



**OPTIMASI MULTIPLE TRAVELLING SALESMAN PROBLEM  
(M-TSP) PADA PENENTUAN RUTE ANGKUTAN SEKOLAH  
MENGUNAKAN ALGORITME PARTICLE SWARM  
OPTIMIZATION (PSO)**

**SKRIPSI**

Untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer

Disusun oleh:

Muhammad Hidayat

NIM: 155150200111125



PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2020

## PENGESAHAN

**OPTIMASI MULTIPLE TRAVELLING SALESMAN PROBLEM  
(M-TSP) PADA PENENTUAN RUTE ANGKUTAN SEKOLAH MENGGUNAKAN  
ALGORITME PARTICLE SWARM OPTIMIZATION (PSO)**

**SKRIPSI**

**Diajukan untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer**

**Disusun Oleh :  
Muhammad Hidayat  
NIM: 155150200111125**

**Skripsi ini telah diuji dan dinyatakan lulus pada  
22 Juli 2020  
Telah diperiksa dan disetujui oleh:**

**Dosen Pembimbing I**

**Dosen Pembimbing 2**


  
**Imam Choissodin, S.Si., M.Kom.**  
**NIK: 201201 850719 1 001**

  
**Bayu Rahayudi, S.T.**  
**NIP: 19740712 200604 1 001**

**Mengetahui**

**Ketua Jurusan Teknik Informatika**



  
**Achmad Basuki, S.T., M.MG., Ph.D.**  
**NIP: 19741118 200312 1 002**

## PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 30 Juli 2020



Muhammad Hidayat

NIM: 155150200111125

## PRAKATA

Puji syukur kehadiran Allah SWT karena atas rahmat dan karunia yang begitu besar sehingga penelitian berjudul “Optimasi *Multiple Travelling Salesman Problem* (M-TSP) Pada Angkutan Sekolah Dengan Algoritme Genetika (Studi Kasus: Yayasan Pembina Muslim Daarussalaam Sangatta)” ini dapat diselesaikan. Shalawat serta salam tidak lupa dicurahkan kepada junjungan Nabi Muhammad SAW beserta keluarga, sahabat, dan pengikutnya hingga akhir zaman.

Dalam kesempatan ini, penulis ingin memberikan ucapan terima kasih sebesar-besarnya atas masukan, bantuan, dan motivasi dalam bentuk materiil maupun moril yang telah diberikan oleh banyak pihak antara lain kepada yang terhormat:

1. Bapak Imam Cholissodin, S.Si., M.Kom., dan Bapak Bayu Rahayudi, S.T., M.T., selaku Dosen pembimbing skripsi yang telah dengan sabar memberikan bantuan, arahan, dan keluangan waktu sehingga penulis dapat menyelesaikan penelitian ini.
2. Bapak Wayan Firdaus Mamudy, S.Si., M.T., Ph.D., selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya Malang.
3. Bapak Achmad Basuki, S.T., M.MG., Ph.D., selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Bapak Adhitya Bhawiyuga, S.Kom., M.Sc., selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang.
5. Seluruh Dosen di Fakultas Ilmu Komputer Universitas Brawijaya Malang yang telah memberikan banyak ilmu kepada penulis.
6. Seluruh keluarga besar penulis yang telah memberikan dukungan dan doa sehingga penulis terus memiliki semangat dalam menyelesaikan penelitian ini.
7. Skypats Industries yaitu Naufal Alif Aldhian, Aldous Elpizochari, Ageng Wibowo, dan Ismail Rabbanii yang telah memberikan banyak dukungan dalam bentuk apapun sehingga penulis dapat menyelesaikan penelitian.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak.

Malang, 07 Juli 2020

Penulis

[muhidayat03@gmail.com](mailto:muhidayat03@gmail.com)



## ABSTRAK

**Muhammad Hidayat, Optimasi *Multiple Travelling Salesman Problem* (M-TSP) Pada Penentuan Rute Angkutan Sekolah Menggunakan Algoritme *Particle Swarm Optimization* (PSO)**

**Pembimbing: Imam Cholissodin, S.Si., M.Kom., dan Bayu Rahayudi, S.T., M.T.**

Yayasan Pembina Muslim (YPM) Daarussalaam merupakan sebuah yayasan pendidikan yang berlokasi di Jl. Jendral Sudirman No 1, Swarga Bara, Sangatta Utara, Kabupaten Kutai Timur, Kalimantan Timur. Dalam YPM Daarussalam terdapat beberapa sekolah yaitu taman kanak-kanak islam terpadu (TKIT), sekolah dasar islam terpadu (SDIT), dan sekolah menengah pertama islam terpadu (SMPIT). Untuk membantu keberangkatan dan kepulangan siswa TK dan SD, pihak yayasan menyediakan fasilitas jasa angkutan sekolah yang dapat digunakan bagi siswa yang membutuhkan. Saat ini jumlah siswa yang menggunakan jasa angkutan sekolah adalah sebanyak 160 siswa dari total 832 siswa. Berdasarkan jumlah tersebut pihak yayasan menyediakan 8 armada dengan kapasitas 20 siswa tiap armada. Penelitian ini bertujuan untuk menentukan rute optimal pada permasalahan *Multiple Travelling Salesman Problem* (M-TSP) dengan menggunakan Algoritme *Particle Swarm Optimization* (PSO). Pada penelitian ini rute sekolah direpresentasikan sebagai partikel yang dibagi menjadi 3 segmen yaitu keberangkatan (segmen 1), kepulangan 1 untuk siswa kelas 1,2, dan TK dan (segmen 2) dan kepulangan 2 untuk siswa kelas 3-6 yang kemudian partikel tersebut akan memperbarui kecepatan setiap iterasi untuk menemukan rute dengan nilai fitness terbaik. Parameter yang digunakan adalah jumlah iterasi = 538, ukuran partikel = 80,  $c_1 = 1$ , dan  $c_2 = 1,5$ . *Fitness* rata-rata yang didapatkan adalah 2,297 dengan nilai *fitness* terbaik 2.393, dan dengan hasil tersebut pihak yayasan dapat memangkas jarak sebanyak 210,884 Km.

**Kata kunci:** optimasi, rute, *multiple travelling salesman problem*, Algoritme *Particle Swarm Optimization* (PSO).

## ABSTRACT

### **Muhammad Hidayat, Optimization of Multiple Traveling Salesman Problems (M-TSP) in Determining School Transport Routes Using Particle Swarm Optimization Algorithm (PSO)**

**Supervisors: Imam Cholissodin, S.Si., M.Kom., and Bayu Rahayudi, S.T., M.T.**

Daarussalaam Muslim Development Foundation (YPM) is an educational foundation located on Jl. Jendral Sudirman No 1, Swarga Bara, Sangatta Utara, East Kutai Regency, East Kalimantan. In YPM Daarussalam there are several schools, namely integrated Islamic kindergartens (TKIT), integrated Islamic elementary schools (SDIT), and integrated Islamic junior high schools (SMPIT). To help the departure and return of kindergarten and elementary school students, the foundation provides school transportation services that can be used for students in need. At present the number of students using school transportation services is 160 students out of a total of 832 students. Based on this number the foundation provides 8 cars with a capacity of 20 students per cars. This study aims to determine the optimal route for the Multiple Traveling Salesman Problem (M-TSP) problem using Particle Swarm Optimization (PSO) algorithm. In this study the school route is represented as particles which are divided into 3 segments, namely departure (segment 1), return 1 for grade 1,2 students, and kindergarten and (segment 2) and return 2 for grade 3-6 students which then the particles will be update the speed of each iteration to find the route with the best fitness value. The parameters used are the number of iterations = 538, particle size = 80,  $c1 = 1$ , and  $c2 = 1.5$ . The average fitness obtained is 2,297 with the best fitness value of 2,393, and with these results the foundation can cut the distance by 210,884 Km.

**Keywords:** optimization, route, multiple travelling salesman problem, Particle Swarm Optimization Algorithm.



## DAFTAR ISI

PENGESAHAN .....	ii
PERNYATAAN ORISINALITAS .....	iii
PRAKATA .....	iv
ABSTRAK .....	v
ABSTRACT .....	vi
DAFTAR ISI .....	vii
DAFTAR TABEL .....	x
DAFTAR GAMBAR .....	xi
DAFTAR LAMPIRAN .....	xii
BAB 1 PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	3
1.3 Tujuan .....	3
1.4 Manfaat .....	3
1.5 Batasan Masalah .....	3
1.6 Sistematika Pembahasan .....	3
BAB 2 LANDASAN KEPUSTAKAAN .....	5
2.1 Kajian Pustaka .....	5
2.2 Yayasan Pembina Muslim (YPM) Daarussalaam .....	6
2.3 Angkutan Sekolah .....	7
2.4 <i>Multiple Travelling Salesman Problem (M-TSP)</i> .....	8
2.5 Algoritme <i>Particle Swarm Optimization (PSO)</i> .....	8
BAB 3 METODOLOGI PENELITIAN .....	10
3.1 Studi Pustaka .....	11
3.2 Pengumpulan Data .....	11
3.3 Perancangan Sistem .....	11
3.4 Implementasi .....	11
3.5 Hasil dan Pembahasan .....	11
3.6 Penarikan Kesimpulan .....	11
BAB 4 PERANCANGAN .....	13



4.1.	Perancangan Algoritme.....	13
4.1.1.	Nilai Parameter.....	14
4.1.2.	Inisialisasi PSO.....	14
4.1.3.	Memperbarui Kecepatan.....	19
4.1.4.	Memperbarui Posisi.....	20
4.1.5.	Menghitung Jarak dan Fitness.....	21
	Memperbarui <i>Pbest</i> dan <i>Gbest</i> .....	22
4.2.	Perhitungan Manual.....	23
4.2.1.	Nilai Parameter.....	23
4.2.2.	Inisialisasi.....	24
4.2.3.	Memperbarui Kecepatan.....	26
4.2.4.	Memperbarui Posisi.....	28
4.2.5.	Memperbarui <i>Pbest</i> .....	29
4.2.6.	Memperbarui <i>Gbest</i> .....	30
4.3.	Perancangan Pengujian.....	31
4.3.1.	Pengujian Konvergensi.....	31
4.3.2.	Pengujian Ukuran Partikel.....	32
4.3.3.	Pengujian Perbandingan Hasil Aktual dengan Hasil Penelitian.....	32
4.4.	Perancangan Antarmuka.....	33
4.4.1.	Halaman Data Alamat.....	33
4.4.2.	Halaman Data Sopir.....	33
4.4.3.	Halaman Data Jarak.....	34
4.4.4.	Halaman Hitung MTSP.....	35
BAB 5	Implementasi.....	37
5.1	Implementasi Proses Inisialisasi.....	37
5.1.1	Implementasi Inisialisasi Posisi Awal.....	37
5.1.2	Implementasi Inisialisasi Kecepatan Awal.....	38
5.1.3	Implementasi Inisialisasi <i>Pbest</i> .....	38
5.1.4	Implementasi Inisialisasi <i>Gbest</i> .....	39
5.2	Implementasi Proses Memperbarui Kecepatan.....	39
5.3	Implementasi Proses Memperbarui Posisi.....	43





5.4 Implementasi Proses Hitung Fitness.....	44
5.5 Implementasi Proses Memperbarui $P_{best}$ .....	45
5.6 Implementasi Proses Memperbarui $G_{best}$ .....	46
<b>BAB 6 Pengujian</b> .....	<b>48</b>
6.1 Pengujian Konvergensi.....	48
6.2 Pengujian Ukuran Partikel.....	49
6.3 Pengujian Perbandingan Hasil Aktual Dan Hasil Penelitian.....	50
<b>BAB 7 PENUTUP</b> .....	<b>52</b>
7.1 Kesimpulan.....	52
7.2 Saran.....	52
<b>DAFTAR REFERENSI</b> .....	<b>53</b>
<b>LAMPIRAN A DATA YANG DIGUNAKAN</b> .....	<b>54</b>
<b>LAMPIRAN B HASIL PENELITIAN</b> .....	<b>58</b>



## DAFTAR TABEL

Tabel 4.1 Dataset Perhitungan Manual.....	23
Tabel 4.2 Keterangan Dataset.....	23
Tabel 4.3 Inisialisasi Kecepatan.....	24
Tabel 4.4 Data Segmen.....	24
Tabel 4.5 Inisialisasi Posisi Awal Partikel.....	24
Tabel 4.6 Tabel Perhitungan jarak Setiap Segmen pada Partikel.....	25
Tabel 4.7 tabel total jarak partikel.....	25
Tabel 4.8 Tabel Perhitungan Fitness.....	25
Tabel 4.9 Tabel Inisialisasi $P_{best}$ .....	25
Tabel 4.10 Tabel Inisialisasi $G_{best}$ .....	26
Tabel 4.11 Hasil Tahap Memperbarui Kecepatan.....	28
Tabel 4.12 Memperbarui Posisi berdasarkan Kecepatan.....	28
Tabel 4.13 Hasil Tahap Memperbarui Posisi.....	29
Tabel 4.14 Memperbarui $P_{best}$ .....	30
Tabel 4.15 Hasil Tahap Memperbarui $P_{best}$ .....	30
Tabel 4.16 Hasil Tahap Memperbarui $G_{best}$ .....	30
Tabel 4.17 Rute Keberangkatan.....	30
Tabel 4.18 Rute Keberangkatan Siswa Kelas 1 dan 2.....	30
Tabel 4.19 Rute Kepulangan Siswa Kelas 3-6.....	31
Tabel 4.20 Hasil Pengujian Konvergensi.....	31
Tabel 4.21 Perancangan Pengujian Ukuran Partikel.....	32
Tabel 6.1 Hasil Pengujian Ukuran Partikel.....	49



## DAFTAR GAMBAR

Gambar 2.2 YPM Daarussalaam.....	7
Gambar 2.3 Angkutan Sekolah pada SDIT Daarussalaam.....	8
Gambar 3.1 Metodologi Penelitian.....	10
Gambar 4.1 Diagram Alir <i>Algoritme Particle Swarm Optimization</i> (PSO).....	14
Gambar 4.2 Diagram Alir Inisialisasi PSO.....	14
Gambar 4.3 Diagram Alir Inisialisasi Kecepatan Awal.....	15
Gambar 4.4 Diagram Alir Inisialisasi Posisi Awal.....	16
Gambar 4.5 Diagram Alir Inisialisasi <i>Pbest</i> .....	17
Gambar 4.6 Diagram Alir Inisialisasi <i>Gbest</i> .....	18
Gambar 4.7 Diagram Alir Memperbarui Kecepatan.....	19
Gambar 4.8 Diagram Alir Memperbarui Posisi Partikel.....	20
Gambar 4.9 Diagram Alir Perhitungan Fitness.....	21
Gambar 4.10 Diagram Alir Perhitungan Total Jarak.....	21
Gambar 4.11 Diagram Alir Memperbarui <i>Pbest</i> .....	22
Gambar 4.12 Graf Rute Kepulangan Siswa.....	31
Gambar 4.13 Graf Rute Keberangkatan Siswa.....	31
Gambar 4.14 Halaman Data Alamat.....	33
Gambar 4.15 Halaman Data Sopir.....	34
Gambar 4.16 Halaman Data Jarak.....	35
Gambar 4.17 Halaman Hitung MTSP.....	35
Gambar 6.1 Grafik Hasil Pengujian Konvergensi.....	48
Gambar 6.2 Grafik Hasil Pengujian Ukuran Partikel.....	50
Gambar 6.3 Hasil Pengujian Ukuran Partikel Terhadap Waktu Komputasi.....	50



## DAFTAR LAMPIRAN

LAMPIRAN A DATA YANG DIGUNAKAN .....	54
A.1 Data Siswa .....	54
LAMPIRAN B HASIL PENELITIAN .....	58
B.1 Hasil Pengujian Ukuran Populasi .....	58
B.2 Hasil Rute pada Segmen1 .....	58
B.3 Hasil Rute pada Segmen2 .....	59
B.4 Hasil Rute pada Segmen2 .....	60



## BAB 1 PENDAHULUAN

### 1.1.1 Latar Belakang

Transportasi saat ini telah menjadi kebutuhan dasar bagi masyarakat dalam melakukan kegiatan sehari-hari seperti aktifitas ekonomi, bekerja, sekolah, maupun sekedar bertamasya. Menurut Menteri Perhubungan Budi Karya Sumadi (2017) menyatakan bahwa transportasi telah menjadi kebutuhan yang mendasar bagi masyarakat sehingga ketersediaan jasa transportasi dalam melakukan produksi, distribusi dan konsumsi memerlukan perhatian yang berkelanjutan.

Menurut nurhanisah (2017) transportasi sangat berperan dalam menunjang pertumbuhan berbagai aspek, beberapa diantaranya adalah ekonomi, sosial dan politik. Dari segi ekonomi, penyediaan transportasi dapat memperluas jaringan distribusi, memperlancar produksi dan memudahkan dalam perluasan cakupan penyediaan barang dan jasa yang dapat dikonsumsi dalam suatu daerah. Dalam segi sosial transportasi memungkinkan manusia untuk memberikan pilihan dalam bermukim dan melakukan berbagai aktifitas pada berbagai tempat. Dari segi politik adanya transportasi memungkinkan bagi pemerintah untuk melaksanakan pemerintahan pada wilayah yang luas.

Pentingnya transportasi membuat beberapa sekolah menyediakan sarana transportasi berupa angkutan sekolah. Angkutan sekolah merupakan layanan jasa transportasi antar-jemput bagi siswa-siswi untuk memperlancar poses belajar mengajar (kariono, 2011). Dengan adanya angkutan sekolah akan mengurangi risiko kecelakaan pada anak sehingga memberikan rasa aman kepada orang tua dibandingkan dengan menggunakan angkutan umum atau kendaraan pribadi.

Yayasan Pembina Muslim (YPM) Daarussalaam merupakan sebuah yayasan pendidikan yang berlokasi di Jl. Jendral Sudirman No 1, Swarga Bara, Sangatta Utara, Kabupaten Kutai Timur, Kalimantan Timur. Dalam YPM Daarussalam terdapat beberapa sekolah yaitu taman kanak-kanak islam terpadu (TKIT), sekolah dasar islam terpadu (SDIT), dan sekolah menengah pertama islam terpadu (SMPIT). Untuk membantu keberangkatan dan kepulangan siswa TK dan SD, pihak yayasan menyediakan fasilitas jasa angkutan sekolah yang dapat digunakan bagi siswa yang membutuhkan. Menurut staf manajemen yayasan, Mansur (2018) adanya angkutan sekolah dikarenakan banyaknya permintaan wali murid dan untuk memudahkan proses belajar mengajar. Saat ini pihak yayasan menyediakan 8 armada angkutan dengan kuota 20 siswa tiap armadanya. Angka ini akan berubah tiap semesternya dikarenakan setiap semester diadakan pendataan ulang bagi siswa yang ingin menggunakan jasa angkutan sekolah maupun siswa yang ingin berhenti menggunakan jasa tersebut serta penambahan murid baru setiap tahun.

Dalam penentuan siswa tiap armada, pihak yayasan akan menentukan secara manual berdasarkan wilayah dan membagikannya kepada para sopir, selanjutnya apabila terdapat siswa dengan alamat penjemputan yang terlalu jauh ataupun tidak sesuai dengan wilayah penjemputan, maka sopir dapat saling menukarkan



siswa tersebut dengan sopir lainnya. Setelah itu setiap harinya para sopir akan melakukan penjemputan dari satu tempat ke tempat lainnya berdasarkan perkiraan jarak terdekat.

Pembagian siswa kepada tiap armada secara manual dan penentuan rute dengan berdasarkan perkiraan jarak terdekat membuat proses penjemputan tidak efisien serta menyulitkan sopir karena harus saling tukar siswa kepada sopir lainnya. Berdasarkan permasalahan tersebut maka dibutuhkan sebuah sistem yang dapat melakukan optimasi terhadap rute yang akan dilalui dalam proses antar-jemput siswa sehingga dapat dihasilkan rute yang efisien dan maksimal. Travelling Salesman Problem merupakan permasalahan dimana seseorang harus mengunjungi seluruh kota dalam rute tepat satu kali dan kemudian kembali ke kota asal, permasalahan yang ingin diselesaikan adalah bagaimana menemukan rute terpendek dengan biaya yang minimum, pada penelitian ini optimasi dilakukan untuk menentukan rute yang akan dilalui angkutan sekolah.

Penelitian dengan permasalahan yang sama sebelumnya pernah dilakukan oleh Azhari, Bachtiar, & Cholissodin (2018) yaitu optimasi TSP pada angkutan sekolah, pada penelitian ini mencoba untuk mengembangkannya menjadi Multiple Travelling Salesman Problem (M-TSP), yaitu dengan menggunakan lebih dari satu armada untuk melakukan penjemputan siswa. Banyak penelitian yang dilakukan untuk menyelesaikan permasalahan pencarian jalur optimal, salah satunya adalah seperti penelitian yang pernah dilakukan oleh Puteri, Widodo, & Cholissodin (2017) tentang pemilihan jalur optimal dalam pendistribusian air minum dengan meneliti studi kasus pada UD. Tosa Malang. Pada penelitian tersebut menggunakan algoritme Algoritme Particle Swarm Optimization (PSO) dan memberikan hasil optimasi yang lebih baik dari rute aktual dengan selisih 45 km.

