OPTIMASI WAKTU EKSEKUSI PENENTUAN RUTE MENUJU OBYEK WISATA DI MALANG RAYA DENGAN ALGORITMA GENETIKA

Mahmud Yunus[1)], Ryan Markus Thobias Rumlaklak[2)]
[1)]Teknik Informatika STMIK PPKIA Pradnya Paramita
email: myoenoes@gmail.com
[2)]Teknik Informatika STMIK PPKIA Pradnya Paramita
email: ryanrumlaklak@outlook.com

Abstract

As one of the tourist destinations in Indonesia, the area of Malang Raya offers a variety of interesting tourist attractions. Based on BPS statistics from Kota Batu in 2015, total tourist visits were 2,089,022 people. While the total tourist visit in 2014 in Malang Regency is 2,118,008 people. Data was taken at 22 tourism object in Batu Town and 10 tourism object in Malang Regency. The location of the destination of several distant attractions, a constraint for tourists to determine the optimal route to the tourist attraction. One alternative solution to the problem can be done using Genetic Algorithms. The focus of this research is to build a web based application that can provide information on the order of route of visits to some tourism objects optimally. Implementation is done based on the completion of Traveling Salesman Problem, using Genetic Algorithm method. Variable optimization is (1) the shortest total distance; and (2) application execution time with Brute Force algorithm as a comparison. The results show that if the location of the chosen destination is numerous, the Genetic Algorithm is more efficient in the use of time and resources than the Brute Force algorithm.

PENDAHULUAN

Sebagai salah satu tujuan wisata di Indonesia, wilayah Malang Raya menawarkan berbagai tempat wisata yang menarik. pesona alam pegunungan, pantai-pantai, air terjun, dan juga taman maupun kebun serta banyak wahana wisata buatan seperti Jatim Park, Batu Night Spectacular, Songgoriti dan sebagainya. Wilayah Malang Raya meliputi Kota Malang, Kabupaten Malang dan Kota Batu dengan luas wilayah 3.989,40 KM^2 (BAPPEDA Kab. Malang, 2014). obyek-obyek wisata yang ada tersebar merata ke seluruh wilayah.

Letak destinasi obyek wisata yang bervariasi namun berjauhan ini menjadi kendala bagi wisatawan untuk menentukan pilihan obyek wisatanya. Apalagi jika wisatawan yang pertama kali berwisata di wilayah Malang Raya, sangat sulit untuk merencanakan perjalanan wisatanya. Berdasarkan statistik yang dikeluarkan oleh Badan Pusat Stistik Kota Batu tahun 2015, total kunjungan wisata adalah 2.089.022 kunjungan, sedangkan total kunjungan wisata pada tahun 2014 di Kabupaten Malang adalah 2.118.008 pengunjung. Data tersebut diambil pada 22 obyek wisata yang terletak di Kota Batu dan 10 obyek wisata yang terletak di Kabupaten Malang.

Belum banyak aplikasi yang dapat membantu wisatawan dalam merencanakan urutan rute (jalur) kunjungan ke beberapa obyek wisata sekaligus dengan total jarak terpendek dan waktu eksekusi aplikasi yang cepat. Salah satu alternatif pemecahan masalah tersebut dapat dilakukan dengan menggunakan Algoritma Genetik. Algoritma Genetika dapat memberikan waktu eksekusi yang lebih stabil dengan kompleksitas yang tinggi dibandingkan metode konvensional seperti metode Brute Force (Tahyudin, 2015).

Algoritma Genetika telah diterapkan secara luas untuk penyelesaian masalah optimasi. Secara khusus penggunaan GA bersifat kompetitif dan direkomendasikan untuk memecahkan masalah optimasi kombinatorial alamiah (Alberto J. Urdaneta, 1999),(D.E Goldberg, 1989). Penelitian mengenai upaya meminimalkan total biaya tetap yang terkait dengan instalasi jaringan dengan optimalisasi alokasi sumber daya reaktif menggunakan GA,

telah berhasil memberikan beberapa alternatif solusi dari permasalahan yang ada (Alberto J. Urdaneta, 1999). Keuntungan penggunaan GA sangat jelas terlihat dari kemudahan implementasi dan kemampuannya untuk menemukan solusi yang dapat diterima secara cepat untuk masalah-masalah berdimensi tinggi (N. Sannomiya, H. Iima, 1992). Keuntungan lain dari penerapan GA adalah dapat dengan mudah dikodekan untuk bekerja pada komputer paralel (Anastasios G. Bakirtzis, 2002).

Fokus penelitian ini adalah untuk membangun aplikasi *web based* yang dapat memberikan informasi urutan rute kunjungan ke beberapa obyek wisata secara optimal. Implementasi dilakukan berdasarkan penyelesaian *Travelling Salesman Problem* (TSP) dengan menggunakan metode Algoritma Genetika. Variable pengukuran optimasi adalah (1) total jarak tempuh terpendek; dan (2) waktu eksekusi aplkasi dengan algoritma *Brute Force* sebagai pembanding.

