PERAKITAN MOBIL RADIO *CONTROL BUGGY NITRO OFFROAD* MENGGUNAKAN ALGORITMA GENETIKA

Ridho Firmansyah[1)], **Safrina Amini**[2)], **Gandung Triyono**[3)]
[1,2,3]Teknik informatika, Teknologi Informasi, Universitas Budi
Jl. Ciledug Raya, Petukangan Utara, Jakarta Selatan, Jakarta, 12260
ridhofirmansyah@gmail.com[1)], safrina.amini@budiluhur.ac.id[2)], gandung.triyono@budiluhur.ac.id[3)]

Abstract

Control radio technology is already experiencing good growth. This is evident by the many areas that utilize these technologies, one of which is a game of toy cars. The Game car by utilizing the radio control are called Car Radio Control (MRC). At this time to have an MRC we can assemble them, with the desired component. However, the problem is to assemble the MRC requires quite a long time. It takes quite a long time due to choosing the components that fit your budget quite difficult. The existence of these problems then developed optimization application to help choose the components of MRC with a genetic algorithm. Applications developed using the Selection Roulette Wheel, Two Point Crossover, and Random Mutation. The results of the trial population size and the number of best generation are 30 and 200. Values parameter combinations crossover probability (Pc) and the probability of mutation (Pm) earn on average the highest fitness is Pc 0.9 and 0.1 Pm. It was concluded that the genetic algorithm is proven to make the process of finding combinations of components to be more efficient than the manual way.

Keywords*: genetic algorithms, optimization, radio control, radio control buggy nitro offroad;*

Abstrak

Teknologi radio control sudah mengalami perkembangan yang baik. Hal tersebut terbukti dengan banyaknya bidang yang memanfaatkan teknologi tersebut, salah satunya adalah permainan mobil-mobilan. Permainan mobil-mobilan dengan memanfaatkan radio control disebut Mobil Radio Control (MRC). Pada saat ini untuk memiliki sebuah MRC kita dapat merakit sendiri, dengan komponen yang diinginkan. Tetapi, permasalahan yang ada adalah untuk merakit MRC memerlukan waktu yang cukup lama. Waktu yang dibutuhkan cukup lama karena untuk memilih komponen yang sesuai dengan anggaran cukup sulit. Adanya masalah tersebut, maka dikembangkan aplikasi optimasi untuk membantu memilih komponen-komponen MRC dengan algoritma genetika. Aplikasi yang dikembangkan menggunakan metode Seleksi *Roulette Wheel, Two Point Crossover dan Random Mutation. Hasil uji coba ukuran populasi dan jumlah generasi terbaik adalah 30 dan 200. Nilai kombinasi parameter probabilitas crossover (Pc) dan probabilitas mutasi (Pm) mendapatkan rata-rata fitness tertinggi adalah Pc 0.9 dan Pm 0.1. Dapat disimpulkan bahwa algoritma genetika terbukti membuat proses pencarian kombinasi komponen menjadi lebih efisien dibandingkan dengan cara manual.*

Kata kunci: *algoritma genetika, optimasi, radio control, radio control buggy nitro offroad;*

1. PENDAHULUAN

Mobil *radio control* merupakan salah satu mainan yang berupa mobil-mobilan. Teknologi yang digunakan untuk mengendalikan adalah gelombang radio. Kelebihan dari teknologi ini adalah jangkauan untuk mengendalikan mainan lebih jauh. Mobil *radio control* sudah menggunakan bahan bakar seperti mobil sunggukan. Karena ketangguhan di medan dan sudah tidak menggunakan baterai, maka harga mainan ini masih relatif mahal. Seiring dengan perkembangan yang pesat *radio control,* minat masyarakat semakin meningkat untuk memilikinya.

Mobil radio control dapat dibeli sudah dalam bentuk mobil atau dapat merakit sendiri, jika merakit sendiri maka diperlukan banyak komponen yang harus disediakan. Untuk membentuk satu unit mobil radio control yang handal perlu pengalaman yang cukup. Selama ini pembeli sering mempunyai kesulitan dalam memilih komponen-komponen yang ada, sehingga sering kali dilakukan *trial and error*. Masalah yang dihadapi saat ini adalah belum adanya aplikasi yang mampu membantu pihak penjual ataupun pembeli dalam mendapatkan kombinasi komponen-komponen dengan kualitas terbaik dan biaya kecil. Dengan mengimplementasikan algoritma genetika dapat dijadikan sebagai solusi dari masalah tersebut.