Pada penelitian ini penulis menggunakan Algoritme PSO untuk melakukan optimasi pada jarak pada rute antar jemput siswa. Algoritme pso merupakan salah satu teknik optimasi yang mengadaptasi perilaku sekelompok hewan dalam mencari makan. Dalam PSO terdapat sekumpulan partikel (individu) yang disebut swarm. Setiap partikel berpindah dengan kecepatan yang terbentuk berdasarkan daerah pencarian dan menyimpan hasil terbaiknya pada penyimpanan global. Algoritme tersebut ditemukan oleh Kennedy dan Eberhart pada tahun 1995. Dalam berbagai permasalahan Algoritme PSO mampu memberikan hasil yang optimal dengan kualitas yang lebih baik serta waktu komputasi yang lebih cepat apabila dibandingkan dengan Algoritme Genetika, selain itu dibandingkan dengan Algoritme Genetika, PSO dinilai lebih baik dalam menangani masalah konvergensi pada optimasi (mohamed b Abdelhalim, 2009).

Berdasarkan uraian diatas penulis mengangkat judul Optimasi Multiple Travelling Salesman Problem (M-TSP) Pada Penentuan Rute Angkutan Sekolah Menggunakan Algoritme *Particle Swarm Optimization* (PSO).



## 1.2 Rumusan Masalah

Berdasarkan pemaparan latar belakang sebelumnya, maka didapatkan rumusan masalah sebagai berikut:

1. Bagaimana penerapan Algoritme PSO dalam optimasi permasalahan Multiple Travelling Salesman Problem (M-TSP) pada penentuan rute angkutan sekolah?
2. Bagaimana kualitas dari solusi yang dihasilkan oleh Algoritme PSO dalam optimasi permasalahan Multiple Travelling Salesman Problem (M-TSP) pada penentuan rute angkutan sekolah?

## 1.3 Tujuan

Berdasarkan penjabaran rumusan masalah diatas, maka tujuan dari penelitian ini adalah sebagai berikut:

1. Dapat menerapkan Algoritme PSO dalam optimasi Multiple Travelling Salesman Problem (M-TSP) pada penentuan rute angkutan sekolah.
2. Mengetahui kualitas dari solusi yang dihasilkan oleh PSO dalam optimasi Multiple Travelling Salesman Problem (M-TSP) pada penentuan rute angkutan sekolah.

## 1.4 Manfaat

Manfaat penelitian dapat diuraikan sebagai dampak atau konsekuensi positif.

Adapun manfaat dari penelitian ini adalah sebagai berikut:

1. Dapat memahami penerapan dan pengaplikasian Algoritme PSO.
2. Dapat menjadi pertimbangan untuk penentuan strategi penjemputan pada pihak Yayasan Pembina Muslim Daarussalaam
3. Dapat mempermudah sopir dalam penentuan rute sehingga lebih efisien

## 1.5 Batasan Masalah

Dalam penelitian ini terdapat beberapa batasan masalah sebagai berikut:

1. Dataset pada penelitian ini merupakan data angkutan dan alamat siswa SDIT dan TKIT Daarussalaam Sangatta.
2. Jarak antar alamat siswa didapat dengan menggunakan google maps.
3. Angkutan sekolah yang dimaksud pada penelitian ini adalah layanan yang disediakan oleh Yayasan Pembina Muslim Daarussalam dengan tipe Daihatsu Luxio dengan kapasitas 20 siswa.

## 1.6 Sistematika Pembahasan

Sistematika pembahasan merupakan metode atau tata cara dalam menyusun sebuah penelitian. Adapun sistematika pembahasan pada penelitian ini adalah sebagai berikut:



## **BAB I Pendahuluan**

Bab ini berisi tentang latar belakang, rumusan masalah, tujuan, manfaat, dan batasan penelitian dari penerapan Algoritme PSO dalam permasalahan optimasi *Multiple Travelling Salesman Problem* (M-TSP) pada penentuan rute angkutan sekolah.

## **BAB II Landasan Kepustakaan**

Bab ini berisi tentang hasil dari penelitian sebelumnya yang berkaitan dengan penerapan Algoritme PSO dalam permasalahan optimasi *Multiple Travelling Salesman Problem* (M-TSP) pada penentuan rute angkutan sekolah, serta penjelasan dasar terkait algoritma tersebut.

## **BAB III Metodologi Penelitian**

Bab ini berisi tentang langkah- langkah dalam melakukan penelitian ini, meliputi studi literatur, pengumpulan data, proses dan analisis data, perancangan, implementasi, pengujian sistem yang telah dibuat.

## **BAB IV Analisis dan Perancangan**

Bab ini berisi tentang analisa dalam perancangan sistem yang akan digunakan dalam melakukan penerapan Algoritme PSO pada optimasi *Multiple Travelling Salesman Problem* (M-TSP) dalam penentuan rute angkutan sekolah.

## **BAB V Implemenasi**

Bab ini berisi tentang penerapan sistem berdasarkan hasil dari analisa dan perancangan dari penerapan Algoritme PSO pada optimasi *Multiple Travelling Salesman Problem* (M-TSP) dalam penentuan rute angkutan sekolah.

## **BAB VI Pengujian dan Analisis**

Bab ini berisi tentang pengujian dari sistem yang telah diimplementasikan, serta menjabarkan hasil akurasi dari penerapan Algoritme PSO pada optimasi *Multiple Travelling Salesman Problem* (M-TSP) dalam penentuan rute angkutan sekolah.

## **BAB VII Penutup**

Bab ini berisi tentang kesimpulan dan saran dari hasil penelitian.





## BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepastakaan berisi uraian dan pembahasan tentang teori, konsep, model, metode, atau sistem dari pustaka ilmiah, yang berkaitan dengan tema, masalah, atau pertanyaan penelitian. Dalam landasan kepastakaan terdapat landasan teori dari berbagai sumber pustaka yang terkait dengan teori dan metode yang digunakan dalam penelitian. Jika dibutuhkan sesuai dengan karakteristik penelitiannya dan syarat kecukupan khusus keminatan tertentu, bisa juga terdapat kajian pustaka yang menjelaskan secara umum penelitian-penelitian terdahulu yang berhubungan dengan topik skripsi dan menunjukkan persamaan dan perbedaan skripsi tersebut terhadap penelitian terdahulu yang dituliskan.

### 2.1 Kajian Pustaka

Azhari, Bachtiar, & Cholissodin (2018) melakukan penelitian mengenai optimasi *Traveling Salesman Problem* (TSP) pada angkutan sekolah MI Salafiyah Kasim Blitar. Penelitian ini menggunakan algoritma genetika untuk melakukan optimasi jarak dan waktu pada proses antar jemput siswa, dengan melakukan analisis pada data angkutan dan alamat siswa dari sekolah tersebut. Dari penelitian yang dilakukan, didapatkan hasil bahwa sistem dapat memberikan rute yang optimal, namun hasil dari sistem tidak selalu lebih baik dari data aktual.

Penelitian lainnya dilakukan oleh Karimah, Widodo, & Cholissodin (2017) untuk melakukan optimasi permasalahan MTSP pada distribusi air minum UD. Tosa Malang. Pada penelitian ini menggunakan Algoritme Genetika dan memberikan hasil yang lebih baik dari rute sebenarnya. Selisih yang antara sistem dan jarak aktual adalah 83.9 km. Dengan objek penelitian yang sama Puteri, Widodo, & Cholissodin (2017) melakukan penelitian dengan Algoritme yang berbeda yaitu Algoritme PSO. Berdasarkan penelitian yang dilakukan, didapatkan bahwa Algoritme PSO dapat memberikan optimasi.

Akhand, Akter, Rahman, & Rahman (2012) melakukan penelitian untuk menguji hasil dari Algoritme *Particle Swarm Optimization* dengan *Partial Search* yang dilakukan menggunakan *dataset* burma14, eil51, berlin52, eil76, kroA100. Penelitian ini memberikan hasil bahwa PSO dengan Algoritme *Partial Search* (PSOPS) terbukti menghasilkan solusi yang optimal dalam waktu minimal dari standar PSO dan Algoritme genetik dalam menyelesaikan *benchmark* TSP.

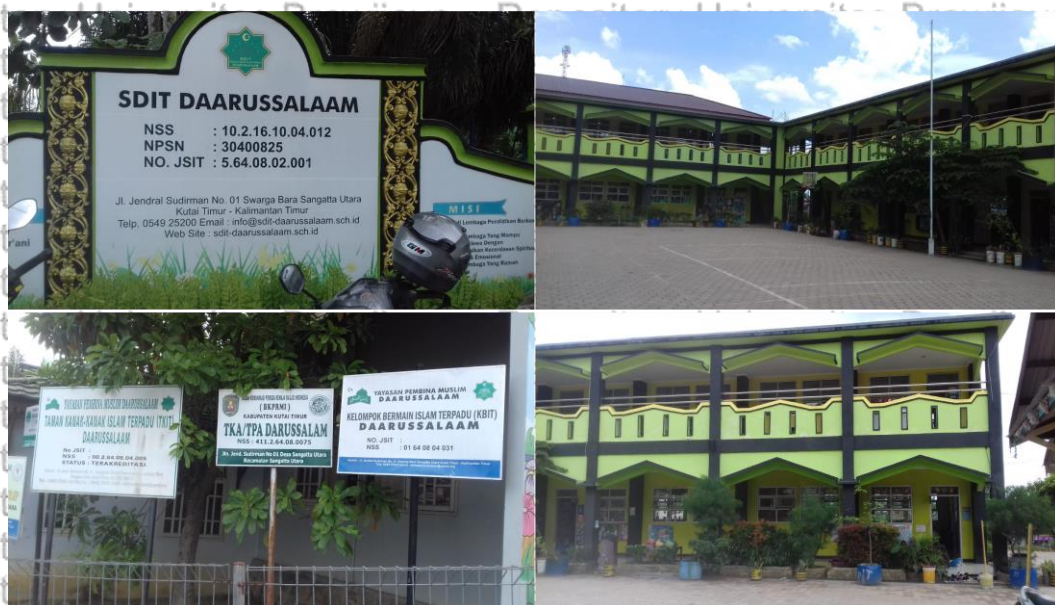
Syafiq, Cholissodin, & Aryadita (2017) melakukan penelitian untuk melakukan optimasi pada penyusunan jadwal kuliah pada PTIIK Universitas Brawijaya menggunakan Algoritme PSO. Pada penelitian ini sistem yang dibangun memberikan hasil optimasi dengan beberapa *constraint*. Perincian kajian pustaka yang ditunjukkan pada Tabel 2.1.

Tabel 2.1 Kajian Pustaka

No.	Penelitian	Objek	Metode	Hasil
1.	(Azhari, Bachtiar, & Cholissodin, 2018)	Angkutan Sekolah	<i>Particle Swarm Optimization</i> (PSO)	Sistem mampu menghasilkan solusi yang baik, namun tidak selalu optimal
2.	(Karimah, Widodo, & Cholissodin, 2017)	Distribusi Air Minum	Algoritme Genetika	Penerapan algoritme Genetika memberikan solusi rute yang telah dioptimasi lebih baik dari data aktual dengan selisih 89.3 Km
3.	(Puteri, Widodo, & Cholissodin, 2017)	Distribusi Air Minum	<i>Particle Swarm Optimization</i> (PSO)	Penerapan algoritme <i>Particle Swarm Optimization</i> (PSO) memberikan solusi rute yang telah dioptimasi lebih baik dari data aktual dengan selisih 45 km
4	( Akhand , Akter , Rahman, & Rahman, 2012)	Dataset burma14, eil51, berlin52, eil76, kroA100 dan kroA200	Particle Swarm Optimization dengan Partial Search	PSO yang diusulkan dengan Algoritma Partial Search (PSOPS) terbukti menghasilkan solusi yang optimal dalam waktu minimal dari standar PSO dan Genetic Algoritma dalam menyelesaikan benchmark TSP.
5	(Syafiq, Cholissodin, & Aryadita, 2017)	Jadwal Perkuliahan PTIIK Universitas Brawijaya	Hybrid Discrete Particle Swarm Optimization	Dapat memberikan solusi dengan beberapa constraint.

## 2.2 Yayasan Pembina Muslim (YPM) Daarussalaam

Yayasan Pembina Muslim (YPM) Daarussalaam merupakan yayasan pendidikan yang berlokasi di Jl. Jendral Sudirman No 1, Swarga Bara, Sangatta Utara, Kabupaten Kutai Timur, Kalimantan Timur. YPM Daarussalaam didirikan pada tahun 2001. Pada YPM Daarussalaam terdapat TKIT, SDIT dan SMPIT. Pada SDIT Daarussalaam terdapat 28 kelas dengan total 832 siswa dan 45 guru sedangkan pada TKIT Daarussalaam terdapat 22 guru dan 228 siswa. (Data Pokok Pendidikan Dasar dan Menengah Direktorat Jenderal Pendidikan Dasar dan Menengah Kementerian Pendidikan dan Kebudayaan, 2019).



Gambar 2.1 YPM Daarussalaam

### 2.3 Angkutan Sekolah

Angkutan sekolah merupakan sarana antar jemput yang disediakan oleh pihak sekolah. Saat ini dikota-kota besar penyediaan layanan angkutan sekolah sudah banyak diterapkan, baik dari pemerintah maupun langsung dari pihak sekolah. Adanya layanan angkutan sekolah mempermudah dalam pengawasan siswa, dan menjadikan program sekolah lebih efektif dan efisien (bafadal, 2017).

Layanan angkutan sekolah pada YPM Daarussalam disediakan oleh pihak yayasan, sehingga untuk dapat menggunakan layanan tersebut maka harus melalui proses pendaftaran terlebih dahulu. Angkutan sekolah disediakan untuk keberangkatan dan kepulangan. Untuk keberangkatan dilakukan pada pukul 06.30 dan untuk kepulangan dibagi menjadi 2 gelombang, gelombang pertama adalah pada pukul 13.00 untuk kepulangan siswa TK dan SD kelas 1 dan 2, dan kepulangan gelombang 2 adalah pada pukul 15.00 untuk siswa SD kelas 3-6. Saat ini jumlah siswa yang menggunakan jasa angkutan sekolah adalah sebanyak 160 siswa dari total 832 siswa. Berdasarkan jumlah tersebut pihak yayasan menyediakan 8 armada dengan kapasitas 20 siswa tiap armada. Angka tersebut dapat berubah tiap semesternya dikarenakan pengadaan pendataan ulang bagi yang ingin mendaftar untuk menggunakan layanan angkutan maupun bagi siswa yang ingin berhenti, serta dapat berubah tiap tahun dikarenakan adanya siswa baru. Jenis mobil yang digunakan sebagai mobil angkutan adalah Daihatsu Luxio.



Gambar 2.2 Angkutan Sekolah pada SDIT Daarussalaam

## 2.4 Multiple Travelling Salesman Problem (M-TSP)

M-TSP merupakan pengembangan dari permasalahan TSP yang dapat digunakan untuk mencari rute dengan jarak yang minimum yang akan dilewati oleh beberapa *sales* untuk mengunjungi beberapa kota dan kembali pada kota awal. Perbedaan dari M-TSP dan TSP adalah pada jumlah *sales*-nya. Pada TSP jumlah *sales* adalah 1 sedangkan pada MTSP jumlah *sales* dapat lebih dari 1. M-TSP dapat diselesaikan dengan metode heuristik, seperti Algoritme Genetika maupun PSO. Kebanyakan metode heuristik yang digunakan menerima solusi baru yang lebih baik, namun pada metode algoritme genetika solusi baru yang lebih buruk kadang dapat diterima, sehingga solusi dapat terhindar dari nilai maksimum lokal (Sari & Mahmudy, 2015).

## 2.5 Algoritme Particle Swarm Optimization (PSO)

Algoritme PSO dikategorikan ke dalam *algoritme swarm intelligence* dan teknik optimasi ini berdasarkan populasi yang awalnya dikembangkan oleh Kennedy dan Eberhart pada tahun 1995. Hal ini didorong oleh kelakuan sosial sekelompok burung atau sekawanan ikan yang berenang bersama-sama. *Particle Swarm Optimization* termasuk *algoritme metaheuristic* dan dapat diterapkan secara luas dalam memecahkan banyak masalah seperti permasalahan kombinatorial. Berikut merupakan langkah-langkah dalam penerapan *Algoritme Particle Swarm Optimization* (Nedjah, 2007):

1. Menginisialisasi populasi partikel dengan posisi dan kecepatan yang dibangkitkan secara acak.



2. Menghitung nilai fitness dari setiap partikel yang ditelaah diinisialisasi. Nilai fitness tersebut menyatakan kesesuaian solusi dari permasalahan yang ingin dipecahkan.
3. Membandingkan nilai fitness dari partikel dengan nilai nilai fitness terbaik yang pernah dicapai oleh partikel tersebut. Apabila nilai fitness yang baru lebih baik, maka nilai tersebut akan menggantikan nilai fitness terbaik dan disimpan sebagai  $P_{best}$ .
4. Mengidentifikasi partikel dengan nilai fitness terbaik dari semua populasi dan menetapkannya sebagai global  $g$  yang terbaik atau  $G_{best}$ .
5. Melakukan perhitungan untuk memperbaiki nilai kecepatan yang didasari oleh nilai  $P_{best}$  dan  $G_{best}$ .

Ulangi kembali ke langkah ke dua, kondisi berhenti ketika jika spesifikasi yang dibutuhkan telah dicapai, atau sampai pada jumlah maksimum iterasi yang dilakukan.



## BAB 3 METODOLOGI PENELITIAN

Dalam penelitian ini dilakukan dengan tahapan sebagai berikut:

1. Mengumpulkan bahan-bahan dan literatur yang berkaitan dengan penelitian, kemudian mempelajarinya.
2. Mengumpulkan data-data yang akan digunakan dan diterapkan dengan menggunakan algoritma Algoritme PSO
3. Melakukan analisa kebutuhan dan perancangan terhadap sistem yang akan dibuat
4. Mengimplementasikan hasil rancangan yang telah dilakukan
5. Menguji hasil ketepatan dari sistem yang telah dibuat
6. Memberikan saran dan kesimpulan

Tahapan penelitian yang dilakukan digambarkan pada gambar 3.1



Gambar 3.1 Metodologi Penelitian



### 3.1 Studi Pustaka

Mempelajari literatur dari berbagai sumber yang berhubungan dengan penerapan Algoritme PSO pada optimasi *Multiple Travelling Salesman Problem* (M-TSP) dalam penentuan rute angkutan sekolah. Literatur tersebut diperoleh dari buku, ebook, jurnal, penelitian sebelumnya dan dokumentasi proyek.

### 3.2 Pengumpulan Data

Penelitian mengambil data bagian Manajemen Yayasan Pembina Muslim Daarussalam, Sangatta Utara, Kabupaten Kutai Timur. Data tersebut merupakan angkutan sekolah SDIT Daarussalaam berupa nama dan alamat siswa, serta sopir yang bertugas dalam penjemputan. Terdapat 160 data alamat siswa yang menggunakan jasa angkutan sekolah dan 8 armada angkutan sekolah. Implementasi sistem diharapkan dapat memberikan rute yang telah dioptimasi sehingga dapat menjadi pertimbangan dalam penentuan strategi penjemputan oleh pihak yayasan.

### 3.3 Perancangan Sistem

Pada tahap ini dilakukan penyusunan langkah – langkah dalam penerapan Algoritme PSO dalam permasalahan optimasi *Multiple Travelling Salesman Problem* (M-TSP) pada penentuan rute angkutan sekolah. Perancangan menjelaskan poses optimasi dengan diagram alir dan melakukan perhitungan manual pada setiap langkah dalam proses optimasi menggunakan Algoritme PSO untuk dibandingkan dengan hasil yang diberikan oleh sistem yang dikembangkan. Pada tahap ini juga dilakukan perancangan antarmuka sistem.

### 3.4 Implementasi

Implementasi merupakan langkah untuk mengembangkan sistem sesuai dengan tahap perancangan. Implementasi pada tugas akhir ini dengan menerapkan Algoritme PSO pada penentuan rute angkutan sekolah dengan alamat yang telah dibreikan oleh pihak Yayasan Pembina Muslim Daarussalaam. Sistem dikembangkan menggunakan bahasa pemrograman PHP. *Ouput* dari sistem yang dibuat berupa rute hasil optimasi dan ditampilkan pada halaman website.

### 3.5 Hasil dan Pembahasan

Pada tahap ini dilakukan pembahasan hasil dari sistem yang telah diimplementasikan seta dilakukan penjelasan bagaimana sistem tersebut akan diuji.

### 3.6 Penarikan Kesimpulan

Dalam tahap ini peneliti akan memberikan hasil kesimpulan dari hasil penelitian yang telah dilakukan, kesimpulan yang diberikan didapat dari hasil analisa dan dan pengujian kebenaran sistem terhadap data yang diujikan.



Kemudian diberikan saran sebagai rekomendasi yang diharapkan dapat membantu dalam penelitian yang akan datang.