KAJIAN LITERATUR

Travelling Salesman Problem

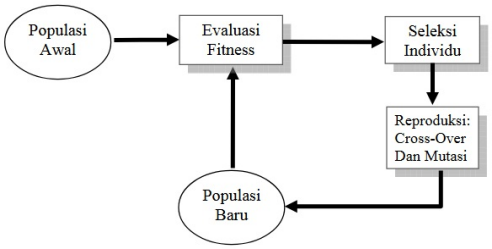
Permasalahan matematika tentang *Traveling Salesman Problem* (TSP) dikemukakan pada tahun 1800 oleh matematikawan Irlandia, William Rowan Hamilton, dan matematikawan Inggris, Thomas Penyngton. TSP adalah permasalahan dimana seorang salesman harus mengunjungi smeua kota dimana tiap kota hanya dikunjungi satu kali, dengan kondisi dimana dia harus mulai dari suatu kota dan kembali ke kota tersebut.

TSP merupakan permasalahan yang memang mudah untuk diselesaikan dengan algoritma Brute Force, tetapi hal itu hanya dapat dilakukan dengan jumlah kota atau simpul yang tidak banyak. Kompleksitas algoritma untuk permasalahan TSP dengan algoritma Brute Force adalah O(n!) dengan catatan n adalah jumlah kota atau simpul dan setiap kota atau simpul terhubung dengan semua kota atau simpul lainnya. Dengan jumlah sebanyak 20 kota, maka kemungkinan solusinya sebanyak 6×10^{16}.

Algoritma Genetika

Algoritma genetik adalah algoritma yang menerapkan pemahaman mengenai evolusi alamiah pada upaya pemecahan masalah. Pendekatan yang diambil algoritma ini adalah dengan menggabungkan secara acak berbagai solusi-solusi terbaik yang dipilih di dalam suatu kumpulan untuk mendapatkan generasi solusi terbaik berikutnya yaitu pada suatu kondisi yang memaksimalkan kecocokan yang akan disebut dengan fitness. Generasi yang terpilih adalah generasi yang merepresentasikan perbaikan-perbaikan pada populasi atau kumpulan sebelumnya. Dengan melakukan proses yang berulang-ulang, algoritma ini dapat mendapatkan solusi yang paling tepat untuk permasalahan yang dihadapi.

Solusi permasalahan dalam penerapan algoritma genetik, direpresentasikan ke dalam bentuk kromosom. Terdapat tiga aspek penting dalam menerapkan algoritma genetik yaitu (1) definisi fungsi fitness; (2) definisi dan implementasi representasi genetik; serta (3) definisi dan implementasi operasi genetik.



Gambar 1. Siklus Algoritma Genetika D. Goldberg

Proses dalam Algoritma Genetika secara bertahap dijelaskan sebagai berikut (Suyanto, 2005).

(1) Pengkodean atau Representasi

Langkah pertama yang dilakukan dalam penggunaan metode GA adalah melakukan pengkodean atau representasi terhadap permasalahan yang akan dioptimasi. Pengkodean yang lazim digunakan dalam GA seperti mengguanakan kode bilangan biner bilangan *real*, dan huruf. Pengkodean digunakan untuk membentuk gen-gen yang ada dalam kromosom.

Gen (genotype) merupakan variabel dasar yang membentuk suatu kromosom individu,

dimana gen ini bisa berupa nilai biner, float, integer maupun karakter (T. Sutojo, Edy Mulyanto, Vincent Suhartono, 2011). Kromosom merupakan gabungan dari gen-gen yang membentuk arti tertentu, dimana dalam aplikasi GA, kromosom merupakan representasi dari solusi yang dicari.

(2) Menghitung Nilai *Fitness* Setiap Individu

Nilai *Fitness* menyatakan seberapa baik nilai dari suatu individu atau solusi yang didapat. Nilai Fitness menyatakan nilai dari fungsi tujuan. Tujuan dari GA adalah memaksimalkan nilai *Fitness*. Jika yang dicari nilai masimal, maka nilai *Fitness* adalah nilai dari fungsi itu sendiri. Tetapi jika yang dibutuhkan adalah nilai minimal, maka nilai *Fitness* merupakan invers dari fungsi itu sendiri (T. Sutojo, Edy Mulyanto, Vincent Suhartono, 2011). Nilai *Fitness* mempengaruhi terpilihnya kromosom-kromosom individu tertentu untuk mengalami proses selanjutnya yaitu kawin silang (c*rossover*) dalam siklus GA.

Fungsi Fitness f(x) dapat berupa suatu persamaan bebas dengan memberikan nilai batasan pada setiap variabelnya. Namun jika batasan nilai variabel dalam fungsi *Fitness* semakin kecil, akan dapat mengakibatkan terjadinya konvergensi dini/prematur(mengarah ke satu titik pertemuan lebih cepat), dimana nilai-nilai *Fitness* yang dihasilkan fungsi tersebut hampir sama (populasi konvergen). Sehingga konvergensi dini dapat mengakibatkan munculnya solusi optimum lokal dan bukan solusi optimum yang sebenarnya.

