TSPLIB yang terdiri dari 1002 titik lokasi kota dalam ruang koordinat dua dimensi, dan sering digunakan sebagai standar uji pengujian algoritma karena skalanya yang besar dan kompleksitasnya yang tinggi. Dataset ini dipilih karena jumlah titiknya yang besar memungkinkan pengujian performa algoritma secara lebih komprehensif. Karena jumlah titik pada dataset pr1002 cukup besar, maka pada penelitian ini hanya ditampilkan sebagian data sebagai representasi.

Tabel 4.2 Dataset pr1002

Titik	**Koordinat X**	**Koordinat Y**
1	1150	4000
2	1050	2750
3	1150	2250
4	1250	2050
5	1350	2350
6	1050	1550
7	3350	1700
8	3450	1450
9	3550	1600
10	3950	1700
…	…	…
1002	14550	11650

Tabel 4.2 hanya menampilkan sebagian dataset pr1002. Dataset lengkap terdiri dari 1002 titik yang digunakan dalam proses eksperimen penelitian. Dataset

pr1002 selanjutnya digunakan dalam proses eksperimen utama untuk membandingkan performa metode NNX dan MPNNX yang diusulkan dalam penelitian ini.

4.2 Demonstrasi Mekanisme Algoritma

Pada bagian ini dilakukan demonstrasi mekanisme algoritma menggunakan dataset contoh yang terdiri dari delapan titik pertama dari dataset pr1002. Demonstrasi ini bertujuan untuk memperjelas tahapan algoritma genetika dalam menyelesaikan permasalahan *Traveling Salesman Problem* secara sistematis dan mudah dipahami.

Dataset contoh yang digunakan mengacu pada Tabel 4.1 yang telah dijelaskan pada bagian sebelumnya. Delapan titik tersebut dipilih sebagai representasi sederhana agar setiap proses dalam algoritma dapat ditampilkan secara jelas.

Tahapan demonstrasi meliputi pembentukan matriks jarak, pembangkitan populasi awal, perhitungan *fitness*, seleksi *parent*, proses *crossover* menggunakan metode NNX dan MPNNX, serta mutasi. Setiap tahapan akan dijelaskan secara rinci pada subbagian berikut.

4.2.1 Pembentukan Matriks Jarak

Tahap pertama dalam demonstrasi mekanisme algoritma adalah pembentukan matriks jarak antar titik. Matriks jarak digunakan untuk mengetahui jarak antara setiap pasangan titik pada dataset contoh. Perhitungan jarak antar titik dilakukan berdasarkan koordinat masing-masing titik yang terdiri dari sumbu x dan y.

Perhitungan jarak pada penelitian ini menggunakan rumus *Euclidean Distance* yang telah dijelaskan pada Bab 3. Proses perhitungan dilakukan untuk

seluruh pasangan titik pada dataset contoh yang terdiri dari delapan titik pertama dari dataset pr1002.

Sebagai contoh, perhitungan jarak antara titik 1 dan titik 2 dilakukan berdasarkan koordinat masing-masing titik. Berdasarkan Tabel 4.1, titik 1 memiliki koordinat (1150, 4000) dan titik 2 memiliki koordinat (1050, 2750). Perhitungan jarak antara kedua titik tersebut dilakukan sebagai berikut:

$$d(1,2) = \sqrt{(1150 - 1050)^2 + (4000 - 2750)^2}$$

$$= \sqrt{(100)^2 + (1250)^2}$$

$$= \sqrt{10000 + 1562500}$$

$$= \sqrt{1572500}$$

$$= 1253{,}994$$

Perhitungan tersebut dilakukan untuk seluruh pasangan titik hingga diperoleh matriks jarak lengkap. Nilai pada matriks jarak ditampilkan hingga tiga angka di belakang koma untuk menjaga presisi perhitungan. Hasil perhitungan jarak antar titik kemudian disusun dalam bentuk matriks jarak yang digunakan dalam proses selanjutnya pada algoritma genetika.

Tabel 4.3 Matriks Jarak 8 Titik

Titik	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	0	1253,994	1750,000	1952,562	1662,077	2452,040	3182,766	3434,021
2	1253,994	0	509,902	728,011	500,000	1200,000	2528,339	2729,469
3	1750,000	509,902	0	223,607	223,607	707,107	2267,708	2435,159
4	1952,562	728,011	223,607	0	316,228	538,516	2128,967	2280,351
5	1662,077	500,000	223,607	316,228	0	854,400	2102,974	2284,732
6	2452,040	1200,000	707,107	538,516	854,400	0	2304,886	2402,082
7	3182,766	2528,339	2267,708	2128,967	2102,974	2304,886	0	269,258
8	3434,021	2729,469	2435,159	2280,351	2284,732	2402,082	269,258	0

Matriks jarak yang diperoleh selanjutnya digunakan dalam proses inisialisasi populasi dan perhitungan *fitness* pada algoritma genetika.

4.2.2 Representasi Kromosom

Pada algoritma genetika, solusi direpresentasikan dalam bentuk kromosom. Pada penelitian ini, kromosom direpresentasikan sebagai urutan titik yang menunjukkan rute perjalanan dalam permasalahan *Traveling Salesman Problem* (TSP). Setiap gen pada kromosom merepresentasikan satu titik yang harus dikunjungi.

Representasi kromosom yang digunakan adalah representasi permutasi, dimana setiap titik hanya muncul satu kali dalam kromosom. Hal ini bertujuan agar setiap titik dikunjungi tepat satu kali sebelum kembali ke titik awal. Berdasarkan delapan titik yang digunakan dalam demonstrasi, contoh representasi kromosom dapat ditunjukkan sebagai berikut:

Tabel 4.4 Contoh Representasi Kromosom

Posisi Gen	1	2	3	4	5	6	7	8
Kromosom	1	2	3	4	5	6	7	8

Kromosom tersebut merepresentasikan urutan kunjungan titik, sehingga rute perjalanan yang terbentuk adalah 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 1. Berdasarkan rute tersebut, diperoleh total jarak sebesar 9166,296. Selanjutnya nilai *fitness* dihitung menggunakan persamaan (2.3), sehingga diperoleh nilai *fitness* sebesar 0,00010910. Nilai *fitness* yang lebih besar menunjukkan solusi yang lebih baik. Representasi kromosom ini selanjutnya digunakan dalam proses pembentukan populasi awal pada algoritma genetika.

4.2.3 Inisialisasi Populasi

Setelah representasi kromosom ditentukan, tahap selanjutnya adalah pembentukan populasi awal. Populasi awal merupakan sekumpulan kromosom yang dibangkitkan secara acak dan digunakan sebagai solusi awal dalam algoritma

genetika. Setiap kromosom dalam populasi merepresentasikan satu kemungkinan rute perjalanan pada permasalahan *Traveling Salesman Problem* (TSP).

Pada penelitian ini, populasi awal dibangkitkan menggunakan metode pengacakan (*random permutation*), sehingga setiap kromosom memiliki urutan titik yang berbeda. Proses ini bertujuan untuk menghasilkan variasi solusi awal yang beragam sehingga algoritma genetika dapat melakukan eksplorasi ruang solusi secara lebih efektif.

Berdasarkan delapan titik yang digunakan dalam demonstrasi, populasi awal dibangkitkan sebanyak beberapa individu. Contoh populasi awal yang dihasilkan dapat ditunjukkan pada Tabel 4.5.

Tabel 4.5 Populasi Awal Demonstrasi

Individu	**Kromosom**
1	7 – 8 – 3 – 5 – 1 – 4 – 2 – 6
2	4 – 3 – 8 – 2 – 7 – 1 – 5 – 6
3	8 – 5 – 3 – 6 – 2 – 1 – 4 – 7
4	4 – 6 – 3 – 2 – 8 – 5 – 7 – 1
5	5 – 3 – 4 – 6 – 8 – 2 – 1 – 7
6	2 – 6 – 4 – 8 – 7 – 1 – 5 – 3

Setiap individu dalam populasi awal merupakan kromosom yang berisi urutan titik yang berbeda. Variasi ini diperlukan agar proses seleksi dan *crossover* dapat menghasilkan solusi yang lebih baik pada generasi berikutnya. Populasi awal yang telah dibentuk kemudian digunakan pada tahap evaluasi *fitness* untuk menghitung kualitas setiap individu berdasarkan total jarak rute yang dihasilkan.

4.2.4 Evaluasi *Fitness*

Setelah populasi awal terbentuk, tahap selanjutnya adalah evaluasi *fitness*. Evaluasi *fitness* bertujuan untuk menilai kualitas setiap individu dalam populasi

berdasarkan total jarak rute yang dihasilkan. Pada permasalahan *Traveling Salesman Problem* (TSP), solusi yang lebih baik ditunjukkan oleh total jarak yang lebih kecil.

Total jarak dihitung berdasarkan urutan titik pada kromosom dengan menjumlahkan jarak antar titik secara berurutan hingga kembali ke titik awal. Perhitungan jarak antar titik menggunakan matriks jarak yang telah diperoleh pada Tabel 4.3.

Sebagai contoh, salah satu kromosom dari populasi awal adalah sebagai berikut:

7 – 8 – 3 – 5 – 1 – 4 – 2 – 6 – 7

Berdasarkan matriks jarak, maka total jarak dihitung sebagai berikut:

7 → 8 = 269,258

8 → 3 = 2435,159

3 → 5 = 223,607

5 → 1 = 1662,077

1 → 4 = 1952,562

4 → 2 = 728,011

2 → 6 = 1200,000

6 → 7 = 2304,886

Sehingga total jarak dihitung sebagai berikut:

$$Total\ Jarak = 269{,}258 + 2435{,}159 + 223{,}607 + 1662{,}077 + 1952{,}562 + 728{,}011 + 1200{,}000 + 2304{,}886$$

$$Total\ Jarak = 10775{,}560706571292$$

Selanjutnya nilai *fitness* dihitung menggunakan Persamaan (2.3) pada Bab II sebagai berikut:

$$f(i) = \frac{1}{Cost(i)}$$

$$f(i) = \frac{1}{10775{,}560706571292}$$

$$f(i) = 0{,}00009280$$

Nilai *fitness* yang lebih besar menunjukkan solusi yang lebih baik karena memiliki total jarak yang lebih kecil. Hasil evaluasi *fitness* untuk seluruh populasi awal ditunjukkan pada Tabel 4.6.

Tabel 4.6 Hasil Evaluasi Fitness Populasi Awal

Individu	**Kromosom**	**Total Jarak**	**Nilai *Fitness***
1	7 – 8 – 3 – 5 – 1 – 4 – 2 – 6 – 7	10775,561	0,00009280
2	4 – 3 – 8 – 2 – 7 – 1 – 5 – 6 – 4	14154,334	0,00007065
3	8 – 5 – 3 – 6 – 2 – 1 – 4 – 7 – 8	10020,227	0,00009980
4	4 – 6 – 3 – 2 – 8 – 5 – 7 – 1 – 4	14008,029	0,00007139
5	5 – 3 – 4 – 6 – 8 – 2 – 1 – 7 – 5	12657,015	0,00007901
6	2 – 6 – 4 – 8 – 7 – 1 – 5 – 3 – 2	9866,477	0,00010135

Tabel tersebut menunjukkan nilai total jarak dan *fitness* dari masing-masing individu. Nilai *fitness* ini kemudian digunakan pada tahap seleksi untuk menentukan individu yang akan dipilih sebagai *parent* pada proses *crossover* berikutnya.