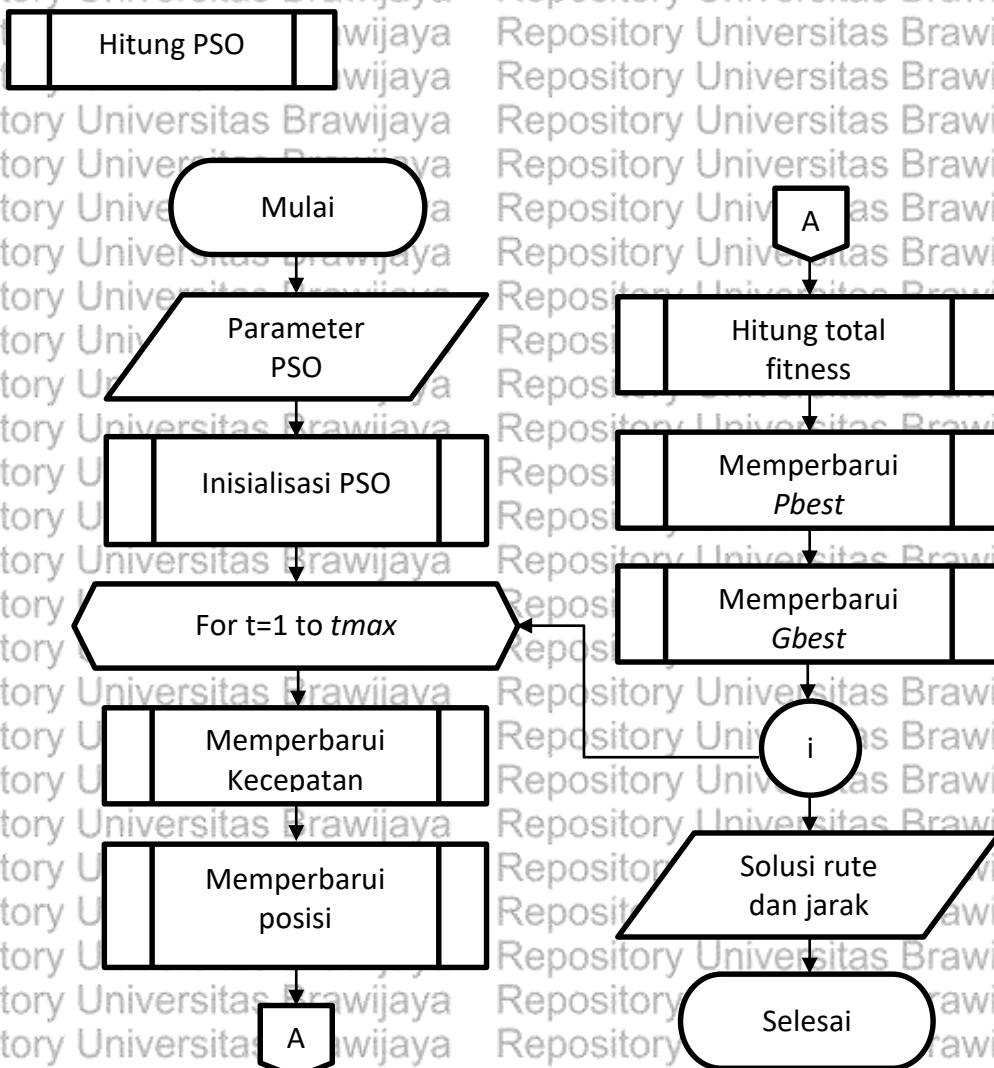


## BAB 4 PERANCANGAN

Pada bab ini akan dijelaskan tentang analisa dalam perancangan sistem yang akan dibangun dan penjelasan secara rinci mengenai Algoritme PSO, serta perancangan antarmuka sistem.

### 4.1. Perancangan Algoritme

Algoritme yang akan digunakan dalam memecahkan permasalahan optimasi *Multiple Travelling Salesman Problem* (M-TSP) adalah Algoritme PSO. pada Algoritme PSO dilakukan inialisasi yang terdiri dari posisi awal, kecepatan, *Pbest* dan *Gbest*, kemudian dalam setiap iterasi seluruh partikel akan diperbarui kecepatannya berdasarkan nilai *Pbest* dan *Gbest* dan kemudian akan diperbarui posisinya berdasarkan nilai kecepatan. Setelah itu akan dilakukan perbaruan pada *Pbest* dan *Gbest* berdasarkan nilai total fitness. Iterasi akan dilanjutkan sampai syarat berhenti terpenuhi atau mencapai konvergensi. Diagram alir Aloritme PSO digambarkan pada Gambar 4.1.





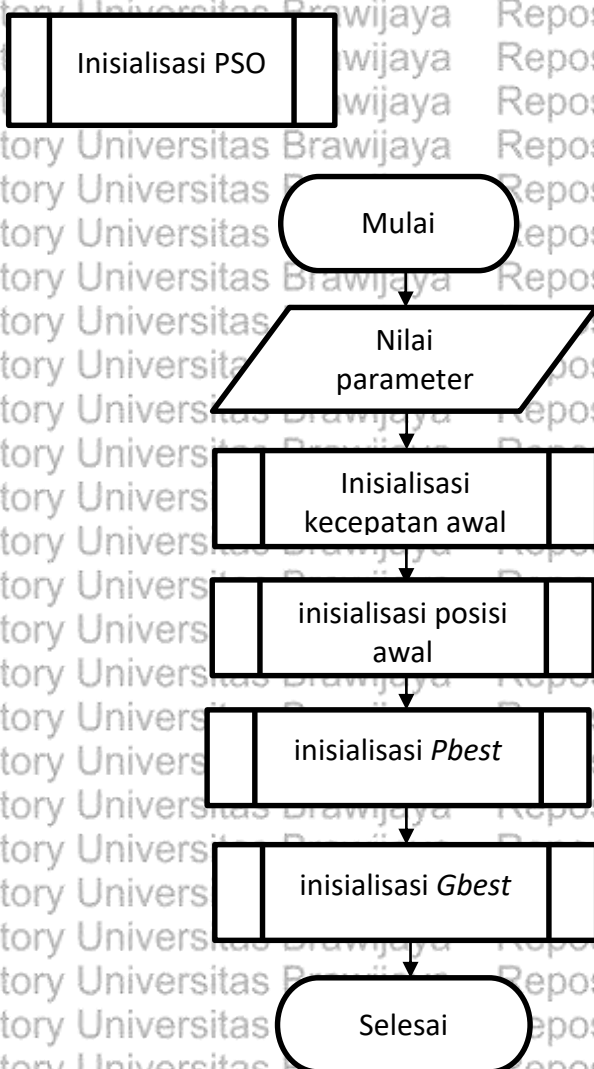
Gambar 4.1 Diagram Alir *Algorithm Particel Swarm Optimization (PSO)*

#### 4.1.1. Nilai Parameter

Tahap ini dilakukan untuk menentukan nilai parameter dari *Algoritme PSO* yaitu jumlah iterasi, jumlah kuota setiap sopir, ukuran *popsize*, *C1* dan *C2*.

#### 4.1.2. Inisialisasi PSO

Pada proses Inisialisasi PSO terdiri dari inisialisasi kecepatan awal, inisialisasi posisi awal partikel dan hitung fitness, inisialisasi *Pbest*, dan inisialisasi *Gbest*. Diagram alir Inisialisasi PSO digambarkan pada Gambar 4.2.



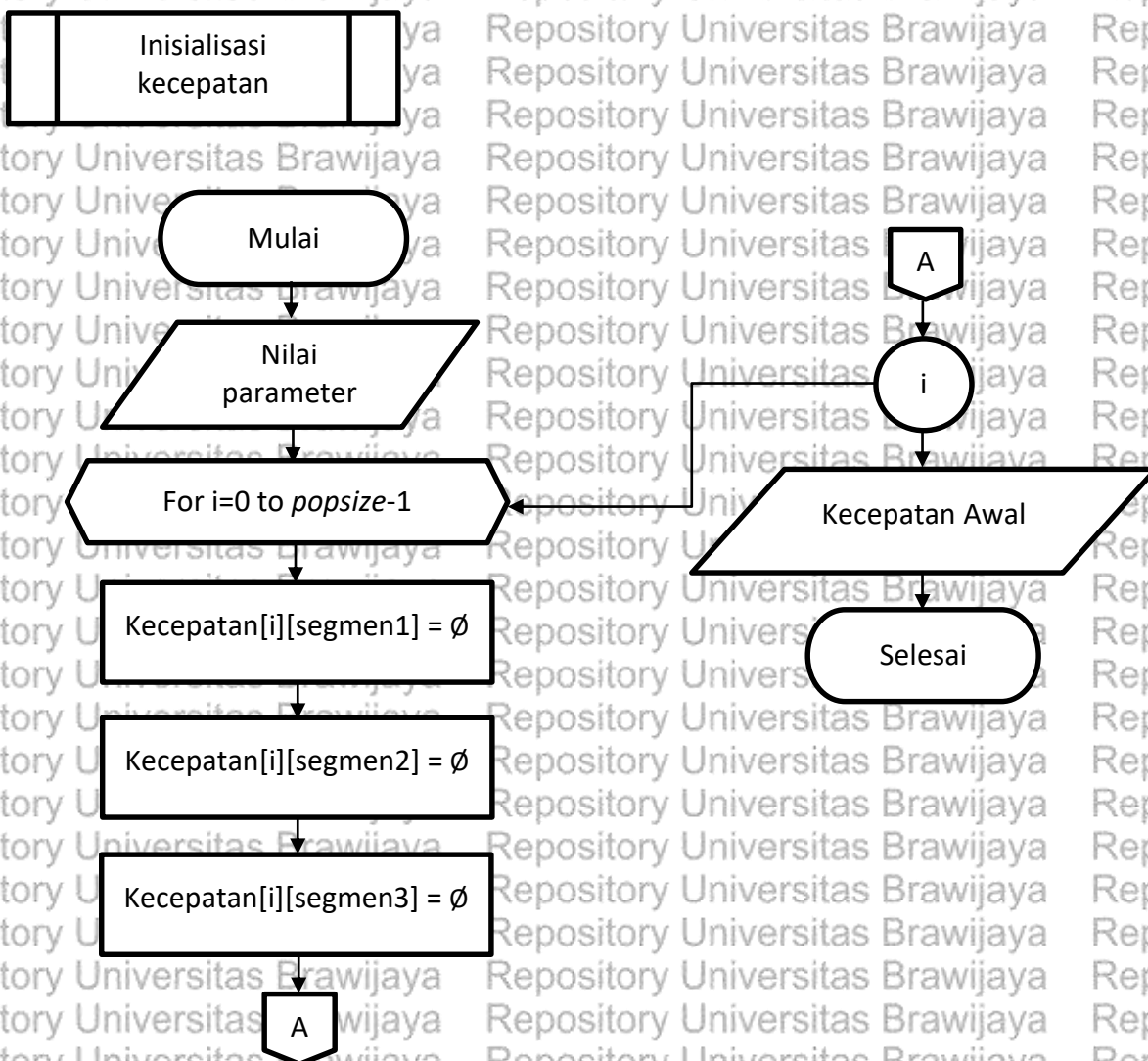
Gambar 4.2 Diagram Alir Inisialisasi PSO



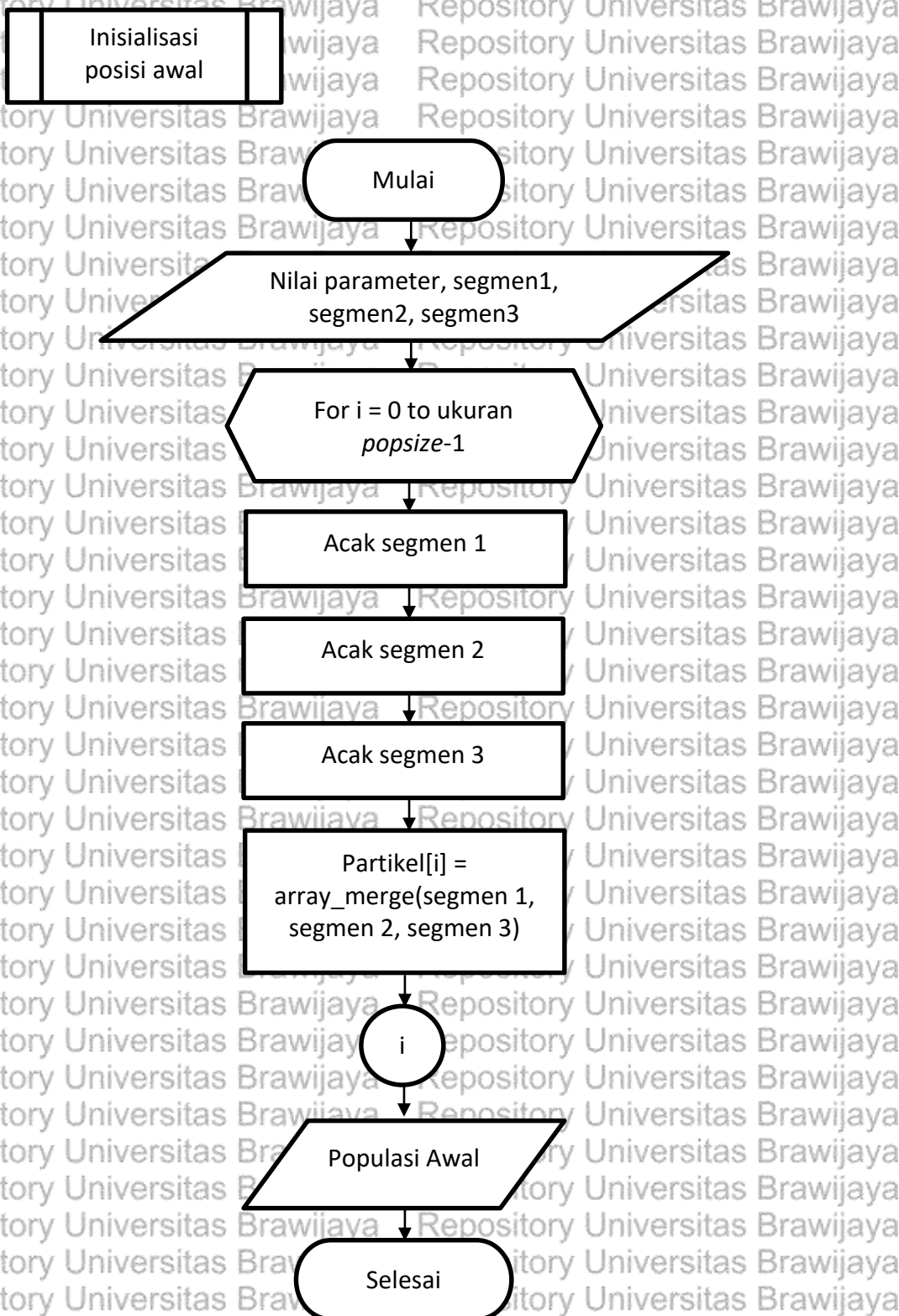
#### 4.1.2.1. Inisialisasi Kecepatan Awal dan Posisi Awal

Kecepatan Awal dan Posisi Awal diinisialisasi sebanyak  $popsiz$ . Pada iterasi pertama ( $t=0$ ) kecepatan awal diinisialisasi dengan nilai  $\emptyset$ . Pada inisialisasi posisi awal partikel, akan dihasilkan array yang terbentuk dari 3 segmen. Segmen pertama menunjukkan rute keberangkatan untuk seluruh siswa, segmen kedua menunjukkan rute kepulangan gelombang pertama untuk siswa TK serta kelas 1 dan 2, sementara segmen 3 yang menunjukkan rute kepulangan gelombang kedua untuk siswa SD kelas 3-6.

Untuk mendapatkan posisi awal partikel dilakukan perulangan sebanyak  $popsiz$ , kemudian pada setiap iterasi dilakukan pengacakan pada tiap-tiap segmen. Hasil dari proses acak tersebut kemudian digabungkan kembali dan disimpan pada suatu array. Diagram alur inisialisasi posisi awal partikel digambarkan pada Gambar 4.3, sedangkan diagram alur inisialisasi kecepatan awal partikel digambarkan pada Gambar 4.4.



Gambar 4.3 Diagram Alir Inisialisasi Kecepatan Awal

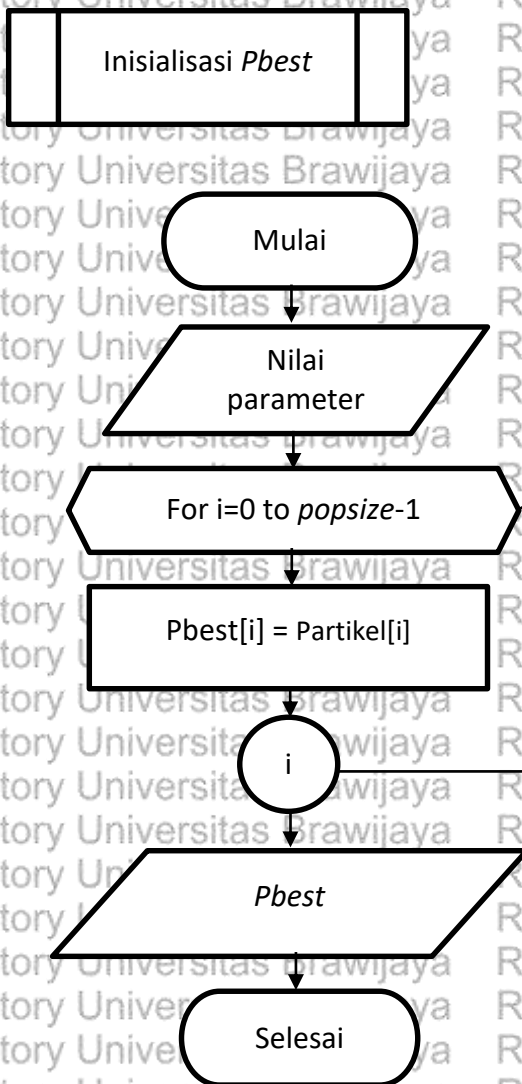


Gambar 4.4 Diagram Alir Inisialisasi Posisi Awal

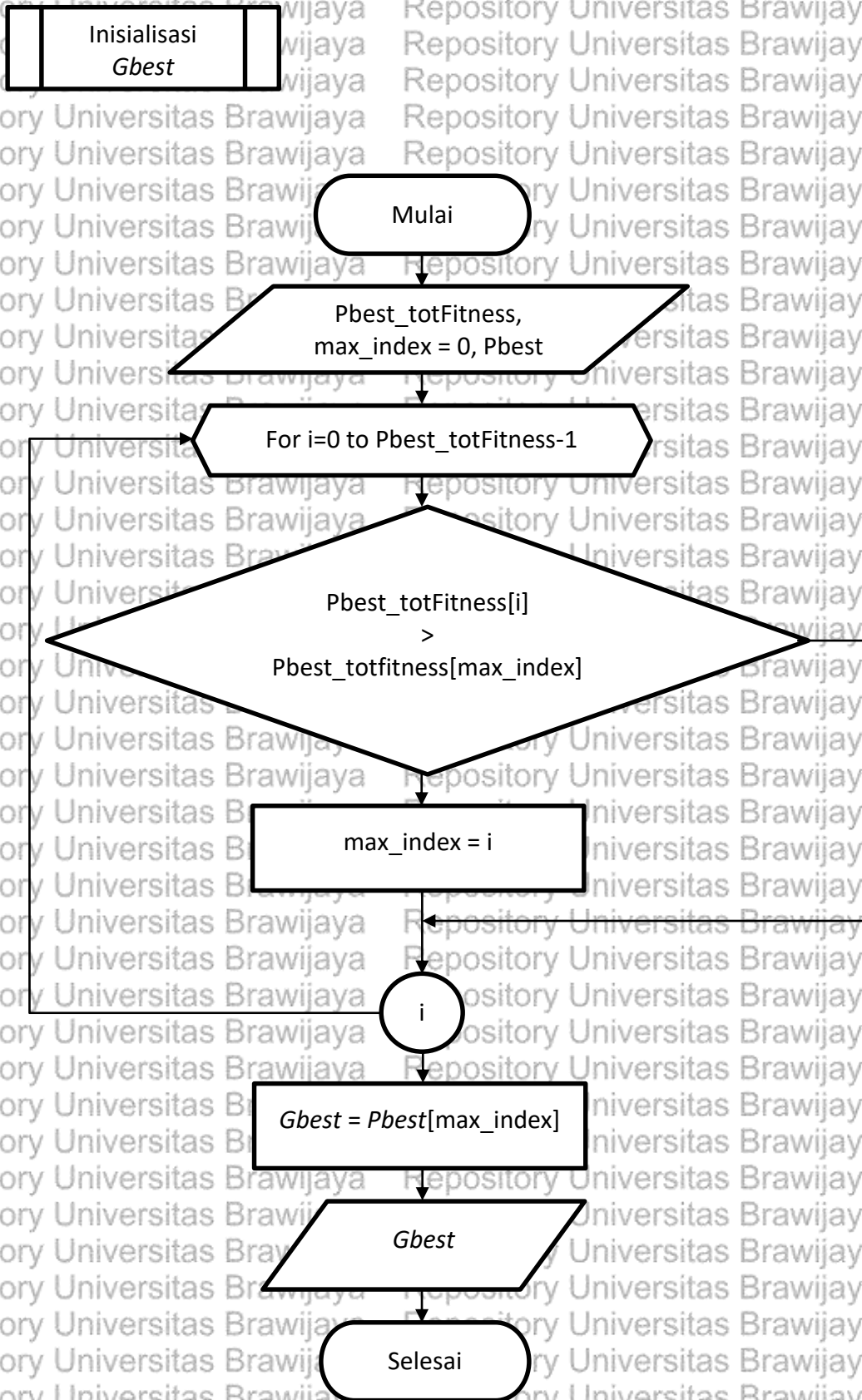


#### 4.1.2.2. Inisialisasi $Pbest$ dan $Gbest$

$Personal Best$  ( $Pbest$ ) merupakan representasi posisi terbaik yang pernah dicapai suatu partikel yang didapatkan dengan membandingkan nilai fitness partikel sekarang dengan yang sebelumnya. Di tahap inisialisasi pada  $t=0$ , nilai  $Pbest$  akan sama dengan nilai posisi awal partikel.  $Global Best$  ( $Gbest$ ) merupakan posisi terbaik partikel yang diperoleh dengan membandingkan nilai fitness terbaik dari seluruh  $Pbest$  partikel. Proses Inisialisasi  $Pbest$  digambarkan pada Gambar 4.5 sedangkan proses Inisialisasi  $Gbest$  digambarkan pada Gambar 4.6.