Tujuan dari penelitian ini adalah dikembangkan aplikasi untuk pemilihan komponen mobil *radio control* buggy *nitro offroad* dengan kualitas terbaik sesuai dengan anggaran dan kriteria yang

diinginkan pembeli. Tenologi yang digunakan untuk menyelesaikan kasus ini adalah algoritma genetika, dengan metode *roulette wheel selection*, *two point* dan *random mutation*.

2. LANDASAN TEORI

Algoritma genetika adalah algoritma pencarian yang digunakan dalam perhitungan untuk mencari solusi perkiraan yang optimal (Padhy, 2005). Kekuatan utama Algoritma genetika adalah kemampuannya untuk menyelesaikan masalah kompleks dalam waktu yang relatif cepat. Dengan keutamaannya algoritma genetika layak diterapkan untuk permasalahan yang dihadapi oleh toko Rctronic.

Suatu pendekatan normatif untuk mengidentifikasikan penyelesaian terbaik dalam pengambilan keputusan dari suatu permasalahan adalah pengertian dari Optimasi. Penyelesaian permasalahan dalam teknik optimasi diarahkan untuk mendapatkan titik maksimum dan titik minimum dari fungsi yang dioptimalkan. Tujuan dari optimasi adalah untuk meminimumkan usaha yang diperlukan atau biaya operasional dan memaksimumkan hasil yang diinginkan (Witary dkk., 2013).

Algoritma Genetika merupakan algoritma pencarian *heuristic* yang didasarkan atas mekanisme evolusi biologis. Diperkenalkan oleh Holland (1975) yang digunakan dalam menyelesaikan masalah optimasi. Algoritma genetika mensimulasikan proses yang terjadi pada populasi alamiah yang merupakan hal penting dalam evolusi. Algoritma ini didasarkan pada proses genetik yang ada dalam makhluk hidup, yaitu individu secara terus-menerus mengalami perubahan gen untuk menyesuaikan dengan lingkungan hidupnya. Hanya individu-individu yang kuat yang mampu bertahan (Sutojo dkk., 2011). Algoritma genetika sangat baik diterapkan untuk masalah optimasi dan dapat digunakan untuk mencari pemecahan optimum (Muthmainnah dan Muntini, 2010).

Dalam algoritma genetika ada beberapa definisi penting yang harus diketahui untuk membangun suatu penyelesaian masalah. Diantaranya adalah sebagai berikut (Fadlisyah, 2009):

(1) *Gen* merupakan nilai yang menyatakan satuan dasar yang membentuk arti tertentu dalam satu kesatuan gen yang dinamakan kromosom.
(2) Kromosom / Individu merupakan gen-gen yang membentuk nilai tertentu dan menyatakan solusi yang mungkin dari suatu permasalahan.
(3) Populasi merupakan sekumpulan individu yang akan diproses bersama dalam satu kesatuan siklus evolusi.
(4) *Fitness* merupakan seberapa baik nilai dari suatu individu atau solusi yang didapatkan.
(5) Seleksi merupakan proses untuk mendapatkan calon induk yang baik
(6) *Crossover* merupakan proses pertukaran atau kawin silang gen-gen dari dua induk tertentu.
(7) Mutasi merupakan proses pergantian salah satu gen yang terpilih dengan nilai tertentu.
(8) Generasi merupakan urutan iterasi.
(9) *Offspring* merupakan kromosom baru yang dihasilkan setelah melewati suatu generasi.

Algoritma genetika memiliki 6 komponen penting, diantaranya (Kusumadewi, 2003):

(1) Teknik Pengkodean

Pengkodean adalah suatu teknik untuk merepresentasikan masalah ke dalam bentuk kromosom. Teknik pengkodean adalah bagaimana mengkodekan gen dari kromosom. Gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variabel. Ada 3 macam teknik pengkodean, yaitu biner, integer dan permutasi.

