PENERAPAN ALGORITMA GENETIKA *HYBRID* PADA PERMASALAHAN *BOUNDED KNAPSACK*

SKRIPSI

Oleh

Ivan Syaikhul Hadi
NIM 111810101027

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2015

PENERAPAN ALGORITMA GENETIKA *HYBRID* PADA PERMASALAHAN *BOUNDED KNAPSACK*

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Matematika (S1)
dan mencapai gelar Sarjana Sains

oleh

Ivan Syaikhul Hadi
NIM 111810101027

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2015

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Ayahanda H. Imam Khudlori dan Ibunda Hj. Umi Zahro' tercinta yang senantiasa memberi doa, semangat, dan kasih sayang;
2. Nenek Hj. Siti Rohmah, serta adikku Erica Laili Rahmawati dan M. Abdullah Kafabihi yang senantiasa memberi semangat, doa dan kasing sayang;
3. M. Ziaul Arif, S. Si, M. Sc selaku Dosen Pembimbing Utama dan Ahmad Kamsyakawuni S. Si, M. Kom selaku Dosen Pembimbing Anggota yang telah memberikan bantuan dan bimbingan secara intensif untuk penyempurnaan skripsi ini;
4. Seluruh guru dan dosen sejak taman kanak-kanak hingga perguruan tinggi yang telah memberi banyak ilmu dan membimbing dengan tulus;
5. Almamater Jurusan Matematika FMIPA Universitas Jember, SMA Darussalam Blokagung, SMP Negeri 3 Tanggul, MI Hidayatul Mubtadiin, dan TK Roudlotul Atfal;
6. Seluruh warga KRAMAT'11, HIMATIKA "Geokompstat", dan sahabat-sahabat Diskusi Akademisi tercinta.

MOTTO

"Allah selalu ada pada setiap niat baik"

(Emha Ainun Nadjib) *)

"Suro diro joyoningrat lebur dening pangastuti"

(Ronggowarsito) **)

"Lebih baik dibenci tapi menjadi diri sendiri daripada disukai tapi menjadi orang lain"

(Kurt Cobain) ***)

---

*) Kenduri Cinta bersama Cak Nun dan Kiai Kanjeng https://youtube.com/watch?v=dIdeZriRKwg.

**) Tembang Kinanthi Karya Ronggowarsito https://jejaktapak.com/2014/03/12/asal-usul-kalimat-suro-diro-jayaningrat-lebur-dening-pangastuti/.

***) Kurt Donald Cobain (NIRVANA) https://goodreads.com/author/quotes/33041.Kurt_Cobain.

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

nama : Ivan Syaikhul Hadi

NIM : 111810101027

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "Penerapan Algoritma Genetika *Hybrid* Pada Permasalahan *Bounded* Knapsack" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan dalam institusi manapun dan juga bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, November 2015

Yang menyatakan,

Ivan Syaikhul Hadi

NIM 111810101027

SKRIPSI

PENERAPAN ALGORITMA GENETIKA *HYBRID* PADA PERMASALAHAN *BOUNDED KNAPSACK*

oleh

Ivan Syaikhul Hadi
NIM 111810101027

Pembimbing

Dosen Pembimbing Utama : M. Ziaul Arif S. Si., M. Sc

Dosen Pembimbing Anggota : Ahmad Kamsyakawuni S. Si., M. Kom

PENGESAHAN

Skripsi berjudul "Penerapan Algoritma Genetika *Hybrid* Pada Permasalahan *Bounded Knapsack*" telah diuji dan disahkan pada:

Hari, tanggal :

Tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember

Tim Penguji:

Ketua,	Sekretaris,
M. Ziaul Arif, S. Si., M. Sc	Ahmad Kamsyakawuni, S. Si., M. Kom
NIP. 198501112008121002	NIP. 197211291998021001
Anggota I,	Anggota II,
Prof. Drs. Kusno, DEA., Ph.D	Dian Anggraeni, S. Si., M. Si
NIP. 196101081986021001	NIP. 198202162006042002

Mengesahkan

Dekan,

Prof. Drs. Kusno, DEA, Ph.D.

NIP. 196101081986021001

RINGKASAN

Penerapan Algoritma Genetika *Hybrid* Pada permasalahan *Bounded Knapsack*; Ivan Syaikhul Hadi, 111810101027; 2015; 49 Halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Knapsack Problem merupakan masalah dimana seseorang dihadapkan pada persoalan optimasi pada pemilihan benda yang akan dimasukkan ke dalam sebuah wadah yang memiliki sebuah keterbatasan ruang atau daya tampung. Wadah atau tempat tersebut hanya dapat menyimpan beberapa objek dengan ketentuan total satuan objek tersebut lebih kecil atau sama dengan satuan kapasitasnya. Ada tiga macam masalah *knapsack*, salah satunya adalah *bounded knapsack*. *Bounded knapsack* adalah permasalahan pengambilan sebagian atau semua objek dari beberapa objek yang jumlahnya terbatas.

Penelitian ini mengaplikasikan algoritma genetika *hybrid* pada permasalahan *bounded knapsack*. Selain untuk mengetahui konsep dan hasil dari algoritma, penelitian ini juga untuk mengetahui peranan dari probabilitas algoritma dalam pencarian nilai optimum. Serta membandingkan solusi optimal algoritma genetika dan algoritma genetika *hybrid* dari segi hasil maupun waktu yang dibutuhkan.

Hasil penelitian menunjukkan bahwa algoritma genetika *hybrid* dapat diimplementasikan pada permasalahan *bounded knapsack* dengan baik. Penggunaan algoritma *tabu search* pada proses *crossover* memberikan hasil yang optimal yakni dapat mengurangi waktu pencarian solusi. Dari 15 pasang parameter probabilitas yang di uji dengan 10 kali *running*, 3 pasang probabilitas menunjukkan hasil *profit* maksimal yang sama yaitu Rp. 4.872.900, namun pada probabilitas *crossover* 0,9 dan probabilitas mutasi 0,05 membutuhkan waktu sedikit lebih cepat (1,3719 detik) dibandingkan dua pasang probabilitas yang lain. Sedangkan hasil perbandingan menunjukkan algoritma genetika *hybrid* dari pada algoritma genetika

karena menghasilkan *profit* lebih maksimal dan *running time* lebih cepat di banding algoritma genetika

PRAKATA

Segala puji syukur kami panjatkan ke hadirat Allah SWT yang telah melimpahkan segala berkah, rahmat, serta hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Peramalan Indeks Harga Konsumen dengan Jaringan Syaraf Tiruan dan *Adaptive Neuro-Fuzzy Inference System* (ANFIS)". Skripsi ini disusun untuk memenuhi salah satu syarat dalam menyelesaikan pendidikan strata 1 (S1) di Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penyusunan skripsi ini tidak lepas dari bantuan berbagai pihak, baik bantuan secara langsung maupun tidak langsung. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Ayahanda H. Imam Khudlori dan Ibunda Hj. Umi Zahro' tercinta yang senantiasa memberi doa, semangat, dan kasih sayang;
2. Nenek Hj. Siti Rohmah, serta adikku Erica Laili Rahmawati dan M. Abdullah Kafabihi yang senantiasa memberi semangat, doa dan kasing sayang;
3. Ibu Hj. Lilik Istiqomah selaku pengasuh PP. AL-JAUHAR.
4. M. Ziaul Arif, S. Si, M. Sc selaku Dosen Pembimbing Utama dan Ahmad Kamsyakawuni S. Si, M. Kom selaku Dosen Pembimbing Anggota yang telah meluangkan waktu, pikiran, dan perhatian dalam penulisan skripsi ini;
5. Prof. Drs. Kusno, DEA., Ph.D dan Dian Anggraeni, S. Si., M. Si selaku Dosen Penguji yang telah memberikan kritik dan saran demi kesempurnaan skripsi ini;
6. Seluruh dosen dan karyawan Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam yang telah memberikan ilmu serta nasehat selama proses perkuliahan;
7. Keluarga besar KRAMAT'11, Diskusi Akademisi (Haki, Emil, Ivan, Rian, Poty, Darul, Jefri, Rafi, Saipul, Hendri, Zulfi), dan sahabat DaBlo (Arif, Azme, Tika, Novi, Richa) yang senantiasa saling mendukung untuk menjadi yang terbaik;