4.2.5 Seleksi

Setelah nilai *fitness* setiap individu diperoleh, tahap selanjutnya adalah proses seleksi. Seleksi bertujuan untuk memilih individu yang akan dijadikan sebagai *parent* pada proses *crossover*. Pada penelitian ini, metode seleksi yang digunakan adalah *Roulette wheel Selection*.

Metode *Roulette wheel Selection* memilih individu berdasarkan peluang yang dihitung dari nilai *fitness*. Individu dengan nilai *fitness* yang lebih besar memiliki peluang lebih besar untuk terpilih sebagai *parent*. Nilai probabilitas setiap individu dihitung menggunakan Persamaan (2.4).

Sebagai contoh, nilai *fitness* dari masing-masing individu adalah sebagai berikut:

0,00009280; 0,00007065; 0,00009980; 0,00007139; 0,00007901; 0,00010135

Total *fitness* dihitung sebagai berikut:

$$Total\ Fitness = 0{,}00009280 + 0{,}00007065 + 0{,}00009980 + 0{,}00007139 + 0{,}00007901 + 0{,}00010135$$

$$Total\ Fitness = 0{,}00051500$$

Selanjutnya, probabilitas untuk individu pertama dihitung sebagai berikut:

$$Probabilitas_1 = \frac{0{,}00009280}{0{,}00051500} = 0{,}180199$$

Kemudian dihitung probabilitas kumulatif. Untuk individu pertama:

$$Probabilitas\ Kumulatif_1 = 0{,}180199$$

Untuk individu kedua:

$$Probabilitas_2 = \frac{0{,}00007065}{0{,}00051500} = 0{,}137184$$

$$Probabilitas\ Kumulatif_2 = 0{,}180199 + 0{,}137184 = 0{,}317384$$

perhitungan yang sama dilakukan hingga individu terakhir.

Hasil perhitungan probabilitas dan probabilitas kumulatif selengkapnya ditunjukkan pada Tabel 4.7.

Tabel 4.7 Hasil Perhitungan Probabilitas dan Probabilitas Kumulatif

Individu	**Total Jarak**	***Fitness***	**Probabilitas**	**Probabilitas Kumulatif**
1	10775,561	0,00009280	0,180199	0,180199
2	14154,334	0,00007065	0,137184	0,317384
3	10020,227	0,00009980	0,193783	0,511167
4	14008,029	0,00007139	0,138617	0,649784
5	12657,015	0,00007901	0,153413	0,803197
6	9866,477	0,00010135	0,196803	1,000000

Setelah nilai probabilitas kumulatif diperoleh, tahap selanjutnya adalah menentukan rentang probabilitas kumulatif untuk setiap individu. Rentang ini digunakan sebagai dasar dalam mekanisme *roulette wheel selection* untuk memilih *parent* yang akan digunakan pada proses *crossover*. Penentuan rentang dilakukan berdasarkan nilai probabilitas kumulatif secara berurutan pada interval 0 sampai 1.

Individu pertama memperoleh rentang dari 0 hingga nilai probabilitas kumulatif pertama. Selanjutnya, rentang individu berikutnya ditentukan dengan melanjutkan batas akhir rentang sebelumnya sampai nilai probabilitas kumulatif individu tersebut. Dengan demikian, rentang probabilitas setiap individu terbentuk secara berurutan sesuai nilai probabilitas kumulatifnya.

Sebagai contoh, apabila individu pertama memiliki probabilitas kumulatif sebesar 0,180199 maka rentang probabilitasnya adalah 0,000000 – 0,180199. Jika individu kedua memiliki probabilitas kumulatif sebesar 0,317384, maka rentang probabilitasnya ditentukan dari batas akhir rentang sebelumnya, yaitu 0,180199 – 0,317384. Proses yang sama dilakukan hingga seluruh individu memperoleh rentang probabilitas kumulatif masing-masing sebagaimana disajikan pada Tabel 4.8 sebagai berikut:

Tabel 4.8 Rentang Probabilitas Kumulatif

Individu	**Nilai Rentang**
1	(0,000000 – 0,180199]
2	(0,180199 – 0,317384]
3	(0,317384 – 0,511167]
4	(0,511167 – 0,649784]
5	(0,649784 – 0,803197]
6	(0,803197 – 1,000000]

Selanjutnya dilakukan pembangkitan bilangan acak pada interval 0 sampai 1 untuk menentukan *parent* yang terpilih pada proses seleksi. Bilangan acak tersebut diperoleh menggunakan fungsi pembangkit bilangan acak pada program, di mana setiap pembangkitan menghasilkan satu nilai riil pada interval [0,1]. Karena pada proses *crossover* diperlukan tiga *parent*, maka dilakukan pembangkitan sebanyak tiga kali sehingga diperoleh nilai bilangan acak untuk *parent* pertama, kedua, dan ketiga masing-masing sebesar 0,301829, 0,034463, dan 0,340153.

Nilai bilangan acak tersebut kemudian dicocokkan dengan rentang probabilitas kumulatif pada Tabel 4.8. Nilai 0,301829 berada pada rentang 0,180199 – 0,317384 sehingga individu ke-2 terpilih sebagai *parent* pertama. Nilai 0,034463 berada pada rentang 0,000000 – 0,180199 sehingga individu ke-1 terpilih sebagai *parent* kedua. Selanjutnya, nilai 0,340153 berada pada rentang 0,317384 – 0,511167 sehingga individu ke-3 terpilih sebagai *parent* ketiga.

Dengan demikian, individu yang terpilih sebagai *parent* adalah individu 2, individu 1, dan individu 3. Individu yang terpilih tersebut kemudian digunakan sebagai *parent* dalam proses *crossover* menggunakan metode NNX dan MPNNX untuk menghasilkan *offspring* pada generasi berikutnya.

Tabel 4.9 *Parent* Hasil Seleksi

Parent	**Individu**	**Kromosom**
1	2	4 – 3 – 8 – 2 – 7 – 1 – 5 – 6
2	1	7 – 8 – 3 – 5 – 1 – 4 – 2 – 6
3	3	8 – 5 – 3 – 6 – 2 – 1 – 4 – 7

4.2.6 *Crossover* dengan Metode *Nearest Neighbor Crossover* (NNX)

Setelah proses seleksi dilakukan, tahap selanjutnya adalah proses *crossover*. *Crossover* bertujuan untuk menghasilkan individu baru (*offspring*) dengan mengkombinasikan informasi dari *parent* yang terpilih. Pada penelitian ini, metode *crossover* yang digunakan adalah *Nearest Neighbor Crossover* (NNX).

Metode NNX bekerja dengan cara memilih kota berikutnya berdasarkan jarak terdekat dari kandidat yang diperoleh dari *parent*. Proses dimulai dari satu kota awal, kemudian pada setiap langkah akan dipilih kota terdekat yang belum dikunjungi. *Parent* yang digunakan dalam proses *crossover* ini diperoleh dari hasil seleksi pada subbab sebelumnya. Meskipun pada tahap seleksi diperoleh tiga *parent*, pada metode *Nearest Neighbor Crossover* (NNX) hanya digunakan dua *parent*, yaitu *parent* 1 dan *parent* 2, sesuai dengan mekanisme dasar metode NNX yang merupakan *two-parent crossover*.

Proses pembentukan *offspring* dimulai dari kota awal, yaitu kota 4 yang merupakan kota pertama pada *parent* 1. Pada setiap langkah, ditentukan kandidat kota berikutnya dari masing-masing *parent.* Selanjutnya dihitung jarak dari kota saat ini ke kandidat tersebut, dan dipilih kota dengan jarak terdekat yang belum dikunjungi. Sebagai contoh pada langkah pertama, kota saat ini adalah kota 4. Berdasarkan *parent* yang digunakan, kandidat kota berikutnya dari *parent* 1 adalah kota 3, sedangkan dari *parent* 2 adalah kota 2. Selanjutnya dihitung jarak dari kota 4 menuju masing-masing kandidat, yaitu jarak ke kota 3 sebesar 223,607

dan jarak ke kota 2 sebesar 728,011. Karena jarak menuju kota 3 lebih kecil, maka kota 3 dipilih sebagai kota berikutnya dalam pembentukan *offspring*.

Proses tersebut diulang hingga seluruh kota telah dikunjungi. Apabila kandidat yang diperoleh sudah pernah dikunjungi, maka dipilih kota terdekat dari himpunan kota yang belum dikunjungi.

Tabel 4.10 Proses *Crossover* Metode NNX

Langkah	**Kota Saat Ini**	**Kandidat P1**	**Kandidat P2**	**Jarak ke Kandidat (P1, P2)**	**Kota Terpilih**	***Offspring* Sementara**
1	4	3	2	(3 = 223,607), (2 = 728,011)	3	4 – 3
2	3	8	5	(5 = 223,607), (8 = 2435,159)	5	4 – 3 – 5
3	5	6	1	(6 = 854,400), (1 = 1662,077)	6	4 – 3 – 5 – 6
4	6	4	7	(7 = 2304,886)	7	4 – 3 – 5 – 6 – 7
5	7	1	8	(8 = 269,258), (1 = 3182,766)	8	4 – 3 – 5 – 6 – 7 – 8
6	8	2	3	(2 = 2729,469)	2	4 – 3 – 5 – 6 – 7 – 8 – 2
7	2	7	6	(1 = 1253,994)	1	4 – 3 – 5 – 6 – 7 – 8 – 2 – 1

Berdasarkan Tabel 4.10, proses *crossover* menggunakan metode *Nearest Neighbor Crossover* (NNX) menghasilkan *offspring* akhir yaitu 4 – 3 – 5 – 6 – 7 – 8 – 2 – 1 dengan total jarak sebesar 9811,783.

4.2.7 *Crossover* dengan Metode *Multi-parent Nearest Neigbor Crossover* (MPNNX)

Setelah proses *crossover* menggunakan metode *Nearest Neighbor Crossover* (NNX) dilakukan, tahap selanjutnya adalah *crossover* menggunakan

metode *Multi-Parent Nearest Neighbor Crossover* (MPNNX). Metode ini merupakan pengembangan dari NNX dengan menggunakan lebih dari dua *parent* dalam proses pembentukan *offspring*.

Berdasarkan hasil seleksi menggunakan metode *Roulette wheel Selection* pada subbab sebelumnya, diperoleh sejumlah *parent* yang digunakan dalam proses *crossover* metode MPNNX. Berbeda dengan metode NNX yang menggunakan dua *parent*, metode MPNNX menggunakan tiga *parent*, yaitu *parent* 1, *parent* 2, dan *parent* 3. Proses pembentukan *offspring* dimulai dari kota awal, yaitu kota pertama pada *parent* 1, yaitu kota 4. Pada setiap langkah, kandidat kota berikutnya ditentukan dari masing-masing *parent* berdasarkan posisi kota saat ini. Selanjutnya, dihitung jarak dari kota saat ini ke seluruh kandidat tersebut, kemudian dipilih kota dengan jarak terdekat yang belum dikunjungi.