(2) Prosedur Inisialisasi

Ukuran populasi tergantung pada masalah yang ingin dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara *random*, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

(3) Evaluasi Fitness

Fungsi fitness untuk mengevaluasi fitness kromosom adalah sebagai berikut:

$$f(x) = \frac{((v/\max v)+(h/\max h))}{2}$$

Sedangkan rumus untuk mengevaluasi individu atau solusi kombinasi adalah sebagai berikut:

$$\mathrm{F} = \frac{f(x1)+f(x2)+\cdots+f(x9)}{9}$$

Keterangan dari rumus diatas adalah:

$f(x)$: Nilai fitness kromosom (0 – 1). 0 untuk nilai *fitness* terendah dan 1 untuk nilai *fitness* tertinggi.

F : Nilai fitness individu atau solusi (0 – 1). 0 untuk nilai *fitness* terendah dan 1 untuk nilai *fitness* tertinggi.

v : Value dari kualitas komponen

$\max v$: Maksimal value dari kualitas komponen

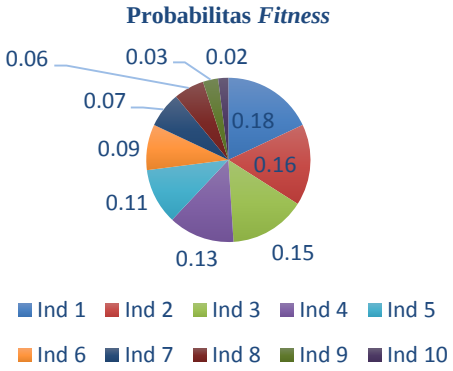
h : Harga dari sebuah komponen

$\max h$: Maksimal harga dari sebuah komponen

(4) Seleksi

Seleksi dilakukan untuk menentukan individu-individu mana saja yang akan dipilih untuk melakukan rekombinasi dan bagaimana *offspring* yang terbentuk dari individu-individu terpilih tersebut. Pencarian nilai *fitness* adalah langkah pertama yang dilakukan didalam proses seleksi. Nilai *fitness* itulah yang nantinya digunakan pada tahap seleksi berikutnya (Kusumadewi, 2003). Seleksi yang digunakan pada penelitian ini adalah seleksi *Roulette Wheel*. *Roulette Wheel* adalah sebuah metode seleksi yang menggunakan analogi dari *roulette wheel* untuk memilih individu-individu dari sebuah populasi. Semakin tinggi *fitness* dari sebuah individu maka semakin besar ruang yang didapatkannya [7]. Cara kerja metode ini adalah: Hitung total *fitness* semua individu, Hitung probabilitas seleksi masing-masing individu, Dari probabilitas tersebut, dihitung jatah interval masing-masing individu pada angka 0 sampai 1, Bangkitkan bilangan random antara 0 sampai, dan Dari bilangan *random* yang dihasilkan, tentukan urutan untuk populasi baru hasil dari proses seleksi.

Contoh dari gambar seleksi *roulette wheel* dapat dilihat sebagai berikut



Gambar 1: Seleksi Roulette Wheel

(5) Operator Genetika

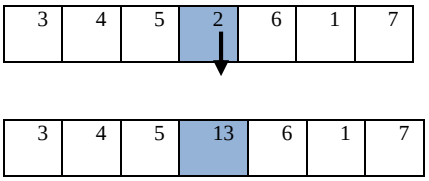
Operator genetika terdiri dari dua operator, yaitu *crossover* dan mutasi. *Crossover* bertujuan untuk menambahkan keanekaragaman individu dalam satu populasi. Mutasi berperan untuk menjaga keanekaragaman populasi dengan cara mengubah nilai dari satu atau beberapa gen dalam suatu individu.

Crossover adalah suatu proses dimana individu-individu dari sebuah populasi bertukar informasi genetik, diharapkan untuk menciptakan sebuah individu yang berisi bagian-bagian terbaik dari gen orangtuanya. Selama proses *crossover*, setiap individu dalam populasi dianggap siap untuk di *crossover*; disinilah digunakannya probabilitas *crossover* (Pc). Dengan membandingkan nilai Pc dengan nilai pembangkitan *random* per individu, kita bisa memilih apakah individu tersebut harus terproses *crossover* atau individu tersebut tidak terproses *crossover* (Jacobson dan Kanber, 2015): *Crossover* yang digunakan adalah *two point crossover* dan dapat dilihat pada tabel dibawah ini.

