Optimasi *Travelling Salesman Problem* Pada Angkutan Sekolah Dengan Algoritme *Particle Swarm Optimization*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
M. Khusnul Azhari
NIM: 145150201111121

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

OPTIMASI *TRAVELLING SALESMAN PROBLEM* PADA ANGKUTAN SEKOLAH
DENGAN ALGORITME *PARTICLE SWARM OPTIMIZATION*
SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
M. Khusnul Azhari
NIM: 145150201111121

Skripsi ini telah diuji dan dinyatakan lulus pada
05 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Imam Cholissodin, S.Si, M.Kom	Dr.Eng., Fitra A. Bachtiar S.T, M.Eng
NIK: 201201 850719 1 001	NIK. 201201 840628 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

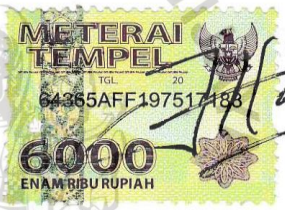
Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

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 pustaka.

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, 05 Juli 2018



M. Khusnul Azhari
NIM: 145150201111121

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah SWT atas anugerah serta limpahan rahmat-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul **"**Optimasi *Travelling Salesman Problem* Pada Angkutan Sekolah Dengan Algoritme *Particle Swarm Optimization***"**. Skripsi ini disusun sebagai syarat memperoleh gelar sarjana pada Program Studi Teknik Informatika, Jurusan Teknik Informatika, Fakultas Ilmu Komputer Universitas Brawijaya.

Dalam proses penyelesaian skripsi ini penulis mendapatkan banyak bantuan, baik bantuan moral maupun materiil dari berbagai pihak. Oleh karena itu, penulis mengucapkan banyak terimakasih kepada:

1. Kedua Orang Tua yaitu Bapak saya Drs. Syaiful Mujahid dan Ibu saya Maimonah beserta keluarga besar yang telah mendukung penulis dengan segala usahanya, mulai dari doa, materi, dukungan moral, semangat hidup, dan tauladan yang semata-mata untuk keberhasilan penulis.
2. Imam Cholissodin, S.Si, M.Kom selaku Pembimbing I yang telah memberikan bimbingan, arahan, ilmu, dan masukan dalam penyelesaian skripsi ini.
3. Fitra Abdurrahman Bachtiar, Dr.Eng., S.T, M.Eng, selaku Pembimbing II yang juga telah memberikan bimbingan, arahan, ilmu, dan masukan dalam penyelesaian skripsi ini.
4. Pihak sekolah MI Salafiyah Kasim Blitar yang sudah mengijinkan penulis untuk melakukan penelitian di sana.
5. Kholifa'ul Khoirin dkk. Serta teman-teman Program Studi Teknik Informatika/Ilmu Komputer yang selalu memberikan semangat, dukungan, dan kebersamaan selama Penulis menempuh pendidikan di Fakultas Ilmu Komputer Universitas Brawijaya.
6. Teman seperjuangan sejak semester satu, Jeriko, Hafid, Aldo, Rama dan Ardhan yang selalu menemami penulis dalam suka maupun duka selama empat tahun di Malang.
7. Teman seatap pada kontrakan BCT, Jeriko, Wildan, Aal, Kevin, Komang serta teman lain yang mampir di kontrakan dengan selalu berbagi canda dan tawa serta semangat menyelesaikan skripsi penulis ini.
8. Semua pihak yang tidak dapat disebutkan satu per satu, yang telah membantu dan terlibat baik secara langsung maupun tidak langsung dalam penulisan skripsi ini.

Penulis menyadari bahwa skripsi ini tidak lepas dari kesalahan dan kekurangan. Oleh karena itu, Penulis bersedia menerima kritik dan saran yang membangun untuk memperbaiki diri. Penulis berharap semoga skripsi ini dapat memberi manfaat.

Malang, 05 Juli 2018

Penulis

khusnulazhari@gmail.com

ABSTRAK

M. Khusnul Azhari, Optimasi *Travelling Salesman Problem* Pada Angkutan Sekolah Dengan Algoritme *Particle Swarm Optimization*

Pembimbing: Imam Cholissodin, S.Si, M.Kom dan Dr.Eng., Fitra A. Bachtiar S.T, M.Eng

Dewasa ini, penerapan angkutan sekolah sudah banyak dilakukan baik dari pihak sekolah, swasta dan bahkan pemerintah. Salah satunya adalah sekolah MI Salafiyah Kasim. Meskipun sistem angkutan sekolah ini sudah diterapkan bertahun-tahun, masih terdapat berbagai kendala seperti siswa-siswi yang diantar tidak selalu sama setiap harinya, keterlambatan supir dalam mengantar sampai tujuan, supir sekolah yang selalu mengedepankan pengalaman pribadi hingga dana operasional angkutan sekolah yang masih belum stabil. Untuk mengatasi permasalahan tersebut, penulis menggunakan Algoritme Particle Swarm Optimization dalam melakukan optimasi guna mendapatkan urutan pengantaran siswa dengan rute terpendek yang bisa dilalui oleh supir sekolah. Hasil dari penelitian ini membandingkan data sampel aktual satu hari pengantaran dengan sistem yang telah dirancang. Dari lima kali percobaan yang diaplikasikan pada masing-masing kloter, tiga diantaranya sistem mampu menghasilkan rekomendasi rute yang lebih baik dari pada yang biasa dilalui oleh supir. Setelah ditinjau secara keseluruhan, sistem dinilai dapat bekerja dengan baik dan menghasilkan solusi yang cukup optimal.

Kata Kunci : *Angkutan Sekolah, Optimasi, Travelling Salesman Problem, Particle Swarm Optimization*.

ABSTRACT

M. Khusnul Azhari, Traveling Salesman Problem On School Transport With Particle Swarm Optimization Algorithm

Advisor: Imam Cholissodin, S.Si, M.Kom and Dr.Eng., Fitra A. Bachtiar S.T, M.Eng

Currently, The implementation of school transport has been done a lot of school, private and even the government. One of them is MI Salafiyah Kasim school. Although the school transport system has been implemented for years, there are obstacles such as students who are delivered not always the same every day, the driver delays in delivering to the destination, the school driver who always prioritizes personal experience and fund of transportation operations that are still unstable . To overcome these problems, the authors use the Particle Swarm Optimization Algorithm in the optimization to get the order of delivery of students with the shortest route that can be passed by the school driver. The results of this study compared actual sample data one day delivery with the system that has been designed. Of the five experiments applied to each kloter, three of them are able to produce a better route recommendation than the usual driver. Once reviewed overall, the system is considered to work well and produce a fairly optimal solution.

Keywords: *School Transportation, Optimization, Travelman Salesman Problem, Particle Swarm Optimization*

DAFTAR ISI

PENGESAHAN ii
KATA PENGANTAR iv
ABSTRAK v
ABSTRACT vi
DAFTAR ISI vii
DAFTAR TABEL x
DAFTAR GAMBAR xii
DAFTAR PERSAMAAN xiii
DAFTAR KODE PROGRAM xiv
DAFTAR LAMPIRAN xv
BAB 1 PENDAHULUAN 1
1.1 Latar belakang 1
1.2 Rumusan masalah 2
1.3 Tujuan 2
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 MI Salafiyah Kasim 8
2.3 Graf 9
2.4 Optimasi 11
2.5 *Travelling Salesman Problem* (TSP) 11
2.6 *Particle Swarm Optimization* (PSO) 11
BAB 3 METODOLOGI 14
3.1 Studi Pustaka 14
3.2 Pengumpulan Data 15
3.3 Analisis Kebutuhan 15
3.3.1 Deskripsi Umum Sistem 15
3.3.2 Spesifikasi Kebutuhan Sistem 15

3.4 Perancangan Sistem 16
3.5 Implementasi 16
3.6 Pengujian dan Analisis 16
3.7 Penarikan Kesimpulan 16
BAB 4 PERANCANGAN 17
4.1 Formulasi Permasalahan 17
4.2 Proses Penyelesaian masalah dengan Algoritma PSO 17
4.2.1 *Input* Data 19
4.2.2 Inisialisasi Populasi Awal 19
4.2.3 *Update* Kecepatan 23
4.2.4 *Update* Posisi 24
4.2.5 *Update Pbest* 28
4.2.6 *Update Gbest* 29
4.3 Perhitungan Manual 31
4.3.1 Inisialisasi Populasi Awal 31
4.3.2 *Update* Kecepatan 33
4.3.3 *Update* Posisi 34
4.3.4 *Update Pbest* dan *Gbest* 36
4.4 Perancangan Antar Muka 38
4.4.1 Halaman Depan 38
4.4.2 Halaman Data 39
4.4.3 Halaman Proses 40
4.4.4 Halaman Hasil 41
4.5 Perancangan Pengujian 42
4.5.1 Pengujian Jumlah Partikel 42
4.5.2 Pengujian Kombinasi *Wmin* dan *Wmax* 42
4.5.3 Pengujian Koefisien Akselerasi 43
4.5.4 Pengujian Konvergensi 44
BAB 5 IMPLEMENTASI 45
5.1 Implementasi Program 45
5.1.1 *Import* Data 45
5.1.2 Inisialisasi Partikel 46

5.1.3 Inisialisasi Kecepatan Awal 47
5.1.4 Hitung *Fitness* 48
5.1.5 *Update* Kecepatan 49
5.1.6 *Update* Posisi 49
5.1.7 *Repair* 50
5.1.8 *Update Pbest* 53
5.1.9 *Update Gbest* 54
5.2 Implementasi Antarmuka 56
5.2.1 Halaman Depan 56
5.2.2 Halaman Data 56
5.2.3 Halaman Proses 57
5.2.4 Halaman Hasil 57
BAB 6 PENGUJIAN DAN ANALISIS 59
6.1 Pengujian dan Analisis Jumlah Partikel 59
6.1.1 Hasil Pengujian Jumlah Partikel 59
6.1.2 Analisis Hasil Pengujian Jumlah Partikel 60
6.2 Pengujian Kombinasi *Wmin* dan *Wmax* 61
6.2.1 Hasil Pengujian Kombinasi *Wmin* dan *Wmax* 62
6.2.2 Analisis Hasil Pengujian Kombinasi *Wmin* dan *Wmax* 63
6.3 Pengujian Koefisien Akselerasi 64
6.3.1 Hasil Pengujian Koefisiean Akselerasi 64
6.3.2 Analisis Hasil Pengujian Koefisien Akselerasi 66
6.4 Pengujian Konvergensi 67
6.4.1 Hasil Pengujian 67
6.4.2 Analisis Hasil Pengujian Konvergensi 68
6.5 Analisis Global Hasil Pengujian 69
BAB 7 PENUTUP 72
7.1 KESIMPULAN 72
7.2 SARAN 72
DAFTAR PUSTAKA 74

DAFTAR TABEL

Tabel 2.1 Kajian Pustaka 5
Tabel 4.1 Matriks Jarak Kloter Pertama (Km) 17
Tabel 4.2 Inisialisasi Kecepatan 19
Tabel 4.3 Data ID Siswa 31
Tabel 4.4 Parameter PSO 31
Tabel 4.5 Inisialisasi Kecepatan 31
Tabel 4.6 Konversi ID Permutasi 32
Tabel 4.7 Pembentukan Partikel Kloter Pertama 32
Tabel 4.8 Pembentukan Partikel Kloter Kedua 32
Tabel 4.9 Nilai *Fitness* Partikel Kloter Pertama 33
Tabel 4.10 Nilai *Fitness* Partikel Kloter Kedua 33
Tabel 4.11 *Pbest* dan *Gbest* Partikel Kloter Pertama 33
Tabel 4.12 *Pbest* dan *Gbest* Partikel Kloter kedua 33
Tabel 4.13 *Update* Kecepatan Kloter Pertama 34
Tabel 4.14 *Update* Kecepatan Kloter Kedua 34
Tabel 4.15 *Update* posisi Kloter Pertama 35
Tabel 4.16 *Update* posisi Kloter Kedua 35
Tabel 4.17 *Repair* Kloter Pertama 35
Tabel 4.18 *Repair* Kloter kedua 35
Tabel 4.19 Perbaruan Nilai *Fitness* Kloter Pertama 36
Tabel 4.20 Perbaruan Nilai *Fitness* Kloter Kedua 36
Tabel 4.21 Perbandingan Nilai *Fitness* Kloter Pertama 36
Tabel 4.22 Perbandingan Nilai *Fitness* Kloter Kedua 37
Tabel 4.23 *Pbest* dan *Gbest* Iterasi 2 Kloter Pertama 37
Tabel 4.24 *Pbest* dan *Gbest* Iterasi 2 Kloter Kedua 37
Tabel 4.25 Hasil Optimasi Kloter Pertama 37
Tabel 4.26 Hasil Optimasi Kloter Kedua 38
Tabel 4.27 Perancangan Pengujian Jumlah Partikel 42
Tabel 4.28 Perancangan Pengujian Kombinasi *Wmin* dan *Wmax* 43
Tabel 4.29 Perancangan Pengujian Koefisien Akselerasi 43

Tabel 4.30 Perancangan Pengujian Koefisien Akselerasi 44
Tabel 6.1 Hasil Pengujian Jumlah Partikel kloter Pertama 59
Tabel 6.2 Hasil Pengujian Jumlah Partikel kloter Kedua 59
Tabel 6.3 Hasil Gabungan Nilai Rata-rata *Fitness* Setiap Kloter 60
Tabel 6.4 Hasil Pengujian Kombinasi *Wmin* dan *Wmax* Kloter Pertama 62
Tabel 6.5 Hasil Pengujian Kombinasi *Wmin* dan *Wmax* Kloter Kedua 62
Tabel 6.6 Hasil Gabungan Nilai Rata-rata *Fitness* Setiap Kloter 63
Tabel 6.7 Hasil Pengujian Koefisien Akselerasi Kloter Pertama 64
Tabel 6.8 Hasil Pengujian Koefisien Akselerasi Kloter Kedua 65
Tabel 6.9 Hasil Gabungan Nilai Rata-rata *Fitness* Setiap Kloter 65
Tabel 6.10 Hasil Pengujian Konvergensi Kloter Pertama 67
Tabel 6.11 Hasil Pengujian Konvergensi Kloter Kedua 67
Tabel 6.12 Data Aktual Rute Pengantaran 69
Tabel 6.13 Hasil Rekomendasi Sistem 70
Tabel 6.14 Hasil Selisih 71

DAFTAR GAMBAR

Gambar 2.1 MI Salafiyah Kasim 8
Gambar 2.2 (a) Mobil sekolah tampak depan (b) Mobil sekolah tampak dalam ... 9
Gambar 2.3 (a) Graf sederhana, (b) Graf ganda dan (c) Graf semu..... 10
Gambar 2.4 (a) Graf berarah, (b) Graf tak berarah..... 10
Gambar 3.1 Metodologi Penelitian..... 14
Gambar 3.2 Perancangan Umum Sistem 16
Gambar 4.1 Diagram Alir Algoritma PSO untuk Optimasi Angkutan Sekolah 18
Gambar 4.2 Diagram Alir Inisialisasi Partikel 20
Gambar 4.3 Diagram Alir Hitung *Fitness*..... 22
Gambar 4.4 Diagram Alir *Update* Kecepatan..... 23
Gambar 4.5 Diagram Alir *Update* Posisi..... 24
Gambar 4.6 Diagram Alir Repair 25
Gambar 4.7 Diagram Alir *Update Pbest*..... 28
Gambar 4.8 Diagram Alir *Update Gbest* 29
Gambar 4.9 Perancangan Halaman Depan 38
Gambar 4.10 Halaman Data..... 39
Gambar 4.11 Halaman Proses..... 40
Gambar 4.12 Halaman Hasil..... 41
Gambar 5.1 Implementasi Halaman Depan..... 56
Gambar 5.2 Implementasi Halaman Data..... 57
Gambar 5.3 Halaman Proses..... 57
Gambar 5.4 Halaman Hasil..... 58
Gambar 6.1 Grafik Hasil Rata-rata *Fitness* 60
Gambar 6.2 Gafik Hasil Waktu Komputasi..... 61
Gambar 6.3 Grafik Hasil Pengujian Kombinasi *Wmin* dan *Wmax* 64
Gambar 6.4 Grafik Hasil Pengujian Koefisien Akselerasi 66
Gambar 6.5 Grafik Hasil Pengujian Konvergensi Kloter Pertama 68
Gambar 6.6 Grafik Hasil Pengujian Konvergensi Kloter Kedua..... 68

DAFTAR PERSAMAAN

Persamaan (2.1) Permutasi TSP 11
Persamaan (2.2) *Update* Kecepatan 12
Persamaan (2.3) *Update* Bobot Inersia 13
Persamaan (2.4) *Update* Posisi 13
Persamaan (2.5) Menentukan *Fitness* 13

DAFTAR KODE PROGRAM

Kode Program 5.1 *Import* Data 46
Kode Program 5.2 Inisialisasi Partikel 47
Kode Program 5.3 Inisialisasi Kecepatan Awal..... 48
Kode Program 5.4 Hitung *Fitness*..... 48
Kode Program 5.5 *Update* Kecepatan..... 49
Kode Program 5.6 *Update* Posisi..... 50
Kode Program 5.7 *Repair* 51
Kode Program 5.8 *Update Pbest*..... 54
Kode Program 5.9 *Update Gbest* 55

DAFTAR LAMPIRAN

LAMPIRAN A HASIL OBSERVASI 76
A.1 Data Siswa-siswi 76
A.2 Persebaran Alamat Siswa 78
A.3 Data Matriks Jarak Siswa-siswi (Km) 80
LAMPIRAN B PERHITUNGAN MANUAL 82
B.1 Data ID Siswa Perhitungan Manual 82
B.2 Konversi ID Permutasi 82
B.3 Pembentukan Partikel 82
B.4 Contoh Perhitungan Total Jarak 82
B.5 Inisialisasi *Pbest* dan *Gbest* 83
B.6 *Update* Kecepatan 83
B.7 *Update* Posisi 83
B.8 *Repair* 84
B.9 *Fitness* 84
B.10 Perbandingan Nilai *Fitness* 85
B.11 *Update Pbest* dan *Gbest* untuk Iterasi 2 85
B.12 Hasil Optimasi 86
LAMPIRAN C HASIL PENGUJIAN SISTEM 87
C.1 Hasil Pengujian Jumlah Partikel 87
C.2 Hasil Pengujian Kombinasi *Wmin* dan *Wmax* 88
C.3 Hasil Pengujian Koefisien Akselerasi 90
C.4 Data Hasil Rekomendasi Sistem 92

BAB 1 PENDAHULUAN

1.1 Latar belakang

Angkutan sekolah merupakan sarana pelayanan transportasi yang diperuntukkan mengantar-jemput siswa-siswi sekolah demi kelancaran proses belajar mengajar (Kariono, 2011). Dengan adanya angkutan sekolah ini tentunya dapat menekan berbagai permasalahan seperti tingkat kemacetan, keamanan lalu lintas hingga kenyamanan siswa untuk menuju sekolah. Permasalahan tersebut memang didasari karena usia pelajar terbilang belum cukup memenuhi persyaratan membawa kendaraan bermotor pribadi untuk pergi ke sekolah. Berbagai pelanggaran dan bahkan kecelakaan lalu lintas disebabkan oleh pelajar. Menurut data Korlantas Polri di 2010-2015, ada 157 ribu anak di bawah umur yang menjadi korban kecelakaan. Sebaliknya, ada setidaknya 25 ribu anak di bawah umur jadi pelaku kecelakaan (Beraunews, 2017).

Dengan adanya sistem angkutan sekolah, maka orang tua siswa akan merasa tenang karena anaknya dapat masuk dan pulang sekolah dengan aman dan tentunya tepat waktu. Oleh karena itu penerapan angkutan sekolah sendiri sudah banyak dilakukan baik murni dari pihak sekolah, swasta dan bahkan pemerintah. Pada saat ini, banyak pemerintah kota seperti Jakarta, Bandung, Depansar, Malang, Tulungagung yang sudah menyediakan bus sekolah gratis bagi pelajar untuk kemudahan dalam menuju atau pulang dari sekolah (Kompasiana, 2017). Namun terdapat kendala seperti keminatan pelajar akan bus sekolah gratis yang terbilang masih kurang. Hal ini mungkin disebabkan karena faktor efisiensi dari kinerja bus ataupun gengsi dari pelajar antar sekolah itu sendiri.

MI Salafiyah Kasim merupakan salah satu lembaga pendidikan dasar yang terletak di Jl. KH. Dimyati Kasim Ploso, kecamatan Selopuro, kabupaten Blitar. Sekolah ini menerapkan sistem pengantaran bagi siswa-siswinya yang sudah berjalan kurang lebih sejak 3 tahun yang lalu. Terdapat dua kloter pengantaran pada setiap harinya yakni kelas 1 dan 2 pada pukul 11.30 untuk kloter pertama dan kelas 3 sampai 6 pada pukul 13.00 untuk kloter kedua. Kendala yang dialami adalah supir selalu mengedepankan pengalaman pribadinya dalam mengantar. Hal ini tentunya dapat dirasa kurang efisien jika pengalaman supir ternyata bukan merupakan suatu pilihan terbaik yang bisa diambil ketika menjalankan tugasnya. Selain itu, sering kali supir terlambat sampai ke sekolah ketika akan mengantar kloter yang kedua. Sehingga menyebabkan siswa menunggu lama. Kendala lain yang dialami adalah siswa-siswi yang diantar setiap harinya tidak selalu sama, sehingga menyebabkan supir agak kesusahan untuk menentukan rute terbaik dan siapa siswa yang akan diantar terlebih dahulu.

Dengan perbandingan besarnya manfaat dan berbagai kendala yang dialami pada angkutan sekolah, maka diperlukan adanya suatu optimasi untuk kelancaran pada penerapan angkutan sekolah ini. Salah satu optimasi yang bisa dilakukan adalah optimasi jarak dengan menentukan rute terpendek yang bisa dilalui. Dengan asumsi semakin pendek jarak yang dilalui, maka dapat menekan biaya

kendaraan dan juga waktu pengantaran. *Travelling Salesman Problem* (TSP) merupakan metode yang berhubungan dengan jarak dan biaya yang mana dapat mencari rute optimal yang dapat dilalui dengan biaya termurah (Mahmudy, 2015). Sedangkan *Particle Swarm Optimization* (PSO) merupakan algoritme optimasi yang mana meniru tingkah laku serangga dalam proses pencarian solusi yang diharapkan.

Sebelumnya pernah dilakukan penelitian oleh (Hermawan, Hidayat, & Setiawan, 2017), mengenai sistem optimasi rute kuliner di Malang, yang menggunakan optimasi *Travelling Salesman Problem* (TSP) Dengan algoritme *Bee Colony* (lebah). Dengan merepresentasikan titik jarak dengan pasukan lebah, dilakukan perhitungan dengan fase *employee bee*, *scout bee* dan *onloker bee*. Penelitian serupa juga pernah dilakukan oleh (Karimah, Widodo, & Cholissodin, 2017), dengan judul optimasi *multiple travelling salesman problem* (M-TSP) pada pendistribusian air minum menggunakan algoritme genetika. Susunan rute jarak antar titik direpresentasikan dengan kromosom yang nantinya akan dilakukan pencarian solusi melalui fase *crossover* dan *mutasi*.

Berdasarkan pertimbangan pada penelitian sebelumnya yang telah disebutkan di atas, penulis berfikiran untuk melakukan penelitian ini dengan menerapkan optimasi pada metode *Travelling Salesman Problem* (TSP) dengan algoritme *Particle Swarm Optimization* (PSO) guna mendapat solusi optimum yang diharapkan. *Particle Swarm Optimization* (PSO) dikenalkan oleh Kennedy dan Eberhard pada tahun 1995. Algoritme ini merupakan salah satu teknik optimasi berbasis populasi yang terdapat banyak partikel sebagai penyusunnya (Kurniawan, 2010). Kelebihan dari algoritme *Particle Swarm Optimization* (PSO) di antaranya adalah memiliki konsep sederhana, dapat diimplementasikan dengan mudah dan dinilai lebih efisien dalam hal perhitungan apabila dibandingkan dengan teknik optimasi heuristik lainnya (Maickel, et al., 2009). Oleh karena itu penulis mengangkat judul "Optimasi *Travelling Salesman Problem* Pada Angkutan Sekolah Dengan Algoritme *Particle Swarm Optimization*" pada penelitian ini.

1.2 Rumusan masalah

Berdasarkan pemaparan latar belakang di atas, maka dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana mengimplementasikan optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah?
2. Bagaimana kualitas hasil optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah?

1.3 Tujuan

Tujuan dilakukan penelitian ini adalah sebagai berikut:

1. Mengimplementasikan optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

2. Mengetahui kualitas hasil optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

1.4 Manfaat

Manfaat penelitian yang bisa didapat antara lain sebagai berikut:

1. Hasil dari penelitian ini dapat menjadi kajian dan referensi bagi sistem angkutan sekolah.
2. Dapat mengoptimalkan kinerja supir dalam menjalankan tugasnya.
3. Meningkatkan kepuasan siswa dengan kenyamanan optimasi waktu antar jemput dan biaya yang dikeluarkan.

1.5 Batasan masalah

Agar penelitian ini dapat dilakukan secara optimal, maka diperlukan Adanya batasan masalah. Diantaranya adalah sebagai berikut:

1. Angkutan sekolah yang diteliti adalah angkutan pribadi dari pihak sekolah.
2. Data sekolah yang dipakai adalah MI Salafiyah Kasim Blitar.
3. Lokasi antar jemput siswa diasumsikan berada di tepi jalan.
4. Penelitian tidak mempertimbangkan kondisi *real* jalan yang dilalui angkutan sekolah.
5. Penelitian berfokus pada optimasi jarak dikarenakan sudah mewakili biaya bahan bakar dan waktu pengantaran dengan asumsi semakin pendek jarak tempuh maka dapat menekan biaya bahan bakar dan waktu pengantaran.

1.6 Sistematika pembahasan

Sistematika pembahasan yang terdapat pada penelitian ini adalah sebagai berikut:

BAB I Pendahuluan

Bab ini membahas latar belakang, rumusan masalah, tujuan, batasan masalah serta manfaat yang terkait dengan implementasi pada metode optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

BAB II Landasan Kepustakaan

Bab ini membahas tentang dasar teori dan kajian pustaka yang digunakan untuk mendukung metode optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

BAB III Metodologi Penelitian

Bab ini membahas tentang metode dan langkah kerja yang dilakukan untuk membangun implementasi optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

BAB IV Perancangan

Bab ini membahas tentang analisa kebutuhan, perancangan antarmuka pengguna dan perancangan perangkat lunak serta proses-proses implementasi optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

BAB V Implementasi

Bab ini membahas tentang jawaban atas pertanyaan bagaimana penerapan optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

BAB VI Pengujian dan Analisis

Bab ini membahas tentang pengujian dan analisis hasil pengujian berdasarkan skenario pengujian yang telah diimplementasikan.

BAB VII Penutup

Bab ini membahas tentang kajian yang berisi kesimpulan dan saran yang diperoleh dari penelitian optimasi *Travelling Salesman Problem* dengan algoritme *Particle Swarm Optimization* pada kasus angkutan sekolah.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Penelitian tentang optimasi rute kendaraan telah banyak dilakukan oleh peneliti sebelumnya. Salah satu contohnya adalah tentang optimasi rute kuliner di Malang oleh Muhammad Afif Hermawan (2017). Algoritme optimasi yang digunakan adalah *Bee Colony* yang meniru tingkah laku kelompok lebah dalam pencarian sumber makanan. Penulis merepresentasikan titik jarak antar tempat kuliner dengan pasukan lebah. Proses perhitungan dalam pencarian solusi dilakukan dengan beberapa fase yaitu *employee bee*, *scout bee* dan *onloker bee*. Hasil yang didapat cukup baik dengan dapat menghasilkan solusi rute terpendek.