Nilai *fitness* setiap kromosom individu diperoleh dengan fungsi objektif. Nilai *fitness* menyatakan dari fungsi tujuan. Tujuan dari GA adalah memaksimalkan nilai *fitness*. Jika yang dicari adalah nilai maksimal, maka nilai *fitness* merupakan nilai dari fungsi tujuan GA. Jika yang dibutuhkan adalah nilai minimal, maka nilai *fitness* merupakan *invers* dari nilai fungsi itu sendiri dengan C = konstanta, ε = bilangan kecil yang ditentukan dengan nilai 0, dan *x* = kromosom individu. Nilai *invers* dapat diperoleh dengan persamaan berikut;

$$Fitnes = \frac{C}{objektif\,(x) + \varepsilon}$$

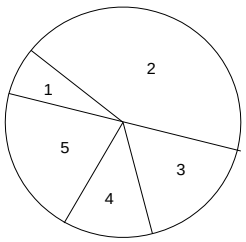
Sehingga nilai *fitness* setiap kromosom dalam penelitian ini diperoleh dengan persamaan:

$$Fitnes\ kromosom[n] = \frac{1}{(objektif\ (kromosom[n]+1))}$$

Proses seleksi dilakukan dengan cara memilih kromosom individu yang mempunyai nilai objektif terkecil atau *fitness* terbesar, hal ini karena optimasi yang dicari sebagai solusi.

(3) Seleksi

Proses seleksi digunakan untuk memilih dua buah individu yang aka dijadikan orang tua, kemudian dilakukan proses pindah silang (*crossover*) untuk mendapatkan keturunan baru (T. Sutojo, Edy Mulyanto, Vincent Suhartono, 2011). Proses pemilihan ini didasarkan pada nilai *Fitness* tiap kromosom yang diranking atau diurutkan berdasarkan besar nilainya dan kemudian urutan tersebut menjadi indeks bagi kromosom yang bersangkutan.



Gambar 2. Roulette Wheel

Kromosom yang berindeks fungsi *Fitness* terbesar akan memiliki peluang paling besar untuk terpilih menjadi kromosom induk dalam proses pindah silang maupun proses mutasi.

Metode seleksi yang sering digunakan adalah metode *Roulette Wheel* (Roda Roulette), dimana masing-masing individu menempati potongan lingkaran roda secara proporsional sesuai dengan nilai *Fitness*-nya (T. Sutojo, Edy Mulyanto, Vincent Suhartono, 2011).

(4) Pindah Silang (*Crossover*)

Proses *crossover* adalah proses menyilangkan dua kromosom induk hasil seleksi. Sebuah individu yang mengarah pada solusi optimal bisa diperoleh melalui proses pindah silang, dengan catatan bahwa pindah silang hanya bisa dilakukan jika sebuah bilangan random ***r*** yang dibangkitkan dalam interval [0 s/d 1], nilainya kurang dari nilai

probabilitas pindah silang tertentu (*r* < *probabilitas*). Cara yang paling sederhana untuk melakukan pindah silang adalah dengan teknik satu titik potong (*one point crossover*), dimana posisi titik potong diperoleh secara random.

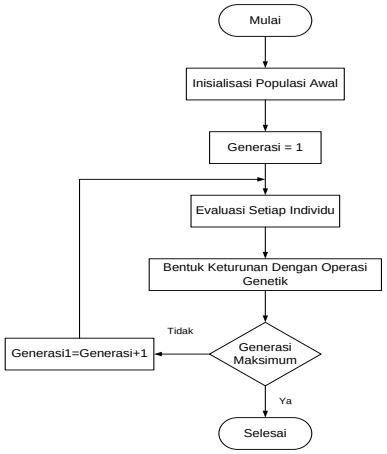
Gambar 3. Pindah Silang Satu Titik

(5) Mutasi

Pada proses mutasi tidak memandang kromosom, melainkan gen-gen dalam kromosom. Probabilitas mutasi akan menentukan gen-gen dari suatu populasi yang akan mengalami proses mutasi. Mutasi adalah proses mengganti nilai gen sebelumnya dengan nilai baru yang ditentukan secara acak (random) dengan range yang ditentukan sebelumnya. Gen-gen dengan nilai baru ini selanjutnya bergabung lagi dengan kromosom induknya, masing-masing untuk ditentukan lagi nilai *fitness* barunya. Nilai *fitness* baru ini yang akan menentukan nilai optimal pada iterasi yang telah dilakukan

(6) *Reinsertion* (Reins) Untuk Penggantian Populasi

Kromosom-kromosom yang telah mengalami proses kawin silang dan mutasi akan digabung dengan kromosom-kromosom lama yang tidak mengalami kawin silang dan mutasi menggunakan proses yang dinamakan *reinsertion* atau reins.



Gambar 4. Siklus Eksekusi Algoritma

Reinsertion merupakan proses penggantian populasi (*generational replacement*) yang akan mengganti semua individu awal dengan individu-individu hasil pindah silang dan mutasi serta individu-individu yang tidak mengalami proses tersebut. Secara garis besar, siklus eksekusi GA ditunjukan pada gambar 4.