8. Keluarga besar HIMATIKA "Geokompstat" yang senantiasa mengajarkan banyak hal;
9. Semua pihak yang tidak dapat disebutkan satu per satu.

Skripsi ini juga tidak lepas dari kekurangan dan kesalahan baik isi maupun susunannya. Oleh karena itu, penulis menerima segala kritik dan saran dari semua pihak demi kesempurnaan skripsi ini. Akhirnya penulis berharap semoga skripsi ini dapat memberi manfaat bagi pembaca dan kehidupan.

Jember, November 2015 Ivan Syaikhul Hadi

DAFTAR ISI

Halaman

HALAMAN JUDUL i
HALAMAN PERSEMBAHAN ii
HALAMAN MOTTO iii
HALAMAN PERNYATAAN iv
HALAMAN PEMBIMBINGAN v
HALAMAN PENGESAHAN vi
RINGKASAN vii
PRAKATA ix
DAFTAR ISI xi
DAFTAR GAMBAR xiii
DAFTAR TABEL xiv
DAFTAR LAMPIRAN xv
BAB 1. PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 3
1.3 Batasan Masalah 3
1.4 Tujuan Penelitian 4
1.5 Manfaat Penelitian 4
BAB 2. TINJAUAN PUSTAKA 5
2.1 *Knapsack Problem* 5
2.2 Algoritma 6
2.3 Algoritma Genetika 7
2.3.1 Parameter Algoritma Genetika 8
2.3.2 Prosedur Umum Algoritma Genetika 9
2.3.3 Komponen Algoritma Genetika 10
2.4 Algoritma *Tabu Search* 17
2.5 Algoritma Genetika *Hybrid* 18

BAB 3. METODE PENELITIAN 20
3.1 Data 20
3.2 Langkah-langkah Penelitian 21
BAB 4. HASIL DAN PEMBAHASAN 25
4.1 Hasil 25
4.1.1 Perhitungan 25
4.1.2 Program 32
4.2 Pembahasan 36
4.2.1 Algoritma Genetika *Hybrid* 36
4.2.2 Algoritma Genetika dan Algoritma Genetika *Hybrid* 38
BAB 5. PENUTUP 46
5.1 Kesimpulan 46
5.2 Saran 46
DAFTAR PUSTAKA 48
LAMPIRAN 50

DAFTAR GAMBAR

Halaman

2.1 Skema prosedur algoritma genetika 10
2.2 Gambaran umum populasi 12
2.3 *Roulette wheel* 14
3.1 Skema langkah-langkah penelitian 24
4.1 *Parent* 1 (K_4) dan *parent* 2 (K_2) 29
4.2 *Offspring* 1 (K_4) dan *offspring* 2 (K_2) 29
4.3 Kromosom *parent* (K_4) proses mutasi 29
4.4 Kromosom *offspring* (K_4) proses mutasi 29
4.5 Kromosom *parent* terpilih 32
4.6 Kromosom *offspring* dari *tabu list* 3 32
4.7 Tampilan awal program 33
4.8 Tampilan hasil proses program 34
4.9 Tampilan *pushbutton* lihat data 34
4.10 Tampilan menu *record* GA atau *record* GA *hybrid* 35
4.11 Tampilan menu *tabu list parent* dan anak 35
4.12 Tampilan menu hasil data 36
4.13 Hasil *running* pertama kedua algoritma 39
4.14 Hasil *running* pertama percobaan kedua 41
4.15 Hasil *running* pertama percobaan ketiga 43

DAFTAR TABEL

Halaman

2.1 Istilah-istilah dalam Algoritma Genetika 8
2.2 Contoh Pengkodean Permutasi 11
2.3 Contoh Pengkodean Biner 11
2.4 Nilai *Fitness* 13
2.5 Contoh *Parent* 1 dan 2 15
2.6 Contoh *Offspring* 1 dan 2 15
2.7 Contoh Mutasi Biner 16
2.8 Contoh Mutasi Permutasi 16
3.1 Data Penelitian 21
4.1 Sampel Data Buah-buahan 25
4.2 Populasi Awal 26
4.3 Hasil Perhitungan Nilai *Fitness* dan Berat Total 27
4.4 Hasil Perhitungan Probabilitas Relatif dan Komulatif 27
4.5 Kromosom Terpilih (*random*) 28
4.6 Kromosom Terpilih untuk *Crossover* 28
4.7 Generasi Baru 30
4.8 *Tabu List Parent* 30
4.9 *Tabu List Offspring* 31
4.10 Hasil *Running* Terbaik Setiap Percobaan 37
4.11 Rentang Waktu Setiap Pengujian 38
4.12 Hasil *Running* Percobaan ke-1 Algoritma Genetika 39
4.13 Hasil *Running* Percobaan ke-1 Algoritma Genetika *Hybrid* 40
4.14 Hasil *Running* Percobaan ke-2 Algoritma Genetika 41
4.15 Hasil *Running* Percobaan ke-2 Algoritma Genetika *Hybrid* 42
4.16 Hasil *Running* Percobaan ke-3 Algoritma Genetika 43
4.17 Hasil *Running* Percobaan ke-1 Algoritma Genetika *Hybrid* 44
4.18 Rangkuman Hasil *Running* Terbaik kedua Algoritma 44

DAFTAR LAMPIRAN

Halaman

A. TABEL PENGUJIAN ALGORITMA GENETIKA *HYBRID* DENGAN MERUBAH PARAMETER PROBABILITAS 50

B. TABEL PENGUJIAN ALGORITMA GENETIKA *HYBRID* DENGAN MERUBAH PARAMETER POPULASI..... 55

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Dalam kehidupan sehari-hari sering dihadapkan dengan masalah media penyimpanan yang bersifat terbatas, padahal seharusnya dapat menyimpan beberapa objek kedalam media tersebut. Bagaimana mengatur pemilihan objek dalam media penyimpanan, merupakan suatu pertanyaan yang lazim. Permasalahan tersebut biasa dikenal dengan istilah "Permasalahan *Knapsack*" atau "*Knapsack Problem*". *Knapsack* merupakan suatu kantong atau tempat yang digunakan untuk memuat suatu objek. Kantong atau tempat tersebut hanya dapat menyimpan beberapa objek dengan ketentuan total satuan objek tersebut lebih kecil atau sama dengan satuan kapasitasnya.

