



TUGAS AKHIR - KS141501

**PENERAPAN ALGORITMA GENETIKA HYPER-HEURISTIC
UNTUK OPTIMASI RENCANA PERJALANAN MENGGUNAKAN
ANGKUTAN UMUM SEBAGAI SEBUAH ORIENTEERING
PROBLEM: STUDI KASUS BIS KOTA DI KOTA SURABAYA**

**A HYPER-HEURISTIC GENETIC ALGORITHM FOR
OPTIMIZING TRAVEL PLAN USING PUBLIC
TRANSPORTATION AS AN ORIENTEERING PROBLEM: CASE
STUDY OF CITY BUS IN SURABAYA CITY**

MUHAMMAD IQBAL IMADUDDIN
NRP 0521 14 400 0116

Dosen Pembimbing
Prof. Ir. Arif Djunaidy, M.Sc., Ph.D
Ahmad Mukhlason, S.Kom., M.Sc., Ph.D

Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

Halaman ini sengaja dikosongkan

TUGAS AKHIR - KS141501

**PENERAPAN ALGORITMA GENETIKA HYPER-
HEURISTIC UNTUK OPTIMASI RENCANA
PERJALANAN MENGGUNAKAN ANGKUTAN UMUM
SEBAGAI SEBUAH ORIENTEERING PROBLEM:
STUDI KASUS BIS KOTA DI KOTA SURABAYA**

**MUHAMMAD IQBAL IMADUDDIN
NRP 0521 14 400 0116**

**Dosen Pembimbing
Prof. Ir. Arif Djunaidy, M.Sc., Ph.D
Ahmad Mukhlason, S.Kom., M.Sc., Ph.D**

**Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018**

Halaman ini sengaja dikosongkan

FINAL PROJECT - KS141501

**A HYPER-HEURISTIC GENETIC ALGORITHM FOR
OPTIMIZING TRAVEL PLAN USING PUBLIC
TRANSPORTATION AS AN ORIENTEERING
PROBLEM: CASE STUDY OF CITY BUS IN
SURABAYA**

**MUHAMMAD IQBAL IMADUDDIN
NRP 0521 14 4000 0116**

Supervisor:

**Prof. Ir. Arif Djunaidy, M.Sc., Ph.D
Ahmad Mukhlason, S.Kom., M.Sc., Ph.D**

**Departement of Information System
Faculty of Information and Communication Technology (ICT)
Institut Teknologi Sepuluh Nopember
Surabaya 2018**

Halaman ini sengaja dikosongkan

LEMBAR PENGESAHAN

PENERAPAN ALGORITMA GENETIKA HYPER- HEURISTIC UNTUK OPTIMASI RENCANA PERJALANAN MENGGUNAKAN ANGKUTAN UMUM SEBAGAI SEBUAH ORIENTEERING PROBLEM: STUDI KASUS BIS KOTA DI KOTA SURABAYA

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

MUHAMMAD IOBAL IMADUDDIN

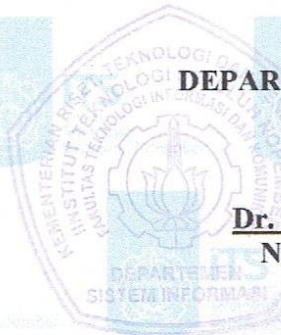
NRP. 0521 14 4000 0116

Surabaya, 18 Juli 2018

**KEPALA
DEPARTEMEN SISTEM INFORMASI**

Dr. Ir. Aris Tjahyanto, M.Kom.

NIP 19650310 199102 1 001



Halaman ini sengaja dikosongkan

LEMBAR PERSETUJUAN

PENERAPAN ALGORITMA GENETIKA HYPER- HEURISTIC UNTUK OPTIMASI RENCANA PERJALANAN MENGGUNAKAN ANGKUTAN UMUM SEBAGAI SEBUAH ORIENTEERING PROBLEM: STUDI KASUS BIS KOTA DI KOTA SURABAYA

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:


MUHAMMAD IQBAL IMADUDDIN
NRP. 0521 14 4000 0116

Disetujui Tim Penguji: Tanggal Ujian: 12 Juli 2018
Periode Wisuda: September 2018 -


Prof. Ir. Arif Djunaidy, M.Sc., Ph.D.


(Pembimbing I)

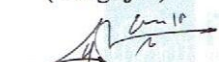
Ahmad Mukhlason, S.Kom., M.Sc., Ph.D.


(Pembimbing II)

Wiwik Anggraeni S.Si., M.Kom.


(Penguji I)

Faizal Mahananto, S.Kom., M.Eng., Ph.D.


(Penguji II)

Halaman ini sengaja dikosongkan

**PENERAPAN ALGORITMA GENETIKA HYPER-
HEURISTIC UNTUK OPTIMASI RENCANA
PERJALANAN MENGGUNAKAN ANGKUTAN UMUM
SEBAGAI SEBUAH ORIENTEERING PROBLEM:
STUDI KASUS BIS KOTA DI KOTA SURABAYA**

Nama Mahasiswa : Muhammad Iqbal Imaduddin
NRP : 0521 14 4000 0116
Jurusan : Sistem Informasi FTIK-ITS
Pembimbing 1 : Prof. Ir. Arif Djunaidy, M.Sc,
Ph.D
Pembimbing 2 : Ahmad Mukhlason, S.Kom,
M.Sc, Ph.D

ABSTRAK

Kota Surabaya memiliki beragam tempat yang dapat dikunjungi oleh wisatawan. Mulai dari wisata belanja, wisata religi, taman hiburan, museum, monument, dan tempat-tempat bersejarah. Mayoritas wisatawan yang berkunjung ke Kota Surabaya masih menggunakan kendaraan pribadi sehingga menambah kemacetan di jalan-jalan raya yang ada di Kota Surabaya. Kepadatan ini bisa semakin parah saat akhir pekan maupun saat musim liburan. Padahal untuk mengunjungi tempat-tempat wisata yang ada di Kota Surabaya bisa menggunakan transportasi umum seperti bis kota dan angkot. Pemerintah Kota Surabaya telah berusaha memperbaiki sarana dan layanan transportasi yang beroperasi di Kota Surabaya. Untuk mendukung program milik Pemerintah Kota Surabaya tersebut, maka dalam penelitian ini dilakukan proses optimasi rute perjalanan wisata dengan menggunakan bis kota yang beroperasi setiap hari di Kota Surabaya.

Sebagai studi kasusnya dipilih sepuluh trayek bis kota yang beroperasi di Kota Surabaya. Data waktu tempuh selama perjalanan didapatkan melalui Google Maps. Berdasarkan

data tersebut, dibangun sebuah solusi menggunakan Orienteering Problem. Dalam penelitian ini dihasilkan model jejaring dari sepuluh trayek tersebut serta model matematisnya. Solusi optimal dari permasalahan tersebut didapatkan dengan menggunakan Algoritma Genetika Hyper-Heuristic dengan bahasa pemrograman Java.

Untuk percobaan, dilakukan perbandingan hasil pencarian solusi terbaik menggunakan tiga algoritma, yaitu Algoritma Genetika Hyper-Heuristic, Algoritma Genetika, dan Algoritma Genetika Modifikasi. Hasilnya, algoritma genetika hyper-heuristic memberikan solusi nilai fitness sebesar 1125, algoritma genetika sebesar 804 dan algoritma genetika modifikasi sebesar 892. Hasil ini menunjukkan bila algoritma genetika hyper-heuristic memberikan solusi yang lebih baik dibanding dua algoritma lainnya untuk menyelesaikan permasalahan orienteering problem.

Kata kunci: Transportasi Umum, Orienteering Problem, Optimasi, Algoritma Genetika, Hyper-Heuristic

**A HYPER-HEURISTIC GENETIC ALGORITHM FOR
OPTIMIZING TRAVEL PLAN USING PUBLIC
TRANSPORTATION AS AN ORIENTEERING
PROBLEM: CASE STUDY OF CITY BUS IN
SURABAYA**

Nama Mahasiswa : **Muhammad Iqbal Imaduddin**
NRP : **0521 14 4000 0116**
Jurusan : **Sistem Informasi FTIK-ITS**
Pembimbing 1 : **Prof. Ir. Arif Djunaidy, M.Sc,
Ph.D**
Pembimbing 2 : **Ahmad Mukhlason, S.Kom,
M.Sc, Ph.D**

ABSTRACT

The city of Surabaya has a variety of places that can be visited by tourists. Starting from shopping center, religious tourism, amusement parks, museums, monuments, and historic places. The majority of tourists who visit the city of Surabaya still use private vehicles that increase the traffic jamming in the existing highways in the city of Surabaya. This density can get worse during weekends or during the holiday season. To visit tourist attractions in the city of Surabaya can use public transportation such as city buses and other public transportations. Surabaya City Government has tried to improve transportation facilities and services that operate in the city of Surabaya. To support the program owned by Surabaya City Government, then in this research is done optimization process of travel route by using city bus that operate daily in Surabaya.

As a case study selected ten city bus routes that operate in the city of Surabaya. Travel time data is obtained through Google Maps. Based on these data, built a solution using the Orienteering Problem. In this study, the network model of the ten routes and mathematical models are produced. The

optimal solution of the problem is obtained by using Hyper-Heuristic Genetic Algorithm with Java programming language.

For the experiment, the best results were compared using three algorithms, the Hyper-Heuristic Genetic Algorithm, Genetic Algorithm, and Modified Genetic Algorithm. As a result, the hyper-heuristic genetic algorithm gave solution with 1125 fitness score, genetic algorithm with 804 fitness score and modified genetic algorithm with 892 fitness score. These results suggest that the hyper-heuristic genetic algorithm gave the better solution than the other two algorithms for orienteering problems.

Keywords: Public Transportation, Orienteering Problem, Optimization, Genetic Algorithm, Hyper-Heuristic

KATA PENGANTAR

Puji syukur selalu penulis panjatkan kepada Allah SWT atas segala berkat, rahmat serta karunia-Nya serta shalawat juga salam selalu penulis panjatkan kepada Rasulullah Muhammad SAW, sehingga penulis dapat menyelesaikan pembuatan buku Tugas Akhir dengan judul:

PENERAPAN ALGORITMA GENETIKA HYPER- HEURISTIC UNTUK OPTIMASI RENCANA PERJALANAN MENGGUNAKAN ANGKUTAN UMUM SEBAGAI SEBUAH ORIENTEERING PROBLEM: STUDI KASUS BIS KOTA DI KOTA SURABAYA

Sebagai salah satu syarat kelulusan pada Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember, Surabaya.

Secara khusus penulis juga menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Ibunda Dra. Nurkhafidloh dan Ayahanda Drs. Mohamad Tohir M.Pd.I, selaku kedua orang tua penulis, yang telah mendidik penulis dari lahir hingga saat ini serta memberikan doa-doa, kebaikan, kasih sayang dan dukungan yang tiada henti-hentinya kepada penulis.
2. Kakak dan Adik tersayang, Muhammad Ikhlasul Amal, Muhammad Ilham Hamidy, dan Amylia Nur Hamidah, yang selalu memberikan dukungan, kehangatan serta kebahagiaan kepada penulis.
3. Keluarga Besar Bani Abdul Aziz dan Bani Djamilin yang selalu memberikan doa-doa terbaik dan juga semangat kepada penulis.
4. Bapak Prof. Ir. Arif Djunaidy, M.Sc., Ph.D dan Bapak Ahmad Mukhlason, S.Kom., M.Sc., Ph.D selaku Dosen Pembimbing, yang telah memberikan ilmu, bimbingan, pengalaman dan juga motivasi kepada penulis
5. Ibu Mahendrawathi ER, S.T., M.Sc., Ph.D selaku Dosen Wali Penulis, yang banyak memberikan wawasan,

bimbingan serta motivasi kepada penulis selama menempuh pendidikan di Departemen Sistem Informasi, FTIK, ITS.

6. Ibu Wiwik Anggraeni S.Si., M.Kom., dan Bapak Faizal Mahananto, S.Kom., M.Sc., Ph.D, selaku Dosen Penguji, yang telah memberikan banyak kritik serta saran yang membangun kepada penulis.
7. Bapak Dr. Ir. Aris Tjahyanto, M.Kom dan Bapak Nisfu Asrul Sani S.Kom., M.Sc selaku Kepala Departemen Sistem Informasi dan Kepala Program Studi S-1 Sistem Informasi, ITS, yang telah memberikan banyak bantuan kepada penulis.
8. Seluruh Dosen dan Staf Tata Usaha serta Karyawan di Departemen Sistem Informasi, yang telah banyak memberikan ilmu dan pengalaman yang bermanfaat serta bantuan yang tidak terhitung jumlahnya kepada penulis selama masa perkuliahan.
9. Teman-teman seperjuangan di rumah tercinta 'E-Home': Fandhi, Bang In, Dewa, Umar, Faiz, Nopal, Oman, Joni, Hendro, Arep, Tatus, Wafu, Bintang dan Dito. Serta Graha, Rama, Adrian, Guntur, Mario, Fai, Endar dan Alfian yang telah memberikan banyak keceriaan, momen-momen tak terlupakan dan berbagi berbagai macam pengalaman kepada penulis.
10. Teman-teman sebimbingan Pasca Sarjana: Mbak Gals, Gusti, Bang Andi, Silfi dan Ujik, yang menjadi tempat diskusi dan belajar bersama, serta berbagi suka-duka selama pengerjaan tugas akhir penulis.
11. Teman-teman Kabinet Kolaborasi: Ammar, Cindy, Erma, Anisa, Adam, Devi, Azis, Fadel, Obik, Mita, April, Ary, Lia, Fufu, Khai, Sasha, Bram dan Arief, yang bersama-sama berjuang di himpunan, memberikan informasi dan motivasi kepada penulis.
12. Teman-teman KP Mas Iwan (MIC): Mas Iwan, Calvin, Udin, Tania, Fata, Khikas, Niken, Iyik, Hans, dan Iqma, yang selalu menemani perjuangan penulis di Departemen Sistem Informasi.

13. Adik-Adik KP Bro Imad: Fadhilah, Ervina, Dhila, Bagus, Puji, Rendra, Awal, Irshad, Ivan, Ratih, dan Yusuf, yang telah memberikan dukungan serta menjadi tempat untuk belajar dan berbagi pengalaman.
14. Teman-teman OSIRIS yang tercinta, yang bersama-sama berjuang dalam satu kapal, yang banyak memberikan momen-momen istimewa yang tak terlupakan dan pengalaman-pengalaman berharga selama perkuliahan.
15. Adik-adik LANNISTER dan ARTEMIS serta Kakak-kakak BELTRANIS, SOLARIS, dan BASILISK yang selalu menemani serta memberikan banyak ilmu, momen serta pengalaman selama masa perkuliahan.
16. Teman-teman Lab RDIB, yang berjuang bersama serta saling berbagi ilmu dan pengalaman dalam pengerjaan tugas akhir.
17. Saudara Dewa dan Fandhi, yang sudah banyak berkontribusi selama pengerjaan tugas akhir.
18. Adik Nur Laili Sholichah dan Fadhilah Rahmah Azizah, yang menjadi inspirasi bagi penulis untuk menjadi pribadi yang lebih baik.
19. Teman-teman IMM Al-Mutsaqqof dan Ranger, KISI, ITS Mengajar, ISE, GERIGI, dan INTERVAL yang telah banyak memberikan momen, ilmu serta pengalaman kepada penulis selama kuliah di ITS.
20. Serta berbagai pihak yang telah membantu penulis dalam pengerjaan tugas akhir ini yang tidak bisa disebutkan satu persatu di atas.

Penyusunan buku tugas akhir ini masih jauh dari kata sempurna, maka dari itu penulis menerima adanya kritik dan saran untuk perbaikan di masa mendatang. Semoga buku tugas akhir ini dapat memberikan manfaat bagi para pembaca.

Surabaya, 19 Juli 2018

Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

LEMBAR PENGESAHAN.....	vii
LEMBAR PERSETUJUAN.....	ix
ABSTRAK.....	xi
ABSTRACT.....	xiii
KATA PENGANTAR.....	xv
DAFTAR ISI.....	xix
DAFTAR TABEL.....	xxiii
DAFTAR GAMBAR.....	xxv
DAFTAR KODE.....	xxvii
BAB I PENDAHULUAN.....	1
1.1. Latar Belakang.....	1
1.2. Perumusan Masalah.....	3
1.3. Batasan Tugas Akhir.....	3
1.4. Tujuan Tugas Akhir.....	4
1.5. Manfaat Tugas Akhir.....	4
1.6. Relevansi.....	4
BAB II DASAR TEORI.....	7
2.1. Optimasi.....	7
2.2. Praproses Data.....	7
2.3. Algoritma Dijkstra.....	8
2.4. Orienteering Problem.....	8
2.5. Algoritma Genetika.....	10
2.6. Hyper-Heuristic.....	13
BAB III METODOLOGI.....	15
3.1. Diagram Metodologi.....	15
3.2. Identifikasi Permasalahan.....	16
3.3. Studi Literatur.....	16
3.4. Pengumpulan Data dan Informasi.....	16
3.5. Pra-proses Data.....	18
3.6. Pembuatan Model Menggunakan <i>Orienteering Problem</i>	18
3.7. Pembuatan Solusi Menggunakan Algoritma Genetika <i>Hyper-Heuristic</i>	19
3.7.1. Mendefinisikan Individu.....	20
3.7.2. Mendefinisikan Nilai <i>Fitness</i>	20

3.7.3.	Melakukan Pembangkitan Populasi Awal	20
3.7.4.	Melakukan Evaluasi <i>Fitness</i>	20
3.7.5.	Melakukan Proses Hyper-Heuristic	20
3.8.	Pelaksanaan Uji Coba dan Analisis	21
3.9.	Penyusunan Laporan Tugas Akhir.....	21
BAB IV	DESAIN APLIKASI	23
4.1.	Pengumpulan dan Deskripsi Data.....	23
4.2.	Pembuatan Model Jejaring	26
4.2.1.	Penentuan Titik dan Skor.....	28
4.2.2.	Penentuan Waktu Tempuh.....	29
4.2.3.	Asumsi Dalam Pembuatan Model Jejaring	30
4.3.	Model Matematika Permasalahan <i>Orienteering Problem</i>	30
4.3.1.	Variabel Keputusan.....	31
4.3.2.	Fungsi Tujuan	32
4.3.3.	Perumusan Batasan	33
4.4.	Penentuan Komponen Algoritma Genetika	35
4.4.1.	Populasi.....	35
4.4.2.	Individu	35
4.4.3.	Gen.....	35
4.5.	Desain Algoritma Genetika	36
4.5.1.	Desain Algoritma Genetika <i>Hyper-Heuristic</i>	36
4.5.2.	Desain Algoritma Genetika.....	40
4.5.3.	Desain Algoritma Genetika Modifikasi	43
4.6.	Desain Aplikasi.....	46
4.6.1.	Tampilan Aplikasi.....	46
4.6.2.	Penentuan Komponen Aplikasi.....	47
BAB V	IMPLEMENTASI.....	49
5.1.	Pembuatan Matriks Waktu Tempuh Keseluruhan	49
5.2.	Penerapan Algoritma Genetika <i>Hyper-Heuristic</i>	51
5.2.1.	Inisialisasi Parameter	52
5.2.2.	Pembangkitan Populasi Awal	52
5.2.3.	Evaluasi <i>Fitness</i>	55
5.2.4.	Melakukan Seleksi.....	56
5.2.5.	Penerapan <i>Hyper-Heuristic</i>	57
5.2.6.	Pemakaian Operator <i>Hyper-Heuristic</i>	57

5.2.7.	Pemberhentian Algoritma Genetika <i>Hyper-Heuristic</i>	62
5.3.	Pembuatan Aplikasi Pencarian Rute Wisata.....	63
5.3.1.	Inisialisasi Rute Perjalanan	63
5.3.2.	Menghasilkan Solusi Rute Wisata Optimal	64
BAB VI	HASIL DAN PEMBAHASAN.....	65
6.1.	Lingkungan Uji Coba.....	65
6.2.	Hasil Penggambaran Model Jejaring	66
6.3.	Hasil Pencarian Waktu Tempuh Keseluruhan	66
6.4.	Hasil Pencarian Solusi menggunakan Algoritma Genetika <i>Hyper-Heuristic</i>	66
6.4.1.	Validasi Pembangkitan Populasi Layak Awal	67
6.4.2.	Validasi Solusi Akhir	69
6.4.3.	Hasil Rekomendasi Rute Wisata Pada Aplikasi... 71	
6.5.	Hasil dan Pembahasan Uji Coba: Membandingkan dengan Algoritma Genetika	72
6.6.	Hasil dan Pembahasan Uji Coba: Merubah titik awal dan titik akhir	79
6.6.1.	Validasi Pembangkitan Populasi Layak Awal	79
6.6.2.	Validasi Solusi Akhir	81
6.6.3.	Hasil Rekomendasi Rute Wisata Pada Aplikasi... 82	
BAB VII	KESIMPULAN DAN SARAN	84
7.1.	Kesimpulan	84
7.2.	Saran	85
DAFTAR	PUSTAKA	87
BIODATA	PENULIS	89
LAMPIRAN A:	LOKASI DAN SKOR TIAP TITIK.....	A-1
LAMPIRAN B:	NILAI WAKTU TEMPUH TIAP TITIK.....	B-1
LAMPIRAN C:	VARIABEL KEPUTUSAN OP	C-1
LAMPIRAN D:	MATRIKS JARAK KESELURUHAN.....	D-1

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Tabel 4-1 Trayek Bis Kota yang Digunakan.....	23
Tabel 4-2 Lokasi dan Skor Tiap Titik	28
Tabel 4-3 Nilai Waktu Tempuh Tiap Titik	29
Tabel 4-4 Asumsi Dalam Pembuatan Model Jejaring.....	30
Tabel 4-5 Variabel Keputusan <i>Orienteering Problem</i>	31
Tabel 6-1 Perangkat Keras Yang Digunakan	65
Tabel 6-2 Perangkat Lunak Yang Digunakan	65
Tabel 6-3 Hasil Pembangkitan Populasi Awal.....	67
Tabel 6-4 Rute Wisata Optimal.....	70
Tabel 6-5 Perbandingan Performa AGHH, AG dan AG Modifikasi	73
Tabel 6-6 Penggunaan Operator Untuk Proses AGHH.....	77
Tabel 6-7 Penggunaan Operator Untuk Proses AG.....	78
Tabel 6-8 Penggunaan Operator Untuk Proses AGM.....	78

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

Gambar 1-1 Roadmap Penelitian Laboratorium Rekayasa Data dan Intelegensi Bisnis (RDIB), Departemen Sistem Informasi, ITS.....	5
Gambar 2-1 Siklus Algoritma Genetika oleh David Goldberg	13
Gambar 3-1 Contoh Hasil Pencarian Skor	18
Gambar 4-1 Penggambaran Jalur Bis Kota di Kota Surabaya	27
Gambar 4-2 Tampilan Aplikasi.....	47
Gambar 4-3 Hasil Pencarian Rute Wisata.....	48
Gambar 6-1 Potongan Hasil Matriks Keseluruhan.....	66
Gambar 6-2 Rekomendasi Rute Wisata Pada Aplikasi	71
Gambar 6-3 Grafik Performa AGHH dan AH (200 Iterasi)...	75
Gambar 6-4 Grafik Performa AGHH dan AH (400 Iterasi)...	75
Gambar 6-5 Grafik Performa AGHH dan AH (600 Iterasi)...	76
Gambar 6-6 Grafik Performa AGHH dan AH (800 Iterasi)...	76
Gambar 6-7 Grafik Performa AGHH dan AH (1000 Iterasi).	77
Gambar 6-8 Rekomendasi Rute Wisata Pada Aplikasi	82

Halaman ini sengaja dikosongkan

DAFTAR KODE

Kode 4-1 <i>Pseudo Code</i> Pembangkitan Populasi Awal Algoritma Genetika <i>Hyper-Heuristic</i>	37
Kode 4-2 <i>Pseudo Code Hyper-Heuristic Single Point Search</i>	38
Kode 4-3 <i>Pseudo Code</i> Pembangkitan Populasi Awal Algoritma Genetika	41
Kode 4-4 <i>Pseudo Code</i> Pembangkitan Populasi Awal Algoritma Genetika Modifikasi	44
Kode 5-1 Algoritma Dijkstra (1)	49
Kode 5-2 Algoritma Dijkstra (2)	50
Kode 5-3 Algoritma Dijkstra (3)	50
Kode 5-4 Algoritma Dijkstra (4)	51
Kode 5-5 Inisialisasi Parameter AGHH	52
Kode 5-6 Pembangkitan Solusi Awal	53
Kode 5-7 Pembuatan <i>Sublist</i>	54
Kode 5-8 Memberi Nilai <i>Fitness</i> dan Waktu Tempuh Pada Populasi	55
Kode 5-9 Mencetak Hasil Pembangkitan Populasi Awal	55
Kode 5-10 Cara Menghitung Nilai <i>Fitness</i>	56
Kode 5-11 Pencarian Individu Terbaik	56
Kode 5-12 Mencetak Individu Terbaik	57
Kode 5-13 Memulai Iterasi <i>Hyper-Heuristic</i>	57
Kode 5-14 Operator <i>Hyper-Heuristic Injection Cross-Over</i> ..	58
Kode 5-15 Operator <i>Hyper-Heuristic Injection Cross-Over</i> (2)	59
Kode 5-16 <i>Hyper-Heuristic Exchange Mutation</i>	60
Kode 5-17 <i>Hyper-Heuristic Add Mutation</i>	61
Kode 5-18 <i>Hyper-Heuristic Omit Mutation</i>	62
Kode 5-19 Pemberhentian Algoritma Genetika <i>Hyper-Heuristic</i>	63
Kode 5-20 Inisialisasi Rute Perjalanan	63
Kode 5-21 Proses Menghasilkan Solusi Rute Wisata Optimal	64

Halaman ini sengaja dikosongkan

BAB I

PENDAHULUAN

Pada bab pertama ini akan dijelaskan proses identifikasi masalah mengenai latar belakang masalah, perumusan masalah, batasan masalah, tujuan tugas akhir, serta manfaat kegiatan tugas akhir. Berdasarkan uraian pada bab ini, diharapkan mampu memberikan gambaran umum mengenai permasalahan dan pemecahan masalah pada tugas akhir.