Penelitian terkait selanjutnya adalah tentang pendistribusian air minum pada UD. Tosa Malang yang menggunakan optimasi *multiple Travelling Salesman Problem* (mTSP) dengan algoritme genetika. Studi kasus ini menerapkan mTSP yang mana terdapat beberapa kendaraan yang beroprasi dalam pengiriman sesuai permintaan sejumlah pelanggan. Algoritme genetika berfungsi sebagai optimasi dari mTSP memiliki beberapa tahapan diantaranya adalah pembangkitan populasi, seleksi dengan *elitism*, *crossoverone cut point* dan mutasi. Hasil dari sistem optimasi yang didapat terbilang cukup baik dengan perbandingan selisih total jarak sebanyak 89.3 km dari sistem manual yang diterapkan oleh UD. Tosa Malang (Sayyidah Karimah, 2017). Tabel 2.1 menunjukkan perbandingan kajian pustaka.

Tabel 2.1 Kajian Pustaka

No	Judul	Objek	Metode	Output
		Input & parameter	Proses	Hasil Penelitian
1	**Muhammad Afif Hermawan, 2017**	Optimasi rute terpendek pada wisata kuliner yang ada di Malang.	*Algoritme Bee Colony*	Rute terpendek
		Data lokasi tujuan yang disediakan dan parameter PSO seperti jumlah *Bee Colony* dan banyak iterasi	- Inisialisasi solusi secara *random* dan perhitungan *fitness* tiap partikel - Fase employeed bee dengan swap operator dan swap sequence - Perhitungan probabilitas tiap employeed bee	- Rute terpendek pada tiap lokasi yang diinputkan - Hasil perhitungan *fitness* pada *Gbest* iterasi partikel

			dengan roulete wheel - Fase onlooker bea dengan insert operator dan insert sequence - Fase scout bee dengan hitung jumlah trial dan penyimpanan solusi yang lebih baik - Perulangan sampai maksiterasi	
2.	**Sayyidah Karimah, 2017**	Pencarian rute terpendek dalam pendistribusian air minum.	*Multiple Travelling Salesman Problem* dan *Algoritma Genetika*	Rute terpendek
		Data jumlah pesanan dan jumlah sales beserta parameter algoritme genetika seperti mr, cr, *popSize* dan maksiterasi	- Inisialisasi solusi secara *random* dan perhitungan *fitness* tiap individu - Penerapan tounament selection pada tiap individu - Crossover One cut point - Mutasi - Perulangan selama maksiterasi - Menampilkan rute dengan *fitness* terbaik	Meskipun menghasilkan hasil yang baik, nilai optimal pada metode ini sangat tergantung dengan pengkodean masalah dan operator crossover mutasi nya.
3.	**Yuji Shigehiro, 2010**	Aplikasi *Particle Swarm Optimization* terhadap *Travelling Salesman Problem*	*Particle Swarm Optimization* dan *Travelling Salesman Problem*	Perbandingan benchmark dengan medote *Local Search* dan *Simulated Annealing*

		Parameter PSO yang sudah dipermutasikan	- Permutasi solusi - Proses perhitungan PSO - Perbandingan benchmark	Hasil yang didapat menunjukkan bahwa benchmark PSO lebih baik dan efisien dibandingkan LS dan SA
4.	***Optimasi Travelling Salesman Problem Pada Angkutan Sekolah Dengan Algoritme Particle Swarm Optimization*** **(usulan penulis)**	Optimasi rute pada angkutan sekolah	*Travelling Salesman Problem* dan *Algoritme Particle Swarm Optimization*	Rute terpendek dan tarif angkutan tiap siswa
		Data jumlah siswa beserta alamat rumah dan parameter PSO seperti jumlah partikel, parameter vektor kecepatan dan maksiterasi	- Inisialisasi awal partikel secara *random* dan perhitungan *fitness* masing-masing partikel - Penentuan *Pbest* dan *Gbest* pada iterasi - *Update* kecepatan partikel - *Update* posisi partikel - *Update Pbest* dan *Gbest* - Perulangan selama maksiterasi - Menampilkan rute dengan *fitness* terbaik	-

Yuji Shigehiro (2010), melakukan penelitian untuk mengaplikasikan *Particle Swarm Optimization* pada *Travelling Salesman Problem.* Langkah awalnya adalah mempermutasikan titik tempat tujuan menjadi *vector* bilangan *real*. Dengan perhitungan metode PSO ini didapat hasil yang cukup memuaskan. Perbandingan *benchmark* pun dilakukan dengan membandingkan metode PSO terhadap metode lain seperti *Local Search* (LS) dan *Simulated Annealing* (SA). Metode PSO terbukti lebih cepat dalam komputasi pencarian solusi terbaik dibandingkan metode

lainnya. Oleh karena itu penulis memilih metode *Particle Swarm Optimization* dalam penelitian optimasi angkutan sekolah ini.

2.2 MI Salafiyah Kasim

MI Salafiyah Kasim merupakan salah satu lembaga pendidikan dasar yang berada di Jl. KH. Dimyati Kasim Ploso, kecamatan Selopuro, kabupaten Blitar. Sekolah ini sudah berdiri sejak 30 Januari 1981 yang mana merupakan LPM NU dengan sistem pondok. Sekolah ini juga terbuka untuk umum sehingga sebagian siswa-siswinya merupakan warga sekitar sekolah tersebut dan bahkan dari luar desa hingga luar kecamatan. MI Salafiyah Kasim ini sudah menerapkan sistem pengantaran bagi siswa-siswinya kurang lebih sejak 3 tahun yang lalu. Gagasan penerapan sistem ini dilandasi dengan rasa perhatian dari pihak sekolah kepada siswa-siswinya.

Gambar 2.1 MI Salafiyah Kasim

Dengan mengadakan pertemuan dengan wali murid, gagasan yang disampaikan mendapat respons baik dari para wali. Sekolah menawarkan antar jemput namun wali meminta hanya pengantaran waktu sepulang sekolah saja. Hal ini dikarenakan jadwal kegiatan sekolah dimulai pada pukul 06.30 WIB dengan sholat dhuha berjamaah terlebih dahulu baru masuk kelas untuk mulai pelajaran. Sedangkan para wali merasa tidak siap dan kasihan pada anak-anak jika diharuskan sudah siap terlalu pagi untuk dijemput. Apalagi ada beberapa siswa yang letak rumahnya terbilang cukup jauh sehingga diharuskan siap lebih awal untuk urutan penjemputan yang tentunya akan memakan waktu lama untuk menjemput siswa lain hingga sampai ke sekolah.

(a) (b)

Gambar 2.2 (a) Mobil sekolah tampak depan (b) Mobil sekolah tampak dalam

Sistem pengantaran pada MI Salafiyah Kasim terbagi menjadi dua pembagian kloter dengan kloter pertama adalah siswa-siswi kelas 1 dan 2 pada jam 11.30 dan Kloter kedua adalah siswa-siswi kelas 3 – 6 pada jam 13.00 WIB. Untuk kebijakan pembayaran biaya antar terdapat banyak perubahan hingga mendapat kebijakan terbaru yang diterapkan saat ini adalah pembayaran setiap kali naik. Besar biayanya adalah Rp2000,00 per anak.

Pendataan siapa saja siswa-siswi yang akan diantar dilakukan pada saat jam istirahat atau terkadang selesai pelajaran terakhir sebelum pulang. Hal ini dilakukan rutin setiap harinya dikarenakan ketidakpastian siswa-siswi yang ingin diantar oleh sekolah. Namun ada beberapa siswa yang memang sudah pasti akan selalu diantar karena orang tua masing-masing yang tidak bisa menjemput ke sekolah.

2.3 Graf

Graf menghubungkan suatu simpul (*node*) dan himpunan sisi (*edges*) yang merupakan struktur diskrit guna merepresentasikan objek maupun hubungan antar objek. Simpul (*node*) dalam Graf digambarkan dengan lingkaran kecil atau titik tebal, dapat direpresentasikan sebagai kota, atom suatu zat, nama orang, bilangan dan lain-lain. Himpunan sisi digambarkan dengan garis atau anak panah yang berarah pada simpul yang lainnya, merupakan hubungan antar simpul yang dapat direpresentasikan sebagai rute perjalanan, jalan raya, ikatan kimia dan lain-lain (Munir, 2005).

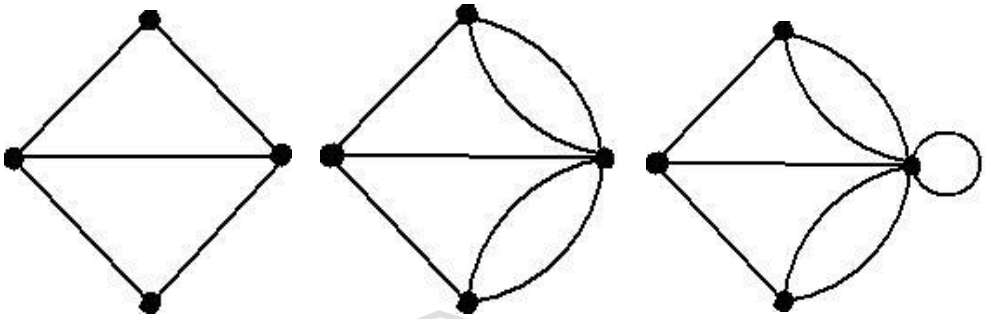
Terdapat beberapa jenis dalam Graf. Apabila dilihat berdasarkan ada atau tidaknya sisi ganda atau sisi gelang (*loop*), maka dapat dibedakan menjadi dua jenis yakni (Lipschutz, Teo dan Hendy, 1985):

1. Graf sederhana

   Merupakan Graf yang tidak mengandung sisi ganda ataupun sisi gelang.

2. Graf tak sederhana

   Merupakan Graf yang terdapat sisi ganda atau sisi gelang. Ada dua macam pada Graf ini yaitu, Graf ganda (*multigraph*) dan Graf semu (*pseudograph*). Bisa dilihat pada Gambar 2.3



(a) (b) (c)

Gambar 2.3 (a) Graf sederhana, (b) Graf ganda dan (c) Graf semu

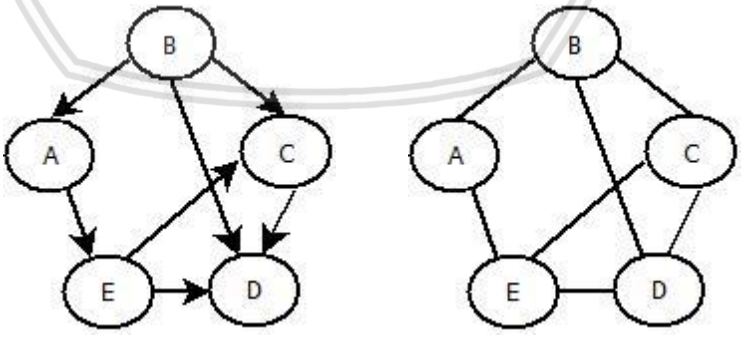
Sedangkan jika berdasarkan orientasi arah Graf, dapat dibedakan menjadi dua jenis yakni (Lipschutz, Teo dan Hendy, 1985):

1. Graf berarah (*Direct Graph*)

   Merupakan Graf yang pada setiap sisinya diberi orientasi arah dan tidak memiliki dua sisi yang berlawanan antara dua simpul. Sisi arah Graf dapat disebut dengan busur (*arc*).

2. Graf tak berarah (*Undirect Graph*)

   Merupakan Graf yang pada setiap sisinya tidak memiliki orientasi arah. Dapat dilihat pada Gambar 2.4.



(a) (b)

Gambar 2.4 (a) Graf berarah, (b) Graf tak berarah

2.4 Optimasi

Optimasi merupakan usaha terbaik yang dilakukan dalam pencarian solusi optimum atas permasalahan dengan *cost* seminimal mungkin (Dorigo, M., dan Stutzle, T., 2004). Permasalahan yang diselesaikan biasanya dinyatakan dalam bentuk fungsi yang dipengaruhi oleh parameter atau variabel tertentu. Keuntungan yang ingin dicapai harus sesuai dengan batasan yang didefinisikan untuk semua variabel. Bidang rekayasa maupun ilmu *engineering* senantiasa dihadapkan dengan masalah optimasi dalam perancangan maupun penyelesaian masalah (Eliantara, 2016).

2.5 *Travelling Salesman Problem* (TSP)

TSP merupakan salah satu persoalan optimasi kombinatorial yang menggunakan representasi Graf untuk memodelkan permasalahan. Berbagai contoh permasalahan yang dapat dimodelkan dengan TSP adalah pencarian rute untuk travel, optimasi pengiriman surat atau barang dan lain-lain. Setelah permodelan, persoalan selanjutnya adalah bagaimana cara mengunjungi suatu node (titik) tujuan, dari satu ke yang lainnya dengan pengeluaran biaya seminimal mungkin. Biaya atau bobot ini dapat mewakili berbagai hal seperti biaya minimum, jarak minimum, bahan bakar minimal, kenyamanan, waktu minimum dan lain-lain. Rumus permutasi yang bisa digunakan pada TSP adalah sebagai berikut:

$$p_n^k = \frac{n!}{(n-k)!} \tag{2.1}$$

Keterangan :

n = jumlah semua node.

k = jumlah node yang dilalui.

2.6 *Particle Swarm Optimization* (PSO)

PSO merupakan salah satu algoritme optimasi stokastik yang berbasis populasi yang telah dikembangkan oleh Russell Eberhart dan James Kennedy pada tahun 1995. Algoritme ini terinspirasi oleh pola kehidupan populasi burung dan ikan untuk bertahan hidup (Haupt, 2004). PSO digunakan untuk memecahkan masalah optimasi. *Particle Swarm Optimization* memiliki banyak persamaan dengan teknik komputasi evolusioner lain seperti Algoritme genetika (GA). Salah satu kelebihan dari Particle Swarm Optimization (PSO) adalah sangat mudah untuk diimplementasikan dan hanya ada beberapa parameter yang bisa disesuaikan.

Dengan melakukan optimasi berbasis populasi, PSO dapat mencari solusi optimal dengan menggunakan partikel dalam populasi itu sendiri. Partikel tersebut berperilaku sebagai agen lingkungan yang cerdas dan setiap partikel juga sangat berkontribusi di dalam lingkungan untuk dapat bisa saling bekerjasama dan berkomunikasi dengan partikel lainnya. PSO didasari ide bahwa setiap partikel

dalam kerumuman populasi merupakan solusi dari ruang solusi itu sendiri (Kurniawan, 2010).

Berikut merupakan istilah yang sering dijumpai pada algoritme *particle swarm optimization* (Sedighizadeh, 2009):

- Swarm: populasi dalam algoritme.
- Partikel: individu dalam populasi yang merepresentasikan solusi. opensial dalam penyelesaian masalah.
- *Pbest* (*Personal Best*): suatu partikel yang menunjukkan posisi terbaik dalam pencarian solusi.
- *Gbest* (*Global Best*): posisi terbaik partikel dalam iterasi.
- *Velocity*: vektor yang menentukan arah partikel akan berpindah untuk perbaikan posisi dalam pencarian solusi optimum.
- *Inertia Weight*: parameter pengontrol dampak adanya *velocity* pada suatu partikel. Biasa dilambangkan dengan simbol ω.
- Koefisien akselerasi: parameter yang mempengaruhi jarak maksimum pada partikel dalam iterasi.

Sedangkan untuk langkah proses yang dilakukan pada algoritme *particle swarm optimization* adalah sebagai berikut (Maickel et al, 2009):

1. Inisialisasi populasi partikel dengan posisi secara *random* dalam suatu ruang dimensi pencarian.
2. Mengevaluasi fungsi *fitness* optimisasi pada setiap partikel yang nantinya akan didapatkan *Pbest* dan *Gbest* nya.
3. *Update velocity* dan posisi untuk masing-masing partikel.
   Persamaan *Update* Kecepatan:

$$V_j^k = WV_j^k + C_1 rand_1 \times \left(Pbest_j - X_j^k\right) + C_2 rand_2 \times \left(Gbest_j - X_j^k\right) \quad \textbf{(2.2)}$$

   Keterangan:

   V_j^k = *velocity* partikel ke-*j* pada *iterasi* ke-*k*

   W = *bobot inertia*

   C_1 = nilai koefisien akselerasi ke-1

   C_2 = nilai koefisiean akselerasi ke-2

   $rand_{1,2}$ = nilai *random* [0, 1]

   X_j^k = posisi partikel ke-*j* pada *Iterasi* ke-*k*

   $Pbest_j$ = nilai *Pbest* dari dimensi ke-*j*

   $Gbest_j$ = nilai *Gbest* dari dimensi ke-*j*

Fungsi *Update* bobot *inersia:*

$$W = Wmax - \frac{Wmax - Wmin}{iter\ max} x\ iter$$

(2.3)

Persamaan *Update* posisi:

$$\boldsymbol{X_j^{k+1}} = \boldsymbol{X_j^k} + \boldsymbol{V_j^{k+1}}$$

(2.4)

4. Perhitungan *fitness* ulang dan *Update* nilai *Pbest* dan *Gbest* nya.
5. Melakukan perulangan hingga mencapai *stop condition* yang ditetapkan.

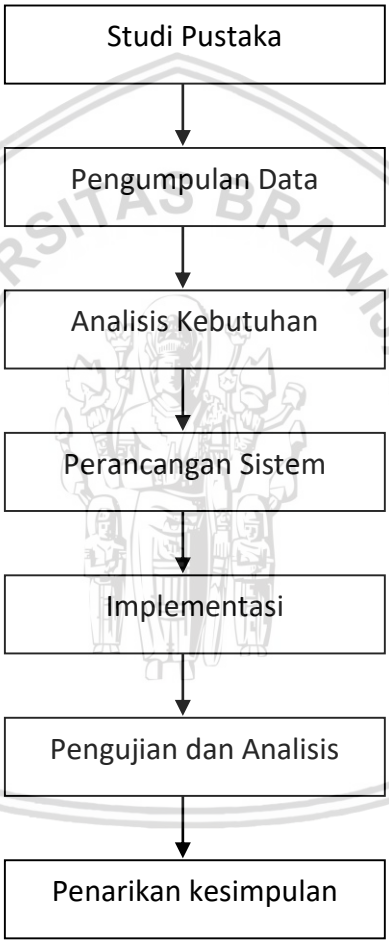
Penerapan PSO pada TSP memiliki peran tersendiri dalam pencarian solusi terbaik. Nilai *fitness* yang didapat merupakan kombinatorial dari kedua metode tersebut. PSO mencari solusi terbaik dengan merepresentasikan alamat siswa yang diantar dengan urutan acak pada suatu partikel dalam populasi. Sedangkan TSP berperan dalam menentukan kualitas solusi terbaik dengan persamaan sebagai berikut:

$$Fitness = \frac{1}{Total\ jarak}$$

(2.5)

BAB 3 METODOLOGI

Penelitian yang dilakukan penulis merupakan penelitian nonimplementatif analitik. Penelitian analitik merupakan penelitian yang dilakukan untuk mengetahui suatu sebab dan akibat dari dua variabel. Bab ini akan membahas tentang langkah-langkah atau proses yang dilakukan oleh penulis dalam penelitian optimasi *Travelling Salesman Problem* pada angkutan sekolah dengan algoritme *Particle Swarm Optimization.* Gambaran umum alur metodologi yang digunakan dapat dilihat pada Gambar 3.1.



Gambar 3.1 Metodologi Penelitian

3.1 Studi Pustaka

Studi pustaka digunakan sebagai teori pendukung dan landasan dasar dalam melakukan penelitian. Pustaka yang digunakan dapat bersumber dari jurnal ilmiah, buku, laporan penelitian maupun situs internet yang berkaitan. Adapun contoh pustaka yang digunakan penulis di antaranya adalah seperti teori, metode dan algoritme yang sesuai dengan topik yang diangkat. Studi pustaka penting

dilakukan agar dapat membandingkan penelitian sebelumnya dan menghindari penelitian yang sama. Dengan adanya pustaka, maka dapat memudahkan pembaca dalam memahami permasalahan.

3.2 Pengumpulan Data

Penulis menerapkan beberapa cara dalam proses pengumpulan data. Salah satunya adalah observasi secara langsung kepada pihak sekolah yakni pada MI Salafiyah Kasim yang terletak di Jl. KH. Dimyati Kasim Ploso, kecamatan Selopuro, kabupaten Blitar. Observasi ini dilakukan pada tanggal 16 Januari 2018 yang mana bertemu dengan bagian sarana prasarana dan petugas supir angkutan sekolah. Adapun data yang didapat adalah jumlah siswa sekolah beserta alamat rumahnya bagi yang berpotensi untuk menggunakan jasa angkutan sekolah. Terdapat 20 siswa untuk kloter pertama dan 24 siswa untuk kloter kedua. Selain itu, penulis juga mendapat data biaya pengantaran hingga dana anggaran untuk perawatan angkutan sekolah tersebut.

3.3 Analisis Kebutuhan

3.3.1 Deskripsi Umum Sistem

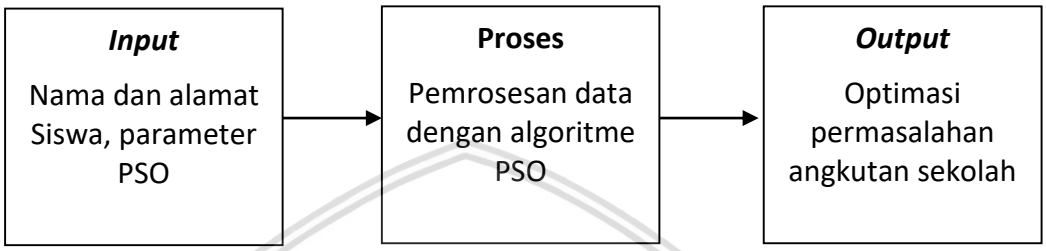
Secara umum, sistem yang akan dibangun dalam penelitian ini adalah sistem *Travelling Salesman Problem* (TSP) pada angkutan sekolah menggunakan algoritme *Particle Swarm Optimization* (PSO). Proses pertama yang harus dilakukan pada sistem ini adalah memasukkan data siswa yang akan diantar meliputi nama dan alamat dari siswa tersebut. Sistem juga sudah menyiapkan data *deafult* apabila *user* tidak ingin memasukkan data siswa. Selain itu sistem juga memerlukan *masukkan* parameter PSO seperti bobot inersia (*Wmin, Wmax*), jumlah partikel dan *iterasi* maksimum. Setelah proses *input* data siswa dan penentuan parameter algoritme, sistem akan menghitung *fitness* pada setiap partikel untuk dijadikan acuan dalam penentuan *Pbest* dan *Gbest*. Berdasarkan *iterasi* maksimum sebagai kondisi berhenti, maka perulangan pada sistem akan berakhir dan memilih partikel dengan nilai *fitness* terbesar sebagai solusi optimum.

3.3.2 Spesifikasi Kebutuhan Sistem

Dalam melakukan penelitian ini, penulis memerlukan kebutuhan perangkat keras dan perangkat lunak dengan seminimalnya memiliki spesifikasi seperti processor Intel Core i3-3210M, RAM 2 GB 2133 MHz, DDR3 2.5GHz dengan sistem operasi minimal *Windows 7* 32 *bit*s, *Database MySQL* dan *Netbeans IDE 7.3*. Sistem harus dapat menerima masukkan data siswa dan parameter algoritme PSO, sistem harus dapat menampilkan parameter masukkan, sistem harus dapat menampilkan hasil perhitungan algoritme PSO dan sistem harus mampu menampilkan hasil optimasi permasalahan angkutan sekolah.

3.4 Perancangan Sistem

Perancangan sistem akan membahas tahapan bagaimana sistem nanti menyelesaikan permasalahan dengan menggunakan metode yang dipakai mulai dari perhitungan manualisasi hingga penjelasan diagram alir dari masing-masing proses yang ada dalam sistem. Perancangan sistem disusun berdasarkan sistem secara umum yakni dimulai dari *input* data dan parameter, proses perhitungan dalam pencarian solusi hingga *output* hasil optimasi permasalahan angkutan sekolah. Dapat dilihat pada Gambar 3.2.



Gambar 3.2 Perancangan Umum Sistem

Adapun proses perhitungan dalam pencarian solusi dengan algoritme PSO meliputi:

1. Inisialisasi populasi partikel dengan posisi secara *random* dalam suatu ruang dimensi pencarian.
2. Mengevaluasi fungsi *fitness* optimisasi pada setiap partikel yang nantinya akan didapatkan *Pbest* dan *Gbest* nya.
3. *Update velocity* dan posisi untuk masing-masing partikel.
4. Perhitungan *fitness* ulang dan *Update* nilai *Pbest* dan *Gbest* nya.
5. Melakukan perulangan hingga mencapai *stop condition* yang ditetapkan.

3.5 Implementasi

Implementasi akan membahas tentang tahapan untuk me-realisasikan perancangan yang telah dibuat sebelumnya. Implementasi sistem diaplikasikan dengan menggunakan bahasa pemrograman JAVA.

3.6 Pengujian dan Analisis

Pengujian akan membahas tentang kinerja sistem terkait akurasi dengan hasil manualisasi pada tahap sebelumnya dan pengujian parameter-parameter yang berpengaruh terhadap hasil akhir dari sistem yang akan diberikan persentase berupa grafik hasil pengujian.

3.7 Penarikan Kesimpulan

Tahap terakhir dari penelitian ini adalah penarikan kesimpulan yang didapatkan dari hasil pengujian sistem. Penambahan saran akan disertakan untuk memperbaiki kekurangan yang terjadi selama atau pada hasil penelitian agar untuk kedepannya sistem dapat dikembangkan lebih lanjut.

BAB 4 PERANCANGAN

4.1 Formulasi Permasalahan

Permasalahan pada penelitian ini adalah bagaimana mengoptimasi rute angkutan sekolah dengan menghasilkan rute terpendek yang bisa dilalui ketika mengantar siswa-siswi. Proses optimasi sendiri menggunakan metode *Travelling Salesman Problem* (TSP) dan *Particle Swarm Optimization* (PSO). Pemodelan awal adalah setiap siswa-siswi akan diberikan ID unik yang mana pasti akan berbeda satu dengan yang lainnya. Langkah pertama adalah membuat matriks jarak dari sekolah ke ID pertama hingga ID terakhir. Sedangkan data jarak pada penelitian ini diambil dari Google Maps. Tabel 4.1 menunjukkan contoh matriks jarak kloter pertama dan untuk data lengkapnya bisa dilihat pada Lampiran A.1.

