PEMILIHAN JALUR WISATA PULAU LOMBOK MENGGUNAKAN HIBRIDISASI ALGORITMA GENETIKA DAN ALGORITMA KOLONI SEMUT

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
MUHAMMAD GUNTUR PUTRA TIMUR
NIM. 115060807111014

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

PEMILIHAN JALUR WISATA PULAU LOMBOK MENGGUNAKAN HIBRIDISASI ALGORITMA GENETIKA DAN ALGORITMA KOLONI SEMUT

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Muhammad Guntur Putra Timur
NIM: 115060807111014

Skripsi ini telah diuji dan dinyatakan lulus pada
18 Januari 2016
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Imam Cholissodin,S.Si,.M.Kom	Edy Santoso, S.Si., M.Kom
NIK: 850719 16 1 1 0268	NIP: 19740414 200312 1 004

Mengetahui
Ketua Program Studi Informatika/ Ilmu Komputer

Drs. Marji, M.T.

NIP: 19670801 199203 1 001

PERNYATAAN ORISINALITAS

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

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

Malang, 3 Januari 2016

Muhammad Guntur Putra Timur

NIM: 115060807111014

KATA PENGANTAR

Alhamdulillah puji syukur kehadirat Allah SWT atas segala limpahan rahmat, karunia, dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi ini. Tidak lupa shalawat dan salam selalu tercurahkan kepada junjungan besar Nabi Muhammad SAW beserta keluarga dan para sahabatnya.

Skripsi ini merupakan bagian dari tugas akhir penulis selama mengikuti perkuliahan dan sebagai salah satu syarat untuk mencapai gelar Sarjana Komputer di Fakultas Ilmu Komputer, Universitas Brawijaya. Judul Skripsi ini adalah **"Pemilihan Jalur Wisata Pulau Lombok Menggunakan Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut"**. Melalui kesempatan ini, Penulis ingin menyampaikan rasa hormat dan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan dan dukungan selama pengerjaan skripsi, diantaranya:

1. Imam Cholissodin,S.Si,. M.Kom selaku dosen pembimbing pertama yang telah meluangkan waktunya untuk membantu dan membimbing penulis dalam pengerjaan tugas akhir ini.
2. Edy Santoso, S.Si., M.Kom selaku dosen pembimbing kedua yang telah meluangkan waktunya untuk membantu dan membimbing penulis dalam pengerjaan tugas akhir ini.
3. Drs. Marji, M.T. dan Issa Arwani, S.Kom., M.Sc. selaku Ketua dan Sekretaris Program Studi Informatika/Ilmu Komputer Fakultas Ilmu Komputer Universitas Brawijaya.
4. Seluruh dosen Informatika/Ilmu Komputer Universitas Brawijaya atas kesediaan membagi ilmunya kepada penulis.
5. Seluruh civitas akademik informatika/ilmu komputer Universitas Brawijaya yang telah banyak membantu dan memberi dukungan selama penulisan skripsi ini.
6. Kedua orang tua Iskandar(Alm) dan Asnisyah, beserta keluarga yang selalu memberikan doa dan dukungan semangat yang tiada henti sehingga skripsi ini dapat terselesaikan.
7. Keluarga besar Kelas H, rekan seperjuangan pengerjaan skripsi Cuntel *(Camarede-Unity-Everlasting)* dan seluruh sahabat Informatika 2011. Terimakasih atas bantuanya selama menempuh studi.
8. Dinas Kebudayaan dan Pariwisata serta Dinas Perhubungan Provinsi NTB yang memberikan data tentang wisata-wisata yang ada di Pulau Lombok.
9. Semua pihak yang telah membantu penulis dalam pengerjaan skripsi yang tidak bisa disebutkan satu persatu.

Penulis menyadari bahwa tugas akhir ini tidak lepas dari kekurangan dan kesalahan. Penulis mengharapkan kritik dan saran dari berbagai pihak demi penyempurnaan tugas akhir ini. Akhir kata, semoga skripsi ini dapat bermanfaat

dan berguna bagi pembaca terutama mahasiswa FILKOM Universitas Brawijaya. Terima kasih.

Malang, 20 Desember 2015

Penulis

mgunturputra@gmail.com

ABSTRAK

Penentuan sebuah jalur pariwisata merupakan salah satu yang diperlukan seseorang untuk berlibur pada suatu tempat. *Travelling Salesman Problem(TSP)* merupakan salah teknik pada algoritma evolusi untuk memecahkan masalah dalam penentuan jalur yang optimal. Namun, pemecahan secara manual untuk masalah TSP membutuhkan banyak waktu terlebih jika data yang berjumlah besar. Algoritma Genetika merupakan salah satu algoritma untuk pemecahan masalah *TSP* karena mudah untuk diimplementasikan. Kelemahan Algoritma Genetika adalah adalah kinerjanya sangat bergantung pada jumlah iterasi yang besar. Selain itu, Algoritma Genetika tidak menjamin hasil yang optimal walaupun iterasi yang digunakan juga besar. Karena pada proses Algoritma Genetika dibutuhkan juga parameter-parameter perhitungan dan proses seleksi. Salah satu pendekatan yang dapat digunakan untuk mengatasi masalah ini adalah menggabungkan dua algoritma yang dapat memecahkan masalah TSP seperti Algoritma Koloni Semut(*ACO*). *ACO* mengeksplorasi ruang mencarian rute dengan cara mencari probabilitas setiap tempat tujuan. Nilai probabilitas digunakan untuk meminimalkan jarak antara sebuah destinasi dengan destinasi lainnya.

Penelitian ini menggunakan Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut (*Hybrid GACO*) untuk pencarian rute pariwisata yang optimal. Proses seleksi yang digunakan adalah seleksi elitism dengan memeriksa semua hasil reproduksi dan memori yang memiliki nilai paling minimum. hasil percobaan menunjukan bahwa algoritma *Hybrid GACO* menghasilkan individu yang lebih baik dibandingkan dengan menggunakan Algoritma Genetika saja.

Kata Kunci : TSP, Algoritma Genetika, Algoritma Koloni Semut, Seleksi Elitism

ABSTRACT

Determining which path to travel on tourism route often needed by someone on vacation. Traveling Salesman Problem (TSP) is one example of evolution algorithm technique used to determine the optimal path. However, the cons of using TSP is that its taking too much time when used to determine travel route using a large amount of data. Genetic Algorithm is another example of algorithm that being used to solve TSP problem because of its easiness on implementation. Genetic Algorithm weakness is that its performance highly dependent on the number of iteration. In addition, Genetic Algorithm cannot guarantee wether using large number of iteration will generate an optimal result. Because some calculation parameters and selection process are required. This paper propose a solution to solve above problem by combining Genetic Algorithm and Ant Colony (ACO). ACO explore the route finding space by calculating the probability of each destination. The probability value then used to minimalize distance between two or more destination.

This research using Hybridation of Genetic Algorithm and Ant Colony (Hybrid GACO) to determine optimal tourism route. Selection process used in this paper is Elitism Selection by checking all reproduction result and memorize the minimum value. The result of this research shows that using Hybrid GACO produce a better individual than using common Genetic Algorithm.

Keyword: *TSP, Genetic Algorithm, Ant Colony Algorithm, Elitism Selection*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE PROGRAM

DAFTAR LAMPIRAN

BAB I PENDAHULUAN

1.1 Latar Belakang

Pulau Lombok merupakan salah satu destinasi pariwisata yang ramai dikunjungi oleh wisatawan domestik maupun mancanegara. Pulau Lombok memiliki setidaknya 40 objek wisata yang menyebar di berbagai kabupaten. Bahkan baru-baru ini Kementerian Pariwisata Indonesia menetapkan Pulau Lombok menjadi tujuan utama destinasi wisata di Indonesia. Wisatawan yang berkunjung di Nusa Tenggara Barat berjumlah 1.629.122 pada tahun 2014 dan sebagian besarnya berwisata di Pulau Lombok [BAP, 2014]. Namun, wisatawan mengalami beberapa masalah dalam menetukan rute tujuan wisata di Pulau Lombok. Hal ini diperkuat dengan sulitnya tempat wisata dengan kendaraan pribadi. Kemudian masih banyaknya transportasi yang tidak layak saat berkunjung ke tempat wisata, kurangnya informasi tentang objek wisata yang dekat satu sama lainnya karena banyaknya pilihan jalur yang berbeda-beda karakteristik dan kondisi jalannya.

Salah satu contoh masalah *Traveling Salesman Problem*(TSP) merupakan pemilihan rute jalur pariwisata. Beberapa jurnal penelitian juga mencoba menyelesaikan masalah TSP dengan metode algortima genetika [GUP-2013] dan algoritma koloni semut [WAN-2014]. Pada penelitian Gupta et al (2013) nilai keoptimalan yang dicapai sangat bergantung pada pengkodean, serta pemilihan *crossover* dan *mutation*. Dan secara umum, algoritma genetika memiliki beberapa kelemahan seperti waktu yang diperlukan untuk mencari nilai maksimum fungsi *fitness* cukup lama, selain itu karena komponen algoritma genetika bersifat random maka menghasilkan solusi yang berbeda setiap dijalankan. Pada penelitian WAN et al (2014), kecepatan dalam mencapai konvergensi cukup lambat karena ruang lingkup yang lebih sempit walaupun dapat menyelesaikan masalah TSP dengan baik. Oleh karena itu, pada penelitian Zukhri dan Paputungan (2013) menggunakan hibridisasi algoritma genetika dan algoritma koloni semut. Hasil kombinasi dari kedua metode tersebut didapatkan solusi dari masalah TSP yang terbaik dan juga didapatkan fitness terbaik dari rute yang akan ditempuh.

Berdasarkan penelitian-penelitian tersebut maka skripsi ini menggunakan judul " Pemilihan Jalur Wisata Pulau Lombok Menggunakan Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut". Metode tersebut digunakan karena jumlah iterasi dan jarak tempuh yang didapatkan lebih kecil dari penelitian yang hanya menggunakan algoritma koloni semut atau algoritma genetik saja. Diharapkan penelitian ini dapat membantu untuk menentukan rute wisata yang tepat, sehingga waktu yang dimiliki dapat digunakan secara optimal.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, dapat dirumuskan masalah-masalah yang akan dibahas sebagai berikut :

1. Bagaimana mengimplementasikan metode Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut untuk menentukan jalur terpendek antar tempat wisata di Pulau Lombok ?
2. Bagaimana nilai *cost* dari hasil implementasi metode Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut untuk pemilihan jalur terpendek di Pulau Lombok?

1.3 Batasan Masalah

Berdasarkan latar belakang dan rumusan masalah yang telah diuraikan, agar permasalahan yang dirumuskan dapat lebih terfokus, maka penelitian ini dibatasi dalam hal :

1. Aplikasi ini hanya dibuat untuk mencari rute terbaik dalam mengunjungi beberapa tempat wisata.
2. Aplikasi ini menggunakan 38 wisata yang berapa di Pulau Lombok.
3. Metode yang digunakan adalah Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut.
4. Pengujian hanya fokus terhadap pengujian metode, tidak pada pengujian hardware.
5. Hasil penerapan metode ini adalah urutan destinasi yang dikunjungi wisatawan agar mendapat jalur terpendek.

1.4 Tujuan

Adapun tujuan yang ingin dicapai dari penelitian ini adalah sebagai berikut:

1. Mengimplementasikan metode Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut untuk menentukan jalur terpendek antar tempat wisata di Pulau Lombok.
2. Mengetahui nilai cost dari hasil implementasi metode Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut untuk pemilihan jalur terpendek di Pulau Lombok.

1.5 Manfaat

Adapun manfaat yang diharapkan dari hasil penelitian ini adalah sebgai berikut :

1. Membantu wisatawan baik domestik maupun mancanegara dalam memilih rute wisata yang optimal di Pulau Lombok secara efektif.
2. Meningkatkan efisiensi waktu wisatawan dalam menentukan rute dan sekaligus waktu kunjungan terhadap tempat-tempat wisata di

Pulau Lombok.

1.6 Sistematika Penulisan

Untuk mencapai tujuan yang diharapkan, maka sistematika penulisan yang disusun dalam laporan tugas akhir ini adalah sebagai berikut :

BAB I Pendahuluan

Berisi tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan, manfaat penelitian dan sistematika penulisan.

BAB II Kajian Pustaka Dan Dasar Teori

Memuat kajian pustaka dan landasan teori yang berhubungan dengan penelitian ini.

BAB III Metode Penelitian Dan Perancangan

Memuat langkah-langkah yang dilakukan dalam penelitian.

BAB IV Implementasi

Berisi tentang penjelasan implementasi metode *Hybrid GA-ACO* untuk optimasi pemilihan jalur wisata di Pulau Lombok

BAB V Pengujian dan Analisis

Berisi tentang pengujian sistem dan analisis data hasil pengujian sistem

BAB VI Kesimpulan Dan Saran

BAB ini berisi hasil kesimpulan yang diperoleh dari pengujian dan saran untuk pengembangan lebih lanjut.

BAB II KAJIAN PUSTAKA DAN DASAR TEORI

Bab ini berisi kajian pustaka dan dasar teori yang berhubungan dengan implementasi pemilihan jalur wisata pulau Lombok menggunakan hibridisasi algoritma genetika dan algoritma koloni semut.

2.1 Kajian Pustaka

Dalam penulisan skripsi ini, telah dilakukan kajian terhadap penelitian-penelitian yang telah dilakukan sebelumnya. Penelitian yang berhubungan dengan optimasi yang menggunakan algoritma genetika dan algoritma koloni semut. Beberapa jurnal penelitian juga mencoba menyelesaikan masalah *TSP* dengan metode algortima genetika [GUP-2013] dan algoritma koloni semut [WAN-2014]. Pada penelitian Gupta *et al* (2013) nilai keoptimalan yang dicapai sangat bergantung pada pengkodean, serta pemilihan *crossover* dan *mutation*. Dan secara umum, algoritma genetika memiliki beberapa kelemahan seperti waktu yang diperlukan untuk mencari nilai maksimum fungsi *fitness* cukup lama, selain itu karena komponen algoritma genetika bersifat random maka menghasilkan solusi yang berbeda setiap dijalankan. Pada penelitian WAN *et al* (2014), kecepatan dalam mencapai konvergensi cukup lambat karena cangkupan ruang lingkup yang lebih sempit walaupun dapat menyelesaikan masalah TSP dengan baik. Oleh karena itu, pada penelitian Zukhri dan Paputungan (2013) menggunakan hibridisasi algoritma genetika dan algoritma koloni semut. Hasil kombinasi dari kedua metode tersebut didapatkan solusi dari masalah TSP yang terbaik dan juga didapatkan *fitness* terbaik dari rute yang akan ditempuh.

Tabel 2.1 Kajian Pustaka

Judul	**Objek**	**Metode**	**Keluaran**
	Masukan dan Parameter	**Proses**	**Hasil Penelitian**
A Hybrid Optimization Algorithm Based on Genetic Algorithm And Ant Colony Optimization [Zukhri dan Paputungan, 2013]	5 buah data set yang terdiri dari 3 data set kota acak yaitu data acak 13 kota, data acak 25 kota dan data acak 35 kota. Dua dataset lainnya adalah 2 data set kota yaitu ulysses16 dan ulysses 22.	*Genetic algorithm* dan *ant colony optimization*	Individu dengan rute terbaik.
	- Jumlah iterasi	- inisialisasi.	Pada pengujian dengan

	- Ukuran populasi - Parameter *crossover* - Parameter *mutation*	- Menentukan kota pertama semut. - Pencarian rute dari tiap semut. - Mencari rute terpendek. - *Pembaruan pheromone matrix* dan populasi berikutnya dengan metode *genetic algorithm.* - Cek kondisi berhenti. - Tampilkan rute terbaik yang diperoleh.	data yang kecil, perbedaan antara metode ini dengan ACO maupun GA tidak terlalu signifikan. Pada data yang besar, metode ini mengungguli ACO dan GA dalam menghasilkan jarak tempuh yang optimal.
Improved Ant Colony Algorithm for Traveling Salesman Problem [WAN-2014]	6 buah data set kota yaitu oliver30, dantzig42, eil51, berlin52, st70 dan pr107	*Ant colony optimization*	Individu dengan rute terpendek dan daftar tabu minimum.
	- Atur jumlah iterasi - Panjang dari sirkuit optimal - Jarak antar kota - Nilai awal *pheromone* dan *pheromone level.*	- Inisialisasi - Membentuk rute dari masing-masing semut. - Memperbarui nilai *phenomenon* lokal. - Melakukan *2-opt local search.* - Perbarui *phenomenon globe dynamic.* - Periksa iterasi proses.	Penelitian dengan menggunakan metode *ant colony optimization* yang telah ditingkatkan ini memiliki performa yang baik dan dapat digunakan untuk menyelesaikan permasalahan optimasi lain yang bersangkutan. Sehingga didapatkan nilai jarak total yang minimum sebagai hasil akhirnya.
Solving Travelling Salesman Problem Using Genetic Algorithm [GUP-13]	Suatu data set dengan jumlah kota pada masing-masing data set 15, 17, 20, 26 dan 42.	Algoritma Genetika	Suatu individu yang memuat suatu rute terpendek pada suatu data set.
	- Jumlah iterasi - Ukuran populasi - Parameter *crossover* - Parameter *mutation*	- Buat populasi - Hitung *fitness* - Menerapkan *tournament selection* ke populasi - Menerapkan *two point crossover* - Menerapkan *interchange mutation*	Walaupun memberikan hasil yang baik, keoptimalan dari metode ini sangat tergantung dari bagaimana masalah itu dikodekan dan juga operator *crossover* dan

		- Pemeriksaan batas iterasi - Tampilkan rute dengan *fitness* terbaik.	*mutation* yang digunakan serta mendapatkan nilai fitness yang terbaik.
Pemilihan Jalur Wisata Pulau Lombok Menggunakan Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut	Data jarak antar 38 tempat wisata di Pulau Lombok	Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut	Individu dengan rute terbaik
	- Jumlah iterasi - Ukuran populasi - Parameter *crossover* - Parameter *mutation*	- Inisialisasi. - Menentukan dilalui oleh semut. - Melakukan pencarian rute dari tiap semut. - Mencari rute terpendek. - Pembaruan *pheromone matrix* dan populasi berikutnya dengan metode *genetic algorithm.* - Cek konndisi berhenti. - Tampilkan rute terbaik yang diperoleh.	

Skripsi ini berisikan tentang optimasi jalur wisata pulau Lombok menggunakan hibridisasi algoritma genetika dan algortima koloni semut. Beberapa dasar teori yang digunakan dalam skripsi ini antara lain ialah algoritma genetika, algoritma koloni semut, *graf* dan Pulau Lombok.

2.2 Pulau Lombok

Pulau Lombok merupakan bagian dari provinsi Nusa Tenggara Barat (NTB). Pulau yang mempunyai luas 4.725 km² ini beribukota di Mataram. Lombok memiliki iklim tropis dengan suhu rata-rata 29˚C - 34˚C, selain itu memiliki dua musim penghujan (Oktober – April) dan kemarau (May – September). Waktu terbaik untuk berkunjung ke Pulau Lombok ialah pada musim hujan karena tanah agak basah dan hijau tidak seperti pada musim kemarau.

Pulau Lombok terdiri dari beberapa bagian, yaitu 4 kabupaten dan 1 kota. Beberapa bagian dari Pulau Lombok seperti kabupaten Lombok Utara, kabupaten Lombok Barat, kabupaten Lombok Tengah, kabupaten Lombok Timur dan kota Mataram. Jumlah penduduk pulau lombok sekitar 3,25 juta jiwa menurut densus pada tahun 2013 serta dominan berasal dari suku Sasak dan beragam Islam.

2.2.1 Wisata Pulau Lombok

Pulau Lombok merupakan salah satu tujuan wisata utama bagi wisatawan domestik maupun internasional yang berkunjung ke Indonesia. Di Lombok wisata yang disediakan cukup lengkap seperti, pantai, gunung, sejarah dan masih banyak lagi. Seperti wisata dalam kota, wisatawan dapat menemui tentang sejarah serta pasar seni. Selain itu juga wisatawan dapat menemui Taman Mayura yang berisikan tentang arsitektur kuno sejarah Pulau Lombok.

Salah satu wisata andalan yang terdapat di Pulau lombok ialah wisata Pantai. Pulau lombok mempunyai banyak sekali wisata pantai, beberapa yang terkenal ialah pantai Senggigi dan pantai Kuta. Pantai Kuta seperti pada Gambar 2.1 mempunyai ciri khas pasir putih dan besar seperti merica, selain itu pantai kuta juga memiliki bibir pantai yang yang luas dan nyaman.

Gambar 2.1 Pantai Kuta

Salah satu yang menjadi tujuan wisata datang ke pantai kuta ialah banyaknya *spot-spot* disekitar pantai kuta yang memiliki ombak besar sehingga banyak disukai oleh para peselancar. Selain memiliki wisata pantai yang menarik, Pulau lombok juga memiliki beberapa wisata air terjun. Salah satu air terjun yang terkenal di pulau lombok ialah air terjun benang kelambu dapat dilihat pada Gambar 2.2. Ciri khas dari air terjun benang kelambu ialah karena pesona air terjun dengan ketinggian 30 meter ini menyerupai tirai atau bambu. Selain itu air turun berasal dari 6 buah titik yang seolah - olah berasal dari rimbunnya tanaman, air terjun ini mempunyai tiga buah tingkatan.

Gambar 2.2 Air Terjun Benang Kelambu

Selain itu juga Pulau Lombok juga memiliki banyak pulau – pulau kecil disekitarnya. Pulau – pulau kecil biasa disebut gili, gili inilah yang biasanya tujuan utama wisatan yang datang ke pulau lombok. Satu gili yang terkenal di pulau lombok adalah gili trawangan. Gili trawangan dapat dilihat pada gambar Gambar 2.3 memiliki banyak wisata yang disediakan. Gili trawangan merupakan salah satu tempat untuk *snorkeling* maupun *diving*, selain itu juga disini disediakan tempat penginapan, alat transportasi tradisional dan *jetski* untuk memanjakan wisatawan yang datang.

Gambar 2.3 Gili Trawangan

2.3 Graf

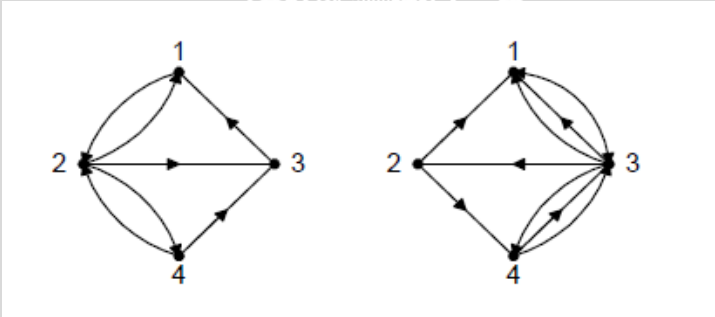
Secara umum, graf adalah suatu diagram yang memuat informasi tertentu jika diinterpretasikan secara tepat. Dalam kehidupan sehari – hari, graf digunakan untuk menggambarkan berbagai macam struktur yang ada. Tujuannya adalah sebagai visualisasi objek – objek agar lebih mudah dimengerti. Teori ini pertama kali diperkenalkan oleh Leonhard Euler pada abad ke 18. Awalnya Euler ingin memecahkan teka – teki yang bernama 'Jembatan Konigsberd' yaitu dengan mencari suatu lintasan dengan melewati ketujuh jembatan yang menghubungkan dua pulau dan satu sungai.

Suatu graf G terdiri dari 2 himpunan yang berhingga, yaitu himpunan titik-titik tidak kosong (symbol *V(G)*) dan himpunan garis – garis (symbol *E(G)*). Setiap garis akan berhubungan dengan satu atau dua titik ujung yang disebut titik ujung, sedangkan garis yang berhubungan dengan satu titik ujung disebut dengan loop. Dan dua garis berbeda yang menghubungkan titik yang sama disebut garis parallel. Dua titik bisa dikatakan berhubungan jika ada garis yang menghubungkan keduanya. Sedangkan titik yang tidak memiliki garis yang berhubungan dengannya disebut titik terasing (*isolating point*). Graf yang tidak memiliki titik (tidak memiliki garis) disebut graf kosong.

2.3.1 Jenis – Jenis Graf

Suatu graf dapat digolongkan berdasar adanya tidaknya suatu gelang, berdasarkan simpulnya, berdasarkan orientasi arah pada sisi, dan graf dilihat dari ada tidaknya bobot yang terdapat pada sisinya.

1. Graf berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, maka graf digolongkan menjadi dua jenis:
    - Graf sederhana : graf yang tidak mempunyai gelang maupun sisi ganda.
    - Graf tak-sederhana : graf yang mengandung ruas ganda atau gelang.
2. Graf yang berdasarkan jumlah simpul, maka secara umum graf dapat digolongkan secara dua jenis:
    - Graf berhingga : graf yang jumlah simpulnya n atau berhingga.
    - Graf tak berhingga : graf yang jumlah simpulnya ,n, tidak berhingga banyaknya.
3. Graf berdasarkan orientasi arah pada sisi, maka secara umum dibedakan menjadi dua jenis yaitu :
    1. Graf tak berarah : graf yang sisinya tidak mempunyai orientasi arah
    2. Graf berarah : graf yang setiap sisinya diberikan orientasi arah. Graf yang berorientasi arah dapat dilihat pada Gambar 2.4.



