OPTIMASI *CAPACITATED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS* MENGGUNAKAN ALGORITMA *ADAPTIVE LARGE NEIGHBORHOOD SEARCH* (ALNS) UNTUK MINIMASI JARAK

(Studi Kasus: PT. Pos Logistik Indonesia)

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Strata-1
Program Studi Teknik Industri - Fakultas Teknologi Industri
Universitas Islam Indonesia

Nama : Rifki Nurul Mukarim
No. Mahasiswa : 21522176

PROGRAM STUDI TEKNIK INDUSTRI PROGRAM SARJANA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2025

PERNYATAAN KEASLIAN

Saya mengakui bahwa tugas akhir ini adalah hasil karya saya sendiri kecuali kutipan dan ringkasan yang seluruhnya sudah saya jelaskan sumbernya. Jika dikemudian hari ternyata terbukti pengakuan saya ini tidak benar dan melanggar peraturan yang sah maka saya bersedia ijazah yang telah saya terima ditarik kembali oleh Universitas Islam Indonesia.

Yogyakarta, 14 Mei 2025



Rifki Nurul Mukarim

21522176

SURAT BUKTI PENELITIAN

UNIVERSITAS ISLAM INDONESIA

FAKULTAS
TEKNOLOGI INDUSTRI

Gedung KH. Mas Mansur
Kampus Terpadu Universitas Islam Indonesia
Jl. Kaliurang km 14,5 Yogyakarta 55584
T. (0274) 898444 ext 4110, 4100
F. (0274) 895007
E. fti@uii.ac.id
W. fti.uii.ac.id

SURAT KETERANGAN PENELITIAN

Nomor : 99/Ka.lab SIMANTI/20/ Lab.SIMANTI/IV/2025

Assalamu'alaikum Warohmatullahi Wabarokaatuh

Dengan hormat,

Yang bertanda tangan dibawah ini, menerangkan bahwa:

Nama : Rifki Nurul Mukarim

Nim : 21522176

Jurusan : Teknik Industri

Dosen Pembimbing : Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM

Menyatakan bahwa mahasiswa tersebut diatas telah melaksanakan penelitian tugas akhir dengan judul "**OPTIMASI *CAPACITATED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS* MENGGUNAKAN ALGORITMA ALNS UNTUK MINIMASI JARAK (Studi Kasus: PT. Pos Logistik Indonesia) ".** Mulai pelaksanaan penelitian 03 Desember 2024 sampai 14 April 2025.

Demikian surat keterangan penelitian ini kami buat. Atas perhatiannya dan kerja samanya kami mengucapkan terima kasih.

Wassalamu'alaikum Warohmatullahi Wabarokaatuh

Yogyakarta, 25 April 2025

Kepala Laboratorium

Sistem Manufaktur Terintegrasi



Putri Dwi Annisa, S.T., M.Sc.

LEMBAR PENGESAHAN PEMBIMBING

OPTIMASI *CAPACITATED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS* MENGGUNAKAN ALGORITMA *ADAPTIVE LARGE NEIGHBORHOOD SEARCH* (ALNS) UNTUK MINIMASI JARAK
(Studi Kasus: PT. Pos Logistik Indonesia)

TUGAS AKHIR

Disusun Oleh :

Nama : Rifki Nurul Mukarim
No. Mahasiswa 21522176

Yogyakarta, 14 Mei 2025

Dosen Pembimbing

(Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM)

LEMBAR PENGESAHAN DOSEN PENGUJI

OPTIMASI *CAPACITATED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS* MENGGUNAKAN ALGORITMA *ADAPTIVE LARGE NEIGHBORHOOD SEARCH* (ALNS) UNTUK MINIMASI JARAK
(Studi Kasus: PT. Pos Logistik Indonesia)

TUGAS AKHIR

Disusun Oleh :
Nama : Rifki Nurul Mukarim
No. Mahasiswa : 21522176

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata-1 Teknik Industri Fakultas Tekonologi Industri Universitas Islam Indonesia

Yogyakarta, 14 Mei 2025

Tim Penguji

Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM	
Ketua	
Chancard Basumerda, S.T., M.Sc	
Anggota I	
Bambang Suratno, S.T., M.T., Ph.D	
Anggota II	

Mengetahui,
Ketua Program Studi Teknik Industri Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., IPM
NIK. 015220101

HALAMAN PERSEMBAHAN

Alhamdulillahhirabal allamin, Tugas akhir ini merupakan salah satu bentuk perjuangan awal yang membuka ke perjalanan berikutnya sebagai bukti perjuangan penulis dalam menyelesaikan program sarjana Teknik Industri Universitas Islam Indonesia. Penulisan Tugas Akhir ini saya persembahkan kepada saya sendiri yang telah berjuang melalui berbagai hambatan yang ada hingga dapat menyelesaikan Tugas Akhir ini dengan maksimal. Selain itu, saya persembahkan kepada kedua orang tua tercinta saya Bapak Sutardi dan Ibu Yati Nuryati, serta saudara kandung Duwi Agung Wibowo, S.Kom, Tri Azi Muharom, S.E, Naufal Ammar Hibatulloh yang senantiasa memberikan cinta kasih yang telah diberikan selama penulis kecil hingga saat ini serta doa yang selalu dipanjatkan. Semoga karya ini dapat menjadi salah satu hal yang dapat dibanggakan dari seseorang yang masih banyak belajar dan akan terus belajar.

MOTTO

"Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya"

(Q.S Al Baqarah: 286)

"*Indeed, with every difficulty, comes relief.*"

(QS. Al-Insirah: 5)

"Try not to become a man of success, but rather become a man of value"

(Albert Einstein)

"Selalu bangkit dan jangan pernah takut untuk terus mencoba hal baru, setiap halangan akan membuat lebih kuat dari sebelumnya."

KATA PENGANTAR

Assalamu'alaikum Warrahmatullahi Wabarakatuh.

Alhamdulillahirabbil'alamin, puji dan syukur penulis panjatkan ke hadirat Allah SWT atas limpahan rahmat, taufik, serta hidayahnya. Shalawat dan salam tercurah kepada Nabi Muhammad SAW beserta keluarga dan para sahabat, serta orang-orang yang bertaqwa, sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "Optimasi *Capacitated Vehicle Routing Problem with Time Windows* (CVRPTW) Menggunakan Algoritma *Adaptive Large Neighborhood Search* (ALNS) untuk Minimasi Jarak".

Tugas Akhir ini disusun sebagai salah satu syarat yang harus dipenuhi oleh setiap mahasiswa yang mengampu pendidikan di Prgram Studi Teknik Industri untuk menyelesaikan studi Strata-1 pada Fakultas Teknologi Industri, Universitas Islam Indonesia. Dalam penyusunan laporan Tugas Akhir ini penulis mendapatkan banyak bantuan dan dukungan dari berbagai pihak, untuk itu penulis menyampaikan ucapan terimakasih dan penghargaan yang setinggi-tingginya kepada pihak-pihak yang telah memberikan dukungannya baik secara langsung maupun tidak langsung, dengan penuh rasa syukur penulis ucapkan terima kasih kepada:

1. Bapak Prof. Dr. Ir. Hari Purnomo, MT. IPU. ASEAN. Eng selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
2. Bapak Dr. Drs. Imam Djati Widodo, M.Eng.Sc selaku Ketua Jurusan Teknik Industri Fakultas Teknologi Industri Universitas Islam Indonesia.
3. Bapak, Ir. M. Ridwan Andi Purnomo, ST., M.Sc., Ph.D., IPM. selaku Ketua Program Studi Teknik Industri Program Sarjana Universitas Islam Indonesia.
4. Bapak Ir. M. Ridwan Andi Purnomo, ST., M.Sc., Ph.D., IPM.selaku Dosen Pembimbing yang telah meluangkan waktu juga pikirannya serta memberikan motivasi, semangat dan dukungan.
5. Bapak Wahyu Kurniawan, S.T., M.T., CISCP dan Ibu Putri Dwi Annisa, S.T., M.Sc yang telah memberikan bimbingan serta wadah untuk peneliti sehingga dapat melaksanakan penelitian tugas akhir di PT. Pos Logistik Indonesia dengan baik.
6. Kedua orang tua tercinta, Ibu Yati Nuryati dan Bp Sutardi yang telah memberikan dukungan baik secara moril maupun material, motivasi serta doa yang tidak pernah terputus sehingga penulis dapat menyelesaikan tugas akhir dengan baik.
7. Saudara kandung Duwi Agung Wibowo, Tri Azi Muharom, Naufal Ammar Hibatulloh, yang telah memberikan bantuan, doa, semangat sehingga penulis dapat menyelesaikan tugas akhir dengan baik.
8. Sahabat sahabat sejak awal perkuliahan, Sastyaviani Irami Putri, Vira Dwi Apryola, Moch. Adib Heramunida, dan Nabila Lintang Kirana yang telah membersamai perjalanan di masa perkuliahan, bertukar canda dan tawa hingga mengukir memori di setiap titik perjalanan yang dilalui penulis serta memberikan motivasi selama pengerjaan tugas akhir penulis.
9. Raras Maheswari Putri Wardhana yang telah menemani perjalanan penulisan Tugas Akhir ini dari awal hingga akhir, memberikan motivasi dan membangkitkan semangat penulis kembali setiap menemui hambatan dalam penulisan Tugas Akhir ini hingga dapat selesai dengan baik.
10. Asisten Laboratorium Sistem Manufaktur 2021, yang telah berjuang bersama sama, saling menguatkan, memberikan dukungan dan doa kepada penulis sehingga dapat menyelesaikan tugas akhir dengan baik.

11. Bapak Heri Susilo serta Asisten Laboratorium Sistem Manufaktur 2020 dan 2022, yang memberikan semangat dan dukungan kepada penulis selama pengerjaan Tugas Akhir ini.
12. Sahabat lainnya, Ngang Ngong Ngeng, Fighter, Tunggu Sukses, Triangel, yang telah memberikan dukungan kepada peneliti sehingga memberikan semangat dalam pengerjaan tugas akhir ini.
13. Mas Demas Emirbuwono Basuki sebagai kakak tingkat yang selalu memberikan dorongan serta memotivasi penulis dan juga memberikan berbagai pengalaman kerja yang sangat berharga dalam menyelesaikan pendidikan sarjana dan mempersiapkan diri di dunia kerja.
14. Segenap keluarga besar teman teman jurusan Teknik Industri Angkatan 2021 yang telah memberikan motivasi dan dukungannya.

Penulis menyadari bahwa tugas akhir ini masih belum sempurna, sehingga penulis menyampaikan permohonan maaf serta mengharapkan kritik dan saran dari semua pembaca guna penyempurnaan dimasa mendatang. Semoga tugas akhir ini dapat bermanfaat bagi semua pihak.

Wassalamu'alaikum Warrahmatullahi Wabarakatuh.

Yogyakarta, 24 April 2025



Rifki Nurul Mukarim
21522176

ABSTRAK

Persaingan industri logistik khususnya perusahaan 3PL, optimasi distribusi dan rute pengiriman menjadi salah satu komponen penting untuk meningkatkan daya saing perusahaan. Oleh karena itu, perusahaan penting mengoptimalkan rute pengiriman produk agar dapat menurunkan biaya pengiriman beriringan dengan meminimalkan jarak pengiriman. PT. Pos Logistik Indonesia merupakan salah satu perusahaan jasa yang bergerak di bidang *Freight Forwarding.* PT. Pos Logistik Indonesia turut menghadapi tantangan terhadap penentuan rute distribusi produk untuk meningkatkan efisiensi dengan melakukan minimasi jarak dalam setiap rute distribusi. Dampak dari penentuan rute yang tidak optimaldirasakan oleh PT. Pos Logistik Indonesia yang berdampak langsung pada jarak tempuh armada yang tidak efisien. Akibatnya, terdapat kendaraan yang menempuh perjalanan yang lebih jauh dari seharusnya, sehingga seringkali kembali ke *main hub* di luar jadwal yang telah ditentukan. Algoritma *Adaptive Large Neighborhood Search* (ALNS) adalah algoritma metaheuristic dengan prinsip kerja meningkatkan solusi awal yang diberikan dengan menjelajahi banyak *large neighborhoods*. Hal ini dapat dilakukan dengan penerapan berbagai operator *destroy* dan *repair*. Penelitian ini bertujuan untuk menentukan rute distribusi yang paling optimal produk di area Jakarta dengan mempertimbangkan *time windows*, dan kapasitas kendaraan sehingga dapat meminimasi jarak tempuh yang dilalui oleh kendaraan. Hasil penelitian menunjukan ALNS membentuk rute usulan dengan 3 rute yang terbentuk menggunakan kendaraan jenis BlindVan. Rute awal yang digunakan oleh PT. Pos Logistik Indonesia memiliki total jarak sejauh 647,4 km, sedangkan rute hasil optimasi ALNS hanya menempuh 276,92 km. Hal tersebut menunjukan terdapat pengurangan jarak sebesar 370,48 km atau sekitar 57,22%.

Kata Kunci: *Adaptive Large Neighborhood Search*, Minimasi Jarak, Optimasi, Rute Distribusi, *Vehicle Routing Problem*

DAFTAR ISI

PERNYATAAN KEASLIAN ..... **ii**
SURAT BUKTI PENELITIAN ..... **iii**
LEMBAR PENGESAHAN PEMBIMBING ..... **iv**
LEMBAR PENGESAHAN DOSEN PENGUJI ..... **v**
HALAMAN PERSEMBAHAN ..... **vi**
MOTTO ..... **vii**
KATA PENGANTAR ..... **viii**
ABSTRAK ..... **x**
DAFTAR ISI ..... **xi**
DAFTAR TABEL ..... **xii**
DAFTAR GAMBAR ..... **xiii**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 4
1.3 Tujuan Penelitian 4
1.4 Manfaat Penelitian 4
1.5 Batasan Penelitian 5
1.6 Sistematika Penulisan 5
BAB II TINJAUAN PUSTAKA **7**
2.1 *Supply Chain Management* 12
2.2 Biaya Operasional 17
2.3 *Vehicle Routing Problem* 18
2.4 *Capacitated Vehicle Routing Problem with Time Windows* (CVRPTW) 21
2.5 Algoritma *Adaptive Large Neighborhood Search* (ALNS) 23
BAB III METODE PENELITIAN **29**
3.1 Objek Penelitian 29
3.2 Instrumen Penelitian 30
3.3 Sumber Data Primer dan Sekunder 30
3.4 Metode Pengumpulan Data 30
3.5 Populasi dan Sampel 31
3.6 Alur Penelitian 31
BAB IV PENGUMPULAN DAN PENGOLAHAN DATA **38**
4.1 Pengumpulan Data 38
4.2 Pengolahan Data 50
BAB V PEMBAHASAN **63**
5.1 Analisis Hasil 63
5.2 Analisis Biaya Distribusi 76
5.3 Perbandingan ALNS dengan Kebijakan Perusahaan 79
BAB VI PENUTUP **81**
6.1 Kesimpulan 81
6.2 Saran 82
DAFTAR PUSTAKA **84**
LAMPIRAN **A-1**

DAFTAR TABEL

Tabel 2. 1 Posisi Penelitian 12
Tabel 2. 2 Jenis Jenis *Destroy Operator* 25
Tabel 2. 3 Jenis Jenis *Repair Operator* 26
Tabel 2. 4 *Acceptance Criterion Method* 28
Tabel 4. 1 Titik Koordinat UTM 43
Tabel 4. 2 Tipe *Time windows* 47
Tabel 4. 3 *Time windows* dan *Demand* 47
Tabel 4. 4 Data Pendukung 50
Tabel 4. 5 Rute Awal (Kebijakan Perusahaan) 50
Tabel 4. 6 Matriks Jarak 52
Tabel 4. 7 *Pseudocode* Inisialisasi Data 53
Tabel 4. 8 *Pseudocode* Penentuan Parameter 55
Tabel 4. 9 *Pseudocode* Total Jarak, Kapasitas, dan Waktu 56
Tabel 4. 10 *Pseudocode Feasibility Check* 57
Tabel 4. 11 *Pseudocode* Penentuan Solusi Awal 58
Tabel 4. 12 *Pseudocode Destroy repair* 58
Tabel 4. 13 *Pseudocode* Fungsi Evaluasi 60
Tabel 4. 14 *Pseudocode* Running Algoritma 61
Tabel 4. 15 Hasil Rute Usulan 62
Tabel 5. 1 Rute Usulan 1 63
Tabel 5. 2 Detail Waktu Rute 1 66
Tabel 5. 3 Rute Usulan 2 68
Tabel 5. 4 Detail Waktu Rute 2 70
Tabel 5. 5 Rute Usulan 3 72
Tabel 5. 6 Detail Waktu Rute 3 74
Tabel 5. 7 Biaya Sewa Kendaraan 76
Tabel 5. 8 Biaya Tenaga Kerja 76
Tabel 5. 9 Biaya Bahan Bakar 77
Tabel 5. 10 Total Biaya Rute Awal 77
Tabel 5. 11 Total Biaya Rute Usulan 78

DAFTAR GAMBAR

Gambar 2. 1 *Supply Chain Management* 13
Gambar 2. 2 *Level of Logistics Management* 15
Gambar 2. 3 Kerangka Permasalahan VRP 20
Gambar 2. 4 Skema Algoritma ALNS 24
Gambar 2. 5 *Framework* Algoritma ALNS 24
Gambar 3. 1 Logo dan Lokasi PT. Pos Logistik Indonesia 29
Gambar 3. 2 Alur Penelitian 32
Gambar 4. 1 Nilai Budaya Perusahaan 38
Gambar 4. 2 *Core value* 39
Gambar 4. 3 Logo PT. Pos Logistik Indonesia 39
Gambar 5. 1 Rute Usulan Kendaraan 1 63
Gambar 5. 2 Rute Usulan Kendaraan 2 68
Gambar 5. 3 Rute Usulan Kendaraan 3 72
Gambar 5. 4 Perbandingan Kebijakan Perusahaan dengan ALNS 79

BAB I
PENDAHULUAN

1.1 Latar Belakang

Supply chain atau rantai pasok merupakan serangkaian proses yang berkolaborasi untuk mengelola aliran barang, informasi, dan keuangan dari hulu ke hilir hingga suatu produk sampai ke konsumen akhir (Fadhlullah, et al., 2018). Proses ini meliputi aktivitas dalam suatu perusahaan mulai dari proses pengadaan bahan baku, produksi, hingga distribusi produk jadi ke *customer* akhir. Efisiensi pengelolahan rantai pasok menjadi sangat penting untuk meningkatkan efektivitas kegiatan operasional, pengurangan biaya, dan peningkatan kepuasan *customer* terhadap kinerja perusahaan (Octaviany & Gunawan, 2023). Salah satu kegiatan penting dalam rantai pasok adalah logistic yang bertanggung jawab untuk mengelola aliran fisik barang dari satu titik ke titik lainnya. Logistik memainkan peranan penting dalam kegiatan transportasi, pergudangan, serta pengelolaan inventaris sehingga dapat mendukung efisiensi operasional di seluruh rantai pasok (Permana, et al., 2024).

Logistik merupakan suatu kegiatan dalam rantai pasok (*supply chain*) yang dilakukan untuk memindahkan atau menempatkan produk pada suatu tempat (Goni, et al., 2022). Berdasarkan data *Compound Annual Growth Rate* (CAGR) menunjukan laju industri logistic pada tahun 2013 – 2017 mengalami pertumbuhan sebesar 11,7%. Laju pertumbuhan tersebut di dukung dengan seiring bertumbuhnya *freight forwarding* yang mencapai 11.8% dan *express & small parcell* mencapai 21.7% (Pramudita, 2020). Pertumbuhan ini me nunjukan peningkatan kebutuhan *customer* terhadap layanan logistik yang cepat, andal, dan terjangkau. Hal tersebut menjadi tantangan perusahan penyedia jasa logistic pihak ketiga atau *third-party logistics* (3PL) menjadi mitra strategis dalam rantai pasok yang menyediakan layanan mulai dari manajemen pergudangan, pengankutan, dan distribusi untuk perusahaan manufaktur dan ritel (Adhana, et al., 2023). Berdasarkan *Allied Market Research Report market size* pasar logistik pihak ketiga (3PL) global tercatat sebesar $1,3 triliun pada tahun 2021 dan diproyeksikan akan mencapai $2,8 triliun pada tahun 2031, dengan tingkat pertumbuhan tahunan rata-rata (CAGR) sebesar 8,8% selama periode 2022 hingga 2031 (Khisty & Katare, 2023).

Namun, seiring dengan pesatnya perkembangan ini, muncul beberapa tantangan operasional yang harus dihadapi perusahaan penyedia layanan logistik, terutama terkait dengan efisiensi biaya, kecepatan pengiriman, dan pengurangan dampak lingkungan. Dalam persaingan

industri logistik khususnya perusahaan 3PL, optimasi distribusi dan rute pengiriman menjadi salah satu komponen penting untuk meningkatkan daya saing perusahaan (Saogo, et al., 2024). Hal tersebut, menjadi penting karena dapat menyebabkan peningkatan biaya bahan bakar, waktu pengiriman yang lebih lama (Pramestiana & Anwar, 2025). Oleh karena itu, perusahaan penting mengoptimalkan rute pengiriman produk agar dapat menurunkan biaya pengiriman beriringan dengan meminimalkan jarak pengiriman. Optimasi *Vehicle Routing Problem* (VRP) menjadi solusi yang sering diterapkan oleh 3PL untuk mengatasi permasalahan tersebut sehingga dapat meningkatkan kinerja operasional perusahaan seiring tuntutan permintaan pelanggan (Paillin, et al., 2019). Salah satu perusahaan 3PL di Indonesia yang menghadapi tantangan tersebut yaitu PT. Pos Logistik Indonesia.

PT. Pos Logistik Indonesia merupakan salah satu perusahaan jasa yang bergerak di bidang *Freight Forwarding.* PT. Pos Logistik Indonesia merupakan anak perusahaan PT Pos Indonesia (Persero) yang merupakan BUMN yang bergerak dalam bidang penyediaan layanan logistik dan distribusi di Indonesia (Hilman & Abdillah, 2021). PT. Pos Logistik Indonesia turut menghadapi tantangan terhadap penentuan rute distribusi produk untuk meningkatkan efisiensi dan pengurangan jarak tempuh beriringan dengan penurunan biaya distribusi dengan melakukan minimasi jarak dalam setiap rute distribusi. Metode penentuan rute yang masih bersifat konvensional mengakibatkan jarak tempuh pada setiap kendaraan kurang optimal, peningkatan biaya operasional, dan waktu pengiriman yang kurang optimal, Selain itu, perusahaan dihadapkan dengan kendala seperti *time windows* dan kapasitas kendaraan yang mana membatasi jam pengiriman barang kepada pelanggan sehingga mengharuskan barang tiba dalam jangka waktu yang telah ditetapkan dan membatasi jumlah barang yang dapat diangkut dari setiap kendaraan.

Berdasarkan hasil wawancara yang dilakukan, penentuan rute distribusi produk di PT. Pos Logistik Indonesia berdasar pada evaluasi tahunan dan juga subjektifitas driver dari masing masing kendaraan. Pendekatan tersebut tidak hanya bersifat reaktif, tetapi juga kurang relevan untuk pengambilan keputusan di tingkat operasional harian yang dinamis. Dampak dari penentuan rute yang tidak optimal ini dirasakan oleh PT. Pos Logistik Indonesia yang berdampak langsung pada jarak tempuh armada yang tidak efisien. Akibatnya, terdapat kendaraan yang menempuh perjalanan yang lebih jauh dari seharusnya, sehingga seringkali kembali ke *main hub* di luar jadwal yang telah ditentukan. Kondisi tersebut juga tidak hanya meningkatkan konsumsi bahan bakar pada setiap kendaraan, namun juga supir atau tenaga kerja

harus bekerja lebih lama yang menyebabkan kelelahan dan penurunan produktivitas. Oleh karena itu, penurunan atau minimasi jarak dari setiap rute yang dilalui kendaraan menjadi hal yang sangat penting agar daya saing perusahaan tidak menurun. Salah satu cara untuk mengatasi permasalahan tersebut yaitu dengan mengoptimalkan rute pengiriman barang pada setiap pelanggan. Hal tersebut dapat dilakukan dengan menyelesaikan permasalahan *Vehicle Routing Problem* (VRP) yang dihadapi PT. Pos Logistik Indonesia menggunakan metode *metaheuristic*. Hal tersebut memungkinkan perusahaan untuk menentukan rute pengiriman paling optimal bagi setiap armada, sehingga efisiensi operasional dapat tercapai, biaya dapat ditekan, dan kualitas layanan tetap terjaga secara berkelanjutan.

Salah satu metode *metaheuristik* yang dapat memecahkan permasalahan VRP adalah dengan menggunakan algoritma ALNS. Algoritma *Adaptive Large Neighborhood Search* (ALNS) adalah algoritma metaheuristik yang diusulkan oleh Ropke dan Pisinger pada tahun 2006, sebagai perpanjangan dari *Large Neighborhood Search* (LNS) yang dicetuskan oleh Shaw (1998). ALNS merupakan sebuah pendekatan adaptif yang digunakan untuk memperbaiki solusi yang ada. Prinsip kerja dasar dari ALNS yaitu meningkatkan solusi awal yang diberikan dengan menjelajahi banyak *large neighborhoods*. Hal ini dapat dilakukan dengan penerapan berbagai operator *destroy* dan *repair* (Mehdi, et al., 2021). Menurut (Pugliese, et al., 2024) dalam penelitiannya menyebutkan bahwa ALNS memiliki waktu komputasi yang menunjukan kinerja baik dalam hal efektivitas, serta memiliki solusi optimal. Hal tersebut sejalan dengan penelitian yang dilakukan oleh (Gunawan, et al., 2020), menunjukan hasil eksperimen menunjukkan bahwa ALNS mampu meningkatkan solusi sebesar 80 dari 90 solusi terbaik yang diketahui dan memberikan hasil yang sama pada 10 instances lainnya dalam waktu komputasi yang singkat. Metode metaheuristik dapat memberikan solusi mendekati optimal dengan waktu penyelesaian yang singkat (Gupta, et al., 2010). Oleh karena itu, penyelesaian masalah penentuan rute yang kompleks dapat diselesaikan dengan metode metaheuristik salah satunya menggunakan metode algoritma ALNS.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk menentukan rute pengantaran barang yang paling optimal di PT. Pos Logistik Indonesia dengan menggunakan metode *Adaptive Large Neighborhood Search* (ALNS) dengan mempertimbangkan batasan berupa *time windows* pelanggan serta kapasitas kendaraan sehingga termasuk dalam kategori *Capacitated Vehicle Routing Problem with Time windows* (CVRPTW). Hasil rute usulan yang

didapatkan nantinya akan dilihat total biaya yang dikeluarkan lalu dibandingkan dengan total biaya yang dikeluarkan perusahaan saat ini.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah di jelaskan diatas, maka rumusan masalah pada penelitian ini sebagai berikut:

1. Bagaimana rute distribusi produk yang optimal pada setiap kendaraan di PT. Pos Logistik Indonesia?
2. Berapa persen penurunan jarak yang dihasilkan berdasarkan dengan rute usulan yang diberikan pada proses distribusi produk di PT. Pos Logistik Indonesia?
3. Bagaimana dampak penurunan jarak tempuh pada rute usulan terhadap total biaya distribusi pada PT. Pos Logistik Indonesia?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah dirancang, berikut merupakan tujuan penelitian yang dilakukan:

1. Menentukan rute distribusi yang paling optimal dengan meminimalkan jarak tempuh pada setiap rute yang dilalui kendaraan dalam proses distribusi produk di area Jakarta dengan mempertimbangkan *time windows*, dan kapasitas kendaraan.
2. Mengetahui penurunan jarak tempuh pada setiap rute yang terbentuk pada rute usulan dibandingkan dengan rute awal.
3. Mengukur dampak penurunan jarak berdasarkan rute usulan terhadap total biaya distribusi pada setiap kendaraan melalui penerapan algoritma ALNS di PT. Pos Logistik Indonesia.

1.4 Manfaat Penelitian

Berikut merupakan manfaat yang diperoleh dari penelitian yang telah dilakukan:

1. Bagi Perusahaan

   Hasil penelitian ini diharapkan mampu memberikan saran dalam penentuan rute distribusi produk yang dilakukan PT. Pos Logistik Indonesia sehingga dapat meminimsi jarak tempuh pada setiap rute yang dilalui kendaraan. Selain itu, metode yang digunakan dapat dijadikan sebagai alat untuk pengambilan keputusan dalam menentukan rute yang optimal dalam proses distribusi produk.
2. Bagi Akademisi

Hasil penelitian ini diharapkan mempu mempu memberikan peluang kerjasama antara program studi Teknik Industri dan PT. Pos Logistik Indonesia kedepannya serta dapat menjadi media yang digunakan untuk melatih kemampuan mahasiswa dalam melakukan penelitian di industri logistik khususnya pada perusahaan ekspedisi.

1.5 Batasan Penelitian

Berikut merupakan batasan penelitian yang digunakan untuk membatasi ruang lingkup penelitian yang dilakukan:

1. Data penelitian yang digunakan salah satunya yaitu titik atau koordinat pengiriman yang berada di wilayah Jakarta dengan satu depot distribusi (*main hub*).
2. Penelitian yang dilakukan hanya mempertimbangkan beberapa constraint yaitu kapasitas kendaraan, dan *time windows*.
3. Metode penelitian yang digunakan adalah menggunakan algoritma *Adaptive Large Neighborhood Search* (ALNS).
4. Penelitian yang dilakukan yang menggunakan 1 jenis kendaraan yaitu BlindVan.
5. Penelitian ini berfokus pada minimasi jarak pada setiap rute kendaraan dengan memenuhi beberapa *constraint* yang telah ditentukan.
6. *Delivery demand* yang digunakan dalam satuan kilogram (kg), tidak mempertimbangkan ukuran produk (panjang, lebar, tinggi).
7. Penelitian ini tidak mencangkup kondisi lalu lintas, waktu dianggap linier terhadap jarak dan tidak terjadi kemacetan.

1.6 Sistematika Penulisan

Dalam penelitian ini terdiri dari beberapa bab yang nantinya sebagai acuan supaya sistematis dalam penulisan penelitian ini, adapun sistematika dalam penulisan penelitian ini sebagai berikut:

BAB I PENDAHULUAN

Pada bab I berisi tentang latar belakang, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan penelitian ini.

BAB II KAJIAN LITERATUR

Bagian ini berisi tentang kajian induktif dan kajian deduktif yang diambil dari jurnal, buku, dan literatur lainnya yang mendukung dalam penyelesaian penelitian ini

BAB III METODE PENELITIAN

Berisi tentang uraian kerangka penelitian, metode yang digunakan, aplikasi, bahan baku, material, alat, tata cara penelitian, dan data yang digunakan.

BAB IV PENGAMBILAN DAN PENGOLAHAN DATA

Dalam bagian ini akan menjelaskan pengumpulan data dengan prosedur yang sudah ditentukan, kemudian diolah sesuai dengan metode yang digunakan dan disertakan gambar dan grafik yang diperoleh dari hasil penelitian

BAB V PEMBAHASAN

Berisi pembahasan kritis mengenai hasil bab sebelumnya dan belum dipaparkan di bab sebelumnya.

BAB VI KESIMPULAN DAN SARAN

Kesimpulan berisi pernyataan singkat yang ditulis dengan menggunakan urutan angka untuk menjabarkan hasil penelitian yang dilakukan. Kesimpulan harus menjawab rumusan permasalahan dan membuktikan hipotesis yang ada. Saran berisi beberapa rekomendasi pengembangan penelitian lanjutan dengan menggunakan cara, alat ataupun metode lain dengan tujuan untuk memperluas pengembangan ilmu. Selain itu, bagian ini juga berisi saran yang diperlukan jika penelitian lanjutan akan dikembangkan berdasarkan keterbatasan/hambatan yang ditemukan selama penelitian dilakukan. Saran dapat dihasilkan dari pembahasan yang telah dilakukan di bab sebelumnya.

DAFTAR PUSTAKA

Bagian ini memuat sumber literatur yang menjadi rujukan dari penelitian ini, baik berupa jurnal, buku, maupun literasi lainnya yang menunjang.

LAMPIRAN

Berisi lampiran daftar tabel, gambar, atau bagian lainnya yang mendukung penelitian.

BAB II
TINJAUAN PUSTAKA

Pada bagian ini menjelaskan hasil kajian pustaka yang mencakup dua pendekatan, yaitu kajian induktif dan deduktif. Kajian induktif mengacu pada analisis yang bersumber dari paper, artikel, dan publikasi sejenis yang membahas penelitian sebelumnya dengan topik atau tema yang relevan dengan penelitian yang akan dilakukan. Kajian ini bertujuan untuk menjadi acuan dalam mengembangkan dan menyempurnakan penelitian yang telah ada. Sementara itu, kajian deduktif berisi landasan teori yang diambil dari buku atau artikel ilmiah lainnya yang mendukung ilmu pengetahuan yang relevan dengan penelitian ini. Kajian deduktif menjadi dasar untuk memahami teori-teori yang mendasari dan memperkuat penelitian yang akan dilakukan. Selain itu, pada bagian ini juga dijelaskan *state of the art* dari penelitian serta perbedaannya dengan penelitian-penelitian terdahulu.

Penelitian sebelumnya yang dilakukan oleh Patmawati & Nugroho (2022), menggunakan metode Algoritma Genetika, yang merupakan salah satu metode perhitungan metaheuristik. Metode ini terinspirasi oleh proses evolusi biologis makhluk hidup dan digunakan untuk menyelesaikan masalah *Capacitated Vehicle Routing Problem* (CVRP) dengan tujuan mengoptimalkan rute distribusi matras. Algoritma Genetik dipilih karena kemampuannya dalam mencari solusi optimal dengan efisiensi waktu dan jarak yang baik. Tujuan dari penelitian ini adalah untuk memaksimalkan kapasitas pengiriman dalam satu kali perjalanan, sehingga distribusi matras dapat dilakukan lebih efisien dibandingkan dengan metode sebelumnya yang menggunakan enam kendaraan dengan kapasitas kurang dari 140 matras. Hasil penelitian menunjukkan adanya pengurangan jarak tempuh sebesar 1,65% atau 39,43 km, yang menunjukkan peningkatan efisiensi dalam distribusi.

Penelitian yang dilakukan oleh Firmansyah, et. al (2021) bertujuan untuk menyelesaikan masalah *Capacitated Vehicle Routing Problem* (CVRP) dalam konteks distribusi es krim di Kota Bandung, dengan fokus pada pengoptimalan rute kendaraan untuk meminimalkan total jarak tempuh sambil mematuhi batasan kapasitas kendaraan. Metode yang digunakan adalah algoritma GASA, yang merupakan kombinasi dari Algoritma Genetika (GA) dan *Simulated Annealing* (SA). Parameter-parameter yang digunakan dalam algoritma GASA meliputi temperatur, penurunan temperatur, kriteria penghentian, banyak generasi, *population rate, crossover rate,* dan *mutation rate.* Hasil penelitian menunjukkan bahwa pendekatan

metaheuristik seperti GASA dapat menjadi solusi efektif untuk masalah CVRP yang termasuk dalam kategori *NP-Hard*, di mana usaha komputasi meningkat seiring dengan kompleksitas masalah.

Penelitian sebelumnya yang dilakukan oleh Anujuprana (2015) bertujuan untuk mengoptimalkan masalah perutean kendaraan di Frisian Flag Indonesia dengan mempertimbangkan *time windows,* yang merupakan tantangan umum dalam proses distribusi hasil produksi berupa susu. Metode yang digunakan dalam penelitian ini adalah Algoritma Genetika, yang dikenal efektif dalam menyelesaikan masalah optimasi kompleks seperti perutean kendaraan. Hasil dari penelitian ini menunjukkan bahwa penerapan Algoritma Genetika dapat memberikan solusi yang efisien dalam mengatur rute kendaraan, dengan total jarak yang bervariasi tergantung pada rute yang diambil, seperti yang ditunjukkan dalam beberapa contoh rute dengan jarak total antara 55.7 km hingga 95 km.

Penelitian yang dilakukan oleh Prawira & Santosa (2021) bertujuan untuk menyelesaikan masalah *Vehicle Routing Problem with Drone* (VRPD) dengan mengembangkan algoritma *Particle Swarm Optimization* (PSO) dan *Simulated Annealing* (SA). Metode yang digunakan adalah pendekatan metaheuristik, di mana PSO meniru perilaku segerombolan organisme seperti ikan atau burung, sementara SA meniru proses annealing dalam metalurgi. Algoritma PSO dan SA dikombinasikan dengan algoritma *Route Drone* untuk mengubah vektor posisi menjadi rute kendaraan dalam VRPD. Hasil penelitian menunjukkan bahwa algoritma SA menghasilkan solusi yang lebih baik dibandingkan dengan PSO, yang disebabkan oleh kemampuan SA untuk keluar dari solusi lokal optimal melalui langkah penerimaan individu berdasarkan peluang meskipun solusi tersebut lebih buruk dari sebelumnya. Hal ini menunjukkan bahwa SA lebih sesuai untuk menyelesaikan masalah kombinatorial seperti VRPD dibandingkan dengan PSO.

