COMPUTATIONAL INTELLIGENT

ALGORITMA GENETIKA

UNTUK PENYELESAIAN MASALAH KNAPSACK (1-0)

Oleh :

Cathrine Nicea Folamauk

Magister Ilmu Komputer dan Elektronika

Fakultas Matematika dan Ilmu Pengetahuan Alam

Universitas Gajah Mada

2016

ALGORITMA GENETIKA

UNTUK PENYELESAIAN MASALAH KNAPSACK (1-0)

1. LATAR BELAKANG

Knapsack problem merupakan salah satu masalah yang dapat ditemukan dalam kehidupan masyarakat saat ini, dimana seseorang dihadapkan pada persoalan optimasi pemilihan barang-barang yang akan dimasukan ke dalam wadah yang memiliki batasan dengan kapasitas daya tampung tertentu, dengan harapan barang-barang yang dimasukan dalam wadah memiliki bobot harga yang tinggi. Adapun benda-benda yang dimasukkan masing-masing memiliki bobot berat dan nilai, yang dapat digunakan untuk menentukan prioritas dalam proses pemilihan nantinya. Sedangkan wadah yang digunakan memiliki nilai konstanta yang merupakan jumlah bobot maksimum dari barang-barang yang dapat dimasukkan. Untuk itu diperlukan suatu cara dalam menentukan barang-barang apasaja yang dapat dimasukkan ke dalam wadah yang jumlah bobotnya tidak melebih kapasitas dari daya tampung namun memiliki hasil yang optimum dengan jumlah bobot harga yang tinggi.

2. PENGERTIAN KNAPSACK PROBLEM

Knapsack problem atau rucksack problem merupakan masalah optimasi dalam kombinatorial. Sesuai dengan namanya Knapsack problem yang berarti sebuah masalah dalam memaksimalkan pemilihan sejumlah barang-barang yang akan dibawa dalam sebuah tas pada suatu perjalanan dengan tepat.

2.1. Jenis-jenis Knapsack Problem :

Terdapat beberapa jenis knapsack problem yaitu :

a) *0/1 Knapsack Problem*. Setiap barang hanya terdiri dari 1 unit, " take it or leave it"
b) *Fractional Knapsack Problem*. Barang-barang yang dibawa hanya beberapa bagian yang dinyatakan dalam pecahan. Versi problem ini menjadi masuk akal apabila barang yang tersedia dapat dibagi-bagi misalnya gula, tepung, dan sebagainya.

c) *Bounded Knapsack Problem*. Setiap barang yang disediakan sebanyak N unit namun memiliki persedian dengan jumlah yang terbatas.

d) *Unbounded Knapsack Problem* Setiap barang tersedia lebih dari 1 unit dan jumlahnya tidak terbatas

3. ALGORITMA GENETIKA

Algoritma Genetika sebagai cabang dari Algoritma Evolusi merupakan metode *adaptive* yang biasa digunakan untuk memecahkan suatu pencarian nilai dalam sebuah masalah optimasi. Algoritma ini didasarkan pada proses genetic yang ada dalam makhluk hidup; yaitu perkembangan generasi dalam sebuah populasi yang alami, secara lambat laun mengikuti prinsip seleksi alam atau "siapa yang kuat, dia yang akan bertahan (*survive)".* Dengan meniru teori evolusi ini, Algoritma Genetika dapat digunakan untuk mencari solusi dari permasalahan-permasalah dalam dunia nyata. Hal-hal yang harus dilakukan dalam algoritma genetika :

1) Mendefinisikan Individu, dimana individu menyatakan salah satu solusi (penyelesaian) yang mungkin dari permasalah yang diangkat
2) Mendefinisikan nilai fitness, yang merupakan ukuran baik-tidaknya sebuah individu atau baik-tidaknya solusi yang didapatkan
3) Menentukan proses pembangkitan populasi awal. Hal ini biasanya dilakukan dengan menggunakan pembangkitan acak seperti random-walk
4) Menentukan proses seleksi yang akan digunakan
5) Menentukan proses perkawinan silang (cross-over) dan mutase gen yang akan digunakan.