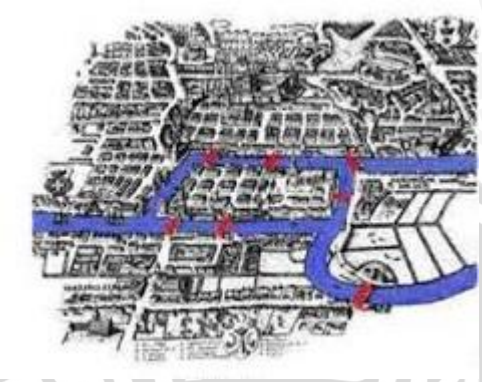
Gambar 2.4 Graf berarah

4. Graf yang dilihat berdasarkan ada atau tidaknya bobot pada sisi atau garis yang menghubungkan antar titik didalam graf tersebut dapat digolongkan menjadi 2 jenis graf yaitu :
    - Graf tak-berbobot : graf yang pada sisi atau garis yang menghubungkan antara titik-titiknya tidak meiliki label taau bobot khusus.
    - Graf berbobot : suatu graf yang pada sisi atau garis yang menghubungkan antara titik-titiknya memiliki label atau bobot khusus.

2.3.2 Sirkuit Euler dan Sirkuit Hamilton

Sirkuit Euler adalah sirkuit dimana setiap titik dalam suatu graf muncul paling sedikit sekali dan setiap garis dalam graf muncul tepat sekali. Teori ini dibuat untuk mengenang ahli matematika Leonhard Euler yang berhasil memperkenalkan graf untuk memecahkan masalah 7 jembatan Konigsberg pada tahun 1736. Kota Konigsberg dibangun pada pertemuan 2 cabang suangi Pregel.

Kota tersebut terdiri dari sebuah pulau di tengah –tengah dan 7 jembatan yang mengelilinginya dapat dilihat pada Gambar 2.5.



Gambar 2.5 Jembatan Konigsberg

Suatu sirkuit disebut sirkuit Hamilton jika ada sirkuit yang mengunjungi setiap titiknya tepat satu kali kecuali titik awal yang sama dengan titik akhirnya. Dalam sirkuit Hamilton semua titik harus dikunjungi tept satu kali dan tidak harus melalui semua garisnya. Dalam sirkuit Euler, yang dipeningkan adalah garisnya. Sebaliknya dalam sirkuit Hamilton, yang dipentingkan adalah kunjungan pada titiknya. Salah satu contoh permasalahan dengan sirkuit Hamilton ialah *travelling salesman problem* (*TSP*).

2.4 Algoritma Genetika

Algoritma Genetika pertama kali diperkenalkan oleh John Holland dari Universitas Michigan tahun 1975. Algoritma genetika adalah cabang dari algoritma evolusi yang merupakan metode *adaptive* dan bisa digunakan untuk memecahkan suatu pencarian nilai dalam masalah optimasi. Algoritma ini mengadaptasi istilah dalam proses genetik yang ada dalam makhluk hidup yaitu seleksi alam.

Adapun 5 komponen penting yang terkandung dalam algoritma genetik, yaitu:

1. Representasi genetik sebagai solusi dari sebuah masalah.
2. Cara untuk membangkitkan populasi awal.
3. Fungsi untuk mengevaluasi solusi dengan nilai *fitness* yang dimiliki masing-masing individu.
4. Beberapa operator genetika yang menghasilkan keturunan /*offspring*.

Nilai untuk beberapa parameter yang digunakan dalam algortima genetika seperti ukuran dan nilai probabilitas yang digunakan dalam operator genetika,dll

2.4.1 Struktur Algoritma Genetika

Pada algoritma genetika, populasi merupakan istilah yang digunakan untuk melakukan teknik pencarian sekaligus atas sejumlah kemungkinan solusi. Solusi yang terdapat dalam satu populasi disebut kromosom atau individu. Populasi pertama dibangun secara acak, sedangkan populasi selanjutnya merupakan hasil evolusi kromosom-kromosom melalui iterasi yang disebut dengan generasi. Pada setiap generasi yang dilakukan, kromosom akan melalui proses evaluasi dengan menggunakan nilai ukur yang disebut *fitness*. Nilai *fitness* ini akan menunjukan kualitas individu dalam populasi.

Individu berikutnya dalam sebuah populasi merupakan offspring (anak) yang terbentuk dari *crossover*. Dalam proses *crossover* dibutuhkan dua operator inidividu yang disebut *parent*, sedangkan untuk proses mutasi hanya dibutuhkan sebuah indiividu untuk menghasilkan inidividu baru melalui modifikasi kromosomnya . Pada populasi, generasi yang baru dibentuk dari hasil seleksi nilai fitness untuk semua individu, termasuk hasil proses *crossover* dan mutasi yang dianggap baik. Serta menolak semua individu lainnya sehingga didapatkan ukuran populasi yang selalu konstan.

Karakteristik pencarian solusi dalam algoritma genetika adalah sebagai berikut :

1. Algoritma Genetika bekerja berdasar pada metode pengkodean untuk sekumpulan parameter yang berkaitan.
2. Algoritma Genetika menggunakan fungsi obyektif (*fitness*).
3. Algoritma Genetika menggunakan aturan transisi dengan probabilitas, bukan dengan aturan deterministik.
4. Algoritma Genetika melakukan pencarian dari sejumlah titik yang tergabung dalam satuan populasi, bukan pada titik pencarian tunggal.

Hal yang perlu diingat dari algoritma genetika adalah bahwa algoritma genetika merupakan algoritma yang dikembangkan dari proses pencarian solusi dengan pencarian acak, hal tersebut dibuktikan dari pembangkitan populasi awal yang dipilih secara acak. Selanjutnya pencarian dilakukan berdasarkan proses-proses teori genetika yang memperhatikan bagaimana memperoleh individu yang lebih baik, sehingga dalam proses evolusi diharapkan mendapat individu yang terbaik.

2.4.2 Representasi Kromosom

Hal paling penting pada penggunaan algoritma genetika untuk menyelesaikan suatu masalah adalah bagaimana mengkodekan permasalah ke dalam kromosom. Hal ini disebabkan karena representasi kromosom dalam setiap permasalahan berbeda-beda dan tidak semua model representasi cocok untuk setiap permasalahan. Representasi kromosom sendiri suatu penyandian

gen dalam kromosom. Representasi gen ini sendiri bisa dalam bentuk *string bit*, pohon, *array*, bilangan *real*, dan lain-lain.

Menurut Guz Eiben dan Jim Smith , jenis representasi kromosom adalah sebgai berikut berikut:

A. Representasi Biner

Representasi ini merupakan bentuk paling sederhana dan umum digunakan untuk menyelesaikan masalah yang tidak terlalu kompleks. Dalam representasi ini, gen hanya bernilai 0 atau 1 sehingga operator *crossover* dan mutasi yang digunakanpun akan sederhana.

Tabel 2.2 Kromosom Biner

Kromosom X	101100011100011100110
Kromosom Y	001101010111110000000

Pada Tabel 2.2 , dapat dijelaskan bahwa terdapat dua kromosom yaitu Kromosom X dan Kromosom Y dimana gennya bernialai 1 atau 0.

B. Representasi *Integer*

Tidak semua permasalahan dapat dikodekan dengan bentuk biner, misalnya masalah yang variabel penyusunnya adalah integer. Pada representasi ini, setiap gen bernilai bulat (*integer*) yang merepresentasikan jenis atau kuantitas objek. Dalam Tabel 2.3 dijelaskan bahwa terdpaat dua kromosom yaitu Kromosom X dan Kromosom Y dimana gen-gennya bernilai bulat.

Tabel 2.3 Kromosom Integer

Kromosom X	3 2 4 7 5 11 9 27 1
Kromosom Y	5 9 1 4 27 6 7 14 2

C. Representasi *Real*

Representsi real bisa digunakan ketika representasi biner dan integer tidak bisa mencapai ketelitian yang diinginkan. Dalam representasi *real*, setiap gen bisa bernialai real dalam interval [0,1]. Penggunaan representasi *real* ditunjukan pada Tabel 2.4 berikut.

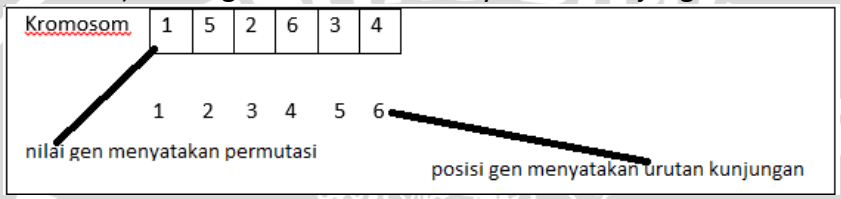
Tabel 2.4 Kromosom Real

Kromosom X	0,234 0,456 1,0 0,541

Kromosom Y	0,165 0,269 0,876 1,0

D. Representasi Permutasi

Dalam masalah tertentu, seperti *Travelling Salesman Problem* (TSP) misalnya, representasi biner, integer maupun *real* tidak bisa digunakan karena dalam solusi permasalahn ini yang dicari adalah bagaimana menemukan kunjungan lokasi dan yang total nilainya paling optimal. Nialai disini bisa berupa jarak, kenyamanan, baiya dan sebagainya. Dalam membangun representasi permutasi ini, ada hal-hal yang harus diperhatikan, yaitu satu kromosom harus menyatakan satu solusi selain itu juga posisi dan nilai gen. Posisi gen (indeks pada kromosom) bisa digunakan untuk menyatakan kunjungan lokasi.



Gambar 2.6 Contoh Permutasi

Untuk permasalahan kombinasi menu makanan ini akan dicari bobot yang optimal dalam gen. bobot pada tiap jenis makaanan bernialai *real*, sehingga representasi kromosom yang digunakan adalah rerpresentasi *real* untuk mengkodekan susunan bibit ke dalam kromosom.

2.4.3 Nilai *Cost*

Nilai *cost* adalah nilai yang menyatakan baik dan tidaknya suatu solusi individu. Nilai inilah yang dijadikan acuan untuk mencapai nilai optimal dalam algoritma genetika. Dalam *Travelling Salesman Problem* (TSP) misalnya, TSP bertujuan untuk meminimalkan jarak, maka nilai fitnessnya adalah inversi dari jarak.

Menurut Suyanto, fungsi *cost* (f) bisa berupa fungsi yang memaksimalkan dan meminimalkan fungsi *(h)* dimana keduanya sama dan bersifat kebalikan. Dalam permasalahn yang memaksimalkan fungsi f (maka akan meminimalkan fungsi h), fungsi ini tidak dapat langsung sebgai fungsi *cost* karena fungsi ini memiliki arti bahwa yang memiliki cost yang tinggi adalah individu yang mampu bertahan hidup dalm sebuah populasi. Secara matematis, fungsi maksimasi tersebut dituliskan dalamm persamaan 2.1 berikut :

$$f = \frac{1}{(h)} \qquad (2.1)$$

2.4.4 Perkawinan Silang/*Crossover*

Proses perkawinan silang (*crossover*) bertujuan menghasilkan anak/keturunan. Dalam *crossover*, diperlukan dua kromosom terpilih sebagai induk atau *parent*. Dalam proses *crossover* ini, akan dihasilkan sebuah kromosom anak. Kromososm anak yang dihasilkan merupakan kombinasi gen dari dua kromosom induk. Dalam *crossover*, terdapat nilai probabilitas *crossover* (pc) yang akan menentukan frekuensi dari *crossover* yang terjadi dalam sebuah populasi. Semakin tinggi nilai *P*c maka akan semakin besar kemungkinan dilakukan kawin silang.

Dalam proses crossover jumlah populasi sangat mempengaruhi hasil dari proses *crossover*. Jumlah populasi yang sangat kecil akan berakibat buruk, hal itu akan menyebabkan suatu kromomsom dengan gen-gen yang mengarah pada solusi akan sangat cepat menyebar ke kromosom-kromsom yang lain. Sehingga solusi yang ditwarkan untuk mengatasi masalah tersebut digunakan suatu aturan bahwa proses *crossover* hanya dapat dilakukan dengan suatu probabilitas tertentu, artinya *crossover* hanya dapat dilakukan dengan membangkitkan bilangan random [0..1] dan nilai bangkitan random adalah kurang dari nilai *Pc* yang telah ditentukan. Pada umumnya, nilai *Pc* adalah mendekati 1.

Memilih dua buah kromosom sebagai induk:

1. Memilih secara acak posisi dalam kromosom biasa disebut *crossover point*, sehingga masing-masing kromosom induk terbagi menjadi dua segmen.
2. Lakukan pertukaran antar bagian induk untuk menghasilkan kromosom anak.

Dalam penggunaannya, tenik *crossover* akan dihasilkan dengan teknik pengkodean kromosom. Metode *crossover* pada *floating point representation* diantaranya adalah *simple arithmetic crossover, single arithmetical crossover,* dan *arithmetical crossover*. Berikut akan dijelaskan mengenai *one-cut point crossover*[ZEO-12].

Crossover dua buah parent yang dilakukan dengan menukar sebesar kebagian satu sama lain untuk menghasilkan anak (*child*), dimana sebelumnya akan ditentukan titik potong. Titik potong untuk membagi segmen ditentukan oleh pengguna atau bebas. Berikut langkah-langkah proses *crossover* menggunakan metode one-cut point.

1. Pilih dua buah induk untuk dijadikan parent dalam proses *crossover*.
2. Pilih dua gen pertama yang terdapat pada induk 1 untuk dijadikan 2 gen pertama pada induk baru.
3. Lakukan pengecekan pada induk 2, setelah itu pilih gen pada induk 2 untuk dijadikan gen ketiga sampai ke-*n*.

Tabel 2.5 *Crossover*

p1	1	4	5	2	3
p2	2	3	1	5	4
c1	1	4	2	3	5

2.4.5 Mutasi

Mutasi merupakan bentuk operator genetika yang menukar nilai gen dengan nilai gen yang lain, dalam *mutasi* biner misalnya, gen yang bernilai 0 menjadi bernilai 1. Proses ini dilakukan secara acak pada posisi gen tertentu pada individu-individu yang terpilih untuk dimutasi. Selain *mutasi* yang bernialai biner, terdapat juga *mutasi* yang bernialai *real*. Banyaknya individu yang mengakami mutasi ditentukan oleh besarnya probabilitas mutasi.

Dalam representasi integer, ada beberapa teknik *mutasi,* diantaranya adalah *uniform tansform*. Dalam *uniform tansform,* akan dibangkitkan nilai secara random dalam tiap gen sepanjang kromosom, dimana sebelumnya sudah ditentukan *Pm*, yang menjadi batasan, lalu nilai random yang kurang dari *Pm* yang ditentukan tadi akan dimutasi. Dalam representasi real atau *floating point* terdapat metode mutasi *uniform mutation*, teknik ini dianalogikan sama dengan *swapping mutation* seperti pada representasi integer[YAY - 2012].

Adapun langkah-langkah dari metode *mutasi* dalam penelitian sebelumnya adalah sebgai berikut:

1. Bangkitkan angka random antara 0 sampai 1 sejumlah gen (panjang kromosom dikalikan dengan jumlah kromosom).
2. Bandingkan angka random dengan gen-gen dalam kromosom
3. Jika bilangan acak yang dihasilkan 0, maka gen yang sesuai akan ditukarkan dengan gen setelahnya.

Tabel 2.6 Ilustrasi Mutasi

angka random	1	1	0	1	1
p1	1	2	3	4	5
m1	1	2	4	3	5

2.4.6 Seleksi

Proses seleksi akan melakukan pemilihan terhadap individu yang akan diikutkan dalam proses reproduksi. Pada proses seleksi, keanekaragaman populasi memegang peranan penting dalam proses seleksi yaitu daerah sampling, probabilitas seleksi, dan mekanisme seleksi.

Perlu diingat, sebelum dilakukan seleksi jumlah anggota populasi ditambah dengan hasil offspring dari proses genetik *crossover* dan *mutasi*. Hasil operasi genetik dan populasi semula selanjutnya di seleksi dengan metode

tertentu untuk diambil sejumlah n anggota populasi yang terbaik sesuai dengan jumlah populasi awal.

Beberapa metode dalam seleksi adalah *Tournament Selection, Roulette Wheel Selection, Rank Based Fitness Selection, Trancation Selection, Seleksi Lokal dan elitism selection .* Berikut akan dijelaskan tentang *Elitism Selection*.

Proses seleksi elitism adalah pengopian satu atau lebih individu yang bernilai *cost* tinggi agar tidak hilang pada proses evolusi selanjutnya. Maka tidak ada jaminan bahwa suatu nilai yang memiliki *cost* tinggi akan selalu terpilih. Jika terpilih maka tidak ada jaminan bahwa individu itu tidak akan rusak karena akan melalui proses *crossover*.

2.5 Algoritma Koloni Semut

Algoritma koloni semut merupakan perkembangan metode dalam *Artificial Intelligence* yang mendapat inspirasi dari alam. Algoritma ini diperkenalkan oleh Moyson dan Manderick pada tahun 1996, setelah itu dikembangkan oleh Marco Dorico. Algoritma ini mendapat inspirasi dari perilaku semut yang mencari makanan dari sarangnya.

2.5.1 Perilaku Semut

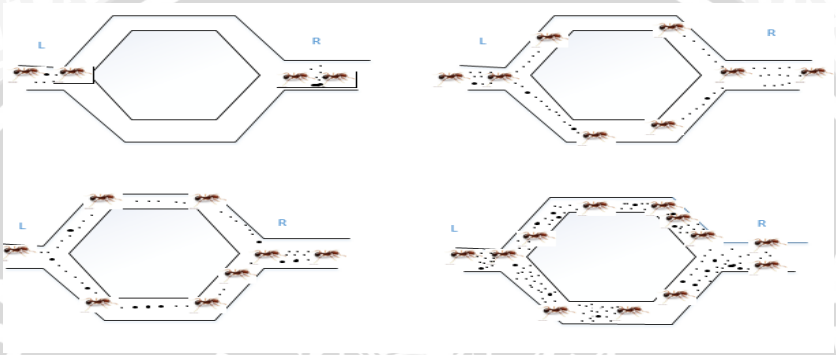
Semut merupakan serangga yang hidup secara berkoloni, dan mampu berkerja sama secara efektif dalam melakukan pekerjaan. Serangga mempunyai alat indera yang sangat kuat dalam menghadapi lingkungan yang kompleks. Alat indera ini digunakan semut dalam mencari makanan dalam kelompoknya. Semut mencari makanan dengan melepaskan feromon sebagai alat penanda dalam jalur yang dilewatinya. Perilaku semut yang melepaskan feromon ini sangat berguna bagi kelompoknya untuk mendapatkan jalur optimal dalam menemukan tempat makanannya. Feromon adalah zat kimia yang berasal dari kelenjar endoktrin dan digunakan oleh makhluk hidup untuk mengenali sesame jenis, individu lain, kelompok dan merupakan alat bantu reproduksi.

Proses meninggalkan feromon ini disebut *stigmergy*, yaitu proses memodifikasi lingkungan untuk mengingat jalan kembali ke sarang dan merupakan alat untuk berkomunikasi bagi sesame semut. Adapun tahapan proses yang dilakukan semut untuk mendapatkan jalur optimal yang dilaluinya ialah sebagai berikut :

1. Setiap semut akan bepergian secara acak dalam menemukan tempat mencari makan.
2. Ketika mendapat tempat makanan, setiap semut kembali ke sarangnya dengan memberikan jejak-jejak feromon pada setiap jalur yang dilaluinya.
3. Jika semut-semut bepergian lagi, maka semut tidak akan bepergian secara acak lagi, melainkan mengikuti jejak feromon tersebut.

4. Jika pada akhirnya semut menemukan makanan, maka semut tersebut akan kembali ke sarangnya dan menguatkan jejak feromon yang dilewatinya.
5. Feromon yang berkonsentrasi tinggi akhirnya akan menarik semut-semut lainnya untuk berpindah jalur, yang akhirnya akan semut tersebut mandapatkan jalur optimal dalam menemukan tempat makanan.

Contoh gambar perilaku semut dalam melalui jalur-jalur yang dilaluinya dalam mencari makanan dapat dilihat pada Gambar 2.7.



Gambar 2.7 Perjalanan Semut Mencari Makanan

2.5.2 Algoritma Semut dalam Travelling Salesman Problem

Secara umum, cara kerja semut dalam menentukan jalur optimal untuk mencari makanan adalah sebagai berikut. Pertama, setiap semut akan bepergian secara acak dalam mencari tempat makan. Selanjutnya setiap semut akan berulangkali mengunjungi kota-kota yang akan dilaluinya (melalukan penjalanan kesetiap kota untuk dapat melakukan tur yang lengkap). Semut akan lebih suka mengunjungi kota terdekat dengan kota awalnya atau dengan tingkat feromon yang lebih tinggi. Setiap semut memiliki memori untuk menyimpan daftar kota yng telah dilaluinya, sehingga mencegah semut mengunjungi kota untuk kedua kalinya.

Setelah setiap semut melakukan tur secara penuh, maka aturan pembaruan feromon akan dilakukan. Penguapan feromon akan dilakukan disetiap sisi yang dilewatinya, setelah itu setiap semut akan melakukan perhitungan panjang jalur yang telah dilewatinya. Semakin pendek sebuah tur yang dilakukan oleh semut, maka semakin besar pula feromon yang diletakan pada setiap sisinya. Hal ini menyebabkan sisi feromon yang dilewatinya akan lebih diminati oleh semut lainnya, sedangkan sisi yang feromon yang lebih sedikit akan kurang diminati oleh semut-semut lainnya dalam menemukan makanan.

2.5.3 Inisialisasi

Dalam algoritma koloni semut langkah awal yang diperlukan melakukan inisialisasi parameter-parameter yang diperluakan dalam perhitunganya. Seperti

halnya menghitung feromon awal pada algoritma koloni semut, selain itu diperluka juga nilai visibiltas dari jarak-jarak setiap kota yang akan dikunjungi semut.

Inisialisasi feromon awal pada algoritma koloni semut ditentukan dengan persamaan sebagai berikut.

$$\tau_0 = \frac{k}{C_{greedy}} \quad (2.4)$$

Keterangan :

k = jumlah semut

C_{greedy} = panjang tur yang dihasilkan melalui algoritma greedy

τo = feromon awal

Sedangkan nilai visibilitas dari jarak antar kota dinyatakan dengan n_{ij}. Nilai n_{ij} dihitung berdasarkan persamaan sebagai berikut :

$$n_{ij} = \frac{1}{d_{ij}} \quad (2.5)$$

Keterangan :

n_{ij} = invers jarak dari kota *i* ke kota *j*

d_{ij} = jarak dari kota *i* ke kota *j*

2.5.4 Pembangunan Solusi

Dalam tahapan ini seekor semut akan menentukan kota tujuan selanjutnya dengan secara acak. Kota tujuan selanjutnya akan ditentukan berdasarkan persamaan berikut :

Keterangan :

$$P_{ij,k,}(t) = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[n_{ij}]^{\beta}}{\sum[\tau_{ij}]^{\alpha}[n_{ij}]^{\beta}}, j \in j_{ik} \\ 0, j \notin j_{ik} \end{cases} \quad (2.6)$$

$P_{ij,k,}(t)$ = peluang semut ke-k untuk mengunjungi kota j ke kota I pada iterasi ke-t

n_{ij} = invers jarak kota *i* ke kota *j*

j_{ik} = kumpulan kota yang akan dikunjungi oleh semut yang berada pada kota i

α = parameter yang mengontrol feromon ($\alpha \geq 0$)

β = parameter yang mengontrol feromon ($\beta \geq 0$)

$\tau_{ij}, (t)$ = inisialisasi feromon antara kota *i* dan kota j pada iterasi ke-t

2.5.5 Pembaruan Feromon Global

Pembaruan feromon dilakukan setelah semua semut selesai melakukan turnya. Feromon yang diperbarui dengan secara menyeluruh untuk menghitung perubahan nilai feromon antar kota. penguapan feromon pada seluruh sisi TSP dengan menggunakan persamaan sebagai berikut :

$$\tau_{ij(baru)} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau_{ij,k} \quad (2.7)$$

Keterangan :

$\tau_{ij\ (baru)}$ = konsentrasi feromon yang baru

p = parameter laju penguapan feromon ($0 < p \leq 1$)

$\Delta\ \tau_{ij,k}$ = jumlah feromon yang ditambahkan oleh semut k

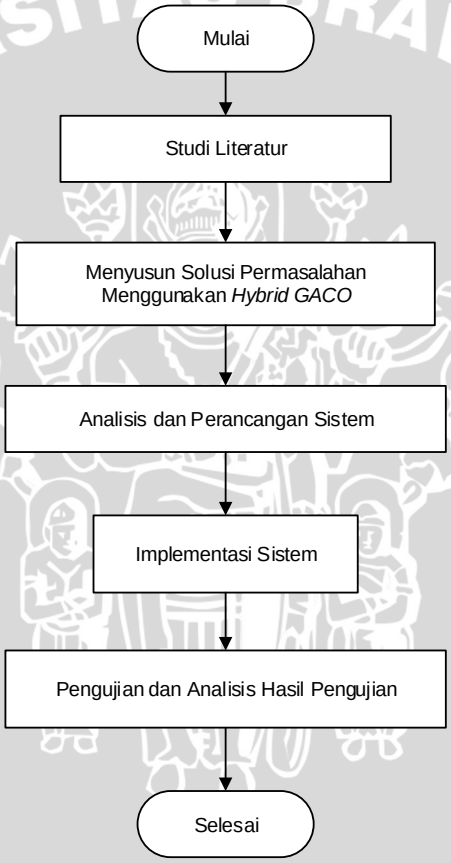
τ_{ij} = inisialisai feromon antara kota *i* dan kota *j*

BAB III METODOLOGI PENELITIAN

Pada bab ini akan membahas beberapa hal, yaitu tahapan penelitian, kebutuhan sistem, formulasi permasalahan, dan siklus penyelesaian masalah menggunakan algoritma genetika dan algoritma koloni semut.