Penelitian yang dilakukan oleh Mohammed et. al (2017) bertujuan untuk menemukan rute optimal untuk *Vehicle Routing Problem* (VRP) di UNITEN menggunakan algoritma genetika. Pendekatan ini dirancang untuk mengurangi waktu tempuh dan jarak pada setiap jalur, sehingga dapat meningkatkan efisiensi transportasi siswa ke lokasi tujuan mereka. Selain itu, penelitian ini juga bertujuan untuk menekan biaya transportasi, seperti konsumsi bahan bakar dan biaya pemeliharaan kendaraan tambahan. Untuk mencapai tujuan tersebut, model *Capacitated Vehicle Routing Problem* (CVRP) diterapkan guna mengoptimalkan layanan shuttle bus di UNITEN. Hasil penelitian ini menunjukan proporsi pengurangan jarak untuk

setiap rute relatif pendek, tetapi penghematan jarak menjadi lebih besar ketika menghitung total jarak yang ditempuh oleh semua bus setiap hari atau bulanan. Hal ini juga berlaku untuk faktor waktu yang telah sedikit berkurang berdasarkan laju pengurangan jarak rute.

Penelitian yang dilakukan oleh Chowmali dan Sukto (2021) bertujuan untuk menyelesaikan masalah *multi-compartment vehicle routing problem* (MCVRP) dalam distribusi bahan bakar dengan menggunakan armada kendaraan yang heterogen. Metode yang digunakan adalah algoritma hibrida FJA-ALNS, yang merupakan pengembangan dari algoritma FJA yang sebelumnya diusulkan oleh penulis yang sama pada tahun 2020. Algoritma ini bertujuan untuk meminimalkan biaya perjalanan total dan jumlah kendaraan yang digunakan dalam distribusi bahan bakar. Hasil dari penelitian ini menunjukkan bahwa algoritma yang diusulkan efektif dalam menyelesaikan MCVRP, dan dapat diadaptasi untuk menyelesaikan masalah VRP lainnya dalam situasi dunia nyata.

Penelitian yang dilakukan oleh Mahmud & Haque Mokammel (2019) bertujuan untuk mengembangkan pendekatan Algoritma Genetika (GA) dalam menyelesaikan masalah *Multiple Depot Vehicle Routing Problem* (MDVRP), yang merupakan masalah optimasi kombinatorial NP-*Hard* yang kompleks. Metode yang digunakan terdiri dari tiga fase: pertama, pembuatan populasi awal dengan mengalokasikan pelanggan ke depot terdekat dan menggunakan algoritma pengelompokan untuk mengoptimalkan penugasan, kedua, optimasi rute dengan menciptakan rute layanan untuk setiap depot sambil memenuhi batasan yang ada, dan ketiga, evaluasi dan seleksi fitness menggunakan metode *weighted-sum* dan perankingan Pareto, diikuti dengan strategi seleksi turnamen untuk menghasilkan generasi solusi baru. Hasil penelitian menunjukkan bahwa GA yang diusulkan mampu menghasilkan solusi berkualitas tinggi untuk MDVRP, mengungguli pendekatan *GenClust* dalam beberapa kasus, serta memberikan dasar untuk pengembangan lebih lanjut dalam varian masalah seperti MDVRP dengan Jendela Waktu dan peningkatan penyetelan parameter.

Penelitian yang dilakukan oleh Shuen & Johar (2023) bertujuan untuk mengoptimalkan rute pengumpulan Sampah Padat Perkotaan dengan meminimalkan biaya perjalanan dan jarak tempuh menggunakan algoritma *Simulated Annealing* (SA). Metode yang digunakan melibatkan pemrograman MATLAB untuk menganalisis data nyata dari Koordinat GPS Malaysia serta tiga set data *benchmark Solomon*, yaitu data acak (R1), kluster (C1), dan kluster acak (RC1). Hasil penelitian menunjukkan bahwa metode yang diusulkan berhasil mengurangi total jarak tempuh untuk pengumpulan MSW, dengan biaya terbaik yang diperoleh sebesar

259.6743 untuk *Capacitated Vehicle Routing Problem* (CVRP), dibandingkan dengan 475.988 untuk masalah *Multi-Trips* (MTVRP), serta menunjukkan bahwa data kluster memberikan jarak perjalanan terpendek dan biaya terendah dibandingkan dengan data acak dan kluster acak.

Penelitian yang dilakukan oleh Maroof, et al. (2024) bertujuan untuk mengembangkan dan menguji efektivitas *Hybrid Genetic Algorithm-Solomon Insertion Heuristic* (HGA-SIH) dalam menyelesaikan *Vehicle Routing Problem with Time windows* (VRPTW), yang merupakan tantangan kompleks dalam logistik. Metode yang digunakan menggabungkan algoritma genetik dengan *Solomon Insertion Heuristic*, dimulai dengan menghasilkan populasi awal dari solusi acak dan solusi heuristik. Proses ini melibatkan teknik seleksi, crossover, dan mutasi untuk meningkatkan kualitas solusi. Hasil penelitian menunjukkan bahwa HGA-SIH berhasil mencapai *Best-Known Solutions* (BKS) untuk sebagian besar *instance* yang diuji, menegaskan efektivitasnya dalam mengoptimalkan rute kendaraan dengan mempertimbangkan batasan waktu dan kapasitas.

Penelitian yang dilakukan oleh Sajid, et al. (2020) memiliki tujuan utama untuk meminimalkan biaya dalam bentuk jarak tempuh melalui pemenuhan permintaan pelanggan yang telah diketahui dengan menggunakan kendaraan identik yang memiliki kapasitas terbatas. Metode yang diusulkan menggabungkan algoritma genetika dan *simulated annealing* untuk mencari solusi optimal secara global. Algoritma HGSA memanfaatkan operator *crossover* tetangga terdekat (*nearest-neighbor crossover*) yang dirancang khusus untuk menghasilkan solusi berdasarkan tetangga terdekat sehingga total jarak tempuh dapat diminimalkan. Evaluasi efektivitas algoritma HGSA dilakukan melalui pengujian terhadap 86 *instance benchmark* CVRP, dan hasilnya menunjukkan bahwa algoritma ini mampu memberikan kinerja yang efektif dalam mengoptimalkan total jarak tempuh.

Penelitian yang dilakukan oleh Yu, et al. (2022) bertujuan untuk mengembangkan algoritma *Simulated Annealing* (SA) untuk menyelesaikan masalah *Vehicle Routing Problem* dengan *Parcel Lockers* (VRPPL), dengan fokus pada efisiensi dalam pengelolaan rute pengiriman. Metode yang digunakan meliputi pengembangan model matematis berbasis *mixed-integer non-linear programming* dan penerapan algoritma SA untuk mencari solusi optimal. Hasil penelitian menunjukkan bahwa algoritma SA dapat memberikan kinerja yang kompetitif dibandingkan dengan solusi yang dihasilkan oleh Gurobi, serta mampu menangani kompleksitas masalah VRPPL secara efisien.

Penelitian yang dilakukan oleh Ferone, et al. (2024) bertujuan untuk mengembangkan metode algoritmik yang efisien dalam menyelesaikan masalah pengumpulan dan pengiriman spesimen medis (MSCDP) dengan menggabungkan matheuristik dan *Adaptive Large Neighborhood Search* (ALNS). Metode yang digunakan terdiri dari dua komponen utama yaitu pertama, pengelompokan spesimen menggunakan algoritma *k-means* untuk membentuk subset yang dapat dilayani oleh kendaraan tunggal, dan kedua, fase peningkatan ALNS yang mengeksplorasi ruang solusi dengan berbagai fungsi tetangga. Hasil eksperimen numerik menunjukkan bahwa pendekatan hibrid ini tidak hanya mampu menemukan solusi berkualitas baik untuk semua *instance* yang diuji, tetapi juga sering kali memperbaiki fungsi objektif yang dihasilkan oleh metode eksak sebelumnya, VeRSA, hingga satu urutan besaran, terutama pada masalah yang lebih besar.

Penelitian yang dilakukan oleh Xu, et al. (2022) bertujuan untuk mengatasi permasalahan *Electric Vehicle Routing Problem with Simultaneous Pickup and Delivery* (EVRPSPD) dengan mempertimbangkan pengisian daya non-linear dan pengosongan daya yang bergantung pada beban. Metode yang diusulkan memformulasikan masalah dalam bentuk *Mixed Integer Linear Programming* (MILP) dan menyelesaikannya menggunakan pendekatan heuristik *Adaptive Large Neighborhood Search* (ALNS). Untuk mengatasi skala besar dan kompleksitas masalah yang bersifat *NP-hard*, ALNS dikembangkan dengan operator baru dan pendekatan prioritas waktu yang dirancang untuk mengurangi waktu komputasi. Hasil pengujian menunjukkan bahwa metode yang diusulkan efektif dalam mengoptimalkan jumlah kendaraan dan waktu kerja total, termasuk waktu perjalanan, pengisian daya, dan pelayanan, yang dievaluasi melalui serangkaian data uji benchmark dengan berbagai skenario operasional.

Penelitian yang dilakukan oleh Lin, et al. (2024) bertujuan mengoptimalkan rute distribusi dalam rantai pasokan produk pertanian segar menggunakan algoritma *Adaptive Large Neighborhood Search* (ALNS) yang telah disempurnakan untuk meningkatkan efisiensi transportasi dan mempertahankan kesegaran produk selama distribusi. Metode yang digunakan mencakup pengembangan model optimasi jalur transportasi yang mempertimbangkan biaya operasional, waktu pengiriman, dan emisi karbon dengan memanfaatkan algoritma ALNS yang ditingkatkan melalui operator penyisipan dan penghapusan yang diadaptasi. Hasil pengujian pada dataset *benchmark Solomon* menunjukkan bahwa algoritma yang diusulkan mampu mengurangi biaya transportasi, mempercepat waktu komputasi, dan menjaga kesegaran produk di atas 80%.

Penelitian yang dilakukan oleh Redi, et al. (2020) bertujuan untuk meminimalkan biaya transportasi yang mencakup biaya perjalanan kendaraan, biaya sewa fasilitas perantara, dan biaya kompensasi bagi pelanggan yang harus menempuh perjalanan untuk mengakses fasilitas tersebut. Metode yang digunakan melibatkan algoritma *Simulated Annealing* sebagai pendekatan untuk menyelesaikan masalah ini. Selain itu, pendekatan pemodelan alternatif juga diusulkan melalui optimasi dua fase, yang terdiri dari masalah p-median dan masalah perutean kendaraan kapasitasi. Hasil percobaan numerik menunjukkan bahwa 2EVRP-LF lebih efektif dibandingkan dengan pendekatan optimasi dua fase. Algoritma *Simulated Annealing* juga terbukti memiliki kinerja yang efisien dalam menyelesaikan masalah 2EVRP-LF, menjadikannya solusi yang optimal untuk optimasi rute distribusi.

Tabel 2. 1 Posisi Penelitian

Penulis	**Obyek (Industri)**			**Metode**		
	Service	**FMCG**	**Manufacturing**	**GA**	**SA**	**ALNS**
(Patmawati & Nugroho, 2022)			√	√		
(Firmansyah, et al., 2021)		√		√	√	
(Anujuprana, 2015)		√		√		
(Prawira & Santosa, 2021)	√				√	
(Mohammed, et al., 2017)	√			√		
(Wasana & Sukto, 2021)			√			√
(Mahmud & Haque Mokammel, 2019)	√			√		
(Shuen & Johar, 2023)	√				√	
(Maroof, et al., 2024)	√			√		
(Sajid, et al., 2020)	√			√	√	
(Yu, et al., 2022)	√				√	
(Ferone, et al., 2024)			√			√
(Xu, et al., 2022)		√				√
(Liu, et al., 2024)		√				√
(Redi, et al., 2020)	√				√	
Usulan	√					√

2.1 *Supply Chain Management*

Supply chain management (SCM) merupakan suatu yang erat kaitannya dengan hubungan antara pemasok dengan perusahaan, hubunga perusahaan dengan pelanggan, serta proses berbagi informasi antara mitra, selain itu *supply chain management* juga memiliki pengaruh terhada kinerja manajemen rantai pasok di sebuah perusahaan. *Supply chain management*

(SCM) dapat membantu perusahaan dalam mengidentifikasi risiko dan mengambil tindakan pencegahan untuk mengurangi dampak negatif pada kinerja perusahaan (Permana, 2023).



Gambar 2. 1 *Supply Chain Management*

Source: britinnica.com

Hubungan yang tidak baik antara pihak pihak terkait dapat menyebabkan suatu perusahaan mengalami kerugian yang cukup besar. Salah satu dampak yang mungkin dapat terjadi karena ketidakefektifan dalam pertukaran informasi antara perusahaan, pemasok, pelanggan, distributor yaitu Bullwhip Effect (Hayati, 2014). Bullwhip Effect adalah kondisi di mana fluktuasi dalam peramalan jumlah permintaan semakin besar akibat buruknya sistem informasi dalam Supply Chain Management (SCM). Penyebab utama bullwhip effect adalah distorsi informasi permintaan dari konsumen akhir (end user) yang terus berlanjut ke rantai pasokan di atasnya, sehingga kuantitas permintaan sering kali tidak dapat dipenuhi secara optimal (Puspitaningtyas & ABdulrahim, 2023). Oleh karena itu, untuk menghindari terjadinya bullwhip effect pada suatu perusahaan perlu mengoptimalkan kinerja dari setiap pihak dalam rantai pasok. Berikut ini merupakan pihak utama yang terlibat dalam rantai pasok suatu perusahaan (Hayati, 2014):

1. *Supplier (chain 1)*

   Rantai dalam supply chain dimulai dari titik awal, yaitu supplier yang menyediakan bahan utama. Pada tahap ini, proses distribusi barang dimulai. Bahan utama tersebut dapat berupa bahan baku, bahan mentah, bahan pendukung, suku cadang, atau barang dagangan

2. *Supplier-Manufacturer (chain 1-2)*

   Rantai pertama tersebut kemudian diteruskan ke rantai kedua, yaitu manufaktur, suatu kegiatan yang berfungsi untuk mengubah atau menyelesaikan bahan baku menjadi produk jadi (*finished good*). Dalam mengoptimalkan rantai ini perusahaan dapat melakukan

efisiensi, seperti pengurangan biaya penyimpanan inventaris dengan mengembangkan konsep kemitraan dengan pemasok.

3. *Supplier-Manufacturer-Distribution (chain 1-2-3)*

   Pada tahap ini, produk jadi yang dihasilkan didistribusikan kepada pelanggan melalui proses logistic yang dapat memanfaatkan perusahaan 3PL. Barang jadi biasanya disalurkan melalui distributor atau grosir, yang berperan sebagai pedagang dengan volume penjualan besar.

4. *Supplier-Manufacturer-Distribution-Retail Outlets (chain 1-2-3-4)*

   Produk dari pedagang besar atau grosir kemudian didistribusikan ke toko pengecer (*retail outlets*). Meskipun ada beberapa perusahaan yang langsung menjual produk mereka kepada pelanggan, jumlahnya relatif sedikit, dan kebanyakan masih menggunakan pola distribusi seperti ini.

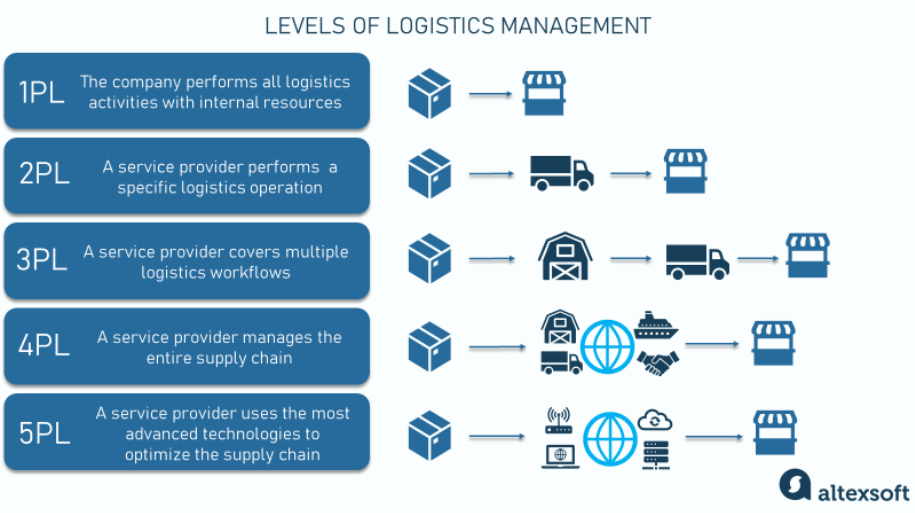
5. *Supplier-Manufacturer-Distribution-Retail Outlets-Customer (chain 1-2-3-4-5).*

   Customer merupakan rantai terakhir yang dilalui dalam *supply chain* yang menjadi *end-user*.

2.1.1 Logistik

Logistik merupakan suatu proses pengelolaan dalam pemindahan dan penyimpanan barang, suku cadang, dan produk jadi dari pemasok, di antara fasilitas perusahaan, atau kepada pelanggan. Logistik merupakan bagian dari manajemen rantai pasokan yang berfokus pada pemindahan produk atau material dengan cara yang paling efisien, sehingga barang tersebut sampai di lokasi yang tepat pada waktu yang tepat. Kegiatan utama logistik adalah pengadaan, penyimpanan, persediaan, pengangkutan, pergudangan, pengemasan, keamanan, dan penanganan barang dan jasa baik dalam bentuk bahan baku, barang antara, dan barang jadi (Kasengkang, et al., 2016). Proses distribusi merupakan salah satu proses dalam manajemen logistic Dimana mengoptimalkan penyaluran barang ke *customer* menggunakan saluran distribusi yang tepat. Saluran Distribusi adalah saluran yang digunakan oleh produsen untuk menyalurkan barang tersebut dari produsen sampai ke konsumen atau pemakai industri (Wijaya, et al., 2021).

Klasifikasi logistic digunakan untuk mengetahui tingkatan atau evolusi pengelolaan logistic yang dapat disesuaikan suatu perusahaan untuk mendukung kegiatan operasional yang disesuikan berdasarkan skala bisnis dan kebutuhan perusahaan. Berikut merupakan klasifikasi layanan logistik dari tingkatan paling rendah ke tingkatan paling tinggi (Zai, et al., 2022):



Gambar 2. 2 *Level of Logistics Management*

Source: (altexsoft.com)

1. 1PL (*First Party Logistics*)

   1PL sering disebut dengan logistik otonom dimana semua operasi logistik, seperti transportasi dan pergudangan, dilakukan sendiri oleh pemilik barang atau perusahaan menggunakan infrastruktur dan sumber daya internal.

2. 2PL (*Second Party Logistics*)

   2PL adalah suatu layanan individu pihak kedua dengan sumber daya sendiri atau eksternal yang digunakan untuk melaksanakan salah satu kegiatan logistik seperti kegiatan transportasi untuk melakukan pengangkutan barang.

3. 3PL (*Third Party Logistics*)

   3PL adalah suatu layanan pihak ketiga yang mengelola berbagai aktivitas logistic seperti pengangkutan, pergudangan, dan distribusi barang. 3PL bisa meliputi layanan seperti transportasi, pergudangan, manajemen inventori, kegiatan yang menambah nilai (perakitan dan pemasangan produk), dan terkait informasi kegiatan (pelacakan, penelusuran, dan perencanaan distribusi).

4. 4PL (*Fourth Party Logistics*)

   4PL adalah layanan logistik yang disesuaikan menggunakan sumber daya terbaik di kelasnya untuk memenuhi permintaan pelanggan mereka dalam rantai pasokan. 4PL bertindak sebagai penghubung utama yang mengelola semua proses logistik.

5. 5PL (*Fifth Party Logistics*)

5PL pada dasarnya merupakan model outsourcing yang menambah kemajuan logis pada pengembangan integrasi logistik penuh termasuk digitalisasi dan otomatisasi, untuk mengoptimalkan seluruh rantai pasok dan menyediakan solusi berbasis data. Layanan ini berpusat di penyediaan solusi e-logistik untuk semua rantai pasok.

2.1.2 Transportasi

Berdasarkan Bahasa latin, Transportasi terdiri dari dua kata yaitu *trans* dan *portare*. *Trans* berarti melintasi atau ke sisi lain, sedangkan *portare* yang berarti mengankut atau membawa. Olehn karena itu dapat disimpulkan bahwa transportasi meruipakan suatu kegiatan pemindahan barang dari suatu tempat asal ke tempat tujuan (Arwini & Juniastra, 2023). Transportasi merupakan proses yang sangat penting dengan memprioritaskan hubungan atau keterkaitan antara Lokasi, jaringan, dan kebutuhan transportasi.

a. Lokasi

   Transportasi memiliki peran utama yang mana menghubungkan antara lokasi satu dengan lokasi lainnya yang mana tiap Lokasi merupakan akses dari suatu sistem distribusi.

b. Jaringan transportasi

   Jaringan transportasi merupakan jaringan yang meliputi kebijakan kebijakan seperti rute perjalanan dari moda transportasi yang digunakan. Jarinagan transportasi ini memegang peranan penting dalam pelaksanaan kegiatan transportasi sebelum maupun sesudah proses produksi.

c. Kebutuhan transportasi

   Kebutuhan transportasi dapat diukur melalui besarnya pasar yang ada, potensi penjualan produk, tingkat konsumsi, dan jumlah populasi. Semakin tinggi permintaan terhadap suatu produk, semakin besar pula kebutuhan akan infrastruktur transportasi yang memadai. Hal ini disebabkan oleh peningkatan jumlah dan keragaman moda transportasi, baik yang sejenis maupun berbeda, untuk memenuhi kebutuhan tersebut.

Transportasi, sebagai bagian dari pengiriman, memiliki peran penting dalam manajemen rantai pasok karena jarang sekali suatu produk diproduksi dan dikonsumsi di lokasi yang sama. Oleh karena itu, keberhasilan strategi rantai pasok memerlukan pengelolaan transportasi yang efektif, termasuk pemilihan metode transportasi dan penentuan rute yang tepat untuk mengoptimalkan biaya pengiriman (Rahayu, et al., 2019).

2.2 Biaya Operasional

Biaya operasional merupakan jumlah atau total biaya yang dikeluarkan yang berkaitan dengan kegiatan yang dilakukan oleh perusahaan dalam menjalankan kegiatannya selain biaya produksi. Biaya ini termasuk biaya penjualan dan administrasi lainnya (Susetyo, et al., 2021). Dalam memperoleh biaya operasional dapat diformulasikan menggunakan rumus berikut:

$$Biaya\ Operasional = Biaya\ Penjualan + Biaya\ Administrasi\ Umum$$

Salah satu biaya operasional yang dikeluarkan oleh suatu perusahaan untuk mendukung kegiatan operasional yaitu biaya logistic. Berikut merupakan jenis jenis biaya yang termasuk kedalam biaya logistic:

a. *Purchase Order Cost*, merupakan biaya yang dikeluarkan dari kegiatan pembelian suatu produk atau material yang digunakan untuk memproduksi produk dari suatu perusahaan. Biaya ini biasanya keluar akibat adanya permintaan produk dari *customer*.
b. *Inbound Cost*, merupakan biaya yang dikeluarkan pada proses umloading produk yang sampai ke Gudang yang besarannya berupa tarif per unit karton sebagai UoM (*Unit of Measurement)* produk. Selain itu, biaya ini termasuk biaya suplai pengaadaan suatu produk.
c. Biaya Penyimpanan dan Penanganan atau *Carring Cost* merupakan biaya yang dikeluarkan untuk menyimpan produk di dalam gudang seperti biaya listrik, biaya perawatan gudang, dll. Biaya ini sebagai biaya internal perusahaan karena dikeluarkan untuk pemeliharaan Gudang perusahaan.
d. Biaya Distribusi (*Delivery Cost*), merupakan biaya yang dikeluarkan untuk proses pengiriman produk dari perusahaan ke *customer*. Umumnya biaya ini sebagai tarif pengiriman dari Gudang DC perusahaan ke *customer*.

Dalam pengelolahan biaya operasional yang dikeluarkan perusahaan terdapat juga klasifikasi biaya berdasarkan perilaku biaya sebagai berikut (Kadriyani, et al., 2022):

a. Biaya Tetap

   Biaya tetap merupakan salah satu jenis biaya yang dikeluarkan secara konstan oleh suatu perusahaan tanpa mempertimbangkan adanya perubahan atau faktor lain pada kegiatan operasional yang berlangsung. Dalam kegiatan distribusi produk, biaya tetap diantaranya meliputi biaya sewa kendaraan, gaji karyawan, dan biaya sewa gudang.
b. Biaya Variable

Biaya variable merupakan salah satu jenis biaya yang dikeluarkan perusahaan yang jumlahnya berubah ubah menyesuaikan dengan kegiatan operasional yang berlangsung. Dalam kegiatan distribusi produk biaya variable meliputi, biaya bahan bakar yang mempertimbangkan rasio penggunaan bahan bakar dan jarak tempuh kendaraan.

2.3 *Vehicle Routing Problem*

Vehicle Routing Problem merupakan masalah dalam penentuan rute kendaraan untuk melakukan distribusi atau pengiriman produk dari suatu depot (Gudang) ke pelanggan. VRP ini memiliki peranan penting dalam dunia industri pada permasalahan manajemen logistik (Pasha, et al., 2022). VRP merupakan salah satu tantangan paling kritis yang dihadapi perusahaan logistik saat ini. Para peneliti telah mempelajari perutean kendaraan dan penjadwalan pengiriman sejak tahun 1959 ketika Dantzig dan Ramser (1959) memperkenalkan *Vehicle Routing Problem*. Permasalahan in dianggap sebagai paradigmatik dari masalah perutean kendaraan (VRP) dan mengacu pada distribusi barang, dari depot pusat ke pelanggan yang tersebar secara geografis (Tan & Yeh, 2021).

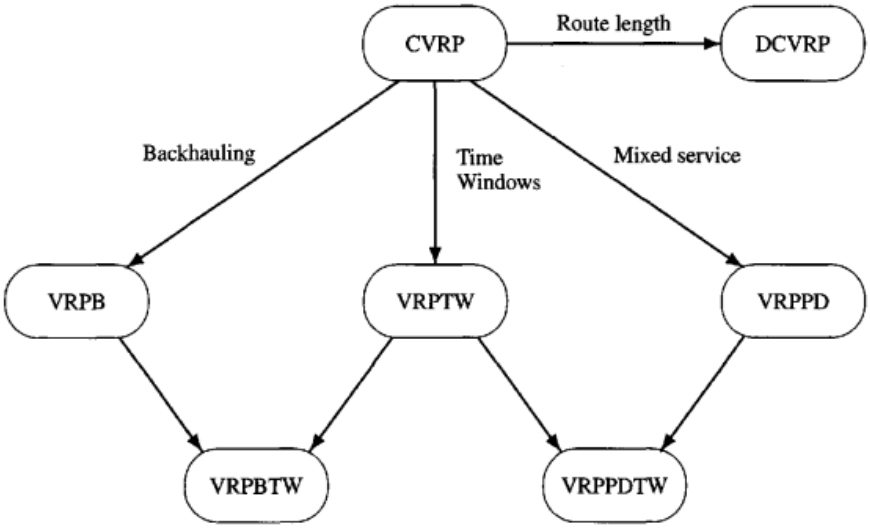
Vehicle Routing Problem ditunjukan dengan suatu graf terhubung G = (V,E), dengan V= {1,2,…..,n} merupakan himpunan tak kosong dari simpul (*Vertex* atau *node*) dan E merupakan himpunan busur (*edge* atau *arc*) yang menghubungkan sepanjang simpul dalam graf. Simpul merepresentasikan sekumpulan pelanggan yang akan dikunjungi, di mana indeks 1 dianggap sebagai depot. Sementara itu, busur menggambarkan perjalanan atau rute yang menghubungkan pelanggan i dengan pelanggan j (Widyastiti & Sumarsa, 2023). Berikut merupakan tujuan utama dalam penyelesaian masalah VRP (Toth P & Vigo D, 2002):

1. Meminimalkan jarak tempuh dari suatu kendaraan serta meminimalkan biaya tetap yang berkaitan dengan penggunaan kendaraan dalam kegiatan distribusi produk ke pelanggan.
2. Mengoptimalkan jumlah kendaraan yang digunakan untuk melakukan pelayanan dalam memenuhi permintaan seluruh pelanggan.
3. Mengoptimalkan rute rute dari setiap titik pengantaran sehingga dapat meminimalkan waktu perjalanan dan muatan yang diangkut.
4. Meminimalkan pinalti sebagai akibat dari pelayanan yang kurang memuaskan terhadap pelanggan, seperti keterlambatan pengiriman dan lain sebagainya.

Untuk mencapai tujuan *Vehicle Routing Problem* (VRP) tersebut, diperlukan pemahaman mendalam terhadap berbagai aspek yang memengaruhi masalah VRP. Menurut

Toth dan Vigo (2002) mengidentifikasi sejumlah komponen utama yang menjadi dasar formulasi VRP, yaitu sebagai berikut:

1. Jaringan jalan, ditunjukan menggunakan sebuah *graph* (diagram) yang terdiri dari arc yang menunjukan bagian bagian jalan dan *vertex* yang menunjukan titik Lokasi konsumen dan depot. Setiap rute perjalanan memiliki nilai yang mewakili biaya dan waktu tempuhnya.
2. Konsumen, ditunjukan dengan vertex (titik) yang mana setiap konsumen biasanya memiliki beberapa hal seperti:
    a. Jumlah permintaan barang yang harus dikirim atau diambis sesuai dengan permintaan konsumen yang mana jenis barang dapat berbeda beda.
    b. Jendela waktu pelayanan dalam rentang tertentu yang mana diluar waktu yang telah ditentukan konsumen tidak dapat menerima pengiriman maupun pengambilan.
    c. Waktu yang digunakan untuk melakukan penurunan atau pemuatan suatu barang kepada konsumen pada Lokasi tertentu yang telah ditentukan dan disesuaikan dengan jenis kendaraan yang digunakan.
    d. Pengelompokan kendaraan yang tersedia untuk melayani konsumen.
    e. Prioritas atau pinalti yang perlu dikeluarkan karena berhubungan dengan kemampuan kendaraan dalam melayani permintaan konsumen.
3. Depot, disimbolkan menggunakan vertex (titik) yang merupakan ujung awal dan akhir dari suatu rute kendaraan. Depot memiliki sejumlah kendaraan dengan jenis dan kapasitas tertentu yang dapat digunakan dalam mendistribusikan barang atau jasa pada jam operasional depot yang telah ditentukan (*time windows depot*).
4. Kendaraan, armada angkut yang digunakan dalam distribusi memiliki kapasitas tertentu yang membatasi jumlah permintaan konsumen, sehingga total permintaan tidak boleh melebihi kapasitas kendaraan tersebut. Selain itu, penggunaan kendaraan juga melibatkan biaya operasional, seperti biaya bahan bakar dan sewa kendaraan.
5. Pengemudi, memiliki kendala seperti jam kerja harian, tambahan waktu lembur apabila diperlukan, jumlah dan jam istirahat, serta durasi maksimum perjalanan.



Gambar 2. 3 Kerangka Permasalahan VRP

Source : (VRP-Toth Vigo)

Berdasarkan penelitian yang dilakukan Toth dan Vigo (2002), terdapat beberapa permasalahan dasar yang ditemukan sehingga dikelompokan dalam kelas yang merepresentasikan variasi VRP sebagai berikut (Toth P & Vigo D, 2002):

a. *Capacitated Vehicle Routing Problem* (CVRP), merupakan variasi permasalahan VRP yang sederhana dengan mempertimbangkan kapasitas kendaraan sebagai suatu kendala yang dihadapi. Variasi VRP ini merupakan variasi yang paling banyak dialami oleh suatu perusahaan.
b. *Distance Constrained Vehicle Routing Problem* (DCVRP), merupakan variasi permasalahan VRP dengan kendala berupa batasan panjang rute pengantaran produk ke konsumen.
c. *Vehicle Routing Problem with Time windows* (VRPTW), merupakan variasi permasalahan VRP dimana terdapat kendala berupa setiap konsumen memiliki batasan waktu pelayanan dalam mengantarkan dan mengambil barang.
d. *Vehicle Routing Problem with Pick up and Delivery* (VRPPD), merupakan variasi permasalahan VRP yang mana memiliki dua pelayanan yaitu pengiriman dan pengambilan barang dalam satu rute distribusi.
e. *Vehicle Routing Problem with Backhauls* (VRPB), merupakan variasi permasalahan VRP yang mana pengambilan baru dapat dilakukan setelah semua pengiriman selesai.

f. *Vehicle Routing Problem with Backhaul and Time windows* (VRPBTW), merupakan variasi VRP dengan linehaul dan backhaul *customer* dimana *customer* harus dilayani dalam interval waktu tertentu.
g. *Vehicle Routing Problem with Pick-up and Delivery and Time windows* (VRPPDTW), merupakan variasi permasalahan VRP yang mana pengambilan barang dari satu pelanggan atau lokasi (jemput) untuk diantar ke pelanggan atau lokasi lain (antar) menggunakan kendaraan yang sama. Selain itu, setiap pelanggan atau lokasi memiliki batasan waktu tertentu untuk pelayanan.

2.4 *Capacitated Vehicle Routing Problem with Time Windows* (CVRPTW)

CVRPTW merupakan salah satu jenis masalah dalam penentuan rute kendaraan dengan kompleksitas tambahan berupa batasan jendela waktu pada setiap titik/node dan juga kapasitas pada setiap kendaraan. Tujuan utama dari CVRPTW adalah untuk melayani semua pelanggan tanpa melanggar kendala kapasitas kendaraan dan kendala jendela waktu dengan jumlah kendaraan minimum sehingga dapayt menentukan rute dengan jarak tempuh paling optimal (Li & Lim, 2003). CVRPTW merupakan generalisasi dari VRP yang termasuk pada masalah NP-Hard. NP-*Hard Problem* adalah jenis masalah komputasi yang sangat kompleks dan sulit untuk diselesaikan. Masalah ini memerlukan waktu komputasi yang cukup lama dalam menemukan Solusi terbaik dengan menggunakan algoritma yang tidak konvensional (Cokrowibowo, et al., 2019).

NP-Hard Problem, seperti *Vehicle Routing Problem* (VRP), yang memiliki tingkat kompleksitas yang sangat tinggi, pendekatan eksak sering kali tidak praktis untuk diterapkan. Oleh karena itu, digunakan pendekatan heuristik yang lebih fleksibel. Dalam menyelesaikan permasalahan ini, model matematis digunakan untuk memberikan *representative* masalah dalam bentuk formulasi matematika yang menjadi acuan dalam pencarian Solusi. Berikut merupakan model matematis dari VRP dengan variasi CVRPTW (Kallehauge, et al., 2005):

Objective Function:

$$\min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \qquad (2.1)$$

Subject to:

$$\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1 \qquad \forall i \in C \qquad (2.2)$$

$$\sum_{i \in C} d_i \sum_{j \in N} x_{ijk} \leq q \qquad \forall k \in V \tag{2.3}$$

$$\sum_{j \in N} x_{0jk} = 1 \qquad \forall k \in V \tag{2.4}$$

$$\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{hjk} = 0 \qquad \forall h \in C, \ \forall k \in V \tag{2.5}$$

$$\sum_{i \in N} x_{i,n+1,k} = 1 \qquad \forall k \in V \tag{2.6}$$

$$x_{ijk}(s_{ik} + t_{ij} - s_{jk}) \leq 0 \qquad \forall i, j \in N, \forall k \in V \tag{2.7}$$

$$a_i \leq s_{ik} \leq b_i \qquad \forall i \in N, \forall k \in V \tag{2.8}$$

$$x_{ijk} \in \{0,1\} \qquad \forall k \in V, \forall (i,j) \in N \tag{2.9}$$

Keterangan:

V	:	Kumpulan kendaraan	q	:	Kapasitas maksimum kendaraan
C	:	Kumpulan pelanggan	d_i	:	Permintaan dari pelanggan i
N	:	Sekumpulan simput, termasuk depot dan pelanggan	a_i	:	Waktu paling awal kendaraan bisa mulai melayani pelanggan i.
c_{ij}	:	Variabel biner, bernilai 1 jika Biaya perjalanan dari simpul i ke simpul j	b_i	:	Waktu paling akhir kendaraan bisa mulai melayani pelanggan i.
t_{ij}	:	Waktu perjalanan dari simpul I ke simpul j	s_{ik}	:	Waktu kendaraan k mulai melayani pelanggan i.