Sebagai contoh pada langkah pertama, kota saat ini adalah kota 4. Berdasarkan ketiga *parent* yang digunakan, kandidat kota berikutnya adalah kota 3 dari *parent* 1, kota 2 dari *parent* 2, dan kota 7 dari *parent* 3. Jarak dari kota 4 menuju masing-masing kandidat tersebut adalah 223,607 untuk kota 3, 728,011 untuk kota 2, dan 2128,967 untuk kota 7. Karena jarak menuju kota 3 merupakan yang paling kecil, maka kota 3 dipilih sebagai kota berikutnya dalam pembentukan *offspring*.

Proses ini diulang hingga seluruh kota telah dikunjungi. Apabila kandidat yang diperoleh sudah pernah dikunjungi, maka pemilihan kota dilakukan dari himpunan kota yang belum dikunjungi berdasarkan jarak terdekat.

Tabel 4.11 Proses *Crossover* Metode MPNNX

Langkah	Kota Saat Ini	Kandidat P1	Kandidat P2	Kandidat P3	Jarak ke Kandidat (P1, P2, P3)	Kota Terpilih	*Offspring* Sementara
1	4	3	2	7	(3 = 223,607), (2 = 728,011), (7 = 2128,967)	3	4 – 3
2	3	8	5	6	(5 = 223,607), (6 = 707,107), (8 = 2435,159)	5	4 – 3 – 5
3	5	6	1	3	(6 = 854,400), (1 = 1662,077)	6	4 – 3 – 5 – 6
4	6	4	7	2	(2 = 1200,000), (7 = 2304,886)	2	4 – 3 – 5 – 6 – 2
5	2	7	6	1	(1 = 1253,994), (7 = 2528,339)	1	4 – 3 – 5 – 6 – 2 – 1
6	1	5	4	4	(7 = 3182,766), (8 = 3434,021)	7	4 – 3 – 5 – 6 – 2 – 1 – 7
7	7	1	8	8	(8 = 269,258)	8	4 – 3 – 5 – 6 – 2 – 1 – 7 – 8

Berdasarkan Tabel 4.11, proses *crossover* menggunakan metode *Multi-parent Nearest Neighbor Crossover* (MPNNX) menghasilkan *offspring* akhir yaitu 4 – 3 – 5 – 6 – 2 – 1 – 7 – 8 dengan total jarak sebesar 9487,983.

4.2.8 Mutasi

Setelah proses *crossover* menggunakan metode *Nearest Neighbor Crossover* (NNX) dan *Multi-parent Nearest Neighbor Crossover* (MPNNX) dilakukan, tahap selanjutnya adalah proses mutasi. Mutasi bertujuan untuk menambah keberagaman solusi dalam populasi serta menghindari algoritma terjebak pada solusi lokal (*local optimum*).

Pada penelitian ini, metode mutasi yang digunakan adalah *swap mutation*, yaitu dengan menukar posisi dua kota secara acak dalam satu rute. Proses mutasi

dilakukan berdasarkan probabilitas mutasi yang telah ditentukan. Probabilitas mutasi yang digunakan dalam penelitian ini adalah sebesar 0.1. Selanjutnya, dibangkitkan bilangan acak antara 0 sampai 1 untuk menentukan apakah mutasi dilakukan atau tidak. Jika nilai bilangan acak lebih kecil dari probabilitas mutasi, maka mutasi dilakukan.

Berdasarkan hasil perhitungan, diperoleh angka acak mutasi sebesar 0,042533. Karena nilai tersebut lebih kecil dari probabilitas mutasi (0,042533 < 0,1), maka proses mutasi dilakukan. Ringkasan proses mutasi disajikan pada Tabel 4.12.

Tabel 4.12 Proses Mutasi

Keterangan	**Nilai**
Angka Acak	0,042533
Probabilitas Mutasi	0,1
Rute Sebelum	4 – 3 – 5 – 6 – 2 – 1 – 7 – 8
Posisi ditukar	2 dan 5
Rute Sesudah	4 – 2 – 5 – 6 – 3 – 1 – 7 – 8
Total Jarak Sebelum Mutasi	9487,983
Total Jarak Sesudah Mutasi	10271,893

Berdasarkan Tabel 4.12, terlihat bahwa proses mutasi menghasilkan perubahan rute serta peningkatan total jarak. Hal ini menunjukkan bahwa mutasi tidak selalu menghasilkan solusi yang lebih baik. Namun demikian, mutasi tetap memiliki peran penting dalam menjaga keberagaman populasi dan membantu algoritma dalam mengeksplorasi ruang solusi secara lebih luas.

Setelah proses mutasi dilakukan, diperoleh individu baru yang selanjutnya akan digunakan pada tahap berikutnya dalam algoritma genetika. Pada tahap demonstrasi ini, proses elitisme tidak ditampilkan secara rinci, namun

penerapannya digunakan pada tahap pengujian sesuai dengan parameter yang telah ditentukan.

4.2.9 Hasil Demonstrasi Algoritma

Setelah melalui tahapan seleksi, *crossover* menggunakan metode *Nearest Neighbor Crossover* (NNX) dan *Multi-parent Nearest Neighbor Crossover* (MPNNX), serta proses mutasi, selanjutnya dilakukan analisis terhadap hasil demonstrasi yang telah diperoleh. Hasil demonstrasi ini bertujuan untuk membandingkan kinerja metode *crossover* yang digunakan dalam menghasilkan solusi rute. Perbandingan kinerja metode dilakukan berdasarkan hasil proses *crossover* sebelum mutasi. Hal ini dikarenakan proses mutasi bersifat acak dan tidak selalu menghasilkan solusi yang lebih baik, sehingga kurang tepat digunakan sebagai dasar utama dalam membandingkan performa metode *crossover*.

Berikut merupakan hasil perbandingan metode NNX dan MPNNX berdasarkan total jarak yang dihasilkan.

Tabel 4.13 Perbandingan Hasil Demonstrasi Metode *Crossover*

Metode	***Offspring***	**Total Jarak**
NNX	4 – 3 – 5 – 6 – 7 – 8 – 2 – 1	9811,783
MPNNX	4 – 3 – 5 – 6 – 2 – 1 – 7 – 8	9487,983

Berdasarkan Tabel 4.13, dapat dilihat bahwa metode *Multi-Parent Nearest Neighbor Crossover* (MPNNX) menghasilkan total jarak yang lebih kecil dibandingkan metode *Nearest Neighbor Crossover* (NNX). Hal ini menunjukkan bahwa metode MPNNX mampu menghasilkan solusi yang lebih optimal pada kasus demonstrasi yang dilakukan. Perbedaan hasil ini disebabkan oleh penggunaan lebih dari dua *parent* pada metode MPNNX, sehingga jumlah kandidat kota yang dipertimbangkan pada setiap langkah menjadi lebih banyak. Dengan semakin beragamnya kandidat yang tersedia, peluang untuk memilih kota

dengan jarak terdekat menjadi lebih besar, yang pada akhirnya menghasilkan rute dengan total jarak yang lebih pendek.

Sementara itu, metode NNX hanya menggunakan dua *parent* sehingga pilihan kandidat kota pada setiap langkah lebih terbatas. Hal ini dapat menyebabkan rute yang dihasilkan kurang optimal dibandingkan dengan metode MPNNX. Namun, metode MPNNX tidak selalu menghasilkan solusi yang lebih baik dibandingkan metode NNX. Performa kedua metode sangat bergantung pada kombinasi *parent* yang digunakan serta kondisi distribusi jarak antar kota. Pada kondisi tertentu, tambahan *parent* dalam metode MPNNX tidak selalu memberikan keuntungan yang signifikan apabila kandidat kota yang dihasilkan memiliki jarak yang relatif serupa atau tidak lebih baik dibandingkan kandidat dari dua *parent*.

Dengan demikian, keunggulan metode MPNNX perlu dibuktikan lebih lanjut melalui pengujian pada dataset yang lebih besar dan beragam pada tahap eksperimen.

4.3 Parameter Pengujian

Parameter pengujian merupakan komponen penting dalam penelitian ini karena berpengaruh terhadap kinerja algoritma genetika dalam menghasilkan solusi. Parameter yang digunakan ditetapkan sebelum proses pengujian dilakukan dan diterapkan secara konsisten pada seluruh percobaan, baik untuk metode *Nearest Neighbor Crossover* (NNX) maupun *Multi-Parent Nearest Neighbor Crossover* (MPNNX). Hal ini bertujuan untuk memastikan bahwa perbandingan hasil kedua metode dilakukan secara adil.

Parameter yang digunakan dalam penelitian ini meliputi ukuran populasi, jumlah generasi, probabilitas mutasi, serta jumlah *run*. Selain itu, pada tahap pengujian utama juga diterapkan mekanisme elitisme untuk mempertahankan individu terbaik pada setiap generasi. Ukuran populasi menentukan jumlah individu dalam setiap generasi yang akan diproses oleh algoritma. Semakin besar ukuran populasi, maka semakin banyak variasi solusi yang dapat dieksplorasi. Jumlah generasi menunjukkan banyaknya iterasi yang dilakukan dalam proses evolusi untuk mencapai solusi optimal.

Pada penelitian ini digunakan dua metode *crossover*, yaitu NNX dan MPNNX, dengan perbedaan utama terletak pada jumlah *parent* yang digunakan. Sementara itu, probabilitas mutasi digunakan untuk menentukan peluang terjadinya perubahan pada individu, yang bertujuan untuk menjaga keberagaman populasi dan menghindari terjebaknya algoritma pada solusi lokal. Pengujian dilakukan sebanyak 35 kali (*run*) untuk masing-masing metode guna memperoleh hasil yang lebih representatif dan mengurangi pengaruh faktor acak dalam algoritma genetika. Secara rinci, parameter yang digunakan dalam pengujian disajikan pada Tabel 4.14 berikut:

Tabel 4.14 Parameter Pengujian

Parameter	**Nilai**
Ukuran Populasi	5
Jumlah Generasi	5
Probabilitas Mutasi	0,1
Jumlah Elitsm	2
Jumlah *Run*	35

Parameter-parameter tersebut digunakan secara konsisten pada seluruh proses pengujian sehingga hasil yang diperoleh dapat dibandingkan secara objektif antara metode NNX dan MPNNX.

4.4 Hasil Pengujian Utama

Setelah parameter eksperimen ditentukan, pengujian dilakukan terhadap metode *Nearest Neighbor Crossover* (NNX) dan *Multi-parent Nearest Neighbor Crossover* (MPNNX) menggunakan dataset pr1002. Masing-masing metode dijalankan sebanyak 35 kali percobaan (*run*) untuk memperoleh hasil yang lebih representatif serta mengurangi pengaruh faktor acak pada algoritma genetika.

Pada setiap *run*, program terlebih dahulu membangkitkan populasi awal yang terdiri atas sejumlah individu yang merepresentasikan kandidat solusi *Traveling Salesman Problem* (TSP). Selanjutnya dilakukan proses evaluasi populasi untuk menghitung total jarak yang dimiliki setiap individu. Individu-individu dengan kualitas solusi terbaik kemudian dipilih untuk membentuk generasi berikutnya melalui mekanisme seleksi, *crossover*, mutasi, dan elitisme.