3.1 Tahapan Penelitian

Pada tahap ini akan dibahas mengenai rancangan dan langkah-langkah yang digunakan dalam membuat sistem "Pemilihan Jalur Wisata Pulau Lombok Menggunakan Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut" dapat dilihat pada Gambar 3.1.



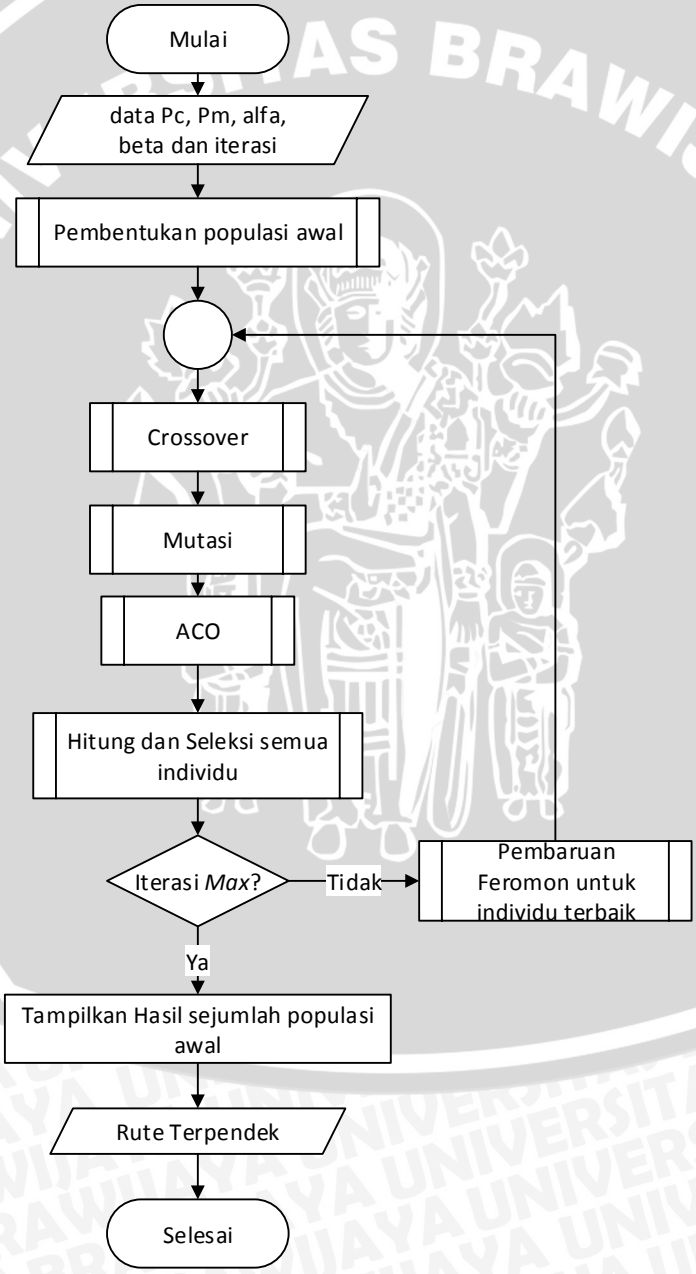
Gambar 3.1 Flowchat studi literatur

3.2 Kebutuhan Sistem

Kebutuhan sistem merupakan tahapan untuk mengalisis tentang hal-hal yang dibutuhkan sistem dalam menjalankan sistem, sehingga berjalan secara efektif dan optimal.

3.2.1 Deskripsi Umum Sistem

Bahasa pemrograman *C#* akan digunakan untuk membuat sistem ini. Sistem ini akan dapat berjalan dengan inputan yang telah diberikan pengguna untuk menentukan parameter hibridisasi algoritma genetika dan algoritma koloni semut (*GA-ACO*) yaitu : jumlah iterasi maksismum untuk menentukan proses berhentinya sistem, jumlah populasi yang ditentukan, dan parameter *crossover* serta *mutasi* untuk proses evaluasi sistem dalam menentukan jalur yang optimal. Berikut alur pengerjaan menggunakan algoritma *GACO* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Flowchart untuk proses Hybrid GACO

3.2.2 Spesifikasi Kebutuhan Sistem

Agar sistem dalam penelitian ini dapat berjalan dibutuhkan perangkat keras dan perangkat lunak. Perangkat lunak yang dibutuhkan antara lain : sistem operasi Windows 8.1 64 bit, Microsoft Office 2013, Microsoft Excel 2013, serta Visual Studio 2013. Adapun perangkat yang dibutuhkan antara lain adalah processor Intel Quad Core i7–4700HQ 2.4Ghz, RAM 8 GB, Hardisk 1 TB, dan monitor 14" dan peralatan masukan seperti keyboard dan mouse.

3.2.2 Data Yang Digunakan

Data yang digunakan dalam penelitian ini adalah data tabel yang berisikan data-data objek wisata penting yang berda di pulau lombok. Data ini juga berisikan jarak tempuh yang dibutuhkan dalam mengunjungi objek wisata satu dengan wisata yang lainnya. Jumlah objek wisata yang digunakan dalam penelitian ini adalah 34 objek dan 4 tempat penting di pulau lombok. Adapun 4 tempat itu antara lain : bandara, pelabuhan Kayangan, pelabuhan Lembar dan kota Mataram. Data objek-objek wisata ini didapatkan dari Dinas Kebudayaan dan Pariwisata Nusa Tenggara Barat. Sedangkan data jarak tempuh objek wisata didapatkan dari Dinas Pekerjaan Umum Nusa Tenggara Barat dan bantuan dari Google Map. Data selengkapnya dapat dilihat Lampiran 1.

3.3 Perhitungan Manual Metode Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut (GA-ACO)

Pada bab ini akan dibahas mengenai contoh perhitungan manual, langkah perhitungan manual dengan menggunakan metode Hibridisasi Algoritma Genetika dan Algoritma Koloni Semut dan juga hasil perhitungannya. Pada contoh perhitungan ini digunakan data jarak antar 5 objek wisata yang berupa sebuah matrix hubung yang dapat dilihat pada Tabel 3.1.

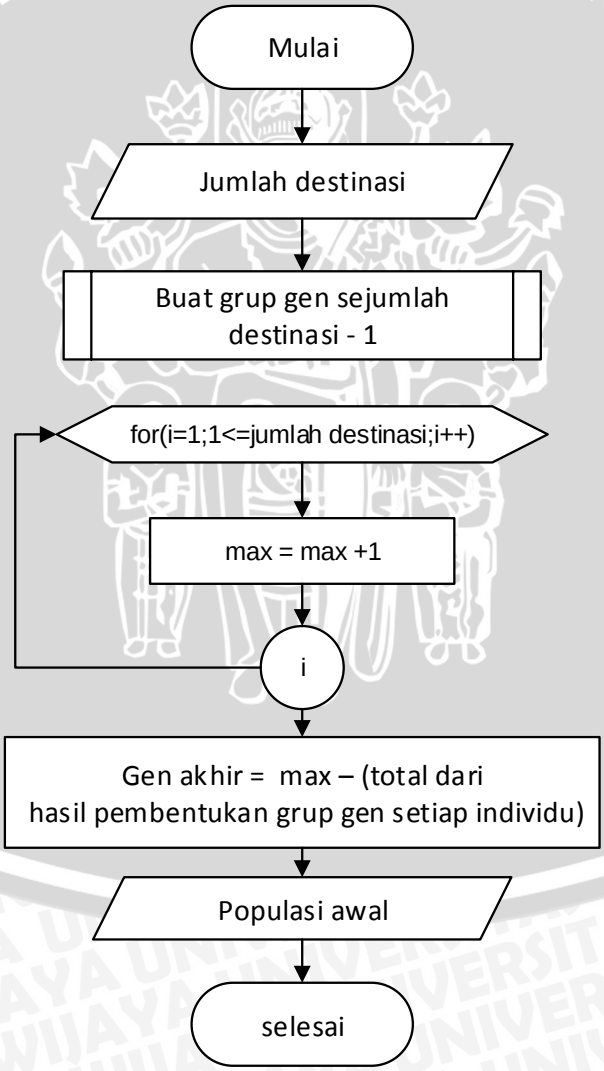
Tabel 3.1 Data jarak antar 5 objek wisata

	BANDARA	LEMBAR	LABUHAN LOMBOK	MATARAM	SEKARBELA
BANDARA (BN)	0	39.1	80.6	36.9	35.1
LEMBAR (L)	39.1	0	101	27	24.9
LABUHAN LOMBOK (LL)	80.6	101	0	84.9	85.2
MATARAM (M)	36.9	27	84.9	0	4
SEKARBELA (SK)	35.1	24.9	85.2	4	0

Dari Tabel 3.1 merupakan data 5 destinasi wisata yang ada dipulau Lombok. Dari data tersebut akan digunakan sebagai bahan perhitungan manual dalam menentukan jalur optimal untuk mengunjungi suatu tempat wisata. Berikut langkah-langkah perhitungan yang digunakan dalam menetukan rute jalur wisata menggunakan hibridisasi algoritma genetika dan algoritma koloni semut:

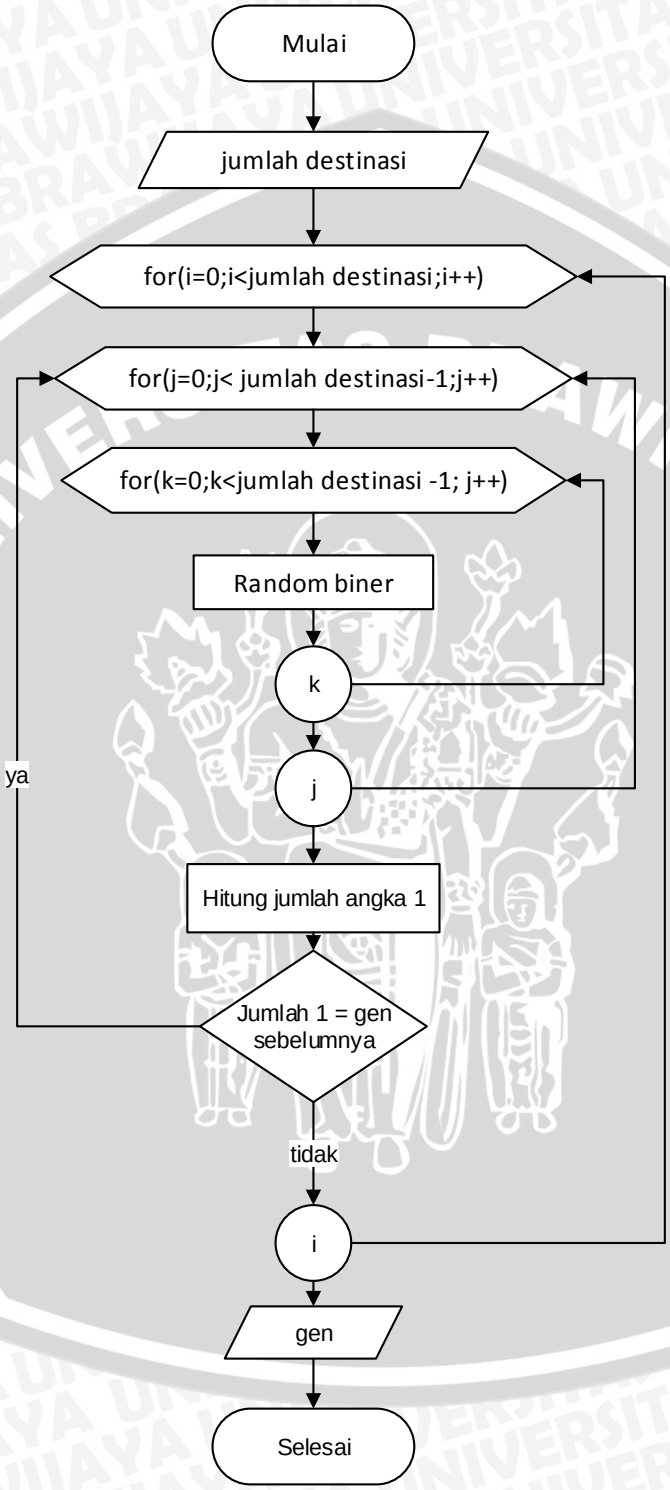
3.3.1 Representasi Kromosom

Dalam kasus hibridisasi ini tujuannya adalah menggabungkan dua metode dasar, yaitu algoritma genetik dan algortima koloni semut. Sehingga perancangan untuk representasi kromosomnya harus dapat digunakan oleh kedua metode tersebut. Sehingga digunakan representasi menggunakan angka biner. Jumlah gen yang terdapat didalam sebuah kromosom dibagi menjadi grup-grup gen. Sehingga langkah selanjutnya adalah menentukan nilai gen dalam sebuah kromosom. Nilai gen yang didapat dalam sebuah grup gen didapat dari perhitungan jumlah angka satu yang didapat dari hasil random. Tetapi, jika sebuah gen tidak menghasilkan satupun angka satu dari hasil random biner maka nilai gen yang terpilih adalah sejumlah destinasi yang dipilih oleh *user*. Alur pembentukan kromosom awal yang ditubuhkan dalam proses bisa dilihat pada Gambar 3.3.



Gambar 3.3 Flowchart untuk pembentukan populasi awal

Pada Gambar 3.4 ialah flowchart untuk proses pembentukan grup gen dan mengisi grup gen dengan nilai random.



Gambar 3.4 Flowchart Pembentukan Grup Gen

Penentuan nilai gen dalam sebuah kromosom dengan menghitung jumlah nilai angka 1 yang dihasilkan dalam setiap grup gen. Hasil penentuan nilai gen dalam setiap kromosom dapat dilihat pada Tabel 3.2.

Tabel 3.2 Representasi kromosom awal

1	1	1	1	1	1	1	0	1	1	0	0	1	0	0	0
4				3				2				1			
1	1	1	0	1	1	0	0	0	0	0	0	1	0	0	0
3				2				5				1			
1	1	1	0	1	0	0	0	1	1	0	0	0	0	0	0
3				1				2				5			
0	0	0	1	1	1	1	0	1	1	0	0	0	0	0	0
1				3				2				5			
1	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0
4				3				5				1			

Dari Tabe 3.2 diatas didapatkan bahwa penentuan nilai sebuah gen dalam kromosom ditentukan menggunakan angka acak berupa biner yang telah di buat menjadi grup-grup. Aturan yang digunakan dalam menentukan nilai gen dalam sebuah grup adalah dengan mnghitung jumlah angka 1 yang berada dalam setiap grup gen. Sedangkan jika dalam sebuah grup gen tidak terdapat nilai 1, maka nilainya adalah sejumlah destinasi yang dituju. Dari proses representasi kromosom diatas maka didapatkan sebuah populasi baru untuk proses GA dan ACO yang dapat dilihat pada Tabel 3.3.

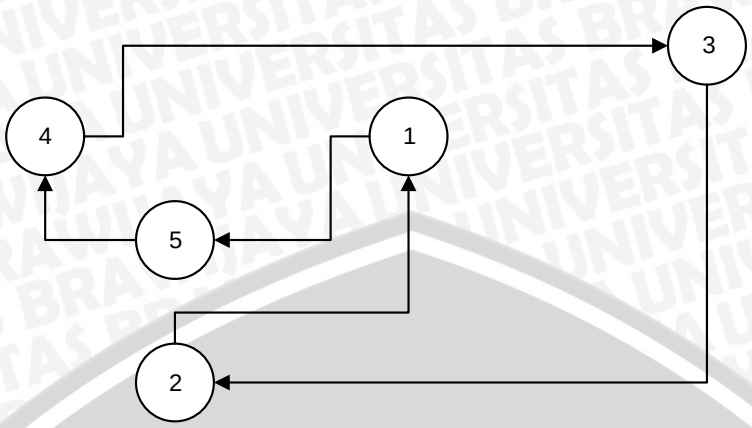
Tabel 3.3 Populasi awal

p1	4	3	2	1	5	264.1
p2	3	2	5	1	4	282.8
p3	3	1	2	5	4	233.5
p4	1	3	2	5	4	247.4
p5	4	3	5	1	2	271.3

Pada Tabel 3.3 didapatkan populasi awal untuk perhitungan hirbridisasi menggunakan GA dan ACO. Selain itu, didapatkan jarak total dari kota-kota yang dilewatinya. Berikut salah satu contoh perhitungan total jarak dalam kilometer(km) pada sebuah kromosom.

$$Kromosom = 4-3-2-1-5 = 85+101+39{,}1+35{,}1+4 = 264$$

Pada Gambar 3.5 merupakan jalur terbaik sementara hasil dari pembentukan populasi awal berdasarkan hasil grup gen yang diisi dengan bilangan random. Dari hasil sementara yang didapatkan akan digunakan sebagai perhitungan dalam proses-proses selanjutnya. Jalur sementara sebelum proses perhitungan menggunakan algoritma genetika dan algoritma koloni semut adalah dari Mataram, Labuhan Lombok, Lembar, Bandara dan terakhir menuju Sekarbela.



Gambar 3.5 Jalur Terbaik Sementara

3.3.2 Algoritma Koloni Semut

Pada tahap ini, pertama-tama akan ditentukan terlebih dahulu nilai awal yang dibutuhkan dalam perhitungan nantinya. Nilai-nilai tersebut seperti *alfa(α), beta(β), rho(ρ)*, banyak semut(k), dan nilai *algoritma greedy* dari kelima kota(C_{greedy}). Nilai *alfa* didapatkan dari jumlah destinasi wisata dibagi dengan minimum total jarak pada populasi pertama. Sedangkan nilai visibilitas antar kota didapatkan dari *invers* nilai jarak antar kota. Setelah menetapkan parameter-parameter awal, proses selanjutnya adalah pencarian nilai feromon awal yang menggunakan persamaan 2.2. Berikut adalah contoh perhitungan untuk mencari nilai feromon awal dalam ACO.

$$\tau_0 = \frac{k}{C_{greedy}} = \frac{5}{233{,}5} = 0{,}0214133 = 0{,}021$$

Dari hasil perhitungan diatas didapatkan bahwa nilai feromon awal dalam ACO adalah 0,025. Nilai tersebut didapatkan dari hasil pembagian antara jumlah semut dengan total jarak minimum sementara dari hasil pembentukan populasi awal. Selanjutnya adalah proses pencarian nilai visibilitas menggunakan persamaan 2.3. Hasil pencarian nilai visibilitas setiap kota dapat dilihat di Tabel 3.4.

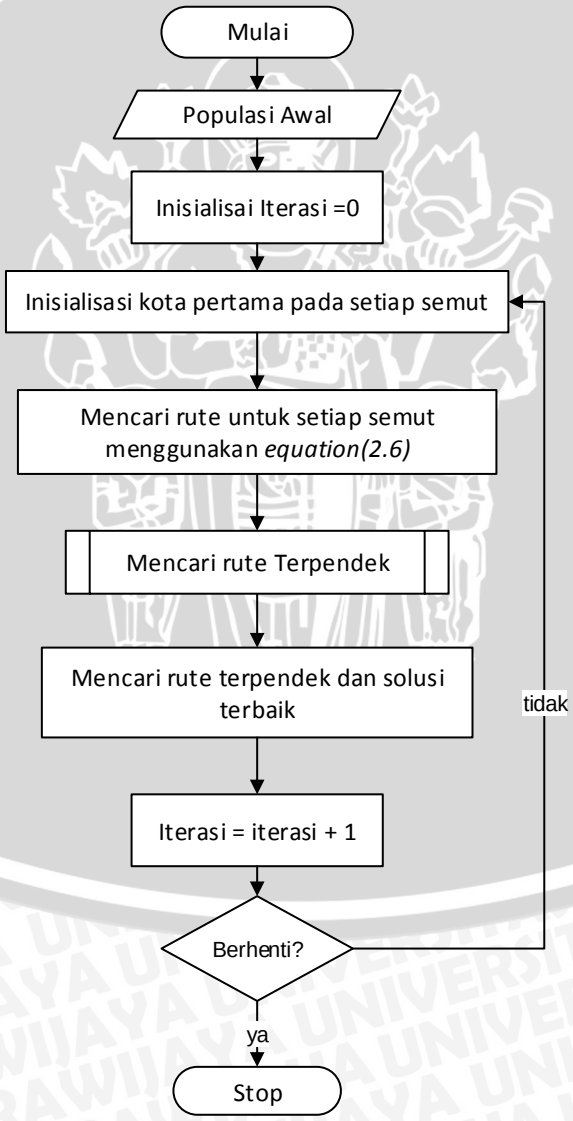
Tabel 3.4 Nilai visibilitas

	LABUHAN				
	BANDARA	LEMBAR	LOMBOK	MATARAM	SEKARBELA
BANDARA	0.000	0.026	0.012	0.027	0.028
LEMBAR	0.026	0.000	0.010	0.037	0.040
LABUHAN LOMBOK	0.012	0.010	0.000	0.012	0.012
MATARAM	0.027	0.037	0.012	0.000	0.250
SEKARBELA	0.028	0.040	0.012	0.250	0.000

Pada Tabel 3.4 merupakan nilai visibilitas antar setiap kota yang dilewati oleh semut. Berikut salah contoh perhitungan untuk mencari nilai visibilitas antar kota.

$$n_{5-4} = \frac{1}{d_{5-4}} = \frac{1}{4} = 0{,}25$$

Pada contoh perhitungan diatas merupakan salah satu contoh pencarian nilai visibilitas antar kota Sekarbela ke kota Mataram. Selanjutnya, adalah proses pencarian nilai untuk menetukan kota tujuan selanjutnya yang akan dikunjungi oleh semut. Untuk melakukan proses kunjungan terdapat beberapa syarat yang akan digunakan semut. Berikut adalah flowchart untuk proses algoritma koloni semut dapat dilihat pada Gambar 3.6.

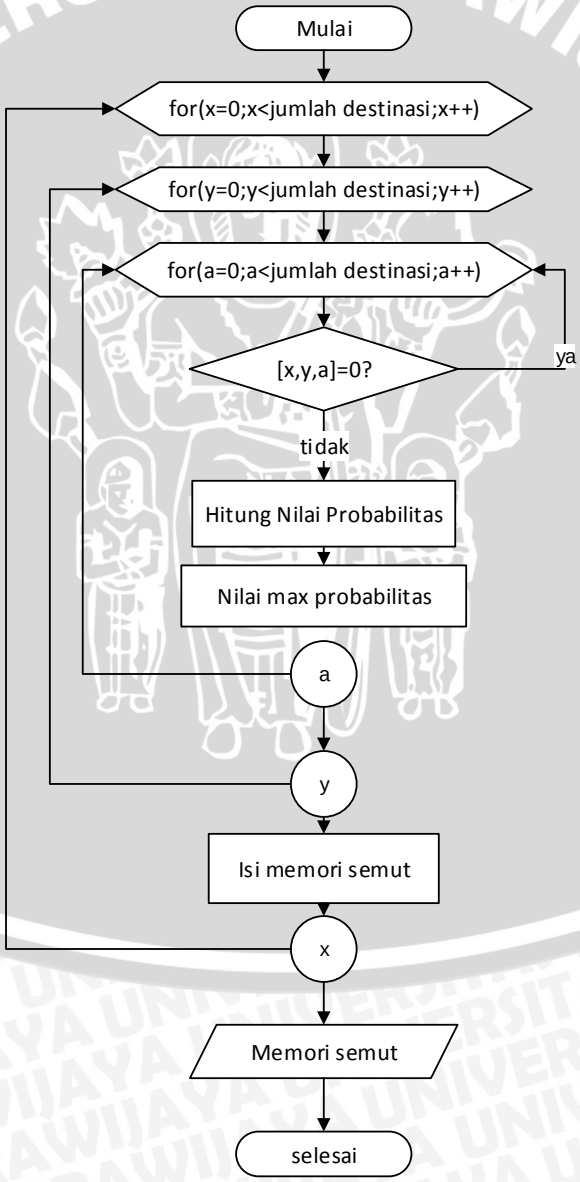


Gambar 3.6 Flowchart proses Algoritma Koloni Semut

Pertama setiap semut harus memulai perjalanan dari kota yang berbeda satu sama lainnya. Terdapat 5 kota yang akan dilalui oleh semut sampai kembali ketempat asalnya, pada setiap langkah setiap harus melakukan hal-hal berikut seperti :

1. Memilih kota yang dikunjungi berdasarkan nilai Probabilitas tertinggi
2. Mencatat setiap kota yang dikunjungi

Pada perhitungan nilai probabilitas kota yang akan dikunjungi oleh semut harus dilakukan pada setiap kota. Proses perhitungan nilai probabilitas ini dilakukan sampai tidak ada lagi kota yang akan dituju lagi probabilitas kota terakhir menjadi 1. Pengisian memori semut dilakukan pada proses perhitungan semua kota selesai, jika telah didapatkan nilai probabilitas yang tertinggi maka kota yang dituju akan dicatat dalam memori setiap semut.



