OPTIMASI KOMPOSISI PAKAN KAMBING POTONG MENGGUNAKAN ALGORITMA GENETIKA

SKRIPSI

KONSENTRASI KOMPUTASI DAN SISTEM CERDAS

Diajukan untuk memenuhi sebagian persyaratan
mencapai gelar Sarjana Komputer

Disusun oleh
DYAH PRAMESTI
NIM. 115060801111010

KEMENTERIAN RISET TEKNOLOGI PENDIDIKAN TINGGI
UNIVERSITAS BRAWIJAYA
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
PROGRAM STUDI INFORMATIKA/ ILMU KOMPUTER
MALANG

2015

LEMBAR PERSETUJUAN

OPTIMASI KOMPOSISI PAKAN KAMBING POTONG MENGGUNAKAN ALGORITMA GENETIKA

SKRIPSI

KONSENTRASI KOMPUTASI DAN SISTEM CERDAS

Untuk memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

Disusun oleh :

DYAH PRAMESTI

NIM. 115060801111010

Skripsi ini telah diperiksa dan disetujui oleh dosen pembimbing

Dosen Pembimbing 1	Dosen Pembimbing 2
Wayan F. Mahmudy, S.Si., MT., Ph.D	**Indriati, ST., M.Kom**
NIP. 19720919 199702 1001	**NIK. 831013 06 1 2 0035**

LEMBAR PENGESAHAN

OPTIMASI KOMPOSISI PAKAN KAMBING POTONG MENGGUNAKAN ALGORITMA GENETIKA

SKRIPSI

LABORATORIUM KOMPUTASI DAN SISTEM CERDAS

Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

DYAH PRAMESTI

NIM. 115060801111010

Skripsi ini telah diuji dan dinyatakan lulus
pada tanggal 15 Mei 2015

Penguji I	Penguji II
Rekyan Regasari MP, ST., M.T.	**Edy Santoso, S.Si., M.Kom**
NIK. 770414 06 1 2 0253	**NIP. 19740414 200312 1 004**

Penguji III

Budi Darma Setiawan, S.Kom., M.Cs
NIP. 19841015 201404 1 002

Mengetahui,
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, M.T.
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 orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 30 April 2015

Mahasiswa,

DYAH PRAMESTI

NIM. 115060801111010

KATA PENGANTAR

Alhamdulillahirobbilalamin, puji syukur kehadirat Allah SWT, karena atas limpahan rahmat, dan hidayah-Nya, penulis dapat menyelesaikan skripsi ini. Tidak lupa shalawat serta salam selalu tercurahkan kepada Nabi Muhammad SAW beserta keluarga dan sahabat Beliau.

Skripsi ini menjadi salah satu syarat untuk mencapai gelar Sarjana Komputer di Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Judul skripsi ini adalah "**Optimasi Komposisi Pakan Kambing Potong Menggunakan Algoritma Genetika**". Penulis mengangkat judul tersebut untuk membantu proses komposisi bahan pakan pada kambing potong.

Penyusunan skripsi ini tidak lepas dari bantuan semua pihak yang terus memberikan semangat, kritik, saran dan bimbingan serta doa. Oleh karena itu penulis menyampaikan ucapan terima kasih kepada:

1. Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D., selaku pembimbing I skripsi yang telah membantu dan membimbing penulis dari penyusunan proposal sampai skripsi ini selesai.
2. Indriati, ST.,M.Kom., selaku pembimbing II skripsi yang telah membantu dan membimbing dalam penyusunan skripsi.
3. Aswin Suharsono, ST, MT., selaku dosen penasehat akademik yang selalu memberikan nasehat kepada penulis selama menempuh masa studi.
4. Drs. Mardji,M.T., selaku Ketua Program Teknik Informatika / Ilmu Komputer yang telah memberikan kesempatan kepada penulis untuk menyelesaikan skripsi ini.
5. Ibu Siti Nurhasah, atas segala pengorbanan Beliau, memberikan dukungan, mendoakan, memotivasi dan memberikan kasih sayang kepada penulis.
6. Ramiadi,S.Sos.,M.Si., atas segala doa, nasehat, kasih sayang, dan dukungan kepada penulis.
7. Seluruh anggota keluarga penulis yang turut memberikan semangat kepada penulis.

8. Segenap bapak dan ibu dosen program studi Teknik Informatika/Ilmu Komputer beserta staff pegawai yang telah membantu penulis selama masa studi.
9. Warga kost 292C yang telah memberikan dukungan kepada penulis selama menempuh studi dan menyelesaikan skripsi ini.
10. Teman- teman program studi Teknik Informatika/Ilmu Komputer Universitas Brawijaya yang telah memberikan masukan kepada penulis selama menempuh studi dan menyelesaikan skripsi ini.
11. Dan semua pihak yang tidak dapat disebutkan satu persatu, yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Penulis menyadari bahwa penulisan skripsi ini masih jauh dari kata sempurna. Oleh karena itu penulis mengharapkan masukan berupa saran dan kritik dari semua pihak demi perbaikan kemampuan penulis. Akhir kata semoga penulisan skripsi ini dapat memberikan manfaat bagi semua pihak.

Malang, 30 April 2015

Penulis

ABSTRAK

Dyah Pramesti, 2015. Optimasi Komposisi Pakan Kambing Potong Menggunakan Algoritma Genetika. Skripsi Program Studi Informatika / Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen pembimbing : Wayan Firdaus Mahmudy, Ph.D dan Indriati, M.Kom.

Permasalahan yang sering terjadi dalam usaha ternak kambing potong adalah penggunaan bahan pakan yang belum efisien. Kesalahan dalam menentukan bahan pakan yang digunakan selama ini berdampak pada rendahnya kandungan nutrisi yang diberikan kepada kambing potong yang mengakibatkan kambing potong tersebut tidak tumbuh dan berkembang dengan baik. Dalam sudut pandang ekonomi, pembelian bahan pakan ternak menjadi biaya tertinggi dalam usaha peternakan, sehingga biaya tersebut harus ditekan serendah mungkin untuk memaksimalkan pendapatan. Mengoptimalkan penyusunan bahan pakan (ransum) kambing potong merupakan cara untuk menekan biaya pembelian bahan pakan serta untuk memaksimalkan keuntungan maupun pendapatan.

Dalam kasus ini digunakan algoritma genetika untuk mengoptimasi komposisi bahan pakan kambing potong. Digunakan representasi kromosom real code dengan panjang kromosom yang didapatkan dari jumlah bahan pakan yang diinputkan. Metode crossover yang digunakan pada penelitian ini adalah extended intermediate crossover dan metode mutasi yang digunakan adalah random mutation. Dan untuk seleksi menggunakan metode ellitsm selection. Solusi optimal diperoleh dari ukuran populasi yaitu 200, crossover rate sebesar 0,1 dan mutation rate 0,5 serta jumlah generasi sebanyak 200 dengan nilai fitness 0.000167046.

Kata Kunci : Algoritma genetika, optimasi pakan, pakan kambing potong

ABSTRACT

Dyah Pramesti, 2015. Optimasi Komposisi Pakan Kambing Potong Menggunakan Algoritma Genetika. Skripsi Program Studi Informatika / Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen pembimbing : Wayan Firdaus Mahmudy, Ph.D dan Indriati, M.Kom.

The common problem that mostly happened in animal husbandry of goat is the insufficient usage of fodder. A simple mistake in choosing the fodder caused the goat to having a little nutrition in their body. That surely make the goat not growing breed well. In the economic point of view, the purchase of fodder becomes the highest cost in factory farming, so the cost should be reduced as low as possible to maximize the revenue. Optimizing the preparations of goat's fodder is a way to reducing the cost, that can also increasing the revenue.

In this case, a genetic algorithm is used to optimizing the goat's fodder. Real code chromosome is used with chromosome length that obtained from the amount of feed ingredients entered. The crossover method that used in this research is the extended intermediate crossover and the mutation method that used is random mutation. The selection used ellitsm selection method. The optimal solution from a population size is 200, 0.1 for the crossover rate, 0.5 for the mutation rate and the generation number of 200 with fitness value 0.000167046.

Keywords : Genetic algorithms, Optimizing the preparations of goat's fodder, feed of goat.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR PERSAMAAN

DAFTAR SOURCE CODE

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kambing potong adalah hewan ternak yang banyak dipelihara oleh masyarakat Indonesia terutama di pedesaan sebagai usaha sampingan atau bahkan untuk dijadikan tabungan. Salah satu faktor masyarakat memilih untuk beternak kambing potong karena Indonesia adalah Negara Tropis yang memiliki iklim yang sesuai untuk pengembangan ternak kambing. Bukan rahasia lagi bahwa kambing potong menjadi salah satu komoditas yang memberikan keuntungan bagi para peternak. Daging kambing banyak disukai konsumen karena aroma, rasa dan keempukannya meskipun daging kambing berlemak. Menurut data staitistik, kebutuhan konsumsi daging kambing dan domba dalam negeri sekitar 5,6 juta ekor setiap tahunnya (Gunawan, 2013).

Tidak semudah yang dibayangkan, dalam usaha ternak kambing terdapat masalah yaitu belum efisiennya penggunaan bahan pakan kambing potong. Serta kesalahan dalam menentukan komposisi bahan pakan yang diberikan kepada kambing potong, karena dapat berpengaruh terhadap rendahnya nutrisi yang diterima kambing potong yang mengakibatkan kambing tidak dapat tumbuh dan berkembang dengan baik, sehingga kualitas kambing potong menurun. Dalam sudut pandang ekonomi, pembelian bahan pakan ternak menjadi biaya tertinggi dalam usaha peternakan, sehingga biaya tersebut harus ditekan serendah mungkin untuk memaksimalkan pendapatan (Nugraha, 2011). Mengoptimalkan penyusunan bahan pakan (ransum) kambing merupakan cara untuk menekan biaya pembelian bahan pakan serta untuk memaksimalkan keuntungan maupun pendapatan.

Untuk membantu proses penentuan komposisi bahan pakan pada kambing potong, diperlukan aplikasi yang diharapkan dapat membantu dalam penentuan komposisi pakan kambing potong dengan memperhatikan kebutuhan kandungan nutrisi pakannya serta biaya yang digunakan untuk membeli pakan ternak karena hal tersebut menjadi hal penting untuk mendapatkan hasil komposisi pakan yang

optimum dengan biaya minimum. Ada beberapa metode untuk menyelesaikan permasalahan optimasi diantaranya adalah *Simulated Annealing (SA), Tabu Search (TS), Particle Swarm Optimization (PSO), Evaluation Strategies (ES), Genetic Algorithm (GA)* dan lain sebagainya. Dalam kasus ini digunakan algoritma genetika. Algoritma genetika adalah salah satu jenis algoritma pencarian heuristik yang cara kerjanya berdasarkan mekanisme evolusi biologis. Algoritma ini didasari oleh konsep evaluasi biologi, sehingga algoritma ini dapat memberikan solusi alternatif terhadap suatu masalah yang ingin diselesaikan (Aribowo, 2008). Prosesnya tersebut dimulai dengan terbentuknya populasi awal secara acak yang terdiri dari individu, kemudian individu tersebut melakukan proses reproduksi untuk menghasilkan anak yang mewarisi sifat-sifat induknya (Turban, 1995 dalam Zukhri, 2014). Algoritma genetika mampu menyelesaikan berbagai masalah kompleks, algoritma genetika ini juga banyak digunakan dan dimanfaatkan dalam berbagai bidang misalkan bidang fisika, biologi, ekonomi, sosiologi dan masih banyak lagi yang lain (Mahmudy, 2013).

Pada penelitian sebelumnya telah membahas mengenai permasalahan optimasi makanan untuk mengoptimasi asupan gizi pada ibu hamil menggunakan algoritma genetika (Sari, 2014). Pada penelitian ini dilakukan dengan mengkombinasikan kromosom dan dilanjutkan dengan proses reproduksi serta seleksi dengan melakukan inisialisasi terlebih dahulu terhadap parameter genetika yaitu ukuran populasi, *crossover rate*, *mutation rate*, dan jumlah generasi. Ada 150 data yang diuji dengan menggunakan metode *single point crossover*, reciprocal *exchange mutation*, dan *ellitsm selection*. Dari penelitian ini didapatkan hasil terbaik dengan nilai fitness tertinggi yang mendekati kebutuhan gizi ibu hamil pada ukuran populasi 150, jumlah generasi 1500, nilai probabilitas crossover 0,4 dan probabilitas mutasi 0,6 (Sari, 2014).

Karena algoritma genetika telah berhasil diterapkan pada beberapa masalah kompleks, pada skripsi ini algoritma genetika digunakan untuk membuat aplikasi ”Optimasi Komposisi Pakan Kambing Potong menggunakan Algoritma Genetika” dengan harapan aplikasi ini dapat membantu mengoptimalkan komposisi pakan kambing potong untuk membantu proses penentuan komposisi pakan kambing yang tepat dengan biaya yang efisien. Sehingga tidak efisein dan

kesalahan dalam pemberian pakan pada kambing potong dapat dihindari untuk mendapatkan kambing potong yang memiliki kandungan gizi sesuai.

1.2 Rumusan Masalah

1. Bagaimana menerapkan Algoritma Genetika untuk menyelesaikan permasalahan komposisi pakan kambing potong?
2. Bagaimana mengukur kualitas solusi yang dihasilkan?
3. Bagaimana pengaruh parameter algoritma genetika (ukuran populasi, *crossover rate*, *mutation rate* dan banyaknya generasi) terhadap hasil dari optimasi komposisi pakan kambing potong ?

1.3 Batasan Masalah

Batasan masalah yang akan dijadikan sebagai pedoman dalam pelaksanaan penelitian ini adalah sebagai berikut :

1. Jenis kambing yang diteliti adalah kambing potong.
2. Jenis kambing yang diteliti adalah kambing potong jantan dewasa untuk penggemukan.
3. Berat Badan yang digunakan hanya yang tertera yang pada Tabel 2.2.
4. Jenis pakan yang digunakan 30 jenis bahan pakan.
5. Penentuan ransum yang disusun berdasarkan berat badan dan target pertambahan berat badan (bobot) perhari.
6. Nutrisi yang digunakan untuk menyusun pakan adalah bahan kering (bk), protein kering (pk) dan *total digestible nutrient* (tdn).

1.4 Tujuan

Menerapkan algoritma genetika untuk menyelesaikan permasalahn optimasi komposisi pakan kambing potong.

1.5 Manfaat

1. Membantu para peternak kambing potong dalam membuat kombinasi pakan kambing potong yang terbaik.
2. Membantu para peternak kambing potong dalam menekan biaya pakan pembelian bahan pakan.

1.6 Sistematika Penulisan

Sistematika penulisan penelitian ditunjukkan untuk memberikan gambaran dan uraian dari penyusun tugas akhir secara garis besar yang meliputi beberapa bab sebagai berikut :

BAB I Pendahuluan

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

BAB II Dasar Teori

Menjelaskan tentang dasar teori dan referensi yang mendasari pembuatan sistem Optimasi Komposisi Pakan Kambing Potong menggunakan Algoritma Genetika.

BAB III Metode Penelitian

Menjelaskan tentang metode dan langkah kerja yang dilakukan dalam penulisan tugas akhir yang terdiri dari studi literatur, metode pengambilan data, analisis kebutuhan, perancangan sistem, implementasi pengujian dan analisis serta pengambilan kesimpulan.

BAB IV Perancangan

Menjelaskan tentang perancangan databse, perancangan antarmuka dan perancangan uji coba dalam pembuatan perangkat lunak system optimasi komposisi pakan kambing potong yang akan dibuat.

BAB V Implementasi

Membahas implementasi dari sistem Optimasi Komposisi Pakan Kambing Potong menggunakan Algoritma Evolusi sesuai dengan perancangan sistem yang telah dibuat beserta impelemntasi antarmuka.

BAB VI Pengujian dan Analisis

Menjelaskan proses dan hasil dari pengujian terhadap sistem yang telah dibangun serta analisis untuk memastikan bahwa program telah sesuai dengan perancangan.

BAB VII Penutup

Bab ini tediri dari kesimpulan yang diperoleh dari pembuatan dan pengujian perangkat lunak yang dibuat dalam skripsi ini serta saran-saran untuk pengembangan lebih lanjut.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

Kajian pustaka dan dasar teori untuk menunjang skripsi pembuatan aplikasi optimasi komposisi pakan kambing potong menggunakan algoritma genetika ini meliputi kajian pustaka, jenis kambing, bahan pakan kambing potong, kebutuhan nutrisi kambing potong dan algoritma genetika yang meliputi struktur algoritma genetika, parameter genetika dan penerapan algoritma genetika.

2.1 Kajian Pustaka

Pada penelitian sebelumnya mengenai komposisi pakan telah dilakukan dengan berbagai macam metode. Referensi yang digunakan pada skripsi ini adalah penelitian dengan judul "Optimasi komposisi Bahan Pakan Ikan Air Tawar Menggunakan Metode Multi-Objective Genetic Algorithm" oleh Luh Kesuma Wardhani, M. Safrizal dan Achmad Chairi (2011), dengan rata-rata tingkat keberhasilan perhitungan pemenuhan kebutuhan nutrisi ikan dapat mencapai 100% dan tingkat efisiensi biaya pakan sekitar 46,5% untuk menghasilkan pakan sejumlah 6 kg pada kasus ikan lele dewasa.

Penelitian selanjutnya adalah "Penerapan Algoritma Genetika Pada Penentuan Komposisi Pakan Ayam Petelur" oleh Arnold Aribowo, Samuel Lukas, Martin Gunawan pada tahun 2008. Pada peneltian ini didapatkan nilai *fitness* optimum yaitu 0.92076 dicapai dengan jumlah gen 20, jumlah kromosom 1000, *crossover rate* 0,5, *mutation rate* 0,1 dan jumlah generasi ke 1084 dari 4000 generasi. Berikut persamaan dan perbedaan dari kedua penelitian dengan skripsi yang diajukan berdasarkan studi literatur yang telah dilakukan pada Tabel 2.1.

Tabel 2.1 Perbedaan Penelitian Sebelumnya dengan Skripsi Penulis

No	Judul	Penulis	Perbandingan	
			Studi Literatur	Skripsi Penulis
1.	Optimasi komposisi Bahan Pakan Ikan Air Tawar Menggunakan Metode Multi-Objective Genetic Algorithm	Luh Kesuma Wardhani, M. Safrizal, Achmad Chairi	Menggunakan Algoritma Genetika Multi-Objective	Menggunakan Algoritma Genetika dengan representasi real code
2.	Penerapan Algoritma Genetika Pada Penentuan Komposisi Pakan Ayam Petelur	Arnold Aribowo, Samuel Lukas, Martin Gunawan	Menggunakan Algoritma Genetika dengan *Roulette Wheel Selection*	Menggunakan Algoritma Genetika Dengan *Ellitsm Selection.*

Pada skripsi ini membahas tentang optimasi komposisi pakan kambing potong. Dalam skripsi ini digunakan algoritma untuk mengoptimasi bahan pakan yang digunakan menggunakan algoritma genetika. Untuk representasi kromosom yang digunakan adalah menggunakan *real code*. Pada tahap *crossover* metode yang digunakan adalah *extended intermediate*, metode *random mutation* digunakan pada proses mutasi dan pada tahap seleksi menggunakan metode *elitism*. Dasar teori yang digunakan pada skripsi ini adalah pakan dan ransum, kebutuhan nutrisi pakan kambing potong, komposisi bahan pakan, formulasi ransum dan algoritma genetika.