METODE PENELITIAN

Bukan hal yang mudah bagi wisatawan untuk menentukan urutan rute kunjungan yang efisien ke obyek-obyek wisata yang tersebar ke seluruh penjuru Malang Raya. Terdapat beberapa cara wisatawan yang ingin berkunjung ke Malang Raya untuk mencari informasi dan menentukan destinasi berwisata mereka, yaitu sebagai berikut;

1. Mencari informasi melalui internet, seringkali wisatawan tertarik untuk mengunjungi suatu tempat wisata setelah melihat unggahan foto di media sosial atau pun merupakan rekomendasi dari kerabat. Namun di media sosial jarang memuat informasi yang sangat detail tentang obyek wisata tersebut.
2. Menggunakan jasa agen/pemandu wisata adalah pilihan yang paling mudah, karena wisatawan tidak perlu lagi merencanakan dan mencari sendiri informasi tentang obyek wisata. Namun menggunakan jasa pemandu wisata tidak bisa menjadi pilihan semua orang karena biaya yang dikeluarkan lebih mahal.

Konsep Solusi Permasalahan

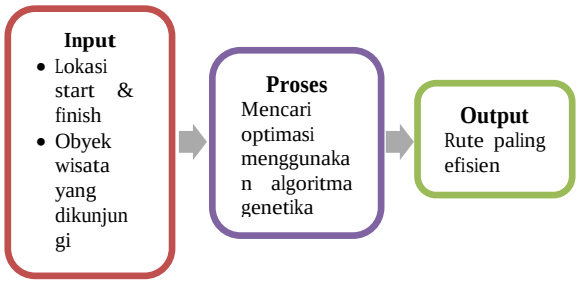
Salah satu konsep solusi dari permasalahan tersebut adalah dengan rancang bangun sebuah aplikasi berbasis web yang dapat memberikan informasi tentang obyek wisata di Malang Raya dan memiliki fungsi untuk membuat rute perjalanan yang paling efisien. Calon wisatawan dapat memilih obyek wisata mana yang akan dikunjungi, kemudian aplikasi ini menghitung dan membuat sebuah rute perjalanan wisata yang efisien.

Metode yang digunakan dalam rancang bangun aplikasi ini adalah implementasi dari Travelling Salesman Problem (TSP) yang dikombinasikan dengan algoritma genetika untuk mencari solusi terbaik dari permasalahan

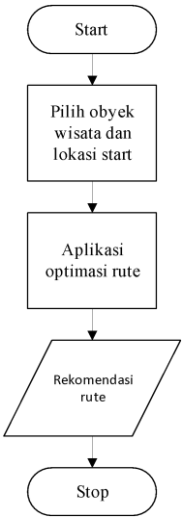
yang ada. Langkah-langkah yang dilakukan adalah sebagai berikut;

(1) Melakukan studi kepustakaan terhadap berbagai referensi yang berkaitan dengan penelitian yang dilakukan. Topik-topik yang dikaji antara lain meliputi implementasi algoritma genetika, implementasi *Travelling Saleman Problem*, dan penggunaan Google Maps API.
(2) Menyiapkan database obyek wisata, yang meliputi nama obyek wisata, deskripsi obyek wisata dan lokasi obyek wisata.
(3) Merancang aplikasi penentuan rute dengan implementasi algoritma genetika pada *Travelling Salesman Problem*.
(4) Melakukan pengujian aplikasi.

Pemodelan Sistem Solusi



Gambar 5. Input, Proses, dan Output Aplikasi



Gambar 6. Flowchart Aplikasi

Pada aplikasi optimasi rute ini pengguna diharapkan memasukkan data titik lokasi tempat start dan obyek-obyek wisata yang ingin dikunjungi, selanjutnya diproses dengan mencari optimasi rute paling efisien dengan Algoritma Genetika. Setelah diproses, maka pengguna akan menerima output berupa rute paling efisien untuk obyek-obyek wisata yang dipilih. Flowchart aplikasi penentuan rute perjalanan wisata yang dibangun adalah pada gambar 6.

HASIL DAN PEMBAHASAN

Implementasi Algoritma Genetika

(1) Fungsi inisialisasi populasi

Fungsi inisialisasi populasi digunakan untuk membangkitkan populasi awal secara acak. Jumlah gen per individu tergantung dari banyaknya lokasi tujuan yang dipilih oleh pengguna. Gen dalam individu disimbolkan dengan karakter abjad.

```
function initialPopulation($jumlah_individu,
$jumlah_gen)
{
    for ($x = 0; $x < $jumlah_individu;
    $x++) {
        $individu[] = "A" .
        $this->pickRandom($jumlah_gen) . "A";
    }
    $check_equal = $this->identical($individu);
    if ($check_equal != true) {
        return $individu;
    } else {
        return FALSE;
    }
    unset($individu);
}
```

(2) Evaluasi Nilai Fitnes Individu

Fungsi hitung nilai fitness digunakan untuk menyatakan nilai dari fungsi tujuan. Semakin tinggi nilai fitness maka semakin optimal solusi yang dihasilkan dari individu tersebut.

```
function calculateFitness($individu,
$distance) {
    $individu_fitness = array();
    $e = 0.1;
    foreach ($individu as $indi):
        $total_distance =
        $this->calculateDistance($indi,
$distance);
```

```
        $fitness = 1 / ($total_distance +
$e);
        $fitness = round($fitness, 15,
        PHP_ROUND_HALF_UP);
        $individu_fitness[$indi]=$fitness;
    endforeach;
    return $individu_fitness;
}

function calculateDistance($indi,
$distance) {
    $length = strlen($indi) - 1;
    $total_distance = 0;
    for ($i = 0; $i < $length; $i++) {
        $j = $i + 1;
        $from = $indi[$i];
        $to = $indi[$j];
        if(!empty($distance[$from][$to])) {
            $indi_distance =
            $distance[$from][$to];
        } else {

    $indi_distance=$distance[$to][$from];
        }
        $total_distance = $total_distance
        + $indi_distance;
    }
    return $total_distance;
}
```