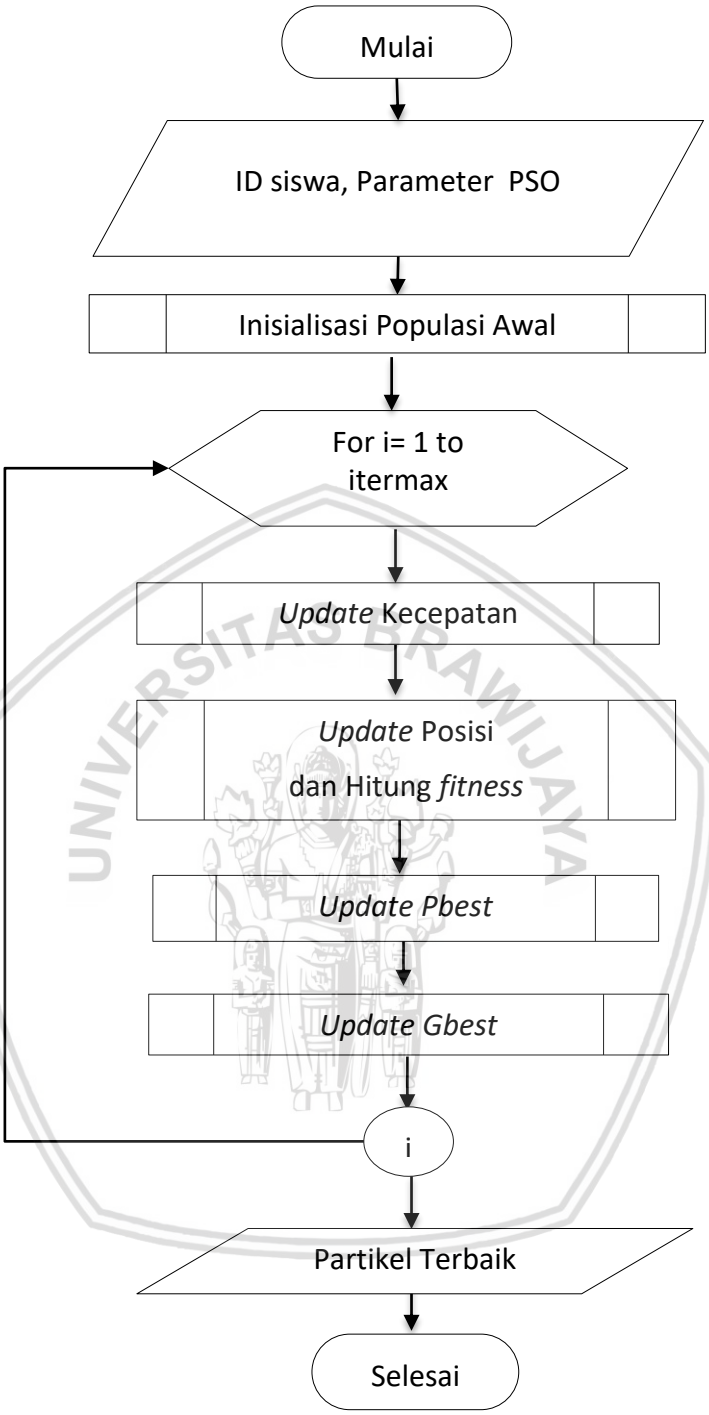
Tabel 4.1 Matriks Jarak Kloter Pertama (Km)

	S	1	2	3	4	5	6	7	8	9	10	11	12	13	..	..	20
S	0	2	1,5	1,6	1,8	1,6	1,9	1,9	1,8	1,6	1,3	2	2,2	3	..	..	2,8
1	2	0	0,5	0,4	0,3	0,4	0,2	1,6	1,6	3,2	1,1	4,9	4,6	4,6	..	..	1,1
2	1,5	0,5	0	0,2	0,4	0,2	0,4	1,8	1,8	3,1	0,4	3,4	3,6	4,5	..	..	1,6
3	1,6	0,4	0,2	0	0,2	0,1	0,3	1,6	1,6	3,3	0,6	3,6	3,8	4,6	..	..	1,3
4	1,8	0,3	0,2	0,2	0	0,2	0,1	1,4	1,4	3,4	0,8	3,8	4,8	4,8	..	..	1,4
5	1,6	0,4	0,2	0,1	0,2	0	0,3	1,6	1,6	3,3	0,6	3,6	3,8	4,6	..	..	1,4
6	1,9	0,2	0,4	0,3	0,1	0,3	0	1,5	1,5	3,3	0,9	3,9	4,7	4,7	..	..	1,2
7	1,9	1,6	1,8	1,6	1,4	1,6	1,5	0	0,1	3,5	2,3	3,9	4,1	4,9	..	..	2,6
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..
20	2,8	1,1	1,6	1,3	1,4	1,4	1,2	2,6	2,7	2,3	1,4	3,9	3,7	3,7	..	..	0

Urutan pengantaran sangat menentukan berapa total jarak yang akan ditempuh. TSP berperan dalam menghitung total jarak dari ID siswa yang terpilih, sedangkan PSO berperan dalam menentukan urutan ID siswa yang diantar tersebut. PSO yang digunakan pada penelitian ini adalah *Real Coded* sehingga mengharuskan adanya penambahan angka permutasi untuk memudahkan perhitungan saat pencarian solusi terbaik. ID siswa akan dikonversi menjadi angka permutasi mulai dari 1 sampai banyaknya ID yang terpilih.

4.2 Proses Penyelesaian masalah dengan Algoritma PSO

Dalam sub bab ini akan memperlihatkan diagram alir proses-proses untuk menyelesaikan permasalahan Optimasi TSP pada angkutan sekolah dengan Algoritme PSO yang diharapkan dapat memberikan rekomendasi urutan pengantaran dengan jarak tempuh seminimal mungkin.



Gambar 4.1 Diagram Alir Algoritma PSO untuk Optimasi Angkutan Sekolah

Gambar 4.1 merupakan diagram alir yang menunjukkan langkah-langkah dalam menyelesaikan permasalahan Optimasi TSP pada angkutan sekolah dengan PSO dengan pembahasan sebagai berikut:

1. Menerima masukkan berupa ID siswa yang diantar pada tiap kloter beserta Parameter PSO
2. Inisialisasi Populasi Awal meliputi Inisialisasi Patikel, *fitness*, posisi, kecepatan, *Pbest* dan *Gbest* awal hingga sebanyak jumlah partikel

3. *Update* Kecepatan
4. *Update* Posisi dan hitung *fitness*
5. *Update Pbest*
6. *Update Gbest*
7. Ulangi langkah 3-7 hingga sejumlah *itermax*
8. *Output* Partikel terbaik

4.2.1 *Input* Data

Pada penelitian ini, data yang dimasukkan adalah data ID siswa yang akan diantar beserta parameter PSO. Setiap ID menyimpan informasi data siswa berupa nama, kelas dan alamat. Sedangkan parameter PSO yang digunakan diantaranya adalah nilai Koefisien Akselerasi (C_1 dan C_2), *Wmin*, *Wmax*, *Rand* (*rand*$_1$ dan *rand*$_2$), Jumlah iterasi *maximum* (*itermax*), dan jumlah partikel.

4.2.2 Inisialisasi Populasi Awal

Tahap ini terdapat beberapa inisialisasi awal yakni inisialisasi kecepatan, posisi partikel dan masing-masing *fitness* partikel beserta *Pbest* dan *Gbest*. Berikut merupakan penjelasan masing-masing inisialisasinya:

4.2.2.1 Inisialisasi Kecepatan

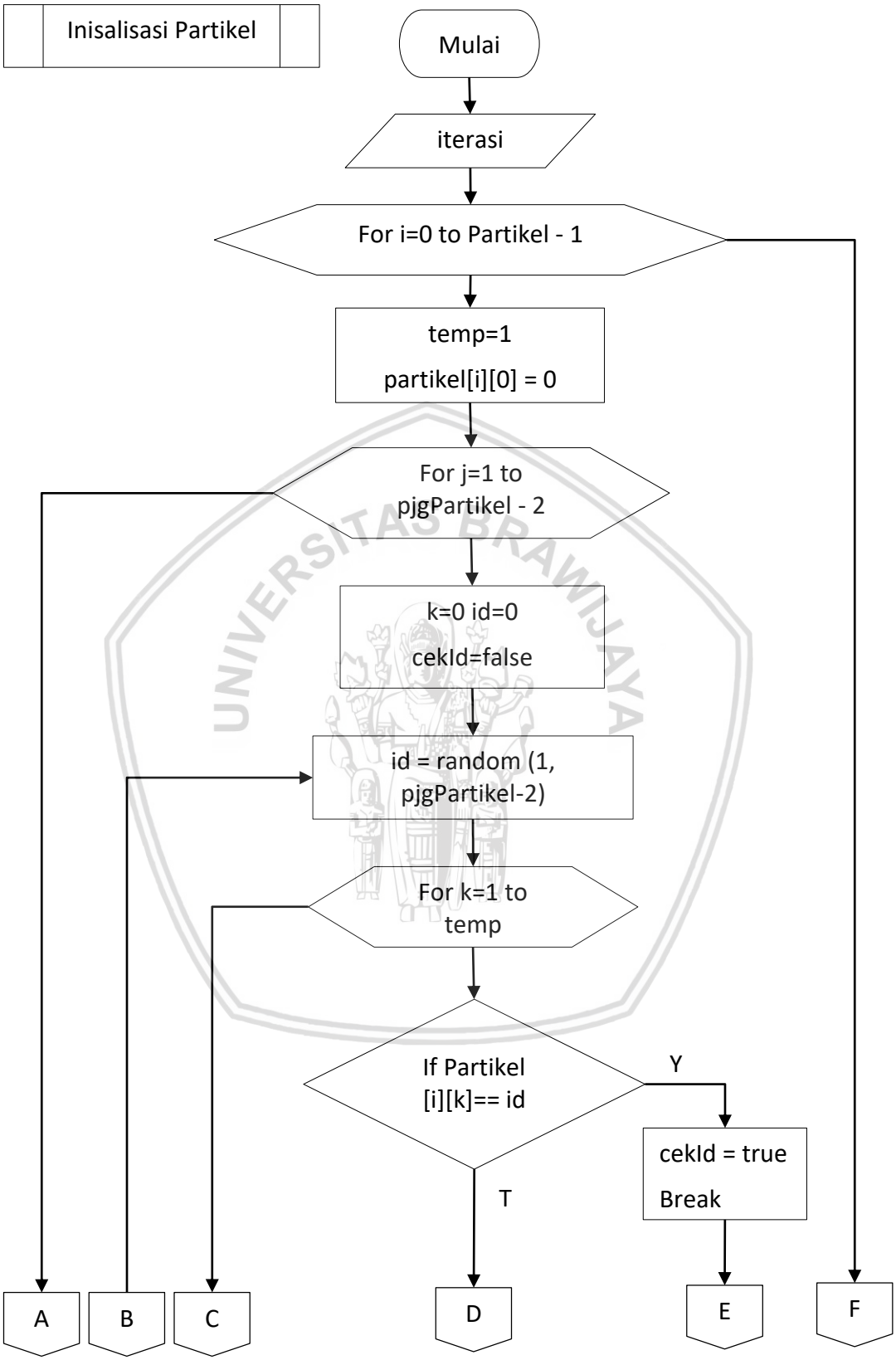
Inisialisasi Kecepatan merupakan kecepatan awal yang dimiliki oleh sebuah partikel. Pada inisialisasi ini, kecepatan (*V*) setiap partikel bernilai 0 (nol). Dapat dilihat pada Tabel 4.2.

Tabel 4.2 Inisialisasi Kecepatan

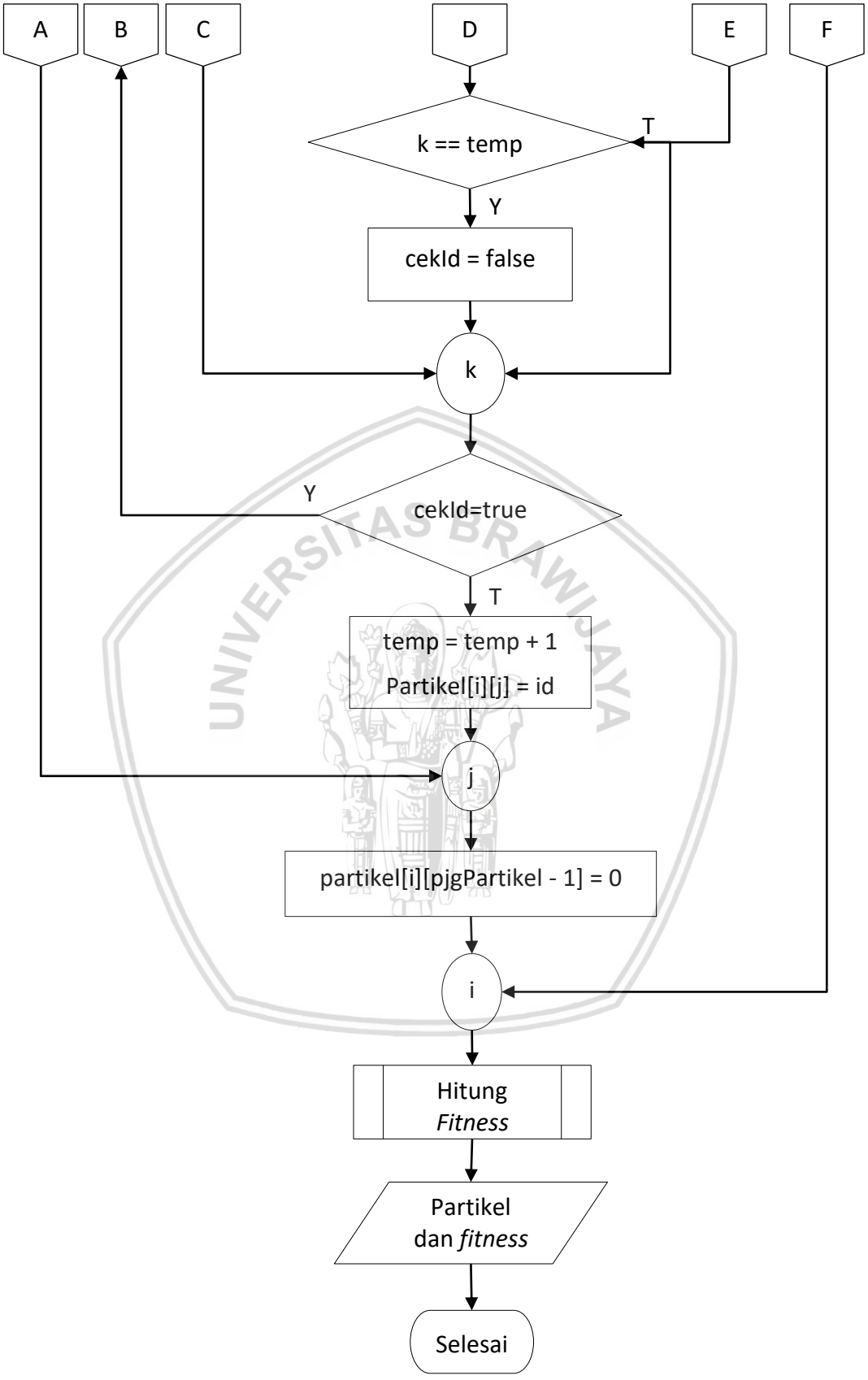
Iterasi =0												
V1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0
V2	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0
V3	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0

4.2.2.2 Inisialisasi Partikel

Ketika masukkan data ID siswa dan parameter PSO sudah diterima, pembentukan partikel akan dibangkitkan sejumlah masukkan dan masing-masing partikel berisi urutan ID siswa terpilih secara acak. Namun sebelum itu, ID siswa terpilih akan dikonversi ke dalam angka permutasi untuk memudahkan pencarian solusi. Pengkonversian dilakukan dengan ID pertama mendapat angka permutasi 1 dan seterusnya hingga sejumlah ID siswa yang terpilih. Gambar 4.2 menjelaskan diagram alir Pembentukan Partikel berdasarkan *random* angka permutasi.

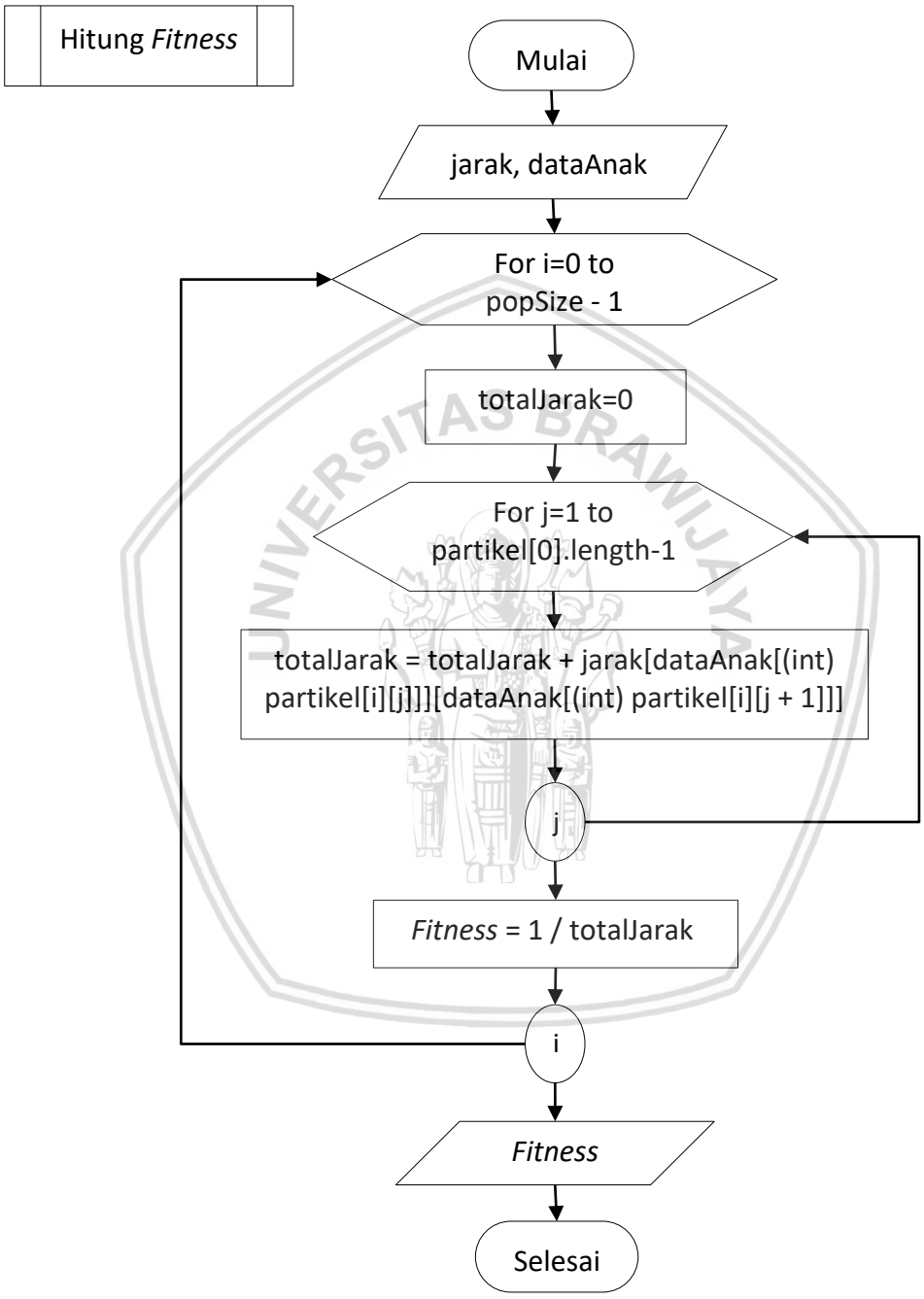


Gambar 4.2 Diagram Alir Inisialisasi Partikel



Gambar 4.2 Diagram Alir Inisialisasi Partikel (Bagian 2)

Nilai *fitness* pada setiap partikel akan berbeda. khususnya pada permasalahan ini dikarenakan urutan pengantaran menentukan total jarak yang akan ditempuh saat supir melakukan tugasnya. Nilai *fitness* diperoleh dengan menggunakan Persamaan 2.5. Untuk lebih jelasnya, dapat dilihat diagram alir hitung *fitness* pada Gambar 4.3.



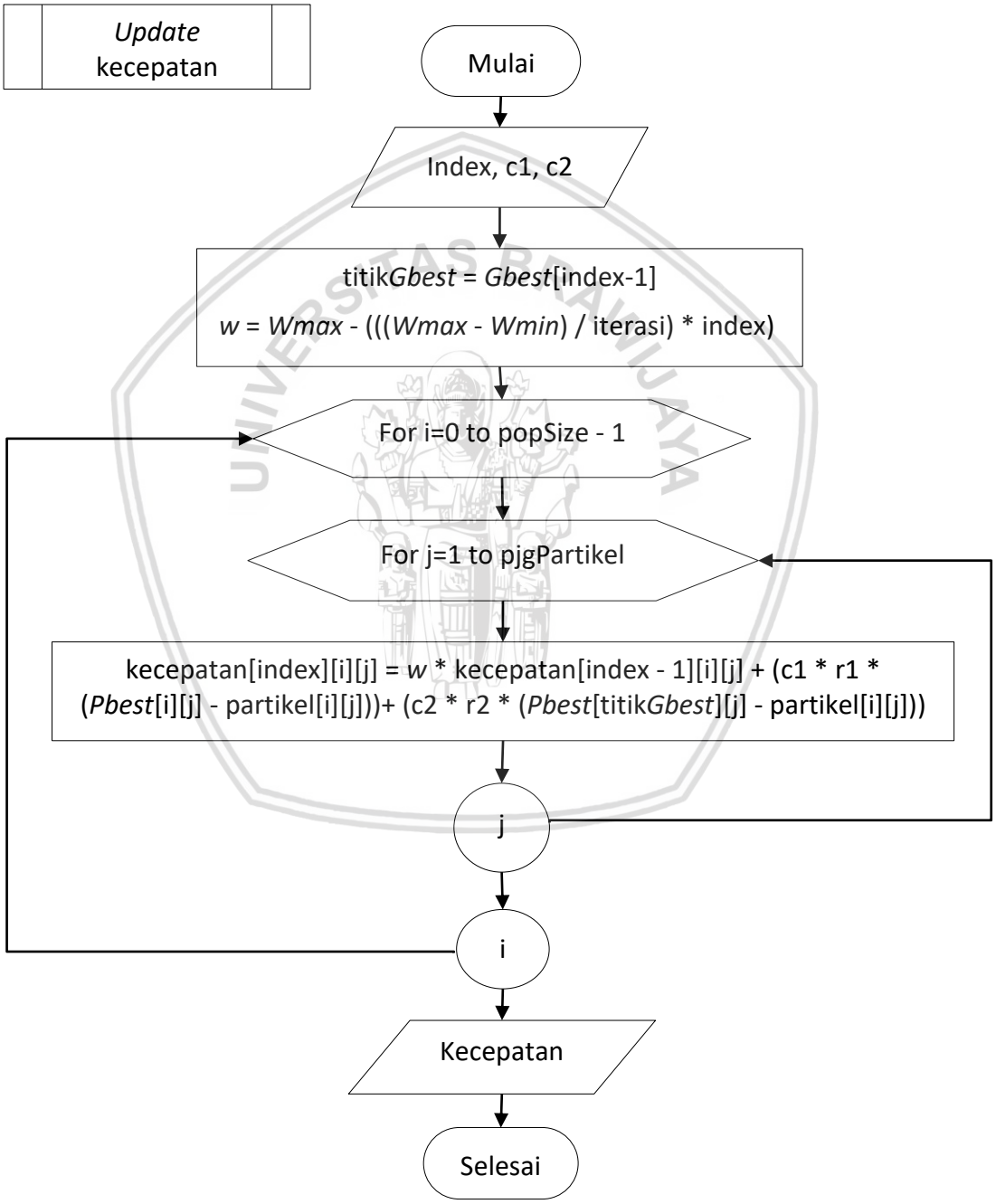
Gambar 4.3 Diagram Alir Hitung *Fitness*

4.2.2.3 Inisialisasi Pbest dan Gbest

Pada awal *iterasi*, nilai *Pbest* disamakan dengan posisi awal partikel dan nilai *Gbest* diperoleh dari *Pbest* dengan nilai *fitness* yang tertinggi.

4.2.3 *Update* Kecepatan

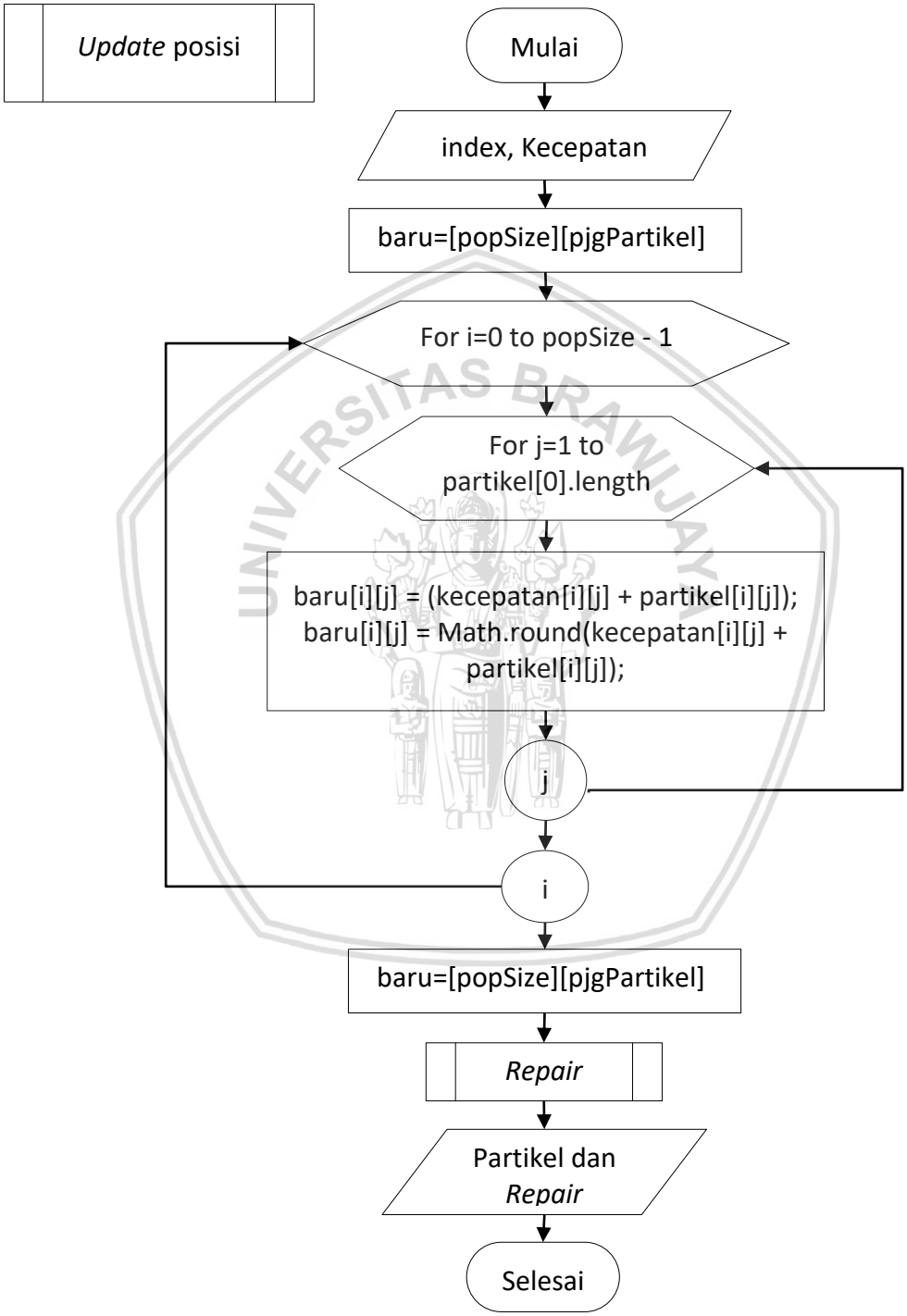
Dalam pencarian solusi, setiap partikel akan terus berpindah posisi hingga mencapai terasi maksimum. *Update* kecepatan dibutuhkan untuk perpindahan posisi partikel. Gambar 4.4 menunjukkan diagram alir *Update* Kecepatan.