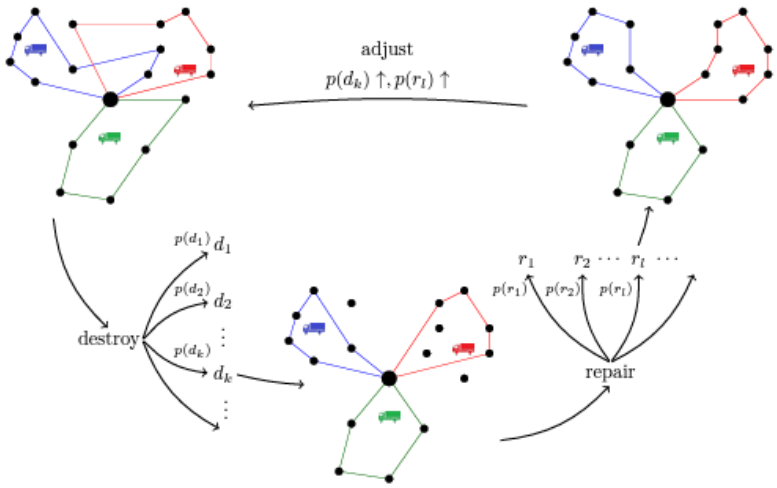
Berdasarkan model matematis diatas diketahui model tersebut bertujuan untuk meminimalkan total biaya perjalanan kendaraan dalam mengantarkan barang kepada pelanggan (2.1). Terdapat beberapa kendala yang dijabarkan dalam beberapa constraint seperti, kendala (2.2) memastikan bahwa setiap pelanggan dikunjungi tepat satu kali oleh salah satu kendaraan, tanpa ada yang terlewat atau dikunjungi lebih dari sekali. Kendala (2.3) mengatur bahwa kapasitas kendaraan tidak boleh melebihi batas maksimal yang telah ditentukan, sehingga setiap kendaraan hanya membawa barang sesuai kapastias kendaraan tersebut. Selanjutnya, kendala (2.4) mewajibkan setiap kendaraan untuk berangkat dari gudang (depot) sebagai titik awal,

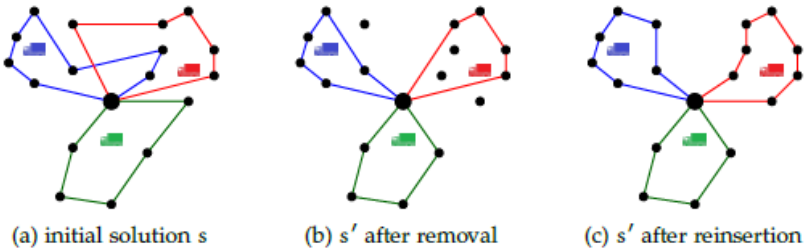
sementara kendala (2.5) memastikan bahwa setelah kendaraan mengantarkan barang ke satu pelanggan, kemudia kendaraan tersebut melanjutkan perjalanan ke tujuan berikutnya. Kendala (2.6) menetapkan bahwa semua kendaraan harus kembali ke gudang atau depot sebagai titik akhir setelah selesai mengantarkan barang.

Selain itu, terdapat kendala (2.7) yang mengatur hubungan waktu antar kunjungan dengan, yang memastikan kendaraan memperhitungkan waktu perjalanan dari satu pelanggan ke pelanggan berikutnya. Kemudian, kendala (2.8) mewajibkan kendaraan untuk mengikuti jendela waktu (*time windows*) yang ditetapkan oleh pelanggan, sehingga pengantaran dilakukan dalam rentang waktu yang diizinkan. Terakhir, kendala (2.9) memastikan bahwa semua variabel dalam model, seperti jumlah kendaraan atau keputusan rute, berbentuk angka bulat (*integer*). Jika terdapat kendaraan yang tidak digunakan, model akan membentuk output yang mana merepresentasikan kendaraan tersebut menempuh rute kosong dari gudang ke gudang tanpa mengunjungi pelanggan, yaitu dengan mengambil rute (0, n + 1).

2.5 Algoritma *Adaptive Large Neighborhood Search* (ALNS)

ALNS merupakan sebuah metode dari perluasan metode *large neighborhood search* yang dikembangkan oleh Shaw, sebuah metode penyelesaian masalah VRP yang juga terkait dengan prinsip *ruin-and-recreate*. Konsep dasar metode ALNS yaitu mencari solusi yang lebih baik pada setiap iterasi dengan cara menghancurkan sebagian dari solusi saat ini dan membangunnya kembali dengan cara yang berbeda (Azi, et al., 2014). Dalam perluasan adaptif, setiap operator diberikan bobot, dan probabilitas pemilihannya terkait dengan bobot tersebut. Bobot ini disesuaikan selama proses pencarian berdasarkan keberhasilan yang dicapai operator tersebut di masa lalu.





Gambar 2. 4 Skema Algoritma ALNS

Source: (Lutz, 2014)

Skema *Adaptive Large Neighborhood Search* (ALNS) dimulai dengan solusi awal yang melalui proses penghancuran (*destroy*) dan perbaikan (*repair*). Heuristik penghancuran maupun perbaikan dipilih dari sekumpulan heuristik yang tersedia. Probabilitas pemilihan heuristik didasarkan pada tingkat keberhasilannya. Setelah sejumlah iterasi, probabilitas tersebut disesuaikan sesuai dengan keberhasilan masing-masing heuristik (Lutz, 2014). Berikut merupakan framework dari algoritma ALNS:

Algorithm 3: Adaptive Large Neighborhood Search

Input: problem instance I
create initial solution $s_{min} = s \in S(I)$
while stopping criteria not met **do**
  for $i = 1, \ldots, p_u$ **do**
    select $r \in R, d \in D$ according to probabilities p
    $s' = r(d(s))$
    if accept(s, s') **then**
      $s = s'$
      if $c(s) < c(s_{min})$ **then**
        $s_{min} = s$
  adjust the weights w and probabilities p of the heuristics
return s_{min}

Gambar 2. 5 *Framework* Algoritma ALNS

Source: (Lutz, 2014)

Repair dan *destroy* merupakan bagian *framework* algortima ALNS yang mana menggantikan metode pencarian solusi tradisional. Dalam hal ini, kedua fungsi tersebut bekerja dengan cara menghancurkan sebagian solusi yang sudah ada, yaitu menghapus beberapa bagiannya, lalu memperbaikinya kembali (Lutz, 2014). Tujuannya adalah untuk menghapus bagian-bagian yang memungkinkan solusi diperbaiki dan menjadi lebih baik.

a. *Destroy Operator*

Destroy Operator merupakan operator yang digunakan dalam algoritma ALNS untuk menghancurkan atau menghapus sebagian node (titik) dari Solusi yang sudah ada yang bertujuan untuk membuka ruang perbaikan melalui proses *repair*. Berikut merupakan beberapa jenis *destroy operator* yang biasa digunakan dalam algoritma ALNS (Mara, et al., 2022):

Tabel 2. 2 Jenis Jenis *Destroy Operator*

Destroy Operator	**Penjelasan**
Random Removal	*Random removal* merupakan salah satu teknik *destroy* yang mana bekerja dengan cara menghapus permintaan secara seragam secara acak. Penghapusan Acak mungkin tampak sebagai strategi penghancuran yang tidak menguntungkan pada pandangan pertama karena dapat menghapus bagian solusi yang sudah pas. Namun, teknik ini memiliki efek yang sangat positif karena memberikan diversifikasi.
Worst Removal	*Worst removal* adalah menghapus bagian-bagian terburuk dari sebuah solusi, yaitu bagian-bagian yang menyebabkan biaya paling besar, dengan harapan bahwa heuristic perbaikan (*repair* heuristic) dapat menghilangkan biaya besar tersebut. Cara umum untuk menentukan biaya dari bagian tertentu dalam solusi adalah dengan menghitung selisih antara total biaya solusi sebelum dan setelah bagian tersebut dihapus.
Related Removal	*Related Removal* adalah salah satu teknik *destroy* yang memanfaatkan hubungan antara bagian-bagian dalam sebuah solusi. Dalam metode ini, diasumsikan bahwa bagian-bagian yang saling terkait lebih mudah untuk diganti atau dipertukarkan. Strateginya adalah menghapus bagian-bagian yang mirip sehingga fungsi perbaikan memiliki peluang untuk menyusunnya kembali dengan cara yang lebih menguntungkan.
History-Based Removal	*History-Based Removal* adalah salah satu teknik *destroy* yang memanfaatkan informasi historis dari semua bagian dalam sebuah solusi. Informasi historis ini digunakan untuk mengevaluasi apakah suatu bagian memiliki potensi untuk dioptimalkan. Jika bagian tersebut dianggap memiliki potensi optimasi, maka bagian tersebut akan dihapus untuk memberi kesempatan fungsi *repair* menyusunnya kembali dengan cara yang lebih baik.

Destroy Operator	**Penjelasan**
Route Removal	*Route Removal* adalah salah satu teknik *destroy* yang memiliki cara kerja menghapus sejumlah rute secara acak beserta titik (node) yang terkait dalam sebuah solusi. Tujuan teknik ini yaitu memberikan ruang untuk memperbaiki solusi dengan cara yang lebih optimal.
Neighborhood Removal	*Neighborhood Removal* merupakan salah satu Teknik *destroy* yang bekerja dengan cara menghapus secara bertahap (iteratif) sebuah titik (node) yang dianggap signifikan berdasarkan jarak rata-rata rute tempat titik tersebut berada. Tujuannya adalah untuk mengoptimalkan rute dengan menghilangkan titik-titik yang memengaruhi efisiensi rute secara keseluruhan.
Cluster Removal	*Cluster Removal* merupakan salah satu Teknik *destroy* yang menggunakan metode cluster, untuk membagi solusi menjadi dua klaster. Setelah itu, semua titik (node) dalam salah satu klaster yang dipilih secara acak akan dihapus dari solusi. Teknik ini bertujuaan untuk memberikan ruang untuk menyusun kembali solusi dengan cara yang lebih optimal.

b. *Repair Operator*

Repair Operator merupakan operator yang digunakan dalam algoritma ALNS untuk memperbaiki atau mengembalikan Solusi yang telah dihancurkan oleh *destroy operator*. *Repair operator* bertugas membangun kembali solusi yang valid dan berpotensi lebih baik untuk meningkatkan kualitas solusi. Berikut merupakan jenis jenis *repair* operator yang digunakan dalam algoritma ALNS (Mara, et al., 2022):

Tabel 2. 3 Jenis Jenis *Repair Operator*

Repair Operator	**Penjelasan**
Greedy Insertion	*Greedy Insertion* merupakan salah satu teknik *repair* yang bekerja dengan cara memiliki dan memasukan kembali sebuah titik (node) dari daftar titik yang telah dihapus secara iteratif hingga semua titik dimasukan kembali. Titik yang dipilih yang memiliki biaya penyisipan paling rendah dibandingkan dengan titik lainnya.

Repair Operator	**Penjelasan**
Regret Insertion	*Regret Insertion* merupakan salah satu teknik *repair* yang bekerja dengan cara memilih dan memasukkan kembali titik (node) yang memiliki nilai *regret* terbesar dari daftar titik yang telah dihapus. Nilai *regret* suatu titik dihitung dari selisih antara biaya posisi penyisipan terbaik dengan biaya posisi penyisipan terbaik ke-k. Proses ini dilakukan secara bertahap (iteratif) hingga semua titik dalam daftar dimasukkan kembali.
Random Insertion	*Random Insertion* merupakan salah satu Teknik *repair* yang bekerja dengan cara secara acak memilih sebuah titik (node) dari daftar titik yang telah dihapus, kemudian memasukkannya kembali ke posisi yang menghasilkan biaya tambahan paling rendah.
Sequential Insertion	*Sequential Insertion* merupakan salah satu teknik *repair* yang berkerja dengan cara memasukan node (titik) dengan mencari posisi penyisipan yang membutuhkan biaya paling kecil (misalnya, jarak atau waktu paling sedikit).
Zone Insertion	*Zone Insertion* merupakan salah satu teknik *repair* yang hampir mirip dengan Greedy Insertion akan tetapi teknik ini bukan berfokus pada jarak terdekat melainkan pada jadwal waktu yang telah ditentukan. Node akan dimasukan kedalam rute dengan memastikan semua selesai tepat waktu.
Swap Insertion	*Swap Insertion* merupakan salah satu teknik *repair* yang bekerja dengan cara memilih dua node (titik) secara acar dari solusi yang sudah ada lalu menukarnya.

Setelah operator *destroy* dan *repair* menghasilkan Solusi baru atau s', *acceptance method* digunakan digunakan untuk memutuskan apakah solusi baru yang dihasilkan oleh operator selama iterasi, akan diterima untuk menggantikan solusi saat ini atau tidak. Metode ini berfungsi sebagai aturan seleksi untuk menjaga keseimbangan antara eksplorasi ruang solusi yang lebih luas dan eksploitasi solusi yang menjanjikan. Berikut merupakan beberapa metode *acceptance criteria* yang dapat digunakan untuk mengevaluasi Solusi baru yang dihasilkan dalam metode ALNS (Mara, et al., 2022):

Tabel 2. 4 *Acceptance Criterion Method*

Criteria Name	***Description***	***Number of Papers***	***Relative Presence***
Greedy Acceptence	Menerima solusi baru S' jika hanya $f(S') < f(S)$. Solusi baru diterima jika mengurangi biaya dibandingkan dengan solusi saat ini.	20	7,94%
Metropolis Criterion	Menerima solusi baru S' jika $f(S') < f(S)$. Akan tetapi, jika $f(S') \geq f(S)$ yang menunjukan solusi lebih buruk maka akan dihitung peluangnya. Solusi baru S' akan diterima jika $rand[0,1] < e^{-(f(S')-f(S))/T}$.	205	81,35%
Record-to-record	Menerima solusi baru S' jika $\frac{f(S')-f(S^*)}{f(S')} < R$, dimana R merupakan nilai ambang batas yang ditentukan sebelumnya .	20	7,94%
Threshold Acceptance	Menerima solusi baru S' jika $\frac{f(S')-f(S^*)}{f(S')} < R$, dimana R merupakan nilai ambang batas yang ditentukan sebelumnya dan nilai R akan berkurang secara bertahap seiring dengan iterasi yang dilakukan dalam algoritma.	3	1,19%
Pareto Dominance	Menerima solusi baru S' jika hunungan antara $\forall_{o=1}^{O} f_o(S') \leqslant f_o(S)$ dan $\exists_{o=1}^{O} f_o(S') \langle f_o(S)$ terpenuhi, atau dalam kata lain solusi baru lebih dominan dalam semua kriteria atau beberapa kriteria $S(S' \prec S)$.	3	1,19%
Other Mechanism		6	2,38%

BAB III
METODE PENELITIAN

3.1 Objek Penelitian

Objek penelitian merupakan sasaran ilmiah yang digunakan untuk mengumpulkan data dengan tujuan penelitian yang bersifat objektif, valid, dan reliabel. Dalam penelitian "Optimasi *Capacitated Vehicle Routing Problem with Time Windows* (CVRPTW) Menggunakan Algoritma ALNS untuk Minimasi Jarak", yang menjadi objek yaitu pada optimasi rute distribusi barang yang dilakukan oleh PT. Pos Logistik Indonesia untuk minimasi jarak pada setiap rute yang nantinya akan diselesaikan menggunakan pendekatan *metaheuristic* untuk menjawab permasalahan dan tujuan penelitian pada laporan tugas akhir ini. Berikut ini merupakan profil dari PT. Pos Logistik Indonesia:

Nama Perusahaan : PT. Pos Logistik Indonesia

Alamat : Gedung Pos Ibukota, Jl. Lap. Banteng Utara No.1, Pasar Baru, Sawah Besar, Central Jakarta City, Jakarta 10710.

No. Telepon : +6281297274831

Bidang Usaha :
- *Warehouse*
- *Distribution*
- *Heavy Lifting*
- *International Cargo*
- *Domestic Cargo*
- *Cold Chain*

Bentuk Perusahaan : Perseroan Terbatas

Tahun Berdiri : 2012



Gambar 3. 1 Logo dan Lokasi PT. Pos Logistik Indonesia

3.2 Instrumen Penelitian

Berikut merupakan instrument penelitian yang digunakan untuk mendukung proses pengumpulan dan analisis data dalam penelitian ini:

1. Dataset, berisikan data yang dibutuhkan dalam pengolahan data seperti koordinat lokasi pelanggan, data batas waktu, dan jumlah permintaan setiap pelanggan.
2. Bahasa Python berisikan kode program untuk mengembangkan algoritma seperti ALNS untuk penyelesaian dan penentuan rute optimal pada pengiriman barang.
3. Visual Code/Google Colab, *software* digunakan dalam melakukan pengolahan data untuk menulis dan menjalankan kode program.

3.3 Sumber Data Primer dan Sekunder

Jenis dan sumber data yang didapatkan berasal dari data primer dan data sekunder. Berikut merupakan data yang digunakan dalam penelitian ini:

1. Data primer yang diperoleh pada penelitian ini melalui wawancara kepada pihak perusahaan dan historical data dengan melihat data historis atau mengidentifikasi kondisi perusahaan melalui berinteraksi secara langsung dengan beberapa bagian yang terlibat. Peneliti mengumpulkan data yang dibutuhkan seperti titik koordinat pengantaran barang, batas waktu pengantaran, *delivery demand* pada setiap pelanggan, jumlah kendaraan, kapasitas maksimal kendaraan, rata rata kecepatan kendaraan, rata rata waktu pelayanan pada setiap *customer*, dan rasio penggunaan bahan bakar.
2. Data sekunder digunakan sebagai penunjang penelitian dan penguat deskripsi-deskripsi kualitatif dalam penelitian. Data sekunder yang digunakan dalam penelitian ini seperti profil perusahaan, dan harga bahan bakar minyak yang digunakan.

3.4 Metode Pengumpulan Data

Data dikumpulkan untuk mendapatkan informasi yang diperlukan dalam proses pengolahan data demi mencapai tujuan penelitian ini. Dalam melakukan pengumpulan data terdapat beberapa metode yang digunakan sesuai dengan kebutuhan peneliti dengan objek penelitian yang telah ditentukan. Berikut merupakan pengumpulan data yang dilakukan:

1. Studi Literatur

   Studi literatur yang digunakan bersumber dari jurnal dan buku sebagai landasan teori dan referensi penelitian digunakan untuk mengumpulkan data/referensi yang berkaitan dengan

topik penelitian yang terdapat dalam literatur, buku, jurnal, dan penelitian terkait lainnya. Studi literatur dapat bertujuan untuk memperkuat topik atau teori yang diterapkan oleh peneliti.

2. Wawancara

   Wawancara dilakukan bersama orang yang memiliki peran penting dalam permasalahan yang akan diatasi yang mana dianggap *expert* di perusahaan tersebut. Wawancara dilakukan secara bertahap bersama dengan *Coorporate Strategy* Manager PT. Pos Logistik Indonesia. Peneliti melakukan wawancara untuk mengidentifikasi permasalahan yang dihadapi oleh perusahaan saat ini. Selain itu, wawancara dilakukan untuk mendapatkan data data yang dibutuhkan dalam penelitian yang dilakukan seperti jumlah *customer*, jam kerja layanan perusahaan, dan data lainnya.

3.5 Populasi dan Sampel

3.5.1 Populasi

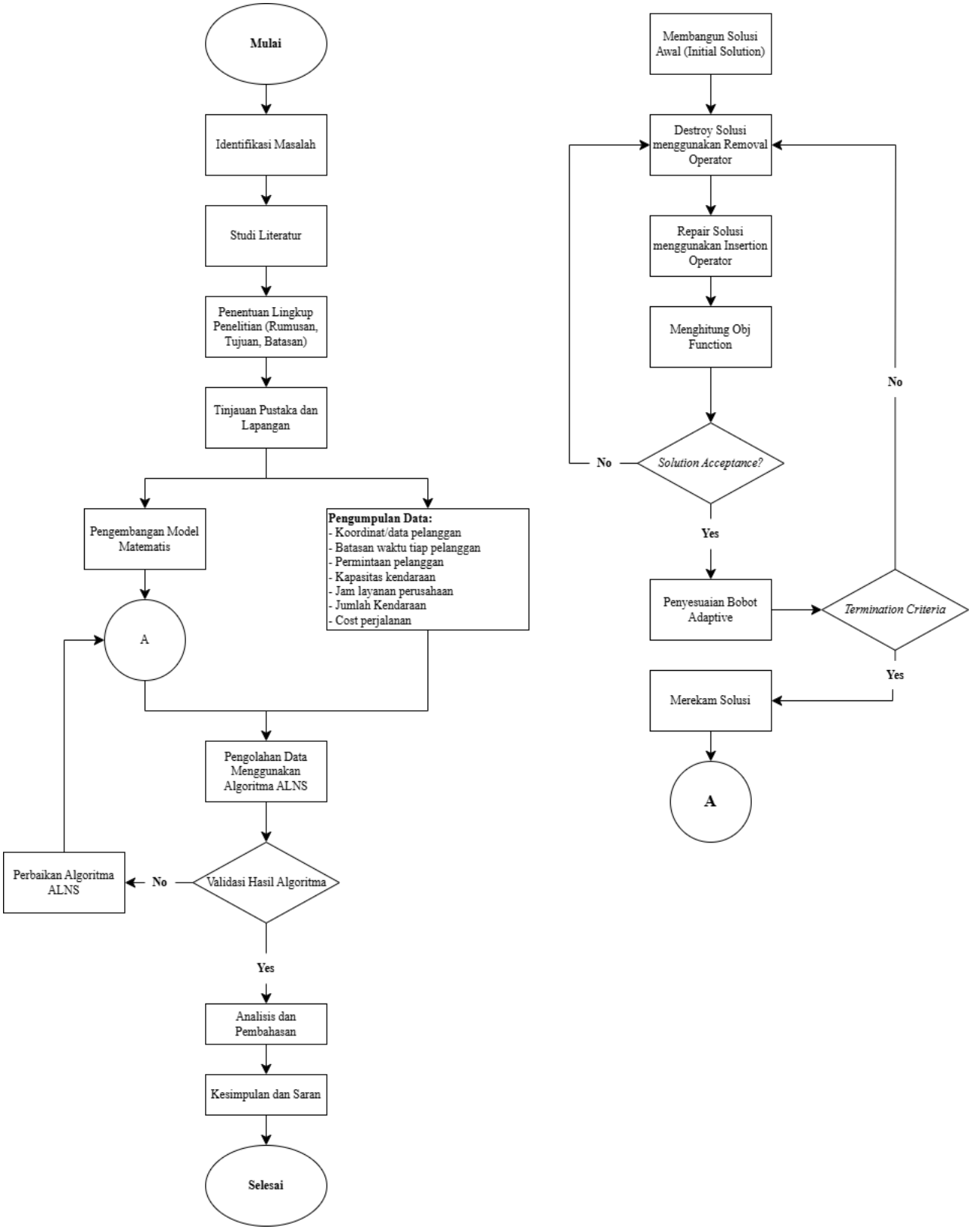
Dalam penelitian yang dilakukan populasi yang digunakan yaitu data jumlah keseluruhan koordinat pengantaran produk PT. Pos Logistik Indonesia. Titik koordinat tersebut merupakan pelanggan atau *customer* yang menjadi tujuan pengantaran produk yang mana menjadi dasar dalam penentuan atau optimasi rute yang dilakukan. Populasi ini mencangkup seluruh Alamat pengiriman aktif yang tercatat dalam sistem perusahaan.

3.5.2 Sampel

Sampel yang digunakan dalam penelitian ini mencakup titik pengantaran produk yang tersebar di wilayah Jakarta, sebanyak 100 titik. Selain itu, menggunakan 1 *main hub* juga sebagai depot atau gudang utama yang berfungsi sebagai titik awal dana khir dari seluruh rute pengiriman. Pemilihan sampel ini digunakan untuk menggambarkan kondisi dilapangan serta mendukung proses simulasi optimasi rute sehingga optimasi rute yang dilakukan dalam skala yang terukur.

3.6 Alur Penelitian

Alur penelitian bertujuan untuk menjabarkan langkah-langkah peneliti dalam mengembangkan dan merancangan penelitian ini dari awal hingga akhir. Berikut adalah *flowchart* dari alur penelitian ini:



Gambar 3. 2 Alur Penelitian

Source: Pribadi

Berikut merupakan tahapan yang dilalui dalam melakukian penelitian ini:

1. Mulai
2. Identifikasi Masalah

   Pada tahap ini peneliti melakukan identifikasi masalah dengan melakukan wawancara sehingga dapat mengetahui permasalahan detail yang dihadapi PT. Pos Logistik Indonesia terutama pada bagian distribusi barang kepada pelanggan. Permasalahan yang dihadapi PT. Pos Logistik Indonesia yaitu terkait penentuan rute distribusi yang saat ini dilakukan hanya berdasarkan hasil evaluasi tahunan dan subjektivitas masing masing driver. Cara tersebut dianggap kurang efektif untuk digunakan sebagai alat pengambilan keputusan pada Tingkat operasional. Selain itu, optimasi rute ini diperlukan karena rute saat ini yang terbentuk memiliki jarak tempuh yang dianggap masih kurang optimal dikarenakan menyebabkan beberapa hal seperti jam lembur pegawai dan juga terkadang beberapa armada kembali ke *main hub* di luar jam layanan yang telah ditentukan. Dengan memiliki rute distribusi yang memiliki jarak tempuh yang pendek, setiap kendaraan diharapkan dapat melayani seluruh *customer* di jam layanan yang telah ditentukan.
3. Studi Literatur

   Tahap kedua melakukan studi literatur yang bertujuan untuk mempelajari dan mencari informasi lebih lanjut terkait permasalahan yang dihadapi perusahaan. Studi literatur ini digunakan untuk mengetahui metode yang dapat digunakan untuk memecahkan permasalahan yang ada. Studi literatur dapat berupa penelitian terdahulu, jurnal, atau buku terkait. Permasalahan yang dihadapi yaitu terkait penentuan rute distribusi barang yang terdapat beberapa batasan yang dipertimbangkan seperti kapasitas kendaraan yang digunakan dan batasan waktu di setiap pelanggan. Berdasarkan studi literatur yang dilakukan algoritma ALNS menjadi metode yang digunakan untuk menyelesaikan permasalahan optimasi rute di PT. Pos Logistik Indonesia.
4. Penentuan Lingkup Penelitian

   Tahap selanjutnya yaitu menentukan ruang lingkup penelitian meliputi rumusan masalah berdasarkan identifikasi masalah yang telah dilakukan sebelumnya dan kemudian dijadikan sebagai tujuan penelitian yang dilakukan. Selain itu, peneliti menentukan batasan masalah untuk membatasi penelitian agar dapat lebih terfokus pada tujuan yang ingin dicapai. Berdasarkan permasalahan yang dihadapi oleh PT. Pos Logistik Indonesia, ruang lingkup penelitian ini difokuskan pada minimasi jarak tempuh dalam proses distribusi produk

melalui perancangan rute usulan yang lebih optimal. Selain itu, penelitian ini juga menganalisis dampak dari rute usulan yang terbentuk dengan minimasi jarak terhadap biaya distribusi yang dikeluarkan perusahaan. Dalam proses optimasi yang dilakukan, terdapat beberapa batasan mencangkup kendala time windows yang harus dipenuhi pada setiap titik *customer* dan batasan terkait kapasitas muatan yang tidak boleh melebihi kapasitas maksimum dari kendaraan. Penelitian yang dilakukan menggunakan satu jenis kendaraan dengan membentuk rute usulan dari 1 *main hub* dan 100 titik *customer* di wilayah Jakarta.

5. Tinjauan Pustaka dan Lapangan
   a. Pengumpulan Data

      Pengumpulan data dilakukan melalui dua teknik yaitu studi literatur dan wawancara secara langsung kepada *expert* perusahaan. Data yang dikumpulkan meliputi titik koordinat pengantaran barang, batas waktu pengantaran, jumlah permintaan pelanggan, jenis kendaraan, jumlah kendaraan, kapasitas maksimal kendaraan, rasio penggunaan bahan bakar, rata rata kecepatan kendaraan, dan waktu pelayanan pada setiap titik.

   b. Pengembangan Algoritma

      Pengembangan algoritma *Adaptive Large Neighborhood Search* (ALNS) untuk menyelesaikan permasalahan VRP yang dilakukan yaitu menambahkan constraint atau batasan pada algoritma meliputi batasan time windows, yang mana time windows pada setiap *customer* harus dipenuhi, dan batasan kapasitas kendaraan yang mana total muatan yang diangkut pada setiap kendaraan tidak diperbolehkan melebihi kapasitas maksimum kendaraan yang telah ditentukan. Selain itu, pengembangan algoritma dilakukan dengan menentukan *destroy repair* operator yang digunakan yaitu untuk *destroy* operator menggunakan *random removal* dan *worst removal*, sedangkan *repair* operator menggunakan *greedy insertion* dan *regret insertion*. Pemilihan operator tersebut didasari oleh kajian pustaka yang telah dilakukan terhadap penelitian terdahulu yang menggunakan algoritma yang sama.

6. Pengolahan Data

   Dari data yang sudah dikumpulkan, peneliti menggunakan *Python* untuk menentukan rute distrubusi barang yang paling optimal dengan menerapkan algoritma ALNS yang bekerja menggukan sistem *destroy repair* yang telah dibangun sehingga dapat menghasilkan solusi

yang lebih baik dari solusi awal. Selain itu, penggunaan algoritma ini mempertimbangkan batasan berupa kapasitas kendaraan yang digunakan dan jendela waktu yang diperbolehkan pada setiap pelanggan. Berikut merupakan langkah langkah dalam penentuan rute distribusi usulan menggunakan algoritma ALNS:

1). Input Data

Pada tahap ini dilakukan input data yang digunakan dalam optimassi penentuan rute PT. Pos Logistik Indonesia menggunakan algoritmaa ALNS. Data yang dimasukan meliputi matriks jarak yang menunjukan jarak tempuh antar *customer*, time windows, *delivery demand* , kecepatan rata rata kendaraan, *service time*, kapasitas maksimal kendaraan, jumlah kendaraan, dan jumlah *customer*.

2). Inisialisasi nilai setiap parameter ALNS

Pada tahap ini dilakukan penentuan setiap nilai dari parameter yang digunakan di dalam algoritma ALNS. Paramater tersebut diantaranya bobot *destroy* operator, bobot *repair* operator, interval pembaruan bobot, nilai sigma 1 (σ_1), sigma 2 (σ_2), sigma 3 (σ_3), *initial temperature* (T_0), *dan cooling rate* (α).

3). Membangun Solusi awal

Pada tahap ini dilakukan dengan membentuk Solusi feasible yang akan menjadi basis awal dalam pencarian rute usulan yang optimal. Solusi ini akan disimpan sebagai bentuk solusi terbaik sementara.

4). Pemilihan *Destroy- Repair* Operator

Setiap proses iterasi yang dilakukan, algoritma ALNS akan melakukan pemilihan operator menggunakan mekanisme *roulette wheel selection*, berdasarkan bobot probabilitas yang menunjukan performa dari setiap operator.

5). *Destroy* Solusi

Pada tahapan ini dilakukan proses penghancuran solusi awal dengan menggunakan beberapa *destroy* operator yaitu *random removal* dan *worst removal*. *Random removal* bekerja dengan menghapus Sebagian elemen pada solusi awal yang mana pada permasalahan VRP yaitu menghilangkan beberapa titik atau node secara acak untuk menciptakan ruang eksplorasi baru. *Worst removal* bekerja dengan menghapus elemen yang memiliki kontribusi paling merugikan yaitu yang memiliki total biaya yang tinggi dalam rute yang terbentuk. Hasil dari proses *destroy* ini akan membentuk rute atau

solusiyang belum lengkap dan perlu di perbaiki di proses *repair* menggunakan *repair* operator.

6). *Repair* Solusi

Solusi atau rute yang belum lengkap pada tahapan ini dilakukan proses *repair* menggunakan beberapa *repair* operator yaitu *greedy insertion* dan *regret insertion*. *Greedy Insertion* bekerja dengan memperbaiki solusi dengan menambahkan kembali titik *customer* yang dihapus dengan titik yang memiliki nilai peningkatan biaya/jarak sekecil mungkin hingga semua titik berhasil disisipkan kembali. *Regret Insertion* bekerja dengan menambahkan titik *customer* yang dihapus berdasarkan risiko biaya penyisipan yang besar sehingga titk yang memiliki nilai regret terbesar akan disisipkan terlebih dahulu pada rute yang terbentuk.

7). *Acceptance Criteria*

Pada tahapan ini Solusi baru yang dihasilkan pada proses sebelumnya akan melalui *Acceptance Criteria* yang menggunakan *metropolis criteria*. Hal ini berujuan untuk menghindari Solusi terjebak dalam solusi local sehingga tetap memberikan peluang Solusi yang lebih buruk untuk dapat diterima. Menerima solusi baru S' jika $f(S') < f(S)$. Akan tetapi, jika $f(S') \geq f(S)$ yang menunjukan solusi lebih buruk maka akan dihitung peluangnya.

8). *Adaptive Weight Adjustment*

Pada tahapan ini, setiap n iterasi yang telah ditentukan akan melakukan proses penyesuaian bobot operator berdasarkan kinerja selana n iterasi sebelumnya. Operator yang menghasilkan solusi berkualitas lebih baik secara konsisten akan mendapatkan bobot yang lebih besar sehingga lebih sering terpilih pada iterasi selanjutnya. Jika proses ini telah dilakukan akan kembali lagi pada proses (4) dan dilakukan secara berulang.

9). *Termination Criteria*

Algoritma ALNS akan berhenti jika mencapai batas iterasi maksimum yang telah ditentukan. Solusi terbaik selama proses akan di simpan sebagai solusi akhir dari algoritma yang membentuk rute usulan pada setiap kendaraan.

7. Validasi Hasil Algoritma

Tahap selanjutnya yaitu tahap validasi hasil algoritma ALNS pada permasalahan VRP (CVRPTW). Validasi ini dilakukan untuk memastikan algoritma yang telah

diimplementasikan mampu memberikan solusi yang efektif, dan stabil. Proses validasi yang dilakukan dalam penelitian ini yaitu dengan membandingkan Solusi akhir yang terbentuk dengan solusi awal untuk melihat sejauh mana perbaikan yang dilakukan. Selain itu, proses validasi dilakukan dengan melihat waktu komputasi pada algoritma dalam menyelesaikan permasalahan optimasi yang dilakukan dengan beberapa kali pengujian (multiple runs)

8. Analisis dan pembahasan

   Pada tahap ini peneliti akan melakukan analisis data dengan pembahasan lebih rinci berdasarkan hasil penelitian atau pengolahan data yang telah dilakukan, dimana pada tahapan ini dilakukan analisis dengan melakukan perbandingan antara rute awal dengan rute usulan yang terbentuk untuk mengetahui minimasi jarak yang diperoleh menggunakan algoritma ALNS. Analisis juga dilakukan untuk mengetahui dampak minimasi jarak yang ada terhadap penurunan biaya distribusi pada PT. Pos Logistik Indonesia. Dengan demikian, dapat dikehatui apakah algoritma yang talah dilakukan untuk menentukan rute usulan mampu memberikan solusi yang optimal.

9. Kesimpulan dan Saran

   Pada tahap akhir ini peneliti akan membuat kesimpulan berdasarkan analisis data yang telah dilakukan menggunakan algoritma ALNS. Pada tahap ini juga peneliti akan menjawab rumusan masalah yang telah dibuat sebelumnya melalui kesimpulan penelitian. Selain itu, pada bagian saran peneliti akan memberikan beberapa saran yang digunakan untuk PT. Pos Logistik Indonesia berdasarkan penelitian yang telah dilakukan dan saran untuk penelitian selanjutnya yang menggunakan algoritma ALNS pada penentuan rute distribuasi khususnya pada perusahaan yang bergerak dibidang *freight forwarding.*

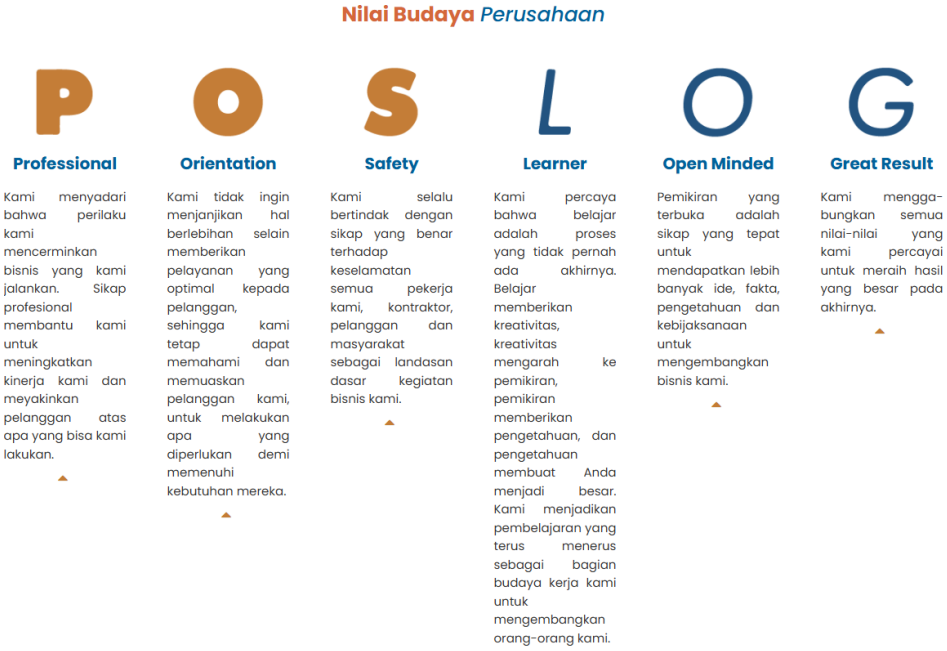
10. Selesai

BAB IV
PENGUMPULAN DAN PENGOLAHAN DATA

4.1 Pengumpulan Data

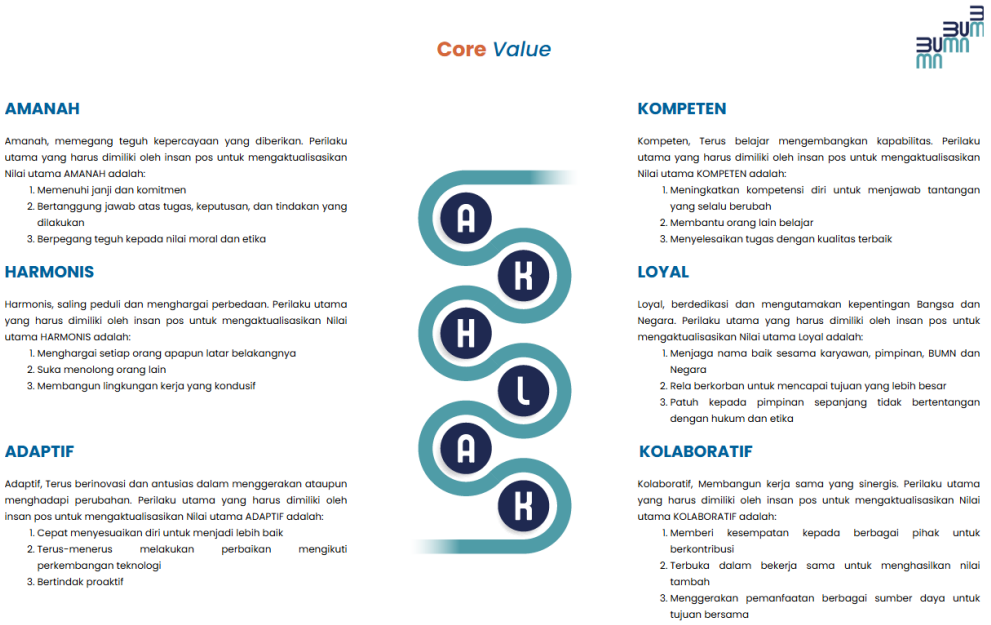
4.1.1 Profil Perusahaan

PT. Pos Logistik Indonesia (POSLOG) merupakan salah satu anak perusahaan dari PT. Pos Indonesia (Persero) yang berdiri pada tahun 2012 yang berawal dari proyek bisnis. PT. Pos Logistik Indonesia salah satu perusahaan yang bergerak di bidang freight forwarding sebagai penyedia layanan logistic. Dalam menghadapi tantangan industry logistic yang ada di Indonesia maupun International, PT. Pos Logistik Indonesia terdapat beberapa layanagan yang dapat digunakan yaitu *warehouse*, *distribution*, *heavy lifting*, *domestic cargo*, *international cargo*, dan *cold chain service.* Dalam menjalankan bisnisnya, PT. Pos Logistis Indonesia mempunyai niolai budaya yang diterapkan untuk memperkuat lini bisnis nya. Nilai budaya yang ada diperusahaan antara lain *professional*, *orientation*, *safety*, *learner*, *open minded*, dan *great result.*



Gambar 4. 1 Nilai Budaya Perusahaan

Selain nilai budaya yang diterapkan perusahaan, PT. Pos Logistik Indonesia juga mempunyai *core value* yang menjadi nilai tambah untuk dapat bersaing dengan perusahaan penyedia layanan logistic lainnya. *Core value* yang dimiliki poslog yaitu amanah, kompoten, harmonis, loyal, adaptif, dan kolaboratif.