Gambar 4.5 Diagram Alir Inisialisasi  $Pbest$



Gambar 4.6 Diagram Alir Inisialisasi *Gbest*



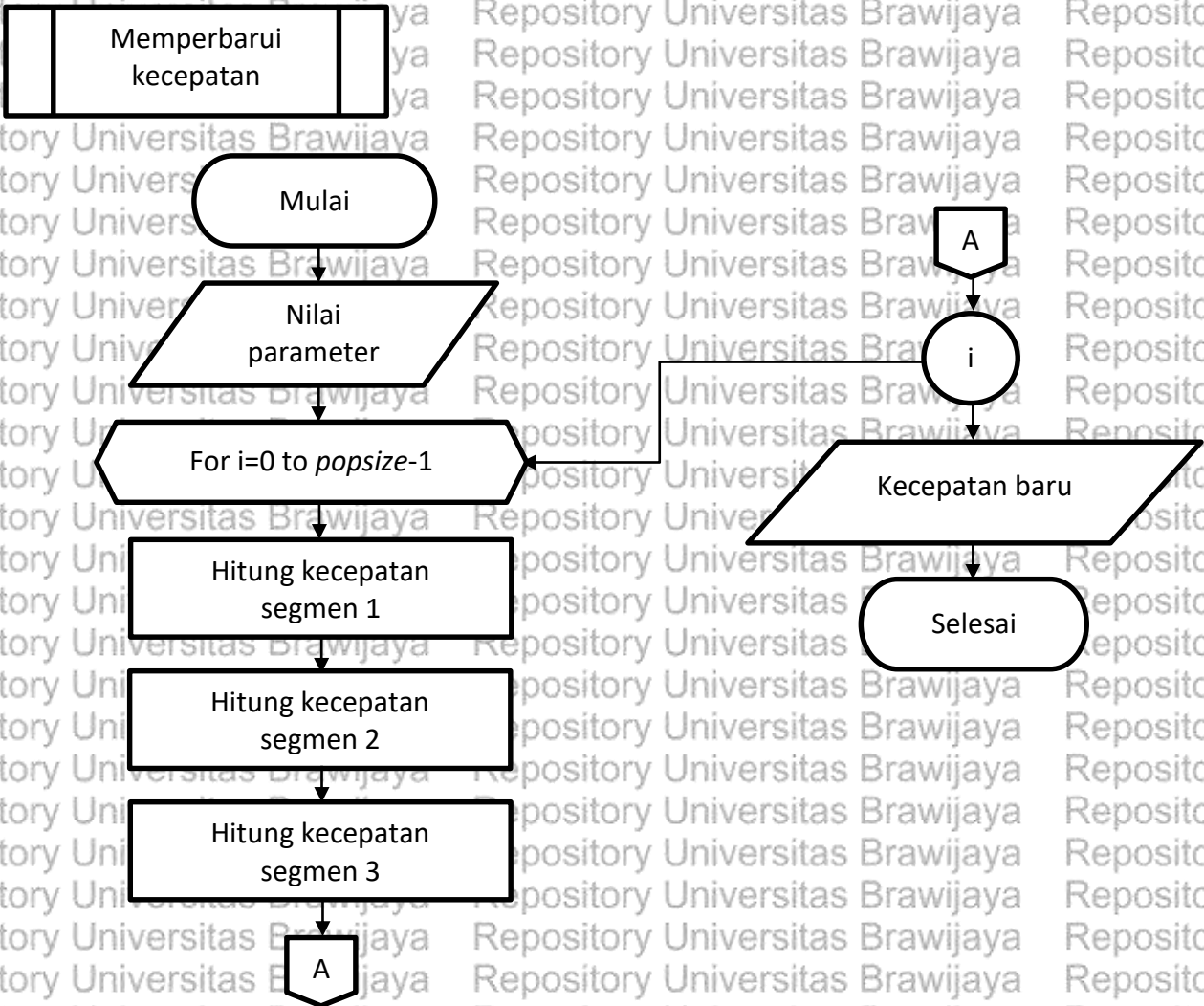
### 4.1.3. Memperbarui Kecepatan

Memperbarui kecepatan dilakukan setiap iterasi dengan tujuan untuk menentukan arah perpindahan dari suatu partikel. Memperbarui kecepatan dihitung menggunakan rumus berdasarkan Clerc:

$$V_i^{t+1} = C_1 \cdot V_i^t \oplus C_2 \left( (Pbest_i^t + \frac{1}{2} (Gbest_g^t - Pbest_i^t)) - X_i^t \right) \quad (4.1)$$

Pada proses perhitungan kecepatan, agar keanggotaan dari tiap-tiap segmen pada partikel tidak rusak maka perhitungan dibagi menjadi 3 proses pula. Hal ini dilakukan dengan membagi  $Pbest$ ,  $Gbest$ , dan posisi partikel menjadi 3 segmen kembali. Sehingga dapat dihitung kecepatan tiap2 segmennya.

Proses dalam memperbarui kecepatan digambarkan pada gambar 4.7.

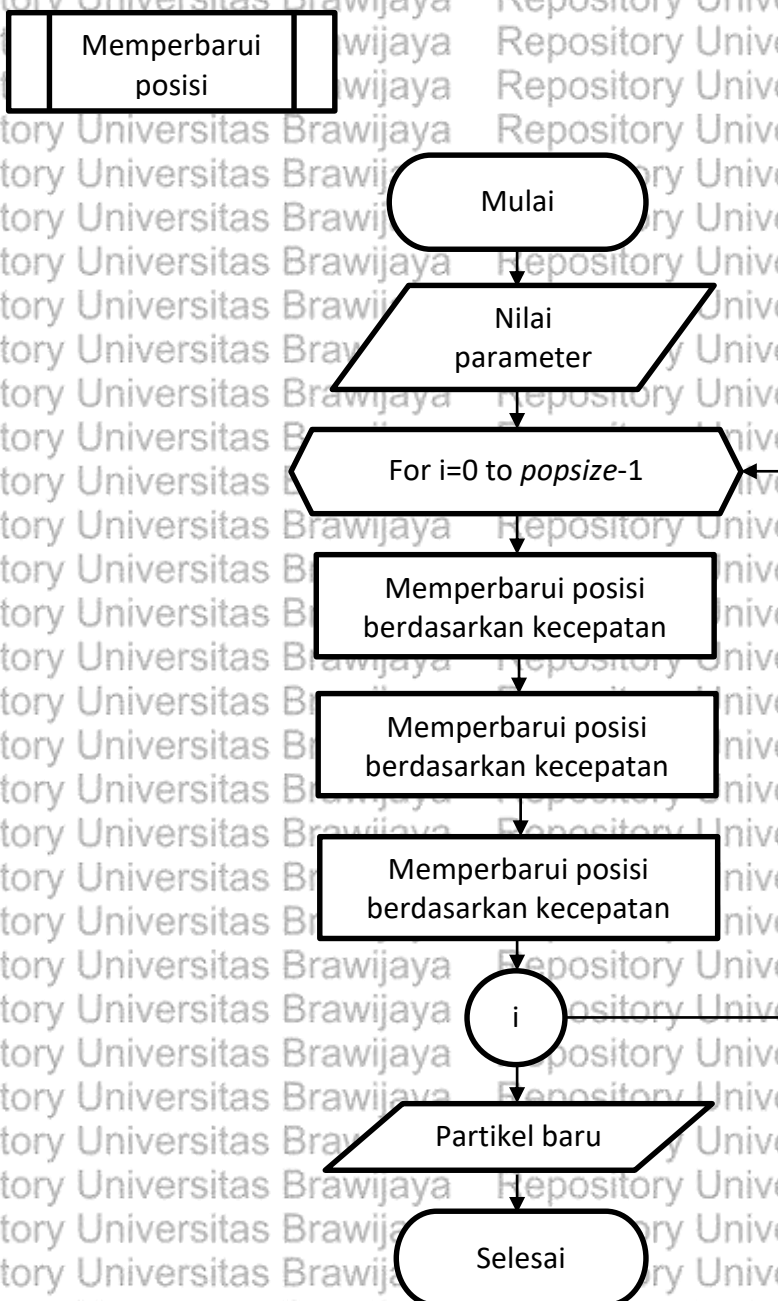


Gambar 4.7 Diagram Alir Memperbarui Kecepatan



#### 4.1.4. Memperbarui Posisi

Memperbarui posisi dilakukan untuk memperbaiki posisi dari suatu partikel berdasarkan kecepatan. Pada proses ini akan menghasilkan partikel baru yang akan dihitung kembali total jarak dan nilai fitnessnya. Proses dalam memperbarui posisi digambarkan pada gambar 4.8.



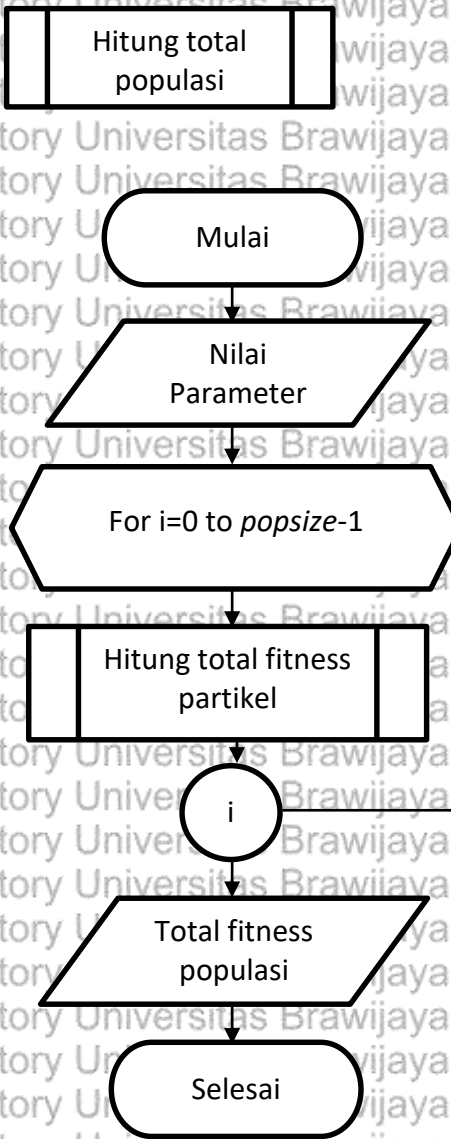
Gambar 4.8 Diaram Alir Memperbarui Posisi Partikel



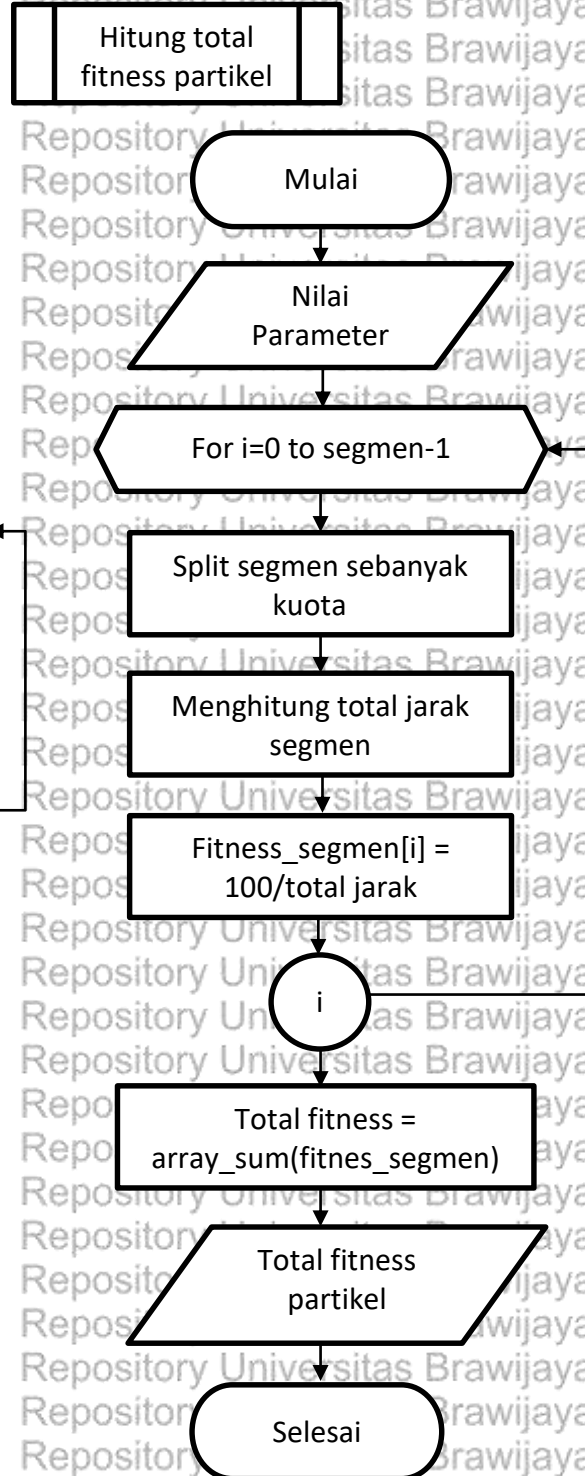


#### 4.1.5. Menghitung Jarak dan Fitness

Tahapan ini dilakukan untuk mengetahui kualitas dari setiap partikel. Nilai fitness dapat dihitung setelah nilai jarak setiap partikel didapatkan, nilai fitness didapatkan dari perhitungan setiap segmen pada tiap-tiap partikel. detail tiap-tiap proses digambarkan pada Gambar 4.9 dan 4.10.



Gambar 4.9 Diagram Alir Perhitungan Fitness

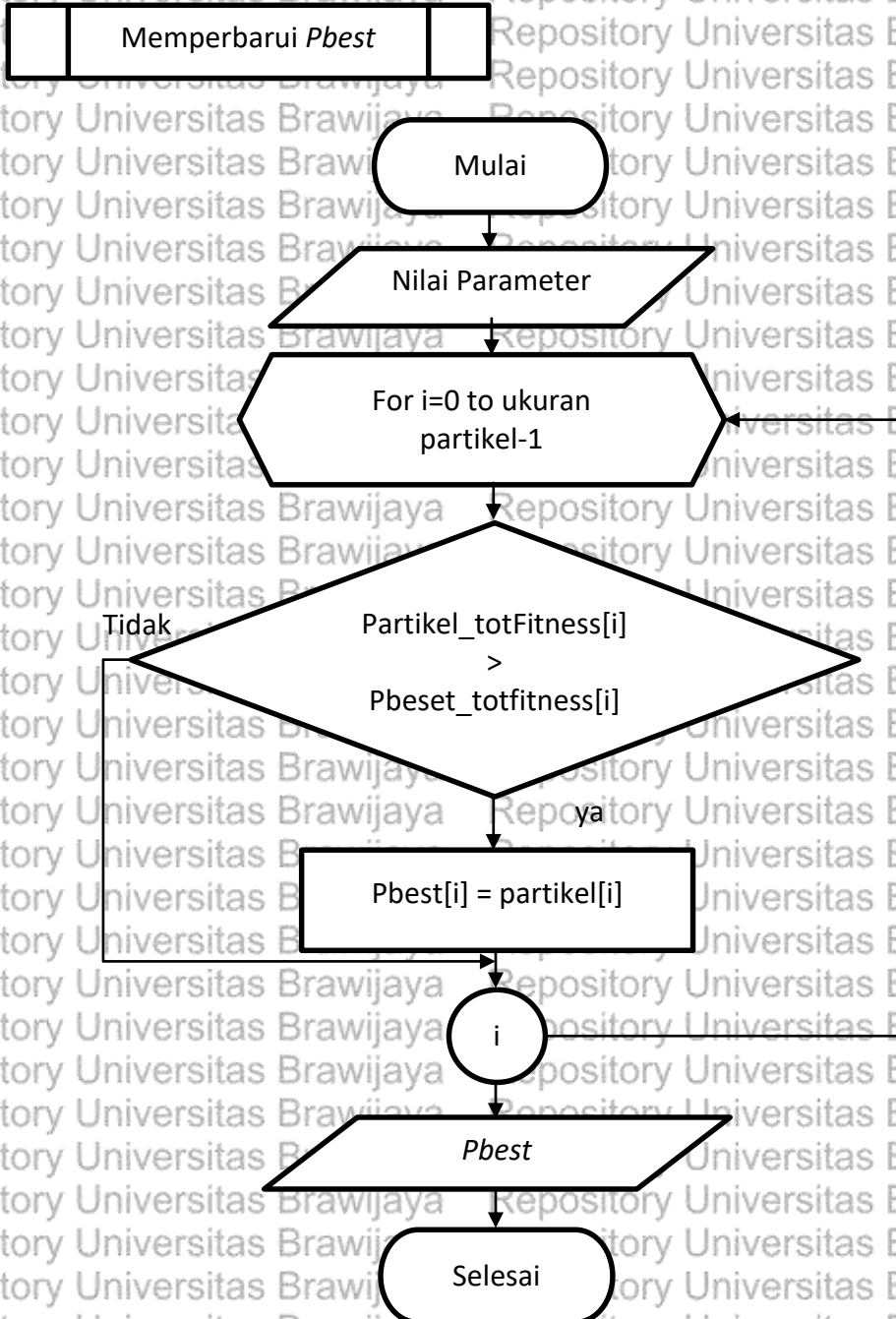


Gambar 4.10 Diagram Alir Perhitungan Total Jarak



### Memperbarui $P_{best}$ dan $G_{best}$

Tahap ini dilakukan untuk memperbarui nilai  $P_{best}$  dan  $G_{best}$  berdasarkan partikel baru tiap iterasi yang didapatkan dengan membandingkan nilai  $P_{best}$  dan  $G_{best}$  sekarang dengan  $P_{best}$  dan  $G_{best}$  sebelumnya. Tahapan Memperbarui  $G_{best}$  sama seperti proses inisialisasi  $G_{best}$  pada Gambar 4.5, sedangkan untuk tahapan memperbarui  $P_{best}$  dapat dilihat pada Gambar 4.11.



Gambar 4.11 Diagram Alir Memperbarui  $P_{best}$



## 4.2. Perhitungan Manual

Pada bagian ini akan dijelaskan perhitungan manual dari permasalahan optimasi MTSP yang akan diselesaikan menggunakan Algoritme *PSO*. Dataset yang digunakan dapat dilihat pada Tabel 4.1 Angka 0 merupakan id dari sekolah sedangkan angka 1-12 merupakan representasi dari id siswa. Isi dari tabel merupakan jarak alamat pada baris dengan alamat pada kolom dan sebaliknya. Jadwal kepulangan untuk siswa dibagi menjadi 2 yaitu untuk siswa TK dan SD kelas 1 dan 2 dengan jam kepulangan 13.00 dan untuk siswa kelas 3-6 dengan jam kepulangan 15.00. Detail dari dataset yang ditunjukkan pada Tabel 4.1 dan 4.2.

**Tabel 4.1 Dataset Perhitungan Manual**

	0	1	2	3	4	5	6
0	0	12.59	17.42	20.17	6.95	1.09	6.91
1	12.59	0	3.85	11.53	23.73	11.55	19.07
2	17.42	3.85	0	14.88	21.54	18.15	21.91
3	20.17	11.53	14.88	0	21.96	6.96	14.73
4	6.95	23.73	21.54	21.96	0	9.05	7.08
5	1.09	11.55	18.15	6.96	9.05	0	11.63
6	6.91	19.07	21.91	14.73	7.08	11.63	0

**Tabel 4.2 Keterangan Dataset**

Id	Nama	Kelas	Alamat
0	SDIT Daarussalaam		Jl. Jenderal Sudirman no. 1, Swarga Bara
1	Aisyabila Auliah	2 Abyar Ali	Jl. Yos Sudarso 1 Gg Seruni 4 No 92
2	Khumaira Hasna Hazima	1 Cordoba	Jl. APT. Pranoto RT. 60 No. 13 Gang Kuda Kaltim
3	Naura Athifa Khairunnisa	2 Muzdalifah	Jl. Apt Pranoto No. 50
4	Alya Triana Parakkasi	3 Al Fath	Perum Swarga Bara Extention Blok G. 22
5	Alfiah Malihah Balqis	4 Jabal Thoriq	Jl. Yos Sudarso II Gg. Sahabat no. 48F
6	Zaskia Alfa Ramadhani	5 Marwah	Jl. Apt. Pranoto Perum Grand Sangatta No. 29

### 4.2.1. Nilai Parameter

Nilai parameter yang dibutuhkan dalam proses perhitungan algoritme particle swarm intelligence (PSO) adalah jumlah iterasi, jumlah *salesman*, jumlah *popsize*, *C1* dan *C2*. Untuk perhitungan manual berikut nilai dari parameter PSO:

Jumlah iterasi: 2

Kuota tiap sopir: 3



Ukuran partikel: 3

$$C1 = 1$$

$$C2 = 1$$

#### 4.2.2. Inisialisasi

Proses inisialisasi dilakukan untuk membentuk populasi awal dalam PSO. Pada proses ini terdiri dari inisialisasi kecepatan awal, posisi awal partikel,  $Pbest$ , dan  $Gbest$ .

##### 4.2.2.1. Inisialisasi Kecepatan Awal dan Posisi Awal Partikel

Pada tahap ini kecepatan awal dan posisi awal partikel diinisialisasi sebanyak  $popsiz$ . Pada tahap inisialisasi kecepatan awal semua partikel memiliki nilai ( $V_i(t) = \emptyset$ ), tabel kecepatan awal ditunjukkan oleh Tabel 4.3.

Posisi partikel terdiri dari 3 segmen, segmen 1 merupakan data yang menunjukkan rute keberangkatan yaitu seluruh siswa, segmen kedua menunjukkan rute kepulangan untuk siswa kelas 1 dan 2, pada kasus ini ditunjukkan oleh siswa dengan id 1-3. Sementara segmen 3 menunjukkan rute kepulangan untuk siswa kelas 3-6 yang ditunjukkan oleh siswa dengan id 4-6. Data segmen ditunjukkan oleh Tabel 4.4. Hasil dari proses acak tersebut kemudian digabungkan kembali sehingga didapatkan data posisi awal partikel seperti pada Tabel 4.5.