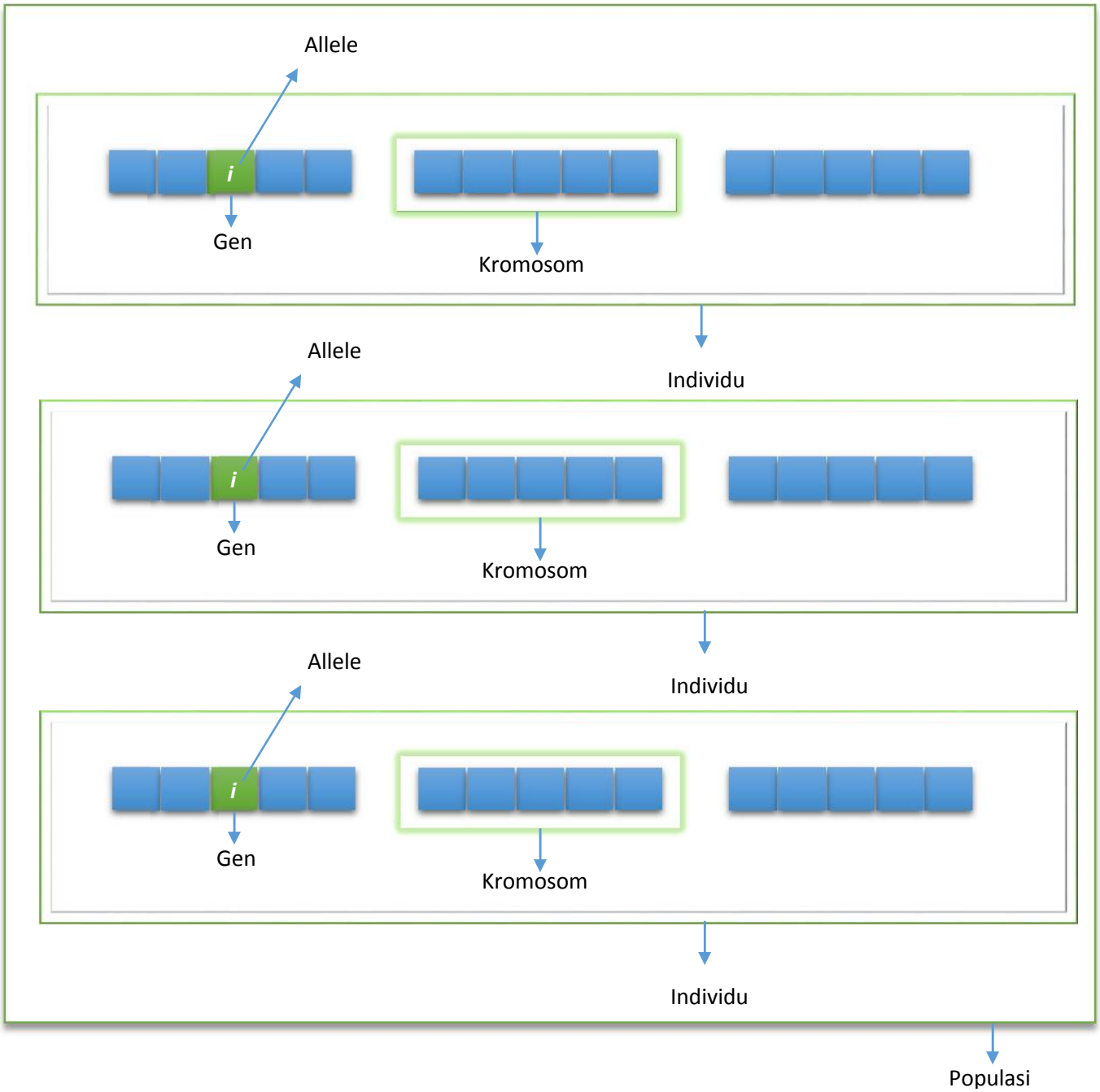
Beberapa bagian penting dalam algoritma genetika :

1) **Genotype (Gen),** sebuah nilai yang menyatakan satuan dasar yag membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Dalam algoritma genetika, gen dapat berupa nilai biner, float, integer maupun karakter atau kombinatorial.
2) **Allele**, nliai dari gen
3) **Kromosom**, gabungan gen-gen yang membentuk suatu nilai tertentu
4) **Individu**, menyatakan satu nilai atau keadaan yang menyatakan salah satu solusi yang

mungkind ari permasalahan yang diangkat

5) **Populasi**, merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi

6) **Generasi**, menyatakan satu siklus proses evolusi atau satu iterasi di dalam algoritma genetika.