1.1. Latar Belakang

Kota Surabaya merupakan salah satu kota besar dan juga kota penting di Indonesia. Setiap harinya, ada ribuan orang yang datang menuju Kota Surabaya untuk bekerja, transit maupun juga berwisata. Di Kota Surabaya sendiri terdapat puluhan objek wisata yang tersebar di 31 kecamatan. Wisatawan yang berkunjung ke Kota Surabaya mayoritas masih menggunakan kendaraan pribadinya masing-masing. Padahal di Kota Surabaya sendiri sering terjadi kemacetan lalu lintas, imbas dari banyaknya jumlah kendaraan yang beroperasi di Kota Surabaya[1]. Salah satu faktor yang membuat wisatawan lebih memilih menggunakan kendaraan pribadi adalah kurangnya kualitas dan kuantitas transportasi umum[2]. Transportasi umum di Kota Surabaya masih sebatas digunakan oleh orang-orang untuk pulang dan berangkat sekolah maupun bekerja.

Selain itu, transportasi umum masih belum bisa menjangkau berbagai macam tempat wisata yang ada di Kota Surabaya karena keterbatasan rutenya. Untuk itu diharapkan adanya rute wisata yang optimal sehingga bisa menjadi rekomendasi rute perjalanan wisata kepada para wisatawan dengan memakai transportasi umum yang beroperasi di Kota Surabaya.

Dengan adanya rekomendasi rute wisata menggunakan transportasi umum, diharapkan masyarakat asli Kota Surabaya dan wisatawan yang berkunjung ke Kota Surabaya bisa berganti menggunakan transportasi umum yang tersedia

daripada menggunakan kendaraan pribadi. Dengan begitu, diharapkan angka kemacetan di Kota Surabaya juga bisa berkurang, sehingga masyarakat lokal dan wisatawan yang datang bisa lebih nyaman berada di Kota Surabaya.

Orienteering Problem (OP) adalah salah satu cara yang dapat digunakan untuk mendukung pembuatan rekomendasi rute wisata. Cara kerja *Orienteering Problem* sendiri adalah dengan menggabungkan proses pemilihan titik dan menentukan jalur optimal dari setiap titik yang telah dipilih. Hal inilah yang membedakan *Orienteering Problem* dengan *Travelling Salesman Problem* (TSP). Dalam TSP, kita harus mengunjungi semua titik untuk mendapatkan jalur optimalnya, sedangkan dengan menggunakan OP kita tidak perlu mengunjungi semua titik yang ada sebelum mencapai tujuan[3] sehingga OP lebih baik untuk permasalahan yang akan diteliti pada tugas akhir ini.

Salah satu metode yang dapat digunakan dalam masalah optimasi adalah Algoritma Genetika (*Genetic Algorithm*). Algoritma ini sudah diuji coba dan dibandingkan dengan *Chao's heuristic*, *Tsiligirides's Stochastic Algorithm* dan *Artificial Neural Network* oleh M. F. Tasgetiren untuk mengoptimalkan sebuah masalah *Orienteering Problem*, lalu hasilnya adalah Algoritma Genetika memberikan hasil yang lebih baik dibanding tiga algoritma yang lain[4]. Jadi, penelitian di atas telah menunjukkan bahwa Algoritma Genetika bisa memberikan solusi yang baik untuk permasalahan *Orienteering Problem*.

Algoritma Genetika *Hyper-Heuristic* juga sudah pernah digunakan untuk menyelesaikan permasalahan *Team Orienteering Problem with Time Windows* untuk mengoptimasi rute perjalanan wisata dan mendapatkan hasil nilai *fitness* yang baik dan memuaskan[5].

Berdasarkan kelebihan-kelebihan dari penyelesaian masalah *Orienteering Problem* dengan metode Algoritma Genetika *Hyper-Heuristic* di atas, maka untuk tugas akhir ini akan digunakan *Orienteering Problem* untuk membuat solusi dan Algoritma Genetika *Hyper-Heuristic* untuk mencari solusi yang optimal. Solusi akhir dari tugas akhir ini adalah rekomendasi rute wisata optimal yang bisa dilalui menggunakan bis kota yang beroperasi setiap hari di Kota Surabaya bagi masyarakat lokal dan wisatawan.

1.2. Perumusan Masalah

Permasalahan yang akan diselesaikan pada tugas akhir ini adalah:

- a. Penyelesaian permasalahan *orienteering problem* rute perjalanan wisata dengan bis kota di Kota Surabaya.
- b. Desain algoritma genetika *hyper-heuristic* yang sesuai untuk permasalahan *orienteering problem*.

1.3. Batasan Tugas Akhir

Batasan permasalahan yang digunakan dalam tugas akhir ini adalah:

- a. Objek penelitian dalam tugas akhir ini adalah transportasi umum, yaitu bis kota yang beroperasi setiap hari di Kota Surabaya.
- b. Jumlah trayek bis kota yang akan dipakai dalam tugas akhir ini ada sepuluh, dengan daerah asal-tujuan beserta kodenya adalah, Purabaya – Ngagel – Terminal Pecindilan (A), Terminal Purabaya – Terminal Bratang (D), Terminal Purabaya – Terminal Joyoboyo (E), Terminal Purabaya – Terminal Jembatan Merah Plaza (F), Terminal Purabaya – Terminal Perak (P1), Terminal Sidoarjo – Terminal JMP Via Tol (P3), Terminal Purabaya – Terminal Perak Via Tol (P4), Terminal Purabaya – Terminal JMP Via Tol (P5), Terminal Purabaya – Terminal Tambak Oso Wilangun (P6), dan Terminal Purabaya – Terminal Tambak Oso Wilangun Via Tol (P8)[6].

- c. Data yang digunakan dalam tugas akhir ini adalah waktu tempuh antar titik/objek yang didapatkan melalui aplikasi *Google Maps* dan tidak memperhatikan ada tidaknya kemacetan lalu lintas.
- d. Dalam tugas akhir ini hanya melibatkan tempat-tempat wisata, monumen, pusat perbelanjaan, museum, taman dan tempat bersejarah serta titik-titik pertemuan antar rute bis kota yang bisa ditempuh dengan menggunakan bis kota yang ada di Kota Surabaya.
- e. Dalam penelitian ini tidak memperhitungkan sisi psikologis dan finansial dari penumpang bis kota.
- f. Waktu tempuh untuk rekomendasi rute wisata pada tugas akhir ini tidak memperhitungkan waktu yang dihabiskan wisatawan di objek wisata tersebut dan waktu menunggu bis untuk pergantian trayek.

1.4. Tujuan Tugas Akhir

Tujuan dari tugas akhir ini adalah membuat rekomendasi rute perjalanan yang optimal untuk mengunjungi berbagai tempat wisata, pusat perbelanjaan dan bangunan bersejarah yang ada di Kota Surabaya dengan bis kota menggunakan *Orienteering Problem* serta mendapatkan solusi optimal dari permasalahan tersebut menggunakan Algoritma Genetika *Hyper-Heuristic*.

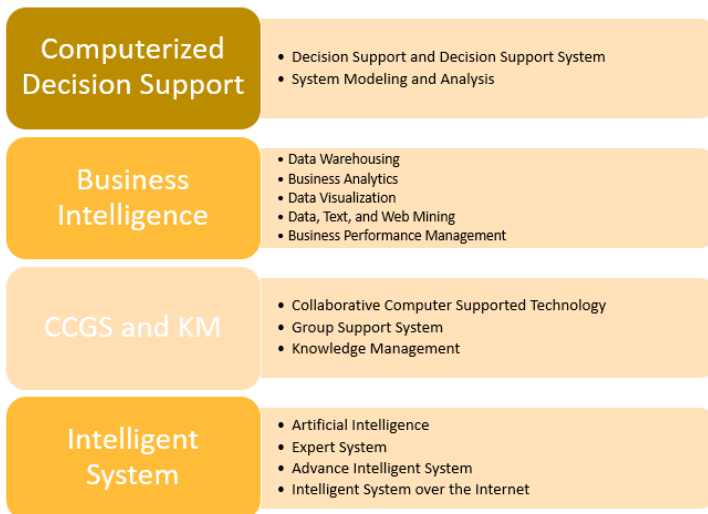
1.5. Manfaat Tugas Akhir

Manfaat yang diharapkan dari pengerjaan tugas akhir ini adalah masyarakat dan wisatawan mendapatkan rekomendasi rute wisata yang optimal saat berpergian menggunakan bis kota di Kota Surabaya serta hasil dari tugas akhir ini bisa diimplementasikan di kota-kota lain yang ada di Indonesia.

1.6. Relevansi

Hasil dari tugas akhir ini diharapkan dapat digunakan sebagai rekomendasi rute wisata menggunakan bis kota yang beroperasi setiap hari di Kota Surabaya. Tugas akhir ini juga dapat digunakan sebagai sumber pustaka untuk penelitian selanjutnya mengenai optimasi rute dengan teknik

Orienteering Problem maupun teknik lain yang memungkinkan, juga sebagai sumber pustaka untuk penggunaan Algoritma Genetika *Hyper-Heuristic* untuk proses optimasi kedepannya. Tugas akhir ini diharapkan juga bisa membantu warga serta wisatawan yang berkunjung ke Surabaya menjadi lebih mudah untuk bepergian menggunakan transportasi umum khususnya bis kota. Topik pada tugas akhir ini terkait dengan beberapa mata kuliah yang ada di Departemen Sistem Informasi, yaitu Sistem Pendukung Keputusan, Sistem Cerdas, Riset Operasi, dan Riset Operasi Lanjut. Selain itu topik tugas akhir ini jugasesuai dengan bidang ilmu riset operasi yang menjadi cakupan *research roadmap* pada laboratorium Rekayasa Data dan Inteligensi Bisnis (RDIB) yaitu *System Modeling and Analysis*.



Gambar 1-1 Roadmap Penelitian Laboratorium Rekayasa Data dan Inteligensi Bisnis (RDIB), Departemen Sistem Informasi, ITS

Halaman ini sengaja dikosongkan

BAB II DASAR TEORI

Pada bab kedua ini dijelaskan mengenai dasar teori yang dijadikan sebagai landasan dalam pengerjaan tugas akhir ini.

2.1. Optimasi

Secara istilah, optimasi adalah disiplin matematis yang menyangkut proses untuk menemukan sesuatu yang ekstrim (minimum ataupun maksimum) dari angka, fungsi, atau sistem. Para filsuf dan matematikawan kuno di masa lalu menciptakan fondasinya dengan mendefinisikan bagaimana nilai yang optimal dari beberapa permasalahan seperti angka, bentuk geometri optik, fisika, astronomi, kualitas kehidupan manusia dan lain-lainnya[7].

Sedang menurut KBBI, optimasi (optimalisasi) berasal dari kata dasar optimal yang berarti terbaik, tertinggi, paling menguntungkan, menjadikan paling baik, menjadikan paling tinggi, pengoptimalan proses, cara, perbuatan mengoptimalkan (menjadikan paling baik, paling tinggi, dan sebagainya)[8].

Sedang menurut kamus Oxford, optimasi adalah *“The action of making the best or most effective use of a situation or resource.”* Jika diterjemahkan dalam Bahasa Indonesia adalah *“Sebuah aksi untuk mendapatkan yang terbaik atau paling efektif dari situasi atau sumber daya yang dimiliki”*[9].

2.2. Praproses Data

Tahap praproses data merupakan proses untuk mempersiapkan data mentah agar bisa digunakan untuk proses selanjutnya. Pada umumnya, praproses data dilakukan dengan cara mengeliminasi data yang tidak sesuai atau mengubah data menjadi bentuk yang lebih mudah diproses oleh sistem[10]. Proses ini diperlukan karena data mentah masih belum siap digunakan untuk proses pengolahan data karena data mentah masih memiliki beberapa kekurangan. Salah satu cara untuk

mengubah data mentah menjadi data yang berkualitas adalah dengan melakukan transformasi data[11].

2.3. Algoritma Dijkstra

Algoritma Dijkstra dinamai sesuai dengan nama penemunya yaitu Edsger Dijkstra. Algoritma Dijkstra menggunakan prinsip greedy, dimana pada setiap langkah dipilih sisi dengan bobot (jarak) minimum yang menghubungkan sebuah simpul yang sudah terpilih dengan simpul lain yang belum terpilih[12]. Algoritma ini bertujuan untuk menemukan jalur terpendek berdasarkan bobot terkecil dari satu titik ke titik lainnya dengan ketentuan bahwa setiap garis penghubung antar titik yang terlibat tidak boleh bernilai negatif.

2.4. Orienteering Problem

Sejarah *Orienteering Problem* (OP) berasal dari sebuah olahraga *outdoor* yang bernama *Orienteering*. Olahraga ini dimainkan dengan cara setiap peserta harus mencari beberapa *checkpoint* yang disediakan sebelum sampai ke garis finis. Di setiap *checkpoint* disediakan poin atau skor yang nantinya akan diakumulasikan untuk mencari siapa yang mendapat poin atau skor yang paling tinggi dan menjadi pemenang. Karena tidak mungkin untuk mengunjungi setiap *checkpoint* yang ada (keterbatasan waktu dan tenaga), maka peserta harus dengan bijak memilih *checkpoint* mana yang akan dia kunjungi agar mendapat poin yang maksimal dengan usaha minimal. Hal inilah yang menjadi awal dari lahirnya metode *Orienteering Problem*[3].

Penjabaran mengenai OP dapat dijelaskan seperti berikut. Jika dimisalkan terdapat satu set titik $N = \{1, \dots, |N|\}$ dimana setiap titik $i \in N$ mempunyai skor non-negative S_i . 1 sebagai titik awal dan $|N|$ sebagai titik akhir. Tujuan dari OP seperti dijelaskan sebelumnya adalah untuk menentukan jalur yang optimal dengan mengunjungi beberapa titik sesuai dengan waktu yang telah ditentukan (dinotasikan dengan T_{max}). Selain itu, OP ditujukan untuk memaksimalkan total skor yang

dikumpulkan dari setiap kunjungan. Maka dari itu, diasumsikan tiap kunjungan ke sebuah titik (*checkpoint*) dapat menambah atau mengurangiskor dan setiap titik hanya boleh dikunjungi paling banyak satu kali. Waktu tempuh dari titik i ke titik j dinotasikan dengan t_{ij} [13].

Dalam pemrograman integer, OP dapat diformulasikan dengan variable keputusan $X_{ij} = 1$ (apabila kunjungan ke titik i diikuti dengan kunjungan ke titik j), dan jika tidak maka $X_{ij} = 0$. Fungsi tujuan dari OP dapat dirumuskan seperti yang ditunjukkan dalam persamaan (2.1)[3].

$$\text{Maximize } \sum_{i=1}^{N-1} \sum_{j=2}^N S_i X_{ij} \quad (2.1)$$

Fungsi tujuan pada persamaan (2.1) bertujuan untuk memaksimalkan skor yang dapat dikumpulkan selama perjalanan dari titik awal sampai dengan titik akhir. Namun, OP sendiri memiliki beberapa batasan yang harus di penuh seperti 5 rumus berikut[3].

$$\sum_{j=2}^{|N|} X_{1j} = \sum_{i=1}^{|N|-1} X_{i|N|} = 1 \quad (2.2)$$

$$\sum_{i=1}^{|N|-1} X_{ik} = \sum_{j=2}^{|N|} X_{kj} \leq 1; \forall k = 2, \dots, (|N| - 1) \quad (2.3)$$

$$\sum_{i=1}^{|N|-1} \sum_{j=2}^{|N|} t_{ij} X_{ij} \leq T_{max} \quad (2.4)$$

$$2 \leq u_i \leq |N|; \forall i = 2, \dots, |N| \quad (2.5)$$

$$u_i - u_j + 1 \leq (|N| - 1)(1 - X_{ij}); \forall i = 2, \dots, |N| \quad (2.6)$$

Batasan (2.2) bertujuan untuk memastikan jalur dimulai dari titik pertama dan akan berhenti di titik ke- $|N|$. Batasan (2.3) bertujuan untuk memastikan setiap jalur terhubung dan menjamin tiap titik hanya dapat dikunjungi maksimal satu kali. Batasan (2.4) bertujuan untuk membatasi total waktu tempuh sesuai dengan T_{\max} . Nilai T_{\max} sendiri akan disesuaikan sesuai kebutuhan. Batasan (2.5) dan (2.6) bertujuan untuk mencegah terjadinya *subtour*, yaitu kondisi dimana lintasan dimulai dan diakhiri pada titik yang sama[3].

2.5. Algoritma Genetika

Algoritma genetika merupakan salah satu tipe algoritma evolusi yang diilhami dari ilmu genetika. Algoritma genetika memiliki kemampuan dalam mengatasi permasalahan yang kompleks di berbagai bidang diantaranya fisika, biologi, ekonomi, sosiologi dan lain-lain. Bahkan di era modern sekarang, algoritma genetika dapat diterapkan pada berbagai bidang industri[14].

Algoritma ini juga mempunyai kemampuan untuk menyelesaikan persoalan optimasi. Kemampuan ini didukung oleh tiga operator genetik yaitu reproduksi, kawin silang (rekombinasi) dan mutasi[15].

Ada beberapa hal yang harus dilakukan untuk menjalankan optimasi menggunakan algoritma genetika, yaitu:

- a. Mendefinisikan individu. Individu, merepresentasikan salah satu solusi yang mungkin dari suatu permasalahan. Berikut dijabarkan istilah-istilah penting terkait dengan individu[16].
 - Gen, merupakan unit terkecil dari individu, membawa nilai yang akan digunakan untuk proses kawin silang dan mutasi.

- Kromosom, merupakan deretan dari beberapa gen yang bersatu.
 - Populasi, merupakan sekumpulan individu yang akan diproses bersama dalam satu generasi.
- b. Mendefinisikan nilai *fitness*, sebagai ukuran baik-tidaknya sebuah individu atau baik-tidaknya solusi yang didapatkan. Nilai ini dijadikan acuan untuk mencapai solusi optimum dalam algoritma genetika. Algoritma genetika bertujuan untuk mencari individu dengan nilai *fitness* yang paling baik. Nilai *fitness* diformulasikan sesuai dengan kondisi dari penelitian tersebut. Misal tujuan penelitian adalah memaksimalkan skor, maka nilai *fitness* bisa berupa total skor suatu individu[16].

Algoritma genetika memiliki beberapa tahap dalam satu generasi. Tahapan-tahapan tersebut digambarkan ke dalam sebuah siklus oleh David Goldberg seperti yang ada pada gambar 2-1[16].

- a. Pembangkitan Populasi Awal
Proses ini merupakan tahapan membangkitkan sejumlah individu secara acak. Ukuran dari populasi awal tergantung pada kondisi permasalahan yang akan diselesaikan.
- b. Evaluasi *fitness*
Pada tahap ini dilakukan evaluasi pada populasi yang telah dibangkitkan. Tujuannya untuk mengetahui pencapaian nilai optimal dari populasi tersebut. Individu dengan nilai *fitness* yang tinggi akan bertahan hidup dan individu dengan nilai *fitness* yang rendah akan tersingkir dari populasi.
- c. Proses Seleksi
Proses ini adalah menentukan individu yang akan digunakan untuk proses kawin silang. Terdapat beberapa teknik yang dapat digunakan untuk proses seleksi, yaitu seleksi berdasarkan ranking *fitness*

(*Rank-based Fitness*), seleksi dengan Roda *Roulette* (*Roulette Wheel Selection*), seleksi Lokal (*Local Selection*), seleksi dengan Pemotongan (*Truncation Selection*), seleksi dengan Turnamen (*Tournament Selection*)[15].

d. Melakukan Kawin Silang

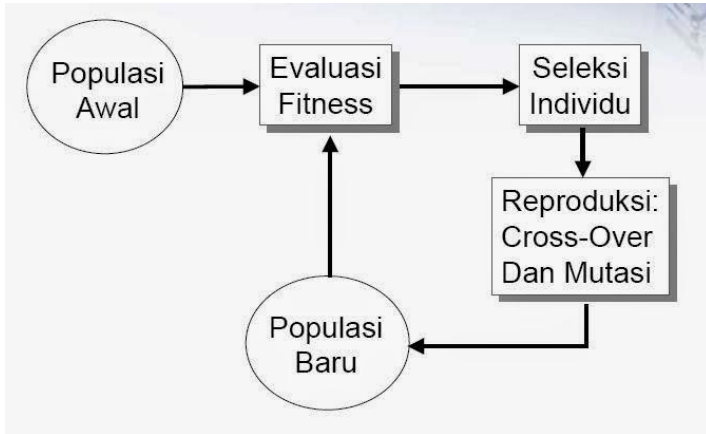
Kawin silang atau rekombinasi adalah proses untuk menyilangkan dua kromosom sehingga membentuk kromosom baru yang harapannya lebih baik dari pada induknya[17]. Ada beberapa macam proses rekombinasi yang ada pada algoritma genetika, yaitu kawin silang diskret, kawin silang menengah, kawin silang garis, kawin silang satu titik, kawin silang banyak titik, kawin silang seragam, kawin silang dengan permutasi, dan kawin silang injeksi[4].

e. Melakukan Mutasi

Mutasi adalah proses perubahan nilai acak yang sangat kecil dengan probabilitas rendah pada variabel keturunan[17]. Proses ini berfungsi untuk menggantikan gen yang hilang saat proses seleksi serta memunculkan gen baru yang belum ada saat pembangkitan populasi awal. Ada beberapa macam proses mutasi yang ada pada algoritma genetika, yaitu mutasi bilangan real dan mutasi biner[15].

f. Terbentuk populasi baru.