Gambar 3.7 Flowchart Pengisian Memori

Pada Gambar 3.7 ialah flowchart untuk proses pencarian nilai probabilitas setiap kota dan pengisian memori semut sebagai catatan kota mana saja yang telah dilewati. Setelah proses selesai, maka semut menyimpan catatan kota yang dikunjungi kemudian akan dihitung jarak dan di seleksi pada proses terakhir.

Kota pertama yang dikunjungi oleh semut adalah kota pertamanya, dimana semut 1 berangkat dari kota BN, semut 2 berangkat dari kota L, semut 3 berangkat dari kota LL, semut 3 berangkat dari kota M, dan semut 5 berangkat dari kota SK setelah itu setiap semut kembali kekotanya masing-masing. Kota pertama yang dikunjungi semut akan disimpan didalam memori masing-masing. Proses kunjungan kota yang dilalui oleh semut dapat dilihat pada Tabel 3.5:

Tabel 3.5 Mengunjungi kota ke-2

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	0.28	0.129	0.290322581	**0.301**	0.301	5	[1 5]
2	0.23	0	0.088	0.327433628	**0.354**	0.354	5	[2 5]
3	**0.26**	0.217	0	0.260869565	0.261	0.261	1	[3 1]
4	0.08	0.113	0.037	0	**0.767**	0.767	5	[4 5]
5	0.08	0.121	0.036	**0.757575758**	0	0.758	4	[5 4]

Hasil Tabel 3.5 didapatkan semut 1 selanjutnya akan mengunjungi kota 5, semut 2 akan mengunjungi kota 5, semut 3 akan mengunjungi kota 1, semut 4 akan mengunjungi kota 5, dan semut 5 akan mengunjungi kota 4. Proses perhitungan besarnya peluang kota tersebut dikunjungi menggunakan persamaan 2.6. Berikut adalah satu contoh pencarian nilai probabilitas kota yang akan dikunjungi oleh semut.

$$P_{15} = \frac{(0{,}028^1 x 0{,}021^1)}{((0^1 x 0{,}021^1) + (0{,}021^1 x 0{,}026^1) + (0{,}021^1 x 0{,}012^1) + (0{,}021^1 x 0{,}027^1) + (0{,}021^1 x 0{,}028^1)}$$

$$= 0{,}3010753 = 0{,}3011$$

Pada perhitungan diatas didapatkan nilai 0,3011 untuk probabilitas semut pertama dalam mengunjungi kota ke-5 sebagai kunjungan selanjutnya. Nilai diatas didapatkan hasil pembagian yang diman pembilang adalah hasil kali nilai feromon awal dengan visibilitas kota ke-5. Sedangkan untuk penyebutnya didapatkan dari hasil sigma dari nilai feromon dengan nilai visibilitas kota masing-masing.

Tabel 3.6 Mengunjungi kota ke-3

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	0.4	0.185	**0.415384615**	0	0.415	4	[1 5 4]
2	0.36	0	0.137	**0.506849315**	0	0.507	4	[2 5 4]
3	0	0.294	0	0.352941176	**0.353**	0.353	5	[3 1 5]
4	0.36	**0.487**	0.158	0	0	0.487	2	[4 5 2]
5	0.35	**0.5**	0.15	0	0	0.5	2	[5 4 2]

Pada Tebel 3.6 merupakan hasil perhitungan nilai probabilitas semut dalam mengujungi kota ke-3. Dari hasil perhitungan probabilitas untuk setiap kota kemudian dilipih yang memiliki nilai probabilitas paling tinggi. Dari Tabel 3.6 dapat dilihat semut 1 dan 2 akan mengunjungi kota 4 sebagai kunjungan selanjutnya. Sedangkan untuk semut 3 memilih kota 5 sebagai tujuan selanjutnya, serta untuk semut 4 dan 5 memilih kota 2 sebagai kunjungan selanjutnya.

Tabel 3.7 Mengunjungi kota ke-4

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	**0.684**	0.316	0	0	0.684	2	[1 5 4 2]
2	**0.72**	0	0.278	0	0	0.722	1	[2 5 4 1]
3	0	0.455	0	**0.545454545**	0	0.545	4	[3 1 5 4]
4	**0.69**	0	0.308	0	0	0.692	1	[4 5 2 1]
5	**0.7**	0	0.3	0	0	0.7	1	[5 4 2 1]

Pada Tebel 3.7 merupakan hasil perhitungan nilai probabilitas semut dalam mengujungi kota ke-4. Dari hasil perhitungan probabilitas untuk setiap kota kemudian dilipih yang memiliki nilai probabilitas paling tinggi. Dari Tabel 3.7 dapat dilihat semut 1 akan mengunjungi kota 2 sebagai kunjungan selanjutnya. Sedangkan untuk semut 2, 4, dan 5 memilih kota 1 sebagai tujuan selanjutnya, serta untuk semut 3 memilih kota 4 sebagai kunjungan selanjutnya.

Tabel 3.8 Mengunjungi Kota ke-5

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	0	**1**	0	0	1	3	[1 5 4 2 3]
2	0	0	**1**	0	0	1	3	[2 5 4 1 3]
3	0	**1**	0	0	0	1	2	[3 1 5 4 2]
4	0	0	**1**	0	0	1	3	[4 5 2 1 3]
5	0	0	**1**	0	0	1	3	[5 4 2 1 3]

Pada Tebel 3.8 merupakan hasil perhitungan nilai probabilitas semut dalam mengujungi kota ke-5. Dari hasil perhitungan probabilitas untuk setiap kota kemudian dilipih yang memiliki nilai probabilitas paling tinggi. Dari Tabel 3.8 dapat dilihat semut 1, 2, 4, dan 5 akan mengunjungi kota 3 sebagai kunjungan selanjutnya. Sedangkan untuk semut 3 memilih kota 1 sebagai tujuan selanjutnya.

Setelah semua semut mengunjungi setiap kota tepat satu kali maka selanjutnya akan dilakukan proses perbandingan panjang keseluruhan rute yang ditempuh oleh setiap semut. Hasil perbandingan panjang rute setiap semut dapat dilihat dalam Tabel 3.9.

Tabel 3.9 Hasil perhitungan ACO

Kota Awal	memori	rute (km)
1	[1 5 4 2 3]	248
2	[2 5 4 1 3]	247
3	[3 1 5 4 2]	248
4	[4 5 2 1 3]	234
5	[5 4 2 1 3]	236

Pada Tabel 3.9 bahwasannya semut 4 yang berangkat dari Mataram mendapatkan rute yang optimal dalam menentukan rute destinasi wisata. Panjang rute yang dilalui oleh semut 4 ialah 234 km.

3.3.3 Algoritma Genetika

Pada tahapan algoritma genetika (GA) melakukan proses *crossover* dan mutasi untuk mendapatkan individu baru. Pada proses dilakukan dengan menggunakan metode *crossover* silang. Untuk proses pada *crossover* dapat dilihat pada Gambar 3.5. Pertama, membangkitkan angka random sejumlah individu awal. Pada perhitungan ini digunakan 5 buah individu, sehingga akan dibangkitkan 5 buah angka random biner. Setelah itu, bandingkan angka random tersebut dengan nilai *pc*. Hasil perbandingan angka random dengan pc dapat dilihat pada Tabel 3.10.

Tabel 3.10 Nilai random

1	random < *Pc*	FALSE
1		FALSE
0		TRUE
0		TRUE
0		TRUE

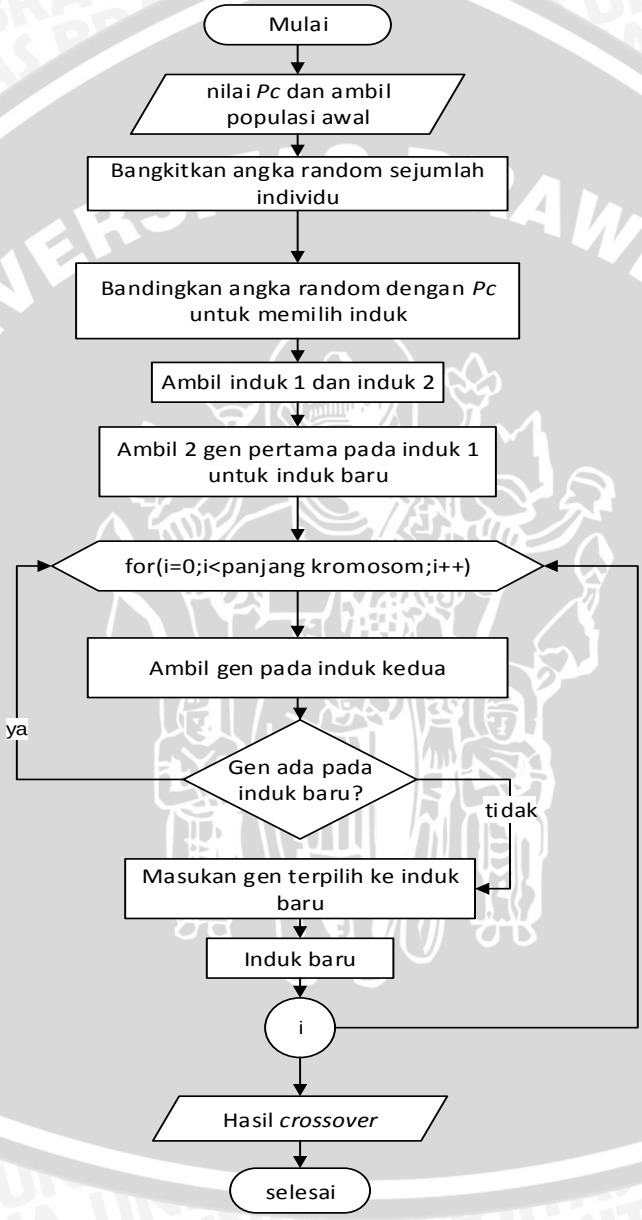
Dari hasil perbandingan Tabel 3.10 didapatkan bahwa yang akan mengalami proses *crossover* adalah kromosom 3, 4, dan 5. Karena menggunakan crossover silang maka diperoleh kromosom 3 X kromosom 4, kromosom 4 X kromosom 5, dan kromosom 5 X kromosom 3. Hasil proses kromosom dapat dilihat pada Tabel 3.11.

Tabel 3.11 Proses *crossover*

p3	3	1	2	5	4
p4	1	3	2	5	4
p6	3	1	2	5	4
p4	1	3	2	5	4
p5	4	3	5	1	2
p7	1	3	4	5	2
p3	3	1	2	5	4

p5	4	3	5	1	2
p8	3	1	4	5	2

Dari Tabel 3.11 dapat dilihat bahwa telah terbentuk 3kromosom baru dari proses *crossover*. Pada Gambar 3.8 dapat dilihat flowchart untuk proses *crossover* menggunakan metode *one-cut point*. Kromosom baru hasil proses *crossover* dinamakan p6, p7, dan p8.



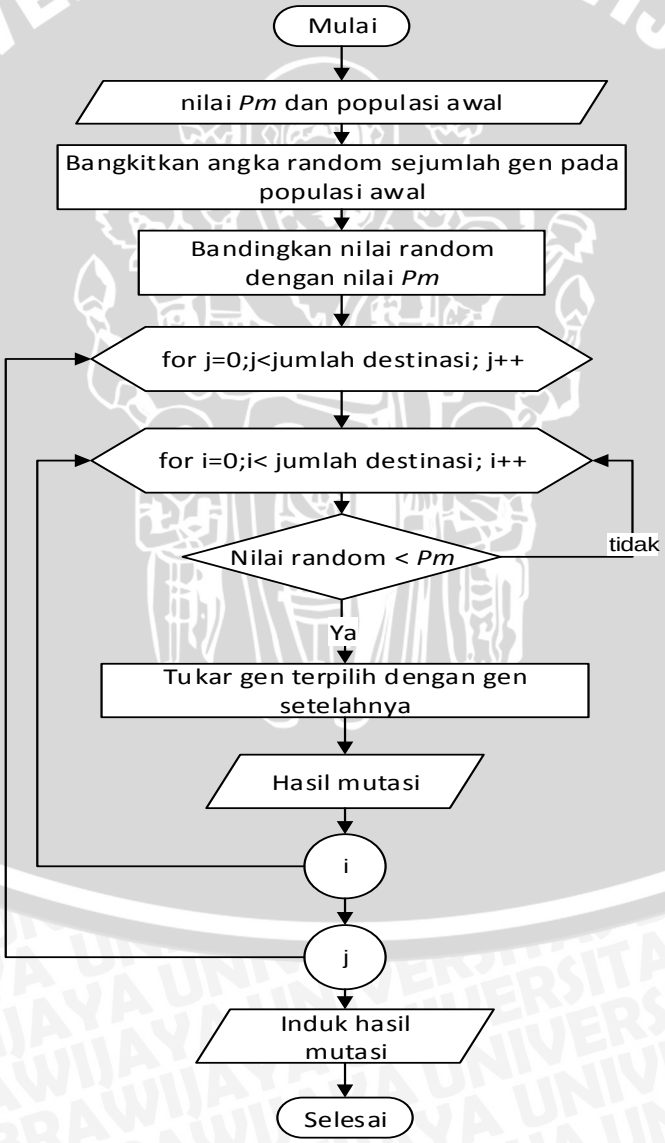
Gambar 3.8 Flowchart proses crossover

Selanjutnya, akan dilakukan proses *mutasi* untuk mendapatkan kromosom baru. Proses *mutasi* disini menggunakan metode *uniform tansform*. Proses mutasi menggunakan metode *uniform tansform* sama seperti pada teknik *swapping mutation* yaitu dengan menukarkan nilai gen terpilih dengan dengan yang

berada didepannya.Tetapi, pada *uniform transform* proses pemilihan gen yang mengalami mutasi berdasarkan perbandingan nilai random dengan *Pc.* Hal itu dilakukan sebanyak gen yang terpilih untuk dilakukan proses mutasi. Jika gen yang terpilih berada di paling terakhir maka gen tersebut akan ditukarkan dengan gen pertama.

Untuk melakukan proses *mutasi* ada beberapa tahapan yang harus dilakukan terlebih dahulu dapat dilihat pada Gambar 3.9. Berikut adalah urutan proses *mutasi* dengan menggunakan *uniform tansform* :

1. Bangkitkan angka random antara 0 sampai 1 sejumlah gen (panjang kromosom dikalikan dengan jumlah kromosom.
2. Bandingkan angka random dengan gen-gen dalam kromosom
3. Jika bilangan acak yang dihasilkan 0, maka gen yang sesuai akan ditukarkan dengan gen setelahnya.



Gambar 3.9 Flowchart proses mutasi

Hasil angka perbandingan angka random dengan gen-gen dalam kromosom dapat dilhat pada Tabel 3.12.

Tabel 3.12 Angka random

1	1	1	0	1
0	1	1	1	1
1	1	1	1	1
1	1	0	0	1
1	1	1	1	0

Pada Tabel 3.12 perbandingan sebuah nilai random dapat dilihat bahwa yang mengalami proses *mutasi* adalah kromosom 1, 2, 4, dan 5. Sehingga gen-gen yang diberi tanda warna kuning tersebut akan digunakan dengan gen setelahnya. Berikut adalah hasil proses *mutasi* dapat dilihat pada Tabel 3.13.

Tabel 3.13 Proses mutasi

p1	4	3	2	1	5
p9	4	3	2	5	1
p2	3	2	5	1	4
p10	2	3	5	1	4
p4	3	2	5	1	4
p11	3	2	1	4	5
p5	4	3	5	1	2
p12	2	3	5	1	4

Pada Tabel 3.13 merupakan hasil proses *mutasi* menggunakan metode *uniform tansform*. Hasil indukan baru dari proses mutasi pada kromosom p9, p10, p11, dan p12.

3.3.4 Proses Seleksi

Dalam tahapan ini menggunakan model seleksi *elitisme*, yaitu dengan membandingkan seluruh individu yang dihasilkan pada metode GA dan ACO. Setelah semua invidu digabungkan, proses selanjutnya adalah mencari nilai *cost* dari setiap individu yang ada. Berikut adalah hasil penggabungan seluruh individu pada metode GA dan ACO dapat dilihat pada Tabel 3.14.

Tabel 3.14 Penggabungan seluruh individu

kromosom						jarak
p1	4	3	2	1	5	264.1
p2	3	2	5	1	4	282.8
p3	3	1	2	5	4	233.5
p4	1	3	2	5	4	247.4
p5	4	3	5	1	2	271.3
c1	3	1	2	5	4	233.5

c2	1	3	4	5	2	233.5
c3	3	1	4	5	2	247.4
m1	4	3	2	5	1	282.8
m2	2	3	5	1	4	285.2
m3	3	2	1	4	5	266.2
m4	2	3	5	1	4	285.2
s1	1	5	4	2	3	247.7
s2	2	5	4	1	3	247.4
s3	3	1	5	4	2	247.7
s4	4	5	2	1	3	233.5
s5	5	4	2	1	3	235.9

Tabel 3.15 Individu terpilih

p3	3	1	2	5	4	233.5
c1	3	1	2	5	4	233.5
c2	1	3	4	5	2	233.5
s4	4	5	2	1	3	233.5
s5	5	4	2	1	3	235.9

Pada Tabel 3.15 diatas didapatkan individu-individu baru hasil seleksi dari proses metode ACO dan GA. Hasil diatas dipotong sesuai banyaknya populasi awal yang terbentuk. Dari populasi ini akan digunakan untuk iterasi selanjutnya, jika tidak ada iterasi lagi maka individu terbaik yang digunakan sebgai hasil akhir adalah individu pertama.

3.3.5 Proses Iterasi

Dalam kasus *travelling salesman problem (TSP)* atau optimasi biasanya akan dilakukan proses iterasi lebih dari satu kali. Untuk masalah hibridisasi *GA-ACO* ini akan ada perubahan dalam sisi pembaruan feromon pada sisi perhitungan *ACO*. Feromon yang digunakan selanjutnya adalah diambil dari rute terbaik pada individu terpilih dalam iterasi sebelumnya. Berikut adalah proses-proses yang dilakukan dalam melakukan iterasi berikutnya.

1. Pembaruan feromon

Tabel 3.16 Individu Awal

urutan kota					rute	feromon
3	1	2	5	4	**233.5**	**0.0043**
3	1	2	5	4	233.5	0.0043
1	3	4	5	2	233.5	0.0043
4	5	2	1	3	233.5	0.0043
5	4	2	1	3	235.9	0.0042

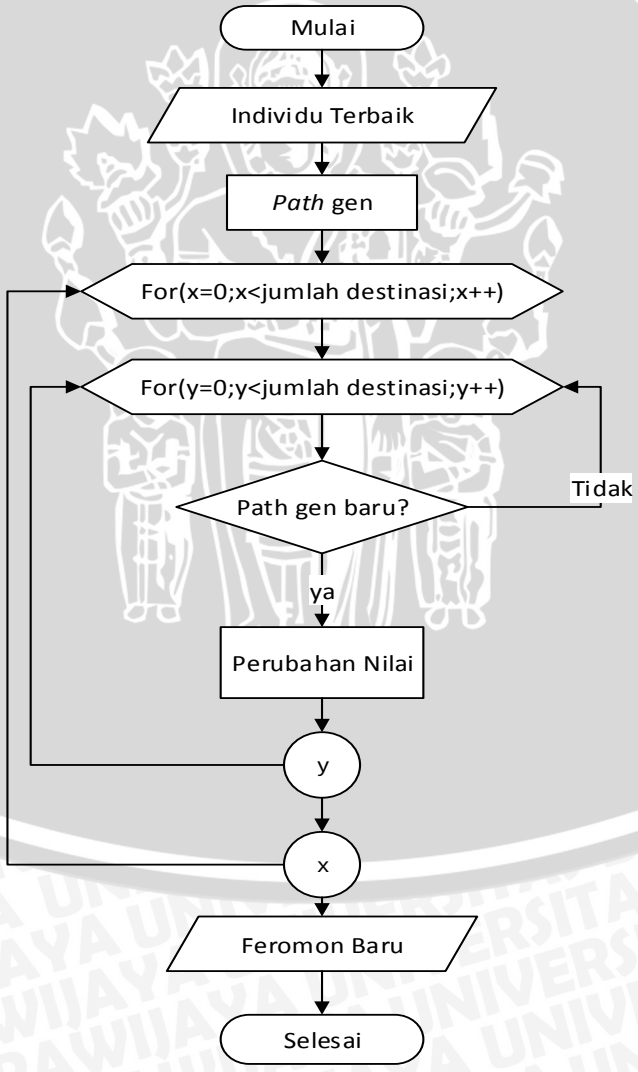
Pada Tabel 3.16 diatas adalah data individu baru dari iterasi sebelumnya beserta dengan nilai feromon masing-masing. Pada Gambar 3.10 ialah *flowchart* untuk pembaruan feromon. Perubahan nilai feromon dilakukan pada jalur individu terbaik. Sehingga pembaruan feromon hanya dilakukan pada beberapa jalur saja. Proses pembaruan feromon ini dilakukan pada setiap iterasi kecuali pada iterasi pertama. Berikut adalah salah satu contoh perhitungan untuk mencari nilai feromon diatas yang menggunakan persamaan 2.7.

$$\Delta\tau = \frac{1}{Lk} = \frac{1}{233,5} = 0,004283 = 0,0043$$

Setelah didapatkan nilai feromon masing-masing, selanjutnya adalah mencari nilai fembaruan feromon untuk proses perhitungan metode ACO selanjutnya.

Berikut adalah contoh perhitungan untuk pembaran feromon menggunakan persamaan 2.8.

$$\tau baru = (1-\rho)x\tau_{ij} + \Delta\tau_{ij} = (1-0)x0,021 + 0,004283 = 0,02355$$



Gambar 3.10 Flowchart Pembaruan Feromon

Karena pembaruan feromon diatas menggunakan individu pertama sebagai acuan, maka nilai feromon rute-rute yang yang sesuai dengan individu tersebut akan dirubah dengan 0,02355. Berikut tabel pembaruan feromon baru dapat dilihat di Tabel 3.17.

Tabel 3.17 Feromon Baru

0	0.023555	0.023555	0.025	0.025
0.023555	0	0.025	0.023555	0.025
0.023555	0.025	0	0.025	0.023555
0.025	0.023555	0.025	0	0.023555
0.025	0.025	0.023555	0.023555	0

2. Algoritma koloni semut

Dalam perhitungan menggunakan metode ACO, langkah-langkah yang digunakan sama dengan iterasi sebelumnya. Untuk semut pertama berangkat dari kota pertama, begitu selanjutnya sampai semut kelima berangkat dari kota kelima. Berikut hasil kunjungan semut untuk kota kedua dapat dilihat pada Tabel 3.18.

Tabel 3.18 Menunjungi kota ke-2

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	0.2698	0.125	0.297	**0.308**	**0.30835986**	5	[1 5]
2	0.224	0	0.091	0.319	**0.366**	**0.36577247**	5	[2 5]
3	**0.253**	0.2242	0	0.269	0.253	**0.26898339**	1	[3 1]
4	0.087	0.1127	0.039	0	**0.761**	**0.76128255**	5	[4 5]
5	0.089	0.127	0.036	**0.748**	0	**0.74811611**	4	[5 4]

Pada Tabel 3.18 dapat dlihat bahwa semut akan mengunjungi kota ke-2 sebagai kunjungan selanjutnya. Berikut contoh perhitungan untuk melihat nilai probabilitas semut dalam mengunjungi kota-kota selanjutnya menggunakan persamaan 2.6.

$$P_{1-2} = \frac{(0{,}026^1 x 0{,}023555^1)}{(0^1 x 0^1) + (0{,}026^1 x 0{,}023555^1) + (0{,}012^1 x 0{,}023555^1) + (0{,}027^1 x 0{,}025^1) + (0{,}028^1 x 0{,}025^1)}$$
$$= 0{,}2698$$

Selanjutnya, untuk melihat hasil kunjungan semut dalam mengunjuni kota-kota berikutnya dapat dilihat pada Tabel 3.19, 3.20, dan 3.21.

Tabel 3.19 Mengunjungi kota ke-3

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	0.3901	0.18	**0.43**	0	0.42991578	4	[1 5 4]

2	0.353	0	0.144	**0.503**	0	0.50262427	4	[2 5 4]
3	0	0.3002	0	**0.36**	0.339	0.36029317	4	[3 1 4]
4	0.366	**0.472**	0.162	0	0	0.47197982	2	[4 5 2]
5	0.353	**0.5044**	0.143	0	0	0.50437412	2	[5 4 2]

Tabel 3.20 Mengunjungi kota ke-4

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	**0.6842**	0.316	0	0	0.68421053	2	[1 5 4 2]
2	**0.71**	0	0.29	0	0	0.71011794	1	[2 5 4 1]
3	0	0.4693	0	0	**0.531**	0.53065327	5	[3 1 4 5]
4	**0.692**	0	0.308	0	0	0.69230769	1	[4 5 2 1]
5	**0.712**	0	0.288	0	0	0.71235563	1	[5 4 2 1]

Tabel 3.21 Mengunjungi kota ke-5

Kota Awal	Probabilitas					probabilitas	tujuan	memori
	kota 1	kota 2	kota 3	kota 4	kota 5			
1	0	0	**1**	0	0	1	3	[1 5 4 2 3]
2	0	0	**1**	0	0	1	3	[2 5 4 1 3]
3	0	**1**	0	0	0	1	2	[3 1 4 5 2]
4	0	0	**1**	0	0	1	3	[4 5 2 1 3]
5	0	0	**1**	0	0	1	3	[5 4 2 1 3]

Setelah semua semut mengunjungi setiap kota tepat satu kali maka selanjutnya akan dilakukan proses perbandingan panjang keseluruhan rute yang ditempuh oleh setiap semut. Hasil perbandingan panjang rute setiap semut dapat dilihat dalam Tabel 3.22.

