PENYUSUNAN BARANG DALAM KONTAINER DENGAN MEMPERHATIKAN ASPEK BERAT BENDA MENGGUNAKAN METODE HYBRID ALGORITMA GENETIK

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk mecapai gelar Sarjana Komputer

EVARDA SYUMAN BAHRISA

0910680014

KEMENTRIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

MALANG

2014

LEMBAR PERSETUJUAN

PENYUSUNAN BARANG DALAM KONTAINER DENGAN MEMPERHATIKAN ASPEK BERAT BENDA MENGGUNAKAN METODE HYBRID ALGORITMA GENETIK

Disusun oleh:

Evarda Syuman Bahrisa
091068014

Skripsi ini telah disetujui oleh dosen pembimbing
pada tanggal 16 Desember 2013

Pembimbing I	Pembimbing II
Dian Eka Ratnawati, S.Si, M.Kom	Novanto Yudistira, S.Kom, M.Sc
NIP. 19730619 200212 2 001	NIP. 831110 16 1 1 0425

LEMBAR PENGESAHAN

PENYUSUNAN BARANG DALAM KONTAINER DENGAN MEMPERHATIKAN ASPEK BERAT BENDA MENGGUNAKAN METODE HYBRID ALGORITMA GENETIK

Disusun oleh:

Evarda Syuman Bahrisa
091068014

Skripsi ini diuji dan dinyatakan lulus
pada tanggal 13 Januari 2014

Penguji I	Penguji II
Suprapto, ST., MT.	Wayan Firdaus Mahmudy, S.Si., MT., Ph.D
NIP. 19710727 199603 1 001	NIP. 19720919 199702 1 001

Penguji III

Wijaya Kurniawan, ST., MT
NIK. 820125 16 1 1 0418

Mengetahui,
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Mardji, MT
NIP. 19670801 199203 1 001

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh pihak lain untuk mendapatkan karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebut dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur jiplakan, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (S-1) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan pasal 70).

Malang, 6 November 2013

Evarda Syuman Bahrisa
0910680014

KATA PENGANTAR

Puji syukur atas kehadirat Tuhan Yang Maha Esa karena atas rahmat dan karunia-Nya penulis dapat menyusun skripsi ini dengan baik dan tepat waktu.

Penelitian ini dibuat dengan banyak bantuan dari berbagai pihak untuk membantu menyelesaikan tantangan dan hambatan selama mengerjakan penelitian ini. Oleh karena itu, peneliti mengucapkan banyak terima kasih kepada semua pihak yang terkait khususnya keluarga, teman, dan pembimbing peneliti.

Peneliti menyadari masih banyak kekurangan dalam penelitian yang diangkat. Sehingga peneliti mengundang pembaca untuk memberikan saran dan kritik yang konstruktif pada penelitian ini.

Akhir kata semoga skripsi ini dapat memberikan manfaat bagi kita sekalian.

Malang, November 2013

Penulis

ABSTRAK

Evarda Syuman Bahrisa, 2013, Penyusunan Barang Dalam Kontainer Dengan Memperhatikan Aspek Berat Benda Menggunakan Hybrid Algoritma Genetik. Skripsi Program Studi Informatika, Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya. Pembimbing: Dian Eka Ratnawati, S. Si, M. Kom dan Novanto Yudistira, S. Kom, M.Sc

Dewasa ini perusahaan jasa ekspedisi semakin banyak peminat akibat dari sistem informasi yang meningkat. Oleh sebab itu penyusunan barang pada suatu kargo atau kontainer harus seefisien mungkin agar tidak memakan biaya terlalu tinggi pada ongkos transportasinya. Pada penelitian ini menggunakan Algoritma Genetik dan DBLF sebagai metode yang digunakan untuk mencari solusi pada permasalahan tersebut. Untuk mencapai hasil yang maksimal, pada penelitian ini menggunakan jumlah generasi 1000, jumlah populasi 50, probabilitas *crossover* sebesar 0.4, dan probabilitas mutasi sebesar 0.004.

Kata kunci: algoritma genetik, DBLF, *3D-Bin Packing Problem*

ABSTRACT

Evarda Syuman Bahrisa, 2013, Penyusunan Barang Dalam Kontainer Dengan Memperhatikan Aspek Berat Benda Menggunakan Hybrid Algoritma Genetik. Skripsi Program Studi Informatika, Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya. Pembimbing: Dian Eka Ratnawati, S. Si, M. Kom dan Novanto Yudistira, S. Kom, M.Sc

Nowdays package delivery company has many customer cause of information technology fast-growth. From that reason, stacking package in to a cargo must be efficient so that transportation cost can be reduced. In this research, genetic algorithm and DBLF method been used for searching solution stacking plan forthose package. For reaching the maximum solution generation, population, crossover and mutation rate are 1000, 50, 0.4, and 0.004 respectively.

Keyword: genetic algorithm, DBLF method, 3D-Bin Packing Problem

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Penyusunan barang dalam kontainer merupakan permasalahan umum yang dihadapi oleh perusahaan-perusahaan ekspedisi. Baik yang lokal seperti Tiki JNE, MSA, dan Megacitra maupun internasional seperti DHL, UPS, dan Federal Express.

Akan tetapi hal tersebut dapat menjadi permasalahan yang serius apabila terjadi kelebihan muatan pada saat kontainer disusun pada alat angkut (kapal sebagai contohnya). Sebuah kapal ferry di tahun 2002 yang bernama MV Joola mengalami sebuah kecelakaan akibat kelebihan muatan. Menurut [BOJ-09], kapal yang seharusnya hanya dapat memuat penumpang sebanyak 580 orang (termasuk awak) diisi oleh 1.800 lebih orang. Dan pada pukul 23.00 waktu setempat kapal terbalik. Tak perlu waktu lama untuk membalik kapal tersebut, hanya 5 menit.

Menurut [MAR-06], perdagangan antar negara menggunakan L/C (letter of credit) sebagai sarana pembayaran. Pada umumnya terdapat perjanjian yang menyatakan bahwa jumlah barang yang dikirim minimal 90% dan maksimal 110% . Sedangkan perjanjian awal transaksi sebelum L/C, ditentukan maksimal 90% pengisian volume kontainer. Sehingga volume kontainer yang harus diisi berkisar 80%-100%.

Dalam masalah penyusunan barang dalam kontainer, biasanya terdapat barang pecah belah yang tidak dapat leluasa untuk diubah tata letaknya. Sehingga rotasi yang diperbolehkan hanya terhadap sumbu-Y (tinggi).

Atas permasalahan di atas, penulis melakukan penelitian dengan judul " **Penyusunan Barang Pada Kontainer Dengan Memperhatikan Aspek Berat Benda Menggunakan Metode Hybrid Algoritma Genetik**". Dengan harapan bahwa aspek berat juga diperhatikan, akan tetapi secara dimensi atau volume tetap maksimal dan memberikan keuntungan bagi perusahaan ekpedisi.

Algoritma Genetik sendiri merupakan metode yang digunakan untuk menyelesaikan masalah rumit (NP). Algoritma Genetik merupakan bagian sistem

evolusioner yang dikemukakan oleh John Holland beserta murid-muridnya. Algoritma Genetik merupakan suatu abstraksi dari evolusi biologis dan memberikan suatu kerangka kerja secara teoritis tentang adaptasi dalam algoritma genetik [IGN-00].

Terdapat 3 operator Algoritma Genetik yaitu, Seleksi (*selection*), Perkawinan Silang (*crossover*), dan Mutasi (*mutation*). Ketiga operator tersebut berfungsi untuk membuat kumpulan individu/solusi menjadi lebih bervariatif. Sehingga diharapkan dapat mencapai nilai *fitness* yang maksimum.

Pada penentuan nilai *fitness* di penelitian ini menggunakan metode *Deepest Bottom-Left with Fill* (DBLF). DBLF merupakan perpanjangan dari algoritma *Bottom-Left with Fill*(BLF) yang ditujukan pada obyek 2D.

Pada penelitian ini menggunakan metode Algoritma Genetik dan DBLF sebagai penyelesaian kasus penyusunan barang pada kontainer. Diharapkan dapat menyelesaikan permasalahan tersebut.

1.2 Rumusan Masalah

Melihat dari latar belakang di atas, rumusan masalah dapat dikerucutkan menjadi 3 yaitu:

1. Bagaimana menentukan representasi kromosom dan operator genetik lainnya pada kasus penyusunan barang pada kontainer
2. Bagaimana cara menghasilkan *fitness* yang optimal berdasarkan volume dan berat barang dengan Algoritma Genetik untuk kasus penyusunan barang pada kontainer.
3. Bagaimana mengevaluasi *fitness* hasil penghitungan metode Algoritma Genetik pada kasus penyusunan barang pada kontainer.

1.3 Batasan Masalah

Untuk menghindari melebarnya pembahasan pada penelitian ini maka penelitian ini berfokus pada:

1. Barang-barang yang disusun berupa benda segi empat.
2. Batas maksimal berat pada kontainer sebesar 26.280 kilogram
3. Dimensi kontainer sebesar 120 dm × 23 dm × 23 dm
4. Aplikasi dibangun di lingkungan JDK 7 *Update* 25

5. Aplikasi dibangun melalui NetBeans IDE 7.3.1
6. Batasan parameter evaluasi berdasarkan volume (utilitas) dan berat barang
7. Pengujian difokuskan pada optimasi penentuan variabel pada Algoritma Genetik

1.4 Tujuan

1. Mengetahui perancangan kromosom dan operator Algoritma Genetik secara optimal untuk masalah penyusunan barang dalam kontainer dengan memperhatikan aspek berat.
2. Mengetahui cara mengevaluasi hasil secara tepat dari perangkat lunak penyusunan barang dalam kontainer.
3. Mengetahui cara menganalisis hasil perhitungan dengan menggunakan Algoritma Genetik pada kasus penyusunan barang pada container.

1.5 Manfaat

1. Dapat membantu menyelesaikan permasalahan muatan kontainer seperti yang banyak dihadapi oleh perusahaan-perusahaan ekpedisi.
2. Untuk memberikan gambaran atau simulasi penyusunan terlebih dahulu. Sehingga pada saat penyusunan barang, tidak memakan waktu berlebih.

1.6 Sistematika Penulisan

Sistematika penulisan laporan skripsi ini adalah:

BAB I PENDAHULUAN

Berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian,manfaat penelitian, dan sistematika penulisan.

BAB II KAJIAN PUSTAKA DAN DASAR TEORI

Berisi uraian kajian pustaka dan dasar teori mengenai Algoritma Genetik, DBLF, dan hubungan antara massa dan volume. Pada bagian ini juga akan dibahas mengenai tinjauan penelitian sebelumnya.

BAB III METODE PENELITIAN DAN PERANCANGAN

Berisi tentang alur penyusunan skripsi, deskripsi umum sistem, representasi penyusunan barang dalam kontainer dengan Algoritma Genetik, analisis data, dan perancangan sistem pada penelitian.

BAB IV IMPLEMENTASI

Bab ini berisi tentang lingkungan implementasi, implementasi program, dan kendala saat implementasi program serta solusi.

BAB V PENGUJIAN DAN ANALISIS

Berisi mengenai analisa hasil penghitungan metode dengan berbagai kombinasi nilai jumlah generasi, jumlah populasi, probabilitas *crossover* dan probabilitas mutasi pada Algoritma Genetik.

BAB VI PENUTUP

Pada bab ini berisi kesimpulan yang diambil berdasarkan analisa penghitungan metode meliputi kelebihan dan kekurangan metode serta saran untuk penelitian selanjutnya.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

2.1 Algoritma Genetik

"Algoritma Genetik merupakan algoritma adaptif yang digunakan untuk memecahkan masalah pencarian dan masalah optimasi [KAR-04]". Sehingga jika ditelaah secara lebih dalam, Algoritma Genetik hanya dapat menyelesaikan permasalahan secara umum. Jika menginginkan menyelesaikan permasalahan khusus perlu ditambahkan algoritma lain agar pencarian solusi lebih cepat dan lebih tepat sasaran.

Hal tersebut sejalan dengan hasil penelitian [JAK-93] yang menyatakan bahwa kombinasi dari algoritma deterministik dan Algoritma Genetik dapat menghasilkan solusi yang diinginkan dari permasalahan yang dihadapi. Selain itu, menurut [KAN-12] Algoritma Genetik secara umum merupakan metode yang "cukup cepat" dan "cukup baik" dalam pencarian solusi.

Algoritma Genetik sendiri memiliki 3 operator di dalamnya. Yaitu, Seleksi (*selection*), Perkawinan Silang (*crossover*), dan Mutasi (*mutation*). dalam penelitian ini penulis menggunakan *Roulette Wh*eel sebagai Seleksi, *Positioning-based Crossover* (POS) sebagai Perkawinan Silang, dan *Swap Mutation* sebagai Mutasi.

2.1.1 *Roulette Wheel*

Adalah salah satu metode Seleksi yang digunakan untuk menentukan pasangan yang digunakan pada proses Perkawinan Silang. Menurut [MAT-05] metode ini berdasarkan probabilitas dari *fitness* yang dimiliki oleh masing-masing individu. Semakin besar nilai *fitness* yang dimiliki oleh individu tersebut semakin besar kemungkinan individu tersebut untuk dipilih sebagai *parent* untuk proses Perkawinan Silang.