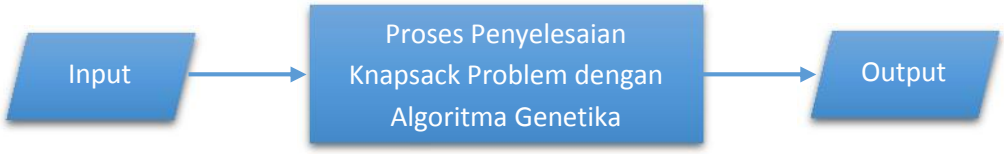


Gambar 1. Ilustrasi Perbedaan Istilah-Istilah dalam Algoritma Genetik

4. IMPLEMENTASI KNAPSACK PADA ALGORITMA GENETIKA

4.1. Perancangan Sistem

Cara kerja program secara garis besar adalah mencari nilai barang yang paling maksimum dengan berat barang tidak melebihi kapasitas yang tersedia. Pencariannya menggunakan algoritma genetika. Berikut adalah gambar blok diagram sistem:



Gambar 2. Diagram Sistem Knapsack Problem

4.2. Perancangan Program

Pada program aplikasi "Genetic Algorithm-Knapsack Problem [1,0] pada jumlah populasi, generasi, batasan kapasitas, persentase crossover dan mutasi, ditentukan oleh user ketika menjalankan program. Adapun banyaknya data *n* yang digunakan (data barang yang akan ditampung) ditentukan berdasarkan banyanknya data yang diinputkan oleh user kedalam program.

4.3. Representasi Kromosom

Dalam kasus Knapsack Problem, Gen direpresentasikan dalam bentuk string bit. Caranya yaitu dengan memilih barang secara manual pada, di mana barang yang terpilih diberi 1 dan untuk barang yang tidak dipilih diberikan angka 0

Contoh :

Tabel 1. Data Genotype

Kode	Nama Barang	Berat (kg)	Nilai (Rp)	Pilihan I	Pilihan II
b1	A	8	100	0	1
b2	B	4	50	1	1
b3	C	10	75	0	0
b4	D	2	45	1	1

4.4. Prosedur Inisialisasi

Proses inisialisasi merupakan proses pemberian bobot pada populasi awal untuk setiap kromosom sesuai dengan jumlah gen yang dimiliki secara acak (*random*) yang di presentasikan dalam bentuk *string bit* (1 dan 0). Jumlah gen yang digunakan berdasarkan jumlah dari jenis barang yang akan dibawa. Seperti pada tabel 1, jumlah barang yang akan dibawa sebanyak 4 jenis dengan masing-masing bobot dan harganya, sehingga didapatkan:

K1	**0**	**1**	**0**	**1**
	b1	b2	b3	b4

K2	**1**	**1**	**0**	**1**
	b1	b2	b3	b4

Untuk populasi berikutnya, nilai masing-masing kromosom diperoleh dari hasil crossover dan mutasi.

4.5. Fungsi Fitness

Fungsi Fitnes pada masalah ini diambil berdasarkan tujuan dan batasan yang diberikan, yaitu menentukan total nilai maksimum barang-barang yang jumlah total beratnya tidak melebihi kapasitas maksimum daya tampung yang diberikan, sehingga dapat dituliskan persamaannya:

$$f(v) = \sum_{i=1}^{n} b_i v_i \qquad ,u \qquad f(w) = \sum_{i=1}^{n} b_i w_i \leq W$$

Sedangkan untuk total nilai barang-barang yang melebihi kapasitas maksimum daya tampung, untuk nilai fitness dihitung menggunakan persamaan :

$$f = \sum_{i=1}^{n} f(w)_i - w_i \qquad ,u \qquad f = \sum_{i=1}^{n} b_i w_i > W$$

Keterangan :

b_i = Nilai Gen Kromosom ke – *i*

v_i = Nilai Harga Barang ke – *i*

w_i = Nilai Berat Barang ke – *i*

W = Nilai Kapasitas Maksimum Daya Tampung

Contoh :

Misalkan kapasitas maksimum untuk daya tampung yang diberikan adalah $\leq \mathbf{1}$ dengan jumlah kromosom pada populasi awal sebanyak 5 kromosom untuk genenasi ke-0, maka diperoleh :

Tabel 2. Jumlah Berat Benda Pada Masing-Masing Kromosom

Kromosom (K)	A	B	C	D	Total (A+B+C+D)
1	1 x 8	0 x 4	1 x 10	0 x 2	18
2	1 x 8	1 x 4	0 x 10	1 x 2	14
3	0 x 8	1 x 4	1 x 10	0 x 2	14
4	0 x 8	1 x 4	1 x 10	1 x 2	16
5	1 x 8	0 x 4	1 x 10	1 x 2	20