2.2 Jenis Kambing

Kambing adalah hewan ternak ruminansia kecil yang dapat dimanfaatkan daging dan susunya. Kambing terdiri dari 2 jenis yaitu kambing potong dan kambing perah (Gunawan, 2013).

a. Kambing Potong

Kambing potong atau pedaging adalah kambing yang diternak dengan tujuan utama adalah produksi daging. Kambing potong adalah komoditas yang menjamur dalam negeri dalam waktu yang tidak sebentar. Kambing potong memiliki ciri-ciri sedikit lemak dan bulu dan pada umumnya ukuran badannya lebih besar.

b. Kambing Perah

Kambing perah adalah kambing yang diternak dengan tujuan utama adalah produksi susu. Untuk meningkatkan produksi susu kambing, maka dipilih kambing unggul dalam menghasilkan susu.

2.3 Bahan Pakan Kambing

Pemberian pakan yang berkualitas akan berdampak pada pertumubhan kambing untuk tumbuh dengan baik dan menghasilkan produksi yang maksimal. Selain itu pakan dibutuhkan kambing untuk berkembang biak. Kambing adalah hewan ternak ruminansia kecil yang pakan utamanya adalah hijauan. Pakan hijauan tersebut dapat berupa daun nangka, waru, atau yang biasa disebut rambanan, tanaman kacang-kacangan dan rumput. Selain diberi pakan hijauan, kambing juga diberi pakan berupa konsentrat, konsentrat merupakan makanan penguat yang kandungan gizi tinggi. Namun, meskipun konsentrat memiliki gizi yang baik namun konsentrat tidak bisa dijadikan makanan tunggal karena dapat menyebabkan gangguan pencernaan pada kambing yang mengakibatkan hewan ternak mati. Sehingga pakan kambing yang baik adalah perpaduan antara hijauan dan konsentrat (Susilawati, 2011).

Bahan pakan untuk domba dan kambing pada umumnya di golongkan dalam 4 golongan sebagai berikut (Cahyono, 2006) :

1. Golongan rumput-rumputan
   a. rumput gajah
   b. rumput setaria
   c. rumput benggala
   d. rumput brachiaria
   e. rumput raja
   f. rumput meksiko
   g. rumput alam
2. Golongan kacang-kacangan atau leguminose
   a. daun lamtoro
   b. daun turi
   c. daun gamal
   d. kaliandra
   e. albesia
   f. gliricidia
   g. siratro
   h. daun kacang tanah
   i. daun kacang-kacangan
   j. daun kacang

3. Hasil Limbah Pertanian

a. daun nangka
b. daun waru
c. daun dadap
d. daun kembang sepatu
e. daun pisang
f. daun jagung
g. daun ketela pohon
h. daun ketela rambat
i. daun beringin

4. Golongan makanan penguat (konsentrat)

a. Dedak
b. jagung kering
c. garam dapur
d. bungkil kelapa
e. tepung ikan
f. bungkil kedelai
g. ampas tahu
h. ampas kecap

2.4 Kebutuhan Nutrisi Pakan Kambing Potong

Beberapa bahan makanan memiliki komposisi yang beragam. Sampai saat ini tidak ada yang dapat membantah bahwa analisi dari suatu bahan makanan yang akan digunakan dalam suatu ransum akan lebih akurat bila menggunakan data yang disusun dalam sebuah tabel. Berikut merupakan nutrisi yang dibutuhkan oleh kambing potong :

1. Bahan Kering (BK)

Bahan pakan yang diberikan pada ternak memiliki kadar air yang bervariasi. Bahan kering atau kadar air mengatur konsumsi sehingga menyusun ransum untuk ternak lebih baik dihitung berdasarkan bahan kering (kadar airnya).

2. Protein Kering (PK)

Protein *Nutrien* yang terdiri dari satu atau lebih ikatan asam amino. Protein ini disebut juga polypeptide sebab beberapa asam amino saling berikatan dalam ikatan peptide. Yang diperlihatkan dalam tabel komposisi adalah protein kasar (Prk).

3. TDN (*Total Digestible Nutrient*)

Pada ternak ruminansia dikenal istilah *Total Digestible Nutrient* (TDN), yaitu suatu asumsi bahwa selisih antara zat gizi yang dikonsumsi dengan zat gizi yang terdapat di dalam faeces merupakan nilai zat gizi yang tercerna dan dapat diubah menjadi enersi. TDN adalah cara/ data yang banyak tersedia dan

telah(pernah) merupakan suatu standar untuk meyatakan nilai energy suatu bahan makanan untuk ruminant (Parakkasi, 1999).

Berikut adalah tabel kebutuhan nutrisi kambing berdasarkan bobot badan dan pertambahan bobot badan (Kearl, 1982 dalam Kusumaningrum, 2009).

Tabel 2.2 Kebutuhan Nutrisi Kambing Berdasarkan Bobot Badan dan Pertambahan Bobot Badan

BB (kg)	PBB (gr)	BK (kg)	TDN (kg)	PK (gr)	Ca (gr)	P (gr)
10	0	0,32	0,16	17	0,9	0,7
	25	0,36	0,21	22	1,2	0,9
	50	0,37	0,25	26	1,5	1,2
	75	0,35	0,30	31	1,9	1,5
15	0	0,44	0,22	23	1,2	0,9
	25	0,45	0,24	25	1,5	1,1
	50	0,50	0,31	33	1,9	1,4
	75	0,50	0,36	37	2,2	1,7
20	0	0,54	0,27	28	1,5	1,1
	25	0,58	0,32	33	1,8	1,3
	50	0,60	0,36	38	2,1	1,6
	75	0,62	0,41	43	2,4	1,9
	100	0,62	0,41	43	2,4	1,9
25	0	0,64	0,32	33	1,8	1,3
	25	0,68	0,37	38	2,1	1,5
	50	0,71	0,41	43	2,4	1,8
	75	0,73	0,46	48	2,7	2,1
	100	0,74	0,51	53	3,1	2,3
30	0	0,74	0,37	38	2,1	1,5
	25	0,77	0,41	43	2,4	1,7
	50	0,80	0,46	48	2,7	2,0
	75	0,83	0,51	53	3,1	2,3
	100	0,84	0,56	58	3,4	2,5
	125	0,84	0,60	63	3,7	2,7
40	0	0,91	0,46	48	2,5	1,9
	25	0,95	0,50	53	2,8	2,1
	50	0,98	0,55	58	3,1	2,4
	75	1,01	0,60	62	3,5	2,7
	100	1,04	0,65	67	3,8	2,9
	125	1,05	0,69	72	4,1	3,1

Sumber : Kearl dalam Kusumaningrum, 2009

Adapun juga tabel kandungan nutrisi bahan ransum pakan seperti dibawah ini.

Tabel 2.3 Kandungan Nutrisi Bahan Rasum Pakan Ruminansia

No	Bahan Pakan	BK	PK	TDN	Harga
1	Gamal	27	25.2	76	2500
2	Kaliandra	16	27.7	62	2000
3	Lamtoro_K	86	23.7	71	3000
4	Lamtoro_S	29	23.4	77	2800
5	R_Benggala	24	5.4	53	3000
6	R_Gajah	18	9.1	51	3400
7	R_Raja	22	13.5	54	3500
8	Tebon_Jagung	22	8	58	2500
9	R_Pangola	23	8.3	53	2700
10	R_Setaria	20	9.5	55	3400
11	J_Padi	86	3.7	39	3300
12	J_Kacang	86	14.7	56	3000
13	J_Kedele	86	19.1	76	1600
14	D_Pisang	16	14.4	70	3500
15	D_Singkong	15	25	18	1400
16	A_Nanas	20	3.4	68	3600
17	A_Tahu	16.2	23.7	78	2500
18	A_Sagu	80.4	1.2	58	3200
19	BijiKapas	86	22.1	74.3	3400
20	B_Kelapa	86	21.6	73	3000
21	B_BijiSawit	86	15	70	3600
22	DPadi_Kasar	86	7.6	14	4000
23	DPadi_Halus	86	13.8	81	3700
24	K_Coklat	88.9	14.6	47	5000
25	Jg_Dedak	86	11.3	81	2800
26	Jg_putih	86	10	81	2550
27	Jg_Kuning	86	10.3	80	2600
28	BijiKapuk_Tepung	86	31.7	74	4650
29	Onggok	28.7	1.2	69	2000
30	Tetes	77	5.4	53	1450

Sumber : Dispet Jatim 2002

2.5 Ransum

Ransum merupakan satu atau beberapa jenis bahan pakan yang diberikan untuk seekor ternak selama sehari semalam. Ransum harus dapat memenuhi zat-zat makanna yang dibutuhkan seekor ternak untuk berbagai fungsi tubuhnya, seperti pokok hidup, produksi, maupun reproduksi (Siregar, 1996).

Kegunaan dari formulasi ransum adalah untuk menuangkan pengetahuan tentang zat atau beberapa zat makanan, bahan/beberapa bahan makanan menjadi suatu bahan makanan (ransum) yang dapat memenuhi kebutuhan ternak yang memepunyai tingkat produksi tertentu yang dikehendaki oleh peternak (Parakkasi, 1999). Adapun langkah-langkah dalam penyusunan ransum adalah (NN, 2013):

1. Menentukan kebutuhan nutrisi ternak.
2. Menentukan bahan makanan yang akan digunakan, yaitu :
   - Jenis bahan pakan yang tersedia
   - Kandungan nutrisinya
   - Harga bahan pakan
3. Memformulasikan berbagai bahan untuk memenuhi kebutuhan ternak dengan teknik perhitungan tertentu.
4. Melakukan pemeriksaan kembali terhadap hasil perhitungan disesuaikan dengan kebutuhan ternak dihubungkan dengan status fisiologisnya.
5. Menyiapkan ransum tersusun sesuai dengan kondisi dan kebutuhan (NN, 2013).

Berikut adalah contoh menyusun ransum untuk kambing dengan bobot badan 30 kg, target pertambahan berat badan 0,1 kg, maka kebutuhan bahan kering (bk) adalah 0,84 kg, protein kering (pk) =0,058 kg dan *total digestible nutrient* (tdn) =0,56 kg. Bahan pakan yang tersedia yaitu kaliandra, rumput gajah dan jagung putih. Jagung putih yang akan digunakan untuk memenuhi kebutuhan bahan kering (bk) 10% dari keseluruhan ransum , sehingga bahan kering (bk) jagung putih adalah (10/100) x 0,84= 0,084 kg bahan kering.

Kadungan protein jagung putih = (10/100) x 0,084 = 0,0084 kg protein. Untuk menyusun ransum dengan kebutuhan bahan kering (bk) sebesar 0,84 kg

dan protein sebesar 0,58 kg masih masih kekurangan sehingga perhitungannya sebagai berikut :

BK = 0,84 – 0,084 = 0,756 kg

Protein = 0,058 – 0,0084 = 0,0496 gram atau (0,0496/0,84) x 100 % = 5.9%. Kekurangan harus dipenuhi dari rumpur gajah dan konsentrat dengan perhitungan sebagai berikut (Edy, 2015):

Kaliandra

27,7 33,6 (33,6/48,6) x 100% = 69,1%

5,9

Rumput Raja 15 (15 /48,6) x 100% = 30,9

9,1 +

48,6

Keterangan :

48,6 didapatkan dari hasil penjumlahan 33,6 dengan 15, begitu juga dengan 33,6 dihasilkan dari penjumlahan dari 27,7 dengan 5,9. Dan 15 dihasilkan dari penjumlahan antara 9,1 dengan 5,9 (Edy, 2015).

- Jumlah bahan kering (BK) yang tersedia :

Jagung Putih $= \frac{10}{100}$ X 840 = 84 gram = 0,084 kg

Kaliandra $= \frac{69,1}{100}$ X 756 = 522,396 gram = 0, 522396 kg

Rumput Gajah $= \frac{30,9}{100}$ X 756 = 233,848 gram = 0,233848 kg

- Jumlah kandungan protein (PK) yang tersedia :

Jagung Putih $= \frac{10}{100}$ X 84 = 8,4 gram = 0,0083 kg

Kaliandra $= \frac{27,7}{100}$ X 522,396 = 144,7 gram = 0,1447 kg

Rumput Gajah $= \frac{9,1}{100}$ X 233,84 = 21,3 gram = 0,0213 kg

- Jumlah kandungan TDN yang tersedia :

Jagung Putih $= \frac{81}{100}$X 84 = 68,04 gram = 0,06804 kg

Kaliandra $= \frac{62}{100}$ X 522,396 = 323,885 gram = 0,3239 kg

Rumput Gajah $= \frac{51}{100}$ X 488,3 = 249,033 gram = 0,2490 kg

2.6 Algoritma Genetika

Algoritma genetika adalah salah satu algoritma evolusi yang popular dalam menyelesaikan masalah optimasi kompleks. Berkembangnya teknologi informasi membuat algoritma genetika semakin berkembang. Algoritma genetika semakin banyak digunakan dalam berbagai bidang seperti fisika sampai sosiologi karena kemampuannya dalam menyelesaikan masalah sederhana sampai kompleks (Mahmudy, 2013). Algoritma genetika ini merupakan metode pencarian heuristik yang didasarkan pada proses alamiah, proses penyelesaiannya sama seperti individu yang berusaha hidup dalam proses evolusi. Prosesnya tersebut dimulai dengan terbentuknya populasi awal secara acak yang terdiri dari individu, kemudian individu tersebut melakukan proses reproduksi untuk menghasilkan anak yang mewarisi sifat-sifat induknya (Turban, 1995 dalam Zukhri, 2014).

Dalam bidang industry manufaktur, algoritma genetika digunakan untuk perencanaan penjadwalan produksi. Algortima genetika juga bisa diterapkam untuk optimasi penugasan mengajar bagi dan masih banyak lagi contoh implementasi algoritma genetika dalam berbagai bidang di kehidupan sekitar kita (Mahmudy, 2013).

2.6.1 Struktur Algoritma Genetika

Proses algoritma genetika berawal dari proses inisialisasi, proses inisialisasi merupakan prosen untuk mebuat individu baru secara acak yang mempunyai kromosom tertentu dimana kromosom tersebut menjadi salah satu solusi untuk masalah yang akan diselesaikan. Setelah melakukan proses insialisasi, selanjutnya melakukan proses reproduksi yang berfungsi untuk menghasilkan keturunan (*offspring*). Proses reproduksi terdiri dari 2 operator genetika yaitu tukar silang (*crossover*) dan mutasi (*mutation*). Setelah melakukan

proses reproduksi dilanjutkan dengan proses evaluasi, dimana proses evaluasi ini berfungsi untuk menghitung *fitness*. Setiap kromosom mempunyai *fitness*, jadi semakin besar *fitness* maka semakin baik kromosom tersebut untuk dijadikan suatu solusi. Langkah selanjutnya adalah proses seleksi, proses seleksi dilakukan untuk memilih individu dari himpunan populasi dan *offspring* yang dipertahankan hidup pada generasi selanjutnya (Mahmudy, 2013). Apabila P(t) dan C(t) merupakan populasi (*parents*) dan *offspring* pada generasi ke-t, maka struktur umum algoritma genetika dapat dideskripsikan sebagai berikut (Gen & Cheng 1997 dalam Mahmudy, 2013).

```
Procedure AlgoritmaGenetika
Begin
t = 0
inisialisasi P(t)
while (bukan kondisi berhenti) do
     reproduksi C(t) dari P(t)
     evaluasi P(t) dan C(t)
          seleksi P(t+1) dari P(t) dan C(t)
     end while
```

Gambar 2.1 Pseudocode Algoritma Genetika

2.6.2 Parameter Algoritma Genetika

Algoritma genetika bekerja berdasarkan parameter – parameter tertentu yang akan mempengaruhi kinerja dan perilaku dari algoritma ini. Parameter penting yang mempengaruhi performa algoritma genetika yaitu :

1. Fungsi *fitness* yang dimiliki oleh masing-masing individu.
2. Jumlah populasi (*popSize*) yang diguanakn pada setiap generasi.
3. *Crossover rate* yang terjadi pada proses *crossover* pada setiap generasi.
4. *Mutation rate* yang terjadi pada proses mutasi pada setiap generasi.
5. Jumlah generasi yang digunakan untuk menentukan lama penerapan algoritma genetika (Desiani,2006).

Rekomendasi untuk menentukan nilai parameter adalah sebagai berikut :

- Untuk permasalahan yang memiliki kawasan solusi cukup besar, De Jong merekomendasikan nilai parameter (*popsize*; pc; pm) = (50;0.6;0.001).

- Bila rata – rata *fitness* setiap generasi digunakan sebagai indicator, maka Grenfenstette merekomendasikan (*popsize*; pc; pm) = (80;0.45;0.01).
- Kombinasi parameter yang sesuai (Mahmudy, Marian, Luong 2104). Ukuran populasi (*popSize*) antara 30 sampai 50, pc antara 0.3 sampai 0.8, dan pm antara 0.1 sampai 0.3.

2.6.3 Penerapan Algoritma Genetika

2.6.3.1 Membangkitkan Populasi Awal

Langkah pertama yang dilakukan adalah membangun populasi awal. Populasi awal dibangkitkan secara acak sehingga didapatkan solusi awal. Populasi ini terdiri dari sejumlah kromosom dan untuk membangkitkan populasi dibutuhkan *popSize* (ukuran populasi) untuk menenentukan jumlah individu.

2.6.3.2 Representasi Kromosom

Representasi kromosom merupakan proses pengkodean dari penyelesaian asli suatu permasalahan (Mahmudy, 2013). Pengkodean merupakan kunci pokok persoalan menggunakan algoritma genetika. Proses pengkodean sangat berkaitan dengan peran kromosom sebagai representasi penyelesaian masalah. Pada prinsipnya tidak ada aturan khusus tentang pengkodean dalam proses algoritma genetika ini, kromosom dapat dibuat dengan kode tertentu dan syarat bisa diproses oleh operator genetika serta merupakan representasi penyelesaian masalah (Zukhri, 2014).

Metode pengkodean diklasifikasikan berdasarkan jenis symbol yang digunakan sebagai nilai suatu gen menurut Gen dan Cheng pada tahun 2000 dalam buku Desiani tahun 2006 yaitu sebagai berikut : pengkodean biner, pengkodean riil, bilangan bulat dan struktur data. Pada contoh persoalan ini menggunakan pengkodean riil, pengkodean riil adalah berbetuk riil. Pengkodean riil tepat sekali jika digunakan pada masalah optimalisasi fungsi dan optimalisasi kendala (Desani, 2006).

