OPTIMASI KOMPOSISI PAKAN BURUNG *LOVEBIRD* MENGGUNAKAN ALGORITME *PARTICLE SWARM OPTIMIZATION (PSO)*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Mauldy Putra Pratama
NIM: 145150207111038

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

OPTIMASI PEMBUATAN PAKAN BURUNG *LOVEBIRD* MENGGUNAKAN ALGORITME *PARTICLE SWARM OPTIMIZATION (PSO)*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Mauldy Putra Pratama
NIM: 145150207111038

Skripsi ini telah diuji dan dinyatakan lulus pada
1 Agustus 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Imam Cholissodin, S.Si., M.Kom	Dr. Muhammad Halim Natsir, S.Pt.,MP
NIK: 201201 850719 1 001	NIP : 197112241998021001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

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 disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam 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, 6 Agustus 2018

Mauldy Putra Pratama

NIM: 145150207111038

KATA PENGANTAR

Dengan segala kerendahan hati dan diri memanjatkan syukur kehadirat Allah SWT atas limpahan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan penulisan skripsi dengan baik. Shalawat serta salam semoga senantiasa terlimpahkan kepada Rasulullah Muhammad SAW beserta keluarga, sahabat-sahabat dan para pengikutnya.

Penulisan skripsi diajukan untuk memenuhi salah satu persyaratan dalam memperoleh gelar Sarjana Komputer pada Fakultas Ilmu Komputer Universitas Brawijaya. Topik skripsi yang diajukan adalah **"Optimasi Komposisi Pakan Burung *Lovebird* Menggunakan Algoritme *Particle Swarm Optimization* (PSO)".**

Kelancaran penulisan skripsi ini tidak lepas dari bantuan berbagai pihak, oleh karena itu penulis mengucapkan terima kasih dan rasa hormat kepada:

1. Bapak Imam Cholissodin, S.Si, M.Kom selaku dosen pembimbing I yang telah memberikan arahan dan menyediakan waktu untuk berdiskusi dengan penulis selama proses penulisan skripsi.
2. Bapak Dr. Muhammad Halim Natsir, S.Pt., MP selaku dosen pembimbing II yang telah memberikan banyak masukan, data nutrisi pakan dan motivasi kepada penulis selama proses penulisan skripsi.
3. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
5. Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
6. Peternak di Guyangan, Kecamatan Tlogomas, Kota Malang selaku narasumber dalam wawancara penentuan komposisi pakan burung *lovebird*.
7. Kedua orang tua penulis yang telah mengorbankan waktu, materiil dan memberikan motivasi yang tiada hentinya sehingga penulis dapat menyelesaikan skripsi penelitian ini. Semoga selalu diberikan kesehatan, rahmat dan ridho-Nya.
8. Seluruh dosen Fakultas Ilmu Komputer yang telah bersedia membagikan ilmunya sehingga penulis dapat menyelesaikan skripsi ini.
9. Teman-teman pengurus BIOS EXALT 2017-2018 yang telah memberikan motivasi, semangat dan membantu penulis dalam proses pengerjaan skripsi ini baik secara langsung maupun tidak langsung.
10. Teman-teman kelas informatika kelas D khususnya Muhammad Aghni Nur L. dan Falih Gozi Febrinanto yang telah memberikan motivasi, semangat dan

membantu penulis dalam proses pengerjaan skripsi ini baik secara langsung maupun tidak langsung.

11. Seluruh *civitas academica* Fakultas Ilmu Komputer Universitas Brawijaya.
12. Dan semua pihak tidak bisa disebutkan satu per satu yang telah membantu penulis dalam penulisan skripsi ini baik secara langsung maupun tidak langsung.

Penulis sadar bahwa skripsi ini masih banyak kekurangan, oleh karena itu kritik dan saran yang bersifat membangun sangat diharapkan untuk menyempurnakan skripsi ini. Penulis berharap skripsi ini dapat memberikan manfaat bagi diri sendiri dan bagi semua pihak.

Malang, 15 Juli 2018

Penulis
mauldyputra@gmail.com

ABSTRAK

Mauldy Putra Pratama, Optimasi Komposisi Pakan Burung *Lovebird* Menggunakan *Particle Swarm Optimization (PSO)*

Dosen Pembimbing: Imam Cholissodin, S.Si., M.Kom dan Dr. Muhammad Halim Natsir, S.Pt.,MP

Burung *lovebird* merupakan salah satu jenis burung yang banyak orang pelihara atau memperlombakannya. Burung ini membutuhkan asupan nutrisi yang cukup untuk keberlangsungan hidupnya. Banyak burung yang kelebihan dan kekurangan nutrisi dalam kenyataannya. Penelitian ini akan membahas bagaimana mengoptimasi komposisi pakan burung *lovebird* dan meminimalkan biaya tanpa mengurangi kebutuhan nutrisi. Metode optimasi, *Particle Swarm Optimization* (PSO) diterapkan pada proses formulasi dan komposisi pakan burung *lovebird* agar tetap memenuhi kebutuhan nutrisi dengan biaya yang minimal. Proses algoritme PSO dimulai dengan proses inisialisasi awal untuk nilai posisi, kecepatan dan *pBest* sebanyak jumlah partikel yang ditentukan serta *gBest*. Kemudian dilanjutkan ke tahap *update* kecepatan, posisi, *pBest* dan *gBest* sebanyak iterasi yang sudah ditentukan. Berdasarkan hasil pengujian yang telah dilakukan pada penelitian ini, diperoleh parameter optimal antara lain jumlah partikel sebanyak 60, jumlah iterasi sebanyak 600 dan nilai koefisien *k* sebesar 0,3. Dengan menggunakan parameter tersebut, selisih harga yang didapatkan adalah Rp 5.540,00.

Kata kunci: optimasi, pakan, komposisi, burung, *particle swarm optimization*

ABSTRACT

Mauldy Putra Pratama, Lovebird Feed Composition Optimization Using Particle Swarm Optimization (PSO)

Supervisors: Imam Cholissodin, S.Si., M.Kom and Dr. Muhammad Halim Natsir, S.Pt.,MP

Lovebird is a type of bird that many people have as a pet or for a competition. This bird needs adequate nutrients for survival. In reality, there are many of them that over or low nutrients. This study will discuss about how to optimized the composition feeds of lovebird and minimize the costs without reducing the nutrition. Particle Swarm Optimization (PSO) is one of the optimization methods that used for the formulation process and the composition to keep fulfilling the nutrient needs with minimal cost. The PSO algorithm process starts with the initialization process for position, velocity, pBest values as much as the particles and gBest. Then proceed to update the velocity, position, pBest and gBest as much as the iteration that has been determined. Based on the results of the test that have been done in this study, the writer obtains the optimal parameters such as the number of particles as much as 60, the number of iterations is 600 and the value of coefficient k is 0.3. By using these parameters, the price difference is Rp 5,540.00.