Permasalahan *knapsack* terbagi menjadi tiga, yaitu *integer knapsack* atau yang sering dikenal dengan *knapsack 0-1, bounded knapsack* dan *unbounded knapsack*. Permasalahan *integer knapsack* adalah menentukan pengambilan seluruh barang yang akan dimuat atau tidak dimuat sama sekali *"take it or leave it"*. Sedangkan *bounded knapsack* adalah permasalahan pengambilan sebagian atau semua objek dari beberapa objek yang jumlahnya terbatas. Sebaliknya, untuk permasalahan jumlah barang yang tidak terbatas sering dikenal dengan *unbounded knapsack problem*.

Permasalahan *knapsack* merupakan suatu permasalahan yang sering dihadapi oleh perusahaan atau pedagang dalam pengiriman dan pengelolaan barang. Permasalahan ini sering juga terjadi pada media transportasi ketika akan mengangkut banyak barang, dimana berat barang yang diangkut tidak boleh melebihi kapasitas daya angkut media transportasi tersebut, dan diharapkan dari pengangkutan barang tersebut didapatkan profit atau keuntungan yang semaksimal mungkin. Terkadang keterbatasan manusia dalam menyelesaikan masalah *knapsack* tanpa menggunakan alat bantu merupakan salah satu kendala dalam pencarian solusi optimum. Apalagi jika dihadapkan dengan jumlah objek terlampau banyak, maka perhitungannya semakin

sulit. Efisiensi waktu juga sering menjadi pertimbangan. Oleh karena itu, dibutuhkan suatu metode sekaligus program aplikasi penerapan metode tersebut yang dapat membantu menyelesaikan masalah *knapsack*.

Masalah *knapsack* dapat diselesaikan dengan berbagai macam algoritma, salah satunya adalah algoritma genetika seperti yang sudah dilakukan oleh Rukmana (2015) pada penelitiannya yang berjudul *"Penerapan Algoritma Genetika dan Algoritma Harmony Search pada permasalahan Knapsack 0-1"*. Penelitian tersebut menyimpulkan bahwa algoritma genetika lebih cepat konvergen dibandingkan algoritma *Harmony Search*.

Selanjutnya Setemen (2009) dalam penelitiannya menyelesaikan permasalahan penjadwalan mata kuliah dengan menggunakan kombinasi algoritma genetika dan *tabu search*. Hasil yang didapat pada jumlah data yang kecil (45 sampai 99 data), kombinasi algoritma genetika dan *tabu search* dapat menyelesaikan masalah penjadwalan dengan baik. Pada penelitian lain, Amelia dan Priharsari (2014) juga menggunakan perpaduan antara algoritma genetika dan *tabu search* untuk menyelesaikan permasalahan penjadwalan auditor pada audit internal mutu Universitas Brawijaya. Hasil penelitian menunjukkan bahwa penggabungan kedua algoritma tersebut memberikan hasil lebih baik dibanding algoritma genetika. Pada prinsipnya, *tabu search* dalam algoritma genetika digunakan untuk memfilter kromosom yang mengalami *crossover* agar kromosom yang sama tidak dilakukan *crossover* berulang-ulang, sehingga diharapkan dapat memberikan hasil optimum dan mengurangi waktu pencarian solusi.

Dari beberapa penelitian diatas dapat disimpulkan bahwa penerapan perpaduan algoritma genetika dan *tabu search* bisa diterapkan dalam berbagai kehidupan nyata, salah satu masalah kehidupan yang menonjol adalah memaksimalkan profit dalam berbisnis. Perolehan keuntungan maksimal merupakan tujuan yang diharapkan suatu perusahaan dalam menggunakan modalnya secara efektif dan efisien. Pemilihan objek sangat berpengaruh pada omset atau keuntungan. Sebuah perusahaan perlu memilih objek yang akan dipasarkan untuk menjamin kelangsungan hidup usahanya. Tanpa adanya pemilihan objek dihawatirkan perusahaan akan dihadapkan pada resiko bahwa

sewaktu-waktu perusahaan tidak dapat memenuhi keinginan konsumen atau pelanggan, dalam kata lain objek yang dipasarkan tidak laku.

Salah satu usaha dagang (UD) yang bernama Buah Barokah, dalam menjaga kelancaran usahanya selalu berusaha memilih objek yang akan dijual. Terdapat berbagai macam pilihan objek yang harus di beli oleh UD Buah Barokah. Namun dalam proses pembelian objek terdapat batasan yang perlu diperhatikan yaitu kapasitas maksimum daya angkut yang tersedia. Berdasarkan masalah diatas, penulis tertarik untuk meneliti tentang penerapan kombinasi algoritma genetika dan *tabu search* atau yang dikenal dengan "*Hybrid Genetic Algorithm"* dalam menghasilkan optimasi terbaik pada masalah *bounded knapsack*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah di uraikan di atas, maka dapat dirumuskan rumusan masalah sebagai berikut:

a. Bagaimana penerapan algoritma genetika *hybrid* untuk solusi permasalahan *bounded knapsack*?
b. Apa pengaruh parameter (probabilitas, populasi) algoritma genetika *hybrid* dalam pencarian nilai optimum?
c. Diantara algoritma genetika dan algoritma genetika *hybrid*, algoritma mana yang memberikan solusi paling optimal untuk permasalahan *bounded knapsack* (dari segi hasil maupun waktu yang dibutuhkan)?

1.3 Batasan Masalah

Adapun batasan permasalahan yang harus dibatasi pada penyelesaian permasalahan *bounded knapsack* adalah tingkat permintaan konsumen untuk masing-masing objek diasumsikan sama.

1.4 Tujuan

Tujuan yang ingin dicapai pada penelitian ini adalah:

a. Mendapatkan solusi optimum pada permasalahan *bounded knapsack* menggunakan algoritma genetika *hybrid*.
b. Mengetahui pengaruh parameter (probabilitas, populasi) algoritma genetika *hybrid* dalam pencarian nilai optimum.
c. Membandingkan algoritma genetika dan algoritma genetika *hybrid* (dari segi hasil dan waktu).

1.5 Manfaat

Manfaat dari penelitian ini diharapkan dapat memberikan alternatif metode dalam menyelesaikan masalah *bounded knapsack* menggunakan algoritma genetika *hybrid*.

BAB 2. TINJAUAN PUSTAKA

2.1 *Knapsack Problem*

Knapsack Problem atau *rucksack problem* merupakan masalah dimana seseorang dihadapkan pada persoalan optimasi pada pemilihan benda yang akan dimasukkan ke dalam sebuah wadah yang memiliki sebuah keterbatasan ruang atau daya tampung. Dengan adanya optimasi dalam pemilihan benda yang akan dimasukkan ke dalam wadah tersebut diharapkan dapat menghasilkan keuntungan yang maksimum. Jadi *Knapsack* adalah permasalahan mengenai optimasi kombinatorial dimana harus mencari solusi terbaik dari banyak kemungkinan yang dihasilkan (Diah *et al*., 2010).

Benda-benda yang akan dimasukkan masing-masing memiliki berat dan sebuah nilai yang digunakan untuk menentukan prioritasnya dalam pemilihan tersebut. Nilainya dapat berupa tingkat kepentingan, harga barang, nilai sejarah, atau yang lainnya. Wadah yang dimaksud disini juga memiliki nilai konstanta yang merupakan nilai pembatas untuk benda-benda yang akan dimasukkan dalam wadah tersebut. Dibutuhkan sebuah cara untuk memasukkan benda-benda tersebut kedalam wadah, supaya didapatkan hasil yang optimum tetapi tidak melebihi kemampuan wadah untuk menampungnya.