2.6.3.3 Inisialisasi

Populasi awal untuk inisialisasi dibangkitkan secara acak. Misalkan ditentukan popSize = 10 maka akan dihasilkan populasi sebagai berikut :

Tabel 2.4 Tabel Inisialisasi

Parents	Chromosome		Fitness
	x1	x2	fx(x1, x2)
P1	1,4898	2,0944	19,8206
P2	8,4917	2,5754	34,7058
P3	1,4054	6,3035	20,6707
P4	5,8114	5,0779	14,5624
P5	-1,8461	1,7097	11,5858
P6	4,0206	4,4355	24,7106
P7	-0,1634	2,974	19,653
P8	5,2742	0,7183	22,1813
P9	9,4374	6,6919	12,4694
P10	-4,5575	0,1679	28,4324

Sumber : (Mahmudy, 2013)

2.6.3.4 Reproduksi

Proses reproduksi ini dilakukan untuk menghasilkan keturunan (*offspring*) dari individu-individu yang ada di populasi. Pada proses reproduksi digunakan 2 operator genetika yaitu adalah tukar silang (*crossover*) dan mutasi (*mutation*). Metode *crossover* dan mutasi yang telah dikembangkan oleh para peneliti yang bersifat spesifik terhadap masalah dan pengkodean sangat beragam (Mahmudy, 2013).

a. Crossover

Metode *crossover* yang digunakan dalam kasus ini adalah *extended intermediate crossover* yang menghasilkan *offspring* dari kombinasi nilai dua induk. Kombinasi dua *parent* untuk menghasilkan *offspring* tersebut dipilih secara acak dari populasi. Misalkan P1 dan P2 adalah dua induk yang telah diseleksi untuk melakukan *crossover*, maka *offspring* C1 dan C2 bisa dibangkitkan sebagai berikut:

$$C1 = P1 + \propto (P2 - P1) \quad (2\text{-}1)$$

$$C2 = P2 + \propto (P1 - P2) \quad (2\text{-}2)$$

α dipilih secara acak pada interval [-0.25, 1.25].

Misalkan yang terpilih sebagai induk adalah P4 dan P9 a=[0.1104 , 1.2336] maka akan dihasilkan dua *offspring* (C1 dan C2) sebagai berikut (Mahmudy, 2013):

C1 : x1 = 5,8114 + 0,1104 (9,4374 - 5,8114) = 6,2118

x2 = 5,0779 + 1,2336 (6,619 - 5,0779) = 7,0690

C2 : x1 = 9,4374 + 0,1104 (5,8114 – 9,4374) = 9,0370

x2 = 6,6919 + 1,2336 (5,0779 – 6,6919) = 4,7008

Jika ditentukan *crossover rate* =0,4 maka ada 0,4 x 10 = 4 *offspring* yang dihasilkan. Karena setiap *crossover* menghasilkan dua anak maka ada dua kali proses *crossover* menggunakan induk yang berbeda. Anggap dua *offspring* berikutnya adalah C3 dan C4 (Mahmudy, 2013).

b. Mutasi

Mutasi digunakan sebagai operator untuk menjaga keragaman populasi dan untuk membuat individu baru dengan memodifikasi satu atau lebih gen dalam satu individu yang sama (Mahmudy, 2013). Fungsi dari mutasi dalam algoritma genetika untuk menggantikan gen yang hilang dari populasi selama proses seleksi serta menyediakan gen yang tidak ada dalam populasi awal. Mutasi akan meningkatkan variasi populasi.

Metode mutasi yang digunakan dalam kasus ini adalah *random mutation* yang menghasilkan *offspring*(anak) yang dilakukan dengan menambah atau mengurangi nilai gen terpilih dengan bilangan random yang kecil. Mutasi dilakukan dengan memilih satu induk secara acak dari populasi. Misalkan domain variabel x_j adalah [min_j, max_j] dan *offspring* yang dihasilkan adalah C=[$x'_1..x'_n$], maka nilai gen *offspring* bisa dibangkitkan sebagai berikut:

$$x_i' = x_i' + r(max_i - min_j) \quad (2\text{-}3)$$

range *r* misalkan [-0,1, 0,1].

Misalkan yang terpilih sebagai induk adalah P2, gen yang terpilih nomer 2 (x_2) dan r = -0,0584. Maka akan dihasilkan *offspring* (C5) sebagai berikut:

C5 : x1 = 8,4917 (tetap)

x2 = 2,5754 – 0,0584(7,3-0,0) = 2,1491

Misal kita tentukan mr=0,2 maka ada 0,2x10=2 *offspring* yang dihasilkan dari proses mutasi. Anggap *offspring* berikutnya adalah C6. Keseluruhan *offspring* yang dihasilkan dari proses reproduksi (*crossover* dan mutasi) adalah sebagai berikut :

Tabel 2.5 Tabel Hasil *Crossover* dan Mutasi

Parents	**Chromosome**		**Fitness**
	x1	**x2**	**fx(x1, x2)**
C1	6,2118	7,0690	22,2048
C2	9,0370	4,7008	22,2313
C3	7,1636	0,0000	15,4774
C4	7,5479	7,3000	9,3531
C5	8,4917	2,1494	31,0389
C6	-1,1238	1,7097	12,0177

Sumber : (Mahmudy, 2013)

Perhatikan bahwa sekarang kita mempunyai 16 individu (10 dari populasi mula-mula ditambah 6 *offspring*) (Mahmudy, 2013).

2.6.3.5 Perhitungan *Fitness*

a. Penalty

Penalty merupakan nilai yang digunakan pada saat suatu individu melakukan suatu pelanggaran terhadap aturan (Tyas, 2013). Aturan yang dimaksud dalam penelitian ini adalah nutrisi yang digunakan untuk memenuhi kebutuhan pakan kambing berdasarkan Tabel 2.2 dan Tabel 2.3. *Penalty* dihitung pada saat kondisi tertentu yaitu kondisi dimana kebutuhan nutrisi rekomendasi dari sistem kurang dari kebutuhan nutrisi dari pakar dan buku. Perhitungan *pinalty* ditunjukkan pada persamaan (2-4) berikut (Sari, Mahmudy, Dewi 2014).

$$Penalty = \alpha 1(Penalty1) + \alpha 2(Penalty2) + \alpha 3(Penalty3) \quad (2\text{-}4)$$

Dimana :

Penalty1 = nilai *pinalty* BK

Penalty2 = nilai *pinalty* PK

Penalty3 = nilai *pinalty* TDN

$\alpha 1$ = nilai prioritas untuk BK

$\alpha 2$ = nilai prioritas untuk PK

$\alpha 3$ = nilai prioritas untuk TDN

Fungsi penalti untuk setiap kebutuhan nutrisi ditunjukkan pada persamaan (2-5), (2-6) dan (2-7) berikut ini.

$$PenaltiBK = \begin{cases} 0, & TBK\ sistem \geq TBK\ pakar \\ TBK\ pakar - TBK\ sistem, & TBK\ sistem < TBK\ pakar \end{cases} \quad (2\text{-}5)$$

$$PenaltiPK = \begin{cases} 0, & TPK\ sistem \geq TPK\ pakar \\ TPK\ pakar - TPK\ sistem, & TPK\ sistem < TPK\ pakar \end{cases} \quad (2\text{-}6)$$

$$PenaltiTDN = \begin{cases} 0, & TTDN\ sistem \geq TTDN\ pakar \\ TTDN\ pakar - TTDN\ sistem, & TTDN\ sistem < TTDN\ pakar \end{cases} \quad (2\text{-}7)$$

Keterangan :

TBK = Total BK

TPK = Total PK

TTDN = Total TDN

b. Fitness

Algoritma genetika dapat menyelesaikan masalah kompleks salah satunya yaitu permasalahan optimasi, untuk mendapatkan nilai *fitness*, suatu individu dievaluasi berdasarkan suatu nilai tertentu. Jika permasalahannya adalah optimasi untuk memaksimalkan fungsi *h*, maka digunakan fungsi *f = h* untuk mencari nilai *fitness*nya. Tapi jika permasalahannya adalah optimasi untuk meminimalkan fungsi *h*, maka digunakan fungsi *f = 1/h* untuk mendapatkan nilai *fitness*nya. Namun jika *h* bernilai 0 akan mengakibatkan *f* bernilai tak terhingga. Sehingga *h* pada fungsi *f = 1/h* perlu ditambah dengan bilangan yang dianggap kecil. Sehingga rumus *fitness* ditunjukkan pada Persamaan (2-8) (Kushardiana, 2013).

$$f = \frac{1}{(h+a)} \quad (2\text{-}8)$$

Dimana :

- *a* adalah bilangan kecil yang sesuai dengan permasalahan yang dihadapi.
- *h* adalah sebuah fungsi yang akan diminimalkan (dalam kasus ini adalah harga).
- *f* adalah fungsi *fitness*

Dalam penelitian ini, fungsi yang akan diminimalkan adalah harga dengan menghitung nilai *Penalty*. Sehingga fungsi *fitness* yang digunakan dalam penelitian ini ditunjukkan pada Persamaan (2-9) (Tyas, 2013)

$$f = \frac{1}{total\ harga + penalty} \quad (2\text{-}9)$$

Dimana :

- Total harga diperoleh dengan mengalikan nilai bahan pakan setiap kilogram dengan harga pakan.
- *Penalty* adalah nilai penalti yang dihitung jika kebutuhan nutrisi rekomendasi system kurang dari kebutuhan nutrisi oleh pakar dan buku.

2.6.3.6 Seleksi

Proses seleksi merupakan proses memilih individu dari himpunan populasi dan *offspring* yang dipertahankan pada generasi berikutnya (Mahmudy, 2013). Tahap awal untuk melakukan seleksi yaitu pencarian nilai *fitness.* Dengan mengetahui nilai *fitness* maka proses seleksi siap untuk dilakukan, karena semakin tinggi nilai *fitness* suatu individu maka semakin besar kesempatan individu tersebut untuk terpulih dan dipertahankan di generasi selanjutnya.

Metode seleksi yang sering digunakan adalah *roulette wheel, binary tournament, dan elitism*. Metode yang digunakan dalam penyelesaian kasus ini adalah *elitism selection*. Seleksi dengan metode *elitism selection* merupakan proses seleksi yang memilih individu yang memiliki *fitness* tertinggi untuk dipertahankan ke generasi selanjutnya. Metode *elitism selection* dilakukan dengan mengumpulkan semua individu baik *parent* maupun *offspring,* kemudian barulah diseleksi dengan melihat nilai *fitness* tertingginya untuk dipertahankan ke generasi selanjutnya. Metode seleksi *elitism* menjamin individu yang terbaik akan selalu lolos (Mahmudy, 2013).

Pseudo-code seleksi elitism dideskripsikan sebagai berikut (Mahmudy, 2013):

```
PROCEDURE ElitismSelection
Input:
POP:  himpunan individu pada populasi
pop_size:  ukuran populasi
OS:  himpunan  individu  anak  (offspring)  hasil
     reproduksi menggunakan crossover dan mutasi
Output :
POP:  himpunan  individu  pada  populasi  setelah  proses
seleksi selesai
     /* gabungkan individu pada POP dan OS ke dalam TEMP */
     TEMP <- Merge (POP,OS)
     /* urutkan individu berdasarkan fitness secara ascending */
     OrderAscending (Temp)
     /* copy pop_size individu terbaik ke POP */
     POP <- CopyBest (Temp, pop_size)

END PROCEDURE
```

Gambar 2.2 Pseudocode Seleksi Elitism

Selengkapnya hasil seleksi ini ada pada Tabel 2.9.

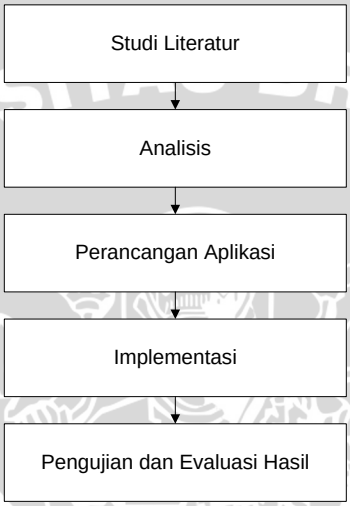
Tabel 2.6 Individu Baru Hasil Seleksi Elitism

P (t+1)	Asal P(t)	Fitness
P1	P2	34,7058
P2	C5	31,0389
P3	P10	28,4324
P4	P6	24,7106
P5	C2	22,2313
P6	C1	22,2048
P7	P8	22,1813
P8	P3	20,6707
P9	P1	19,8206
P10	P7	19,653

Kelemahan metode seleksi *elitism* adalah metode ini tidak mengijinkan individu dengan *fitness* rendah untuk masuk ke populasi berikutnya. Dalam beberapa kasus solusi optimum dicapai pada individu dengan *fitness* rendah (Mahmudy, 2013).

BAB III
METODOLOGI PENELITIAN

Pada bab metode penelitian ini akan dibahas langkah-langkah yang digunakan dalam pembuatan aplikasi optimasi komposisi pakan kambing potong dengan menggunakan algoritma genetika. Adapun langkah-langkahnya dapat dilihat pada Gambar 3.1.



Gambar 3.1 Metodologi Penelitian

Berdasarkan Gambar 3.1, dapat dilihat diagram metodologi penelitian, sehingga penjelasan dari setiap tahapnya sebagai berikut :

1. Melakukan studi literatur tentang algortima genetika dan penyusunan atau formulasi pakan (ransum) pada kambing potong yang akan dibuat dan dibahas pada skripsi ini.
2. Menganalisa kebutuhan system dalam permasalahan optimasi komposisi pakan kambing potong menggunakan algoritma genetika.
3. Merancang sistem yang akan dibuat untuk menyelesaikan permasalahan optimasi komposisi pakan kambing potong menggunakan algoritma genetika.
4. Membuat sistem (implementasi) berdasarkan analisa dan perancangan yang telah dilakukan.
5. Melakukan pengujian dan evaluasi hasil terhadap hasil yang diperoleh dari uji coba tersebut.

3.1 Studi Literatur

Mempelajari literatur dari beberapa bidang ilmu yang berhubungan dengan pembuatan aplikasi optimasi komposisi pakan kambing potong dengan menggunakan algoritma genetika, diantaranya:

a. Algoritma Genetika
b. Pakan (Ransum) Kambing Potong
c. Penyusunan Ransum Kambing Potong

Literatur tersebut diperoleh dari buku, jurnal, website, e-book dan penelitian sebelumnya.

3.2 Analisa Kebutuhan Sistem

Analisa kebutuhan sistem merupakan tahapan menganalisa hal yang dibutuhkan untuk membangun sistem sehingga sistem dapat berjalan sesuai dengan tujuan yang diinginkan.

3.2.1 Deskripsi Umum Sistem

Sistem yang akan dibangun pada penelitian ini adalah sistem yang mengimplementasikan algoritma genetika ke dalam permasalahan penyusunan komposisi bahan pakan pada kambing potong. Menyusun ransum kambing adalah pekerjaan yang tidak mudah dilakukan sehingga untuk mendapatkan kombinasi bahan pakan dengan nutrisi sesuai dengan harga minimal dibutuhkan perhitungan menggunakan algoritma genetika. Terdapat beberapa bahan makanan dari Dispet Jatim 2002 dan Tabel NRC 1994 yang menjadi patokan kebutuhan nutrisi dan kandungan nutrisi dalam skripsi ini. Algoritma Genetika diharapkan dapat memberikan solusi terbaik dalam pembuatan aplikasi optimasi komposisi pakan kambing potong. Dengan menghasilkan komposisi bahan makanan yang memiliki kandungan nutrisi seimbang serta harga bahan pakan yang minimal.

3.2.2 Data yang Digunakan

Data yang digunakan dalam penelitian ini adalah sebagai berikut :

- Data bahan makanan dari Dispet Jatim 2002.

- Data harga harga makanan didapatkan dengan melakukan survey ke beberapa tempat yang menjual bahan makanan di Kota Malang pada bulan November 2014.

3.2.3 Kebutuhan Perangkat

Pada bab ini akan dibahas tentang kebutuhan sistem apakah sudah menghasilkan tujuan yang diinginkan. Kebutuhan perangkat dalam sub bab ini akan membahas tentang perangkat keras (*hardware*) dan perangkat lunak (*software*) yang digunakan dalam sistem yang dibuat.

3.2.3.1 Perangkat Keras

Perangkat keras yang digunakan dalam pembuatan system ini adalah sebagai berikut :

1. Prosesor Intel ® Core ™ i5-3337U 1.80 GHz
2. RAM 3.89 GB
3. Harddisk 350 Gb
4. Monitor 14”

3.2.3.2 Perangkat Lunak

Perangkat lunak yang digunakan dalam pembuatan system ini adalah sebagai berikut :

1. Operating System Windows 8.1 64 bit
2. Visual Studio Professional 2013
3. Bahasa Pemrograman C#
4. Microsoft Office 2010
5. Database MySql
6. Microsoft Visio 2007

3.3 Formulasi Permasalahan

Pada sub bab formulasi permasalahan menjelaskan tentang permasalahan yang akan diselesaikan. Permasalahan dalam penelitian ini yaitu bagaimana menyusun komposisi pakan kambing potong agar nutrisi terpenuhi dan biaya minimal menggunnakan algoritma genetika. Dalam proses menyusun komposisi pakan kambing potong agar nutrisi terpenuhi dan biaya minimal karena

penyusunan pakan kambing mempengaruhi kualitas kambing yang akan digemukkan. Untuk menyelesaikan masalah komposisi pakan kambing potong dengan algoritma genetika adapun diberikan contoh kasus seperti berikut :

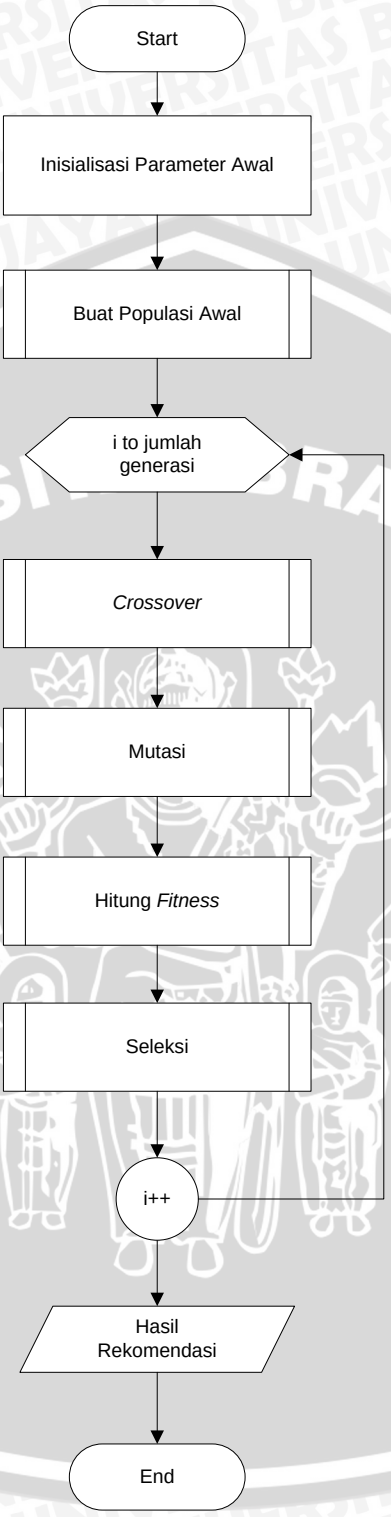
Sebuah kambing dengan berat badan 30 kg akan dinaikkan berat badannya sebesar 0,1 kg, bagaimana menyusun komposisi pakan kambing potong tersebut agar nutrisinya terpenuhi dan harganya minimal?

Langkah pertama yang dilakukan adalah melihat bahan pakan yang tersedia (disesuaikan dengan Tabel 2.3). Dalam contoh permasalah ini bahan pakan yang tersedia yaitu kaliandra, rumput raja dan dedak padi kasar. Lalu melihat kandungan nutrisi bahan kering (bk), protein kering (pk) dan dan *total digestible nutrient* (tdn) nya dari kambing potong tersebut. Dengan melihat Tabel 2.2 berdasarkan berat bobot kambing dan target pertamabahan berat bobot kambing makan kebutuhan yang dibutuhkan adalah bahan kering (bk) = 0,84 kg, dan *total digestible nutrient* (tdn) = 0,56 kg, protein kering (pk) = 0,058 kg .