Gambar 4.4 Diagram Alir *Update* Kecepatan

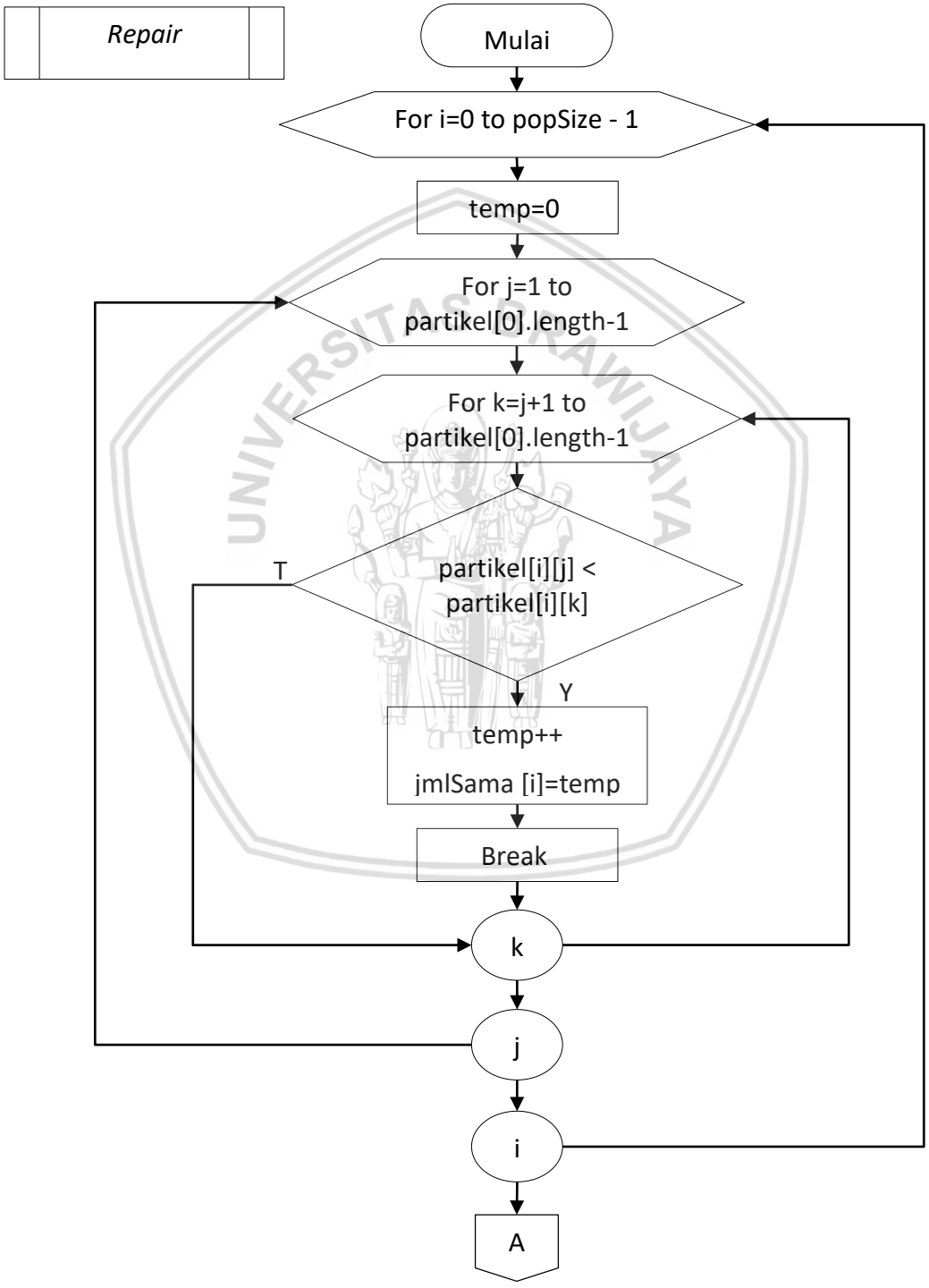
4.2.4 *Update* Posisi

Update Posisi merupakan perpindahan posisi partikel karena *Update* Kecepatan pada tahap sebelumnya. Dalam pengimplementasian pada permasalahan ini, terdapat tahapan khusus yaitu adalah *Repair*. Gambar 4.5 menunjukkan diagram alir *Update* posisi.

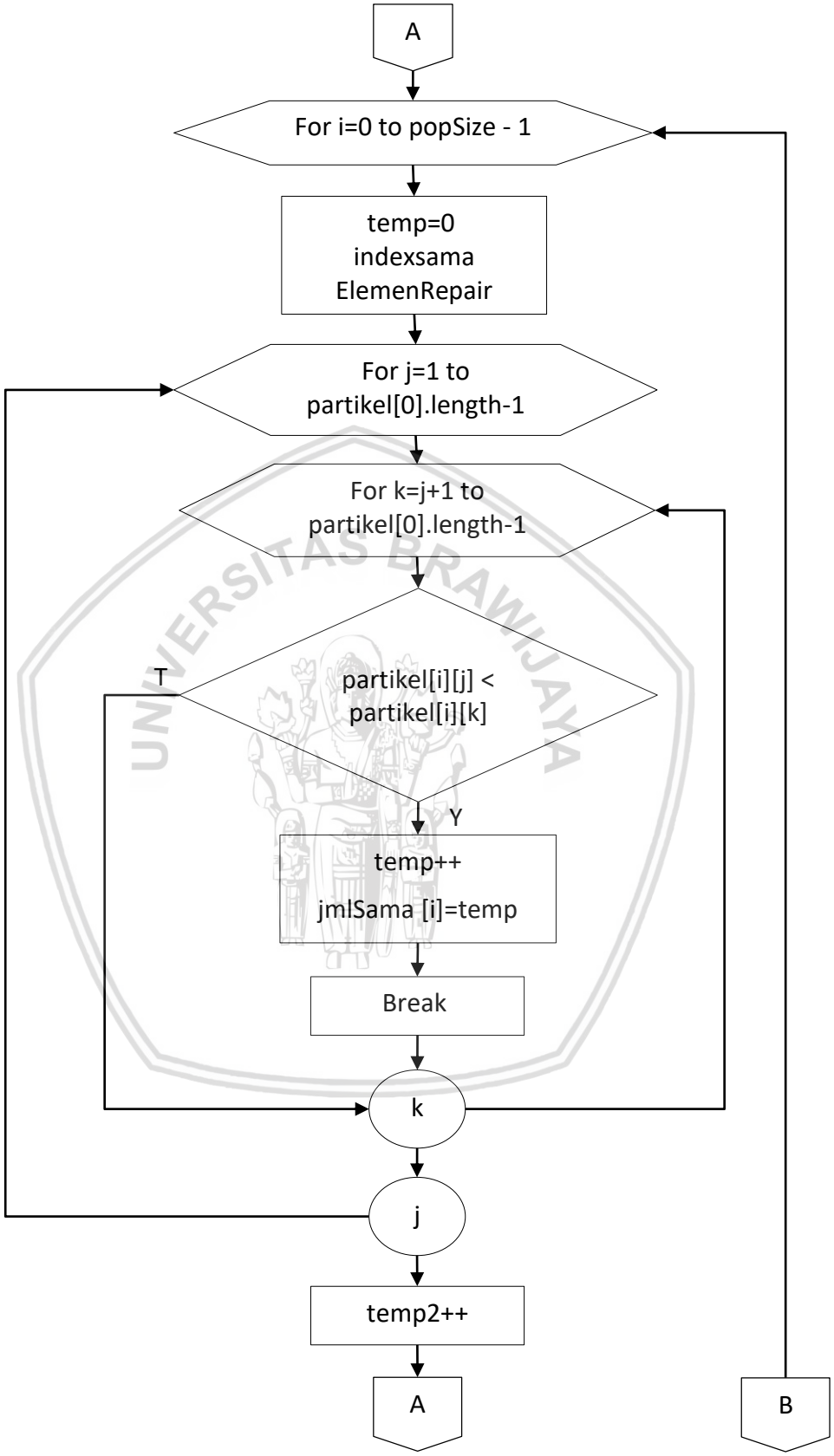


Gambar 4.5 Diagram Alir *Update* Posisi

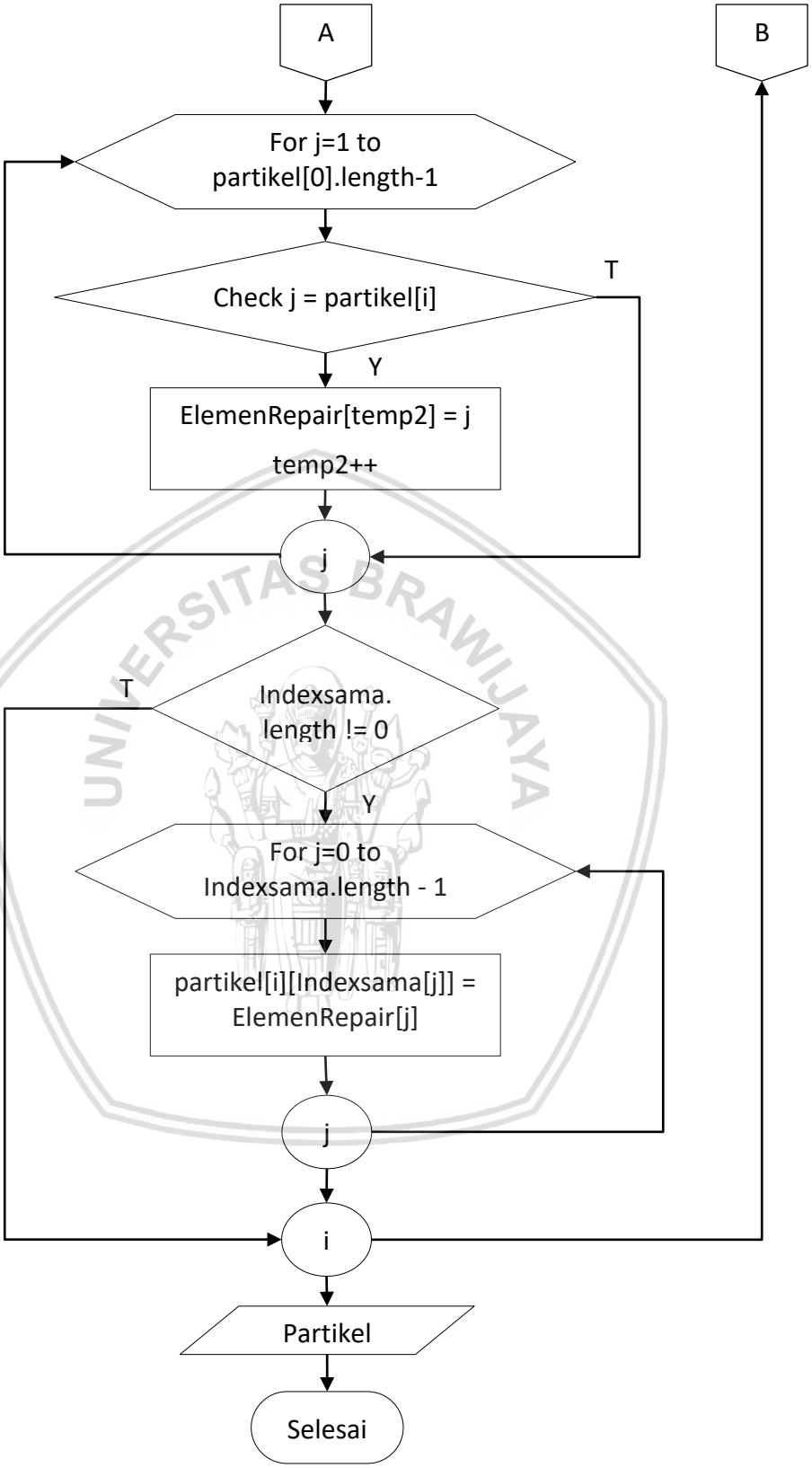
Repair diterapkan secara khusus pada permasalahan ini dikarenakan Algoritme PSO yang digunakan adalah *Real Coded*. Hal ini mengakibatkan perlunya ada penambahan angka permutasi yang diatur secara *Unique* untuk mewakili masing-masing ID siswa. Setelah *Update* kecepatan, setiap partikel akan memperbarui posisinya dan perlu dilakukan pengecekan dan perbaikan untuk menghindari kerancuan angka permutasinya. Gambar 4.6 menunjukkan diagram alir *Repair.*



Gambar 4.6 Diagram Alir Repair



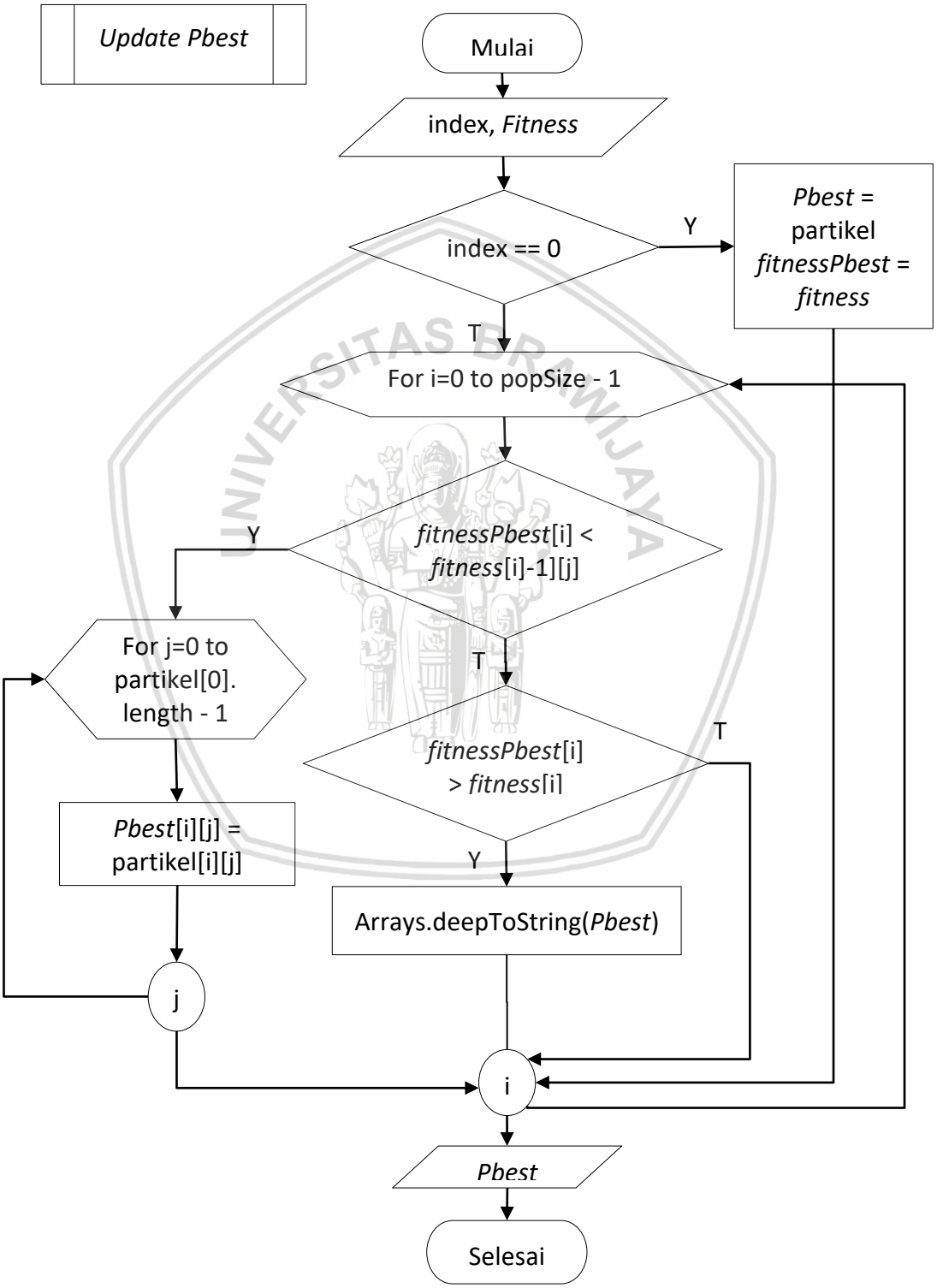
Gambar 4.6 Diagram Alir Repair (Bagian 2)



Gambar 4.6 Diagram Alir Repair (Bagian 3)

4.2.5 *Update Pbest*

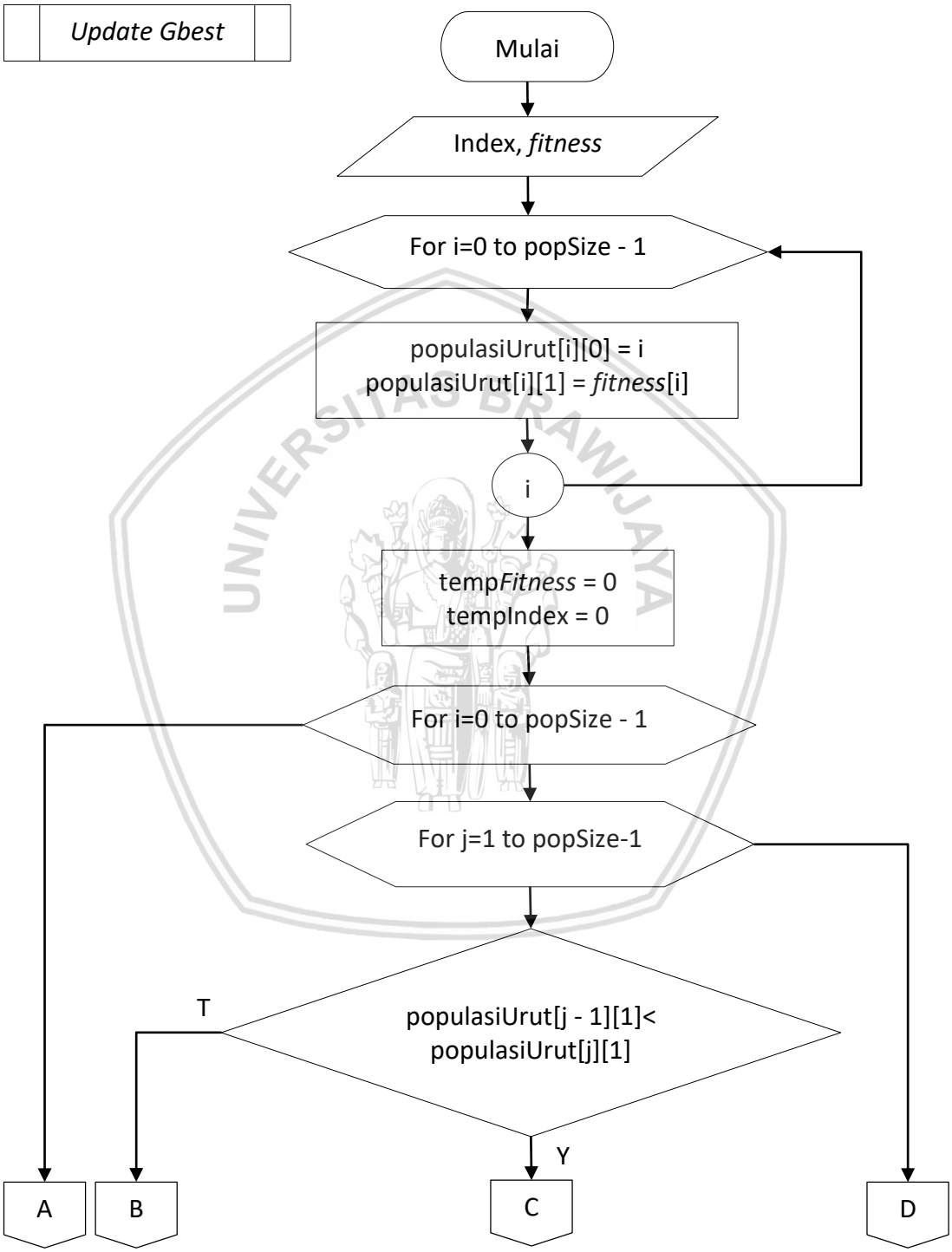
Nilai *Pbest* akan selalu diperbarui pada setiap iterasi dengan *Update Pbest*, yang didapatkan dengan cara membandingkan nilai *fitness* partikel pada iterasi saat ini dengan *fitness Pbest* iterasi sebelumnya. Gambar 4.7 menunjukkan diagram alir *Update Pbest*.



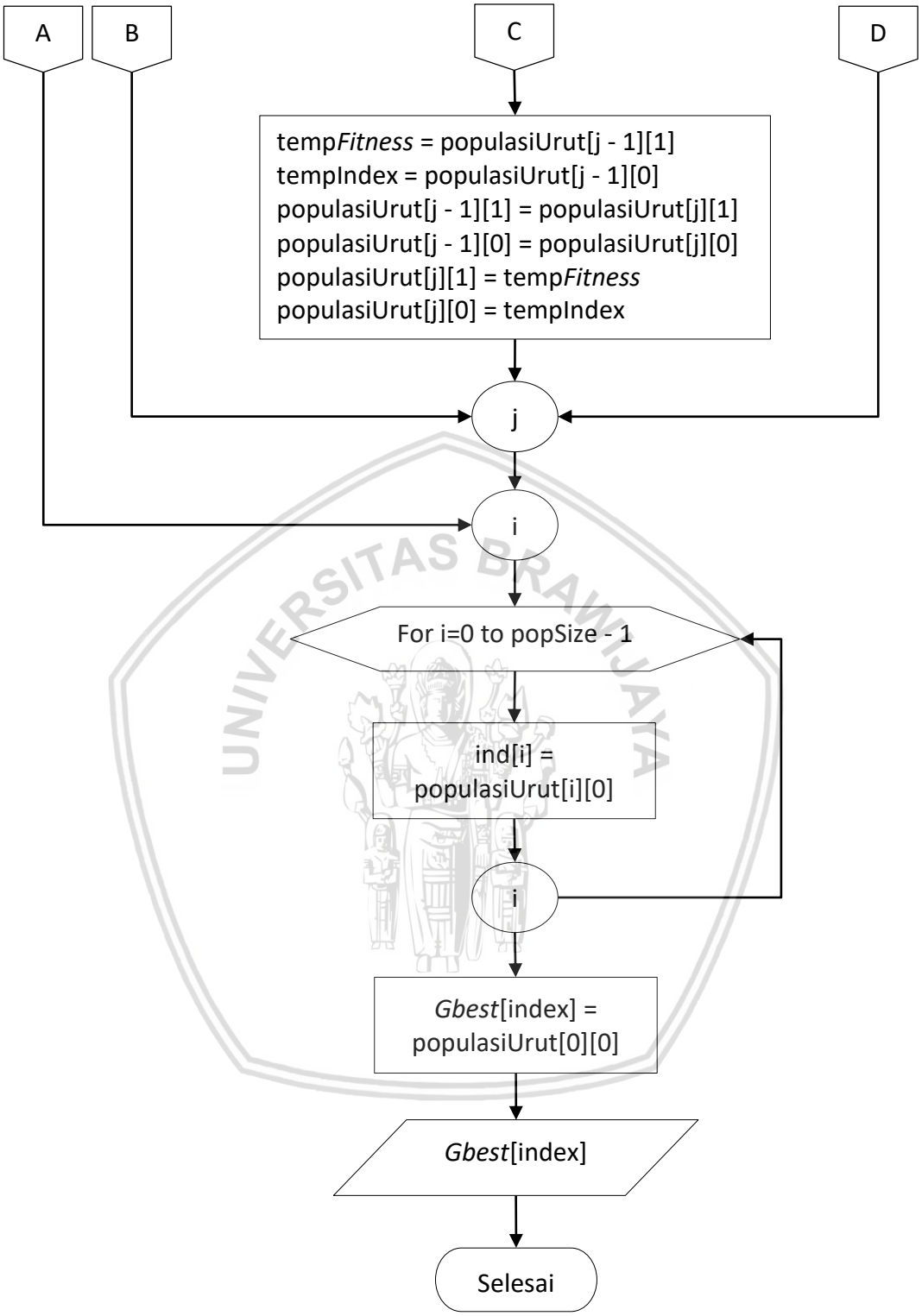
Gambar 4.7 Diagram Alir *Update Pbest*

4.2.6 *Update Gbest*

Nilai *Gbest* didapatkan dari *Pbest* dengan nilai *fitness* terbesar dalam populasi. *Gbest* pada tahap ini akan dijadikan sebagai solusi terbaik suatu permasalahan. Gambar 4.8 menunjukkan Diagram Alir *Update Gbest.*



Gambar 4.8 Diagram Alir *Update Gbest*



Gambar 4.8 Diagram Alir *Update Gbest* (Bagian 2)

4.3 Perhitungan Manual

Pada sub bab ini akan membahas contoh perhitungan manual dari optimasi TSP pada angkutan sekolah dengan PSO. Proses perhitungan akan dilakukan secara terpisah pada masing-masing kloter sehingga setiap partikel dan nilai *fitness*nya akan berbeda. Berikut ini merupakan contoh data ID siswa dan Parameter PSO yang digunakan sebagai contoh perhitungan manual, dapat dilihat pada Tabel 4.3 dan Tabel 4.4 yang lebih lengkapnya bisa dilihat di Lampiran B.1.

Tabel 4.3 Data ID Siswa

Kloter Pertama																	
ID	19	14	18	15	17	11	13	9	20	10	2	3	4	5	..	..	8
Kloter Kedua																	
ID	13	14	11	12	19	10	9	8	7	24	20	23	1	2	..	..	17

Tabel 4.4 Parameter PSO

Jumlah Partikel	***C_1***	***C_2***	***$rand_1$***	***$rand_2$***	***Itermax***	***Wmin***	***Wmax***
3	1	1	[0,1]	[0,1]	2	0.4	0,9

4.3.1 Inisialisasi Populasi Awal

4.3.1.1 Inisialisasi Kecepatan

Pada inisialisasi awal atau iterasi pertama, kecepatan (*V*) setiap partikel bernilai 0 (nol). Dapat dilihat pada Tabel 4.5.

Tabel 4.5 Inisialisasi Kecepatan

Iterasi =0												
V1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0
V2	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0
V3	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0

4.3.1.2 Inisialisasi Partikel dan Hitung Fitness

Masukkan ID siswa terpilih pada setiap kloter di atas akan dikonversi ke dalam angka permutasi untuk memudahkan pencarian solusi. Pengkonversian dilakukan dengan ID pertama mendapat angka permutasi 1 dan seterusnya hingga sejumlah ID siswa yang terpilih. Tabel 4.6 menunjukkan contoh sederhana representasi ID siswa ke dalam angka permutasi yang lebih lengkapnya bisa dilihat di Lampiran B.2.

Tabel 4.6 Konversi ID Permutasi

Kloter Pertama																	
ID	19	14	18	15	17	11	13	9	20	10	2	3	4	5	..	..	8
Permutasi	1	2	3	4	5	6	7	8	9	10	11	12	13	14	..	..	17
Kloter kedua																	
ID	13	14	11	12	19	10	9	8	7	24	20	23	1	2	..	..	17
Permutasi	1	2	3	4	5	6	7	8	9	10	11	12	13	14	..	..	20

Pembangkitan partikel secara acak akan dilakukan sesuai data masukkan pada parameter PSO pada Tabel 4.4. Terdapat 3 partikel pada masing-masing kloter yang bisa dilihat pada Tabel 4.7 dan 4.8 yang lebih lengkapnya bisa dilihat di Lampiran B.3.