Gambar 4. 2 *Core value*

4.1.2 Visi Misi Perusahaan



Gambar 4. 3 Logo PT. Pos Logistik Indonesia

PT. Pos Logistik Indonesia mempunya visi misi yang untuk mendukung perusahaan sehingga dapat merumuskan strategi dan positioning bisnis menjadi Indonesia's #1 Logistics. Berikut merupakan visi misi dari perusahaan:

Visi:

"Menjadi penyedia solusi logistik terpadu yang terpercaya, terluas, dan terkemuka di Indonesia"

Misi:

1. Memberikan solusi logistik yang efisien dan terintegrasi bagi pelanggan serta mendukung daya saing logistic nasional.
2. Memberikan kontribusi laba yang maksimal dan membangun sinergi usaha dengan PT Pos Indonesia.

3. Membangun kemitraan usah a dengan mitra kerja strategis yang saling menguntungkan.
4. Terus berupaya mengembangkan kompetensi karyawan dan organisasi agar memiliki daya saing nasional.

4.1.3 Formulasi Masalah

Pada penelitian yang dilakukan yaitu menyelesaikan permasalahan dalam penentuan rute distribusi produk PT. Pos Logistik Indonesia yang mana untuk mengurangi biaya yang dikeluarkan. Permasalahan penentuan rute ini memperhatikan tingkat ketepatan waktu pada setiap *customer* dan juga yang ada di daerah Jakarta yang mana setiap *customer* harus dilayani sesuai jendela waktu yang telah ditentukan. Selain itu, terbatasnya jumlah kapasitas pada setiap kendaraan (k = 1000kg.) menjadi salah satu batasan yang menjadi pertimbangan dalam penentuan rute ini. Setiap kendaraan tidak diperbolehkan mengangkut produk melebih kapasitasnya sehingga diperlukan pengoptimalan rute sesuai dengan permintaan dari setiap *customer* agar tidak melebih kapasitas yang ada. Penentuan rute yang optimal akan dilihat dari pengurangan jarak pada rute distribusi masing masing kendaraan, sehingga minimasi jarak yang dilakukan akan berdampak pada pengurangan biaya distribusi yang dikeluarkan PT. Pos Logistik Indonesia. Penelitian yang dilakukan menggunakan 1 *main hub* dan 100 titik yang di simbolkan dengan node (i = 0, 1,2,3,….n) yang mana i = 0 menunjukan *main hub* dan i = 1,2,3,..,n menunjukan *customer* di daerah Jakarta.

Bedasarkan masalah yang dihadapi dalam penelitian ini masuk pada jenis kasus penentuan rute sering dikategorikan CVRPTW. Dengan demikian, algoritma yang digunakan dalam penelitian ini mampu menyelesaikan permasalahan dengan membentuk rute untuk jumlah *customer* yang relative banyak. Algoritma yang digunakan juga terfokus pada penentuan rute terpendek pada setiap kendaraan dengan waktu komputasi yang relatif singkat yaitu dengan menggunakan algoritma *metaheuristic* ALNS.

4.1.4 Model Matematis

Model matematis yang digunakan disesuaikan dengan formulasi permasalahan yang ada yaitu penentuan rute optimal dengan jenis CVRPTW. Model matematis ini digunakan untuk menentukan fungsi tujuan dan fungsi kendala yang akan dibentuk dalam algoritma penyelesaian dalam kasus ini. Adapun fungsi tujuan dalam model matematis ini adalah ini yaitu untuk meminimalkan atau mencari jarak paling pendek pada masing masing rute yang terbentuk di setiap kendaraan. Berdasarkan fungsi tujuan tersebut, terdapat beberapa hal yang digunakan sebagai acuan dalam penelitian yang dilakukan:

1. Setiap rute yang terbentuk dimulai dan berakhir pada satu depot (*main hub*) yang sama.
2. Pada rute kendaraan yang terbentuk dapat melayani beberapa *customer*.
3. Kendaraan harus mengunjungi setiap *customer* yang ada berdasarkan data yang diberikan.
4. Setiap *customer* dikunjungi tepat satu kali oleh satu kendaraan.
5. Setiap *customer* memiliki jangka waktu yang telah ditentukan untuk melakukan pengiriman produk.
6. Seluruh *customer* memiliki jumlah waktu layanan yang sama.
7. Jenis kendaraan yang dalam proses distribusi yang dilakukan menggunakan jenis kendaraan BlindVan.

Dalam pengembangan model algoritma CVRPTW menggunakan algoritma ALNS, terdapat beberapa notasi, parameter, variable keputusan, dan performance measure yang akan mempermudah dalam pembuatan model matematis yang sesuai dengan kasus yang ada. Berikut merupakan notasi, parameter, variable Keputusan, dan *performance measure*:

a. Notasi
   - V : Himpunan kendaraan $(k \in V)$
   - C : Himpunan pelanggan termasuk $(i \in C)$; i = 1,2,3,….n.
   - N : Himpunan semua simpul (termasuk *main hub*, dan pelanggan)
   - 0 : Simpul *main hub* (titik awal dan akhir rute)
   - $n+1$: Simpul *main hub* dummy (untuk kendaraan kembali)

b. Parameter
   - d_{ij} : Jarak dari node i ke j
   - t_{ij} : Waktu perjalanan dari node i ke j
   - q_k : Kapasitas maksimum kendaraan k
   - d_i : Permintaan delivery (barang dikirim) di simpul i (kg)
   - a_i : Waktu mulai layanan tercepat (*earliest service time*) di simpul i
   - b_i : Waktu mulai layanan terlambat (*latest service time*) di simpul i

c. *Decision Variable*
   - s_{ik} : Waktu kendaraan k mulai melayani simpul i
   - L_{ik} : Muatan kendaraan k setelah meninggalkan simpul i
   - x_{ijk} : Bernilai 1 apabila terdapat perjalanan dari titik i ke j dengan kendaraan k. Bernilai 0 apabila tidak terdapat perjalanan dari simpul i ke j dengan kendaraan k

d. *Performance Measure*

- Total jarak tempuh setiap kendaraan
- Total waktu layanan pelanggan
- Total kapasitas kendaraan yang terpakai

Berdasarkan data tersebut, maka diperoleh model matematis yang terdiri dari fungsi tujuan dan fungsi kendala yang digunakan dalam penelitian ini. Berikut merupakan model matematis untuk model CVRPTW dalam permasalahan distribusi PT. Pos Logistik Indonesia:

Fungsi Tujuan:

$$\min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ijk} \qquad (4.1)$$

Fungsi tujuan dari model matematis ini yaitu meminimalkan total jarak distribusi produk ke *customer* dari i ke j.

Fungsi Kendala:

1. Kendala 1: Setiap pelanggan dikunjungi tepat satu kali oleh salah satu kendaraan, tanpa ada yang terlewat atau dikunjungi lebih dari sekali.

$$\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1 \qquad \forall i \in C \qquad (4.2)$$

2. Kendala 2: Total muatan pada rute yang terbentuk di setiap kendaraan tidak diperbolehkan melebihi kapasitas kendaraan yang telah ditentukan.

$$\sum_{i \in C} d_i \sum_{j \in N} x_{ijk} \leq q \qquad \forall k \in V \qquad (4.3)$$

3. Kendala 3: Setiap kendaraan harus berangkat dari gudang (depot) sebagai titik awal dari rute distribusi

$$\sum_{j \in N} x_{0jk} = 1 \qquad \forall k \in V \qquad (4.4)$$

4. Kendala 4: Setelah kendaraan mengantarkan barang ke satu pelanggan, kemudia kendaraan tersebut melanjutkan perjalanan ke tujuan berikutnya

$$\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{hjk} = 0 \qquad \forall h \in C, \ \forall k \in V \qquad (4.5)$$

5. Kendala 5: Semua kendaraan harus kembali ke gudang atau depot sebagai titik akhir setelah selesai mengantarkan barang.

$$\sum_{i \in N} x_{i,n+1,k} = 1 \qquad \forall k \in V \qquad (4.6)$$

6. Kendala 6: Mengatur hubungan waktu antar kunjungan dengan memastikan kendaraan memperhitungkan waktu perjalanan dari satu pelanggan ke pelanggan berikutnya

$$x_{ijk}(s_{ik} + t_{ij} - s_{jk}) \leq 0 \qquad \forall i, j \in N, \forall k \in V \qquad (4.7)$$

7. Kendala 7: Kendaraan wajib mengikuti jendela waktu (*time windows*) yang ditetapkan oleh pelanggan, sehingga pengantaran dilakukan dalam rentang waktu yang diizinkan

$$a_i \leq s_{ik} \leq b_i \qquad \forall i \in N, \forall k \in V \qquad (4.8)$$

8. Kendala 8: memastikan bahwa semua variabel dalam model, seperti jumlah kendaraan atau keputusan rute, berbentuk angka bulat (*integer*).

$$x_{ijk} \in \{0,1\} \qquad \forall k \in V, \forall (i,j) \in N \qquad (4.9)$$

4.1.5 Data Penelitian

Data yang digunakan dalam pengolahan CVRPTW dengan menggunakan algoritma ALNS ini meliputi beberapa data seperti titik koordinat dari *main hub*, *customer*, *time windows* pada masing masing node, jumlah kendaraan, kapasitas dari masing masing kendaraan, *delivery demand*, *service time*, dan kecepatan rata rata kendaraan. Koordinat titik ini digunakan sebagai input untuk membuat matriks jarak yang akan digunakan dalam mencari jarak terdekat dan penentuan rute dengan mempertimbangkan *time windows* pada masing masing *customer*.

Titik koordinat yang digunakan dalam penelitian ini yaitu *customer* PT. Pos Logistik Indonesia yang berada di daerah Jakarta yang meliputi 1 *main hub* dan 100 *customer*. Berikut merupakan data titik koordinat dari setiap node:

Tabel 4. 1 Titik Koordinat UTM

Nama Node	**ID *Number***	***Decimal Degrees***		**UTM**	
		Latitude	***Longitude***	***East* (X)**	***North* (Y)**
Main hub	0	-6.20241	106.91688	712.0994	9314.0371
Customer 1	1	-6.20005	106.86950	706.8569	9314.3158
Customer 2	2	-6.26438	106.84996	704.6687	9307.2094
Customer 3	3	-6.26062	106.84947	704.6162	9307.6256
Customer 4	4	-6.266101	106.87786	707.7563	9307.0083
Customer 5	5	-6.201513	106.85215	704.9349	9314.1617
Customer 6	6	-6.210823	106.94136	714.8056	9313.0967
Customer 7	7	-6.281572	106.88730	708.7944	9305.2933

Nama Node	**ID *Number***	***Decimal Degrees***		**UTM**	
		Latitude	***Longitude***	***East* (X)**	***North* (Y)**
Customer 8	8	-6.279968	106.81923	701.2618	9305.4974
Customer 9	9	-6.233342	106.85007	704.6932	9310.6423
Customer 10	10	-6.269443	106.77413	696.2748	9306.6785
Customer 11	11	-6.212111	106.80952	700.2131	9313.0059
Customer 12	12	-6.288054	106.79509	698.5876	9304.6123
Customer 13	13	-6.292434	106.84513	704.1230	9304.1086
Customer 14	14	-6.274435	106.81169	700.4294	9306.1122
Customer 15	15	-6.268981	106.93584	714.1719	9306.6663
Customer 16	16	-6.196040	106.78103	697.0662	9314.7940
Customer 17	17	-6.291090	106.88195	708.1989	9304.2428
Customer 18	18	-6.200544	106.79031	698.0917	9314.2924
Customer 19	19	-6.279059	106.81821	701.1493	9305.5983
Customer 20	20	-6.253398	106.98133	719.2122	9308.3712
Customer 21	21	-6.251575	106.73320	691.7517	9308.6698
Customer 22	22	-6.278261	106.75016	693.6183	9305.7122
Customer 23	23	-6.267833	106.79343	698.4114	9306.8493
Customer 24	24	-6.289595	106.89197	709.3081	9304.4041
Customer 25	25	-6.209064	106.89812	710.0204	9313.3086
Customer 26	26	-6.225719	106.88118	708.1395	9311.4732
Customer 27	27	-6.204504	106.90071	710.3094	9313.8119
Customer 28	28	-6.225583	107.00348	721.6762	9311.4385
Customer 29	29	-6.269489	106.80965	700.2058	9306.6600
Customer 30	30	-6.230990	106.82348	701.7506	9310.9127
Customer 31	31	-6.287387	106.88501	708.5390	9304.6511
Customer 32	32	-6.265529	106.86182	705.9810	9307.0778
Customer 33	33	-6.209836	106.79974	699.1314	9313.2612
Customer 34	34	-6.240286	106.84409	704.0288	9309.8766
Customer 35	35	-6.222611	106.881953	708.2254	9311.8167
Customer 36	36	-6.271854	106.756113	694.2795	9306.4186
Customer 37	37	-6.270860	106.880841	708.0833	9306.4807

Nama Node	**ID *Number***	***Decimal Degrees***		**UTM**	
		Latitude	***Longitude***	***East* (X)**	***North* (Y)**
Customer 38	38	-6.281991	106.896347	709.7949	9305.2434
Customer 39	39	-6.276553	106.792414	698.2950	9305.8853
Customer 40	40	-6.240736	106.739010	692.3982	9309.8664
Customer 41	41	-6.278588	106.956858	716.4932	9305.5951
Customer 42	42	-6.233353	106.905816	710.8624	9310.6191
Customer 43	43	-6.226198	106.93371	713.9527	9311.3992
Customer 44	44	-6.227234	106.79961	699.1100	9311.3376
Customer 45	45	-6.230924	106.750892	693.7169	9310.9472
Customer 46	46	-6.268196	106.714327	689.6566	9306.8385
Customer 47	47	-6.219344	106.825823	702.0143	9312.1998
Customer 48	48	-6.206014	106.796047	698.7236	9313.6853
Customer 49	49	-6.213234	106.894372	709.6038	9312.8488
Customer 50	50	-6.231975	106.816648	700.9940	9310.8063
Customer 51	51	-6.215032	106.796623	698.7841	9312.6877
Customer 52	52	-6.205822	106.964001	717.3137	9313.6406
Customer 53	53	-6.249451	106.737492	692.2271	9308.9031
Customer 54	54	-6.207175	106.828957	702.3657	9313.5445
Customer 55	55	-6.221041	106.828281	702.2857	9312.0112
Customer 56	56	-6.243603	106.835927	703.1232	9309.5128
Customer 57	57	-6.274117	106.972113	718.1835	9306.0833
Customer 58	58	-6.295155	106.789333	697.9470	9303.8291
Customer 59	59	-6.240121	106.822205	701.6058	9309.9032
Customer 60	60	-6.243571	106.954533	716.2504	9309.4692
Customer 61	61	-6.286299	106.796021	698.6905	9304.8060
Customer 62	62	-6.254802	106.784211	697.3954	9308.2939
Customer 63	63	-6.298913	106.842677	703.8488	9303.3930
Customer 64	64	-6.282974	106.808191	700.0385	9305.1692
Customer 65	65	-6.281341	106.928593	713.3640	9305.3023
Customer 66	66	-6.253956	106.810636	700.3202	9308.3775
Customer 67	67	-6.267852	106.905446	710.8076	9306.8035

Nama Node	**ID *Number***	***Decimal Degrees***		**UTM**	
		Latitude	***Longitude***	***East* (X)**	***North* (Y)**
Customer 68	68	-6.264153	106.842021	703.7897	9307.2377
Customer 69	69	-6.296323	106.866565	706.4933	9303.6700
Customer 70	70	-6.206968	106.987402	719.9037	9313.5041
Customer 71	71	-6.257163	106.763728	695.1277	9308.0405
Customer 72	72	-6.262841	106.802915	699.4623	9307.3978
Customer 73	73	-6.268792	106.957371	716.5540	9306.6784
Customer 74	74	-6.245723	106.968306	717.7739	9309.2255
Customer 75	75	-6.266832	106.968362	717.7714	9306.8905
Customer 76	76	-6.245014	106.961681	717.0409	9309.3066
Customer 77	77	-6.259981	106.894461	709.5949	9307.6784
Customer 78	78	-6.273181	106.919096	712.3162	9306.2086
Customer 79	79	-6.224792	106.920733	712.5169	9311.5600
Customer 80	80	-6.207003	106.924558	712.9475	9313.5260
Customer 81	81	-6.230163	106.763518	695.1145	9311.0266
Customer 82	82	-6.253615	106.738448	692.3313	9308.4422
Customer 83	83	-6.297994	106.930443	713.5619	9303.4595
Customer 84	84	-6.246302	106.818465	701.1896	9309.2210
Customer 85	85	-6.218771	106.885036	708.5682	9312.2402
Customer 86	86	-6.278014	106.825127	701.9146	9305.7112
Customer 87	87	-6.271007	106.754148	694.0623	9306.5130
Customer 88	88	-6.250797	106.797788	698.8994	9308.7318
Customer 89	89	-6.202589	106.850735	704.7780	9314.0433
Customer 90	90	-6.295848	106.804147	699.5861	9303.7468
Customer 91	91	-6.294899	106.854689	705.1796	9303.8322
Customer 92	92	-6.208897	106.728317	691.2264	9313.3913
Customer 93	93	-6.231823	106.916202	712.0125	9310.7841
Customer 94	94	-6.212998	106.903571	710.6221	9312.8713
Customer 95	95	-6.252885	106.896008	709.7691	9308.4627
Customer 96	96	-6.182069	106.702447	688.3727	9316.3676
Customer 97	97	-6.280521	106.902647	710.4928	9305.4035

Nama Node	ID *Number*	*Decimal Degrees*		UTM	
		Latitude	***Longitude***	***East* (X)**	***North* (Y)**
Customer 98	98	-6.254798	106.836067	703.1344	9308.2746
Customer 99	99	-6.241405	106.738714	692.3652	9309.7924
Customer 100	100	-6.270141	106.866163	706.4591	9306.5659

Selain koordinat pada main, dan *customer*, dalam penelitian ini juga terdapat batasan waktu pelayanan yang telah dipilih oleh setiap *customer* berdasarkan pilihan waktu pengantaran yang disediakan oleh PT. Pos Logistik Indonesia. Berikut merupakan waktu pelayanan/*time windows* yang disediakan oleh poslog:

Tabel 4. 2 Tipe *Time windows*

***Time windows* 1**	08.00 WIB - 13.00 WIB
***Time windows* 2**	13.01 WIB -17.00 WIB
***Time windows* 3**	17.01 WIB - 21.00 WIB

Selain itu, pada setiap *customer* juga memiliki data *delivery demand*. Berikut merupakan data *earliest time* dan *latest time* beserta dengan *demand* dari masing masing node yang digunakan dalam penelitian ini:

Tabel 4. 3 *Time windows* dan *Demand*

ID *Number*	*Time Windows*	*Earliest Time*	*Latest Time*	*Delivery Demand* (kg)
0	-	-	-	-
1	1	0	300	20
2	2	301	540	26
3	3	541	780	20
4	3	541	780	27
5	3	541	780	11
6	1	0	300	5
7	2	301	540	22
8	3	541	780	13
9	2	301	540	21
10	3	541	780	23
11	1	0	300	7
12	1	0	300	15
13	3	541	780	16
14	2	301	540	29
15	1	0	300	3

ID *Number*	*Time Windows*	*Earliest Time*	*Latest Time*	*Delivery Demand* (kg)
16	2	301	540	26
17	2	301	540	28
18	3	541	780	27
19	2	301	540	11
20	2	301	540	14
21	3	541	780	18
22	3	541	780	13
23	1	0	300	30
24	3	541	780	21
25	3	541	780	24
26	2	301	540	1
27	2	301	540	10
28	1	0	300	30
29	2	301	540	26
30	3	541	780	12
31	2	301	540	13
32	2	301	540	30
33	3	541	780	10
34	1	0	300	22
35	2	301	540	12
36	2	301	540	17
37	2	301	540	27
38	3	541	780	1
39	3	541	780	14
40	2	301	540	3
41	2	301	540	28
42	2	301	540	17
43	2	301	540	3
44	3	541	780	23
45	1	0	300	11
46	2	301	540	18
47	3	541	780	30
48	3	541	780	30
49	3	541	780	7
50	3	541	780	25
51	3	541	780	7
52	1	0	300	8
53	2	301	540	3
54	3	541	780	11

ID *Number*	*Time Windows*	*Earliest Time*	*Latest Time*	*Delivery Demand* (kg)
55	1	0	300	11
56	1	0	300	27
57	2	301	540	16
58	1	0	300	30
59	1	0	300	9
60	1	0	300	4
61	2	301	540	26
62	2	301	540	21
63	2	301	540	17
64	2	301	540	20
65	1	0	300	29
66	1	0	300	6
67	1	0	300	25
68	1	0	300	4
69	1	0	300	27
70	1	0	300	19
71	2	301	540	2
72	1	0	300	1
73	3	541	780	23
74	2	301	540	13
75	2	301	540	27
76	2	301	540	25
77	2	301	540	15
78	1	0	300	6
79	1	0	300	30
80	3	541	780	18
81	1	0	300	12
82	1	0	300	25
83	1	0	300	26
84	3	541	780	24
85	3	541	780	8
86	3	541	780	5
87	3	541	780	2
88	2	301	540	1
89	3	541	780	20
90	2	301	540	29
91	3	541	780	17
92	2	301	540	2
93	3	541	780	13

ID *Number*	*Time Windows*	*Earliest Time*	*Latest Time*	*Delivery Demand* (kg)
94	1	0	300	24
95	2	301	540	20
96	3	541	780	2
97	2	301	540	1
…	…	…	…	…
99	3	541	780	15
100	1	0	300	20

Berdasarkan tabel 4. merupakan batasan waktu pelayanan (*Time windows*) yang telah ditetapkan oleh *main hub*, dan *customer* sehingga proses pendistribusian produk harus sesuai Batasan waktu yang diperbolehkan. Pada proses distribusi juga PT. Pos Logistik Indonesia mempunyai rata rata waktu pelayanan pada masing masing *customer* selama 10 menit terhitung dari awal waktu kedatangan hingga meninggalkan titik tersebut. Selain itu terdapat data lainnya yang mendukung pengolahan data pada penelitian ini, sebagai berikut:

Tabel 4. 4 Data Pendukung

Jenis Kendaraan	Mobil BlindVan
Jumlah Kendaraan	3 unit
Kapasitas Kendaraan	1000kg
Rata Rata Kecepatan Kendaraan	22km/jam
Waktu Layanan Rata Rata	5 menit/*customer*
Rasio Konsumsi BBM	1: 11
Harga BBM/km	Rp909.09

4.2 Pengolahan Data

4.2.1 Rute Perusahaan

Berdasarkan pengumpulan data yang telah dilakukan, dari sampel *customer* yang diambil pada PT. Pos Logistik Indonesia, diperoleh rute distribusi awal yang telah ditentukan perusahaan sebagai berikut:

Tabel 4. 5 Rute Awal (Kebijakan Perusahaan)

Rute ke-	Rute	Total Jarak
Rute 1	0 – 80 – 25 - 6 – 42 – 26 – 95 – 77 - 67 – 70 – 4 – 97 – 20 – 32 – 41 – 28 – 56 – 2 – 98 – 30 – 69 – 91	205,58 km

Rute ke-	Rute	Total Jarak
Rute 2	– 66 – 48 – 8 – 29 – 72 – 23 – 90 – 81 – 71 – 87 – 22 – 92 – 96 – 0 0 – 27 – 79 – 43 – 85 – 52 – 60 – 5 – 74 – 78 – 37 – 65 – 38 – 100 – 54 – 55 – 24 – 47 – 83 – 59 – 11 – 13 – 86 – 51 – 19 – 14 – 64 – 62 – 61 – 10 – 45 - 40 – 53 – 21 – 0	205,76 km
Rute 3	0 – 94 – 49 – 93 – 35 – 1 – 76 – 89 – 15 – 9 – 73 – 34 – 75 – 7 – 3 – 57 – 31 – 17 – 68 – 50 – 84 – 33 – 44 – 63 – 18 – 88 – 16 – 39 – 12 – 58 – 36 – 99 – 82 – 46 – 0	236,06 km

Proses distribusi produk dilakukan dengan total menggunakan 3 unit armada BlindVan, dengan total jarak rute 1 sebesar 205,58 km, rute 2 sebesar 205,76 km, dan rute 3 sebesar 236,06 km dengan total jarak keseluruhan sebesar 647,4 km.

4.2.2 Matriks Jarak

Berdasarkan hasil pengumpulan data yang dilakukan, terdapat 101 titik koordinat yang meliputi 1 *main hub* dan 100 *customer* pada PT. Pos Logistik Indonesia. Data tersebut digunakan untuk mencari jarak antar dua titik sehingga dihasilkan matriks jarak dari keseluruhan titik koordinat yang ada. Matriks ini digunakan sebagai acuan untuk menentukan rute dengan jarak terpendek dari masing masing kendaraan yang digunakan. Selain itu, matriks jarak juga digunakan untuk menentukan waktu tempuh yang dibutuhkan untuk setiap kendaraan melalui titik pada masing masing rute sehingga dapat mengetahui waktu kendaraan tiba di titik tertentu. Hal tersebut digunakan untuk memastikan setiap kendaraan tiba di titik tertentu sesuai dengan time windows yang telah ditentukan. Dalam pengolahan data titik koordinat menjadi matriks jarak menggunakan rumus sebagai berikut:

$$s = \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)} \quad (4.10)$$

Berikut merupakan matriks jarak dari keseluruhan titik koordinat yang meliputi *main hub* dan *customer* dari PT. Pos Logistik Indonesia:

Tabel 4. 6 Matriks Jarak

i/j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	n	100
0	0.00	5.25	10.09	6.04	8.26	7.17	2.87	9.35	13.80	8.15	17.45	11.93	16.47	12.74	14.11	7.66	…	9.36
1	5.25	0.00	7.44	6.54	7.36	1.93	8.04	9.23	10.44	4.26	13.05	6.77	12.75	10.57	10.42	10.58	…	7.76
2	10.09	7.44	0.00	5.25	3.09	6.96	11.72	4.55	3.81	3.43	8.41	7.31	6.61	3.15	4.38	9.52	…	1.90
3	6.04	6.54	5.25	0.00	2.48	7.47	6.84	3.32	9.01	5.52	13.61	10.58	11.83	7.13	9.63	4.76	…	3.81
4	8.26	7.36	3.09	2.48	0.00	7.69	9.31	2.00	6.67	4.75	11.49	9.64	9.48	4.65	7.38	6.42	…	1.37
5	7.17	1.93	6.96	7.47	7.69	0.00	9.93	9.67	9.41	3.53	11.45	4.86	11.47	10.09	9.22	11.90	…	7.75
6	2.87	8.04	11.72	6.84	9.31	9.93	0.00	9.85	15.53	10.41	19.61	14.59	18.30	13.96	15.98	6.46	…	10.60
7	9.35	9.23	4.55	3.32	2.00	9.67	9.85	0.00	7.54	6.74	12.60	11.54	10.23	4.82	8.40	5.55	…	2.66
8	13.80	10.44	3.81	9.01	6.67	9.41	15.53	7.54	0.00	6.18	5.13	7.58	2.82	3.18	1.03	12.96	…	5.31
9	8.15	4.26	3.43	5.52	4.75	3.53	10.41	6.74	6.18	0.00	9.30	5.07	8.58	6.56	6.22	10.28	…	4.44
10	17.45	13.05	8.41	13.61	11.49	11.45	19.61	12.60	5.13	9.30	0.00	7.45	3.10	8.26	4.19	17.90	…	10.18
11	11.93	6.77	7.31	10.58	9.64	4.86	14.59	11.54	7.58	5.07	7.45	0.00	8.55	9.72	6.90	15.33	…	8.97
12	16.47	12.75	6.61	11.83	9.48	11.47	18.30	10.23	2.82	8.58	3.10	8.55	0.00	5.56	2.38	15.72	…	8.11
13	12.74	10.57	3.15	7.13	4.65	10.09	13.96	4.82	3.18	6.56	8.26	9.72	5.56	0.00	4.20	10.37	…	3.39
14	14.11	10.42	4.38	9.63	7.38	9.22	15.98	8.40	1.03	6.22	4.19	6.90	2.38	4.20	0.00	13.75	…	6.05
15	7.66	10.58	9.52	4.76	6.42	11.90	6.46	5.55	12.96	10.28	17.90	15.33	15.72	10.37	13.75	0.00	…	7.71
n	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
100	9.36	7.76	1.90	3.81	1.37	7.75	10.60	2.66	5.31	4.44	10.18	8.97	8.11	3.39	6.05	7.71	…	0.00

4.2.3 Waktu Tempuh

Dalam menentukan rute distribusi produk dari PT. Pos Logistik Indonesia, Waktu tempuh digunakan untuk memngetahui waktu total perjalanan dari antar titik sehingga dapat melihat apakah waktu tersebut masi masuk kedalam rentang batasan waktu (*time windows*) yang telah ditentukan. Waktu tempuh ini dapat diperoleh dengan melakukan pembagian antara data jarak dua titik dengan kecepatan rata rata kendaraan yang digunakan dalam proses distribusi. Pada penelitian ini menggunakan satuan waktu berupa menit. Berikut merupakan contoh perhitungan total waktu tempuh dari *customer* A dan *customer* B dari PT. Pos Logistik Indonesia:

Jarak antara *Main hub* dan *Customer* 1 = 5,25 km

Kecepatan rata rata kendaraan = 22 km/jam

Waktu tempuh $= \frac{jarak\ tempuh}{kecepatan\ rata\ rata} = \frac{5{,}25}{22} = 0{,}238\ jam = 14{,}31\ menit$

4.2.4 Proses Algoritma *Adaptive Large Neighborhood Search*

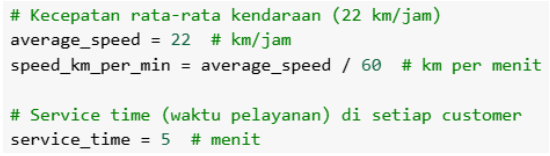
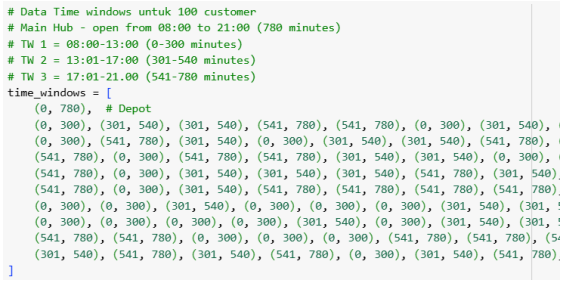
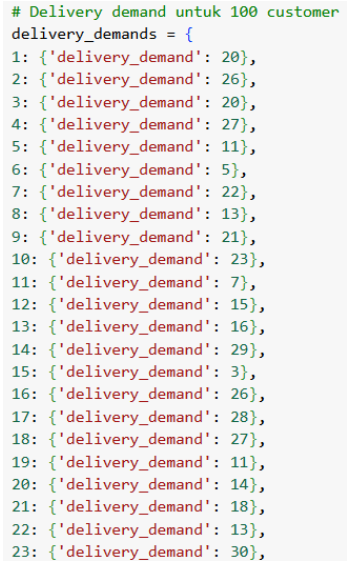
Algoritma *Adaptive Large Neighborhood Search* (ALNS) akan bekerja melakukan pencarian solusi yang paling optimal dalam penentuan rute distrbusi produk dengan memanfaatkan metode *local search*, yang mana algoritma ini mencari solusi terbaik dalam lingkungan yang terbentuk dari suatu solusi awal. Dalam penyelesaian permasalahan CVRPTW pada penentuan rute distribusi produk PT. Pos Logistik Indonesia dengan menggunakan bahasa pemrograman python, terdapat beberapa tahapan yang dilakukan, sebagai berikut:

1. Inisialisasi Data

   Inisialisasi data pada tahap ini digunakan untuk memasukan data awal yang digunakan dalam pembentukan algoritma ini sebagai input dari permasalahan yang akan diselesaikan. Berikut merupakan *pseudocode* dalam tahapan inisialisasi data:

Tabel 4. 7 *Pseudocode* Inisialisasi Data

Pseudocode	**Fungsi**
`import numpy as np` `import random` `from typing import List, Tuple` `import math`	Code tersebut digunakan untuk memasukan *library* yang digunakan dalam fungsi algoritma yang akan dijalankan. Library ini meliputi *numpy*, *random*, *math*, dan *list/tuple*.

Pseudocode	Fungsi
	Mamasukan matriks jarak yang telah didapatkan berdasarkan koordinat titik yang meliputi *customer* dengan total 100 titik + 1 *main hub*.
	Memasukan data terkait kecepatan rata rata dari kendaraan yang digunakan dan rata rata waktu pelayanan di setiap *customer*.
	Menghitung waktu tempuh dari setiap *customer* dengan melakukan pembagian berdasarkan jarak dan kecepatan rata rata kendaraan.
	Memasukan data *time windows* dari setiap *customer* termasuk jam pelayanan *main hub* yang digunakan sebagai batas waktu yang diperbolehkan.
	Memasukan data d*elivery demand* dari setiap titik sebagai data perhitungan kapasitas pada masing masing kendaraan di setiap rute yang terbentuk.