3.4 Siklus Penyelesaian Masalah Menggunakan Algoritma Genetika

Untuk menyelesaikan permasalahan optimasi komposisi pakan kambing potong diatas maka adapun tahap-tahapnya akan dijelaskan pada Gambar 3.2.

Parameter yang digunakan pada contoh ini yang meliput: *population size* (nilai yang menyatakan banyaknya individu/kromosom yang ditampung dalam populasi), *crossover rate* (nilai yang menyatakan rasio *offspring* yang dihasilkan proses *crossover* terhadap ukuran populasi sehingga akan dihasilkan *offspring* sebanyak *crossover rate* x *popSize*) dan *mutation rate* (nilai yang menyatakan rasio *offspring* yang dihasilkan proses mutasi terhadap ukuran populasi sehingga akan dihasilkan *offspring* sebanyak *mutation rate* x *popSize*. Untuk memperoleh komposisi bahan pakan yang memenuhi nutrisi seimbang dan harga minimal, langkah pertama adalah menginputkan berat bobot kambing, bahan pakan yang dipilih serta target pertambahan berat bobot (pbb). Kemudian sistem akan mengambil data kebutuhan nutrisi yang diperlukan dari tabel kebutuhan kambing sesuai berat badan dan targer bobot yang telah diinputkan. Setelah mengetahui nutrisi yang dibutuhkan kambing tersebut maka selanjutnya sistem akan melihat kandungan dari setiap bahan pakan, kemudian barulah disusun ransumnya.



Gambar 3.2 Flowchart Formulasi Ransum

Setelah mendapatkan berapa banyak kebutuhan makanan kambing, maka tahap berikutnya adalah melakukan inisialisasi awal. Jumlah bahan pakan yang

dipilih oleh user akan menjadi kromosom, kemudian dibangkitkan populasi sebanyak parameter yang dimasukkan. Proses reproduksi dilakukan dengan *crossover* dan mutasi. *Offspring* yang dihasilkan merupakan hasil kali *popSize* dengan *crossover rate* dan juga hasil kali *popSize* dengan *mutation rate*. Selanjutnya individu dari populasi awal dan *offspring* hasil *crossover* dan mutasi digabungkan untuk proses seleksi. Metode seleksi yang digunakan adalah seleksi *ellitsm*. Seleksi dilakukan dengan mengambil sejumlah populasi yang diinputkan dengan melihat besar *fitness* nya. Sehingga invidu yang memiliki *fitness* terbesar akan menjadi individu baru untuk generasi selanjutnya.

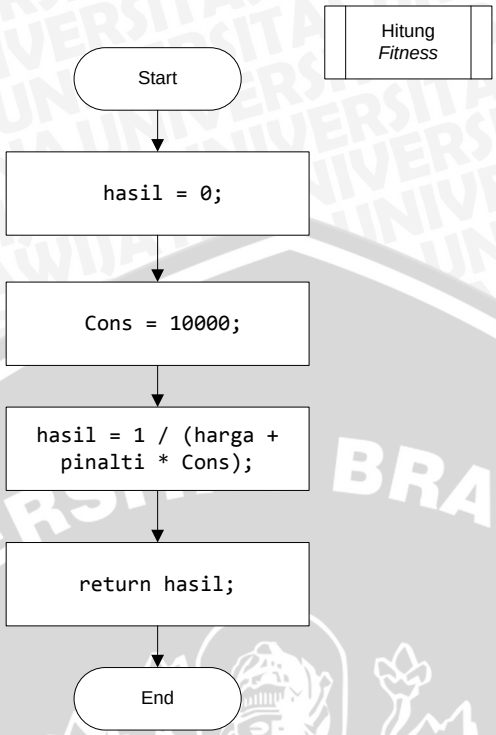
3.4.1 Representasi Kromosom dan Perhitungan Fitness

Representasi kromosom merupakan proses pengkodean dari penyelesaian asli suatu permasalahan (Mahmudy, 2013). Pengkodean merupakan kunci pokok persoalan menggunakan algoritma genetika. Proses pengkodean sangat berkaitan dengan peran kromosom sebagai representasi penyelesaian masalah. Pada prinsipnya tidak ada aturan khusus tentang pengkodean dalam proses algoritma genetika ini, kromosom dapat dibuat dengan kode tertentu dan syarat bisa diproses oleh operator genetika serta merupakan representasi penyelesaian masalah (Zukhri, 2014). Representasi kromosom yang digunakan pada kasus ini adalah *real code* yang dibangkitkan secara acak dengan rang 3 sampai 10. Tabel 3.1 merupakan contoh kromosom.

Tabel 3.1 Contoh Representasi Kromosom

Kromosom		
3	4	5

Setiap individu akan dihitung nilai *fitness* nya. Fungsi dari nilai *fitness* itu sendiri adalah untuk mengetahui seberapa baik suatu individu tersebut dijadikan sebagai solusi (Mahmudy, 2013). Gambar 3.3 merupakan flowchart proses perhitungan nilai *fitness* .



Gambar 3.3 Flowchart Perhitungan *Fitness*

Untuk mendapatkan nilai *fitness*, maka perlu diketahui berapa kebutuhan nutrisi seekor kambing, yaitu kebutuhan BK, TDN dan PK. Berdasarkan dengan bobot badan 30 kg, target pertambahan berat badan 0,1 kg, maka kebutuhan BK adalah 0,84 kg, PK= 0,58 kg dan TDN= 0,56 kg, kebutuhan ini didapatkan dari Tabel 2.3 pada bab 2.

Tabel 3.2 Kebutuhan

Bahan	**Harga**	**Kebutuhan**		
		0,84 kg	0,058	0,56 kg
		BK	**PK**	**TDN**
Kaliandra	2000	0.16	0,277	0,62
Rumput Raja	3500	0,22	0,135	0,54
D Padi Kasr	4000	0,86	0,076	0,14

Tabel 3.3 Merupakan representasi kromosom yang digunakan pada contoh perhitungan.

Tabel 3.3 Representasi Kromosom

Kaliandra	Rumput Raja	Dedak Padi Kasar
3	4	5

Contoh kromosom pada Tabel 3.3 yang terdiri dari tiga gen penyusun yaitu kaliandra, rumput raja dan dedak padi kasar. Maka proses perhitungan untuk mencari nilai *fitness* adalah sebagai berikut:

1. Menghitung nilai nutrisi pada setiap invidivu

Menghitung nilai nutrisi BK, PK dan TDN untuk setiap bahan pakan dengan melihat tabel 2.3. Kaliandra, rumput raja dan dedak padi kasar masing-masing memiliki kandungan nutrisi BK, PK dan TDN. Tabel 3.4 merupakan tabel nilai nutrisi bk, pk dan tdn untuk setiap bahan pakan yang tersedia.

Tabel 3.4 Nilai BK, PK dan TDN Setiap Bahan Pakan

Kaliandra			Raja			Padi		
Bk	**Pk**	**Tdn**	**Bk**	**Pk**	**Tdn**	**Bk**	**Pk**	**Tdn**
0.48	0.831	1.86	0.88	0.54	2.16	4.3	0.38	0.7

Berikut merupakan contoh perhitungan untuk mendapatkan nilai nutrisi setiap bahan pakan.

a. Nilai nutrisi Kaliandra
   - BK = Nilai nutrisi BK kaliandra x bobot kaliandra
     = 0,16 x 3
     = 0,48
   - PK = Nilai nutrisi PK kaliandra x bobot kaliandra
     = 0,277 x 3
     = 0,831
   - TDN = Nilai nutrisi TDN kaliandra x bobot kaliandra
     = 0,62 x 3
     = 1,86

b. Nilai nutrisi rumput raja
   - BK = Nilai nutrisi BK rumput raja x bobot kaliandra
     = 0,22 x 4
     = 0,88

- PK = Nilai nutrisi PK rumput raja x bobot kaliandra
  = 0,135 x 4
  = 0,54
- TDN = Nilai nutrisi TDN rumput raja x bobot kaliandra
  = 0,54 x 4
  = 2,16

c. Nilai nutrisi dedak padi kasar

- BK = Nilai nutrisi BK rumput raja x bobot kaliandra
  = 0,86 x 5
  = 4,3
- PK = Nilai nutrisi PK rumput raja x bobot kaliandra
  = 0,076 x 5
  = 0,38
- TDN = Nilai nutrisi TDN rumput raja x bobot kaliandra
  = 0,14 x 5
  = 0,7

2. Menghitung nilai *pinalty*

Dari Tabel 3.4 maka dapat diketahui nilai nutrisi dari masing-masing bahan pakan. Untuk mendapatkan nilai *penalty*, nilai bobot di kalikan dengan nilai setiap nutrisinya, kemudian dilakukan pemeriksaan ulang dengan kebutuhan kambing, jika kebutuhannya nutrisi terpenuhi nilai *penalty* nya 0 dan jika ada nutrisi yang tidak terpenuhi akan diperoleh nilai *penalty* nya. Tabel 3.5 merupakan tabel kebutuhan dan ketersediaan nutrisi.

Tabel 3.5 Kebutuhan dan ketersedian

Kebutuhan			**Ketersediaan**			***Pinalty***
BK	**PK**	**TDN**	**BK**	**PK**	**TDN**	
0,84	0,058	0,56	5,66	1,751	4,72	0

Sehingga dari perhitungan jumlah nutrisi setiap bahan pakan dalam 1 individu maka dapat diketahui nilai *penalty* nya. Berikut merupakan contoh perhitungan untuk mendapatkan nilai *penalty*.

- BK = 0,48 + 0,88 + 4,3

= 5,66

- PK = 0,831 + 0,54 + 0,38
  = 1,751
- TDN =1,86 +2,16 + 0,7
  = 4,72

Karena tidak ada nilai nutrisi yang kurang dari kebutuhan maka nilai Penalty nya adalah 0.

- Dianggap *penalty* jika total bk < kebutuhan bk kambing yang telah di hitung di awal.
- Dianggap *penalty* jika total pk < kebutuhan pk kambing yang telah di hitung di awal.
- Dianggap *penalty* jika total tdn < kebutuhan tdn kambing yang telah di hitung di awal.

3. Menghitung harga

Selanjutnya adalah menghitung total harga dalam satu individu dengan cara:

$$\text{Harga} = \frac{\text{gen pakan}}{\text{Total kromosom}} \text{ x range akhir x harga bahan pakan} \quad (3\text{-}1)$$

Daftar harga setiap bahan pakan dapat dilihat pada halaman lampiran. Tabel 3.6 merupakan tabel totl harga setiap individu dalam satu populasi.

Tabel 3.6 Harga Setiap Individu

Parent	Kromosom			Harga
	Kaliandra	Rumput Raja	D. Padi Kasar	
p1	3	4	5	33332
p2	4	7	6	33231
p3	5	3	3	29544
p4	4	4	8	33750
p5	4	3	4	31362
p6	4	3	6	32691
p7	7	5	2	28214
p8	2	8	4	34285
p9	6	3	4	29614
p10	3	6	3	32500

Keterangan :

$\left(\left(\frac{3}{15}\right)\text{x}10\right)\text{x}\,4000 = 8000$Berikut merupakan contoh perhitungan harga sehingga akan didapatkan harga masing-masing bahan pakan :

- Harga untuk kaliandra $= \left(\left(\frac{3}{12}\right) x10\right) x\ 2000$

  $= 5000$

- Harga untuk kaliandra $= \left(\left(\frac{4}{12}\right) x10\right) x\ 3500$

  $= 11655$

- Harga untuk kaliandra $= \left(\left(\frac{5}{12}\right) x10\right) x\ 4000$

  $= 16640$

4. Menghitung *Fitness*

Setelah menghitung harga setiap individu, maka langkah berikutnya adalah menghitung nilai *fitness*. Rumus perhitungan nilai *fitness* dapat dilihat pada subbab 2.6.3.4. Dengan menggunakan rumus *fitness* yang sudah dijelaskan pada bab sebelumnya maka didapatkan nilai *fitness* untuk setiap individu dapat dilihat pada Tabel 3.7.

Tabel 3.7 Nilai *Fitness*

Parent	**Kromosom**			***Fitness***
	Kaliandra	**Rumput Raja**	**Dedak Padi Kasar**	
p1	3	4	5	3.00012E-05
p2	4	7	6	3.00924E-05
p3	5	3	3	3.38478E-05
p4	4	4	8	2.96296E-05
p5	4	3	4	3.18857E-05
p6	4	3	6	3.05895E-05
p7	7	5	2	3.54434E-05
p8	2	8	4	2.91673E-05
p9	6	3	4	3.37678E-05
p10	3	6	3	3.07692E-05

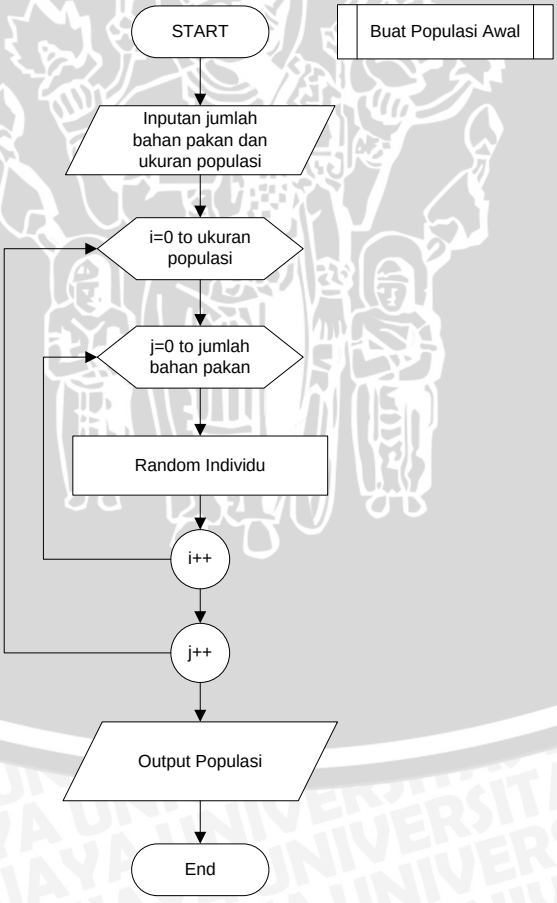
Berikut merupakan contoh perhitungan nilai *fitness* :

$$f = \left(\frac{1}{33332 + (0\ x\ 10000)}\right) = 3.00012\text{E-}05$$

3.4.2 Inisialisasi Populasi Awal

Pada tahap ini dibangkitkan populasi awal sebanyak jumlah populasi yang telah ditentukan. Panjang kromosom adalah sama dengan jumlah bahan pakan yang diinpukan pada halaman nutrisi pada aplikasi. Jumlah generasi digunakan untuk menentukan berapa banyak ukuran generasi yang diinginkan. Parameter untuk membentuk populasi awal pada algoritma genetika adalah *popSize, crossover rate, mutation rate* dan generasi.

PopSize berfungsi untuk menyatakan banyaknya individu yang akan digunakan dalam 1 populasi. *Crossover rate* dan *mutation rate* berfungsi untuk menghasilkan jumlah *offspring* (*child*) yang akan dihasilkan. Menentukan panjang setiap kromosom dengan membangkitkan bilangan acak antara 1-10. Berikut merupaka flowchart dari proses membuat populasi awal dapat dilihat pada Gambar 3.4.



Gambar 3.4 Proses Buat Populasi Awal

Untuk menyelesaikan contoh permasalahan diatas maka diperlukan inisialisasi parameter sebagai berikut :

a. Jumlah populasi (*popSize*) = 10
b. C*rossover rate* (cr) = 0,2
c. *Mutation rate* (mr) = 0,1
d. Jumlah generasi = 1

Dalam contoh kasus ini terdapat 3 bahan pakan yaitu dedak padi, ampas tahu dan tepung ikan, maka representasi kromosom dapat dilihat pada Tabel 3.8. Bahan pakan yang tersedia kaliandra, rumput raja dan dedak padi kasar.

Tabel 3.8 Populasi Awal

Parent	**Kromosom**		
	Kaliandra	**Rumput Raja**	**Dedak Padi Kasar**
p1	3	4	5
p2	4	7	6
p3	5	3	3
p4	4	4	8
p5	4	3	4
p6	4	3	6
p7	7	5	2
p8	2	8	4
p9	6	3	4
p10	3	6	3

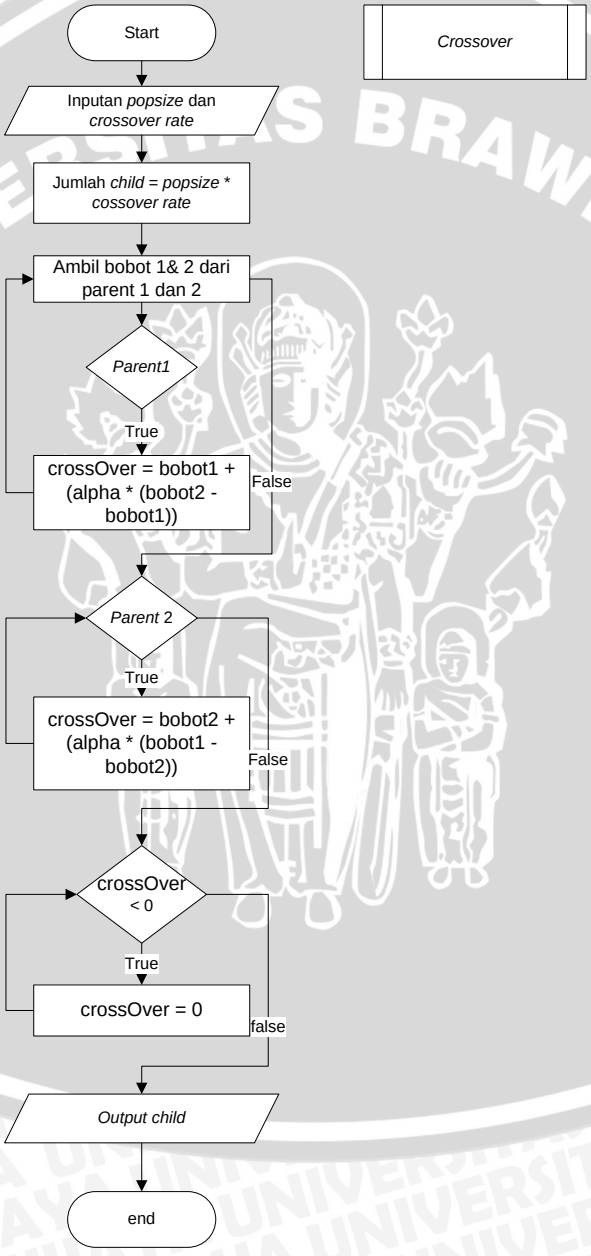
3.4.3 Reproduksi

Proses selanjutnya adalah proses reproduksi dimana di dalam proses ini terdapat 2 proses yaitu proses *crossover* dan proses mutasi. Proses *crossover* dilakukan dengan metode *extended intermediate crossover* sedangkan proses mutasi menggunakan metode *random mutation*.