Tabel 3. Jumlah Harga Benda Pada Masing-Masing Kromosom

Kromosom (K)	A	B	C	D	Total (A+B+C+D)
1	1 x 100	0 x 50	1 x 75	0 x 45	175
2	1 x 100	1 x 50	0 x 75	1 x 45	195
3	0 x 100	1 x 50	1 x 75	0 x 45	125
4	0 x 100	1 x 50	1 x 75	1 x 45	160
5	1 x 100	0 x 50	1 x 75	1 x 45	225

Tabel 4. Nilai Fitness Pada Masing-Masing Kromosom

Kromosom	Total Berat	Total Harga	Fitness	Ket.
1010	18	175	**-4**	**Invalid**
1101	14	195	**195**	**Valid**
0110	14	125	**125**	**Valid**
0111	16	160	**-2**	**Invalid**
1011	20	225	**-6**	**Invalid**

Pada tabel 4, terdapat 2 kromosom yang ***valid*** yaitu kromosom yang bernilai positif (memiliki total berat $\leq$ kapasitas maksimum daya tampung 14), dan 3 kromosom yang ***invalid*** yaitu kromosom yang bernilai negative (memiliki total berat > kapasitas maksimum daya tampung)

4.6. Proses Seleksi

Untuk menyelesaikan masalah knapsack problem, menggunakan metode "Roulette Wheel" untuk melakukan seleksi terhadap parent yang akan digunakan pada proses crossover dan mutase untuk menghasilkan individu baru. Langkah-langkah penyeleksiannya:

1. Menghitung fungsi fitness untuk masing-masing kromosom $k = 1,2,3, \dots, p\_s$ (ukuran populasi)
2. Menghitung total fitness dari populasi tersebut:

$$F = \sum_{i=1}^{p} f(v)_k$$

3. Menghitung nilai probabilitas dari seleksi p_k untuk masing-masing kromosom $k = 1,2,3, \dots, p\_s$, dimana :

$$p_k = f(v)_k / F$$

4. Menghitung total komulatif p, q_i untuk masing-masing kromosom $v_i = 1, 2, 3, \dots, p\_s$, dimana :

$$q_i = \sum_{j=1}^{i} p_j$$

Proses seleksi dilakukan berdasarkan atas pemutaran *Roulette Wheel* sebanyak pop_size yang ditentukan. Pada proses dilakukan pemilihan kromosom tunggal sebagai populasi baru dengan cara membangkitkan bilangan acak r. Jika pada proses pengacakan diperoleh $r \leq q_i$, maka akan dipilih kromosom pertama v_1, sebaliknya akan dipilih kromosom ke$-i$ untuk v_i $(2 \leq i \leq p\_s)$, sedemikian rupa sehingga diperoleh $q_i - 1 < r \leq q_i$.

Pada contoh diperoleh :

Tabel 5. Data Nilai Fitness, Probabilitas dan Total Komulatif Pada Kromosom

Kromosom (K)	Fitness $(f(v))$	Probabilitas (p) $(f(v))/F$	Total Komulatif (q)
1	-4	-0.01299	-0.01299
2	195	0.63312	0.62013
3	125	0.40584	1.025974
4	-2	-0.00649	1.019481
5	-6	-0.01948	1
	$F = 308$		

Pada proses akhir dari seleksi dilakukan pembangkitan bilangan acak sebanyak jumlah *pop_size* untuk menentukan urutan kromosom-kromosom yang dijadikan sebagai *parent*.

Misalnya :

K_b1	K_b2	K_b3	K_b4	K_b5
0.3	0.89	0.59	0.44	0.99

Kromosom Terpilih

K2	K2	K3	K2	K3
1101	0110	1101	1101	0110

Gambar 3. Proses Pengurutan Kromosom Terpilih Secara Acak (Random)

4.7. Operator Genetika

Metode *crossover* yang digunakan pada kromosom berbentuk *string biner* merupakan *crossover* satu titik (*one-point crossover*), dan mutase yang digunakan merupakan mutasi yang bernilai biner.

4.7.1. Proses Crossover :

Pemilihan titik yang akan digunakan pada proses pembagian sebelum melakukan crossover

dilakukan secara acak (*random*).

Misalnya : titik terpilih adalah **2** , diperoleh :

1101 X ➔ 1110 ← Kromosom Baru (K_1)

0110 X ➔ 0101 ← Kromosom Baru (K_2)

Gambar 4. Proses Crossover

4.7.2. Proses Mutasi

Pada tahap ini, dilakukan proses pengubahan terhadap nilai gen terpilih. Proses mutasi dilakukan menggunakan mutase string biner dimana untuk gen bernilai “1” diubah menjadi “0”, sebaliknya untuk gen bernilai “0” diubah menjadi “1”. Penentuan letak gen pada kromosom dan parent yang akan diubah dipilih secara acak (*random*). Banyaknya jumlah mutasi ditentukan berdasarkan persentase (%) dari mutasi yang diberikan terhadap jumlah kromosom parent pada masing-masing populasi.