Keywords: *optimization, feed, composition, bird, particle swarm optimization*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Hewan peliharaan adalah hewan yang dipelihara manusia sebagai teman sehari-hari. Banyak manfaat yang bisa dirasakan jika kita memiliki hewan peliharaan seperti mengurangi stres (Barker, 2012), mengurangi depresi, memperbaiki suasana hati dan lain-lain. Ada banyak hewan yang bisa dipelihara untuk di rumah, misalnya kucing, anjing, hamster dan lain-lain. Selain hewan-hewan di atas, banyak juga orang yang senang untuk memelihara burung. Pada dasarnya, memelihara burung dapat memberikan kepuasan karena penampilannya, warna bulu dan kicauannya yang indah dan merdu (Hamiyanti, et al., 2011). Salah satu burung yang sering dijadikan hewan peliharaan adalah burung *lovebird.*

Burung *lovebird* memiliki ukuran yang kecil, yaitu sekitar 13 sampai 17 cm dengan berat 40 sampai 60 gram dan burung *lovebird* memiliki sifat sosial. Burung *lovebird* memiliki warna bulu yang cantik dan indah. Terdapat dua cara untuk mendapatkan warna bulu pada burung yaitu melalui pigmen dan pewarnaan struktural (Zhang, et al., 2014). Burung ini bisa menjadi sangat mahal apabila memiliki warna yang langka dan kicauannya. Selain itu, burung *lovebird* juga memiliki suara atau kicauan yang merdu dan lantang. Selain untuk menjadi peliharaan, burung ini dapat juga dilombakan kicauannya (Achmad, 2018). Jika burung dapat berkicau dengan durasi yang panjang maka burung tersebut bisa menjadi pemenang. Untuk dapat berkicau dengan durasi yang panjang maka perawatan dan pakan yang diberikan merupakan faktor yang dapat menentukan hal tersebut yang tentunya berbeda dengan perawatan dan pakan yang diberikan kepada burung yang akan dijadikan peliharaan saja. Dalam pakan burung dibutuhkan kandungan-kandungan nutrisi penting yang dibutuhkan burung agar dapat menunjang keberlangsungan hidup burung tersebut.

Setiap makhluk hidup membutuhkan kebutuhan nutrisi yang berbeda-beda, begitu juga dengan burung *lovebird*. Kebutuhan nutrisi yang dibutuhkan burung *lovebird* tergantung dari jenisnya, tujuan dari pemeliharaan apakah itu untuk dilombakan atau hanya untuk dipelihara dan umur dari burung itu sendiri (Natsir, 2018). Jika ketiga hal tersebut sudah ditentukan, maka pemberian pakan yang tepat bisa dilakukan untuk keberlangsungan hidup burung. Pakan yang dibutuhkan burung bermacam-macam. Terdapat dua kelompok dalam pakan burung yaitu pakan utama dan pakan tambahan (Natsir, 2018). Pakan utama meliputi millet merah, millet putih, biji kenari *(canary seed)*, biji oat dan lain-lain, sedangkan untuk pakan tambahan meliputi jagung, kecambah, kangkung, kuaci (biji bunga matahari) dan lain-lain. Dalam proses pertumbuhan burung juga memiliki tiga langkah yaitu burung muda, burung pertumbuhan dan burung dewasa di mana kebutuhan nutrisi yang dibutuhkan masing-masing langkah berbeda-beda. Hal-hal seperti ini haruslah diperhatikan dalam pemberian pakan untuk burung *lovebird*.

Harga pakan yang terbilang lumayan merogoh kocek para peternak menjadi salah satu masalah yang dihadapi oleh para peternak. Dari harga-harga yang ditawarkan oleh para pedagang, sebenarnya dengan biaya yang lebih sedikit dari harga yang ditawarkan, para peternak sudah dapat memenuhi kebutuhan nutrisi burung *lovebird*-nya. Jika burung kelebihan nutrisi atau pakan, burung bisa terkena penyakit seperti kegemukan, tidak bisa bertelur, birahi meningkat yang bisa menyebabkan burung untuk mencabut bulunya sendiri bahkan memakan anaknya sendiri (Achmad, 2018). Sebaliknya, jika burung kekurangan makanan maka burung akan kekurangan nutrisi sehingga dapat menyebabkan burung menjadi kurus bahkan mati karena kandungan nutrisi yang berada dalam pakan berguna sebagai energi yang digunakan burung untuk beraktivitas seperti terbang, berjalan dan lain-lain.

Dalam pemberian pakan haruslah dapat mencukupi kebutuhan nutrisi burung agar hal-hal di atas tidak terjadi. Kurangnya pengetahuan para peternak dalam pemberian pakan juga menjadi salah satu masalah yang dihadapi dalam pemberian pakan burung. Selama ini para peternak dalam pemberian pakan untuk burung masih mengira-ngira takaran yang diberikan sehingga takaran dalam pemberian pakan tidak pasti (Achmad, 2018). Karena itu dibutuhkan sebuah sistem yang dapat menghitung kebutuhan nutrisi pakan yang digunakan agar kebutuhan dapat tercukupi dan optimal.

Penelitian ini dilakukan untuk menentukan komposisi pakan burung *lovebird* yang dapat memenuhi nutrisi yang dibutuhkan serta memberikan solusi alternatif agar dapat mengganti bahan pakan sebagai upaya optimasi komposisi pakan burung *lovebird* dengan menggunakan algoritme PSO (*Particle Swarm Optimization)* dan juga bisa mengurangi biaya. PSO didasari dari sebuah ide yang setiap kerumunannya partikel merupakan solusi dari ruang solusi (Permana & Hashim, 2010). PSO digunakan untuk mencari nilai atau solusi yang optimal berbasis populasi. Algoritme PSO mempunyai kelebihan dalam hal konsep yang sederhana, mudah dalam implementasinya dan lebih efisien dalam melakukan perhitungan dibandingkan dengan teknik optimasi heuristik lainnya (Maickel, dkk., 2009). Selain itu, hasil akurasi yang didapatkan dengan menggunakan algoritme PSO lebih baik dibanding dengan menggunakan *Genetic Algoritm (GA)*. Posisi dan kecepatan partikel pada algoritme PSO dapat menghasilkan solusi baru yang lebih baik (Erny, 2013). Dari hal semua hal tersebut, penulis melakukan penelitian dengan judul "Optimasi Komposisi Pakan Burung *Lovebird* Dengan Menggunakan Algoritme *Particle Swarm Optimization*".

