

## ALGORITMA GENETIKA UNTUK MENYELESAIKAN COIN PROBLEM: APLIKASI PADA MESIN ATM

**Elliana Gautama**

Program Studi Sistem Informasi Institute Keuangan Perbankan dan Informatika Perbanas  
e-mail: <sup>1</sup>[elliana@perbanas.id](mailto:elliana@perbanas.id); <sup>2</sup>[gautamaelly@gmail.com](mailto:gautamaelly@gmail.com)

### **Abstract**

*At the present in Indonesia an Automatic Teller Machine (ATM) only able to withdraw specific denomination, such as Rp. 50.000 or Rp. 100.000. Sometimes, client wants to withdraw specific denomination but the ATM only provides Rp. 100.000 or other denomination, then the client have to switch to other ATM and sometime they have to walk so far to withdraw specific denomination in ATM as they wants. The main obstacle at present is the absence of an algorithm application of ATM that can cope problem to solve value of money to combination of denomination of money as client wants. This problem was called Coin Change Problem and it is usually used to find optimal combination from many different money combinations. This paper will use the theory of Genetic Algorithms to solve the Coin Change Problem. Genetic algorithm is computational algorithm was inspired by evolution theory then was adopted from it self and it was become computational algorithm to find solution from problem in nature's way. One of application genetic algorithm is the combination of optimization problems, which get a value of an optimal solution to a problem that has many possible solutions. With this Genetic Algorithm in this paper is the resolve to the Coin Change Problem on application of ATM, so that an ATM can issued a optimal combination from many different money combinations. In this study developed an algorithm using Genetic Algorithm which can solve the problems of Coin Problem in ATM applications, resulting in an ATM can issue some money with a optimum combination specific denomination.*

**Keywords :** Genetic Algorithms , Coin Problem , Optimization Combination , ATM

### **Abstrak**

*Pada saat ini di Indonesia, sebuah mesin ATM untuk fasilitas penarikan uang tunai hanya dapat mengeluarkan satu jenis pecahan mata uang saja, misalnya pecahan 50.000 rupiah saja atau pecahan 100.000 rupiah saja, sedangkan ada saatnya seorang nasabah memerlukan beberapa jenis pecahan mata uang dalam satu kali transaksi tanpa berpindah mesin ATM. Mesin ATM saat ini walaupun hanya dapat mengeluarkan satu jenis pecahan mata uang saja, sebenarnya memiliki tempat untuk beberapa pecahan mata uang. Permasalahan untuk memecahkan kasus tersebut di dalam algoritma disebut dengan Coin Problem. Coin Problem ini berguna untuk memecahkan masalah dalam mencari jumlah kombinasi pecahan optimum untuk mendapatkan sejumlah nilai uang yang diinginkan dengan kombinasi beberapa pecahan mata uang yang berbeda. Untuk memecahkan masalah Coin Problem ini penulis menggunakan Algoritma Genetika. Algoritma genetika adalah algoritma komputasi yang diinspirasi teori evolusi yang kemudian diadopsi menjadi algoritma komputasi untuk mencari solusi suatu permasalahan dengan cara yang lebih "alamiah". Berdasarkan hasil pengujian yang telah dilakukan terbukti bahwa dengan menggunakan Algoritma Genetika dapat menyelesaikan permasalahan Coin Problem dalam Aplikasi Mesin ATM, sehingga dalam satu mesin ATM dapat mengeluarkan kombinasi jumlah pecahan mata uang yang optimum.*

**Kata Kunci :** Algoritma Genetika, Coin Problem, Optimasi Kombinasi, mesin ATM.

## **1. Pendahuluan**

Ketergantungan masyarakat akan mesin ATM pada saat ini sangat tinggi, karena itu semua bank saat ini berlomba-lomba meningkatkan pelayanannya dengan menyediakan mesin ATM sebanyak-banyaknya di semua lokasi yang strategis. Dari data yang dikumpulkan Infobanknews.com[6], per akhir tahun 2012, posisi bank-bank yang memiliki mesin ATM

lebih dari 2 ribu unit di urutan pertama adalah PT Bank Rakyat Indonesia (Persero) Tbk (BRI) dengan 14.292 unit, disusul PT Bank Central Asia Tbk (BCA) dengan 12.026 unit, PT Bank Mandiri (Persero) Tbk sebanyak 10.985 unit, PT Bank Negara Indonesia (Persero) Tbk (BNI) sebanyak 8.227 unit, dan PT Bank CIMB Niaga Tbk (CIMB Niaga) 2.257 unit. Pada tahun 2012, sudah 47 ribu mesin ATM yang tersebar di seluruh Indonesia dengan transaksi per harinya sebesar Rp. 7 triliun dan transaksi per mesin per harinya adalah Rp. 157 juta.