2.1.2 *Position-Based Crossover* (POS)

Merupakan salah satu metode yang digunakan pada Perkawinan Silang. Teknik ini dapat diterapkan pada kromosom yang tersusun (*Ordered*

Chromosome). Hampir sama dengan *Order Crossover* (OX) hanya saja pada POS *parent* 1 digunakan untuk menentukan titik-titik kromosom *offspring* yang digunakan pada *crossover* sedangkan *parent* 2 berfungsi untuk mengisi sisa dari kromosom *offsring* yang belum terisi.

Penentuan titik-titik yang digunakan untuk *crossover* berdasarkan angka acak yang diberikan dan nilainya berada di bawah ambang probabilitas *crossover* yang telah dibangkitkan sebelumnya.

2.1.3 *Swap Mutation*

Mutasi yang digunakan pada penelitian ini adalah *Swap Mutation*. *Swap Mutation* juga merupakan salah satu metode yang digunakan pada *Ordered Chromosome*. Jadi secara garis besar pada suatu individu, pada tiap kromosom akan dibangkitkan nilai secara acak.

Kemudian dicocokkan dengan nilai probabilitas mutasi yang telah dibangkitkan sebelumnya. Jika lebih kecil dari nilai probabilitas mutasi maka akan terjadi mutasi antar kromosom. Akan tetapi harus ada 2 kromosom yang terlibat dalam mutasi.

2.2 DBLF

Algoritma DBLF merupakan perpanjangan algoritma BLF yang dikembangkan oleh [KAR-04]. Algoritma BLF yang diperkenalkan oleh [JAK-93] membahas masalah mengenai penempatan bentuk segiempat dan polygon lain secara 2D yang biasa ditemui pada industri logam dan pakaian. Menurut [KAR-04] DBLF sendiri merupakan metode yang digunakan untuk memecahkan permasalahan penyusunan benda pada 3D.

Dalam DBLF, benda digerakkan sejauh mungkin ke arah dalam (sejajar sumbu z), kemudian sejauh mungkin ke arah bawah (sejajar sumbu y), kemudian sejauh mungkin ke arah kiri (sejajar sumbu x).

Sehubungan dengan tingkat kompleksitas algoritma DBLF yang tinggi maka diperlukan komputasi yang efisien.

2.3 Hubungan Antara Massa dan Volume

Menurut [CHE-12] antara massa dan volume tidak ada hubungannya. Sebab kedua hal tersebut merupakan ciri atau properti dari tiap obyek. Massa menunjukkan ukuran materi dari sebuah obyek. Biasa dilambangkan dengan miligram, gram, atau kilogram. Sedangkan volume merupakan ukuran untuk menunjukkan bahwa suatu obyek menempati dimensi yang dilambangkan dengan milimeter, meter, atau kilometer.

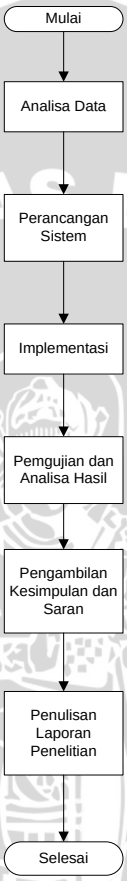
Akan tetapi lain hal dengan obyek yang sama pembentuk materinya. Semisal kubus tembaga dan silinder tembaga. Kedua obyek tersebut memiliki pembentuk materi yang sama yaitu tembaga. Sehingga massa jenis kedua obyek sama dan massa berbanding lurus dengan volume.

Dalam penelitian ini, penulis mengasumsikan bahwa benda yang lebih berat memiliki kestabilan lebih daripada benda yang lebih ringan. Dan massa tidak berbanding lurus dengan volume benda (diasumsikan bahwa massa jenis tiap benda berbeda).

BAB III
METODE PENELITIAN

3.1 Alur Penyusunan Skripsi

Pada penyusunan skripsi di penelitian ini diatur seperti pada Gambar 3.1.



Gambar 3.1 Diagram Alur Penyusunan Skripsi

3.2 Deskripsi Umum Sistem

Dalam menentukan solusi, program membutuhkan masukan berupa daftar barang yang akan disusun, jumlah populasi, jumlah generasi, dimensi kontainer, batas maksimum berat pada kontainer, probabilitas *crossover*, probabilitas mutasi, porsi fitness berat, serta porsi fitness utilitas.

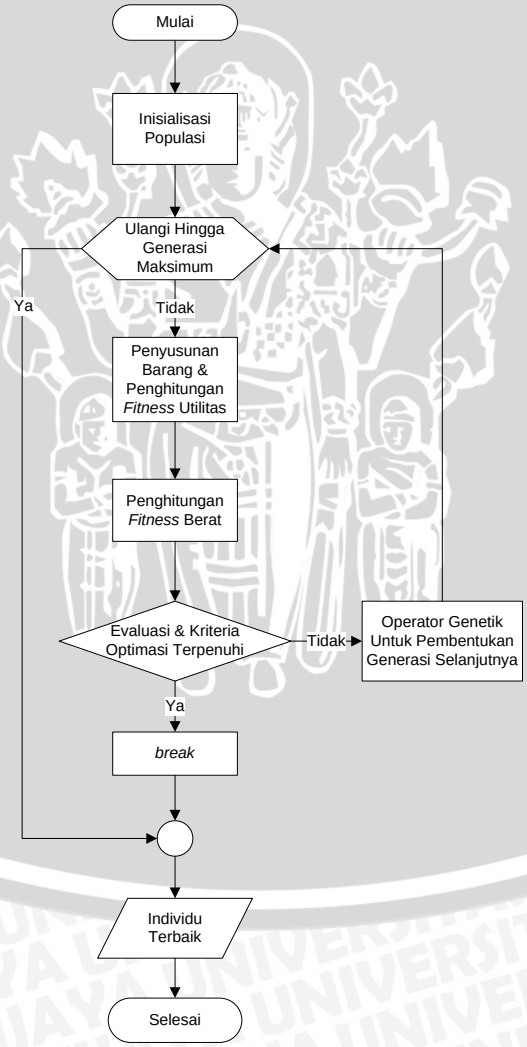
Untuk daftar barang yang akan disusun terdapat 4 buah fitur yang harus diisi. Yaitu panjang barang, lebar barang, tinggi barang, dan berat barang. Keempat fitur tersebut akan disimpan pada database untuk digunakan pada perhitungan selanjutnya.

Untuk penentuan probabilitas *crossover*, probabilitas mutasi, penentuan dimensi kontainer, batas penentuan berat pada kontainer, serta porsi fitness utilitas dan berat dilakukan kemudian.

Kemudian sistem menggunakan Algoritma Genetik dan DBLF dalam perhitungan untuk menghasilkan solusi terbaik yang akan ditawarkan kepada *user*.

3.3 Representasi Penyusunan Barang dalam Kontainer dengan Algoritma Genetik

Proses penyusunan barang dalam kontainer menggunakan Algoritma Genetik dan DBLF terdiri dari beberapa tahapan seperti yang ditunjukkan pada Gambar 3.2.

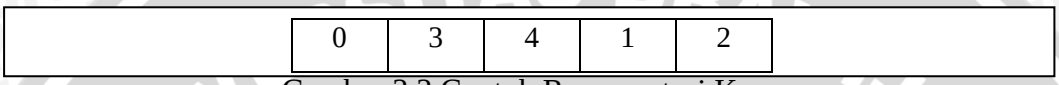


Gambar 3.2 Diagram Alir Sistem

3.3.1 Inisialisasi Populasi Awal

Inisialisasi populasi awal bertujuan untuk membentuk individu sebanyak jumlah populasi yang diinginkan sesuai dengan banyaknya populasi yang dituliskan pada masukan sistem. Penentuan individu sendiri adalah acak.

Kromosom dari individu terdiri dari kode dari barang yang telah dimasukkan dalam database. Dan urutan penyusunan dari barang-barang tersebut berurutan tergantung dari penyusunan kromosom. Untuk lebih jelas dapat dilihat pada Gambar 3.3.

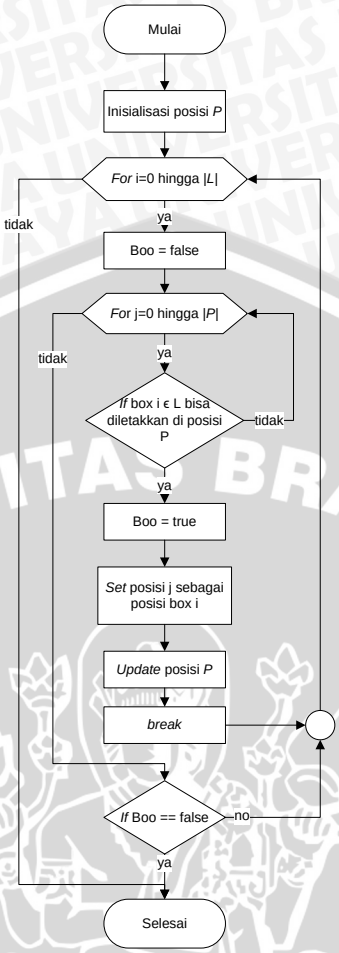


Gambar 3.3 Contoh Representasi Kromosom

Pada Gambar 3.3 dapat dilihat terdapat angka 0, 3, 4, 1, 2. Jadi urutan penyusunan barangnya dimulai dari barang dengan kode 0 terlebih dahulu kemudian disusul dengan barang berkode 3 dan seterusnya. Untuk penentuan panjang kromosom tergantung dari jumlah barang yang akan disusun.

3.3.2 Penyusunan Barang dan Penghitungan *Fitness* Utilitas

Penyusunan barang dilakukan dengan menggunakan metode DBLF. Jadi sesuai dengan yang telah disampaikan pada bab 2.2, maka alur metode DBLF dijelaskan pada Gambar 3.4.



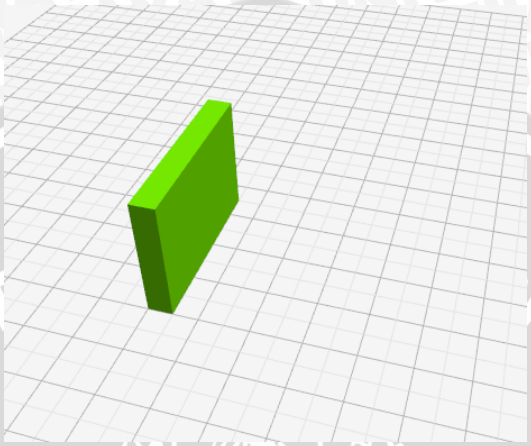
Gambar 3.4 Diagram Alir Algoritma DBLF

Pada Gambar 3.4 terdapat simbol P, *L*, |P|, |*L*|. P menunjukkan posisi yang dapat diisi oleh barang, |P| menunjukkan banyaknya posisi yang tersedia. *L* merupakan daftar susunan dari barang, sedangkan |*L*| merupakan banyaknya data dari daftar susunan barang. Dimisalkan dimensi barang seperti pada Tabel 3.1.

Tabel 3.1 Contoh daftar dimensi barang

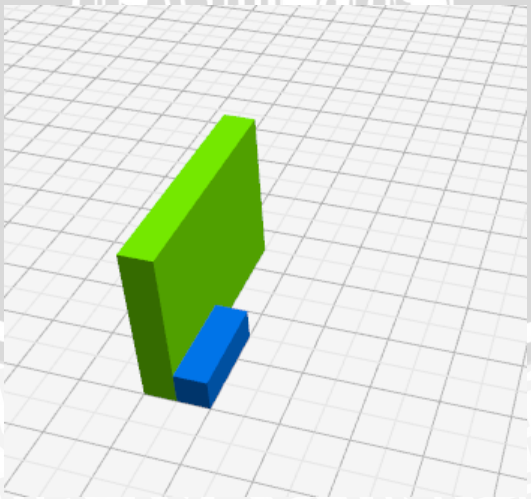
Kode Barang	Panjang	Lebar	Tinggi
0	1	7	5
1	1	4	2
2	2	3	3
3	1	3	1
4	3	2	2

Kemudian diasumsikan bahwa panjang kontainer 4, lebar kontainer 7, dan tinggi kontainer 5. Jika terdapat individu dengan kromosom yang sama dengan Gambar 3.3, maka penyusunan barang dengan algoritma DBLF adalah sebagai berikut.



Gambar 3.5 Langkah 1 Penyusunan Barang dengan DBLF

Sesuai dengan urutan pada individu, maka barang pertama yang disusun adalah barang dengan kode 0. Barang dengan kode 0 memiliki dimensi panjang 1 (x), lebar 7 (z), dan tinggi 5 (y). Kemudian diletakkan pada posisi yang terdalam, terbawah, dan terkiri. Karena baru memulai maka otomatis posisi yang dapat diisi adalah x = 0, y = 0, z = 0 (0,0,0). Setelah terisi seperti pada Gambar 3.5 maka posisi yang tersedia berubah menjadi (1,0,0), (0,5,0), (0,0,7).