**Tabel 4.3 Inisialisasi Kecepatan**

$V_i(t)$	kecepatan		
$v1(0)$	$\emptyset$	$\emptyset$	$\emptyset$
$v2(0)$	$\emptyset$	$\emptyset$	$\emptyset$
$v3(0)$	$\emptyset$	$\emptyset$	$\emptyset$

**Tabel 4.4 Data Segmen**

Segmen	id					
Segmen1	1	2	3	4	5	6
Segmen2	1	2	3			
Segmen3	4	5	6			

**Tabel 4.5 Inisialisasi Posisi Awal Partikel**

$X_i(t)$	Partikel											
$x1(0)$	2	3	4	6	5	1	1	2	3	4	5	6
$x2(0)$	1	4	5	2	6	3	3	2	1	5	4	6
$x3(0)$	5	1	2	4	3	6	1	2	3	5	6	4



Selanjutnya setelah proses inisialisasi dilakukan perhitungan total jarak pada tiap-tiap partikel. Total jarak dibutuhkan untuk mendapatkan nilai fitness tiap-tiap partikel. Tiap-tiap segmen pada partikel dibagi berdasarkan kuota, pada kasus ini dimisalkan kuota untuk setiap sopir adalah 3 siswa, kemudian setelah dibagi akan dihitung jaraknya dimulai dari alamat sekolah dan akan berakhir ke sekolah pula. Proses perhitungan jarak setiap segmen pada partikel dapat dilihat pada Tabel 4.6 dan total jarak yang dihasilkan ditunjukkan pada Tabel 4.7.

**Tabel 4.6 Tabel Perhitungan jarak Setiap Segmen pada Partikel**

Xi(t)	Segmen1		Segmen2	Segmen3
	Sopir1	Sopir2	Sopir1	So1
X1(0)	0-2-3-4-0 = 61.21	0-6-5-1-0 = 42.68	0-1-2-3-0 = 51.49	0-4-5-6-0 = 34.54
X2(0)	0-1-4-5-0 = 46.46	0-2-6-3-0 = 74.23	0-3-2-1-0 = 51.49	0-5-4-6-0 = 24.13
X3(0)	0-5-1-2-0 = 33.91	0-4-3-6-0 = 50.55	0-1-2-3-0 = 51.49	0-5-6-4-0 = 26.75

**Tabel 4.7 tabel total jarak partikel**

Xi(t)	partikel												jarak1	jarak2	jarak3
X1(0)	2	3	4	6	5	1	1	2	3	4	5	6	103.89	51.49	34.54
X2(0)	1	4	5	2	6	3	3	2	1	5	4	6	120.69	51.49	24.13
X3(0)	5	1	2	4	3	6	1	2	3	5	6	4	84.46	51.49	26.75

Setelah mendapatkan total jarak tahap selanjutnya adalah melakukan perhitungan nilai fitness dengan rumus  $Fitness = 100/\text{total jarak}$ . Proses perhitungan fitness dapat dilihat pada Tabel 4.8.

**Tabel 4.8 Tabel Perhitungan Fitness**

jarak1	jarak2	jarak3	fitness1	fitness2	fitness3	totFitness
103.89	51.49	34.54	0.962557	1.942125	2.895194	5.799875
120.69	51.49	24.13	0.828569	1.942125	4.144219	6.914913
84.46	51.49	26.75	1.183992	1.942125	3.738318	6.864435

**4.2.2.2. Inisialisasi Pbest dan Gbest**

Personal Best (*Pbest*) merupakan representasi posisi terbaik yang pernah dicapai suatu partikel. Pada tahap inisialisasi, dikarenakan masih pada tahap inisialisasi ( $t=0$ ), sehingga nilai dari *Pbest* tiap partikel akan sama dengan nilai dari posisi awal partikel. Nilai *Pbest* dapat dilihat pada Tabel 4.9.

**Tabel 4.9 Tabel Inisialisasi Pbest**

Pbest <sub>i</sub> (t)	partikel												totFitness
Pbest <sub>1</sub> (0)	2	3	4	6	5	1	1	2	3	4	5	6	5.799875
Pbest <sub>2</sub> (0)	1	4	5	2	6	3	3	2	1	5	4	6	6.914913
Pbest <sub>3</sub> (0)	5	1	2	4	3	6	1	2	3	5	6	4	6.864435



Global Best merupakan posisi terbaik partikel yang diperoleh dengan membandingkan nilai fitness terbaik dari seluruh partikel. Pada Tabel 4.9 nilai totFitness tertinggi terapat pada  $Pbest_2(0)$  dengan nilai 6.914913 sehingga  $Gbest_1(0) = Pbest_2(0)$ .  $Gbest$  dapat dilihat pada Tabel 4.10.

Tabel 4.10 Tabel Inisialisasi  $Gbest$

Xi(t)	partikel												totFitness
X1(0)	1	4	5	2	6	3	3	2	1	5	4	6	6.914913

### 4.2.3. Memperbarui Kecepatan

Untuk mendapatkan hasil perbaruan kecepatan, dilakukan perhitungan pada setiap segmen untuk setiap partikel. Sehingga  $Pbest$ ,  $Gbest$ , dan posisi partikel di bagi menjadi 3 segmen kembali. Kemudian kecepatan dihitung menggunakan rumus berdasarkan Clerc pada persamaan 4.1

Misal untuk menghitung perbaruan kecepatan untuk  $V_1(0)$  pada segmen 1 (keberangkatan) :

$$V_1^{0+1} = c_1 \cdot V_1^0 \oplus c_2 \left( (Pbest_1^0 + \frac{1}{2} (Gbest_1^0 - Pbest_1^0)) - X_1^0 \right)$$

$$V_1^1 = 1.0 \oplus 1 \left( ([2 \ 3 \ 4 \ 6 \ 5 \ 1] + \frac{1}{2} ([1 \ 4 \ 5 \ 2 \ 6 \ 3] - [2 \ 3 \ 4 \ 6 \ 5 \ 1])) - [2 \ 3 \ 4 \ 6 \ 5 \ 1] \right)$$

Untuk menyelesaikan persamaan diatas, langkah pertama adalah dengan melakukan perhitungan pada kurung yang terdalam, yaitu dengan melakukan operasi *subtraction* (*position minus position*).

$$V_1^1 = 1.0 \oplus 1 \left( ([2 \ 3 \ 4 \ 6 \ 5 \ 1] + \frac{1}{2} ([1 \ 4 \ 5 \ 2 \ 6 \ 3] - [2 \ 3 \ 4 \ 6 \ 5 \ 1])) - [2 \ 3 \ 4 \ 6 \ 5 \ 1] \right)$$

Transposisi

$$(1, 6)$$

$$[1 \ 3 \ 4 \ 6 \ 5 \ 2]$$

Transposisi

$$(2, 3)$$

$$[1 \ 4 \ 3 \ 6 \ 5 \ 2]$$

Transposisi

$$(3, 5)$$

$$[1 \ 4 \ 5 \ 6 \ 3 \ 2]$$

Transposisi

$$(4, 6)$$

$$[1 \ 4 \ 5 \ 2 \ 3 \ 6]$$

Transposisi

$$(5, 6)$$

$$[1 \ 4 \ 5 \ 2 \ 6 \ 3]$$

Maka setelah perhitungan diatas, didapatkan hasil sebagai berikut:



$$V_1^1 = 1.0 \oplus 1(( [2\ 3\ 4\ 6\ 5\ 1] + \frac{1}{2}((1,6), (2,3), (3,5), (4,6), (5,6)) - [2\ 3\ 4\ 6\ 5\ 1] ))$$

Kemudian dilakukan perhitungan dengan operasi *multiplication (coefficient times velocity)*.

$$V_1^1 = 1.0 \oplus 1(( [2\ 3\ 4\ 6\ 5\ 1] + \frac{1}{2}((1,6), (2,3), (3,5), (4,6), (5,6)) - [2\ 3\ 4\ 6\ 5\ 1] ))$$

$$\|v\| = 5 \quad c = 0.5$$

Dimana  $\|v\|$  adalah panjang array sedangkan  $c$  adalah konstanta

$$\|v'\| = [c * \|v\|] = [0.5 * 5] = [2.5] = 3$$

$$v' = ((1,6), (2,3), (3,5))$$

Maka setelah perhitungan diatas, didapatkan hasil sebagai berikut:

$$V_1^1 = 1.0 \oplus 1(( [2\ 3\ 4\ 6\ 5\ 1] + ((1,6), (2,3), (3,5)) - [2\ 3\ 4\ 6\ 5\ 1] ))$$

Kemudian dilakukan perhitungan dengan operasi *move (position plus velocity)*.

$$V_1^1 = 1.0 \oplus 1(( [2\ 3\ 4\ 6\ 5\ 1] + ((1,6), (2,3), (3,5)) - [2\ 3\ 4\ 6\ 5\ 1] ))$$

$$[2\ 3\ 4\ 6\ 5\ 1] + ((1,6), (2,3), (3,5))$$

$$= [1\ 3\ 4\ 6\ 5\ 2] + ((2,3), (3,5))$$

$$= [1\ 4\ 3\ 6\ 5\ 2] + ((3,5))$$

$$= [1\ 4\ 5\ 6\ 3\ 2]$$

Maka setelah perhitungan diatas, didapatkan hasil sebagai berikut:

$$V_1^1 = 1.0 \oplus 1(( [1\ 4\ 5\ 6\ 3\ 2] - [2\ 3\ 4\ 6\ 5\ 1] ))$$

Kemudian dilakukan perhitungan dengan operasi *substraction (position minus position)*.

$$V_1^1 = 1.0 \oplus 1(( [1\ 4\ 5\ 6\ 3\ 2] - [2\ 3\ 4\ 6\ 5\ 1] ))$$

$$\text{Transposisi } (1,6)$$

$$[1\ 3\ 4\ 6\ 5\ 2]$$

$$\text{Transposisi } (2,3)$$

$$[1\ 4\ 3\ 6\ 5\ 2]$$

$$\text{Transposisi } (3,5)$$

$$[1\ 4\ 5\ 6\ 3\ 2]$$

Maka setelah perhitungan diatas, didapatkan hasil sebagai berikut:

$$V_1^1 = 1.0 \oplus 1((1,6), (2,3), (3,5))$$



Kemudian dilakukan perhitungan dengan operasi *multiplication (coefficient times velocity)*.

$$V_1^1 = 1.0 \oplus 1((1, 6), (2, 3), (3, 5))$$

$$\|v\| = 3 \quad c = 1$$

$$\|v'\| = [c * \|v\|] = [3 * 1] = 3$$

$$v' = ((1, 6), (2, 3), (3, 5))$$

Maka setelah perhitungan diatas, didapatkan hasil sebagai berikut:

$$V_1^1 = 1.0 \oplus ((1, 6), (2, 3), (3, 5))$$

Sehingga hasil dari proses memperbaiki kecepatan  $V_1(1)$  pada segmen 1 adalah sebagai berikut:

$$V_1^1 = ((1, 6), (2, 3), (3, 5))$$

Kemudian langkah tersebut dilanjutkan dengan langkah yang sama untuk mencari  $V_1(1)$  pada segmen 2 dan 3 dan kemudian diulangi untuk mendapatkan  $V_2(1)$  dan  $V_3(1)$  sehingga akan didapatkan hasil kecepatan seperti pada tabel 4.11.

**Tabel 4.11 Hasil Tahap Memperbarui Kecepatan**

$V_i(t)$	kecepatan		
$V_1(1)$	$((1, 6), (2, 3), (3, 5))$	$(1, 3)$	$(1, 2)$
$V_2(1)$	$\emptyset$	$\emptyset$	$\emptyset$
$V_3(1)$	$((1, 2), (2, 4))$	$(1, 3)$	$(2, 3)$

**4.2.4. Memperbarui Posisi**

Tahap ini untuk memperbaiki posisi dari partikel berdasarkan nilai kecepatan. Untuk menghitung nilai dari posisi partikel yang baru dapat menggunakan persamaan sebagai berikut:

$$X_i^{t+1} = X_i^t + V_i^{t+1}$$

Kemudian partikel dipecah kembali menjadi 3 segmen dan diperbarui kecepatannya pada masing2 segmen. Tabel dari posisi partikel dan kecepatan dapat dilihat pada Tabel 4.12.

**Tabel 4.12 Memperbarui Posisi berdasarkan Kecepatan**

$X_i(t)$	partikel												$V_i(t+1)$	kecepatan		
$X_1(0)$	2	3	4	6	5	1	1	2	3	4	5	6	$V_1(1)$	$((1, 6), (2, 3), (3, 5))$	$(1, 3)$	$(1, 2)$
$X_2(0)$	1	4	5	2	6	3	3	2	1	5	4	6	$V_2(1)$	$\emptyset$	$\emptyset$	$\emptyset$
$X_3(0)$	5	1	2	4	3	6	1	2	3	5	6	4	$V_3(1)$	$((1, 2), (2, 4))$	$(1, 3)$	$(2, 3)$





Misal untuk menghitung  $X_1(1)$ :

$$X_1^0 \text{ Segmen 1} = [2 \ 3 \ 4 \ 6 \ 5 \ 1], \quad V_1^1 \text{ Segmen 1}$$

$$X_1^1 = X_2^0 + V_2^1$$

$$X_1^1 = [2 \ 3 \ 4 \ 6 \ 5 \ 1] + ((1, 6), (2, 3), (3, 5))$$

$$X_1^1 = [1 \ 3 \ 4 \ 6 \ 5 \ 2] + ((2, 3), (3, 5))$$

$$X_1^1 = [1 \ 4 \ 3 \ 6 \ 5 \ 2] + ((3, 5))$$

$$X_1^1 = [1 \ 4 \ 5 \ 6 \ 3 \ 2]$$

$$X_1^0 \text{ Segmen 2} = [1 \ 2 \ 3], \quad V_1^1 \text{ Segmen 2}$$

$$X_1^1 = X_2^0 + V_2^1$$

$$X_1^1 = [1 \ 2 \ 3] + ((1, 3))$$

$$X_1^1 = [3 \ 2 \ 1]$$

$$X_1^0 \text{ Segmen 3} = [4 \ 5 \ 6], \quad V_1^1 \text{ Segmen 3}$$

$$X_1^1 = X_2^0 + V_2^1$$

$$X_1^1 = [4 \ 5 \ 6] + ((1, 2))$$

$$X_1^1 = [5 \ 4 \ 6]$$

Kemudian masing-masing segmen digabungkan kembali sehingga didapatkan  $X_1^1$  sebagai berikut:

$$X_1^1 = [1 \ 4 \ 5 \ 6 \ 3 \ 2 \ 3 \ 2 \ 1 \ 5 \ 4 \ 6]$$

Sehingga setelah melakukan proses memperbaiki posisi pada seluruh partikel, akan didapatkan hasil seperti pada Tabel 4.13.

**Tabel 4.13 Hasil Tahap Memperbarui Posisi**

$X_i(t)$	partikel											totFitness	
$X_1(1)$	1	4	5	6	3	2	3	2	1	5	4	6	7.0824
$X_2(1)$	1	4	5	2	6	3	3	2	1	5	4	6	6.9149
$X_3(1)$	1	4	2	5	3	6	3	2	1	5	4	6	7.039

#### 4.2.5. Memperbarui $Pbest$

Nilai pada proses memperbaiki  $Pbest$  didapatkan dengan membandingkan  $Pbest$  pada iterasi sebelumnya dengan nilai dari posisi yang sekarang pada tiap-tiap partikel. Setelah melakukan proses memperbaiki posisi partikel pada Tabel 4.13 maka nilai tersebut akan dibandingkan dengan nilai  $Pbest$  pada Tabel 4.9, apabila nilai fitness partikel baru lebih baik maka nilai dari  $Pbest$  akan diperbarui dengan nilai partikel tersebut.



**Tabel 4.14 Memperbarui  $Pbest$**

Pbesti(t)	Partikel	totFitness	$X_i(t)$	Partikel	totFitness
X1(0)	[2 3 46 5 1 1 2 3 4 5 6]	5.799875	X1(1)	[1 4 5 6 3 2 3 2 1 5 4 6]	7.0824
X2(0)	[1 4 5 2 6 3 3 2 1 5 4 6]	6.914913	X2(1)	[1 4 5 2 6 3 3 2 1 5 4 6]	6.9149
X3(0)	[5 1 2 4 3 6 1 2 3 5 6 4]	6.864435	X3(1)	[1 4 2 5 3 6 3 2 1 5 4 6]	7.039

Dipebarui

Sehingga hasil dari  $Pbest$  yang baru dapat dilihat pada Tabel 4.15

**Tabel 4.15 Hasil Tahap Memperbarui  $Pbest$**

$X_i(t)$	partikel												totFitness
Pbest1(1)	1	4	5	6	3	2	3	2	1	5	4	6	7.0824
Pbest2(1)	1	4	5	2	6	3	3	2	1	5	4	6	6.9149
Pbest3(1)	1	4	2	5	3	6	3	2	1	5	4	6	7.039

**4.2.6. Memperbarui  $Gbest$**

Didapatkan dari nilai tertinggi  $Pbest$  pada Tabel 4.15 bahwa nilai fitness tertinggi terdapat pada  $Pbest_3(1)$  dengan nilai 2.24972 sehingga  $Gbest_1(1) = Pbest_3(1)$ .  $Gbest$  dapat dilihat pada Tabel 4.12.

**Tabel 4.16 Hasil Tahap Memperbarui  $Gbest$**

$X_i(t)$	partikel												totFitness
$Gbest_1(0)$	1	4	5	6	3	2	3	2	1	5	4	6	7.0824

Nilai dari  $Gbest$  inilah yang nantinya akan menjadi solusi dari permasalahan. Kemudian partikel tersebut akan dibagi menjadi 3 segmen, segmen 1 merupakan data yang menunjukkan rute keberangkatan, segmen 2 menunjukkan rute kepulangan untuk siswa kelas 1 dan 2, sementara segmen 3 menunjukkan rute kepulangan untuk siswa kelas 3-6. Dan pada tiap-tiap segmen dibagi kembali sesuai kuota. Sehingga untuk keberangkatan didapatkan hasil pada Tabel 4.17, dan untuk kepulangan ditunjukkan pada Tabel 4.18 dan 4.19, bentuk graf dari rute keberangkatan ditunjukkan pada Gambar 4.12, sedangkan bentuk graf untuk rute kepulangan ditunjukkan pada Gambar 4.13

**Tabel 4.17 Rute Keberangkatan**

segmen1	sopir1			sopir2		
	1	4	5	6	3	2

**Tabel 4.18 Rute Keberangkatan Siswa Kelas 1 dan 2**

segmen2	sopir1		
	3	2	1



Tabel 4.19 Rute Kepulangan Siswa Kelas 3-6

segmen3	sopir1		
	5	4	6



Gambar 4.13 Graf Rute Keberangkatan Siswa



Gambar 4.12 Graf Rute Kepulangan Siswa

### 4.3. Perancangan Pengujian

Pada bagian ini membahas perancangan pengujian untuk menguji hasil dari sistem yang dibuat. Perancangan pengujian terdiri dari 3 pengujian yaitu pengujian jumlah iterasi, ukuran populasi, sistem, dan koefisien akselerasi.

#### 4.3.1. Pengujian Konvergensi

Pengujian konvergensi dilakukan untuk mengetahui pengaruh banyaknya iterasi yang dilakukan terhadap fitness yang dihasilkan. Pengujian ini dilakukan 10 kali menggunakan 1000 iterasi. Pada pengujian iterasi ini menggunakan 160 siswa dengan 8 salesman, jumlah *popsize* = 10, nilai koefisien akselerasi yang digunakan adalah berdasarkan pengujian kombinasi koefisien akselerasi yang dilakukan oleh M. Khusnul Azhari (2018) yaitu  $C1 = 1$  dan  $C2 = 1.5$  sehingga akan didapatkan hasil seperti Tabel 4.20.

Tabel 4.20 Hasil Pengujian Konvergensi

iterasi ke	Avg Fitness
1	...
2	...
3	...
4	...
5	...
6	...
7	...
8	...
9	...
10	...



### 4.3.2. Pengujian Ukuran Partikel

Pengujian ukuran populasi dilakukan untuk mengetahui pengaruh besarnya ukuran partikel yang dilakukan terhadap fitness yang dihasilkan. Pengujian ini dilakukan 10 kali menggunakan partikel 10 - 100 dengan kelipatan 10. Pada pengujian iterasi ini menggunakan 160 siswa dengan 8 salesman, jumlah, nilai  $C1 = 1$  dan  $C2 = 1.5$ , dengan jumlah iterasi terbaik yang didapatkan dari hasil pengujian iterasi sehingga akan didapatkan hasil seperti Tabel 4.21.

**Tabel 4.21 Perancangan Pengujian Ukuran Partikel**

Ukuran Populasi	Nilai <i>Fitness-i</i>							Avg <i>Fitness</i>	Max <i>Fitness</i>
	P1	P2	P3	P4	P5	...	P10		
10	...	...	...	...	...	...	...	...	...
20	...	...	...	...	...	...	...	...	...
30	...	...	...	...	...	...	...	...	...
40	...	...	...	...	...	...	...	...	...
50	...	...	...	...	...	...	...	...	...
60	...	...	...	...	...	...	...	...	...
70	...	...	...	...	...	...	...	...	...
80	...	...	...	...	...	...	...	...	...
90	...	...	...	...	...	...	...	...	...
100	...	...	...	...	...	...	...	...	...