(3) Seleksi Individu Dengan Roulette Wheel

Proses seleksi Roulette Wheel dimulai dengan mencari nilai acak dari sampai 1, lalu dipilih berdasarkan probailitas yang dihasilkan oleh tiap individu. Individu-individu yang terpilih akan diproses di proses crossover untuk menghasil generasi yang baru.

```
function calculateProbability($population,
$population_key) {
    echo "<br/>";
    $f_max = max($population);
    $f_min = min($population);
    $n = count($population);
    $lfr_kumulatif = 0.0000;
    $sorted_key = array_keys($population);
    foreach ($sorted_key as $key):
        $r = array_search($key,$population_key);
        $lfr[] = $f_max - (($f_max -
        $f_min) * ($r / ($n - 1)));
    endforeach;
    $lfr_total = array_sum($lfr);
    echo "<br/>LFP total: " . $lfr_total;
    foreach ($lfr as $item):
        $lfr_kumulatif = $lfr_kumulatif+$item;
        $probability[] = ROUND($lfr_kumulatif /
$lfr_total, 2,
        PHP_ROUND_HALF_UP);
    endforeach;
    $counter = 1;
    foreach ($probability as $prob):
        echo "<br/>Probabilitas individu "
        . $counter++ . ": " . $prob;
    endforeach;
    return $probability; unset($probability);
unset($lfr);
}

function randomRw($jumlah_individu){
    echo "<br/><br/>Random roulette wheel";
    for ($i = 0; $i < $jumlah_individu;$i++) {
    $random[]= round($this->random_0_1(), 2,
        PHP_ROUND_HALF_UP);
        $j = $i + 1;
        echo "<br/>random " . $j . ": " .
        $random[$i];
    }
    return $random;
    unset($random);
}
```

(4) Proses Crossover

Proses cross over atau pindah silang ini melibatkan 2 individu untuk menghasilkan 2 individu baru. Proses cross over yang dipakai adalah dengan menukarkan 2 buah gen pada posisi acak antara 2 individu yang menjadi orang tua.

```
function crossover($parents) {
    $couple = (int) count($parents)/2;
    for ($i = 0; $i < $couple; $i++) {
        $m = 2 * $i;
        $f = $m + 1;
        $male = $parents[$m];
        $male = substr($male, 1, -1);
        $female = $parents[$f];
        $female = substr($female, 1, -1);
        $children[] = $this->breeding($male,
$female);
    }
    return $children; unset($children);
}

function breeding($male, $female) {
    $length = strlen($male); $batas = $length - 1;
    $child1 = array(); $child2 = array();
    $sequence1 = array(); $sequence2 = array();
    $taken1 = array(); $taken2 = array();
```

```
    do {
        $acak1 = mt_rand(0, $batas);
        $acak2 = mt_rand(1, $batas);
    } while ($acak1 == $acak2 ||
    $acak1 > $acak2);
    for ($i = 0; $i < $length; $i++) {
        if ($i>=$acak1 && $i<=$acak2) {
            array_push($child1,substr($male,$i,1));
            array_push($child2,
            substr($female, $i, 1));
            $taken1[] = substr($male,$i,1);
            $taken2[] = substr($female,$i,1);
        } else {
            array_push($child1, "x");
            array_push($child2, "x");
        }
        for ($j=$acak2+1;$j<$length;$j++) {
            array_push($sequence1, substr($male,
$j, 1));
            array_push($sequence2,
substr($female, $j, 1));
        }
        for ($k = 0; $k <= $acak2; $k++){
            array_push($sequence1,
            substr($male, $k, 1));
            array_push($sequence2,
substr($female, $k, 1));
        }
        $sisa1 = array_diff($sequence1,$taken2);
        $sisa2 = array_diff($sequence2,$taken1);
        $sisa1 = array_values($sisa1);
        $sisa2 = array_values($sisa2);
        for ($l = $acak2 + 1; $l < $length;$l++) {
            if (count($sisa1) > 0) {
                $child1[$l] = $sisa2[0];
                $child2[$l] = $sisa1[0];
                array_splice($sisa1, 0, 1);
                array_splice($sisa2, 0, 1);
            }
        }
        for ($m = 0; $m <= $acak2;$m++) {
            if (count($sisa1) > 0) {
                $child1[$m] = $sisa2[0];
                $child2[$m] = $sisa1[0];
                array_splice($sisa1, 0, 1);
                array_splice($sisa2, 0, 1);
            }
        }
        $child_gen1 = join("", $child1);
        $child_gen2 = join("", $child2);
        $children[0] = $child_gen1;
        $children[1] = $child_gen2;
        return $children; unset($children);
        unset($taken1); unset($taken2);
    }
}
```

