PENGARUH VARIASI PELUANG CROSSOVER DAN MUTASI DALAM ALGORITMA GENETIKA UNTUK MENYELESAIKAN MASALAH KNAPSACK

Sutikno

Program Studi Teknik Informatika
Fakultas Sains dan Matematika UNDIP
tik@undip.ac.id

Abstrak

Algoritma Genetika telah banyak digunakan untuk menyelesaikan masalah optimasi, salah satunya yaitu masalah *knapsack*. Masalah *knapsack* merupakan masalah optimasi yang berusaha memaksimalkan keuntungan. Untuk menghasilkan optimasi yang terbaik dipengaruhi beberapa variabel diantaranya yaitu jumlah kromosom, nilai peluang *crossover* dan nilai peluang mutasi, sehingga perlu dilakukan penelitian dengan cara memberikan beberapa variasi peluang *crossover* dan mutasi untuk mendapatkan nilai *fitness* terbaik sehingga penelitian-penelitian selanjutnya yang berhubungan dengan penerapan algortima generika dalam menyelesaikan masalah *knapsack* tidak lagi memberikan nilai peluang *crossover* dan mutasi dengan cara *trial and error*. Pada penelitian ini dibuat aplikasi penerapan algoritma genetika dalam menyelesaikan masalah *knapsack* dengan menggunakan Visual Basic dan Microsoft Access, dan dilakukan pengujian untuk mencari nilai *fitness* terbaik pada pemberian beberapa variasi peluang *crossover* dan peluang mutasi. Hasil pengujian didapatkan bahwa nilai *fitness* terbaik (terbesar) pada pemberian nilai peluang mutasi rata-rata 0,1 dan nilai peluang *crossover* rata-rata 0,3.

Kata Kunci : Algoritma Genetika, Masalah *Knapsack*, Peluang *Crossover*, Peluang Mutasi

1. Latar Belakang

Algoritma genetika merupakan metode adaptif berdasarkan pada evolusi alami yang digunakan untuk menyelesaikan masalah optimasi. Algoritma ini memproses populasi dari pencarian ruang solusi dengan tiga operasi yaitu seleksi, kawin silang dan mutasi [3].

Salah satu masalah yang dapat di selesaikan dengan algoritma genetika adalah *Knapsack Problem*. Untuk menghasilkan optimasi yang terbaik dipengaruhi beberapa variabel diantaranya yaitu jumlah kromosom, peluang *crossover* dan peluang mutasi, sehingga perlu dilakukan penelitian dengan cara memberikan beberapa variasi peluang *crossover* dan mutasi untuk mendapatkan nilai *fitness* terbaik sehingga penelitian-penelitian selanjutnya yang berhubungan dengan penerapan algoritma generika dalam menyelesaikan masalah *knapsack* tidak lagi memberikan nilai peluang *crossover* dan mutasi dengan cara *trial and error*.

Pada penelitian ini akan dilakukan pembuatan aplikasi algoritma genetika untuk menyelesaikan *knapsack problem* dan dilakukan pengujian beberapa variasi peluang *crossover* dan peluang mutasi, kemudian dicari nilai peluang *crossover* dan mutasi yang terbaik dalam menyelesaikan permasalahan *knapsack*.

2. Knapsack Problem

Knapsack problem merupakan masalah dimana orang dihadapkan pada persoalan optimasi pada pemilihan benda yang dapat dimasukkan ke dalam sebuah wadah yang memiliki keterbatasan ruang atau daya tampung. Dengan adanya optimasi dalam pemilihan benda yang akan dimasukkan kedalam wadah tersebut diharapkan dapat menghasilkan keuntungan yang maksimum [1].