3.4.3.1 Perhitungan Crossover

Metode *crossover* yang digunakan dalam perhitungan ini adalah *extended intermediate* yang menghasilkan *offspring* dari kombinasi nilai dua induk yang terpilih secara acak. Jumlah *offspring* yang dihasilkan dalam proses *crossover* dapat dilihat berdasarkan *crossover rate* (cr) yang akan diinputkan pada program nantinya. Misalkan P3 dan P4 adalah dua induk yang telah terpilih secara acak

untuk dilakukan proses *crossover*, maka *offspring* C1 dan C2 bisa dibangkitkan dengan Persamaan (2-1) dan Persamaan (2-2) (Mahmudy, 2013). Banyaknya *offspring* yang dihasilkan pada proses *crossover* adalah hasil perkalian jumlah populasi (*popSize*) dengan *crossover rate* (Pc x *popSize* = 0,2 x 10 = 2) sedangkan nilai α dipikih secara acak pada interval [0-1] yakni 0.65, 0,40 dan 0.34. Flowchart dalam melakukan proses *crossover* dapat dilihat pada Gambar 3.5



Gambar 3.5 Flowchart proses *crossover*

Proses *crossover* ditunjukkan pada Tabel 3.9. Contoh perhitungan proses crossover dengan induk yang terpilih adalah P1 dan P4 adalah sebagai berikut :

Tabel 3.9 Perhitungan manual *crossover*

	kromosom			harga	Penalty	*fitness*
p3	5	3	3	29544	0	3.38478E-05
p4	4	4	8	33750	0	2.96296E-05
Offspring 1	4.35	3.4	4.7	31646	0	3.15996E-05
Offspring 2	4.65	3.6	6.3	32371	0	3.08918E-05

Keterangan :

C1 : x1 = 5 + 0,65 (4 – 5) = 4.35

x2 = 3 + 0,40 (4 – 3) = 3,4

x3 = 3 + 0,34 (8 – 3) = 4,7

C2 : x1 = 4 + 0,65 (5 – 4) = 4,65

x2 = 4 + 0,40 (3 – 4) = 3,6

x3 = 8 + 0,34 (3 – 8) = 6,3

3.4.3.2 Perhitungan Mutasi

Metode mutasi yang digunakan adalah *random mutation* . Metode mutasi ini dilakukan dengan menambah atau mengurangi nilai gen terpilih dengan bilangan *random* yang kecil (Mahmudy, 2013) seperti pada persamaan (2-3). Banyaknya *offspring* yang dihasilkan pada proses mutasi adalah dengan mengalikan *mutation rate* dengan *popSize* atau mr x *popSize* = 0,1 x 10 = 1. Sedangkan nilai *r* di bangkitkansecara acak dengan interval [-0,1, 0,1] yakni 0,025 dan individu yang terpilih adalah nomor 2. Maka akan dihasilkan *offspring 3* pada. *Flowchart* dalam melakukan proses mutasi dapat dilihat pada Gambar 3.6.
Pada Tabel 3.10 Contoh perhitungan proses mutasi dengan induk yang terpilih adalah P2 adalah sebagai berikut :

Tabel 3.10 Perhitungan manual mutasi

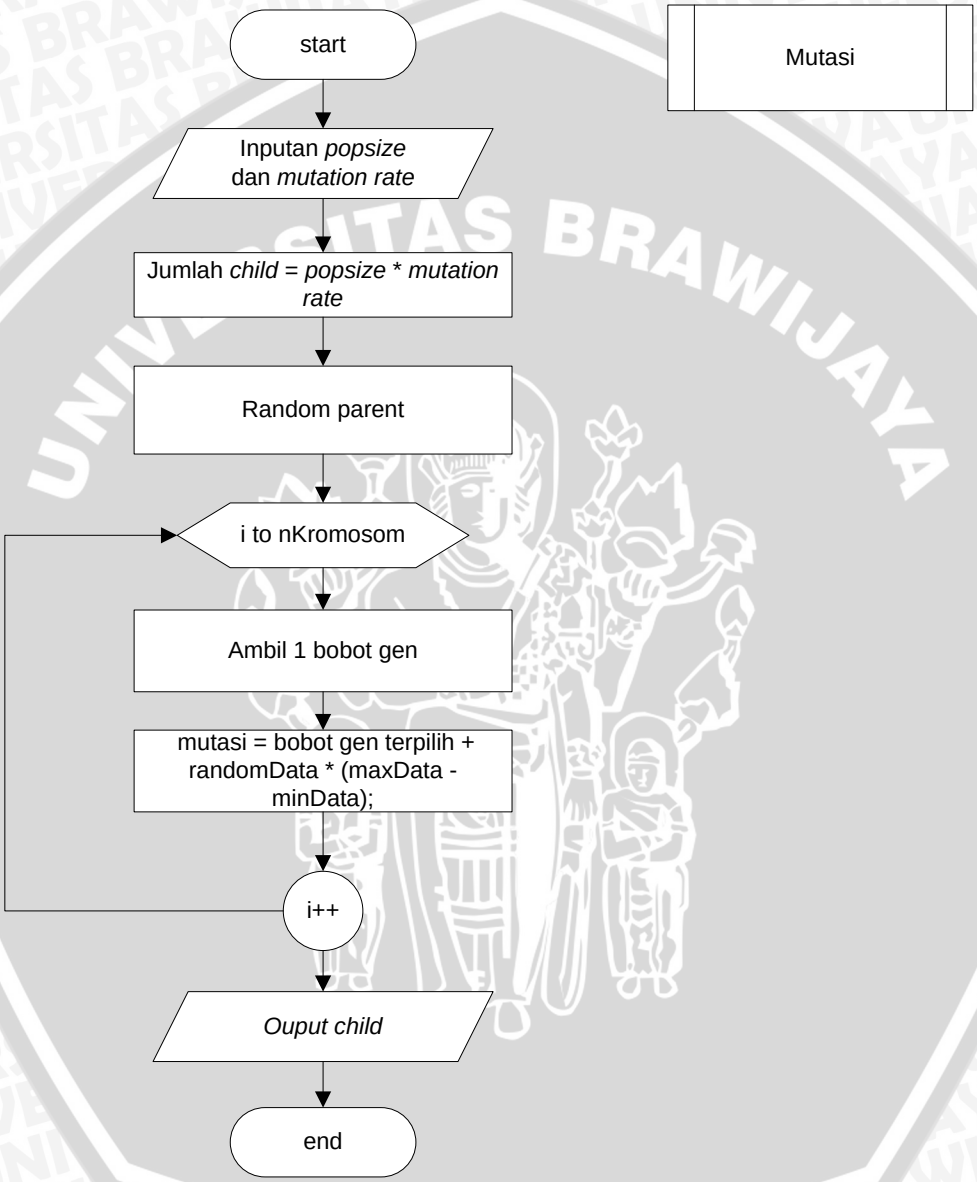
	Kromosom			**Harga**	**Penalty**	***fitness***
p2	4	7	6	33231	0	3.01E-05
Offspring 3	3	6	1.25	31220	0	3.2E-05

Keterangan :

C1 : x1 = 4 (tetap)

x2 = 7 – ((0,25)*(10-3)) = 1,25

x3 = 6 (tetap)



Gambar 3.6 Flowchart proses mutasi

3.4.4 Evaluasi dan Seleksi

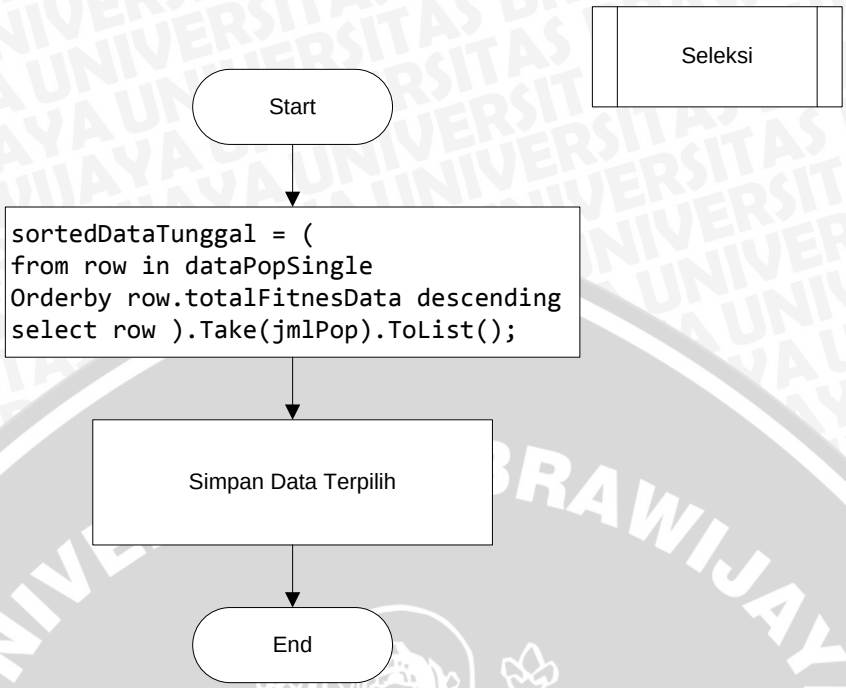
Setelah melakukan tahap reproduksi dimana tahap reproduksi ada 2 proses yaitu *crossover* dan mutasi maka selanjutnya dilakukan proses evaluasi. Proses evaluasi digunakan untuk menghitung kebugaran (*fitness*) dari setiap kromosom.

Proses evaluasi dilakukan dengan menghimpun seluruh individu yakni *parent* dan *offspring* yang kemudian menghitung nilai *fitness* masing-masing. Semakin besar nilai *fitness* maka semakin baik kromosom tersebut untuk dijadikan solusi (Mahmudy, 2013). Berikut ini adalah tabel hasil perhitungan *fitness* seluruh individu.

Tabel 3.11 Tabel Evaluasi

	Kromosom			**Harga**	**Penalty**	***Fitness***
	Kaliandra	**Rumput Raja**	**Dedak Padi Kasar**			
p1	3	4	5	33332	0	3.00012E-05
p2	4	7	6	33231	0	3.00924E-05
p3	5	3	3	29544	0	3.38478E-05
p4	4	4	8	33750	0	2.96296E-05
p5	4	3	4	31362	0	3.18857E-05
p6	4	3	6	32691	0	3.05895E-05
p7	7	5	2	28214	0	3.54434E-05
p8	2	8	4	34285	0	2.91673E-05
p9	6	3	4	29614	0	3.37678E-05
p10	3	6	3	32500	0	3.07692E-05
c1	4.35	3.4	4.7	31646	0	3.15996E-05
c2	4.65	3.6	6.3	32371	0	3.08918E-05
c3	3	6	1.25	31220	0	3.20307E-05

Seleksi dilakukan untuk memilih individu dari himpunan populasi dan *offspring* yang dipertahankan hidup pada generasi berikutnya (Mahmudy, 2013). Metode seleksi yang digunakan adalah *elitism*, metode seleksi ini bekerja dengan menggabungkan semua individu dalam populasi (*parent*) dan *offspring* dalam satu penampungan (Mahmudy, 2013). Seleksi *elitism* dilakukan dengan memilih *fitness* paling tinggi, sejumlah dengan jumlah populasi awal. Individu yang memiliki nilai *fitness* tinggi akan menjadi populasi baru di generasi selanjutnya. Flowchart proses seleksi dengan metode *elitism* dapat dilihat pada Gambat 3.7.



Gambar 3.7 Flowchart proses seleksi *elitism*

Hasil seleksi *elitism* dapat dilihat pada Tabel 3.12.

Tabel 3.12 Hasil Seleksi Elitism

P(t+1)	Asal P(t)	Kromosom			Harga	Penalty	*Fitness*
		Kaliandra	Rumput Raja	Dedak Padi Kasar			
p1	p7	7	5	2	28214	0	3.54434E-05
p2	p3	5	3	3	29544	0	3.38478E-05
p3	p9	6	3	4	29614	0	3.37678E-05
p4	c3	3	6	1.25	31220	0	3.20307E-05
p5	p5	4	3	4	31362	0	3.18857E-05
p6	c1	4.35	3.4	4.7	31646	0	3.15996E-05
p7	c2	4.65	3.6	6.3	32371	0	3.08918E-05
p8	p10	3	6	3	32500	0	3.07692E-05
p9	p6	4	3	6	32691	0	3.05895E-05
p10	p2	4	7	6	33231	0	3.00924E-05
c1	p1	3	4	5	33332	0	3.00012E-05
c2	p4	4	4	8	33750	0	2.96296E-05
c3	p8	2	8	4	34285	0	2.91673E-05

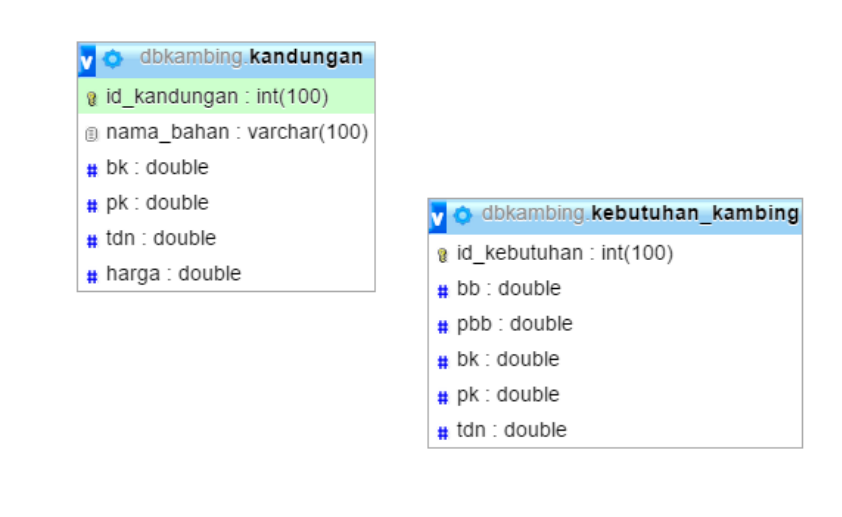
Jadi, komposisi bahan pakan yang terbaik adalah pada parent ke 7 dengan kromosom 7 5 2 dengan harga 28214 rupiah.

BAB IV
PERANCANGAN

Pada bab perancangan ini akan dibahas perancangan database, perancangan *user interface* dan perancangan uji coba yang akan digunakan dalam pembuatan aplikasi optimasi komposisi pakan kambing potong dengan menggunakan algoritma genetika.

4.1 Perancangan Database

Rancangan database pada aplikasi ini terdiri dari 2 tabel, yaitu tabel kebutuhan kambing dan tabel kandungan. Pada tabel kebutuhan kambing terdapat *field* yaitu : id_kebutuhan, bb (berat badan), pbb (pertambahan berat badan), bk, pk dan tdn. Sedangkan pada tabel kandungan terdapat *field* yaitu id_kandungan, nama_bahan, bk, pk, tdn dan harga, dimana harga disini adalah harga setiap bahan pakan. Berikut adalah desain *database* dapat dilihat pada Gambar 4.1.



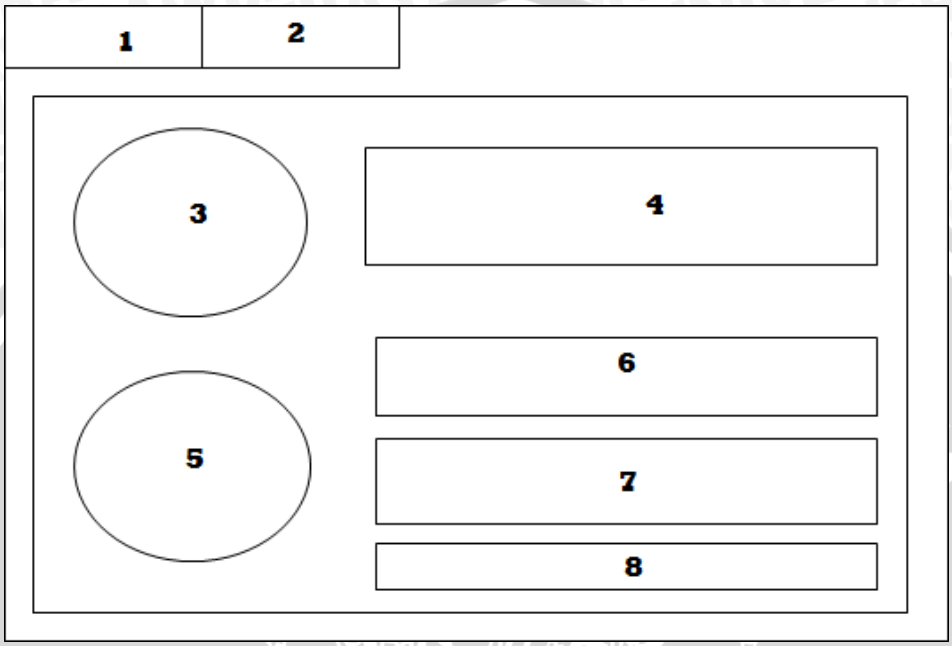
Gambar 4.1 Desain *Database*

4.2 Perancangan Antarmuka

Rancangan antar muka dari aplikasi optimasi komposisi pakan kambing potong menggunakan algortima genetika terdiri dari 2 halaman yaitu halaman nutrisi dan halaman algen.

4.2.1 Rancangan Tampilan Nutrisi

Rancangan antarmuka halaman nutrisi dari aplikasi optimasi komposisi pakan kambing potong menggunakan algoritma genetika dapat dilihat pada Gambar 4.2.



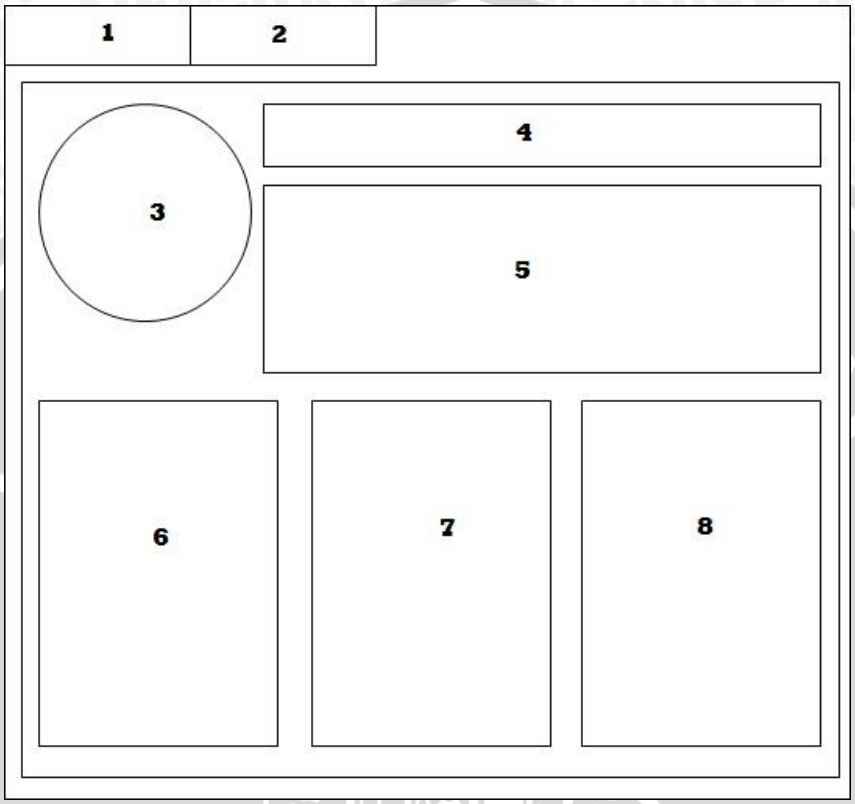
Gambar 4.2 Rancangan Antarmuka Halaman Nutrisi

Keterangan :

1. Menu *File* yang berisi menu *exit* untuk keluar program.
2. Menu yang bernama menu yang berisi fitur untuk memulai program.
3. Tempat untuk menginputkan berat badan dan target pertambahan berat badan.
4. Menampilkan kebutuhan nutrisi dari tabel kebutuhan sesuai dengan bobot badan dan target pertambahan bobot badan.
5. Digunakan untuk memilih bahan pakan untuk kemudian disusun sebagai pakan atau ransum.
6. Menampilkan bahan pakan yang sudah dipilih
7. Menampilkan informasi bahan pakan yang berupa nutrisi setiap bahan pakan.
8. Digunakan untuk menuju halaman algen.