(5) Proses Mutasi

Proses mutasi ini dilakukan pada semua gen yang tersapat pada individu dalam suatu generasi. Proses dimulai dengan menentukan secara acak 2 posisi gen yang akan ditukarkan dalam satu individu.

```
function mutation($hasil_co) {
    $string = $hasil_co;
    $length = strlen($string[0]);
    $batas = $length - 1;
    foreach ($string as $individu):
        do {
            $acak1 = mt_rand(0, $batas);
            $acak2 = mt_rand(0, $batas);
        } while ($acak1 == $acak2);
        $temp1 = substr($individu,
        $acak1, 1);
        $temp2 = substr($individu,
        $acak2, 1);
        $hasil = substr_replace($individu,
        $temp2, $acak1, 1);

    $hasil_mutasi[]=substr_replace($hasil
,
        $temp1, $acak2, 1);
    endforeach;
    return $hasil_mutasi;
    unset($hasil_mutasi);
}
```

(6) Fungsi Menampilkan Peta Dengan Google Maps API

Fungsi ini menampilkan peta dari Google Maps dengan titik-titik tujuan wisata dan lokasi start yang dipilih oleh pengguna.

```
<script type="text/javascript">
    var locations = [
    <?php
        foreach ($wisata as $item):
        echo "[" . $item . "],";
        $object[] = explode(',', $item);
        endforeach;
        echo "['Start'," . $startLat . "," .
        $startLong . "]";
    ?>];
    var map = new google.maps.Map
    (document.getElementById('map'), {
        zoom: 9,
        center: new google.maps.LatLng(-
        7.982601, 112.630853), mapTypeId:
        google.maps.MapTypeId.ROADMAP
    });
```

```
    var infowindow = new
google.maps.InfoWindow();
    var marker, i;
    var labels =
    'ABCDEFGHIJKLMNOPQRSTUVWX YZ';
    var labelIndex = 0;
    for (i = 0; i < locations.length; i++) {
        marker = new google.maps.Marker({
        position:new
google.maps.LatLng(locations[i][1],
        locations[i][2]) label
labels[labelIndex++%
        labels.length], map: map});
        google.maps.event.addListener(marker,
        'click', (function (marker, i) {
            return function () {
            infowindow.setContent(locations[i][0]);
                infowindow.open(map, marker);
            }
        })(marker, i));
    }
</script>
```

Hasil Pengujian

Berikut adalah data yang dipakai dan hasil dari 30 sampel dengan variasi titik dan jumlah tujuan wisata yang berbeda.

(1) Pengujian Sampel Menggunakan Uji T Berpasangan

Uji T berpasangan digunakan untuk membandingkan nilai rata-rata yang diperoleh dari pengukuran dua sampel yang saling berhubungan satu sama lain. Penelitian ini membandingkan hasil dari 30 kasus percobaan yang sudah dilakukan.

Pengujian akurasi hasil jarak terpendek dari 30 kasus percobaan yang ada, terdapat 18 sampel dimana algoritma Brute Force tidak menampilkan hasil. Pengujian dilakukan pada 12 kasus percobaan lainnya dimana Algoritma Genetika dan Brute Force sama-sama berhasil menampilkan hasil. Data hasil pengujian akurasi hasil jarak terpendek adalah sebagai berikut.

Tabel 1. Lokasi Obyek Wisata Malang Raya

No	Obyek Wisata	Latitude	Longitude
1	Jatim Park 1	-7.884502	112.52488
2	Jatim Park 2	-7.88852	112.53009
3	Kusuma Agrowisata	-7.88393	112.51551
4	Candi Badut	-7.956699	112.598282
5	Coban Rondo	-7.872324	112.48171
6	Goa Cina	-8.44678	112.6515
7	Gunung Bromo	-7.909038	112.95159
8	Keramik Dinoyo	-7.943487	112.60983
9	Wisata Sumber Brantas	-7.741544	112.53485
10	Kebun Raya Purwodadi	-7.798182	112.74067
11	Batu Night Spectacular	-7.896648	112.53496
12	Museum Brawijaya	-7.972085	112.62109
13	BakpaoTelo	-7.8218	112.70493
14	Paralayang	-7.854867	112.49694
15	Pantai Sendang Biru	-8.431598	112.68575
16	Pasar Minggu	-7.977226	112.62347
17	Masjid Tiban Turen	-8.150999	112.71382
18	Hutan Kota Malabar	-7.968516	112.6265
19	Wisata Madu Lebah Lawang	-7.848805	112.69526
20	Bendungan Karangkates	-8.156151	112.44939
21	Taman Senaputra	-7.975495	112.63139

Tabel 2. Hasil Uji Jarak Terpendek

No	Jarak (dalam KM.)	
	Algoritma Genetika	Brute Force
1	54.4	54.4
2	118.6	118.6
3	129.4	129.4
4	104.1	104.1
5	88.5	88.5
6	184.7	184.7
7	49.6	49.6
8	183.2	182.2
9	62.4	62.4
10	100.9	100.9
11	106	106
12	243.6	243.3

Dari sampel tersebut dilakukan uji T berpasangan dan menghasilkan hasil sebagai berikut.