Benda-benda yang akan dimasukkan ini masing-masing memiliki berat dan nilai yang digunakan untuk menentukan prioritasnya dalam pemilihan tersebut. Nilainya dapat berupa tingkat kepentingan, harga barang, nilai sejarah, atau yang lainnya. Wadah yang dimasukkan disini juga memiliki nilai konstanta yang merupakan nilai pembatas untuk benda-benda

yang akan dimasukkan ke dalam wadah tersebut sehingga harus diambil sebuah cara memasukkan benda-benda tersebut kedalam wadah sehingga menghasilkan hasil optimum tetapi tidak melebihi kemampuan wadah untuk menampungnya. Secara matematis, nilai dan berat total dapat dituliskan seperti pada persamaan 1 dan 2.

$$F = \sum_{i=1}^{n} b_i \, v_i \quad (1)$$

$$Wtot = \sum_{i=1}^{n} b_i \, w_i \quad (2)$$

Dimana:

F = nilai *finess* setiap kromosom
b = bit bernilai 1 atau 0 (nilai 1 jika barang dibawa, dan 0 jika tidak dibawa)
v = nilai dari setiap barang
n = jumlah semua barang
Wtot = Berat Total yang terkandung pada setiap kromosom
W = berat yang terkandung pada setiap barang

3. Algoritma Genetika

Algoritma genetika adalah pencarian terkomputerisasi dan algoritma optimasi berdasarkan mekanika alami genetika dan seleksi alam. Algoritma dimulai dengan sebuah paket solusi yang memungkinkan. Satuan solusi yang mungkin disebut dengan populasi. Setiap solusi yang mungkin dalam populasi juga disebut dengan kromosom. Setiap kromosom ditugaskan dengan nilai *finesss* berdasarkan fungsi finesss. Solusi dari satu populasi diambil dan biasa membangun populasi baru. Ini adalah motivasi dengan harapan bahwa populasi baru akan menjadi lebih baik daripada yang sebelumnya [2].

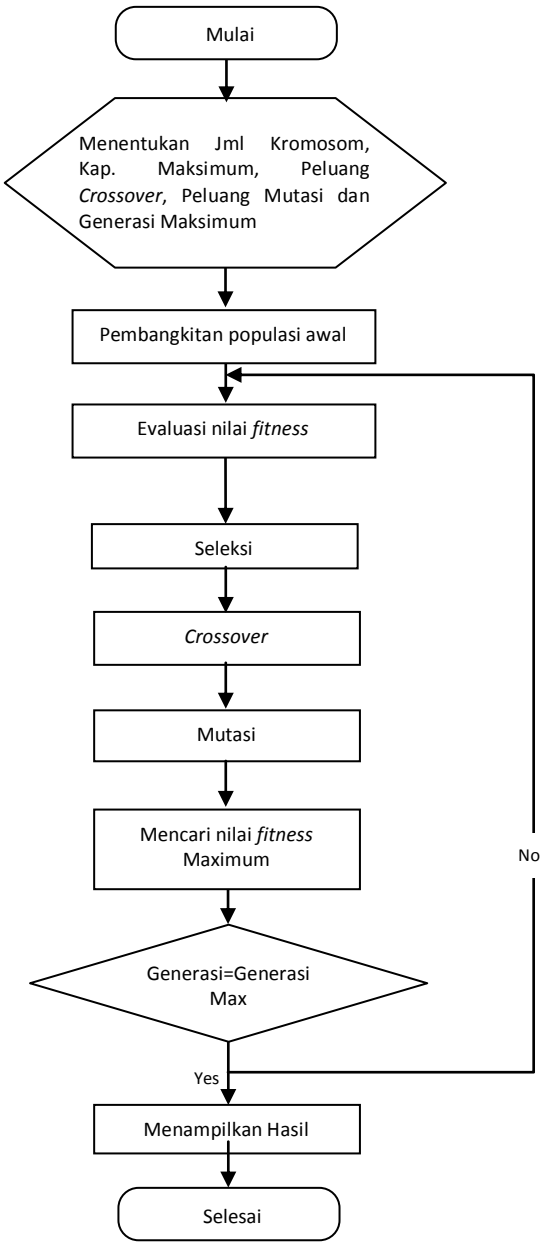
Solusi merupakan hal yang dipilih untuk membangun solusi baru (keturunan), dipilih menurut nilai finesssnya, banyak kesempatan mereka untuk dapat melakukan reproduksi. Keturunan menggantikan populasi kaum tua dan generasi tercapai. Proses ini diulang sampai kriteria tertentu ditemukan.