Knapsack terdiri dari beberapa persoalan yaitu sebagai berikut (Martello *et al.*, 2006).

a. *Knapsack* 0-1 (*Integer Knapsack*)

   Objek yang dimasukkan ke dalam media penyimpanan dimensinya harus dimasukkan semua atau tidak sama sekali (*take it or leave it*).

b. *Knapsack* terbatas (*Bounded Knapsack*)

   Objek yang dimasukkan ke dalam media penyimpanan dimensinya bisa dimasukkan sebagian atau seluruhnya namun jumlah objeknya terbatas.

c. *Knapsack* tak terbatas (*Unbounded Knapsack*)

Jumlah objek yang dimasukkan ke dalam media dimensinya tidak terbatas.

Knapsack yang akan dibahas pada penelitian ini adalah jenis *bounded knapsack*. Dalam persoalan ini, diberikan n buah objek untuk dikemas dalam sebuah media penyimpanan yang memiliki daya tampung maksimal senilai c. Setiap objek (j) memiliki bobot (w_j) dengan nilai keuntungan profit (p_j), dan sebuah batas ketersediaan objek (m_j). Permasalahannya adalah memilih jumlah barang x_j dimana $0 \leq x_j \leq m_j$ dari setiap barang yang jumlah profitnya berbeda, dan tidak boleh melebihi daya tampung maksimal *c*.

Secara matematis masalah *bounded knapsack* dapat dirumuskan sebagai berikut:

Fungsi tujuan maksimasi

$$z = \sum_{j=1}^{n} p_j x_j \tag{2.1}$$

kendala

$$\sum_{j=1}^{n} w_j x_j \leq c, \tag{2.2}$$

dengan $x_j \epsilon\{0,1,\ldots,m_j\}, j = 1,\ldots,n,$

dimana semua koefisien adalah bilangan bulat positif. Tanpa menghilangan sifat umum, asumsikan bahwa $m_j w_j \leq c$ untuk $j = 1,\ldots,n,$ sehingga semua jenis barang yang tersedia bisa dimuat ke dalam *knapsack* (Pisinger, 1995).

2.2 Algoritma

Algoritma adalah logika, metode, dan tahapan (urutan) sistematis yang digunakan untuk memecahkan suatu permasalahan. Algoritma dapat juga diartikan sebagai urutan langkah secara sistematis dan logis dalam penyelesaian masalah. Algoritma ditulis dengan notasi khusus yang mudah dimengerti dan notasi dapat diterjemahkan kedalam suatu bahasa pemrograman. Algoritma akan memerlukan masukan (*input*) tertentu untuk memulai dan akan menghasilkan keluaran (*output*) tertentu pada akhirnya. Dalam suatu algoritma, mencari langkah yang paling sesuai

dalam menyelesaikan masalah merupakan hal yang perlu diperhatikan, karena setiap algoritma memiliki karakteristik tertentu yang memiliki kekurangan maupun kelebihan tersendiri (Nugraha, 2012).

Algoritma yang baik adalah algoritma yang dapat meminimumkan kebutuhan ruang dan waktu. Dalam suatu algoritma yang menjadi perhatian adalah ruang memori yang dibutuhkan dan lamanya waktu tempuh untuk menjalankan suatu algoritma tersebut. Meskipun suatu algoritma memberikan hasil yang mendekati optimal tetapi waktu yang dibutuhkan sangat lama maka algoritma tersebut biasanya jarang dipakai (Munir, 2005).

2.3 Algoritma Genetika

Algoritma genetika pertama kali dikembangkan oleh John Holland pada tahun 1975. Holland mengatakan bahwa setiap masalah yang berbentuk adaptasi (alami maupun buatan) dapat diformulasikan dalam terminologi genetika. Algoritma genetika adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom (Kusumadewi dan Hari, 2005). Algoritma genetika bekerja berdasarkan pada proses genetika yang ada pada mahluk hidup, yaitu perkembangan generasi dalam sebuah populasi yang alami, secara lambat laun mengikuti proses seleksi alam “siapa yang kuat, dialah yang akan bertahan”.

Teknik pendekatan pada algoritma genetika berbeda dengan algoritma yang lain, karena pada algoritma genetika, pertama di mulai dengan membangkitkan secara random solusi-solusi yang sering dikenal dengan sebutan *population* (Seryadi, 2003). Algoritma genetika juga memakai mekanisme seleksi alam dan ilmu genetika, sehingga istilah-istilah yang digunakan dalam algoritma genetika bersesuaian dengan istilah-istilah pada seleksi alam dan ilmu genetika. Berikut diberikan tabel beberapa istilah dalam algoritma genetika.

Tabel 2.1 Istilah-istilah dalam Algoritma Genetika

Istilah	Keterangan
Kromosom	Individu yang merupakan calon solusi / kumpulan gen
Gen	Bagian dari kromosom
Populasi	Kumpulan kromosom / calon solusi

Golberg pada tahun 1989 mengemukakan bahwa algoritma genetika mempunyai karakteristik-karakteristik yang perlu diketahui sehingga dapat terbedakan dari prosedur pendekatan yang lain, yaitu:

a. algoritma genetika diawali dengan pengkodean dari himpunan solusi permasalahan berdasarkan parameter yang telah ditetapkan;
b. algoritma genetika melakukan pendekatan pada sebuah solusi dari sejumlah individu-individu yang merupakan solusi permasalahan bukan hanya dari sebuah individu;
c. algoritma genetika menggunakan informasi fungsi *fitness* sebagai cara untuk mengevaluasi individu yang mempunyai solusi terbaik;
d. algoritma genetika menggunakan aturan-aturan transisi peluang, bukan aturan deterministik.

2.3.1 Parameter Algoritma Genetika

Menurut Gen dan Cheng (1997) parameter yang digunakan dalam algoritma genetika yaitu:

a. Ukuran populasi

   Jumlah individu atau kromosom dinyatakan sebagai ukuran dari populasi.

b. Probabilitas *crossover* (Pc)

   Probabilitas *crossover* bernilai $0 \leq Pc \leq 1$. Semakin tinggi nilai Pc yang digunakan, maka semakin besar kemungkinan algoritma Genetika mengeksplorasi

ruang pencarian, sekaligus mempercepat ditemukannya solusi optimum. Nilai probabilitas yang baik berkisar antara 0,6 sampai 1.

c. Probabilitas mutasi (Pm)

Probabilitas mutasi menunjukkan jumlah kromosom yang mengalami proses mutasi dalam satu populasi. Probabilitas mutasi bernilia $0 \leq Pm \leq 1$, namun nilai yang digunakan umumnya berkisar antara 0,001 sampai 0,2. Jika nilai Pm terlalu besar akan banyak bermunculan kromosom yang kemungkinan tidak memiliki potensi dalam pencapaian solusi optimum, begitu juga sebaliknya jika nilai Pm terlalu kecil maka akan banyak kromosom yang memiliki potensi tidak akan pernah muncul. *Offspring* akan kehilangan kemiripan dengan kromosom induk (*parent*) yang memiliki potensi pada populasi sebelumnya dan algoritma Genetika akan kehilangan kemampuan untuk belajar dari proses pencarian yang lalu.

d. Kriteria pemberhentian

Kriteria pemberhentian adalah *Number of Improvisation* (NI) yang merepresentasikan jumlah n iterasi yang akan digunakan dan telah ditentukan sebelumnya.