Gambar 3.6 Langkah 2 Penyusunan Barang dengan DBLF

Kemudian menuju ke urutan selanjutnya yaitu barang dengan kode 3. Barang dengan kode 3 memiliki dimensi panjang 1 (x), lebar 3 (z), dan tinggi 1 (y). Menurut hukum algoritma DBLF, posisi yang terpilih adalah yang terdalam, terbawah terlebih dahulu. Jadi posisi yang digunakan adalah (1,0,0). Pemetaannya dapat diilustrasikan pada Gambar 3.6. Kemudian posisi yang tersedia di-*update* menjadi (2,0,0), (1,1,0), (1,0,3), (0,5,0), (0,0,7). Dan begitu seterusnya hingga urutan barang yang ada di dalam individu mencapai akhir.

Untuk penentuan Fitness Utilitas dapat dilihat pada persamaan (3-1).

$$F_u = \frac{\sum V_{Barang}}{V_{Kontainer}} \quad (3\text{-}1)$$

F_u = *Fitness* utilitas

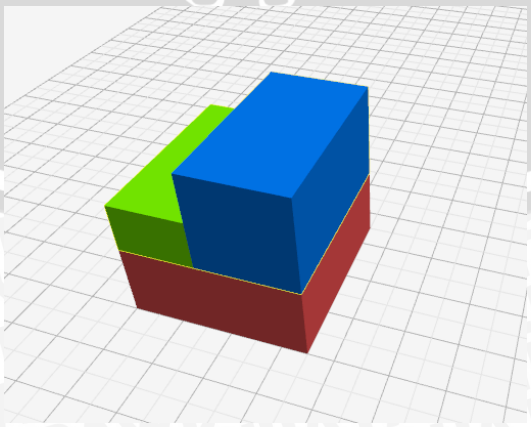
V_{Barang} = Volume barang yang tersusun

$V_{Kontainer}$ = Volume kontainer

3.3.3 Penentuan Nilai Fitness Berat

Penentuan nilai fitness berat suatu pola penyusunan berdasarkan penalti yang dilakukan. Jika semakin besar melakukan pelanggaran semakin kecil *fitness* berat yang diperoleh oleh pola penyusunan tersebut begitu pula sebaliknya.

Merujuk dengan apa yang telah dijabarkan pada bab 2.3, jika benda yang berada di susunan terbawah lebih berat daripada yang berada di susunan atasnya maka tidak dikenakan penalti. Akan tetapi jika sebaliknya maka dikenakan penalti sebesar 1. Dimisalkan terdapat kondisi seperti pada Gambar 3.7.



Gambar 3.7 Contoh Penyusunan Barang

Dengan barang berwarna merah memiliki berat 100 Kg, barang berwarna hijau memiliki berat 25 Kg, dan barang berwarna biru memiliki berat 50 Kg. Persamaan untuk menentukan penalti adalah sebagai berikut.

$$P = \begin{cases} 1, (M_{Bawah} < \sum M_{Atas}) \\ 0, (M_{Bawah} \geq \sum M_{Atas}) \end{cases} \quad (3\text{-}2)$$

P = Penalti
M_{Bawah} = Berat pada susunan bawah
M_{Atas} = Berat pada susunan atas

Sehingga nilai penalti yang diberikan pada barang merah adalah 0. Begitu pula seterusnya. Ketika telah mendapat seluruh penalti maka *fitness* berat dapat dihitung dengan persamaan (3-3).

$$F_b = \frac{B_{tersusun} - \sum P}{B_{tersusun}} \quad (3\text{-}3)$$

F_b = *Fitness* berat
$B_{tersusun}$ = Banyaknya barang yang tersusun
P = Penalti yang diperoleh

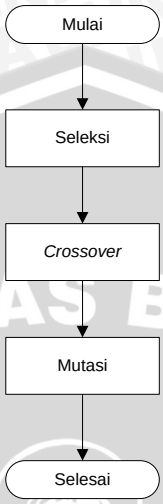
Untuk memperoleh *fitness* secara keseluruhan (berdasarkan utilitas dan berat) menggunakan persamaan (3-4).

$$F = P_u \times F_u + P_b \times F_b \quad (3\text{-}4)$$

F = Fitness keseluruhan
P_u = Porsi fitness utilitas
F_u = Fitness utilitas
P_b = Porsi fitness berat
F_b = Fitness berat

3.3.4 Operator Algoritma Genetik

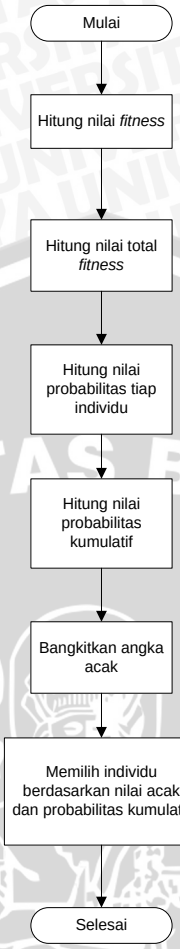
Ada tiga operator Algoritma Genetik yang digunakan pada penelitian ini. Yaitu seleksi, *crossover*, mutasi.



Gambar 3.8 Alur proses operator Algoritma Genetik

3.3.4.1 Seleksi

Seleksi dalam Algoritma Genetik bertujuan untuk memilih indukan yang digunakan untuk proses selanjutnya. Seleksi yang digunakan adalah *Roulette Wheel*. Tahapan *Roulette Wheel* sebagai berikut.



Gambar 3.9 Alur Seleksi

1) Menghitung nilai *fitness* dari masing-masing individu. Penghitungannya mengacu pada persamaan (3-4).
2) Menghitung total nilai *fitness*.
3) Memberikan nilai probabilitas seleksi pada masing-masing individu. Nilai probabilitas seleksi dapat dilihat pada persamaan (3-5)

$$P_{seleksi} = \frac{F_i}{F_t} \quad (3\text{-}5)$$

$P_{seleksi}$ = Probabilitas seleksi individu
F_i = Fitness individu
F_t = Fitness total

4) Menghitung nilai probabilitas kumulatif pada tiap individu. Nilai probilitas kumulatif dapat dilihat pada persamaan (3-6)

$$P_k = \sum_{i=n}^{i} P_n \quad (3\text{-}6)$$

P_k = Probabilitas kumulatif

P_n = Probabilitas individu ke-n

5) Bangkitkan angka acak dari 0 hingga 1 pada tiap individu.
6) Kemudian bandingkan nilai acak dengan nilai probabilitas kumulatif. Jika nilai acak kurang dari sama dengan nilai probabilitas kumulatif maka individu yang terdekat nilainya dengan probabilitas kumulatif yang terpilih untuk proses selanjutnya.

Dimisalkan terdapat populasi sebagai berikut.

Individu 1:

0	2	4	1	3

Individu 2:

2	1	3	0	4

Individu 3:

3	2	0	4	1

Gambar 3.10 Contoh populasi

Individu 1 memiliki *fitness* sebesar 67.34, individu 2 memiliki *fitness* 70.3, dan individu 3 memiliki *fitness* sebesar 74.5. Kemudian dihitung fitness total dan ditemukan hasil 212.14. Kemudian dicari masing-masing probabilitas seleksinya.

- $Individu\ 1 = \frac{67.34}{212.14} = 0.32$
- $Individu\ 2 = \frac{70.3}{212.14} = 0.33$
- $Individu\ 3 = \frac{74.5}{212.14} = 0.35$

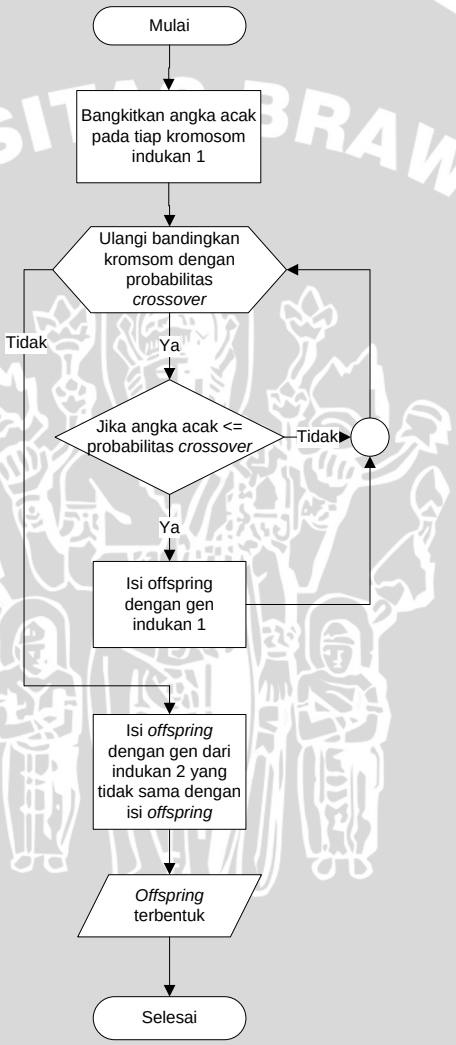
Kemudian untuk mendapatkan probabilitas kumulatif dapat melihat pada persamaan (3-6). Dan didapatkan hasil sebagai berikut.

- $Individu\ 1 = 0.32$
- $Individu\ 2 = 0.32 + 0.33 = 0.65$
- $Individu\ 3 = 0.32 + 0.33 + 0.35 = 1$

Bangkitkan angka acak. Misalkan didapat angka 0.7. Maka individu yang terpilih adalah individu yang memiliki probabilitas kumulatif yang kurang dari 0.7 dan yang paling mendekati yaitu Individu 2.

3.3.4.2 *Crossover*

Pada *crossover* mengambil indukan dari hasil seleksi yang dilakukan sebelumnya. Untuk langkah-langkah *crossover* dapat dilihat pada Gambar 3.11.



Gambar 3.11 Alur *crossover*

1) Bangkitkan angka acak antara 0 dan 1 pada tiap kromosom indukan 1. Hal ini bertujuan untuk menentukan kromosom mana saja yang akan digunakan untuk individu selanjutnya.

2) Kemudian periksa hasil angka acak yang diberikan pada tiap kromosom dengan probabilitas *crossover* yang telah diberikan sebelumnya. Ualngi hingga seluruh kromosom pada indukan 1 telah diperiksa.
3) Jika nilai dari angka acak lebih kecil atau sama dengan nilai probabilitas *crossover* maka kromosom tersebut digunakan untuk individu selanjutnya (*offspring*).
4) Kemudian isi kromosom yang masih kosong pada *offspring* dengan kromosom yang terdapat pada indukan 2 yang berbeda dengan yang ada di dalam *offspring*.
5) Individu baru telah terbentuk dan dapat digunakan untuk proses operator Algoritma Genetik selanjutnya.

Dimisalkan terdapat dua indukan sebagai berikut.

Indukan 1:

2	1	3	0	4

Indukan 2:

3	2	0	4	1

Gambar 3.12 Contoh indukan

Kemudian dibangkitkan angka acak pada indukan 1 sehingga menghasilkan kondisi sebagai berikut.

Indukan 1:

0.4	0.6	0.5	0.7	0.8
2	1	3	0	4

Indukan 2:

3	2	0	4	1

Gambar 3.13 Pembangkitan nilai acak pada kromosom indukan

Pada probabilitas crossover diberikan nilai 0.5. Maka kromosom yang terpilih untuk dilakukan kromosom adalah sebagai berikut.

Indukan 1:

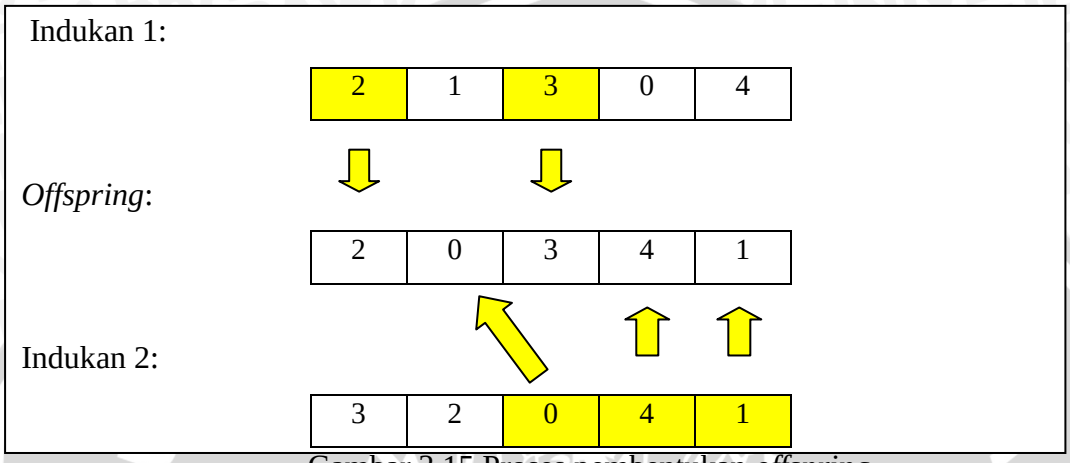
2	1	3	0	4

Indukan 2:

3	2	0	4	1

Gambar 3.14 Kromosom indukan yang terpilih

Pada indukan 1 yang memiliki warna kuning pada kolom adalah kromosom yang terpilih sebagai kromosom *offspring*. Untuk kolom yang berwarna merah pada indukan 2 tidak dapat digunakan sebagai kromosom *offspring*. Sebab akan menimbulkan *redundant* pada kromosom *offspring*. Pembentukan *offspring* dapat dilihat pada Gambar 3.15.