Secara umum garis besar dari algoritma genetika dasar yaitu sebagai berikut [2]:

a. [Awal] membangkitkan populasi acak n kromosom (solusi yang cocok untuk masalah)
b. [Finesss] mengevaluasi nilai finesss f(x dari tiap kromosom x di populasi)
c. [Populasi Baru] membuat populasi baru dengan mengulang langkah-langkah berikut sampai populasi baru tercapai.
   i. [Pilihan] pilih dua kromosom orang tua dari populasi menurut nilai finesss mereka (makin baik nilai finesss, lebih besar kesempatan untuk dipilih)
   ii. [Kawin Silang] kawin silang orang tua ke bentuk keturunan baru (keturunan). Kawin silang terjadi dengan kemungkinan tertentu. Jika kawin silang tidak dilakukan, keturunan salinan akan sama dari salah satu orang tua.
   iii. [Mutasi] mengubah tempat sifat keturunan baru (posisi di kromosom dengan kemungkinan tertentu). Jika tidak terjadi mutasi, keturunan langsung hasil kawin silang, atau salinan salah satu dari orang tua.
   iv. [Penerimaan] tempat keturunan baru di populasi baru
d. [Penggantian] menggunakan populasi baru untuk lebih jauh perjalanan algoritma baru
e. [Tes] jika akhir kondisi dipuaskan, hentikan, dan kembali ke solusi terbaik diarus populasi
f. [Pengulangan] menuju ke langkah b.

4. Perancangan sistem

Implementasi aplikasi algoritma genetika pada persoalan knapsack ini secara umum terdiri dari beberapa tahapan, seperti pada flowchart gambar 1.



Gambar 1. Fowchart Algoritma Genetika pada Masalah *Knapsack*

Penjelasan lebih rinci dari proses-proses diatas yaitu seperti dibawah ini.

a. Pembakitan populasi awal

Populasi merupakan kumpulan dari beberapa kromosom dan kromosom merupakan kumpulan dari gen-gen yang merupakan salah satu solusi dari masalah yang akan diselesaikan. Pada permasalahan knapsack ini kromosom di representasikan didalam bentuk string bit. Jumlah gen pada setiap kromosom dikodekan sebanyak jumlah semua barang yang akan dipilih. Barang yang dibawa diberi kode 1 dan yang tidak dibawa diberi kode 0. Misalnya terdapat 6 barang yang akan dibawa seperti pada tabel 1, dari tabel tersebut barang yang akan dibawa yaitu B001, B003, dan B004, yang lainnya tidak karena keterbatasan berat. Sehingga kromosom dapat direpresentasikan seperti pada gambar 2.

Tabel 1. Contoh Barang pada permasalahan knapsack

Kode Barang	Berat (w)	Nilai (v)
B001	3	5
B002	6	8
B003	6	5
B004	2	6
B005	4	9
B006	5	5

1	0	1	1	0	0

Gambar 2. Representasi kromosom pada masalah Knapsack

b. Evaluasi Nilai *Fitness*

Nilai *finess* dihitung dengan menjumlahkan nilai semua barang yang dibawa, tetapi dibatasi berat maksimalnya. Nilai *finess* dan berat total dapat dihitung dengan menggunakan rumus 1 dan 2.

Nilai f*inesss* bernilai 0 jika berat total barang yang dibawa lebih besar dari batas maksimal yang diijinkan dan bernilai NF jika kurang atau sama dengan batas maksimal yang diijinkan. Kromosom yang mempunyai nilai *finess* 0 akan diperbaiki sampai berat total dari kromosom kurang atau sama dengan batas maksimalnya. Caranya dengan melakukan mutasi sebagian bit 1 menjadi 0.

Contoh dari perhitungan nilai *finess* ini misalnya kromosom pada gambar 2 yaitu 101100, dan misalnya dibatasi dengan berat 12, maka dapat dihitung berat total dan nilai finessnya.

$$F = \sum_{i=1}^{n} b_i v_i$$

$$= \sum_{i=0}^{6} b_i v_i$$

=1x3+0x6+1x6+1x2+0x4+0x5=11

Wtot $=\sum_{i=1}^{n} b_i w_i$

$=\sum_{i=1}^{6} b_i w_i$

=1x5+0x8+1x5+1x6+0x9+0x5=16

c. Seleksi

Proses seleksi dilakukan dengan menggunakan metode *roullette wheel selection*. Pada metode ini terdapat beberapa tahap yaitu perhitungan total nilai finesss semua kromosom, perhitungan peluang setiap kromosom, perhitungan peluang komulatif setiap kromosom, dan pembangkitan bilangan acak sebanyak jumlah kromosom untuk dibandingkan dengan peluang komulatif setiap kromosom, sehingga didapatkan kromosom baru.

i. Perhitungan nilai *finesss* total

Setelah semua nilai finesss setiap kromosom didapat selanjutnya di jumlahkan, dengan memakai persamaan 3.