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijelaskan, maka permasalahan yang diangkat pada penelitian ini adalah:

1. Bagaimana cara menerapkan algoritme *Particle Swarm Optimization* (PSO) dalam mengoptimalkan komposisi pakan burung *lovebird?*
2. Bagaimana parameter algoritme *Particle Swarm Optimization* (PSO) yang optimal untuk menghasilkan komposisi pakan burung *lovebird* terbaik?

3. Bagaimana kualitas solusi algoritme *Particle Swarm Optimization* (PSO) dalam mengoptimasi komposisi pakan burung *lovebird?*

1.3 Tujuan

Tujuan dari penelitian ini adalah :

1. Mengetahui cara menerapkan algoritme *Particle Swarm Optimization* (PSO) dalam mengoptimalkan komposisi pakan burung *lovebird*.
2. Mengetahui parameter algoritme *Particle Swarm Optimization* (PSO) yang optimal untuk menghasilkan komposisi pakan burung *lovebird* terbaik.
3. Mengetahui kualitas solusi algoritme dari *Particle Swarm Optimization* (PSO) dalam mengoptimalkan komposisi pakan burung *lovebird*.

1.4 Manfaat

Penulis mengharapkan penelitian ini mendapatkan komposisi pakan burung *lovebird* yang optimal atau mendekati optimal bagi peternak maupun yang hobi memelihara burung *lovebird.* Untuk sisi akademis, penelitian ini dapat dimanfaatkan sebagai referensi pada penelitian-penelitian selanjutnya.

1.5 Batasan masalah

Penelitian ini di batasi oleh hal-hal sebagai berikut:

1. Pakar berasal dari Dosen Fakultas Peternakan Universitas Brawijaya, Malang.Data pakan burung didapatkan dari peternak burung *lovebird* di Tlogomas, Malang.
2. Data pakan burung didapatkan dari peternak burung *lovebird* di Tlogomas, Malang.
3. Optimasi komposisi bahan pakan sesuai dengan referensi dari peternak burung *lovebird.*
4. Terdapat 17 bahan pakan burung *lovebird*.
5. Metode yang dipakai adalah *Particle Swarm Optimization* (PSO).

1.6 Sistematika pembahasan

Pada penelitian ini, terdapat beberapa bab secara garis besar, antara lain :

BAB 1 PENDAHULUAN

Berisi terkait dengan latar belakang penelitian, rumusan masalah, tujuan dan manfaat penelitian ini, batasan masalah serta sistematika penulisan dari Optimasi Komposisi Pakan Burung *Lovebird* Menggunakan Algoritme *Particle Swarm Optimization (PSO).*

BAB 2 LANDASAN KEPUSTAKAAN

Menguraikan teori-teori dan referensi yang erat hubungannya dengan burung *lovebird* dan algoritme *Particle Swarm Optimization (PSO)* yang mendukung pada penelitian ini.

BAB 3 METODOLOGI

Pada bab ini akan dijelaskan mengenai metode dan tahapan-tahapan perancangan dan implementasi yang dilakukan pada saat meneliti Optimasi Komposisi Pakan Burung *Lovebird* Menggunakan Algoritme *Particle Swarm Optimization (PSO).*

BAB 4 PERANCANGAN

Membahas tentang merancang sistem optimasi komposisi pakan burung *lovebird* menggunakan algoritme *Particle Swarm Optimization* (PSO).

BAB 5 IMPLEMENTASI

Bab ini membahas pengimplementasian algoritme *Particle Swarm Optimization* (PSO) untuk mengoptimasi komposisi pakan burung *lovebird*.

BAB 6 PENGUJIAN DAN ANALISIS

Berisi tentang pengujian-pengujian yang dilakukan serta analisis hasil dari pengujian tersebut

BAB 7 PENUTUP

Berisi kesimpulan dari penelitian yang telah dilakukan serta saran yang sekiranya dapat membantu untuk penelitian-penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Penelitian meliputi penentuan komposisi pakan ternak untuk memenuhi kebutuhan nutrisi ayam petelur dengan biaya minimum menggunakan *Particle Swarm Optimization* (PSO), optimasi formulasi pakan pada proses budidaya ikan bandeng menggunakan *Particle Swarm Optimization* (PSO) dan optimasi komposisi pakan sapi perah menggunakan algoritme *Particle Swarm Optimization* (PSO) merupakan kajian yang dilakukan terhadap penelitian-penelitian sebelumnya.

Penelitian pertama yang dilakukan oleh Esmin, et al. (2015) dalam me-*review* algoritme PSO dan variannya dalam *clustering high-dimensional data.* Penelitian ini bertujuan untuk memberikan survei dengan mengulas algoritme PSO dan variannya yang akan digunakan pada *clustering* data yang besar dan *high-dimensional*. Hasil dari penelitian ini adalah Penelitian membuktikan bahwa *Particle Swarm Optimization* (PSO) dapat digunakan pada *cluster* dalam ruang *multi-dimensional* dengan menunjukkan performa yang baik. Tetapi, tingkat konvergensi pada saat mencari *gBest* masih kurang baik. Selanjutnya, PSO yang sudah dimodifikasi dan di *hybrid* dengan algoritme lain akan berhasil dalam memecahkan *clustering high-dimensional data* dan memberikan hasil yang lebih baik.

Penelitian kedua yang dilakukan oleh Khaqqo, et al. (2016) dalam menggunakan algoritme *Particle Swarm Optimization* (PSO) dalam mengoptimalkan komposisi pakan sapi perah. Penelitian ini bertujuan untuk menambah produksi susu sapi tetapi biaya dalam pembuatan pakan sapi bisa di hemat agar kesehatan dan reproduksi sapi perah tetap terpenuhi. Dari penelitian yang telah dilakukan didapatkan hasil, inisialisasi partikel dengan menggunakan bahan pakan yang sudah ditentukan sebagai panjang dimensi, menghitung nilai *fitness* setiap partikel, meng-*update* posisi dan kecepatan partikel pada setiap iterasi, solusi terbaik tiap partikel ditentukan dari nilai *personal best* serta solusi terbaik dari semua partikel yang didapatkan dari penentuan *global best* di mana nilai *fitness global best* iterasi terakhir menjadi solusi. Di mana parameter yang digunakan adalah rata-rata nilai *fitness* terbesar 1,523184, nilai w_1 = 0,2 dan w_2 = 0,9 yang nilai *fitness*-nya 1,26156177, kombinasi akselerasi c_1 dan c_2 masing-masing sebesar 2, 350 iterasi yang nilai *fitness*-nya 1,47298298 dan ukuran *swarm* sebanyak 400 partikel yang nilai *fitness*-nya 1,44519179. Perbandingan komposisi pakan sapi perah didapatkan dengan membandingkan hasil observasi dengan rekomendasi dari sistem. Dari perbandingan tersebut dihasilkan pakan sapi perah yang sistem rekomendasikan dapat menghemat biaya pakan kurang lebih Rp2.600,00 serta meningkatkan produksi susu sekitar 0,57 liter di mana nutrisi yang sapi butuh tetap terpenuhi.