Tabel 3. Paired Samples Statistics

	Mean	N	Std. Deviasi	Std. Error Mean
Distance Algoritma Genetik	118.78	12	58.7081	16.948
Distance Brute Force	118.68	12	58.5509	16.90

Tabel 4. Paired Samples Correlations

	N	Corelation	Sig.
Distance Algoritma Genetik & Distance Brute Force	12	1.000	0.000

Tabel 5. Paired Samples Test

	Mean	Std. Deviation	Std. Error
Distance Algoritma Genetik- Time Brute Force	0.1083	0.2937	0.0848

	95% Confidence Interval of the Difference		(2-taile d)		
	Lower	Upper			
Distance Algoritma Genetik- Time Brute Force	-0.078	0.295	1.278	11	0.228

Tabel 6. Hasil Uji Waktu Eksekusi

No	Waktu Eksekusi (dalam MS)	
	Algoritma Genetika	Brute Force
1	0.029092073440552000	0.000040054321289062
2	0.038208961486816000	0.000054836273193359
3	0.046123027801514000	0.000138998031616210
4	0.050101995468140000	0.000616073608398440
5	0.054955005645752000	0.003762960433960000
6	0.059437990188599000	0.029155015945435000
7	0.064618110656738000	0.290423154830930000
8	0.072858095169067000	3.697235107421900000
9	0.074270963668823000	3.697235107421900000
10	0.076208114624023000	3.697235107421900000
11	0.081863164901733000	3.697235107421900000
12	0.084616899490356000	3.697235107421900000
13	0.093037128448486000	3.697235107421900000
14	0.105159997940060000	3.697235107421900000
15	0.112433910369870000	3.697235107421900000
16	0.110876083374020000	3.697235107421900000
17	0.114498853683470000	3.697235107421900000
18	0.119351863861080000	3.697235107421900000
19	0.124213933944700000	3.697235107421900000
20	0.134221076965330000	3.697235107421900000
21	0.031466007232666000	0.000030994415283203
22	0.045607089996338000	0.000293016433715820
23	0.056088924407959000	0.003961086273193400
24	0.063810110092163000	0.270946025848390000
25	0.074354887008667000	3.697235107421900000
26	0.083584070205688000	3.697235107421900000
27	0.094172954559326000	3.697235107421900000
28	0.115406036376950000	3.697235107421900000
29	0.116683959960940000	3.697235107421900000
30	0.124078989028930000	3.697235107421900000

Tabel 7. Paired Samples Statistics

	Mean	N	Std. Deviasi	Std. Error Mean
Time Algoritma Genetik	0.0817133427000	30	0.0306836824000	0.005602048
Time Brute Force	2.3615629750000	30	1.7866424730000	0.326194795

Tabel 8. Paired Samples Correlations

	N	Corelation	Sig.
Distance Algoritma Genetik & Distance Brute Force	30	0.829	0.000

Pengujian waktu eksekusi terhadap 30 kasus percobaan yang ada, terdapat 18 kasus dimana algoritma Brute Force tidak dapat menampilkan hasil karena melewati batas penggunaan *resource* yang ditentukan oleh server. Pada proses uji T berpasangan, data tersebut diganti dengan waktu tertinggi dari sampel yang ada hasilnya. Data kasus percobaan yang diujikan adalah pada tabel 6

Dari sampel tersebut dilakukan uji T berpasangan dan menghasilkan hasil pada tabel 7

Tabel 9. Paired Samples Test

	Mean	Std. Deviation	Std. Error
Time Algoritma Genetik- Time Brute Force	-2.27985	1.76129	0.32157

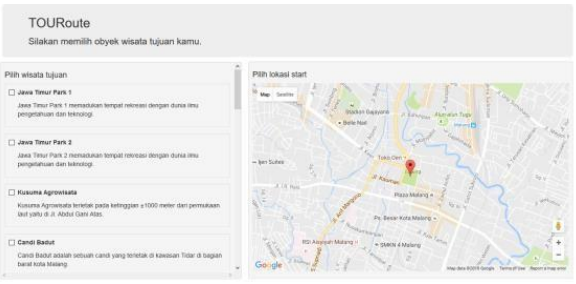
	95% Confidence Interval of the Difference		(2-taile d)		
	Lower	Upper			
Time Algoritma Genetik- Time Brute Force	-2.93753	-1.62217	-7.09	29	0

Hasil uji T berpasangan di atas menunjukan bahwa kecepatan eksekusi algoritma genetika dan algoritma brute force berbeda karena nilai sig. (2-tailed) kurang dari 0,05. Dari nilai rata-rata dapat diambil kesimpulan bahwa algoritma genetika yang kecepatan rata-rata eksekusinya 0,0817133427 microsecond lebih cepat dari algoritma brute force yang kecepatan rata-rata eksekusinya 2.361562975 microsecond.