Tabel 1: Two Point Crossover

P1	1	3	5	4	2	1	6
P 2	3	1	2	7	6	4	5
O1	1	3	2	7	6	1	6
O2	3	1	5	4	2	4	5

Mutasi merupakan sebuah proses yang jarang terjadi, menggambarkan sebuah perubahan dalam gen. Mutasi dapat menyebabkan peningkatan yang signifikan pada *fitness* sebuah kromosom. Sering kali proses seleksi dan *crossover* menyebabkan susunan gen pada sebuah individu menjadi itu-itu saja. Dalam kondisi seperti itu, semua kromosom identik dan dengan demikian *fitness*nya pun sama dan sulit untuk ditingkatkan. Tugas mutasi adalah untuk menjamin agar tidak hilang keanekaragaman genetik (Negnevitsky, 2015). Mutasi yang digunakan adalah *random mutation* dan dapat dilihat pada Gambar 2.



Gambar 2: Random Mutation

Efek dari proses crossover dan mutasi sering membuat kita kehilangan individu terbaik kita. Oleh karena itu, kita membutuhkan operator yang dapat memberikan kita solusi dari masalah tersebut. Salah satu teknik dasar yang digunakan untuk mengatasi masalah tersebut adalah dengan selalu memungkinkan individu dengan fitness terbesar untuk ditambahkan ke generasi berikutnya. Proses yang mempertahankan yang terbaik untuk generasi selanjutnya disebut dengan Elitism. Elitism berkerja dengan mengurutkan individu berdasarkan fitness terbesar ke terkecil (Jacobson dan Kanber, 2015).

(6) Penentuan Parameter

Algoritma Genetika memiliki setidaknya beberapa parameter yang perlu diperhatikan selama proses. Diantaranya adalah *mutation rate* (Pm), *crossover rate* (Pc), *Population Size*

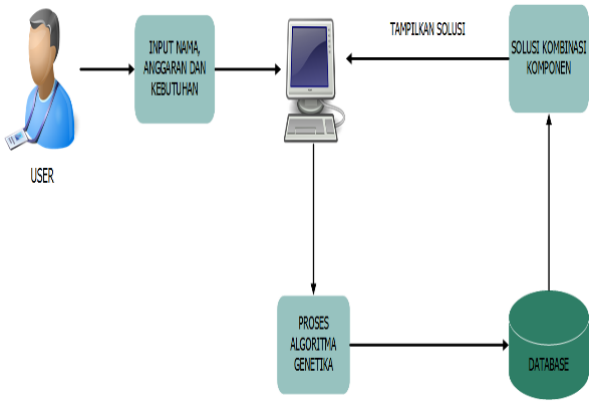
(Ukuran Populasi) (Jacobson dan Kanber, 2015). *Mutation Rate* adalah probabilitas dimana gen tertentu didalam kromosom solusi akan bermutasi. Secara teknis, tidak ada nilai yang sangat tepat untuk probabilitas ini. Namun, beberapa nilai biasanya akan menghasilkan hasil yang lebih baik dari yang lain. Nilai dari Pm itu sendiri adalah 0-1. Nilai Pm yang terlalu tinggi dapat menyebabkan terlalu banyak variasi genetik antara setiap generasi, yang menyebabkan berpeluang kehilangan solusi baik yang telah ditemukan dalam populasi sebelumnya. *Crossover Rate* adalah probabilitas dimana indiviidu yang ada didalam populasi akan saling bertukar informasi genetik. Nilai dari Pc itu sendiri adalah 0-1. Nilai Pc yang besar memungkinkan untuk menghasilkan banyak solusi yang baik dalam proses *crossover*. Nilai Pc yang rendah akan membantu untuk mempertahankan solusi yang lebih baik untuk generasi selanjutnya. *Population Size* atau kuran populasi adalah jumlah individu yang terdapat dalam populasi. Semakin besar ukuran populasi semakin besar pula ruang Algoritma Genetika untuk mendapatkan sampel solusi, ini akan membantu mendapatkan solusi yang lebih optimal. Ukuran populasi yang kecil mengakibatkan didapatkannya solusi yang kurang optimal.

3. RANCANGAN APLIKASI

Dalam kasus yang didapatkan bahwa jika pembeli memiliki anggaran sebanyak "X" maka pihak toko atau penjual akan melakukan pemilihan komponen sesuai dengan angaran. Dari komponen-komponen tersebut akan menjadi satu mobil *radio control*. Pada proses tersebut memakan waktu yang lama karena harus memilih komponen-komponen yang optimal dengan anggaran yang ada. Pada kasus tersebut pokok permasalahanya adalah mencari kombinasi komponen secara manual tidak efisien karena pihak penjual harus mengecek komponen satu persatu sampai menjadi sebuah kombinasi yang sesuai anggaran.