2.3.2 Prosedur Umum Algoritma Genetika

Secara umum struktur dari algoritma genetika dapat didefinisikan dengan langkah-langkah sebagai berikut (Gambar 2.1).

a. Membangkitkan populasi awal

Populasi awal dibangkitkan secara random, sehingga didapatkan sebuah solusi awal. Populasi sendiri terdiri atas sejumlah kromosom yang merepresentasikan solusi yang diinginkan.

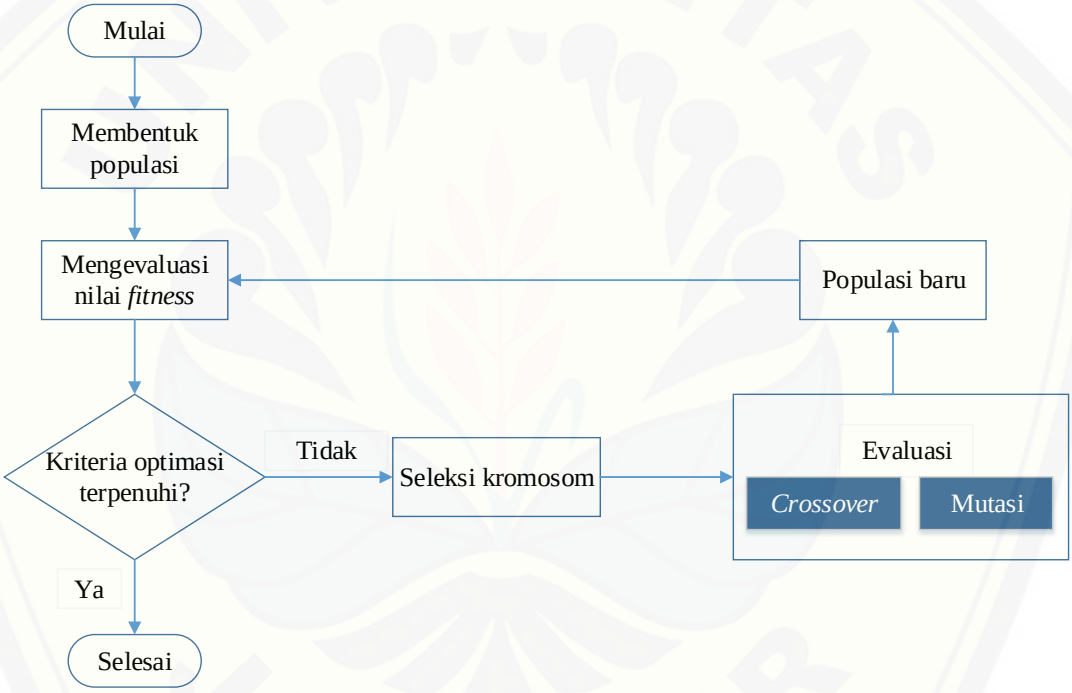
b. Evaluasi solusi

Sebuah kromosom akan memlalui proses evaluasi pada setiap generasi dengan menggunakan alat ukur yang dinamakan *fitness*. Proses ini akan mengevaluasi setiap populasi dengan menghitung nilai *fitness* setiap kromosom sampai terpenuhi kriteria

pemberhentian. Nilai *fitness* suatu kromosom menggambarkan kualitas kromosom dalam populasi tersebut.

c. Membentuk generasi baru

Proses pembentukan generasi baru menggunakan tiga operator yaitu seleksi, *crossover* dan mutasi. Proses ini dilakukan berulang-ulang hingga kriteria pemberhentian terpenuhi. Generasi baru ini merupakan representasi dari solusi baru yang dikenal dengan istilah *offspring*.



Gambar 2.1 Skema prosedur algoritma genetika

2.3.3 Komponen Algoritma Genetika

Beberapa komponen algoritma Genetika adalah sebagai berikut.

a. Representasi Kromosom

Kromosom direpresentasikan sebagai calon solusi suatu masalah dalam populasi awal dengan teknik pengkodean, teknik ini meliputi pengkodean gen dari kromosom. Gen merupakan bagian dari kromosom, dimana satu gen biasanya akan

mewakili satu variabel. Gen dapat direpresentasikan dalam bentuk nilai bit, pohon, *array* bilangan *real*, elemen permutasi dan lain-lain.

Ada beberapa cara pengkodean, antara lain:

1) Pengkodean Permutasi (*Permutation Encoding*)

   Pengkodean permutasi adalah pengkodean yang digunakan dalam masalah pengurutan data, *travelling Salesman Problem* (TSP) atau masalah pengurutan tugas. Setiap kromosom dalam pengkodean permutasi merupakan serangkaian gen yang mewakili angka dalam suatu urutan tertentu. Contohnya seperti yang ditunjukkan pada Tabel 2.2 berikut.

Tabel 2.2 Contoh Pengkodean Permutasi

Kromosom	Pengkodean					
Kromosom 1	4	6	2	1	5	3
Kromososm 2	2	3	1	4	6	5

2) Pengkodean Biner (*Binary Encoding*)

   Pengkodean biner adalah pengkodean yang paling umum digunakan, karena sebagian besar karya awal algoritma genetika dikerjakan dengan menggunakan pengkodean jenis ini. Dalam *binary encoding* setiap kromosom direpresentasikan dalam barisan bit 0 atau 1. Contohnya seperti yang ditunjukkan pada Tabel 2.3 berikut.

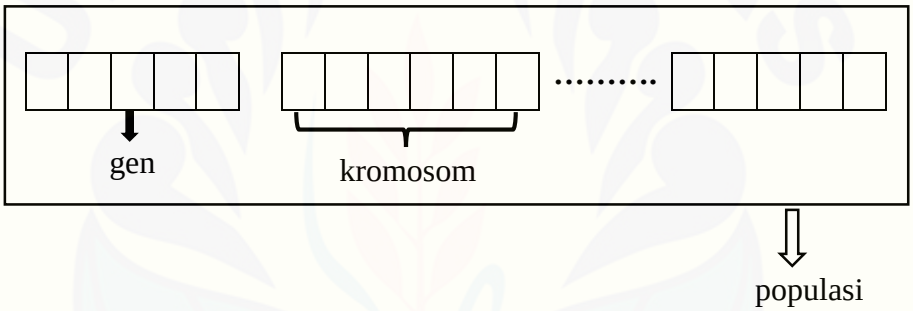
Tabel 2.3 Contoh Pengkodean Biner

Kromosom	Pengkodean					
Kromosom 1	1	0	0	1	1	0
Kromososm 2	1	1	0	0	1	1

Pengkodean kromosom tergantung pada ragam masalah yang dihadapi. Pada permasalahan *bounded knapsack* pengkodean yang digunakan adalah pengkodean biner (*Binary Encoding*).

b. Pembentukan Populasi Awal

Proses pada algoritma Genetika diawali dengan pembentukan populasi awal yang bertujuan untuk membangkitkan generasi awal yang akan menjadi dasar untuk membangkitkan generasi-generasi selanjutnya. Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian dilakukan inisialisasi terhadap kromosom yang terdapat dalam kromosom tersebut. Inisialisasi kromosom dilakukan secara acak, namun harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada. Secara umum populasi berbentuk seperti Gambar 2.2.