Terjadi peningkatan pada triwulan satu tahun 2013, PT Bank Rakyat Indonesia (Persero) Tbk (BRI) memiliki 14.397 unit ATM, PT Bank Centra Asia Tbk (BCA) sebanyak 12.026 unit, PT Bank Mandiri (Persero) Tbk sebanyak 10.986 unit, PT Bank Negara Indonesia (Persero) Tbk (BNI) 8.279 unit, PT Bank CIMB Niaga Tbk (CIMB Niaga) 2.257 unit. Bank-bank terbesar ini secara signifikan dan kontinyu melakukan investasi ATM. Pangsa bank-bank ini 67% dari seluruh total ATM. Dari data di atas menunjukkan perkembangan yang sangat pesat dan semakin meningkatnya kebutuhan masyarakat akan mesin ATM.

Mesin ATM merupakan mesin yang memberikan kemudahan kepada nasabah dalam melakukan transaksi perbankan secara otomatis selama 24 jam dalam 7 hari termasuk hari libur, ATM (*Automatic Teller Machine*) atau dalam bahasa Indonesia menjadi Anjungan Tunai Mandiri adalah suatu sistem pelayanan bank secara elektronik yang melaksanakan fungsi teller secara otomatis, dimana kini banyak bank yang menerapkan sistem ATM guna menambah kecepatan dalam melayani kebutuhan nasabah akan uang tunai khususnya, disamping memberi kenyamanan bagi nasabah dalam melakukan transaksi perbankan pada umumnya[1].

Penggunaan ATM di Indonesia sepanjang tahun 2013 dengan nilai transaksi sebesar Rp 3.797.400.000.000 (100,0%) digunakan untuk pengambilan uang (tarik tunai) sebesar 44,1%, transfer intrabank sebesar 39,7%, transfer antarbank 12,3%, belanja 3,9% [2].

Kondisi saat ini sebuah mesin ATM untuk fasilitas penarikan uang tunainya hanya dapat mengeluarkan satu jenis pecahan mata uang saja, misalnya pecahan 50.000 rupiah saja atau pecahan 100.000 rupiah saja, sedangkan ada saatnya seorang nasabah memerlukan beberapa jenis pecahan mata uang dalam satu kali transaksi tanpa berpindah mesin ATM, kenyataannya saat ini jika seseorang ingin mengambil uang dalam pecahan yang berbeda maka orang tersebut harus mencari beberapa mesin ATM dan melakukan beberapa kali transaksi, hal ini sangat merugikan tidak hanya kerugian waktu tetapi juga nasabah tersebut harus menanggung biaya jika transaksi dilakukan pada mesin ATM yang bukan bank dari nasabah tersebut. Mesin ATM saat ini walaupun hanya dapat mengeluarkan satu jenis pecahan mata uang saja, sebenarnya memiliki tempat untuk beberapa pecahan mata uang.

Permasalahan untuk menyelesaikan kasus di atas di dalam algoritma disebut dengan Coin Problem. Coin Problem ini berguna untuk menyelesaikan masalah dalam mencari jumlah pecahan optimum untuk mendapatkan sejumlah nilai uang yang diinginkan, di mana pecahan-pecahan uang tersebut berjumlah tidak terbatas. Contoh masalah : Jika seorang nasabah ingin melakukan penarikan tunai sebesar Rp. 270.000,-, dan mesin ATM yang dirancang memiliki pecahan mata uang Rp. 100.000,-, Rp. 50.000,-, Rp. 20.000,- dan Rp.

10.000,-, maka nasabah yang bersangkutan akan menerima uang dengan beberapa kombinasi pecahan, kombinasi pertama sebanyak 2 lembar Rp. 100.000,-, 1 lembar Rp. 50.000,- dan 1 lembar Rp. 20.000,-, kombinasi kedua sebanyak 2 lembar Rp. 100.000,-, 1 lembar Rp. 50.000,- dan 2 lembar Rp. 10.000,-, dan masih banyak kombinasi lainnya. Ada banyak algoritma dan metode matematika untuk mencari kombinasi yang optimum dalam menyelesaikan Coin Problem ini, salah satunya adalah dengan menggunakan Algoritma Genetika. Algoritma Genetika merupakan salah satu metode heuristik yang merupakan cabang dari evolutionary algorithm, yaitu suatu teknik untuk memecahkan masalah-masalah optimasi yang rumit dengan menirukan proses evolusi makhluk hidup. Algoritma ini banyak digunakan dalam bidang fisika, biologi, ekonomi sosiologi dan lain-lain yang sering menghadapi masalah optimasi dengan model matematika yang kompleks atau bahkan sulit dibangun [3].