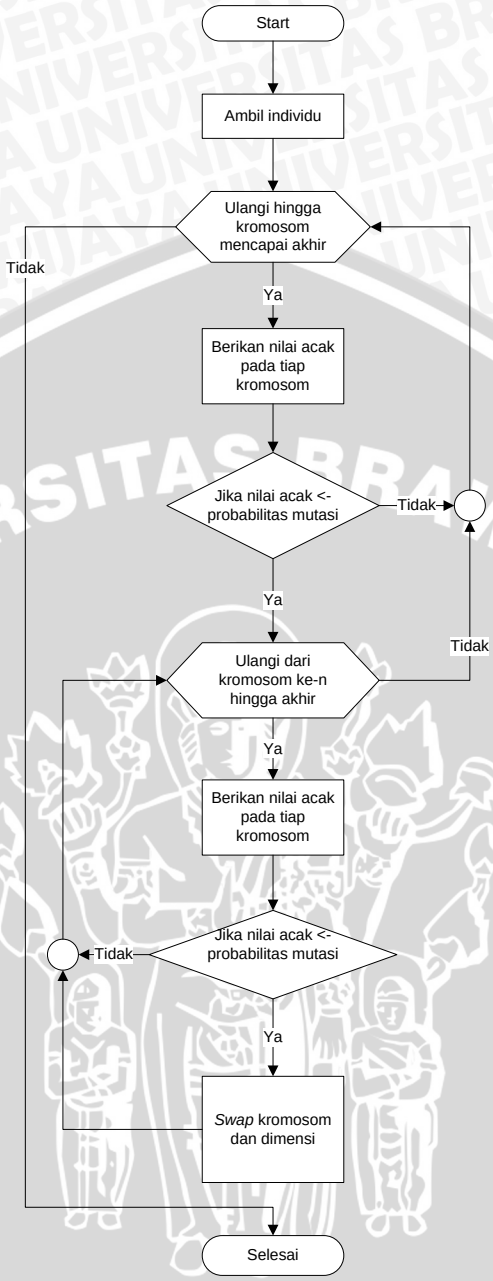


Gambar 3.15 Proses pembentukan *offspring*

Offspring dapat digunakan untuk proses operator Algoritma Genetik selanjutnya.

3.3.4.3 Mutasi

Pada proses mutasi ini bergantung pada probabilitas mutasi yang diberikan oleh *user*. Akan tetapi biasanya mutasi yang baik adalah probabilitas mutasi yang kurang dari 1%. Untuk proses mutasi dapat dilihat pada Gambar 3.16.

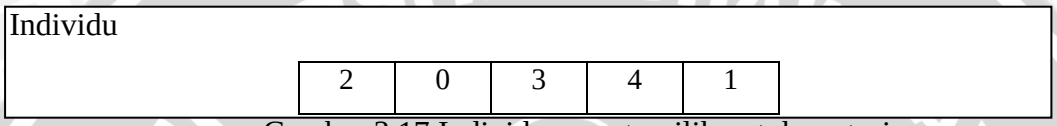


Gambar 3.16 Alur mutasi

1) Ambil individu untuk dilakukan proses mutasi. Individu yang terpilih adalah individu hasil *crossover*.
2) Jika sudah terpilih maka dilakukan perulangan untuk menentukan kromosom mana yang terjadi mutasi.
3) Kemudian memberikan nilai acak pada tiap kromosom. Jika nilai probabilitas mutasi lebih besar sama dengan nilai acak di kromosom maka pada kromosom tersebut akan terjadi mutasi

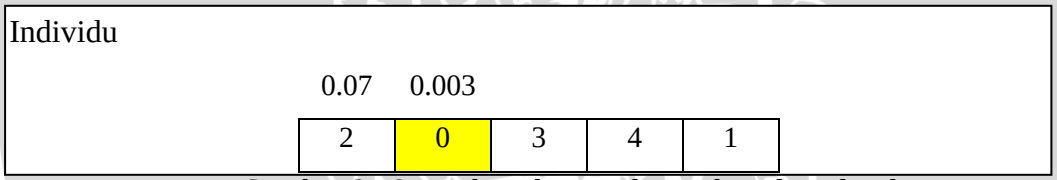
4) Lakukan pengulangan lagi dimulai dari kromosom yang akan terjadi mutasi. Perulangan dilakukan hingga kromosom akhir.
5) Kemudian memberikan nilai acak pada kromosom. Jika nilai acak lebih kecil sama dengan nilai probabilitas maka terjadi mutasi di kromosom ini dengan kromosom sebelumnya. Selain itu, terjadi pengubahan peletakan barang dengan cara mengubah panjang barang menjadi lebar barang. Begitu pula sebaliknya.

Dimisalkan individu yang terpilih untuk melakukan mutasi adalah sebagai berikut.



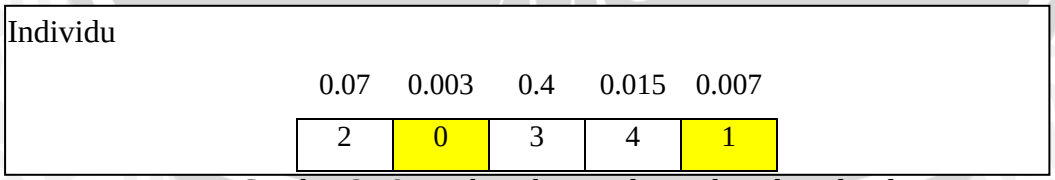
Gambar 3.17 Individu yang terpilih untuk mutasi

Kemudian bangkitkan angka acak pada individu tersebut. Sehingga individu tersebut memiliki kondisi sebagai berikut.



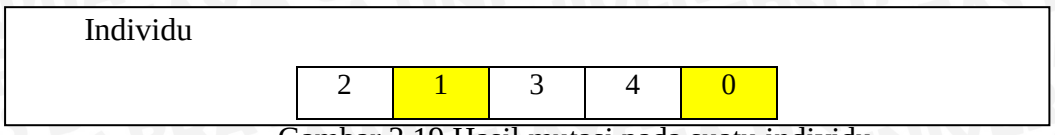
Gambar 3.18 Pembangkitan nilai acak pada individu

Untuk nilai probabilitas mutasi adalah 0.01. Jadi kromosom yang terpilih untuk mutasi adalah kromosom yang kedua. Kemudian menuju ke perulangan yang kedua. Dimulai dari kromosom ketiga (setelah kolom kuning pada gambar 3.18).



Gambar 3.18 Pembangkitan nilai acak pada individu

Kemudian dilakukan *swap* seperti yang diilustrasikan pada Gambar 3.19.



Gambar 3.19 Hasil mutasi pada suatu individu

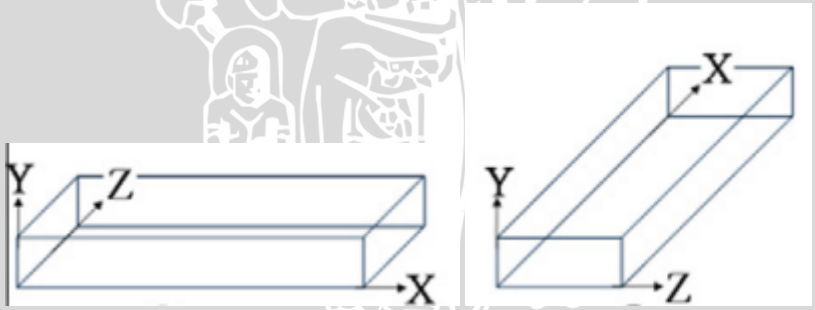
Selain itu, pada dimensi dari barang mengalami perubahan. Merujuk pada Tabel 3.1 maka daftar dimensi berubah seperti pada Tabel 3.2.

Tabel 3.2 Daftar dimensi barang setelah dimutasi

Kode Barang	Panjang	Lebar	Tinggi
0	7	1	5
1	4	1	2
2	2	3	3
3	1	3	1
4	3	2	2

Dengan harapan bahwa pengubahan peletakkan barang dapat memberikan lebih banyak pilihan dalam menyusun barang dalam kontainer. Penulis hanya mencantumkan 2 cara dalam peletakkan sebab jika bisa membalik barang untuk mendapatkan banyak pilihan dalam penyusunan dapat merusak isi barang. Sebab terdapat barang yang perlakuannya khusus dalam penyusunan. Seperti tidak boleh dibalik.

Untuk ilustrasi 2 cara dapat dilihat pada Gambar 3.20.



Gambar 3.20 Kondisi peletakkan barang

Sumber: [KAN-12]

3.4 Analisis Data

Pada penenlitian ini, data yang digunakan dalam pengujian menggunakan data yang terdapat pada penelitian [KAN-12] yang telah disimpan dalam *database*. *Database* tersebut berisi informasi mengenai daftar dimensi panjang, lebar, dan tinggi barang. Serta berat dari masing-masing barang. Untuk dimensi dan maksimum berat pada kontainer, *user* mengisi sendiri pada program. Sama

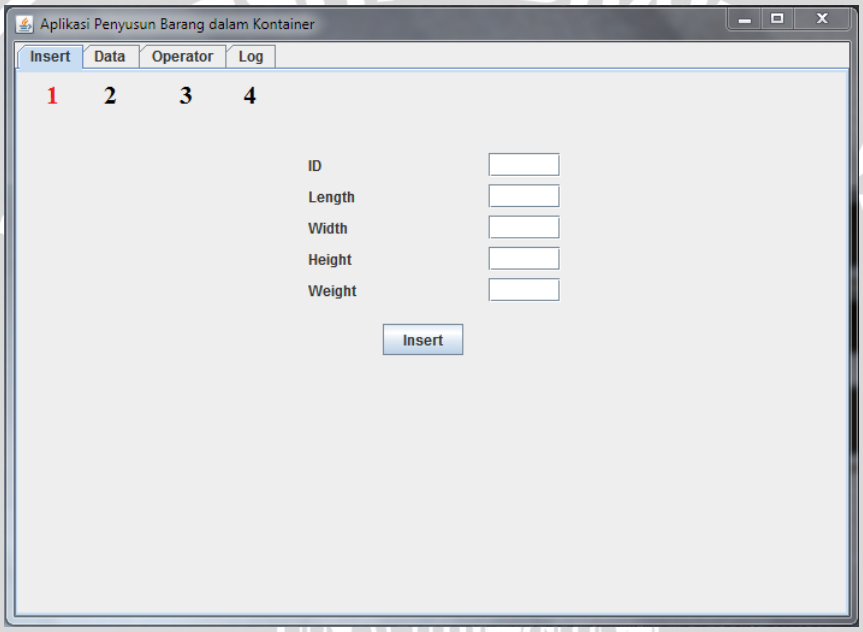
halnya dengan porsi *fitness* utilitas, porsi *fitness* berat, probabilitas mutasi, dan probabilitas *crossover*.

3.5 Perancangan Sistem

Perancangan sistem pada penelitian ini dibagi menjadi 2 yaitu perancangan antar muka dan perancangan *database*.

3.5.1 Perancangan Antar Muka

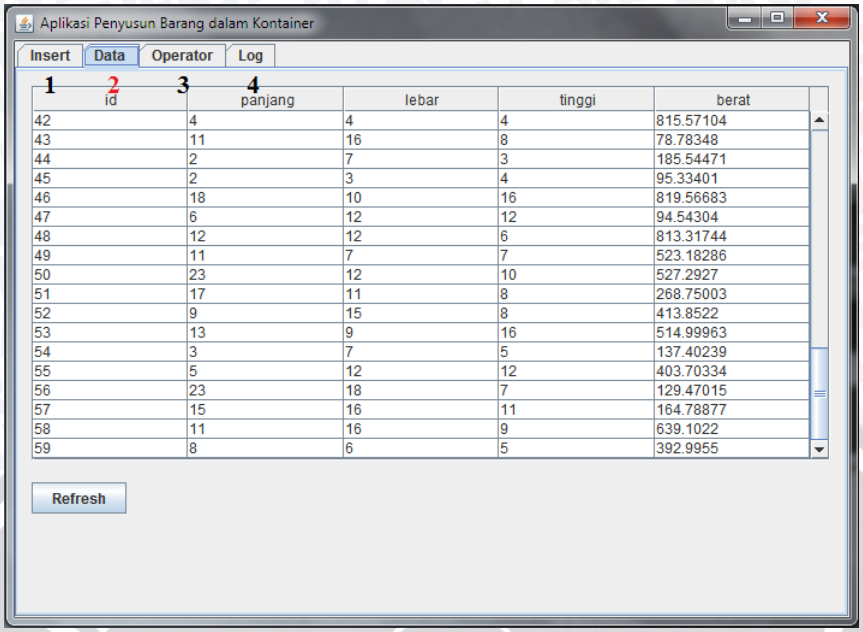
Pada *tab* 1 pada sistem berfungsi untuk memasukkan data ke dalam *database*. Hal ini dapat dilihat pada Gambar 3.21.



Gambar 3.21 Antar muka pengisian *database*

Pada *tab* 1 terdapat kolom ID untuk mengisi kode barang yang akan dimasukkan, kolom Length untuk memasukkan panjang barang, kolom Width untuk memasukkan lebar barang, kolom Height untuk memasukkan tinggi barang, dan kolom Weight untuk memasukkan berat barang yang akan disusun pada kontainer.