Untuk merakit satu mobil radio control diperlukan komponen seperti *engine, servo throttle, servo steering, fuel* dan *reciever battery. Engine* sebagai mesin utama dari mobile *radio control. Servo* sebagai pengatur pergerakan mobil ke arah kiri atau kanan. *Fuel* merupakan komponen yang digunakan untuk memompa bahan bakar. Dan *reciever* fungsinya digunakan untuk menerima sinyal gelompang dario dari pusat pengendali atau remote.

Dikembangkanya aplikasi ini dapat meringankan kerja pihak penjual untuk mendapatkan kombinasi komponen yang sesuai dengan keadaan yang ada secara cepat.

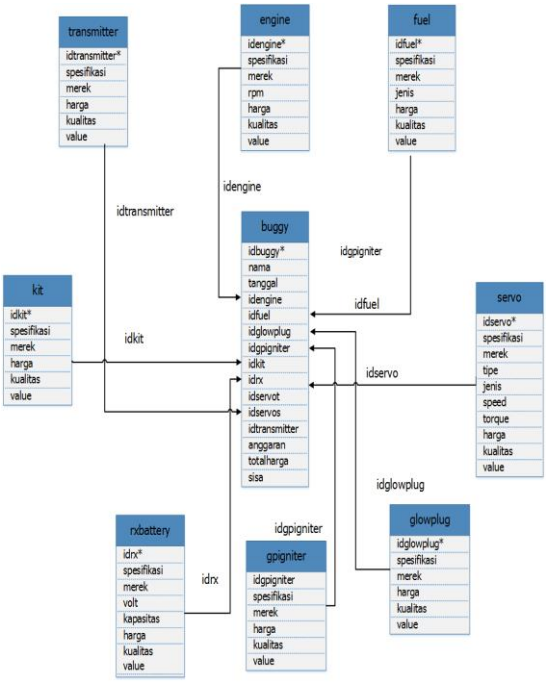


Gambar 3: Arsitektur Sistem.

Pada Gambar 3 dijelaskan bahwa alur proses dalam aplikasi yang dikembangkan ada 3 bagian proses, yaitu:

(1) *User* menginput nama pembeli, anggaran yang dimiliki dan kebutuhan atas mobil *radio control* dari pembeli.
(2) Proses selanjutnya adalah melakukan perhitungan untuk mendapatkan komponen yang dibutuhkan. Metode yang digunakan adalah algoritma genetika.
(3) Setelah proses (2) selesai, maka daftar komponen yang dibutuhkan diberikan ke user sebagai rekomendasi.

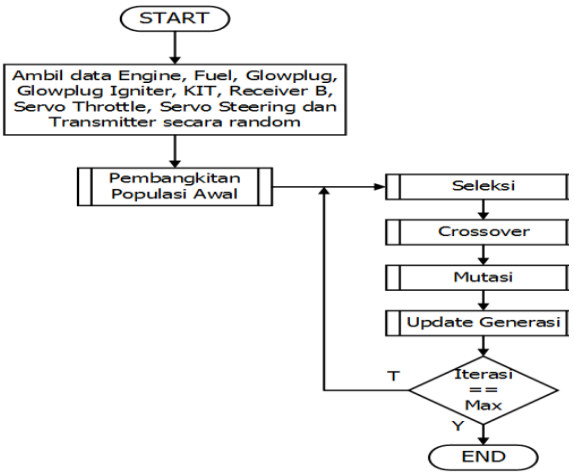
Dari hasil rancangan didapatkan model struktur data. Struktur data yang digunakan adalah model data relasi. Untuk memperlihatkan struktur data digunakan *diagram entity relationship.* Dalam penelitian ini software basis data yang digunakan adalah Mysql. Struktur data yang dibentuk dapat dilihat pada Gambar 4.



Gambar 4: Logical Record Structure (LRS).

Pada Gambar 4 dijelaskan bahwa data-data yang digunakan untuk mendukung proses pemilihin komponen seperti data *buggy*, *kit*, *battery*, *transmitter*, *servo*, *fuel* dan *engine*. Data-data tersebut yang digunakan untuk membentuk satu mobil *radio control.*

Tahapan yang dilakukan dalam algoritma genetika adalah pengambilan data, pembangkitan populasi awal, melakukan seleksi, *crossover*, mutasi dan update generasi. *Flowchart* proses algoritma genetika dapat dilihat pada Gambar 5.