Oleh karena itu, pada penelitian ini dikembangkan Algoritma Genetika yang dipakai untuk mencari optimasi dalam aplikasi yang akan diterapkan dalam mesin ATM yang dapat membantu untuk memudahkan seorang nasabah dalam mengambil uang dengan beberapa pecahan mata uang yang berbeda dengan satu kali transaksi untuk mesin ATM yang sama. Sehingga dapat meningkatkan fungsionalitas sebuah mesin ATM yang pastinya akan memiliki keunggulan dibanding dengan mesin ATM yang lainnya. Hal ini tentunya akan sangat menguntungkan bagi sebuah bank. Beberapa batasan yang digunakan dalam penelitian ini adalah aplikasi masih berupa simulasi yang belum diterapkan pada mesin ATM sesungguhnya. Pecahan mata uang yang disimulasikan adalah pecahan Rp. 100.000, Rp. 50.000, Rp. 20.000 dan Rp. 10.000 dan kombinasi yang dihasilkan adalah kombinasi minimal dan kombinasi maksimal.

## **2. Tinjauan Pustaka**

### **2.1. Penelitian Terdahulu**

Penelitian ini dilakukan berdasarkan penelitian sebelumnya yang dilakukan oleh Richa Garg dan Saurabh Mittal [9] yang membahas mengenai optimasi dengan menggunakan Algoritma Genetika dan penelitian Dita Anindhika[7] membahas mengenai kompleksitas algoritma untuk menyelesaikan persoalan penukaran koin dengan menggunakan Algoritma Greedy, serta penelitian Agus Wahyu Widodo dan Wayan Firdaus Mahmudy[8] yang membahas mengenai bagaimana menerapkan Algoritma Genetika pada Sistem Rekomendasi Wisata Kuliner. Pada penelitian ini penulis menerapkan algoritma genetika untuk menyelesaikan coin problem.

### **2.2. Coin Problem**

Misalnya terdapat A nilai uang yang ditukar. Dengan himpunan koin  $\{d_1, d_2, d_3, \dots, d_n\}$  dan himpunan solusi  $X = \{x_1, x_2, x_3, \dots, x_n\}$  dimana  $x_i = 1$  jika dipilih dan  $x_i = 0$  jika tidak dipilih. Maka objektif persoalan di atas adalah:

$$\text{Minimasi } F = \sum_{i=1}^n (x_i) \quad (\text{fungsi obyektif})$$

Dengan kendala  $\sum_{i=1}^n (d_i x_i) = A$

Contoh : pecahan uang logam Indonesia terdiri dari 1000, 500, 200, 100, 50. Uang senilai 1150 bisa dipecah menjadi beberapa kombinasi yaitu :

Tabel 1. Kombinasi Pecahan 1150

Kombinasi	1000	500	200	100	50	Jumlah Keping
1	0	0	0	11	1	12
2	1	0	0	1	1	3
3	0	2	0	1	1	4
4	0	1	3	0	1	5
5	0	1	1	4	1	7
6	0	1	2	2	1	6
7	0	1	0	6	1	8

Dari beberapa kombinasi di atas terlihat jumlah kombinasi yang maksimal adalah pada kombinasi ke 1 yaitu 12 keping dengan kombinasi 11 keping 100, dan 1 keping 50.

Sedangkan kombinasi minimal adalah pada kombinasi ke 2 yaitu 3 keping dengan kombinasi 1 keping 1000, 1 keping 100, dan 1 keping 50.

### 2.3. Algoritma Genetika

Algoritma genetika yang dikembangkan oleh Goldberg adalah algoritma komputasi yang diinspirasi teori evolusi Darwin yang menyatakan bahwa kelangsungan hidup suatu makhluk dipengaruhi aturan “yang kuat adalah yang menang”. Darwin juga menyatakan bahwa kelangsungan hidup suatu makhluk dapat dipertahankan melalui proses reproduksi, *crossover*, dan mutasi. Konsep dalam teori evolusi Darwin tersebut kemudian diadopsi menjadi algoritma komputasi untuk mencari solusi suatu permasalahan dengan cara yang lebih “alamiah” [3].

Sebuah solusi yang dibangkitkan dalam algoritma genetika disebut sebagai *chromosome*, sedangkan kumpulan *chromosome-chromosome* tersebut disebut sebagai **populasi**. Sebuah *chromosome* dibentuk dari komponen-komponen penyusun yang disebut sebagai gen dan nilainya dapat berupa bilangan numerik, biner, simbol ataupun karakter tergantung dari permasalahan yang ingin diselesaikan. *Chromosome-chromosome* tersebut akan berevolusi secara berkelanjutan yang disebut dengan **generasi**. Dalam tiap generasi *chromosome-chromosome* tersebut dievaluasi tingkat keberhasilan nilai solusinya terhadap

masalah yang ingin diselesaikan (fungsi\_objektif) menggunakan ukuran yang disebut dengan *fitness*. Untuk memilih *chromosome* yang tetap dipertahankan untuk generasi selanjutnya dilakukan proses yang disebut dengan **seleksi**. Proses seleksi *chromosome* menggunakan konsep aturan evolusi Darwin yang telah disebutkan sebelumnya yaitu *chromosome* yang mempunyai nilai *fitness* tinggi akan memiliki peluang lebih besar untuk terpilih lagi pada generasi selanjutnya.