2. Penentuan Parameter

Penentuan parameter pada tahap ini digunakan untuk mengatur ukuran dari suatu populasi sebagai salah satu indikator pengukuran/evaluasi hasil. Berikut merupakan *pseudocode* dalam tahapan penentuan parameter dalam penelitian ini:

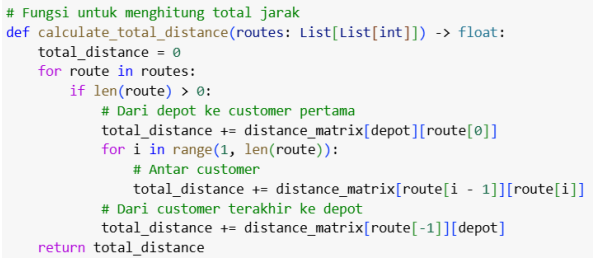
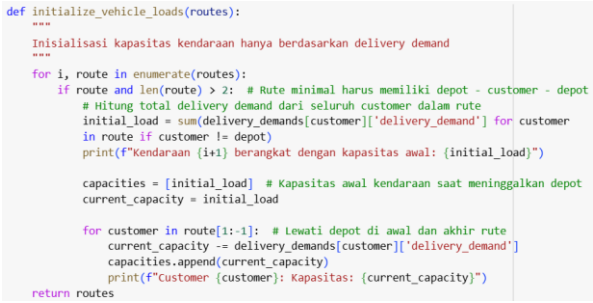
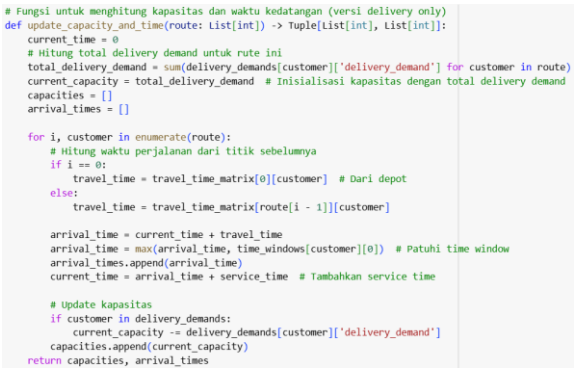
Tabel 4. 8 *Pseudocode* Penentuan Parameter

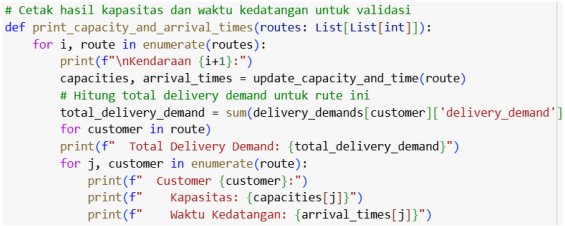
Pseudocode	**Fungsi**
`# Parameter` `num_vehicles = 3  # Jumlah kendaraan` `num_customers = 100  # Jumlah customer` `depot = 0  # Depot adalah customer 0` `max_capacity = 1000  # Kapasitas maksimal kendaraan` `max_iterations = 10000  # Jumlah iterasi`	Memasukan parameter terkait data yang diperoleh dari perusahaan meliputi jumlah kendaraan yaitu 3, total jumlah *customer* yaitu 100, kapasitas maksimum kendaraan yaitu 1000kg. Selain itu, menetapkan max iteration untuk membatasi iterasi pada algoritma ALNS yaitu 10.000 iterasi.
`# Parameter untuk adaptive weight adjustment` `destroy_operators = ["random_removal", "worst_removal"]` `repair_operators = ["greedy_insertion", "regret_insertion"]` `destroy_weights = [1.0, 1.0]  # Bobot awal untuk destroy operators` `repair_weights = [1.0, 1.0]  # Bobot awal untuk repair operators` `weight_update_interval = 100  # Update bobot setiap 100 iterasi` `sigma1 = 1.0  # Faktor peningkatan bobot untuk solusi yang lebih baik` `sigma2 = 0.5  # Faktor peningkatan bobot untuk solusi yang diterima` `sigma3 = 0.1  # Faktor penurunan bobot untuk solusi yang ditolak`	Memasukan parameter *adaptive weight adjustment* yaitu jenis *destroy repair* operator yang digunakan, bobot pada masing masing operator, dan nilai dari sigma 1,2, dan 3 sebagai reward yang akan diberikan pada setiap iterasi.
`# Parameter untuk acceptance criteria (Metropolis Criterion)` `initial_temperature = 1000` `cooling_rate = 0.99`	Memasukan parameter untuk *acceptance criteria* yang menggunakan *metropolis criterion* meliputi *initial temperature* dan *cooling rate.*

3. Menghitung Total Jarak, Kapasitas Kendaraan, dan Waktu Kedatangan

Berdasarkan data yang telah diinputkan meliputi data matriks jarak, *time windows*, dan *delivery demand* , pada tahap ini dilakukan fungsi perhitungan untuk menemukan total jarak dari setiap rute yang terbentuk, kapasitas kendaraan, dan waktu kedatangan dari setiap *customer*. Berikut merupakan *pseudocode* dalam tahapan ini pada penelitian yang dilakukan:

Tabel 4. 9 *Pseudocode* Total Jarak, Kapasitas, dan Waktu

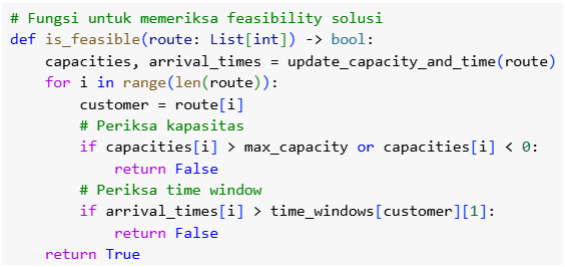
Pseudocode	**Fungsi**
	Fungsi tersebut digunakan untuk melakukan perhitungan total jarak dari sebuah rute kendaraan yang terbentuk. Jarak dihitung mulai dari kendaraan berangkat dari *main hub* ke *customer*, lalu dilanjutkan antar *customer* dalam rute tersebut, kemudian kembali lagi ke *main hub*. Keseluruhan jarak pada rute tersebut dijumlahkan sehingga mendapatkan total jarak pada rute yang terbentuk.
	Melakukan perhitungan kapasitas kendaraan dengan memastikan kendaraan berangkat dari *main hub* dengan kapasitas awal untuk mengirimkan semua *delivery deman*. Kemudian kapasitas akan berkurang dan diperbarui berdasarkan *delivery demand* saat berada di setiap titik pengiriman.
	Fungsi ini digunakan untuk menghitung kapasitas dan waktu kedatangan di setiap titik dalam rute yang terbentuk. Kapasitas akan berkurang sesuai *delivery demand* pada setiap titik yang dilalui kendaraan dan waktu kedatangan dihitung dengan mempertimbangkan waktu perjalanan dan *service time* pada setiap *customer*.

Pseudocode	Fungsi
	Fungsi ini digunakan untuk mencetak hasil validasi yang memuat informasi tentang total permintaan pengiriman, kapasitas kendaraan tersisa, dan waktu kedatangan.

4. Memeriksa *Feasibility* Solusi

Setelah memasukan fungsi yang dugunakan untuk melakukan perhitungan kapasitas kendaraan dan juga waktu kedatangan dari masing masing *customer* pada rute yang terbentuk, pada tahap ini dilakukan *feasibility test* pada suatu rute berdasarkan kapasitas kendaraan dan batasan waktu (*time windows*). Berikut merupakan *Pseudocode* dalam tahapan ini pada penelitian yang dilakukan:

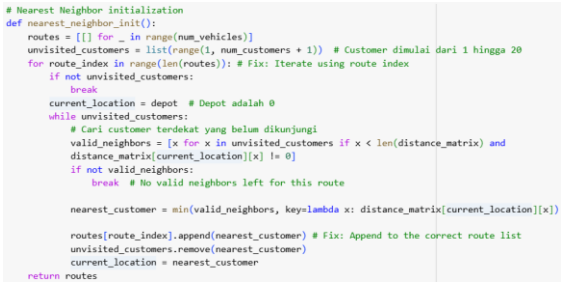
Tabel 4. 10 *Pseudocode Feasibility Check*

Pseudocode	Fungsi
	Fungsi ini digunakan untuk memeriksa rute kendaraan yang terbentuk tanpa melanggar Batasan kapasitas dan *time windows*. Jika kendaraan membawa muatan melebihi kapasitas atau jika kendaraan tiba di *customer* setelah batas waktu yang diperbolehkan, maka rute dianggap tidak layak/tidak valid.

5. Menentukan Solusi Awal

Dalam langkah awal pada algoritma ALNS yaitu membentuk *initial solution* yang digunakan sebagai acuan perbaikan atau optimasi rute pada setiap rute kendaraan yang terbentuk berdasarkan data yang diberikan. Berikut merupakan *Pseudocode* pada tahap ini dalam penelitian yang dilakukan:

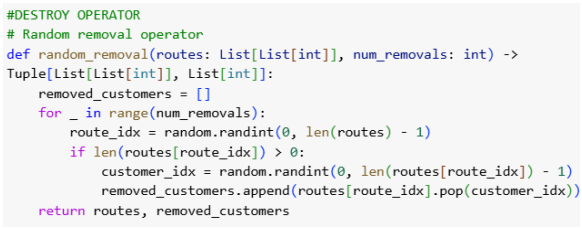
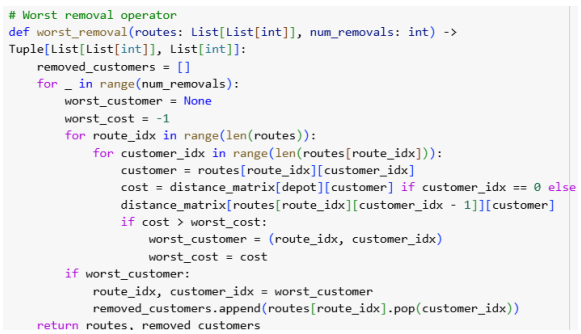
Tabel 4. 11 *Pseudocode* Penentuan Solusi Awal

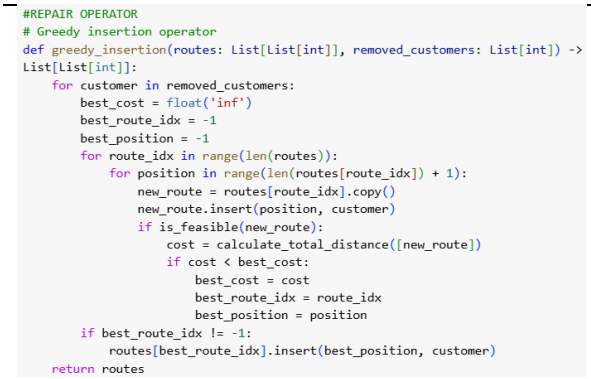
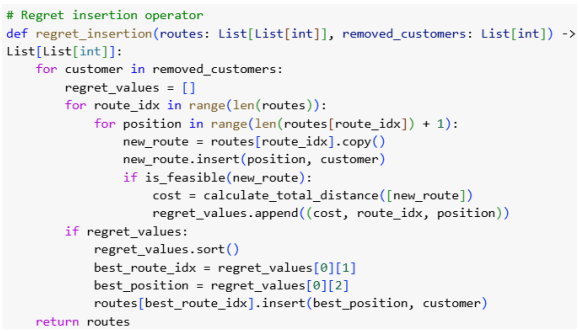
Pseudocode	**Fungsi**
 `# Nearest Neighbor initialization` `def nearest_neighbor_init():` `routes = [[] for _ in range(num_vehicles)]` `unvisited_customers = list(range(1, num_customers + 1))  # Customer dimulai dari 1 hingga 20` `for route_index in range(len(routes)): # Fix: Iterate using route index` `if not unvisited_customers:` `break` `current_location = depot  # Depot adalah 0` `while unvisited_customers:` `# Cari customer terdekat yang belum dikunjungi` `valid_neighbors = [x for x in unvisited_customers if x < len(distance_matrix) and` `distance_matrix[current_location][x] != 0]` `if not valid_neighbors:` `break  # No valid neighbors left for this route` `nearest_customer = min(valid_neighbors, key=lambda x: distance_matrix[current_location][x])` `routes[route_index].append(nearest_customer) # Fix: Append to the correct route list` `unvisited_customers.remove(nearest_customer)` `current_location = nearest_customer` `return routes`	Membentuk rute awal untuk masing masing kendaraan menggunakan metode *Nearest Neighbor*. Setiap kendaraan dimulai dari *main hub* dan memilih *customer* terdekat yang belum dikunjungi hingga semua *customer* masuk kedalam rute.

6. Pembentukan Solusi Sub-lingkaran

Algoritma ALNS bekerja dengan menggunakan mekanisme *destroy-repair* sehingga memungkinkan menemukan solusi yang paling optimal berdasarkan initial solution yang ada. Pada tahap ini sub lingkaran dari masing masing rute akan dibentuk dengan menggunakan *destroy* operator lalu akan diperbaiki menggunakan *repair* operator. Berikut merupakan *pseudocode* yang digunakan dalam tahap ini:

Tabel 4. 12 *Pseudocode Destroy repair*

Pseudocode	**Fungsi**
 `#DESTROY OPERATOR` `# Random removal operator` `def random_removal(routes: List[List[int]], num_removals: int) ->` `Tuple[List[List[int]], List[int]]:` `removed_customers = []` `for _ in range(num_removals):` `route_idx = random.randint(0, len(routes) - 1)` `if len(routes[route_idx]) > 0:` `customer_idx = random.randint(0, len(routes[route_idx]) - 1)` `removed_customers.append(routes[route_idx].pop(customer_idx))` `return routes, removed_customers`	*Random removal* bekerja dengan menghapus beberapa *customer* secara acak dari rute kendaraan. Setiap iterasi memilih rute secara acak dan menghapusnya lalu disimpan dalam daftar terpisah untuk dimasukan kembali pada proses *repair*.
 `# Worst removal operator` `def worst_removal(routes: List[List[int]], num_removals: int) ->` `Tuple[List[List[int]], List[int]]:` `removed_customers = []` `for _ in range(num_removals):` `worst_customer = None` `worst_cost = -1` `for route_idx in range(len(routes)):` `for customer_idx in range(len(routes[route_idx])):` `customer = routes[route_idx][customer_idx]` `cost = distance_matrix[depot][customer] if customer_idx == 0 else` `distance_matrix[routes[route_idx][customer_idx - 1]][customer]` `if cost > worst_cost:` `worst_customer = (route_idx, customer_idx)` `worst_cost = cost` `if worst_customer:` `route_idx, customer_idx = worst_customer` `removed_customers.append(routes[route_idx].pop(customer_idx))` `return routes, removed_customers`	*Worst Removal* digunakan untuk menghapus beberapa *customer* dengan biaya perjalanan tertinggi berdasarkan matriks jarak dan dihapus secara *iterative* hingga mencapai *num_removals.*

Pseudocode	Fungsi
	*Greedy Insertion* digunakan untuk melakukan perbaikan dengan menyisipkan kembali *customer* yang telah dihapus berdasarkan biaya perjalanan paling rendah yang mengacu pada jarak atau waktu tempuh.
	*Regret Insertion* digunakan sebagai *repair* operator untuk menyisipkan kembali *customer* yang telah dihapus berdasarkan *regret value* atau selisih biaya antara dua kemungkinan Solusi yang terbaik sehingga *customer* dimasukan kembali ke tempat yang paling menghindari biaya penyisipan yang tinggi.

7. Perhitungan Fungsi Evaluasi

Setelah memasukan fungsi *destroy repair* operator, pada tahap ini merupakan tahap *adaptive weight adjustment* dan *acceptance criteria* yang merupakan tahapan evaluasi hasil dengan menggunakan bobot yang telah di tetapkan. Berikut merupakan tahapan fungsi evaluasi yang dilakukan:

a. *Adaptive Weight Adjustment*

Pada tahap ini dilakukan *weight adjustment* yang digunakan untuk meningkatkan kemungkinan terpilihnya Solusi yang optimal dengan memberikan bobot yang lebih tinggi sehingga akan lebih sering terpilih pada iterasi berikutnya daripada solusi yang kurang optimal. Ketika setiap iterasi yang dilakukan selesai, nilai ψ untuk *destroy repair* operator akan dihitung menggunakan rumus:

$$\psi = max$$

σ_1 jika solusi baru lebih baik dari pada solusi saat ini

σ_2 jika solusi baru diterima

σ_3 jika solusi baru ditolak

Dimana jika nilai ψ memiliki nilai yang tinggi, algoritma tersebut berhasil dengan nilai $\sigma_1 \geq \sigma_2 \geq \sigma_3$. Jika suatu operator kurang optimal maka akan diberikan bobot dengan rumus sebagai berikut:

$$w_i = max(w_i \times (1 - \sigma_3), 0.1)$$

Akan tetapi jika suatu operator dapat menghasilkan solusi yang optimal maka akan diberikan peningkatan bobot sehingga meningkatkan *probabilitas* operator tersebut akan terpilih di iterasi berikutnya. Peningkatan bobot tersebut mengacu pada rumus sebagai berikut:

$$w_i = w_i + \sigma$$

b. *Acceptance Criteria*

Pada tahap ini solusi baru yang dihasilkan oleh *destroy repair* operator akan di evaluasi apakah diterima atau tidak. Solusi baru akan diterima S' jika $f(S') < f(S)$. Akan tetapi, jika $f(S') \geq f(S)$ yang menunjukan solusi lebih buruk maka akan dihitung peluangnya. Solusi baru S' akan diterima jika $rand[0,1] < e^{-(f(S')-f(S))/T}$.

Tabel 4. 13 *Pseudocode* Fungsi Evaluasi

Pseudocode	**Fungsi**
`# Fungsi untuk memilih operator berdasarkan bobot` `def select_operator(operators: List[str], weights: List[float]) -> str:` `  total_weight = sum(weights)` `  probabilities = [weight / total_weight for weight in weights]` `  return random.choices(operators, probabilities)[0]`	Fungsi ini digunakan untuk memilih satu operator berdasarkan bobot melalui perhitungan probabilitas.
`# Fungsi untuk update bobot operator` `def update_weights(weights: List[float], index: int, sigma: float):` `  weights[index] = max(weights[index] * (1 - sigma3), 0.1)` `  # Pastikan bobot tidak terlalu kecil` `  weights[index] += sigma`	Fungsi ini digunakan untuk memperbarui bobot operator dengan mempertimbangkan nilai sigma. Bobot yang lebih baik akan mendapatkan peningkatan (*reinforcement*), sedangkan yang kurang efektif akan dikurangi tetapi tetap memiliki bobot minimum tertentu.
`# Acceptance criteria (Metropolis Criterion)` `def acceptance_criteria(current_cost: float, new_cost: float, temperature: float) -> bool:` `  if new_cost < current_cost:` `  return True` `  else:` `  probability = math.exp((current_cost - new_cost) / temperature)` `  return random.random() < probability`	Fungsi ini digunakan untuk menentukan apakah solusi baru yang dihasilkan akan diterima atau tidak.

8. Menjalankan ALNS

Tahap terakhir yaitu menjalankan algoritma sesuai dengan alur algoritma ALNS. Berikut merupakan *Pseudocode* yang digunakan untuk menjalankan algoritma ALNS:

Tabel 4. 14 *Pseudocode* Running Algoritma

***Pseudocode* ALNS**

```
# ALNS algorithm
def alns_vrp():
    # Inisialisasi solusi awal menggunakan Nearest Neighbor
    routes = nearest_neighbor_init()
    global_best_routes = routes.copy()
    global_best_distance = calculate_total_distance(global_best_routes)
    current_routes = routes.copy()
    current_distance = global_best_distance

    temperature = initial_temperature

    for iteration in range(max_iterations):
        # Pilih destroy dan repair operator berdasarkan bobot
        destroy_operator = select_operator(destroy_operators, destroy_weights)
        repair_operator = select_operator(repair_operators, repair_weights)

        # Destroy phase
        num_removals = random.randint(1, 3)  # Hapus 1-3 customer secara acak
        if destroy_operator == "random_removal":
            new_routes, removed_customers = random_removal(current_routes.copy(), num_removals)
        elif destroy_operator == "worst_removal":
            new_routes, removed_customers = worst_removal(current_routes.copy(), num_removals)
```

```
# Repair phase
if repair_operator == "greedy_insertion":
    new_routes = greedy_insertion(new_routes, removed_customers)
elif repair_operator == "regret_insertion":
    new_routes = regret_insertion(new_routes, removed_customers)

# Evaluasi solusi baru
new_distance = calculate_total_distance(new_routes)
if all(is_feasible(route) for route in new_routes) and acceptance_criteria(current_distance, new_distance, temperature):
    current_routes = new_routes.copy()
    current_distance = new_distance

    # Update solusi terbaik global
    if new_distance < global_best_distance:
        global_best_routes = new_routes.copy()
        global_best_distance = new_distance
```

```
    # Update bobot operator setiap weight_update_interval iterasi
    if iteration % weight_update_interval == 0:
        if new_distance < current_distance:
            update_weights(destroy_weights, destroy_operators.index(destroy_operator), sigma1)
            update_weights(repair_weights, repair_operators.index(repair_operator), sigma1)
        elif new_distance == current_distance:
            update_weights(destroy_weights, destroy_operators.index(destroy_operator), sigma2)
            update_weights(repair_weights, repair_operators.index(repair_operator), sigma2)
        else:
            update_weights(destroy_weights, destroy_operators.index(destroy_operator), sigma3)
            update_weights(repair_weights, repair_operators.index(repair_operator), sigma3)

    # Cooling schedule
    temperature *= cooling_rate

return global_best_routes, global_best_distance
```

Berdasarkan code tersebut algortima ALNS akan bekerja dengan membentuk Solusi awal yang kemudian *destroy repair* operator akan dipilih untuk membentuk solusi baru yang lebih optimal sesuai dengan iterasi yang telah ditentukan. Bobot operator akan

***Pseudocode* ALNS**
diperbarui seiring tahap iterasi dilakukan. Kemudian, solusi baru akan terbentuk dan dievaluasi apakah diterima atau tidak.

4.2.5 Hasil Rute Terbaik

Dari pengolahan data yang telah dilakukan menggunakan algoritma *Adaptive Large Neighborhood Search* untuk permasalahan CVRPTW di PT. Pos Logistik Indonesia menghasilkan solusi yang optimal dengan membentuk 3 rute yang melayani 100 titik *customer* dengan 1 *main hub*. Berikut merupakan rekapitulasi rute yang dihasilkan meliputi jarak dan waktu tempuh dari masing masing kendaraan:

Tabel 4. 15 Hasil Rute Usulan

Rute Ke-	Rute	Total Jarak
Rute 1	0 - 6 - 52 - 70 - 28 - 60 - 76 - 74 - 20 - 75 - 57 - 41 - 73 - 43 - 27 - 35 - 26 - 9 - 3 - 2 - 32 - 4 - 38 - 24 - 91 - 13 - 86 - 84 - 30 - 50 - 48 - 18 - 47 - 54 - 89 - 25 - 93 - 79 – 80 - 0	91.53 km
Rute 2	0 - 1 - 34 - 98 - 66 - 72 - 23 - 58 - 12 - 39 - 14 - 29 - 88 - 62 - 71 - 16 - 92 - 96 - 99 - 21 - 22 - 87 - 10 - 8 - 44 - 51 - 33 - 5 - 85 – 49 - 0	92.07 km
Rute 3	0 - 94 - 15 - 65 - 83 - 67 - 77 - 100 - 69 - 68 - 56 - 59 - 55 - 11 - 81 - 45 - 82 - 53 - 40 - 46 - 36 - 61 - 90 - 64 - 19 - 63 - 37 - 17 - 31 - 7 - 97 - 78 - 95 – 42 - 0	93.32 km

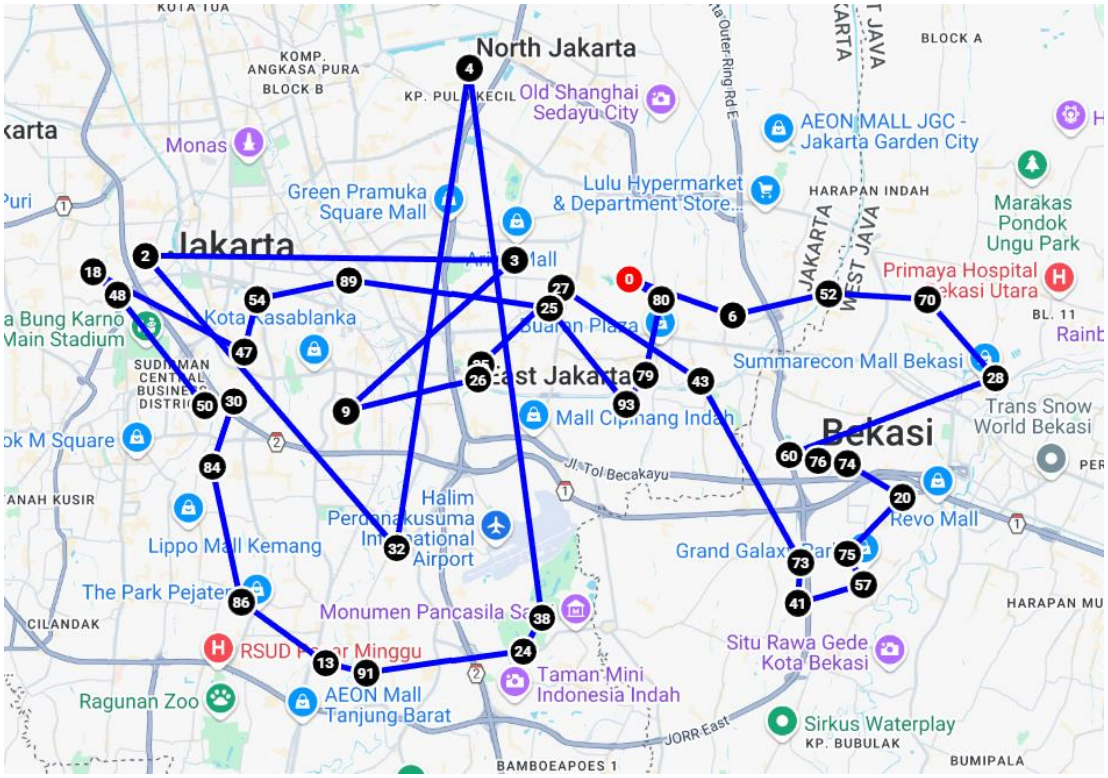
BAB V
PEMBAHASAN

5.1 Analisis Hasil

Hasil dari implementasi algoritma *Adaptive Large Neighborhood Search* (ALNS) dalam menyelesaikan permasalahan *Capacitated Vehicle Routing Problem with Time Windows* (CVRPTW) di PT Pos Logistik Indonesia menunjukkan bahwa algoritma yang digunakan mampu menghasilkan solusi rute yang efisien dan memenuhi kendala kapasitas serta jendela waktu pengantaran. Berdasarkan data pengiriman yang berjumlah 100 titik *customer* dengan jenis armada BlindVan yang memiliki kapasitas maksimum 1000kg, proses optimasi menghasilkan tiga rute utama, sebagai berikut:

a. Rute 1

Berikut merupakan rute pertama yang terbentuk hasil pengolahan menggunakan algoritma ALNS:



Gambar 5. 1 Rute Usulan Kendaraan 1

Tabel 5. 1 Rute Usulan 1

Jenis Kendaraan	ID Number	Rute	Demand (kg)	Total Muatan	Jarak Tempuh	Total Jarak Tempuh
BlindVan	0	*Main hub*	-	686	-	91.53
	6	*Customer* 6	5		2.86	
	52	*Customer* 52	8		2.57	
	70	*Customer* 70	19		2.59	
	28	*Customer* 28	30		2.72	
	60	*Customer* 60	4		5.77	
	76	*Customer* 76	25		0.81	
	74	*Customer* 74	13		0.73	
	20	*Customer* 20	14		1.67	
	75	*Customer* 75	27		2.06	
	57	*Customer* 57	16		0.91	
	41	*Customer* 41	28		1.75	
	73	*Customer* 73	23		1.08	
	43	*Customer* 43	3		5.39	
	27	*Customer* 27	10		4.37	
	35	*Customer* 35	12		2.88	
	26	*Customer* 26	1		0.35	
	9	*Customer* 9	21		3.54	
	3	*Customer* 3	20		3.01	
	2	*Customer* 2	26		0.41	
	32	*Customer* 32	30		1.31	
	4	*Customer* 4	27		1.77	
	38	*Customer* 38	1		2.69	
	24	*Customer* 24	21		0.97	
	91	*Customer* 91	17		4.16	
	13	*Customer* 13	16		1.09	
	86	*Customer* 86	5		2.72	
	84	*Customer* 84	24		3.58	
	30	*Customer* 30	12		1.78	
	50	*Customer* 50	25		0.76	

Jenis Kendaraan	ID Number	Rute	Demand (kg)	Total Muatan	Jarak Tempuh	Total Jarak Tempuh
	48	*Customer* 48	30		3.66	
	18	*Customer* 18	27		0.87	
	47	*Customer* 47	30		4.44	
	54	*Customer* 54	11		1.38	
	89	*Customer* 89	20		2.46	
	25	*Customer* 25	24		5.29	
	93	*Customer* 93	13		3.21	
	79	*Customer* 79	30		0.92	
	80	*Customer* 80	18		2.01	
	0	*Main hub*	-		0.99	

Pada tabel 5.1 merupakan rute yang terbentuk yang memuat 38 titik yang dilalui oleh kendaraan untuk mendistribusikan produk. Dimana dalam proses distribusi ini menggunakan kendaraan dengan jenis BlindVan yang memiliki kapasitas sebesar 1000kg (k=1000). Kendaraan memulai perjalanan dari *Main hub* atau Gudang utama (ID 0) lalu mengunjungi berbagai *customer* sesuai rute yang telah terbentuk. Setiap *customer* memiliki permintaan pengiriman barang dengan berat yang bervariasi dengan total muatan yang diangkut sebesar 686kg. Oleh kerena itu, total muatan yang diangkut oleh kendaraan tidak melebihi kapasitas yang tersedia, sehingga masih memungkinkan kendaraan tersebut melayani *customer* lain untuk memaksimalkan kapasitas yang dimiliki dengan tetap mempertimbangkan *time windows* yang telah ditentukan.

Selain itu, pada tabel diberikan informasi terkait jarak tempuh dari satu titik ke titik lainnya, dengan total jarak tempuh keseluruhan sebesar 91,53 km. Jarak tempuh dari tiap masing masing titik ditentukan berdasarkan titik koordinat yang tersedia yang mana perjalanan terdekat yaitu pada *customer* 35 – *customer* 26 dengan menempuh jarak sebesar 0.35 km, sedangkan perjalanan terjauh yaitu dari *customer* 28 – *customer* 60 dengan menempuh jarak sebesar 5,77 km. Kemudian, total jarak yang dihasilkan akan menjadi salah satu parameter pengukuran untuk mengetahui biaya yang dikeluarkan dengan menggunakan algoritma ALNS.

Tabel 5. 2 Detail Waktu Rute 1

ID *Number*	Jarak Tempuh	Waktu Tempuh	Waktu Kedatangan	Selesai *Service*
6	2.86	7.81	7.81	12.81
52	2.57	7	19.81	24.81
70	2.59	7.07	31.89	36.89
28	2.72	7.42	44.31	49.31
60	5.77	15.74	65.05	70.05
76	0.81	2.2	72.25	77.25
74	0.73	2.01	301	306
20	1.67	4.56	310.56	315.56
75	2.06	5.63	321.2	326.2
57	0.91	2.47	328.67	333.67
41	1.75	4.8	338.47	343.47
73	1.08	2.96	346.43	351.43
43	5.39	14.7	366.13	371.13
27	4.37	11.92	383.04	388.04
35	2.88	7.87	395.91	400.91
26	0.35	0.97	401.88	406.88
9	3.54	9.67	416.55	421.55
3	3.01	8.23	429.78	434.78
2	0.41	1.14	435.92	440.92
32	1.31	3.6	444.52	449.52
4	1.77	4.85	541	546
38	2.69	7.35	553.35	558.35
24	0.97	2.65	561	566
91	4.16	11.37	577.37	582.37
13	1.09	2.98	585.35	590.35
86	2.72	7.44	597.79	602.79
84	3.58	9.77	612.56	617.56
30	1.78	4.86	622.42	627.42
50	0.76	2.08	629.51	634.51
48	3.66	10	644.51	649.51

ID *Number*	Jarak Tempuh	Waktu Tempuh	Waktu Kedatangan	Selesai *Service*
18	0.87	2.39	651.9	656.9
47	4.44	12.13	669.02	674.02
54	1.38	3.79	677.81	682.81
89	2.46	6.72	689.53	694.53
25	5.29	14.44	708.97	713.97
93	3.21	8.77	722.74	727.74
79	0.92	2.52	730.26	735.26
80	2.01	5.49	740.75	745.75
0	0.99	2.7	743,45	-

Berdasarkan data waktu tempuh kendaraan pada rute 1, diketahui bahwa kendaraan memulai perjalanan dari main hub (ID = 0) dan kembali lagi ke main hub setelah menyelesaikan seluruh pelayanan customer. Waktu kembali ke main hub pada rute 1 yaitu 748.45 satuan waktu, sedangkan batas waktu layanan ditetapkan dari pukul 08.00 hingga 21.00, atau setara dengan 0 hingga 780. Dengan demikian, kendaraan menyelesaikan seluruh pelayanannya hingga kembali ke main hub dalam batas waktu layanan, sehingga tidak terjadi keterlambatan kendaraan kembali ke main hub. Seluruh proses distribusi diselesaikan dalam jam kerja yang ditentukan, sehingga driver tidak perlu melakukan lembur (*overtime*). Hal ini menunjukkan bahwa rute yang direncanakan telah sesuai dengan ketentuan time window yang berlaku.

Selain itu, terdapat sisa waktu sebesar 31.55 satuan waktu (780 – 748.45) sebelum jam layanan yang telah ditentukan berakhir. Hal ini menunjukkan adanya peluang optimalisasi rute, di mana waktu yang tersisa tersebut dapat dimanfaatkan untuk menambahkan satu pelanggan tambahan ke dalam rute dengan mempertimbangkan beberapa batasan yang telah ditentukan sebelumnya.

b. Rute 2

Berikut merupakan rute kedua yang terbentuk hasil pengolahan menggunakan algoritma ALNS:



Gambar 5. 2 Rute Usulan Kendaraan 2

Tabel 5. 3 Rute Usulan 2

Kendaraan	ID *Number*	Rute	*Demand* (kg)	Total Muatan	Jarak Tempuh (Km)	Total Jarak Tempuh
BlindVan	0	*Main hub*	-	407	-	92.07
	1	*Customer* 1	20		5.24	
	34	*Customer* 34	22		5.26	
	98	*Customer* 98	10		1.83	
	66	*Customer* 66	6		2.81	
	72	*Customer* 72	1		1.31	
	23	*Customer* 23	30		1.18	
	58	*Customer* 58	30		3.05	
	12	*Customer* 12	15		1.01	
	39	*Customer* 39	14		1.31	

Kendaraan	ID *Number*	Rute	*Demand* (kg)	Total Muatan	Jarak Tempuh (Km)	Total Jarak Tempuh
	14	*Customer* 14	29		2.14	
	29	*Customer* 29	26		0.59	
	88	*Customer* 88	1		2.44	
	62	*Customer* 62	21		1.56	
	71	*Customer* 71	2		2.28	
	16	*Customer* 16	26		7.02	
	92	*Customer* 92	2		6.01	
	96	*Customer* 96	2		4.12	
	99	*Customer* 99	15		7.69	
	21	*Customer* 21	18		1.27	
	22	*Customer* 22	13		3.49	
	87	*Customer* 87	2		0.91	
	10	*Customer* 10	23		2.21	
	8	*Customer* 8	13		5.12	
	44	*Customer* 44	23		6.22	
	51	*Customer* 51	7		1.38	
	33	*Customer* 33	10		0.67	
	5	*Customer* 5	11		5.87	
	85	*Customer* 85	8		4.11	
	49	*Customer* 49	7		1.21	
	0	*Main hub*	-		2.76	

Pada tabel 5.2 merupakan rute yang terbentuk yang memuat 29 titik yang dilalui oleh kendaraan untuk mendistribusikan produk. Dimana dalam proses distribusi ini menggunakan kendaraan dengan jenis BlindVan yang memiliki kapasitas sebesar 1000kg (k=1000). Kendaraan memulai perjalanan dari *Main hub* atau Gudang utama (ID 0) lalu mengunjungi berbagai *customer* sesuai rute yang telah terbentuk. Setiap *customer* memiliki permintaan pengiriman barang dengan berat yang bervariasi dengan total muatan yang diangkut sebesar 407kg. Oleh kerena itu, total muatan yang diangkut oleh kendaraan tidak melebihi kapasitas yang tersedia, sehingga masih memungkinkan kendaraan tersebut melayani *customer* lain untuk

memaksimalkan kapasitas yang dimiliki dengan tetap mempertimbangkan *time windows* yang telah ditentukan.

Selain itu, pada tabel diberikan informasi terkait jarak tempuh dari satu titik ke titik lainnya, dengan total jarak tempuh keseluruhan sebesar 92,07 km. Jarak tempuh dari tiap masing masing titik ditentukan berdasarkan titik koordinat yang tersedia yang mana perjalanan terdekat yaitu pada *customer* 14 – *customer* 29 dengan menempuh jarak sebesar 0.59 km, sedanghkan perjalanan terjauh yaitu dari *customer* 96 – *customer* 99 dengan menempuh jarak sebesar 7,69 km. Kemudian, total jarak yang dihasilkan akan menjadi salah satu parameter pengukuran untuk mengetahui biaya yang dikeluarkan dengan menggunakan algoritma ALNS.