Contoh :

Jumlah Gen per kromosom = 4

Jumlah Kromosom = 5

Persentase (%) = 20

Jumlah Mutasi = $5\ x\ 20\% = 1$

Random Parent = 2

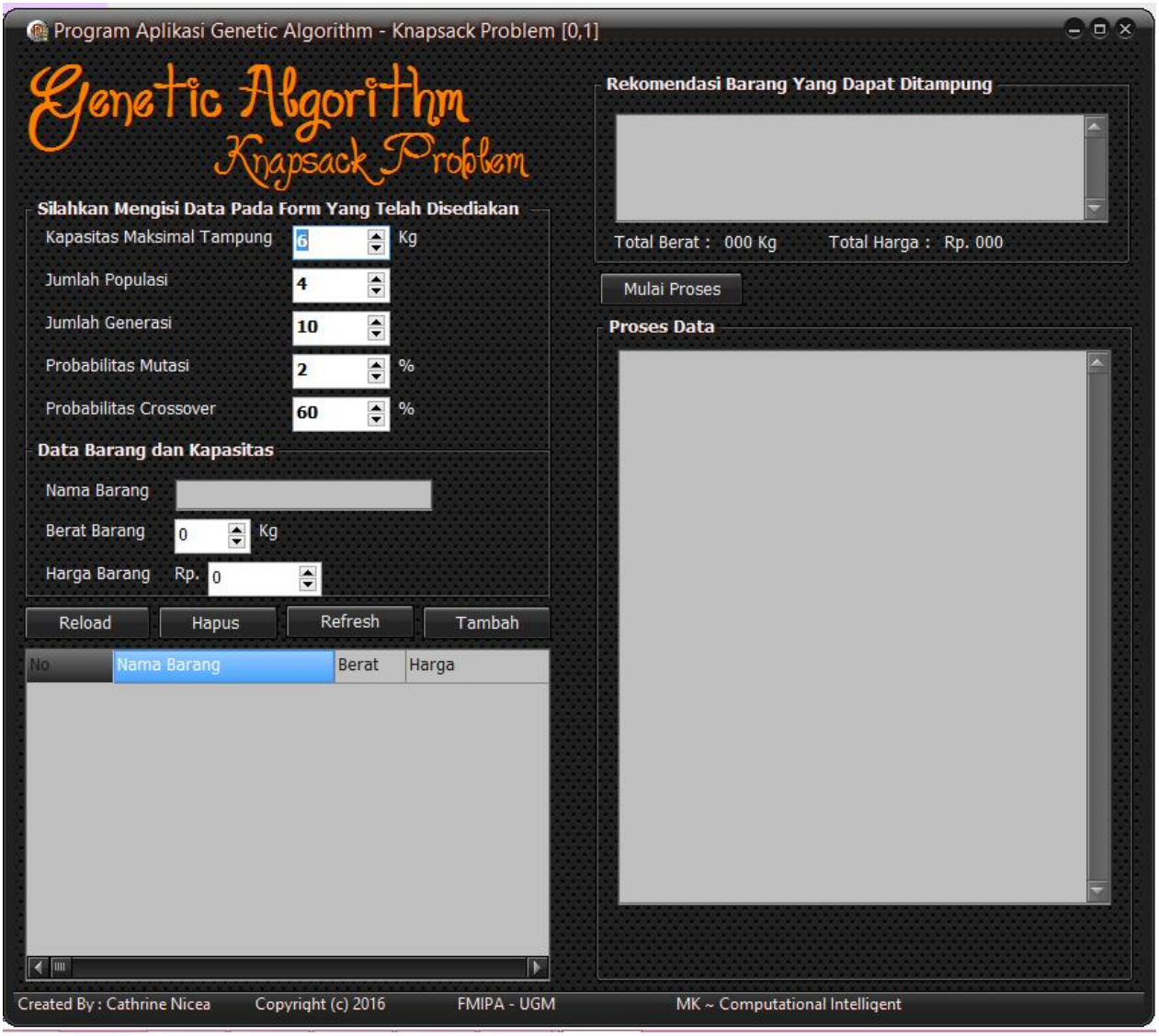
Random Gen =3

Maka diperoleh :