4.2.2 Rancangan Halaman Algen

Rancangan antarmuka halaman algen dari aplikasi optimasi komposisi pakan kambing potong menggunakan algoritma genetika dapat dilihat pada Gambar 4.3.



Gambar 4.3 Rancangan Antarmuka Halaman Algoritma Genetika

Keterangan :

1. Menu *File* yang berisi menu *exit* untuk keluar program.
2. Menu yang bernama menu yang berisi fitur untuk memulai program.
3. Tempat untuk menginputkan *popsize*, *crossover rate*, *mutation rate* dan jumlah generasi.
4. Menampilkan hasil rekomendasi.
5. Menampilkan populasi pada generasi terakhir.
6. Menampilkan grafik *fitness* terbaik setiap generasi.
7. Menampilkan grafik bk, pk dan tdn terbaik setiap generasi.
8. Menampilkan grafik harga terbaik setiap generasi.

4.3 Perancangan Uji Coba dan Evaluasi

Dikarenakan tidak adanya metode yang pasti untuk menentukan parameter algoritma genetika, maka untuk mengevaluasi program dilakukan beberapa uji coba antara lain :

1. Uji coba untuk menentukan ukuran populasi yang optimal.
2. Uji coba untuk menentukan banyaknya generasi yang optimal.
3. Uji coba untuk mencari kombinasi *crossover rate*(cr) dan *mutation rate* (mr) yang terbaik.

4.3.1 Uji Coba Ukuran Populasi

Uji coba ukuran populasi ini dilakukan untuk mengetahui ukuran populasi yang tepat untuk menghasilkan komposisi pakan kambing terbaik. Kemampuan algoritma genetika dalam menemukan solusi yang terbaik dipengaruhi oleh ukuran populasi yang digunakan. Banyaknya populasi yang digunakan adalah kelipatan 40. Rancangan uji coba ukuran populasi dapat dilihat pada Tabel 4.1.

Tabel 4.1 Rancangan Uji Coba Ukuran Populasi

Banyak Populasi	**Nilai *fitness***										**Rata - rata *fitness***
	Percobaan populasi ke -										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
80											
160											
240											
320											
400											
480											
560											

4.3.2 Uji Coba Banyaknya Generasi

Uji coba banyaknya generasi ini dilakukan untuk mengetahui banyaknya generasi yang tepat untuk menghasilkan komposisi pakan kambing terbaik pada aplikasi ini. Kemampuan algortima genetika dalam menemukan solusi yang optimal dipengaruhi oleh banyaknya generasi yang digunakan. Banyaknya genersi yang digunakan adalah kelipatan 50 .Rancangan uji coba ukuran populasi dapat dilihat pada Tabel 4.2.

Tabel 4.2 Rancangan Uji Coba Banyaknya Generasi

Banyak Generasi	Nilai *fitness*										Rata - rata *fitness*
	Percobaan generasi ke -										
	1	2	3	4	5	6	7	8	9	10	
100											
200											
300											
400											
500											
600											
700											
800											

4.3.3 Uji Coba Kombinasi *Crossover Rate* dan *Mutation Rate*

Uji coba berdasarkan *crossover rate* (*cr*) dan *mutation rate* (*mr*) dilakukan untuk mendapatkan kombinasi *cr* dan *mr* yang tepat untuk menghasilkan komposisi pakan kambing potong terbaik. Pada uji coba ini digunakan nilai yang berbeda pada parameter algoritma genetika, yaitu *crossover rate* (*cr*) dan *mutation rate* (*mr*). Nilai *cr* dan *mr* yang digunakan antara 0 dan 1. Rancangan uji coba kombinasi cr dan mr ditampilkan pada Tabel 4.3 Parameter yang digunakan dalam uji coba adalah:

a. Jumlah populasi = Hasil populasi terbaik pada uji coba populasi
b. Jumlah generasi = Hasil generasi terbaik pada uji coba generasi

Tabel 4.3 Rancangan Uji Coba Kombinasi *cr* dan *mr*

Kombinasi		Nilai Fitness										Rata -rata nilai *fitness*
		Percobaan Kombinasi *pc* dan *pm ke* -										
Pc	*Pm*	1	2	3	4	5	6	7	8	9	10	
0.6	0											
0.5	0.1											
0.4	0.2											
0.3	0.3											
0.2	0.4											
0.1	0.5											
0	0.6											

BAB V
IMPLEMENTASI

Pada bab implementasi ini akan dibahas tentang implementasi sistem yang telah dibuat. Pembahasan implementasi meliputi implementasi program dan impelemntasi antarmuka.

5.1 Implementasi Program

Pada subbab ini akan dijelaskan tentang implementasi program berdasarkan perancangan sistem yang sudah dijelaskan pada bab 4. Pada penjelasan implementasi program ini akan dijelaskan proses-proses dan *source code* setiap fungsi menggunakan bahasa pemrograman c#.

5.1.1 Implementasi Kelas Koneksi

Berikut ini merupakan potongan program untuk menjelaskan koneksi database MySQL yang akan dijelaskan pada *Source Code* 5.1.

```
MySqlConnection conn = new
MySqlConnection("server=localhost;database=dbkambing;uid=root;p
assword=;");
              List<TBLkebutuhan> TBLkeb = new
List<TBLkebutuhan>();
              List<TBLkandungan> TBLkan = new
List<TBLkandungan>();
      public koneksiMySQL()
     {
            loadTableKebutuhan();
           loadTableKandungan();
      }
```

Source Code 5.1 Kelas Koneksi

Source Code 5.1 Menjelaskan proses implementasi kelas koneksi yang berfungsi untuk membuat koneksi antara program yang dibuat dengan database MySQL dimana kelas koneksi ini merupakan tempat penyimpanan data *connection string.*

5.1.2 Implementasi Proses Inisialisasi

Berikut ini merupakan potongan program untuk menjelaskan proses inisialisasi yang akan dijelaskan pada *Source Code* 5.2.

```
public Form1()
            {
            InitializeComponent();
                connectData = new koneksiMySQL();
                TBLkeb = new List<TBLkebutuhan>();
                TBLkan = new List<TBLkandungan>();
                TBLkebGet = new List<TBLkebutuhan>();

                TBLkeb = connectData.getDataKebutuhan();
                TBLkan = connectData.getDataKandungan();

                dataKandPersen = new List<TBLkandungan>();
                dataKandKilo = new List<TBLkandungan>();
dataKebutuhanPrimer = new TBLkebutuhan();
          Inisialisasi();
}
private void Inisialisasi()
{
TBLkebDistinct = TBLkeb.GroupBy(p => p.bb).Select(g =>
g.First()).ToList();

foreach (var item in TBLkebDistinct)
            {
            BeratBadan.Items.Add(item.bb);
            }
     foreach (var item in TBLkan)
     {
                Bahan_Pakan.Items.Add(item.nama_bahan);
      }
     gridKandungan.ColumnCount = 4;
     gridKandungan.Columns[0].Name = "Nama";
     gridKandungan.Columns[1].Name = "Pk";
     gridKandungan.Columns[2].Name = "Bk";
     gridKandungan.Columns[3].Name = "Tdn";
}
```

Source Code 5.2 Proses Inisialisasi

Source Code 5.2 Menjelaskan proses inisialisaisi dalam program yang dibuat. Proses ini merupakan bertujuan untuk inisialisai variabel yang akan digunakan untuk pemrosesan data.

5.1.3 Implementasi Proses Perhitungan Kandungan Bahan Pakan

Berikut ini merupakan potongan program untuk menjelaskan proses perhitungan kandungan setiap bahan pakan yang akan dijelaskan pada *Source Code* 5.3.

```
foreach (var item in dataKandPersen)
            {
                  TBLkandungan itemKand = new TBLkandungan();
            itemKand.pk_kan = item.pk_kan / 100d;
            itemKand.bk_kan = item.bk_kan / 100d;
            itemKand.tdn_kan = item.tdn_kan / 100d;
            itemKand.nama_bahan = item.nama_bahan;
            itemKand.harga = item.harga;
            dataKandKilo.Add(itemKand);
            }
```

Source Code 5.3 Proses Perhitungan Kandungan Bahan Pakan

Source Code 5.3 Menjelaskan tentang proses perhitungan kandungan bahan pakan. Perhitungan yang dimaksud dalam program ini adalah menghitung kandungan setiap bahan pakan yang masih dalam bentuk persen ke dalam kilogram menggunakan rumus perhitungan nutrisi. Data kandungan dalam bentuk persen didapatkan dari sumber yang sudah dibahas di bab 2, data kandungan bahan pakan disimpan di database MySQL untuk di ubah dalam bentuk kilogram.

5.1.4 Implementasi Proses Pembangkitan Populasi Awal

Berikut ini merupakan potongan program untuk menjelaskan proses pembangkitan populasi yang akan dijelaskan pada *Source Code* 5.4

```
private void buatPopulasiMany(int popSize)
      {
            int rangeAwal = 0;
           int rangeAkhir = 10;
           Random rnd = new Random();
           List<DTPopChild> listTemp = new List<DTPopChild>();
            for (int i = 0; i < popSize; i++)
            {
                  double hitungsatu = 0;
               foreach (var satu in dataItemPakan)
               {
                double bilRand =
System.Math.Round((Double)rnd.Next(rangeAwal, rangeAkhir) *
rnd.NextDouble(), 2);
                      satu.bilanganAcak = bilRand;
                }
                  foreach (var totalsatu in dataItemPakan)
                {
      hitungsatu = totalsatu.bilanganAcak + hitungsatu;
                    }
                    foreach (var item in dataItemPakan)
                    {
      DTPopChild itemPopulasi = new DTPopChild(); //buat objek
                        itemPopulasi.namaData =
item.nama_bahan;
```

```
                            itemPopulasi.bkData =
System.Math.Round(item.bk_kan * item.bilanganAcak, 2);
                            itemPopulasi.pkData =
System.Math.Round(item.pk_kan * item.bilanganAcak, 2);
                            itemPopulasi.tdnData =
System.Math.Round(item.tdn_kan * item.bilanganAcak, 2);
                            itemPopulasi.jenisData = "p" + (i +
1);
                            itemPopulasi.randomData =
item.bilanganAcak;
      if (hitungsatu <= rangeAkhir ){
             itemPopulasi.hargaData = item.bilanganAcak *
item.harga;}
      else{
             itemPopulasi.hargaData = ((item.bilanganAcak /
hitungsatu) * rangeAkhir) * item.harga;}
          dataPopMany.Add(itemPopulasi);
                        }
                    }}
```

Source Code 5.4 Proses Pembangkitan Populasi Awal

Source Code 5.4 Menjelaskan tentang proses pembangkitan populasi awal. Proses pembangkitan populasi awal tersebut di awali dengan membangkitkan kromosom setiap individu dengan cara random, panjang kromosom disesuaikan dengan banyak bahan pakan yang diinputkan. Kemudian menghitung atribut setiap individu. Atribut setiap individu yang dimaksud adalah harga, penalti dan *fitness*.

5.1.5 Implementasi Proses Perhitungan Penalti

Berikut ini merupakan potongan program untuk menjelaskan proses perhitungan Penalty yang akan dijelaskan pada *Source Code* 5.5.

```
foreach (var item in dataPopSingle)
            {
            double pinaltiPk = 0;
                double pinaltiBk = 0;
                double pinaltiTdn = 0;

                  if (item.totalPkData >=
dataKebPrimer.pk_keb) pinaltiPk = 0;
                        if (item.totalPkData <
dataKebPrimer.pk_keb)
                        {
                            pinaltiPk = dataKebPrimer.pk_keb -
item.totalPkData;
                        }
```

```
                    if (item.totalBkData >=
dataKebPrimer.bk_keb) pinaltiBk = 0;
                    if (item.totalBkData <
dataKebPrimer.bk_keb)
                    {
                        pinaltiBk = dataKebPrimer.bk_keb -
item.totalBkData;
                    }
                    if (item.totalTdnData >=
dataKebPrimer.tdn_keb) pinaltiTdn = 0;
                    if (item.totalTdnData <
dataKebPrimer.tdn_keb)
                    {
                        pinaltiTdn = dataKebPrimer.tdn_keb -
item.totalTdnData;
                    }
                    item.pinaltiData = pinaltiPk + pinaltiBk
+ pinaltiTdn;
                }
```

Source Code 5.5 Proses Perhitungan Penalti

Source Code 5.5 Menjelaskan proses perhitungan penalti. Penalti merupakan hasil dari perhitungan kurangnya nutrisi yang dibutuhkan. Jika nutrisi bahan pakan yang diinputkan sama dengan atau lebih dari kebutuhan kambing yang telah dihitung maka penaltinya dianggap 0, namun jika nutrisi bahan pakan kurang dari kebutuhan nutrisi kambing maka dihitung jumlah penalti setiap nutrisi. Hasil dari penalti ini digunakan untuk menghitung *fitness*.

5.1.6 Implemntasi Proses Perhitungan *Fitness*

Berikut ini merupakan potongan program untuk menjelaskan proses perhitungan niali *fitness* yang akan dijelaskan pada *Source Code* 5.6.

```
    public double hitungFitnes(double pinalti, double
harga)
            {
                double hasil = 0;
                double Cons = 10000;
                hasil = 1 / (harga + pinalti * Cons);
                return hasil;
            }
```

Source Code 5.6 Porses Perhitungan *Fitness*

Source Code 5.6 Menjelaskan tentang proses perhitungan *fitness*. Nilai *fitness* didapatkan dengan rumus 1 dibagi dengan total jumlah harga setiap

kromosom ditambah dengan hasil kali *Penalty* dan konstanta. Perhitungan *fitness* dilakukan pada setiap individu.

5.1.7 Implementasi Proses Perhitungan *Crossover*

Berikut ini merupakan potongan program untuk menjelaskan proses perhitungan *crossover* yang akan dijelaskan pada *Source Code* 5.7.

```
      public double hitungCrossOver(double alpha, DTPopParent
p1, DTPopParent p2, string namaPakan, int loops)
               {
                   double bobot1 = 0;
                   double bobot2 = 0;
                   double crossOver = 0;
                   foreach (var item in dataPopOriMany)
                   {
                       if ((item.jenisData.Equals(p1.jenisData))
&& (item.namaData.Equals(namaPakan)))
                       {
                           bobot1 = item.randomData;
                       }
                   }
      foreach (var item in dataPopOriMany)
                   {
                       if ((item.jenisData.Equals(p2.jenisData))
&& (item.namaData.Equals(namaPakan)))
                  {
                      bobot2 = item.randomData;
                   }
             }
             if (loops == 1)
             {
             crossOver = bobot1 + (alpha * (bobot2 - bobot1));
             }
             if (loops == 2)
             {
             crossOver = bobot2 + (alpha * (bobot1 - bobot2));
             }
             return crossOver;
        }
```

Source Code 5.7 Proses Perhitungan *Crossover*

Source Code 5.7 Menjelaskan tentang proses perhitungan *crossover*. Metode *crossover* yang digunakan adalah *extended intermediate crossover*. Untuk mendapatkan nilai *crossover* terlebih dahulu dilakukan pemilihan individu untuk dilakukan *crossover*. Untuk mendapatkan nilai *crossover* dilakukan juga pembangkitan nilai acak antara 0-1 sebanyak jumlah kromosom dibangkitkan.

Proses *crossover* ini dilakukan untuk menghasilkan *offspring*, jumlah *offspring* yang dihasilkan adalah sejumlah hasil kali *crossover rate* dengan jumlah populasi.

5.1.8 Implementasi Proses Perhitungan Mutasi

Berikut ini merupakan potongan program untuk menjelaskan proses perhitungan mutasi yang akan dijelaskan pada *Source Code* 5.8.

```
      public double hitungMutasi(double randomData, DTPopParent
p1, string namaPakan, bool isSelected)
              {
                  double mutasi = 0;
                  double minData = 3;
                  double maxData = 10;
                  double bobot1 = 0;
                  double bobot2 = 0;

                  foreach (var item in dataPopOriMany)
            {
                   if((item.jenisData.Equals(p1.jenisData))   &&
(item.namaData.Equals(namaPakan)))
     if((item.jenisData.Equals(p1.jenisData))                 &&
(item.namaData.Equals(namaPakan)))
                      {
                          bobot1 = item.randomData;
                      }
                  }
                  foreach (var item in dataPopOriMany)
                  {
                      if  ((item.jenisData.Equals(p1.jenisData))
&& (item.namaData.Equals(namaPakan)))
                      {
                          bobot2 = item.randomData;
                      }
                  }
                  if (isSelected) mutasi = bobot1 + randomData
* (maxData - minData);
                  else mutasi = bobot2;
                  return mutasi;
              }
```

Source Code 5.8 Proses Perhitungan Mutasi

Source Code 5.8 Menjelaskan tentang proses perhitungan mutasi. Metode mutasi yang digunakan adalah *random mutation*. Untuk mendapatkan nilai mutasi terlebih dahulu dilakukan pemilihan individu untuk dilakukan mutasi. Untuk mendapatkan nilai *crossover* dilakukan juga pembangkitan nilai acak antara -0,1 sampai 0,1 sebanyak 1 untuk setiap individu. Kromosom yang terpilih dari 1 individu akan dihitung menggunakan rumus *random mutation*, untuk kromosom

yang tidak terpilih maka nilai kromosomnya tetap. Proses mutasi ini dilakukan untuk menghasilkan *offspring* , jumlah *offspring* yang dihasilkan adalah sejumlah hasil kali *mutation rate* dengan jumlah populasi.