Tabel 3.22 Hasil perhitungan ACO

s1	1	5	4	2	3	247.7
s2	2	5	4	1	3	247.4
s3	3	1	4	5	2	247.4
s4	4	5	2	1	3	233.5
s5	5	4	2	1	3	235.9

Dari Tabel 3.22 didapatkan bahwa semut keempat dan kelima yang berangkat dari Bandara dan Mataram yang mendapatkan rute yang optimal dalam menentukan rute destinasi wisata. Panjang rute yang dilalui oleh semut ialah sama-sama 233,5 km.

3. Algoritma Genetika

Pada proses algoritma, langkah-langkah yang dilakukan sama seperti pada iterasi sebelumnya yaitu proses *crossover* dan *mutasi*. Untuk proses

crossover pertama membangkitkan nilai random biner sebanyak jumlah individu. Selanjutnya bandingkan nilai pc, jika nilai random lebih kecil dari nilai pc maka individu tersebut mengalami proses *crossover.*

Tabel 3.23 Perbandingan bilangan random *crossover*

1		FALSE
1		FALSE
0	BANDINGKAN DENGAN PC	TRUE
1		FALSE
0		TRUE

Pada Tabel 3.23 diatas dapat dilihat bahwa yang mengalami proses mutasi adalah kromosom 3 dan 5. Dua buah kromosom inilah yang akan di *crossover* untuk mendapatkan individu baru. Berikut proses *crossover* dapat dilihat pada Tabel 3.24.

Tabel 3.24 Proses *crossover*

p3	1	3	4	5	2
p5	5	4	2	1	3
p6	1	3	5	4	2

Dari proses *crossover* pada Tabel 3.24 diatas didapatkan individu baru dengan rute kunjungan kota 1-3-5-4-2. Selanjutnya adalah melakukan proses mutasi. Proses *mutasi* akan dilakukan menggunakan metode *uniform tansform*. Langkah pertama adalah membangkitkan angka random biner sejumlah gen. berikutnya adalah membandingkan angka random dengan gen-gen dalam setiap individu. Jika bilangan random yang dihasilkan lebih kecil dari pm, maka gen tersebut akan ditukar dengan gen setelahnya.

Tabel 3.25 Perbandingan bilangan random mutasi

0	1	1	0	1	gen pada populasi yang terpilih untuk dimutasi	3	1	2	5	4
0	1	1	1	1		3	1	2	5	4
0	1	1	0	1		1	3	4	5	2
1	1	0	1	1		4	5	2	1	3
0	1	1	1	1		5	4	2	1	3

Pada Tabel 3.25 diatas dapat dilihat bahwa terdapat 6 buah gen yang mengalami proses mutasi. Proses mutasi yaitu dengan menukar gen yang ditandai dengan gen setelahnya. Proses mutasi pada iterasi ini dapat dilihat pada Tabel 3.26.

Tabel 3.26 Proses Mutasi

p1	3	1	2	5	4
p7	1	3	2	4	5
p2	3	1	2	5	4
p8	1	3	2	5	4
p3	1	3	4	5	2

p9	3	1	4	2	5
p5	5	4	2	1	3
p10	4	5	2	1	3

4. Proses Seleksi

Proses seleksi akan menggunakan metode seleksi *elitism* seperti pada iterasi sebelumnya. Proses seleksi dilakukan dengan membandingkan seluruh individu yang dihasilkan pada metode GA dan ACO. Setelah semua invidu digabungkan, proses selanjutnya adalah mencari nilai *cost* dari setiap individu yang ada.

Tabel 3.27 Proses seleksi

kromosom						jarak
p1	3	1	2	5	4	233.5
p2	3	1	2	5	4	233.5
p3	1	3	4	5	2	233.5
p4	4	5	2	1	3	233.5
p5	5	4	2	1	3	235.9
c1	1	3	5	4	2	235.9
m1	1	3	2	4	5	247.7
m2	1	3	2	5	4	247.4
m3	3	1	4	2	5	254.6
m4	4	5	2	1	3	233.5
s1	1	5	4	2	3	247.7
s2	2	5	4	1	3	247.4
s3	3	1	4	5	2	247.4
s4	4	5	2	1	3	233.5
s5	5	4	2	1	3	235.9

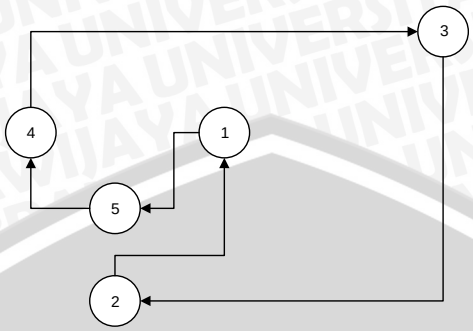
Pada Tabel 3.28 diatas didapatkan 5 individu terbaik untuk dengan rute yang paling optimal. Individu-individu hasil seleksi untuk proses iterasi selanjutnya dapat dilihat pada Tabel 3.28.

Tabel 3.28 Individu terbaik

p1	3	1	2	5	4	233.5
p2	3	1	2	5	4	233.5
p3	1	3	4	5	2	233.5
p4	4	5	2	1	3	233.5
m4	4	5	2	1	3	233.5

Pada Tabel 3.28 adalah populasi terbaik dari proses perhitungan menggunakan hibridisasi algoritma genetika dan algoritma koloni semut. Tetapi individu terbaik

hasil perhitungan adalah individu teratas atau p1. Berikut jalur akhir yang didapat dari hasil proses perhitungan dapat dilihat pada Gambar 3.11.



Gambar 3.11 Jalur Akhir

3.4 Skenario Pengujian

Untuk menentukan parameter *Hybrid GACO* perlu dibuatnya skenario pengujian untuk mendapatkan parameter yang optimal. Beberapa skenario yang perlu dibuat antara lain adalah :

1. Pengujian untuk pengaruh jumlah iterasi dalam metode.
2. Pengujian pengaruh nilai *Pc* terhadap metode.
3. Pengujian pengaruh nilai *Pm* terhadap metode.
4. Pengujian pengaruh nilai *Alfa* terhadap metode.
5. Pengujian pengaruh nilai *Beta* terhadap metode.
6. Pengujian untuk pengaruh jumlah destinasi dalam metode.

3.4.1 Pengujian Untuk Pengaruh Jumlah Iterasi Dalam Metode

Pengujian pengaruh jumlah iterasi dilakukan untuk mencari seberapa besar pengaruh banyaknya jumlah iterasi dalam mencari nilai optimal. Besarnya jumlah iterasi akan berpengaruh terhadap kecepatan proses dalam mencari kombinasi individu yang optimal. *Range* nilai iterasi yang akan diuji adalah dengan kelipatan 10. Rancangan pengujian terhadap jumlah iterasi dapat dilihat pada Tabel 3.29.

Tabel 3. 29 Skenario Pengujian Jumlah Iterasi

Jumlah Iterasi	Percobaan ke- *i*										Rata-rata *Cost*
	1	2	3	4	5	6	7	8	9	10	
1											
5											
10											

15											
20											
25											
50											
75											
100											
1000											

3.4.2 Pengujian Pengaruh Nilai Pc Terhadap Metode

Pengujian pengaruh nilai *Pc* dilakukan untuk mencari seberapa panjang nilai range yang agar dapat menghasilkan nilai parameter algoritma genetika yang optimal. Nilai *range* tersebut akan berpengaruh terhadap jumlah induk anak yang akan terbentuk. Rancangan pengujian terhadap nilai *Pc* dapat dilihat pada Tabel 3.30.

Tabel 3.30 Skenario Pengujian Nilai *Pc*

Nilai *Pm*	Percobaan ke-*i*										Rata-rata *Cost*
	1	2	3	4	5	6	7	8	9	10	
0,1											
0,2											
0,3											
0,4											
0,5											
0,6											
0,7											
0,8											
0,9											

3.4.3 Pengujian Pengaruh Nilai Pm Terhadap Metode

Pengujian pengaruh nilai *Pm* dilakukan untuk mencari seberapa panjang nilai range yang agar dapat menghasilkan nilai parameter algoritma genetika yang optimal. Nilai *range* tersebut akan berpengaruh terhadap jumlah induk anak yang akan terbentuk. Rancangan pengujian terhadap nilai *Pm* dapat dilihat pada Tabel 3.31.

Tabel 3.31 Skenario Pengujian Nilai *Pm*

Nilai *Pc*	Percobaan ke- *i*										Rata-rata *Cost*
	1	2	3	4	5	6	7	8	9	10	
0,1											
0,2											
0,3											
0,4											

0,5											
0,6											
0,7											
0,8											
0,9											

3.4.4 Pengaruh Nilai *Alfa* Terhadap Metode

Pengujian pengaruh nilai *alfa* dilakukan untuk mencari seberapa panjang nilai range yang agar dapat menghasilkan nilai parameter algoritma koloni semut yang optimal. Nilai *range* tersebut akan berpengaruh terhadap jumlah induk anak yang akan terbentuk. Rancangan pengujian terhadap nilai *alfa* dapat dilihat pada Tabel 3.32.

Tabel 3.32 Skenario Pengujian Nilai *Alfa*

Nilai *Alfa*	Percobaan ke- *i*										Rata-rata *Cost*
	1	2	3	4	5	6	7	8	9	10	
0,5											
0,6											
0,7											
0,8											
0,9											
1,1											
1,2											
1,3											
1,4											

3.4.5 Pengaruh Nilai *Beta* Terhadap Metode

Pengujian pengaruh nilai *beta* dilakukan untuk mencari seberapa panjang nilai range yang agar dapat menghasilkan nilai parameter algoritma koloni semut yang optimal. Nilai *range* tersebut akan berpengaruh terhadap jumlah induk anak yang akan terbentuk. Rancangan pengujian terhadap nilai *beta* dapat dilihat pada Tabel 3.32.

Tabel 3.33 Skenario Pengujian Nilai *Beta*

Nilai *Beta*	Percobaan ke- *i*										Rata-rata *Cost*
	1	2	3	4	5	6	7	8	9	10	
0,5											
0,6											
0,7											
0,8											
0,9											
1,1											

1,2											
1,3											
1,4											

3.4.6 Pengujian Untuk Pengaruh Jumlah Destinasi Dalam Metode

Pengujian pengaruh jumlah destinasi dilakukan untuk mencari seberapa besar pengaruh banyaknya jumlah destinasi dalam mencari nilai optimal. Besarnya jumlah iterasi akan berpengaruh terhadap kecepatan proses dalam mencari kombinasi individu yang optimal. *Range* nilai iterasi yang akan diuji adalah dengan kelipatan 5. Rancangan pengujian terhadap jumlah iterasi dapat dilihat pada Tabel 3.34.

Tabel 3.34 Skenario Pengujian Jumlah Destinasi

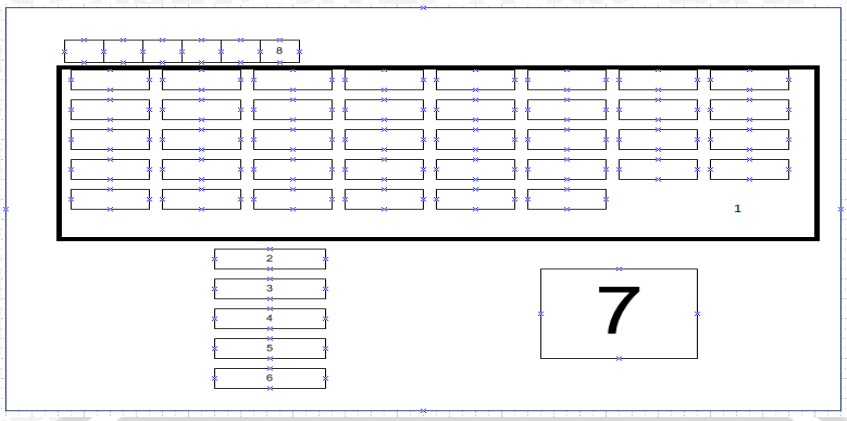
Jumlah Destinasi	Percobaan ke- *i*										Rata-rata (dalam detik)
	1	2	3	4	5	6	7	8	9	10	
5											
10											
15											
20											
25											

3.5 Perancangan *User Interface*

Perancangan *user interface* untuk permasalahan pencarian rute wisata Lombok terdiri dari enam halaman. Halaman pertama berisikan parameter-parameter inputan untuk proses perhitungan menggunakan hibridisasi algoritma genetika dan algoritma koloni semut. Halaman pertama dapat dilihat pada Gambar 3.12.

Keterangan Gambar 3.12.

1. Daftar tempat-tempat wisata Pulau Lombok.
2. Parameter perhitungan algoritma genetika *Pc.*
3. Parameter perhitungan algoritma genetika *Pm.*
4. Parameter perhitungan algoritma koloni semut *Alfa*.
5. Parameter perhitungan algoritma koloni semut *Beta.*
6. Parameter perhitungan untuk jumlah iterasi.
7. Tombol proses untuk memulai proses perhitungan.
8. Halaman destinasi

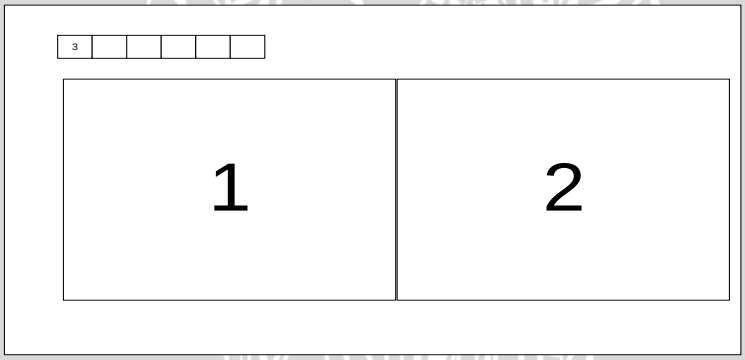


Gambar 3.12 Halaman Pertama

Halaman kedua *user interface* berisi daftar tempat-tempat wisata yang berada di Pulau Lombok. Terdiri dari data jarak antar setiap tempat wisata dan tampilan matrix jarak yang dipilih user dapat dilihat pada Gambar 3.13.

Keterangan pada Gambar 3.13.

1. Menampilkan semua data jarak setiap tempat wisata
2. Menampilkan data jarak antar kota setelah dipilih user pada halaman 6.
3. Halaman data jarak.

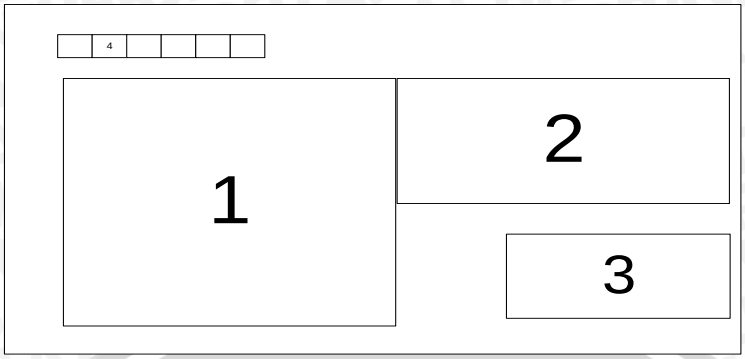


Gambar 3.13 Halaman Kedua

Halaman ketiga berisi representasi kromosom berdasarkan hasil grup gen yang diisi dengan bilangan random biner. Selain itu, pada halaman ketiga juga berisi populasi awal yang terbentuk berdasarkan representasi kromosom dan juga berisi total jarak. Halaman ketiga dapat dilihat pada Gambar 3.14.

Keterangan pada Gambar 3.14.

1. Tampilan hasil representasi kromosom berdasarkan grup gen biner.
2. Tampilan populasi awal yang terbentuk.
3. Total jarak yang didapat dari perhitungan populasi dan hasil akhir.
4. Halaman kromosom.

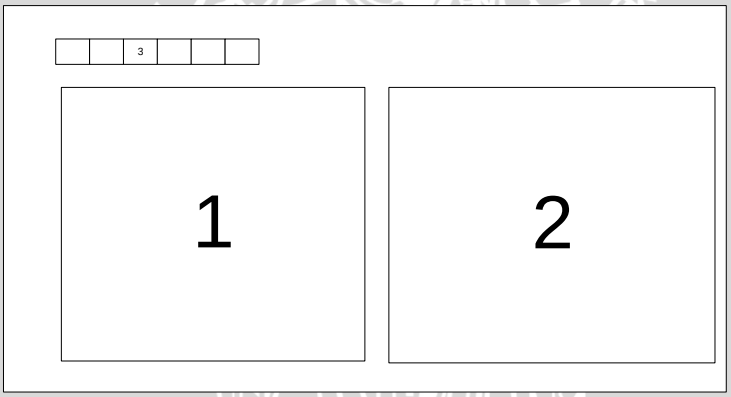


Gambar 3.14 Halaman Ketiga

Halaman keempat pada perancangan user interface berisikan hasil proses crossover dan mutasi pada setiap iterasi yang diinputkan. Halaman keempat pada Gambar 3.15.

Keterangan pada Gambar 3.15.

1. Hasil proses *crossover*.
2. Hasil proses mutasi.
3. Halaman *GA.*

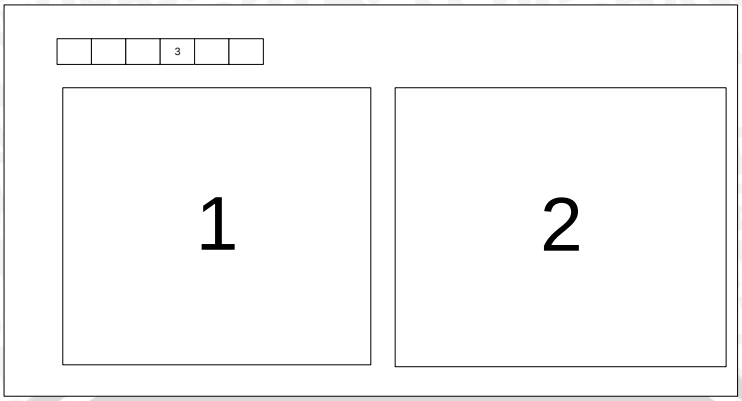


Gambar 3.15 Halaman Keempat

Pada halaman kelima user interface berisikan tampilan perhitungan probabilitas untuk pemilihan rute selanjutnya bagi setiap semut dan memori semut yang berisi daftar jalu yang telah dilalui. Halaman keempat dapat dilihat Gambar 3.16.

Keterangan pada Gambar 3.16

1. Hasil Perhitungan probabilitas tiap kota untuk pemilihan rute semut.
2. Isi memori semut tentang rute yangtelah dilewati.
3. Halaman *ACO.*

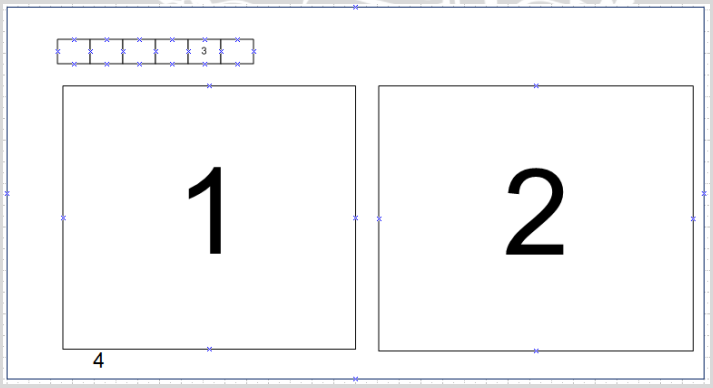


Gambar 3.16 Halaman Kelima

Pada halaman terakhir atau halaman keenam berisikan tampilan semua individu yang terbentuk dan populasi yang telah seleksi berdasarkan seleksi *elitism.* Selain itu, pada halaman keenam berisikan nama-nama urutan tempat wisata yang dikunjungi berdasarkan jarak minimum. Halaman keenam dapat dilihat pada Gambar 3.17.

Keterangan pada Gambar 3.17.

1. Tampilan semua individu yang terbentuk pada setiap iterasi.
2. Tampilan populasi hasil seleksi pada setiap iterasi.
3. Halaman hasil.
4. Nama urutan tempat wisata yang akan dikunjungi.



Gambar 3.17 Halaman Keenam

BAB IV IMPLEMENTASI

Pada bab implementasi ini merupakan tahap penerapan sistem, dimana aplikasi ini dijalankan pada keadaaanya sebenarnya penerapan sistem yang kaan dijelaskan terdiri dari lingkungan implementasi, implementasi aplikasi dan implementasi tampilan user interface.

4.1 Lingkungan Implementasi

Lingkungan implementasi merupakan ruang lingkup kebutuhan dalam penerapan sistem hibridisasi algoritma genetika dan algoritma koloni semut untu penentuan rute pariwisata Pulau Lombok. Terdapat dua lingkungan yangyang digunakan dalam membangun sistem aplikasi tersebut, yaitu lingkungan perangkat keras dan lingkungan perangkat lunak.

4.1.1 Lingkungan Perangkat Keras

Lingkungan perangkat keras yang digunakan dalam implementasi aplikasi hibridisasi algoritma genetika dan algoritma koloni semut untuk penentuan rute pariwisata Pulau Lombok adalah sebagai berikut :

1. Processor 2.4 GHz Intel Core i7.
2. Memory 8 GB
3. Hardisk dengan kapasista 1 TB.
4. VGA Card NVIDIA GEFORCE 750 M.

4.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam implementasi hibridisasi algoritma genetika dan algoritma koloni semut untuk penentuan rute pariwisata Pulau Lombok adalah sebagai berikut :

1. Sistem Operasi yang digunakan adalah Windows 8.1 Single Language 64 bit.
2. Editor yang digunakan adalah Visual Studio 2013.
3. Penyimpanan data yang digunakan adalah Microsoft Excel 2013.

4.2 Implementasi Program

Berdasarkan metodologi dan perancangan yang sistem yang telah dilakukan pada Bab 3, maka selanjutnya akan dijelaskan implementasi proses-proses tersebut kedalam bentuk sistem dengan bahasa pemrograman C#.

4.2.1 Membangkitkan Populasi Awal

Membangkitkan populasi dimulai dengan membangkitkan individu. Satu individu dibentuk melalui grup-grup gen yang berisikan nilai random biner. Panjang grup gen yang dibentuk ialah sepanjang jumlah destinasi tempat yang

pilih dan dikurangi satu. Sedangkan panjang kromosom sebuah individu disesuaikan dengan jumlah wisata yang dipilih oleh *user*. Setelah itu proses pembentulan dilakukan dengan menghitung angka satu dalam setiap grup gen, jika tidak ada angka satu dalam grup gen nilai gen tersebut diisikan dengan jumlah destinasi. Proses pembangkitan kromosom dalam sebuah individu dilakukan pengecekan agar tidak terdapat nilai gen yang sama pada setiap individunya. Proses pembangkitan populasi awal dapat dilihat pada Kode Program 4.1.

```
1  private void KromosomAwal()
2      {
3          int n = count - 1;
4          int[] temp = new int[n];
5          int[, ,] repKromosom = new int[count, n, n];
6          int[] number;
7          for (int i = 0; i < count; i++)
8          {
9              number = generateNumber(count);
10             for (int j = 0; j < n; j++)
11             {
12                 temp = callRandom(n, number[j]);
13                 //Console.Write(number[j] + " ");
14                 for (int k = 0; k < n; k++)
15                 {
16                     repKromosom[i, j, k] = temp[k];
17                     //Console.Write(repKromosom[i, j, k]);
18                 }
19                 //Console.Write(" ");
20             }
21             //Console.WriteLine();
22         }
```

Kode Program 4.1 Pembentukan Populasi Awal

1. Baris 3-6 merupakan pendefinisian kromosom dan gen
2. Baris 9 fungsi untuk menghitung nilai 1 dalam grup gen dari hasil random.
3. Baris 12-18 untuk melakukan proses random biner untuk setiap grup gen.