Gambar 5: Flowchart Proses Algoritma Genetika

Logika dalam proses algoritma genetika adalah sebagai berikut:

1. Mulai
2. Ambil data *Engine*, *Fuel*, *Glowplug*, *Glowplug Igniter*, KIT, *Receiver* B, Servo *Throttle*, Servo *Streeing* dan *Transmitter* secara *random*
3. Pembangkitan populasi awal
4. Seleksi
5. *Crossover*
6. Mutasi
7. *Update* Generasi
8. *If* Iterasi == Max *Then*
9. Selesai
10. *Else*
11. Kembali ke proses 4
12. *End If*
13. Selesai

4. HASIL DAN PEMBAHASAN

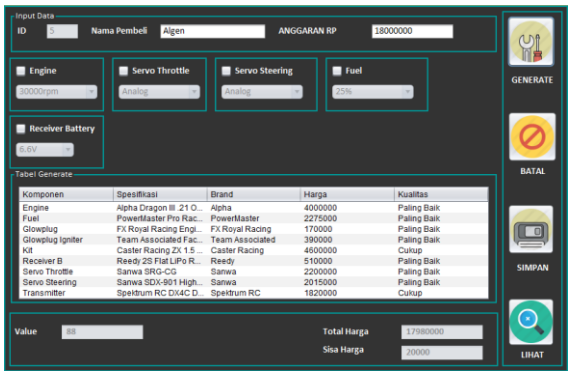
4.1 Fungsi untuk Pemilihan Komponen

Aplikasi yang dihasilkan dapat digunakan untuk melakukan pemilihan komponen-komponen yang optimal berdasarkan kondisi keuangan pembeli. Tapi sebelum dapat menggunakan aplikasi ini user harus memasukan beberapa kompenen terlebih dahulu. Jadi semua komponen harus di input ke fungsi yang telah disediakan.

Pada aplikasi ini fungsi utama yang digunakan untuk memilih komponen yang paling optimal adalah fungsi pada menu "Generate" pada aplikasi. Pada menu ini user tinggal memasukan data pembeli dan harga yang dimiliki.

Di dalam menu "Generate" terdapat tombol "*Generate".* Tombol ini digunakan untuk kombinasi semua komponen secara otomatis dengan teknik *random*. Tetapi, jika pembeli menginginkan beberapa komponen yang spesifik maka user dapat memilih beberapa *checkbox* yang telah tersedia.

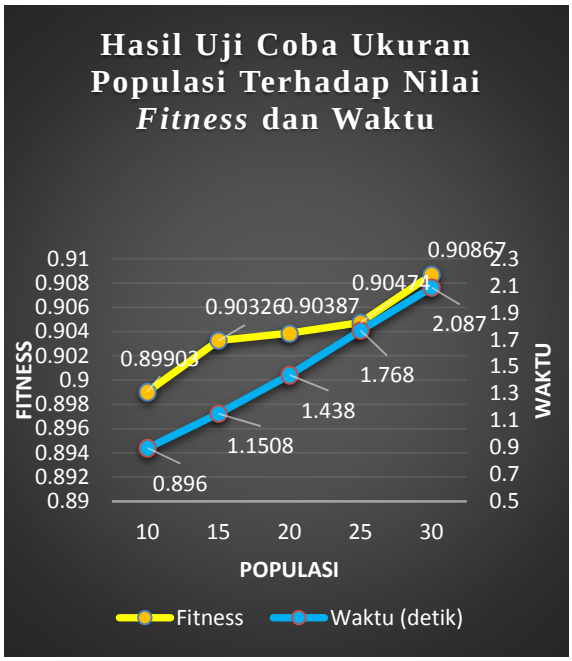
Apabila *user* sudah yakin dengan inputan yang telah dipilih, *user* dapat menekan tombol *Generate* untuk melihat hasil dari kombinasi komponen. Jika sudah tampil kombinasi komponen didalam tabel *generate*, *user* dapat memilih tombol "Simpan" jika ingin menyimpan. Kemudian, memilih tombol "Lihat" jika ingin melihat data pembeli yang telah tersimpan, tampilan aplikasi dapat dilihat pada Gambar 6.