Perbedaan utama antara kedua metode terletak pada operator *crossover* yang digunakan. Metode NNX membentuk *offspring* menggunakan dua *parent* yang dipilih melalui *Roulette wheel selection*, sedangkan metode MPNNX membentuk *offspring* menggunakan tiga *parent*. Setelah proses *crossover* selesai, *offspring* yang dihasilkan berpeluang mengalami mutasi menggunakan operator *swap mutation* sesuai probabilitas mutasi yang telah ditentukan pada parameter eksperimen.

Proses evaluasi, seleksi, *crossover*, mutasi, dan elitisme tersebut diulang selama jumlah generasi yang telah ditentukan. Pada setiap generasi dicatat nilai jarak terbaik yang diperoleh. Apabila ditemukan solusi yang lebih baik dibandingkan solusi terbaik sebelumnya, maka nilai tersebut disimpan sebagai

solusi terbaik global pada *run* yang sedang dijalankan. Implementasi proses eksperimen utama ditunjukkan pada kode program sebagai berikut:

```
def jalankan_satu_run(matriks_jarak, parameter, metode):
    ukuran_populasi = parameter["ukuran_populasi"]
    jumlah_generasi = parameter["jumlah_generasi"]
    probabilitas_mutasi = parameter["probabilitas_mutasi"]
    jumlah_elit = parameter["jumlah_elit"]

    jumlah_titik = matriks_jarak.shape[0]
    populasi = bangkitkan_populasi_awal(ukuran_populasi,
jumlah_titik)

    riwayat_jarak_terbaik = []
    rute_terbaik_global = None
    jarak_terbaik_global = float("inf")

    waktu_mulai = time.time()

    for _ in range(jumlah_generasi):

        tabel_evaluasi = evaluasi_populasi(
            populasi,
            matriks_jarak
        ).sort_values("Total_Jarak").reset_index(drop=True)

        jarak_terbaik_generasi = tabel_evaluasi.loc[0,
"Total_Jarak"]
        riwayat_jarak_terbaik.append(jarak_terbaik_generasi)

        if jarak_terbaik_generasi < jarak_terbaik_global:
            jarak_terbaik_global = jarak_terbaik_generasi
            rute_terbaik_global = tabel_evaluasi.loc[0,
"Rute"]

        elit = list(tabel_evaluasi.head(jumlah_elit)["Rute"])
        populasi_baru = elit.copy()

        while len(populasi_baru) < ukuran_populasi:
            anak = buat_offspring(
                populasi,
                tabel_evaluasi,
                matriks_jarak,
                metode,
                probabilitas_mutasi
            )
```

```
            populasi_baru.append(anak)

        populasi = populasi_baru

    waktu_selesai = time.time()

    return {
        "Metode": metode,
        "Jarak_Terbaik": jarak_terbaik_global,
        "Rute_Terbaik": rute_terbaik_global,
        "Riwayat_Jarak_Terbaik": riwayat_jarak_terbaik,
        "Waktu_Komputasi": waktu_selesai - waktu_mulai
    }
```

Kode program tersebut menunjukkan proses utama algoritma genetika yang digunakan dalam penelitian. Pada fungsi tersebut, populasi dievaluasi pada setiap generasi, kemudian dibentuk populasi baru melalui proses reproduksi hingga diperoleh solusi terbaik untuk satu kali *run* pengujian.

Setelah fungsi eksperimen utama selesai dibuat, pengujian dilakukan sebanyak 35 kali untuk setiap metode. Pada setiap *run*, program menyimpan informasi berupa nama metode, nomor *run*, nilai jarak terbaik yang diperoleh, serta waktu komputasi. Implementasi proses pengujian ditunjukkan pada kode program sebagai berikut:

```
for metode in daftar_metode:
    semua_riwayat[metode] = []
    for run_ke in range(1, parameter_eksperimen["jumlah_run"]
+ 1):
        print(f"Menjalankan metode {metode}, run ke-{run_ke}
...")
        hasil_run = jalankan_satu_run(matriks_jarak_utama,
parameter_eksperimen, metode)
        hasil_run["Run"] = run_ke
        semua_hasil_run.append({
            "Metode": hasil_run["Metode"],
            "Run": run_ke,
            "Jarak_Terbaik": hasil_run["Jarak_Terbaik"],
            "Waktu_Komputasi": hasil_run["Waktu_Komputasi"]
        })
```

Hasil dari seluruh pengujian kemudian direkap ke dalam sebuah dataframe sehingga membentuk tabel hasil pengujian setiap *run*. Implementasi pembentukan tabel hasil ditunjukkan pada kode program sebagai berikut:

```
tabel_hasil_run = pd.DataFrame(semua_hasil_run)
print("\nTabel hasil per run:")
display(tabel_hasil_run)
```

Karena algoritma genetika mengandung unsur acak pada proses pembangkitan populasi awal, pemilihan *parent*, dan mutasi, maka nilai jarak terbaik yang diperoleh pada setiap *run* dapat berbeda. Oleh karena itu, pengujian dilakukan sebanyak 35 kali untuk memperoleh gambaran performa metode yang lebih objektif dan mengurangi pengaruh faktor acak terhadap hasil penelitian. Hasil pengujian untuk setiap *run* ditunjukkan sebagai berikut:

Tabel 4.15 Hasil Pengujian Jarak Terbaik Setiap *Run*

Run	**Jarak Terbaik NNX**	**Jarak Terbaik MPNNX**
1	2091897,234	2172247,85
2	2108702,222	1727603,169
3	2276350,859	1710182,997
4	2712530,667	2816674,754
5	2099715,199	1536076,7
6	2158292,345	1909596,98
7	2306234,19	1557280,453
8	2091981,635	1733961,97
9	2754893,4	1799520,622
10	2875402,673	1974018,004
11	2294867,695	1816746,231
12	2872468,406	2187052,489
13	1931371,533	1598947,715
14	2546674,267	1943622,338
15	1989636,154	1833271,505
16	2341626,099	1894714,909
17	2914680,092	2441838,712
18	1611346,468	2622885,896
19	2845262,273	1720872,102
20	3039305,936	2036244,337
21	2447745,701	1496166,741
22	3052363,253	1735520,447
23	1925235,473	2731952,994
24	2869127,195	1923843,181
25	2585453,531	2015286,959
26	2473130,865	1601271,022
27	2645995,588	1517116,76
28	2107292,705	1492421,327
29	2444837,325	1480099,216
30	2236650,178	2030918,638
31	2044412,977	2740429,008
32	2243348,833	2066064,725
33	2102042,865	1600982,69
34	2138582,581	3066036,441
35	2554969,716	1912051,552

Setelah seluruh hasil pengujian dari 35 kali *run* pada masing-masing metode diperoleh, data hasil eksperimen kemudian direkap untuk dilakukan analisis statistik. Data yang digunakan berasal dari tabel hasil pengujian yang telah menyimpan informasi berupa metode, nomor *run*, dan jarak terbaik yang diperoleh pada setiap percobaan.

Analisis statistik deskriptif dilakukan untuk memberikan gambaran umum mengenai performa masing-masing metode berdasarkan hasil pengujian yang telah diperoleh. Statistik yang digunakan meliputi jumlah *run*, nilai jarak minimum, nilai jarak rata-rata, dan simpangan baku. Nilai minimum digunakan untuk mengetahui solusi terbaik yang berhasil ditemukan selama seluruh percobaan, sedangkan nilai rata-rata dan simpangan baku digunakan untuk melihat kecende*run*gan hasil serta tingkat kestabilan metode yang diuji. Proses rekapitulasi statistik dilakukan sebagai berikut:

```
tabel_ringkasan = tabel_hasil_run.groupby("Metode").agg(
    Jumlah_Run=("Run", "count"),
    Jarak_Minimum=("Jarak_Terbaik", "min"),
    Jarak_Rata_Rata=("Jarak_Terbaik", "mean"),
    Simpangan_Baku=("Jarak_Terbaik", "std"),
).reset_index()

print("Ringkasan statistik hasil eksperimen:")
display(tabel_ringkasan)
```

Hasil rekap statistik pengujian utama ditunjukkan pada Tabel 4.16.

Tabel 4.16 Ringkasan Statistik Hasil Pengujian

Metode	Jarak Minimum	Jarak Rata-Rata	Simpangan Baku
MPNNX	1.480.099,216	1.955.529,184	416.327,544
NNX	1.611.346,468	2.392.412,232	362.023,759

Berdasarkan Tabel 4.16, kedua metode menunjukkan adanya variasi hasil pada setiap *run*. Hal ini merupakan karakteristik umum dari algoritma genetika yang menggunakan proses probabilistik dalam pembentukan solusi. Dilihat dari

nilai jarak minimum dan rata-rata, metode MPNNX menghasilkan nilai total jarak yang lebih kecil dibandingkan NNX. Hal ini menunjukkan bahwa MPNNX memiliki kemampuan yang lebih baik dalam menghasilkan solusi rute yang lebih optimal pada dataset yang digunakan.

Nilai simpangan baku yang diperoleh pada kedua metode menunjukkan bahwa hasil pengujian masih mengalami variasi antar *run*. Variasi tersebut dipengaruhi oleh proses acak dalam pembangkitan populasi awal, pemilihan *parent*, dan mutasi. Semakin kecil nilai simpangan baku, maka semakin konsisten hasil yang diperoleh suatu metode. Berdasarkan nilai simpangan baku pada Tabel 4.16, dapat diamati tingkat kestabilan masing-masing metode selama 35 kali percobaan.

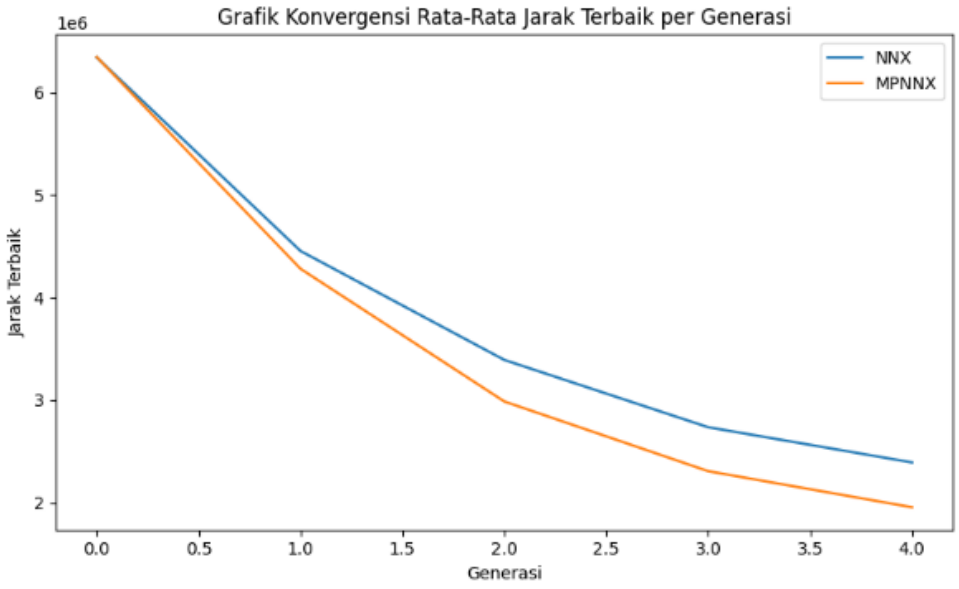
Selain dilakukan rekapitulasi statistik hasil pengujian, analisis juga dilakukan terhadap proses perkembangan solusi selama evolusi algoritma genetika menggunakan grafik konvergensi. Grafik konvergensi digunakan untuk menggambarkan perubahan nilai jarak terbaik pada setiap generasi sehingga dapat diketahui kecende*rung*an proses pencarian solusi yang dilakukan oleh masing-masing metode.