Populasi baru ini nanti akan mengalami proses seperti yang dijelaskan sebelumnya sampai akhirnya algoritma genetika sampai pada generasi yang terakhir dan berhenti.



Gambar 2-1 Siklus Algoritma Genetika oleh David Goldberg

Hasil dari proses-proses di atas adalah terbentuknya populasi-populasi baru. Nantinya, populasi baru tersebut akan menjadi populasi awal untuk generasi (iterasi) yang diproduksi selanjutnya. Proses perulangan generasi tersebut akan terus berlanjut hingga didapatkan generasi yang sudah sesuai dengan kriteria optimal yang sudah ditentukan.

2.6. Hyper-Heuristic

Hyper-heuristic terdiri dari serangkaian pendekatan yang memiliki tujuan untuk mengotomatisasi metode heuristik untuk memecahkan masalah pencarian komputasi yang kompleks dan sulit. Ini karena komponen dan strategi pencarian dari proses *hyper-heuristic* dapat dengan mudah diterapkan dalam domain masalah yang berbeda. Investigasi empiris hingga kini menunjukkan bahwa *hyper-heuristic* adalah teknik yang bisa menghasilkan solusi dengan kualitas yang baik dalam waktu yang singkat[18].

Ada beberapa macam komponen untuk melakukan proses *hyper-heuristic*, yaitu Simple Random, Random Permutation, Greedy, Peekish, Choice Function, Reinforcement Learning

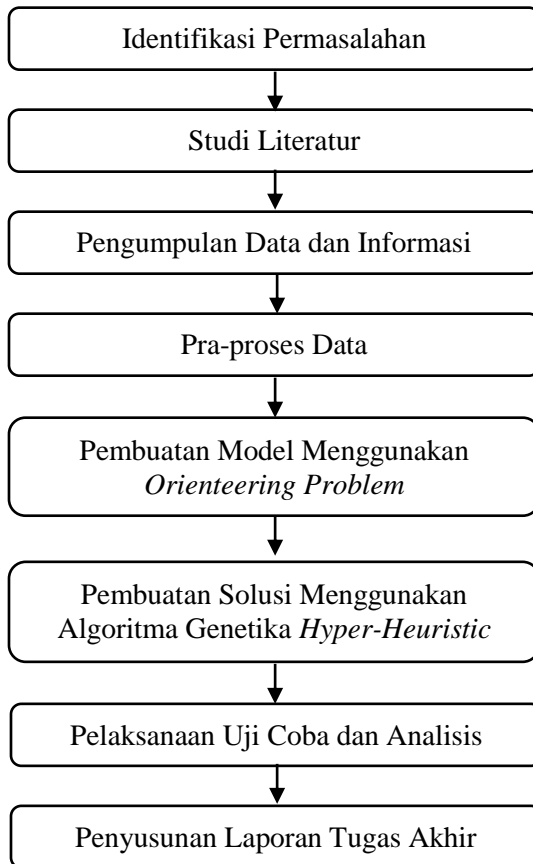
dan lain-lain[19]. Untuk kriteria penerimaan hasil dari proses *hyper-heuristic* juga ada beberapa cara, yakni *All Moves*, *Only Improvements*, *Improving and Equal* [19].

BAB III METODOLOGI

Pada bab ketiga ini dijelaskan mengenai metodologi yang digunakan untuk menyelesaikan tugas akhir.

3.1. Diagram Metodologi

Pada Gambar 3.1 di jelaskan alur metodologi yang akan digunakan untuk menyelesaikan penelitian tugas akhir :



Gambar 3.1 Metodologi Tugas Akhir

3.2. Identifikasi Permasalahan

Dalam pengerjaan tugas akhir, yang pertama di lakukan adalah mengidentifikasi masalah yang terjadi. Seperti yang sudah dijelaskan pada sub-bab latar belakang, salah satu masalah yang saat ini terjadi di Kota Surabaya adalah belum adanya rekomendasi rute wisata untuk para wisatawan yang ingin berkeliling Kota Surabaya menggunakan bis kota. Oleh karena itu, salah satu solusi yang bisa digunakan untuk mengatasinya adalah dengan membuat rekomendasi rute wisata di Kota Surabaya menggunakan bis kota sehingga wisatawan bisa memilih untuk menggunakan transportasi umum dibanding kendaraan pribadi. Jumlah trayek bis kota yang digunakan sebagai objek penelitian adalah 10 trayek[6].

3.3. Studi Literatur

Studi literatur dilakukan dengan mengumpulkan berbagai referensi seperti buku, jurnal terkait, penelitian sebelumnya, dan data-data dari website yang mendukung penyelesaian tugas akhir. Studi literatur dilakukan pada beberapa referensi yang sesuai dengan topik yang dipilih, yaitu mengenai optimasi jalur transportasi umum menggunakan metode tertentu untuk membuat formulasi penyelesaian dan mendapatkan solusi optimal berdasarkan model yang dibuat dengan algoritma tertentu.

Karena permasalahan yang dibahas adalah pembuatan model jejaring dan optimasi rute, maka diusulkan metode yang digunakan adalah *Orienteering Problem* dengan Algoritma Genetika *Hyper-Heuristic* sebagai cara untuk mendapatkan solusi rute yang optimal.

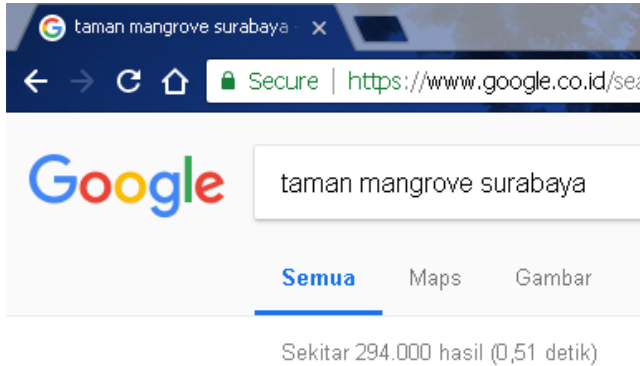
3.4. Pengumpulan Data dan Informasi

Setelah memahami konsep dari metode dan algoritma yang digunakan, langkah selanjutnya adalah mengumpulkan data. Berikut adalah sepuluh rute bis kota yang dipilih dalam penelitian tugas akhir ini :

- a. Bis Kota jurusan Terminal Purabaya – Semut – Terminal Pecindilan (A)
- b. Bis Kota jurusan Terminal Purabaya – Terminal Bratang (D)
- c. Bis Kota jurusan Terminal Purabaya – Terminal Joyoboyo (E)
- d. Bis Kota jurusan Terminal Purabaya – Terminal Jembatan Merah Plaza (F)
- e. Bis Kota jurusan Terminal Purabaya – Terminal Perak (P1)
- f. Bis Kota jurusan Terminal Sidoarjo – Terminal JMP Via Tol (P3)
- g. Bis Kota jurusan Terminal Purabaya – Terminal Perak Via Tol (P4)
- h. Bis Kota jurusan Terminal Purabaya – Terminal JMP Via Tol (P5)
- i. Bis Kota jurusan Terminal Purabaya – Terminal Tambak Oso Wilangun (P6)
- j. Bis Kota jurusan Terminal Purabaya – Terminal Tambak Oso Wilangun via Tol (P8)[6].

Dari kesepuluh jalur bis kota tersebut, diperlukan data mentah yang akan diproses ditahap selanjutnya. Data tersebut bisa didapatkan dari pencarian melalui mesin pencari *Google* dan aplikasi *Google Maps*, data yang dimaksud adalah :

- a. Data skor, yaitu poin dari setiap titik berdasarkan dari jumlah hasil pencarian yang dilakukan melalui mesin pencari *Google*. Dari jumlah tersebut, tiga angka terakhir dihilangkan sehingga nilai skor menjadi nilai ratusan dan tidak terlalu besar. Data skor ini diambil pada tanggal 9 Mei 2018.



Gambar 3-1 Contoh Hasil Pencarian Skor

Dari gambar 3-1 maka skor untuk taman mangrove Surabaya adalah 294.

- b. Data waktu tempuh dari satu titik ke titik yang lainnya melalui aplikasi *Google Maps*.

3.5. Pra-proses Data

Data mentah yang telah dikumpulkan perlu diolah agar data tersebut menjadi data yang berkualitas dan siap dipakai. Teknik praproses data yang digunakan pada penelitian ini adalah transformasi data[11] dengan menggunakan algoritma Dijkstra. Tujuannya adalah agar data mentah tadi diubah sesuai dengan format masukan yang bisa dibaca oleh aplikasi. Data masukan ini akan berupa data matriks sesuai jumlah titik yang digunakan.

3.6. Pembuatan Model Menggunakan *Orienteering Problem*

Ada beberapa hal yang harus diperhatikan sebelum proses pembuatan model, yaitu :

- a. Titik (*node*) adalah representasi dari tempat-tempat wisata yang ada di Kota Surabaya yang bisa ditempuh dengan menggunakan bis kota.

- b. Nilai pada sebuah garis disebut busur (*arc*). Busur adalah representasi dari waktu tempuh dari satu titik ke titik lainnya.
- c. Jumlah skor melambangkan jumlah hasil pencarian suatu tempat dengan mesin pencari *Google*. Skor digunakan untuk melihat apakah solusi yang ditemukan baik atau buruk mengingat semakin banyak skor berarti semakin banyak tempat terkenal yang bisa dikunjungi selama perjalanan menggunakan bis kota di Kota Surabaya.

Tahap selanjutnya adalah membuat solusi berdasarkan variabel-variabel yang ada, yaitu variabel keputusan, fungsi tujuan, dan batasan[3]. Perinciannya adalah seperti berikut :

a. Variabel Keputusan

Adalah kunjungan bis kota dari satu titik ke titik yang lainnya.

b. Fungsi Tujuan

Adalah untuk memaksimalkan jumlah skor yang dapat dicapai selama perjalanan menggunakan bis kota dari titik awal berangkat sampai ke titik akhir.

c. Batasan

- Batasan 1 : Menetapkan titik awal dan titik akhir perjalanan.
- Batasan 2 : Setiap titik saling terhubung dan setiap titik hanya dapat dikunjungi maksimal satu kali.
- Batasan 3 : Terdapat batasan waktu untuk satu kali perjalanan.
- Batasan 4 : Perjalanan tidak boleh berhenti di titik awal keberangkatan.

3.7. Pembuatan Solusi Menggunakan Algoritma Genetika *Hyper-Heuristic*

Setelah tahap pra-proses data menggunakan Algoritma Dijkstra, maka ditemukan jarak antar titik. Data matriks tersebut lalu digunakan sebagai masukan dalam penerapan Algoritma Genetika *Hyper-Heuristic* untuk menghasilkan

solusi optimal dari permasalahan rute wisata dengan bis kota di Kota Surabaya. Penjelasan selengkapnya ada pada poin-poin berikut:

3.7.1. Mendefinisikan Individu

Individu di sini adalah representasi dari sebuah solusi dari masalah yang ada. Individu berisikan gen yang mewakili titik-titik yang dikunjungi menggunakan bis kota. Titik awal dan akhir yang ada pada tugas akhir ini sudah ditentukan dan tidak akan berubah.

3.7.2. Mendefinisikan Nilai *Fitness*

Nilai *fitness* adalah representasi dari penilaian baik-tidaknya sebuah individu (solusi), semakin tinggi nilai *fitness* semakin kuat suatu individu. Nilai *fitness* digunakan untuk mengevaluasi setiap individu yang dibangkitkan. Pada tugas akhir ini nilai *fitness* berasal dari total skor suatu individu.

3.7.3. Melakukan Pembangkitan Populasi Awal

Populasi awal berisikan individu yang memenuhi kriteria dalam pemodelan OP. Pembangkitan ini dilakukan secara acak selama tidak melanggar batasan. Individu terbaik dari Pembangkitan Populasi Awal akan menjadi acuan dari penerimaan hasil *hyper-heuristic*.

3.7.4. Melakukan Evaluasi *Fitness*

Evaluasi *fitness* digunakan untuk mengevaluasi apakah individu di dalam populasi yang dibangkitkan sudah layak menjadi solusi optimal atau belum.

3.7.5. Melakukan Proses *Hyper-Heuristic*

Pada tahap ini dimulai proses *hyper-heuristic*. Proses ini akan terus berlangsung hingga iterasi yang ditentukan terpenuhi.

a. Seleksi individu untuk proses *Hyper-Heuristic*

Proses seleksi ini akan memilih individu dengan nilai *fitness* yang paling tinggi untuk dijadikan induk.

Individu terbaik ini akan digunakan pada tahap selanjutnya.

b. Pemilihan Operator *Hyper-Heuristic*

Individu pada proses sebelumnya akan diproses dengan operator *hyper-heuristic* yang dipilih secara random. Operator tersebut adalah kawin silang *injection*, mutasi *add*, mutasi *omit*[4], dan mutasi *exchange*[20].

c. Pembentukan Solusi Baru

Hasil dari penerapan operator *hyper-heuristic* adalah lahirnya individu baru. Individu ini nanti akan dievaluasi nilai fitness dan nilai waktu tempuhnya, apabila lebih baik dari individu solusi awal, maka dia akan menggantikan posisi individu solusi awal tersebut, namun jika tidak, maka individu solusi awal akan dipertahankan. Proses ini berulang sampai iterasi yang ditentukan.

d. Pengambilan Solusi Terbaik

Individu terbaik hasil dari proses *hyper-heuristic* pada iterasi terakhir akan menjadi solusi optimal untuk permasalahan yang dibahas pada tugas akhir ini.

3.8. Pelaksanaan Uji Coba dan Analisis

Uji coba dilakukan dengan membandingkan hasil pencarian solusi terbaik menggunakan algoritma genetika *hyper-heuristic*, algoritma genetika dan algoritma genetika modifikasi. Uji coba ini dilakukan beberapa kali dengan menggunakan iterasi/generasi yang berbeda-beda, yakni 200, 400, 600, 800 dan 1000. Hasil dari uji coba akan disajikan dalam bentuk tabel dan grafik perbandingan performa dari tiga algoritma di atas.

3.9. Penyusunan Laporan Tugas Akhir

Tahap terakhir adalah pembuatan laporan tugas akhir sebagai bentuk dokumentasi atas terlaksananya tugas akhir ini. Di dalam laporan tersebut mencakup:

a. Bab I Pendahuluan

Dalam bab ini dijelaskan mengenai latar belakang, rumusan dan batasan masalah, tujuan dan manfaat pengerjaan tugas akhir ini.

b. Bab II Dasar Teori

Dalam bab ini dijelaskan mengenai penelitian-penelitian serupa yang telah dilakukan serta teori – teori yang menunjang permasalahan yang dibahas pada tugas akhir ini.

c. Bab III Metodologi

Dalam bab ini dijelaskan mengenai tahapan – tahapan apa saja yang harus dilakukan dalam pengerjaan tugas akhir.

d. Bab IV Desain Aplikasi

Dalam bab ini dijelaskan mengenai rancangan penelitian, bagaimana penelitian dilakukan, pemilihan objek penelitian, dan sebagainya.

e. Bab V Implementasi

Dalam bab ini dijelaskan mengenai proses penelitian, langkah-langkah yang dilakukan dari awal sampai akhir dan sebagainya.

f. Bab VI Hasil dan Pembahasan

Dalam bab ini dijelaskan mengenai pembahasan dalam penyelesaian permasalahan yang dikerjakan pada penelitian tugas akhir ini.

g. Bab VII Kesimpulan dan Saran

Dalam bab ini dijelaskan mengenai kesimpulan dari penelitian dan juga saran yang ditujukan untuk penyempurnaan tugas akhir ini kedepannya.

BAB IV DESAIN APLIKASI

Pada bab keempat ini dijelaskan desain dari penelitian tugas akhir yang meliputi obyek dari penelitian dan bagaimana desain aplikasi dibuat.

4.1. Pengumpulan dan Deskripsi Data

Pengumpulan data dilakukan dengan mengambil data trayek bis kota dari website resmi Dinas Perhubungan Kota Surabaya. 10 trayek bis kota yang digunakan pada penelitian ini dijelaskan pada tabel 4-1.

Tabel 4-1 Trayek Bis Kota yang Digunakan

Kode	Rute
A	<u>Purabaya – Ngagel – Semut</u> Purabaya/Bungurasih – Ahmad Yani – Stasiun Wonokromo – Ngagel – Raya Gubeng – Stasiun Gubeng – Kusuma Bangsa – D. Gembong – Kalianyar – Pengampon - Bunguran – Putar Kya Kya (Kembang Jepun-Kapasan) – Gembong – Pecindilan – Kembali dengan rute yang sama
D	<u>Purabaya – Bratang</u> Berangkat : Purabaya/Bungurasih – Ahmad Yani – Jemursari – Prapen – (D. Panjang Jiwo) – Nginden – Bratang Kembali : Bratang – Bratang Jaya – Barata Jaya XIX – Barata Jaya XVII – Nginden - (D. Panjang Jiwo) – Prapen – Jemursari – Ahmad Yani – Purabaya/Bungurasih
E	<u>Purabaya – Joyoboyo</u> Purabaya/Bungurasih – Ahmad Yani – Wonokromo – Joyoboyo – Kembali dengan rute yang sama

F	<p><u>Purabaya - Kupang - Raden Saleh – JMP</u></p> <p>Berangkat : Surabaya/Bungurasih – Ahmad Yani – Wonokromo – Diponegoro – (D. Kupang) – Pasar Kembang – Arjuno – Semarang – Stasiun Pasar Turi - Raden Saleh – Bubutan – Indrapura – Rajawali – Jembatan Merah Plasa (JMP)</p> <p>Kembali : Jembatan Merah Plasa (JMP) – Jembatan Merah – Veteran – Pahlawan – Gemblongan – Siola – Praban – Bubutan – Raden Saleh – Stasiun Pasar Turi – Semarang – Arjuno – Pasar Kembang - (D. Kupang) – Diponegoro – Wonokromo – Ahmad Yani – Surabaya/Bungurasih</p>
P1	<p><u>Purabaya - Darmo/TP/Siola - Indrapura – Perak</u></p> <p>Berangkat : Surabaya/Bungurasih – Ahmad Yani – Wonokromo – Darmo – Urip Sumoharjo – Basuki Rahmat – (D. Tunjungan / TP) – Embong Malang – Blauran – Bubutan – Indrapura – Rajawali – Perak Barat – Prapat Kurung – Kalimas Baru – Perak (Pelabuhan)</p> <p>Kembali : Perak (Pelabuhan) – Kalimas Baru – Prapat Kurung – Perak Timur – Rajawali – Jembatan Merah Plasa (JMP) - Jembatan Merah – Veteran – Pahlawan – Kramat Gantung – Siola – Tunjungan – Gubernur Suryo – Panglima Sudirman – Bambu Runcing – Panglima Sudirman – Urip Sumoharjo – Darmo – Wonokromo – Ahmad Yani – Surabaya/Bungurasih</p>
P3	<p><u>Sidoarjo - Dupak - Rajawali – JMP</u></p> <p>Berangkat : Terminal Sidoarjo – Pahlawan – (Gor Delta Sidoarjo) – Pahlawan – Masuk Tol Sidoarjo Keluar Tol Dupak – Pasar Loak - Dupak – Pasar Turi – Bubutan – Indrapura – Rajawali – Jembatan</p>

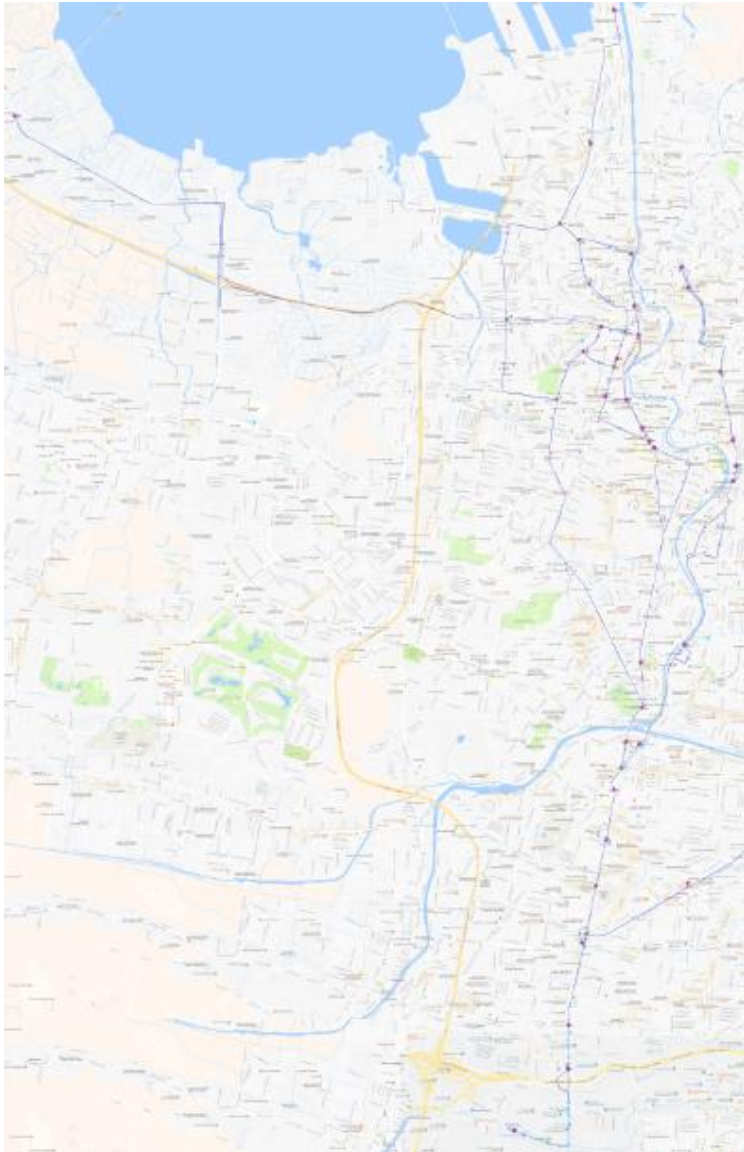
	<p>Merah Plasa (JMP)</p> <p>Kembali : Jembatan Merah Plasa (JMP) – Jembatan Merah – Veteran – Pahlawan – Tembaan – Pasar Turi – Dupak – Pasar Loak – Masuk Tol Dupak Keluar Tol Sidoarjo - Pahlawan – (Gor Delta Sidoarjo) – Pahlawan – Gajah Mada – Terminal Sidoarjo</p>
P4	<p><u>Purabaya - Dupak - Perak (Lewat Tol)</u></p> <p>Berangkat : Purabaya/Bungurasih – Masuk Tol Waru Keluar Tol Dupak - Pasar Loak – Dupak – Demak – Gresik Gadukan – Gresik - Perak Barat – Prapat Kurung – Kalimas Baru – Perak (Pelabuhan)</p> <p>Kembali : Perak (Pelabuhan) – Kalimas Baru – Prapat Kurung – Perak Timur – Gresik – Gresik Gadukan – Demak – Dupak – Pasar Loak – Masuk Tol Dupak Keluar Tol Waru – Purabaya/Bungurasih</p>
P5	<p><u>Purabaya - Dupak - JMP (Lewat Tol)</u></p> <p>Berangkat : Purabaya/Bungurasih – Masuk Tol Waru Keluar Tol Dupak - Pasar Loak – Dupak - Pasar Turi – Bubutan – Indrapura – Rajawali – Jembatan Merah Plasa (JMP)</p> <p>Kembali : Jembatan Merah Plasa (JMP) – Jembatan Merah – Veteran – Pahlawan – Tembaan – Pasar Turi – Dupak – Pasar Loak – Masuk Tol Dupak Keluar Tol Waru – Purabaya/Bungurasih</p>
P6	<p><u>Purabaya - Kupang - Demak - Tambak Oso Wilangun</u></p> <p>Purabaya/Bungurasih – Ahmad Yani – Wonokromo – Diponegoro – (D. Kupang) – Pasar Kembang – Arjuno – Tembok Dukuh – Demak – Dupak – Pasar Loak – Masuk Tol Dupak Keluar Tol Tandes –</p>

	Margomulyo – Tambak Osowilangun – Kembali dengan rute yang sama
P8	<u>Purabaya - Tambak Oso Wilangun (Lewat Tol)</u> Purabaya/Bungurasih – Masuk Tol Waru Keluar Tol Tandes – Margomulyo – Tambak Osowilangun – Kembali dengan rute yang sama

4.2. Pembuatan Model Jejaring

Pembuatan model jejaring diawali dengan menggambarkan rute trayek yang digunakan, setiap rute trayek digambarkan di atas peta digital menggunakan aplikasi Paint sebagai gambaran awal jalur tiap trayek sekaligus untuk mengetahui pada jalur mana saja tiap trayek bersinggungan.