Gambar 6: Tampilan Layar Form Generate

4.2 Uji Coba Ukuran Populasi

Pada pengujian berikut, diukur ukuran populasi untuk mengetahui berapa ukuran populasi yang menghasilkan rata-rata nilai *fitness* tinggi. Ukuran populasi yang diuji yaitu 10, 15, 20, 25 dan 30 individu dengan masing-masing 10 kali pengujian. Nilai probabilitas *crossover* dan probabilitas mutasi yang digunakan yaitu 0.7 dan 0.3, lalu jumlah generasi yang digunakan adalah 100, hasil dari pengujian dapat dilihat pada Gambar 7.



Gambar 7: Uji Coba Ukuran Populasi

Pada Gambar 7 diperlihatkan bahwa hasil uji coba didapatkan bahwa *fitness* tertinggi didapat pada ukuran populasi 30 individu dengan nilai *fitness* dan waktunya adalah 0.90687 dan 2.087 detik. Sedangkan nilai *fitness* terendah didapat pada ukuran populasi 10 dengan nilai *fitness* dan waktunya adalah 0.89903 dan 0.896 detik. Dapat disimpulkan bahwa ukuran populasi dapat membengaruhi besar kecilnya *fitness* yang didapat, namun ukuran populasi juga dapat membuat proses algoritma genetika membutuhkan waktu yang semakin lama.

4.3 Uji Coba Jumlah Generasi

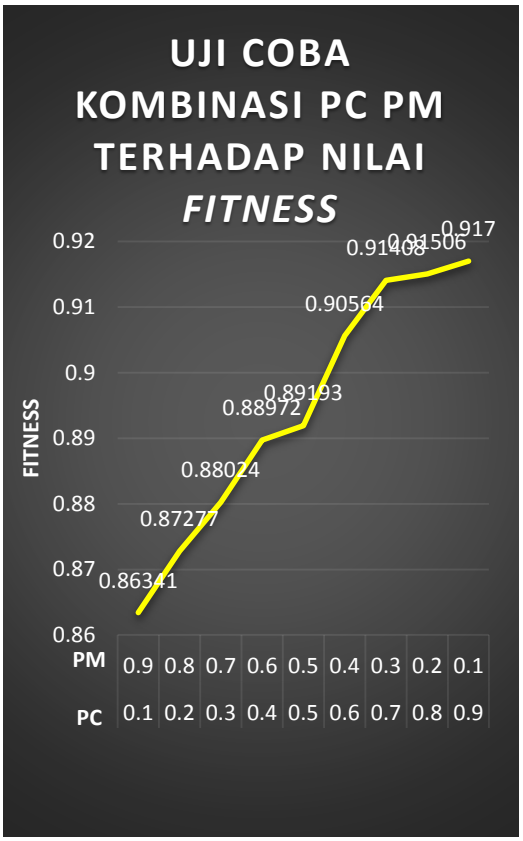
Pada pengujian berikut, diukur banyaknya jumlah generasi pada proses algoritma genetika untuk mengetahui berapa ukuran generasi yang menghasilkan rata-rata nilai *fitness* tinggi. Jumlah generasi yang diuji adalah 100, 125, 150, 175 dan 200 dengan masing-masing 10 kali pengujian. Sesuai dengan hasil uji coba ukuran populasi, maka ukuran populasi yang digunakan adalah sebesar 30 individu. probabilitas *crossover* dan probabilitas mutasi yang digunakan adalah 0.7 dan 0.3, hasil dari pengujian dapat dilihat pada Gambar 8.

Gambar 8: Uji Coba Jumlah Generasi

Pada Gambar 7 diperlihatkan bahwa hasil uji coba didapatkan bahwa *fitness* tertinggi didapat pada generasi yang berjumlah 200 dengan nilai *fitness* dan waktunya adalah 0.91274 dan 6.087 detik. Sedangkan nilai *fitness* terendah ada pada generasi yang berjumlah 100 dengan nilai *fitness* dan waktunya adalah 0.90872 dan 3.244 detik. Dengan kata lain, jumlah generasi dapat mempengaruhi Algoritma Genetika untuk menghasilkan nilai *fitness* yang semakin baik, namun juga dapat membuat proses Algoritma Genetika semakin lama.