Tabel 5. 4 Detail Waktu Rute 2

ID *Number*	Jarak Tempuh	Waktu Tempuh	Waktu Kedatangan	Selesai *Service*
1	5.25	14.32	14.32	19.32
34	5.26	14.36	33.67	38.67
98	1.83	5	43.68	48.68
66	2.82	7.68	56.36	61.36
72	1.3	3.55	64.91	69.91
23	1.19	3.23	73.14	78.14
58	3.06	8.33	86.48	91.48
12	1.01	2.76	94.23	99.23
39	1.31	3.56	301	306
14	2.15	5.85	311.85	316.85
29	0.59	1.61	318.47	323.47
88	2.45	6.68	330.15	335.15
62	1.57	4.27	339.42	344.42
71	2.28	6.22	350.64	355.64
16	7.03	19.16	374.81	379.81
92	6.01	16.38	396.19	401.19
96	4.12	11.25	541	546
99	7.69	20.98	566.98	571.98
21	1.28	3.49	575.47	580.47
22	3.5	9.54	590.01	595.01
87	0.92	2.5	597.5	602.5
10	2.22	6.05	608.55	613.55
8	5.13	13.98	627.53	632.53

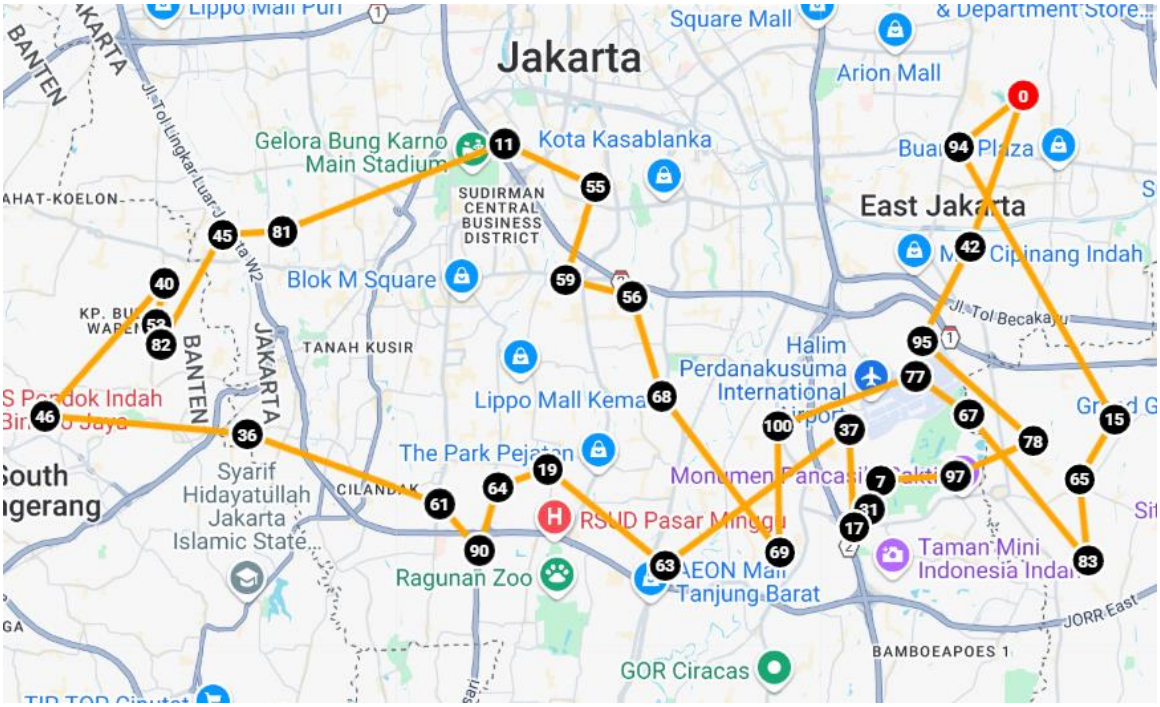
ID *Number*	Jarak Tempuh	Waktu Tempuh	Waktu Kedatangan	Selesai *Service*
44	6.22	16.97	649.51	654.51
51	1.39	3.79	658.29	663.29
33	0.67	1.83	665.12	670.12
5	5.87	16.02	686.14	691.14
85	4.11	11.21	702.35	707.35
49	1.2	3.28	710.63	715.63
0	2.76	7.52	718.15	-

Berdasarkan data waktu tempuh kendaraan pada rute 2, diketahui bahwa kendaraan memulai perjalanan dari main hub (ID = 0) dan kembali lagi ke main hub setelah menyelesaikan seluruh pelayanan customer. Kendaraan mengunjungi sejumlah titik pelanggan, dan kembali lagi ke main hub pada waktu 718.15 satuan waktu dengan waktu layanan yang ditetapkan dari pukul 08.00 hingga 21.00, yaitu setara pada interval waktu 0 hingga 780. Dengan demikian, dapat disimpulkan bahwa kendaraan telah kembali ke main hub dalam batas waktu layanan yang ditentukan, yaitu sekitar 61.85 satuan waktu sebelum waktu tutup layanan. Hal ini menunjukkan bahwa seluruh aktivitas pelayanan dilakukan secara tepat waktu dan tidak menyebabkan driver melakukan overtime.

Selain itu, sisa waktu layanan yang masih tersedia menunjukkan adanya potensi untuk meningkatkan efisiensi operasional melalui penambahan pelanggan ke dalam rute yang sama. Dengan sisa waktu sebesar 61.85 satuan waktu, masih terdapat ruang untuk melayani satu atau bahkan dua pelanggan tambahan, selama waktu tempuh dan pelayanan tambahan tersebut tidak menyebabkan kendaraan kembali ke hub melewati batas waktu 780. Penambahan pelanggan dapat dipertimbangkan dengan melihat lokasi pelanggan tambahan berada dalam jalur dan waktu pelayanan relatif singkat. Selain itu, time window dari pelanggan tambahan juga harus memenuhi dengan waktu yang tersisa agar tetap memenuhi batasan pelayanan.

c. Rute 3

Berikut merupakan rute ketiga yang terbentuk hasil pengolahan menggunakan algoritma ALNS:



Gambar 5. 3 Rute Usulan Kendaraan 3

Tabel 5. 5 Rute Usulan 3

Jenis Kendaraan	ID *Number*	Rute	*Demand* (kg)	Total Muatan (Kg)	Jarak Tempuh (Km)	Total Jarak Tempuh
	0	*Main hub*	-		-	
	94	*Customer* 94	24		1.88	
	15	*Customer* 15	3		7.14	
	65	*Customer* 65	29		1.58	
	83	*Customer* 83	26		1.85	
BlindVan	67	*Customer* 67	25	553	4.33	93.32
	77	*Customer* 77	15		1.49	
	100	*Customer* 100	20		3.32	
	69	*Customer* 69	27		2.89	
	68	*Customer* 68	4		4.47	
	56	*Customer* 56	27		2.37	

Jenis Kendaraan	ID *Number*	Rute	*Demand* (kg)	Total Muatan (Kg)	Jarak Tempuh (Km)	Total Jarak Tempuh
	59	*Customer* 59	9		1.56	
	55	*Customer* 55	11		2.21	
	11	*Customer* 11	7		2.29	
	81	*Customer* 81	12		5.47	
	45	*Customer* 45	11		1.39	
	82	*Customer* 82	25		2.86	
	53	*Customer* 53	3		0.47	
	40	*Customer* 40	3		0.97	
	46	*Customer* 46	18		4.08	
	36	*Customer* 36	17		4.64	
	61	*Customer* 61	26		4.69	
	90	*Customer* 90	29		1.38	
	64	*Customer* 64	20		1.49	
	19	*Customer* 19	11		1.19	
	63	*Customer* 63	17		3.48	
	37	*Customer* 37	27		5.24	
	17	*Customer* 17	28		2.24	
	31	*Customer* 31	13		0.53	
	7	*Customer* 7	22		0.69	
	97	*Customer* 97	1		1.7	
	78	*Customer* 78	6		1.99	
	95	*Customer* 95	20		3.4	
	42	*Customer* 42	17		4.41	
	0	*Customer* 0	-		3.63	

Pada tabel 5.3 merupakan rute yang terbentuk yang memuat 33 titik yang dilalui oleh kendaraan untuk mendistribusikan produk. Dimana dalam proses distribusi ini menggunakan kendaraan dengan jenis BlindVan yang memiliki kapasitas sebesar 1000kg (k=1000). Kendaraan memulai perjalanan dari *Main hub* atau Gudang utama (ID 0) lalu mengunjungi berbagai *customer* sesuai rute yang telah terbentuk. Setiap *customer* memiliki permintaan

pengiriman barang dan pengambilan barang dengan berat yang bervariasi dengan total muatan yang diangkut sebesar 553kg. Oleh kerena itu, total muatan yang diangkut oleh kendaraan tidak melebihi kapasitas yang tersedia, sehingga masih memungkinkan kendaraan tersebut melayani *customer* lain untuk memaksimalkan kapasitas yang dimiliki dengan tetap mempertimbangkan *time windows* yang telah ditentukan.

Selain itu, pada tabel diberikan informasi terkait jarak tempuh dari satu titik ke titik lainnya, dengan total jarak tempuh keseluruhan sebesar 93,32 km. Jarak tempuh dari tiap masing masing titik ditentukan berdasarkan titik koordinat yang tersedia yang mana perjalanan terdekat yaitu pada *customer* 82 – *customer* 53 dengan menempuh jarak sebesar 0.47 km, sedanghkan perjalanan terjauh yaitu dari *Customer* 94 – *Customer* 15 dengan menempuh jarak sebesar 7,14 km. Kemudian, total jarak yang dihasilkan akan menjadi salah satu parameter pengukuran untuk mengetahui biaya yang dikeluarkan dengan menggunakan algoritma ALNS.

Tabel 5. 6 Detail Waktu Rute 3

ID Number	**Jarak Tempuh**	**Waktu Tempuh**	**Waktu Kedatangan**	**Selesai *Service***
94	1.88	5.13	5.13	10.13
15	7.14	19.5	29.63	34.63
65	1.58	4.32	38.95	43.95
83	1.85	5.05	49/.01	54.01
67	4.33	11.82	65.82	70.82
77	1.49	4.08	74.9	79.9
100	3.32	9.07	88.98	93.98
69	2.89	7.9	101.87	106.87
68	4.47	12.21	119.08	124.08
56	2.37	6.47	130.55	135.55
59	1.56	4.27	139.82	144.82
55	2.21	6.04	150.86	155.86
11	2.29	6.27	162.13	167.13
81	5.47	14.92	182.05	187.05
45	1.39	3.82	190.87	195.87
82	2.86	7.81	203.67	208.67
53	0.47	1.29	301	306
40	0.97	2.67	308.67	313.67

ID Number	Jarak Tempuh	Waktu Tempuh	Waktu Kedatangan	Selesai *Service*
46	4.08	11.14	324.81	329.81
36	4.64	12.66	342.47	347.47
61	4.69	12.81	360.28	365.28
90	1.38	3.78	369.06	374.06
64	1.49	4.07	378.13	383.13
19	1.19	3.25	386.38	391.38
63	3.48	9.51	400.88	405.88
37	5.24	14.29	420.18	425.18
17	2.24	6.11	431.29	436.29
31	0.53	1.45	437.74	442.74
7	0.69	1.88	444.62	449.62
97	1.7	4.64	454.26	459.26
78	1.99	5.44	464.7	469.7
95	3.4	9.28	478.98	483.98
42	2.41	6.59	490.57	495.57
0	3.63	9.9	500.47	-

Berdasarkan data waktu tempuh kendaraan pada rute 3, diketahui bahwa kendaraan memulai perjalanan dari main hub (ID = 0) dan kembali lagi ke main hub setelah menyelesaikan seluruh pelayanan customer. Kendaraan kembali ke main hub pada waktu 500.47 satuan waktu. Sementara itu, jam layanan yang ditetapkan berada dalam rentang pukul 08.00 hingga 21.00, atau setara dengan 0 hingga 780 satuan waktu. Dengan demikian, dapat disimpulkan bahwa kendaraan kembali ke hub jauh sebelum batas akhir waktu layanan, yaitu masih menyisakan waktu sebesar 279,53 satuan waktu. Kondisi ini menunjukkan bahwa seluruh proses pelayanan dilakukan secara tepat waktu dan tidak melebihi jam operasional yang telah ditetapkan. Hal ini juga berarti pengemudi (driver) tidak perlu melakukan lembur atau overtime karena pekerjaan telah selesai dalam batas waktu kerja yang diperbolehkan.

Selain itu, sisa waktu yang tersisa sebesar 279,53, yang mana dapat dimanfaatkan untuk menambahkan satu atau lebih pelanggan ke dalam rute yang sama. Penambahan pelanggan tentu harus mempertimbangkan beberapa hal, seperti lokasi pelanggan yang tidak terlalu jauh dari jalur rute, waktu tempuh, serta kecocokan dengan time window dari pelanggan tambahin. Oleh karena itu, rute ini berpotensi untuk melayani 5-10 pelanggan tambahin diluar sample

yang diambil tanpa mengganggu batas akhir layanan dan tidak menyebabkan kendaraan kembali ke main hub terlambat.

5.2 Analisis Biaya Distribusi

Biaya distribusi merupakan biaya yang dikeluarkan perusahaan untuk melakukan kegiatan distribusi yang meliputi biaya tetap dan biaya variable. Berikut merupakan analisis biaya distribusi yang dikeluarkan PT. Pos Logistik Indonesia berdasarkan rute awal dan rute usulan yang terbentuk:

a. Biaya Tetap

Fixed cost atau biaya tetap merupakan biaya yang dikeluarkan oleh PT. Pos Logistik Indonesia untuk melakukan proses distribusi produk ke *customer* secara tetap. Biaya ini tidak memperhatikan jarak tempuh kendaraan dan kapasitas yang diangkut. Biaya tetap ini meliputi biaya sewa kendaraan, dan gaji supir. Berikut merupakan biaya tetap yang dikeluarkan dalam proses distribusi produk PT. Pos Logistik Indonesia:

Tabel 5. 7 Biaya Sewa Kendaraan

Pengeluaran	Jenis Kendaraan	Biaya Sewa/Bulan	Biaya Sewa/Hari
Sewa Kendaraan	BlindVan	Rp4.120.000,00	Rp137.333,33

Tabel 5. 8 Biaya Tenaga Kerja

Pengeluaran	Petugas per Rute	Gaji/Bulan	Gaji/Hari
Tenaga Kerja	1	Rp5.600.000,00	Rp186.666,67

Berdasarkan tabel diatas, terdapat biaya tetap yang dikeluarkan perusahaan meliputi biaya sewa kendaraan dan biaya tenaga kerja. Biaya sewa kendaraan yang dikeluarkan bervariasi tergantung pada jenis kendaraan yang digunakan dalam proses distribusi produk. Pada penelitian ini jenis kendaraan yang digunakan yaitu BlindVan yang mana memiliki biaya sewa per hari sebesar Rp137.333,33. Kemudian, biaya tenaga kerja yang dikeluarkan mengacu pada Upah Minimum Provinsi DKI Jakarta yaitu sebesar Rp186.666,67.

b. Biaya Variable

Biaya variable merupakan biaya yang dikeluarkan oleh PT. Pos Logistik Indonesia dalam proses distribusi produk yang dapat berubah ubah sesuai dengan jarak yang ditempuh oleh masing masing kendaraan. Berikut merupakan biaya variable yang dikeluarkan oleh perusahaan dalam proses distribusi produk dengan jenis kendaraan BlindVan:

Tabel 5. 9 Biaya Bahan Bakar

Jenis BBM	Rasio Konsumsi BBM	Harga BBM/Liter	Harga BBM/km
Pertalite	1 : 11	Rp10.000,00	Rp909,09

Berdasarkan tabel diatas, kendaraan BlindVan yang digunakan untuk proses distribusi menggunakan jenis BBM pertalite dengan rasio konsumsi BBM 1 : 11. Hal tersebut berarti setiap 1 liter BBM dapat digunakan untuk menempuh jarak sejauh 11 km. Harga BBM/liter berdasarkan harga standard yang telah ditetapkan yaitu sebesar Rp10.000,00 dengan Rp909,09 per Hari.

c. Rekapitulasi Total Biaya

Total biaya distribusi yang dilakukan merupakan hasil penjumlahan antara biaya tetap dan biaya variable dari masing masing rute yang terbentuk. Pada penelelitian yang dilakukan, jarak tempuh kendaraan merupakan salah satu faktor yang menyebabkan tinggi rendahnya biaya distribusi pada setiap kendaraan. Semakin pendek jarak tempuh kendaraan akan menyebabkan biaya yang dikeluarkan semakin rendah. Hal tersebut dikarenakan biaya variable yang digunakan mengacu pada jarak tempuh dari setiap kendaraan sehingga semakin jauh jarak yang ditempuh akan semakin besar biaya variable yang dikeluarkan perusahaan. Berikut merupakan total biaya yang perlu dikeluarkan perusahaan untuk proses distribusi produk berdasarkan rute yang terbentuk:

1) Rute Awal

Tabel 5. 10 Total Biaya Rute Awal

Rute Ke-	Total Kunjungan	Jarak Tempuh (km)	Biaya Tetap (Rp)	Biaya Variable (Rp)
1	34	205.58	Rp324.000,00	Rp186.890,90
2	33	205.76	Rp324.000,00	Rp187.054,54
3	33	236.06	Rp324.000,00	Rp214.599,99
	Sub Total		**Rp972.000,00**	**Rp588.545,43**
	TOTAL		**Rp1.560.545,44**	

Berdasarkan tabel diatas, dapat diketahui total biaya yang dikeluarkan untuk melakukan distribusi produk sebesar Rp1.560.545,44 untuk melakukan distribusi ke 100 titik *customer* yang tersebar di daerah Jakarta. Biaya tersebut dihitung berdasarkan biaya tetap dan biaya variable. Biaya tetap meliputi biaya sewa kendaraan sebesar Rp137.333,33 dan biaya tenaga kerja sebesar Rp186.666,7 dengan total biaya tetap untuk 3 rute yang

terbentuk sebesar Rp972.000.00. Biaya variable pada setiap rute berbeda beda menyesuaikan dengan total jarak yang ditempuh masing masing kendaraan, yang mana total biaya variable yang dikeluarkan sebesar Rp588.545,43.

Pada tabel diatas, rute 3 merupakan rute yang memiliki biaya paling tinggi diantara rute 1 dan rute 2 dengan biaya sebesar Rp538.599,99, yang mana berbanding lurus dengan jarak tempuh yang dilalui kendaraan yaitu sebesar 236,06 km dengan total kunjungan 33 titik. Sementara itu, rute dengan biaya paling rendah yaitu rute 1 dengan total biaya sebesar Rp510.810,90 dan jarak tempuh kendaraan yaitu 205,58 km dengan total kunjungan 34 titik.

2) Rute Usulan

Tabel 5. 11 Total Biaya Rute Usulan

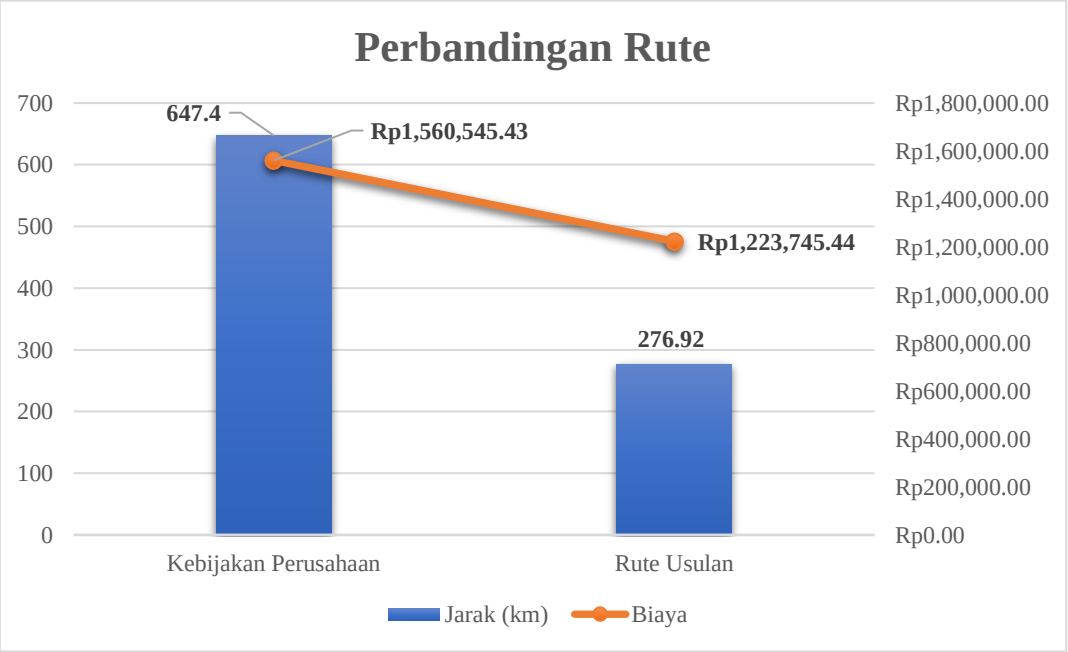
Rute Ke-	Total Kunjungan	Jarak Tempuh (km)	Biaya Tetap (Rp)	Biaya Variable (Rp)
1	38	91.53	Rp324.000,00	Rp83.209,09
2	29	92.07	Rp324.000,00	Rp83.699,99
3	33	93.32	Rp324.000,00	Rp84.868,36
	Sub Total		**Rp972.000,00**	**Rp251.745,44**
	TOTAL		**Rp1.223.745,44**	

Berdasarkan tabel diatas, dapat diketahui total biaya yang dikeluarkan untuk melakukan distribusi produk sebesar Rp1.223.745,44 untuk melakukan distribusi ke 100 titik *customer* yang tersebar di daerah Jakarta. Biaya tersebut dihitung berdasarkan biaya tetap dan biaya variable. Biaya tetap meliputi biaya sewa kendaraan sebesar Rp137.333,33 dan biaya tenaga kerja sebesar Rp186.666,7 dengan total biaya tetap untuk 3 rute yang terbentuk sebesar Rp972.000.00. Biaya variable pada setiap rute berbeda beda menyesuaikan dengan total jarak yang ditempuh masing masing kendaraan, yang mana total biaya variable yang dikeluarkan sebesar Rp251.745,44.

Pada tabel diatas, rute 3 merupakan rute yang memiliki biaya paling tinggi diantara rute 1 dan rute 2 dengan biaya sebesar Rp408.868,36, yang mana berbanding lurus dengan jarak tempuh yang dilalui kendaraan yaitu sebesar 93.32 km dengan total kunjungan 33 titik. Sementara itu, rute dengan biaya paling rendah yaitu rute 1 dengan total biaya sebesar Rp407.209,09 dan jarak tempuh kendaraan yaitu 91.53 km dengan total kunjungan 38 titik.

5.3 Perbandingan ALNS dengan Kebijakan Perusahaan

Berdasarkan hasil wawancara yang dilakukan, penentuan rute yang dilakukan PT. Pos Logistik Indonesia saat ini yaitu dengan berdasarkan historical data atau pengalaman dari masing masing driver. Penentuan rute juga didasari oleh hasil evaluasi per semester atau per tahun yang dilakukan oleh perusahaan. Akan tetapi, cara tersebut dianggap kurang efektif untuk mengambil keputusan pada level operasional. Berdasarkan hasil wawancara yang dilakukan terdapat 3 kendaraan yang digunakan untuk mengunjungi 100 titik *customer*. Terdapat 3 rute yang terbentuk dengan total jarak sebesar 647,4 km. Sementara itu, rute usulan terbentuk dengan melakukan optimasi menggunakan algoritma ALNS. Hasil optimasi yang dilakukan terbentuk 3 rute distribusi untuk mengunjungi 100 titik dengan total jarak sebesar 276,92 km. Berikut merupakan perbandingan rute antara kebijakan perusahaan dan rute usulan:



Gambar 5. 4 Perbandingan Kebijakan Perusahaan dengan ALNS

Berdasarkan gambar diatas menunjukan hasil perbandingan antara rute distribusi berdasarkan kebijakan awal perusahaan dengan rute usulan hasil optimasi menggunakan algoritma *Adaptive Large Neighborhood Search* (ALNS) menunjukkan adanya perbedaan yang cukup siginifikan dalam hal jarak tempuh. Rute awal yang digunakan oleh PT. Pos Logistik Indonesia memiliki total jarak sejauh 647,4 km, sedangkan rute hasil optimasi ALNS hanya menempuh 276,92 km. Hal tersebut menunjukan terdapat pengurangan jarak sebesar 370,48 km atau sekitar 57,22%. Pengurangan jarak dapat mengatasi terhadap permasalahan yang selama ini dihadapi perusahaan, seperti keterlambatan kendaraan kembali ke *main hub*, dan hingga jam lembur bagi

pegawai akibat rute yang terlalu panjang dan tidak efisien. Algoritma ALNS, melalui proses *destroy and repair* yang adaptif, mampu menghasilkan optimasi rute yang jauh lebih optimal.

Efisiensi jarak ini juga berdampak langsung pada pengurangan biaya distribusi, di mana rute awal membutuhkan biaya sebesar Rp1.560.545,43, sedangkan rute hasil usulan membutuhkan biaya sebesar Rp1.223.745,44. Dengan demikian, hal tersebut menunjukan adanya penghematan sebesar Rp336.799,99 atau sekitar 21,58%. Selain menekan biaya, rute yang lebih pendek juga memperkecil risiko keterlambatan dan kelelahan kerja, serta mengurangi kebutuhan lembur yang bisa berdampak pada penurunan produktivitas jangka panjang. Oleh karena itu, mengacu dampak signifikan dari efisiensi jarak dan biaya yang dihasilkan, diperlukan langkah strategis untuk memastikan bahwa optimasi ini dapat diterapkan secara konsisten dalam operasional harian perusahaan.

Melihat kondisi operasional PT. Pos Logistik Indonesia yang memiliki karakteristik permintaan pelanggan yang dinamis setiap harinya, di mana titik layanan cenderung berubah-ubah, penerapan algoritma Adaptive Large Neighborhood Search (ALNS) dapat membantu menentukan rute kunjungan dari setiap kendaraan dengan menemukan rute dengan jarak tempuh terpendek. ALNS yang bersifat fleksibel mampu menangani variasi data secara efisien dan tetap menghasilkan rute optimal dalam waktu yang relatif singkat. Untuk mendukung implementasi algoritma ini dalam proses harian, data pelanggan dan titik distribusi dapat secara otomatis diambil dari file Excel menggunakan fungsi *pandas* dalam Python, seperti pd.read_excel( ). Dengan demikian, proses input data menjadi lebih cepat, sehingga proses optimasi dalam penentuan rute dapat berjalan secara *real-time* dan adaptif mengikuti perubahan permintaan harian.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan penelitian yang dilakukan dalam menentukan rute distribusi yang optimal menggunakan algoritma ALNS, diperoleh beberapa kesimpulan sebagai berikut:

1. *Adaptive Large Neighborhood Search* merupakan algoritma *metaheuristic* yang digunakan untuk melakukan optimasi rute distribusi produk pada PT. Pos Logistik Indonesia. Algoritma ini mampu menyelesaikan dan memberikan hasil optimasi untuk 100 titik yang *customer* dengan 1 *main hub* sebagai titik awal dan akhir proses distribusi. ALNS membentuk rute usulan dengan 3 rute yang terbentuk menggunakan kendaraan jenis BlindVan dengan kapasitas maksimum kendaraan k = 1000 kg. Rute 1 terbentuk dengan total kunjungan 38 titik dan total muatan sebesar 686kg. Rute ini melalui 38 *customer* yang tersebar diwilayah Jakarta dengan total jarak tempuh sebesar 91,53 km. Rute 2 terbentuk dengan total kunjungan 29 titik dan total muatan sebesar 407kg. Rute ini melalui 29 *customer* di wilayah Jakarta dengan total jarak tempuh sebesar 92,07 km. Kemudian, rute 3 terbentuk dengan total kunjungan 33 titik dan total muatan sebesar 553kg. Rute ini melalui 33 *customer* di wilayah Jakarta dengan total jarak tempuh sebesar 93,32 km. Dengan demikian, berdasarkan hasil optimasi dengan menggunakan algoritma ALNS membentuk 3 rute dengan total jarak tempuh keseluruhan sebesar 276,92 km.
2. Hasil optimasi rute menunjukan adanya perbedaan yang signifikan antara rute distribusi berdasarkan kebijakan awal perusahaan dengan rute hasil optimasi menggunakan algoritma *Adaptive Large Neighborhood Search* (ALNS), berdasarkan jarak tempuh yang dilalui setiap kendaraan. Pada rute awal, total jarak yang harus ditempuh oleh tiga kendaraan adalah 647,40 km, yang masing-masing mencakup 205,58 km, 205,76 km, dan 236,06 km. Sementara itu, pada rute usulan hasil optimasi ALNS, total jarak berkurang menjadi 276,92 km, dengan masing-masing rute sejauh 91,53 km, 92,07 km, dan 93,32 km. Hal ini menunjukkan adanya pengurangan jarak total sebesar 370,48 km atau sekitar 57,22%. Jarak tempuh yang terlalu panjang pada rute awal menyebabkan kendaraan bekerja dalam durasi yang lebih lama sehingga pengurangan jarak ini menjadi solusi utama terhadap beberapa kendala operasional yang sebelumnya sering terjadi, seperti keterlambatan

kendaraan kembali ke *main hub*, beban kerja berlebih bagi pengemudi, dan jam lembur yang harus dilakukan oleh driver.

3. Optimalisasi rute distribusi yang dilakukan menggunakan algoritma ALNS menghasilkan pengurangan total jarak tempuh yang cukup signifikan, yang mana berdampak juga pada penurunan biaya distribusi. Dibandingkan dengan rute awal berdasarkan kebijakan perusahaan yang menggunakan 3 rute kendaraan jenis Blindvan dengan total jarak tempuh 647,40 km dan biaya distribusi sebesar Rp1.560.545,43, rute usulan hasil optimasi hanya menempuh jarak dengan total sebesar 276,92 km dengan biaya Rp1.223.745,44. Dengan demikian, terdapat penghematan biaya sebesar Rp336.799,99 atau sekitar 21,58%. Hal ini menunjukkan bahwa algoritma ALNS mampu menemukan rute terpendek secara efektif, dan karena jarak tempuh berbanding lurus dengan biaya operasional, maka semakin pendek jarak yang ditempuh, semakin rendah pula biaya yang dikeluarkan oleh perusahaan.

6.2 Saran

Berdasarkan penelitian yang telah dilakukan di PT. Pos Logistik Indonesia, terdapat beberapa saran atau rekomendasi yang dapat dijadikan sebagai pertimbangan perusahaan serta penelitian selanjutnya yang akan dilakukan dengan menggunakan metode serupa. Berikut merupakan saran yang diberikan:

1. Bagi Perusahaan
   - Berdasarkan pada penelitian yang telah dilakukan oleh peneliti di PT. Pos Logistik Indonesia, perusahaan dapat menggunakan algoritma ALNS sebagai salah satu bahan pertimbangan untuk mengambil keputusan dalam penentuan rute distribusi produk. Algoritma ini mampu memberikan minimasi atau pengurangan jarak tempuh yang cukup signifikan dengan mencari rute dengan jarak terpendek dari masing masing kendaraan dengan tetap mempertimbangkan beberapa batasan lainnya seperti *time windows*, dan kapasitas kendaraan. Berdasarkan hasil pengolahan data dengan menggunakan sampel 100 titik *customer* di daerah Jakarta, ALNS mampu memberikan pengurangan jarak sebesar 370,48 km atau sekitar 57,22%.
   - Algoritma ini dapat dijadikan sebagai alat pengambil keputusan dalam menentukan rute distribusi di seluruh wilayah agar pengurangan jarak yang dilakukan dapat dicapai secara maksimal dan meminimalisir risiko akibat jarak tempuh kendaraan yang tidak optimal.

2. Bagi Penelitian Selanjutnya
   - Penelitian selanjutnya disarankan untuk mempertimbangkan variabel kondisi lalu lintas dengan memanfaatkan data *Estimated Time of Arrival* (ETA) sebagai input. Hal ini menjadi relevan terutama apabila lokasi penelitian berada di wilayah dengan tingkat kemacetan tinggi, seperti Jakarta, sehingga estimasi waktu dalam proses perencanaan rute yang terbentuk dan analisis dapat lebih akurat dan representatif terhadap kondisi nyata di lapangan.
   - Penentuan rute pada penelitian ini didasarkan pada area pengiriman yang telah terbentuk secara langsung berdasarkan algoritma yang digunakan. Oleh karena itu, penelitian selanjutnya, disarankan untuk menerapkan metode *clustering* terlebih dahulu dalam membentuk area pengiriman berdasarkan pola rutin harian pengiriman yang dilakukan oleh masing-masing driver di perusahaan, sehingga perencanaan rute dapat disesuaikan dengan kondisi operasional actual.
   - Penelitian selanjutnya juga dapat mempertimbangkan penggunaan berbagai jenis kendaraan dalam proses distribusi, tidak terbatas hanya pada satu jenis kendaraan seperti Blindvan. Selain itu, pada algoritma ALNS dapat menambahkan *destroy* dan *repair* operator lainnya sehingga ALNS memiliki eksplorasi yang lebih luas sehingga solusi yang dihasilkan lebih optimal.

DAFTAR PUSTAKA

Adhana, D. M., Rivani & Hendriyani, C., 2023. Analysis of Third Party Logistic Service in Indonesia. *Image: Jurnal Riset Manajemen,* pp. 226-232.

Anujuprana, A. H., 2015. Shipping and Transport Optimization on Vehicle Routing. *European Journal of Business and Management* .

Arwini, N. P. D. & Juniastra, I. M., 2023. PERAN TRANSPORTASI DALAM DUNIA INDUSTRI. *VASTUWIDY,* Volume 6.

Azi, N., Gendreau, M. & Potvin, J.-Y., 2014. An adaptive large neighborhood search for a vehicle routing problem with multiple routes. *Computers & Operations Research,* pp. 167-173.

Cokrowibowo, S., Ismail & Indra, 2019. Multiple Traveling Salesman Problem menggunakan Algoritma Ant Colony Optimization dengan Operasi Elitism. *JCIS (Journal of Computer and Information System),* Volume 1, pp. 23-28.

Fadhlullah, A. D., Ekowati, T. & Mukson, 2018. Analisis Rantai Pasok (Supply Chain) Kedelai di UD Adem Ayem Kecamatan Pulokulon Kabupaten Grobogan. *BISE: Jurnal Pendidikan Bisnis dan Ekonomi,* Volume 4 No 2, pp. 86-95.

Ferone, D., Festa, P., Fugaro, S. & Pastore, T., 2024. Hybridizing a matheuristic with ALNS for the optimal collection and delivery of medical specimens. *INTERNATIONAL TRANSACTIONS INOPERATIONAL RESEARCH,* pp. 90-116.

Firmansyah, Y. S., Novianingsih, K. & Husain, H. S., 2021. Penyelesaian Capacitated Vehicle Routing Problem Menggunakan. *Jurnal EurekaMatika.*

Goni, A. G., Palandeng, I. D. & Pondaag, J. J., 2022. ANALISIS RANTAI PASOK (SUPPLY CHAIN) MINUMAN CAP TIKUS (STUDI PADA PETANI DESA PALAMBA KECAMATAN LANGOWAN SELATAN). *Jurnal EMBA,* Volume 10 No.2, pp. 358-367.

Gunawan, A., Widjaja, u. T., Vansteenwegen, P. & Yu, V. F., 2020. Adaptive Large Neighborhood Search for Vehicle Routing Problem with Cross-Docking. *IEEE World Congress,* pp. 1-8.

Gupta, Radha, Singh, B. & Pandey, D., 2010. Multi-Objective Fuzzy Vehicle Routing Problem: A Case Study. *Int. J. Contemp. Math.,* Volume 5, pp. 1439-54.

Hayati, E. N., 2014. SUPPLY CHAIN MANAGEMENT (SCM) DAN LOGISTIC MANAGEMENT. *Jurnal DINAMIKA TEKNIK,* Volume 8, p. .25 – 34.

Hilman, T. & Abdillah, B., 2021. ANALISIS KOMPARATIF BIAYA ANGKUTAN MILIK SENDIRI DENGAN SEWA DI PT POS LOGISTIK INDONESIA. *Jurnal Logistik Bisnis,* pp. 25-32.

Kadriyani, E., Mislinawati & Aksarina, 2022. PENERAPAN BIAYA DIFERENSIAL DALAM RANGKA MENERIMA ATAU MENOLAK PESANAN KHUSUS PADA KUPI BROWNIES ATJEH, BANDA ACEH. *Jurnal Ilmiah Akuntansi,* Volume 09 No 2, pp. 114-128.

Kallehauge, B., Larsen, J., Madsen, O. B. & Solomon, M. M., 2005. VEHICLE ROUTING PROBLEM WITH *TIME WINDOWS*. In: *Column Generation.* s.l.:Springer, pp. 67-98.

Kasengkang, R. A., Nangoy, S. & Sumarauw, J., 2016. ANALISIS LOGISTIK (STUDI KASUS PADA PT. REMENIA SATORI TEPAS-KOTA MANADO). *Jurnal Berkala Ilmiah Efisiensi,* Volume 16, pp. 750-759.

Khisty, M. & Katare, L. J., 2023. *Third-party Logistics (3PL) Market*, s.l.: Allied Market Research.

Li, H. & Lim, A., 2003. Local search with annealing-like restarts to solve the VRPTW. *European Journal of Operational Research,* pp. 115-127.

Liu, L., Chen, Y. & Li, A., 2024. Optimization of Distribution Routes in Agricultural Product Supply Chain Decision Management Based on Improved ALNS Algorithm. *International Journal of Advanced Computer Science and Applications,* pp. 269-278.

Lutz, R., 2014. *Adaptive Large Neighborhood Search, A heuristic for the Rich Pickup and Delivery Problem with Time windows.* Germany: Ulm University.

Mahmud, N. & Haque Mokammel, 2019. Solving Multiple Depot Vehicle Routing Problem (MDVRP) using Genetic Algorithm. *International Conference on Electrical, Computer and Communication Engineering (ECCE).*

Mara, S. T. W. et al., 2022. A survey of adaptive large neighborhood search algorithms and applications. *Computers and Operations Research* .

Maroof, A., Ayvaz, B. & Naeem, K., 2024. Logistics Optimization Using Hybrid Genetic Algorithm (HGA): A Solution to the Vehicle Routing Problem With *Time windows* (VRPTW). *IEEE.*

Mehdi, N., Abdelmountalib, M. & Imad, H., 2021. A modified ALNS algorithm for vehicle routing. *Journal of Physics: Conference Series,* pp. 1-10.

Mohammed, M. A. et al., 2017. Solving vehicle routing problem by using improved genetic algorithm for optimal solution. *Journal of Computational Science.*

Octaviany, T. & Gunawan, A., 2023. Mengoptimalkan Manajemen Persediaan Melalui Teknologi Rantai. *Journal Of Informatics And Busisnes,* Volume Vol. 01 No. 03, pp. 150-155.

Paillin, D. B., Tupan , J. M. & Utami Putri, R. A., 2019. PENERAPAN ALGORITMA DIFFERENTIAL EVOLUTION UNTUK PENYELESAIANPERMASALAHAN CAPACITATED VEHICLE ROUTING PROBLEM (CVRP). *PT. Paris Jaya Mandiri,* pp. 147-153.