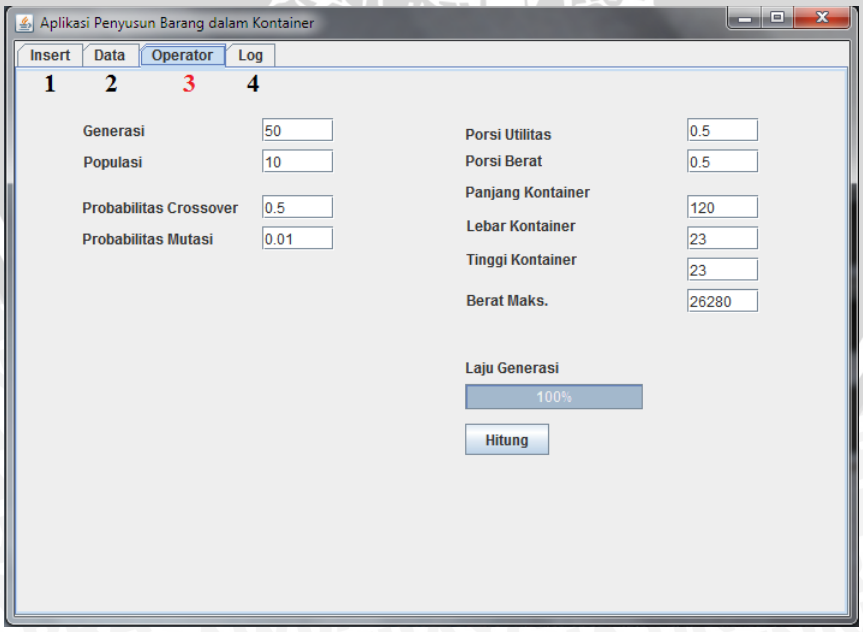
Pada *tab* 2 berfungsi untuk menampilkan daftar barang yang telah dimasukkan ke dalam *database*. Untuk ilustrasinya dapat dilihat pada Gambar 3.22.



Gambar 3.22 Antar muka daftar barang

Pada antar muka di *tab* 2 berfungsi untuk menampilkan daftar barang yang akan disusun pada kontainer. Terdapat kolom id, panjang, lebar, tinggi, dan berat masing-masing barang. Tombol "Refresh" berfungsi untuk menampilkan ulang daftar barang jika terjadi penambahan daftar barang.

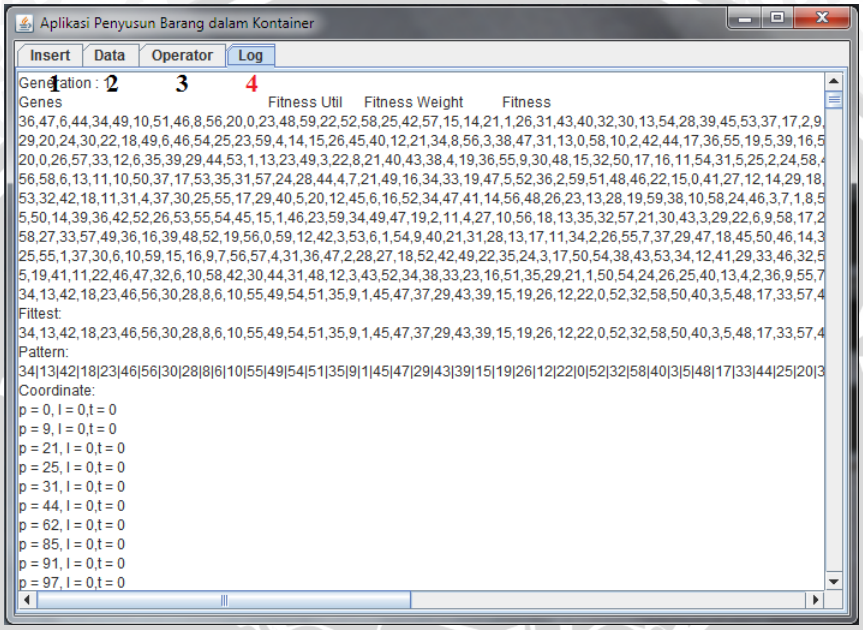
Di *tab* 3 berfungsi untuk memasukkan operator Algoritma Genetik, porsi *fitness*, dimensi kontainer, dan maksimum berat.



Gambar 3.23 Antar muka operator Algoritma Genetik

Pada *tab* 3 berfungsi untuk memasukkan variabel yang diperlukan untuk operasi Algoritma Genetik mulai dari Generasi yang digunakan hingga Berat Maksimum yang diijinkan.

Tab 4 berfungsi untuk menampilkan hasil perhitungan dengan Algoritma Genetik. Untuk ilustrasinya dapat dilihat pada Gambar 3.24.

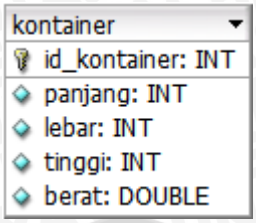


Gambar 3.24 Antar muka logging Algoritma Genetik

Pada *tab* 4 menampilkan individu apa saja yang terbentuk dengan direpresentasikan sebagai susunan kromosom. Kemudian *fitness* utilitas, *fitness* berat, dan *fitness* keseluruhan. Kemudian ditampilkan pula cara penyusunan barang pada individu terbaik.

3.5.2 Perancangan *Database*

Database pada sistem ini menggunakan 1 tabel yaitu tabel kontainer. Untuk jumlah dan macam *field*-nya dapat dilihat pada Gambar 3.25.



Gambar 3.25 Tabel Kontainer

Pada kolom id_kontainer berfungsi untuk menyimpan data nomor identitas dari masing-masing barang. Kolom panjang berfungsi untuk menyimpan data dimensi panjang dari masing-masing barang. Kolom lebar berfungsi untuk menyimpan data dimensi lebar dari masing-masing barang. Kolom tinggi berfungsi untuk menyimpan data dimensi tinggi dari masing-masing barang. Kolom berat berfungsi untuk menyimpan data berat dari masing-masing barang.

BAB IV
IMPLEMENTASI

4.1 Lingkungan Implementasi

4.1.1 Lingkungan Implementasi Perangkat Keras

Pada penelitian ini menggunakan perangkat keras sebagai berikut.

- Processor Intel i3 4 Core @ 3.30 GHz
- RAM 4 GB
- Hard Drive Seagate 300 GB
- Keyboard Logitech
- Monitor Skyview
- VGA NVIDIA GeForce Zotac 430 1 GB
- Motherboard Asus P8H61-M LX R2.0
- DVD-RW Samsung
- Casing Power Up
- Power Supply Sturdy 500 W

4.1.2 Lingkungan Implementasi Perangkat Lunak

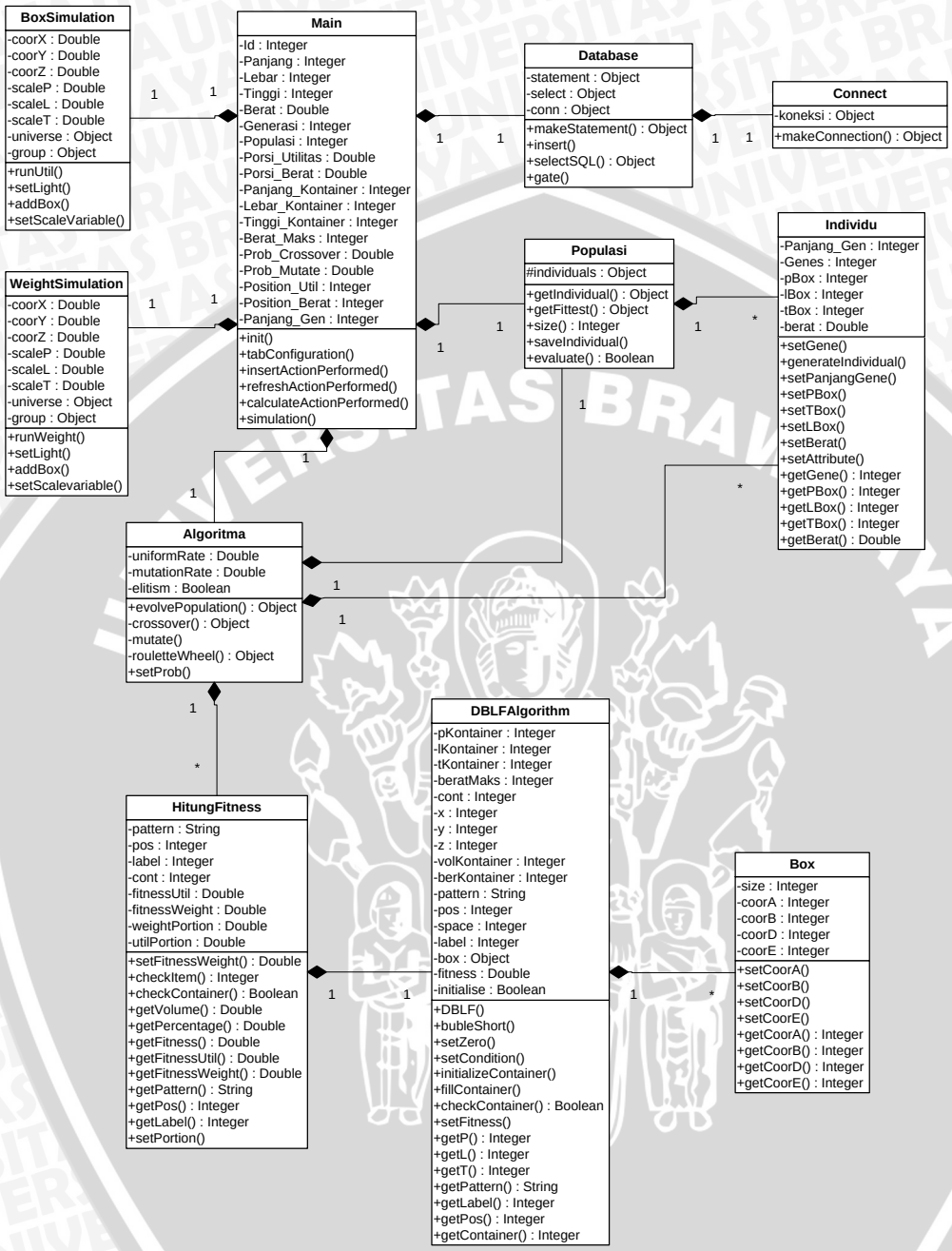
Pada penelitian ini menggunakan perangkat lunak sebagai berikut.

- Windows 7 64-bit
- NetBeans IDE 7.3.1
- JDK 7 Update 25
- Library MySQL JDBC Driver
- Library J3D 1.5.2
- Library rs2xml

4.2 Implementasi Program

4.2.1 Hubungan Antar Kelas

Pada sistem ini menggunakan *class* dan hubungan seperti pada Gambar 4.1.



Gambar 4.1 *Class* diagram sistem

Untuk proses memasukkan data ke dalam *database* menggunakan 3 kelas yaitu kelas Main, Database, dan Connect. Pada *class* Main berfungsi untuk mengirimkan data yang akan disimpan pada *database. Class* Connect berfungsi

untuk membuka koneksi *database*. Sedangkan pada *class* Database berfungsi untuk mengeksekusi *query* yang dibutuhkan pada sistem.

Penginisialisasian populasi menggunakan *class* Main, Populasi, dan Individu. Pada *class* Main berfungsi untuk mengirimkan jumlah individu yang diinginkan pada tiap generasi. Kemudian pada *class* Populasi berfungsi untuk menyimpan jumlah individu yang diinginkan serta memicu penginisialisasian tiap individu. Pada *class* Individu berfungsi untuk pengacakan awal kromosom individu.

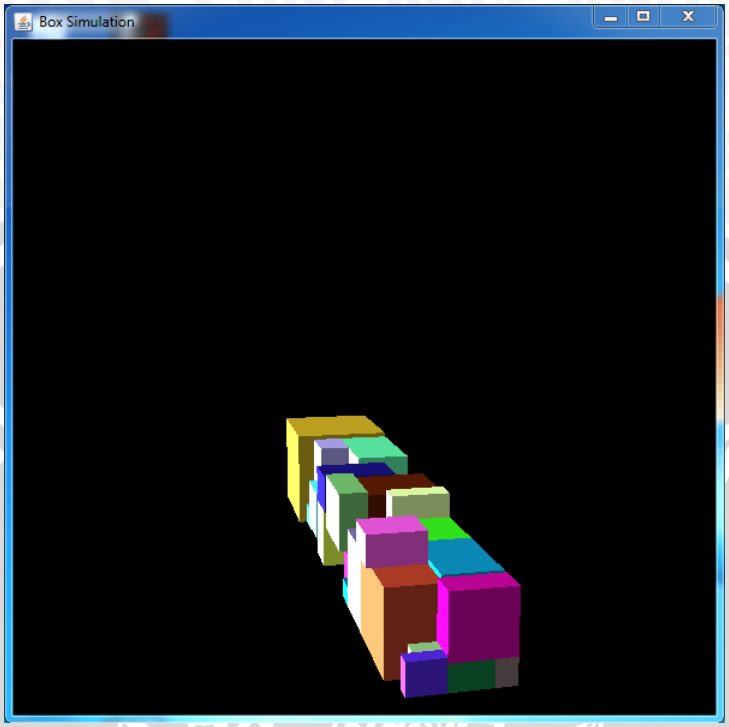
Pada proses Algoritma Genetik menggunakan *class* Main, Algoritma, Populasi, Individu, HitungFitness, DBLFAlgorithm, Box. Pada *class* Main bertugas untuk menentukan banyaknya generasi yang dilakukan sesuai dengan masukan yang dilakukan oleh *user*. Pada *class* Algoritma berfungsi untuk melakukan operasi Algoritma Genetik. Dan *class* HitungFitness berfungsi untuk memicu penghitungan *fitness* utilitas dengan algoritma DBLF dan melakukan penghitungan fitness berat. Pada *class* DBLFAlgorithm berfungsi untuk melakukan penghitungan *fitness* utilitas dan data dimensi barang dan dimensi posisi yang tersedia pada tiap barang disimpan pada *class* Box.