Penelitian selanjutnya yang dilakukan oleh Wardhany, et al. (2017) dalam menggunakan algoritme *Particle Swarm Optimization* (PSO) yang bertujuan untuk menentukan komposisi bahan pakan ternak untuk memenuhi kebutuhan nutrisi

ayam petelur dengan biaya minimum. Dari pengujian yang telah dilakukan, peneliti mendapatkan hasil setelah iterasi 330 dengan nilai *fitness* sebesar 4,02190223 yang bisa menghemat biaya sebesar 42% atau setara dengan Rp226.77 tetapi tetap dapat memenuhi kebutuhan nutrisi ayam dengan nilai penalti sebesar 9.746823404%. Untuk mendapatkan hasil tersebut, parameter yang diuji adalah ukuran *swarm* sebanyak 350, jumlah iterasi sebanyak 500, w_{max} sebesar 0.9, w_{min} sebesar 0.4, c_{1i} & c_{1f} sebesar 2.5 & 0.5 serta c_{2i} & c_{2f} sebesar 0.5 & 2.5.

Penelitian selanjutnya yang dilakukan oleh Darmawan, et al. (2017) dalam menggunakan algoritme *Particle Swarm Optimization* (PSO) yang bertujuan untuk mengoptimalkan formulasi pakan pada proses budidaya ikan bandeng. Dari pengujian yang telah dilakukan yaitu dengan pengujian 100 partikel, 70 iterasi, batas atas dan batas bawah partikel sebesar 9 dan 1 serta nilai koefisien *k* sebesar 0,4. Dengan total biaya Rp15.017,625 dan nilai *fitness* 4,635699E^{-5}, komposisi pakan ikan terbaik didapatkan untuk ikan bandeng berusia 10 minggu dengan jumlah populasi sebesar 500 serta berat 0,25 kg per ekornya ialah tepung daun lamtoro sebanyak 2,129 kg, tepung gaplek sebanyak 1,384 kg dan tepung ikan sebanyak 0,237 kg.

Tabel 2.1 Kajian Pustaka

No.	Judul	Objek	Metode	Hasil
1.	*A Review on Particle Swarm Optimization Algorithm and Its Variants to Clustering High-Dimensional Data* (Esmin, et al., 2015)	Algoritme *Particle Swarm Optimization*	Algoritme *Particle Swarm Optimization*	Penelitian membuktikan bahwa *Particle Swarm Optimization* (PSO) dapat digunakan pada *cluster* dalam ruang *multi-dimensional* dengan menunjukkan performa yang baik. Tetapi, tingkat konvergensi pada saat mencari *gBest* masih kurang baik. Selanjutnya, PSO yang sudah dimodifikasi dan di *hybrid* dengan algoritme lain akan berhasil dalam memecahkan *clustering high-dimensional data* dan memberikan hasil yang lebih baik.

2.	Optimasi Komposisi Pakan Sapi Perah Menggunakan Algoritme *Particle Swarm Optimization* (PSO) (Khaqqo, et al., 2016)	Kebutuhan komposisi dan nutrisi pakan sapi perah	Algoritme *Particle Swarm Optimization*	Pengujian yang dilakukan mendapatkan rata-rata *fitness* terbesar 1.523184 dengan w_1 = 0.2 dan w_2 = 0.9, kombinasi c_1 dan c_2 = 2, banyaknya ukuran *swarm* yaitu 400 dan 350 iterasi yang dilakukan, membuktikan bahwa biaya pakan dan penambahan produksi susu bisa di hemat dengan PSO.
3.	Penentuan Komposisi Pakan Ternak untuk Memenuhi Kebutuhan Nutrisi Ayam Petelur dengan Biaya Minimum Menggunakan Algoritme *Particle Swarm Optimization* (PSO) (Wardhany, et al., 2017)	Kebutuhan komposisi dan nutrisi pakan ayam petelur	Algoritme *Particle Swarm Optimization*	Pengujian ukuran swarm, bobot inersia, koefisien akselerasi dan jumlah iterasi mendapatkan hasil yang konvergen setelah iterasi ke 330 dengan nilai *fitness* sebesar 4.2190223 yang bisa menghemat biaya pakan sebesar 42% atau Rp226.77.-. Parameter yang paling optimal pada penelitian ini adalah ukuran *swarm* sebesar 350, c_{2i} & c_{2f} = 0.5 & 2.5, jumlah iterasi sebanyak 500, w_{max} = 0.9, w_{min} = 0.4 dan c_{1i} & c_{1f} = 2.5 & 0.5
4.	Optimasi Formulasi Pakan Pada Proses Budidaya Ikan Bandeng Menggunakan *Particle Swarm*	Kebutuhan komposisi dan nutrisi ikan bandeng	Algoritme *Particle Swarm Optimization*	Tepung daun lamtoro sebanyak 2.129 kg, tepung gaplek sebanyak 1.384 kg dan tepung ikan 0.237 kg dengan total biaya Rp15.017,625

	Optimization (PSO) (Darmawan, et al., 2017)			merupakan hasil dari pengujian 100 partikel, 70 iterasi, x_{max} dan x_{min} masing-masing sebesar 9 dan 1 serta nilai koefisien *k* sebesar 0,4.

2.2 Burung *Lovebird*

Burung *lovebird* berasal dari Afrika dan pertama kali diimpor ke Eropa pada tahun 1800-an. Burung yang merupakan genus *Agarponis* ini merupakan burung yang berukuran kecil, antara 13 cm sampai 17 cm dengan berat mencapai 40 gr sampai 60 gr. Burung ini memiliki 9 spesies yaitu:

1. *Agarponis roseicollis*
2. *Agarponis personate*
3. *Agarponis fischeri*
4. *Agarponis lilianae*
5. *Agarponis nigrigenis*
6. *Agarponis cana*
7. *Agarponis taranta*
8. *Agarponis pullaria*
9. *Agarponis swindernia*

Masa mengerami telur burung *lovebird* adalah selama 2 minggu dan bisa dibuahi kembali setelah 1,5 bulan (Achmad, 2018). Setelah telur menetas, bayi *lovebird* diberi bubur bayi dengan campuran bahan-bahan lain. Harga burung ini beragam, mulai dari harga ±Rp150.000 sampai jutaan rupiah.