Hasil dari penggambaran jalur tersebut dapat dilihat pada Gambar 4-1.



Gambar 4-1 Penggambaran Jalur Bis Kota di Kota Surabaya

4.2.1. Penentuan Titik dan Skor

Setelah tiap jalur trayek bis kota digambarkan pada peta digital, maka kemudian ditentukan letak titik pada jalur trayek. Yang berperan sebagai titik adalah tiap – tiap objek wisata yang dilewati bis kota. Tiap titik juga memiliki nilai skor yang digunakan untuk penyelesaian rute *Orienteering Problem*. Nilai skor pada titik dihitung dari jumlah hasil pencarian pada objek tersebut menggunakan mesin pencari *google*.

Tabel 4-2 merupakan potongan tabel lokasi dimana tiap titik berada beserta skornya. Untuk keseluruhan lokasi titik dan skor tiap titik dapat dilihat pada halaman LAMPIRAN A.

Tabel 4-2 Lokasi dan Skor Tiap Titik

Titik	Lokasi	Skor	Skor Total
1	Terminal Bungurasih	27	27
2	City of Tomorrow Mall	16	16
3	Masjid Al-Akbar	107	107
4	Taman Pelangi	8	8
5	Pertigaan Ahmad Yani Jemur Andayani	0	0
6	Taman Flora	7	7
7	DBL Arena	65	65
8	Maspion Square	22	22
9	Royal Plaza	239	239
10	Pertigaan Wonokromo	0	0
11	Stasiun Wonokromo	11	11
12	Marvel City	15	15
13	Monumen Kapal Selam	24	95
	Plaza Surabaya	71	
14	Stasiun Gubeng	34	34
15	Grand City Mall	69	69

4.2.2. Penentuan Waktu Tempuh

Setelah ditentukannya lokasi-lokasi wisata menjadi titik-titik, seluruh titik tersebut akan dihubungkan dengan sebuah garis yang memiliki waktu tempuh berbeda-beda. Waktu tempuh saat berangkat dan pulang dari satu titik ke titik yang lainnya dapat berbeda karena rute yang dilaluinya bisa saja berbeda.

Seluruh nilai waktu tempuh untuk tiap titik didapatkan menggunakan aplikasi *Google Maps* yang diakses melalui *web browser*.

Tabel 4-3 merupakan tabel nilai waktu tempuh masing – masing titik yang ada di dalam model jejaring. Untuk keseluruhan tabel waktu tempuh tiap titik dapat dilihat pada halaman LAMPIRAN B.

Tabel 4-3 Nilai Waktu Tempuh Tiap Titik

Titik Awal	Titik Akhir	Waktu tempuh (menit)
1	2	5
2	1	5
2	3	3
3	2	3
3	4	4
4	3	4
4	5	3
4	7	3
5	4	2
5	6	12
6	5	12
7	5	3
7	8	1
8	7	1

Titik Awal	Titik Akhir	Waktu tempuh (menit)
8	9	2
9	8	2
9	10	2
10	9	2
10	11	2
10	19	1

4.2.3. Asumsi Dalam Pembuatan Model Jejaring

Karena terdapat beberapa tempat/objek wisata yang tidak tepat bersebelahan dengan jalan raya yang dilalui oleh bus kota, maka dibuat beberapa asumsi yang digunakan dalam pembuatan model jejaring. Asumsi yang digunakan pada penelitian ini dapat dilihat pada tabel 4-4.

Tabel 4-4 Asumsi Dalam Pembuatan Model Jejaring

1	Asumsi: Masjid Al-Akbar Surabaya (Titik No. 3)
	Alasan: Karena tidak bersebelahan dengan Jalan Ahmad Yani, maka pemberhentian dilakukan di Carefour Ahmad Yani, lalu diteruskan berjalan ke barat sekitar 500 meter.
2	Asumsi: Taman Flora (Titik No. 6)
	Alasan: Karena taman flora berada di utara terminal bratang, jadi setelah sampai di terminal bratang, wisatawan harus berjalan ke utara sejauh 100 meter.

4.3. Model Matematika Permasalahan *Orienteering Problem*

Setelah pembuatan model jejaring di atas, maka selanjutnya dibuatlah model matematika yang dipakai untuk menyelesaikan permasalahan *Orienteering Problem* (OP) yang berisikan fungsi tujuan, variable keputusan, dan batasan.

Berikut adalah penjelasan mengenai apa saja variable keputusannya, dan bagaimana fungsi tujuannya serta apa saja batasannya.

4.3.1. Variabel Keputusan

Hasil tugas akhir ini berfokus pada bagaimana menentukan rute optimum dari satu titik ke titik lainnya. Maka dari itu, variable keputusan yang digunakan adalah kunjungan dari satu titik ke titik lainnya yang direpresentasikan dengan x_{ij} . Dimana x_{ij} akan bernilai 1 apabila kunjungan dimulai dari titik i dan berakhir di titik j dan apabila tidak, maka x_{ij} bernilai 0. Tabel 4-5 merupakan sebagian variable keputusan untuk permasalahan OP. Untuk keseluruhan variabel keputusan dapat dilihat pada halaman LAMPIRAN C.

Tabel 4-5 Variabel Keputusan *Orienteering Problem*

No	Variabel	Deskripsi
1	$x_{1.2}$	Kunjungan dari titik 1 ke titik 2
2	$x_{2.1}$	Kunjungan dari titik 2 ke titik 1
3	$x_{2.3}$	Kunjungan dari titik 2 ke titik 3
4	$x_{3.2}$	Kunjungan dari titik 3 ke titik 2
5	$x_{3.4}$	Kunjungan dari titik 3 ke titik 4
6	$x_{4.3}$	Kunjungan dari titik 4 ke titik 3
7	$x_{4.5}$	Kunjungan dari titik 4 ke titik 5
8	$x_{4.7}$	Kunjungan dari titik 4 ke titik 7
9	$x_{5.4}$	Kunjungan dari titik 5 ke titik 4
10	$x_{5.6}$	Kunjungan dari titik 5 ke titik 6
11	$x_{6.5}$	Kunjungan dari titik 6 ke titik 5
12	$x_{7.5}$	Kunjungan dari titik 7 ke titik 5
13	$x_{7.8}$	Kunjungan dari titik 7 ke titik 8
14	$x_{8.7}$	Kunjungan dari titik 8 ke titik 7
15	$x_{8.9}$	Kunjungan dari titik 8 ke titik 9

No	Variabel	Deskripsi
16	x9.8	Kunjungan dari titik 9 ke titik 8
17	x9.10	Kunjungan dari titik 9 ke titik 10
18	x10.9	Kunjungan dari titik 10 ke titik 9
19	x10.11	Kunjungan dari titik 10 ke titik 11
20	x10.19	Kunjungan dari titik 10 ke titik 19

4.3.2. Fungsi Tujuan

Fungsi tujuan dalam penelitian ini mengikuti fungsi tujuan OP pada umumnya yaitu mencari rute terpendek dengan jumlah skor maksimum.

Dalam studi kasus ini, titik awal keberangkatan adalah Terminal Bungurasih (Titik No. 1) dan titik akhir pemberhentian adalah Terminal Pelabuhan Tanjung Perak (Titik No. 41). Seperti yang telah dijabarkan sebelumnya, terdapat 41 titik pada model jejaring. Seluruh titik akan dimasukkan untuk mencari rute dengan skor tertinggi. Fungsi Tujuan dalam OP adalah sebagai berikut:

$$\text{Maximize } \sum_{i=1}^{|N|-1} \sum_{j=2}^{|N|} S_i X_{ij}$$

Sehingga penulisan fungsi tujuan pada studi kasus ini adalah:

$$\text{Maximize } \sum_{i=1}^{40} \sum_{j=2}^{41} S_i X_{ij}$$

Dimana:

S_i : Skor pada titik i

X_{ij} : Kunjungan dari titik i ke titik j

i : titik 1, 2, 3, ..., 40

j : titik 2, 3, 4, ..., 41

Dimana skor untuk tiap titik telah didefinisikan sebelumnya.

4.3.3. Perumusan Batasan

Batasan yang digunakan dalam pemodelan *Orienteering Problem* adalah sebagai berikut:

- **Batasan 1:** Rute harus dimulai dari titik ke 1 dan berakhir di titik ke 41.

Batasan ini digunakan untuk memastikan rute yang dipilih dimulai dari lokasi awal dan berhenti di lokasi akhir yang dituju. Sehingga variable keputusan yang terkait dengan kunjungan dari titik 1 dan kunjungan ke titik 41 harus bernilai 1. Batasan 1 dijabarkan sebagai berikut:

$$\sum_{j=2}^{|N|} X_{1j} = \sum_{i=1}^{|N|-1} X_{i|N|} = 1$$

Batasan titik awal adalah:

$$\sum_{j=2}^{41} X_{1j} = 1$$

Sedangkan batasan titik akhir adalah:

$$\sum_{j=1}^{40} X_{i41} = 1$$

- **Batasan 2:** Setiap jalur harus terhubung dan setiap titik hanya dapat dikunjungi maksimal satu kali.

Batasan ini digunakan untuk memastikan bahwa setiap jalur yang dilalui pada model jejaring terhubung dan menjamin setiap titik hanya dapat dikunjungi satu kali. Terhubungnya semua titik dalam suatu model jejaring ditunjukkan dari adanya percabangan dan pertemuan

antar titik. Bentuk model matematika dari batasan ini adalah sebagai berikut:

$$\sum_{i=1}^{|N|-1} X_{ik} = \sum_{j=2}^{|N|} X_{kj} \leq 1; \forall k = 2, \dots, (|N| - 1)$$

Untuk batasan pertemuan antar titik adalah seperti:

$$\sum_{i=1}^{40} X_{ik} \leq 1; \forall k = 2, \dots, 40$$

Untuk batasan percabangan antar titik adalah seperti:

$$\sum_{j=2}^{41} X_{kj} \leq 1; \forall k = 2, \dots, 40$$

- **Batasan 3:** Total waktu tempuh dalam satu kali perjalanan tidak boleh melebihi waktu yang telah ditentukan

Pada studi kasus ini, penentuan waktu maksimum dari titik ke 1 menuju ke titik 41 (tMax) adalah 60 menit. Maka dari itu batasan dapat dijabarkan sebagai berikut:

$$\sum_{i=1}^{40} \sum_{j=2}^{41} t_{ij} X_{ij} \leq 60 \text{ (Menit)}$$

- **Batasan 4 dan 5:** Tidak diperbolehkan terjadi *subtour*, yaitu kondisi dimana lintasan dimulai dan berakhir pada titik yang sama. Batasan ini secara tidak langsung telah tergambarkan pada batasan – batasan yang lain sebelumnya.

4.4. Penentuan Komponen Algoritma Genetika

Pada langkah ini, dilakukan penentuan beberapa komponen dari algoritma genetika sehingga bisa digunakan untuk menyelesaikan permasalahan *orienteering problem*. Komponen tersebut adalah:

4.4.1. Populasi

Populasi pada studi kasus ini adalah sekumpulan rekomendasi rute (individu) mulai dari titik awal sampai titik akhir yang memenuhi batasan. Dalam suatu populasi, panjang kromosom pada tiap individu bisa berbeda-beda. Namun setiap individu harus memiliki titik awal dan titik akhir yang sama..

4.4.2. Individu

Individu pada studi kasus ini adalah kumpulan dari titik-titik yang direkomendasikan untuk dilalui mulai dari titik awal sampai ke titik akhir. Penulisannya adalah sebagai berikut:

[titik awal, titik-titik mana saja yang dilalui, titik akhir] (waktu tempuh, skor)

Contoh:

[1, 9, 21, 38, 41] (55, 327)

4.4.3. Gen

Gen pada studi kasus ini adalah titik-titik yang dikunjungi dalam setiap rekomendasi rute. Jumlah gen dalam suatu individu adalah tidak tetap karena penetapan jumlah gen ditentukan secara acak. Maksimum jumlah gen dalam suatu individu adalah 42 sesuai dengan jumlah titik yang ada pada model jejaring.

4.5. Desain Algoritma Genetika

Pada tahap ini, dilakukan pembuatan desain algoritma genetika yang digunakan pada tugas akhir ini.

4.5.1. Desain Algoritma Genetika *Hyper-Heuristic*

Pada tahap ini, dijabarkan mengenai bagaimana alur kerja dari Algoritma Genetika *Hyper-Heuristic* untuk menyelesaikan permasalahan OP. Algoritma akan ditulis menggunakan bahasa pemrograman Java. Berikut adalah tahapannya:

4.5.1.1. Inisialisasi Parameter

Pada tahap ini dilakukan pengaturan parameter dasar yang digunakan pada algoritma genetika *hyper-heuristic*:

- a. Jumlah Titik = 42
- b. Titik Awal = 1
- c. Titik Akhir = 41
- d. Ukuran Populasi = 40
- e. Jumlah Generasi = 1000

4.5.1.2. Pembangkitan Populasi Awal

Pada tahap ini akan dibangkitkan individu sebagai solusi layak awal dengan titik-titik yang bervariasi sejumlah populasi yang diinisialisasi pada parameter. Secara detail, pembentukan solusi layak awal dibuat berdasarkan pembentukan populasi yang sudah dijelaskan oleh Tasgetiren [4].


```

Define tMax
Define loopMaksPPA
Set loopPPA = 0
Do {
    Produce a random number,  $R_N$ , between
    [1,N]
    Produce a list,  $R_L$ , between [1,N]
    randomly
    Generate a sub list,  $R_S$ , by taking
    the first  $R_N$  part of the random list
     $R_L$ 
    Compute the total traveling time,  $t_{RS}$ ,
    of the sub list  $R_S$ 
    If  $t_{RS} \leq tMax$ , then loopPPA=loopPPA+1
} while (loopPPA <= loopMaksPPA)

```

**Kode 4-1 Pseudo Code Pembangkitan Populasi Awal Algoritma
Genetika *Hyper-Heuristic***

Proses pencarian solusi dimulai dengan menentukan $tMax$ sebagai batasan maksimal total waktu tempuh setiap individu solusi lalu jumlah $loopMaksPPA$ sebagai maksimal iterasi pembangkitan populasi awal. Setiap individu solusi layak awal harus memiliki titik awal dan titik akhir yang sama dan waktu tempuhnya tidak boleh melebihi nilai $tMax$.

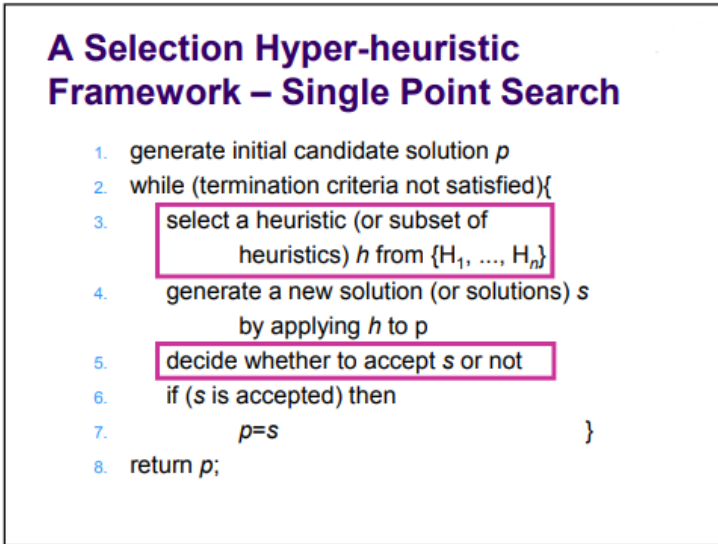
4.5.1.3. Evaluasi Nilai *Fitness*

Pada tahap ini, dilakukan evaluasi nilai *fitness*. Nilai ini didapatkan dengan mengkalkulasikan skor di tiap titik (tempat wisata) yang ada dalam suatu individu/rute. Skor tiap tempat didapatkan dari hasil pencarian dengan menggunakan mesin pencari *Google*.

4.5.1.4. Memulai Proses *Hyper-Heuristic*

Pada penelitian ini, digunakan salah satu framework *Hyper-Heuristic*, yaitu *Single Point Search*[19]. Proses ini dimulai dengan mencari individu terbaik dari pembangkitan populasi

awal. Individu terbaik ini yang akan diproses menggunakan operator *hyper-heuristic*.



Kode 4-2 Pseudo Code Hyper-Heuristic Single Point Search

4.5.1.5. Penerapan Operator *Hyper-Heuristic*

Pada bagian ini, individu terbaik hasil pembangkitan populasi awal dipakai sebagai input untuk melakukan proses *hyper-heuristic*. Pada penelitian ini, digunakan 4 macam operator *hyper-heuristic*. Berikut adalah penjelasan mengenai operator *hyper-heuristic* yang dipakai pada penelitian ini:

a. Operator Kawin Silang Injection

Operator ini bekerja untuk menjalankan proses kawin silang dengan cara membuat *insertion point* pada induk pertama dimulai dari gen terdepan, selanjutnya untuk induk kedua dilakukan pengambilan beberapa gen dari belakang kecuali gen terakhir, lalu gen-gen tersebut dimasukkan ke dalam *insertion point* induk pertama. Bila

ada gen yang terduplikasi atau tidak sesuai rute maka gen tersebut akan dihapus, selanjutnya hasil dari injeksi tersebut terbentuklah keturunan baru [4].

b. Operator Mutasi *Exchange*

Operator ini bekerja dengan melakukan pertukaran posisi dari gen yang ada di dalam individu. Apabila setelah dilakukan *exchange* membuat nilai *fitness* individu menjadi lebih baik, maka hasil *exchange* akan dipertahankan, namun apabila semakin buruk maka hasil *exchange* akan dikembalikan posisinya. Gen yang bisa ditukar adalah gen selain titik awal dan titik akhir [20].

c. Operator Mutasi *Add*

Operator ini bekerja dengan memasukkan gen secara acak pada individu yang ada. Apabila hasil penambahan secara acak membuat nilai *fitness* individu menjadi lebih baik, maka perubahan akan disimpan, namun apabila menjadi lebih buruk, maka penambahan gen akan dibatalkan [4].

d. Operator Mutasi *Omit*

Operator ini berkerja dengan menghapus gen secara acak pada individu yang ada. Apabila hasil penghapusan secara acak membuat nilai waktu tempuh individu menjadi lebih kecil, maka perubahan akan disimpan, namun apabila menjadi lebih besar, maka penambahan gen akan dibatalkan. Gen yang memungkinkan untuk dihapus adalah gen selain titik awal dan titik akhir [4].

4.5.1.6. Pemberhentian Algoritma Genetika *Hyper-Heuristic*

Pada penelitian ini, algoritma genetika hyper-heuristic akan berhenti ketika telah mencapai nilai maksimal generasi/iterasi yang telah ditentukan. Setelah sampai pada iterasi terakhir, individu hasil *hyper-heuristic* pada putaran tersebut akan dipilih menjadi solusi optimal.

4.5.2. Desain Algoritma Genetika

Pada tahap ini, dijabarkan mengenai bagaimana alur kerja dari Algoritma Genetika untuk menyelesaikan permasalahan OP.

4.5.2.1. Inisialisasi Parameter

Pada tahap ini dilakukan pengaturan parameter dasar yang digunakan pada algoritma genetika:

- a. Jumlah Titik = 42
- b. Titik Awal = 1
- c. Titik Akhir = 41
- d. Ukuran Populasi = 40
- e. Jumlah Seleksi = 10
- f. Jumlah Generasi = 1000
- g. Probabilitas Kawin Silang = 0.9
- h. Probabilitas Mutasi = 0.1

4.5.2.2. Pembangkitan Populasi Awal

Pada tahap ini akan dibangkitkan individu sebagai solusi layak awal dengan titik-titik yang bervariasi sejumlah populasi yang diinisialisasi pada parameter. Secara detail, pembentukan solusi layak awal dibuat berdasarkan pembentukan populasi yang sudah dijelaskan oleh Tasgetiren [4].

```

Define tMax
Define maxloop
Set loop = 0
Do {
    Produce a random number,  $R_N$ , between
    [1,N]
    Produce a list,  $R_L$ , between [1,N]
    randomly
    Generate a sub list,  $R_S$ , by taking
    the first  $R_N$  part of the random list
     $R_L$ 
    Compute the total traveling time,
     $t_{RS}$ , of the sub list  $R_S$ 
    If  $t_{RS} \leq tMax$ , then loop=loop+1
} while (loop <= maxloop)

```

Kode 4-3 Pseudo Code Pembangkitan Populasi Awal Algoritma Genetika

Proses pencarian solusi dimulai dengan menentukan $tMax$ sebagai batasan maksimal total waktu tempuh setiap individu solusi lalu jumlah $maxloop$ sebagai maksimal iterasi pembangkitan populasi awal. Setiap individu solusi layak awal harus memiliki titik awal dan titik akhir yang sama dan waktu tempuhnya tidak boleh melebihi nilai $tMax$.

4.5.2.3. Evaluasi Nilai *Fitness*

Pada tahap ini, dilakukan evaluasi nilai *fitness*. Nilai ini didapatkan dengan mengkalkulasikan skor di tiap titik (tempat wisata) yang ada dalam suatu individu/rute. Skor tiap tempat didapatkan dari hasil pencarian dengan menggunakan mesin pencari *Google*.

4.5.2.4. Melakukan Seleksi Individu

Setelah setiap individu pada populasi diketahui nilai *fitness* dan waktu tempuh, maka tahap selanjutnya adalah melakukan seleksi. Seleksi dilakukan dengan metode roda *roulette* [15]

sehingga didapatkan 10 individu yang akan menjadi induk dari proses kawin silang.

4.5.2.5. Memulai Proses Kawin Silang

Pada tahap kawin silang ini, digunakan salah satu jenis kawin silang yang sesuai dengan permasalahan *orienteering problem*, yaitu kawin silang secara injeksi. Proses kawin silang dilakukan dengan cara membuat *insertion point* pada induk pertama dimulai dari gen terdepan, selanjutnya untuk induk kedua dilakukan pengambilan beberapa gen dari belakang kecuali gen terakhir, lalu gen-gen tersebut dimasukkan ke dalam *insertion point* induk pertama. Bila ada gen yang terduplikasi atau tidak sesuai rute maka gen tersebut akan dihapus, selanjutnya hasil dari injeksi tersebut terbentuklah keturunan baru [4].

4.5.2.6. Memulai Proses Mutasi

Pada tahap mutasi ini, digunakan salah satu jenis mutasi yang sesuai dengan permasalahan *orienteering problem*, yaitu mutasi *Add*. Mutasi ini bekerja dengan memasukkan gen secara acak pada individu yang ada. Apabila hasil penambahan secara acak membuat nilai *fitness* individu menjadi lebih baik, maka perubahan akan disimpan, namun apabila menjadi lebih buruk, maka penambahan gen akan dibatalkan [4].

4.5.2.7. Pembentukan Populasi Baru

Setelah proses mutasi selesai, individu hasil mutasi yang memenuhi kriteria ditambah dengan sisa populasi sebelumnya akan menjadi populasi baru. Populasi baru ini akan diproses kembali dengan seleksi, kawin silang dan mutasi sampai iterasi terakhir.