Pada proses simulasi *class* yang digunakan adalah Main, WeightSimulation, dan BoxSimulation. Pada *class* Main berfungsi untuk mengirimkan barang apa saja yang disusun serta posisi yang dimaksud. *Class* WeightSimulation berfungsi untuk menunjukkan posisi barang berdasarkan peletakkan berat. Sedangkan pada *class* BoxSimulation berfungsi untuk menunjukkan posisi barang saja.

4.2.2 Simulasi 3D

Untuk mempertajam analisa pada pengujian penelitian maka diperlukan sebuah simulasi sebagai hasil penghitungan tiap algoritma. Pada simulasi ini menggunakan *library* j3d versi 1.5.2 untuk pengaplikasiannya. Konsep yang diusung oleh j3d berbeda dengan pemrograman grafis pada umumnya. Jika pemrograman grafis pada umumnya menggunakan konsep terstruktur sedangkan pada j3d menggunakan konsep *inheritance* yang hampir sama dengan konsep Java.

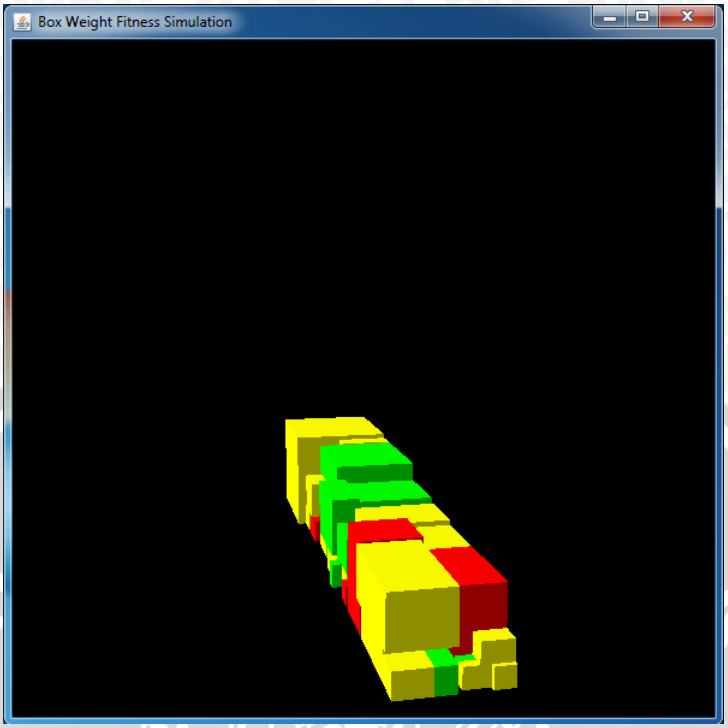
Pada proses simulasi penyusunan barang, pemilihan nilai warnanya. Dimensi barang yang dimasukkan sebagai *input* adalah panjang, lebar, tinggi.



Gambar 4.2 Contoh simulasi penyusunan barang

Berbeda pada simulasi berat terdapat masukan berupa kategori. Kategori terdiri dari 3 macam yaitu barang berbobot ringan, berbobot menengah, dan berbobot berat. Penentuan kriteria kategori berdasarkan nilai median dari data berat yang telah diurutkan secara *ascending*. Jika berat barang berada di antara nilai awal data berat hingga setengah nilai median maka barang tersebut termasuk kategori ringan. Jika berat barang berada di antara setengah nilai median hingga satu setengah nilai median maka barang tersebut termasuk dalam kategori menengah. Jika selain dari tersebut maka barang termasuk dalam kategori berat.

Dalam simulasi, barang kategori berat direpresentasikan dengan warna merah. Barang kategori menengah direpresentasikan dengan warna kuning. Dan barang kategori ringan direpresentasikan dengan warna hijau.



Gambar 4.3 Contoh simulasi berat barang

Sedangkan untuk koordinat barang-barang yang tersusun dapat dilihat pada Tabel 4.1

Tabel 4.1 Daftar lokasi barang yang tersusun

no	dimensi			berat	posisi			no	dimensi			berat	posisi		
37	26	23	23	638.5	0	0	0	50	23	12	10	527.3	96	10	8
38	6	7	10	789.9	26	0	0	14	16	6	12	179.3	55	16	7
21	11	7	15	273	32	0	0	22	4	4	5	573.8	26	13	0
10	9	16	13	281.1	43	0	0	18	6	7	5	296.9	113	14	0
19	3	5	6	490.4	52	0	0	59	8	6	5	393	60	15	0
17	5	4	7	674.7	55	0	0	1	4	5	4	510.2	43	16	0
49	11	7	7	523.2	60	0	0	8	4	4	3	217	30	13	0
13	12	17	17	351.9	71	0	0	29	6	4	5	489.1	26	17	0
53	13	9	16	515	83	0	0	6	12	12	12	719.5	83	9	6
46	18	10	16	819.6	96	0	0	24	6	3	7	498.1	71	17	0
0	4	6	8	324.1	114	0	0	52	9	15	8	413.9	34	7	7
7	7	6	3	349.8	52	5	0	12	4	4	4	552.5	48	16	0
3	8	6	3	255.9	26	7	0	27	7	3	5	718.1	34	18	0
57	15	16	11	164.8	55	0	7	32	2	3	4	244.3	116	6	0
56	23	18	7	129.5	32	0	15	55	5	12	12	403.7	26	0	10
48	12	12	6	813.3	83	9	0	42	4	4	4	815.6	77	17	0
45	2	3	4	95.33	114	6	0	35	2	8	4	217.7	52	18	0
28	6	11	7	800.5	34	7	0	5	3	7	6	425.2	40	7	0
51	17	11	8	268.8	96	10	0	40	1	4	4	348.5	118	0	0
33	6	8	5	530.3	60	7	0	26	11	4	7	215.3	71	17	7
39	5	6	4	678.4	66	7	0	44	2	7	3	185.5	58	11	0
30	6	4	5	214.9	113	10	0	9	2	4	4	806.6	40	14	0
20	7	6	5	477.9	52	11	0	54	3	7	5	137.4	68	15	0

4.2.3 Struktur Data

Penyimpanan data pada sistem ini menggunakan *database*. Akan tetapi saat program berjalan *class* yang berfungsi untuk menyimpan data yang telah diolah adalah *class* Individu. Pada *class* Individu terdapat *array* Genes yang bertipe Integer yang berfungsi untuk menyimpan urutan peletakkan barang. *Array* pBox, lBox, tBox yang bertipe Integer berfungsi untuk menyimpan dimensi barang. Jika terjadi mutasi pada individu maka nilai dari *array* pBox dan lBox akan bertukar. *Array* berat yang bertipe double berfungsi untuk menyimpan data berat dari masing-masing barang.

4.3 Kendala Implementasi Program

Dalam pengimplementasian program penulis mengalami kendala dan pemecahan masalah sebagai berikut.

1. Penentuan *fitness* berat terlalu ketat. Jika melanggar lebih dari 1 barang maka nilai *fitness*-nya jauh berkurang.
   Solusi:
   Mengubah nilai *fitness* agar tidak terjadi jarak terlalu jauh jika terjadi pelanggaran
2. Pengimplementasian program memerlukan spesifikasi perangkat keras yang cukup tinggi.
   Solusi:
   Meningkatkan perangkat keras dengan mengganti *processor* dual core dengan *processor* i3
3. Eksekusi program memerlukan waktu lama (antara 4-6 jam) sehingga cukup sulit untuk memangkas waktu dalam pengujian.
   Solusi:
   Menggunakan 3 buah komputer Laboratorium Komputer Dasar untuk melakukan pengujian.

BAB V
PENGUJIAN DAN ANALISIS

Dalam pengujian kali ini menggunakan data uji yang terdapat pada penelitian [KAN-12] yang telah diadaptasi dengan sistem dan ditambah dengan data berat hasil pengacakan pada tiap barang. Untuk daftar data uji dapat dilihat pada Tabel 5.1.

Tabel 5.1 Data uji

ID	Panjang	Lebar	Tinggi	Berat
0	4	6	8	324.06158
1	4	5	4	510.2276
2	8	8	17	569.175
3	8	6	3	255.90749
4	15	8	23	542.9703
5	3	7	6	425.1569
6	12	12	12	719.5298
7	7	6	3	349.79483
8	4	4	3	216.95891
9	2	4	4	806.628
10	9	16	13	281.07104
11	12	6	7	820.076
12	4	4	4	552.48376
13	12	17	17	351.941
14	16	6	12	179.27766
15	12	10	11	1.3731356
16	9	9	23	431.6416
17	5	4	7	674.7005
18	6	7	5	296.90503
19	3	5	6	490.3626
20	7	6	5	477.8786
21	11	7	15	272.95474
22	4	4	5	573.83716
23	13	12	17	457.49026
24	6	3	7	498.0543
25	9	11	6	562.3798
26	11	4	7	215.32184
27	7	3	5	718.0554
28	6	11	7	800.5403
29	6	4	5	489.10648
30	6	4	5	214.87929

31	9	15	7	696.2703
32	2	3	4	244.34961
33	6	8	5	530.31976
34	9	12	11	517.5793
35	2	8	4	217.65382
36	13	12	6	637.33295
37	26	23	23	638.4907
38	6	7	10	789.9341
39	5	6	4	678.39606
40	1	4	4	348.5059
41	6	11	8	182.22412
42	4	4	4	815.57104
43	11	16	8	78.78348
44	2	7	3	185.54471
45	2	3	4	95.33401
46	18	10	16	819.56683
47	6	12	12	94.54304
48	12	12	6	813.31744
49	11	7	7	523.18286
50	23	12	10	527.2927
51	17	11	8	268.75003
52	9	15	8	413.8522
53	13	9	16	514.99963
54	3	7	5	137.40239
55	5	12	12	403.70334
56	23	18	7	129.47015
57	15	16	11	164.78877
58	11	16	9	639.1022
59	8	6	5	392.9955

Pengujian dilakukan dengan memberikan nilai yang bervariatif pada Generasi, Populasi, Probabilitas Mutasi, dan Probabilitas *Crossover*. Bertujuan untuk mengetahui apakah variabel tersebut berpengaruh pada nilai *fitness* masing-masing kandidat solusi.

Untuk memperoleh hasil yang diinginkan, menurut [KOU-06] diperlukan nilai populasi yang tepat agar waktu yang digunakan untuk pengeksekusian menjadi relatif singkat tetapi tetap menghasilkan nilai *fitness* yang optimal.

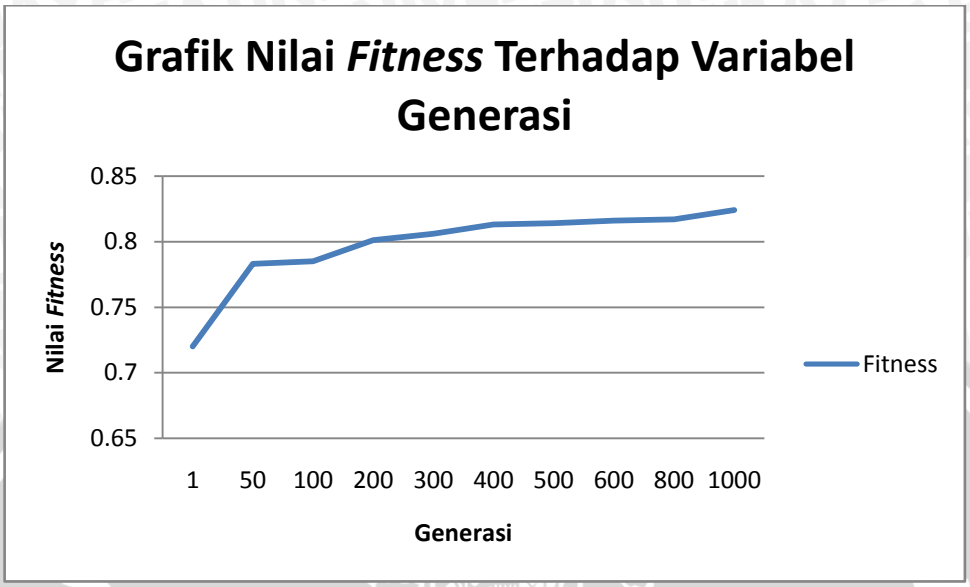
Pada penelitian ini pengujian dilakukan tidak hanya pada populasi saja melainkan pada variabel generasi juga. Pertama pengujian dilakukan dengan

memberikan nilai variatif pada variabel Generasi. Dengan diberikan nilai Populasi 50, Probabilitas Mutasi 0.002, dan Probabilitas *Crossover* 0.3 maka dihasilkan seperti pada Tabel 5.2.