Pasha, J. et al., 2022. Exact and Metaheuristic Algorithms for the Vehicle Routing Problem with a Factory-in-aBox in Multi-Objective Settings. *Elsevier,* pp. 1-33.

Patmawati, H. & Nugroho, Y. A., 2022. OPTIMALISASI RUTE DISTRIBUSI MATRAS PADA PENYELESAIAN CAPACITATED. *Jurnal Cakrawala Ilmiah.*

Permana, D. S., Febriyanto, A. S. & Revansa, R. R., 2024. Peran Strategis Manajemen Logistik dalam Keunggulan Bersaing Perusahaan. *Jurnal Manajemen dan SIstem Informasi,* pp. 85-96.

Permana, K. B. P., 2023. Analisis Peran Supply Chain Management (Scm) Dalam Meningkatkan Kinerja Perusahaan. *Jurnal Penelitian dan Pengabdian Masyarakat,* pp. 2275-2287.

Pramestiana, I. & Anwar, A., 2025. Penentuan Rute Terpendek pada Pengiriman Kendaraan Motor menggunakan Algoritma Dijkstra di PT X. *Jurnal Logic: Logistics & Supply Chain Center,* Volume 03.

Pramudita, A. S., 2020. Pengukuran Kualitas Pelayanan dan Kepuasan Pelanggan Terhadap Loyalitas Pelanggan pada Industri Logistik. *Jurnal Logistik Bisnis,* pp. 15-21.

Prawira, H. A. & Santosa, B., 2021. Development of Particle Swarm Optimization and Simulated Annealing Algorithms to. *Prozima.*

Pugliese, L. d. P. et al., 2024. An adaptive large neighborhood search for the order picking process: the case of a retail distribution company in italy. *Procedia Computer Science,* pp. 2606-2615.

Puspitaningtyas, A. & ABdulrahim, M., 2023. PENERAPAN METODE PERAMALAN SEBAGAI UPAYA MENGURANGI BULLWHIP EFFECT PADA UMKM PENTOL X. *Jurnal Taguchi: Jurnal Ilmiah Teknik dan Manajemen Industri,* pp. 761-766.

Rahayu, W. I., Riza, N. & Ramadhan, N., 2019. Aplikasi Estiquent Untuk Estimasi Biaya Transportasi Logistik Di PT.Sukarasa Menggunakan Algoritma North West Corner. *Jurnal Teknik Informatika,* Volume 11, pp. 7-11.

Redi, A. A. N. P. et al., 2020. A Simulated Annealing Algorithm for Solving Two-Echelon Vehicle Routing Problem with Locker Facilities. *algorithms,* pp. 1-14.

Sajid, M., Jafar, A. & Surbhi, S., 2020. Hybrid Genetic and Simulated Annealing Algorithm for Capacitated Vehicle Routing Problem. *Sixth International Conference on Parallel.*

Saogo, W., Yanti, S. & Mursal, 2024. Pengaruh Kualitas Layanan, Kualitas Informasi, dan Presepsi Harga Terhadap Kepuasan Pelanggan (Studi Kasus pada Pengguna Aplikasi Pos Aja! PT. Pos Indonesia Cabang Batam). *Zona Manajerial,* Volume 14, pp. 94-107.

Shuen, H. N. & Johar, F., 2023. Optimization of Capacitated Vehicle Routing Problem for Municipal Solid Waste Collection Using Simulated Annealing Algorithm. *Proc. Sci.*.

Sugiyono, 2011. Metode Penelitian Kuantitatif, Kualitatif dan R&D. *Bandung. Alfabeta.*

Susetyo, D. P., Firmansyah, D. & Aisah, A. S., 2021. Perubahan Biaya Operasional Dan Jumlah Penjualan Terhadap Laba Bersih Sebelum Dan Pada Masa Pandemic Covid-19 (Studi Pada PT. Nuansa Ilham Prima Sukabumi). *Jurnal Sekuritas,* Volume 5, pp. 208-219.

Tan, S.-Y. & Yeh, W.-C., 2021. The Vehicle Routing Problem: State-of-the-Art Classification and Review. *Applied Sciences,* pp. 1-28.

Toth P & Vigo D, 2002. An Overview of vehicle routing problems.. *Philadelphia: Siam.,* pp. 1-26.

Wasana, C. & Sukto, S., 2021. A hybrid FJA-ALNS algorithm for solving the multi-compartment vehicle routing problem. *Decision Science Letters,* pp. 497-510.

Widyastiti, M. & Sumarsa, A., 2023. IMPLEMENTASI SIMULTANEOUS PICK UP AND DELIVERY VEHICLE ROUTING PROBLEM WITH *TIME WINDOWS* MENGGUNAKAN INTEGER NON-LINEAR PROGRAMMING. *JURNAL PENDIDIKAN MATEMATIKA DAN SAINS,* Volume 04, pp. 084-093.

Wijaya, H. M., Deswantoro, G. & Hidayat, R., 2021. ANALISIS PERENCANAAN SUPPLY CHAIN MANAGEMENT (SCM) PADA PT. KYLO KOPI INDONESIA. *Jurnal Ekonomi Manajemen Sistem Informasi,* 2(6), pp. 795-806.

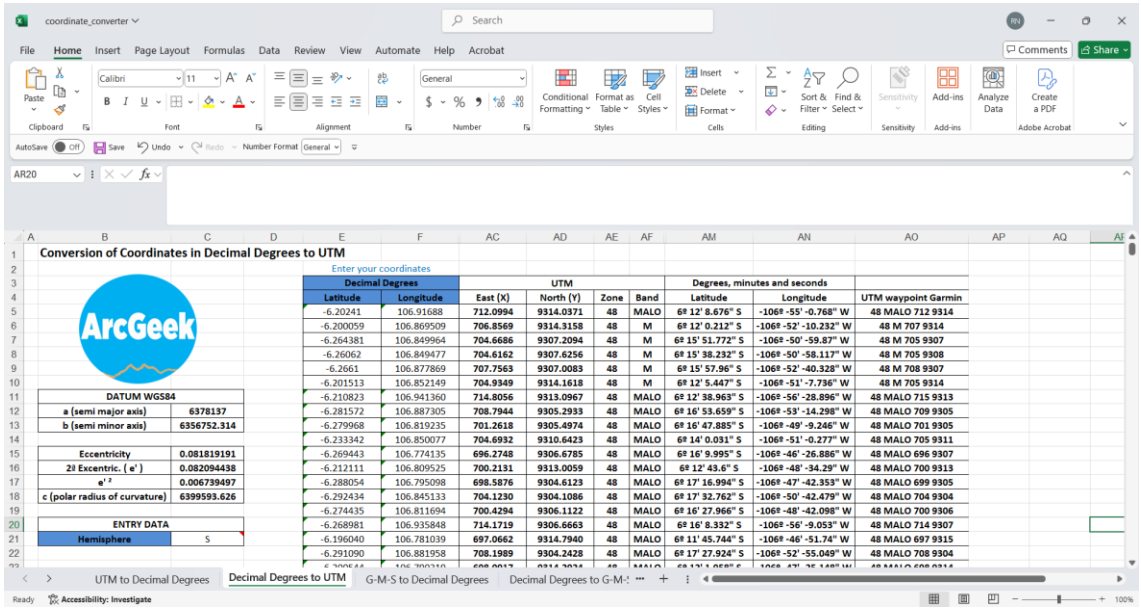
Xu, W., Zhang, . C., Cheng, M. & Huang, Y., 2022. Electric Vehicle Routing Problem with Simultaneous Pickup and Delivery: Mathematical Modeling and Adaptive Large Neighborhood Search Heuristic Method. *Energies,* pp. 1-25.

Yu, V. F. et al., 2022. A Simulated Annealing Algorithm for the Vehicle Routing Problem With Parcel Lockers. *IEEE.*

Zai, I. et al., 2022. Perbandingan Elemen Model Bisnis Third-Party Logistics (3PL)dan Fourth-Party Logistics (4PL) Pada Logistik Angkutan Laut. *Jurnal Ilmu Komputer dan Bisnis (JIKB),* Volume 8, pp. 110-121 .

LAMPIRAN

A - *Coordinate Converter* (*Latitude Longitude* – UTM)



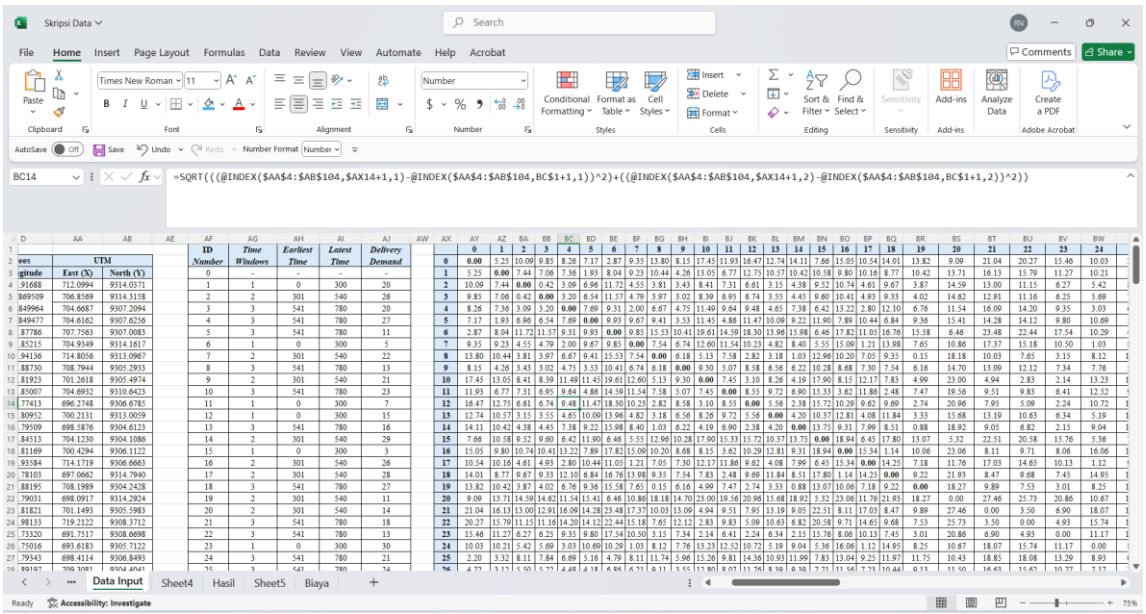
Conversion of Coordinates in Decimal Degrees to UTM

Enter your coordinates

Decimal Degrees		UTM				Degrees, minutes and seconds		
Latitude	Longitude	East (X)	North (Y)	Zone	Band	Latitude	Longitude	UTM waypoint Garmin
-6.20241	106.91688	712.0994	9314.0371	48	MALO	6º 12' 8.676" S	-106º -55' -0.768" W	48 MALO 712 9314
-6.200059	106.869509	706.8569	9314.3158	48	M	6º 12' 0.212" S	-106º -52' -10.232" W	48 M 707 9314
-6.264381	106.849964	704.6686	9307.2094	48	M	6º 15' 51.772" S	-106º -50' -59.87" W	48 M 705 9307
-6.26062	106.849477	704.6162	9307.6256	48	M	6º 15' 38.232" S	-106º -50' -58.117" W	48 M 705 9308
-6.2661	106.877869	707.7563	9307.0083	48	M	6º 15' 57.96" S	-106º -52' -40.328" W	48 M 708 9307
-6.201513	106.852149	704.9349	9314.1618	48	M	6º 12' 5.447" S	-106º -51' -7.736" W	48 M 705 9314
-6.210823	106.941360	714.8056	9313.0967	48	MALO	6º 12' 38.963" S	-106º -56' -28.896" W	48 MALO 715 9313
-6.281572	106.887305	708.7944	9305.2933	48	MALO	6º 16' 53.659" S	-106º -53' -14.298" W	48 MALO 709 9305
-6.279968	106.819235	701.2618	9305.4974	48	MALO	6º 16' 47.885" S	-106º -49' -9.246" W	48 MALO 701 9305
-6.233342	106.850077	704.6932	9310.6423	48	MALO	6º 14' 0.031" S	-106º -51' -0.277" W	48 MALO 705 9311
-6.269443	106.774135	696.2748	9306.6785	48	MALO	6º 16' 9.995" S	-106º -46' -26.886" W	48 MALO 696 9307
-6.212111	106.809525	700.2131	9313.0059	48	MALO	6º 12' 43.6" S	-106º -48' -34.29" W	48 MALO 700 9313
-6.288054	106.795098	698.5876	9304.6123	48	MALO	6º 17' 16.994" S	-106º -47' -42.353" W	48 MALO 699 9305
-6.292434	106.845133	704.1230	9304.1086	48	MALO	6º 17' 32.762" S	-106º -50' -42.479" W	48 MALO 704 9304
-6.274435	106.811694	700.4294	9306.1122	48	MALO	6º 16' 27.966" S	-106º -48' -42.098" W	48 MALO 700 9306
-6.268981	106.935848	714.1719	9306.6663	48	MALO	6º 16' 8.332" S	-106º -56' -9.053" W	48 MALO 714 9307
-6.196040	106.781039	697.0662	9314.7940	48	MALO	6º 11' 45.744" S	-106º -46' -51.74" W	48 MALO 697 9315
-6.291090	106.881958	708.1989	9304.2428	48	MALO	6º 17' 27.924" S	-106º -52' -55.049" W	48 MALO 708 9304

DATUM WGS84	
a (semi major axis)	6378137
b (semi minor axis)	6356752.314
Eccentricity	0.081819191
2ª Excentric. (e')	0.082094438
e' ²	0.006739497
c (polar radius of curvature)	6399593.626
ENTRY DATA	
Hemisphere	S

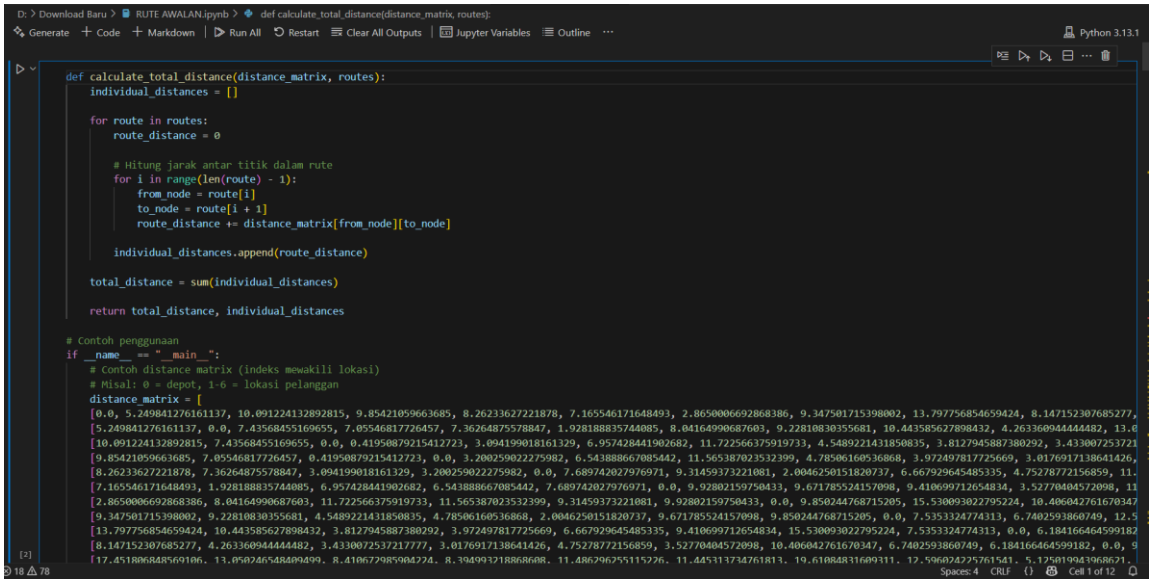
B - Pengolahan Matriks Jarak



	ID Number	Time Windows	Earliest Time	Latest Time	Delivery Demand
ees	UTM				
gitude	East (X)	North (Y)			
.91688	712.0994	9314.0371	1	1	0	300	20
869509	706.8569	9314.3158	2	2	301	540	26
849964	704.6687	9307.2094	3	3	541	780	20
849477	704.6162	9307.6216	4	3	541	780	27
.87786	707.7563	9307.0083	5	3	541	780	11
.85215	704.9349	9314.1617	6	1	0	300	5
.94136	714.8056	9313.0967	7	2	301	540	22
.88730	708.7944	9305.2933	8	3	541	780	13
.81923	701.2618	9305.4974	9	2	301	540	21
.85007	704.6932	9310.6423	10	3	541	780	23
.77413	696.2748	9306.6785	11	1	0	300	7
.80952	700.2131	9313.0059	12	1	0	300	15
.79509	698.5876	9304.6123	13	3	541	780	16
.84513	704.1230	9304.1086	14	2	301	540	29
.81169	700.4294	9306.1122	15	1	0	300	3
.93584	714.1719	9306.6663	16	2	301	540	26
.78103	697.0682	9314.7940	17	2	301	540	28
.88195	708.1989	9304.2428	18	3	541	780	27
.79031	698.0917	9314.2924	19	2	301	540	11
.81821	701.1493	9305.5983	20	2	301	540	14
.98133	719.2122	9308.3712	21	3	541	780	18
.73320	691.7517	9308.6698	22	3	541	780	13
.75016	693.6183	9305.7122	23	1	0	300	30
.79343	698.4114	9306.8493	24	3	541	780	21

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	**0.00**	5.25	10.09	9.85	8.26	7.17	2.87	9.35	13.80	8.15	17.45	11.93	16.47	12.74	14.11	7.66	15.05	10.54	14.01	13.82	9.09	21.04	20.27	15.46	10.03
1	5.25	**0.00**	7.44	7.06	7.36	1.93	8.04	9.23	10.44	4.26	13.05	6.77	12.75	10.57	10.42	10.58	9.80	10.16	8.77	10.42	13.71	16.13	15.79	11.27	10.21
2	10.09	7.44	**0.00**	0.42	3.09	6.96	11.72	4.55	3.81	3.43	8.41	7.31	6.61	3.15	4.38	9.52	10.74	4.61	9.67	3.87	14.59	13.00	11.15	6.27	5.42
3	9.85	7.06	0.42	**0.00**	3.20	6.54	11.57	4.79	3.97	3.02	8.39	6.95	6.74	3.55	4.45	9.60	10.41	4.93	9.33	4.02	14.62	12.91	11.16	6.25	5.69
4	8.26	7.36	3.09	3.20	**0.00**	7.69	9.31	2.00	6.67	4.75	11.49	9.64	9.48	4.65	7.38	6.42	13.22	2.80	12.10	6.76	11.54	16.09	14.20	9.35	3.03
5	7.17	1.93	6.96	6.54	7.69	**0.00**	9.93	9.67	9.41	3.53	11.45	4.86	11.47	10.09	9.22	11.90	7.89	10.44	0.84	9.36	15.41	14.28	14.12	9.80	10.69
6	2.87	8.04	11.72	11.57	9.31	9.93	**0.00**	9.85	15.53	10.41	19.61	14.59	18.30	13.96	15.98	6.46	17.82	11.05	16.76	15.58	6.46	23.48	22.44	17.54	10.29
7	9.35	9.23	4.55	4.79	2.00	9.67	9.85	**0.00**	7.54	6.74	12.60	11.54	10.23	4.82	8.40	5.55	15.09	1.21	13.98	7.65	10.86	17.37	15.18	10.50	1.03
8	13.80	10.44	3.81	3.97	6.67	9.41	15.53	7.54	**0.00**	6.18	5.13	7.58	2.82	3.18	1.03	12.96	10.20	7.05	9.35	0.15	18.18	10.03	7.65	3.15	8.12
9	8.15	4.26	3.43	3.02	4.75	3.53	10.41	6.74	6.18	**0.00**	9.30	5.07	8.58	6.56	6.22	10.28	8.68	7.30	7.54	6.16	14.70	13.09	12.12	7.34	7.76
10	17.45	13.05	8.41	8.39	11.49	11.45	19.61	12.60	5.13	9.30	**0.00**	7.45	3.10	8.26	4.19	17.90	8.15	12.17	7.83	4.99	23.00	4.94	2.83	2.14	13.23
11	11.93	6.77	7.31	6.95	9.64	4.86	14.59	11.54	7.58	5.07	7.45	**0.00**	8.55	9.72	6.90	15.33	3.62	11.86	2.48	7.47	19.56	9.51	9.83	6.41	12.52
12	16.47	12.75	6.61	6.74	9.48	11.47	18.30	10.23	2.82	8.58	3.10	8.55	**0.00**	5.56	2.38	15.72	10.29	9.62	9.69	2.74	20.96	7.95	5.09	2.24	10.72
13	12.74	10.57	3.15	3.55	4.65	10.09	13.96	4.82	3.18	6.56	8.26	9.72	5.56	**0.00**	4.20	10.37	12.81	4.08	11.84	3.33	15.68	13.19	10.63	6.34	5.19
14	14.11	10.42	4.38	4.45	7.38	9.22	15.98	8.40	1.03	6.22	4.19	6.90	2.38	4.20	**0.00**	13.75	9.31	7.99	8.51	0.88	18.92	9.05	6.82	2.15	9.04
15	7.66	10.58	9.52	9.60	6.42	11.90	6.46	5.55	12.96	10.28	17.90	15.33	15.72	10.37	13.75	**0.00**	18.94	6.45	17.80	13.07	5.32	22.51	20.58	15.76	5.36
16	15.05	9.80	10.74	10.41	13.22	7.89	17.82	15.09	10.20	8.68	8.15	3.62	10.29	12.81	9.31	18.94	**0.00**	15.34	1.14	10.06	23.06	8.11	9.71	8.06	16.06
17	10.54	10.16	4.61	4.93	2.80	10.44	11.05	1.21	7.05	7.30	12.17	11.86	9.62	4.08	7.99	6.45	15.34	**0.00**	14.25	7.18	11.76	17.03	14.65	10.13	1.12
18	14.01	8.77	9.67	9.33	12.10	6.84	16.78	13.98	9.35	7.54	7.83	2.48	9.69	11.84	8.51	17.80	1.14	14.25	**0.00**	9.22	21.93	8.47	9.68	7.45	14.95
19	13.82	10.42	3.87	4.02	6.76	9.36	15.58	7.65	0.15	6.16	4.99	7.47	2.74	3.33	0.88	13.07	10.06	7.18	9.22	**0.00**	18.27	9.89	7.53	3.01	8.25
20	9.09	13.71	14.59	14.62	11.54	15.41	6.46	10.86	18.18	14.70	23.00	19.56	20.96	15.68	18.92	5.32	23.06	11.76	21.93	18.27	0.00	27.46	25.73	20.86	10.67
21	21.04	16.13	13.00	12.91	16.09	14.28	23.48	17.37	10.03	13.09	4.94	9.51	7.95	13.19	9.05	22.51	8.11	17.03	8.47	9.89	27.46	0.00	3.50	6.90	18.07
22	20.27	15.79	11.15	11.16	14.20	14.12	22.44	15.18	7.65	12.12	2.83	9.83	5.09	10.63	6.82	20.58	9.71	14.65	9.68	7.53	25.73	3.50	0.00	4.93	15.74
23	15.46	11.27	6.27	6.25	9.35	9.80	17.54	10.50	3.15	7.34	2.14	6.41	2.24	6.34	2.15	15.76	8.06	10.13	7.45	3.01	20.86	6.90	4.93	0.00	11.17
24	10.03	10.21	5.42	5.69	3.03	10.69	10.29	1.03	8.12	7.76	13.23	12.52	10.72	5.19	9.04	5.36	16.06	1.12	14.95	8.25	10.87	18.07	15.74	11.17	0.00
25	2.20	3.32	8.11	7.84	6.69	5.16	4.79	8.11	11.74	5.96	15.26	9.81	14.36	10.93	11.99	7.83	13.04	9.25	11.97	11.75	10.43	18.85	18.08	13.29	8.93

C - *Pseudocode* Perhitungan Jarak Rute Awal



```
def calculate_total_distance(distance_matrix, routes):
    individual_distances = []

    for route in routes:
        route_distance = 0

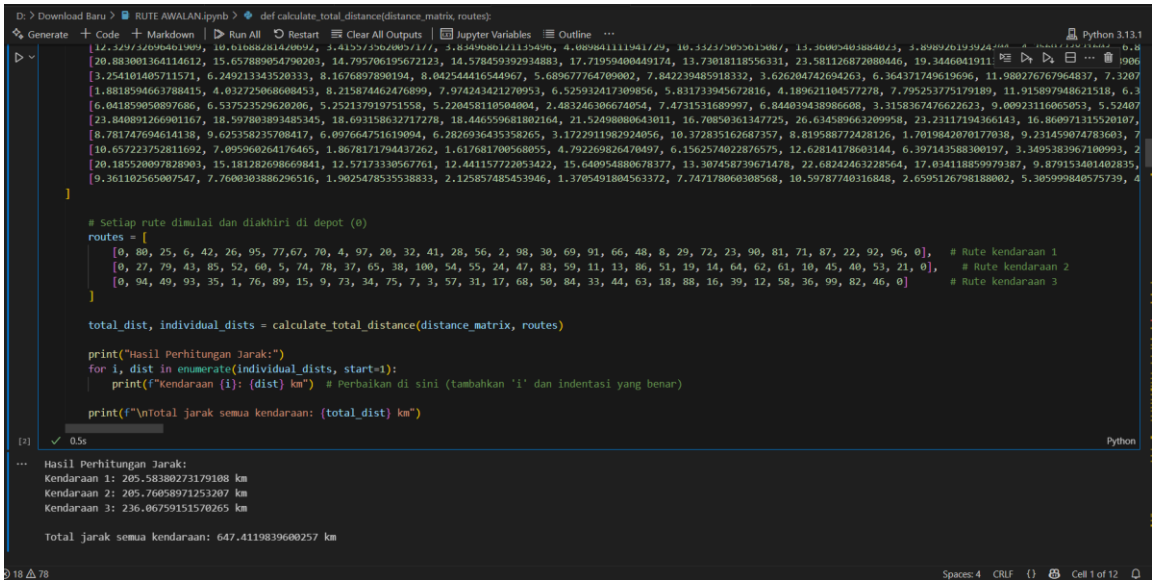
        # Hitung jarak antar titik dalam rute
        for i in range(len(route) - 1):
            from_node = route[i]
            to_node = route[i + 1]
            route_distance += distance_matrix[from_node][to_node]

        individual_distances.append(route_distance)

    total_distance = sum(individual_distances)

    return total_distance, individual_distances

# Contoh penggunaan
if __name__ == "__main__":
    # Contoh distance matrix (indeks mewakili lokasi)
    # Misal: 0 = depot, 1-6 = lokasi pelanggan
    distance_matrix = [
    [0.0, 5.249841276161137, 10.091224132892815, 9.85421059663685, 8.26233627221878, 7.165546171648493, 2.8650006692868386, 9.347501715398002, 13.797756854659424, 8.147152307685277,
    [5.249841276161137, 0.0, 7.435684551696555, 7.05546817726457, 7.36264875578847, 1.928188835744085, 8.04164990687603, 9.22810830355681, 10.443585627898432, 4.263360944444482, 13.8
    [10.091224132892815, 7.435684551696555, 0.0, 0.41950879215412723, 3.094199018161329, 6.957428441902682, 11.722566375919733, 4.548922143850835, 3.8127945887380292, 3.433007253721
    [9.85421059663685, 7.05546817726457, 0.41950879215412723, 0.0, 3.200259022275982, 6.543888667085442, 11.565387023532399, 4.785061605368 68, 3.972497817725669, 3.0176917138641426,
    [8.26233627221878, 7.36264875578847, 3.094199018161329, 3.200259022275982, 0.0, 7.689742027976971, 9.31459373221081, 2.0046250151820737, 6.667929645485335, 4.75278772156859, 11.
    [7.165546171648493, 1.928188835744085, 6.957428441902682, 6.543888667085442, 7.689742027976971, 0.0, 9.92802159750433, 9.671785524157098, 9.410699712654834, 3.527704045720 98, 11
    [2.8650006692868386, 8.04164990687603, 11.722566375919733, 11.565387023532399, 9.314593732210 81, 9.92802159750433, 0.0, 9.850244768715205, 15.530093022795224, 10.406042761670347
    [9.347501715398002, 9.228108303556 81, 4.548922143185 0835, 4.785061605368 68, 2.0046250151820737, 9.671785524157098, 9.850244768715205, 0.0, 7.535332477431 3, 6.740259386 0749, 12.5
    [13.797756854659424, 10.443585627898432, 3.8127945887380292, 3.972497817725669, 6.667929645485335, 9.410699712654834, 15.530093022795224, 7.535332477431 3, 0.0, 6.184166464599182
    [8.147152307685277, 4.263360944444482, 3.433007253721 7777, 3.0176917138641426, 4.752787721568 59, 3.527704045720 98, 10.406042761670347, 6.740259386 0749, 6.184166464599182, 0.0, 9
    [17.451806848569106, 13.050246548409499, 8.410672985904274, 8.394993218868608, 11.486296255115226, 11.445313734761813, 19.610848316093 11, 12.596024225761541, 5.125019941960621,
```



```
    [12.329732696461989, 10.618882814206 92, 3.415573562005717, 3.834968812115496, 4.0898411119417 29, 10.332375055615087, 13.360054038840 23, 3.898926193924
    [20.883001364114612, 15.657889054790203, 14.795706195672123, 14.578459392934883, 17.71959400449174, 13.730181118556331, 23.581126872080446, 19.344604191 1
    [3.2541014057115 71, 6.249213343520333, 8.167689789019 4, 8.042544416544967, 5.689677764709002, 7.842239485918332, 3.626204742694263, 6.364371749619696, 11.980276767964837, 7.3207
    [1.8818594663788415, 4.032725068608453, 8.215874462476899, 7.974243421270953, 6.525932417309856, 5.831733945672816, 4.189621104577278, 7.795253775179189, 11.915897948621518, 6.3
    [6.041859050897686, 6.537523529620206, 5.252137971975 1558, 5.220458110504004, 2.483246306674054, 7.473153168999 7, 6.844039438986608, 3.315836747662262 3, 9.009231160650 53, 5.52407
    [23.840891266901167, 18.597803893485345, 18.693158632717278, 18.446559681802164, 21.524980806430 11, 16.708503613477 25, 26.634589663209958, 23.231171943661 43, 16.860971315520107,
    [8.781747694614138, 9.625358235708417, 6.097664751619094, 6.282693643535 8265, 3.172291198292 4056, 10.372835162687357, 8.819588772428126, 1.701984207017 7038, 9.231459074783603, 7
    [10.657223752811692, 7.095960264176465, 1.867817179443 7262, 1.617681700568055, 4.792269826470497, 6.156257402287 6575, 12.628141786031 44, 6.397143588300197, 3.349538396710 0993, 2
    [20.185520097828903, 15.181282698669841, 12.571733305677 61, 12.441157722053422, 15.640954880678377, 13.307458739671478, 22.682424632285 64, 17.034118859979387, 9.879153401402835,
    [9.361102565007547, 7.760030388629651 6, 1.902547853553 8833, 2.125857485453946, 1.370549180456 3372, 7.747178060308568, 10.597877403168 48, 2.659512679818 8002, 5.305999840575739, 4
]

    # Setiap rute dimulai dan diakhiri di depot (0)
    routes = [
        [0, 80, 25, 6, 42, 26, 95, 77,67, 70, 4, 97, 20, 32, 41, 28, 56, 2, 98, 30, 69, 91, 66, 48, 8, 29, 72, 23, 90, 81, 71, 87, 22, 92, 96, 0],   # Rute kendaraan 1
        [0, 27, 79, 43, 85, 52, 60, 5, 74, 78, 37, 65, 38, 100, 54, 55, 24, 47, 83, 59, 11, 13, 86, 51, 19, 14, 64, 62, 61, 10, 45, 40, 53, 21, 0],    # Rute kendaraan 2
        [0, 94, 49, 93, 35, 1, 76, 89, 15, 9, 73, 34, 75, 7, 3, 57, 31, 17, 68, 50, 84, 33, 44, 63, 18, 88, 16, 39, 12, 58, 36, 99, 82, 46, 0]      # Rute kendaraan 3
    ]

    total_dist, individual_dists = calculate_total_distance(distance_matrix, routes)

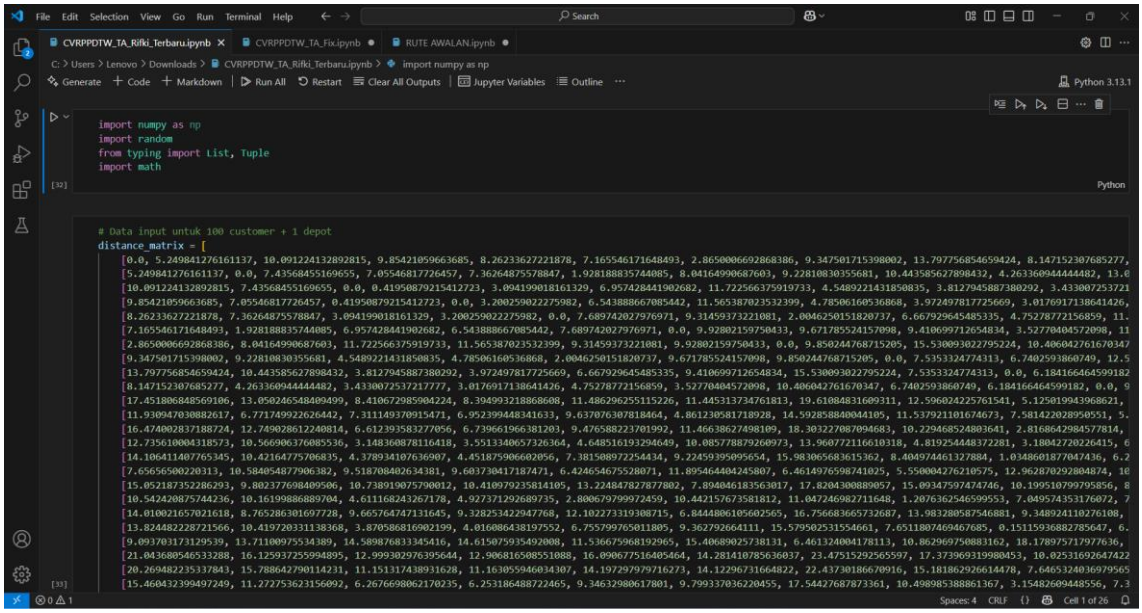
    print("Hasil Perhitungan Jarak:")
    for i, dist in enumerate(individual_dists, start=1):
        print(f"Kendaraan {i}: {dist} km")  # Perbaikan di sini (tambahkan 'i' dan indentasi yang benar)

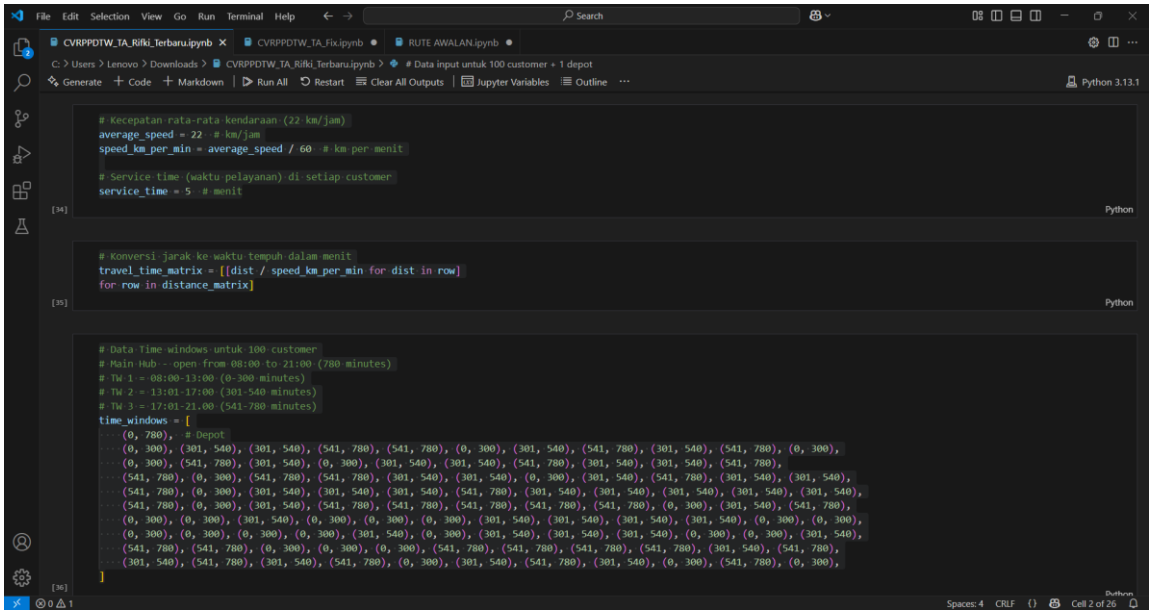
    print(f"\nTotal jarak semua kendaraan: {total_dist} km")
```