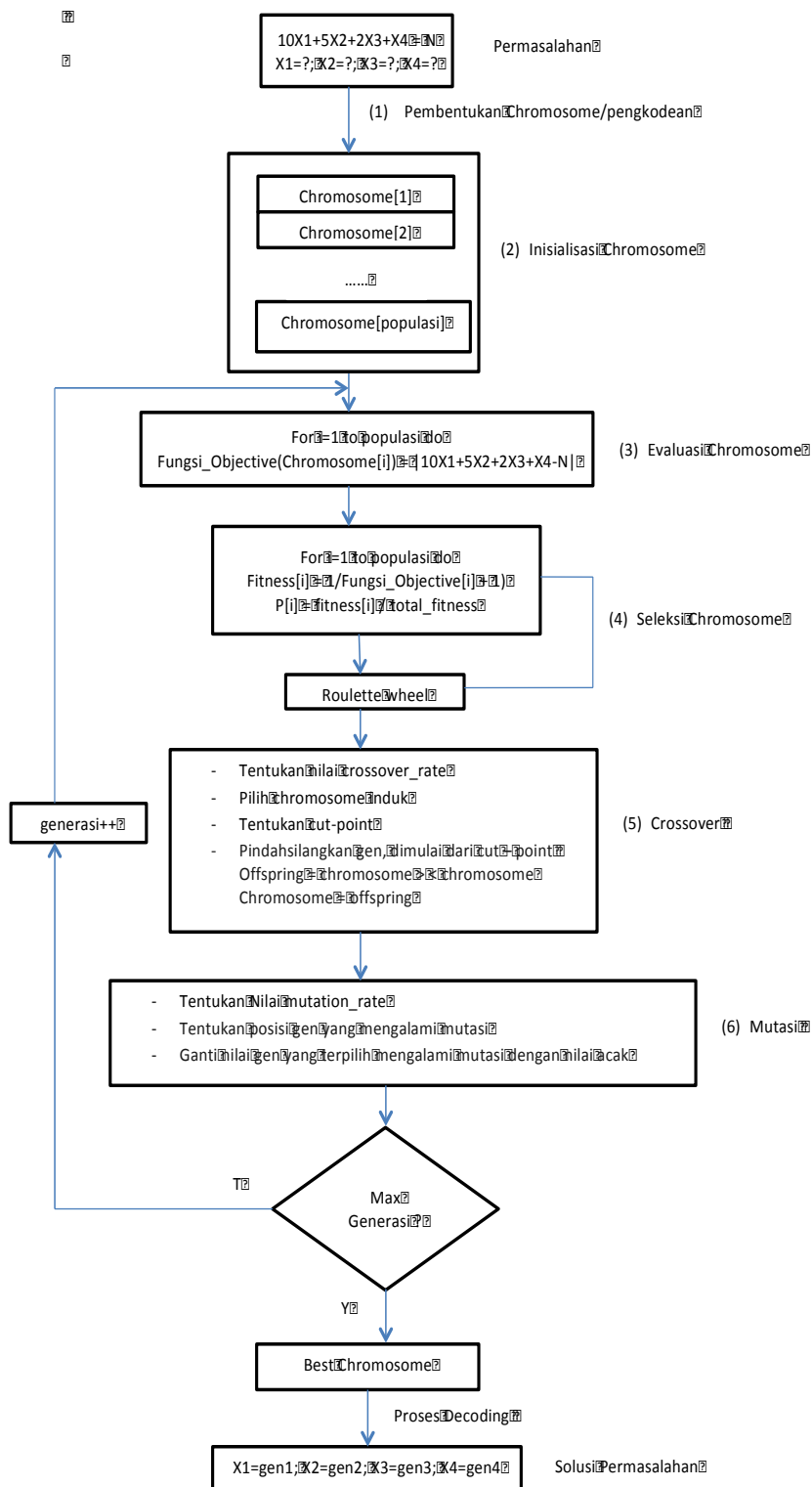
*Chromosome-chromosome* baru yang disebut dengan *offspring*, dibentuk dengan cara melakukan perkawinan antar *chromosome-chromosome* dalam satu generasi yang disebut sebagai proses *crossover*. Jumlah *chromosome* dalam populasi yang mengalami *crossover* ditentukan oleh parameter yang disebut dengan *crossover\_rate*. Mekanisme perubahan susunan unsur penyusun makhluk hidup akibat adanya faktor alam yang disebut dengan mutasi direpresentasikan sebagai proses berubahnya satu atau lebih nilai gen dalam *chromosome* dengan suatu nilai acak. Jumlah gen dalam populasi yang mengalami mutasi ditentukan oleh parameter yang dinamakan *mutation\_rate*. Setelah beberapa generasi akan dihasilkan *chromosome-chromosome* yang nilai gen-gennya konvergen ke suatu nilai tertentu yang merupakan solusi terbaik yang dihasilkan oleh algoritma genetika terhadap permasalahan yang ingin diselesaikan [3][4].