4.2.2 Proses *Crossover*

Metode *crossover* yang digunakan adalah *one-cut point.* Sebelum dilakukan proses *crossover,* maka dilakukan dulu pemilihan induk yang mengalami *crossover* sesuai dengan nilai *Pc* yang dimasukan. Proses *crossover* dapat dilihat pada Kode Program 4.2.

```
1  for (int i = 0; i < biner_pc.Length; i++)
```

```
2           {
3              if (biner_ pc [i] < pc)
4              {
5                 crossed[i] = 1;
6                 jumlah_cross++;
7              }
8              //Console.WriteLine("index cross " + biner_cr[i] + " " + i);
9           }
10 long jmlkombinasi;
11          long atas = 1;
12          for (int i = 1; i <= jumlah_cross; i++ )
13          {
14             atas = atas * i;
15          }
16          long bawah = 1;
17          for (int i = 1; i <= (jumlah_cross-2); i++)
18          {
19             bawah = bawah * i;
20          }
21          jmlkombinasi = Math.Abs(atas) / (2 * bawah);
22 int[,] cross_sementara = new int[jmlkombinasi, biner_cr.Length];
23                int flag = 0;
24                // perulangan untuk mengisi gen ke 1 dan gen 2 dalam setiap kromosom
25                for (int i = 0; i < biner_ pc.Length - 1; i++)
26                {
27                   if (biner_ pc [i] > pc) continue;
28                   for (int j = i + 1; j < biner_cr.Length; j++)
29                   {
30                      if (biner_ pc [j] > pc) continue;
31                      //Console.WriteLine("k : " + i);
32                      cross_sementara[flag, 0] = pop[i].getgen(0);
33                      cross_sementara[flag, 1] = pop[i].getgen(1);
34                      int flagchild = 2;
35                      //perulangan untuk mengisi gen 3, 4, dan 5
36                      for (int l = 0; l < count; l++)
37                      {
38                         if (pop[j].getgen(l) != cross_sementara[flag, 0]
39                            && pop[j].getgen(l) != cross_sementara[flag, 1])
40                         {
41                            cross_sementara[flag, flagchild] = pop[j].getgen(l);
42                            flagchild++;
43                         }
44                      }
```

```
45                  flag++;
46                }
47              }
```

Kode Program 4.2 Proses *Crossover*

1. Baris 1-9 merupakan perbandingan nilai *Pc* untuk pemilihan induk crossover.
2. Baris 10-21 untuk menghitung kombinasi individu yang dihasilkan dari proses crossover.
3. Baris 28-34 merupakan proses *crossover one-cut point* untuk mengambil dua gen pertama pada *parent 1*.
4. Baris 38-41 merupkan proses untuk mengisi kromosom pada induk baru yang berasal pada *parent 2*.

4.2.3 Proses Mutasi

Pada proses mutasi metode yang digunakan adalah *uniform transform*. Pada uniform transform proses pertama yang dilakukan adalah membangkitkan angka random sejumlah populasi awal. Kemudian memilih gen yang akan dilakukan proses mutasi. Proses mutasi dapat dilihat pada Kode Program 4.3.

```
1  for (int i = 0; i < biner_pm.GetLength(0); i++)
2          {
3             for (int j = 0; j < biner_pm.GetLength(1); j++)
4             {
5                biner_pm [i, j] = rand.NextDouble();
6             }
7          }
8  for (int j = 0; j < biner_pm.GetLength(1); j++)
9                {
10                  if (biner_pm [i, j] < mr && perulangan == 0)
11                  {
12
13                     if (j == count - 1)
14                     {
15                        mutasi_sementara[flag, j] = mutasi_sementara[flag, 0];
16                        mutasi_sementara[flag, 0] = cloning[flag, j];
17                     }
18                     Else
19                     {
20                        mutasi_sementara[flag, j] = cloning[flag, j + 1];
21                        mutasi_sementara[flag, j + 1] = cloning[flag, j];
22                     }
23                     biner_pm [i, j] = 1;
24                     j++;
```

```
25                  //}
26                  continue;
27              }
```

Kode Program 4.3 Proses Mutasi

1. Baris 1-7 merupakan proses pembangkitan nilai random sejumlah populasi awal
2. Baris 13-17 merupakan pertukaran gen yang terpilih dengan gen ayng berada didepannya.
3. Baris 18-22 proses pertukaran gen jika gen yang terpilih berada di gen terakhir, maka gen tersebut akan ditukarkan dengan gen hasil cloning pada gen pertama.

4.2.4 Proses *Ant Colony*

Pada proses untuk metode *ant-colony* proses yang dilakukan adalah dengan melakukan perhitungan untuk mencari nilai probabilitas tiap kota. Hasil hasil probabilitas tersebut maka dipilih yang nilainya paling besar untuk dikunjungi oleh semut. Kemudian terakhir ialah menampilkan isi memori memori dari setiap semut yang berisi urutan kota ang telah dikunjungi. Proses *ant-colony* dapat dilihat pada Kode Program 4.4.

```
1   alfabeta();
2           invers();
3           int bantuan = 1;
4           double[,] _aco = new double[jarak.Count, jarak.Count];
5           int[,] _memori= new int[jarak.Count,jarak.Count];
6           for(int i=0;i<_memori.GetLength(0);i++)
7           {
8              _memori[i, 0] = i + 1;
9           }
10          bool hitung = true;
11          dataGridView8.ColumnCount = _aco.GetLength(1);
12          dataGridView9.ColumnCount = _memori.GetLength(1);
13          Do
14          {
15             for (int x = 0; x < _aco.GetLength(0); x++)
16             {
17                DataGridView row = new DataGridView();
18                string[] row2 = new string[_memori.GetLength(1)];
19                for (int y = 0; y < _aco.GetLength(1); y++)
20                {
21                   for (int a = 0; a < _aco.GetLength(1); a++)
22                   {
23                      if (_memori[x, a] == y + 1)
```

```
24                      hitung = false;
25                  }
26                  if (hitung == false)
27                  {
28                      _aco[x, y] = 0;
29
30                  }
31                  else
32                      _aco[x, y] = (Math.Pow(feromonawal[x, y], _alfa) *
33                      Math.Pow(visibilitas[x, y], _beta)) / sigma(_memori, x);
34                  row2[y] = _aco[x, y].ToString();
35                  hitung = true;
36              }
37              dataGridView8.Rows.Add(row2);
38          }
39          for (int x = 0; x < _aco.GetLength(0); x++)
40          {
41              double max = 0;
42              int imax = 0;
43              for (int y = 0; y < _aco.GetLength(1); y++)
44              {
45                  if (max < _aco[x, y])
46                  {
47                      max = _aco[x, y];
48                      imax = y + 1;
49
50                  }
51                  _memori[x, bantuan] = imax;
52              }
53          }
54          for (int x = 0; x < _memori.GetLength(0); x++)
55          {
56              string[] row3 = new string[_memori.GetLength(1)];
57              for (int y = 0; y < _memori.GetLength(1); y++)
58              {
59                  row3[y] = _memori[x, y].ToString();
60              }
61              dataGridView9.Rows.Add(row3);
```

Kode Program 4.4 Proses *Ant Colony*

1. Baris 6-9 ialah untuk membuat panjang memori yang dimiliki setiap semut.
2. Baris 19-35 merupakan proses perhitungan probabilitas setiap kota yang akan dikunjungi oleh setiap semut.

3. Baris 43-52 ialah proses pemilihan kota yang akan dikunjungi dengan memilih probabilitasnya yang paling tinggi.
4. Baris 57-60 proses pengisian memori untuk semut untuk kota yang telah dikunjungi.

4.2.5 Perhitungan Jarak

Proses perhitungan dilakukan dengan menghitung total jarak yang diperlukan untuk mengunjungi setiap tempat pariwisata tepat satu kali. Perhitungan total jarak dilakukan untuk setiap individu dari populasi awal, algoritma genetika dan algorit koloni semut. Proses perhitungan total jarak dapat dilihat pada Kode Program 4.5.

```
1  for (int i = 0; i < gen2.GetLength(0); i++)
2          {
3             for (int j = 0; j < gen2.GetLength(1); j++)
4             {
5                gen2[i, j] = pop[i].getgen(j);
6             }
7             gen2[i, count] = pop[i].getgen(0);
8          }
9   for (int i = 0; i < gen2.GetLength(0); i++)
10         {
11            DataGridViewRow row2 = new DataGridViewRow();
12            row2.CreateCells(this.dataGridView10);
13
14            for (int j = 0; j < gen2.GetLength(1) - 1; j++)
15            {
16               total = total + jarak[gen2[i, j] - 1].getjarak(gen2[i, j + 1] - 1);
17               Console.WriteLine("Gen "+ sd + " " + gen2[i, j] + " " + gen2[i, j + 1]);
18               row2.Cells[j].Value = gen2[i, j].ToString();
19               gen_baru[i, j] = gen2[i, j];
20               sd++;
21            }
22            gen_baru[i, count] = total;
```

Kode Program 4.5 Proses Perhitungan Jarak

1. Baris 1-8 merupakan proses cloning untuk gen pertama pada setiap individu.
2. Baris 16-22 merupakan proses perhitungan total jarak untuk setiap individu sampai kembali ketempat semut berangkat.

4.2.6 Proses Seleksi

Proses seleksi yang dilakukan untuk implementasi ini adalah metode seleksi elitism. Seleksi dilakukan dengan memilih beberapa individu yang mempunyai total jarak minimum dari semua individu sejumlah populasi awal. Proses seleksi dapat dilihat pada Kode Program 4.6.

```
1   for (int i = 0; i < count; i++)
2           {
3               double min = double.MaxValue;
4               int mini = 0;
5               for (int j = 0; j < pop.Count; j++)
6               {
7                   if (min > pop[j].getTotalJarak() && flag[j] != 1)
8                   {
9                       min = pop[j].getTotalJarak();
10                      mini = j;
11                      flag[j] = 1;
12                  }
13
14              }
15              for (int k = 0; k < count; k++)
16              {
17                  potong_sementara[i, k] = pop[mini].getgen(k);
18              }
19              potong_sementara[i, count] = pop[mini].getTotalJarak();
20          }
```

Kode Program 4.6 Proses Seleksi

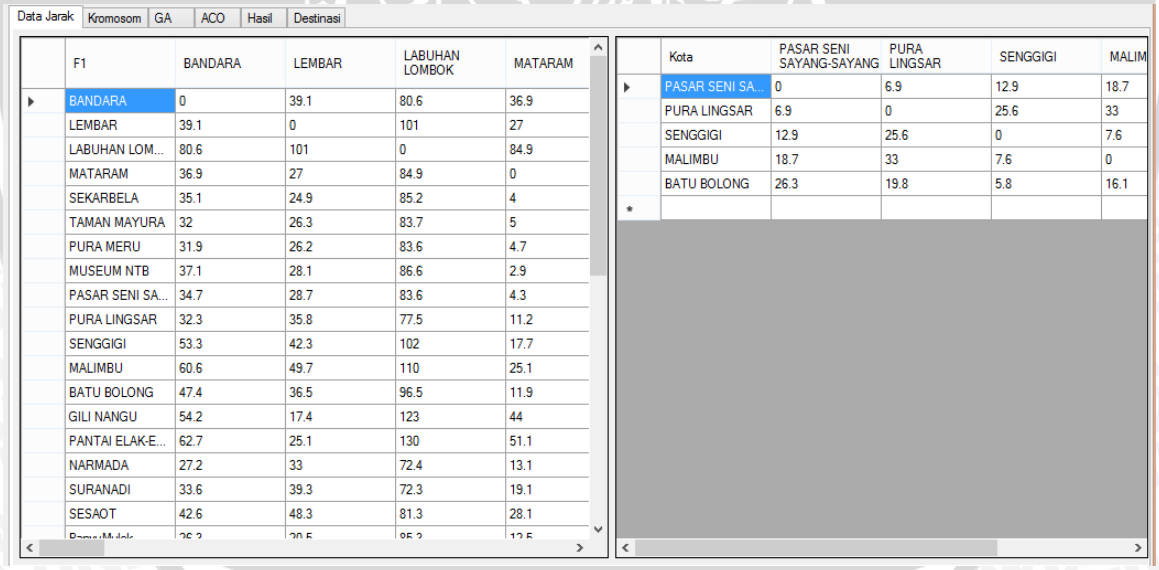
1. Baris 5-11 merupakan proses pencarian total jarak minimum dari semua individu yang terbentuk.
2. Baris 15-18 ialah proses pembentukan populasi baru dari individu-individu yang mempunyai jarak minimum.

4.3 Implementasi *User Interface* 4

Implementasi *user interface* untuk proses pencarian rute menggunakan algoritma genetika dan algoritma koloni semut ini menggunakan 6 halaman utama. Halaman pertama berisikan daftar nama tempat wisata Pulau Lombok. Kemudian pada halaman pertama juga berisikan parameter-paramter perhitungan dalam mencari rute wisita yang paling minimum. Pada Gambar 4.1 merupakan implementasi *user interface* untuk halaman pertama.

Gambar 4.1 *User Interface* halaman pertama

Selanjutnya pada halaman kedua *user interface* berisikan data jarak antar semua tempat wisata . selain itu pada halaman kedua juga berisikan matrix jarak berdasarkan tempat-temat wisata yang ingin dikunjungi oleh *user.* Pada Gambar 4.2 dapat dilihat hasil implementasi halaman kedua.

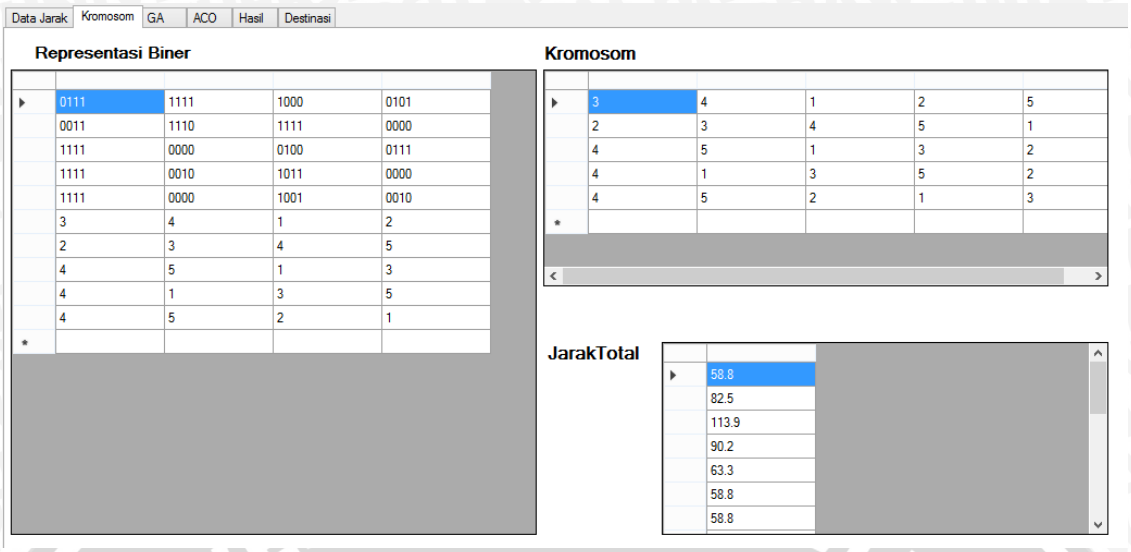


F1	BANDARA	LEMBAR	LABUHAN LOMBOK	MATARAM
BANDARA	0	39.1	80.6	36.9
LEMBAR	39.1	0	101	27
LABUHAN LOM...	80.6	101	0	84.9
MATARAM	36.9	27	84.9	0
SEKARBELA	35.1	24.9	85.2	4
TAMAN MAYURA	32	26.3	83.7	5
PURA MERU	31.9	26.2	83.6	4.7
MUSEUM NTB	37.1	28.1	86.6	2.9
PASAR SENI SA...	34.7	28.7	83.6	4.3
PURA LINGSAR	32.3	35.8	77.5	11.2
SENGGIGI	53.3	42.3	102	17.7
MALIMBU	60.6	49.7	110	25.1
BATU BOLONG	47.4	36.5	96.5	11.9
GILI NANGU	54.2	17.4	123	44
PANTAI ELAK-E...	62.7	25.1	130	51.1
NARMADA	27.2	33	72.4	13.1
SURANADI	33.6	39.3	72.3	19.1
SESAOT	42.6	48.3	81.3	28.1

Kota	PASAR SENI SAYANG-SAYANG	PURA LINGSAR	SENGGIGI	MALIM
PASAR SENI SA...	0	6.9	12.9	18.7
PURA LINGSAR	6.9	0	25.6	33
SENGGIGI	12.9	25.6	0	7.6
MALIMBU	18.7	33	7.6	0
BATU BOLONG	26.3	19.8	5.8	16.1

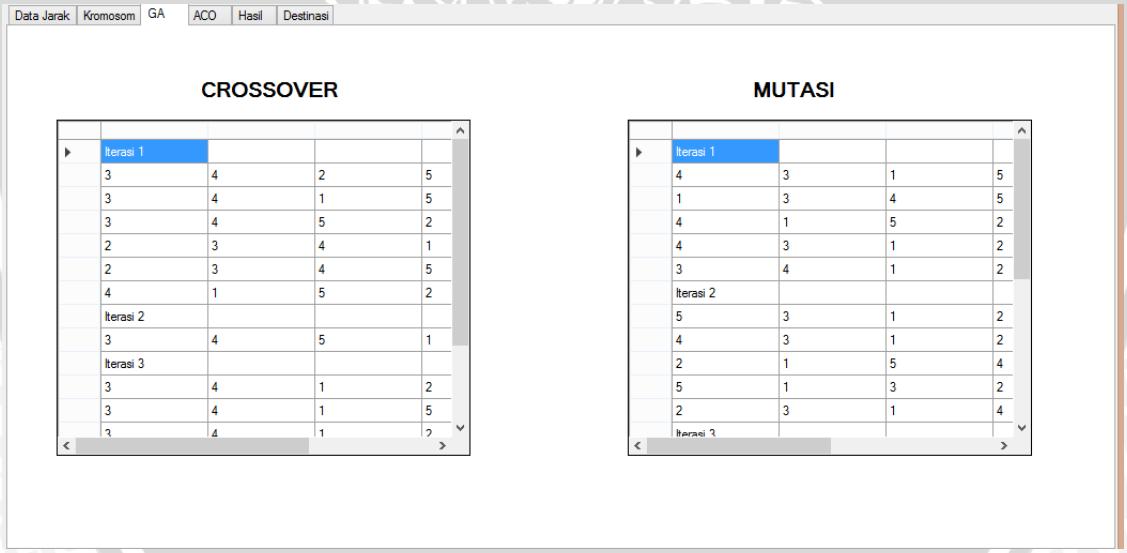
Gambar 4.2 *User Interface* halaman kedua

Pada halaman ketiga user interface terdapat hasil pemebentukan kromosom menggunakan grup gen yang diisikan dengan nilai random biner. Selain itu, pada halaman ketiga juga berisikan populasi awal yang terbentuk beserta total jarak yang didapat untuk setiap individu. Hasil implementasi halaman ketiga dapat dilihat pada Gambar 4.3.



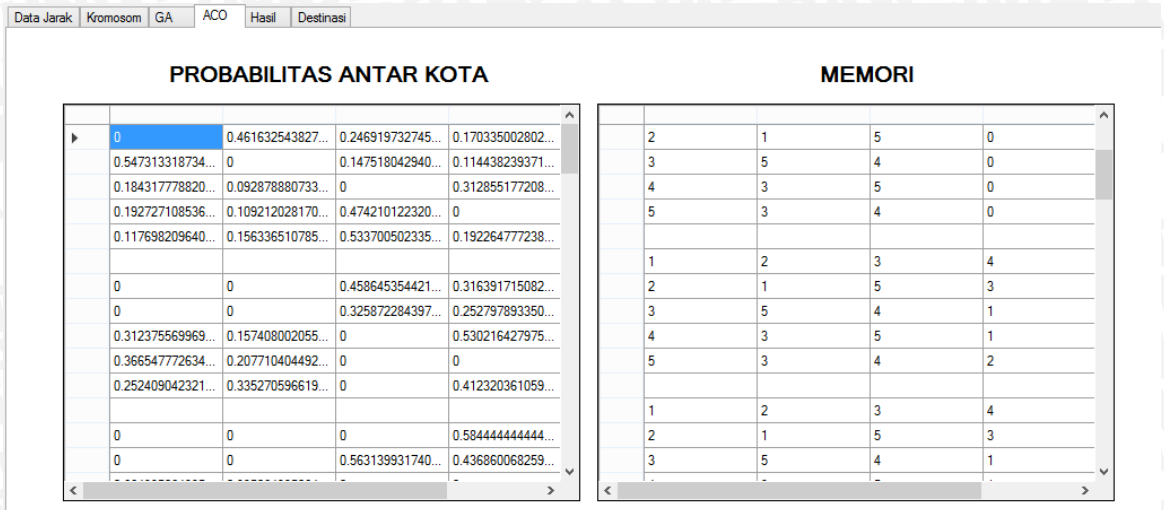
Gambar 4.3 *User Interface* halaman ketiga

Pada implementasi untuk halaman keempat berisikan hasil pembentukan individu baru untuk proses crossover dan mutasi. Hasil implementasi halaman keempat dapat dilihat pada Gambar 4.4.



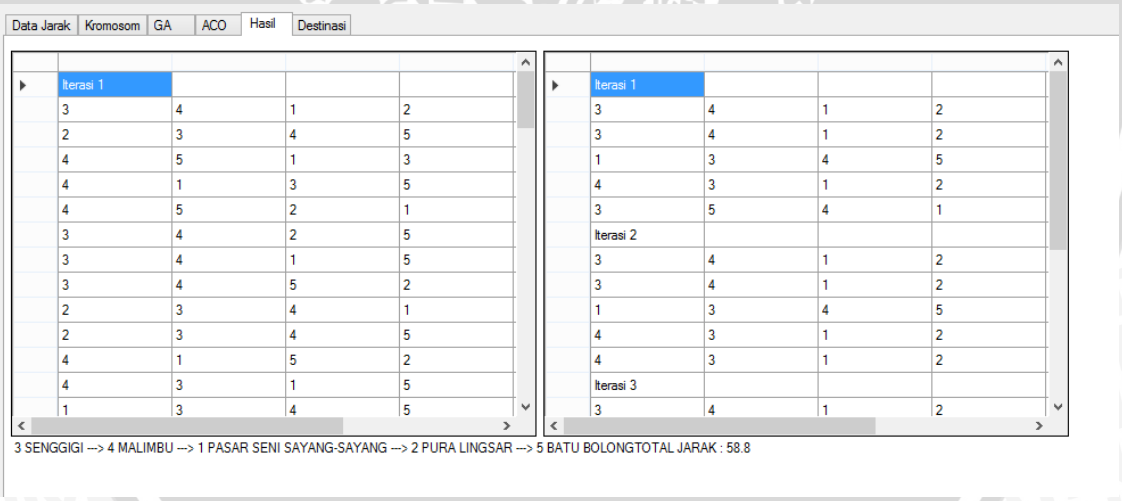
Gambar 4.4 *User Interface* proses Algoritma Genetika

Implementasi user interface untuk halaman kelima berisi hasil perhitungan probabilitas nilai setiap kota yang akan dikunjungi. Pada implementasi user interface halaman kelima juga berisikan memori yang berisikan daftar urutan tempat-tempat wisata yang akan dikunjungi. Hasil implementasi untuk halaman kelima dapat dilihat pada Gambar 4.5.



Gambar 4.5 *User Interface* proses Ant-Colony

Kemudian implementasi *user interface* untuk halaman terakhir berisi semua individu yang terbentuk untuk setiap iterasi dan hasil populasi baru yang didapat berdasarkan metode seleksi *elitism*. Pada halaman terakhir juga berisik individu terbaik beserta dengan total jarak yang dilalui. Hasil implementasi user interface halaman terakhir dapat dilihat pada Gambar 4.6.



Gambar 4. 6 *User Interface* halaman keenam

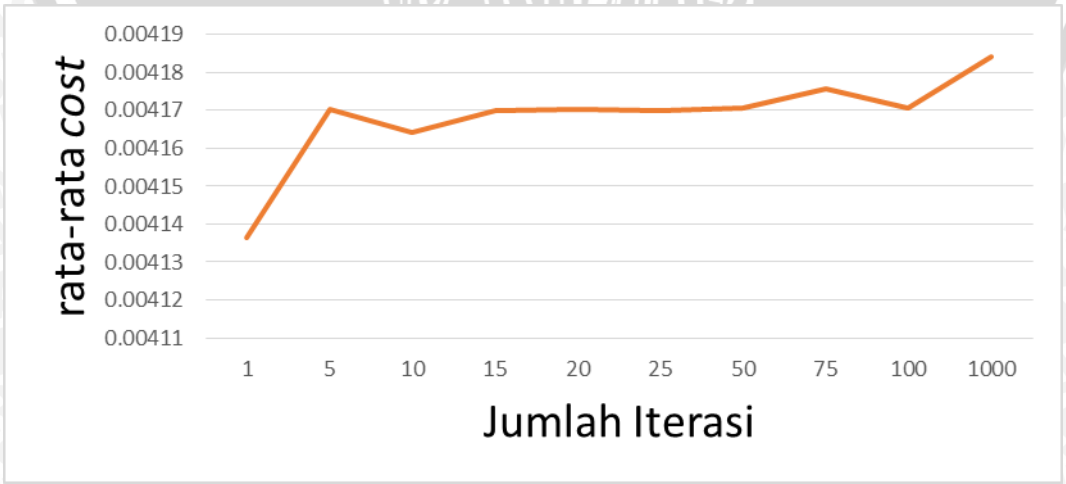
BAB V PENGUJIAN DAN ANALISIS

5.1 Pengujian dan Analisi Jumlah Iterasi

Pengujian ini dilakukan untuk mengetahui seberapa banyak generai yang optimal untuk pada permasalahan penentuan rute pariwisata Pulau Lombok menggunakan metode hiridisasi algoritma genetika dan algoritma koloni semut. Pengujian dilakukan sebanyak 10 kali dengan banyk generasi yang bervariasi 1, 5, 10, 15, 20, 25, 50, 75, 100, dan 1000 iterasi. Jumlah destinasi wisata yang digunakan adalah 8 destinasi. Kombinasi nilai *Pm* dan *Pc* yang digunakan adalah 0,5 : 0,5. Sedangkan untuk kombinasi nilai *Alfa* dan *Beta* yang digunakan adalah 1 : 1. Hasil dari percobaan akan didapatkan nilai rata-rata total jarak untuk tiap iterasi yang berbeda. Pada Tabel 5.1 dapat dilihat hasil pengujian untuk jumlah iterasi.