### 4.3.3. Pengujian Perbandingan Hasil Aktual dengan Hasil Penelitian

Pengujian ukuran partikel dilakukan untuk perbandingan antara hasil penelitian dengan data aktual yang digunakan oleh pihak YPM Daarussalaam. Pada pengujian ini ada aktual jempitan akan dibandingkan dengan solusi dengan nilai fitness terbaik dari hasil perhitungan PSO dengan parameter optimum yang telah didapat pada pengujian sebelumnya. Setelah pengujian akan didapatkan hasil seperti Tabel 4.23.

**Tabel 4.23 Perancangan Pengujian Perbandingan Hasil Aktual dengan Hasil Penelitian**

Metode	Cost (KM)			Fitness			Total Fitness
	Segmen 1	Segmen 2	Segmen 3	Segmen 1	Segmen 2	Segmen 3	
Hasil Asli	...	...	...	...	...	...	...
hasil 1	...	...	...	...	...	...	...
hasil 2	...	...	...	...	...	...	...
hasil 3	...	...	...	...	...	...	...
hasil 4	...	...	...	...	...	...	...
hasil 5	...	...	...	...	...	...	...



#### 4.4. Perancangan Antarmuka

Pada bagian ini membahas perancangan antarmuka yang akan dibuat pada sistem. Antarmuka merupakan penghubung antara pengguna dan sistem sehingga pengguna dapat menggunakan sistem dengan mudah. Sistem akan dibuat berbentuk website menggunakan Bahasa HTML dan CSS. Sistem terdiri dari beberapa halaman yaitu halaman data sopir, data alamat, data jarak, hitung demo dan perhitungan MTSP.

##### 4.4.1. Halaman Data Alamat

Halaman data alamat digunakan untuk melihat data alamat dari tiap tiap siswa dan sekolah, seperti yang ditunjukkan pada Gambar 4.14. Pada halaman ini terdapat tabel alamat sekolah, dan tabel alamat masing masing siswa. Keterangan dari halaman data alamat sebagai berikut:

1. Logo yayasan
2. Sidebar menu
3. Judul
4. Tabel alamat sekolah
5. Tabel alamat siswa

Gambar 4.14 Halaman Data Alamat

##### 4.4.2. Halaman Data Sopir

Halaman data Sopir digunakan untuk melihat data dari tiap tiap sopir seperti yang ditunjukkan pada Gambar 4.15. Pada halaman ini terdapat tabel



untuk menunjukkan data sopir seperti yang ditunjukkan pada Gambar 4.13 pada angka 1.

Data Sopir	

**Gambar 4.15 Halaman Data Sopir**

#### 4.4.3. Halaman Data Jarak

Halaman data Sopir digunakan untuk melihat data jarak dari tiap-tiap alamat siswa seperti yang ditunjukkan pada Gambar 4.16. Pada halaman ini terdapat tabel untuk menunjukkan data jarak antar alamat tiap-tiap siswa. Keterangan dari tampilan halaman data jarak adalah sebagai berikut:

1. Tabel data jarak antar siswa
2. *Scroll y*
3. *Scroll x*



Gambar 4.16 Halaman Data Jarak

#### 4.4.4. Halaman Hitung MTSP

Halaman hitung MTSP merupakan tampilan untuk melakukan perhitungan optimasi dengan *Algoritme Particle Swarm Optimization*. Pada halaman ini terdapat form untuk menginputkan parameter PSO sehingga dapat dilakukan perhitungan oleh sistem seperti yang ditunjukkan pada Gambar 4.17. Keterangan dari halaman Hitung MTSP sebagai berikut:

Gambar 4.17 Halaman Hitung MTSP

1. Form hitung MTSP
2. *Input select* jumlah sales
3. *Input select* jumlah iterasi
4. *Input select* ukuran popsize



5. *Input select nilai C1*

6. *Input select nilai C2*

7. Tombol hitung untuk melakukan perhitungan pada sistem





## BAB 5 IMPLEMENTASI

Pada bagian implementasi, Algoritme *PSO* dibuat menggunakan bahasa pemrograman PHP. Pada algoritme ini terdapat beberapa proses, yang dibagi menjadi 5 bagian, yaitu (1) Inisialisasi *PSO*, (2) Memperbarui kecepatan, (3) Memperbarui posisi, (4) Memperbarui *Pbest*, dan (5) Memperbarui *Gbest*.

### 5.1 Implementasi Proses Inisialisasi

Pada proses Inisialisasi *PSO* terdiri dari inisialisasi kecepatan awal, inisialisasi posisi awal partikel dan hitung fitness, inisialisasi *Pbest*, dan inisialisasi *Gbest*.

#### 5.1.1 Implementasi Inisialisasi Posisi Awal

Posisi awal diinisialisasi dengan membuat array sebanyak *popsi*ze. Untuk setiap Index pada array dilakukan pembentukan array dengan panjang jumlah siswa yang di representasikan dengan angka. Kemudian dibentuk 3 segmen, segmen 1 untuk rute keberangkatan siswa, segmen 2 untuk kepulangan siswa kelas 1 dan 2, dan segmen 3 untuk kepulangan siswa kelas 3-6. Masing-masing segmen dilakukan pengacakan kemudian disatukan untuk membuat posisi awal. Implementasi proses inisialisasi posisi awal ditunjukkan pada Kode Sumber 5.1.

Algoritme 1.1: Inisialisasi Partikel Awal	
1	<code>\$partikel = [];</code>
2	<code>for(\$i=0; \$i&lt;\$popsi; \$i++){</code>
3	<code>  shuffle(\$temp_berangkat);</code>
4	<code>  shuffle(\$temp_pulang1);</code>
5	<code>  shuffle(\$temp_pulang2);</code>
6	<code>  \$partikel[\$i] = array_merge(\$temp_berangkat, \$temp_pulang1,</code>
7	<code>  \$temp_pulang2);</code>
8	<code>}</code>

Kode Sumber 5.1 Inisialisasi Partikel Awal

Berikut merupakan penjelasan dari Kode Sumber 5.1:

1. Baris 1 adalah proses inisialisasi array partikel.
2. Baris 2 adalah proses perulangan yang akan dilakukan sebanyak *popsi*ze.
3. Baris 3 adalah proses pengacakan pada segmen 1.
4. Baris 4 adalah proses pengacakan pada segmen 2.
5. Baris 5 adalah proses pengacakan pada segmen 3.
6. Baris 6-7 adalah proses penggabungan kembali segmen 1, 2, dan 3 menjadi 1 array dan disimpan pada array partikel index ke-*i*.
7. Baris 8 adalah akhir dari for.



### 5.1.2 Implementasi Inisialisasi Kecepatan Awal

Kecepatan Awal dan Posisi Awal diinisialisasi sebanyak `popsize`. Pada iterasi pertama ( $t=0$ ) kecepatan awal diinisialisasi dengan nilai  $\emptyset$ . Implementasi proses implementasi inisialisasi kecepatan awal ditunjukkan pada Kode Sumber 5.2.

Algoritme 1.2: Inisialisasi Kecepatan Awal

```

1  $kecepatan = [];
2  for($i=0; $i<$popsize; $i++){
3  $kecepatan[$i]['segmen1'] = [];
4  $kecepatan[$i]['segmen2'] = [];
5  $kecepatan[$i]['segmen3'] = [];
6  }

```

#### Kode Sumber 5.2 Inisialisasi Kecepatan Awal

Berikut merupakan penjelasan dari Kode Sumber 5.2:

1. Baris 1 adalah proses inisialisasi array kecepatan.
2. Baris 2 adalah proses perulangan yang akan dilakukan sebanyak `popsize`.
3. Baris 3 adalah proses inisialisasi kecepatan segmen 1 dengan array kosong.
4. Baris 4 adalah proses inisialisasi kecepatan segmen 2 dengan array kosong.
5. Baris 5 adalah proses inisialisasi kecepatan segmen 3 dengan array kosong.
6. Baris 6 adalah akhir dari for.

### 5.1.3 Implementasi Inisialisasi *Pbest*

pada tahap inisialisasi, nilai *Pbest* akan sama dengan nilai partikel awal, sehingga untuk mendapatkan *Pbest* awal dilakukan perulangan pada setiap partikel dalam partikel dan kemudian nilai dari partikel tersebut disimpan pada setiap *Pbest*. proses implementasi inisialisasi *Pbest* ditunjukkan pada Kode Sumber 5.3.

Algoritme 1.3: Inisialisasi *Pbest*

```

1  $pbest = [];
2  for ($i = 0; $i < sizeof($partikel); $i++) {
3  $pbest[$i] = $partikel[$i];
4  }

```

#### Kode Sumber 5.3 Inisialisasi *Pbest*

Berikut merupakan penjelasan dari Kode Sumber 5.3:

1. Baris 1 adalah proses inisialisasi array *Pbest*.
2. Baris 2 adalah proses perulangan yang akan dilakukan sebanyak `popsize`.
3. Baris 3 adalah proses inisialisasi kecepatan *pbest* pada array dengan index ke-*i* dengan nilai array partikel pada index ke-*i*.
4. Baris 4 adalah akhir dari for.



#### 5.1.4 Implementasi Inisialisasi *Gbest*

*Global Best (Gbest)* merupakan posisi terbaik partikel yang diperoleh dengan membandingkan nilai fitness terbaik dari seluruh *Pbest* partikel. Pada implementasinya *Gbest* didapat dengan melakukan perulangan pada array yang menyimpan nilai dari fitness total *Pbest* kemudian membandingkan nilai masing-masing index, index dari nilai fitness terbers akan disimpan sebagai acuan untuk index yang akan dijadikan *Gbest* pada array *Pbest*. proses implementasi inisialisasi *Gbest* ditunjukkan pada Kode Sumber 5.4.

```

(Algoritme 1.3: Inisialisasi Pbest
1   $max_index = 0;
2   for ($i = 0; $i < sizeof($pbest_totFitness); $i++) {
3     if ($pbest_totFitness[$i] > $pbest_totFitness[$max_index]) {
4       $max_index = $i;
5     }
6   }

```

#### Kode Sumber 5.4 Inisialisasi *Gbest*

Berikut merupakan penjelasan dari Kode Sumber 5.4:

1. Baris 1 adalah proses inisialisasi variabel `max_index` dengan nilai 0.
2. Baris 2 adalah proses perulangan yang akan dilakukan sebanyak `popsize`.
3. Baris 3 adalah proses percabangan `if` dengan syarat array `pbest_totFitness` index ke-*i* lebih besar dari array `pbest_totFitness` pada index dengan nilai dari variabel `max_index`.
4. Baris 4 adalah Proses inisialisasi `max index` dengan nilai *i* apabila syarat `if` sebelumnya terpenuhi.
5. Baris 5 adalah akhir dari `if`.
6. Baris 6 adalah akhir dari `for`.

#### 5.2 Implementasi Proses Memperbarui Kecepatan

Proses memperbarui kecepatan dilakukan menggunakan perulangan sebanyak jumlah partikel. Pada perhitungan proses memperbarui kecepatan menggunakan persamaan Clarc pada persamaan 4.1. Untuk dapat menghitung persamaan tersebut terdapat beberapa fungsi pada tahap implementasi yaitu `multiplication`, `subtraction`, `addition`, `potong`, `move`, dan `oposite`. Proses implemenetasi perbarui kecepatan ditunjukkan pada Kode Sumber 5.5, sedangkan implementasi fungsi ditunjukkan pada Kode Sumber 5.6.

Berikut merupakan penjelasan dari Kode Sumber 5.5:

1. Baris 1 adalah proses perulangan yang akan dilakukan sebanyak `popsize`.
2. Baris 3-4 adalah proses untuk mendapatkan `segmen1` dari partikel dengan memotong array partikel pada index ke-*j* dimulai dari index ke 0 sampai dengan panjang dari `segmen1` kemudian disimpan pada `partikel_segmen1`.



3. Baris 5-6 adalah proses untuk mendapatkan segmen2 dari partikel dengan memotong array partikel pada index ke- $j$  dimulai dari index setelah segmen1 sampai dengan panjang dari segmen2 kemudian disimpan pada partikel\_segmen2.
4. Baris 7-9 adalah proses untuk mendapatkan segmen3 dari partikel dengan memotong array partikel pada index ke- $j$  dimulai dari index setelah segmen2 sampai dengan panjang dari segmen3 kemudian disimpan pada partikel\_segmen1.
5. Baris 11-12 adalah proses untuk mendapatkan segmen1 dari Pbest dengan memotong array Pbest pada index ke- $j$  dimulai dari index ke 0 sampai dengan panjang dari segmen1 kemudian disimpan pada pbest\_segmen1.
6. Baris 13-14 adalah proses untuk mendapatkan segmen2 dari Pbest dengan memotong array Pbest pada index ke- $j$  dimulai dari index setelah segmen1 sampai dengan panjang dari segmen2 kemudian disimpan pada pbest\_segmen2.
7. Baris 15-16 adalah proses untuk mendapatkan segmen3 dari Pbest dengan memotong array Pbest pada index ke- $j$  dimulai dari index setelah segmen2 sampai dengan panjang dari segmen3 kemudian disimpan pada pbest\_segmen3.
8. Baris 18-19 adalah proses untuk mendapatkan segmen1 dari gbest dengan memotong array gbest dimulai dari index ke 0 sampai dengan panjang dari segmen1 kemudian disimpan pada gbest\_segmen1.
9. Baris 20-21 adalah proses untuk mendapatkan segmen2 dari gbest dengan memotong array gbest dimulai dari index setelah segmen1 sampai dengan panjang dari segmen2 kemudian disimpan pada gbest\_segmen2.
10. Baris 22-23 adalah proses untuk mendapatkan segmen3 dari gbest dengan memotong array gbest dimulai dari index setelah segmen2 sampai dengan panjang dari segmen3 kemudian disimpan pada gbest\_segmen3.
11. Baris 25-32 adalah proses perhitungan memperbarui kecepatan dengan persamaan Clarc untuk segmen1.
12. Baris 34-41 adalah proses perhitungan memperbarui kecepatan dengan persamaan Clarc untuk segmen2.
13. Baris 43-50 adalah proses perhitungan memperbarui kecepatan dengan persamaan Clarc untuk segmen3.
14. Baris 52 adalah akhir dari for.



Algoritme 2.1: Memperbarui Kecepatan

```

1  for ($j = 0; $j < $popsize; $j++) {
2
3      $partikel_segmen1 = array_slice($partikel[$j], 0,
4      sizeof($temp_berangkat));
5      $partikel_segmen2 = array_slice($partikel[$j],
6      sizeof($temp_berangkat), sizeof($temp_pulang1));
7      $partikel_segmen3 = array_slice($partikel[$j],
8      sizeof($temp_berangkat) + sizeof($temp_pulang1),
9      sizeof($temp_pulang2));
10
11     $pbest_segmen1 = array_slice($pbest[$j], 0,
12     sizeof($temp_berangkat));
13     $pbest_segmen2 = array_slice($pbest[$j],
14     sizeof($temp_berangkat), sizeof($temp_pulang1));
15     $pbest_segmen3 = array_slice($pbest[$j], sizeof($temp_berangkat)
16     + sizeof($temp_pulang1), sizeof($temp_pulang2));
17
18     $gbest_segmen1 = array_slice($gbest, 0,
19     sizeof($temp_berangkat));
20     $gbest_segmen2 = array_slice($gbest, sizeof($temp_berangkat),
21     sizeof($temp_pulang1));
22     $gbest_segmen3 = array_slice($gbest, sizeof($temp_berangkat) +
23     sizeof($temp_pulang1), sizeof($temp_pulang2));
24
25     $sub1 = $this->subtraction($pbest_segmen1, $pbest_segmen1);
26     $mul1 = $this->multiplication(0.5, $sub1);
27     $mov1 = $this->move($pbest_segmen1, $mul1);
28     $sub2 = $this->subtraction($mov1, $partikel_segmen1);
29     $mul2 = $this->multiplication($c2, $sub2);
30     $mul3 = $this->multiplication($c1, $kecepatan[$j]['segmen1']);
31     $add1 = $this->addition($mul3, $mul2);
32     $kecepatan[$j]['segmen1'] = $add1;
33
34     $sub1 = $this->subtraction($pbest_segmen2, $pbest_segmen2);
35     $mul1 = $this->multiplication(0.5, $sub1);
36     $mov1 = $this->move($pbest_segmen2, $mul1);
37     $sub2 = $this->subtraction($mov1, $partikel_segmen2);
38     $mul2 = $this->multiplication($c2, $sub2);
39     $mul3 = $this->multiplication($c1, $kecepatan[$j]['segmen2']);
40     $add1 = $this->addition($mul3, $mul2);
41     $kecepatan[$j]['segmen2'] = $add1;
42
43     $sub1 = $this->subtraction($pbest_segmen3, $pbest_segmen3);
44     $mul1 = $this->multiplication(0.5, $sub1);
45     $mov1 = $this->move($pbest_segmen3, $mul1);
46     $sub2 = $this->subtraction($mov1, $partikel_segmen3);
47     $mul2 = $this->multiplication($c2, $sub2);
48     $mul3 = $this->multiplication($c1, $kecepatan[$j]['segmen3']);
49     $add1 = $this->addition($mul3, $mul2);
50     $kecepatan[$j]['segmen3'] = $add1;
51 }
52

```

Kode Sumber 5.5 Memperbarui Kecepatan



Algoritme 2.2: Fungsi Operasi PSO

```

1 public function multiplication($c, $velocity){
2     $hasil = [];
3     if ($c > 1) {
4         $k = floor($c);
5         $new_c = $c - $k;
6         $kali = ceil($new_c * sizeof($velocity));
7         $potongan = $this->potong($velocity, $kali);
8         for ($i = 0; $i < $k; $i++) {
9             $hasil = array_merge($hasil, $velocity);
10        }
11        $hasil = array_merge($hasil, $potongan);
12    }
13    } else if ($c > 0) {
14        $nilai = ceil(sizeof($velocity) * $c);
15        $hasil = array_slice($velocity, 0, $nilai);
16        return $hasil;
17    } else if ($c == 0) {
18        $hasil = [];
19        return $hasil;
20    } else {
21        $balik = array_reverse($velocity);
22        return $this->multiplication($c * -1, $balik);
23    }
24 }
25
26 public function subtraction($acuan, $array){
27     $arr = $array;
28     $hasil = [];
29     for ($i = 0; $i < sizeof($arr); $i++) {
30         if ($acuan[$i] != $arr[$i]) {
31             $index = array_search($acuan[$i], $arr);
32             $temp = $arr[$i];
33             $arr[$i] = $arr[$index];
34             $arr[$index] = $temp;
35             array_push($hasil, [$i, $index]);
36         }
37     }
38     return $hasil;
39 }
40
41 public function move($position, $velocity){
42     $hasil = $position;
43     for ($i = 0; $i < sizeof($velocity); $i++) {
44         $temp = $hasil[$velocity[$i][0]];
45         $hasil[$velocity[$i][0]] = $hasil[$velocity[$i][1]];
46         $hasil[$velocity[$i][1]] = $temp;
47     }
48     return $hasil;
49 }
50
51 public function addition($array1, $array2){
52     $hasil = array_merge($array1, $array2);
53     return $hasil;
54 }
55
56 public function opposite($array){
57     $hasil = array_reverse($array);
58     return $hasil;
59 }

```

Kode Sumber 5.6 Fungsi Operasi PSO



Berikut merupakan penjelasan dari Kode Sumber 5.6:

1. Baris 1-24 adalah fungsi multiplication.
2. Baris 26-39 adalah fungsi subtraction.
3. Baris 41 – 49 adalah fungsi move.
4. Baris 51 – 54 adalah fungsi addition.
5. Baris 56-59 adalah fungsi opposite.

### 5.3 Implementasi Proses Memperbarui Posisi

Proses memperbarui posisi dilakukan menggunakan perulangan sebanyak jumlah partikel. Pada proses memperbarui posisi setiap partikel akan diubah posisinya menggunakan fungsi move sesuai dengan kecepatan masing-masing partikel. Implementasi fungsi ditunjukkan pada Kode Sumber 5.7.

```

Algorithm 3: Memperbarui Posisi
1   for ($j = 0; $j < $popsize; $j++) {
2
3       $partikel_segmen1 = array_slice($partikel[$j], 0,
4       sizeof($temp_berangkat));
5       $partikel_segmen2 = array_slice($partikel[$j],
6       sizeof($temp_berangkat), sizeof($temp_pulang1));
7       $partikel_segmen3 = array_slice($partikel[$j],
8       sizeof($temp_berangkat) + sizeof($temp_pulang1),
9       sizeof($temp_pulang2));
10
11      $update1 = $this->move($partikel_segmen1,
12      $kecepatan[$j]['segmen1']);
13      $update2 = $this->move($partikel_segmen2,
14      $kecepatan[$j]['segmen2']);
15      $update3 = $this->move($partikel_segmen3,
16      $kecepatan[$j]['segmen3']);
17
18      $partikel[$j] = array_merge($update1, $update2, $update3);
19  }
  
```

#### Kode Sumber 5.7 Memperbarui Posisi

Berikut merupakan penjelasan dari Kode Sumber 5.7:

1. Baris 1 adalah proses perulangan for yang akan dilakukan sebanyak popsize.
2. Baris 3-4 adalah proses untuk mendapatkan segmen1 dari partikel dengan memotong array partikel pada index ke-j dimulai dari index ke 0 sampai dengan panjang dari segmen1 kemudian disimpan pada partikel\_segmen1.
3. Baris 5-6 adalah proses untuk mendapatkan segmen2 dari partikel dengan memotong array partikel pada index ke-j dimulai dari index setelah segmen1 sampai dengan panjang dari segmen2 kemudian disimpan pada partikel\_segmen2.
4. Baris 7-9 adalah proses untuk mendapatkan segmen3 dari partikel dengan memotong array partikel pada index ke-j dimulai dari index setelah



segmen2 sampai dengan panjang dari segmen3 kemudian disimpan pada partikel\_segmen1.