Hasil Pengujian Output Aplikasi

(1) Pemilihan Lokasi Obyek Wisata

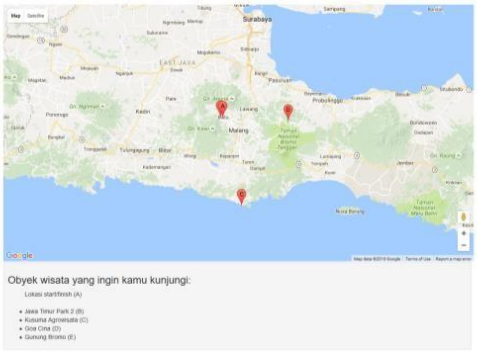
(2) Halaman ini berisi pilihan obyek wisata di Malang Raya dan peta kota Malang untuk menentukan lokasi start. Pada halaman ini user diminta untuk menginputkan data obyek wisata tujuan dan lokasi start.



Gambar 7. Pemilihan Lokasi Obyek Wisata

(3) Konfirmasi Lokasi Obyek Wisata Terpilih

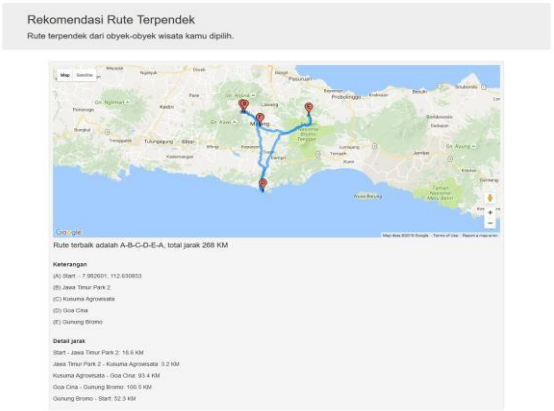
Halaman ini berisi tentang informasi dari input yang telah dipilih oleh user sebelumnya. Jika user menginginkan perubahan user dapat memilih tombol "Kembali" atau "Confirm" untuk menuju halaman rekomendasi rute.



Gambar 8. Konfirmasi Lokasi Obyek Wisata

(4) Rekomendasi Rute Ke Obyek Wisata

Halaman ini berisi peta rekomendasi rute perjalanan wisata dengan detail jarak dari tiap tempat yang dipilih.



Gambar 8: Rekomendasi Rute Ke Lokasi Wisata

KESIMPULAN

Berdasarkan serangkaian tahap pengujian yang telah dilakukan, dapat disimpulkan bahwa aplikasi optimasi rute kunjungan ke beberapa obyek wisata yang telah dibangun dapat memberikan kemudahan dalam memilih rute perjalanan wisata di Malang Raya. Aplikasi optimasi rute dapat membantu proses pembuatan rute dari lokasi dimulai dan obyek-obyek wisata yang ingin dikunjungi. Algoritma Genetika lebih efisien dalam penggunaan waktu dan sumber daya daripada algoritma Brute Force, jika lokasi tujuan yang dipilih banyak.

REFERENSI

1. Badan Perencanaan Pembangunan Daerah. 2015. Statistik Pembangunan Daerah (Kabupaten Malang Dalam Angka) Tahun 2014. Malang: Pemerintah Kabupaten Malang.
2. Badan Pusat Statistik Kota Batu. 2015. Kota Batu dalam Angka 2015. Batu: Badan Pusat Statistik Kota Batu.
3. Kusumadewi, S., dkk. 2005. Penyelesaian Masalah Optimasi dengan Teknik-teknik Heuristi. Yogyakarta: Graha Ilmu.
4. Lukas, dkk. 2005. Penerapan Algoritma Genetika untuk Travelling Salesman Problem dengan Menggunakan Metode Order Crossover dan Insertion Mutation (hlm. 1-5). Seminar Nasional Aplikasi Teknologi Informasi
5. Tahyudin, Imam dan Susanti, Ika. 2015. Pencarian Rute Terbaik pada Obyek Wisata di Kabupaten Banyumas Menggunakan Algoritma Genetika Metode TSP. JUITA ISSN: 2086-9398, 3 (4): 165-173
6. Alberto J. Urdaneta, Juan F. Gomez, Elmer Sorrentino, Luis Flores, Ricardo Diaz. "A Hybrid Genetic Algorithm For Optimal Reactive Power Planning Based Upon Successive Linear Programming". IEEE Transactions on Power Systems, Vol. 14, No.42, November 1999.
7. D.E Goldberg,"Genetic Algorithm in Search, Optimization & Machine Learning," Addison-Wesley Publishing Company, Inc., Canada, 1989, hlm. 59-86.
8. Anastasios G. Bakirtzis, Pandel N. Biskas, Christoforos E. Zoumas, Vasilios Petridis. "Optimal Power Flow by Enhanced Genetic Algorithm". IEEE Transactions on Power Systems, Vol. 17, No.02, May 2002.
9. N. Sannomiya , H. Iima, "Genetic algorithm approach to a production ordering problem in an assembly process with buffers". Proc. of 7th IFAC &mp. On Information Control Problems in a Manufacturing Technology, pp. 403-408,1992.
10. Suyanto, "Algoritma Genetika dalam MATLAB". Yogyakarta: ANDI, 2005, hlm. 1-2.
11. T. Sutojo, Edy Mulyanto, Vincent Suhartono. "Kecerdasan Buatan". Yogyakarta: CV. ANDI Offset, 2011.