Gambar 2.1 *Agarponis fischeri*

2.2.2 Kebutuhan Nutrisi Burung *Lovebird*

Nutrisi adalah substansi organik yang dibutuhkan organisme untuk fungsi normal dari sistem tubuh, pertumbuhan dan pemeliharaan kesehatan (Afrianto & Leviawaty, 2005). Nutrisi didapat dari makanan dan cairan yang selanjutnya diasimilasikan oleh tubuh.

Nutrisi yang dibutuhkan burung sebagai pakan adalah (Natsir, 2018):

1. Protein

Gambar 2.2 Sumber Protein

Protein ini bermanfaat sebagai bahan dasar pembentukan sel-sel dan jaringan baru dalam tubuh burung. Selain untuk pembentukan sel, protein juga bermanfaat untuk pertumbuhan dan perbaikan jaringan tubuh yang rusak serta sebagai penopang sumber energi dalam pembentukan telur, daging dan pertunasan bulu-bulu baru. Protein dibutuhkan burung sebesar 35%. Sumber protein bisa didapatkan dari nabati dan hewani. Untuk sumber protein nabati bisa dari taoge, sedangkan untuk protein hewani bisa dari kroto, ulat daun dan lain-lain.

2. Karbohidrat

Gambar 2.3 Sumber Karbohidrat

Karbohidrat adalah sumber energi yang burung butuhkan agar bisa beraktivitas sehari-hari. Karbohidrat berfungsi agar burung dapat bergerak, berkicau, membantu kelancaran metabolisme dan pembentukan sel darah merah. Karbohidrat yang dapat dicerna dan diserap bagi burung adalah buah-buahan, jagung muda dan sayuran.

3. Lemak

Gambar 2.4 Sumber Lemak

Lemak yang dibutuhkan burung tidak boleh melebihi 8%. Lemak ini berfungsi sebagai sumber energi selain karbohidrat, membawa vitamin (A, D, E dan K), melindungi organ tubuh, untuk menghangatkan tubuh dan lain-lain. Lemak menghasilkan energi yang lebih besar dibandingkan dengan karbohidrat, tetapi

jika terlalu banyak lemak yang dicerna bisa membuat burung menjadi malas, bulu rontok sebelum waktunya bahkan terkena serangan jantung.

4. Fosfor

Fosfor dapat ditemukan pada kuaci (biji bunga matahari). Akibat yang dapat terjadi jika kekurangan fosfor ialah keroposnya tulang belakang dan turunnya pertumbuhan. Pelepasan kalsium tulang dan terbentuknya kalsium karbonat merupakan contoh dari kelebihan fosfor (Marginingtyas, et al., 2015).

5. Serat Kasar

Serat kasar dikategorikan sebagai serat yang dapat larut dan tidak dapat larut berdasarkan dari manfaat nutrisinya. Manfaat serat kasar adalah untuk melindungi saluran pencernaan agar tetap bekerja sesuai dengan fungsinya serta berguna untuk memperbaiki fungsi serapan nutrisi agar nutrisi tercukupi. Kanibalisme juga dapat dicegah dengan serat kasar (Marginingtyas, et al., 2015).

Untuk menghitung kadar nutrisi protein, karbohidrat, lemak, fosfor dan serat kasar dapat menggunakan Persamaan 2.1.

$$Nutrisi(\%) = \frac{Bobot\ pakan_{i,j}(\%)}{100} \times kadar\ nutrisi\ bahan\ i\ (\%) \quad (2.1)$$

6. Energi Metabolisme (ME)

Semua makhluk hidup pasti membutuhkan energi untuk melakukan sesuatu. Sumber penyusun energi metabolisme terdiri dari karbohidrat, lemak dan protein. Komponen-komponen tersebut bisa didapatkan dengan mengonsumsi semua biji-bijian, seperti jagung, millet, kuaci dan lain-lain. Komponen pertama yang diproses untuk menjadi energi adalah karbohidrat. Apabila karbohidrat masih kurang dalam menghasilkan energi, maka lemak yang akan diproses untuk menjadi energi. Apabila tetap kurang, maka memproses protein untuk menjadi energi langkah terakhir yang dilakukan walaupun tidak efisien (Marginingtyas, et al., 2015). Untuk menghitung kadar ME dapat menggunakan Persamaan 2.2.

$$ME\left(\frac{Kkal}{kg}\right) = \frac{Bobot\ pakan_{i,j}(\%)}{100} \times kadar\ ME\ bahan\ i\ (\frac{Kkal}{kg}) \quad (2.2)$$

2.2.3 Bahan dan Harga Pakan Burung *Lovebird*

Dalam pemberian pakan burung *lovebird*, terdapat dua kelompok pakan burung yaitu pakan utama dan pakan tambahan. Pakan utama meliputi millet merah, millet putih, biji kenari *(canary seed)*, biji oat dan lain-lain, sedangkan untuk pakan tambahan meliputi jagung, kecambah, kangkung, kuaci (biji bunga matahari) dan lain-lain. Bahan pakan memiliki harga yang beragam baik itu eceran ataupun per karung. Untuk lebih lengkapnya dapat dilihat pada Lampiran A.1 sampai Lampiran A.4.

Menurut wawancara dengan peternak, pemberian pakan dengan menggunakan millet putih, millet merah, kuaci dan *canary seed* dicampur menjadi 1. Untuk jagung, 1 jagung di potong menjadi 3 bagian. Dalam pencampuran millet putih, millet merah, kuaci dan *canary seed* perbandingannya adalah 25kg millet putih banding 2kg kuaci banding 3 kali dan untuk pengambilan millet merah menggunakan tangan. Pakan tersebut diberikan sekali sehari setiap harinya.

2.3 Algoritme *Particle Swarm Optimization* (PSO)

Particle Swarm Optimization (PSO) adalah suatu algoritme yang terinspirasi dari perilaku sekawanan burung dalam hal bekerja sama dan berkomunikasi. Kecerdasan yang muncul dari perilaku tersebut menyebabkan kawanan burung membentuk pola global yang kompleks. Algoritme merupakan salah satu cabang dalam *Swarm Intelligence* dalam memecahkan permasalahan optimasi yang ditemukan oleh Russel Eberhart dan James Kennedy pada tahun 1995 (Cholissodin & Riyandani, 2016).

2.3.1 Komponen-Komponen Algoritme PSO

Terdapat empat komponen utama yang dimiliki PSO, yaitu kecepatan, partikel, komponen kognitif dan komponen sosial. Terdapat dua faktor pembelajaran partikel, yaitu kombinasi pembelajaran keseluruhan kelompok (*social learning*) dan pengalaman partikel (*cognitive learning*). Posisi terbaik yang pernah dicapai sebuah partikel merupakan *Cognitive learning* sebagai *pBest* (*personal best*), sedangkan *social learning* sebagai *gBest* (*global best*) adalah posisi terbaik dari keseluruhan partikel dalam kelompok. *pBest (personal best)* dan *gBest* (*global best*) untuk menghitung kecepatan partikel dan kecepatan untuk menghitung posisi selanjutnya.