5. Baris 11-12 adalah proses untuk mengubah posisi berdasarkan kecepatan pada segmen1 dengan memanggil fungsi move dah hasilnya disimpan pada update1.
6. Baris 13-14 adalah proses untuk mengubah posisi berdasarkan kecepatan pada segmen2 dengan memanggil fungsi move dah hasilnya disimpan pada update2.
7. Baris 15-16 adalah proses untuk mengubah posisi berdasarkan kecepatan pada segmen3 dengan memanggil fungsi move dah hasilnya disimpan pada update3.
8. Baris 18 adalah proses menggabungkan hasil dari proses memperbaiki posisi pada update1, update2, dan update3 dengan memanggil fungsi array\_merge kemudian disimpan pada array kecepatan pada index ke-j.
9. Baris 19 adalah akhir dari perulangan for.

#### 5.4 Implementasi Proses Hitung Fitness

Proses Hitung Fitness dapat dilakukan setelah total jarak didapatkan. Total jarak didapatkan dari perhitungan jarak yang akan dilalui oleh masing-masing sopir pada setiap segmen. Implementasi fungsi ditunjukkan pada Kode Sumber 5.8.

Algoritme 4: Hitung Fitness

```

1 public function hitung_fitness($partikel, $sizeof_part1,
2   $sizeof_part2, $sizeof_part3, $kuota){
3   for ($i = 0; $i < sizeof($partikel); $i++) {
4     $data_berangkat = array_slice($partikel[$i], 0,
5       $sizeof_part1);
6     $data_pulang1 = array_slice($partikel[$i], $sizeof_part1,
7       $sizeof_part2);
8     $data_pulang2 = array_slice($partikel[$i], $sizeof_part1 +
9       $sizeof_part2, $sizeof_part3);
10
11     $part1_split = $this->split_array($data_berangkat, $kuota);
12     $part2_split = $this->split_array($data_pulang1, $kuota);
13     $part3_split = $this->split_array($data_pulang2, $kuota);
14
15     $partikel_jarak[$i][0] = $this->total_jarak($part1_split);
16     $partikel_jarak[$i][1] = $this->total_jarak($part2_split);
17     $partikel_jarak[$i][2] = $this->total_jarak($part3_split);
18
19     $partikel_fitness[$i][0] = 100 / $partikel_jarak[$i][0];
20     $partikel_fitness[$i][1] = 100 / $partikel_jarak[$i][1];
21     $partikel_fitness[$i][2] = 100 / $partikel_jarak[$i][2];
22     $partikel_totFitness[$i] = array_sum($partikel_fitness[$i]);
23   }
24   return array('partikel_jarak' => $partikel_jarak,
25     'partikel_fitness' => $partikel_fitness, 'partikel_totFitness'
26     => $partikel_totFitness);
27 }

```

Kode Sumber 5.8 Hitung Fitness





Berikut merupakan penjelasan dari Kode Sumber 5.8:

1. Baris 1 - 2 adalah deklarasi fungsi hitung `fitness`.
2. Baris 3 adalah proses perulangan sebanyak panjang array partikel.
3. Baris 4-5 adalah untuk mendapatkan array `segmen1` dan disimpan pada `data_berangkat`.
4. Baris 6-7 adalah untuk mendapatkan array `segmen2` dan disimpan pada `data_pulang1`.
5. Baris 8-9 adalah untuk mendapatkan array `segmen3` dan disimpan pada `data_pulang2`.
6. Baris 11 adalah untuk mendapatkan siswa untuk masing-masing sopir pada `segmen1` dan disimpan pada `part1_split`.
7. Baris 12 adalah untuk mendapatkan siswa untuk masing-masing sopir pada `segmen2` dan disimpan pada `part2_split`.
8. Baris 13 adalah untuk mendapatkan siswa untuk masing-masing sopir pada `segmen3` dan disimpan pada `part3_split`.
9. Baris 15-17 adalah proses menghitung total jarak pada tiap-tiap segmen.
10. Baris 19-21 adalah proses menghitung `fitness` pada tiap-tiap segmen.
11. Baris 22 adalah proses menjumlahkan `fitness` setiap segmen dan kemudian disimpan pada `partikel_totFitness`.
12. Baris 23 adalah akhir dari perulangan `for`.
13. Baris 24-26 adalah proses mengembalikan nilai berupa array sebagai hasil dari perhitungan total `fitness`.
14. Baris 27 adalah akhir dari fungsi hitung `fitness`.

### 5.5 Implementasi Proses Memperbarui *Pbest*

Nilai pada proses memperbarui *Pbest* didapatkan dengan membandingkan *Pbest* pada iterasi sebelumnya dengan nilai dari posisi yang sekarang pada tiap-tiap partikel. Implementasi fungsi ditunjukkan pada Kode Sumber 5.9.

Algoritme 5: Memperbarui <i>Pbest</i>	
1	<code>for (\$j = 0; \$j &lt; \$popsize; \$j++)</code>
2	<code>  if (\$partikel_totFitness[\$j] &gt; \$pbest_totFitness[\$j]) {</code>
3	<code>    \$pbest[\$j] = \$partikel[\$j];</code>
4	<code>    \$pbest_jarak[\$j] = \$partikel_jarak[\$j];</code>
5	<code>    \$pbest_fitness[\$j] = \$partikel_fitness[\$j];</code>
6	<code>    \$pbest_totFitness[\$j] = \$partikel_totFitness[\$j];</code>
7	<code>  }</code>
8	<code>}</code>

**Kode Sumber 5.9 Memperbarui *Pbest***



Berikut merupakan penjelasan dari Kode Sumber 5.9:

1. Baris 1 adalah proses perulangan for yang akan dilakukan sebanyak popsize.
2. Baris 2 adalah proses percabangan if dengan syarat partikel\_totFitness index ke- $j$  lebih besar dari nilai fitness pbest\_totFitness index ke- $j$ .
3. Baris 3 adalah proses inialisasi nilai pbest index ke- $j$  dengan nilai dari array partikel index ke- $j$  apabila syarat if terpenuhi.
4. Baris 4 adalah proses inialisasi pbest\_jarak index ke- $j$  dengan partikel\_jarak index ke- $j$ .
5. Baris 5 adalah proses inialisasi pbest\_fitness index ke- $j$  dengan partikel\_fitness index ke- $j$ .
6. Baris 6 adalah proses inialisasi pbest\_totFitness index ke- $j$  dengan partikel\_totFitness index ke- $j$ .
7. Baris 7 adalah akhir dari percabangan if.
8. Baris 8 adalah akhir dari perulangan for.

## 5.6 Implementasi Proses Memperbarui Gbest

Tahapan Memperbarui Gbest sama seperti proses Inialisasi Gbest yaitu dengan mencari Pbest dengan nilai total fitness yang tertinggi. Implementasi fungsi ditunjukkan pada Kode Sumber 5.10.

Algoritme 6: Memperbarui Gbest

```

1  $max_index = 0;
2  for ($k = 0; $k < sizeof($pbest_totFitness); $k++) {
3      if ($pbest_totFitness[$k] > $pbest_totFitness[$max_index]) {
4          $max_index = $k;
5      }
6  }
7  $gbest = $pbest[$max_index];
8  $gbest_totFitness = $pbest_totFitness[$max_index];

```

Kode Sumber 5.10 Memperbarui Gbest

Berikut merupakan penjelasan dari Kode Sumber 5.10:

1. Baris 1 adalah deklarasi variabel \$max\_index dengan nilai 0.
2. Baris 2 adalah proses perulangan for yang akan dilakukan sebanyak panjang index dari array pbest\_totFitness.
3. Baris 3 adalah proses percabangan if dengan syarat pbest\_totFitness index ke- $k$  lebih besar dari nilai fitness pbest\_totFitness dengan index max\_index.
4. Baris 4 adalah proses inialisasi variabel max\_index dengan nilai dari variabel k apabila syarat if sebelumnya terpenuhi.
5. Baris 5 adalah akhir dari proses percabangan if.



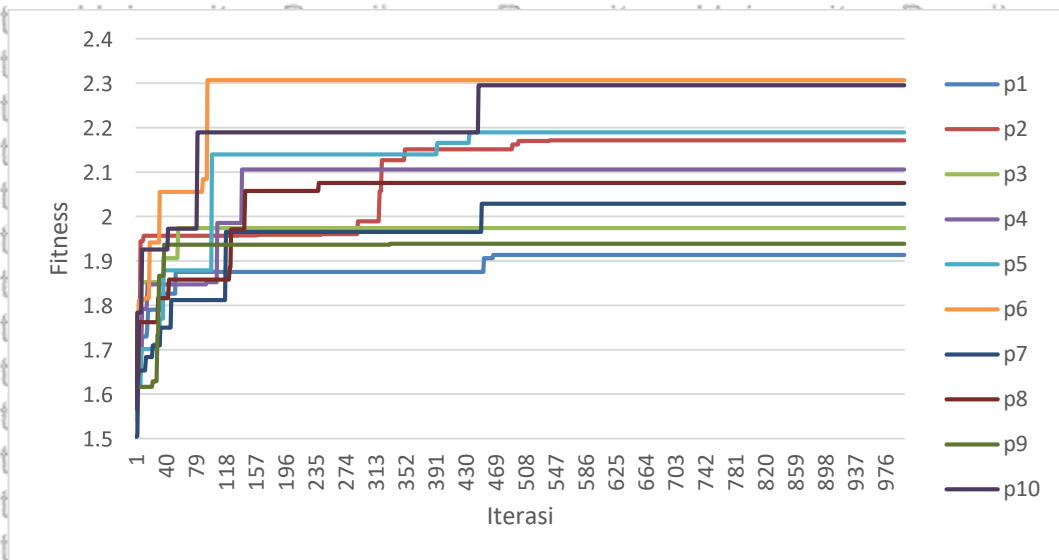
6. Baris 6 adalah akhir dari proses perulangan for.
7. Baris 7 adalah inialisasi gbest dengan array pbest dengan index max\_index.
8. Baris 8 adalah proses inialisasi totFitness dengan array pbest\_totFitness dengan index max\_index.

## BAB 6 PENGUJIAN

Pada bab ini akan membahas hasil dari pengujian dari skenario yang telah dipaparkan pada bab sebelumnya. Pengujian terdiri dari 3 pengujian yaitu pengujian jumlah iterasi, ukuran partikel dan pengujian sistem. Hasil dari masing-masing pengujian akan digunakan sebagai parameter untuk pengujian berikutnya.

### 6.1 Pengujian Konvergensi

Pengujian konvergensi dilakukan untuk mendapatkan jumlah iterasi yang dibutuhkan pada *Algoritme Particle Swarm Optimization* untuk mendapatkan nilai fitness yang konvergen. Pengujian dilakukan sebanyak 10 kali dengan 1000 iterasi pada setiap pengujian. Pada pengujian ini menggunakan parameter jumlah iterasi = 1000, ukuran partikel = 10, jumlah salesman = 8, nilai  $C1 = 1$ , dan nilai  $C2 = 1$ . Hasil dari pengujian konvergensi digambarkan pada Gambar 6.1.



Gambar 6.1 Grafik Hasil Pengujian Konvergensi

Berdasarkan grafik pada Gambar 6.1, dari hasil pengujian konvergensi didapat nilai fitness akhir dengan rata-rata 2.1 dengan range nilai maksimum 2.307 dan nilai minimum 1.913. Dapat diamati bahwa pada iterasi setelah iterasi ke-538 nilai fitness tidak mengalami perubahan atau mengalami konvergen. Dengan pengujian ini didapat bahwa iterasi yang dibutuhkan untuk mendapatkan nilai fitness yang konvergen adalah 538. Nilai ini akan digunakan sebagai parameter untuk pengujian selanjutnya.



## 6.2 Pengujian Ukuran Partikel

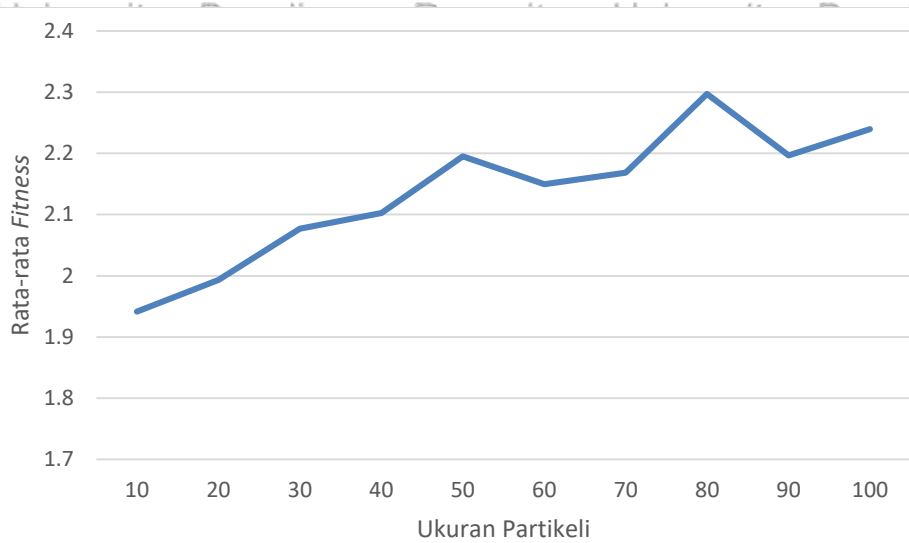
Ukuran partikel menyatakan banyaknya individu atau partikel yang ditampung dalam partikel. Partikel sendiri merupakan representasi solusi yang akan dibangkitkan secara acak pada tahap inisialisasi. Pengujian ukuran partikel dilakukan untuk mengetahui pengaruh besarnya ukuran partikel terhadap nilai *fitness* yang dihasilkan. Pengujian ini dilakukan 10 kali menggunakan partikel 10 - 100 dengan kelipatan 10. Pada pengujian iterasi ini menggunakan jumlah iterasi = 538 yang didapat dari pengujian jumlah iterasi, jumlah *salesman* = 8, nilai  $C1 = 1$  dan  $C2 = 1.5$ . Hasil dari pengujian ukuran partikel dapat dilihat pada Tabel 6.1

Tabel 6.1 Hasil Pengujian Ukuran Partikel

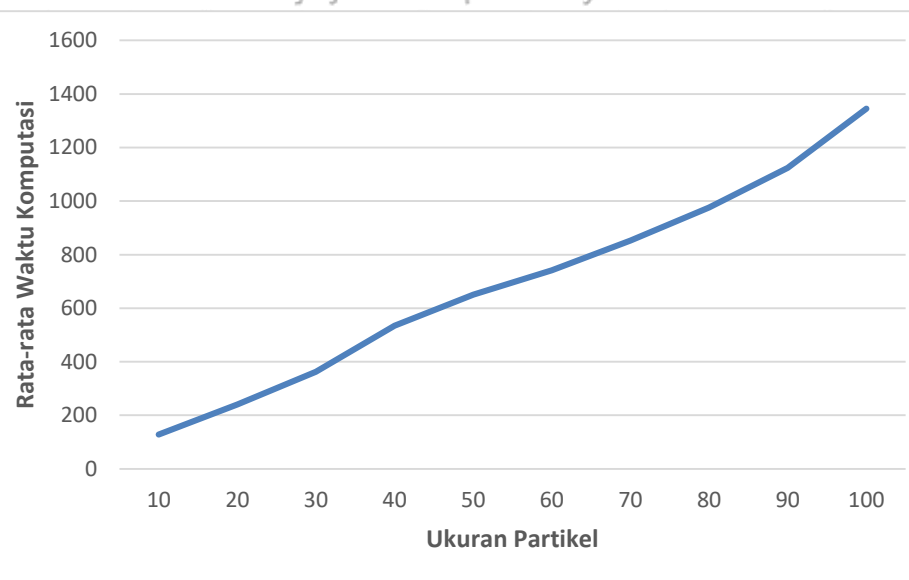
Ukuran partikel	Nilai <i>Fitness</i> -i							Avg <i>Fitness</i>	Max <i>Fitness</i>
	P1	P2	P3	P4	P5	...	P10		
10	2.019	1.976	1.942	1.994	1.959	...	2.034	1.941	2.153
20	2.022	1.846	2.046	2.077	1.969	...	2.070	1.993	2.077
30	2.060	2.068	2.134	1.986	2.006	...	2.274	2.077	2.274
40	2.120	2.051	2.185	2.128	2.063	...	2.069	2.102	2.185
50	2.314	2.148	2.088	2.149	2.236	...	2.088	2.195	2.326
60	2.175	2.162	2.069	2.317	2.105	...	2.136	2.150	2.317
70	2.166	2.152	2.062	2.250	2.082	...	2.316	2.168	2.316
80	2.287	2.379	2.379	2.393	2.309	...	2.238	2.297	2.393
90	2.300	2.170	2.160	2.080	2.159	...	2.189	2.197	2.300
100	2.333	2.138	2.224	2.221	2.201	...	2.321	2.240	2.333

Berdasarkan hasil dari pengujian ukuran partikel pada tabel 6.1, nilai rata-rata *fitness* tertinggi yang didapat 2.297 dengan ukuran partikel sebanyak 80 dan rata-rata terendah adalah 1.941 dengan ukuran partikel sebanyak 10. Nilai *fitness* paling optimum dicapai dengan ukuran partikel 80 pada percobaan ke-4 dengan nilai *fitness* 2.393. Data keseluruhan dari pengujian ukuran populasi dapat dilihat pada Lampiran B. Grafik dari hasil pengujian ukuran populasi dapat dilihat pada Gambar 6.2.

Berdasarkan Gambar 6.2 ukuran partikel cukup mempengaruhi dari *fitness* yang akan dihasilkan. Dari grafik dapat dilihat semakin tinggi nilai ukuran populasi maka nilai rata-rata *fitness* relatif naik. Hal tersebut dikarenakan semakin banyak ukuran partikel maka akan semakin bervariasi partikel yang akan dibentuk sehingga ruang pencarian untuk solusi terbaik akan semakin luas dan besar. Hanya saja semakin banyak partikel yang akan dibentuk maka komputasi yang akan dilakukan sistem akan semakin besar pula sehingga waktu komputasi akan semakin lama. Grafik pengaruh ukuran partikel terhadap waktu komputasi dapat dilihat pada Gambar 6.3.



Gambar 6.2 Grafik Hasil Pengujian Ukuran Partikel



Gambar 6.3 Hasil Pengujian Ukuran Partikel Terhadap Waktu Komputasi

### 6.3 Pengujian Perbandingan Hasil Aktual Dan Hasil Penelitian

Pengujian ukuran partikel dilakukan untuk perbandingan antara hasil penelitian dengan data aktual yang digunakan oleh pihak YPM Daarussalaam. Pada pengujian ini ada aktual jempuran akan dibandingkan dengan solusi dengan nilai fitness terbaik dari hasil perhitungan PSO dengan parameter optimum yang telah didapat yaitu jumlah iterasi = 538, ukuran partikel = 80,  $C1 = 1$ , dan  $C2 = 1.5$ . hasil Pengujian perbandingan hasil aktual dan hasil penelitian dapat dilihat pada Tabel 6.2.



Tabel 6.2 Pengujian Perbandingan Hasil Aktual dengan Hasil Penelitian

Metode	Cost (KM)			Fitness			Total Fitness
	Segmen 1	Segmen 2	Segmen 3	Segmen 1	Segmen 2	Segmen 3	
Hasil Asli	390.588	265.042	140.646	0.256	0.377	0.711	1.344
Hasil Sistem 1	302.607	220.639	62.146	0.330	0.453	1.609	2.393
Hasil Sistem 2	303.298	237.207	67.046	0.330	0.422	1.492	2.243
Hasil Sistem 3	315.498	255.390	72.959	0.317	0.392	1.371	2.079
Hasil Sistem 4	336.158	225.907	66.003	0.297	0.443	1.515	2.255
Hasil Sistem 5	339.499	240.411	68.583	0.295	0.416	1.458	2.169

Pada Tabel 6.4 dapat dilihat bawah hasil dari sistem yang dibuat mendapatkan nilai *fitness* lebih baik dari hasil asli. Nilai *fitness* rata-rata yang didapat adalah 2.2277 dengan nilai *fitness* terbaik 2.3928 yaitu hasil pada pengujian pertama. Total jarak yang didapatkan adalah Hasil Asli = 796.276, Hasil Sistem 1 = 585.392, Hasil Sistem 2 = 607.551, Hasil Sistem 3 = 643.847, Hasil Sistem 4 = 628.068, Hasil Sistem 5 = 648.493. Berdasarkan hasil terbaik pihak yayasan dapat memangkas jarak sebanyak 210.884 Km dari rute yang asli. Rute lengkap dari hasil perhitungan sistem dapat dilihat pada lampiran C.



## BAB 7 PENUTUP

### 7.1 Kesimpulan

Perdasarkan penelitian yang telah dilakukan, didapatkan kesimpulan sebagai berikut:

1. Algoritme PSO merupakan metode metaheuristik yang mengadaptasi sekumpulan ikan atau burung dalam mencari makan. Untuk mendapatkan solusi permasalahan perhitungan Algoritme PSO dilakukan dengan menginisialisasi populasi awal dengan membangkitkan partikel dengan posisi dan kecepatan acak kemudian menghitung nilai fitness masing-masing partikel. Setelah menghitung fitness dilanjutkan dengan membandingkan nilai fitness tersebut dengan nilai terbaik yang pernah partikel tersebut capai dan disimpan pada  $Pbest$ , kemudian  $Pbest$  dari setiap partikel dibandingkan dan hasilnya disimpan pada nilai terbaik global  $Gbest$ . Setelah itu dilakukan perbaikan kecepatan yang menyesuaikan dari nilai  $Pbest$  setiap partikel dengan nilai  $Gbest$ . Langkah tersebut diulang hingga memenuhi suatu kondisi atau telah mencapai batas iterasi maksimal.
2. Berdasarkan pengujian yang dilakukan didapatkan parameter terbaik untuk optimasi rute angkutan sekolah Yayasan Daarussalaam adalah, jumlah iterasi = 538, ukuran populasi = 80, dengan nilai  $C1 = 1$ , nilai  $C2 = 1.5$ , nilai fitness rata-rata yang didapatkan adalah 2.227, dengan nilai fitness terbaik 2.393. Dari hasil tersebut apabila dibandingkan dengan data aktual pihak sekolah dapat memangkas jarak sebanyak 210.884 Km.