4.5.2.8. Pemberhentian Algoritma Genetika

Algoritma genetika akan berhenti ketika ia telah mencapai nilai maksimal generasi/iterasi yang telah ditentukan. Setelah sampai pada iterasi terakhir, individu dengan nilai *fitness* yang

paling besar dari populasi pada iterasi terakhir tersebut akan dipilih menjadi solusi yang optimal.

4.5.3. Desain Algoritma Genetika Modifikasi

Pada tahap ini, dijabarkan mengenai bagaimana alur kerja dari Algoritma Genetika Modifikasi untuk menyelesaikan permasalahan OP.

4.5.3.1. Inisialisasi Parameter

Pada tahap ini dilakukan pengaturan parameter dasar yang digunakan pada algoritma genetika modifikasi:

- a. Jumlah Titik = 42
- b. Titik Awal = 1
- c. Titik Akhir = 41
- d. Ukuran Populasi = 40
- e. Jumlah Seleksi = 10
- f. Jumlah Generasi = 1000
- g. Probabilitas Kawin Silang = 0.9

4.5.3.2. Pembangkitan Populasi Awal

Pada tahap ini akan dibangkitkan individu sebagai solusi layak awal dengan titik-titik yang bervariasi sejumlah populasi yang diinisialisasi pada parameter. Secara detail, pembentukan solusi layak awal dibuat berdasarkan pembentukan populasi yang sudah dijelaskan oleh Tasgetiren [4].

```

Define tMax
Define maxloop
Set loop = 0
Do {
    Produce a random number,  $R_N$ , between
    [1,N]
    Produce a list,  $R_L$ , between [1,N]
    randomly
    Generate a sub list,  $R_S$ , by taking
    the first  $R_N$  part of the random list
     $R_L$ 
    Compute the total traveling time,
     $t_{RS}$ , of the sub list  $R_S$ 
    If  $t_{RS} \leq tMax$ , then loop=loop+1
} while (loop <= maxloop)

```

**Kode 4-4 Pseudo Code Pembangkitan Populasi Awal Algoritma
Genetika Modifikasi**

Proses pencarian solusi dimulai dengan menentukan $tMax$ sebagai batasan maksimal total waktu tempuh setiap individu solusi lalu jumlah $loopMaksPPA$ sebagai maksimal iterasi pembangkitan populasi awal. Setiap individu solusi layak awal harus memiliki titik awal dan titik akhir yang sama dan waktu tempuhnya tidak boleh melebihi nilai $tMax$.

4.5.3.3. Evaluasi Nilai *Fitness*

Pada tahap ini, dilakukan evaluasi nilai *fitness*. Nilai ini didapatkan dengan mengkalkulasikan skor di tiap titik (tempat wisata) yang ada dalam suatu individu/rute. Skor tiap tempat didapatkan dari hasil pencarian dengan menggunakan mesin pencari *Google*.

4.5.3.4. Melakukan Seleksi Individu

Setelah setiap individu pada populasi diketahui nilai *fitness* dan waktu tempuh, maka tahap selanjutnya adalah melakukan seleksi. Seleksi dilakukan dengan metode roda *roulette* [15]

sehingga didapatkan 10 individu yang akan menjadi induk dari proses kawin silang.

4.5.3.5. Memulai Proses Kawin Silang

Pada tahap kawin silang ini, digunakan salah satu jenis kawin silang yang sesuai dengan permasalahan *orienteeing problem*, yaitu kawin silang secara injeksi. Proses kawin silang dilakukan dengan cara membuat *insertion point* pada induk pertama dimulai dari gen terdepan, selanjutnya untuk induk kedua dilakukan pengambilan beberapa gen dari belakang kecuali gen terakhir, lalu gen-gen tersebut dimasukkan ke dalam *insertion point* induk pertama. Bila ada gen yang terduplikasi atau tidak sesuai rute maka gen tersebut akan dihapus, selanjutnya hasil dari injeksi tersebut terbentuklah keturunan baru [4].

4.5.3.6. Memulai Proses Mutasi

Pada tahap ini dilakukan modifikasi untuk memilih operator mutasi yang digunakan. Ada 3 jenis operator mutasi yang digunakan secara acak pada setiap iterasi. Penjabarannya operator tersebut ada di bawah ini:

a. Operator Mutasi *Add*

Operator ini bekerja dengan memasukkan gen secara acak pada individu yang ada. Apabila hasil penambahan secara acak membuat nilai *fitness* individu menjadi lebih baik, maka perubahan akan disimpan, namun apabila menjadi lebih buruk, maka penambahan gen akan dibatalkan [4].

b. Operator Mutasi *Omit*

Operator ini berkerja dengan menghapus gen secara acak pada individu yang ada. Apabila hasil penghapusan secara acak membuat nilai waktu tempuh individu menjadi lebih kecil, maka perubahan akan disimpan, namun apabila menjadi lebih besar, maka penambahan gen akan

dibatalkan. Gen yang memungkinkan untuk dihapus adalah gen selain titik awal dan titik akhir [4].

c. Operator Mutasi *Exchange*

Operator ini bekerja dengan melakukan pertukaran posisi dari gen yang ada di dalam individu. Apabila setelah dilakukan *exchange* membuat nilai *fitness* individu menjadi lebih baik, maka hasil *exchange* akan dipertahankan, namun apabila semakin buruk maka hasil *exchange* akan dikembalikan posisinya. Gen yang bisa ditukar adalah gen selain titik awal dan titik akhir [20].

4.5.3.7. Pembentukan Populasi Baru

Setelah proses mutasi selesai, individu hasil mutasi yang memenuhi kriteria ditambah dengan sisa populasi sebelumnya akan menjadi populasi baru. Populasi baru ini akan diproses kembali dengan seleksi, kawin silang dan mutasi sampai iterasi terakhir.

4.5.3.8. Pemberhentian Algoritma Genetika Modifikasi

Algoritma genetika modifikasi akan berhenti ketika ia telah mencapai nilai maksimal generasi/iterasi yang telah ditentukan. Setelah sampai pada iterasi terakhir, individu dengan nilai *fitness* yang paling besar dari populasi pada iterasi terakhir tersebut akan dipilih menjadi solusi yang optimal.

4.6. Desain Aplikasi

Pada tahap ini, dilakukan pembuatan desain aplikasi yang dihasilkan pada tugas akhir ini.

4.6.1. Tampilan Aplikasi

Gambar 4-2 merupakan tampilan dari aplikasi yang dihasilkan dalam tugas akhir ini.

APLIKASI PENCARIAN RUTE WISATA DI KOTA SURABAYA

Pilih Titik Awal Pemberangkatan Anda
Terminal Bungurasih

Pilih Titik Akhir Pemberhentian Anda *Titik Awal dan Akhir tidak boleh sama
Pelabuhan Tanjung Perak

Isikan waktu maksimal perjalanan Anda
 Menit

MULAI PENCARIAN!

Rekomendasi rute wisata Anda adalah :

Gambar 4-2 Tampilan Aplikasi

Aplikasi ini dibuat menggunakan bahasa pemrograman *Java* menggunakan aplikasi NetBeans IDE 7.4.

4.6.2. Penentuan Komponen Aplikasi

Pada tahap ini ditentukan komponen yang dipakai pada aplikasi yang dihasilkan dari tugas akhir ini.

4.6.2.1. Komponen Masukan

Masukan yang dibutuhkan pada aplikasi ini adalah:

a. Titik Awal

Titik awal adalah titik yang dipilih sebagai titik keberangkatan wisatawan.

b. Titik Akhir

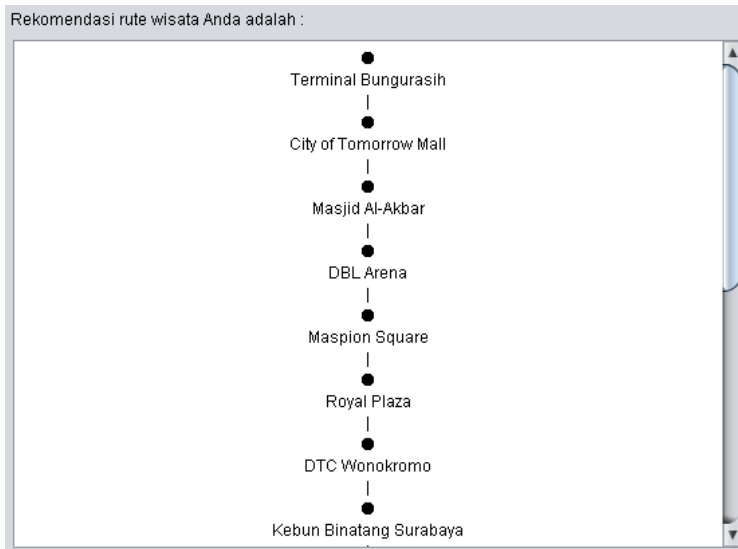
Titik akhir adalah titik yang dipilih sebagai titik akhir perjalanan wisata.

c. Waktu Tempuh

Waktu tempuh adalah durasi perjalanan maksimal yang diinginkan wisatawan.

4.6.2.2. Komponen Keluaran

Hasil keluaran dari aplikasi ini adalah rekomendasi rute perjalanan wisata dari satu tempat ke tempat lainnya. Gambar 4-3 menunjukkan potongan hasil rekomendasi rute yang ditemukan.



Gambar 4-3 Hasil Pencarian Rute Wisata

Hasil pencarian rute wisata ditampilkan dengan titik hitam disertai nama tempat wisata tersebut.

BAB V IMPLEMENTASI

Pada bab kelima ini dijelaskan mengenai proses implementasi algoritma untuk mencari solusi yang optimal dari studi kasus tugas akhir. Bahasa pemrograman yang dipakai adalah Java dengan tools yang digunakan adalah NetBeans 7.4.

5.1. Pembuatan Matriks Waktu Tempuh Keseluruhan

Pada matriks antar titik, masih banyak sel yang terisi dengan angka nol karena banyak node yang tidak terhubung secara langsung. Maka dari itu perlu dibuat matriks yang berisi waktu tempuh keseluruhan untuk tiap titik. Untuk mengisi nilai nol pada matriks antar titik, maka perlu diterapkan algoritma Dijkstra. Nilai nol akan diisi dengan waktu tempuh tercepat antar node yang dihasilkan oleh algoritma tersebut. Matriks antar titik yang telah dibuat sebelumnya dengan format. csv akan menjadi input untuk algoritma Dijkstra dan algoritma dijalankan pada NetBeans IDE 7.4.

```
22     static int jumlahNode = 42; //inisialisasi jumlah titik
23     static int src = 0;
24
25     //fungsi untuk menemukan node dengan nilai waktu tempuh minimum
26     //dari set node yang belum memiliki nilai terpendek
27     int minDistance(int dist[], Boolean sptSet[]) {
28         // Initialize min value
29         int min = Integer.MAX_VALUE, min_index = 0;
30
31         for (int v = 0; v < jumlahNode; v++) {
32             if (sptSet[v] == false && dist[v] <= min) {
33                 min = dist[v];
34                 min_index = v;
35             }
36         }
37         return min_index;
38     }
```

Kode 5-1 Algoritma Dijkstra (1)

Kode 5-1 ini memiliki fungsi untuk menemukan titik dengan nilai waktu tempuh minimum pada titik yang belum memiliki waktu tempuh terpendek.

```

40 // Method untuk mencetak array jarak yang telah di susun
41 void printSolution(int dist[], int n) {
42     for (int i = 0; i < jumlahNode; i++) {
43         System.out.print(dist[i] + ",");
44     }
45 }

```

Kode 5-2 Algoritma Dijkstra (2)

Kode 5-2 ini berfungsi untuk mencetak semua solusi yang di hasilkan oleh algoritma Dijkstra dengan tanda koma sebagai pemisah.

```

47 void dijkstra(int graph[][], int src) {
48     int dist[] = new int[jumlahNode]; // array yang berfungsi untuk menyimpan
49     // jalur terpendek dari source ke i
50
51     // sptSet[i] akan bernilai true jika jarak terpendek dari node src
52     // ke node i sudah final
53     Boolean sptSet[] = new Boolean[jumlahNode];
54
55     // Initialize all distances as INFINITE and sptSet[] as false
56     for (int i = 0; i < jumlahNode; i++) {
57         dist[i] = Integer.MAX_VALUE;
58         sptSet[i] = false;
59     }
60
61     // jarak dari node sumber ke node yang sama selalu bernilai 0
62     dist[src] = 0;
63
64     // Temukan jalur terpendek dari node sumber ke semua node
65     for (int count = 0; count < jumlahNode - 1; count++) {
66         // Pilih jalur terpendek ke semua node
67         // Pada iterasi pertama, u selalu sama dengan source
68         int u = minDistance(dist, sptSet);
69
70         // Tandai node yang telah diproses
71         sptSet[u] = true;
72
73         // update nilai dist
74         for (int v = 0; v < jumlahNode; v++)

```

Kode 5-3 Algoritma Dijkstra (3)

Kode 5-3 merupakan inti dari algoritma Dijkstra. Pada kode ini akan dilakukan pembuatan array satu dimensi untuk menyimpan nilai waktu tempuh. Lalu dilakukan proses algoritma Dijkstra untuk mencari jalur terpendek antar titik.

```

92 public static void main(String[] args) throws FileNotFoundException,
93
94     String thisLine;
95     BufferedReader nodeMatrix = new BufferedReader
96     (new FileReader("data/MatriksJarakTitikTerhubung.csv"));
97
98     ArrayList<String[]> lines = new ArrayList<>();
99     while ((thisLine = nodeMatrix.readLine()) != null) {
100         lines.add(thisLine.split(", "));
101     }
102
103     //Convert our list to a String array
104     String[][] array = new String[lines.size()][0];
105     lines.toArray(array);
106
107     //Convert String array to Integer array
108     for (int i = 0; i < array.length; i++) {
109         for (int j = 0; j < array.length; j++) {
110             arrNode[i][j] = Integer.parseInt(array[i][j]);
111         }
112     }
113
114     TugasAkhirImad t = new TugasAkhirImad();
115     for (int i = 0; i < jumlahNode; i++) {
116         t.dijkstra(arrNode, i);
117         System.out.println();

```

Kode 5-4 Algoritma Dijkstra (4)

Kode 5-4 merupakan kode untuk membaca file yang akan diolah. Pada tugas akhir ini, file yang digunakan adalah MatriksJarakTerhubung.csv . Setiap baris dari file akan dibaca kemudian dikonversi menjadi string array dan dilanjutkan menjadi integer array.

5.2. Penerapan Algoritma Genetika *Hyper-Heuristic*

Pada bagian ini menjelaskan bagaimana Algoritma Genetika *Hyper-Heuristic* (AGHH) yang telah dirancang pada bab sebelumnya diterapkan. Tahapan-tahapan utama dari AGHH akan disertakan dengan potongan kode dibuat dengan Bahasa pemrograman Java menggunakan aplikasi NetBeans IDE 7.4.

5.2.1. Inisialisasi Parameter

Pada bagian ini diinisialisasikan parameter apa saja yang digunakan untuk AGHH. Parameter yang diinisialisasikan adalah titik awal, titik akhir, jumlah titik, tMax, jumlah populasi, dan jumlah generasi.

```
26         static int titikAwal = 1; //Menetar
27         static int titikAkhir = 41; //Menet
28         static int jumlahTitik = 42; //Juml
29         static int tMax = 60; //Menetapkan
30         static int jumlahPopulasi = 40; //E
31         static int jumlahSeleksi = 1; //Mer
32         static int jumlahGenerasi = 1000; /
```

Kode 5-5 Inisialisasi Parameter AGHH

Kode 5.4 menunjukkan bila pada tugas akhir ini jumlah titik yang digunakan adalah 42, tMax sebesar 60 menit, jumlah populasi sebesar 40, dan jumlah generasi sebesar 1000.

5.2.2. Pembangkitan Populasi Awal

Pada bagian dilakukan proses pembangkitan populasi awal. Populasi awal yang dibangkitkan harus memenuhi semua batasan seperti yang di sebutkan pada bab sebelumnya.


```

132     int populasi = 0; //Menetapkan inisiasi jumlah individu
133     int loopPPA = 0; //Inisiasi looping pencarian populasi
134     int loopMaksPPA = 100000; //Maksimal looping pencarian
135
136     do {
137
138         //1.1. Menghasilkan nilai random, Rn , diantara inde
139         //     digunakan untuk menentukan panjang solusi yan
140         int RN = new Random().nextInt(jumlahTitik - 2) + 1;
141
142         //1.2. Menghasilkan list Rl , diantara [1,N] secara
143         ArrayList<Integer> listRL = new ArrayList();
144         //Array yang digunakan untuk mengisi titik random d:
145         ArrayList<Integer> listRL_rand = new ArrayList();
146
147         //1.3. Menggenerate isi listRL_rand diantara titik
148         for (int i = 1; i < jumlahTitik + 1; i++) {
149             if (i == titikAwal || i == titikAkhir) {
150                 continue; //Memastikan tidak ada titik awal
151             } //yang muncul di tengah individu
152             listRL_rand.add(i);
153         }
154
155         Collections.shuffle(listRL_rand); //Mencacak titik
156
157         //1.4. Mengisi listRL
158         listRL.add(titikAwal); //Merepresentasikan node awal
159         for (int i = 0; i < listRL_rand.size(); i++) {
160             listRL.add(listRL_rand.get(i)); //Representasi
161         } //ke N-1 yang te
162         listRL.add(titikAkhir); //Merepresentasikan node aki

```

Kode 5-6 Pembangkitan Solusi Awal

Kode 5-6 menunjukkan solusi awal dibuat dan disimpan pada ArrayList bernama listRI, list ini memiliki panjang maksimal 42 sesuai jumlah titik. Setiap solusi memiliki titik awal dan akhir yang sama, yaitu 1 dan 41.

```

166 ArrayList<Integer> subListRS = new ArrayList();
167 subListRS.add(listRL.get(0));
168 for (int i = 1; i < (RN + 1); i++) {
169     subListRS.add(listRL.get(i));
170 }
171 subListRS.add(listRL.get(listRL.size() - 1));
172
173 //1.6. Hanya memasukkan subList Rs (individu) yang unique
174 //dan tidak melebihi batasan waktu tMax ke dalam populasi.
175 for (int i = 0; i < subListRS.size(); i++) {
176     if (hitungWaktuTempuh(subListRS) <= tMax
177         && !ArrayPopulasi.contains(subListRS)) {
178         ArrayPopulasi.add(subListRS);
179         populasi++;
180     }
181 }
182
183 loopPPA++;
184
185 //Menghentikan pembangkitan individu.
186 if (loopPPA == loopMaksPPA) {
187     //Bila tidak ada individu yang memenuhi, maka akan keluar
188     //pesan berikut hasil running.
189     System.out.println("Tidak dapat menemukan solusi yang "
190         + "layak untuk tMax sebesar " + tMax);
191     System.exit(0);
192 }

```

Kode 5-7 Pembuatan Sublist

Kode 5-7 menunjukkan pembuatan ArrayList baru yang berguna untuk membentuk sublist, dengan nama subListRS. Setiap individu yang dihasilkan melalui proses ini harus memenuhi ke 5 batasan yang sudah ditetapkan pada bab sebelumnya. Semua hasil pembangkitan akan disimpan pada ArrayPopulasi. Pembangkitan akan terus dilakukan hingga batas maksimal iterasi, yaitu 10000. Apabila tidak bisa ditemukan individu yang memenuhi syarat sampai iterasi yang terakhir maka program secara otomatis akan berhenti dan memberikan notifikasi.

```

198 //1.7. Memasukkan waktu tempuh dan nilai fitness untuk setiap
199 // individu dari populasi awal yang dibangkitkan.
200 for (int i = 0; i < ArrayPopulasi.size(); i++) {
201     ArrayPopulasi_WaktuTempuh.add(
202         hitungWaktuTempuh(ArrayPopulasi.get(i)));
203     ArrayPopulasi_Fitness.add(
204         hitungFitness(ArrayPopulasi.get(i)));
205 }

```

Kode 5-8 Memberi Nilai *Fitness* dan Waktu Tempuh Pada Populasi

Kode 5-8 adalah untuk memberikan nilai fitness dan waktu tempuh pada semua individu yang dibangkitkan.

```

219 //1.9. Mencetak hasil pembangkitan populasi awal
220 pw.println("+++> MULAI ALGORITMA GENETIKA HYPER-HEURISTIC
221 pw.println(" ");
222 pw.println("Hasil Pembangkitan Populasi Awal :");
223 for (int i = 0; i < ArrayPopulasi.size(); i++) {
224     pw.println(ArrayPopulasi.get(i) + " ("
225         + ArrayPopulasi_WaktuTempuh.get(i) + ", "
226         + ArrayPopulasi_Fitness.get(i) + ")");
227 }

```

Kode 5-9 Mencetak Hasil Pembangkitan Populasi Awal

Kode 5-9 digunakan untuk mencetak semua individu yang telah dibangkitkan. Jumlah individu yang dibangkitkan adalah 40.

5.2.3. Evaluasi *Fitness*

Setiap individu yang dibangkitkan atau dihasilkan melalui proses kawin silang dan mutasi akan dievaluasi nilai *fitness*-nya. Dari evaluasi *fitness* ini akan ditemukan individu yang terbaik yang nanti akan dijadikan input untuk proses selanjutnya.

```

712 //Fungsi untuk menghitung fitness setiap individu.
713 public static int hitungFitness(ArrayList<Integer> varList) {
714     int fitness = 0; //Inisiasi nilai awal fit adalah 0
715     for (int i = 0; i < varList.size(); i++) {
716         fitness = fitness + nilaiSkor[varList.get(i) - 1];
717     } //Loop di atas utk menghitung skor tiap titik
718     return fitness;
719 }

```

Kode 5-10 Cara Menghitung Nilai *Fitness*

Kode 5-10 menunjukkan *method* untuk menghitung nilai *fitness* pada individu sehingga masing-masing individu bisa dibandingkan mana yang lebih baik untuk menjadi solusi.

5.2.4. Melakukan Seleksi

Seleksi dilakukan untuk mencari individu terbaik yang akan digunakan sebagai input untuk proses *hyper-heuristic*.

```

209 //1.8. Mencari individu terbaik dari proses pembangkitan
210 // populasi awal.
211 int individuSolusiAwal = 0;
212 int optimalFitnessAwal = ArrayPopulasi_Fitness.get(0);
213 for (int i = 1; i < ArrayPopulasi.size(); i++) {
214     if (ArrayPopulasi_Fitness.get(i) >= optimalFitnessAwal) {
215         individuSolusiAwal = i;
216         optimalFitnessAwal = ArrayPopulasi_Fitness.get(i);
217     }
218 }
219 ArraySeleksi.add(ArrayPopulasi.get(individuSolusiAwal));

```

Kode 5-11 Pencarian Individu Terbaik

Kode 5-11 menunjukkan proses untuk mendapatkan individu terbaik dari array populasi. Individu terbaik tersebut selanjutnya dimasukkan pada array seleksi.

```

239 | pw.println(" ");
240 | pw.println("Individu Terbaik Saat Pembangkitan Populasi Awal");
241 | pw.println("Individu ini akan menjadi input awal untuk operator"
242 |         + " low-level heuristic");
243 | pw.println("Rule: "
244 |         + ArraySeleksi.get(0) + " | Waktu Tempuh: "
245 |         + ArraySeleksi_WaktuTempuh.get(0) + " | Fitness: "
246 |         + ArraySeleksi_Fitness.get(0));
247 | pw.println(" ");

```

Kode 5-12 Mencetak Individu Terbaik

Kode 5-12 digunakan untuk mencetak individu terbaik pada pembangkitan populasi awal. Individu terbaik ini yang akan menjadi input untuk proses *hyper-heuristic*.