Data yang digunakan untuk membentuk grafik konvergensi berasal dari riwayat nilai jarak terbaik yang disimpan pada setiap generasi selama 35 kali *run* pengujian. Karena setiap metode dijalankan sebanyak 35 kali percobaan, maka nilai yang ditampilkan pada grafik merupakan rata-rata jarak terbaik pada setiap generasi dari seluruh *run* yang dilakukan. Dengan demikian, grafik yang dihasilkan dapat memberikan gambaran umum mengenai perilaku konvergensi masing-masing metode dan tidak hanya merepresentasikan satu kali percobaan tertentu. Proses pembentukan grafik konvergensi dilakukan sebagai berikut:

```
plt.figure(figsize=(8, 5))
for metode, daftar_riwayat in semua_riwayat.items():
    rata_rata_per_generasi = np.mean(np.array(daftar_riwayat),
axis=0)
    plt.plot(rata_rata_per_generasi, label=metode)

plt.title("Grafik Konvergensi Rata-Rata Jarak Terbaik per
Generasi")
plt.xlabel("Generasi")
plt.ylabel("Jarak Terbaik")
plt.legend()
plt.tight_layout()
plt.show()
```

Hasil grafik konvergensi rata-rata jarak terbaik pada setiap generasi untuk metode NNX dan MPNNX ditampilkan sebagai berikut:



Gambar 4.1 Grafik Konvergansi Rata-rata Jarak Terbaik per Generasi

Berdasarkan grafik tersebut, terlihat bahwa kedua metode mengalami penu*run*an nilai jarak seiring bertambahnya generasi, yang menunjukkan adanya proses perbaikan solusi. Namun, metode MPNNX menunjukkan penu*run*an yang lebih signifikan dibandingkan NNX, sehingga mampu mencapai nilai jarak yang lebih rendah pada generasi akhir. Hal ini menunjukkan bahwa penggunaan lebih

dari satu *parent* pada MPNNX meningkatkan kemampuan eksplorasi solusi sehingga konvergensi menuju solusi optimal menjadi lebih cepat.

4.5 Analisis Statistik Deskriptif

Analisis statistik deskriptif dilakukan untuk mengevaluasi performa metode *Nearest Neighbor Crossover* (NNX) dan *Multi-parent Nearest Neighbor Crossover* (MPNNX) berdasarkan hasil pengujian yang telah diperoleh. Analisis ini mencakup parameter utama, yaitu nilai minimum, rata-rata, serta simpangan baku dari total jarak yang dihasilkan.

Berdasarkan hasil yang disajikan pada Tabel 4.16, metode MPNNX menghasilkan nilai rata-rata total jarak sebesar 1.955.529,184, yang lebih rendah dibandingkan dengan NNX sebesar 2.392.412,232. Hal ini menunjukkan bahwa secara umum MPNNX cende*run*g menghasilkan solusi rute dengan jarak yang lebih pendek.

Selain itu, nilai minimum yang dihasilkan oleh MPNNX adalah sebesar 1.480.099,216 yang diperoleh pada *run* ke-29. Sementara itu, NNX menghasilkan nilai minimum sebesar 1.611.346,468 yang diperoleh pada *run* ke-18. Hasil ini menunjukkan bahwa pada kondisi terbaiknya, MPNNX mampu menghasilkan rute dengan total jarak yang lebih kecil dibandingkan NNX.

Namun, nilai simpangan baku pada metode MPNNX sebesar 416.327,544 lebih tinggi dibandingkan dengan NNX sebesar 362.023,759. Hal ini menunjukkan bahwa hasil yang diperoleh MPNNX memiliki tingkat variasi yang lebih besar antar *run*, sehingga tingkat konsistensinya relatif lebih rendah dibandingkan NNX.

Secara keseluruhan, hasil analisis menunjukkan bahwa metode MPNNX cende*run*g menghasilkan jarak yang lebih kecil, baik dari sisi nilai rata-rata maupun

nilai minimum. Sementara itu, metode NNX menunjukkan hasil yang lebih stabil, yang ditunjukkan oleh nilai simpangan baku yang lebih rendah.

4.6 Analisis Uji Statistik

Analisis uji statistik dilakukan untuk mengetahui apakah terdapat perbedaan kinerja yang signifikan antara operator *Nearest Neighbor Crossover* (NNX) dan *Multi-parent Nearest Neighbor Crossover* (MPNNX) dalam algoritma genetika berdasarkan nilai jarak terbaik yang diperoleh dari 35 kali percobaan pada dataset pr1002. Sebelum dilakukan pengujian hipotesis, terlebih dahulu dilakukan uji normalitas terhadap data hasil jarak terbaik yang diperoleh dari 35 kali *run* pada masing-masing metode. Uji normalitas bertujuan untuk mengetahui apakah data hasil pengujian mengikuti distribusi normal atau tidak. Hasil uji normalitas selanjutnya digunakan sebagai dasar dalam menentukan jenis uji statistik yang sesuai untuk membandingkan performa metode NNX dan MPNNX.

Pada penelitian ini, uji normalitas dilakukan menggunakan metode Shapiro-Wilk. Metode ini dipilih karena memiliki kemampuan yang baik dalam menguji normalitas pada ukuran sampel kecil hingga menengah. Statistik uji *Shapiro-Wilk* dihitung menggunakan persamaan (Bruce et al., 2020):

$$W = \frac{\left(\sum_{i=1}^{n} a_i x_{(i)}\right)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

dengan W merupakan statistik uji *Shapiro-Wilk*, $x_{(i)}$ adalah data yang telah diurutkan, a_i merupakan konstanta *Shapiro-Wilk*, x_i adalah data ke-i, $\bar{x}$ adalah rata-rata sampel, dan n adalah jumlah data. Pada uji ini, hipotesis nol (H_0) menyatakan bahwa data berdistribusi normal, sedangkan hipotesis alternatif (H_1) menyatakan bahwa data tidak berdistribusi normal. Kriteria pengujian yang digunakan adalah

menerima H_0 apabila nilai p-value lebih besar dari 0,05 dan menolak H_0 apabila nilai p-value kurang dari atau sama dengan 0,05.

Pengujian normalitas dilakukan menggunakan fungsi `shapiro()` dari pustaka SciPy. Implementasi pengujian ditunjukkan sebagai berikut:

```
stat_norm_a, p_norm_a = stats.shapiro(data_a)
stat_norm_b, p_norm_b = stats.shapiro(data_b)
```

Hasil uji normalitas *Shapiro-Wilk* ditunjukkan pada Tabel 4.17.

Tabel 4.17 Hasil Uji Normalitas Shapiro Wilk

Metode	Statistik Uji	p-value	Keterangan
NNX	0,9581	0,1998	Data berdistribusi normal
MPNNX	0,8765	0,0010	Data tidak berdistribusi normal

Berdasarkan Tabel 4.17, metode NNX memperoleh nilai statistik uji sebesar 0,9581 dengan p-value sebesar 0,1998. Karena nilai p-value lebih besar dari 0,05, maka data hasil jarak terbaik pada metode NNX dapat dinyatakan berdistribusi normal. Sementara itu, metode MPNNX memperoleh nilai statistik uji sebesar 0,8765 dengan p-value sebesar 0,0010. Nilai p-value tersebut lebih kecil dari 0,05 sehingga data hasil jarak terbaik pada metode MPNNX tidak berdistribusi normal. Karena terdapat setidaknya satu kelompok data yang tidak memenuhi asumsi normalitas, maka pengujian hipotesis dilanjutkan menggunakan uji nonparametrik Mann-Whitney U.

Uji Mann-Whitney U digunakan untuk mengetahui apakah terdapat perbedaan yang signifikan antara dua kelompok data independen tanpa mensyaratkan distribusi normal. Pada penelitian ini, uji Mann–Whitney U digunakan untuk membandingkan nilai jarak terbaik yang dihasilkan oleh metode NNX dan MPNNX selama 35 kali *run* pengujian. Statistik uji Mann-Whitney U dihitung menggunakan persamaan (Mann & Whitney, 1947):

$$U_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1$$

$$U_2 = n_1 n_2 + \frac{n_2(n_2+1)}{2} - R_2$$

$$U = min(U_1, U_2)$$

dengan U merupakan statistik Mann-Whitney, n_1 dan n_2 masing-masing adalah jumlah sampel pada kelompok pertama dan kedua, sedangkan R_1 dan R_2 adalah jumlah peringkat dari masing-masing kelompok. Hipotesis nol (H_0) menyatakan bahwa tidak terdapat perbedaan yang signifikan antara hasil yang diperoleh NNX dan MPNNX, sedangkan hipotesis alternatif (H_1) menyatakan bahwa terdapat perbedaan yang signifikan antara kedua metode tersebut.

Pengujian dilakukan menggunakan fungsi `mannwhitneyu()` dari pustaka SciPy. Implementasi pengujian ditunjukkan sebagai berikut:

```
statistik_uji, p_value = stats.mannwhitneyu(
        data_a, data_b, alternative="two-sided"
    )
```

Hasil uji Mann-Whitney U ditunjukkan pada Tabel 4.18.

Tabel 4.18 Hasil Uji Mann-Whitney U

Statistik Uji	p-value	Kesimpulan
1003,0000	0,0000046281	Terdapat perbedaan yang signifikan

Berdasarkan hasil pengujian diperoleh nilai statistik uji sebesar 1003,0000 dengan p-value sebesar 0,0000046281. Nilai p-value tersebut lebih kecil daripada taraf signifikansi 0,05 sehingga hipotesis nol (H_0) ditolak. Dengan demikian, dapat disimpulkan bahwa terdapat perbedaan yang signifikan antara kinerja operator NNX dan MPNNX dalam menyelesaikan permasalahan *Traveling Salesman Problem* pada dataset pr1002.

Hasil ini mengindikasikan bahwa perbedaan performa yang diperoleh tidak terjadi secara kebetulan, melainkan dipengaruhi oleh perbedaan metode yang digunakan. Namun, uji statistik ini hanya menunjukkan adanya perbedaan yang signifikan antara kedua metode, dan tidak secara langsung menunjukkan bahwa salah satu metode lebih baik. Oleh karena itu, interpretasi hasil perlu dikaitkan dengan analisis statistik deskriptif yang telah dilakukan sebelumnya, seperti nilai rata-rata dan simpangan baku.

4.7 Pembahasan Hasil

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, dapat diketahui bahwa metode *Nearest Neighbor Crossover* (NNX) dan *Multi-Parent Nearest Neighbor Crossover* (MPNNX) menunjukkan perbedaan performa dalam menyelesaikan permasalahan *Traveling Salesman Problem* (TSP).