5.1.9 Implementasi Proses Seleksi dengan Metode *Ellitsm*

Berikut ini merupakan potongan program untuk menjelaskan proses perhitungan seleksi menggunakan metode *ellitsm* yang akan dijelaskan pada *Source Code* 5.9.

```
      public void urutDanSeleksi(int jmlPop, int iterasi)
              {
                  List<DTPopParent> sortedDataTunggal = (
                    from row in dataPopSingle
                    orderby row.totalFitnesData
                    select row).Take(jmlPop).ToList();
                  List<DTPopChild> sortedDataJamak = new
List<DTPopChild>();
                  foreach (var item in sortedDataTunggal)
                  {
                      foreach (var item2 in dataPopMany)
                      {
                          if
(item2.jenisData.Equals(item.jenisData))
                          {
                              DTPopChild data = new
DTPopChild();
                              data.hargaData = item2.hargaData;
                              data.alphaData = item2.alphaData;
                              data.jenisData = item2.jenisData;
                              data.pkData = item2.pkData;
                              data.bkData = item2.bkData;
                              data.tdnData = item2.tdnData;
                              data.namaData = item2.namaData;
                              data.randomData =
item2.randomData;
                              Console.WriteLine("Sorting and
Selecting >> " + item2.jenisData + "---- fitness " +
item.totalFitnesData + "--- iteration " + iterasi);
                              sortedDataJamak.Add(data);
                          }
                      }
                  }
                  dataPopMany = new
List<DTPopChild>(sortedDataJamak);
                  gridHasil.DataSource = null;
                  gridHasil.Rows.Clear();
                  gridHasil.Columns.Clear();
              }
```

Source Code 5.9 Proses Seleksi *Elltism*

Source Code 5.9 Menjelaskan tentang proses seleksi *ellitsm*. Pada seleksi dengan *ellitsm* ini semakin besar nilai *fitness* suatu invididu maka akan semakin besar pula peluang untuk terpilih menjadi individu untuk generasi selanjutnya dan semakin kecil nilai *fitness* dari suatu individu maka semakin kecil pula kemungkinan untuk terpilih menjadi individu untuk generasi selanjutnya. Proses seleksi ini dilakukan dengan cara mengurutkan nilai *fitness* dari yang paling besar sampai kecil kemudian untuk menghasilkan invidu untuk generasi selanjutnya dipilih sebanyak jumlah sesuai populasi awal.

5.1.10 Implementasi Pemilihan Kromosom Terbaik

Berikut ini merupakan potongan program untuk menjelaskan proses pemilihan kromosom terbaik yang akan dijelaskan pada *Source Code* 5.10.

```
      private void tampilRekomendasi()
              {
                  DTPopParent hasilSeleksi =
dataPopSingle.LastOrDefault();

                  string hasil = string.Empty;

                  foreach (var item in dataItemPakan)
                  {
                      foreach (var item2 in dataPopMany)
                      {
                          if
((item2.jenisData.Equals(hasilSeleksi.jenisData)) &&
(item2.namaData.Equals(item.nama_bahan)))
                          {
                              hasil += (item2.randomData + " kg
" + item2.namaData + "  ");
                          }
                      }
                  }
                  hasil += " harga " +
hasilSeleksi.totalHargaData;

                  lblHasil.Text = hasil;
              }
```

Source Code 5.10 Pemilihan Kromosom Terbaik

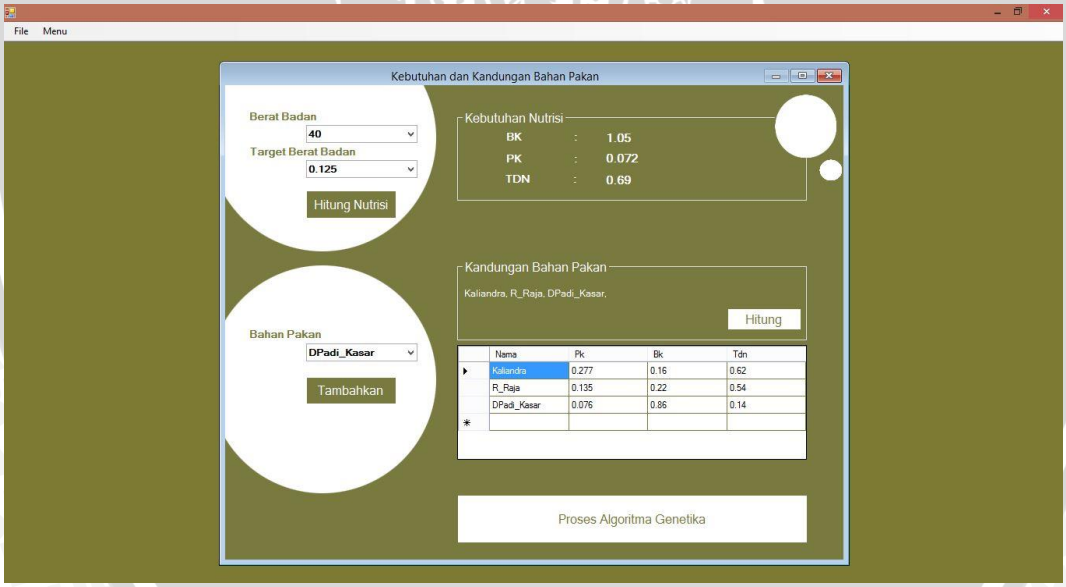
Source Code 5.10 Menjelaskan tentang proses pemilihan kromosom terbaik. Kromosom terbaik adalah kromosom yang mempunyai nilai *fitness* tertinggi dari kromosom yang ada untuk kemudian dijadikan populasi di generasi selanjutnya.

5.2 Implementasi Atarmuka

Implementasi antar muka meruapakah hasil implementasi dari perancangan yang telah dirancang pada bab sebelumnya yaitu perancangan antarmuka. Antarmuka yang telah dibuat adalah halaman nutrisi dan halaman algen.

5.2.1 Implementasi Antarmuka Halaman Nutrisi

Pada halaman nutrisi terdapat 2 bagian yaitu bagian untuk menghitung nutrisi dan untuk menghitung kandungan bahan pakan. Bagian untuk menghitung nilai nutrisi yang dibutuhkan adalah dengan cara memasukkan berat badan dan target pertambahan berat badan. Untuk mendapatkan nilai nutrisi setiap bahan pakan maka terlebih dahulu untuk mendapatkan satuan kilogram. Nilai kebutuhan nutrisi dan nilai kandungan bahan pakan tersebut berpengaruh terhadap perhitungan nilai *penalty* . Tampilan menu halaman nutrisi sistem ini dilihat pada Gambar 5.1.



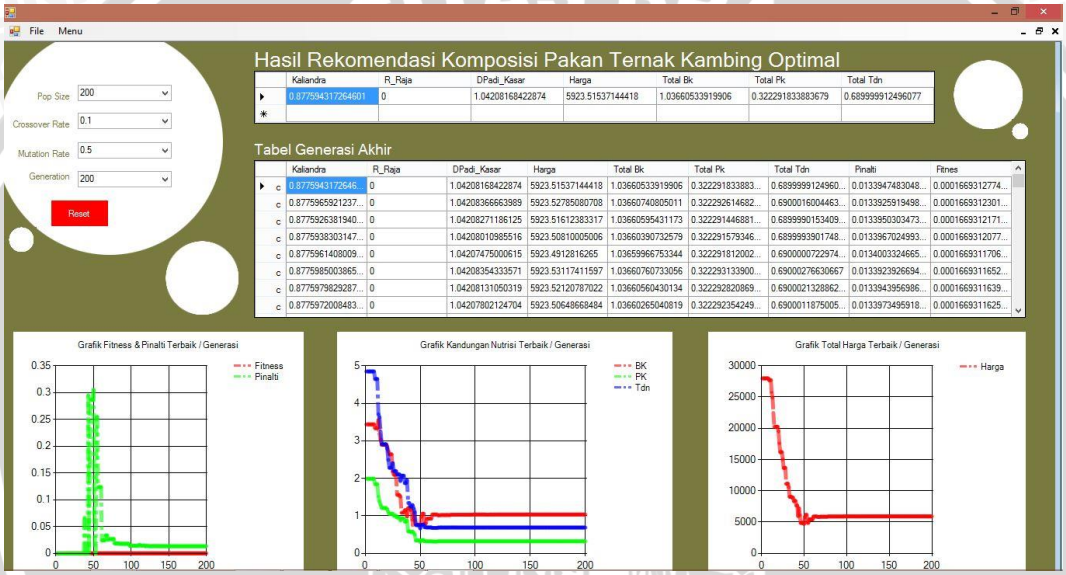
Gambar 5.1 Implemntasi Antarmuka Halaman Nutrisi

Pada antarmuka halaman nutrisi ini menghasilakn *output* atau solusi berupa nutrisi yang dibutuhkan kambing sesuai dengan berat badan dan target pertambahan berat badan yang diinputkan *user*. Pada halaman ini juga dihasilkan *output* berupa kandungan bahan pakan yang diinputkan dengan nutrisi yang sudah ditentukan pada Tabel 2.2. Dan *output* pada form ini akan digunakan pada

perhitungan di antarmuka selanjutnya yaitu antarmuka halaman algoritma genetika.

5.2.2 Implementasi Antarmuka Halaman Algen

Pada halaman algen ini akan dihitung untuk mendapatkan solusi terbaik. Untuk mendapatkan solusi terbaik dengan perhitungan algoritma genetika terlebih dahulu menginputkan jumlah populasi, nilai *crossover rate*, nilai *mutation rate* dan jumlah generasi yang akan digunakan dalam perhitungan. Tampilan menu halaman algoritma genetika sistem ini dilihat pada Gambar 5.2.



Gambar 5.2 Implemntasi Antarmuka Halaman Algen

Pada antarmuka halaman algoritma genetika ini menghasilakn *output* atau solusi berupa komposisi bahan pakan yang tersedia dengan kandungan nutrisi yang dibutuhkan kambing potong dengan berat badan dan target berat badan yang sudah ditentukan pada Tabel 2.2, sehingga didapatkan komposisi pakan yang mendekati terpenuhinya nutrisi kambing dengan harga yang minimal.

BAB VI
PENGUJIAN DAN ANALISIS

Pada bab pengujian dan analisis ini membahas mengenai pengujian serta analisis terhadap hasil implementasi yang sudah dirancang pada bab sebelumnya. Pengujian ini dilakukan untuk mengetahui parameter algoritma genetika yang optimal dilihat dari nilai *fitness* terbaik dari setiap pengujian yang dilakukan. Pengujian pada bab ini meliputi pengujian parameter algoritma genetika yaitu pengujian ukuran populasi, pengujian banyaknya generasi serta pengujian kombinasi *crossover rate* dan *mutation rate*.

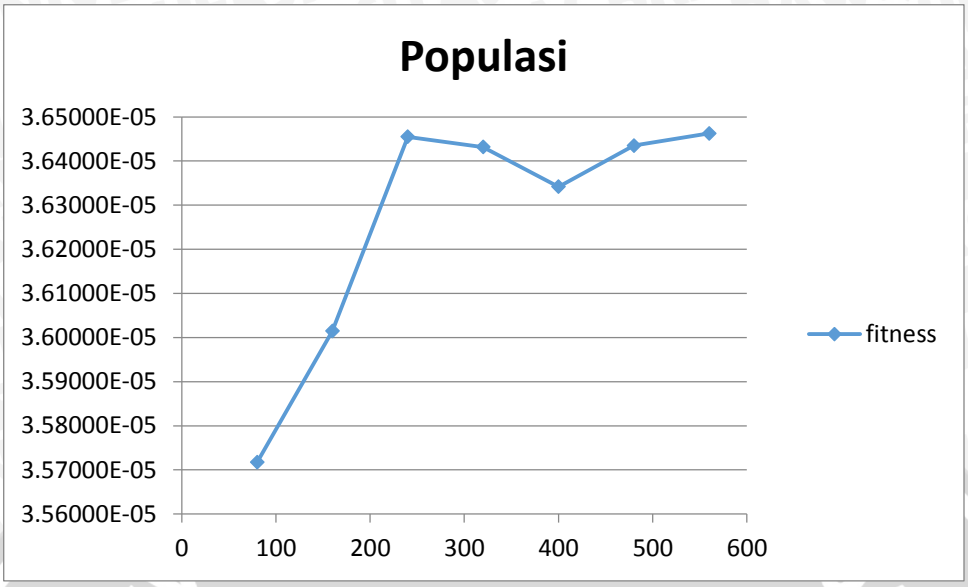
6.1 Pengujian dan Analisis Ukuran Populasi

Pada pengujian pertama dilakukan pengujian terhadap parameter ukuran populasi. Pengujian ukuran populasi dilakukan untuk mengetahui pengaruh jumlah populasi terhadap nilai *fitness* yang dihasilkan. Bahan pakan yang digunakan pada pengujian ini adalah 3 bahan pakan. Untuk menguji ukuran populasi digunakan ukuran populasi kelipatan 80, dimulai dari 80 sampai 560. Untuk mendapatkan hasil yang lebih *valid* maka setiap percobaan akan dilakukan sebanyak 10 kali untuk diambil nilai rata-rata *fitness*nya. Pada percobaan ini digunakan generasi = 250, cr = 0,5 dan mr = 0,1. Tabel hasil uji coba ukuran populasi dapat dilihat pada Tabel 6.1.

Tabel 6.1 Hasil Uji Coba Ukuran Populasi 1-10

	1	2	3	4	5	6	7	8	9	10	rata-rata
80	3.475E-05	3.61E-05	3.535E-05	3.506E-05	3.624E-05	3.661E-05	3.583E-05	3.576E-05	3.554E-05	3.595E-05	3.572E-05
160	3.62E-05	3.497E-05	3.605E-05	3.562E-05	3.594E-05	3.675E-05	3.64E-05	3.59E-05	3.636E-05	3.597E-05	3.601E-05
240	3.675E-05	3.611E-05	3.597E-05	3.677E-05	3.659E-05	3.673E-05	3.67E-05	3.711E-05	3.633E-05	3.55E-05	3.645E-05
320	3.569E-05	3.661E-05	3.571E-05	3.662E-05	3.63E-05	3.68E-05	3.614E-05	3.664E-05	3.718E-05	3.664E-05	3.643E-05
400	3.634E-05	3.598E-05	3.642E-05	3.66E-05	3.637E-05	3.648E-05	3.665E-05	3.646E-05	3.608E-05	3.604E-05	3.634E-05
480	3.662E-05	3.619E-05	3.63E-05	3.64E-05	3.634E-05	3.659E-05	3.611E-05	3.667E-05	3.673E-05	3.64E-05	3.644E-05
560	3.662E-05	3.662E-05	3.602E-05	3.602E-05	3.654E-05	3.654E-05	3.64E-05	3.64E-05	3.674E-05	3.674E-05	3.646E-05

Dari tabel Tabel 6.1 dapat dibuat sebuah grafik untuk melihat dengan jelas hasil dari uji coba ukuran populasi terhadap nilai *fitness*. Dan grafik hasil uji coba ukuran populasi dapat dilihat pada Gambar 6.1.



Gambar 6.1 Grafik Percobaan Ukuran Populasi

Berdasarkan Gambar 6.1 dapat dilihat bahwa ukuran populasi berpengaruh terhadap nilai *fitness* yang dihasilkan dari perhitungan algoritma genetika. Grafik mengalami kenaikan pada ukuran populasi 80 sampai 240, dilakukan 10 kali percobaan pada pengujian ukuran populasi untuk menghasilkan nilai rata-rata *fitness* dan dihasilkan *fitness* terbaik dengan nilai 0.000036455. Pada ukuran populasi 320 sampai 560 tidak terjadi kenaikan yang signifikan. Sehingga populasi paling optimal untuk kasus ini yaitu pada ukuran populasi 240. Dan dapat disimpulkan bahwa nilai *fitness* yang dihasilkan oleh sistem dipengaruhi oleh ukuran populasi yang semakin tinggi, namun jika ukuran populasi semakin tinggi maka tidak didapatkan kenaikan *fitness* yang signifikan dan waktu komputasi yang digunakan untuk menghasilkan nilai *fitness* pun semakin lama. Pola seperti ini juga didapatkan oleh Pratiwi (2014) yang menerapkan algoritma genetika untuk pemenuhan kebutuhan nutrisi.

6.2 Pengujian dan Analisis Banyaknya Generasi

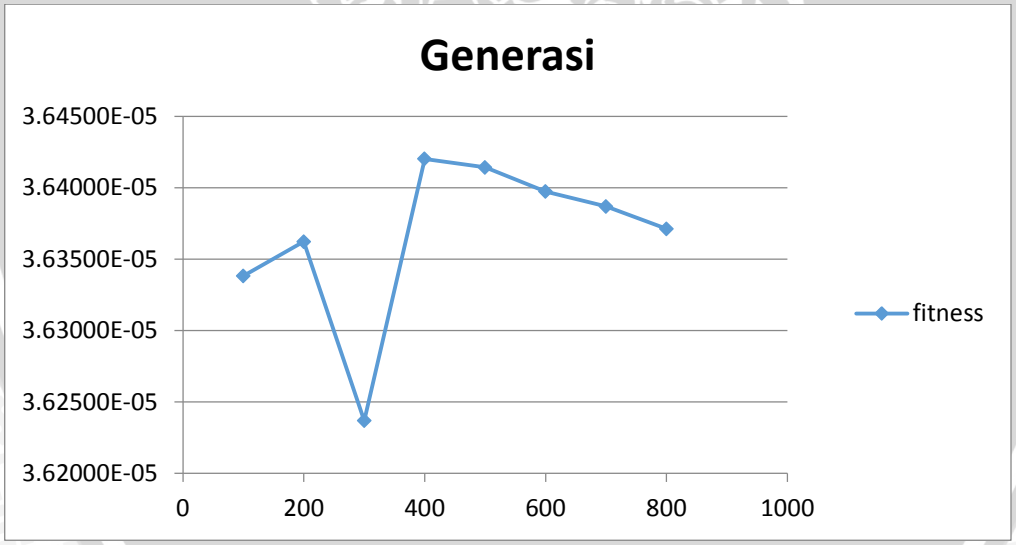
Pengujian kedua dilakukan pengujian terhadap parameter banyaknya generasi yang digunakan. Pengujian banyaknya generasi dilakukan untuk mengetahui pengaruh jumlah generasi terhadap nilai *fitness* yang dihasilkan. Bahan pakan yang digunakan pada pengujian ini adalah 3 bahan pakan. Untuk menguji banyaknya generasi digunakan banyaknya generasi mulai 100 sampai 800. Untuk mendapatkan hasil yang lebih *valid* maka setiap percobaan akan

dilakukan sebanyak 10 kali untuk diambil nilai rata-rata *fitness*nya. Pada percobaan ini digunakan populasi=240 yang didapatkan dari percobaan sebelumnya, cr=0,5 dan mr=0,1. Tabel hasil uji coba ukuran populasi dapat dilihat pada Tabel 6.2.