### 3. Metode Penelitian

Pada penelitian ini penulis menggunakan Algoritma Genetika untuk menyelesaikan Coin Problem, Tahapan Algoritma Genetika yang digunakan adalah sebagai berikut [5][9]:

- a. Pembentukan *Chromosome*/Pengkodean
- b. Inisialisai *Chromosome*
- c. Evaluasi *Chromosome*
- d. Seleksi *Chromosome*
- e. *Crossover*
- f. Mutasi
- g. Proses *Decoding*
- h. Solusi Permasalahan

Dalam rancangan simulasi aplikasi untuk fasilitas tarikan tunai pada mesin ATM, diasumsikan terdapat 4 pecahan mata uang yaitu pecahan Rp. 100.000, Rp. 50.000, Rp. 20.000 dan Rp. 10.000, sehingga dapat dibuat persamaan sebagai berikut  $100.000X_1 + 50.000X_2 + 20.000X_3 + 10.000X_4 = N$ , agar nilai tidak terlalu besar maka persamaan dibuat menjadi  $10X_1 + 5X_2 + 2X_3 + X_4 = N$ , akan dicari nilai  $X_1$ ,  $X_2$ ,  $X_3$ , dan  $X_4$  yang memenuhi persamaan di atas, dimana nilai  $N$  adalah jumlah nominal yang dimasukan oleh nasabah. Diagram alir dari algoritma genetika untuk menyelesaikan permasalahan di atas dapat dilihat pada gambar 1.



Gambar 1. Diagram Alir Algoritma Genetika

#### 4. Hasil dan Pembahasan

Penjelasan mengenai langkah-langkah penyelesaian permasalahan di atas menggunakan algoritma genetika adalah sebagai berikut:

##### a. Pembentukan *Chromosome*

Karena yang dicari adalah nilai  $X_1, X_2, X_3, X_4$  maka variabel  $X_1, X_2, X_3, X_4$  dijadikan sebagai gen-gen pembentuk *chromosome*. Batasan nilai variabel  $X_1, X_2, X_3, X_4$ ,  $N$  adalah bilangan *integer*  $\geq 0$  dan variabel  $X_1, X_2, X_3, X_4 \leq N$ .

##### b. Inisialisasi

Proses inisialisasi dilakukan dengan cara memberikan nilai awal gen-gen dengan nilai acak sesuai batasan yang telah ditentukan. Jumlah populasi yang diambil dalam penelitian ini ada 200 :

##### c. Evaluasi *Chromosome*

Permasalahan yang ingin diselesaikan adalah nilai variabel  $X_1, X_2, X_3$  dan  $X_4$  yang memenuhi persamaan  $10X_1 + 5X_2 + 2X_3 + X_4 = N$ , untuk nilai  $N$  diasumsikan bernilai 50 sehingga persamaan menjadi  $10X_1 + 5X_2 + 2X_3 + X_4 = 50$  maka fungsi\_objektif yang dapat digunakan untuk mendapatkan solusi adalah *fungsi\_objektif(chromosome)* =

$$| (10X_1 + 5X_2 + 2X_3 + X_4) - 50 |$$

##### d. Seleksi *Chromosome*

Proses seleksi dilakukan dengan cara membuat *chromosome* yang mempunyai fungsi\_objektif kecil mempunyai kemungkinan terpilih yang besar atau mempunyai nilai probabilitas yang tinggi. Untuk itu dapat digunakan fungsi

$$fitness = (1 / (1 + fungsi\_objektif))$$

fungsi\_objektif perlu ditambah 1 untuk menghindari kesalahan program yang diakibatkan pembagian oleh 0.

Rumus untuk mencari probabilitas :

$$P[i] = fitness[i] / total\_fitness$$

Dari probabilitas di atas, *chromosome* dengan *fitness* paling besar mempunyai probabilitas untuk terpilih pada generasi selanjutnya lebih besar dari *chromosome* lainnya. Untuk proses seleksi digunakan *roulette wheel*, karena itu harus mencari dahulu nilai *cumulative* probabilitasnya:

Setelah dihitung *cumulative* probabilitasnya maka proses seleksi menggunakan *roulette-wheel* dapat dilakukan. Prosesnya adalah dengan membangkitkan bilangan acak  $R$  dalam range 0-1.