Dari hasil statistik deskriptif, metode MPNNX secara umum menghasilkan nilai total jarak yang lebih rendah dibandingkan dengan metode NNX. Hal ini menunjukkan bahwa MPNNX cenderung menghasilkan rute dengan jarak yang lebih pendek. Namun demikian, jika dilihat dari nilai simpangan baku, metode MPNNX memiliki nilai yang lebih besar dibandingkan NNX. Nilai simpangan baku yang lebih tinggi menunjukkan bahwa hasil yang diperoleh MPNNX cende*run*g lebih bervariasi antar percobaan, sehingga tingkat konsistensinya relatif lebih rendah.

Hasil uji statistik menggunakan metode Mann-Whitney U menunjukkan bahwa perbedaan performa antara kedua metode tersebut bersifat signifikan secara statistik. Dengan nilai *p-value* yang lebih kecil dari tingkat signifikansi 0,05, dapat disimpulkan bahwa perbedaan yang terjadi tidak disebabkan oleh faktor kebetulan.

Namun, hasil uji statistik ini hanya menunjukkan adanya perbedaan performa, dan tidak secara langsung menunjukkan bahwa salah satu metode lebih unggul.

Dengan mempertimbangkan hasil analisis statistik deskriptif, dapat dipahami bahwa metode MPNNX cenderung memberikan hasil yang lebih baik dari sisi nilai jarak, tetapi dengan tingkat variasi hasil yang lebih tinggi. Sebaliknya, metode NNX menunjukkan hasil yang lebih stabil, meskipun nilai jarak yang dihasilkan relatif lebih besar. Dengan demikian, kedua metode memiliki karakteristik yang berbeda, di mana MPNNX lebih berfokus pada kualitas solusi (jarak), sedangkan NNX lebih unggul dalam hal konsistensi hasil.

4.8 Optimasi Rute dalam Perspektif Islam

Penelitian mengenai optimasi rute menggunakan algoritma genetika dengan penerapan mekanisme *multi-parent* pada *Nearest Neighbor Crossover* (NNX) tidak hanya memiliki nilai ilmiah dalam bidang komputasi, tetapi juga dapat dianalisis dalam perspektif nilai-nilai Islam. Penggunaan metode *Multi-parent Nearest Neighbor Crossover* (MPNNX) dalam mencari rute terpendek mencerminkan prinsip kerja optimal, sistematis, dan berorientasi pada kualitas hasil. Hal ini memiliki keselarasan dengan ajaran Islam yang menekankan kesungguhan usaha, kualitas amal, dan ketelitian dalam setiap proses. Untuk memahami prinsip usaha dalam perspektif Islam, salah satu dasar yang dapat digunakan adalah firman Allah SWT dalam Q.S. An-Najm ayat 39 sebagai berikut:

وَاَنْ لَّيْسَ لِلْاِنْسَانِ اِلَّا مَا سَعٰىۙ ﴿٣٩﴾

(Kementerian Agama Republik Indonesia, 2022)

"Bahwa manusia hanya memperoleh apa yang telah diusahakannya," (Q.S. An-Najm: 39)

Menurut *Tafsir Ibnu Katsir* (Katsir & Umar, 2005), ayat ini menegaskan bahwa hasil yang diperoleh manusia bergantung pada usaha yang dilakukan. Tidak ada hasil tanpa proses, dan tidak ada pencapaian tanpa kerja nyata. Dalam tafsir tersebut juga dijelaskan bahwa pahala maupun konsekuensi tidak dapat dipindahkan kepada orang lain, sehingga setiap individu memiliki tanggung jawab penuh atas proses dan hasil usahanya. Hal ini menunjukkan bahwa dalam Islam, proses kerja memiliki peran penting dalam menentukan hasil akhir. Dalam penjelasan tersebut, usaha yang dilakukan secara terbatas atau terhenti di tengah proses cende*rung* menghasilkan hasil yang berbeda dibandingkan dengan usaha yang dilakukan secara maksimal dan berkelanjutan. Oleh karena itu, keberhasilan tidak hanya ditentukan oleh adanya usaha, tetapi juga oleh konsistensi usaha tersebut hingga tujuan tercapai.

Dalam konteks penelitian ini, prinsip tersebut sangat relevan dengan mekanisme algoritma genetika. Algoritma tidak dirancang untuk menemukan solusi dalam satu langkah, melainkan melalui proses iteratif yang panjang, yaitu melalui generasi demi generasi. Setiap generasi merupakan bentuk usaha sistem dalam mendekati solusi optimal. Jika proses dihentikan terlalu cepat (misalnya sebelum konvergensi tercapai), maka solusi yang diperoleh cende*rung* masih suboptimal.

Dengan demikian, prinsip usaha dalam Islam tidak hanya menekankan pentingnya melakukan pekerjaan, tetapi juga menuntut konsistensi dan kesungguhan dalam prosesnya. Namun, usaha yang maksimal saja belum cukup apabila tidak diarahkan pada hasil yang berkualitas. Oleh karena itu, Islam tidak hanya menilai seberapa besar usaha yang dilakukan, tetapi juga seberapa baik kualitas hasil dari usaha tersebut. Prinsip tersebut ditegaskan dalam Q.S. Al-Mulk

ayat 2, yang menunjukkan bahwa penilaian terhadap amal tidak didasarkan pada banyaknya, melainkan pada kualitasnya:

الَّذِيْ خَلَقَ الْمَوْتَ وَالْحَيٰوةَ لِيَبْلُوَكُمْ اَيُّكُمْ اَحْسَنُ عَمَلًاۗ وَهُوَ الْعَزِيْزُ الْغَفُوْرُۙ ٢

(Kementerian Agama Republik Indonesia, 2022)

"Yaitu yang menciptakan kematian dan kehidupan untuk menguji kamu, siapa di antara kamu yang lebih baik amalnya. Dia Maha Perkasa lagi Maha Pengampun." (Q.S. Al-Mulk: 2)

Menurut (Katsir & Umar, 2005) pada "*tafsir Ibnu Katsir*" dijelaskan bahwa konsep *ahsanu 'amala* menekankan kualitas amal, bukan kuantitasnya. Artinya, ukuran keberhasilan bukan terletak pada banyaknya aktivitas yang dilakukan, tetapi pada seberapa baik hasil yang dicapai. Penekanan pada kualitas ini menunjukkan bahwa dalam Islam, suatu pekerjaan dinilai dari tingkat ketepatan, kebermanfaatan, dan kesempurnaannya, bukan sekadar dari jumlah usaha yang terlihat secara kuantitatif. Dengan demikian, aktivitas yang banyak namun tidak menghasilkan kualitas yang baik tidak memiliki nilai yang lebih tinggi dibandingkan aktivitas yang lebih sedikit tetapi menghasilkan hasil yang optimal.

Dalam konteks penelitian ini, prinsip tersebut tidak bertentangan dengan tujuan untuk memperoleh jarak yang paling kecil. Hal ini karena nilai jarak yang lebih kecil tidak merepresentasikan jumlah yang lebih sedikit, melainkan menunjukkan tingkat optimalitas solusi. Jarak dalam permasalahan *Traveling Salesman Problem* merupakan indikator efisiensi, sehingga semakin kecil nilai yang dihasilkan, semakin tinggi kualitas solusi tersebut. Dengan kata lain, pengurangan jarak bukan berarti mengurangi "kuantitas kerja", tetapi merupakan bentuk peningkatan kualitas hasil melalui proses optimasi yang lebih baik.

Lebih lanjut, dalam algoritma genetika, kualitas solusi direpresentasikan melalui nilai *fitness*. Solusi dengan nilai *fitness* terbaik (jarak paling kecil) akan memiliki peluang lebih besar untuk dipilih dan dikembangkan pada generasi berikutnya. Hal ini menunjukkan bahwa sistem secara alami menerapkan prinsip seleksi berbasis kualitas, di mana solusi yang kurang optimal akan dieliminasi. Dengan demikian, proses ini tidak menekankan banyaknya solusi yang dihasilkan, tetapi pada kemampuan sistem dalam menemukan dan mempertahankan solusi terbaik dari sekumpulan alternatif yang ada.

Oleh karena itu, optimasi dalam algoritma genetika bukan berfokus pada memperbanyak jumlah solusi, tetapi pada proses seleksi dan perbaikan solusi secara bertahap. Banyaknya solusi dalam setiap generasi hanya berfungsi sebagai ruang eksplorasi untuk menemukan kemungkinan kombinasi yang lebih baik. Namun, tujuan akhirnya tetap pada konvergensi menuju solusi yang paling optimal. Hal ini mencerminkan bahwa kualitas hasil merupakan tujuan utama, sedangkan kuantitas hanya menjadi sarana dalam proses pencapaiannya.

Dengan demikian, konsep *ahsanu 'amala* dalam ayat ini memiliki keselarasan yang kuat dengan tujuan penelitian, yaitu menghasilkan solusi dengan kualitas paling optimal melalui proses seleksi dan perbaikan yang berkelanjutan. Penelitian ini tidak hanya menghasilkan banyak alternatif solusi, tetapi secara sistematis mengarahkan proses menuju solusi terbaik, sehingga sejalan dengan prinsip Islam yang menekankan pencapaian kualitas terbaik dalam setiap usaha.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa metode *Nearest Neighbor Crossover* (NNX) dan *Multi-Parent Nearest Neighbor Crossover* (MPNNX) dapat digunakan untuk menyelesaikan permasalahan *Traveling Salesman Problem* (TSP). Hasil penelitian menunjukkan bahwa metode MPNNX menghasilkan nilai rata-rata total jarak sebesar 1.955.529,184 yang lebih rendah dibandingkan metode NNX sebesar 2.392.412,232. Selain itu, nilai minimum yang dihasilkan oleh MPNNX sebesar 1.480.099,216 juga lebih kecil dibandingkan NNX sebesar 1.611.346,468, sehingga MPNNX cende*run*g menghasilkan rute dengan jarak yang lebih pendek.

Di sisi lain, metode MPNNX memiliki nilai simpangan baku yang lebih tinggi dibandingkan NNX, yang menunjukkan bahwa hasil yang diperoleh lebih bervariasi dan tingkat konsistensinya relatif lebih rendah. Sebaliknya, metode NNX menunjukkan hasil yang lebih stabil meskipun menghasilkan jarak yang lebih besar. Selain itu, hasil uji statistik menggunakan *Mann-Whitney U Test* menunjukkan bahwa terdapat perbedaan yang signifikan antara kedua metode, dengan nilai p-value sebesar 0,0000046281 < 0,05, sehingga perbedaan performa yang terjadi tidak disebabkan oleh faktor kebetulan.

Secara keseluruhan, pada eksperimen yang dilakukan dalam penelitian ini, metode MPNNX menunjukkan performa yang lebih baik dibandingkan metode NNX, khususnya dalam menghasilkan nilai total jarak yang lebih kecil. Namun demikian, hasil tersebut masih terbatas pada dataset dan parameter yang digunakan,

sehingga diperlukan pengujian lebih lanjut pada dataset yang berbeda serta variasi parameter yang lebih beragam untuk memastikan konsistensi dan generalisasi hasil penelitian.