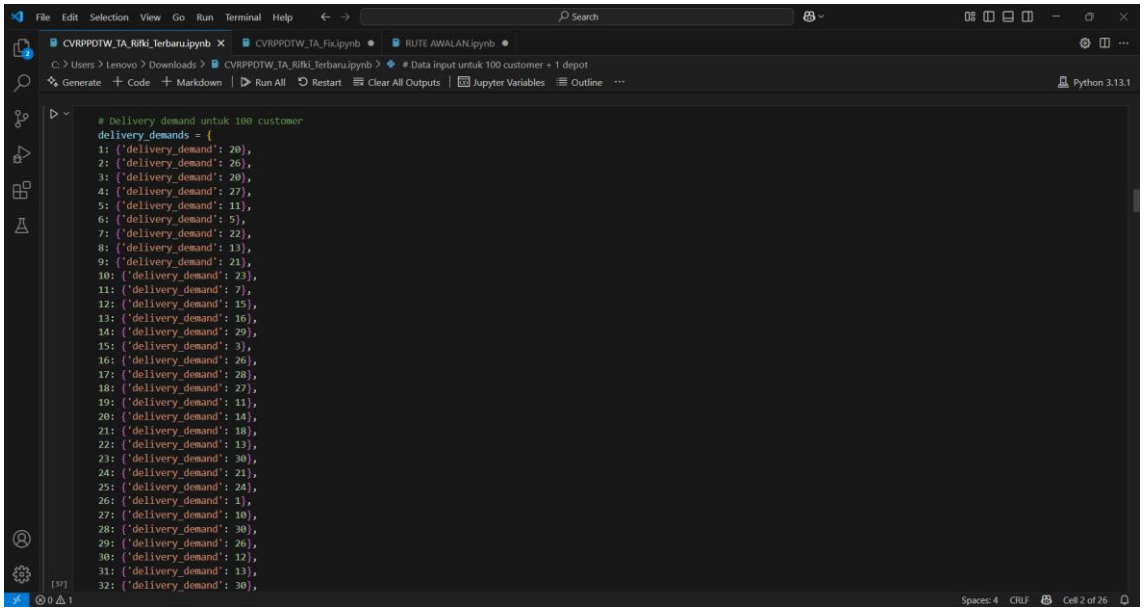
```
Hasil Perhitungan Jarak:
Kendaraan 1: 205.58380273179108 km
Kendaraan 2: 205.76058971253207 km
Kendaraan 3: 236.06759151570265 km

Total jarak semua kendaraan: 647.4119839600257 km
```

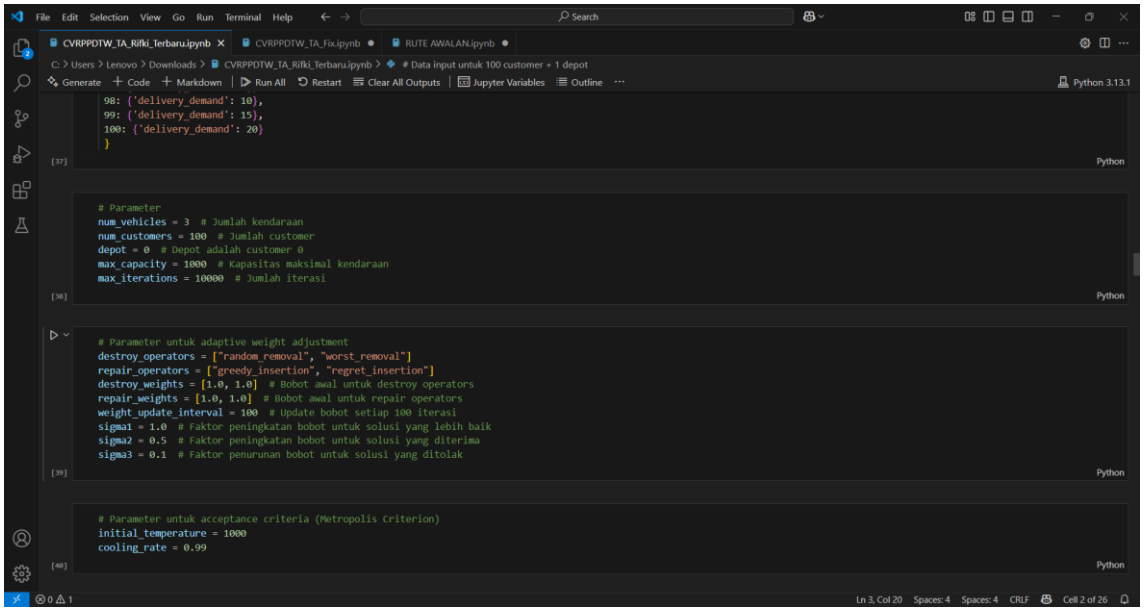
D - *Pseudocode* Algoritma ALNS - CVRPTW







```
# Delivery demand untuk 100 customer
delivery_demands = {
1: {'delivery_demand': 20},
2: {'delivery_demand': 26},
3: {'delivery_demand': 20},
4: {'delivery_demand': 27},
5: {'delivery_demand': 11},
6: {'delivery_demand': 5},
7: {'delivery_demand': 22},
8: {'delivery_demand': 13},
9: {'delivery_demand': 21},
10: {'delivery_demand': 23},
11: {'delivery_demand': 7},
12: {'delivery_demand': 15},
13: {'delivery_demand': 16},
14: {'delivery_demand': 29},
15: {'delivery_demand': 3},
16: {'delivery_demand': 26},
17: {'delivery_demand': 28},
18: {'delivery_demand': 27},
19: {'delivery_demand': 11},
20: {'delivery_demand': 14},
21: {'delivery_demand': 18},
22: {'delivery_demand': 13},
23: {'delivery_demand': 30},
24: {'delivery_demand': 21},
25: {'delivery_demand': 24},
26: {'delivery_demand': 1},
27: {'delivery_demand': 10},
28: {'delivery_demand': 30},
29: {'delivery_demand': 26},
30: {'delivery_demand': 12},
31: {'delivery_demand': 13},
32: {'delivery_demand': 30},
```

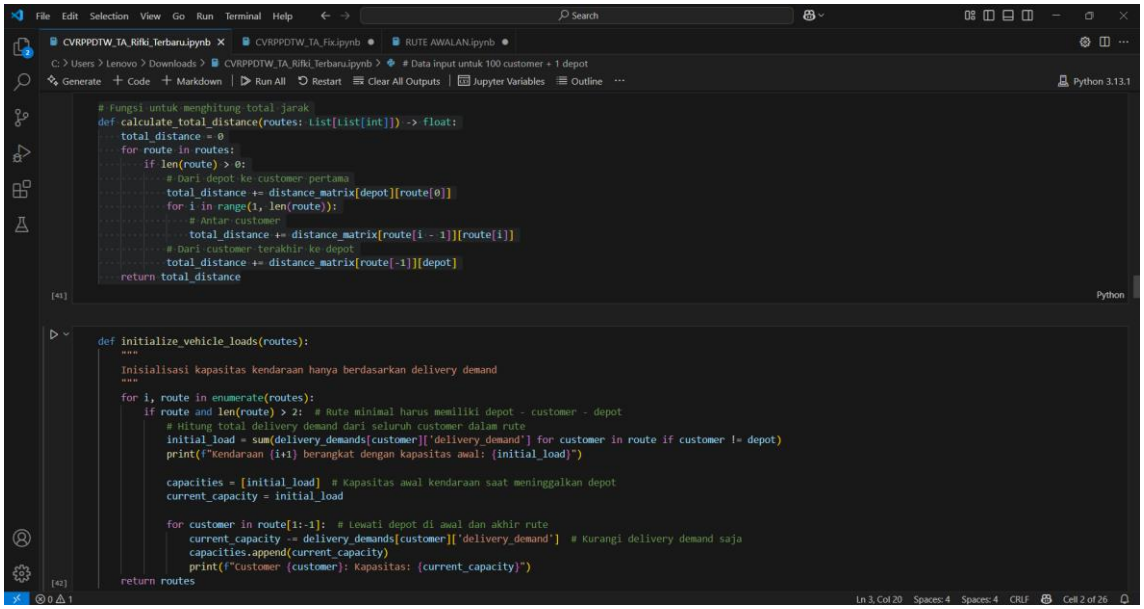


```
98: {'delivery_demand': 10},
99: {'delivery_demand': 15},
100: {'delivery_demand': 20}
}
```

```
# Parameter
num_vehicles = 3  # Jumlah kendaraan
num_customers = 100  # Jumlah customer
depot = 0  # Depot adalah customer 0
max_capacity = 1000  # Kapasitas maksimal kendaraan
max_iterations = 10000  # Jumlah iterasi
```

```
# Parameter untuk adaptive weight adjustment
destroy_operators = ["random_removal", "worst_removal"]
repair_operators = ["greedy_insertion", "regret_insertion"]
destroy_weights = [1.0, 1.0]  # Bobot awal untuk destroy operators
repair_weights = [1.0, 1.0]  # Bobot awal untuk repair operators
weight_update_interval = 100  # Update bobot setiap 100 iterasi
sigma1 = 1.0  # Faktor peningkatan bobot untuk solusi yang lebih baik
sigma2 = 0.5  # Faktor peningkatan bobot untuk solusi yang diterima
sigma3 = 0.1  # Faktor penurunan bobot untuk solusi yang ditolak
```

```
# Parameter untuk acceptance criteria (Metropolis Criterion)
initial_temperature = 1000
cooling_rate = 0.99
```

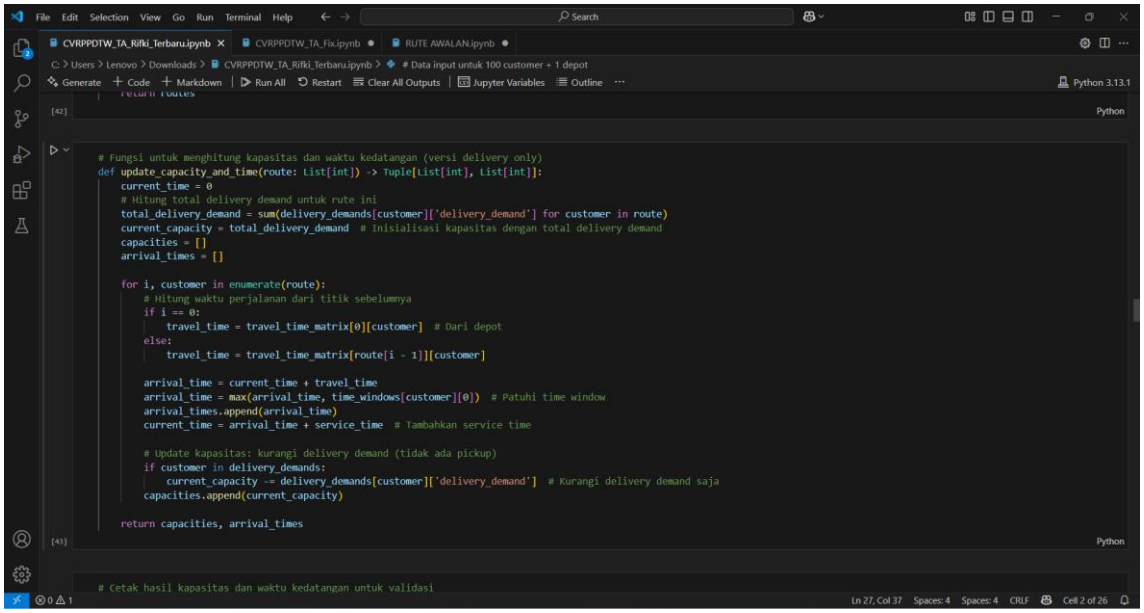


```python
# Fungsi untuk menghitung total jarak
def calculate_total_distance(routes: List[List[int]]) -> float:
    total_distance = 0
    for route in routes:
        if len(route) > 0:
            # Dari depot ke customer pertama
            total_distance += distance_matrix[depot][route[0]]
            for i in range(1, len(route)):
                # Antar customer
                total_distance += distance_matrix[route[i - 1]][route[i]]
            # Dari customer terakhir ke depot
            total_distance += distance_matrix[route[-1]][depot]
    return total_distance
```

```python
def initialize_vehicle_loads(routes):
    """
    Inisialisasi kapasitas kendaraan hanya berdasarkan delivery demand
    """
    for i, route in enumerate(routes):
        if route and len(route) > 2:  # Rute minimal harus memiliki depot - customer - depot
            # Hitung total delivery demand dari seluruh customer dalam rute
            initial_load = sum(delivery_demands[customer]['delivery_demand'] for customer in route if customer != depot)
            print(f"Kendaraan {i+1} berangkat dengan kapasitas awal: {initial_load}")

            capacities = [initial_load]  # Kapasitas awal kendaraan saat meninggalkan depot
            current_capacity = initial_load

            for customer in route[1:-1]:  # Lewati depot di awal dan akhir rute
                current_capacity -= delivery_demands[customer]['delivery_demand']  # Kurangi delivery demand saja
                capacities.append(current_capacity)
                print(f"Customer {customer}: Kapasitas: {current_capacity}")
    return routes
```



```python
# Fungsi untuk menghitung kapasitas dan waktu kedatangan (versi delivery only)
def update_capacity_and_time(route: List[int]) -> Tuple[List[int], List[int]]:
    current_time = 0
    # Hitung total delivery demand untuk rute ini
    total_delivery_demand = sum(delivery_demands[customer]['delivery_demand'] for customer in route)
    current_capacity = total_delivery_demand  # Inisialisasi kapasitas dengan total delivery demand
    capacities = []
    arrival_times = []

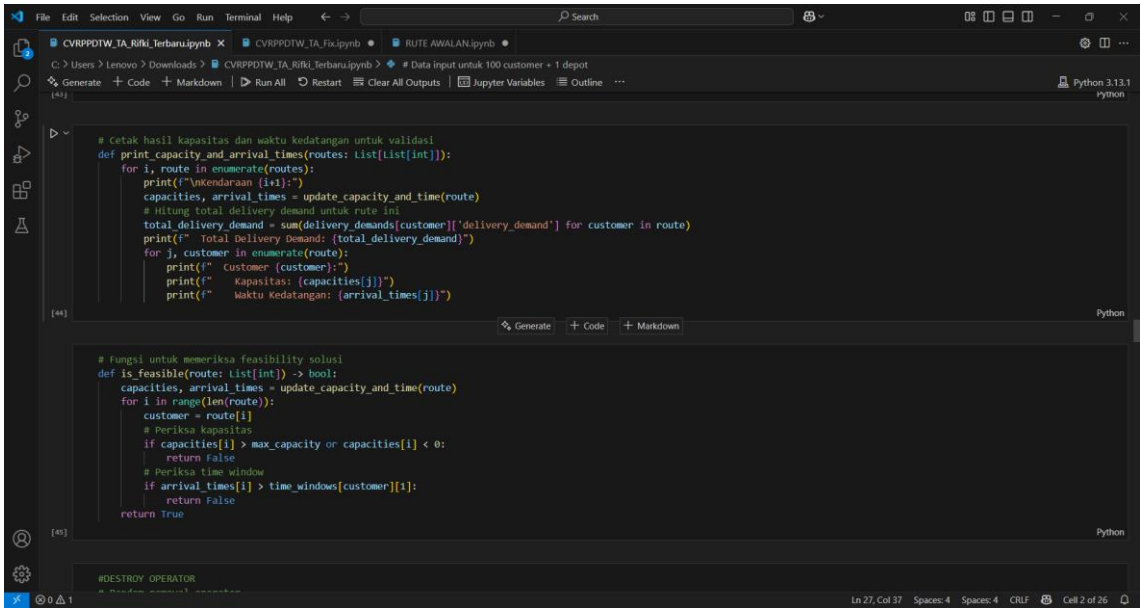
    for i, customer in enumerate(route):
        # Hitung waktu perjalanan dari titik sebelumnya
        if i == 0:
            travel_time = travel_time_matrix[0][customer]  # Dari depot
        else:
            travel_time = travel_time_matrix[route[i - 1]][customer]

        arrival_time = current_time + travel_time
        arrival_time = max(arrival_time, time_windows[customer][0])  # Patuhi time window
        arrival_times.append(arrival_time)
        current_time = arrival_time + service_time  # Tambahkan service time

        # Update kapasitas: kurangi delivery demand (tidak ada pickup)
        if customer in delivery_demands:
            current_capacity -= delivery_demands[customer]['delivery_demand']  # Kurangi delivery demand saja
        capacities.append(current_capacity)

    return capacities, arrival_times
```

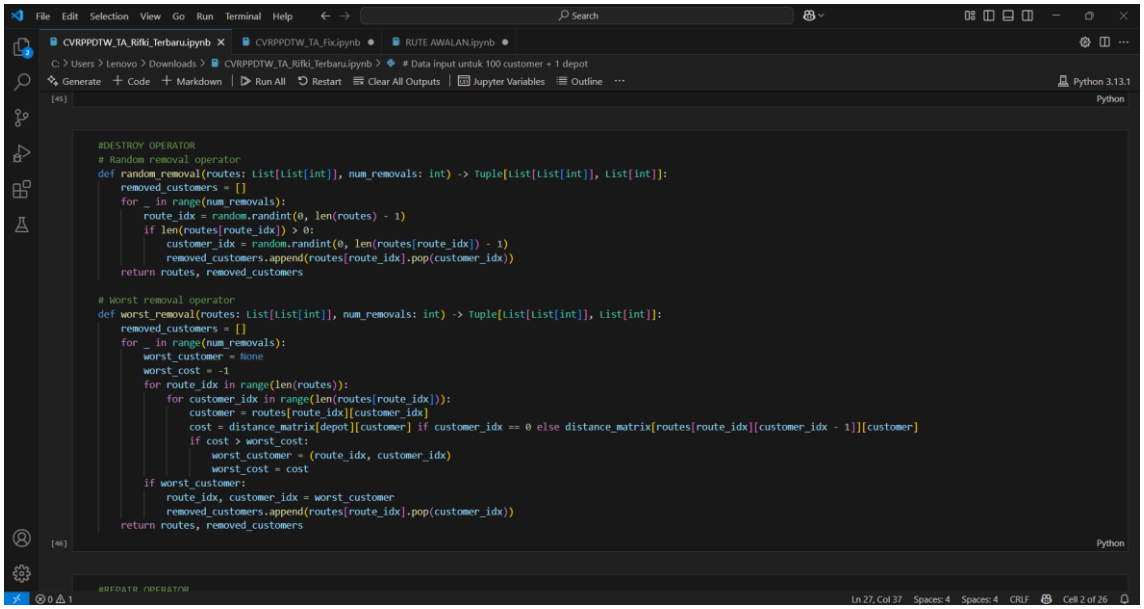
```python
# Cetak hasil kapasitas dan waktu kedatangan untuk validasi
```



```python
# Cetak hasil kapasitas dan waktu kedatangan untuk validasi
def print_capacity_and_arrival_times(routes: List[List[int]]):
    for i, route in enumerate(routes):
        print(f"\nKendaraan {i+1}:")
        capacities, arrival_times = update_capacity_and_time(route)
        # Hitung total delivery demand untuk rute ini
        total_delivery_demand = sum(delivery_demands[customer]['delivery_demand'] for customer in route)
        print(f"  Total Delivery Demand: {total_delivery_demand}")
        for j, customer in enumerate(route):
            print(f"  Customer {customer}:")
            print(f"    Kapasitas: {capacities[j]}")
            print(f"    Waktu Kedatangan: {arrival_times[j]}")
```

```python
# Fungsi untuk memeriksa feasibility solusi
def is_feasible(route: List[int]) -> bool:
    capacities, arrival_times = update_capacity_and_time(route)
    for i in range(len(route)):
        customer = route[i]
        # Periksa kapasitas
        if capacities[i] > max_capacity or capacities[i] < 0:
            return False
        # Periksa time window
        if arrival_times[i] > time_windows[customer][1]:
            return False
    return True
```

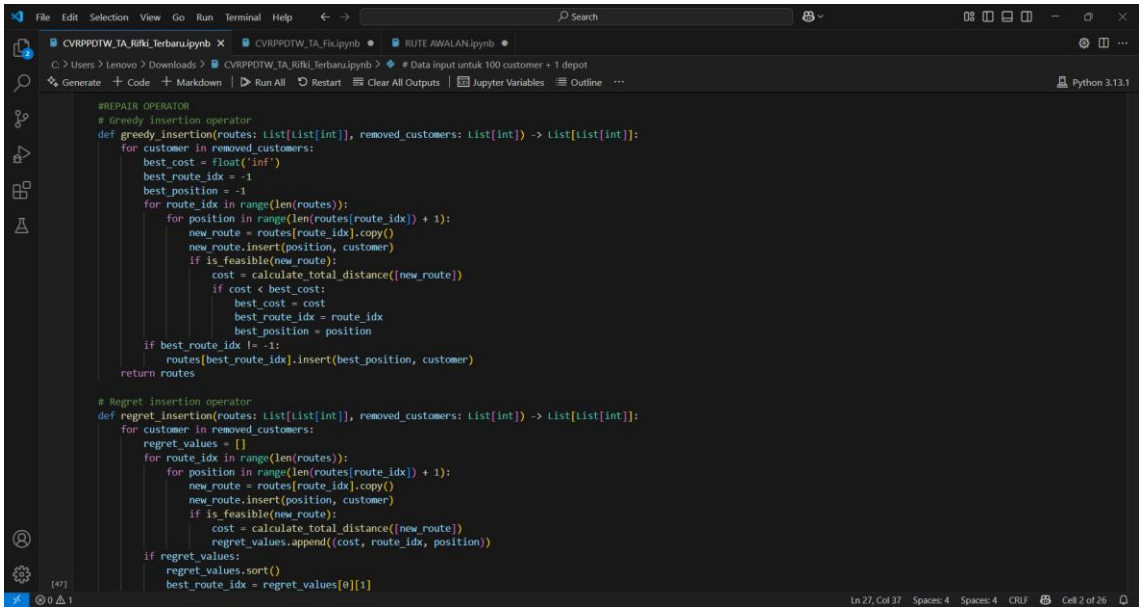
```python
#DESTROY OPERATOR
```



```python
#DESTROY OPERATOR
# Random removal operator
def random_removal(routes: List[List[int]], num_removals: int) -> Tuple[List[List[int]], List[int]]:
    removed_customers = []
    for _ in range(num_removals):
        route_idx = random.randint(0, len(routes) - 1)
        if len(routes[route_idx]) > 0:
            customer_idx = random.randint(0, len(routes[route_idx]) - 1)
            removed_customers.append(routes[route_idx].pop(customer_idx))
    return routes, removed_customers

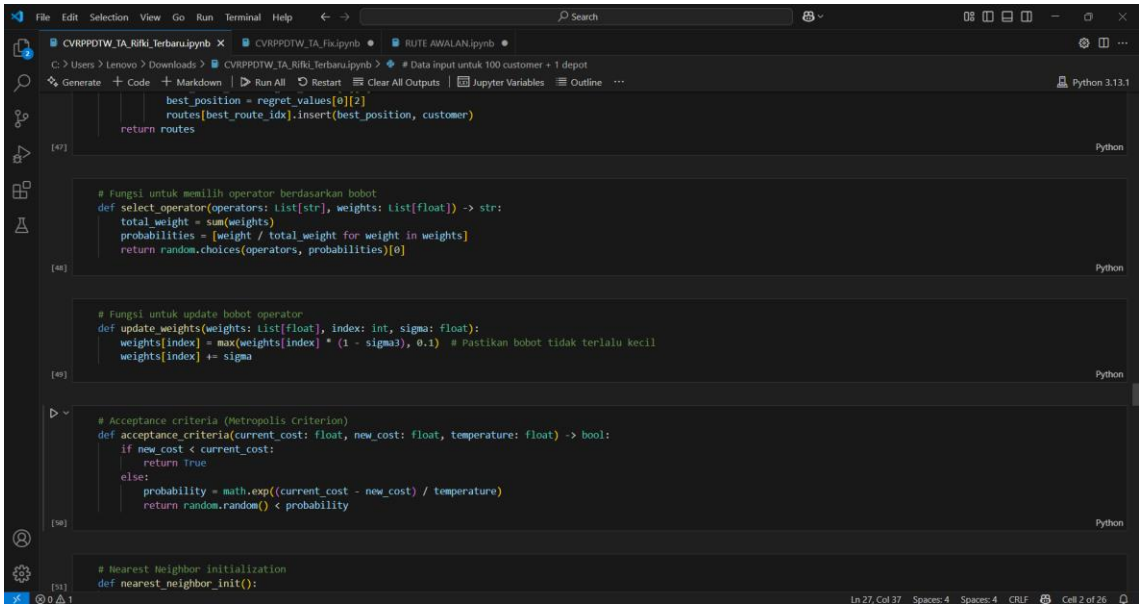
# Worst removal operator
def worst_removal(routes: List[List[int]], num_removals: int) -> Tuple[List[List[int]], List[int]]:
    removed_customers = []
    for _ in range(num_removals):
        worst_customer = None
        worst_cost = -1
        for route_idx in range(len(routes)):
            for customer_idx in range(len(routes[route_idx])):
                customer = routes[route_idx][customer_idx]
                cost = distance_matrix[depot][customer] if customer_idx == 0 else distance_matrix[routes[route_idx][customer_idx - 1]][customer]
                if cost > worst_cost:
                    worst_customer = (route_idx, customer_idx)
                    worst_cost = cost
        if worst_customer:
            route_idx, customer_idx = worst_customer
            removed_customers.append(routes[route_idx].pop(customer_idx))
    return routes, removed_customers
```

```python
#REPAIR OPERATOR
```



```python
#REPAIR OPERATOR
# Greedy insertion operator
def greedy_insertion(routes: List[List[int]], removed_customers: List[int]) -> List[List[int]]:
    for customer in removed_customers:
        best_cost = float('inf')
        best_route_idx = -1
        best_position = -1
        for route_idx in range(len(routes)):
            for position in range(len(routes[route_idx]) + 1):
                new_route = routes[route_idx].copy()
                new_route.insert(position, customer)
                if is_feasible(new_route):
                    cost = calculate_total_distance([new_route])
                    if cost < best_cost:
                        best_cost = cost
                        best_route_idx = route_idx
                        best_position = position
        if best_route_idx != -1:
            routes[best_route_idx].insert(best_position, customer)
    return routes

# Regret insertion operator
def regret_insertion(routes: List[List[int]], removed_customers: List[int]) -> List[List[int]]:
    for customer in removed_customers:
        regret_values = []
        for route_idx in range(len(routes)):
            for position in range(len(routes[route_idx]) + 1):
                new_route = routes[route_idx].copy()
                new_route.insert(position, customer)
                if is_feasible(new_route):
                    cost = calculate_total_distance([new_route])
                    regret_values.append((cost, route_idx, position))
        if regret_values:
            regret_values.sort()
            best_route_idx = regret_values[0][1]
```



```python
                best_position = regret_values[0][2]
                routes[best_route_idx].insert(best_position, customer)
    return routes
```

[47] Python

```python
# Fungsi untuk memilih operator berdasarkan bobot
def select_operator(operators: List[str], weights: List[float]) -> str:
    total_weight = sum(weights)
    probabilities = [weight / total_weight for weight in weights]
    return random.choices(operators, probabilities)[0]
```

[48] Python

```python
# Fungsi untuk update bobot operator
def update_weights(weights: List[float], index: int, sigma: float):
    weights[index] = max(weights[index] * (1 - sigma3), 0.1)  # Pastikan bobot tidak terlalu kecil
    weights[index] += sigma
```

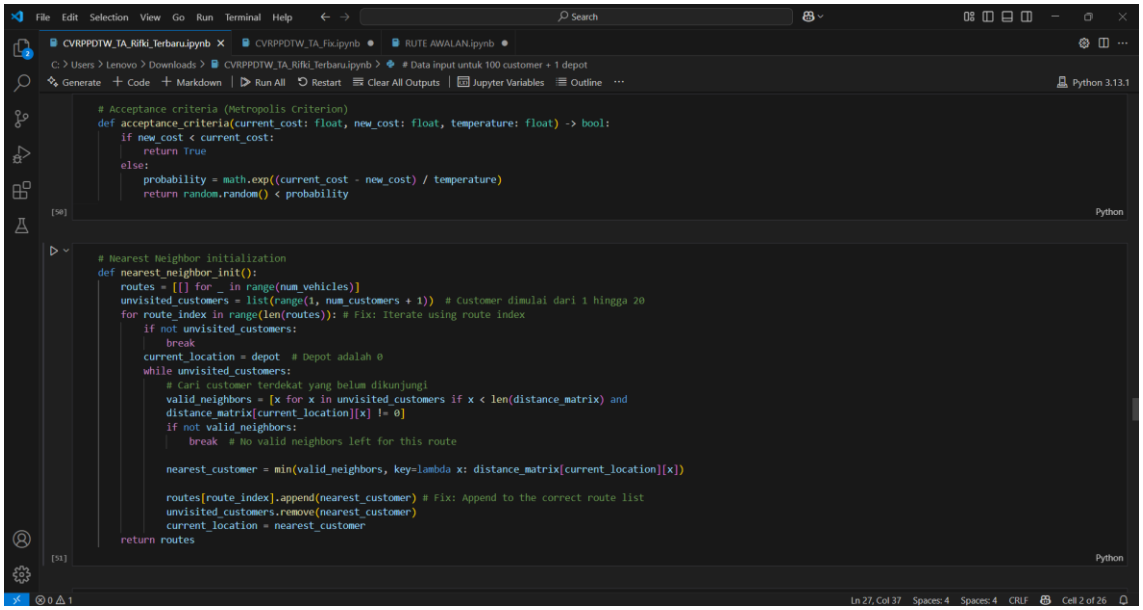
[49] Python

```python
# Acceptance criteria (Metropolis Criterion)
def acceptance_criteria(current_cost: float, new_cost: float, temperature: float) -> bool:
    if new_cost < current_cost:
        return True
    else:
        probability = math.exp((current_cost - new_cost) / temperature)
        return random.random() < probability
```

[50] Python

```python
# Nearest Neighbor initialization
def nearest_neighbor_init():
```

[51]



```
# Acceptance criteria (Metropolis Criterion)
def acceptance_criteria(current_cost: float, new_cost: float, temperature: float) -> bool:
    if new_cost < current_cost:
        return True
    else:
        probability = math.exp((current_cost - new_cost) / temperature)
        return random.random() < probability
```

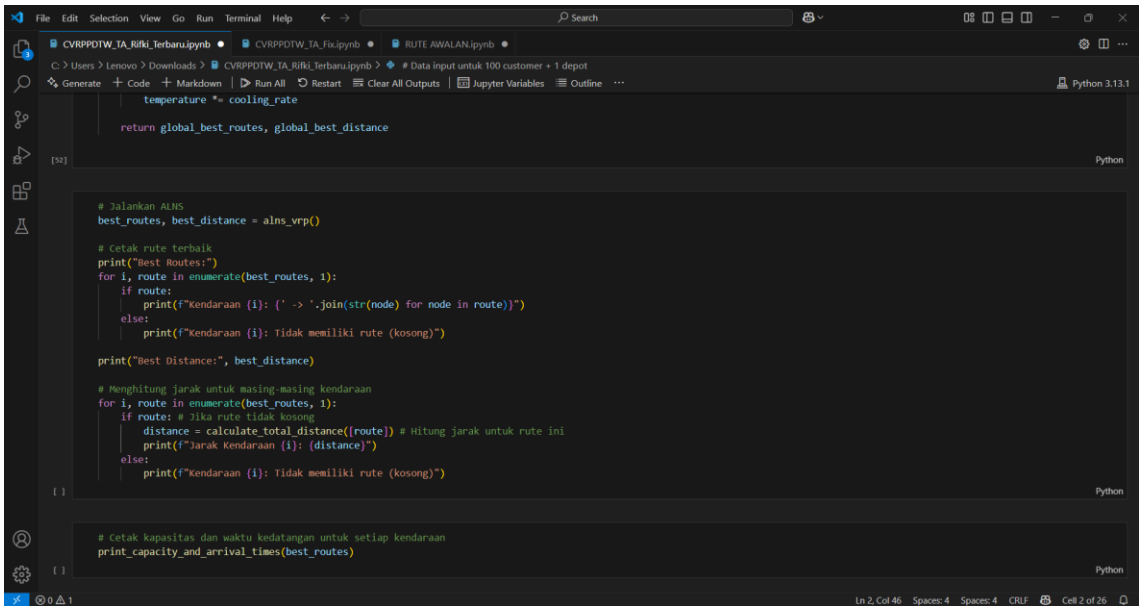
[50] Python

```
# Nearest Neighbor initialization
def nearest_neighbor_init():
    routes = [[] for _ in range(num_vehicles)]
    unvisited_customers = list(range(1, num_customers + 1))  # Customer dimulai dari 1 hingga 20
    for route_index in range(len(routes)): # Fix: Iterate using route index
        if not unvisited_customers:
            break
        current_location = depot  # Depot adalah 0
        while unvisited_customers:
            # Cari customer terdekat yang belum dikunjungi
            valid_neighbors = [x for x in unvisited_customers if x < len(distance_matrix) and
            distance_matrix[current_location][x] != 0]
            if not valid_neighbors:
                break  # No valid neighbors left for this route

            nearest_customer = min(valid_neighbors, key=lambda x: distance_matrix[current_location][x])

            routes[route_index].append(nearest_customer) # Fix: Append to the correct route list
            unvisited_customers.remove(nearest_customer)
            current_location = nearest_customer
    return routes
```

[51] Python



```
        temperature *= cooling_rate

    return global_best_routes, global_best_distance
```

[52] Python

```
# Jalankan ALNS
best_routes, best_distance = alns_vrp()

# Cetak rute terbaik
print("Best Routes:")
for i, route in enumerate(best_routes, 1):
    if route:
        print(f"Kendaraan {i}: {' -> '.join(str(node) for node in route)}")
    else:
        print(f"Kendaraan {i}: Tidak memiliki rute (kosong)")

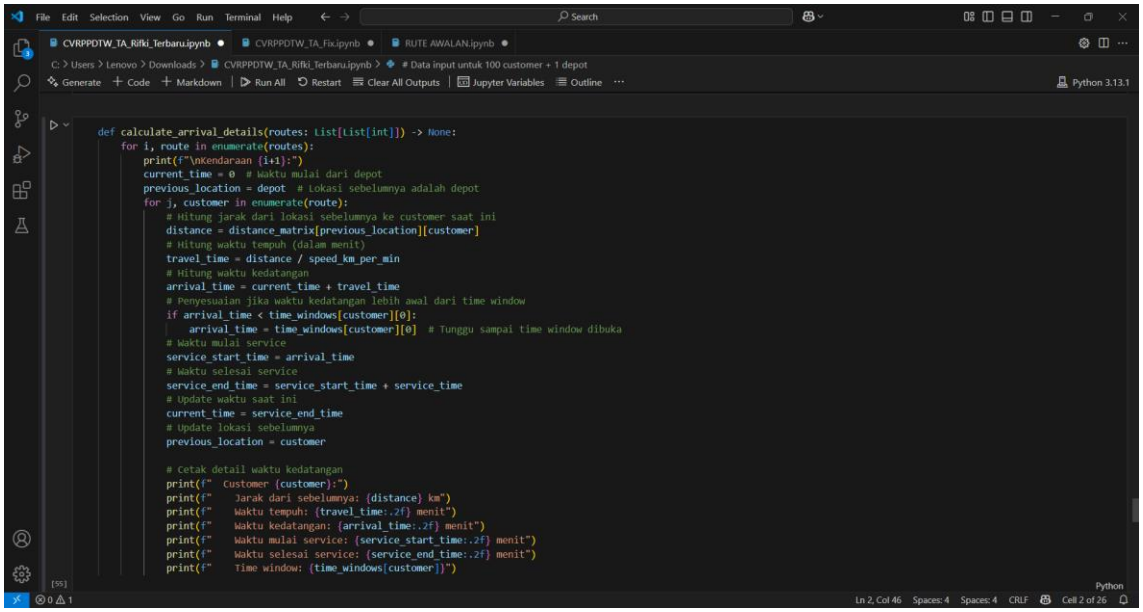
print("Best Distance:", best_distance)

# Menghitung jarak untuk masing-masing kendaraan
for i, route in enumerate(best_routes, 1):
    if route: # Jika rute tidak kosong
        distance = calculate_total_distance([route]) # Hitung jarak untuk rute ini
        print(f"Jarak Kendaraan {i}: {distance}")
    else:
        print(f"Kendaraan {i}: Tidak memiliki rute (kosong)")
```

[ ] Python

```
# Cetak kapasitas dan waktu kedatangan untuk setiap kendaraan
print_capacity_and_arrival_times(best_routes)
```

[ ] Python



```python
def calculate_arrival_details(routes: List[List[int]]) -> None:
    for i, route in enumerate(routes):
        print(f"\nKendaraan {i+1}:")
        current_time = 0  # Waktu mulai dari depot
        previous_location = depot  # Lokasi sebelumnya adalah depot
        for j, customer in enumerate(route):
            # Hitung jarak dari lokasi sebelumnya ke customer saat ini
            distance = distance_matrix[previous_location][customer]
            # Hitung waktu tempuh (dalam menit)
            travel_time = distance / speed_km_per_min
            # Hitung waktu kedatangan
            arrival_time = current_time + travel_time
            # Penyesuaian jika waktu kedatangan lebih awal dari time window
            if arrival_time < time_windows[customer][0]:
                arrival_time = time_windows[customer][0]  # Tunggu sampai time window dibuka
            # Waktu mulai service
            service_start_time = arrival_time
            # Waktu selesai service
            service_end_time = service_start_time + service_time
            # Update waktu saat ini
            current_time = service_end_time
            # Update lokasi sebelumnya
            previous_location = customer

            # Cetak detail waktu kedatangan
            print(f"  Customer {customer}:")
            print(f"    Jarak dari sebelumnya: {distance} km")
            print(f"    Waktu tempuh: {travel_time:.2f} menit")
            print(f"    Waktu kedatangan: {arrival_time:.2f} menit")
            print(f"    Waktu mulai service: {service_start_time:.2f} menit")
            print(f"    Waktu selesai service: {service_end_time:.2f} menit")
            print(f"    Time window: {time_windows[customer]}")
```