Tabel 5.2 Hasil percobaan variabel Generasi

Perc.	Data Fitness	Generasi									
		1	50	100	200	300	400	500	600	800	1000
1	Utilitas	0.61	0.59	0.6	0.64	0.69	0.66	0.52	0.67	0.67	0.61
	Berat	0.84	0.9	0.94	0.96	0.92	0.98	0.96	0.98	0.98	0.96
	Keseluruhan	0.72	0.75	0.77	0.8	0.8	0.82	0.74	0.83	0.82	0.79
2	Utilitas	0.62	0.63	0.64	0.69	0.67	0.72	0.65	0.6	0.65	0.73
	Berat	0.85	0.96	0.94	0.94	0.96	1	0.98	0.94	0.94	0.94
	Keseluruhan	0.74	0.8	0.79	0.81	0.81	0.86	0.81	0.77	0.79	0.84
3	Utilitas	0.61	0.66	0.65	0.67	0.69	0.7	0.66	0.64	0.66	0.7
	Berat	0.87	0.96	0.94	0.94	0.92	0.94	0.98	0.98	1	0.98
	Keseluruhan	0.74	0.81	0.79	0.8	0.8	0.82	0.82	0.81	0.83	0.84
4	Utilitas	0.55	0.65	0.56	0.73	0.69	0.68	0.68	0.69	0.71	0.68
	Berat	0.9	0.92	0.96	0.94	0.96	0.98	1	0.96	0.96	0.98
	Keseluruhan	0.73	0.78	0.76	0.83	0.82	0.83	0.84	0.82	0.83	0.83
5	Utilitas	0.49	0.62	0.66	0.62	0.69	0.58	0.69	0.72	0.68	0.67
	Berat	0.9	0.96	0.94	1	0.94	0.96	0.92	0.92	0.98	0.96
	Keseluruhan	0.7	0.79	0.8	0.81	0.82	0.77	0.8	0.82	0.83	0.81
6	Utilitas	0.68	0.69	0.68	0.57	0.55	0.58	0.69	0.69	0.7	0.6
	Berat	0.78	0.92	0.92	0.96	0.98	0.94	1	0.96	1	1
	Keseluruhan	0.73	0.8	0.8	0.76	0.76	0.76	0.84	0.82	0.85	0.83
7	Utilitas	0.63	0.61	0.61	0.66	0.71	0.66	0.69	0.7	0.68	0.65
	Berat	0.79	0.98	0.95	0.96	0.96	0.98	0.96	0.94	0.96	0.98
	Keseluruhan	0.71	0.79	0.78	0.81	0.83	0.82	0.83	0.82	0.82	0.81
8	Utilitas	0.56	0.56	0.62	0.68	0.67	0.65	0.68	0.71	0.69	0.69
	Berat	0.88	0.94	0.89	0.96	0.98	0.98	0.96	0.94	0.94	0.96
	Keseluruhan	0.72	0.75	0.76	0.82	0.82	0.81	0.82	0.82	0.81	0.82
9	Utilitas	0.52	0.67	0.66	0.56	0.6	0.7	0.65	0.68	0.66	0.68
	Berat	0.9	0.86	0.96	0.96	0.96	0.94	0.98	0.96	0.98	1
	Keseluruhan	0.71	0.77	0.81	0.76	0.78	0.82	0.81	0.82	0.82	0.84
10	Utilitas	0.53	0.62	0.64	0.66	0.67	0.66	0.66	0.68	0.57	0.7
	Berat	0.86	0.96	0.94	0.96	0.98	0.98	1	0.98	0.98	0.96
	Keseluruhan	0.7	0.79	0.79	0.81	0.82	0.82	0.83	0.83	0.77	0.83
Fitness Minimal		0.7	0.75	0.76	0.76	0.76	0.76	0.74	0.77	0.77	0.79
Fitness Maksimal		0.74	0.81	0.81	0.83	0.83	0.86	0.84	0.83	0.85	0.84
Rata-rata		0.72	0.783	0.785	0.801	0.806	0.813	0.814	0.816	0.817	0.824

Pada percobaan terhadap variabel Generasi terdapat 10 buah nilai yang diujikan. Yaitu 1, 50, 100, 200, 300, 400, 500, 600, 800, dan 1000.



Gambar 5.1 Grafik nilai *fitness* terhadap variabel Generasi

Disimpulkan bahwa jumlah Generasi mempengaruhi nilai *fitness* dan Generasi yang bernilai 600 adalah nilai ambang konstan.

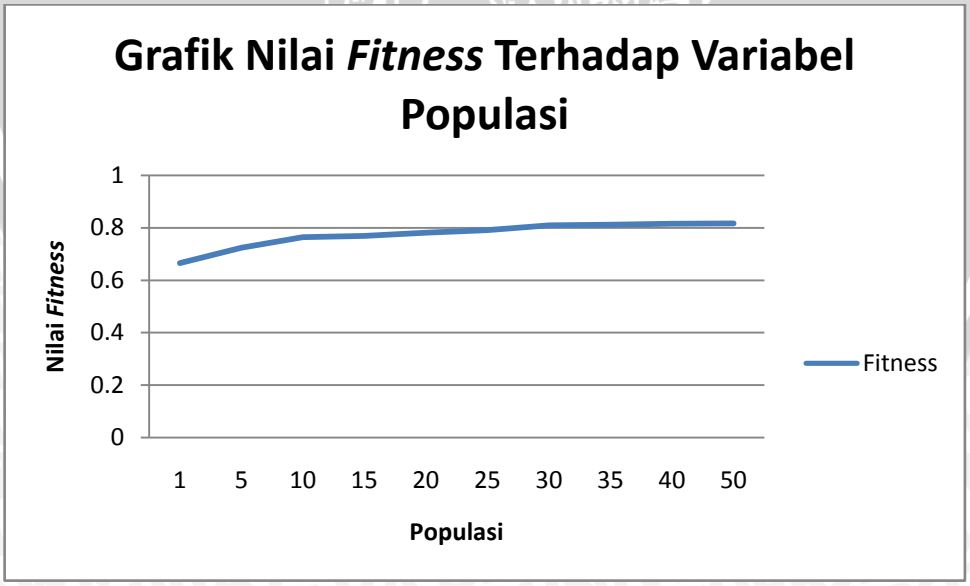
Pada pengujian variabel Populasi diberikan nilai jumlah Generasi 600, Probabilitas Mutasi 0.002, dan Probabilitas *Crossover* 0.3. Hasilnya dapat dilihat pada Tabel 5.3.

Tabel 5.3 Hasil percobaan variabel Populasi

Perc.	Data Fitness	Populasi									
		1	5	10	15	20	25	30	35	40	50
1	Utilitas	0.49	0.54	0.59	0.57	0.71	0.65	0.68	0.59	0.68	0.67
	Berat	0.76	0.91	0.94	0.9	0.92	0.94	0.96	0.91	0.96	0.98
	Keseluruhan	0.62	0.72	0.76	0.74	0.81	0.79	0.82	0.75	0.82	0.83
2	Utilitas	0.48	0.51	0.52	0.52	0.61	0.7	0.67	0.7	0.72	0.6
	Berat	0.86	0.9	0.96	1	0.92	0.98	0.98	0.98	0.94	0.94
	Keseluruhan	0.67	0.71	0.74	0.76	0.77	0.84	0.83	0.84	0.83	0.77
3	Utilitas	0.54	0.51	0.55	0.53	0.53	0.53	0.66	0.67	0.66	0.64
	Berat	0.78	0.96	0.96	0.94	0.98	0.98	1	0.98	0.98	0.98
	Keseluruhan	0.66	0.74	0.76	0.74	0.75	0.75	0.83	0.83	0.82	0.81
4	Utilitas	0.45	0.5	0.64	0.68	0.69	0.58	0.55	0.67	0.68	0.69
	Berat	0.84	0.94	0.94	0.96	0.89	0.98	0.98	1	0.98	0.96
	Keseluruhan	0.64	0.72	0.79	0.82	0.79	0.78	0.76	0.84	0.83	0.82
5	Utilitas	0.5	0.52	0.55	0.53	0.55	0.65	0.71	0.72	0.66	0.72

	Berat	0.69	0.92	0.98	0.94	0.98	0.96	0.98	0.96	0.96	0.92
	Keseluruhan	0.59	0.72	0.76	0.74	0.76	0.8	0.84	0.84	0.81	0.82
6	Utilitas	0.54	0.63	0.68	0.61	0.57	0.55	0.56	0.62	0.66	0.69
	Berat	0.73	0.82	0.94	0.98	0.98	0.98	0.98	0.96	1	0.96
	Keseluruhan	0.64	0.72	0.81	0.79	0.78	0.76	0.77	0.79	0.82	0.82
7	Utilitas	0.5	0.51	0.52	0.66	0.54	0.58	0.67	0.65	0.66	0.7
	Berat	0.83	0.96	0.94	0.96	0.94	0.98	0.98	0.98	0.96	0.94
	Keseluruhan	0.66	0.73	0.73	0.81	0.74	0.78	0.83	0.82	0.81	0.82
8	Utilitas	0.5	0.53	0.67	0.55	0.67	0.66	0.66	0.68	0.68	0.71
	Berat	0.84	0.96	0.92	0.94	0.98	1	0.96	0.94	0.96	0.94
	Keseluruhan	0.67	0.75	0.8	0.75	0.83	0.83	0.81	0.81	0.82	0.82
9	Utilitas	0.47	0.52	0.59	0.61	0.66	0.68	0.72	0.71	0.66	0.68
	Berat	0.78	0.92	0.96	0.96	0.96	0.94	0.94	0.94	0.98	0.96
	Keseluruhan	0.83	0.72	0.77	0.79	0.81	0.81	0.83	0.82	0.82	0.82
10	Utilitas	0.52	0.52	0.52	0.55	0.57	0.58	0.56	0.56	0.54	0.68
	Berat	0.81	0.9	0.92	0.94	0.98	0.94	0.98	0.98	1	0.98
	Keseluruhan	0.67	0.71	0.72	0.75	0.77	0.76	0.77	0.77	0.77	0.83
Fitness Minimal		0.59	0.71	0.72	0.74	0.74	0.75	0.76	0.75	0.77	0.77
Fitness Maksimal		0.83	0.75	0.81	0.82	0.83	0.84	0.84	0.84	0.83	0.83
Rata-rata		0.665	0.724	0.764	0.769	0.781	0.79	0.809	0.811	0.815	0.816

Pada percobaan terhadap variabel Populasi terdapat 10 buah nilai yang diujikan. Yaitu 1, 5, 10, 15, 20, 25, 30, 35, 40, dan 50.



Gambar 5.2 Grafik nilai *fitness* terhadap variabel Populasi

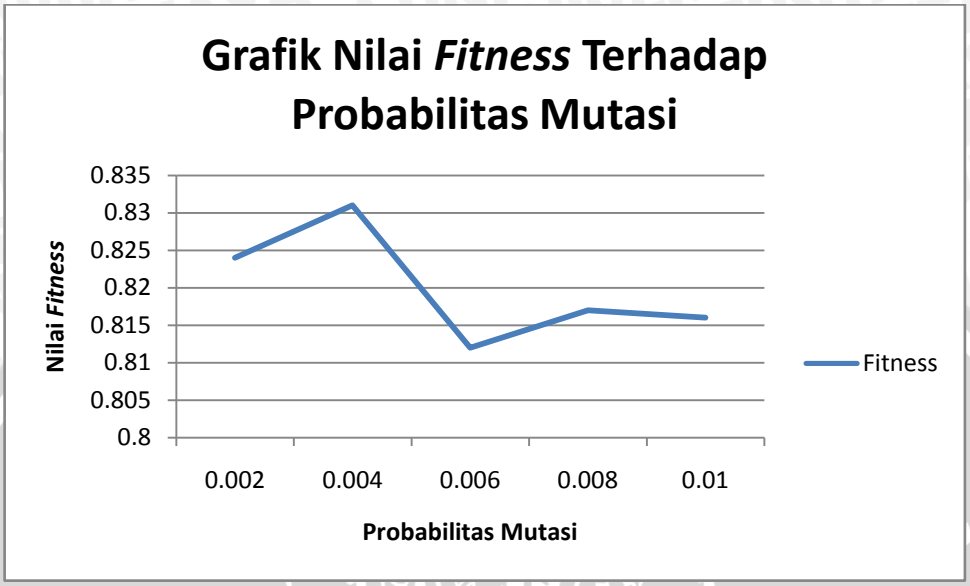
Disimpulkan bahwa jumlah Populasi mempengaruhi nilai *fitness* dan Populasi bernilai 25 adalah nilai ambang konstan.

Pengujian terhadap variabel Probabilitas Mutasi sejalan dengan pengujian yang dilakukan oleh [HAN-10]. Diberikan nilai jumlah Generasi 600, jumlah Populasi 25, dan Probabilitas *Crossover* 0.3. Hasilnya dapat dilihat pada Tabel 5.4.