Penjelasan detail komponen algoritme PSO adalah sebagai berikut:

1. *Swarm*
   Swarm adalah banyaknya partikel dalam populasi pada suatu algoritme. Kompleksnya masalah yang dihadapi bergantung pada ukuran *swarm*. *Swarm* merepresentasikan banyaknya partikel yang ada dalam suatu permasalahn.
2. Partikel
   Merupakan individu dalam *swarm* yang merepresentasikan solusi penyelesaian masalah. Representasi solusi menentukan posisi dan kecepatan setiap partikel pada saat itu.
3. *Personal Best (pBest)*
   Perbandingan antara *fitness* pada posisi partikel sekarang dan sebelumnya dapat menghasilkan posisi terbaik yang pernah dicapai (*pBest)*.
4. *Global Best (gBest)*
   Perbandingan nilai *fitness* terbaik dari keseluruhan partikel dalam *swarm* dapat menghasilkan posisis terbaik partikel (*gBest*).
5. Kecepatan (*velocity*)
   Arah perpindahan posisi partikel dapat ditentukan oleh kecepatan. Posisi partikel saat ini diperbaiki dengan adanya perubahan kecepatan setiap iterasi.
6. Bobot inersia (*inertia weight*)
   Perubahan kecepatan yang diberikan oleh partikel dikontrol dengan menggunakan bobot inersia.
7. Koefisien akselerasi

Sejauh mana suatu partikel dalam satu iterasi dapat menggunakan koefisien akselerasi ini. Nilai ini dapat ditentukan sendiri.

2.3.2 Langkah-Langkah Algoritme PSO

Proses algoritme *Particle Swarm Optimization* dalam mencari solusi terbaik adalah sebagai berikut :

1. Inisialisasi
   Inisialisasi dilakukan untuk membangkitkan himpunan solusi baru secara acak yang terdiri atas sejumlah *string* dimensi partikel dan ditempatkan pada penampungan yang disebut populasi. Pada langkah ini harus ditentukan ukuran populasi (*popSize).* Nilai ini menyatakan banyaknya individu/partikel yang berada dalam populasi.

   a. Inisialisasi kecepatan awal partikel
      Pada iterasi awal (*t*=0) nilai kecepatan semua partikel adalah 0 dan konversi partikelnya menjadi *x*.

   b. Inisialisasi posisi awal partikel
      Nilai *x* ditentukan secara acak dengan interval. Interval tersebut memiliki batas bawah dan batas atas. Pada penelitian ini, batas atas dapat ditentukan oleh kita sendiri atau sesuai dengan masukkan dari *user* (Natsir, 2018). Setelah batas bawah dan batas atas ditentukan, maka inisialisasi posisi partikel bisa dihitung dengan menggunakan Persamaan 2.3.

      $$X_{i,j}(t) = X_{min} + rand[0,1] * (X_{max} - X_{min}) \quad (2.3)$$

      Di mana:

      X_{min} = batas bawah posisi
      X_{max} = batas atas posisi
      $rand[0,1]$ = nilai acak antara 0 dan 1

   c. Perhitungan nilai *fitness*
      Pada perhitungan nilai *fitness*, terdapat 3 proses yaitu:

      - Normalisasi nutrisi pakan
        Untuk mendapatkan nilai normalisasi nutrisi pakan dari total bobot yang belum mencapai 100%, maka dapat menggunakan Persamaan 2.4 untuk normalisasi.

        $$Normalisasi\ bobot\ pakan_j(\%) = \frac{bobot\ bahan_{i,j(\%)}}{Total\ bobot\ bahan} \times 100\% \quad (2.4)$$

      - Menghitung penalti
        Untuk menghitung penalti dihitung dengan cara nilai total dibandingkan dengan kebutuhan nutrisi di mana dapat didapatkan dengan menggunakan Persamaan 2.5.

        $$penalti\begin{Bmatrix} 0, totalNutrisi \geq KebNutisi \\ KebNutrisi - TotalNutrisi, TotalNutrisi < KebNutrisi \end{Bmatrix} \quad (2.5)$$

      - Menghitung harga pakan

Untuk menghitung harga pakan dapat menggunakan Persamaan 2.6.

$$harga = \left(\frac{bobot\ bahan_{i,j}}{100} \times kebutuhan\ pakan/hari\right) \times harga\ pakan \quad (2.6)$$

- Menghitung *Fitness*
  Setelah mendapatkan nilai-nilai di atas, maka nilai *fitness* sudah bisa didapatkan dengan menggunakan Persamaan 2.7.

$$Fitness = \left(\frac{1}{(harga\ pakan \times \alpha) \times (penalti \times \beta)}\right) \times K \quad (2.7)$$

  Di mana nilai α dan β adalah konstanta pengali bernilai 20 agar selisih antara harga dan penalti pakan tidak terlalu besar. Agar nilai *fitness* tidak terlalu kecil, maka konstanta pengali(konstanta *K*) bernilai 10000.

d. Inisialisasi *pBest* dan *gBest*
   Untuk inisialisasi *pBest* didapatkan dari posisi awal partikel dan *fitness*-nya. Sedangkan untuk *gBest* didapatkan dari nilai *fitness* terbaik dari seluruh partikel.

e. Menentukan batas kecepatan
   Batas kecepatan dapat ditentukan dengan menggunakan Persamaan 2.8 dan 2.9.

$$V_{max} = k \times \left(\frac{(x_{max} - x_{min})}{2}\right) \quad (2.8)$$

$$V_{min} = -V_{max} \quad (2.9)$$

   Di mana:
   X_{min} = batas bawah posisi
   X_{max} = batas atas posisi
   V_{min} = batas bawah kecepatan
   V_{max} = batas atas kecepatan
   k = nilai acak antara 0 dan 1.

2. *Update* Kecepatan
   Arah perpindahan posisi partikel yang ada di populasi ditentukan oleh *update* kecepatan. Untuk menghitung *update* kecepatan dapat menggunakan Persamaan 2.10.

$$v_{i,j}^{t+1} = w.\, v_{i,j}^{t} + c_1 r_1 (pBest_{i,j}^{t} - x_{i,j}^{t}) + c_2 r_2 (gBest_{g,j}^{t} - x_{i,j}^{t}) \quad (2.10)$$

   Di mana:
   $v_{i,j}^{t+1}$ = kecepatan partikel ke-*i*, pada dimensi ke-*j,* pada iterasi ke *t.*
   w = bobot inersia.
   c_1 = parameter kognitif.
   c_2 = parameter sosial.
   r_1 dan r_2 = nilai acak dengan rentang [0,1].
   $x_{i,j}^{t}$ = posisi partikel ke-*i*, pada dimensi ke-*j,* pada iterasi ke *t.*
   $pBest_{i,j}^{t}$ = posisi terbaik partikel ke-*i*, pada dimensi ke-*j,* pada iterasi ke *t.*

$gBest_{g,j}^{t}$ = nilai *pBest* terbaik dari populasi.