Gambar 2.2 Gambaran umum populasi

c. Fungsi *Fitness*

Suatu individu dievaluasi berdasarkan suatu fungsi tertentu sebagia ukuran performansinya. Fungsi ini dinamakan fungsi *Fitness*. Dalam evolusi alam, individu yang bernilai *fitness* tinggi yang akan hidup, sebaliknya individu yang bernilai *fitness* rendah akan mati (Setemen, 2008). Fungsi ini membedakan kualitas dari kromosom untuk mengetahui seberapa baik kromosom yang dihasilkan. Semakin tinggi nilai *fitness* akan semakin besar kemungkinan kromosom tersebut terpilih untuk generasi berikutnya. Pada kebanyakan optimasi, fungsi *fitness* dibentuk berdasarkan fungsi objektifnya. Jika masalah optimasinya memaksimumkan maka fungsi *fitness* sama dengan fungsi objektif (Ayuningtyas, 2008). Jika solusi yang dicari adalah meminimalkan sebuah fungsi $h(x)$ (masalah minimasi), maka dapat

menggunakan persamaan $(x) = \frac{1}{h(x)}$. Tetapi jika masalahnya adalah memaksimalkan fungsi $h(x)$ (masalah maksimasi), maka nilai *fitness* yang digunakan adalah nilai dari fungsi $h(x)$ tersebut, seperti yang ditunjukkan pada persamaan (2.3) berikut.

$$f\ (x) = h\ (x) \qquad (2.3)$$

dengan,

$f\ (x)$ = nilai *fitness*

$h\ (x)$ = nilai optimasi (minimasi atau maksimasi)

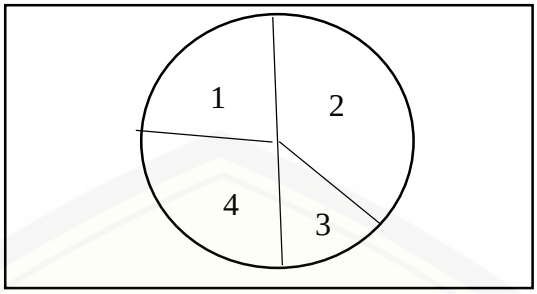
x = kromosom

d. Seleksi

Seleksi dilakukan pada setiap generasi untuk memilih kromosom-kromosom dari populasi yang akan menjadi *parent* bagi generasi berikutnya. Seleksi ini bertujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang memiliki nilai *fitness* yang tinggi untuk bereproduksi. Salah satu metode seleksi yang umum digunakan adalah *roulette-wheel* (roda *roulette*). Sesuai dengan namanya, metode ini menirukan permainan *roulette-wheel* dimana masing-masing kromosom menempati potongan lingkaran pada roda *roulette* secara proporsional sesuai dengan nilai *fitness*nya. Kromosom yang memiliki nilai *fitness* lebih besar menempati potongan lingkaran yang lebih besar dibanding dengan kromosom yang memiliki nilai *fitness* rendah, seperti contoh pada Tabel 2.4 dan Gambar 2.3.

Tabel 2.4 Nilai *Fitness*

Kromosom (k)	*Fitness*
1	220
2	375
3	125
4	280
Jumlah	1000



Gambar 2.3 *Roulette wheel*

Langkah-langkah yang dilakukan dalam seleksi roda *roullete* yaitu :

1) Menghitung total *fitness* dari semua kromosom;

$$total\ fitness = \sum_{i=1}^{n} h(x) \quad (2.4)$$

2) Menghitung probabilitas relatif (p_k) setiap kromosom dengan

$$p_k = \frac{fitness\ (k)}{total\ fitness} \quad (2.5)$$

3) Menghitung probabilitas komulatif (q_k) dengan

$$q_k = q_{k-1} + p_k \text{ dan } q_0 = 0 \quad (2.6)$$

4) Menciptakan nilai *random* $(r_i), 0 \leq r_i \leq 1$, sebanyak ukuran populasi;

5) Memilih kromosom ke-*k* untuk dijadikan *parent* pada proses *crossover* dengan syarat $q_{k-1} \leq r_i \leq q_k$ dengan $k, i = 1, 2, 3, \ldots, N$. Kromosom yang terpilih pada proses seleksi akan melewati proses genetika yaitu *crossover* dan mutasi (Tuloli, 2007).

e. Pindah Silang (*Crossover*)

Crossover atau pindah silang merupakan salah satu komponen penting dalam algoritma genetika. Sebuah kromosom yang mengarah pada solusi yang bagus bisa diperoleh dari proses memindahsilangkan dua buah kromosom (Diah *et al.*, 2010). Dua buah kromosom bisa berpindah silang jika suatu bilangan yang dibangkitkan secara *random* (r) nilainya kurang dari atau sama dengan P_c ($0 \leq r \leq 1$). Bilangan *random* tersebut dibangkitkan setiap kali akan melakukan *crossover* pada kromosom.

Ada banyak metode *crossover* diantaranya *Partial Mapped Crossover* (PMX), *Order Crossover* (OX), *crossover* satu titik, *crossover* banyak titik dan lain sebagainya. Dalam penelitian ini metode yang digunakan ialah *crossover* satu titik dan *crossover* banyak titik, karena biasanya metode ini digunakan untuk representasi kromosom dalam biner. Pada *crossover* satu titik, posisi k ($k = 1, 2, 3, \dots, N-1$) dimana *N*= panjang kromosom yang diseleksi secara random. Contohnya seperti yang ditunjukkan pada Tabel 2.5 dan Tabel 2.6 berikut.

Tabel 2.5 Contoh *Parent* 1 dan 2

Kromosom	Pengkodean				
Parent 1	1	0	**1**	**0**	**1**
Parent 2	1	1	**0**	**1**	**0**

Tabel 2.6 Contoh *Offspring* 1 dan 2

Kromosom	Pengkodean				
Offspring 1	1	0	**0**	**1**	**0**
Offspring 2	1	1	**1**	**0**	**1**

f. Mutasi

Mutasi adalah proses memodifikasi kromosom anak secara *random*. Mutasi berperan untuk menggantikan gen yang hilang dari populasi akibat proses seleksi dan memungkinkan munculnya gen yang tidak ada dalam populasi awal (Kusumadewi & Purnomo, 2005). Proses mutasi akan terjadi pada suatu gen, jika suatu bialangan yang dibangkitkan secara *random* (r), nilainya kurang dari atau sama dengan Pm yaitu $0 \leq r \leq 1$. Metode mutasi, diantaranya yaitu.

1) Mutasi pengkodean biner

Mutasi ini dilakukan dengan mengubah nilai bit pada posisi tertentu yang dipilih secara *random* atau menggunakan skema tertentu pada kromosom, yang disebut dengan *inverse* bit. Contohnya seperti yang ditunjukkan pada Tabel 2.7 berikut.

Tabel 2.7 Contoh Mutasi Biner

Kromosom	Pengkodean					
Parrent	1	0	0	1	0	1
Offspring	1	0	1	1	0	1

2) Mutasi pengkodean permutasi

Proses mutasi pada pengkodean biner dengan mengubah langsung gen-gen pada kromosom tidak dapat dilakukan pada pengkodean ini, karena konsistensi urutan permutasi sangat diperhatikan. Salah satu cara yang dapat dilakukan adalah memilih dua posisi dari kromosom dan kemudian nilainya saling dipertukarkan. Contohnya seperti yang ditunjukkan pada Tabel 2.8 berikut.