$Ftot = \sum_{i=1}^{n} F_i$ (3)

Ftot = Nilai Finess Total

F_i = nilai Finess kromosom ke-i

N = jumlah kromosom

ii. Peluang setiap kromosom

Untuk menghitung peluang setiap kromosom dengan cara membagi nilai finesss masing-masing kromosom dengan nilai finesss total.

$P_i = \frac{F_i}{Ftot}$ (4)

iii. Peluang komulatif setiap kromosom

Peluang komulatif kromosom pertama sama dengan peluang kromosom pertama, sedangkan peluang komulatif kromosom kedua dan selanjutnya dihitung dengan cara menambahkan peluang komulatif kromosom sebelumnya dengan peluang kromosom yang dihitung.

$PK_i = PK_{i-1} + P_i$

Dimana:

P_i = Peluang kromomosm ke-i

PK_i =Peluang Komulatif Kromosom Ke-i

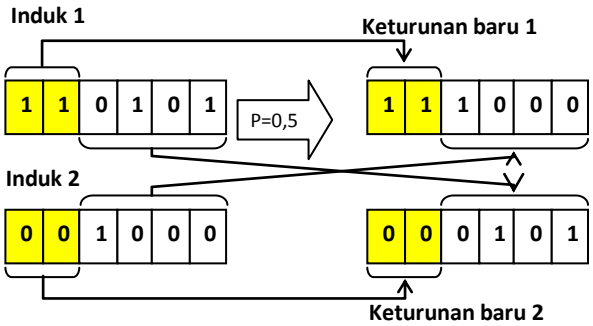
PK_0 =0

iv. Membangkitkan bilangan acak

Langkah terakhir pada proses seleksi yaitu membangkitkan bilangan acak sebanyak jumlah kromosom, kemudian dibandingkan dengan peluang komulatif pada masing-masing kromosom, jika bilangan acak yang dibangkitkan lebih besar dari peluang komulatif sebelumnya, dan kurang dari peluang kromosom yang dibandingkan maka kromosom tersebut terpilih.

d. Crossover

Crossover dilakukan dengan kawin silang satu titik. Langkah pertama cara ini yaitu menentukan bilangan acak pada proses *crossover*. Kemudian menentukan batas *crossover*, jika bilangan acak proses *crossover* kurang dari batas *crossover* maka dilakukan proses *crossover*. Langkah ini dilakukan dengan cara menentukan satu titik potong, kemudian bagian pertama dari kromosom pertama di kombinasikan bagian kedua dari kromosom kedua menjadi keturunan pertama. Sedangkan bagian kedua kromosom pertama di kombinasikan dengan bagian pertama kromosom kedua menjadi keturunan kedua. Hal ini dilakukan pada masing-masing pasangan induk yang akan di *crossover*. Contoh dari kawin silang satu titik ini terlihat pada gambar 3.



Gambar 3. Contoh Proses *Crossover* Peluang *Crossover* 0,5

e. Mutasi

Mutasi dilakukan dengan cara mengubah bit 0 menjadi 1 atau sebaliknya pada kromosom yang terpilih secara acak. Langkah-langkah dari proses mutasi ini yaitu menghitung jumlah gen pada semua kromosom, menentukan jumlah gen yang dimutasi, dan mencari gen yang dimutasi.

i. Menghitung jumlah gen

Jumlah Gen = (jumlah gen 1 kromosom) x (jumlah kromosom)

ii. Jumlah gen yang dimutasi

Untuk menentukan jumlah gen yang dimutasi dilakukan dengan cara membangkitkan bilangan acak mulai dari 1 sampai dengan jumlah gen.