5.2.5. Penerapan *Hyper-Heuristic*

Hyper-heuristic akan secara acak memilih operator yang dipakai pada tiap iterasi.

```

251 | //memulai iterasi hyper-heuristic
252 | for (int i = 0; i < ArraySeleksi.size(); i++) {
253 |     int iterasiHH = 1;
254 |     do {
255 |
256 |         int randomOperator = new Random().nextInt(4) + 1;
257 |
258 |         //pw.println(" ");
259 |         //pw.println("Hasil Hyper-Heuristic Generasi ke : "
260 |         //+ ArraySeleksi.get(0));
261 |
262 |         switch (randomOperator) {

```

Kode 5-13 Memulai Iterasi *Hyper-Heuristic*

Kode 5-13 adalah proses untuk memulai iterasi *hyper-heuristic*. Di setiap iterasi, akan digunakan 1 dari 4 operator *hyper-heuristic* yang akan dipilih secara acak. Proses ini akan terus berlanjut sampai iterasi terakhir dicapai.

5.2.6. Pemakaian Operator *Hyper-Heuristic*

Pada bagian ini, operator yang sudah ditentukan secara random akan berjalan untuk mengolah individu terbaik

menjadi individu keturunan, apabila hasil keturunan lebih baik, maka dia akan menggantikan posisi individu terbaik sebelumnya, namun apabila tidak, maka individu sebelumnya akan dipertahankan.

```

265         case 1:
266             //3. MELAKUKAN KAWIN SILANG DENGAN INJECTION CROSS-OVER
267             // membuat array untuk menyimpan keturunan
268             List<Integer> keturunan = new ArrayList<Integer>();
269
270             if (ArraySeleksi.get(0).size()
271                 <= ArraySeleksi.get(0).size()) {
272
273                 //Membuat insertion point untuk Induk 1.
274                 int insertionPoint = new Random().nextInt(
275                     ArraySeleksi.get(0).size() - 2) + 1;
276                 //Mengambil gen sejumlah insertion point
277                 //pada Induk 1 dimulai dari indeks 0.
278                 List<Integer> head = ArraySeleksi.get(0).subList(0,
279                     insertionPoint);
280                 //Merupakan list yang nantinya akan digunakan jika
281                 //proses perkawinan silang tidak berhasil.
282                 List<Integer> cdg = ArraySeleksi.get(0).subList(
283                     insertionPoint,
284                     ArraySeleksi.get(0).size() - 1);
285                 //Mengambil gen sejumlah insertion point pada
286                 //Induk 2 dimulai dari indeks sebelum node akhir.
287                 List<Integer> tail = ArraySeleksi.get(0).subList(
288                     ArraySeleksi.get(0).size() - insertionPoint
289                     - 1, ArraySeleksi.get(0).size() - 1);
290
291                 //Fungsi untuk menghasilkan keturunan. Jika proses
292                 //perkawinan silang gagal, maka dihasilkan keturunan
293                 //yang sama dengan induk 1.
294                 if (head.size() + tail.size() + 1
295                     >= ArraySeleksi.get(0).size()) {
296                     //Memasukkan semua gen dari head ke keturunan.
297                     keturunan.addAll(head);

```

Kode 5-14 Operator *Hyper-Heuristic Injection Cross-Over*

Kode 5-14 adalah proses untuk melakukan injection cross-over. Pada induk 1 akan dibuat *insertion point* dari titik awal hingga titik *insertion point*-nya, selanjutnya pada individu 2 diambil beberapa gen dari sebelum titik terakhir. Kedua potongan dari kedua induk tersebut akan digabung untuk menghasilkan individu baru.

```

335 List<Integer> tail = ArraySeleksi.get(0).subList(
336     ArraySeleksi.get(0).size() - insertionPoint
337     - 1, ArraySeleksi.get(0).size() - 1);
338
339 if (head.size() + tail.size() + 1
340     >= ArraySeleksi.get(0).size()) {
341     keturunan.addAll(head);
342     int index = 0;
343     while (keturunan.size()
344         < ArraySeleksi.get(0).size() - 1) {
345         if (index < tail.size()) {
346             if (!keturunan.contains(
347                 tail.get(index))) {
348                 keturunan.add(tail.get(index));
349             }
350             index++;
351         } else {
352             keturunan.addAll(cdg);
353         }
354     }
355     keturunan.add(titikAkkhir);
356 } else {
357     keturunan.addAll(ArraySeleksi.get(0));
358 }
359 }
360
361 //3.3. Memasukkan keturunan hasil crossover ke dalam array
362 //yang unik dan memenuhi batasan tMax
363 if (!ArraySeleksi.contains(keturunan) && hitungWaktuTempuh(
364     (ArrayList<Integer>) keturunan) <= tMax) {
365     ArraySeleksi.add((ArrayList<Integer>) keturunan);

```

Kode 5-15 Operator *Hyper-Heuristic Injection Cross-Over* (2)

Kode 5-15 adalah proses untuk memasukkan individu keturunan ke array seleksi apabila individu tersebut nilai waktu tempuhnya memenuhi batasan tMax.

```

406 case 2:
407 //4.1. Menjalankan Operator Exchange.
408
409 //memilih 2 titik pada individu yang akan di mutasi dengan exchange
410 int tukar1 = new Random().nextInt(ArraySeleksi.get(0).size() - 2) + 1;
411 int tukar2 = new Random().nextInt(ArraySeleksi.get(0).size() - 2) + 1;
412
413 //pw.println("Low-Level Heuristic = Exchange");
414
415 if (ArraySeleksi.get(0).size() > 3) {
416
417     //Menyimpan sementara nilai dari gen yang di tukar
418     int sementara = ArraySeleksi.get(0).get(tukar1);
419     int sementara2 = ArraySeleksi.get(0).get(tukar2);
420
421     Collections.swap(ArraySeleksi.get(0), tukar1, tukar2);
422
423     //Menyimpan waktu tempuh & fitness hasil proses exchange
424     int currentWaktuTempuhS
425         = hitungWaktuTempuh(ArraySeleksi.get(0));
426     int currentFitnessS
427         = hitungFitness(ArraySeleksi.get(0));
428     if (currentWaktuTempuhS
429         < ArraySeleksi_WaktuTempuh.get(0)) {
430         ArraySeleksi_WaktuTempuh.set(0, currentWaktuTempuhS);
431         ArraySeleksi_Fitness.set(0, currentFitnessS);
432

```

Kode 5-16 Hyper-Heuristic Exchange Mutation

Kode 5-16 adalah proses untuk melakukan operator mutasi *exchange*, operator ini bertujuan untuk melakukan pertukaran posisi antar gen di dalam individu. Apabila waktu tempuh setelah proses penukaran menjadi lebih baik, maka gen yang ditukar tersebut akan dipermanenkan, namun apabila tidak maka gen yang ditukar tersebut akan dikembalikan posisinya.


```

440 case 3:
441 //4.2. Menjalankan Operator Add.
442 //menentukan posisi penambahan gen pada individu yang akan
443 //dimutasi dengan operator Add
444 int InsertRandom = new Random().nextInt(ArraySeleksi.get(0).size());
445 //menginisialisasi nilai random yang akan ditambahkan
446 int randomInsertion
447     = new Random().nextInt(jumlahTitik) + 1;
448 if (ArraySeleksi.get(0).contains(randomInsertion)
449     || randomInsertion == titikAwal
450     || randomInsertion == titikAkhir) {
451     continue; //Memastikan gen yang dimasukkan bukan
452 } //merupakan titik awal dan titik akhir
453 ArraySeleksi.get(0).add(InsertRandom, randomInsertion);
454
455 //Menyimpan waktu tempuh dan fitness dari hasil proses
456 //add.
457 int currentWaktuTempuh
458     = hitungWaktuTempuh(ArraySeleksi.get(0));
459 int currentFitness
460     = hitungFitness(ArraySeleksi.get(0));
461
462 //Jika proses add menghasilkan waktu tempuh dan fitness
463 //yang lebih baik atau sama dengan sebelumnya, maka
464 //hasil mutasi akan disimpan menggantikan yang lama.
465 if (currentWaktuTempuh <= tMax
466     && currentFitness >= ArraySeleksi_Fitness.get(0)) {
467     ArraySeleksi_WaktuTempuh.set(0, currentWaktuTempuh);

```

Kode 5-17 Hyper-Heuristic Add Mutation

Kode 5-17 adalah proses untuk menjalankan operator mutasi *Add*. Operator ini bertujuan untuk menambahkan 1 gen acak pada individu sehingga diharapkan nilai *fitness* individu tersebut menjadi lebih baik. Apabila setelah penambahan gen, individu tersebut memiliki nilai *fitness* yang lebih baik dan tidak melanggar batasan *tMax*, maka individu tersebut diterima dan menggantikan individu yang sebelumnya.

```

479     case 4:
480         //memilih posisi gen pada individu yang akan di
481         //mutasi dengan operator omit
482         int DeleteRandom = new Random().nextInt(ArraySeleksi.get(0).size() - 2) + 1;
483         if (!ArraySeleksi.get(0).contains(DeleteRandom)
484             || DeleteRandom == titikAwal
485             || DeleteRandom == titikAkhir) {
486             continue; //Memastikan gen yang dimasukkan bukan
487         }
488         //Proses bisa dilakukan jika ukuran keturunan lebih dari 3
489         if (ArraySeleksi.get(0).size() > 3) {
490             //Menyimpan sementara nilai dari gen/node yang dihapus
491             int sementara = ArraySeleksi.get(0).get(DeleteRandom);
492             ArraySeleksi.get(0).remove(DeleteRandom);
493             if (DeleteRandom == titikAwal || DeleteRandom == titikAkhir)
494             {
495                 continue; //Memastikan gen yang dimasukkan bukan
496             }
497             //Menyimpan waktu tempuh & fitness hasil proses delete
498             int currentWaktuTempuhD
499                 = hitungWaktuTempuh(ArraySeleksi.get(0));
500             int currentFitnessD
501                 = hitungFitness(ArraySeleksi.get(0));
502             if (currentWaktuTempuhD
503                 < ArraySeleksi_WaktuTempuh.get(0)
504                 && currentFitnessD >= ArraySeleksi_Fitness.get(0)) {
505                 ArraySeleksi_WaktuTempuh.set(0, currentWaktuTempuhD);
506                 ArraySeleksi_Fitness.set(0, currentFitnessD);

```

Kode 5-18 Hyper-Heuristic Omit Mutation

Kode 5-18 adalah proses untuk menjalankan operator mutasi *omit*. Operator ini bertujuan untuk menghapus salah 1 gen pada individu dan diharapkan waktu tempuh pada individu tersebut menjadi lebih sedikit. Apabila setelah proses penghapusan gen waktu tempuh menjadi lebih baik, maka gen tersebut akan benar-benar dihapus dari individu, namun apabila tidak, maka gen tersebut akan dikembalikan sesuai dengan posisinya.

5.2.7. Pemberhentian Algoritma Genetika *Hyper-Heuristic*

Apabila sudah sampai iterasi terakhir dari generasi algoritma genetika *hyper-heuristic*, maka algoritma ini secara otomatis akan berhenti dan semua hasil akan dicetak pada file dengan ekstensi `' .txt'`.

```

679         generasi++;
680
681     } while (generasi < jumlahGenerasi);
682
683     pw.close();

```

Kode 5-19 Pemberhentian Algoritma Genetika *Hyper-Heuristic*

Kode 5-19 adalah proses pemberhentian algoritma genetika *hyper-heuristic*. Individu hasil dari proses *hyper-heuristic* generasi terakhir ini akan menjadi solusi yang paling optimal pada tugas akhir ini.

5.3. Pembuatan Aplikasi Pencarian Rute Wisata

Pada bagian ini dijelaskan bagaimana Aplikasi Pencarian Wisata yang telah dirancang pada bab sebelumnya dibuat. Tahapan-tahapan dari pembuatan aplikasi akan disertakan dengan potongan kode dibuat dengan Bahasa pemrograman Java menggunakan aplikasi NetBeans IDE 7.4.

5.3.1. Inisialisasi Rute Perjalanan

Untuk menghasilkan rute perjalanan wisata, diperlukan beberapa komponen masukan seperti titik awal, titik akhir dan waktu maksimal perjalanan. Ketiga komponen tersebut setelah dimasukkan oleh pengguna, maka perlu diolah sehingga bisa masuk ke program algoritma genetika *hyper-heuristic* (AGHH) untuk mencari solusi rute optimal.

```

164 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
165     AlGeHyHe.titikAwal = TitikAwal.getSelectedIndex() + 1;
166     AlGeHyHe.titikAkhir = TitikAkhir.getSelectedIndex() + 1;
167     AlGeHyHe.tMax = Integer.parseInt(TMAX.getText());
168     String []args = new String[0];
169     Scanner s = new Scanner(System.in);
170     try {
171         AlGeHyHe.main(args);
172     } catch (IOException ex) {
173     }
174
175 }

```

Kode 5-20 Inisialisasi Rute Perjalanan

Kode 5-20 berfungsi untuk mengganti parameter titik awal, titik akhir dan tMax pada AGHH dengan data masukan yang telah dipilih oleh pengguna aplikasi.

5.3.2. Menghasilkan Solusi Rute Wisata Optimal

Setelah komponen masukan dibaca, maka program AGHH bisa berjalan untuk mencari rute wisata yang optimal. Rute optimal yang dihasilkan AGHH masih berupa deretan angka, selanjutnya deretan angka tersebut dirubah menjadi list rekomendasi tempat-tempat mana saja yang dikunjungi.

```

176 ArrayList<Integer> seleksi = AlGeHyHe.ArraySeleksi.get(0);
177 System.out.println(seleksi.size());
178 HasilPencarian.setText("");
179 int j = 1;
180 for (int i = 0; i < seleksi.size(); i++) {
181     System.out.println(seleksi.get(i).toString());
182     String namaTempat = TitikAval.getItemAt(seleksi.get(i)-1).toString();
183     System.out.println(namaTempat);
184     String text = HasilPencarian.getText();
185     if (i==0) {
186         HasilPencarian.setText(text + "●\n" +namaTempat + " (Titik Awal)");
187     }
188     else if (i==seleksi.size() - 1){
189         HasilPencarian.setText(text + "\n\n" + "●\n" +namaTempat + " (Titik Akhir)");
190     }
191     else{
192         HasilPencarian.setText(text + "\n\n" + "●\n" + namaTempat);
193     } j++;
194 }
195 StyledDocument doc = HasilPencarian.getStyledDocument();
196 SimpleAttributeSet center = new SimpleAttributeSet();
197 StyleConstants.setAlignment(center, StyleConstants.ALIGN_CENTER);
198 doc.setParagraphAttributes(0, doc.getLength(), center, false);
199 AlGeHyHe.reset();

```

Kode 5-21 Proses Menghasilkan Solusi Rute Wisata Optimal

Kode 5-21 berfungsi untuk merubah deretan angka hasil program AGHH menjadi list rekomendasi tempat-tempat wisata sehingga pengguna lebih mudah untuk membaca solusi rute wisata yang optimal.

BAB VI HASIL DAN PEMBAHASAN

Pada bab keenam ini dijelaskan mengenai hasil yang didapatkan dari penelitian ini dan pembahasannya.

6.1. Lingkungan Uji Coba

Lingkungan uji coba meliputi perangkat keras dan perangkat lunak yang digunakan selama proses penelitian. Perangkat keras yang dipakai adalah sebagai berikut:

Tabel 6-1 Perangkat Keras Yang Digunakan

PERANGKAT KERAS	SPESIFIKASI
Jenis	Laptop
Prosesor	Intel Core i3 2348M 2.30GHz
RAM	4,00 GB
<i>Hard Disk Drive</i> (HDD)	500 GB

Kemudian untuk perangkat lunak yang digunakan adalah sebagai berikut:

Tabel 6-2 Perangkat Lunak Yang Digunakan

PERANGKAT LUNAK	FUNGSI
Windows 7 Ultimate	Sistem Operasi
Paint	Membuat model jejaring
NetBeans IDE 7.4	Membuat dan menjalankan algoritma
Java 1.8	Bahasa pemrograman
Ms. Word Office 365	Membuat laporan
Ms. Word Excel 365	Membuat matriks dan merekap data

6.2. Hasil Penggambaran Model Jejaring

Dalam model jejaring, setiap tempat wisata yang dipakai akan digambarkan dengan titik ungu, titik pertemuan rute digambarkan dengan titik abu-abu dan lalu jalur bis digambarkan dengan garis biru. Nomor untuk setiap tempat diletakkan disebelah titik tersebut.

6.3. Hasil Pencarian Waktu Tempuh Keseluruhan

Setelah menjalankan algoritma Dijkstra untuk merubah matriks waktu tempuh antar titik menjadi matriks waktu tempuh keseluruhan. Gambar 6-1 adalah potongan hasil matriks waktu tempuh keseluruhan, untuk waktu tempuh keseluruhan selengkapnya ada pada LAMPIRAN D.

	A	B	C	D	E
1	0	5	8	12	14
2	5	0	3	7	9
3	8	3	0	4	6
4	12	7	4	0	2
5	14	9	6	2	0
6	26	21	18	14	12

Gambar 6-1 Potongan Hasil Matriks Keseluruhan

Dari penggalan matriks di atas, dapat dilihat jika algoritma dijkstra bisa digunakan untuk menemukan waktu tempuh antar titik sehingga setiap titik pada model jejaring menjadi terhubung.

6.4. Hasil Pencarian Solusi menggunakan Algoritma Genetika *Hyper-Heuristic*

Program akan dijalankan untuk mencari solusi dari permasalahan pencarian rute optimal dari titik awal berangkat Terminal Bungurasih, sampai ke titik akhir Terminal

Pelabuhan Tanjung Perak. Waktu tempuh maksimal adalah 60 menit.

6.4.1. Validasi Pembangkitan Populasi Layak Awal

Sebelum proses mencari solusi optimal, maka perlu dilakukan validasi solusi awal yang dibangkitkan. Solusi awal berjumlah 75 individu sesuai batasan yang dibuat. Berikut adalah tabel dari pembangkitan solusi awal.

Tabel 6-3 Hasil Pembangkitan Populasi Awal

Individu	[Solusi] (Waktu Tempuh, Skor)
1	[1, 31, 41] (52, 0)
2	[1, 3, 20, 22, 39, 41] (56, 500)
3	[1, 11, 41] (56, 11)
4	[1, 2, 41] (52, 16)
5	[1, 19, 41] (52, 55)
6	[1, 4, 41] (52, 8)
7	[1, 23, 32, 41] (57, 72)
8	[1, 10, 29, 41] (52, 14)
9	[1, 9, 10, 26, 41] (60, 309)
10	[1, 34, 41] (52, 0)
11	[1, 2, 8, 19, 20, 41] (52, 210)
12	[1, 7, 9, 41] (52, 304)
13	[1, 33, 41] (52, 0)
14	[1, 8, 41] (52, 22)
15	[1, 4, 32, 31, 41] (52, 10)
16	[1, 26, 41] (56, 70)
17	[1, 32, 41] (52, 2)
18	[1, 10, 20, 41] (57, 117)
19	[1, 21, 22, 41] (56, 364)
20	[1, 10, 41] (52, 0)
21	[1, 27, 41] (56, 0)

22	[1, 22, 31, 41] (56, 276)
23	[1, 9, 41] (52, 239)
24	[1, 29, 36, 41] (52, 20)
25	[1, 5, 7, 9, 33, 41] (59, 304)
26	[1, 21, 41] (55, 88)
27	[1, 29, 41] (52, 14)
28	[1, 19, 23, 40, 41] (56, 126)
29	[1, 40, 41] (52, 1)
30	[1, 20, 21, 41] (55, 205)
31	[1, 29, 32, 41] (52, 16)
32	[1, 5, 32, 41] (53, 2)
33	[1, 11, 28, 41] (60, 92)
34	[1, 20, 41] (52, 117)
35	[1, 23, 41] (56, 70)
36	[1, 2, 23, 41] (56, 86)
37	[1, 7, 41] (52, 65)
38	[1, 9, 36, 41] (52, 245)
39	[1, 36, 41] (52, 6)
40	[1, 2, 29, 41] (52, 30)

Solusi yang dihasilkan di atas bisa dikatakan layak dan valid apabila tidak melanggar batasan yang sudah didefinisikan pada bab sebelumnya, yaitu:

1. Batasan 1 terpenuhi. Semua individu yang dibangkitkan dimulai dari titik ke 1 dan berakhir di titik ke 41.
2. Batasan 2 terpenuhi. Semua titik saling terhubung dan setiap titik hanya dikunjungi maksimal satu kali.
3. Batasan 3 terpenuhi. Tidak ada individu yang memiliki t_{Max} lebih dari 60 menit.
4. Batasan 4 dan 5 terpenuhi. Tidak ada *subtours* yaitu lintasan dimulai dan berakhir di titik yang sama. Ini

dibuktikan dengan gen awal dan akhir hanya muncul 1 kali pada setiap individu solusi.

Dari keempat batasan di atas, dapat disimpulkan bahwa semua solusi awal yang dihasilkan dari proses pembangkitan populasi awal adalah layak dan valid.

Berdasarkan nilai *fitness*, individu terbaik saat pembangkitan populasi awal adalah individu dengan nilai *fitness* sebesar 434. Selengkapnya ada di bawah ini:

[1, 3, 20, 22, 39, 41] Waktu Tempuh: 56 Fitness: 500
--

6.4.2. Validasi Solusi Akhir

Setelah program berjalan hingga iterasi ke 1000, maka algoritma yang berjalan dihentikan. Individu terbaik pada generasi terakhir akan terpilih menjadi solusi optimal.

Berdasarkan nilai *fitness*, maka didapatkan solusi dengan nilai *fitness* tertinggi, yaitu individu dengan nilai *fitness* sebesar 1125. Selengkapnya ada di bawah ini:

[1, 2, 3, 4, 7, 9, 19, 20, 21, 22, 23, 26, 27, 29, 39, 41] Waktu Tempuh: 57 Fitness: 1125

Solusi optimal ini dikatakan layak apabila tidak melanggar batasan yang sudah didefinisikan pada bab sebelumnya, yaitu:

1. Batasan 1 terpenuhi. Individu solusi optimal dimulai dari titik ke 1 dan berakhir di titik ke 41.
2. Batasan 2 terpenuhi. Semua titik saling terhubung dan setiap titik hanya dikunjungi maksimal satu kali.
3. Batasan 3 terpenuhi. Individu solusi optimal memiliki tMax kurang dari 60 menit.

4. Batasan 4 dan 5 terpenuhi. Tidak ada *subtours* yaitu lintasan dimulai dan berakhir di titik yang sama. Ini dibuktikan dengan titik awal dan akhir hanya muncul 1 kali pada individu solusi optimal.

Dari keempat batasan di atas, dapat disimpulkan bahwa solusi optimal akhir yang dihasilkan adalah layak dan valid.

Solusi optimal tersebut dianggap sebagai rekomendasi rute wisata paling optimal untuk studi kasus ini, dengan lintasannya ada pada tabel 6-4 berikut:

Tabel 6-4 Rute Wisata Optimal

No	Titik	Nama Objek
1	1	Terminal Bungurasih
2	2	City of Tomorrow Mall
3	3	Masjid Al-Akbar
4	4	Taman Pelangi
5	7	DBL Arena
6	9	Royal Plaza
7	19	DTC Wonokromo
8	20	Kebun Binatang Surabaya
		Patung Suro dan Boyo
9	21	Taman Bungkul
10	22	Tunjungan Plaza
11	23	Monumen Pers Perjuangan Surabaya
12	26	BG Junction
		Pasar Blauran
13	27	Pertigaan Bubutan Raden Saleh
14	29	Stasiun Pasar Turi
15	39	Pertigaan Perak Rajawali
16	41	Pelabuhan Tanjung Perak

Tanda merah pada tabel menunjukkan bahwa obyek pada titik tersebut bukan merupakan tempat wisata, namun sebagai titik pertemuan jalur bis kota sehingga wisatawan tidak perlu berhenti pada titik tersebut kecuali bila akan berganti bis kota.