Tabel 2.8 Contoh Mutasi Permutasi

Kromosom	Pengkodean
Kromosom sebelum mutasi	123465879
Kromososm setelah mutasi	127465839

Peluang mutasi (Pm) menunjukkan seberapa sering mutasi akan dilakukan. Jika tidak dilakukan mutasi ($Pm = 0\%$), artinya kromosom dalam populasi tidak mengalami perubahan setelah *crossover*. Jika $Pm = 100\%$, maka semua kromosom mengalami perubahan. Biasanya Pm diset sebagai $1/n$, dimana n adalah jumlah gen dalam kromosom (Rukmana, 2015).

g. Evaluasi

Evaluasi digunakan untuk menghitung kebugaran (*fitness*) setiap kromosom. Semakin besar *fitness* maka semakin baik kromosom tersebut untuk dijadikan calon solusi. Karena masalah ini adalah pencarian nilai maksimum, maka nilai *fitness* untuk tiap individu bisa dihitung secara langsung $fitness = f(x)$.

h. Kriteria Pemberhentian

Proses algoritma genetika akan terus menerus berlangsung hingga kriteria pemberhentian terpenuhi. Kondisi berhenti pada algoritma genetika terjadi apabila jumlah generasi (iterasi) yang diinginkan telah tercapai, n generasi berurutan selanjutnya tidak dijumpai solusi yang lebih baik dari sebelumnya dan setelah t satuan waktu tercapai (biasanya digunakan untuk membandingkan performa algoritma) (Mahmudi, 2013).

2.4 Algoritma *Tabu Search*

Tabu seacrh pertama kali diperkenalkan oleh Fred Glover sekitar tahun 1986. *Tabu search* merupakan sauatu metode optimasi yang menggunakan *shortterm memory* untuk menjaga agar proses pencarian tidak terjebak pada nilai optimum lokal. Sehingga maksud dari algoritma ini adalah mencegah terjadinya perulangan dan ditemukannya solusi yang sama pada suatu iterasi. Metode ini menggunakan *tabu list* untuk menyimpan beberapa solusi yang baru saja dievaluasi (Amelia dan Priharsari, 2014).

Tabu List yang ada pada *tabu search* digunakan untuk menyimpan sekumpulan solusi yang baru saja dievaluasi. Selama proses optimasi pada setiap iterasi, solusi yang akan dievaluasi akan dicocokkan terlebih dahulu dengan isi *tabu list*. Apabila solusi tersebut sudah ada pada pada *tabu list*, maka solusi tersebut tidak akan dievaluasi lagi pada iterasi berikutnya. Apabila sudah tidak ada lagi solusi yang tidak menjadi anggota *tabu list*, maka nilai terbaik yang baru saja diperoleh merupakan solusi yang sebenarnya.

Kelebihan *tabu search* terletak pada struktur memori yang fleksibel. Struktur memori itu akan memperbolehkan pencarian terus dilakukan meskipun solusi yang diperoleh saat ini tidak ada yang lebih baik dari solusi terbaik yang telah diperoleh. Struktur memori dalam *tabu search* menggunakan empat prinsip utama, yaitu :

a. *recency based memory*, yaitu memori yang tetap menjaga struktur terbaik dari solusi awal yang ditemukan selama proses pencarian pada setiap iterasinya,

sehingga apabila pada suatu iterasi ditemukan solusi yang lebih baik, maka solusi ini akan tetap dipertahankan sampai ditemukannya solusi baru yang lebih baik. Pada metode *tabu search* ada kemungkinan terjadi perulangan perhitungan, sehingga untuk mengantisipasi hal tersebut dibuatlah struktur yang disebut *tabu list. Tabu list* berfungsi untuk menyimpan sekumpulan solusi yang telah dievaluasi, sehingga untuk itersi selanjutnya sebuah solusi yang telah dievaluasi akan dicocokkan terlebih dahulu dengan isi *tabu list*. Apabila solusi itu sudah ada pada *tabu list,* maka solusi tersebut tidak akan dievaluasi lagi pada iterasi selanjutnya;

b. *frequency,* menyediakan sebuah informasi yang telah direkam oleh *recency based memory*. Sehingga keduanya dapat saling melengkapi untuk membentuk suatu informasi permanen guna mengevaluasi pergerakan yang terjadi;
c. *quality,* adalah kemampuan untuk membedakan solusi terbaik yang dikunjungi selama pencarian atau iterasi berlangsung;
d. *influence*, mempertimbangkan efek yang terjadi dari pemilihan solusi yang dipilih selama pencarian berlangsung, tidak hanya kualitasnya melainkan juga strukturnya (Berlianty dan Arifin, 2010).

2.5 Algoritma Genetika *Hybrid*

Algoritma genetika *hybrid* merupakan kombinasi metode-metode heuristik lain ke dalam algoritma genetika dengan harapan mampu meningkatkan kinerja algoritma genetika (Syarif *et al*., 2007). Pada prinsipnya hibridisasi ini diharapkan mampu memberikan solusi lain yang lebih baik di sekitar lokal optimum yang diberikan oleh algoritma genetika atau dikenal dengan istilah (*local search*). Algoritma genetika *hybrid* menggunakan fungsi random sehingga menyebabkan algoritma genetika *hybrid* menjadi suatu algoritma berbasis komputer. *Best Improvement Local Search* adalah memperhatikan semua solusi tepat diantara semua lingkungan solusi dan mengambil solusi yang paling baik diantara semua solusi. Kelebihan dari algoritma genetika *hybrid* yang merupakan kombinasi dari algoritma genetika dan algoritma *tabu search* terletak pada proses *crossover* yang menggunakan *tabu list* untuk mencegah kromosom yang

sama agar tidak di-*crossover* lagi, sehingga akan lebih cepat ditemukannya solusi optimum (Amelia dan Priharsari, 2014).

Pada kasus atau penelitian ini, *tabu search* hanya digunakan untuk memfilter kromosom yang mengalami *crossover* agar kromosom yang sama tidak dilakukan *crossover* berulang-ulang. Saat iterasi pertama, semua kromosom yang mengalami *crossover* dan kromosom hasil *crossover* disimpan ke dalam *tabu list*. Untuk iterasi berikutnya, kromosom terpilih pada proses *crossover*, terlebih dahulu di cek pada *tabu list* apakah kromosom tersebut sudah ada atau belum. Jika kromosom tersebut sudah ada pada *tabu list* maka kromosom tersebut tidak perlu di~*crossover*, karena hasilnya sudah tersedia pada *tabu list*, sehingga tinggal melanjutkan ke langkah berikutnya (mutasi). Sebaliknya jika kromosom tersebut belum ada pada *tabu list*, maka kromosom tersebut beserta hasilnya akan disimpan pada *tabu list*.

BAB 3. METODE PENELITIAN

Langkah-langkah penelitian masalah *bounded knapsack* dengan menggunakan kombinasi algoritma genetika dan algoritma *tabu search* (Genetika hybrid) akan dipaparkan pada bab ini.