iii. Mencari gen yang dimutasi

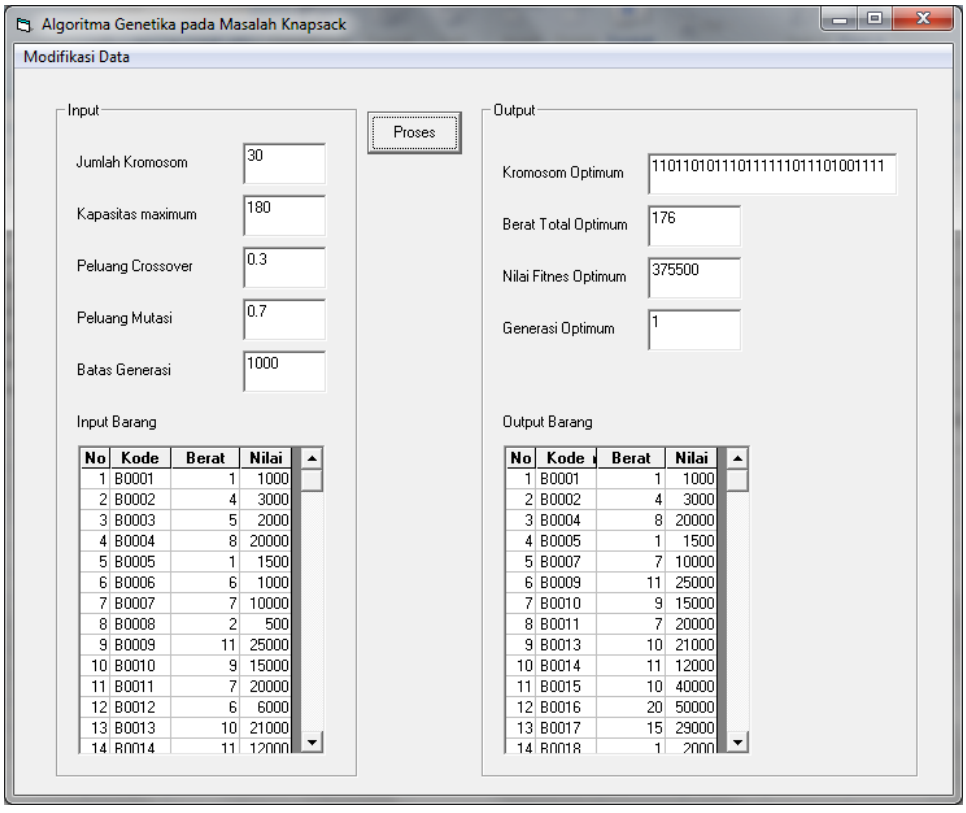
Setelah dicari jumlah gen yang dimutasi selanjutnya mencari gen mana saja yang akan dimutasi yaitu dengan cara membangkitkan bilangan acak 1 sampai dengan jumlah gen sebanyak jumlah gen yang dimutasi.

iv. Proses mutasi

Proses mutasi dilakukan dengan membalikkan bit 0 menjadi 1 atau sebaliknya

5. Implementasi Sistem

Sistem ini dibuat dengan menggunakan bahasa pemrograman Visual basic dan menggunakan database Microsoft Access. Tampilan dari sistem ini yaitu seperti pada gambar 4. Sistem ini terbagi menjadi 2 bagian yaitu bagian input dan output. Bagian input digunakan untuk memberikan variabel input yaitu input barang, jumlah kromosom, kapasitas maksimal berat yang dapat di bawa, peluang *crossover*, peluang mutasi, dan batas generasi maximum. Sedangkan bagian output digunakan untuk menampilkan hasil dari proses algoritma genetika yaitu kromosom optimum yang dihasilkan, berat total pada finesss optimum, nilai finesss optimum dan generasi pada *finesss* optimum.