Setelah melakukan hitungan Persamaan 2.8, apabila kecepatan melebihi dari V_{max} maka nilainya akan diubah menjadi V_{max} dan apabila nilai kecepatan kurang dari V_{min} maka nilainya akan diubah menjadi V_{min}. Penjelasan di atas sama dengan Persamaan 2.11 dan Persamaan 2.12.

$$jika\ V_{i,j}(t+1) > V_{max}\ maka\ V_{i,j}(t+1) = V_{max} \quad (2.11)$$

$$jika\ V_{i,j}(t+1) < V_{min}\ maka\ V_{i,j}(t+1) = V_{min} \quad (2.12)$$

3. Update Posisi
   Persamaan 2.13 dan Persamaan 2.14 dapat digunakan untuk proses *update* posisi.

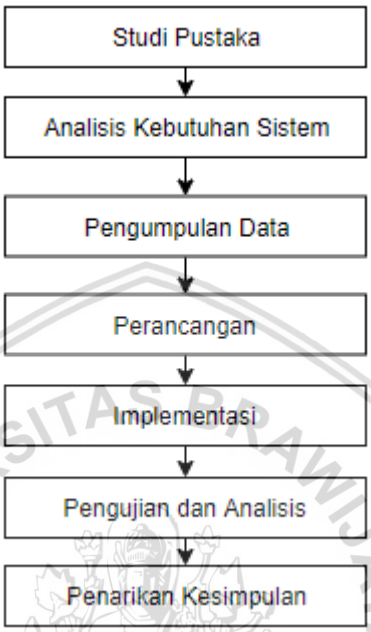
$$x_{i,j}^{t+1} = x_{i,j}^{t} + v_{i,j}^{t+1} \quad (2.13)$$

$$x_{i,j}^{t+1} = \begin{cases} jika\ x_{i,j}^{t} + v_{i,j}^{t+1} > x_{max}\ maka\ x_{max} \\ jika\ x_{i,j}^{t} + v_{i,j}^{t+1} < x_{min}\ maka\ x_{min} \end{cases} \quad (2.14)$$

4. Update *pBest* dan *gBest*
   Untuk memperbarui nilai *pBest* didapatkan dari melihat apakah nilai *fitness* pada iterasi sekarang lebih baik dari iterasi sebelumnya. Nilai *gBest* didapatkan dari nilai *pBest* dengan nilai *fitness* tertinggi.

BAB 3 METODOLOGI

Metodologi penelitian dalam penelitian ini dibagi menjadi langkah-langkah yang dilakukan dalam perhitungan optimasi. Gambar 3.1 merupakan diagram alir metodologi penelitian.



Gambar 3.1 Diagram Alir Metodologi Penelitian

3.1 Studi Pustaka

Dasar teori yang telah dijelaskan pada bab sebelumnya merupakan studi literatur yang digunakan dalam menunjang penelitian. Teori-teori tersebut dapat dikumpulkan dari berbagai macam sumber, dapat melalui buku, skripsi sebelumnya, jurnal ilmiah atau media yang lainnya. Dari sumber-sumber tersebut, peneliti bisa mendapatkan data tentang kandungan nutrisi pakan yang dibutuhkan oleh burung *lovebird,* harga bahan untuk pembuatan pakan serta tentang algoritme yang dipakai dalam penelitian yaitu *Particle Swarm Optimization* (PSO).

3.2 Analisis Kebutuhan

Kebutuhan-kebutuhan apa saja yang dibutuhkan oleh peneliti dalam melakukan penelitian ini baik itu perangkat lunak (*software*) maupun perangkat keras (*hardware*) ditentukan oleh analisis ini.

Analisis ini dilakukan untuk mengetahui kebutuhan-kebutuhan apa saja yang dibutuhkan oleh peneliti dalam melakukan penelitian ini baik itu perangkat keras (*hardware*) maupun perangkat lunak (*software*).

Penelitian ini menggunakan perangkat keras (*hardware*) maupun perangkat lunak (*software*) sebagai berikut:

1. Perangkat Keras (*Hardware*), meliputi:

- Laptop dengan spesifikasi, RAM 8 GB, Harddisk 1 TB dan monitor 15.6 inch.

2. Perangkat Lunak (*Software*), meliputi:
   - *Microsoft Windows* 10 64 bit.
   - *Netbeans IDE* 8.2.
   - *Website* draw.io dan Microsoft Word sebagai alat dalam pembuatan diagram.

3.3 Pengumpulan Data

Penelitian ini menggunakan data didapatkan dari beberapa sumber. Data-data yang sudah didapatkan akan digunakan dalam penelitian ini untuk perhitungan dan analisis hasil dari optimasi yang dilakukan. Data-data tersebut adalah:

- Data jenis pakan yang digunakan didapatkan melalui wawancara dengan peternak dan pakar pada Februari 2018.
- Data harga jenis dan kandungan nutrisi pakan untuk burung *lovebird* sebanyak 17 data.
- Data kebutuhan nutrisi burung *lovebird* sebanyak 6 fitur.

3.4 Perancangan

Perancangan sistem dibuat berdasarkan pengumpulan data yang dilakukan dan hasil dari analisis kebutuhan. Perancangan ini dilakukan untuk menyelesaikan masalah optimasi komposisi pakan burung *lovebird* dengan menggunakan *Particle Swarm Optimization* (PSO), penggunaan parameter dan manualisasi dengan PSO. Perancangan yang dilakukan adalah perancangan sistem. Perancangan sistem adalah perancangan yang dilakukan untuk merancang sistem yang dapat mengimplementasikan algoritme PSO.

3.5 Implementasi

Implementasi sistem adalah menerapkan hal yang telah didapatkan pada Bab Landasan Kepustakaan dan tahap membangun sistem yang sesuai dengan perancangan yang telah dibuat. Java sebagai *platform* pengembangan digunakan untuk membangun sistem optimasi komposisi pakan burung *lovebird* menggunakan algoritme. Implementasi algoritme digunakan pada tahap implementasi. Komposisi pakan burung *lovebird* yang optimal menghasilkan keluaran dari implementasi sistem.