K_b2			
1	2	3	4
0	1	0	1
Hasil Mutasi			
0	1	1	1

Gambar 5. Proses Mutasi

5. PROGRAM APLIKASI ALGORITMA GENETIK - KNAPSACK PROBLEM [0,1]



Gambar 6. Tampilan Awal Program Aplikasi Algoritma Genetik - Knapsack Problem

Gambar 6. Merupakan tampilan awal dari program aplikasi “Genetic Algorithm – Knapsack Problem”. Pada program terdiri dari 5 tombol utama dan 8 form untuk mengisi data input sebelum melakukan proses dan 3 memo yang digunakan untuk menampilkan hasil dan proses selama program berlangsung secara detail. Adapun masing-masing komponen dan fungsi detailnya:

1. **Tombol ;** terdiri dari 5 tombol utama yaitu Reload, Hapus, Refresh, Tambah dan Mulai Proses.
   a. **Reload ;** digunakan untuk memasukan data dari file *.txt (DataBarang.txt) ke dalam tabel

program untuk dieksekusi

Note : Data pada file DataBarang.txt dapat ditambahkan secara manual.

b. **Hapus ;** digunakan untuk membersihkan data form secara keseluruhan apabila ingin melakukan pembatalan data input atau setelah menyelesaikan program
c. **Refresh** ; digunakan untuk merefresh data yang ada menggunakan data default yang ada dalam program
d. **Tambah ;** digunakan untuk menambahkan data dalam tabel data secara manual. Tombol ini digunakan ketika user sudah mengisi form Nama Barang, Berat dan Harga.
e. **Mulai Proses ;** digunakan untuk memulai proses Algoritma Genetik untuk menyelesaikan Knapsack Problem sesuai dengan inputan data barang, batasan kapasitas tampung, persentasi mutasi dan *crossover*, banyaknya populasi dan generasi yang diinginkan.

2. **Form ;** terdiri dari 8 form penting yang harus diisi oleh user sebelum menekan tombol proses. Pada form sendiri dibagi dalam 2 kategori yaitu :
   a. **Penginputan data barang,** terdiri dari 3 form utama yang harus di input oleh user jika ingin menambahkan data barang baru pada tabel. Ketiga form dimaksud yaitu : Nama Barang, Berat Barang dan Harga Barang. Apabila salah satu dari ke-3 form tidak dilengkapi maka data barang tersebut tidak dapat disimpan dalam tabel. Untuk menyimpan data inputan dari user, maka selanjutnya user menekan tombol “Tambah”
   b. **Penginputan Data Algoritma Genetik,** terdiri dari 5 form utama yang wajib diisi oleh user sebelum menjalankan program dengan menekan tombol “Mulai Proses”, yaitu :
      - **Kapasitas Maksimal Tampung.** Digunakan oleh user untuk menginput bobot maksimal daya tampung yang dijadikan sebagai batasan dalam memasukan barang dalam kantong
      - **Jumlah Populasi.** User menentukan jumlah populasi yang akan dibentuk per generasinya
      - **Jumlah Generasi.** User menentukan jumlah generasi yang akan digunakan. Jumlah generasi dijadikan sebagai syarat berhenti pada program ini
      - **Probabilitas Mutasi.** User menentukan berapa persentase (%) mutasi yang dilakukan

dalam proses mutasi untuk menghasilkan individu baru

- **Probabilitas Crossover.** User menentukan berapa persentase (%) crossover yang akan dilakukan.