Tabel 5.4 Hasil percobaan variabel Probabilitas Mutasi

Perc.	Data Fitness	Probabilitas Mutasi				
		0.002	0.004	0.006	0.008	0.01
1	Utilitas	0.61	0.66	0.71	0.68	0.67
	Berat	0.96	1	0.96	0.98	0.96
	Keseluruhan	0.79	0.83	0.83	0.83	0.81
2	Utilitas	0.73	0.73	0.67	0.67	0.54
	Berat	0.94	0.96	0.96	0.98	0.98
	Keseluruhan	0.84	0.85	0.81	0.83	0.76
3	Utilitas	0.7	0.67	0.66	0.63	0.7
	Berat	0.98	0.98	1	1	0.98
	Keseluruhan	0.84	0.82	0.83	0.82	0.84
4	Utilitas	0.68	0.68	0.66	0.7	0.64
	Berat	0.98	0.98	0.98	0.96	0.98
	Keseluruhan	0.83	0.83	0.82	0.83	0.81
5	Utilitas	0.67	0.68	0.59	0.71	0.69
	Berat	0.96	0.98	0.98	0.94	0.96
	Keseluruhan	0.81	0.83	0.78	0.83	0.82
6	Utilitas	0.66	0.66	0.57	0.67	0.71
	Berat	1	0.96	0.98	0.96	0.92
	Keseluruhan	0.83	0.81	0.77	0.81	0.81
7	Utilitas	0.65	0.66	0.68	0.64	0.69
	Berat	0.98	1	0.96	0.98	0.98
	Keseluruhan	0.81	0.83	0.82	0.81	0.84
8	Utilitas	0.69	0.72	0.67	0.55	0.66
	Berat	0.96	0.96	0.96	0.96	0.98
	Keseluruhan	0.82	0.84	0.82	0.75	0.82
9	Utilitas	0.68	0.7	0.68	0.66	0.69
	Berat	1	0.98	0.96	1	0.98
	Keseluruhan	0.84	0.84	0.82	0.83	0.83
10	Utilitas	0.7	0.65	0.69	0.69	0.67
	Berat	0.96	1	0.96	0.96	0.96
	Keseluruhan	0.83	0.83	0.82	0.83	0.82
Fitness Minimal		0.79	0.81	0.77	0.75	0.76
Fitness Maksimal		0.84	0.85	0.83	0.83	0.84
Rata-rata		0.824	0.831	0.812	0.817	0.816

Pada percobaan terhadap variabel Probabilitas Mutasi terdapat 5 buah nilai yang diujikan. Yaitu 0.002, 0.004, 0.006, 0.008, dan 0.01.



Gambar 5.3 Grafik nilai *fitness* terhadap variabel Probabilitas Mutasi

Disimpulkan bahwa pemilihan Probabilitas Mutasi yang tepat mempengaruhi nilai *fitness* dan Probabilitas Mutasi bernilai 0.004 adalah yang terbaik. Sebab memiliki nilai rata-rata yang tertinggi.

Pengujian terhadap variabel Probabilitas *Crossover* diberikan nilai jumlah Generasi 600, jumlah Populasi 25, dan Probabilitas Mutasi 0.004. Hasilnya dapat dilihat pada Tabel 5.5.

Tabel 5.5 Hasil percobaan variabel Probabilitas *Crossover*

Perc.	Data Fitness & Generasi	Probabilitas Crossover				
		0.3	0.4	0.6	0.7	0.9
1	Utilitas	0.66	0.69	0.66	0.62	0.59
	Berat	1	0.98	1	0.94	0.94
	Keseluruhan	0.83	0.83	0.83	0.78	0.77
2	Utilitas	0.73	0.67	0.66	0.71	0.72
	Berat	0.96	0.98	0.96	0.96	0.96
	Keseluruhan	0.85	0.82	0.81	0.83	0.84
3	Utilitas	0.67	0.71	0.53	0.66	0.59
	Berat	0.98	0.94	0.98	0.98	0.98
	Keseluruhan	0.83	0.82	0.75	0.82	0.78
4	Utilitas	0.68	0.67	0.69	0.61	0.69

	Berat	0.98	0.98	0.96	0.96	0.98
	Keseluruhan	0.83	0.82	0.82	0.79	0.83
5	Utilitas	0.68	0.69	0.71	0.66	0.64
	Berat	0.98	0.94	0.96	0.96	0.96
	Keseluruhan	0.83	0.81	0.83	0.81	0.8
6	Utilitas	0.66	0.67	0.71	0.68	0.68
	Berat	0.96	0.98	0.98	0.98	1
	Keseluruhan	0.81	0.82	0.84	0.83	0.84
7	Utilitas	0.66	0.68	0.66	0.67	0.7
	Berat	1	0.98	0.98	0.96	0.98
	Keseluruhan	0.83	0.83	0.82	0.82	0.84
8	Utilitas	0.72	0.69	0.69	0.63	0.69
	Berat	0.96	0.96	0.96	1	0.98
	Keseluruhan	0.84	0.82	0.82	0.81	0.84
9	Utilitas	0.7	0.68	0.69	0.69	0.73
	Berat	0.98	0.94	0.98	0.94	0.94
	Keseluruhan	0.84	0.81	0.83	0.81	0.83
10	Utilitas	0.65	0.64	0.67	0.66	0.7
	Berat	1	0.98	0.98	0.98	0.98
	Keseluruhan	0.83	0.81	0.83	0.82	0.84
Fitness Minimal		0.81	0.81	0.75	0.78	0.77
Fitness Maksimal		0.85	0.83	0.84	0.83	0.84
Rata-rata		0.832	0.819	0.818	0.812	0.821
Varian		0.000096	0.000049	0.000576	0.000236	0.000669
Standar Deviasi		0.0098	0.0070	0.0240	0.0154	0.0259

Pada percobaan terhadap variabel Probabilitas *Crossover* menghasilkan rata-rata dan *fitness* maksimal yang berbeda tidak terlalu jauh. Oleh sebab itu untuk membedakan tiap hasil digunakan rumus Standar Deviasi yang dapat dilihat pada persamaan (5-1).

$$\sigma = \sqrt[2]{\frac{\sum_{i=1}^{i=n}(\bar{s}-s_i)^2}{n}} \quad (5\text{-}1)$$

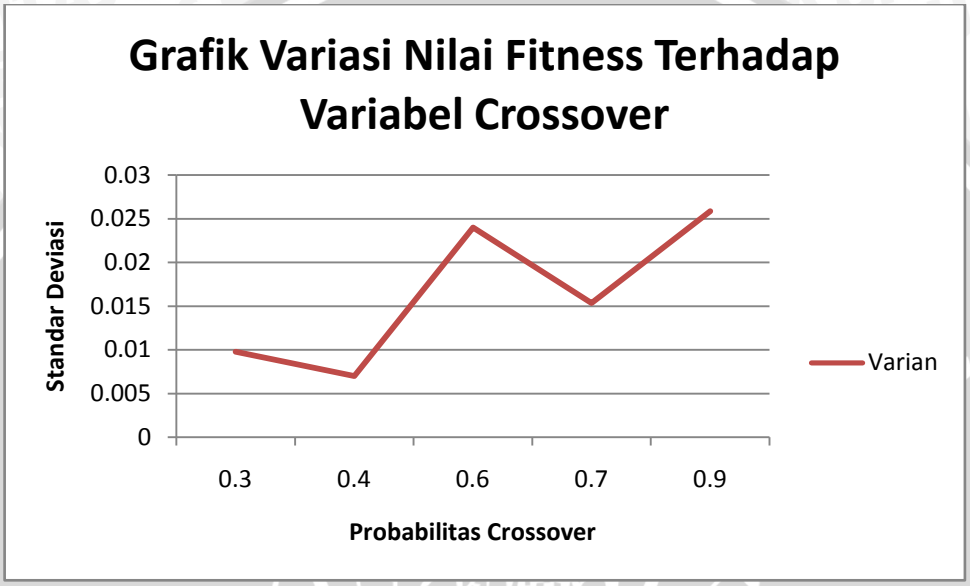
σ = Standar Deviasi

$\bar{s}$ = Rata-rata data

s_i = Data ke-i

n = Banyak data

Untuk memilih hasil yang terbaik, maka dipilih nilai Standar Deviasi yang paling mendekati 0 dari masing-masing variabel Probabilitas *Crossover*. Sebab dengan nilai Standar Deviasi yang mendekati 0 maka kecil kemungkinan bagi variabel tersebut jika dieksekusi memberikan penilaian yang timpang antar tiap eksekusi. Dan ditentukan bahwa Probabilitas *Crossover* yang bernilai 0.4 yang terbaik.



Gambar 5.4 Grafik nilai *fitness* terhadap variabel Probabilitas *Crossover*

Disimpulkan bahwa pemilihan nilai Probabilitas Crossover yang tepat mempengaruhi nilai *fitness*.

BAB VI
PENUTUP

Berdasarkan hasil percobaan pada bab sebelumnya maka dapat disimpulkan sebagai berikut.

1. Perancangan kromosom yang tepat untuk penelitian ini adalah dengan menggunakan metode *ordered chromosome*. Untuk panjang kromosom ditentukan oleh banyaknya barang yang akan disusun pada kontainer. Pada kasus ini panjang yang digunakan adalah 60 kromosom. Untuk operator genetik menggunakan *Roulette Wheel*, *Position-Based Crossover*, dan *Swap Mutation*.
2. Perancangan *fitness* dibagi menjadi dua yaitu *fitness* utilitas dan *fitness* berat. Untuk *fitness* utilitas menggunakan algoritma DBLF dalam penyelesaiannya. Sedang *fitness* berat menggunakan metode konvensional dalam penyelesaiannya.
3. Perancangan evaluasi *fitness* menggunakan variasi nilai Generasi, Populasi, Probabilitas Mutasi, dan Probabilitas *Crossover*. Disimpulkan bahwa Generasi 600, populasi 25, Probabilitas Mutasi 0.004, dan Probabilitas *Crossover* 0.004 adalah solusi yang terbaik pada penelitian ini. Standar deviasi sebagai penentu akhir. Jika semakin kecil standar deviasinya maka nilai dari operator tersebut yang dipakai sebab terkait dengan stabilnya variabel tersebut untuk memberikan solusi.

Saran yang dapat diambil dari penelitian ini adalah sebagai berikut.

1. Perancangan fitness berat sebaiknya diubah dengan melihat titik berat. Hal ini dimaksudkan agar penentuan fitness berat dapat lebih valid sehingga solusi yang terbentuk dapat lebih dipercaya
2. Untuk melakukan rotasi pada barang sebaiknya tidak hanya di operator mutasi. Sebab mutasi kejadiannya sangat kecil. Bahkan pada tiap generasi bisa jadi tidak terjadi mutasi. Perlu perancangan yang lebih mendalam saat melakukan rotasi barang.

DAFTAR PUSTAKA

[BOJ-09] Bojang Jnr, Sheriff. 2009, "Le Joola – seven years since Africa's worst maritime disaster", http://www.rnw.nl/africa/article/le-joola-seven-years-africa%E2%80%99s-worst-maritime-disaster [27 Oktober 2013]

[MAR-06] Martono, M. 2006, Simulasi 3D Container Problem Loading Problem Menggunakan Algoritma Genetik, Teknik Informatika Bina Nusantara, Jakarta.

[HAN-10] Handayani, Sri Rejeki. 2010, Penyusunan Barang Pada Kontainer Dengan Algoritma Genetik, Jurusan Ilmu Komputer Universitas Brawijaya, Malang.

[IGN-00] Ignatius Purnomo, Oskar. 2000, Algoritma Genetik Pada Pencarian Rute Terpendek Dalam Masalah Travelling Salesman Problem (TSP), Program Studi Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Gadjah Mada, Yogyakarta.

[KAN-12] Kang, K., Moon, I. dan Wang, H. 2012, "A Hybrid Genetic Algorithm with a New Packing Strategy for The Three-Dimensional Bin Packing Problem", Applied Mathematics And Computation, Vol. 219, Issue 3, hal. 1287-1299

[KAR-04] Karabulut, K. dan Inceoglu, M.M. 2004, "A Hybrid Genetic Algorithm for Packing in 3D with Deepest Bottom Left with Fill Method", ADVIS, Vol. 3261, hal. 441-450

[KOU-06] Koumousis, Vlasis K dan Katsaras, Christos K. 2006, "A Saw-Tooth Genetic Algorithm Combining the Effects of Variable Population Size and Reinitialization to Enhance Performance", IEEE Transactions on Evolutionary Computation, Vol. 10, hal. 19-28

[JAK-93] Jakobs, S. 1993, "Theory and Methodology On Genetic Algorithms for The Packing Polygons", European Journal of Operational Research, Vol. 88, hal. 165-181

[MAT-05] Matthew, Tom V. 2005, "GENETIC ALGORITHM", http://www.civil.iitb.ac.in/tvm/2701_dga/2701-ga-notes/gadoc/ [25 September 2013]

[CHE-12] S., Cherish. 2012, "Mass and Volume Relationship", http://www.newton.dep.anl.gov/askasci/gen01/gen01652.htm [6 Oktober 2013]