Tabel 4.7 Pembentukan Partikel Kloter Pertama

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	17	S
X2	S	11	12	13	14	15	16	17	8	10	6	7	2	..	..	9	S
X3	S	9	1	3	5	4	2	7	6	8	10	17	16	..	..	11	S

Tabel 4.8 Pembentukan Partikel Kloter Kedua

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	13	..	..	16	S
X2	S	13	14	16	17	18	7	8	9	6	3	4	1	2	..	..	11	S
X3	S	10	12	15	19	11	5	20	2	1	4	6	3	7	..	..	18	S

Nilai *S* di awal dan di akhir mewakili sekolah, dikarenakan titik awal pemberangkatan dan titik akhir tujuan adalah sekolah itu sendiri. Jadi pengacakan pada dimensi hanya berlaku untuk ID siswa terpilih yang sudah dipermutasikan seperti di atas. Untuk selanjutnya akan dilakukan perhitungan *fitness* pada setiap partikel. *Fitness* didapatkan dari Persamaan 2.5 yang membutuhkan total jarak dari setiap partikel. Tabel 4.9 dan 4.10 merupakan *fitness* dari partikel pada setiap kloter yang lebih lengkapnya bisa dilihat di Lampiran B.4.

Contoh perhitungan Total Jarak:

X1	S	1	2	3	4	5	6	7	8	9	10	11	..	..	17	S	Total jarak
Jarak		3	3,6	0	0,2	0,9	1,5	2,7	1,4	2,3	1,4	0,4	..	..	0,1	1,8	22,58

$$Fitness = \frac{1}{Total\ Jarak}$$

$$Fitness = \frac{1}{22{,}58} = 0{,}04429$$

Tabel 4.9 Nilai *Fitness* Partikel Kloter Pertama

Partikel	Total Jarak	*Fitness*
X1	22,58	0,04429
X2	35,05	0,02853
X3	32,21	0,03105

Tabel 4.10 Nilai *Fitness* Partikel Kloter Kedua

Partikel	Total Jarak	*Fitness*
X1	33,27	0,03005
X2	49,17	0,02033
X3	50,06	0,01997

4.3.1.3 Inisialisasi Pbest dan Gbest

Pada awal iterasi, nilai *Pbest* disamakan dengan posisi awal partikel dan nilai *Gbest* diperoleh dari *Pbest* dengan nilai *fitness* yang tertinggi. Tabel 4.11 dan 4.12 menunjukkan nilai *Pbest* dan *Gbest* pada setiap kloter yang lebih lengkapnya bisa dilihat di Lampiran B.5.

Tabel 4.11 *Pbest* dan *Gbest* Partikel Kloter Pertama

Pbest	Nilai dimensi																	*Fitness*
X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	17	S	0,04429
X2	S	11	12	13	14	15	16	17	8	10	6	7	2	..	..	9	S	0,02853
X3	S	9	1	3	5	4	2	7	6	8	10	17	16	..	..	11	S	0,03105
Gbest	Nilai dimensi																	*Fitness*
X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	17	S	0,04429

Tabel 4.12 *Pbest* dan *Gbest* Partikel Kloter kedua

Pbest	Nilai Dimensi																	*Fitness*
X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	20	S	0,03005
X2	S	13	14	16	17	18	7	8	9	6	3	4	1	..	..	10	S	0,02033
X3	S	10	12	15	19	11	5	20	2	1	4	6	3	..	..	13	S	0,01997
Gbest	Nilai Dimensi																	*Fitness*
X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	20	S	0,03005

4.3.2 *Update* Kecepatan

Dalam menghitung kecepatan, perlu diketahui nilai *w* terlebih dahulu yang bisa didapatkan dengan menggunakan Persamaan 2.3.

Berdasarkan nilai masukkan parameter PSO pada Tabel 4.3, diketahui *Wmin*=0.4, *Wmax*=0.9, itermax=2, *rand1*=[0,1], *rand2*=[0,1], C1 & C2= 1, jmlPartikel=3, maka nilai *W* pada *iterasi* ke-1 adalah:

$$W = 0.9 - \frac{0.9 - 0.4}{2} x\ 1 = 0.65$$

Misalkan diambil contoh partikel *X2* pada masing-masing kloter. Maka nilai dimensi masing-masing kloter adalah 11 dan 13. Untuk perubahan kecepatan, maka nilai perpindahan masing-masing partikel adalah:

kecepatan dimensi ke-1 pada *X2* kloter pertama *iterasi* ke-1:

$$V_{2,1}^{1} = WV_{2,1}^{0} + C_1 rand_1 \times \left(Pbest_{1,1} - X_{2,1}^{0}\right) + C_2 rand_2 \times \left(Gbest_{1,1} - X_{2,1}^{0}\right)$$
$$= 0.65 * 0 + 1 * 0.9 \times (11 - 11) + 1 * 0.3 \times (1 - 11) = -3$$

kecepatan dimensi ke-1 pada *X2* kloter kedua *iterasi* ke-1:

$$V_{2,1}^{1} = WV_{2,1}^{0} + C_1 rand_1 \times \left(Pbest_{1,1} - X_{2,1}^{0}\right) + C_2 rand_2 \times \left(Gbest_{1,1} - X_{2,1}^{0}\right)$$
$$= 0.65 * 0 + 1 * 0.9 \times (13 - 13) + 1 * 0.3 \times (1 - 13) = -3.6$$

Tabel 4.13 dan 4.14 merupakan hasil dari *Update* kecepatan semua dimensi dalam partikel pada masing-masing kloter saat iterasi =1. Data lengkapnya bisa dilihat pada lampiran.

Tabel 4.13 *Update* Kecepatan Kloter Pertama

V1	S	0	0	0	0	0	0	0	0	0	..	..	0	S
V2	S	-3,0	-3,0	-3,0	-3,0	-3,0	-3,0	-3,0	0,0	-0,3	..	..	2,4	S
V3	S	-2,4	0,3	0,0	-0,3	0,3	1,2	0,0	0,6	0,3	..	..	1,8	S

Tabel 4.14 *Update* Kecepatan Kloter Kedua

V1	S	0	0	0	0	0	0	0	0	0	..	..	0	S
V2	S	-3,6	-3,6	-3,9	-3,9	-3,9	-0,3	-0,3	-0,3	0,9	..	..	3	S
V3	S	-2,7	-3	-3,6	-4,5	-1,8	0,3	-3,9	1,8	2,4	..	..	2,1	S

4.3.3 *Update* Posisi

Dalam menghitung *Update* posisi dapat menggunakan Persamaan 2.4 yang telah dibahas pada bab sebelumnya. Berikut merupakan contoh perhitungan *Update* posisi pada partikel *X2* masing-masing kloter:

Update posisi dimensi ke-1 pada *X2* kloter pertama *iterasi* ke-1:

$$X_j^{k+1} = X_j^k + V_j^{k+1}$$
$$X_{2,1}^{0+1} = X_{2,1}^0 + V_{2,1}^{0+1} = 11 + (-3) = 8$$

Update posisi dimensi ke-1 pada *X2* kloter kedua *iterasi* ke-1:

$$X_j^{k+1} = X_j^k + V_j^{k+1}$$

$$X_{2,1}^{0+1} = X_{2,1}^0 + V_{2,1}^{0+1} = 13 + (-3.6) = 9$$

Tabel 4.15 dan 4.16 menunjukkan *Update* posisi partikel pada masing-masing kloter. Data lengkapnya bisa dilihat pada Lampiran B.7.

Tabel 4.15 *Update* posisi Kloter Pertama

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	17	S
X2	S	8	9	10	11	12	13	14	8	10	7	8	5	..	..	11	S
X3	S	7	1	3	5	4	3	7	7	8	10	15	15	..	..	13	S

Tabel 4.16 *Update* posisi Kloter Kedua

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	13	..	..	20	S
X2	S	9	10	12	13	14	7	8	9	7	5	6	4	5	..	..	13	S
X3	S	7	9	11	15	9	5	16	4	3	6	8	6	9	..	..	15	S

4.3.3.1 Repair

Perpindahan dimensi partikel tidak selalu tepat sesuai permutasi yang didefinisikan sebelumnya. Oleh karena itu akan dilakukan perbaikan. Nilai dimensi yang tidak sesuai akan diganti dengan nilai permutasi yang tidak ada pada set kesatuan permutasi ID siswa. Tabel 4.17 dan 4.18 menunjukkan contoh pengecekkan dan perbaikan dimensi partikel. Warna kuning merupakan nilai dimensi yang salah dan warna hijau merupakan nilai baru dimensi setelah perbaikan. Data lengkapnya bisa dilihat pada Lampiran B.8.

Tabel 4.17 *Repair* Kloter Pertama

Sebelum

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	17	S
X2	S	8	9	10	11	12	13	14	8	10	7	8	5	..	..	11	S
X3	S	7	1	3	5	4	3	7	7	8	10	15	15	..	..	13	S

Sesudah

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	..	..	17	S
X2	S	8	9	10	11	12	13	14	1	2	7	4	5	..	..	3	S
X3	S	7	1	3	5	4	6	2	9	8	10	15	11	..	..	17	S

Tabel 4.18 *Repair* Kloter kedua

Sebelum

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	13	..	..	20	S
X2	S	9	10	12	13	14	7	8	9	7	5	6	4	5	..	..	13	S
X3	S	7	9	11	15	9	5	16	4	3	6	8	6	9	..	..	15	S

Sesudah

X1	S	1	2	3	4	5	6	7	8	9	10	11	12	13	..	..	20	S
X2	S	9	10	12	13	14	7	8	1	11	5	6	4	17	..	..	20	S
X3	S	7	9	11	15	1	5	16	4	3	6	8	18	13	..	..	12	S

4.3.3.2 Hitung *Fitness*

Perpindahan partikel menyebabkan nilai *fitness* juga berubah. Perubahan yang terjadi tidak selalu menghasilkan nilai *fitness* yang lebih baik dari pada sebelumnya. Jika nilai *fitness* lebih tinggi dari sebelumnya, maka nilai *Pbest* dan *Gbest* akan diperbarui. Tabel 4.19 dan 4.20 merupakan perbaruan nilai *fitness* setelah *Update* posisi. Data lengkapnya bisa dilihat pada Lampiran B.9.

Tabel 4.19 Perbaruan Nilai *Fitness* Kloter Pertama

X1	S	1	2	3	4	5	6	7	8	..	..	17	S	0,0443
X2	S	8	9	10	11	12	13	14	1	..	..	3	S	0,0291
X3	S	7	1	3	5	4	6	2	9	..	..	17	S	0,0288

Tabel 4.20 Perbaruan Nilai *Fitness* Kloter Kedua

X1	S	1	2	3	4	5	6	7	8	..	..	20	S	0,0301
X2	S	9	10	12	13	14	7	8	1	..	..	20	S	0,0186
X3	S	7	9	11	15	1	5	16	4	..	..	12	S	0,0162

4.3.4 *Update Pbest* dan *Gbest*

Personal Best dan *Global Best* akan selalu diperbarui. Partikel dalam iterasi terbaru dengan nilai *fitness* yang lebih baik dari sebelumnya akan menggantikan partikel lama untuk dipakai dalam iterasi selanjutnya. Hasil dari Tabel 4.21 dan 4.22 menunjukkan bahwa partikel *X2* pada kloter pertama dengan warna hijau muda memiliki nilai *fitness* yang lebih baik dari iterasi sebelumnya. Sedangkan untuk partikel iterasi 1 pada kloter kedua tidak menunjukkan adanya peningkatan nilai *fitness*. Data lengkapnya bisa dilihat pada Lampiran B.10.

Tabel 4.21 Perbandingan Nilai *Fitness* Kloter Pertama

Iterasi 0

X1	S	1	2	3	4	5	6	7	8	9	..	..	17	S	0,0443
X2	S	11	12	13	14	15	16	17	8	10	..	..	9	S	0,0285
X3	S	9	1	3	5	4	2	7	6	8	..	..	11	S	0,0311

Iterasi 1

X1	S	1	2	3	4	5	6	7	8	9	..	..	17	S	0,0443
X2	S	8	9	10	11	12	13	14	1	2	..	..	3	S	0,0291
X3	S	7	1	3	5	4	6	2	9	8	..	..	17	S	0,0288

Tabel 4.22 Perbandingan Nilai *Fitness* Kloter Kedua

Iterasi 0

X1	S	1	2	3	4	5	6	7	8	9	..	..	20	S	0,0301
X2	S	13	14	16	17	18	7	8	9	6	..	..	10	S	0,0203
X3	S	10	12	15	19	11	5	20	2	1	..	..	13	S	0,0199

Iterasi 1

X1	S	1	2	3	4	5	6	7	8	9	..	..	20	S	0,0301
X2	S	9	10	12	13	14	7	8	1	11	..	..	20	S	0,0186
X3	S	7	9	11	15	1	5	16	4	3	..	..	12	S	0,0162

Pembaruan nilai *Pbest* dan *Gbest* perlu dilakukan sebagai acuan untuk perhitungan iterasi selanjutnya. Tabel 4.23 dan 4.24 akan menunjukkan pembaruan *Pbest* dan *Gbest* yang akan digunakan untuk iterasi kedua. Data lengkapnya bisa dilihat pada Lampiran B.11.

Tabel 4.23 *Pbest* dan *Gbest* Iterasi 2 Kloter Pertama

Pbest	Nilai Dimensi														Fitness
X1	S	1	2	3	4	5	6	7	8	9	..	..	17	S	0,0443
X2	S	8	9	10	11	12	13	14	1	2	..	..	3	S	0,0291
X3	S	9	1	3	5	4	2	7	6	8	..	..	11	S	0,0311
Gbest	Nilai Dimensi														Fitness
X1	S	1	2	3	4	5	6	7	8	9	..	..	17	S	0,0443

Tabel 4.24 *Pbest* dan *Gbest* Iterasi 2 Kloter Kedua

Pbest	Nilai Dimensi														Fitness
X1	S	1	2	3	4	5	6	7	8	9	..	..	20	S	0,0301
X2	S	13	14	16	17	18	7	8	9	6	..	..	10	S	0,0203
X3	S	10	12	15	19	11	5	20	2	1	..	..	13	S	0,0199
Gbest	Nilai Dimensi														Fitness
X1	S	1	2	3	4	5	6	7	8	9	..	..	20	S	0,0301

Namun jika iterasi dihentikan hanya pada iterasi 1, maka nilai *Gbest* akan menjadi solusi terbaik yang dihasilkan. Berikut merupakan hasil optimasi jika *itermax*=1. Dapat dilihat pada Tabel 4.25 dan 4.26. Data lengkapnya bisa dilihat pada Lampiran B.12.

Tabel 4.25 Hasil Optimasi Kloter Pertama

Urutan pengantaran (kiri ke kanan)											
ID	S	19	14	18	15	17	11	13	..	..	S
Nama	Sekolah	atiq	alissa	firli	melissa	arin	arum	syafa	..	..	Sekolah

Tabel 4.26 Hasil Optimasi Kloter Kedua

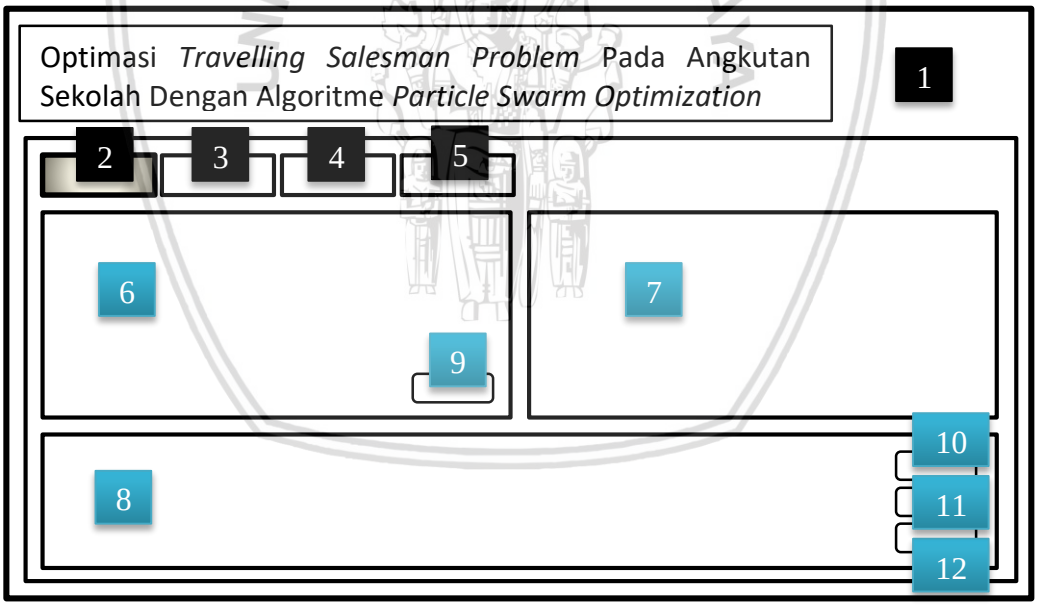
Urutan pengantaran (kiri ke kanan)											
ID	S	13	14	11	12	19	10	9	..	..	S
Nama	Sekolah	indah	hanum	najib	syifa'i	ririn	asfira	fauzi	..	..	Sekolah

4.4 Perancangan Antar Muka

Perancangan Antarmuka merupakan perancangan untuk komunikasi antara *system* dengan *user*. Pada penelitian ini, penulis membagi menjadi 4 halaman antar muka yang terdiri dari Halaman Depan atau Halaman Masukkan, Halaman Data, Halaman Proses dan Halaman Hasil. Berikut merupakan pembahasan masing-masing Halaman antarmuka:

4.4.1 Halaman Depan

Halaman depan adalah halaman yang selalu ditampilkan secara *Default* sebagai halaman awal ketika program dijalankan. Di dalamnya berisi masukkan ID masing-masing kloter beserta parameter PSO yang akan digunakan untuk proses perhitungan. Oleh karena itu, Halaman Depan juga bisa disebut sebagai Halaman Masukkan. Gambar 4.9 menunjukkan perancangan antarmuka Halaman depan.



Gambar 4.9 Perancangan Halaman Depan

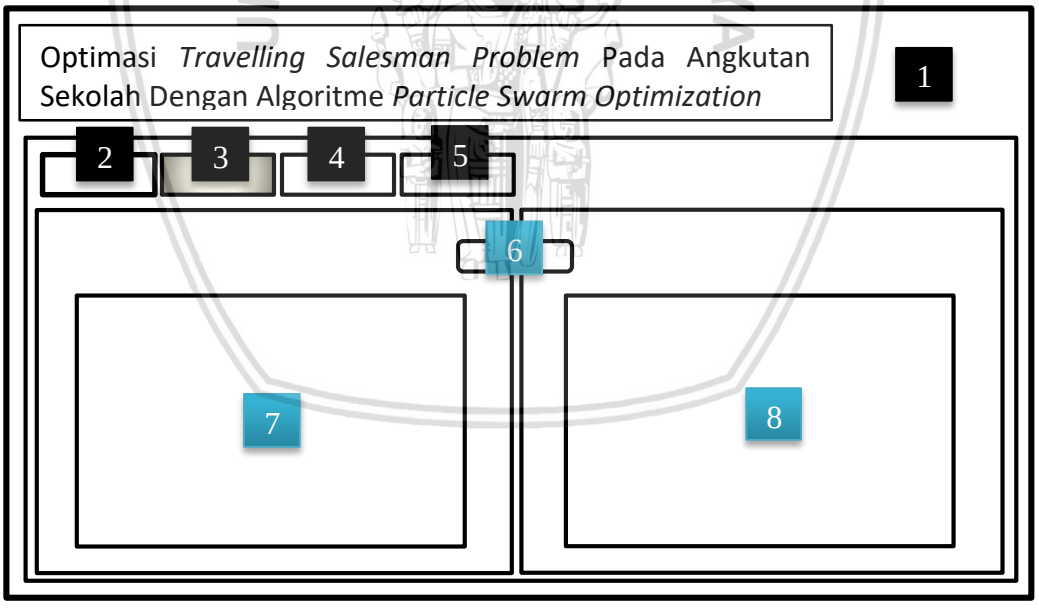
Keterangan:

1. Judul Sistem dari penelitian.
2. Tab “Data Masukkan” merupakan Tab yang aktif saat ini untuk menampilkan Halaman Depan.
3. Tab “Data” untuk menampilkan Halaman Data.
4. Tab “Proses” untuk menampilkan Halaman Proses.

5. Tab “Hasil” untuk menampilkan Halaman Hasil.
6. Pannel untuk data masukkan ID dan nama siswa pada kloter pertama.
7. Pannel untuk data masukkan ID dan nama siswa pada kloter pertama.
8. Pannel untuk data masukkan Parameter PSO yang akan digunakan untuk proses optimasi, Parameter yang digunakan adalah: Jumlah Partikel, C1, C2, *Wmin*, *Wmax* dan ITERMAX.
9. *Button* “*Default*” untuk memilih ID siswa pada setiap kloter yang sudah disediakan oleh sistem.
10. *Button* “*START*” digunakan untuk memulai proses perhitungan.
11. *Button* “*Default*” untuk menampilkan parameter PSO yang sudah disediakan oleh sistem.
12. *Button* “*Reset*” untuk menghapus semua record data masukkan, parameter PSO dan proses yang sudah dijalankan.

4.4.2 Halaman Data

Halaman Data adalah halaman yang akan menampilkan data siswa pada masing-masing kloter. Data tersebut meliputi ID, Nama panggilan, Kelas dan Alamat rumah siswa. Gambar 4.10 menunjukkan perancangan antarmuka Halaman Data.



Gambar 4.10 Halaman Data

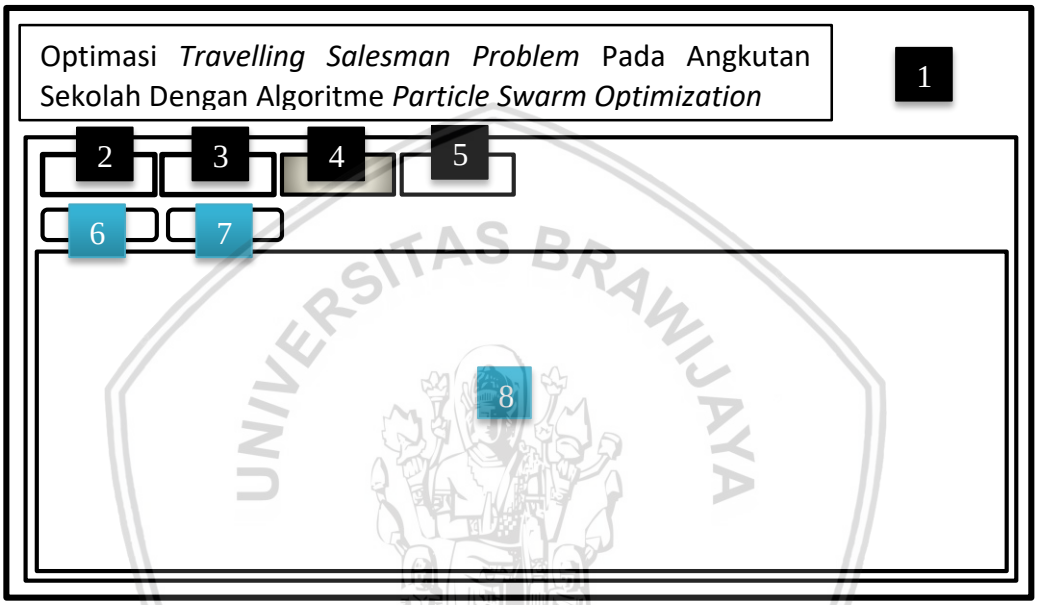
Keterangan:

1. Judul Sistem dari penelitian.
2. Tab “Data Masukkan” untuk menampilkan Halaman Depan.
3. Tab “Data” merupakan Tab yang aktif saat ini untuk menampilkan Halaman Data.
4. Tab “Proses” untuk menampilkan Halaman Proses.

5. Tab “Hasil” untuk menampilkan Halaman Hasil.
6. *Button* “Tampil Data” untuk menampilkan data siswa-siswi kloter pertama.
7. Pannel Data siswa-siswi kloter pertama dalam bentuk Tabel.
8. Pannel Data siswa-siswi kloter kedua dalam bentuk Tabel.

4.4.3 Halaman Proses

Halaman Proses adalah halaman yang akan menampilkan proses perhitungan optimasi pada masing-masing kloter. Gambar 4.11 menunjukkan perancangan antarmuka Halaman Proses.



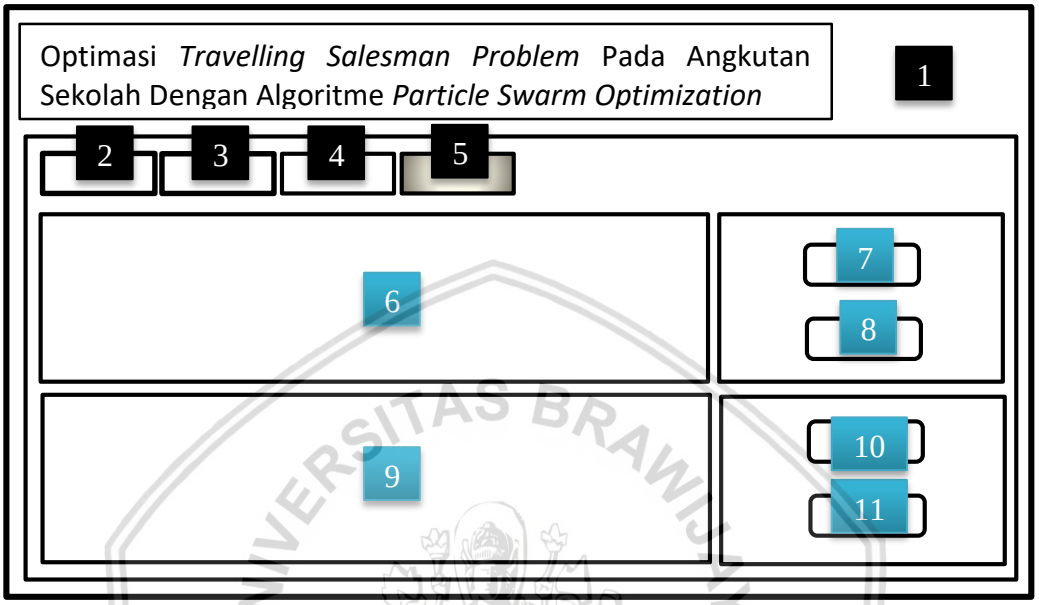
Gambar 4.11 Halaman Proses

Keterangan:

1. Judul Sistem dari penelitian.
2. Tab “Data Masukkan” untuk menampilkan Halaman Depan.
3. Tab “Data” untuk menampilkan Halaman Data.
4. Tab “Proses” merupakan Tab yang aktif saat ini untuk menampilkan Halaman Proses.
5. Tab “Hasil” untuk menampilkan Halaman Hasil.
6. Tab “Kloter Pertama” untuk menampilkan proses optimasi pada kloter pertama.
7. Tab “Kloter Kedua” untuk menampilkan proses optimasi pada kloter kedua.
8. *Text Field* untuk menampilkan proses perhitungan optimasi dari kloter pertama dan kloter kedua.