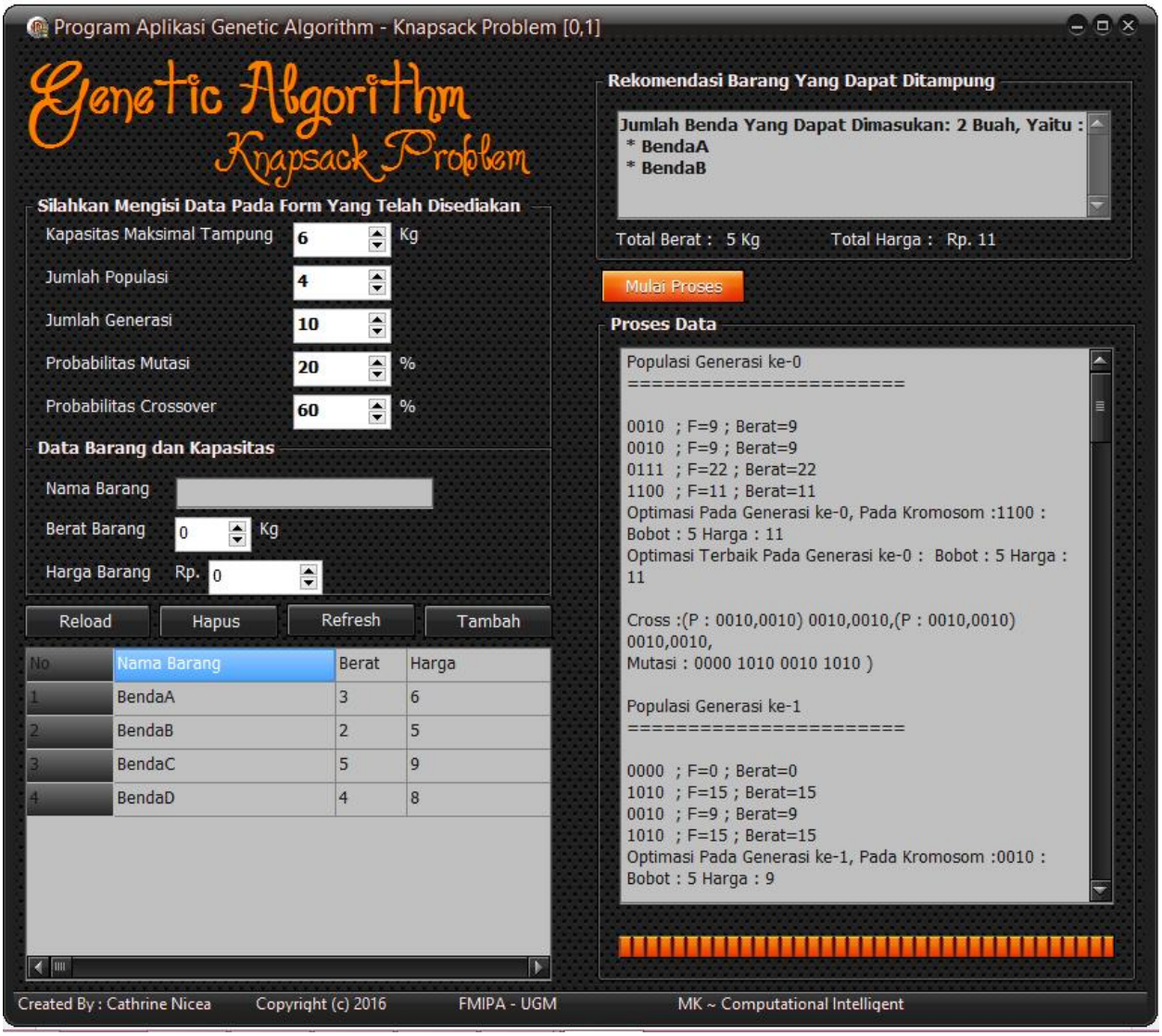
3. **Memo ;** terbagi dalam 2 bagian utama dengan masing-masing fungsinya. Yaitu :
   a. **Memo Hasil,** digunakan untuk manampilkan data barang-barang terpilih yang dapat dimasukan dalam media penampung. Memo ini terletak pada bagian kanan atas pada program.
   b. **Memo Proses,** digunakan untuk menampilkan proses algoritma genetik dalam menentukan solusi knapsack problem. Adapun informasi-informasi yang termuat didalamnya yaitu : informasi Kromosom, Bobot masing-masing Kromosom, Nilai Fitness Kromosom, Kromosom Hasil *Crossover*, Kromosom Hasil Mutasi, Optimasi Kromosom per Generasi dan Optimasi Kromosom Terbaik untuk keseluruhan Generasi.