Jika  $R[k] < C[1]$  maka pilih *chromosome* 1 sebagai induk, selain itu pilih *chromosome* ke- $k$  sebagai induk dengan syarat  $C[k-1] < R < C[k]$ . Putar *roulette wheel* sebanyak jumlah populasi yaitu 200 kali (bangkitkan bilangan acak  $R$ ) dan pada tiap putaran, pilih satu *chromosome* untuk populasi baru. *Chromosome* baru hasil seleksi adalah :

##### e. Crossover

Setelah proses seleksi maka proses selanjutnya adalah proses *crossover*. Metode yang digunakan salah satunya adalah *one-cut point*, yaitu memilih secara acak satu posisi dalam



*chromosome* induk kemudian saling menukar gen. *Chromosome* yang dijadikan induk dipilih secara acak dan jumlah *chromosome* yang mengalami *crossover* dipengaruhi oleh parameter *crossover\_rate*( *pc* ).

*Pseudocode* untuk proses *crossover* adalah sebagai berikut.

Begin

$k \leftarrow 0$ ;

    While ( $k < \text{populasi}$ ) do

$R[k] \leftarrow \text{random}(0-1)$ ;

        If ( $R[k] < pc$ ) then

            Select *Chromosome*[*k*] as parent;

        End;

$k = k + 1$ ;

    End;

End.

Misal ditentukan *crossoverprobability* adalah sebesar 25%, maka diharapkan dalam satu generasi ada 50% *Chromosome* (3 *chromosome*) dari satu generasi mengalami proses *crossover*.

Maka *Chromosome* ke *k* akan dipilih sebagai induk jika  $R[k] < pc$ , dari bilangan acak *R* di atas maka yang dijadikan induk.

Setelah melakukan pemilihan induk proses selanjutnya adalah menentukan posisi *crossover*. Ini dilakukan dengan cara membangkitkan bilangan acak dengan batasan 1 sampai (panjang *chromosome*-1), dalam kasus ini bilangan acak yang dibangkitkan adalah 1 – 3. Misalkan didapatkan posisi *crossover* adalah 1 maka *chromosome* induk akan dipotong mulai gen ke 1 kemudian potongan gen tersebut saling ditukarkan antar induk. Posisi *cut-point crossover* dipilih menggunakan bilangan acak 1-3 sebanyak jumlah *crossover* yang terjadi.

#### f. Mutasi

Jumlah *chromosome* yang mengalami mutasi dalam satu populasi ditentukan oleh parameter *mutation\_rate*. Proses mutasi dilakukan dengan cara mengganti satu gen yang terpilih secara acak dengan suatu nilai baru yang didapat secara acak. Prosesnya adalah sebagai berikut. Pertama dihitung dahulu panjang total gen yang ada dalam satu populasi. Dalam kasus ini panjang total gen adalah ***total\_gen = (jumlah gen dalam chromosome) \* jumlah populasi***.

Untuk memilih posisi gen yang mengalami mutasi dilakukan dengan cara membangkitkan bilangan integer acak antara 1 sampai *total\_gen*. Jika bilangan acak yang dibangkitkan lebih kecil daripada variabel *mutation\_rate* (*pm*) maka pilih posisi tersebut sebagai *sub-chromosome* yang mengalami mutasi. Misal *pm* ditentukan 10% maka diharapkan ada 10% dari *total\_gen* yang mengalami populasi.

Setelah proses mutasi maka telah menyelesaikan satu iterasi dalam algoritma genetika atau disebut dengan satu generasi.



Tabel 2. Tabel hasil satu generasi

<i>Chromosome</i>	<i>X1</i>	<i>X2</i>	<i>X3</i>	<i>X4</i>	Fungsi Objektif	<i>Fitness</i>	Probabilitas	<i>Cumulative Probabilitas</i>
[1]	04	10	05	02	19	0.05	0.0348	0.0348
[2]	15	10	03	01	14	0.0667	0.0464	0.0812
[3]	20	05	05	01	15	0.0625	0.0435	0.1247
[4]	10	05	01	03	5	0.1667	0.1160	0.2407
[5]	14	03	02	02	0	1	0.6960	0.9367
[6]	04	08	04	02	10	0.0909	0.0633	1

Dapat dilihat dari hasil perhitungan fungsi objektif yang telah dilakukan bahwa setelah satu generasi, nilai hasil rata-rata fungsi\_objektif lebih menurun dibandingkan hasil fungsi\_objektif pada saat sebelum mengalami seleksi, *crossover* dan mutasi. Hal ini menunjukkan bahwa *chromosome* atau solusi yang dihasilkan setelah satu generasi lebih baik dibandingkan generasi sebelumnya.