4.4.4 Halaman Hasil

Halaman Hasil adalah halaman yang akan menampilkan hasil proses optimasi. Hasil yang dikeluarkan oleh sistem ini adalah urutan rute pengantaran siswa dengan nilai *fitness* terbaik pada masing-masing kloter. Gambar 4.12 menunjukkan perancangan antarmuka Halaman Hasil.



Gambar 4.12 Halaman Hasil

Keterangan:

1. Judul Sistem dari penelitian.
2. Tab "Data Masukkan" untuk menampilkan Halaman Depan.
3. Tab "Data" untuk menampilkan Halaman Data.
4. Tab "Proses" untuk menampilkan Halaman Proses.
5. Tab "Hasil" merupakan Tab yang aktif saat ini untuk menampilkan Halaman Hasil.
6. *Text Field* untuk menampilkan hasil optimasi yang berupa urutan rute pengantaran siswa pada Kloter pertama.
7. *Text Field* untuk menampilkan fitness terbaik pada kloter pertama.
8. *Text Field* untuk menampilkan Total Jarak terpendek pada kloter pertama.
9. *Text Field* untuk menampilkan hasil optimasi yang berupa urutan rute pengantaran siswa pada Kloter kedua.
10. *Text Field* untuk menampilkan fitness terbaik pada kloter kedua.
11. *Text Field* untuk menampilkan Total Jarak terpendek pada kloter kedua.

4.5 Perancangan Pengujian

Perancangan pengujian akan dilakukan untuk mendapatkan parameter Algoritme PSO yang optimal dalam menyelesaikan permasalahan optimasi rute terpendek angkutan sekolah ini. Terdapat 4 pengujian yang akan dilakukan, diantaranya adalah:

1. Pengujian Jumlah Partikel
2. Pengujian kombinasi *Wmin* dan *Wmax*
3. Pengujian Koefisien Akselerasi
4. Pengujian Konvergensi

Pengujian akan dilakukan setelah sistem selesai diimplementasikan. Berikut merupakan penjelasan beberapa Tabel rancangan pengujian:

4.5.1 Pengujian Jumlah Partikel

Pengujian jumlah partikel dilakukan sebanyak 10 kali berdasarkan jumlah partikel dan lamanya waktu komputasi. Tabel 4.27 menunjukkan perancangan pengujian untuk jumlah partikel.

Tabel 4.27 Perancangan Pengujian Jumlah Partikel

Jumlah Partikel	Percobaan Ke-*i*							Rata-rata
	1	2	3	4	5	..	10	
	Fitness	*Fitness*	*Fitness*	*Fitness*	*Fitness*	..	*Fitness*	*Fitness*

4.5.2 Pengujian Kombinasi *Wmin* dan *Wmax*

Pengujian Kombinasi *Wmin* dan *Wmax* digunakan untuk mengetahui nilai bobot inersia (*W*) yang optimal untuk proses perhitungan dengan nilai [0,1]. Pengujian akan dilakukan dengan sepuluh kali percobaan berdasarkan rata-rata *fitness* yang dihasilkan. Tabel 4.28 menunjukkan perancangan Pengujian pada *Wmin* dan *Wmax.*

Tabel 4.28 Perancangan Pengujian Kombinasi *Wmin* dan *Wmax*

Bobot Inersia		Nilai *Fitness*					Rata-rata *Fitness*
Wmax	*Wmin*	Percobaan ke-*i*					
		1	2	3	..	10	
0,9	0,2						
	0,3						
	0,4						
0,8	0,2						
	0,3						
	0,4						
0,7	0,2						
	0,3						
	0,4						

4.5.3 Pengujian Koefisien Akselerasi

Pengujian koefisien akselerasi bertujuan untuk mengetahui kombinasi nilai koefisien akselerasi optimal dalam menentukan solusi terbaik yang akan dihasilkan oleh sistem. Pengujian akan dilakukan dengan sepuluh kali percobaan berdasarkan rata-rata *fitness* yang dihasilkan. Tabel 4.29 menunjukkan perancangan pengujian yang akan dilakukan.

Tabel 4.29 Perancangan Pengujian Koefisien Akselerasi

Koefisien Akselersi		Nilai *Fitness*						Rata-rata *Fitness*
C_1	C_2	Percobaan ke-*i*						
		1	2	3	4	..	10	
1	1					..		
	1,5					..		
	2					..		
1,5	1					..		
	1,5					..		
	2					..		
2	1					..		
	1,5					..		
	2					..		

4.5.4 Pengujian Konvergensi

Pengujian Konvergensi bertujuan untuk mengetahui kinerja maksimal sistem dalam menghasilkan solusi. Pengujian ini dilakukan dengan menganalisis hasil stagnasi nilai solusi berdasarkan iterasi. Pengujian akan dilakukan dengan lima kali percobaan berdasarkan rata-rata *fitness* yang dihasilkan. Tabel 4.30 menunjukkan perancangan pengujian yang akan dilakukan.

Tabel 4.30 Perancangan Pengujian Koefisien Akselerasi

Iterasi ke-	Percobaan Ke-*i*				
	1	2	3	4	5
	Fitness	*Fitness*	*Fitness*	*Fitness*	*Fitness*
1					
4					
8					
12					
16					
20					
25					
30					
35					
40					
45					
50					

BAB 5 IMPLEMENTASI

5.1 Implementasi Program

Subbab ini akan menjelaskan tentang tahapan implementasi sistem yang telah dirancang hingga dapat menghasilkan optimasi rute terpendek dari angkutan sekolah. Implementasi program menggunakan bahasa pemrograman JAVA yang meliputi *Import* data, Inisialisasi Partikel, Inisialisasi Kecepatan Awal, Perhitungan *Fitness*, *Update* Kecepatan, *Update* Posisi, *Update Pbest* dan *Gbest*.

5.1.1 *Import* Data

Data yang dimaksud di sini adalah data ID siswa yang meliputi Nama, Kelas dan Matriks jarak alamat siswa satu dengan lainnya pada setiap kloter. Data tersebut tersimpan pada file Ms. Excel yang nantinya akan di*import* dengan kode program agar dapat diambil nilainya untuk menjadi bagian dalam proses optimasi. Kode Program 5.1 menjelaskan implementasi *Import* data.

Baris	Kode Program
1	`public void ImportExcel() throws IOException {`
2	`  File fileExcel = new File(fileInput);`
3	`  Workbook w;`
4	`  try {`
5	`    w = Workbook.getWorkbook(fileExcel);`
6	
7	`    Sheet sheetPagi = w.getSheet(0);`
8	`    for(int j = 0; j<sheetPagi.getColumns(); j++){`
9	`      for(int i = 0; i<sheetPagi.getRows()-1; i++) {`
10	`        Cell cell = sheetPagi.getCell(j, i);`
11	`        jarakPagi[j][i]=`
12	`        Double.parseDouble(cell.getContents());`
13	`      }`
14	`    }`
15	
16	`    Sheet sheetSiang = w.getSheet(1)`
17	`    for(int j = 0; j<sheetSiang.getColumns(); j++) {`
18	`      for(int i = 0; i<sheetSiang.getRows()-1; i++){`
19	`        Cell cell2 = sheetSiang.getCell(j, i);`
20	`        jarakSiang[j][i]=`
21	`        Double.parseDouble(cell2.getContents());`
22	`      }`
23	`    }`
24	
25	`    Sheet sheetIdPagi = w.getSheet(2);`
26	`    for(int j = 0; j<sheetIdPagi.getColumns(); j++){`
27	`      for(int i = 0; i<sheetIdPagi.getRows(); i++){`
28	`        Cell cell3 = sheetIdPagi.getCell(j, i);`
29	`        if(j == 0) {`
30	`          kodePagi[j][i]=`
31	`          Integer.parseInt(cell3.getContents());`
32	`        }else if (j == 1){`

```
33              kodePagi[j][i] = cell3.getContents();
34            }else if (j == 2){
35              kodePagi[j][i]=
36              Integer.parseInt(cell3.getContents());
37            }
38          }
39        }
40
41        Sheet sheetIdSiang = w.getSheet(3);
42        for(int j = 0; j<sheetIdSiang.getColumns(); j++){
43          for(int i = 0; i<sheetIdSiang.getRows(); i++){
44            Cell cell4 = sheetIdSiang.getCell(j, i);
45            if(j == 0) {
46              kodeSiang[j][i]=
47              Integer.parseInt(cell4.getContents());
48            }else if (j == 1){
49              kodeSiang[j][i] = cell4.getContents();
50            }else if (j == 2){
51              kodeSiang[j][i]=
52              Integer.parseInt(cell4.getContents());
53            }
54          }
55        }
56      }catch (BiffException e) {
57           e.printStackTrace();
58      }
59    }
```

Kode Program 5.1 *Import* Data

Pembahasan Kode Program 5.1 *Import* Data:

Baris 7-23 merupakan kode program yang berfungsi untuk mengambil data matriks jarak kloter pertama dan kedua dari file Ms. Excel. Baris 7 dan 16 berfungsi mengambil data lembar halaman satu dan dua pada file Ms. Excel untuk masing-masing matriks jaraknya.

Baris 25-55 merupakan kode program yang berfungsi untuk mengambil data ID siswa pada masing-masing kloter dari file Ms. Excel. Baris 25 dan 41 berfungsi mengambil data lembar halaman tiga dan empat pada file Ms. Excel yang berisi ID siswa dengan Nama dan Kelas pada masing-masing kloternya.

5.1.2 Inisialisasi Partikel

Inisialisasi partikel merupakan tahap awal penyelesaian masalah dengan PSO. Pembangkitan partikel akan dilakukan secara acak dengan angka permutasi seperti yang telah dibahas pada bab sebelumnya. Kode Program 5.2 membahas implementasi dari rancangan inisialisasi partikel.

Baris	Kode Program

1
2
3
	`public void setPartikel(int iterasi) {`
	`  this.partikel=new double[popSize][pjgPartikel];`
	`  Random r = new Random();`

```
4    int batas = pjgPartikel - 1;
5    for (int i = 0; i < popSize; i++) {
6       int temp = 1;
7       partikel[i][0] = 0;
8       for (int j = 1; j <= (pjgPartikel - 2); j++) {
9          int k;
10         boolean cekId = false;
11         double id;
12         do{
13            id = r.nextInt(batas - 1) + 1;
14            for (k = 1; k < temp; k++){
15               if(partikel[i][k] == id){
16                  cekId = true;
17                  break;
18               } else if (k == temp - 1){
19                  cekId = false;
20               }
21            }
22
23         }while (cekId);
24                  temp = temp + 1;
25                  partikel[i][j] = id;
26      }
27      partikel[i][pjgPartikel - 1] = 0;
28   }
29
30    Pbest = partikel;
31    Gbest = new double[iterasi];
32    kecepatan=new
33    double[iterasi][popSize][pjgPartikel];
34
35   }
```

Kode Program 5.2 Inisialisasi Partikel

Pembahasan Kode Program 5.2 Inisialisasi Partikel:

Baris 5-28 merupakan kode program untuk pelakukan perulangan sejumlah banyak partikel yang akan dibangkitkan sesuai dengan masukkan.

Baris 8-26 merupakan perulangan sebanyak panjang partikel -2 yang berfungsi untuk membangkitkan nilai dimensi partikel secara acak.

Baris 14-21 merupakan perulangan untuk melakukan pengecekan nilai dimensi partikel agar menghindari nilai yang sama pada dimensi setiap partikel.

5.1.3 Inisialisasi Kecepatan Awal

Inisialisasi kecepatan awal sebelum masuk iterasi selalu diberi nilai=0 pada seluruh dimensi setiap partikel. Kode Program 5.3 membahas implementasi inisialisasi kecepatan awal.

Baris	Kode Program
1	`public void setKecepatan() {`
2	`   for (int i = 0; i < popSize; i++) {`
3	`      for (int j = 0; j < pjgPartikel; j++) {`
4	`          kecepatan[0][i][j] = 0;`
5	`      }`
6	`   }`
7	`}`

Kode Program 5.3 Inisialisasi Kecepatan Awal

Pembahasan Kode Program 5.3 Inisialisasi Kecepatan Awal:

Baris 2-6 merupakan perulangan sebanyak partikel yang dibangkitkan.

Baris 3-5 merupakan perulangan sebanyak panjang nilai dimensi setiap partikel.

Baris 4 merupakan pemberian nilai 0 pada dimensi setiap partikel.

5.1.4 Hitung *Fitness*

Nilai *Fitness* setiap partikel akan berbeda. Perhitungan nilai *Fitness* seperti yang dijelaskan pada bab sebelumnya membutuhkan total jarak dari setiap partikel. Kode Program 5.4 membahas implementasi perancangan hitung *Fitness*.

Baris	Kode Program
1	`public double[] hitung*Fitness*(double[][] jarak, int[]`
2	`dataAnak) {`
3	`  for (int i=0; i < popSize; i++) {`
4	`     double totalJarak = 0;`
5	`     for (int j=0; j < (partikel[0].length-1); j++) {`
6	`       totalJarak = totalJarak + jarak[dataAnak[(int)`
7	`       partikel[i][j]]][dataAnak[(int)  partikel[i][j`
8	`       + 1]]];`
9	`     }`
10	`          *fitness*[i] = 1 / totalJarak;`
11	`  }`
12	`               return *fitness*;`
13	`}`

Kode Program 5.4 Hitung *Fitness*

Pembahasan Kode Program 5.4 Hitung *Fitness*:

Baris 3-11 merupakan perulangan sebanyak jumlah partikel untuk mencari nilai *Fitness* masing-masing.

Baris 5-9 merupakan perulangan sejumlah panjang tiap partikel untuk mencari total jarak dengan menjumlahkan data jarak sekolah ke ID siswa pertama hingga terakhir lalu kembali ke sekolah lagi.

Baris 10 merupakan deklarasi perhitungan nilai *Fitness*.

Baris 12 digunakan untuk mengembalikan nilai variabel *Fitness*.

5.1.5 *Update* Kecepatan

Update Kecepatan digunakan untuk perpindahan partikel dalam pencarian solusi setiap iterasinya. Dalam pengimplementasian, *Update* Kecepatan memerlukan parameter indeks sebagai iterasi serta *C*1 dan *C*2. Kode Program 5.5 membahas implementasi perancangan dari *Update* Kecepatan.

Baris	Kode Program
1	`public void UpdateKecepatan(int indeks, double c1,`
2	`double c2) {`
3	`    titikGbest = (int) Gbest[indeks - 1];`
4	`    double w = Wmax - (((Wmax - Wmin) / iterasi)`
5	`    * indeks);`
6	`    for (int i = 0; i < popSize; i++) {`
7	`       for (int j = 0; j < pjgPartikel; j++) {`
8	`          kecepatan[indeks][i][j]=w*  kecepatan`
9	`          [indeks - 1][i][j] + (c1 * r.nextDouble() *`
10	`          (Pbest[i][j] - partikel[i][j])) + (c2 *`
11	`          r.nextDouble() * (Pbest[titikGbest][j] -`
12	`          partikel[i][j]));`
13	
14	`       }`
15	`    }`
16	`}`

Kode Program 5.5 *Update* Kecepatan

Pembahasan Kode Program 5.5 *Update* Kecepatan:

Baris 4 merupakan deklarasi nilai *W* yang selalu berubah pada setiap iterasi sesuai dengan Persamaan yang dibahas pada bab sebelumnya.

Baris 6-15 merupakan perulangan sejumlah banyak partikel yang telah dibangkitkan sesuai dengan masukkan.

Baris 7-14 merupakan perulangan sebanyak panjang partikel yang nilai dimensinya akan diperbarui kecepatannya.

Baris 8-12 merupakan pendeklarasian rumus *Update* Kecepatan sesuai dengan Persamaan yang dibahas sebelumnya.

5.1.6 *Update* Posisi

Update Posisi merupakan perpindahan posisi partikel hasil dari *Update* kecepatan. Dalam pengimplementasiannya, *Update* Posisi memerlukan parameter indeks sebagai iterasi dan juga kecepatan. Kode Program 5.6 membahas implementasi perancangan *Update* Posisi.

Baris	Kode Program
1	`public double[][] UpdatePosisi(int indeks, double[][]`
2	`kecepatan) {`
3	`   baru= new double[popSize][pjgPartikel];`
4	`   System.out.println("UPDATE POSISI");`

```
5       for(int i = 0; i < popSize; i++) {
6         for(int j = 0; j < partikel[0].length; j++) {
7             baru[i][j]=(kecepatan[i][j]+
8             partikel[i][j]);
9             baru[i][j]=Math.round(kecepatan[i][j]+
10            partikel[i][j]);
11        }
12        System.out.println("");
13      }
14      partikel = baru;
15      return partikel;
16  }
```

Kode Program 5.6 *Update* Posisi

Pembahasan Kode Program 5.6 *Update* Posisi:

Baris 3 merupakan deklarasi variabel *baru* untuk menyimpan partikel *baru*.

Baris 5-13 merupakan perulangan sejumlah banyak partikel yang telah dibangkitkan sesuai dengan masukkan.

Baris 6-11 merupakan panjang partikel yang nilai dimensinya akan diperbarui posisinya.

Baris 7-8 merupakan deklarasi rumus *Update* Posisi dengan menambahkan kecepatan pada nilai dimensi saat iterasi tersebut.

Baris 9-10 merupakan fungsi untuk membulatkan nilai posisi dimensi.

Baris 14-15 mengembalikan nilai *baru* dengan partikel.

5.1.7 *Repair*

Repair merupakan tahapan khusus yang diimplementasikan pada penelitian ini. *Repair* berfungsi untuk memperbaiki nilai dimensi yang terindikasi sama dengan nilai dimensi lain dalam satu partikel. Kode Program 5.7 membahas implementasi perancangan *Repair*.

Baris	Kode Program
1	`public double[][] Repair() {`
2	`  int[] jmlSama = new int[popSize];`
3	`  int[] Indekssama;`
4	`  int[] ElemenRepair;`
5	`  for (int i = 0; i < popSize; i++) {`
6	`    int temp = 0;`
7	`    for(int j = 1; j<partikel[0].length - 1; j++){`
8	`      for(int k = j+1; k<partikel[0].length-1; k++){`
9	`        if(partikel[i][j] == partikel[i][k]) {`
10	`            temp++;`
11	`            jmlSama[i] = temp;`
12	`            break;`
13	`        }`
14	`      }`
15	`    }`

```
16    }
17
18    for(int i = 0; i < popSize; i++){
19      Indekssama = new int[jmlSama[i]];
20      ElemenRepair = new int[jmlSama[i]];
21      int temp = 0;
22
23      for(int j = 1; j<partikel[0].length-1; j++){
24        for(int k = j+1; k<partikel[0].length-1; k++){
25          if(partikel[i][j] == partikel[i][k]){
26            temp++;
27            Indekssama[temp - 1] = k;
28
29             break;
30          }
31        }
32      }
33      int temp2 = 0;
34      for(int j = 1; j<=partikel[0].length - 2; j++){
35        if(check(j, partikel[i])) {
36          ElemenRepair[temp2] = j;
37          temp2++;
38        }
39      }
40
41      System.out.println("PopSize ke (" + i + ") indeks
42      replace" + Arrays.toString(Indekssama));
43      System.out.println("PopSize ke (" + i + ") Elemenz
44      Repair" + Arrays.toString(ElemenRepair));
45
46      if(Indekssama.length != 0){
47        for (int j = 0; j < Indekssama.length; j++) {
48          partikel[i][Indekssama[j]] = ElemenRepair[j];
49        }
50      }
51    }
52    return partikel;
53  }
```

Kode Program 5.7 *Repair*

Pembahasan Kode Program 5.7 *Repair*:

Baris 2 deklarasi variabel *jmlSama* sebanyak nilai variabel *popSize*.

Baris 3-4 deklarasi variabel *indekssama* dan *ElemenRepair*.

5-16 merupakan perulangan variabel *i* sejumlah banyak *popSize* yang telah dibangkitkan sesuai dengan masukkan.

Baris 6 deklarasi variabel *temp* dengan nilai 0.

Baris 7-15 merupakan perulangan *j* mulai dari *j*=1 sampai panjang partikel ke-0 sampai 1.

Baris 8-14 merupakan perulangan *k* mulai dari *j*=1 sampai panjang partikel ke-0 sampai 1.

Baris 9-13 merupakan percabangan dengan kondisi jika indeks partikel ke *i*, *j* sama dengan indeks partikel ke *i*, *k*, maka variabel *temp* nilainya akan ditambah. Lalu nilai variabel *temp* akan disimpan pada variabel *jmlSama* ke *i*. Melakukan *break*, untuk keluar dari percabangan.

Baris 18-51 merupakan perulangan variabel *i* sejumlah banyak *popSize* yang telah dibangkitkan sesuai dengan masukkan.

Baris 19 deklarasi variabel *IndeksSama* sebanyak nilai variabel *jmlSama* ke *i*.

Baris 20 deklarasi variabel *ElemenRepair* sebanyak nilai variabel *jmlSama* ke *i*.

Baris 21 deklarasi variabel *temp* dengan nilai 0.

Baris 23-33 merupakan perulangan *j* mulai dari *j*=1 sampai panjang partikel ke-0 dikurangi 1.

Baris 24-31 merupakan perulangan *k* mulai dari *j*=1 sampai panjang partikel ke-0 dikurangi 1.

Baris 25-30 merupakan percabangan dengan kondisi jika indeks partikel ke *i*, *j* sama dengan indeks partikel ke *i, k,* maka variabel *temp* nilainya akan ditambah. Lalu nilai variabel *k* akan disimpan pada variabel *IndeksSama* indeks ke nilai *temp* dikurangi 1. Melakukan *break*, untuk keluar dari percabangan.

Baris 34 deklarasi variabel *temp2* dengan nilai 0.

Baris 34-39 merupakan perulangan *j* mulai dari *j*=1 sampai panjang partikel ke-0 dikurangi 2.

Baris 35-38 merupakan percabangan dengan kondisi jika nilai kembalian dari fungsi pemanggilan fungsi *check* dengan parameter *j* dan partikel ke *i* bernilai *true,* maka variabel *temp2* nilainya akan ditambah. Lalu nilai variabel *j* akan disimpan pada variabel *ElemenRepair* indeks ke nilai temp2.

Baris 36-37 deklarasi variabel *indekssama* dan *ElemenRepair*.

Baris 41-42 menampilkan variabel Indekssama sebagai variabel yang menyimpan indeks yang akan diganti.

Baris 43-44 menampilkan variabel *ElemenRepair* sebagai variabel yang menyimpan nilai yang sebagai pengganti.

Baris 46-50 merupakan percabangan dengan kondisi jika panjang dari variabel *Indekssama* tidak bernilai 0 maka akan melakukan perulangan *j* mulai dari *j*=0 sampai panjang dari variabel *indekssama*. Kemudian, partikel dengan indeks *i* dan indeks dari nilai variabel *Indekssama* ke *j* sama dengan nilai dari variabel *ElemenRepair* ke *j.*

Baris 52 merupakan pengembalkan nilai kembalian dari Fungsi *Repair* yang disimpan dalam variable *Partikel*.

5.1.8 *Update Pbest*

Nilai *Pbest* akan selalu diperbarui pada setiap iterasi. Jika nilai *Pbest* pada iterasi sebelumnya lebih kecil dari pada nilai *Pbest* pada iterasi saat ini, maka *Pbest* akan menyimpan nilai yang lebih besar untuk digunakan pada iterasi selanjutnya. Kode Program 5.8 membahas implementasi perancangan *Update Pbest.*

Baris	Kode Program
1	`public double[] pBest(int index, double[] fitness) {`
2	`  double[] hasil = new double[popSize];`
3	`  ind2 = new double[popSize];`
4	`  double temp;`
5	`  for (int i = 0; i < popSize; i++) {`
6	`      ind2[i] = 0;`
7	`  }`
8	`  for (int i = 0; i < popSize; i++) {`
9	`      pbest[i][index] = fitness[i];`
10	`  }`
11	`  if (index == 0) {`
12	`     for(int i = 0; i < popSize; i++) {`
13	`        pbest[i][index] = fitness[i];`
14	`        titikpbest= partikel;`
15	`     }`
16	`  } else {`
17	`     for(int i = 0; i < popSize; i++) {`
18	`        for(int j = index; j < index + 1; j++) {`
19	`            if (pbest[i][j] < pbest[i][j - 1]) {`
20	`               ind2[i] = 1;`
21	`            } else {`
22	`               ind2[i] = 0;`
23	`            }`
24	`        }`
25	`     }`
26	`     for (int i = 0; i < popSize; i++) {`
27	`        for (int j = index; j < index + 1; j++) {`
28	`            if (pbest[i][j] < pbest[i][j - 1]) {`
29	`               temp = pbest[i][j - 1];`
30	`               pbest[i][j - 1] = pbest[i][j];`
31	`               pbest[i][j] = temp;`
32	`            }`
33	`        }`
34	`     }`
35	`  }`
36	`  for (int i = 0; i < popSize; i++) {`
37	`      hasil[i] = pbest[i][index];`
38	`       iPbest[i] = i;`
39	`  }`
40	`  for (int i = 0; i < popSize; i++) {`
41	`        if (ind2[i]==0) {`
42	`           titikpbest[i] = titikpbest[i];`

```
43          }else  {
44              titikpbest[i] = partikel[i];
45          }
46      }
47      return hasil;
48  }
```

Kode Program 5.8 *Update Pbest*

Pembahasan Kode Program 5.8 *Update Pbest*:

Baris 2-4 merupakan deklarasi variabel *hasil, ind2* dan *temp.*

Baris 5-7 merupakan perulangan sejumlah banyak partikel yang telah dibangkitkan sesuai dengan masukkan. Mengisi variabel *array ind2* dengan nilai 0.

Baris 8-10 merupakan perulangan sejumlah banyak partikel yang telah dibangkitkan sesuai dengan masukkan. Menyamakan nilai variabel *pbest* dengan nilai *fitness*.

Baris 11-16 merupakan kondisi jika nilai *index*=0, maka melakukan perulangan sejumlah banyak partikel yang telah dibangkitkan sesuai dengan masukkan. Menyamakan nilai variabel *pbest* dengan nilai *fitness*.

Baris 16-25 merupakan kondisi lain jika nilai *index* tidak sama dengan 0, melakukan perulangan sejumlah banyak partikel untuk mengecek indeks di dalamnya ketika nilai *pbest* pada iterasi tersebut tidak lebih besar dari sebelumnya maka variabel *ind2* diberi nilai 1 dan jika lebih besar maka *ind2* diberi nilai 0.

Baris 26-34 merupakan perulangan sejumlah banyak partikel lagi untuk mengecek indeks di dalamnya ketika nilai *pbest* pada iterasi tersebut tidak lebih besar dari sebelumnya maka nilai *pbest* akan disimpan sementara pada variabel *temp* untuk dilakukan pergeseran dengan nilai *pbest* yang lebih tinggi.

Baris 36-39 merupakan perulangan sejumlah banyak partikel untuk menyimpan nilai *pbest* indeks pada variabel *hasil.*

Baris 40-46 merupakan perulangan sejumlah banyak partikel dengan kondisi jika nilai variabel *ind2*=0 maka *titikpbest* disamakan dengan *titikpbest* dan selain itu maka *titikpbest* disamakan dengan *partikel*.

5.1.9 *Update Gbest*

Nilai *Gbest* diambil dari nilai *fitness Pbest* terbesar dalam setiap iterasinya. *Gbest* juga akan diperbarui jika menemukan nilai *fitness Pbest* terbesar yang lebih baik dari pada iterasi sebelumnya. Kode Program 5.9 membahas implementasi perancangan *Update Gbest*.