5.1. Cara Penggunaan

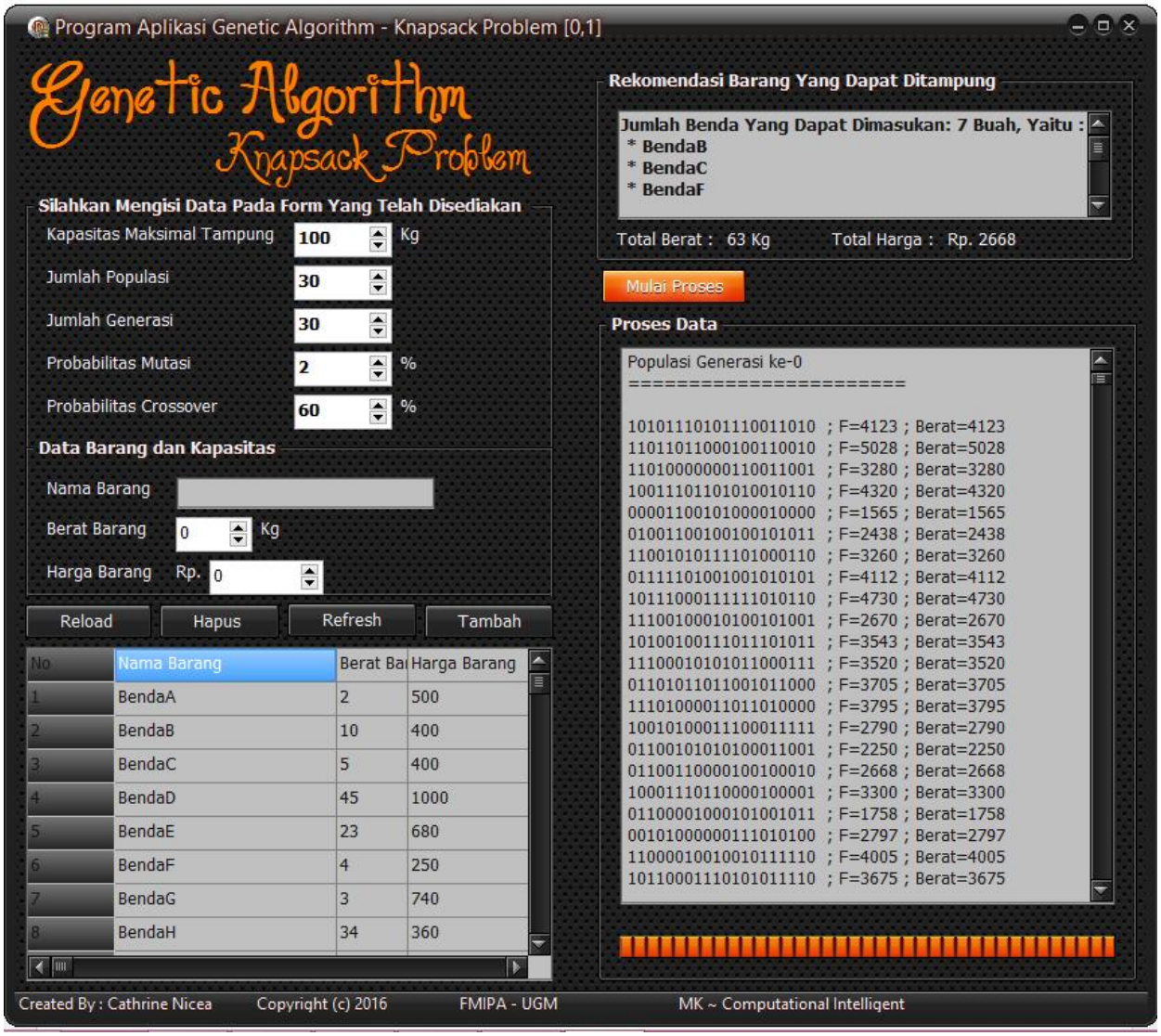
Langkah pertama yang harus dilakukan oleh user untuk menjalankan program aplikasi *"Genetic Algorithm – Knapsack Problem"* adalah dengan membuka file **"KnapsackProblem.exe".** Setelah muncul tampilan awal program, user dapat menginput data secara manual dengan memasukan data pada form barang dan menekan tombol "Tambah", atau menginput data barang secara otomatis menggunakan tombol "Reload" atau *"Refresh*". Sebelum memulai proses user harus menginput jumlah Maksimum kapasitas daya tampung, jumlah populasi, jumlah generasi, presentasi mutasi dan presentasi *crossover* yang akan dilakukan. Setelah semuanya sudah lengkap, maka user dapat menjalankan program dengan menekan tombol "Mulai Proses". User selanjutnya menunggu hingga proses selesai maka pada program akan ditampilkan hasil barang-barang yang terpilih dan proses algoritma genetiknya. Lamanya proses bergantung pada banyaknya data inputan, jumlah populasi dan generasi yang digunakan, semakin banyak populasi, generasi ataupun data yang digunakan, maka untuk memproses hasil akan membutuhkan waktu yang lebih lama.

5.2. Output :

Hasil dari program aplikasi ini tidak bersifat tetap, dikarenakan proses random yang digunakan membuat hasil dari program ini sulit ditebak. Namun tentunya pada program akan memberikan solusi yang diinginkan. Apabila jumlah kapasitas yang diinputkan terlalu kecil dan dengan jumlah generasi yang digunakan juga sedikit maka kemungkinan solusi tidak ditemukan.



Gambar 7. Running Program dengan 4 data barang.



Gambar 7. Running Program dengan 20 data barang.