Gambar 4. Implementasi Sistem

6. Pengujian Sistem

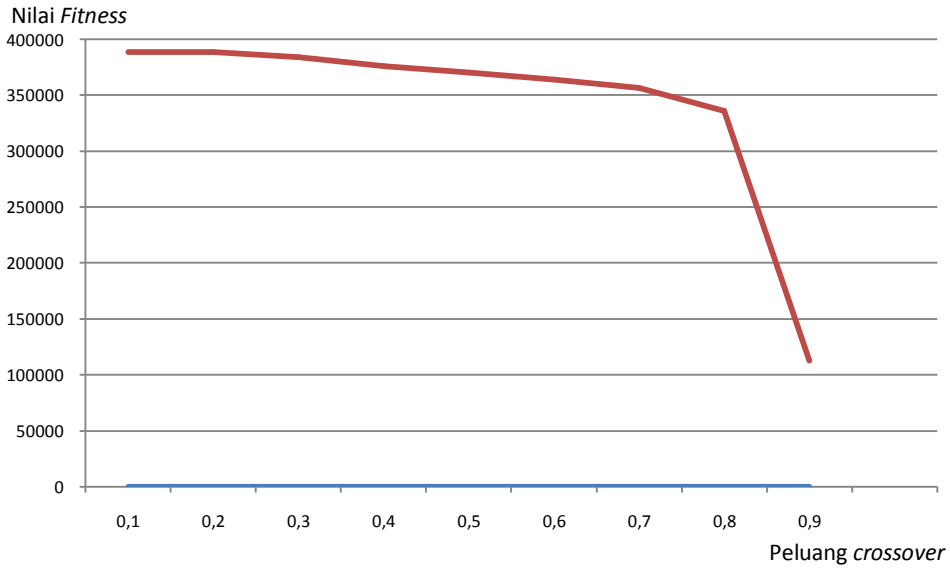
a. Pengujian variasi peluang mutasi

Pengujian dilakukan pada kapasitas maksimum, peluang *crossover*, dan batas generasi yang sama, yaitu kapasitas maksimum 180 , peluang *crossover* 0.3, dan batas generasi sebanyak 1000. Akan di uji beberapa variasi peluang mutasi yaitu diantara 0,1 sampai dengan 0,9 dan dicari nilai *fitness* paling besar, seperti terlihat pada tabel 2.

Tabel 2. Pengujian pada batas kapasitas maximal 180, peluang *crossover* 0.3, dan generasi maximum 1000.

Peluang Mutasi	**Nilai *Fitness* pada Jumlah Kromosom**					**Rata-Rata Nilai *Fitness***
	10	**20**	**30**	**40**	**50**	
0,1	365500	389000	385500	403500	400000	388700
0,2	367000	392500	395500	396500	391500	388600
0,3	373500	379500	386500	393000	387500	384000
0,4	362000	362000	396500	373000	387000	376100
0,5	362000	370000	374500	362000	384500	370600
0,6	361500	368000	362000	362000	368000	364300
0,7	325000	360500	375500	362000	361500	356900
0,8	329000	305000	356000	353000	337000	336000
0,9	0	0	333000	230000	0	112600

Dari Tabel 2 terlihat bahwa setelah dilakukan pengujian sebanyak 9 variasi peluang mutasi yaitu antara 0,1 sampai dengan 0,9 dan masing-masing 4 variasi jumlah kromosom yaitu 10, 20, 30 dan 40 pada kapasitas maksimal, peluang *crossover*, dan batas generasi maksimum yang sama dihasilkan bahwa rata-rata nilai *fitness* terbaik pad pemberian peluang mutasi 0,1. Terlihat bahwa semakin besar pemberian variabel peluang mutasi maka akan menghasilkan nilai *fitness* yang semakin kecil, seperti terlihat pada grafik gambar 5.



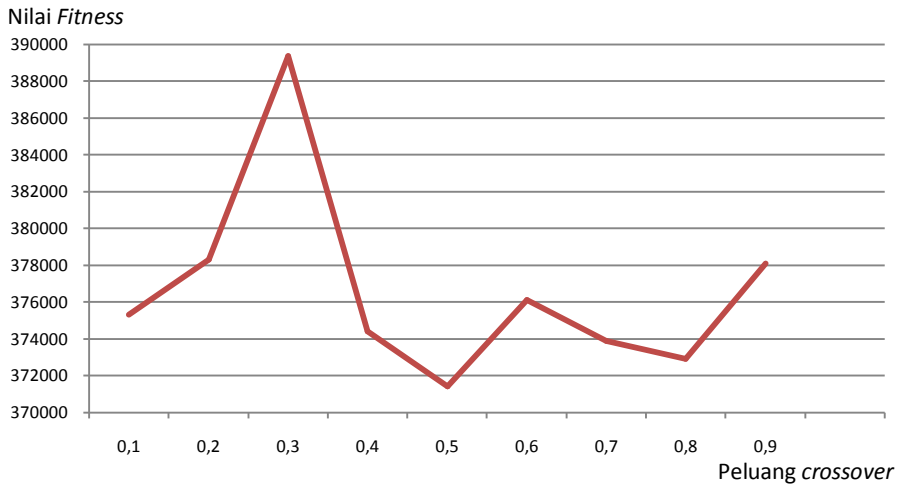
Gambar 5. Grafik peluang mutasi terhadap nilai *fitness*