Baris	Kode Program
1	`public double Gbest(int indeks, double[] fitness) {`
2	`   double tempF[][] = new double[popSize][2];`
3	`   ind = new double[popSize];`
4	`   for (int i = 0; i < popSize; i++) {`

```
5           populasiUrut[i][0] = i;
6           populasiUrut[i][1] = fitness[i];
7       }
8       double tempFitness, tempIndeks;
9       for(int i = 0; i < popSize; i++) {
10        for(int j = 1; j < (popSize - i); j++) {
11          if(populasiUrut[j-1][1]<populasiUrut[j][1]) {
12             tempFitness = populasiUrut[j - 1][1];
13             tempIndeks = populasiUrut[j - 1][0];
14             populasiUrut[j-1][1]=populasiUrut[j][1];
15             populasiUrut[j-1][0]=populasiUrut[j][0];
16             populasiUrut[j][1] = tempFitness;
17             populasiUrut[j][0] = tempIndeks;
18          }
19        }
20      }
21      for(int i = 0; i < popSize; i++) {
22         ind[i] = populasiUrut[i][0];
23      }
24      gbest[indeks]=populasiUrut[0][0];
25      System.out.println("Gbest :"
26      +populasiUrut[0][0];
27      return Gbest[indeks];
28  }
```

Kode Program 5.9 *Update Gbest*

Pembahasan Kode Program 5.9 *Update Gbest*:

Baris 2 merupakan deklarasi variabel *tempF*.

Baris 3 merupakan deklarasi variabel *Ind*.

Baris 4-7 merupakan perulangan sebanyak partikel dengan menyamakan nilai indeks *popSize* dan *fitness* ke dalam variabel *populasiUrut*.

Baris 8 merupakan deklarasi variabel *tempFitness* dan *tempIndeks*.

Baris 9-20 merupakan perulangan sejumlah banyak partikel yang telah dibangkitkan sesuai dengan masukkan.

Baris 10-19 merupakan perulangan dengan teknik *bubble Short* dengan kondisi di dalamnya.

Baris 11-18 merupakan kondisi jika nilai *fitness* indeks 0 kurang dari *fitness* indeks 1, maka nilai *fitness* dan indeks partikel akan disimpan sementara pada variabel *tempFitness* dan *tempIndeks* untuk dilakukan pergeseran atau pengurutan nilai *fitness* pada indeks 1.

Baris 21-23 merupakan perulangan sejumlah banyak partikel dengan menyimpan nilai indeks partikel pada variabel *tempF* ke variabel *populasiUrut*.

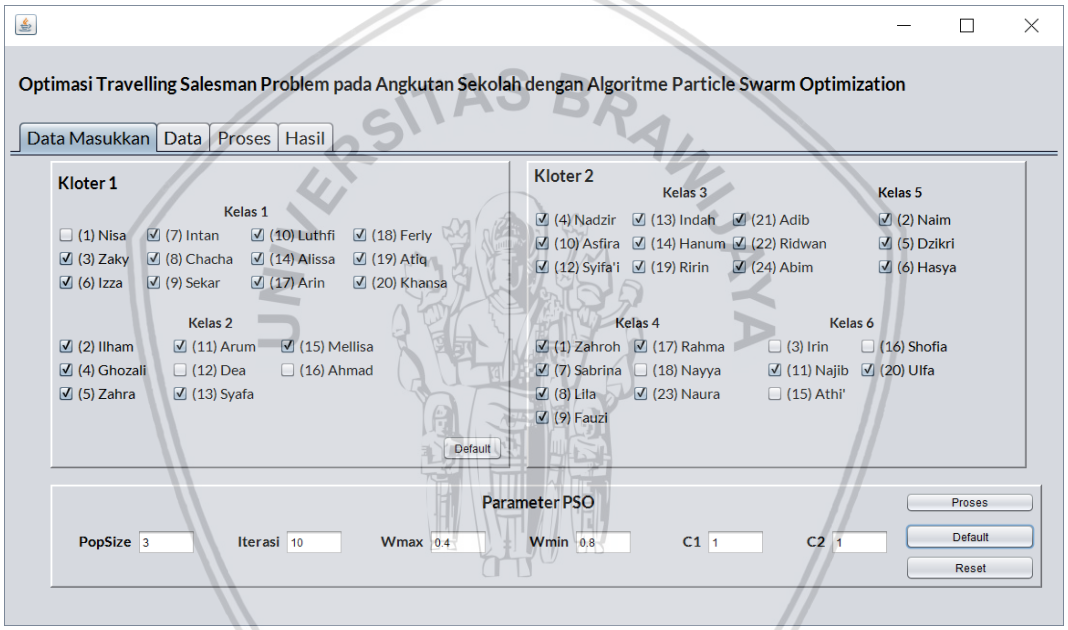
Baris 24-26 merupakan deklarasi dan perintah uruk menampilkan hasil *Gbest* dari variabel *pupulasiUrut*.

5.2 Implementasi Antarmuka

Antarmuka merupakan perantara komunikasi dari *system* dengan *user*. Seperti pada perancangan sebelumnya, penulis membagi menjadi 4 halaman antar muka yang terdiri dari Halaman depan atau halaman masukkan, Halaman Data, Halaman Proses dan Halaman Hasil. Berikut merupakan implementasi masing-masing Halaman antarmuka:

5.2.1 Halaman Depan

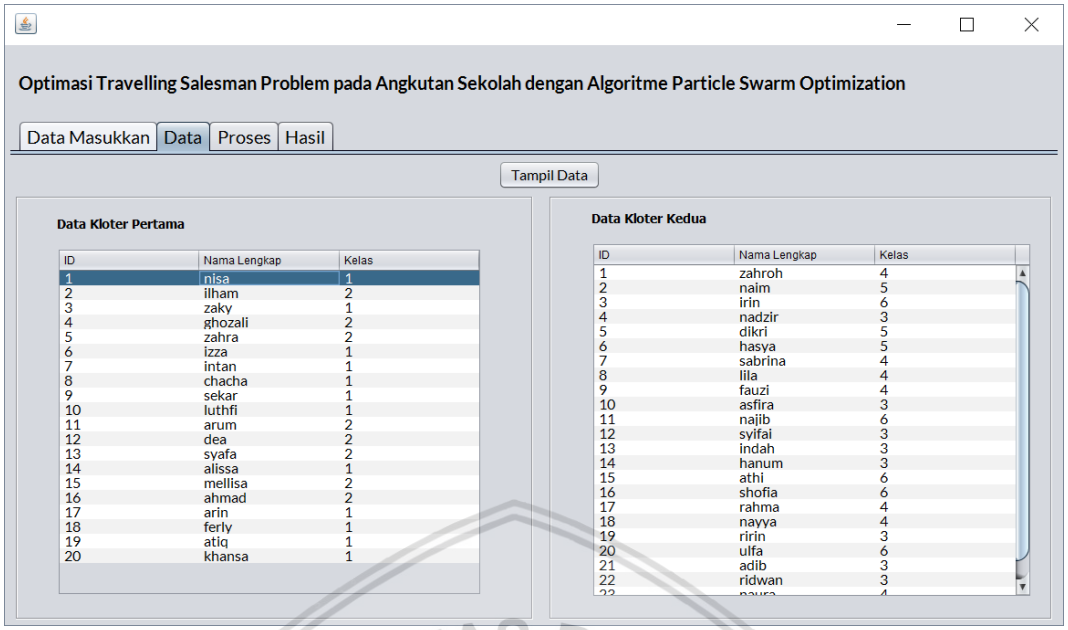
Halaman depan adalah halaman yang selalu ditampilkan secara *Default* sebagai halaman awal ketika program dijalankan. Di dalamnya berisi masukkan ID masing-masing kloter beserta parameter PSO yang akan digunakan untuk proses perhitungan. Oleh karena itu, Halaman Depan juga bisa disebut sebagai Halaman Masukkan. Gambar 5.1 menunjukkan implementasi antarmuka Halaman depan.



Gambar 5.1 Implementasi Halaman Depan

5.2.2 Halaman Data

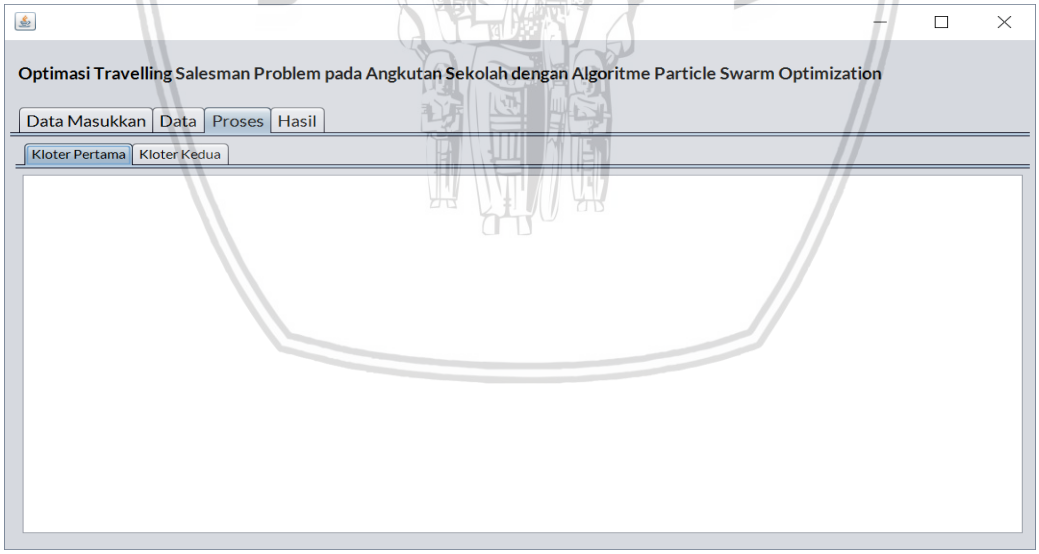
Halaman Data adalah halaman yang akan menampilkan data siswa pada masing-masing kloter. Data tersebut meliputi ID, Nama panggilan, Kelas dan Alamat rumah siswa. Gambar 5.2 menunjukkan implementasi antarmuka Halaman Data.



Gambar 5.2 Implementasi Halaman Data

5.2.3 Halaman Proses

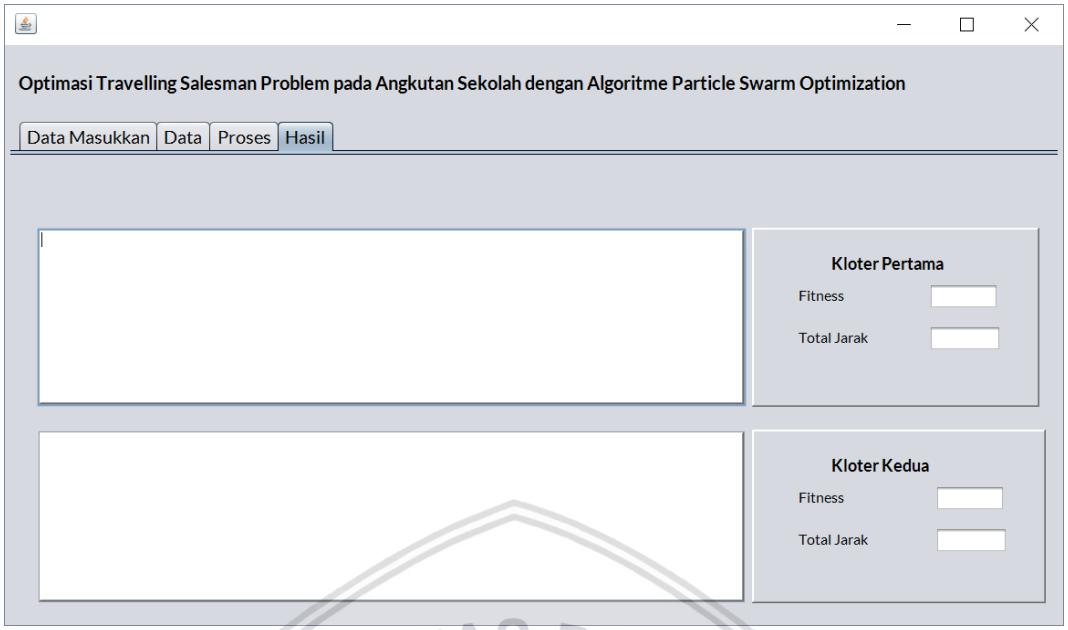
Halaman Proses adalah halaman yang akan menampilkan proses perhitungan optimasi pada masing-masing kloter. Gambar 5.3 menunjukkan implementasi antarmuka Halaman Proses.



Gambar 5.3 Halaman Proses

5.2.4 Halaman Hasil

Halaman Hasil adalah halaman yang akan menampilkan hasil proses optimasi. Hasil yang dikeluarkan oleh sistem ini adalah urutan rute pengantaran siswa dengan nilai *fitness* terbaik pada masing-masing kloter. Gambar 5.4 menunjukkan implementasi antarmuka Halaman Hasil.



Gambar 5.4 Halaman Hasil

BAB 6 PENGUJIAN DAN ANALISIS

6.1 Pengujian dan Analisis Jumlah Partikel

Pengujian Jumlah Partikel bertujuan untuk mengetahui berapa jumlah partikel optimal dalam menentukan solusi terbaik yang akan dihasilkan oleh sistem. Beberapa parameter yang digunakan dalam pengujian ini adalah:

Jumlah Iterasi : 100

Wmin : 0.4

Wmax : 0.9

C_1 : 1

C_2 : 1

6.1.1 Hasil Pengujian Jumlah Partikel

Hasil Pengujian Jumlah Partikel masing-masing kloter ditunjukkan pada Tabel 6.1 dan 6.2. Hasil lengkapnya bisa dilihat pada Lampiran C.1.

Tabel 6.1 Hasil Pengujian Jumlah Partikel kloter Pertama

Jumlah Partikel	Percobaan Ke-*i*							Rata-rata	
	1		2		..	10			
	Fitness	Waktu	*Fitness*	Waktu	..	*Fitness*	Waktu	*Fitness*	Waktu
50	0,033	0,672	0,0347	0,687	..	0,0361	0,716	0,03427	0,707
100	0,0328	0,722	0,0339	0,713	..	0,0372	0,717	0,03536	0,712
150	0,0393	0,734	0,0364	0,732	..	0,0366	0,738	0,03569	0,733
200	0,0419	0,728	0,0354	0,73	..	0,0362	0,734	0,03681	0,731
250	0,0375	0,754	0,0365	0,748	..	0,0384	0,779	0,03718	0,754
300	0,038	0,747	0,0395	0,749	..	0,0348	0,749	0,03778	0,764
350	0,0393	0,756	0,0409	0,784	..	0,0387	0,762	0,03838	0,782
400	0,0387	0,779	0,0376	0,826	..	0,0391	0,805	0,0379	0,784
450	0,0387	0,795	0,0399	0,789	..	0,0357	0,778	0,03853	0,786
500	0,0343	0,824	0,0386	0,805	..	0,0384	0,811	0,03794	0,802

Tabel 6.2 Hasil Pengujian Jumlah Partikel kloter Kedua

Jumlah Partikel	Percobaan Ke-i							Rata-rata	
	1		2		..	10			
	Fitness	Waktu	Fitness	Waktu	..	Fitness	Waktu	Fitness	Waktu
50	0,0225	0,672	0,0225	0,687	..	0,0226	0,716	0,02311	0,707
100	0,0273	0,722	0,0268	0,713	..	0,0247	0,717	0,02584	0,712
150	0,0271	0,734	0,0262	0,732	..	0,0266	0,738	0,02633	0,733

200	0,0254	0,728	0,0249	0,73	..	0,0253	0,734	0,0255	0,731
250	0,0264	0,754	0,0255	0,748	..	0,0272	0,779	0,026793	0,754
300	0,026	0,747	0,0256	0,749	..	0,0281	0,749	0,02591	0,764
350	0,0294	0,756	0,0265	0,784	..	0,0277	0,762	0,02695	0,782
400	0,0266	0,779	0,0282	0,826	..	0,0275	0,805	0,0271	0,784
450	0,0282	0,795	0,0271	0,789	..	0,0266	0,778	0,02703	0,786
500	0,0286	0,824	0,0276	0,805	..	0,0278	0,811	0,02708	0,802

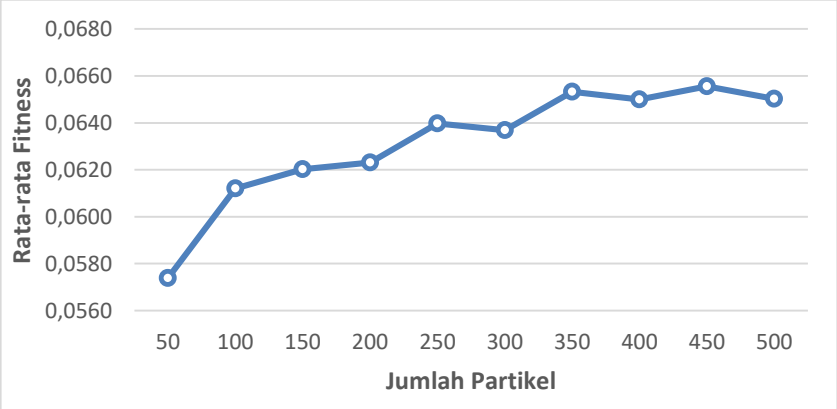
Untuk memudahkan penentuan jumlah partikel optimal dari kedua kloter maka dilakukan penjumlahan rata-rata nilai *fitness* pada setiap kloter. Tabel 6.3 menunjukkan gabungan nilai rata-rata *fitness*.

Tabel 6.3 Hasil Gabungan Nilai Rata-rata *Fitness* Setiap Kloter

Jumlah Partikel	Penjumlahan *fitness*	Waktu Komputasi (Detik)
50	0,0574	0,707
100	0,0612	0,712
150	0,0620	0,733
200	0,0623	0,731
250	0,0640	0,754
300	0,0637	0,764
350	0,0653	0,782
400	0,0650	0,784
450	0,0656	0,786
500	0,0650	0,802

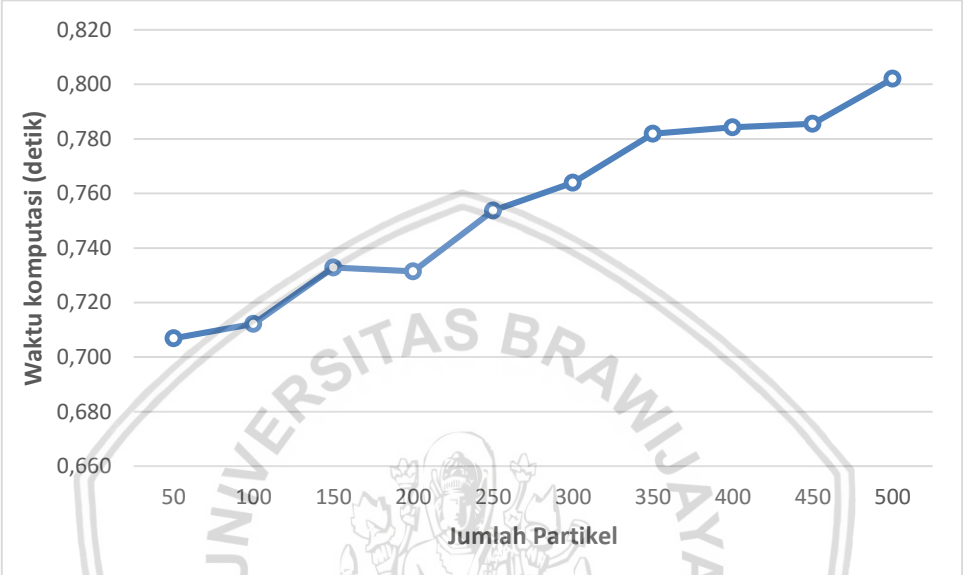
6.1.2 Analisis Hasil Pengujian Jumlah Partikel

Berdasarkan Tabel 6.3, maka dibuatlah Grafik Hasil Pengujian Jumlah Partikel yang ditunjukkan pada Gambar 6.1 dan Gambar 6.2.



Gambar 6.1 Grafik Hasil Rata-rata *Fitness*

Grafik Hasil Pengujian pada Gambar 6.1 menunjukkan bahwa semakin banyak jumlah partikel maka akan semakin tinggi nilai *fitness* yang dihasilkan. Jumlah partikel yang banyak dapat memberi nilai atau solusi dengan lebih bervariasi, sehingga ruang pencarian solusi akan semakin besar dan luas. Pada pengujian ini, didapatkan nilai rata-rata *fitness* terendah sebesar 0,0574 pada 50 partikel dan nilai rata-rata *fitness* tertinggi sebesar 0,0656 pada 450 Partikel.



Gambar 6.2 Gafik Hasil Waktu Komputasi

Grafik Hasil pengujian waktu komputasi pada Gambar 6.2 menunjukkan bahwa semakin banyak jumlah partikel, maka semakin lama juga waktu yang dibutuhkan untuk proses komputasi. Meskipun semakin lamanya waktu komputasi, namun nilai *fitness* yang dihasilkan juga semakin besar. Oleh karena itu, penulis menentukan bahwa jumlah partikel sebanyak 450 merupakan jumlah partikel paling optimal dengan waktu 0,786 detik dan menghasilkan nilai *fitness* yang paling baik.

6.2 Pengujian Kombinasi *Wmin* dan *Wmax*

Pengujian Kombinasi *Wmin* dan *Wmax* bertujuan untuk mengetahui kombinasi nilai Bobot inersia terendah dan tertinggi optimal dalam menentukan solusi terbaik yang akan dihasilkan oleh sistem. Pengujian ini dilakukan rentang 0 hingga 1. Berikut ini parameter yang digunakan untuk pengujian kombinasi *Wmin* dan *Wmax*:

Jumlah Partikel : 450

Jumlah Iterasi : 100

C_1 : 1

C_2 : 1

6.2.1 Hasil Pengujian Kombinasi *Wmin* dan *Wmax*

Hasil pengujian kombinasi *Wmin* dan *Wmax* masing-masing kloter ditunjukkan pada Tabel 6.4 dan 6.5. Hasil selengkapnya dapat dilihat pada Lampiran C.2.

Tabel 6.4 Hasil Pengujian Kombinasi *Wmin* dan *Wmax* Kloter Pertama

Bobot Inersia		Nilai *Fitness*							Rata-rata *Fitness*
Wmax	*Wmin*	Percobaan ke-*i*							
		1	2	3	4	5	..	10	
0,9	0,2	0,0371	0,0344	0,0382	0,035	0,0379	..	0,0376	0,03758
	0,3	0,036	0,0358	0,0388	0,0374	0,0397	..	0,0405	0,03773
	0,4	0,039	0,039	0,0375	0,0357	0,0343	..	0,0351	0,03736
0,8	0,2	0,0386	0,0385	0,0367	0,0387	0,0386	..	0,0399	0,03889
	0,3	0,0353	0,0404	0,0376	0,0429	0,0367	..	0,0393	0,0385
	0,4	0,0353	0,0372	0,0358	0,0384	0,0416	..	0,0381	0,03773
0,7	0,2	0,0397	0,0367	0,0381	0,0378	0,0405	..	0,043	0,03885
	0,3	0,0395	0,0397	0,0381	0,0366	0,0387	..	0,0371	0,03805
	0,4	0,0344	0,0365	0,0384	0,041	0,0351	..	0,0368	0,03809

Tabel 6.5 Hasil Pengujian Kombinasi *Wmin* dan *Wmax* Kloter Kedua

Bobot Inersia		Nilai *Fitness*							Rata-rata *Fitness*
Wmax	*Wmin*	Percobaan ke-*i*							
		1	2	3	4	5	..	10	
0,9	0,2	0,0266	0,0268	0,0263	0,0267	0,0252	..	0,0275	0,02668
	0,3	0,0259	0,0264	0,0275	0,0291	0,026	..	0,0268	0,0269
	0,4	0,0267	0,0284	0,027	0,0275	0,0268	..	0,0259	0,02702
0,8	0,2	0,0297	0,0267	0,0253	0,0257	0,0272	..	0,0266	0,02678
	0,3	0,0284	0,0287	0,0276	0,0267	0,0285	..	0,0286	0,02747
	0,4	0,0261	0,027	0,027	0,0266	0,0263	..	0,0291	0,02695
0,7	0,2	0,0263	0,0263	0,0272	0,0275	0,0302	..	0,0264	0,02729
	0,3	0,0267	0,0259	0,026	0,0269	0,0257	..	0,0266	0,02656
	0,4	0,027	0,028	0,0268	0,0275	0,028	..	0,0264	0,02694

Untuk memudahkan penentuan Kombinasi *Wmin* dan *Wmax* optimal dari kedua kloter maka dilakukan penjumlahan rata-rata nilai *fitness* pada setiap kloter. Tabel 6.6 menunjukkan gabungan nilai rata-rata *fitness*.

Tabel 6.6 Hasil Gabungan Nilai Rata-rata *Fitness* Setiap Kloter

Bobot Inersia		Rata-rata *Fitness*
Wmax	Wmin	
0,9	0,2	0,0643
	0,3	0,0646
	0,4	0,0644
0,8	0,2	0,0657
	0,3	0,0660
	0,4	0,0647
0,7	0,2	0,0661
	0,3	0,0646
	0,4	0,0650

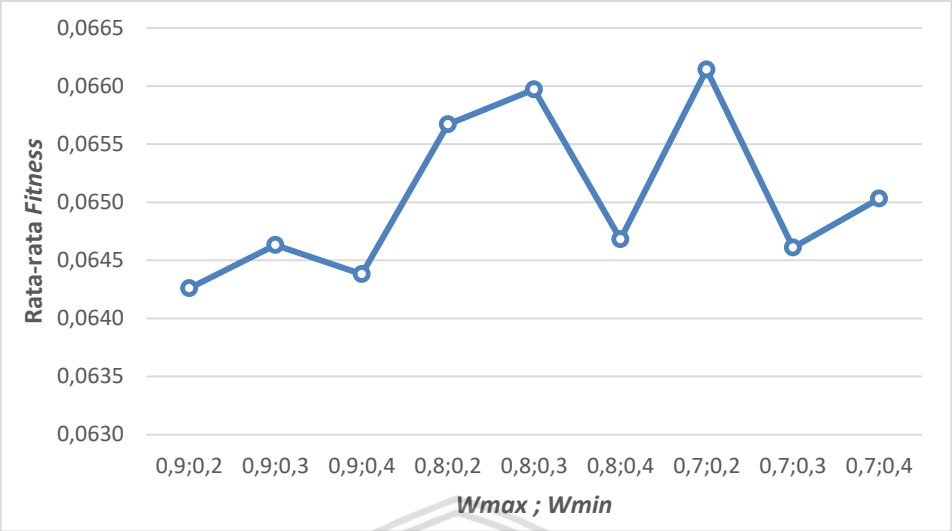
6.2.2 Analisis Hasil Pengujian Kombinasi *Wmin* dan *Wmax*

Bobot inersia berfungsi untuk memperluas daya eksplorasi (pencarian global) dan eksploitasi (pencarian lokal) perpindahan partikel yang berpengaruh pada keragaman partikel dan nilai solusi yang akan dihasilkan. Besar kecilnya nilai *W*, memberikan pengaruh tersendiri dalam menghasilkan solusi. Nilai *W* yang semakin besar membuat kecepatan partikel dalam berpindah semakin tinggi. Dampaknya adalah partikel dapat melakukan pencarian global lebih maksimal namun sering melewatkan solusi pada area lokal. Sebaliknya jika semakin kecil nilai *W*, maka kecepatan partikel dalam berpindah akan semakin kecil. Dampaknya adalah partikel dapat melakukan pencarian lokal pada suatu area secara maksimal namun peluang untuk pencarian globalnya berkurang. Oleh karena itu, diperlukan kombinasi nilai bobot inersia yang seimbang untuk pencarian lokal dan global. Persamaan 2.2 dan 2.3 memperjelas pengaruh nilai *W* dalam perpindahan partikel.

$$\boldsymbol{V_J^k} = \boldsymbol{W}V_J^k + C_1 rand_1 \times \left(Pbest_j - X_j^k\right) + C_2 rand_2 \times \left(Gbest_j - X_j^k\right)$$

$$\boldsymbol{W} = Wmax - \frac{\boldsymbol{Wmax - Wmin}}{iter\ max} x\ iter$$

Berdasarkan Tabel 6.6, maka dibuatlah Grafik Hasil Pengujian Kombinasi *Wmin* dan *Wmax* yang ditunjukkan pada Gambar 6.3.