5.2 Saran

Berdasarkan hasil dan keterbatasan penelitian yang telah dilakukan, terdapat beberapa saran untuk pengembangan lebih lanjut:

1. Penelitian ini menggunakan parameter algoritma genetika yang ditentukan secara tetap, seperti ukuran populasi, jumlah generasi, dan probabilitas mutasi. Penelitian selanjutnya dapat mengeksplorasi metode penentuan parameter secara adaptif atau menggunakan pendekatan optimasi parameter untuk memperoleh konfigurasi yang lebih optimal.
2. Fungsi *fitness* yang digunakan dalam penelitian ini hanya mempertimbangkan total jarak sebagai satu-satunya kriteria. Penelitian selanjutnya dapat mengembangkan fungsi *fitness* yang lebih kompleks atau *multi-objective*, dengan mempertimbangkan faktor lain seperti waktu tempuh, biaya, atau kondisi tertentu, sehingga solusi yang dihasilkan menjadi lebih realistis dan aplikatif.
3. Metode *crossover* yang digunakan dalam penelitian ini terbatas pada NNX dan MPNNX. Penelitian selanjutnya dapat mengkaji dan membandingkan dengan metode *crossover* lain atau mengkombinasikannya dengan teknik optimasi lokal seperti 2-opt atau *local search* untuk meningkatkan kualitas solusi.
4. Dataset yang digunakan dalam penelitian ini adalah dataset statis dari TSPLIB. Penelitian selanjutnya dapat menguji metode pada dataset dengan

karakteristik yang lebih beragam atau pada permasalahan nyata yang bersifat dinamis untuk melihat konsistensi performa algoritma.

5. Evaluasi dalam penelitian ini difokuskan pada kualitas solusi berdasarkan total jarak dan uji statistik. Penelitian selanjutnya dapat menambahkan analisis performa dari segi waktu komputasi dan efisiensi algoritma secara lebih mendalam.
6. Penelitian ini menggunakan jumlah *parent* tetap pada metode MPNNX. Penelitian selanjutnya dapat mengeksplorasi variasi jumlah *parent* untuk menganalisis pengaruhnya terhadap keseimbangan antara kualitas solusi dan stabilitas hasil.

DAFTAR PUSTAKA

Alkafaween, E., Hassanat, A., Essa, E., & Elmougy, S. (2024). An Efficiency Boost for Genetic Algorithms: Initializing the GA with the Iterative Approximate Method for Optimizing the *Traveling Salesman Problem*—Experimental Insights. *Applied Sciences (Switzerland)*, *14*(8). https://doi.org/10.3390/app14083151

Applegate, D. L., Bixby, R. E., Chvátal, V., & Cook, W. J. (2011). *The Traveling Salesman Problem: A Computational Study*. https://books.google.co.id/books?id=zfIm94nNqPoC&printsec=frontcover&hl=id&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Baty, L., Jungel, K., Klein, P. S., Parmentier, A., & Schiffer, M. (2024). Combinatorial Optimization-Enriched Machine Learning to Solve the Dynamic Vehicle Routing Problem with Time Windows. *Transportation Science*, *58*(4), 708–725. https://doi.org/10.1287/trsc.2023.0107

Boyd, S., & Vandenberghe, L. (2004). *Convex Optimization*.

Bruce, P., Bruce, A., & Gedeck, P. (2020). *Practical Statistics for Data Scientists*. https://datapot.vn/wp-content/uploads/2023/12/datapot.vn-Practical-Statistics-for-Data-Scientists.pdf?srsltid=AfmBOoqtKct7hU8yKl5T7sR2wwxswotOIHEnEgHI5tu_-pUE8vUk4g46

Dalkilic, S. B., Ozgur, A., & Erdem, H. (2025). Comparison of Genetic *Crossover* Operators for *Traveling Salesman Problem*. *Gazi University Journal of Science*, *38*(2), 751–778. https://doi.org/10.35378/gujs.1582521

Fontaine, R., Dibangoye, J., & Solnon, C. (2023). Exact and anytime approach for solving the time dependent *Traveling Salesman Problem* with time windows. *European Journal of Operational Research*, *311*(3), 833–844. https://doi.org/10.1016/j.ejor.2023.06.001

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization & Machine Learning. In *American Society of Mechanical Engineers, Dynamic Systems and Control Division (Publication) DSC* (Vol. 22, pp. 39–45).

Hillier, F. S., & Lieberman, G. J. (2015). *Introduction to operations research*.

Holland, J. H. (1975). Adaption in Natural and Artificial Systems. *Vegetatio*, *30*(3), 213–219.

Isa, F. M., Ariffin, W. N. M., Jusoh, M. S., & Putri, E. P. (2024). A Review of Genetic Algorithm: Operations and Applications. *Journal of Advanced Research in Applied Sciences and Engineering Technology*, *40*(1), 1–34. https://doi.org/10.37934/araset.40.1.134

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A Review On Genetic Algorithm Past Present And Future.pdf. In *Multimedia Tools and Applications* (Vol. 80). Multimedia Tools and Applications.

Katsir, I., & Umar, I. bin. (2005). *Tafsir Ibnu Katsir* (Cetakan Pe). Pustaka Imam

Asy-Syafi'i. https://www.a*run*gsejarah.com/2023/02/tafsir-ibnu-katsir-jilid-1-8.html

Kementerian Agama Republik Indonesia. (2022). *Al-Qur'an dan Terjemahannya*. Lajnah Pentashihan Mushaf Al-Qur'an.

Kharisma, M. L., Ahmad, A., & Malik, R. (2024). Product Distribution Route Optimization Using The *Traveling Salesman Problem* (Tsp) Method At Pt. Inbisco Niagatama Semesta (Mayora Group) In Makassar. *Journal of Industrial System Engineering and Management*, *3*(1), 29–37. https://doi.org/10.56882/jisem.v3i1.27

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (2023). Optimization by Simulated Annealing. *Springer Tracts in Civil Engineering*, *220*(4598), 15–24. https://doi.org/10.1007/978-981-99-2213-0_2

Mahmoudinazlou, S., & Kwon, C. (2024). A hybrid genetic algorithm for the min–max Multiple *Traveling Salesman Problem*. *Computers and Operations Research*, *162*, 1–29. https://doi.org/10.1016/j.cor.2023.106455

Mann, H. B., & Whitney, D. R. (1947). *On A Test Of Whether One Of Two Random Variables Is Stochastically Larger Than The Other*. *18*, 50–60. https://www.jstor.org/stable/2236101

Melanie, M. (1998). *An Introduction to Genetic Algorithms*.

Onder, I. (2007). *A Genetic Algorithm For TSP with Backhauls Based on Conventional Heuristics*. *September*. http://www.thesis.bilkent.edu.tr/0002464.pdf

Pop, P. C., Cosma, O., Sabo, C., & Sitar, C. P. (2024). A comprehensive survey on the generalized *Traveling Salesman Problem*. *European Journal of Operational Research*, *314*(3), 819–835. https://doi.org/10.1016/j.ejor.2023.07.022

Potvin, J.-Y. (n.d.). *Genetic Algorithms for the Traveling Salesman Problem*. 1–42.

Pretorius, K., & Pillay, N. (2024). Neural network *crossover* in genetic algorithms using genetic programming. *Genetic Programming and Evolvable Machines*, *25*(1), 1–30. https://doi.org/10.1007/s10710-024-09481-7

Reinelt, G. (1991). *Tsplib 95*. 1–16.

Sharma, M. K., Chaudhary, S., Rathour, L., & Mishra, V. N. (2024). Modified genetic algorithm with novel *crossover* and mutation operator for travelling salesman problem. *Sigma Journal of Engineering and Natural Sciences*, *42*(6), 1876–1883. https://doi.org/10.14744/sigma.2023.00105

Shenmaier, V. (2022). *Asymptotic Optimality of the Greedy Patching Heuristic for Max TSP in Doubling Metrics*. *2*, 1–7. http://arxiv.org/abs/2201.03813

Soni, N., & Kumar, D. . T. (2007). A Study of *Crossover* Operators in Genetic Algorithms. *Frontiers in Nature-Inspired Industrial Optimization*, *5*(6), 7235–7238. https://link.springer.com/chapter/10.1007/978-981-16-3128-3_2

Srinath Murthy, A., & P Mankame, D. (2024). Genetic Algorithms - A Brief Study. *International Journal of Science and Research (IJSR)*, *13*(7), 1195–1200. https://doi.org/10.21275/sr24721004409

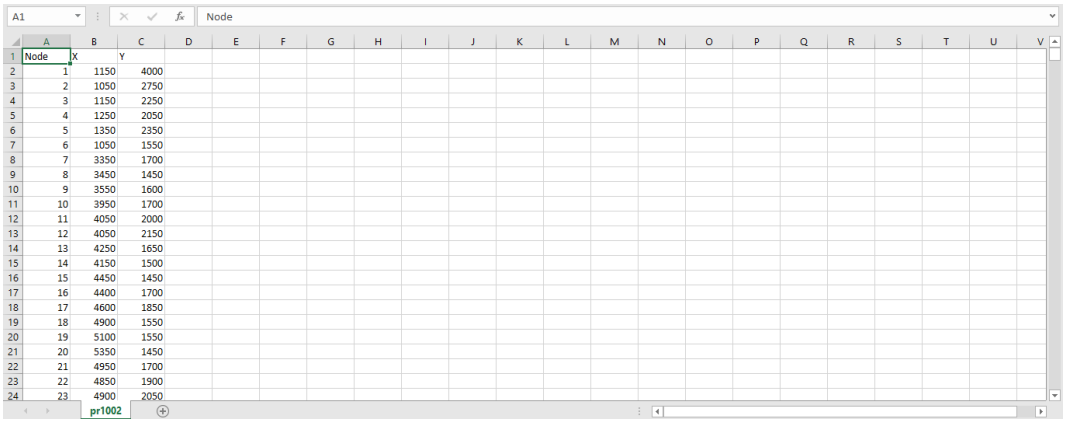
Srinivas, M., & Patnaik, L. M. (1994). Adaptive Probabilities of *Crossover* and Mutation in Genetic Algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, *24*(4), 656–667. https://doi.org/10.1109/21.286385

Zibaei, H., & Mesgari, M. S. (2023). *Improved discrete particle swarm optimization using Bee Algorithm and multi-parent crossover method (Case study : Allocation problem and benchmark functions)*.