b. Pengujian variasi peluang *crossover*
Pengujian dilakukan pada kapasitas maksimum, peluang muasi, dan batas generasi yang sama, yaitu kapasitas maksimum 180 , peluang mutasi 0.3, dan batas generasi sebanyak 1000. Akan di uji beberapa variasi *crossover* yaitu diantara 0,1 sampai dengan 0,9 dan dicari nilai *fitness* paling besar, seperti terlihat pada tabel 3.

Tabel 3. Pengujian pada batas kapasitas maximal 180, peluang mutasi 0.3, dan generasi maximum 1000.

Peluang *Crossover*	**Nilai *Fitness* pada Jumlah Kromosom**					**Rata-Rata Nilai *Fitness***
	10	**20**	**30**	**40**	**50**	
0,1	365500	373000	368000	393000	377000	375300
0,2	380500	379500	388000	362000	381500	378300
0,3	411000	389500	382000	383500	381000	389400
0,4	362000	370000	380000	371000	389000	374400
0,5	362000	367000	367000	386000	375000	371400
0,6	368500	376500	384500	377000	374000	376100
0,7	362000	377500	367000	381500	381500	373900
0,8	362000	362000	378000	382000	380500	372900
0,9	367000	368000	389000	378000	388500	378100

Dari Tabel 3 terlihat bahwa setelah dilakukan pengujian sebanyak 9 variasi peluang *crossover* yaitu antara 0,1 sampai dengan 0,9 dan masing-masing 4 variasi jumlah kromosom yaitu 10, 20, 30 dan 40 pada kapasitas maksimal, peluang mutasi, dan batas generasi maksimum yang sama dihasilkan bahwa rata-rata nilai *fitness* terbaik pada pemberian peluang *crossover* 0,3. Grafik rata-rata nilai *fitness* untuk semua peluang *crossover* seperti terlihat pada grafik gambar 6.



Gambar 6. Grafik peluang *crossover* terhadap nilai *fitness*

7. Kesimpulan

Kesimpulan dari penelitian ini diperoleh sebagai berikut:

a. Penerapan algoritma genetika untuk menyelesaikan masalah *knapsack* akan menghasilkan nilai *fitness* terbaik (terbesar) pada pemberian nilai peluang mutasi rata-rata 0,1.
b. Semakin besar pemberian variabel peluang mutasi maka akan menghasilkan nilai *fitness* rata-rata semakin kecil.
c. Penerapan algoritma genetika untuk menyelesaikan masalah *knapsack* akan menghasilkan nilai *fitness* terbaik (terbesar) pada pemberian nilai peluang *crossover* rata-rata 0,3.

8. Referensi

[1] Gen, Mitsuo dan runwaei Cheng.2000, Genetic Algorithms and Enggineering Optimization, Canada, John Wiley & Sons, Inc.

[2] Nopriadi, Putu Niko, Algoritma Genetika Dasar Komputasi Cerdas, Teknik Elektro Universitas Udayana

[3] http://lecturer.eepis-its.edu/~entin/Kecerdasan%20Buatan/Buku/Bab%207%20Algoritma%20Genetika.pdf. Diakses 10 Januari 2012, jam 10.15 WIB

[4] http://repository.upnyk.ac.id/400/1/D-5_PENYELESAIAN_KNAPSACK_PROBLEM_MENGGUNAKAN_ALGORITMA_GENETIKA.pdf. Diakses 10 Januari 2012, jam 10.15 WIB