Tabel 3. *Chromosome* baru setelah satu generasi

<i>Chromosome</i>	<i>X1</i>	<i>X2</i>	<i>X3</i>	<i>X4</i>
[1]	10	05	01	03
[2]	14	03	02	02
[3]	04	10	05	02
[4]	10	05	01	03
[5]	14	03	02	02
[6]	14	03	02	02

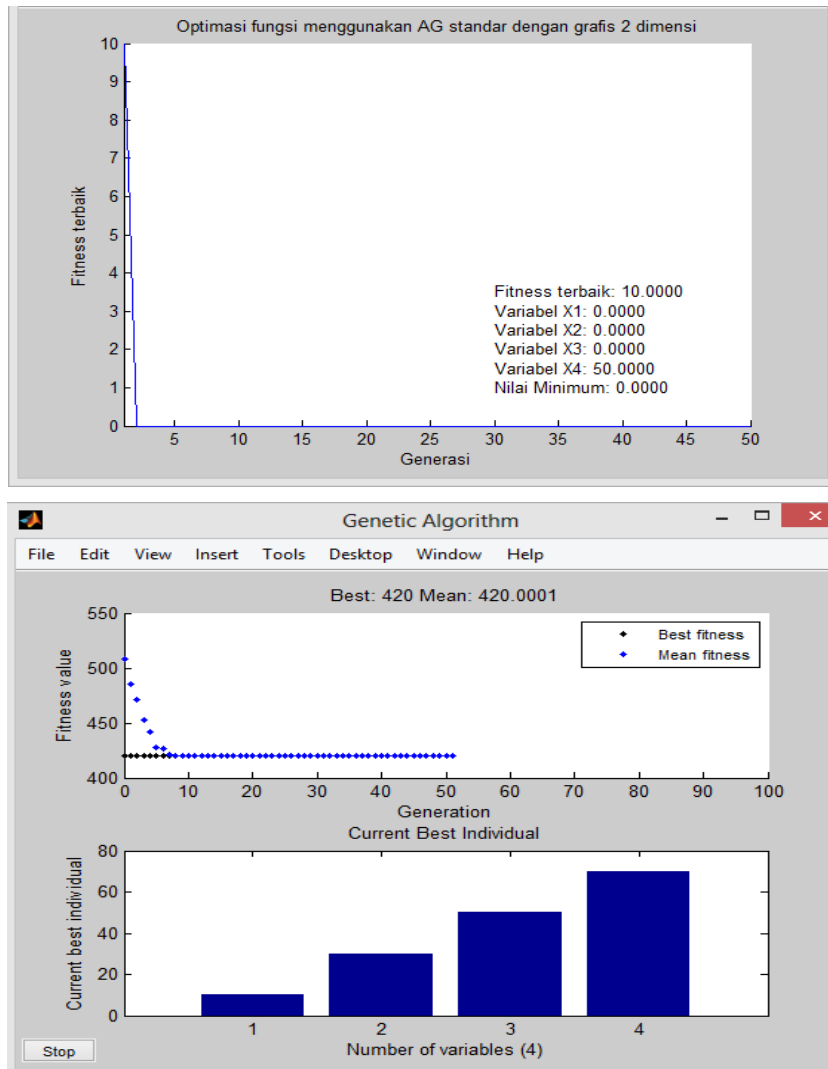
*Chromosome-chromosome* yang baru ini akan mengalami proses yang sama seperti generasi sebelumnya yaitu proses evaluasi, seleksi, *crossover* dan mutasi yang kemudian akan menghasilkan *chromosome-chromosome* baru untuk generasi yang selanjutnya. Proses ini akan berulang sampai sejumlah generasi yang telah ditetapkan sebelumnya.

Setelah 50 generasi untuk contoh kombinasi maksimal didapatkan *chromosome* yang terbaik untuk nilai  $N = 50$ , adalah:

*Chromosome* = [00;00;00;50]

Jika didekode maka :  $X1=0$  ;  $X2=0$  ;  $X3=0$  ;  $X4=50$

Jika dihitung terhadap persamaan  $f = 10X_1 + 5X_2 + 2X_3 + X_4$  :  
 $(10*0) + (5*0) + (2*0) + (1*50) = 50$



Gambar 2. Hasil Algoritma Genetika setelah 50 generasi untuk Nilai Maksimal  
 g. Implementasi Sistem



Gambar 3. Aplikasi Penarikan Uang Tunai pada ATM

Gambar 3. Merupakan menu utama dari simulasi aplikasi penarikan uang tunai pada ATM, dimana nasabah mengetikkan jumlah uang yang ingin diambil dan terdapat dua tombol untuk pemilihan yaitu pemilihan pecahan maksimal atau pemilihan pecahan minimal.