6.4.3. Hasil Rekomendasi Rute Wisata Pada Aplikasi

Hasil solusi rute wisata yang ditemukan ditampilkan pada aplikasi sebagaimana gambar 6-2 berikut:

APLIKASI PENCARIAN RUTE WISATA DI KOTA SURABAYA

Pilih Titik Awal Pemberangkatan Anda

Pilih Titik Akhir Pemberhentian Anda *Titik Awal dan Akhir tidak boleh sama

Isikan waktu maksimal perjalanan Anda
 Menit MULAI PENCARIAN!

Rekomendasi rute wisata Anda adalah :

●
 Terminal Bungurasih (Titik Awal)
 |
 ●
 City of Tomorrow Mall
 |
 ●
 Masjid Al-Akbar
 |
 ●
 Taman Pelangi
 |
 ●
 DBL Arena
 |
 ●
 Royal Plaza

Gambar 6-2 Rekomendasi Rute Wisata Pada Aplikasi

Solusi rekomendasi rute wisata yang ditemukan menggunakan algoritma genetika *hyper-heuristic* ditampilkan pada aplikasi. Hasil ini sama seperti tabel 6-4 di atas.

6.5. Hasil dan Pembahasan Uji Coba: Membandingkan dengan Algoritma Genetika

Percobaan ini dilakukan dengan membandingkan hasil dari pencarian solusi optimal menggunakan Algoritma Genetika *Hyper-Heuristic* (AGHH), Algoritma Genetika (AG) dan Algoritma Genetika Modifikasi (AGM).

Percobaan ini dilakukan sebanyak 5 kali dengan jumlah iterasi (generasi) yang berbeda-beda, yaitu 200, 400, 600, 800, dan 1000. Hasil dari percobaan ini ditunjukkan pada tabel 6-5.

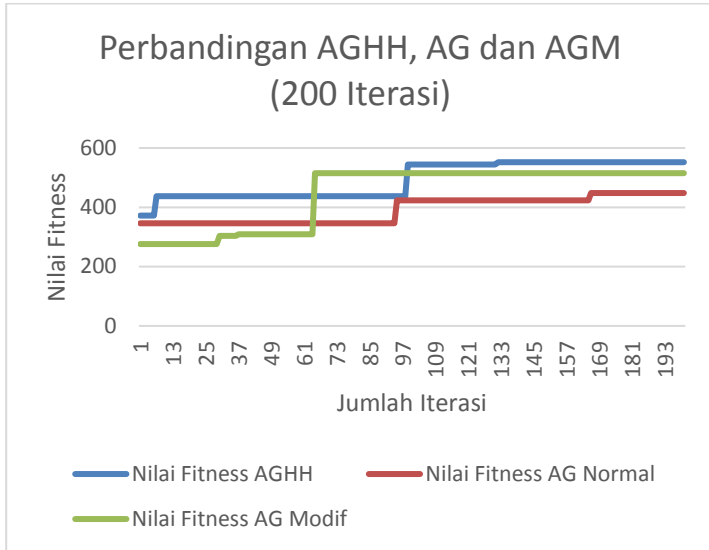
Untuk grafik performa dari masing-masing percobaan perbandingan AGHH, AG dan AG Modifikasi adalah seperti pada gambar 6-2 hingga gambar 6-6.

Tabel 6-5 Perbandingan Performa AGHH, AG dan AG Modifikasi

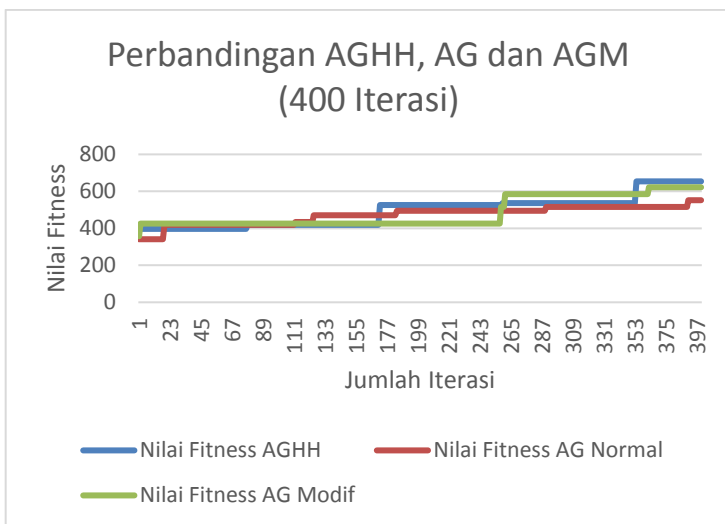
No.	Jumlah Iterasi	AG-HH		AG		AG Modifikasi	
		Nilai Fitness	Waktu Komputasi (detik)	Nilai Fitness	Waktu Komputasi (detik)	Nilai Fitness	Waktu Komputasi (detik)
1	200	552	0.106094275	448	0.332038853	515	0.402557011
2	400	654	0.154538891	552	0.405843858	622	0.422469516
3	600	750	0.142610575	677	0.479376668	703	0.456847983
4	800	993	0.153163646	774	0.464845113	757	0.555661756
5	1000	1125	0.160102344	804	0.646157567	892	0.74191965

Dari hasil tabel 6-5, dapat disimpulkan apabila Algoritma Genetika *Hyper-Heuristic* (AGHH) memberikan solusi dengan nilai fitness yang lebih tinggi daripada Algoritma Genetika (AG) dan Algoritma Genetika Modifikasi (AGM), sehingga ini menunjukkan jika AGHH memberikan solusi lebih baik pada kasus ini dibandingkan dengan AG dan AGM. Selain itu, waktu yang diperlukan untuk proses pencarian solusi menunjukkan apabila AGHH lebih cepat daripada AG dan AGM.

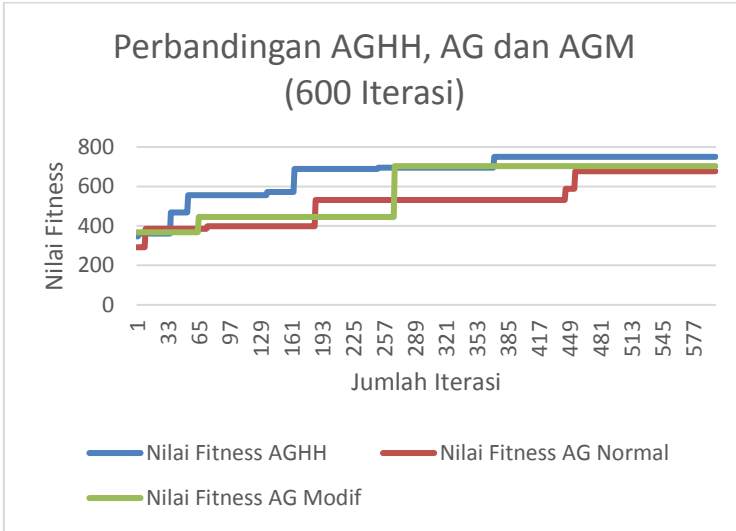
Halaman ini sengaja dikosongkan



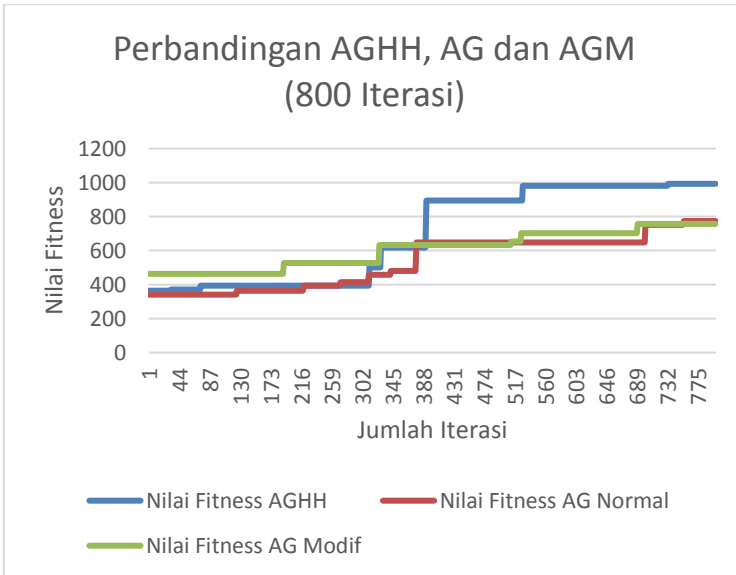
Gambar 6-3 Grafik Performa AGHH dan AH (200 Iterasi)



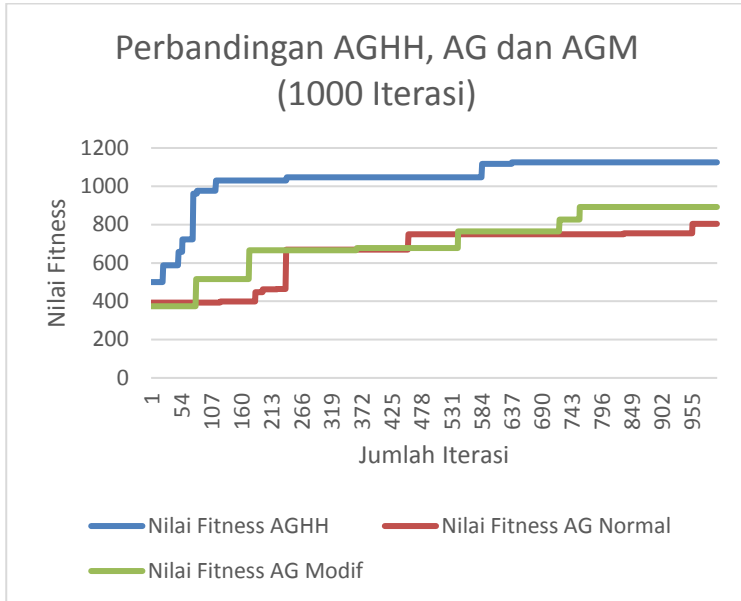
Gambar 6-4 Grafik Performa AGHH dan AH (400 Iterasi)



Gambar 6-5 Grafik Performa AGHH dan AH (600 Iterasi)



Gambar 6-6 Grafik Performa AGHH dan AH (800 Iterasi)



Gambar 6-7 Grafik Performa AGHH dan AH (1000 Iterasi)

Dari grafik pada gambar 6-2 sampai 6-6 dapat diamati bila nilai *fitness* AGHH selalu unggul dibandingkan dengan AG dan AGM dalam setiap iterasi yang berbeda.

Tabel 6-6 sampai 6-8 adalah hasil statistik penggunaan operator pada saat percobaan perbandingan performa AGHH, AG dan AGM.

Tabel 6-6 Penggunaan Operator Untuk Proses AGHH

No.	Jumlah Iterasi	Algoritma Genetika <i>Hyper-Heuristic</i>			
		KW <i>Injection</i>	Mutasi <i>Exchange</i>	Mutasi <i>Add</i>	Mutasi <i>Omit</i>
1	200	62	64	60	14
2	400	138	138	112	12
3	600	188	211	161	40

4	800	258	248	208	86
5	1000	332	329	221	118

Pada tabel 6-6 menunjukkan saat menjalankan program AGHH dalam 5 percobaan, operator yang paling sering digunakan adalah mutasi *exchange* dan yang paling sedikit digunakan adalah mutasi *omit*.

Tabel 6-7 Penggunaan Operator Untuk Proses AG

No.	Jumlah Iterasi	Algoritma Genetika	
		KW <i>Injection</i>	Mutasi <i>Add</i>
1	200	200	200
2	400	400	400
3	600	600	600
4	800	800	800
5	1000	1000	1000

Pada tabel 6-7 menunjukkan saat menjalankan program AG dalam 5 percobaan, jumlah operator yang digunakan adalah sama seperti jumlah iterasi, ini dikarenakan pada setiap iterasi, kedua operator tersebut digunakan secara berurutan.

Tabel 6-8 Penggunaan Operator Untuk Proses AGM

No.	Jumlah Iterasi	Algoritma Genetika Modifikasi			
		KW <i>Injection</i>	Mutasi <i>Exchange</i>	Mutasi <i>Add</i>	Mutasi <i>Omit</i>
1	200	200	61	72	67
2	400	400	140	112	148
3	600	600	185	203	212
4	800	800	263	265	272
5	1000	1000	332	345	323

Pada tabel 6-8 menunjukkan penggunaan operator saat menjalankan program AGM. Untuk operator kawin silang *injection* digunakan terus menerus disetiap iterasi dan operator

mutasi dipilih secara random. Operator *Add* digunakan paling banyak pada percobaan 200 iterasi. Operator *Omit* digunakan paling banyak saat percobaan 400 dan 600 iterasi. yang paling sering digunakan adalah mutasi *exchange* dan yang paling sedikit digunakan adalah mutasi *omit*.

6.6. Hasil dan Pembahasan Uji Coba: Merubah titik awal dan titik akhir

Percobaan dilakukan dengan merubah titik awal dan titik akhir. Jika sebelumnya rute dimulai dari titik ke 1 dan berakhir di titik ke 41, pada percobaan ini titik awal dirubah menjadi titik ke 6 (Taman Flora) dan titik akhir dirubah menjadi titik ke 42 (Terminal Tambak Osowilangun).

6.6.1. Validasi Pembangkitan Populasi Layak Awal

Sebelum proses mencari solusi optimal, maka perlu dilakukan validasi solusi awal yang dibangkitkan. Solusi awal berjumlah 40 individu sesuai batasan yang dibuat. Berikut adalah tabel dari pembangkitan solusi awal.

Individu	[Solusi] (Waktu Tempuh, Skor)
1	[6, 4, 42] (55, 15)
2	[6, 7, 32, 34, 42] (54, 74)
3	[6, 33, 42] (52, 7)
4	[6, 10, 42] (52, 7)
5	[6, 11, 42] (56, 18)
6	[6, 27, 42] (59, 7)
7	[6, 32, 42] (54, 9)
8	[6, 23, 42] (59, 77)
9	[6, 8, 42] (52, 29)
10	[6, 8, 27, 42] (59, 29)
11	[6, 7, 42] (52, 72)
12	[6, 5, 26, 42] (59, 77)

13	[6, 7, 20, 42] (52, 189)
14	[6, 21, 42] (55, 95)
15	[6, 4, 34, 42] (55, 15)
16	[6, 20, 42] (52, 124)
17	[6, 34, 42] (52, 7)
18	[6, 31, 42] (57, 7)
19	[6, 19, 42] (52, 62)
20	[6, 21, 27, 42] (59, 95)
21	[6, 9, 42] (52, 246)
22	[6, 22, 34, 42] (59, 283)
23	[6, 5, 42] (52, 7)
24	[6, 29, 42] (54, 21)
25	[6, 22, 29, 42] (59, 297)
26	[6, 8, 33, 42] (52, 29)
27	[6, 10, 29, 42] (54, 21)
28	[6, 20, 21, 19, 42] (58, 267)
29	[6, 4, 7, 42] (55, 80)
30	[6, 4, 20, 42] (55, 132)
31	[6, 7, 26, 42] (59, 142)
32	[6, 22, 42] (59, 283)
33	[6, 21, 33, 42] (55, 95)
34	[6, 7, 31, 42] (57, 72)
35	[6, 7, 33, 42] (52, 72)
36	[6, 29, 34, 42] (54, 21)
37	[6, 26, 42] (59, 77)
38	[6, 8, 10, 42] (60, 29)
39	[6, 20, 23, 42] (59, 194)
40	[6, 5, 8, 42] (58, 29)

Solusi yang dihasilkan di atas bisa dikatakan layak dan valid apabila tidak melanggar batasan yang sudah didefinisikan pada bab sebelumnya, yaitu:

1. Batasan 1 terpenuhi. Semua individu yang dibangkitkan dimulai dari titik ke 6 dan berakhir di titik ke 42.
2. Batasan 2 terpenuhi. Semua titik saling terhubung dan setiap titik hanya dikunjungi maksimal satu kali.
3. Batasan 3 terpenuhi. Tidak ada individu yang memiliki tMax lebih dari 60 menit.
4. Batasan 4 dan 5 terpenuhi. Tidak ada *subtours* yaitu lintasan dimulai dan berakhir di titik yang sama. Ini dibuktikan dengan gen awal dan akhir hanya muncul 1 kali pada setiap individu solusi.

Dari keempat batasan di atas, dapat disimpulkan bahwa semua solusi awal yang dihasilkan dari proses pembangkitan populasi awal adalah layak dan valid.

6.6.2. Validasi Solusi Akhir

Setelah program berjalan hingga iterasi ke 1000, maka algoritma yang berjalan dihentikan. Individu terbaik pada generasi terakhir akan terpilih menjadi solusi optimal.

Berdasarkan nilai *fitness*, maka didapatkan solusi dengan nilai *fitness* tertinggi, yaitu individu dengan nilai *fitness* sebesar 748. Selengkapnya ada di bawah ini:

[6, 5, 7, 8, 9, 10, 19, 22, 26, 27, 29, 42] | Waktu Tempuh: 59 | Fitness: 748

Solusi optimal ini dikatakan layak apabila tidak melanggar batasan yang sudah didefinisikan pada bab sebelumnya, yaitu:

1. Batasan 1 terpenuhi. Individu solusi optimal dimulai dari titik ke 6 dan berakhir di titik ke 42.
2. Batasan 2 terpenuhi. Semua titik saling terhubung dan setiap titik hanya dikunjungi maksimal satu kali.
3. Batasan 3 terpenuhi. Individu solusi optimal memiliki tMax kurang dari 60 menit.

4. Batasan 4 dan 5 terpenuhi. Tidak ada *subtours* yaitu lintasan dimulai dan berakhir di titik yang sama. Ini dibuktikan dengan titik awal dan akhir hanya muncul 1 kali pada individu solusi optimal.

Dari keempat batasan di atas, dapat disimpulkan bahwa solusi optimal akhir yang dihasilkan adalah layak dan valid.

6.6.3. Hasil Rekomendasi Rute Wisata Pada Aplikasi

Hasil solusi rute wisata dengan merubah titik awal dan akhir ditampilkan pada aplikasi sebagaimana gambar 6-8 berikut:

APLIKASI PENCARIAN RUTE WISATA DI KOTA SURABAYA

Pilih Titik Awal Pemberangkatan Anda

Pilih Titik Akhir Pemberhentian Anda *Titik Awal dan Akhir tidak boleh sama

Isikan waktu maksimal perjalanan Anda
 Menit MULAI PENCARIAN!

Rekomendasi rute wisata Anda adalah :

```

graph TD
    A[Taman Flora (Titik Awal)] --- B[Pertigaan Ahmad Yani Jemur Andayani]
    B --- C[DBL Arena]
    C --- D[Maspion Square]
    D --- E[Royal Plaza]
  
```

Gambar 6-8 Rekomendasi Rute Wisata Pada Aplikasi

Solusi rekomendasi rute wisata yang ditemukan menggunakan algoritma genetika *hyper-heuristic* sudah sesuai dengan perubahan titik awal dan titik akhir yang dilakukan saat uji

coba, yaitu berangkat dari Taman Flora (Titik No. 6) dan berakhir di Terminal Tambak Osowilangun (Titik No. 42).

BAB VII KESIMPULAN DAN SARAN

Pada bab ketujuh ini dijelaskan kesimpulan dari tugas akhir beserta saran yang bermanfaat untuk perbaikan pada penelitian kedepannya.

7.1. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, maka berikut ini adalah kesimpulan yang diperoleh:

1. Algoritma Genetika *Hyper-Heuristic* dapat digunakan untuk mencari solusi yang optimal dari permasalahan OP. Dari permasalahan tugas akhir ini didapatkan nilai *fitness* dari solusi yang terbaik adalah sebesar 1125.
2. Algoritma Genetika *Hyper-Heuristic* (AGHH) mampu menghasilkan solusi yang lebih baik dibandingkan dengan Algoritma Genetika (AG) biasa dan Algoritma Genetika Modifikasi (AGM) pada beberapa kali percobaan menggunakan iterasi yang berbeda-beda. Pada percobaan dengan 200 iterasi, AGHH menghasilkan nilai *fitness* sebesar 552, lebih besar dari AG dengan nilai *fitness* 448 dan AGM dengan nilai *fitness* 515. Pada percobaan dengan 400 iterasi, AGHH menghasilkan nilai *fitness* sebesar 654, lebih besar dari AG dengan nilai *fitness* 552 dan AGM dengan nilai *fitness* 622. Pada percobaan dengan 600 iterasi, AGHH menghasilkan nilai *fitness* sebesar 750, lebih besar dari AG dengan nilai *fitness* 677 dan AGM dengan nilai *fitness* 703. Pada percobaan dengan 800 iterasi, AGHH menghasilkan nilai *fitness* sebesar 993, lebih besar dari AG dengan nilai *fitness* 774 dan AGM dengan nilai *fitness* 757, dan pada percobaan dengan 1000 iterasi, AGHH menghasilkan nilai *fitness* sebesar 1125, lebih besar dari AG dengan nilai *fitness* 804 dan AGM dengan nilai *fitness* 892. Untuk waktu komputasi dari Algoritma Genetika *Hyper-Heuristic*

lebih cepat dibandingkan dengan AG dan AGM, yakni kurang dari 0,2 detik.

3. Hasil dari pencarian solusi menggunakan AGHH, AG maupun AGM bisa berubah pada setiap percobaan karena faktor probabilitas yang ada pada setiap algoritma.
4. Penentuan operator-operator dan jumlah iterasi pada algoritma genetika *hyper-heuristic* bisa mempengaruhi hasil optimasi yang didapatkan.

7.2. Saran

Saran yang diberikan penulis berdasarkan hasil dari penelitian yang dikerjakan untuk penelitian kedepannya adalah:

1. Penggunaan operator *hyper-heuristic* pada tugas akhir ini masih terbatas, untuk penelitian selanjutnya bisa ditambahkan lagi operator-operator yang lain, seperti *single point cross-over*, *move* dan *replace*.
2. Proses *hyper-heuristic* yang digunakan pada tugas akhir ini menggunakan pemilihan secara random untuk operator yang dipakai disetiap iterasi, untuk penelitian selanjutnya bisa menggunakan metode *reinforcement learning* atau *greedy*.
3. Waktu tempuh yang dipakai masih statis karena berdasarkan hasil pencarian melalui *Google Maps*. Penelitian selanjutnya bisa dikembangkan lagi berdasarkan jam keberangkatan pagi atau malam dan juga hari-hari tertentu melalui observasi langsung di lapangan sehingga data waktu tempuh menjadi lebih akurat dan bermanfaat.
4. Objek penelitian ini hanya sebatas 10 trayek bis kota di Kota Surabaya, penelitian selanjutnya bisa diterapkan untuk bis kota yang beroperasi di kota-kota lain yang lebih besar dan kompleks, seperti di Kota Jakarta, Kota Bandung atau Kota Yogyakarta.
5. Rekomendasi rute wisata pada tugas akhir ini hanya menunjukkan tempat-tempat wisata yang bisa

dikunjungi, penelitian selanjutnya bisa ditambahkan keterangan trayek bis yang bisa digunakan untuk mengunjungi tempat-tempat wisata tersebut.