LAMPIRAN

Lampiran 1 Dataset TSPLIB pr1002

https://docs.google.com/spreadsheets/d/1or8LHXw2n5EBaJwgbZqxcjtS-cwKj4Ik/edit?usp=drive_link&ouid=117294901494638134019&rtpof=true&sd=true

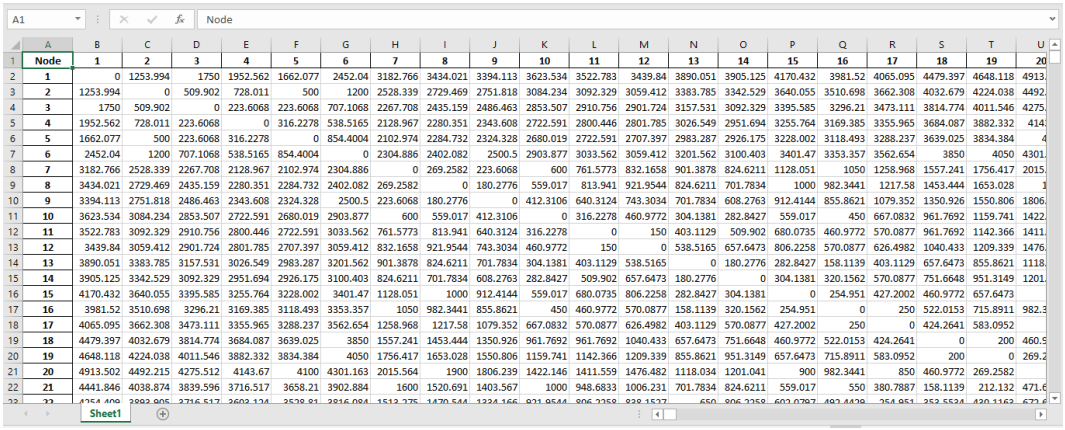


Node	X	Y
1	1150	4000
2	1050	2750
3	1150	2250
4	1250	2050
5	1350	2350
6	1050	1550
7	3350	1700
8	3450	1450
9	3550	1600
10	3950	1700
11	4050	2000
12	4050	2150
13	4250	1650
14	4150	1500
15	4450	1450
16	4400	1700
17	4600	1850
18	4900	1550
19	5100	1550
20	5350	1450
21	4950	1700
22	4850	1900
23	4900	2050

Gambar 1 Cuplikan Dataset TSPLIB pr1002

Lampiran 2 Matriks Jarak

https://docs.google.com/spreadsheets/d/11jxRkhfIlNn73S8mh1i-jCAuwnf4-yp1/edit?usp=sharing&ouid=117294901494638134019&rtpof=true&sd=true



Node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	1253.994	1750	1952.562	1662.077	2452.04	3182.766	3434.021	3394.113	3623.534	3522.783	3439.84	3890.051	3905.125	4170.432	3981.52	4065.095	4479.397	4648.118	4913.
2	1253.994	0	509.902	728.011	500	1200	2528.339	2729.469	2751.818	3084.234	3092.329	3059.412	3383.785	3342.529	3640.055	3510.698	3662.308	4032.679	4224.038	4492.
3	1750	509.902	0	223.6068	223.6068	707.1068	2267.708	2435.159	2486.463	2853.507	2910.756	2901.724	3157.531	3092.329	3395.585	3296.21	3473.111	3814.774	4011.546	4275.
4	1952.562	728.011	223.6068	0	316.2278	538.5165	2128.967	2280.351	2343.608	2722.591	2800.446	2801.785	3026.549	2951.694	3255.764	3169.385	3355.965	3684.087	3882.332	414
5	1662.077	500	223.6068	316.2278	0	854.4004	2102.974	2284.732	2324.328	2680.019	2722.591	2707.397	2983.287	2926.175	3228.002	3118.493	3288.237	3639.025	3834.384	4
6	2452.04	1200	707.1068	538.5165	854.4004	0	2304.886	2402.082	2500.5	2903.877	3033.562	3059.412	3201.562	3100.403	3401.47	3353.357	3562.654	3850	4050	4301.
7	3182.766	2528.339	2267.708	2128.967	2102.974	2304.886	0	269.2582	223.6068	600	761.5773	832.1658	901.3878	824.6211	1128.051	1050	1258.968	1557.241	1756.417	2015.
8	3434.021	2729.469	2435.159	2280.351	2284.732	2402.082	269.2582	0	180.2776	559.017	813.941	921.9544	824.6211	701.7834	1000	982.3441	1217.58	1453.444	1653.028	1
9	3394.113	2751.818	2486.463	2343.608	2324.328	2500.5	223.6068	180.2776	0	412.3106	640.3124	743.3034	701.7834	608.2763	912.4144	855.8621	1079.352	1350.926	1550.806	1806.
10	3623.534	3084.234	2853.507	2722.591	2680.019	2903.877	600	559.017	412.3106	0	316.2278	460.9772	304.1381	282.8427	559.017	450	667.0832	961.7692	1159.741	1422.
11	3522.783	3092.329	2910.756	2800.446	2722.591	3033.562	761.5773	813.941	640.3124	316.2278	0	150	403.1129	509.902	680.0735	460.9772	570.0877	961.7692	1142.366	1411.
12	3439.84	3059.412	2901.724	2801.785	2707.397	3059.412	832.1658	921.9544	743.3034	460.9772	150	0	538.5165	657.6473	806.2258	570.0877	626.4982	1040.433	1209.339	1476.
13	3890.051	3383.785	3157.531	3026.549	2983.287	3201.562	901.3878	824.6211	701.7834	304.1381	403.1129	538.5165	0	180.2776	282.8427	158.1139	403.1129	657.6473	855.8621	1118.
14	3905.125	3342.529	3092.329	2951.694	2926.175	3100.403	824.6211	701.7834	608.2763	282.8427	509.902	657.6473	180.2776	0	304.1381	320.1562	570.0877	751.6648	951.3149	1201.
15	4170.432	3640.055	3395.585	3255.764	3228.002	3401.47	1128.051	1000	912.4144	559.017	680.0735	806.2258	282.8427	304.1381	0	254.951	427.2002	460.9772	657.6473	
16	3981.52	3510.698	3296.21	3169.385	3118.493	3353.357	1050	982.3441	855.8621	450	460.9772	570.0877	158.1139	320.1562	254.951	0	250	522.0153	715.8911	982.3
17	4065.095	3662.308	3473.111	3355.965	3288.237	3562.654	1258.968	1217.58	1079.352	667.0832	570.0877	626.4982	403.1129	570.0877	427.2002	250	0	424.2641	583.0952	
18	4479.397	4032.679	3814.774	3684.087	3639.025	3850	1557.241	1453.444	1350.926	961.7692	961.7692	1040.433	657.6473	751.6648	460.9772	522.0153	424.2641	0	200	460.9
19	4648.118	4224.038	4011.546	3882.332	3834.384	4050	1756.417	1653.028	1550.806	1159.741	1142.366	1209.339	855.8621	951.3149	657.6473	715.8911	583.0952	200	0	269.2
20	4913.502	4492.215	4275.512	4143.67	4100	4301.163	2015.564	1900	1806.239	1422.146	1411.559	1476.482	1118.034	1201.041	900	982.3441	850	460.9772	269.2582	
21	4441.846	4038.874	3839.596	3716.517	3658.21	3902.884	1600	1520.691	1403.567	1000	948.6833	1006.231	701.7834	824.6211	559.017	550	380.7887	158.1139	212.132	471.6

Gambar 2 Cuplikan Matriks Jarak

Lampiran 3 Codingan menggunakan Google Colab

```
# ==========================================
# 1. IMPORT PUSTAKA
# ==========================================

import math
import random
import time
import statistics
from pathlib import Path

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

try:
    from scipy import stats
    SCIPY_TERSEDIA = True
except Exception:
    SCIPY_TERSEDIA = False

# Pengaturan tampilan tabel
pd.set_option("display.max_columns", 50)
pd.set_option("display.width", 120)
pd.set_option("display.max_rows", 20)
pd.options.display.float_format = "{:,.3f}".format


print("Semua pustaka utama berhasil dimuat.")
print(f"Scipy tersedia: {SCIPY_TERSEDIA}")
```

```
# ==========================================
# 2. PENGATURAN UMUM DAN LOKASI FILE
# ==========================================

# Ganti bagian ini jika lokasi file berbeda
nama_file_dataset = "pr1002.xlsx"
sheet_dataset = "pr1002"

# Lokasi keluaran
folder_hasil = Path("hasil_bab4")
folder_hasil.mkdir(exist_ok=True)

print("Nama file dataset :", nama_file_dataset)
print("Nama sheet        :", sheet_dataset)
print("Folder hasil      :", folder_hasil.resolve())
```

Dokumentasi implementasi program dan hasil pengujian lengkap dapat diakses melalui tautan berikut:

https://colab.research.google.com/drive/1nH28Z00HNEfvB9OcXxNEtfRG0gjvn6CE#scrollTo=4_Dlfq3zz0Td

RIWAYAT HIDUP

Badi'atul Khoiroh, biasa dipanggil Badi', lahir di Mojokerto, 19 Maret 2004. Penulis merupakan anak keempat dari empat bersaudara dari pasangan Bapak Suwantah Ahmad Muhsin dan Ibu Shofiyatun. Penulis bertempat tinggal di Kecamatan Jetis, Kabupaten Mojokerto. Penulis menempuh pendidikan awal di RA Roudlotul Muta'allim pada tahun 2009–2010. Selanjutnya, penulis melanjutkan pendidikan di MI Roudlotul Muta'allim pada tahun 2010–2016. Setelah itu, penulis menempuh pendidikan di SMP An-Nur Bululawang pada tahun 2016–2019 dan SMA An-Nur Bululawang pada tahun 2019–2022. Selanjutnya penulis menempuh pendidikan tinggi di Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang pada tahun 2022 melalui jalur SNMPTN. Selama menempuh pendidikan tinggi, penulis pernah berpartisipasi dalam kegiatan kepanitiaan Duta Saintek. Selain itu, penulis memiliki pengalaman mengajar sebagai tutor les privat matematika. Penulis dapat dihubungi melalui e-mail: badiatulkhoiroh6@gmail.com

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

BUKTI KONSULTASI SKRIPSI

Nama : Badi'atul Khoiroh
NIM : 220601110009
Fakultas / Jurusan : Sains dan Teknologi / Matematika
Judul Skripsi : Pengaruh Mekanisme *Multi-Parent* pada *Nearest Neighbor Crossover* dalam Algoritma Genetika untuk Penyelesaian *Traveling Salesman Problem*
Pembimbing I : Mohammad Nafie Jauhari, M.Si.
Pembimbing II : Erna Herawati, M.Pd.

No	Tanggal	Hal	Tanda Tangan
1.	08 Agustus 2025	Konsultasi Topik dan Data	1.
2.	13 Agustus 2025	Konsultasi Bab I, II, dan III	2.
3.	15 September 2025	Konsultasi Bab I, II, dan III	3.
4.	27 Oktober 2025	ACC Bab I, II, dan III	4.
5.	28 Oktober 2025	Konsultasi Kajian Agama Bab I dan II	5.
6.	30 Oktober 2025	Konsultasi Kajian Agama Bab I dan II	6.
7.	13 November 2025	ACC Kajian Agama Bab I dan II	7.
8.	25 November 2025	ACC Seminar Proposal	8.
9.	2 Maret 2026	Konsultasi Revisi Seminar Proposal	9.
10.	13 April 2026	Konsultasi Bab IV dan V	10.
11.	28 April 2026	ACC Bab IV dan V	11.
12.	29 April 2026	Konsultasi Kajian Agama Bab IV	12.
13.	30 April 2026	Konsultasi Kajian Agama Bab IV	13.
14.	05 Mei 2026	ACC Kajian Agama Bab IV	14.
15.	13 Mei 2026	ACC Seminar Hasil	15.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No.50 Dinoyo Malang Telp. / Fax. (0341)558933

16.	03 Juni 2026	Konsultasi Revisi Seminar Hasil	16.
17.	08 Juni 2026	ACC Sidang Skripsi	17.
18.	17 Juni 2026	ACC Keseluruhan	18.

Malang, 17 Juni 2026
Mengetahui,
Ketua Program Studi Matematika

Dr. Fachrur Rozi, M.Si.
NIP. 19800527 200801 1 012