Gambar 6.3 Grafik Hasil Pengujian Kombinasi *Wmin* dan *Wmax*

Kombinasi *Wmin* sebesar 0,2 dan *Wmax* sebesar 0,7 menghasilkan nilai rata-rata *fitness* tertinggi yakni 0,0661. Hal ini menunjukkan bahwa kombinasi tersebut merupakan kombinasi paling optimal dalam menghasilkan solusi terbaik pada permasalahan optimasi rute terpendek angkutan sekolah.

6.3 Pengujian Koefisien Akselerasi

Pengujian koefisien akselerasi bertujuan untuk mengetahui kombinasi nilai koefisien akselerasi optimal dalam menentukan solusi terbaik yang akan dihasilkan oleh sistem. Nilai yang diujikan ini adalah 1, 1.5 dan 2. Berikut ini parameter yang digunakan untuk pengujian Koefisien akselerasi:

Jumlah Partikel : 450

Jumlah Iterasi : 100

Wmin : 0.2

Wmax : 0.7

6.3.1 Hasil Pengujian Koefisiean Akselerasi

Hasil Pengujian Koefisien Akselerasi masing-masing kloter ditunjukkan pada Tabel 6.7 dan 6.8. Hasil selengkapnya dapat dilihat pada Lampiran C.3.

Tabel 6.7 Hasil Pengujian Koefisien Akselerasi Kloter Pertama

Koefisien Akselersi		Nilai *Fitness*						Rata-rata *Fitness*
C_1	C_2	Percobaan ke-*i*						
		1	2	3	4	..	10	
1	1	0,0382	0,0376	0,0374	0,0371	..	0,0399	0,03813
	1,5	0,0433	0,0412	0,0393	0,0354	..	0,0354	0,03925

	2	0,0369	0,0405	0,0361	0,0381	..	0,0408	0,03835
1,5	1	0,0343	0,0396	0,0359	0,0405	..	0,0361	0,0379
	1,5	0,0392	0,0389	0,0376	0,0359	..	0,0426	0,03881
	2	0,034	0,0359	0,0358	0,0363	..	0,0429	0,03569
2	1	0,0364	0,0419	0,037	0,0363	..	0,0401	0,03814
	1,5	0,0357	0,039	0,0396	0,0371	..	0,039	0,03888
	2	0,0352	0,0385	0,04	0,0352	..	0,0358	0,03703

Tabel 6.8 Hasil Pengujian Koefisien Akselerasi Kloter Kedua

Koefisien Akselersi		Nilai *Fitness*						Rata-rata *Fitness*
C_1	C_2	Percobaan ke-*i*						
		1	2	3	4	..	10	
1	1	0,027	0,0275	0,0265	0,0262	..	0,0275	0,02733
	1,5	0,026	0,027	0,0264	0,0274	..	0,0282	0,02713
	2	0,0277	0,0261	0,028	0,0264	..	0,0268	0,02733
1,5	1	0,0263	0,0284	0,0284	0,0283	..	0,0262	0,02726
	1,5	0,0262	0,0279	0,0274	0,0263	..	0,0271	0,02712
	2	0,0271	0,0263	0,0271	0,0267	..	0,0269	0,02689
2	1	0,0273	0,0263	0,0257	0,0276	..	0,0264	0,02653
	1,5	0,0259	0,0268	0,0265	0,0284	..	0,0269	0,02676
	2	0,0266	0,0256	0,0283	0,0257	..	0,0273	0,02674

Untuk memudahkan penentuan Kombinasi Nilai Koefisien Akselerasi optimal dari kedua kloter maka dilakukan penjumlahan rata-rata nilai *fitness* pada setiap kloter. Tabel 6.9 menunjukkan gabungan nilai rata-rata *fitness*.

Tabel 6.9 Hasil Gabungan Nilai Rata-rata *Fitness* Setiap Kloter

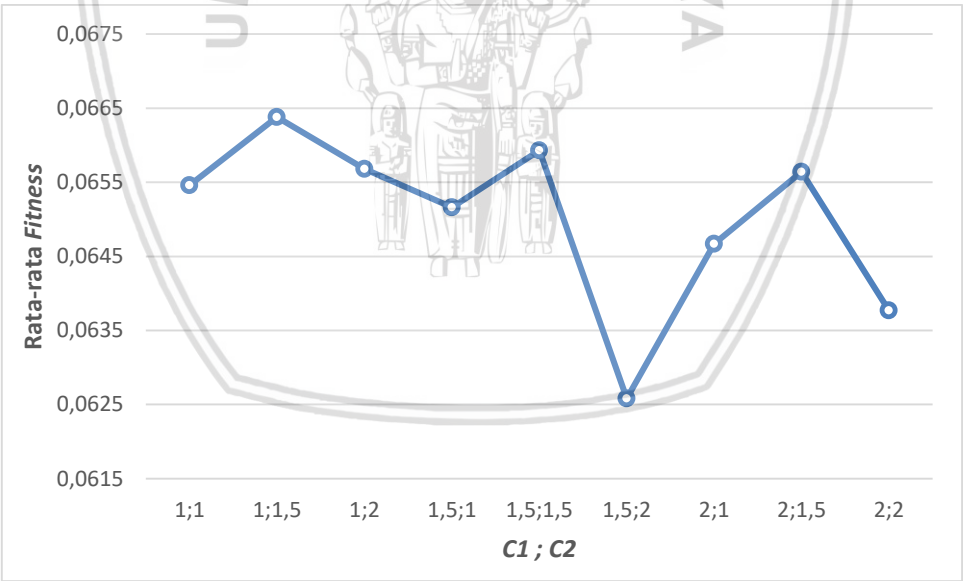
Koefisien Akselersi		Penjumlahan *Fitness*
C_1	C_2	
1	1	0,0655
	1,5	0,0664
	2	0,0657
1,5	1	0,0652
	1,5	0,0659

	2	0,0626
2	1	0,0647
	1,5	0,0656
	2	0,0638

6.3.2 Analisis Hasil Pengujian Koefisien Akselerasi

Seperti kombinasi *Wmin* dan *Wmax*, Koefisien akselerasi juga berperan dalam menentukan pergerakan partikel pada ruang pencarian. Nilai dari C_1 dan C_2 yang besar dapat mempengaruhi partikel untuk melakukan perpindahan yang relatif lebih luas dalam pencarian solusi, sehingga kemampuan eksplorasi partikel akan menjadi lebih baik namun bisa saja terlampau melebihi batas ruang pencarian. Sebaliknya, jika nilai C_1 dan C_2 terlalu kecil, maka partikel akan cenderung untuk melakukan eksploitasi di sekitar areanya pada posisi saat itu. Oleh karena itu dibutuhkan kombinasi nilai C_1 dan C_2 agar dapat memberikan pergerakan partikel yang menuju pada posisi terbaik dan menghasilkan banyak variasi untuk setiap partikel.

Berdasarkan Tabel 6.9, maka dibuatlah Grafik Hasil Pengujian Koefisien akselerasi yang ditunjukkan pada Gambar 6.4.



Gambar 6.4 Grafik Hasil Pengujian Koefisien Akselerasi

Kombinasi nilai C_1 dan C_2 sebesar 1 dan 1,5 menghasilkan nilai rata-rata *fitness* tertinggi yakni 0,0664. Hal ini menunjukkan bahwa kombinasi nilai tersebut dianggap sebagai kombinasi koefisien akselerasi paling optimal untuk menyelesaikan permasalahan Optimasi Rute Angkutan Sekolah ini.

6.4 Pengujian Konvergensi

Pengujian Konvergensi bertujuan untuk mengetahui kinerja maksimal sistem dalam menghasilkan solusi. Pengujian ini dilakukan dengan menganalisis hasil stagnasi nilai solusi berdasarkan iterasi. Ketika solusi yang dihasilkan tidak mengalami perubahan signifikan pada iterasi tertentu, maka dapat dianggap bahwa sistem telah mencapai konvergensi. Percobaan pengujian ini akan dilakukan sebanyak 5 kali. Adapun jumlah iterasi yang diujikan adalah 50 iterasi dan dikombinasi dengan beberapa parameter sebagai berikut:

Jumlah Partikel : 450

Wmin : 0.2

Wmax : 0.7

C_1 : 1

C_2 : 1.5

6.4.1 Hasil Pengujian

Tabel 6.10 dan 6.11 merupakan hasil percobaan yang menunjukkan tanda-tanda konvergensi sistem pada masing-masing kloter.

Tabel 6.10 Hasil Pengujian Konvergensi Kloter Pertama

Iterasi ke-	Percobaan Ke-*i*				
	1	2	3	4	5
	Fitness	*Fitness*	*Fitness*	*Fitness*	*Fitness*
1	0,0353	0,0337	0,0361	0,0359	0,0371
4	0,0365	0,0362	0,0363	0,0363	0,0396
8	0,0372	0,0369	0,0374	0,0371	0,0396
12	0,0391	0,0371	0,0374	0,0371	0,0427
16	0,0423	0,0389	0,0402	0,0428	0,0427
20	0,0423	0,041	0,0402	0,0428	0,0427
25	0,0423	0,041	0,0402	0,0428	0,0427
30	0,0423	0,041	0,0402	0,0428	0,0427
35	0,0423	0,041	0,0402	0,0428	0,0427
40	0,0423	0,041	0,0402	0,0428	0,0427
45	0,0423	0,041	0,0402	0,0428	0,0427
50	0,0423	0,041	0,0402	0,0428	0,0427

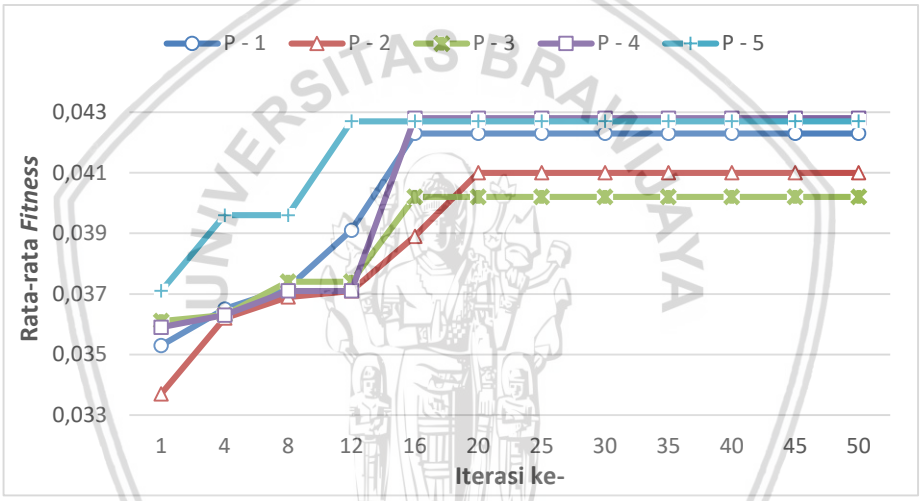
Tabel 6.11 Hasil Pengujian Konvergensi Kloter Kedua

Iterasi ke-	Percobaan Ke-*i*				
	1	2	3	4	5
	Fitness	*Fitness*	*Fitness*	*Fitness*	*Fitness*
1	0,0212	0,0205	0,021	0,0203	0,0198
4	0,0213	0,0209	0,0217	0,0213	0,0231
8	0,0222	0,0217	0,0234	0,0223	0,0238

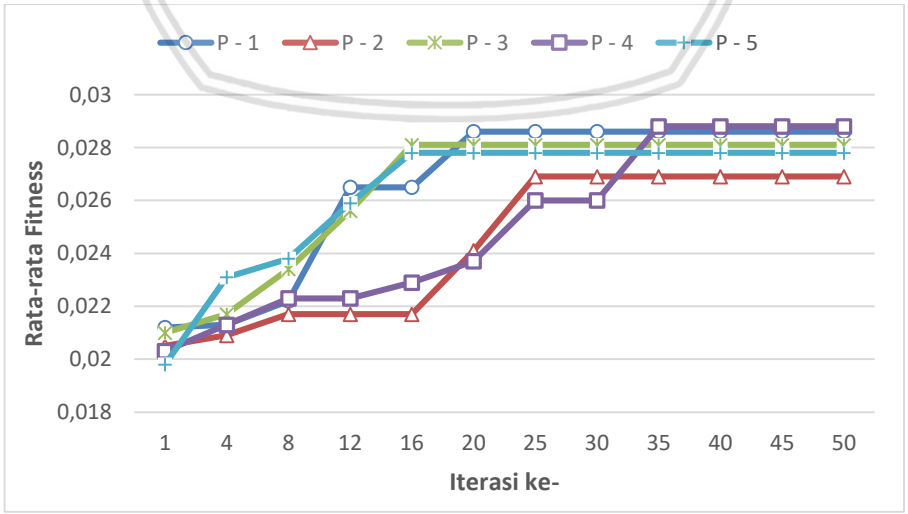
12	0,0265	0,0217	0,0256	0,0223	0,0259
16	0,0265	0,0217	0,0281	0,0229	0,0278
20	0,0286	0,0241	0,0281	0,0237	0,0278
25	0,0286	0,0269	0,0281	0,026	0,0278
30	0,0286	0,0269	0,0281	0,026	0,0278
35	0,0286	0,0269	0,0281	0,0288	0,0278
40	0,0286	0,0269	0,0281	0,0288	0,0278
45	0,0286	0,0269	0,0281	0,0288	0,0278
50	0,0286	0,0269	0,0281	0,0288	0,0278

6.4.2 Analisis Hasil Pengujian Konvergensi

Berdasarkan Tabel 6.10 dan 6.11 maka dibuatlah Grafik Hasil Pengujian Konvergensi yang ditunjukkan pada Gambar 6.5 dan 6.6 untuk masing-masing kloternya.



Gambar 6.5 Grafik Hasil Pengujian Konvergensi Kloter Pertama



Gambar 6.6 Grafik Hasil Pengujian Konvergensi Kloter Kedua

Hasil di atas menunjukkan bahwa kloter pertama mulai menunjukkan konvergen pada iterasi ke- 20. Sedangkan untuk kloter kedua menunjukkan tanda-tanda konvergen pada iterasi ke- 35. Untuk selanjutnya sistem secara stagnan menunjukkan hasil solusi yang stabil tanpa ada perubahan hingga mencapai iterasi terakhir. Oleh karena itu dianggap bahwa iterasi sejumlah 35 merupakan iterasi optimal mengingat sistem akan segera berhenti jika telah mencapai konvergensi.

6.5 Analisis Global Hasil Pengujian

Dari beberapa pengujian yang telah dilakukan sebelumnya, maka didapat parameter yang dianggap optimal dalam menyelesaikan permasalahan optimasi rute angkutan sekolah ini. Adapun beberapa parameter tersebut adalah:

Jumlah Partikel : 450

Jumlah Iterasi : 35

Wmin : 0.2

Wmax : 0.7

C_1 : 1

C_2 : 1.5

Selanjutnya akan dilakukan analisis global untuk menerapkan parameter di atas untuk menguji data sampel aktual dari pengantaran siswa-siswi pada MI Salafiyah Kasim. Terdapat 1 hari data sampel aktual yang ditunjukkan pada Tabel 6.12.

Tabel 6.12 Data Aktual Rute Pengantaran

Hari	Data Aktual Urutan Pengantaran					
	Kloter Pertama			Kloter Kedua		
	ID	Nama	Total Jarak (Km)	ID	Nama	Total Jarak (Km)
1	S	Sekolah	22,58	S	sekolah	33,27
	19	atiq		13	indah	
	14	sekar		14	hanum	
	18	firli		11	najib	
	15	arin		12	syifa'i	
	17	melissa		19	ririn	
	11	alissa		10	asfira	
	13	arum		9	fauzi	
	9	khansa		8	lila	
	20	Iza		7	shabrina	
	10	caca		24	abimanyu	
	2	intan		20	ulfa	
	3	ilham		23	naura	
	4	zaki		1	zahroh	
	5	zahra		2	naim	
	6	lutfi		22	ridwan	
	7	syafa		4	nadzir	

	8	Ghozali		5	dzikri
	S	Sekolah		6	hasya
				21	adib
				17	rahma
				S	sekolah

Berdasarkan data di atas maka telah dilakukan pengujian sistem sesuai dengan data dan parameter yang telah ditentukan. Hasil dari keluaran sistem ditunjukkan pada Tabel 6.13. Data lengkap bisa dilihat pada Lampiran C.4.

Tabel 6.13 Hasil Rekomendasi Sistem

Percobaan ke-	Data Hasil Rekomendasi Sistem					
	Kloter Pertama			Kloter Kedua		
	ID	Nama	Total Jarak (Km)	ID	Nama	Total Jarak (Km)
1	S	Sekolah	23,17	S	sekolah	33,80
	17	alissa		13	indah	
	11	Ghozali		14	hanum	
	13	arin		11	najib	
	9	syafa		12	syifa'i	
	19	melissa		19	ririn	
	15	caca		10	asfira	
	18	lutfi		9	fauzi	
	14	zaki		8	lila	
	7	zahra		7	shabrina	
	8	firli		24	abimanyu	
	6	arum		23	naura	
	4	atiq		20	Ulfa	
	2	khansa		1	zahroh	
	5	sekar		2	naim	
	3	intan		22	ridwan	
	20	ilham		4	nadzir	
	10	Iza		5	dzikri	
	S	Sekolah		6	hasya	
				21	adib	
				17	rahma	
				5	sekolah	
2	S	Sekolah	22,13	S	sekolah	32,41
	..	..		..	..	
	S	sekolah		S	sekolah	
3	S	Sekolah	22,97	S	sekolah	31,66
	..	..		..	..	
	S	sekolah		S	hasya	

4	S	Sekolah	21,93	S	sekolah	32,73
	..	..		..	..	
	S	sekolah		S	hasya	
5	S	Sekolah	22,13	S	sekolah	34,2
	..	..		..	..	
	S	sekolah		S	hasya	

Dari data hasil rekomendasi tersebut, maka akan dibandingkan dengan data aktual dan dihitung selisihnya. Hasil selisih ditunjukkan oleh Tabel 6.14.

Tabel 6.14 Hasil Selisih

Hari	Data Hasil Selisih					
	Kloter Pertama			Kloter Kedua		
	Jarak Aktual (Km)	Jarak Sistem (Km)	Selisih (Km)	Jarak Aktual (Km)	Jarak Sistem (Km)	Selisih (Km)
1	22,58	23,17	-0,59	33,27	33,8	-0,53
		22,13	0,45		32,41	0,86
		22,97	-0,39		31,66	1,61
		21,93	0,65		32,73	0,54
		22,13	0,45		34,2	-0,93

Hasil di atas merupakan perbandingan data aktual dengan data hasil sistem dalam lima kali percobaan. Dari lima kali percobaan pada masing-masing kloter, nilai negatif berwarna merah menunjukkan jika jarak hasil dari sistem tidak lebih baik dari pada jarak aktual. Variasi yang dihasilkan pada setiap percobaan beragam. Ada dalam suatu percobaan yang tidak optimal pada kedua kloter, salah satu kloter optimal dan kedua kloter optimal. Jika ditinjau secara keseluruhan, sistem dinilai mampu untuk menghasilkan solusi yang baik, namun tidak selalu optimal. Fungsi *Repair* sangat berpengaruh dalam menghasilkan total jarak yang menghasilkan *fitness* masing-masing partikel. Terdapat beberapa rute pengantaran yang mengantar dari titik desa A ke desa B lalu kembali ke desa A lagi. Maka dari itu, diperlukan untuk memaksimalkan fungsi *Repair* pada penelitian selanjutnya.

BAB 7 PENUTUP

7.1 KESIMPULAN

Berikut merupakan kesimpulan yang dapat diambil dari hasil penelitian Optimasi *Travelling Salesman Problem* Dengan Algoritme *Particle Swarm Optimization* Pada Angkutan Sekolah:

1. Implementasi Algoritme PSO untuk Optimasi Penentuan Rute Pada Angkutan Sekolah mula-mula dilakukan dengan inisialisasi partikel awal secara acak. Penentuan nilai partikel dilakukan setelah melakukan konversi ID siswa ke dalam angka permutasi mulai dari 1 sampai jumlah siswa terpilih, sehingga hasil permutasi merepresentasikan ID siswa. Perhitungan *fitness* didasarkan pada jumlah jarak dari sekolah ke alamat siswa pertama hingga siswa terakhir lalu kembali ke sekolah lagi berdasarkan urutan rute yang telah ditentukan oleh setiap partikel yang dibangkitkan. Setelah *fitness* setiap partikel diketahui maka dapat dilakukan *Update* Kecepatan, *Update* Posisi, *Repair, Update Pbest* dan *Gbest*. Berdasarkan Langkah-langkah tersebut, maka dapat diperoleh hasil optimasi berdasarkan nilai *fitness* terbesar dari seluruh partikel yang ditunjukkan oleh nilai *Global Best* pada Iterasi terakhir.
2. Parameter yang digunakan Algoritme PSO sangat berpengaruh terhadap hasil Optimasi Rute Angkutan Sekolah. Berdasarkan pengujian dan analisis sistem, didapatkan parameter PSO yang dianggap optimal dalam menyelesaikan permasalahan yakni, Jumlah Partikel=450, *Wmin*=0.2, *Wmax*= 0.7, C_1=1, C_2=1,5 dan Jumlah Iterasi=35. Parameter tersebut diimplementasikan pada data sampel aktual dengan lima kali percobaan dari sistem. Lima percobaan pada setiap kloter, tiga diantaranya menunjukkan hasil yang lebih baik dari pada data aktual. Hal ini menunjukkan bahwa sistem mampu menghasilkan solusi yang baik, namun tidak selalu optimal.

7.2 SARAN

1. Penelitian selanjutnya dapat memaksimalkan fungsi *repair* agar dapat lebih optimal dalam memperbaiki nilai partikel yang tidak sesuai dengan yang ditentukan di awal. Selain itu, bisa juga diterapkan tipe Algoritme PSO lain seperti HDPSO yang memungkinkan dapat bekerja lebih efisien karena tidak memerlukan fungsi *repair* dalam menghasilkan solusi.
2. Penelitian selanjutnya dapat menambahkan fitur untuk menentukan ongkos kendaraan yang bisa dijadikan pedoman bagi sekolah dalam menentukan biaya setiap siswa yang berbeda satu sama lain sesuai dengan jarak rumah ke sekolah. Dengan adanya fitur penentuan harga tersebut diharapkan pihak sekolah dapat mengatur pendapatan dan pengeluaran dalam angkutan sekolah agar lebih stabil.

3. Penelitian selanjutnya dapat menambahkan fitur untuk mengolah data siswa di dalam *database* agar sistem dapat digunakan oleh sekolah dengan fleksibel sesuai dengan keadaan terbaru pada sekolah tersebut.
4. Penelitian selanjutnya dapat menjadikan sistem ini menjadi berbasis *Mobile* dengan keluaran hasil rute terbaik dari sistem bisa terintegrasi dengan GPS sehingga lebih mudah untuk digunakan baik untuk pihak sekolah maupun supir sekolah.

DAFTAR PUSTAKA

Beraunews. 2017. *Data Indonesia dan Dunia, Korban Laka Lantas Usia Remaja Urutan Kedua Terbanyak*. [Online]. Tersedia di: http://www.beraunews.com/lipsus-real-life/liputan-khusus/3002-data-indonesia-dan-dunia-korban-laka-lantas-usia-remaja-urutan-kedua-terbanyak?showall=1&limit*Start*= [Diakses pada tanggal 9 Januari 2018].

Felia, Eliantara. Cholissodin, I., & Indriati. 2016. *Implementasi Algoritma Particle swarm Optimization dalam Optimasi Pemenuhan Kebutuhan Gizi Keluarga.*S1.Universitas Brawijaya.

Haupt, R.L., Haupt, S.E. 2004. *Practical Genetic Algorithm*, Second Edition. s.l. : Wiley, 2004.

Hermawan, Arif, Hidayat, I., & Budi. 2017. *Sistem Optimasi Rute Tempat Wisata Kuliner Di Malang Menggunakan Algoritma Bee Colony.*S1.Universitas Brawijaya.

Istiqomah, Leni, Cholissodin, I., & Marji. 2017. *Implementasi Algoritma Particle swarm Optimization untuk Optimasi Pemenuhan Kebutuhan Gizi Balita.*S1.Universitas Brawijaya.

Karimah, Sayyidah. Cholissodin, I., & Indriati. 2017. *optimasi multiple travelling salesman problem (M-TSP) pada pendistribusian air minum menggunakan algoritme genetika.*S1.Universitas Brawijaya.

Kariono. 2011. *Layanan transportasi sekolah untuk menekan tidak masuk dan terlambat ke sekolah bagi siswa*. [Online]. Tersedia di: http://library.um.ac.id/free-contents/indeks.php/pub/detail/layanan-transportasi-sekolah-untuk-menekan-tidak-masuk-dan-terlambat-ke-sekolah-bagi-siswa-studi-kasus-di-sekolah-dasar-islam-terpadu-al-hikmah-bence-garum-blitar-kariono-50928.html [Diakses pada tanggal 9 Januari 2018].

Kompasnia. 2017. *Perlukah Ada Bus Sekolah di Kabupaten atau Kota?*. [Online]. Tersedia di https://www.kompasiana.com/penaulum/5a44cc485e13731e5b530382/pentingkah-ada-bus-sekolah-di-kabupaten-kota [Diakses pada tanggal 9 Januari 2018].

Kurniawan, E.P. & M., Z.Z. 2010. *Fuzzy Membership Function Generation Using Particle Swarm Optimization.* s.l. : International Journal Open Problems Computation Math, No 3, 2010.

Lipschutz, S., Teo, K.L., dan Hendy, M.D. 1985, "*Notes on Graphs and Matrix Inequalities*", American Mathematical Monthly, Vol.92, No.4, hal.277-280.

M.Dorigo, T.Stutzle. 2004. "*Ant Colony optimization*". 2004.

Mahmudy, W.F. 2013. *Modul Algoritma Evolusi Semester Ganjil 2013-2014.* Universitas Brawijaya Malang.

Maickel, Tuegeh, Soeprijanto dan Purnomo, Mauridhi H. 2009. "*Modified Improved Particle Swarm Optimization For Optimal*". 2009.

Munir, Rinaldi. 2005. *Matematika Diskrit.* Bandung: Deepublish.

Sedighizadeh, Davoud and Masehian, Ellips . 2009. "*Particle Swarm Optimization Methods, Taxonomy and Applications*". 2009.

Shigehiro, Yuji, Katsura, T., & Masuda, T. 2010. *An Application of Particle Swarm Optimization to Traveling Salesman Problem.* IEEE.