3.1 Data

Data yang akan digunakan dalam penelitian ini berupa data primer yang diperoleh dari hasil wawancara kepada pemilik UD Buah Barokah (H. Edi Sungkono), usaha dagang ini berlokasi di dusun Rejosari kecamatan Cluring kabupaten Banyuwangi. Dari proses wawancara didapatkan beberapa informasi mengenai aktifitas, stok barang (jumlah yang tersedia), profit, berat setiap kemasan, biaya operasional, waktu perjalanan. Dari beberapa informasi yang didapatkan, penulis hanya mengambil beberapa informasi yang diperlukan dalam menyelesaikan permasalahan *bounded knapsack.* Informasi yang diambil berupa nama buah, berat setiap kemasan, jumlah yang tersedia, dan profit (Tabel 3.1), karena pada penelitian ini bertujuan untuk memaksimalkan keuntungan dengan syarat berat barang tidak melebihi daya tampung maksimal yang tersedia. Sedangkan data yang tidak relevan dengan definisi *bounded knapsack* diabaikan.

Proses pengambilan sejumlah *n* barang dilakukan secara acak, sehingga setiap barang memiliki peluang yang sama, namun sejumlah barang yang diambil dibatasi dengan *c* kapasitas daya angkut. Data yang diambil adalah data mentah yang berupa bobot kemasan per-item (w_j), banyak item (m_j), nilai keuntungan per-item (p_j) dan kapasitas daya angkut yang tersedia (c). Data tersebut disajikan pada Tabel 3.1 berikut.

Tabel 3.1 Data Penelitian

No.	Nama Buah	Berat (kg)	Jumlah (kemasan)	*Profit*/Item (Rp)
1	Anggur	0,5	10	2300
2	Alpukat	5	7	20000
3	Apel	15	12	47000
4	Belimbing	10	5	12000
5	Buah Naga	20	11	52000
6	Duku	1	8	2200
7	Durian	1	0	3000
8	Jambu Merah	10	4	14000
9	Jambu Kristal	10	8	18500
10	Jeruk	20	9	75000
11	Kesemek	3	2	7500
12	Mangga	15	13	25500
13	Manggis	5	3	9700
14	Melon	10	5	22000
15	Nanas	12	7	23500
16	Nangka	1	14	1850
17	Pepaya	20	6	41000
18	Salak	20	12	112000
19	Sawo	3	3	6500
20	Semangka	4,5	2	8500
21	Strowberi	0,5	16	1800
22	Anggur Merah	0,5	11	2500
23	Langsep	1	4	1250
24	Klengkeng	2,5	6	5000
25	Rambutan	5	12	5750

3.2 Langkah-Langkah Penelitian

Langkah-langkah yang dilakukan untuk menyelesaikan permasalahan *bounded knapsack* menggunakan algoritma genetika *hybrid* yaitu:

a. Mengumpulkan data

Proses pengumpulan data dilakukan dengan cara wawancara langsung kepada pemilik UD Buah Barokah, dari proses wawancara didapatkan berbagai informasi yang kemudian diidentifikasi sehingga didapatkan data seperti pada Tabel 3.1.

b. Mengolah data menggunakan algoritma genetika *hybrid*

Data yang diperoleh selanjutnya dibawa kedalam masalah *bounded knapsack* dan diolah menggunakan algoritma genetika *hybrid* dengan langkah-langkah sebagai berikut:

1. Inisialisasi parameter

   Parameter digunakan untuk memberikakan suatu probabilitas (kemungkinan atau peluang) yang berupa persentase. Dalam algoritma genetika ada dua probabilitas yang digunakan yaitu probabilitas *crossover* dan probabilitas mutasi, dimana keduanya memiliki rentang nilai 0 sampai 1. Dalam penelitian ini nilai probabilitas *crossover* dan mutasi yang digunakan mengacu pada bab sebelumnya, yaitu masing-masing antara 0,6 sampai 1 dan 0,001 sampai 0,2.

2. Membentuk populasi awal

   Pembentukan populasi awal dilakukan dengan menggunakan pengkodean biner, dimana nilai 1 berarti barang tersebut diambil dan nilai 0 berarti sebaliknya, yaitu barang tersebut tidak diambil. Populasi awal ini akan menjadi dasar untuk membangkitkan generasi-generasi selanjutnya;

3. Menghitung nilai *fitness*

   Menghitung nilai *fitness* dilakukan pada setiap solusi, dengan tujuan meminimalkan berat dan memaksimalkan keuntungan;

4. Menyeleksi solusi

   Seleksi dilakukan dengan metode *roulette-wheel* (roda *roulette*). Seleksi dilakukan dalam sebuah populasi untuk memilih induk bagi generasi selanjutnya;

5. Melakukan proses reproduksi

   Induk yang terpilih pada proses seleksi akan melewati proses *Crossover* dan mutasi, dimana pada proses *crossover* dilengkapi dengan *tabu list* dengan menggunakan algoritma *tabu search*;

6. Mengevaluasi calon generasi baru

   Setelah melewati proses reproduksi maka akan dievaluasi untuk memilih generasi baru;

7. Kriteria pemberhentian

   Setelah didapatkan generasi baru, ulangi langkah (3) hingga mencapai iterasi maksimal yang telah ditentukan.

c. Membuat program

   Membuat program dari permasalahan *bounded knapsack* dengan algoritma genetika *hybrid* menggunakan *software Matlab 7.8.0* R2009a.

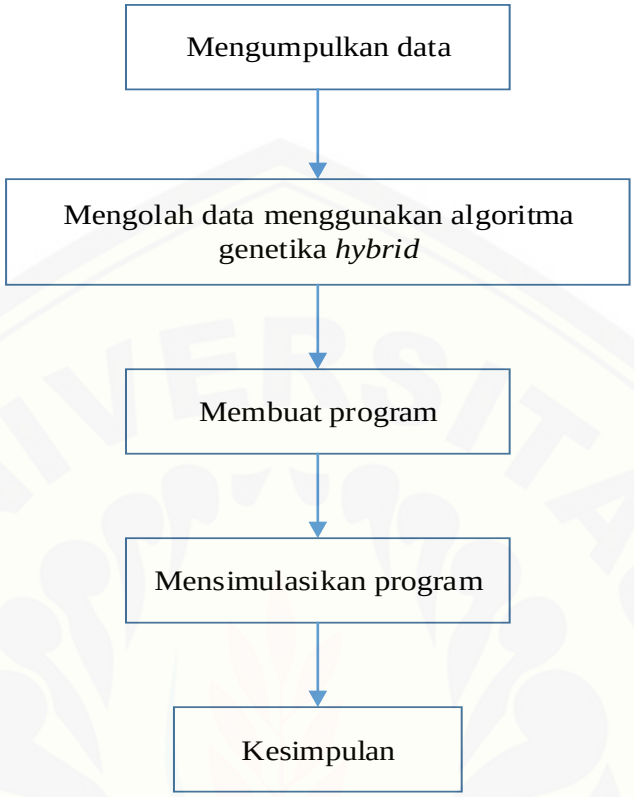
d. Mensimulasikan program

   Simulasi program menggunakan data yang telah dikumpulkan dari UD Buah Barokah.

e. Menyimpulkan

   Membuat kesimpulan dari simulasi program *software Matlab* yang telah dijalankan pada permasalahan *bounded knapsack* menggunakan algoritma genetika *hybrid*.

Gambar 3.1 berikut merupakan skema dari langkah-langkah penyelesaian algoritma genetika *hybrid*.



Gambar 3.1 Skema langkah-langkah penelitian