DAFTAR PUSTAKA

- [1] “Kemacetan Surabaya.” [Online]. Available: <https://tesarakram.wordpress.com/2013/12/27/kemacetan-surabaya/>. [Accessed: 19-Jul-2018].
- [2] “Ada Apa dengan Transportasi Umum di Surabaya?” [Online]. Available: <https://ehsurabaya.wordpress.com/2017/02/16/2130/>. [Accessed: 19-Jul-2018].
- [3] P. Vansteenwegen, S. Wouter, and D. Van Oudheusden, “The orienteering problem: a survey,” vol. 209, pp. 1–10, 2011.
- [4] M. F. Tasgetiren, “A Genetic Algorithm with an Adaptive Penalty Function for the Orienteering Problem,” vol. 4, no. 2, pp. 1–26.
- [5] Y. Mei, “Genetic Programming Hyper-heuristic for Stochastic Team Orienteering Problem with Time Windows.”
- [6] “Transportasi Umum di Kota Surabaya.” [Online]. Available: <http://www.surabaya.go.id/berita/8263-transportasi>. [Accessed: 10-Mar-2018].
- [7] S. Kiranyaz, T. Ince, and M. Gabbouj, *Multidimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition*, vol. 15. 2014.
- [8] “Arti Kata Optimal.” [Online]. Available: <https://kbbi.web.id/optimal>. [Accessed: 10-Mar-2018].
- [9] “Definition of Optimization.” [Online]. Available: <https://en.oxforddictionaries.com/definition/optimization>. [Accessed: 10-Mar-2018].
- [10] S. Mujilawati, “Pre-Processing Text Mining Pada Data Twitter,” *Semin. Nas. Teknol. Inf. dan Komun.*, vol. 2016, no. Sentika, pp. 2089–9815, 2016.
- [11] B. W. Taylor, *Introduction to Management Science*, vol. 58, no. 3. 2013.
- [12] D. Pugas, M. Somantri, and K. Satoto, “Pencarian Rute Terpendek Menggunakan Algoritma Dijkstra dan Astar (A*) pada SIG Berbasis Web untuk Pemetaan

- Pariwisata Kota Sawahlunto,” *Transmisi*, vol. 13, no. 1, pp. 27–32, 2011.
- [13] A. Gunawan, H. Chuin, and P. Vansteenwegen, “Orienteering Problem: A survey of recent variants , solution approaches and applications,” vol. 255, pp. 315–317, 2016.
- [14] W. F. M. Candra Aditya1), “Optimasi Persediaan Baju Menggunakan Algoritma Genetika,” *Pros. Semin. Nas. Teknol. dan Rekayasa Inf. Tahun 2016 “Peran Teknol. dan Rekayasa Inf. dalam Implementasi Geostrategi Indones.*, no. December, pp. 65–70, 2016.
- [15] I. Mutakhirroh, F. Saptono, N. Hasanah, and R. Wiryadinata, “Pemanfaatan Metode Heuristik Dalam Pencarian Jalur Terpendek Dengan Algoritma Semut dan Algoritma Genetika,” *SNATI (Seminar Nas. Apl. Teknol. Informasi) 2007*, vol. 2007, no. Snati, pp. B33–B39, 2007.
- [16] E. Satriyana, “Algoritma Genetika,” pp. 81–87, 2011.
- [17] F. Saptono and H. Taufiq, “PERANCANGAN ALGORITMA GENETIKA UNTUK MENENTUKAN JALUR TERPENDEK,” vol. 2007, no. Snati, pp. 75–79, 2007.
- [18] E. K. Burke *et al.*, “Hyper-heuristics: A survey of the state of the art,” *J. Oper. Res. Soc.*, vol. 64, no. 12, pp. 1695–1724, 2013.
- [19] G. E. Moglen and E. Ozcan, “Fundamentals of Hyper-Heuristic,” *Thirty-fourth SGAI Int. Conf. Artif. Intell.*, pp. 36–72, 2014.
- [20] P. Larranaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, “Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators,” *Artif. Intell. Rev.*, vol. 13, no. 2, pp. 129–170, 1999.

BIODATA PENULIS



Penulis lahir di Kabupaten Bojonegoro pada hari Kamis, tanggal 20 Oktober 1994. Penulis merupakan anak kedua dari empat bersaudara. Penulis telah menempuh beberapa pendidikan formal yaitu, SDN Ngronggo VIII Kota Kediri, MTsN 2 Kota Kediri, dan SMAN 2 Kota Kediri. Pada tahun 2014, penulis melanjutkan pendidikan ke jenjang perguruan tinggi di Departemen Sistem Informasi, FTIK, Institut Teknologi Sepuluh Nopember (ITS) Surabaya melalui jalur SBMPTN dan terdaftar sebagai mahasiswa dengan NRP 05211440000116.

Selama menjadi mahasiswa, penulis mengikuti berbagai kegiatan kemahasiswaan seperti beberapa organisasi dan kepanitiaan. Penulis pernah bertugas sebagai Kakak Pendamping selama Manage 2015/2016 di Himpunan Mahasiswa Sistem Informasi (HMSI), ITS pada tahun kedua dan pernah menjabat sebagai Bendahara Umum HMSI, ITS pada tahun ketiga perkuliahan. Penulis juga aktif di kepanitiaan Information System Expo (ISE) tahun 2016 dan 2017 serta GERIGI ITS 2016. Penulis juga pernah diberikan kesempatan untuk melakukan kerja praktik di Krakatau Information Technology di Kota Cilegon, Banten pada bagian IT Development.

Pada tahun keempat, karena penulis memiliki ketertarikan pada bidang perekayasaan data, maka penulis memilih untuk bergabung di Laboratorium Rekayasa Data dan Inteligensi Bisnis (RDIB), Sistem Informasi, ITS. Untuk keperluan penelitian dan lain-lain, penulis dapat dihubungi melalui surel: imadshinn.yukikolatifa@gmail.com .

Halaman ini sengaja dikosongkan

LAMPIRAN A: LOKASI DAN SKOR TIAP TITIK

Berikut adalah tabel lokasi beserta skor:

No.	Lokasi	Skor	Skor Total
1	Terminal Bungurasih	0	0
2	City of Tomorrow Mall	16	16
3	Masjid Al-Akbar	107	107
4	Taman Pelangi	8	8
5	Pertigaan Ahmad Yani Jemur Andayani	0	0
6	Taman Flora	7	7
7	DBL Arena	65	65
8	Maspion Square	22	22
9	Royal Plaza	239	239
10	Pertigaan Wonokromo	0	0
11	Stasiun Wonokromo	11	11
12	Marvel City	15	15
13	Monumen Kapal Selam	24	95
	Plaza Surabaya	71	
14	Stasiun Gubeng	34	34
15	Grand City Mall	69	69
16	Hi-Tech Mall	59	59
17	Pasar Atom Surabaya	29	69
	ITC Surabaya Mega Grosir	40	
18	Wisata Religi Ampel	3	3
19	DTC Wonokromo	55	55
20	Kebun Binatang Surabaya	110	117
	Patung Suro dan Boyo	7	
21	Taman Bungkul	88	88
22	Tunjungan Plaza	276	276

A-2

No.	Lokasi	Skor	Skor Total
23	Monumen Pers Perjuangan Surabaya	70	70
24	Hotel Majapahit (Yamato)	126	126
25	Museum Surabaya (Gedung Siola)	48	48
26	BG Junction	36	70
	Pasar Blauran	34	
27	Pertigaan Bubutan Raden Saleh	0	0
28	Museum Dr. Soetomo GNI	81	81
29	Stasiun Pasar Turi	14	14
30	Tugu Pahlawan	92	92
31	Perempatan Tembaan Bubutan	0	0
32	Pasar Turi Baru	2	2
33	Perempatan Kranggan Tembok Dukuh	0	0
34	Perempatan Demak Dupak	0	0
35	Stasiun Surabaya Kota (Semut)	3	3
36	Museum Kesehatan Dr. Adhyatma	6	6
37	Jembatan Merah Plaza	42	42
38	Pertigaan Rajawali Indrapura	0	0
39	Pertigaan Perak Rajawali	0	0
40	Monumen Pelayaran	1	1
41	Pelabuhan Tanjung Perak	0	0
42	Terminal Tambak Osowilangun	0	0

LAMPIRAN B: NILAI WAKTU TEMPUH TIAP TITIK

Berikut adalah tabel nilai waktu tempuh antar titik pada model jejaring:

Titik Awal	Titik Akhir	Waktu tempuh (menit)
1	2	5
2	1	5
2	3	3
3	2	3
3	4	4
4	3	4
4	5	3
4	7	3
5	4	2
5	6	12
6	5	12
7	5	3
7	8	1
8	7	1
8	9	2
9	8	2
9	10	2
10	9	2
10	11	2
10	19	1
11	10	2
11	12	4
12	11	5
12	13	7
13	12	8

B-2

Titik Awal	Titik Akhir	Waktu tempuh (menit)
13	14	1
14	13	1
14	15	1
15	14	1
15	16	3
16	15	3
16	17	7
17	18	1
18	17	1
18	16	8
19	10	1
19	20	2
20	19	1
20	21	2
20	33	10
21	20	1
21	22	8
22	21	8
22	23	1
23	22	1
23	26	4
24	23	1
25	24	1
25	26	1
26	27	1
27	28	1
27	29	1
28	31	1

Titik Awal	Titik Akhir	Waktu tempuh (menit)
29	32	1
29	33	2
30	25	3
30	31	1
31	32	2
31	36	3
32	31	1
32	34	3
33	20	10
33	29	2
33	34	4
34	32	4
34	33	4
34	39	6
34	42	17
35	30	2
36	38	2
37	35	2
38	37	2
38	39	1
39	34	5
39	38	1
39	40	3
40	39	3
40	41	7
41	40	7
42	34	16

Halaman ini sengaja dikosongkan

LAMPIRAN C: VARIABEL KEPUTUSAN OP

Berikut adalah tabel variabel keputusan *orienteeering problem* yang digunakan dalam tugas akhir:

No	Variabel	Deskripsi
1	x1.2	Kunjungan dari titik 1 ke titik 2
2	x2.1	Kunjungan dari titik 2 ke titik 1
3	x2.3	Kunjungan dari titik 2 ke titik 3
4	x3.2	Kunjungan dari titik 3 ke titik 2
5	x3.4	Kunjungan dari titik 3 ke titik 4
6	x4.3	Kunjungan dari titik 4 ke titik 3
7	x4.5	Kunjungan dari titik 4 ke titik 5
8	x4.7	Kunjungan dari titik 4 ke titik 7
9	x5.4	Kunjungan dari titik 5 ke titik 4
10	x5.6	Kunjungan dari titik 5 ke titik 6
11	x6.5	Kunjungan dari titik 6 ke titik 5
12	x7.5	Kunjungan dari titik 7 ke titik 5
13	x7.8	Kunjungan dari titik 7 ke titik 8
14	x8.7	Kunjungan dari titik 8 ke titik 7
15	x8.9	Kunjungan dari titik 8 ke titik 9
16	x9.8	Kunjungan dari titik 9 ke titik 8
17	x9.10	Kunjungan dari titik 9 ke titik 10
18	x10.9	Kunjungan dari titik 10 ke titik 9
19	x10.11	Kunjungan dari titik 10 ke titik 11
20	x10.19	Kunjungan dari titik 10 ke titik 19
21	x11.10	Kunjungan dari titik 11 ke titik 10
22	x11.12	Kunjungan dari titik 11 ke titik 12
23	x12.11	Kunjungan dari titik 12 ke titik 11
24	x12.13	Kunjungan dari titik 12 ke titik 13
25	x13.12	Kunjungan dari titik 13 ke titik 12
26	x13.14	Kunjungan dari titik 13 ke titik 14

No	Variabel	Deskripsi
27	x14.13	Kunjungan dari titik 14 ke titik 13
28	x14.15	Kunjungan dari titik 14 ke titik 15
29	x15.14	Kunjungan dari titik 15 ke titik 14
30	x15.16	Kunjungan dari titik 15 ke titik 16
31	x16.15	Kunjungan dari titik 16 ke titik 15
32	x16.17	Kunjungan dari titik 16 ke titik 17
33	x17.18	Kunjungan dari titik 17 ke titik 18
34	x18.17	Kunjungan dari titik 18 ke titik 17
35	x18.16	Kunjungan dari titik 18 ke titik 16
36	x19.10	Kunjungan dari titik 19 ke titik 10
37	x19.20	Kunjungan dari titik 19 ke titik 20
38	x20.19	Kunjungan dari titik 20 ke titik 19
39	x20.21	Kunjungan dari titik 20 ke titik 21
40	x20.33	Kunjungan dari titik 20 ke titik 33
41	x21.20	Kunjungan dari titik 21 ke titik 20
42	x21.22	Kunjungan dari titik 21 ke titik 22
43	x22.21	Kunjungan dari titik 22 ke titik 21
44	x22.23	Kunjungan dari titik 22 ke titik 23
45	x23.22	Kunjungan dari titik 23 ke titik 22
46	x23.26	Kunjungan dari titik 23 ke titik 26
47	x24.23	Kunjungan dari titik 24 ke titik 23
48	x25.24	Kunjungan dari titik 25 ke titik 24
49	x25.26	Kunjungan dari titik 25 ke titik 26
50	x26.27	Kunjungan dari titik 26 ke titik 27
51	x27.28	Kunjungan dari titik 27 ke titik 28
52	x27.29	Kunjungan dari titik 27 ke titik 29
53	x28.31	Kunjungan dari titik 28 ke titik 31
54	x29.32	Kunjungan dari titik 29 ke titik 32
55	x29.33	Kunjungan dari titik 29 ke titik 33

No	Variabel	Deskripsi
56	x30.25	Kunjungan dari titik 30 ke titik 25
57	x30.31	Kunjungan dari titik 30 ke titik 31
58	x31.32	Kunjungan dari titik 31 ke titik 32
59	x31.36	Kunjungan dari titik 31 ke titik 36
60	x32.31	Kunjungan dari titik 32 ke titik 31
61	x32.34	Kunjungan dari titik 32 ke titik 34
62	x33.20	Kunjungan dari titik 33 ke titik 20
63	x33.29	Kunjungan dari titik 33 ke titik 29
64	x33.34	Kunjungan dari titik 33 ke titik 34
65	x34.32	Kunjungan dari titik 34 ke titik 32
66	x34.33	Kunjungan dari titik 34 ke titik 33
67	x34.39	Kunjungan dari titik 34 ke titik 39
68	x34.42	Kunjungan dari titik 34 ke titik 42
69	x35.30	Kunjungan dari titik 35 ke titik 30
70	x36.38	Kunjungan dari titik 36 ke titik 38
71	x37.35	Kunjungan dari titik 37 ke titik 35
72	x38.37	Kunjungan dari titik 38 ke titik 37
73	x38.39	Kunjungan dari titik 38 ke titik 39
74	x39.34	Kunjungan dari titik 39 ke titik 34
75	x39.38	Kunjungan dari titik 39 ke titik 38
76	x39.40	Kunjungan dari titik 39 ke titik 40
77	x40.39	Kunjungan dari titik 40 ke titik 39
78	x40.41	Kunjungan dari titik 40 ke titik 41
79	x41.40	Kunjungan dari titik 41 ke titik 40
80	x42.34	Kunjungan dari titik 42 ke titik 34

Halaman ini sengaja dikosongkan

LAMPIRAN D: MATRIKS JARAK KESELURUHAN

Berikut adalah tabel matriks jarak keseluruhan yang digunakan dalam tugas akhir:

Titik	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	0	5	8	12	14	26	14	15	17	19	21	25	32	33	34	37	44	45	20	22	24
2	5	0	3	7	9	21	9	10	12	14	16	20	27	28	29	32	39	40	15	17	19
3	8	3	0	4	6	18	6	7	9	11	13	17	24	25	26	29	36	37	12	14	16
4	12	7	4	0	2	14	2	3	5	7	9	13	20	21	22	25	32	33	8	10	12
5	14	9	6	2	0	12	1	2	4	6	8	12	19	20	21	24	31	32	7	9	11
6	26	21	18	14	12	0	13	14	16	18	20	24	31	32	33	36	43	44	19	21	23
7	17	12	9	5	3	15	0	1	3	5	7	11	18	19	20	23	30	31	6	8	10
8	18	13	10	6	4	16	1	0	2	4	6	10	17	18	19	22	29	30	5	7	9
9	20	15	12	8	6	18	3	2	0	2	4	8	15	16	17	20	27	28	3	5	7
10	22	17	14	10	8	20	5	4	2	0	2	6	13	14	15	18	25	26	1	3	5
11	24	19	16	12	10	22	7	6	4	2	0	4	11	12	13	16	23	24	3	5	7
12	29	24	21	17	15	27	12	11	9	7	5	0	7	8	9	12	19	20	8	10	12
13	37	32	29	25	23	35	20	19	17	15	13	8	0	1	2	5	12	13	16	18	20
14	38	33	30	26	24	36	21	20	18	16	14	9	1	0	1	4	11	12	17	19	21
15	39	34	31	27	25	37	22	21	19	17	15	10	2	1	0	3	10	11	18	20	22
16	42	37	34	30	28	40	25	24	22	20	18	13	5	4	3	0	7	8	21	23	25
17	51	46	43	39	37	49	34	33	31	29	27	22	14	13	12	9	0	1	30	32	34
18	50	45	42	38	36	48	33	32	30	28	26	21	13	12	11	8	15	0	29	31	33
19	23	18	15	11	9	21	6	5	3	1	3	7	14	15	16	19	26	27	0	2	4
20	24	19	16	12	10	22	7	6	4	2	4	8	15	16	17	20	27	28	1	0	2
21	25	20	17	13	11	23	8	7	5	3	5	9	16	17	18	21	28	29	2	1	0
22	33	28	25	21	19	31	16	15	13	11	13	17	24	25	26	29	36	37	10	9	8
23	34	29	26	22	20	32	17	16	14	12	14	18	25	26	27	30	37	38	11	10	9
24	35	30	27	23	21	33	18	17	15	13	15	19	26	27	28	31	38	39	12	11	10
25	36	31	28	24	22	34	19	18	16	14	16	20	27	28	29	32	39	40	13	12	11
26	38	33	30	26	24	36	21	20	18	16	18	22	29	30	31	34	41	42	15	14	16
27	37	32	29	25	23	35	20	19	17	15	17	21	28	29	30	33	40	41	14	13	15
28	44	39	36	32	30	42	27	26	24	22	24	28	35	36	37	40	47	48	21	20	22
29	36	31	28	24	22	34	19	18	16	14	16	20	27	28	29	32	39	40	13	12	14
30	39	34	31	27	25	37	22	21	19	17	19	23	30	31	32	35	42	43	16	15	14
31	43	38	35	31	29	41	26	25	23	21	23	27	34	35	36	39	46	47	20	19	21
32	41	36	33	29	27	39	24	23	21	19	21	25	32	33	34	37	44	45	18	17	19
33	34	29	26	22	20	32	17	16	14	12	14	18	25	26	27	30	37	38	11	10	12
34	38	33	30	26	24	36	21	20	18	16	18	22	29	30	31	34	41	42	15	14	16
35	41	36	33	29	27	39	24	23	21	19	21	25	32	33	34	37	44	45	18	17	19
36	46	41	38	34	32	44	29	28	26	24	26	30	37	38	39	42	49	50	23	22	22
37	43	38	35	31	29	41	26	25	23	21	23	27	34	35	36	39	46	47	20	19	18
38	44	39	36	32	30	42	27	26	24	22	24	28	35	36	37	40	47	48	21	20	20
39	43	38	35	31	29	41	26	25	23	21	23	27	34	35	36	39	46	47	20	19	21
40	46	41	38	34	32	44	29	28	26	24	26	30	37	38	39	42	49	50	23	22	24
41	53	48	45	41	39	51	36	35	33	31	33	37	44	45	46	49	56	57	30	29	31
42	54	49	46	42	40	52	37	36	34	32	34	38	45	46	47	50	57	58	31	30	32

Halaman ini sengaja dikosongkan

Titik	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
1	32	33	51	50	37	38	39	34	47	36	35	32	36	45	39	43	41	42	45	52	53
2	27	28	46	45	32	33	34	29	42	31	30	27	31	40	34	38	36	37	40	47	48
3	24	25	43	42	29	30	31	26	39	28	27	24	28	37	31	35	33	34	37	44	45
4	20	21	39	38	25	26	27	22	35	24	23	20	24	33	27	31	29	30	33	40	41
5	19	20	38	37	24	25	26	21	34	23	22	19	23	32	26	30	28	29	32	39	40
6	31	32	50	49	36	37	38	33	46	35	34	31	35	44	38	42	40	41	44	51	52
7	18	19	37	36	23	24	25	20	33	22	21	18	22	31	25	29	27	28	31	38	39
8	17	18	36	35	22	23	24	19	32	21	20	17	21	30	24	28	26	27	30	37	38
9	15	16	34	33	20	21	22	17	30	19	18	15	19	28	22	26	24	25	28	35	36
10	13	14	32	31	18	19	20	15	28	17	16	13	17	26	20	24	22	23	26	33	34
11	15	16	34	33	20	21	22	17	30	19	18	15	19	28	22	26	24	25	28	35	36
12	20	21	39	38	25	26	27	22	35	24	23	20	24	33	27	31	29	30	33	40	41
13	28	29	47	46	33	34	35	30	43	32	31	28	32	41	35	39	37	38	41	48	49
14	29	30	48	47	34	35	36	31	44	33	32	29	33	42	36	40	38	39	42	49	50
15	30	31	49	48	35	36	37	32	45	34	33	30	34	43	37	41	39	40	43	50	51
16	33	34	52	51	38	39	40	35	48	37	36	33	37	46	40	44	42	43	46	53	54
17	42	43	61	60	47	48	49	44	57	46	45	42	46	55	49	53	51	52	55	62	63
18	41	42	60	59	46	47	48	43	56	45	44	41	45	54	48	52	50	51	54	61	62
19	12	13	31	30	17	18	19	14	27	16	15	12	16	25	19	23	21	22	25	32	33
20	10	11	29	28	15	16	17	12	25	14	13	10	14	23	17	21	19	20	23	30	31
21	8	9	30	29	13	14	15	13	26	15	14	11	15	24	18	22	20	21	24	31	32
22	0	1	23	22	5	6	7	7	19	8	8	9	11	17	11	15	13	14	17	24	28
23	1	0	22	21	4	5	6	6	18	7	7	8	10	16	10	14	12	13	16	23	27
24	2	1	0	22	5	6	7	7	19	8	8	9	11	17	11	15	13	14	17	24	28
25	3	2	1	0	1	2	3	3	15	4	4	5	7	13	7	11	9	10	13	20	24
26	20	19	18	17	0	1	2	2	14	3	3	4	6	12	6	10	8	9	12	19	23
27	19	18	17	16	17	0	1	1	13	2	2	3	5	11	5	9	7	8	11	18	22
28	18	17	16	15	16	17	0	12	12	1	3	10	6	10	4	8	6	7	10	17	23
29	19	18	17	16	17	18	19	0	13	2	1	2	4	11	5	9	7	8	11	18	21
30	6	5	4	3	4	5	6	6	0	1	3	8	6	10	4	8	6	7	10	17	23
31	17	16	15	14	15	16	17	11	11	0	2	9	5	9	3	7	5	6	9	16	22
32	18	17	16	15	16	17	18	9	12	1	0	7	3	10	4	8	6	7	10	17	20
33	20	20	19	18	19	20	21	2	15	4	3	0	4	13	7	11	9	10	13	20	21
34	19	18	17	16	17	18	19	6	13	5	4	4	0	11	8	9	7	6	9	16	17
35	8	7	6	5	6	7	8	8	2	3	5	10	8	0	6	10	8	9	12	19	25
36	14	13	12	11	12	13	14	14	8	9	11	12	8	6	0	4	2	3	6	13	25
37	10	9	8	7	8	9	10	10	4	5	7	12	10	2	8	0	10	11	14	21	27
38	12	11	10	9	10	11	12	12	6	7	9	10	6	4	10	2	0	1	4	11	23
39	13	12	11	10	11	12	13	11	7	8	9	9	5	5	11	3	1	0	3	10	22
40	16	15	14	13	14	15	16	14	10	11	12	12	8	8	14	6	4	3	0	7	25
41	23	22	21	20	21	22	23	21	17	18	19	19	15	15	21	13	11	10	7	0	32
42	35	34	33	32	33	34	35	22	29	21	20	20	16	27	24	25	23	22	25	32	0