4.4 Hasil Uji Coba Kombinasi Pc dan Pm

Pada pengujian berikut, akan diukur kombinasi nilai Pc dan Pm terhadap nilai *fitness* untuk mengetahui kombinasi nilai Pc dan Pm yang menghasilkan *fitness* tertinggi. Sesuai dengan hasil uji coba ukuran populasi dan jumlah generasi pada uji coba sebelumnya, didapat ukuran populasi yang optimal adalah 30 individu dan jumlah generasinya adalah 200. Maka, digunakan ukuran populasi dan jumlah generasi tersebut. Kombinasi Pc dan Pm yang akan diuji adalah 0.1:0.9, 0.2:0.8, 0.3:0.7, 0.4:0.6, 0.5:0.5, 0.6:0.4, 0.7:0.3, 0.8:0.2 dan 0.9:0.1 dengan 10 kali pengujian pada masing-masing kombinasi, hasil dari pengujian dapat dilihat Gambar 9.



Gambar 9: Uji Coba Kombinasi Pc dan Pm

Pada Gambar 7 diperlihatkan bahwa hasil uji coba didapatkan bahwa *fitness* tertinggi didapatkan pada kombinasi Pc 0.9 dan Pm 0.1. Maka, pada aplikasi ini kombinasi Pc dan Pm yang digunakan adalah 0.9 dan 0.1.

4.5 Evaluasi Program

Dari hasil evaluasi dari program maka aplikasi yang dikembangkan memiliki kelebihan seperti:

(1) Aplikasi ini dapat membuat proses pencarian kombinasi komponen lebih cepat dibandingkan dengan cara manual.
(2) Aplikasi ini dapat menyimpan hasil kombinasi komponen.
(3) Aplikasi ini memiliki *interface* yang mudah untuk digunakan oleh *user*.

Selain memiliki kelebihan, juga memiliki kelemahan. Dari hasil evaluasi aplikasi yang dikembangkan masih memiliki beberapa kekurangan, antara lain:

(1) Dikarenakan Algoritma Genetika mencari solusi-solusi secara *random*, maka solusi yang sekarang belum tentu sama dengan solusi sebelumnya dan setelahnya.
(2) Jumlah data perkomponen yang masih perlu diperbanyak agar mendapatkan solusi yang lebih bervariasi.

5. KESIMPULAN

Berdasarkan pembahasan dan hasil pengujian, maka dapat disimpukan bahwah algoritma genetika dapat diimplementasikan ke dalam proses pencarian kombinasi komponen terbaik untuk perakitan *radio control* buggy *nitro offroad*. Aplikasi ini membuat proses pemilihan komponen lebih efisien dari sisi waktu yang dibutuhkan memilih komponen. Aplikasi ini hanya memberikan saran untuk kombinasi komponen. Hasil uji coba menunjukkan bahwa nilai parameter algoritma genetika berpengaruh terhadap jalannya aplikasi dan solusi yang didapat. Seperti besar kecilnya *fitness* dan waktu proses pencarian solusi.

6. DAFTAR PUSTAKA

[1] Padhy, N., P., 2005, *Artificial Intelligence and Intelligent Systems*. New Delhi: Oxford University Press.
[2] Witary, V., Rachmat, N., dan Inayatullah, 2013, Optimasi Penjadwalan Perkuliahan dengan Menggunakan Algoritma Genetika (Studi Kasus : AMIK MDP , STMIK GI MDP dan STIE MDP), *J. STMIK GI MDP*, pp. 1–7.
[3] Sutojo, T., Mulyanto, E., dan Suhartono, V., 2011, *Kecerdasan Buatan Pertama*. Yogyakarta: Andi.
[4] Muthmainnah, dan Muntini, M., S., 2010, Penerapan Algoritma genetik Untuk Optimasi Transfer Daya, *Pros. Semin. Nas. Sains*, pp. 189–195.
[5] Fadlisyah., Arnawan., Faisal., 2009, *Algoritma Genetika*, 1st ed. Yogyakarta: Graha Ilmu.
[6] Kusumadewi, S., 2003, *Artificial Intelligence (Teknik dan Aplikasinya)*, 1st ed. Yogyakarta: Graha Ilmu.
[7] Jacobson, L., dan Kanber, B., 2015, *Genetic Algorithms in Java Basics*. Apress.
[8] Negnevitsky, M., 2015, *Artificial Intelligence: A Guide to Intelligent Systems*, 2nd ed. Addison-Wesley.