Tabel 6.2 Hasil Uji Coba Banyaknya Generasi 1-10

	1	2	3	4	5	6	7	8	9	10	rata2
100	3.62201E-05	3.718E-05	3.667E-05	3.617E-05	3.609E-05	3.604E-05	3.665E-05	3.666E-05	3.57E-05	3.601E-05	3.634E-05
200	3.56087E-05	3.579E-05	3.66E-05	3.689E-05	3.562E-05	3.651E-05	3.703E-05	3.647E-05	3.651E-05	3.658E-05	3.636E-05
300	3.63848E-05	3.573E-05	3.615E-05	3.619E-05	3.614E-05	3.673E-05	3.625E-05	3.673E-05	3.632E-05	3.575E-05	3.624E-05
400	3.62161E-05	3.647E-05	3.632E-05	3.565E-05	3.662E-05	3.669E-05	3.679E-05	3.698E-05	3.573E-05	3.673E-05	3.642E-05
500	3.68297E-05	3.652E-05	3.662E-05	3.63E-05	3.576E-05	3.669E-05	3.676E-05	3.66E-05	3.578E-05	3.629E-05	3.641E-05
600	3.59479E-05	3.666E-05	3.612E-05	3.587E-05	3.661E-05	3.67E-05	3.671E-05	3.624E-05	3.675E-05	3.637E-05	3.64E-05
700	3.62766E-05	3.703E-05	3.649E-05	3.605E-05	3.592E-05	3.658E-05	3.664E-05	3.641E-05	3.578E-05	3.669E-05	3.639E-05
800	3.65524E-05	3.597E-05	3.63E-05	3.615E-05	3.68E-05	3.589E-05	3.603E-05	3.686E-05	3.65E-05	3.666E-05	3.637E-05

Dari tabel Tabel 6.2 dapat dibuat sebuah grafik untuk melihat dengan jelas hasil dari uji coba banyaknya generasi terhadap nilai *fitness.* Dan grafik hasil uji coba banyaknya generasi dapat dilihat pada Gambar 6.2



Gambar 6.2 Grafik Percobaan Banyaknya Generasi

Berdasarkan Gambar 6.2 dapat dilihat bahwa ukuran populasi berpengaruh terhadap nilai *fitness* yang dihasilkan dari perhitungan algoritma genetika Nilai *fitness* paling rendah didapatkan pada ukuran generasi 300 karena proses algoritma genetika berproses secara optimal. Namun jumlah generasi yang tinggi tidak bisa dikatakan generasi terbaik karena belum tentu niali *fitness* yang dihasilkan paling optimal dan waktu untuk proses algoritma genetika lebih lama. Dapat dilihat pada Gambar 6.2, grafik mengalami kenaikan pada jumlah generasi

100 sampai 400, dilakukan 10 kali percobaan pada pengujian ukuran populasi untuk menghasilkan nilai rata-rata *fitness* dan dihasilkan *fitness* terbaik dengan nilai 0.00003642. Pada jumlah generasi 500 sampai 800 tidak terjadi kenaikan yang signifikan dan grafik hampir membentuk garis turun. Sehingga generasi paling optimal untuk kasus ini yaitu pada jumlah generasi 400.

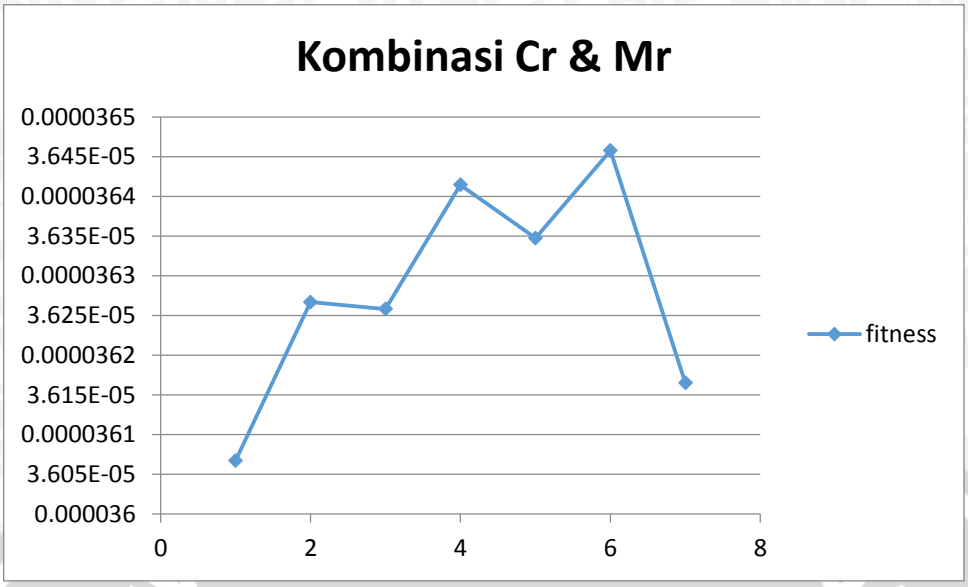
6.3 Pengujian dan Analisis Kombinasi *Crossover Rate* dan *Mutation Rate*

Pengujian terakhir adalah pengujian terhadap parameter kombinasi cr dan mr yang digunakan proses algoritma genetika. Pengujian kombinasi cr dan mr dilakukan untuk mengetahui pengaruh kombinasi cr dan mr terhadap nilai *fitness* yang dihasilkan. Bahan pakan yang digunakan pada pengujian ini adalah 3 bahan pakan. Kombinasi cr dan mr yang digunakan pada uji coba ini antara 0 sampai 0,6 dan untuk mendapatkan hasil yang lebih *valid* maka setiap percobaan akan dilakukan sebanyak 10 kali untuk diambil nilai rata-rata *fitness*nya. Pada percobaan ini digunakan populasi optimal pada uji coba sebelumnya yaitu ukuran populasi=240 dan generasi=400 yang didapatkan dari hasil uji coba sebelumnya. Tabel hasil uji coba kombinasi cr dan mr dapat dilihat pada Tabel 6.3.

Tabel 6.3 Hasil Uji Kombinasi Cr dan Mr

cr	mr	1	2	3	4	5	6	7	8	9	10	rata2
0.6	0	3.571E-05	3.571E-05	3.617E-05	3.617E-05	3.669E-05	3.599E-05	3.669E-05	3.577E-05	3.599E-05	3.577E-05	3.607E-05
0.5	0.1	3.671E-05	3.709E-05	3.66E-05	3.605E-05	3.639E-05	3.641E-05	3.639E-05	3.566E-05	3.577E-05	3.56E-05	3.627E-05
0.4	0.2	3.646E-05	3.619E-05	3.695E-05	3.578E-05	3.644E-05	3.565E-05	3.66E-05	3.571E-05	3.669E-05	3.611E-05	3.626E-05
0.3	0.3	3.665E-05	3.671E-05	3.61E-05	3.65E-05	3.626E-05	3.585E-05	3.659E-05	3.698E-05	3.672E-05	3.578E-05	3.641E-05
0.2	0.4	3.633E-05	3.695E-05	3.714E-05	3.631E-05	3.619E-05	3.698E-05	3.625E-05	3.591E-05	3.497E-05	3.643E-05	3.635E-05
0.1	0.5	3.673E-05	3.667E-05	3.626E-05	3.71E-05	3.653E-05	3.646E-05	3.625E-05	3.605E-05	3.573E-05	3.679E-05	3.646E-05
0	0.6	3.556E-05	3.595E-05	3.609E-05	3.641E-05	3.592E-05	3.703E-05	3.624E-05	3.6E-05	3.556E-05	3.689E-05	3.617E-05

Dari Tabel 6.3 dapat dibuat sebuah grafik untuk melihat dengan jelas hasil dari uji coba kombinasi cr dan mr terhadap nilai *fitness.* Dan grafik hasil uji coba kombinasi cr dan mr dapat dilihat pada Gambar 6.3.



Gambar 6.3 Grafik Uji Coba Kombinasi Cr dan Mr

Berdasarkan Gambar 6.3 dapat dilihat bahwa dapat dilihat bahwa nilai *fitness* terbaik ada pada kombinasi cr dan mr 0,1:0,5 dengan nilai *fitness* 0.00003646. Uji coba kombinasi cr dan mr berpengaruh terhadap nilai *fitness* yang dihasilkan. Tingkat *crossover rate* yang terlalu besar dan *mutation rate yang* terlalu kecil akan menurunkan kemampuan algoritma genetika untuk memperlebar area pencarian. Dan *crossover rate* yang kecil dan *mutation rate* yang besar akan menurunkan kemampuan algoritma genetika untuk belajar dari generasi sebelumnya dan tidak mampu untuk mengeksploitasi area *optimum local* (Mahmudy, WF, Marian, RM & Luong, LHS 2014).

6.4 Analisis Hasil Pengujian

Algoritma genetika merupakan algoritma yang akan mencari solusi mendekati optimal dari suatu permasalahan. Pada penelitian ini nilai fitness terbaik yang didapatkan yaitu 3,646E-05. Solusi ini didapatkan dengan nilai parameter ukuran populasi = 240, jumlah generasu = 400, dan kombinasi cr dan mr yaitu 0,1 dan 0,5.

Untuk mengukur kualitas solusi terbaik pada permasalahan optimasi komposisi pakan kambing potong yaitu dengan melihat nilai *fitness* tertinggi. *Fitness* tertinggi adalah *fitness* yang memiliki nilai *penalty* paling kecil dan harga paling minimum. Pada pengujian didapatkan nilai *fitness* tertinggi sebesar 3,646E-05 yang memiliki nilai *penalty* yaitu 0 dan harga yaitu Rp 28.000,-.

Setelah melakukan beberapa pengujian terhadap parameter algoritma genetika, algoritma genetika mampu menyelesaikan permasalahan optimasi komposisi pakan kambing potong. Dengan solusi yang diberikan maka para penyuluh peternakan dan para peternak dapat mengurangi biaya pengeluaran untuk pembelian bahan pakan ternak. Seekor kambing dengan bobot 40 kg dan target pertambahan berat badan 0,125 kg, jika diberi pakan hijauan dan konsentrat dengan komposisi yang telah diberikan oleh system yaitu kaliandra 5.484 kg , rumput raja 2,07 kg dan dedak padi kasar 2,45 kg.

BAB VII
KESIMPULAN DAN SARAN

7.1 Kesimpulan

Kesimpulan yang didapatkan dari hasil uji coba yang telah dilakukan mengenai penerapan algoritma genetika untuk menyelesaikan permasalahan optimasi komposisi pakan kambing potong yaitu sebagai berikut :

1. Algoritma genetika dapat diterapkan pada permasalahan optimasi komposisi pakan kambing potong dengan menggunakan representasi kromosom *real code*, metode *extended intermediate crossover*, *random mutation* dan *ellitsm selection.*
2. Untuk mengukur kualitas solusi terbaik pada permasalahan optimasi komposisi pakan kambing potong yaitu dengan melihat nilai *fitness* tertinggi. *Fitness* tertinggi adalah *fitness* yang memiliki nilai *penalty* paling kecil dan harga paling minimum. Pada pengujian didapatkan nilai *fitness* tertinggi sebesar 3,646E-05 yang memiliki nilai *penalty* yaitu 0 dan harga yaitu Rp 28.000,-.
3. Pengujian parameter genetika berpengaruh terhadap nilai *fitness* yang dihasilkan. Pengujian pada parameter genetika dilakukan dengan mengkombinasikan *crossover rate* (cr) dan *mutation rate* (mr) yang terbaik terdapat pada kombinasi cr dan mr 0,1:0,5 dengan nilai *fitness* 0.00003646. Kenaikan nilai *fitness* dipengaruhi oleh nilai *crossover rate* (cr) dan *mutation rate* (mr) yang semakin naik karena individu akan lebih sering mengalami persilangan sehingga terbentuk invidu yang bervariasi sehingga nilai *fitness* juga bervariasi. Parameter genetika yang juga diuji dalam penelitian ini adalah populasi dan generasi. Kedua parameter tersebut berpengaruh terhadap nilai *fitness* yang dihasilkan dari proses algoritma genetika. Ukuran populasi yang optimal terdapat pada ukuran 240 dengan nilai *fitness* 0,000036455. Sehingga dapat disimpulkan bahwa nilai *fitness* yang dihasilkan oleh sistem dipengaruhi oleh ukuran populasi yang semakin tinggi, namun jika ukuran populasi semakin tinggi maka tidak didapatkan kenaikan *fitness* yang

signifikan dan waktu komputasi yang digunakan untuk menghasilkan nilai *fitness* pun semakin lama. Pengujian jumlah generasi optimal dihasilkan *fitness* terbaik dengan nilai 0,00003642 dan terdapat pada jumlah generasi ke 400. Sehingga disimpulkam pada jumlah generasi yang terlalu rendah tidak didapatkan generasi terbaik sebagai solusi karena proses algoritma genetika berproses secara optimal. Namun jumlah generasi yang tinggi tidak bisa dikatakan generasi terbaik karena belum tentu niali *fitness* yang dihasilkan paling optimal dan waktu untuk proses algoritma genetika lebih lama.

7.2 Saran

Pada penelitian ini, terdapat beberapa hal yang dapat ditambahkan dan dikembangkan untuk penelitian selanjutnya antara lain :

1. Pada penelitian ini digunakan algoritma genetika untuk menyelesaiakan permasalahan optimasi komposisi bahan pakan kambing potong namun bisa juga menggunakan algoritma lain dalam menyelesaikan permasalahan ini.
2. Dapat menggunakan representasi kromosom lain selain *real code* yang digunakan pada penelitian ini. Dapat menggunakan metode *crossover* selain *extended intermediate crossover* dan metode mutasi yang selain *random mutation.* Dan dapat menggunakan metode seleksi yang lain selain *ellitsm.*
3. Jumlah bahan pakan bisa ditambahkan, dalam permasalahan ini hanya terdapat 30 jenis bahan pakan. Dan bisa menambahkan nutrisi yang digunakan, misal ditambahan Ca dan P.

DAFTAR PUSTAKA

Aribowo , Arnold dkk. 2008, *" Penerapan Algoritma Genetika Pada Penentuan Komposisi Pakan Ayam Petelur"*, Universitas Pelita Harapan, Indonesia.

Wardhani, Luh Kesuma. dkk. 2011, *" Optimasi Komposisi Bahan Pakan Ikan Air Tawar Menggunakan Metode Multi-Objective Genetic Algorithm"*, Fakultas Sains dan Teknologi, UIN Sultan Syarif Kasim, Riau.

Nugroho, Romada Adi. 2011 ,"*Optimalisasi Formulasi Pakan Ternak Terhadap Ayam Pedaging Dengan Menggunakan Metode Linear Programming*", Jurusan Teknik Industri, Fakultas Teknologi Indusri, Universitas Gunadarma, Jakarta

Mahmudy, WF. 2013, "*Algoritma Evolusi"*, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang.

Susilawati, Trinil dkk. 2011. *"Agribisnis kambing",*UB Press, Cetakan Pertama, Malang.

Dispet Jatim. 2002. *"Gizi Ternak Ruminansia Sesuai Stadia Fisiologis"*, Reorientasi Formulator Pakan Ternak, Jatim.

Murtidjo, Bambang Agus. 2008, *"Kambing sebagai Ternak Potong dan Perah"*, Kanisius, Cetakan Kesebelas, Jogyakarta.

Desiani, Anita dan Arhami, Muhammad. 2006, "*Konsep Kecerdasan Buatan",* ANDI OFFSET, Cetakan Pertama, Yogyakarta.

Mulyono, Subangkit dan Sarwono, B. 2008, *"Penggemukan Kambing Potong"*, Penebar Swadaya, Cetakan V, Depok.

Haryadi, NK. 2014, *"Teknik Jitu Penggemukan kambing Potong"*, Trans Idea Publishing, Cetakan Kedua, Jogjakarta.

Sari, AP, Mahmudy, WF & Dewi, C. 2014, *"Optimasi Asupan Gizi Pada Ibu Hamil dengan Menggunakan Algoritma Genetika"*, DORO: Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya, vol. 4, no. 5.

Pratiwi, MI, Mahmudy, WF & Dewi, C. 2014, 'Implementasi algoritma genetika pada optimasi biaya pemenuhan kebutuhan gizi', DORO: Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya, vol. 4, no. 6.

Zukhri, Zainudin. 2014, "*Algoritma Genetika Metode Komputasi Evolusioner untuk Menyelesaikan Masalah Optimasi*", C. VANDI OFFSET, Yogyakarta.

Mahmudy, WF, Marian, RM & Luong, LHS. 2014, *"Hybrid genetic algorithms for part type selection and machine loading problems with alternative production plans in flexible manufacturing system"*, *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, vol. 8, no. 1, pp. 80-93.

Siregar, Soribasya. 1996. *Sapi Perah Jenis, Teknik Pemeliharaan, dan Analisa Usaha*. PT Penebar Swadaya: Jakarta.

Edy. 2015, *"Penyusunan Ransum Kambing Potong"*, Bidang Studi Hijauan Makanan Ternak, SMK Peternakan, Dau.

Guawan, Hadi. 2013, *"Prospek Usaha Penggemukan Kambing Potong"*, Pustaka Baru Press, Cetakan Pertama, Yogyakarta.

Cahyono, Bambang. 2006, *"Beternak Domba dan Kambing"* , Kanisius , Cetakan Ke Sembilan, Yogyakarta.

Parakkasi, Aminuddin. 1999, "*Ilmu Nutrisi dan Makanan Ternak Ruminan*", Universitas Indonesia, Jakarta.

Kusumaningrum, Brilyan Indah. 2009, "*Kajian Kualitas Ransum Kambing Peranakan Ettawa Di Balai Pembibitan Dan Budidaya Ternak Ruminansia Kendal*" , Jurusan Nutrisi Dan Makanan Ternak Fakultas Peternakan Universitas Diponegoro, Semarang

NN . 2013*,"Dasar-Dasar Pakan Ternak"*, Buku Teks Bahan Ajar Siswa, Direktorat Pembinaan Sekolah Menengah Kejuruan Kementerian Pendidikan dan Kebudayaan Republik Indonesia.

Kartadisastra, H.R. 1997. *"Penyediaan dan Pengelolaan Pakan Ternak Ruminansia"*, Kanisius, Cetakan Ke Enam, Yogyakarta.

LAMPIRAN

No	Bahan Pakan	BK	PK	TDN
1	Gamal	27	25.2	76
2	Kaliandra	16	27.7	62
3	Lamtoro_K	86	23.7	71
4	Lamtoro_S	29	23.4	77
5	R_Benggala	24	5.4	53
6	R_Gajah	18	9.1	51
7	R_Raja	22	13.5	54
8	Tebon_Jagung	22	8	58
9	R_Pangola	23	8.3	53
10	R_Setaria	20	9.5	55
11	J_Padi	86	3.7	39
12	J_Kacang	86	14.7	56
13	J_Kedele	86	19.1	76
14	D_Pisang	16	14.4	70
15	D_Singkong	15	25	18
16	A_Nanas	20	3.4	68
17	A_Tahu	16.2	23.7	78
18	A_Sagu	80.4	1.2	58
19	BijiKapas	86	22.1	74.3
20	B_Kelapa	86	21.6	73
21	B_BijiSawit	86	15	70
22	DPadi_Kasar	86	7.6	14
23	DPadi_Halus	86	13.8	81
24	K_Coklat	88.9	14.6	47
25	Jg_Dedak	86	11.3	81
26	Jg_putih	86	10	81
27	Jg_Kuning	86	10.3	80
28	BijiKapuk_Tepung	86	31.7	74
29	Onggok	28.7	1.2	69
30	Tetes	77	5.4	53

No	Bahan Pakan	Harga
1	Gamal	2500
2	Kaliandra	2000
3	Lamtoro_K	3000
4	Lamtoro_S	2800
5	R_Benggala	3000
6	R_Gajah	3400
7	R_Raja	3500
8	Tebon_Jagung	2500
9	R_Pangola	2700
10	R_Setaria	3400
11	J_Padi	3300
12	J_Kacang	3000
13	J_Kedele	1600
14	D_Pisang	3500
15	D_Singkong	1400
16	A_Nanas	3600
17	A_Tahu	2500
18	A_Sagu	3200
19	BijiKapas	3400
20	B_Kelapa	3000
21	B_BijiSawit	3600
22	DPadi_Kasar	4000
23	DPadi_Halus	3700
24	K_Coklat	5000
25	Jg_Dedak	2800
26	Jg_putih	2550
27	Jg_Kuning	2600
28	BijiKapuk_Tepung	4650
29	Onggok	2000
30	Tetes	1450