Tabel 5.1 Pengujian Jumlah Iterasi.

Jumlah Iterasi	Percobaan ke- *i*										Rata-rata *Cost*
	1	2	3	4	5	6	7	8	9	10	
1	242	242	242	241.4	242	242	242	242	241.8	241.8	0.004136
5	239	242	239	238.9	242	239	239	239	238.9	241.8	0.00417
10	242	239	239	239.1	239	239	242	242	241.8	239.1	0.004164
15	242	239	239	241.8	239	239	239	239	241.8	238.9	0.00417
20	242	239	239	238.9	239	239	239	239	241.8	241.8	0.00417
25	239	242	239	241.8	239	239	242	239	238.9	239.1	0.00417
50	242	239	239	238.9	242	239	239	239	238.9	241.8	0.004171
75	238.9	238.9	238.9	241.8	241.8	238.9	238.9	238.9	238.9	238.9	0.0041757
100	238.9	241.8	238.9	238.9	238.9	241.8	241.8	238.9	238.9	238.9	0.0041707
1000	238.9	238.9	239.1	239.1	239.1	238.9	238.9	239.1	238.9	239.1	0.0041841



Gambar 5.1 Grafik Hasil Pengujian Jumlah Iterasi

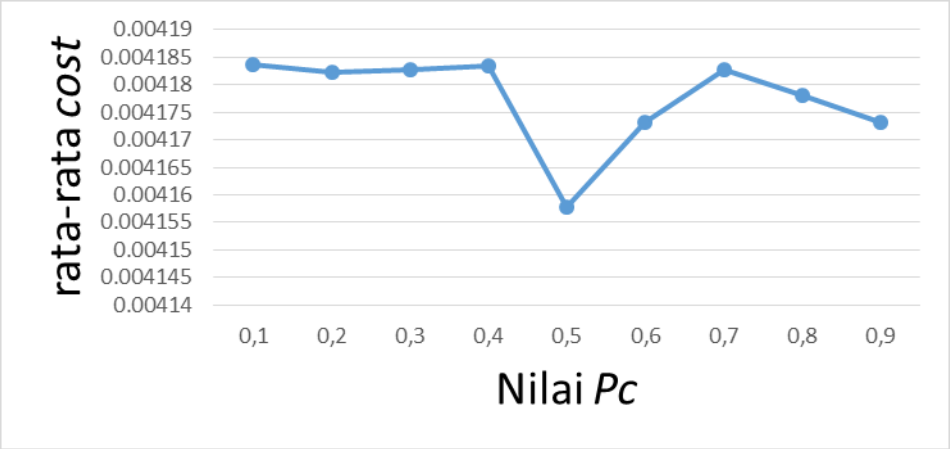
Pada Gambar 5.1 merupakan grafik hasil pengujian pengaruh jumlah iterasi. Pada grafik tersebut dapat terihat bahwa pengaruh jumlah iterasi dapat membuat metode yang digunakan semakin mendapat jarak yang optimal atau minimum. Dapat dilihat dari variasi jumlah iterasi yang digunakan bahwasannya dapat memperbaiki nilai minimum yang digunakan. Peningkatan yang cukup signifikan terlihat pada iterasi 1-5 dan 100-1000. Sedangkan untuk iterasi 5-50 tidak adanya perubahan yang signifikan sehingga terjadi konvergensi. Kondisi seperti ini mengakibatkan proses reproduksi untuk menghasilkan individu baru hampir sama dengan induknya. Dari hasil pengujian ini, terlalu banyak generasi belum tentu menghasilkan solusi yang paling optimal. Karena dengan jumlah iterasi yang banyak solusi yang dihasilkan tidak terlalu signifikan, tetapi solusi yang banyak membuat proses perhitungan menjadi lebih lama. Hal ini dapat terlihat dari grafik yang dihasilkan, pola yang terbentuk dari jumlah iterasi yang berbeda-beda tidak terlalu signifikan atau setara.

5.2 Pengujian dan Analisis Nilai *Pc*

Pada pengujian ini dilakukan untuk mengukur nilai *Pc* yang tepat untuk permasalahan penentuan rute pariwisata Pulau Lombok dengan menggunakan metode hibridisasi algoritma genetika dan algoritma koloni semut. Pengujian ini dilakukan sebanyak 9 kali percobaan pada jumlah iterasi sebanyak 4. Sedangkan kombinasi nilai Alfa dan Beta yang digunakan adalah 1 : 1. Pada nilai *Pm* digunakan nilai *default* yang telah diset sebelumnya yaitu 0,5. Populasi yang digunakan adalah 8 tempat wisata. Dari hasil setiap percobaan didapatkan rata-rata nilai jarak optimal untuk setiap variasi niali *Pc*. Pada pengujian ini variasi nilai *Pc* yang digunakan adalah 0,1-0,9. Hasil pengujian pengaruh nilai *Pc* dapat dilihat pada Tabel 5.2.

Tabel 5.2 Hasil Pengujian Nilai *Pc*

Nilai *Pc*	Percobaan ke-*i*										Rata-rata Cost
	1	2	3	4	5	6	7	8	9	10	
0,1	239.1	239.1	239.1	239.1	239.1	239.1	239	238.9	238.9	238.9	0.00418
0,2	239.1	239.1	239.1	239.1	239.1	239.1	239	239.1	239.1	239.1	0.00418
0,3	239.1	239.1	239.1	239.1	239.1	239.1	239	239.1	239.1	239.1	0.00418
0,4	239.1	239.1	239.1	239.1	239.1	239.1	239	238.9	238.9	238.9	0.00418
0,5	243.1	241.3	239.1	241.8	241.8	238.9	239	239.1	241.8	239.1	0.00416
0,6	239.1	239.1	239.1	239.1	241.8	238.9	239	241.8	239.1	239.1	0.00417
0,7	239	239.1	239.1	239.1	239.1	239.1	239	239.1	239.1	238.9	0.00418
0,8	238.9	239.1	239.1	239.1	239.1	241.8	239.1	239.1	239.1	239.1	0.004178
0,9	241.8	239.1	239.1	239.1	239.1	239.1	238.9	239.1	241.8	239.1	0.0041733



Gambar 5.2 Grafik Hasil Pengujian Nilai *Pc*

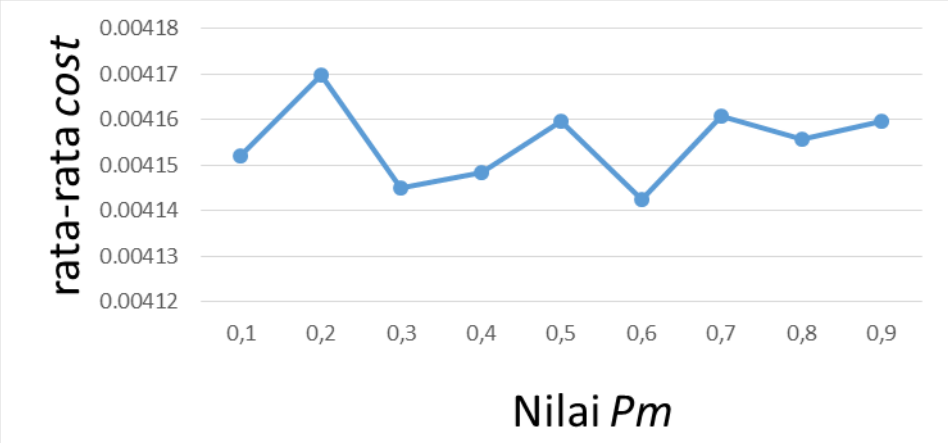
Pada Gambar 5.2 merupakan grafik hasil pengujian pengaruh nilai *Pc* yang dilakukan sebanyak 9 kali. Rata-rata nilai total jarak didapatkan cukup bervariasi. Pada nilai *Pc* dengan rentang 0,1-0,4 nilai jarak yang didapatkan tidak bepengaruh besar dalam mendapatkan total jarak yang optimal. Pada nilai *Pc* 0,5 merupakan nilai terburuk untuk mendapatkan nilai jarak yang paling minimum. Kemudian untuk nilai *Pc* 0,7-0,9 nilai jarak yang didapatkan semakin membesar. Sedangkan untuk kasus *TSP* seperti ini yang diharapkan adalah jarak yang seminimum mungkin. Solusi optimal yang didapat dari pengujian diatas adalah pada percobaan nilai *Pc* 0,1. Nilai *cost* relatif tinggi tersebut didapatkan dari seringnya proses *crossover* serta pemilihan metode seleksi yang digunakan. Dapat dilihat pada grafik, untuk nilai *Pc* 0,1 - 0,4 pola yang dihasilkan hampir sama. Sehingga proses *crossover* yang banyak dan beragam menentukan dalam menghasilkan induk anakan yang lebih baik, tetapi harus di dukung dengan proses seleksi. Sebab dengan proses seleksi *elitism* induk-induk yang mengalami proses *crossover* adalah induk terbaik dari populasi sebelumnya.

5.3 Pengujian dan Analisa Nilai *Pm*

Pada pengujian ini dilakukan untuk mengukur nilai *Pm* yang tepat untuk permasalahan penentuan rute pariwisata Pulau Lombok dengan menggunakan metode hibridisasi algoritma genetika dan algoritma koloni semut. Pengujian ini dilakukan sebanyak 9 kali percobaan pada jumlah iterasi sebanyak 4. Sedangkan kombinasi nilai Alfa dan Beta yang digunakan adalah 1 : 1. Pada nilai *Pc* digunakan nilai *default* yang telah diset sebelumnya yaitu 0,5. Populasi yang digunakan adalah 8 tempat wisata. Dari hasil setiap percobaan didapatkan rata-rata nilai jarak optimal untuk setiap variasi niali *Pc.* Pada pengujian ini variasi nilai *Pm* yang digunakan adalah 0,1-0,9. Hasil pengujian pengaruh nilai *Pm* dapat dilihat pada Tabel 5.3.

Tabel 5.3 Hasil Pengujian Nilai *Pm*

Nilai *Pm*	Percobaan ke-*i*										Rata-rata Cost
	1	2	3	4	5	6	7	8	9	10	
0,1	241.3	241.8	238.9	241.8	238.9	238.9	241	241.8	241.8	241.8	0.00415
0,2	239.1	241.8	239.1	239.1	239.1	241.8	239	239.1	238.9	241.3	0.00417
0,3	241.3	241.4	241.8	241.8	241.3	241.4	241	241.8	239.1	241.3	0.00415
0,4	239.1	241.3	241.3	241.8	241.3	239.1	242	241.8	241.8	241.3	0.00415
0,5	241.8	239.1	239.1	241.8	239.1	241.8	242	241.3	239.1	239.1	0.00416
0,6	241.3	241.8	241.8	241.8	241.4	241.8	241	239.1	241.8	241.8	0.00414
0,7	239.1	238.9	241.8	241.8	241.4	241.8	239.1	241.3	239.1	239.1	0.0041608
0,8	241.3	241.4	241.8	239.1	239.1	239.1	241.8	241.8	239.1	241.8	0.0041558
0,9	241.8	239.1	239.1	241.8	239.1	241.8	241.8	241.3	239.1	239.1	0.0041597



Gambar 5.3 Grafik Pengujian Nilai *Pm*

Pada Gambar 5.3 merupakan grafik hasil pengujian nilai *Pm* yang dilakukan sebanyak 9 kali. Hasil grafik dapat dilihat perubahan jarak yang didapatkan sangat bervariasi. Pada Gambar 5.3 dapat diambil nilai terburuk untuk total jarak didapatkan adalah untuk nilai *Pm* 0,6. Sedangkan untuk nilai pada rentang 0,7-0,9 tidak terlalu banyak perubahan. Dari Gambar 5.3 dapat disimpulkan bahwa nilai *Pm* untuk mendapatkan hasil yang optimal yaitu pada 0,2. Sehingga pemilihan parameter nilai *Pm* tidak menentukan dalam menghasilkan nilai *cost* yang baik. Hal ini terlihat dari pola grafik yang dinamis. Nilai parameter yang kecil tetapi jika kemungkinan nilai proses mutasi lebih besar maka hasil yang dihasilkan akan lebih bervariasi dan beragam. Tetapi harus ada metode seleksi yang tepat, agar proses mutasi dapat dilakukan dari individu-individu yang terbaik dalam proses sebelumnya.

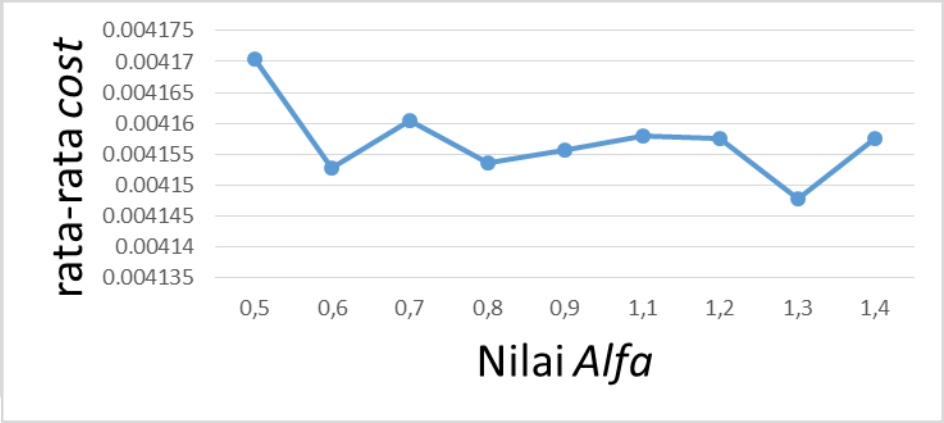
5.4 Pengujian dan Analisis Nilai *Alfa*

Pada pengujian ini dilakukan untuk mengukur nilai *alfa* yang paling tepat untuk masalah penentuan rute pariwisata Pulau Lombok dengan menggunakan metode hibridisasi algoritma genetika dan algoritma koloni semut. Pengujian ini

dilakukan sebanyak 9 kali percobaan. Nilai *alfa* yang digunakan percobaan bervariasi yaitu 0,5 - 0,9 dan 1,1 – 1,4. Sedangkan untuk nilai kombinasi *Pc* dan *Pm* digunakan 0,5 : 0,5. Pada nilai *beta* diset *default* yaitu 1. Pada jumlah destinasi dan jumlah iterasi yang digunakan adalah 8 dan 4. Hasil uji coba nilai *alfa* dapat dilihat pada Tabel 5.4.

Tabel 5.4 Hasil Pengujian Nilai *Alfa*

Nilai *Alfa*	Percobaan ke-*i*										Rata-rata Cost
	1	2	3	4	5	6	7	8	9	10	
0,5	239.1	239.1	239.1	238.9	241.3	241.8	238.9	238.9	238.9	241.8	0.00417049
0,6	241.8	241.4	241.8	239.1	241.3	239.1	241.3	241.8	241.3	239.1	0.004152824
0,7	238.9	239.1	239.1	239.1	241.8	238.9	241.8	241.3	241.8	241.8	0.004160426
0,8	241.8	241.3	238.9	239.1	241.4	241.8	241.3	241.3	238.9	241.8	0.004153514
0,9	241.8	238.9	241.8	241.8	239.1	239.1	238.9	241.4	241.8	241.8	0.004155585
1,1	241.4	241.8	238.9	241.3	241.3	241.8	238.9	238.9	238.9	241.8	0.004158004
1,2	241.8	238.9	241.3	238.9	241.8	241.3	241.8	239.1	241.3	239.1	0.004157486
1,3	241.3	241.8	241.8	241.8	241.3	239.1	238.9	241.8	241.8	241.3	0.004147829
1,4	238.9	241.3	239.1	241.3	241.8	241.8	239.1	241.3	241.8	238.9	0.004157486



Gambar 5.4 Grafik Pengujian Nilai *Alfa*

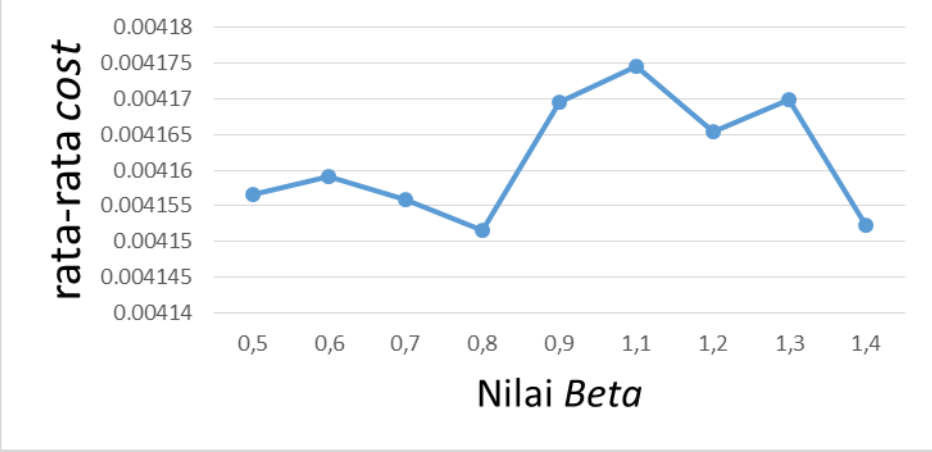
Pada Gambar 5.4 merupakan grafik hasil pengujian nilai *alfa* yang dilakukan sebanyak 9 kali. Pada grafik dapat dilihat bahwa terdapat perubahan signifikan yang buruk pada nilai *alfa* 1,3. Pada nilai rata-rata yang didapat untuk rentang nilai *alfa* 0,8 – 1,2 cenderung membaik, tetapi perubahan yang didapat tidak terlalu signifikan. Pada nilai *alfa* 0,5 didapatkan nilai total jarak yang paling minimum. Solusi untuk pengujian ini didapatkan pada nilai alfa 0,5 dengan rata-rata nilai *cost* yang didapatkan adalah 0,00417. Pengaruh nilai *alfa* cukup bervariasi dalam hasil perhitungan. Sehingga nilai parameter *alfa* yang bervariasi tidak dapat menghasilkan grafik yang pasti, ini terlihat dari pola yang selalu dinamis. Hal ini menyebabkan nilai parameter, jumlah destinasi dan jarak yang yang dihasilkan tidak memiliki pola tertentu tergantung dengan besarnya maslah yang digunakan[Boko, et al 2011].

5.5 Pengujian dan Analisis Nilai *Beta*

Pada pengujian ini dilakukan untuk mengukur nilai *beta* yang paling tepat untuk masalah penentuan rute pariwisata Pulau Lombok dengan menggunakan metode hibridisasi algoritma genetika dan algoritma koloni semut. Pengujian ini dilakukan sebanyak 9 kali percobaan. Nilai *beta* yang digunakan percobaan bervariasi yaitu 0,5 - 0,9 dan 1,1 – 1,4. Sedangkan untuk nilai kombinasi *Pc* dan *Pm* digunakan 0,5 : 0,5. Pada nilai *alfa* diset *default* yaitu 1. Pada jumlah destinasi dan jumlah iterasi yang digunakan adalah 8 dan 4. Hasil uji coba nilai *beta* dapat dilihat pada Tabel 5.5.

Tabel 5.5 Hasil Pengujian Nilai *Beta*

Nilai Beta	Percobaan ke-*i*										Rata-rata Cost
	1	2	3	4	5	6	7	8	9	10	
0,5	238.9	241.8	241.8	239.1	239	239	241.3	241.3	241.8	241.8	0.004157
0,6	241.8	238.9	241.8	239.1	239	242	241.8	239.1	241.8	239.1	0.004159
0,7	239.1	241.3	241.3	241.8	239	242	241.8	241.8	239.1	239.1	0.004156
0,8	238.9	241.8	241.8	241.3	241	242	239.1	239.1	241.8	241.8	0.004152
0,9	238.9	239.1	241.8	238.9	239	239	241.8	239.1	239.1	241.8	0.00417
1,1	238.9	238.9	238.9	239.1	241	239	241.8	239.1	239.1	239.1	0.004175
1,2	239.1	241.8	241.4	241.3	239	239	239.1	239.1	241.8	239.1	0.004165
1,3	239.1	239.1	241.8	241.8	241	239	238.9	238.9	238.9	239.1	0.00417
1,4	241.3	241.8	241.3	241.8	239	242	238.9	241.4	239.1	241.8	0.004152



Gambar 5.5 Grafik Hasil Pengujian Nilai *Beta*

Pada Gambar 5.5 merupakan grafik hasil pengujian nilai Beta yang dilakukan sebanyak 9 kali. Pada grafik dapat dilihat bahwa nilai rata-rata yang didapat sangat bervariasi terutama untuk nilai pada rentang 0,7 – 1,3. Dan untuk nilai beta pada 0,8 merupakan rata-rata yang paling buruk. Solusi yang paling optimal dari hasil percobaan nilai *beta* adalah untuk nilai *beta* 1,1 dengan rata-rata nilai *cost* yang didapat adalah 0,0041. Pengaruh nilai *beta* menghasilkan nilai jarak

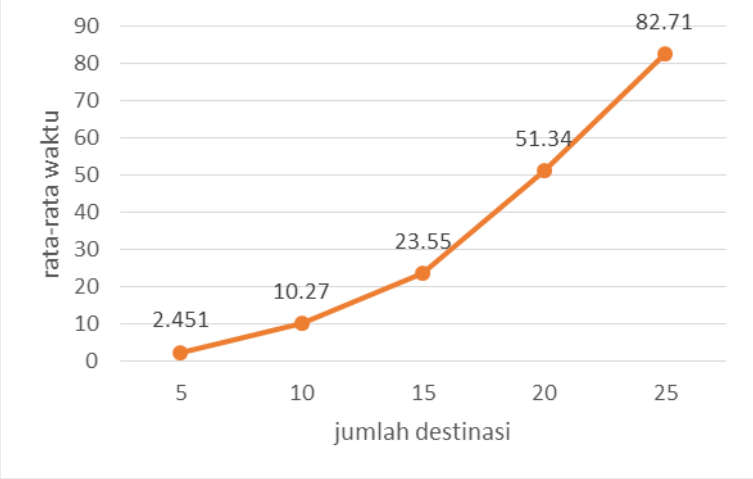
yang bervariasi, hal ini terlihat dalam grafik. Sehingga tidak ada pemilihan nilai parameter dalam perhitungan untuk mencari nilai jarak minimum[Boko, et al 2011].

5.6 Pengujian dan Analisis Terhadap Jumlah Destinasi

Pada pengujian ini akan dilakukan untuk mengetahui banyaknya populasi dengan solusi terbaik pada permasalahan pencarian rute pariwisata Pulau Lombok dengan menggunakan hibridisasi algoritma genetika dan algoritma koloni semut. Pengujian ini dilakukan dengan 5 kali percobaan. Pada parameter nilai *Pc, Pm, alfa, beta* dan jumlah iterasi semua diisi dengan nilai *default* yaitu 0,5, 0,5, 1, 1 dan 4. Hasil dari setiap percobaan akan didapatkan rata-rata nilai jarak yang paling optimal untuk setiap populasi yang berbeda. Hasil uji coba jumlah destinasi dapat dilihat pada Gambar 5.6.

Tabel 5.6 Hasil Pengujian Jumlah Iterasi

Jumlah Destinasi	Percobaan ke- *i*										Rata-rata (dalam detik)
	1	2	3	4	5	6	7	8	9	10	
5	2.4	2.45	2.46	2.6	2.55	2.4	2.4	2.45	2.4	2.4	2.451
10	10.3	10.45	9.9	10.05	10.2	10.2	10.6	10.25	10.85	9.9	10.27
15	24.4	24	24.6	24.9	24.35.	22.8	22.5	21.5	21.45	25.8	23.55
20	50.7	52.15	51.55	50.8	50.5	52.45	53.1	49.9	50.7	51.55	51.34
25	81.55	82.45	84.55	85.15	80.8	80.45	83.3	81.6	84.55	82.7	82.71



Gambar 5.6 Grafik Hasil Pengujian Jumlah Destinasi

Pada Gambar 5.6 merupakan grafik hasil pengujian untuk jumlah destinasi yang berbeda-beda. Pada grafik tersebut dapat dilihat bahwa semakin banyak jumlah destinasi yang digunakan maka semakin besar pula waktu yang dibutuhkan untuk mencari solusi yang optimal. Grafik tersebut menunjukan bahwa jumlah destinasi berbanding lurus dengan waktu komputasi sistem.