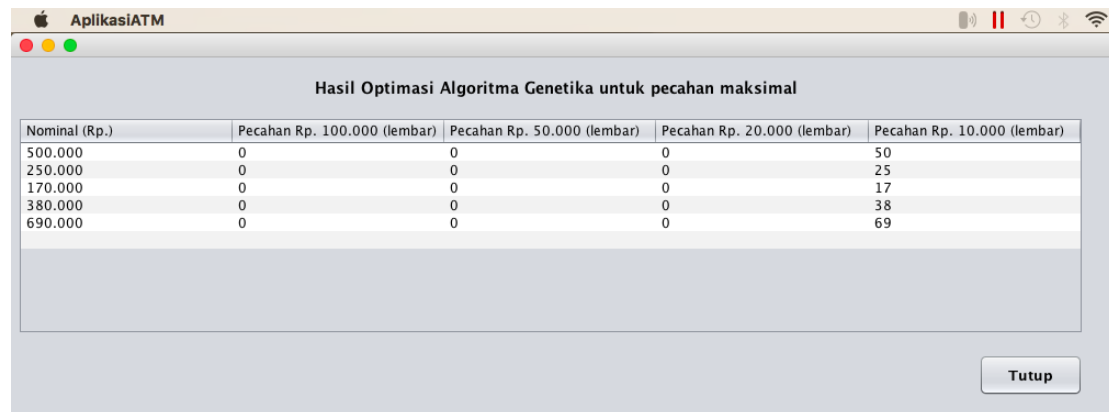
Gambar 4. Menu Pecahan Maksimal

Gambar 5. Menu Pecahan Minimal

Gambar 4 dan 5 menampilkan berapa jumlah pecahan uang yang didapat sesuai dengan nominal yang dimasukan dan kombinasi pecahan yang dipilih.

Nominal (Rp.)	Pecahan Rp. 100.000 (lembar)	Pecahan Rp. 50.000 (lembar)	Pecahan Rp. 20.000 (lembar)	Pecahan Rp. 10.000 (lembar)
500.000	5	0	0	0
250.000	2	1	0	0
170.000	1	1	1	0
380.000	3	1	1	1
690.000	6	1	2	0

Gambar 6. Hasil Optimasi Algoritma Genetika untuk Pecahan Minimal



Nominal (Rp.)	Pecahan Rp. 100.000 (lembar)	Pecahan Rp. 50.000 (lembar)	Pecahan Rp. 20.000 (lembar)	Pecahan Rp. 10.000 (lembar)
500.000	0	0	0	50
250.000	0	0	0	25
170.000	0	0	0	17
380.000	0	0	0	38
690.000	0	0	0	69

Gambar 7. Hasil Optimasi Algoritma Genetika untuk Pecahan Maksimal

## 5. Kesimpulan

Dari penelitian dapat diambil beberapa kesimpulan sebagai berikut:

- Dalam simulasi sistem aplikasi penarikan uang tunai dengan menggunakan Algoritma Genetikaini, didapatkan kombinasi *chromosome* yang maksimal dan minimal sesuai dengan jumlah nominal yang di-input.
- Dari langkah-langkah Algoritma Genetika yang sudah dijalankan di atas, setelah beberapa generasi bisa didapatkan jumlah kombinasi maksimal dengan jumlah koin terbanyak dan bisa juga menghasilkan jumlah kombinasi minimal dengan jumlah koin yang paling sedikit.
- Jumlah kombinasi maksimal dari contoh nominal Rp. 500.000,- adalah 50 lembar uang pecahan Rp. 10.000,-. Sedangkan untuk kombinasi minimal yang didapat adalah 5 lembar uang pecahan Rp. 100.000,-.
- Dalam penelitian ini dibuktikan bahwa Algoritma Genetika dapat menyelesaikan permasalahan Coin Problem yang terdapat pada Aplikasi Tarik Tunai di mesin ATM.

## DAFTAR PUSTAKA

- [1] Kasmir, SE.MM., Bank dan Lembaga Keuangan Lainnya, Rajawali Pers. 2012
- [2] Kusnandar, Viva, "Transaksi Via Kartu ATM", Bloomberg Businessweek Indonesia edisi 23, 2014.
- [3] Zukhri, Zainudin, "Algoritma Genetika Metode Komputasi Evolusioner untuk Menyelesaikan Masalah Optimas", CV. Andi Offset, 2014.
- [4] Suyanto, "Algoritma Genetika dalam Matlab", Penerbit Andi Yogyakarta, 2005.
- [5] Hermawanto, Denny, <http://dennyhermawanto.webhop.org>, akses terakhir: 6 Mei 2014
- [6] <http://www.infobanknews.com/2013/08/empat-bank-besar-dominasi-mesin-atm-di-tanah-air/>, akses terakhir: 6 Mei 2014

- [7] Anindhika, Dita, “Kompleksitas Algoritma untuk Penyelesaian Persoalan Penukaran Koin dengan Algoritma Greedy”, Makalah IF2091 Struktur Diskrit – Sem. I Tahun 2011/2012.
- [8] Widodo, Agus Wahyu dan Wayan Firdaus Mahmudy, “Penerapan Algoritma Genetika pada Sistem Rekomendasi Wisata Kuliner”, Jurnal Ilmiah Kursor Vol. 5, No. 4, Juli 2010.
- [9] Garg, Richa and Saurabh Mittal, “Optimization by Genetic Algorithm”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 4, April 2014 , ISSN: 2277 128X.