### 7.2 Saran

Perdasarkan penelitian yang telah dilakukan, didapatkan saran sebagai berikut:

1. Mencoba melakukan optimasi dengan algoritme lainnya, agar didapatkan hasil yang lebih bervariasi dan dapat dibandingkan hasilnya dengan algoritme pada penelitian sebelumnya yang sudah ada.
2. menambahkan perhitungan waktu tempuh sebagai pertimbangan dalam perhitungan fitness sehingga bisa didapatkan solusi yang lebih baik dengan waktu sebagai pertimbangannya dan sesuai dengan permasalahan sekolah.





## DAFTAR REFERENSI

Akhand, M. A. H., Akter, S., Rahman, S. S. & Rahman, M. M. H., 2012. *Particle Swarm Optimization with Partial Search to*. Kuala Lumpur, IEEE, pp. 118-121.

Anon., 2019. *Data Pokok Pendidikan Dasar dan Menengah Direktorat Jenderal Pendidikan Dasar dan Menengah Kementerian Pendidikan dan Kebudayaan*. [Online] Available at: <http://dapo.dikdasmen.kemdikbud.go.id/sekolah/4842F75F036A6E2144B4>

Azhari, M. K., Bachtiar, F. A. & Cholissodin, I., 2018. Optimasi Travelling Salesman Problem pada Angkutan Sekolah Dengan Algoritma Particle Swarm Intelligence. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, Volume 2, pp. 5691 - 5699.

Karimah, S., Widodo, A. W. & Cholissodin, i., 2017. Optimasi Multiple Salesman Problem Pada Pendistribusian Air Minum Menggunakan Algoritme Genetika. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, pp. 849-850.

mohamed b Abdelhalim, S. E. D. H., 2009. *Particle Swarm Optimization for HW/SW Partitioning*. 1 penyunt. s.l.: InTech Education and Publishing.

Puteri, R. N., Widodo, A. W. & Cholissodin, I., 2017. Optimasi Multiple Salesman Problem Pada Pendistribusian Air Minum Menggunakan Algoritme Particle Swarm Intelligence Optimization (Studi Kasus: UD Tosa Malang). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, pp. 642-648.

Syafiq, M., Cholissodin, I. & Aryadita, H., 2017. Optimasi Penjadwalan Perkuliahan Dengan Menggunakan Hybrid Discrete Particle Swarm Optimization (Studi Kasus: PTIIK Universitas Brawijaya). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, pp. 249-256.



## LAMPIRAN A DATA YANG DIGUNAKAN

### A.1 Data Siswa

ID	Kelas	Latitude	Longitude
1	2	0.528822	117.537875
2	1	0.494197	117.528480
3	1	0.537983	117.506964
4	1	0.534892	117.521051
5	1	0.508257	117.533013
6	1	0.527590	117.536259
7	1	0.525872	117.529689
8	1	0.528781	117.537940
9	1	0.485954	117.545219
10	1	0.535159	117.520699
11	1	0.488332	117.533748
12	1	0.515661	117.539520
13	1	0.508073	117.547329
14	1	0.508764	117.537159
15	1	0.535857	117.512154
16	1	0.511343	117.545122
17	1	0.527967	117.538329
18	1	0.527723	117.519847
19	1	0.488066	117.529142
20	1	0.509405	117.542855
21	1	0.536954	117.511110
22	1	0.527925	117.518190
23	1	0.499030	117.516028
24	1	0.503057	117.537063
25	1	0.491983	117.530415
26	1	0.536312	117.504241
27	1	0.507999	117.538840
28	1	0.509908	117.539026
29	1	0.528060	117.537729
30	1	0.511641	117.540026
31	1	0.517925	117.536540
32	1	0.515689	117.542405
33	1	0.527711	117.521903
34	1	0.491225	117.528049
35	1	0.489440	117.542838
36	1	0.511364	117.545114
37	2	0.513510	117.539582
38	2	0.534861	117.520095
39	2	0.530934	117.504384
40	2	0.528503	117.518560
41	2	0.510749	117.534851
42	2	0.512635	117.534205
43	2	0.484751	117.544265
44	2	0.526818	117.521194



ID	Kelas	Latitude	Longitude
45	2	0.525625	117.528406
46	2	0.536339	117.513234
47	2	0.497499	117.506275
48	2	0.510683	117.536649
49	2	0.517756	117.532241
50	2	0.490374	117.530709
51	2	0.534564	117.521512
52	2	0.484509	117.544406
53	2	0.511779	117.534360
54	2	0.526927	117.526977
55	2	0.492531	117.537059
56	2	0.517061	117.544239
57	2	0.511476	117.539113
58	2	0.506313	117.537772
59	2	0.489797	117.530755
60	2	0.492143	117.534648
61	2	0.528887	117.518656
62	2	0.494524	117.527222
63	2	0.500634	117.495050
64	B	0.486465	117.526865
65	B	0.512794	117.545220
66	B	0.486247	117.532629
67	B	0.527874	117.519925
68	B	0.526223	117.520150
69	B	0.536622	117.512702
70	B	0.526092	117.520051
71	B	0.490974	117.527939
72	A	0.518306	117.542462
73	A	0.526031	117.519704
74	A	0.528333	117.518527
75	A	0.507606	117.529493
76	A	0.516661	117.535506
77	B	0.473531	117.527909
78	A	0.503008	117.537399
79	B	0.534442	117.520839
80	A	0.528290	117.535964
81	A	0.504495	117.535064
82	A	0.510552	117.536309
83	B	0.528405	117.518532
84	B	0.495903	117.553579
85	A	0.509321	117.544969
86	A	0.526576	117.519592
87	B	0.528502	117.534333
88	B	0.512094	117.534187
89	B	0.526600	117.521285
90	B	0.529016	117.517936
91	A	0.534533	117.521264
92	B	0.485877	117.532729
93	B	0.506111	117.538547



ID	Kelas	Latitude	Longitude
94	A	0.512896	117.571779
95	B	0.508748	117.544855
96	B	0.527237	117.537512
97	B	0.524511	117.531743
98	B	0.526818	117.521194
99	B	0.528102	117.524741
100	B	0.528146	117.517817
101	A	0.527913	117.517814
102	A	0.528255	117.525575
103	B	0.485964	117.545177
104	A	0.488798	117.530000
105	B	0.536070	117.512686
106	B	0.538394	117.507271
107	B	0.506776	117.540059
108	A	0.541941	117.510724
109	A	0.500636	117.537030
110	A	0.542175	117.511366
111	3	0.527897	117.519791
112	3	0.492053	117.534624
113	3	0.528019	117.538355
114	3	0.517965	117.534701
115	3	0.490139	117.530788
116	3	0.487619	117.544561
117	3	0.516442	117.535408
118	3	0.518115	117.536861
119	3	0.535147	117.512402
120	3	0.525684	117.520468
121	3	0.52881	117.518302
122	3	0.527145	117.537091
123	3	0.491753	117.5352
124	3	0.488129	117.531145
125	3	0.521097	117.536493
126	3	0.495303	117.528483
127	3	0.513525	117.539708
128	3	0.497768	117.526783
129	3	0.497768	117.526783
130	3	0.499439	117.532018
131	3	0.509246	117.542794
132	3	0.517947	117.536199
133	3	0.499759	117.540473
134	4	0.495215	117.52834
135	4	0.517838	117.536133
136	4	0.506664	117.535482
137	4	0.504549	117.535504
138	4	0.506272	117.537437
139	4	0.528971	117.537604
140	4	0.528045	117.537559
141	4	0.518134	117.542352
142	4	0.530581	117.508276



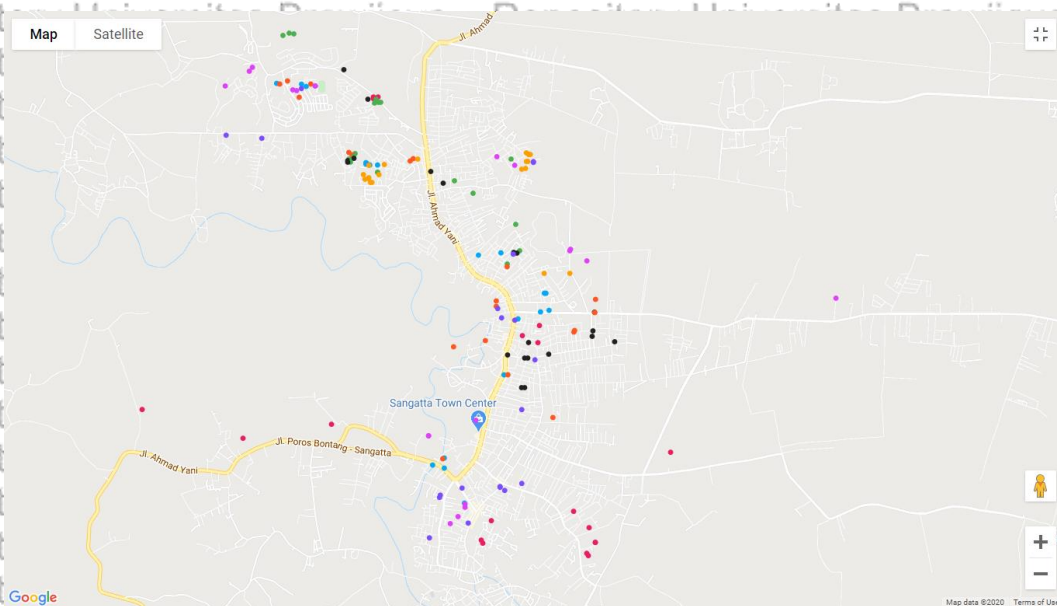
ID	Kelas	Latitude	Longitude
143	4	0.528816	117.537979
144	4	0.508039	117.537848
145	4	0.536576	117.51028
146	4	0.542077	117.511874
147	4	0.528379	117.525005
148	4	0.536662	117.509947
149	4	0.535938	117.511741
150	4	0.5257	117.520358
151	4	0.53661	117.513755
152	4	0.536322	117.514263
153	5	0.536322	117.514263
154	5	0.538147	117.517362
155	5	0.528268	117.518021
156	5	0.527719	117.520252
157	5	0.527572	117.520151
158	5	0.527713	117.52106
159	6	0.534951	117.520691
160	6	0.535163	117.52118

## LAMPIRAN B HASIL PENELITIAN

### B.1 Hasil Pengujian Ukuran Populasi

Ukuran Populasi	Nilai <i>Fitness-i</i>										Avg <i>Fitness</i>	Max <i>Fitness</i>
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10		
10	2.019	1.976	1.942	1.994	1.959	2.153	1.867	1.688	1.783	2.034	1.941	2.153
20	2.022	1.846	2.046	2.077	1.969	2.027	2.060	1.770	2.047	2.070	1.993	2.077
30	2.060	2.068	2.134	1.986	2.006	2.252	1.875	1.932	2.180	2.274	2.077	2.274
40	2.120	2.051	2.185	2.128	2.063	2.112	2.032	2.146	2.119	2.069	2.102	2.185
50	2.314	2.148	2.088	2.149	2.236	2.264	2.326	2.169	2.169	2.088	2.195	2.326
60	2.175	2.162	2.069	2.317	2.105	2.090	2.142	2.128	2.176	2.136	2.150	2.317
70	2.166	2.152	2.062	2.250	2.082	2.164	2.175	2.196	2.120	2.316	2.168	2.316
80	2.287	2.379	2.379	2.436	2.309	2.095	2.239	2.345	2.397	2.238	2.311	2.436
90	2.300	2.170	2.160	2.080	2.159	2.157	2.285	2.394	2.171	2.189	2.207	2.394
100	2.333	2.138	2.224	2.221	2.201	2.211	2.216	2.204	2.327	2.321	2.240	2.333

### B.2 Hasil Rute pada Segmen1



1. Urutan segmen 1 pada driver 1 ditandai dengan titik berwarna biru muda:

Yayasan => 69 => 48 => 57 => 50 => 46 => 62 => 114 => 126 => 2 => 81 => 37  
=> 127 => 148 => 30 => 49 => 111 => 18 => 67 => 156 => 158 => Yayasan

2. Urutan segmen 1 pada driver 2 ditandai dengan titik berwarna ungu:

Yayasan => 130 => 19 => 104 => 59 => 115 => 149 => 129 => 128 => 94 => 56  
=> 141 => 72 => 6 => 87 => 106 => 3 => 26 => 15 => 153 => 152 => Yayasan



- Urutan segmen 1 pada driver 3 ditandai dengan titik berwarna jingga:

Yayasan => 140 => 139 => 143 => 1 => 8 => 29 => 122 => 96 => 32 => 12 => 157  
=> 89 => 150 => 120 => 68 => 70 => 73 => 86 => 33 => 102 => Yayasan

- Urutan segmen 1 pada driver 4 ditandai dengan titik berwarna hijau:

Yayasan => 79 => 91 => 51 => 4 => 155 => 22 => 44 => 98 => 61 => 40 => 83 =>  
7 => 97 => 76 => 125 => 118 => 80 => 108 => 110 => 146 => Yayasan

- Urutan segmen 1 pada driver 5 ditandai dengan titik berwarna merah muda:

Yayasan => 160 => 10 => 159 => 14 => 27 => 43 => 52 => 116 => 35 => 92 => 66  
=> 11 => 23 => 63 => 47 => 77 => 9 => 103 => 84 => 28 => Yayasan

- Urutan segmen 1 pada driver 6 ditandai dengan titik berwarna hitam:

Yayasan => 74 => 101 => 100 => 45 => 31 => 132 => 136 => 138 => 58 => 144  
=> 85 => 13 => 95 => 107 => 78 => 24 => 16 => 54 => 38 => 154 => Yayasan

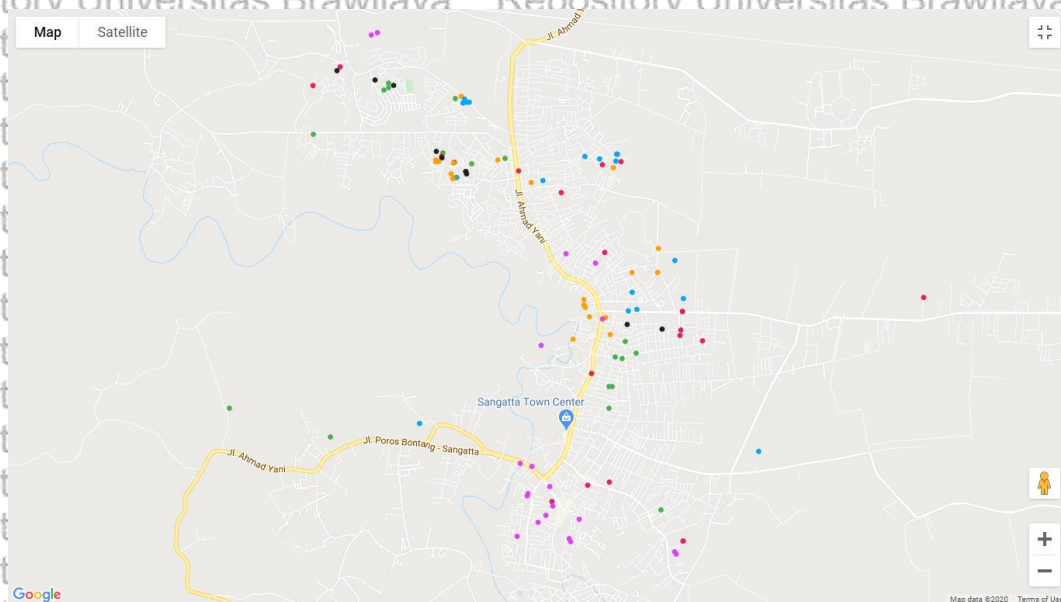
- Urutan segmen 1 pada driver 7 ditandai dengan titik berwarna biru:

Yayasan => 113 => 82 => 25 => 60 => 55 => 123 => 112 => 124 => 17 => 34 =>  
71 => 109 => 93 => 41 => 53 => 135 => 39 => 142 => 105 => 64 => Yayasan

- Urutan segmen 1 pada driver 8 ditandai dengan titik berwarna merah:

Yayasan => 121 => 90 => 119 => 21 => 151 => 117 => 20 => 36 => 99 => 131 =>  
88 => 147 => 133 => 65 => 137 => 42 => 134 => 5 => 75 => 145 => Yayasan

### B.3 Hasil Rute pada Segmen2



- Urutan segmen 2 pada driver 1 ditandai dengan titik berwarna biru muda:

Yayasan => 80 => 83 => 84 => 68 => 7 => 23 => 79 => 91 => 51 => 4 => 87 => 1  
=> 8 => 29 => 56 => 37 => 65 => 16 => 30 => 57 => Yayasan



2. Urutan segmen 2 pada driver 2 ditandai dengan titik berwarna ungu:

Yayasan => 76 => 82 => 25 => 19 => 104 => 64 => 2 => 62 => 43 => 52 => 59 => 11 => 66 => 92 => 71 => 34 => 75 => 49 => 110 => 108 => Yayasan

3. Urutan segmen 2 pada driver 3 ditandai dengan titik berwarna jingga:

Yayasan => 96 => 72 => 32 => 12 => 48 => 14 => 5 => 41 => 53 => 88 => 42 => 86 => 73 => 22 => 101 => 100 => 18 => 99 => 10 => 45 => Yayasan

4. Urutan segmen 2 pada driver 4 ditandai dengan titik berwarna hijau:

Yayasan => 102 => 61 => 70 => 33 => 38 => 63 => 47 => 77 => 35 => 109 => 27 => 107 => 78 => 24 => 93 => 58 => 39 => 15 => 105 => 69 => Yayasan

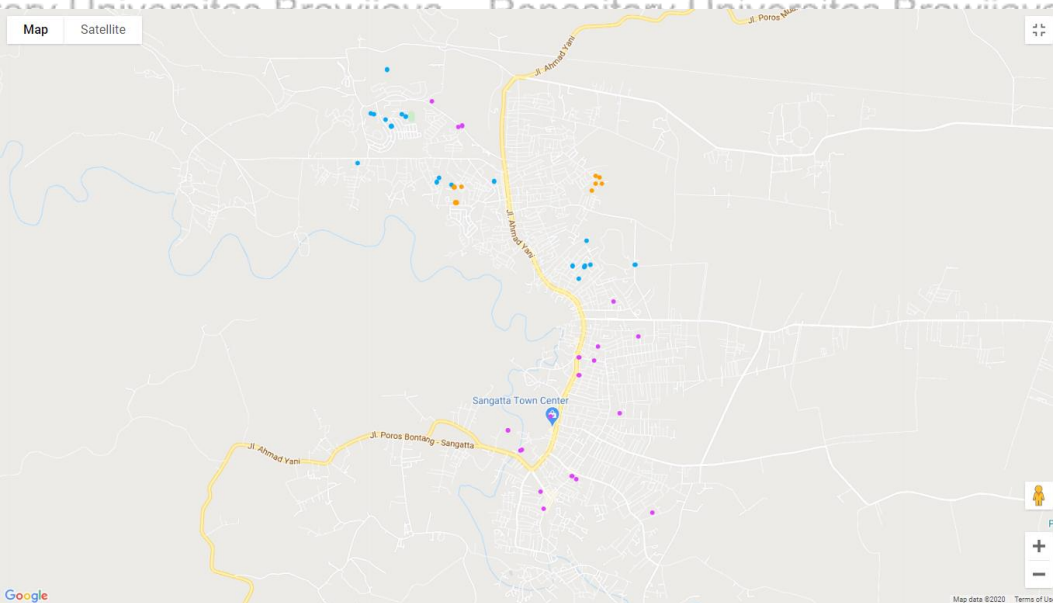
5. Urutan segmen 2 pada driver 5 ditandai dengan titik berwarna merah muda:

Yayasan => 6 => 17 => 67 => 54 => 97 => 31 => 81 => 50 => 60 => 55 => 103 => 9 => 40 => 95 => 13 => 94 => 85 => 36 => 26 => 106 => Yayasan

6. Urutan segmen 2 pada driver 6 ditandai dengan titik berwarna hitam:

Yayasan => 21 => 98 => 90 => 3 => 74 => 20 => 44 => 46 => 28 => 89 => Yayasan

#### B.4 Hasil Rute pada Segmen2



Keterangan dari gambar diatas:

1. Urutan segmen 2 pada driver 1 ditandai dengan titik berwarna merah muda:

Yayasan => 153 => 152 => 111 => 125 => 135 => 132 => 118 => 141 => 117 => 147 => 114 => 121 => 155 => 142 => 148 => 149 => 145 => 119 => 151 => 146 => Yayasan

2. Urutan segmen 2 pada driver 2 ditandai dengan titik berwarna ungu:





Yayasan => 130 => 134 => 126 => 129 => 128 => 116 => 123 => 112 => 115 =>  
124 => 133 => 137 => 138 => 136 => 144 => 131 => 127 => 159 => 160 => 154  
=> Yayasan

3. Urutan segmen 2 pada driver 3 ditandai dengan titik berwarna jingga:

Yayasan => 122 => 140 => 139 => 143 => 113 => 156 => 150 => 158 => 120 =>  
157 => Yayasan