BAB VI KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan hasil pengujian pada penerapan hibridisasi algoritma genetika dan algoritma koloni semut untuk penentuan rute pariwisata Pulau Lombok, terdapat beberapa kesimpulan yaitu :

1. Hibridisasi algoritma genetika dan algoritma koloni semut dapat menyelesaikan masalah TSP untuk pencarian rute pariwisata Pulau Lombok. Implementasi pertama dilakukan pembentukan representasi kromosom berdasarkan grup gen yang setiap grup gennya diisikan dengan random biner. Selanjutnya proses reproduksi dengan menggunakan metode *one-cut point* dan *uniform transform.* Sedangkan pada proses algoritma koloni semut dengan mencari probabilitas setiap kunjungan tempat wisata yang ada dan kemudian disimpan ke dalam memori. Kemudian mengumpulkan hasil populasi awal, reproduksi, dan memori untuk mencari nilai total jarak masing-masing individu. Hasil terbaik yang diperoleh yaitu total jarak minimum. Hasil akhirnya adalah berupa urutan tempat wisata yang memiliki total jarak minimum.

2. Perubahan parameter algoritma genetika dan algoritma koloni semut rata-rata mempengaruhi hasil yang didapatkan. Pada data yang berukuran besar, hibridisasi algoritma genetika dan algoritma koloni semut memperoleh solusi yang optimal dengan waktu proses yang sebanding. Solusi yang optimal diperoleh jika rata-rata hasil akhir mencapai terjadinya konvergensi. Nilai optimal yang didapatkan dari penelitian ini terdapat pada jumlah iterasi sebanyak 75 iterasi, untuk kombinasi nilai *Pc* dan *Pm* adalah 0.1 : 0.2, serta kombinasi nilai alfa dan beta adalah 0.5 : 1.4. Hal ini dapat dikatakan bahwa kunjungan ideal di Pulau Lombok dengan mengunjungi 5 tempat wisata dalam sehari.

6.2 Saran

Berdasarkan hasil penelitian mengenai penerapan hibridisasi algoritma genetika dan algoritma koloni semut untuk penentuan rute jalur wisata di Pulau Lombok, terdapat beberapa saran yaitu :

1. Pada penelitian lebih lanjut dapat dikembangkan untuk menyelesaikan permasalahan rute jalur pariwisata menggunakan metode *crossover*, mutasi, dan seleksi yang lainnya. Sedangkan untuk algoritma koloni semut dapat dikembangkan untuk parameter nilai *alfa* dan *beta.* Hal ini dapat mempengaruhi nilai akhir yang didapat sehingga dapat memungkinkan untuk mendapat jarak yang lebih optimal.

2. Pada penelitian yang lebih lanjut agar menambahkan jumlah objek wisata yang dapat dikunjungi dan jarak yang variatif dengan memperhatikan kondisi jalan yang dilalui, waktu buka tutup tempat wisata dan kondisi jalan yang satu arah ataupun arah berlawanan.
3. Pada penelitian yang lebih lanjut dapat dikembangkan dengan menambahkan lama waktu untuk mengunjungi sebuah tempat wisata. Hal ini dilakukan agar dapat menambah jumlah objek yang dikunjungi dalam suatu waktu.

DAFTAR PUSTAKA

[Andri, et al 2015]. Aplikasi *Travelling Salesman Problem* Dengan Metode *Artifcial Bee Colony*. Indonesia. STMIK Mikroskil.

[Riyanti, Eka 2004]. Penerapan Algoritma *Branch And Bound* Untuk Penentuan Rute Objek Wisata. Indonesia. Jakarta.

[Rachmayadi, Tedy 2010]. Pencarian Solusi *TSP(Travelling Salesman Problem)* Menggunakan Algoritma Genetika. Bandung. ITB

[Boko et al, 2011]. Aplikasi Algoritma *Ant System (As)* Dalam Kasus *Travelling Salesman Problem(TSP)*. Indonesia. UIN Syarif Hidayatullah. Indonesia. Yogyakarta.

[I'ing et al 2007]. Pencarian Jalur Terpendek Menggunakan Algoritma Semut. Seminar NAsional Aplikasi Teknologi Informasi 2007(SNATI 2007). Indonesia. Yogyakarta

[Hardjasutanto, F 2013]. Penerapan Algoritma Semut untuk Pencarian Jalur Terpendek. Indonesia. ITB

[Zukhri dan Paputungan, 2013] Zukhri dan Paputungan, september 2013 A *Hybrid Optimization Algorithm Based on Genetic Algorithm And Ant Colony Optimization* dalam International Journal of Artificial Intelligence & Applications halaman 63-75.

[WAN-2014] Pei-dong Wan, Gong-ou Tang, Yang Li, dan Xi-Xin Yang, 2014. *Improved Ant Colony Algorithm for Traveling Salesman Problems* dalam 2012 24th Chinese Control and Decision Confrence halaman 660-664.

[GUP-2013] Gupta, Saloni dan Panwar, Poonam, 2013. *Solving Travelling Salesman Problem Using Genetic Algorithm* dalam International Journal of Advanced Research in Computer Science and Software Engineering halaman 376 – 380.

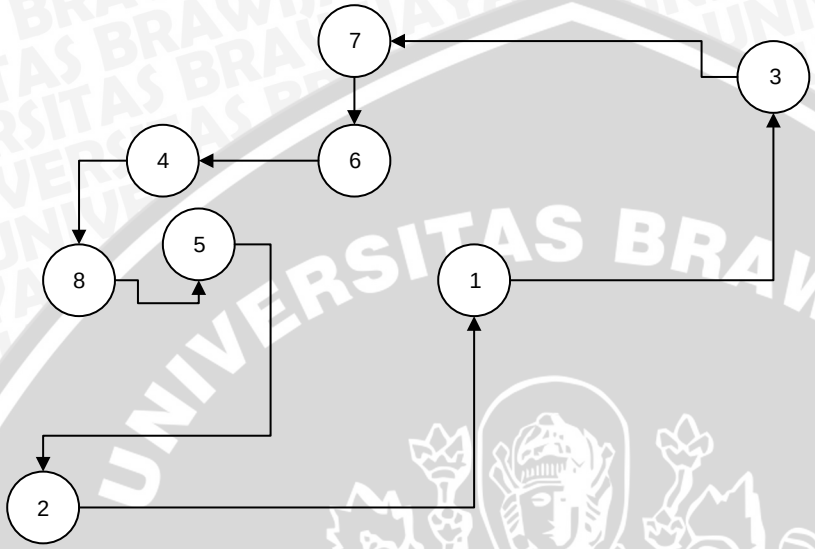
[BAP- 2014] Badan Perencanaan Pembangunan Daerah Provinsi Nusa Tenggara Barat. Nusa Tenggara Barat dalam Angka 2014. Badan Perencanaan Pembangunan Daerah Provinsi Nusa Tenggara Barat. Tahun 2014.

[ZEO-12] http://zero-fisip.web.unair.ac.id/artikel_detail-69833-ArtikelLainOperasiAlgoritmaGenetika.html. diakses tanggal 22 Oktober 2015 . pukul 20.00 WIB

[YAY - 2012] Penerapan Algoritma Genetika Dalam Penyelesaian *Travelling Salesman Problem With Precedence Constraints (Tsppc)*

LAMPIRAN

LAMPIRAN 1 JALUR TERBAIK PADA HASIL PENGUJIAN



Keterangan :

1. Bandara
2. Lembar
3. Labuhan Lombok
4. Mataram
5. Sekarbela
6. Taman Mayura
7. Pura Meru
8. Museum NTB

LAMPIRAN 2 DATA JARAK TEMPAT WISATA

	BANDARA	LEMBAR	LABUHAN LOMBOK	MATARAM	SEKARBELA	TAMAN MAYURA	PURA MERU	MUSEUM NTB
BANDARA	0	39.1	80.6	36.9	35.1	32	31.9	37.1
LEMBAR	39.1	0	101	27	24.9	26.3	26.2	28.1
LABUHAN LOMBOK	80.6	101	0	84.9	85.2	83.7	83.6	86.6
MATARAM	36.9	27	84.9	0	4	5	4.7	2.9
SEKARBELA	35.1	24.9	85.2	4	0	5.7	5.00	2.9
TAMAN MAYURA	32	26.3	83.7	5	5.7	0	0.1	6.1
PURA MERU	31.9	26.2	83.6	4.7	5.00	0.1	0	6.1
MUSEUM NTB	37.1	28.1	86.6	2.9	2.9	6.1	6.1	0
PASAR SENI SAYANG-SAYANG	34.7	28.7	83.6	4.3	7.3	3.3	3.3	7.1
PURA LINGSAR	32.3	35.8	77.5	11.2	23	6.9	6.9	12.8
SENGGIGI	53.3	42.3	102	17.7	19.3	22.1	22.1	15.9
MALIMBU	60.6	49.7	110	25.1	26.7	29.6	29.6	23.5
BATU BOLONG	47.4	36.5	96.5	11.9	13.5	16.2	16.2	10.1
GILI NANGU	54.2	17.4	123	44	40.9	41.9	41.9	43.1
PANTAI ELAK-ELAK	62.7	25.1	130	51.1	48.4	49.5	49.5	50.6
NARMADA	27.2	33	72.4	13.1	13.4	8.5	8.5	14.2
SURANADI	33.6	39.3	72.3	19.1	19.6	14.8	14.8	22
SESAOT	42.6	48.3	81.3	28.1	28.6	23.8	23.8	31
BanyuMulek	26.3	20.5	85.3	12.5	10.6	11.8	11.8	13.7
GILI AIR, MENO, TRAWANGAN	60.6	52.7	113	26	29	29.3	29.3	28.8
SENARU	115	113.9	74.8	87.3	90.3	90.8	90.8	90.1
SEMBALUN	153.6	152.5	68.6	119	122	122.3	122.3	121.8
sendang gile	118	116	76.9	89.4	92.4	92.7	92.7	92.2
PUSUK	50.7	45.4	120	16.1	19.1	19.4	19.4	19
BAYAN	108.3	107	67.9	80.4	83.4	83.7	83.7	83.2
DESA SADE	10.7	46.9	82.8	43.4	42.4	39.4	39.4	44.6
PANTAI AAN	23.4	49.8	96.1	56.1	55.7	51.7	51.7	57.2
KUTA	18.3	46.9	91	51	50	46	46	52.2
PENUJAK	4.4	34.8	75.9	31.3	30.2	27.2	27.2	32.4
seger	22.6	49	96.2	55.3	54.2	50.2	50.2	56.5
Mawun	26.2	35.5	95	50.3	53.8	58.6	58.6	61.1
selong belanak	23.6	26.5	94.1	49.3	70	65.5	65.5	71.5
PANTAI PINK	48.6	87.3	77.6	80.4	73.8	69.9	69.9	76
TANJUNG RINGGIT	51	86.2	79.9	82.7	76.1	72.3	72.3	78.3

PANTAI CEMARA	44.2	79.5	73.2	76	75.4	65.6	65.6	71.6
tanjung beloam	49.4	84.7	78.4	81.2	74.6	70.7	70.7	76.8
TETE BATU	47.7	63.3	79.7	47.1	44	40.1	40.1	45.8
GiliKondo	83.9	104	7.00	89.9	87.2	82.9	82.9	88.6

	PASAR SENI SAYANG-SAYANG	PURA LINGSAR	SENGGIGI	MALIMBU	BATU BOLONG	GILI NANGU
BANDARA	34.7	32.3	53.3	60.6	47.4	54.2
LEMBAR	28.7	35.8	42.3	49.7	36.5	17.4
LABUHAN LOMBOK	83.6	77.5	102	110	96.5	123
MATARAM	4.3	11.2	17.7	25.1	11.9	44
SEKARBELA	7.3	23.00	19.3	26.7	13.5	40.9
TAMAN MAYURA	3.3	6.9	22.1	29.6	16.2	41.9
PURA MERU	3.3	6.9	22.1	29.6	16.2	41.9
MUSEUM NTB	7.1	12.8	15.9	23.5	10.1	43.1
PASAR SENI SAYANG-SAYANG	0	6.9	12.9	18.7	26.3	44.8
PURA LINGSAR	6.9	0	25.6	33	19.8	49.2
SENGGIGI	12.9	25.6	0	7.6	5.8	59.4
MALIMBU	18.7	33	7.6	0	16.1	69.6
BATU BOLONG	26.3	19.8	5.8	16.1	0	53.5
GILI NANGU	44.8	49.2	59.4	69.6	53.5	0
PANTAI ELAK-ELAK	52.3	56.7	66.9	77.2	61.1	7.1
NARMADA	11.2	5	29.9	40.2	24.1	49.5
SURANADI	14.9	8.7	33.6	43.9	27.8	55.8
SESAOT	23.9	17.7	42.6	52.9	36.8	64.6
BanyuMulek	11.3	15.7	29.9	40.2	23.4	35.3
GILI AIR, MENO, TRAWANGAN	25.7	32.8	21.8	14.4	27.6	69.1
SENARU	87	94.2	83.1	75.7	88.9	130.9
SEMBALUN	118.7	125.9	114.8	107.4	120.6	162.6
sendang gile	89.1	96.3	85.2	77.8	91	133
PUSUK	15.8	23	23.5	21.8	17.7	59.3
BAYAN	80.1	87.3	76.2	68.8	82	124
DESA SADE	40.7	42.3	60.9	71.1	55	63.4
PANTAI AAN	53.3	55	73.5	84.4	68.3	63.5
KUTA	48.3	49.9	68.4	78.7	62.6	58.4
PENUJAK	28.5	30.1	48.7	59	42.9	51.2
seger	52.5	54.2	72.7	83	66.9	62.7
Mawun	59.3	53.8	72.3	82.6	63.9	47.6
selong belanak	47.7	49.4	66.7	76.9	61	40.7

PANTAI PINK	72.2	68.2	90.9	101	85.1	100
TANJUNG RINGGIT	74.6	70.6	93.3	104	87.5	103
PANTAI CEMARA	67.9	63.9	86.7	96.9	80.8	95.9
tanjung beloam	73	69	91.8	102	85.9	101
TETE BATU	42.8	36.6	61.5	71.8	55.7	79.7
GiliKondo	85.6	79.4	104	115	98.5	120

	PANTAI ELAK-ELAK	NARMADA	SURANADI	SESAOT	BanyuMulek	GILI AIR, MENO, TRAWANGAN
BANDARA	62.7	27.2	33.6	42.6	26.3	60.6
LEMBAR	25.1	33	39.3	48.3	20.5	52.7
LABUHAN LOMBOK	130	72.4	72.3	81.3	85.3	113
MATARAM	51.1	13.1	19.1	28.1	12.5	26
SEKARBELA	48.4	13.4	19.6	28.6	10.6	29
TAMAN MAYURA	49.5	8.5	14.8	23.8	11.8	29.3
PURA MERU	49.5	8.5	14.8	23.8	11.8	29.3
MUSEUM NTB	50.6	14.2	22	31	13.7	28.8
PASAR SENI SAYANG-SAYANG	52.3	11.2	14.9	23.9	11.3	25.7
PURA LINGSAR	56.7	5	8.7	17.7	15.7	32.8
SENGGIGI	66.9	29.9	33.6	42.6	29.9	21.8
MALIMBU	77.2	40.2	43.9	52.9	40.2	14.4
BATU BOLONG	61.1	24.1	27.8	36.8	23.4	27.6
GILI NANGU	7.1	49.5	55.8	64.6	35.3	69.1
PANTAI ELAK-ELAK	0	56.6	62.9	71.9	42.5	76.2
NARMADA	56.6	0	6.3	15.4	15.3	37.1
SURANADI	62.9	6.3	0	9	22.7	40.8
SESAOT	71.9	15.4	9	0	31.5	49.8
BanyuMulek	42.5	15.3	22.7	31.5	0	37.3
GILI AIR, MENO, TRAWANGAN	76.2	37.1	40.8	49.8	37.3	0
SENARU	137.9	98.5	102.1	110.9	98.3	58.8
SEMBALUN	169.6	137	133	142.6	129.3	95.5
sendang gile	140	100.6	104.2	113	100.7	65.9
PUSUK	66.4	27.3	30.9	40	27.4	9.8
BAYAN	131	91.6	95.2	104	91.7	56.9
DESA SADE	70.5	34.4	40	47.7	33.9	66.6
PANTAI AAN	71.3	47	52.7	60.3	46.5	79.3
KUTA	70.7	42	47.6	55.2	41.5	74.2
PENUJAK	58.3	22.2	27.9	35.5	21.7	54.4

seger	69.9	46.2	51.9	59.5	45.7	78.5
Mawun	54.7	45.8	51.5	59.1	45.3	78.1
selong belanak	47.5	40.2	46	53.5	39.7	72.4
PANTAI PINK	107	63.2	68.8	76.8	66.9	98.1
TANJUNG RINGGIT	110	65.5	71.2	79.1	69.3	100
PANTAI CEMARA	103	58.8	64.5	72.4	62.6	93.8
tanjung beloam	108	64	69.6	77.6	67.7	98.9
TETE BATU	86.8	31.6	31.4	37.6	45	68.7
GiliKondo	128	74.4	74.2	80.4	85.7	110

	SENARU	SEMBALUN	sendang gile	PUSUK	BAYAN	DESA SADE	PANTAI AAN
BANDARA	115	153.6	118	50.7	108.3	10.7	23.4
LEMBAR	113.9	152.5	116	45.4	107	46.9	49.8
LABUHAN LOMBOK	74.8	68.6	76.9	120	67.9	82.8	96.1
MATARAM	87.3	119	89.4	16.1	80.4	43.4	56.1
SEKARBELA	90.3	122	92.4	19.1	83.4	42.4	55.7
TAMAN MAYURA	90.8	122.3	92.7	19.4	83.7	39.4	51.7
PURA MERU	90.8	122.3	92.7	19.4	83.7	39.4	51.7
MUSEUM NTB	90.1	121.8	92.2	19	83.2	44.6	57.2
PASAR SENI SAYANG-SAYANG	87	118.7	89.1	15.8	80.1	40.7	53.3
PURA LINGSAR	94.2	125.9	96.3	23	87.3	42.3	55
SENGGIGI	83.1	114.8	85.2	23.5	76.2	60.9	73.5
MALIMBU	75.7	107.4	77.8	21.8	68.8	71.1	84.4
BATU BOLONG	88.9	120.6	91	17.7	82	55	68.3
GILI NANGU	130.9	162.6	133	59.3	124	63.4	63.5
PANTAI ELAK-ELAK	137.9	169.6	140	66.4	131	70.5	71.3
NARMADA	98.5	137	100.6	27.3	91.6	34.4	47
SURANADI	102.1	133	104.2	30.9	95.2	40	52.7
SESAOT	110.9	142.6	113	40	104	47.7	60.3
BanyuMulek	98.3	129.3	100.7	27.4	91.7	33.9	46.5
GILI AIR, MENO, TRAWANGAN	58.8	95.5	65.9	9.8	56.9	66.6	79.3
SENARU	0	35	2.4	64.1	6.9	121	133
SEMBALUN	35	0	35.1	90.9	38.6	99.9	104
sendang gile	2.4	35.1	0	86.8	9.2	123	136
PUSUK	64.1	90.9	86.8	0	64.3	56.8	69.4
BAYAN	6.9	38.6	9.2	64.3	0	121	134
DESA SADE	121	99.9	123	56.8	121	0	13.2
PANTAI AAN	133	104	136	69.4	134	13.2	0
KUTA	128	107	131	64.3	129	8.2	5.2

PENUJAK	109	83.8	111	44.6	109	12.2	24.8
seger	135	105	135	68.6	133	12.4	4.5
Mawun	132	117	135	68.2	132	18.6	17.7
selong belanak	127	102	129	62.7	127	24.3	24.6
PANTAI PINK	137	93.9	137	88.3	140	49.5	50.7
TANJUNG RINGGIT	139	96.2	139	90.6	143	51.9	53
PANTAI CEMARA	133	89.5	133	83.9	136	45.2	46.3
tanjung beloam	138	94.7	138	89.1	141	50.3	51.5
TETE BATU	144	59.5	86.8	58.9	90.3	49.9	59.6
GiliKondo	60.8	65.2	60.9	118	64.4	84.8	89.9

	KUTA	PENUJAK	seger	mawun	selong belanak	PANTAI PINK	TANJUNG RINGGIT
BANDARA	18.3	4.4	22.6	26.2	23.6	48.6	51
LEMBAR	46.9	34.8	49	35.5	26.5	87.3	86.2
LABUHAN LOMBOK	91	75.9	96.2	95	94.1	77.6	79.9
MATARAM	51	31.3	55.3	50.3	49.3	80.4	82.7
SEKARBELA	50	30.2	54.2	53.8	70	73.8	76.1
TAMAN MAYURA	46	27.2	50.2	58.6	65.5	69.9	72.3
PURA MERU	46	27.2	50.2	58.6	65.5	69.9	72.3
MUSEUM NTB	52.2	32.4	56.5	61.1	71.5	76	78.3
PASAR SENI SAYANG-SAYANG	48.3	28.5	52.5	59.3	47.7	72.2	74.6
PURA LINGSAR	49.9	30.1	54.2	53.8	49.4	68.2	70.6
SENGGIGI	68.4	48.7	72.7	72.3	66.7	90.9	93.3
MALIMBU	78.7	59	83	82.6	76.9	101	104
BATU BOLONG	62.6	42.9	66.9	63.9	61	85.1	87.5
GILI NANGU	58.4	51.2	62.7	47.6	40.7	100	103
PANTAI ELAK-ELAK	70.7	58.3	69.9	54.7	47.5	107	110
NARMADA	42	22.2	46.2	45.8	40.2	63.2	65.5
SURANADI	47.6	27.9	51.9	51.5	46	68.8	71.2
SESAOT	55.2	35.5	59.5	59.1	53.5	76.8	79.1
BanyuMulek	41.5	21.7	45.7	45.3	39.7	66.9	69.3
GILI AIR, MENO, TRAWANGAN	74.2	54.4	78.5	78.1	72.4	98.1	100
SENARU	128	109	135	132	127	137	139
SEMBALUN	107	83.8	105	117	102	93.9	96.2
sendang gile	131	111	135	135	129	137	139
PUSUK	64.3	44.6	68.6	68.2	62.7	88.3	90.6
BAYAN	129	109	133	132	127	140	143
DESA SADE	8.2	12.2	12.4	18.6	24.3	49.5	51.9
PANTAI AAN	5.2	24.8	4.5	17.7	24.6	50.7	53

KUTA	0	19.7	4.4	10.00	19.5	52.1	54.4
PENUJAK	19.7	0	24	23.6	19.2	50.1	52.4
seger	4.4	24	0	16.9	23.9	51.5	53.8
Mawun	10.00	23.6	16.9	0	8.6	65.2	67.5
selong belanak	19.5	19.2	23.9	8.6	0	65.1	67.4
PANTAI PINK	52.1	50.1	51.5	65.2	65.1	0	14.7
TANJUNG RINGGIT	54.4	52.4	53.8	67.5	67.4	14.7	0
PANTAI CEMARA	47.7	48.9	47.1	60.8	60.7	18.5	20.8
tanjung beloam	52.9	50.9	52.3	66	65.9	13.2	3.2
TETE BATU	57.4	41.2	61.8	64.8	59.2	60.1	63.9
GiliKondo	92.4	85.4	90.8	101	100	79.6	81.9

	PANTAI CEMARA	tanjung beloam	TETE BATU	GiliKondo
BANDARA	44.2	49.4	47.7	83.9
LEMBAR	79.5	84.7	63.3	104
LABUHAN LOMBOK	73.2	78.4	79.7	7.00
MATARAM	76	81.2	47.1	89.9
SEKARBELA	75.4	74.6	44	87.2
TAMAN MAYURA	65.6	70.7	40.1	82.9
PURA MERU	65.6	70.7	40.1	82.9
MUSEUM NTB	71.6	76.8	45.8	88.6
PASAR SENI SAYANG-SAYANG	67.9	73	42.8	85.6
PURA LINGSAR	63.9	69	36.6	79.4
SENGGIGI	86.7	91.8	61.5	104
MALIMBU	96.9	102	71.8	115
BATU BOLONG	80.8	85.9	55.7	98.5
GILI NANGU	95.9	101	79.7	120
PANTAI ELAK-ELAK	103	108	86.8	128
NARMADA	58.8	64	31.6	74.4
SURANADI	64.5	69.6	31.4	74.2
SESAOT	72.4	77.6	37.6	80.4
BanyuMulek	62.6	67.7	45	85.7
GILI AIR, MENO, TRAWANGAN	93.8	98.9	68.7	110
SENARU	133	138	144	60.8
SEMBALUN	89.5	94.7	59.5	65.2

sendang gile	133	138	86.8	60.9
PUSUK	83.9	89.1	58.9	118
BAYAN	136	141	90.3	64.4
DESA SADE	45.2	50.3	49.9	84.8
PANTAI AAN	46.3	51.5	59.6	89.9
KUTA	47.7	52.9	57.4	92.4
PENUJAK	48.9	50.9	41.2	85.4
seger	47.1	52.3	61.8	90.8
Mawun	60.8	66	64.8	101
selong belanak	60.7	65.9	59.2	100
PANTAI PINK	18.5	13.2	60.1	79.6
TANJUNG RINGGIT	20.8	3.2	63.9	81.9
PANTAI CEMARA	0	19.3	55.8	75.2
tanjung beloam	19.3	0	60.9	80.4
TETE BATU	55.8	60.9	0	57.5
GiliKondo	75.2	80.4	57.5	0