3.6 Pengujian dan Analisis

Sistem yang telah dibangun diuji untuk mengetahui apakah sistem sinkron dengan kebutuhan dan spesifikasi. Parameter optimal dalam implementasi sistem ditentukan oleh uji coba yang dilakukan pada tahap perancangan pengujian sistem.

3.7 Penarikan Kesimpulan

Apabila semua tahapan mulai dari perancangan, implementasi dan pengujian dari sistem optimasi komposisi pakan burung *lovebird* menggunakan algoritme *Particle Swarm Optimization* (PSO) dilakukan maka tahap pengambilan keputusan dilakukan. Hasil pengujian dan analisis sistem dapat menghasilkan kesimpulan. Saran yang berhubungan dengan hasil yang telah tercapai yang bermanfaat merupakan tahap terakhir dari penulisan penelitian ini.

BAB 4 PERANCANGAN

Bab Perancangan ini membahas perihal perancangan sistem penelitian ini, mencakup formulasi permasalahan, alur algoritme *Particle Swarm Optimization* (PSO) serta alur penyelesaian penelitian ini.

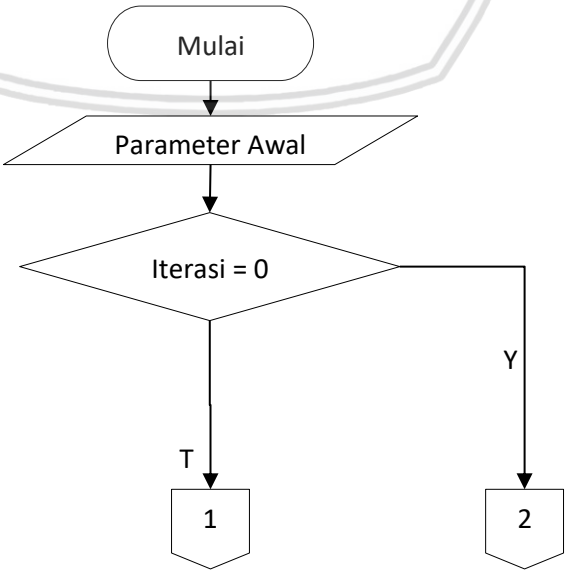
4.1 Formulasi Permasalahan

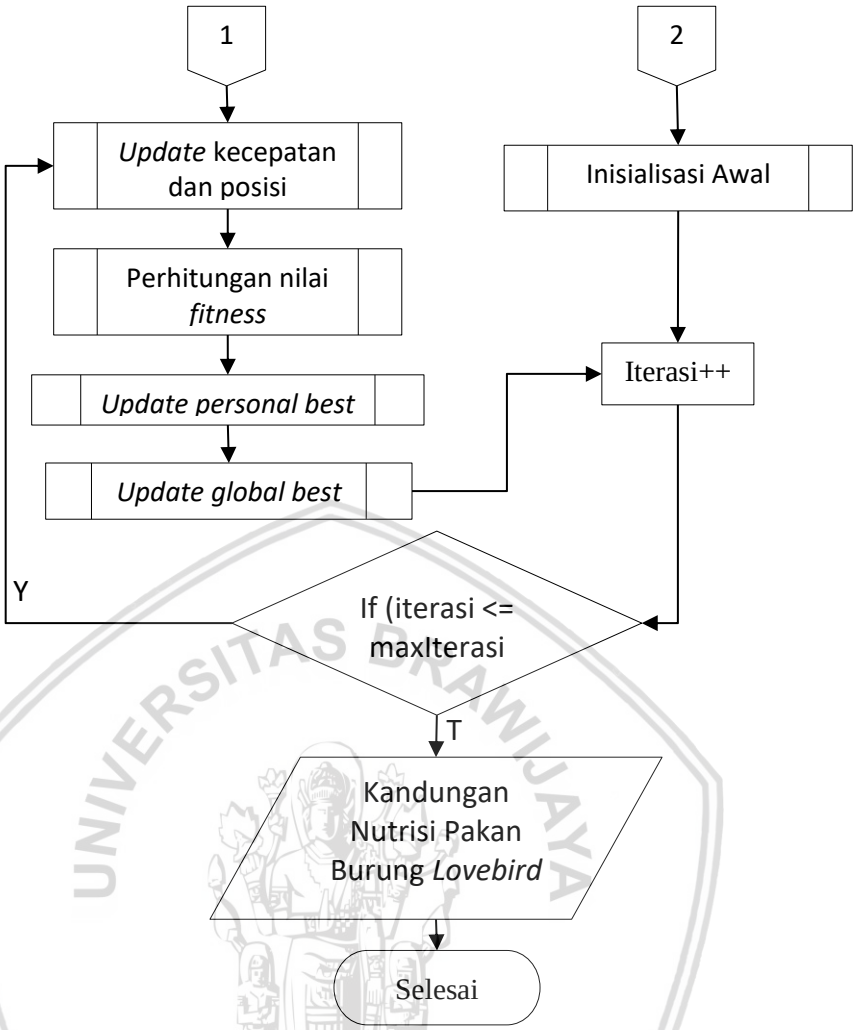
Dalam pemberian pakan burung, dibutuhkan komposisi yang tepat agar kebutuhan burung sehari-hari dapat terpenuhi. Untuk menentukan komposisi yang tepat, algoritme PSO dapat digunakan dalam menentukan komposisi tersebut. Selain menentukan komposisi, PSO juga dapat meminimalkan biaya yang dibutuhkan dalam pemberian pakan.

Pada sistem yang nantinya akan dibuat, data yang digunakan dalam penelitian ini adalah jenis bahan pakan, harga bahan pakan, kandungan nutrisi bahan pakan dan kebutuhan nutrisi burung per harinya. Selain itu, masukkan dari *user* juga digunakan dalam sistem ini. Nantinya dalam sistem ini, *user* dapat memasukkan bahan pakan apa saja yang akan digunakan. Setelah *user* memasukkan *input*-annya, sistem akan menghitung apakah komposisi yang dimasukkan oleh *user* tepat atau tidak dan memberitahukan biaya yang dibutuhkan *user.*

4.2 Alur Algoritme *Particle Swarm Optimization* (PSO)

Pada sub bab ini menjelaskan pengunaan algoritme PSO dalam penyelesaian masalah secara sekuensial. Hasil yang optimal berlandaskan *fitness* partikel sebagai representasi solusi dan permasalahan komposisi pakan burung *lovebird* diharapkan dapat diselesaikan dengan menggunakan algoritme PSO. Pada Gambar 4.1 merupakan diagram alir dari proses pengoptimalan komposisi pakan burung *lovebird* menggunakan algoritme PSO:





Gambar 4.1 Diagram Alir Proses Pengoptimalan Komposisi Pakan Burung *Lovebird* Menggunakan Algoritme PSO

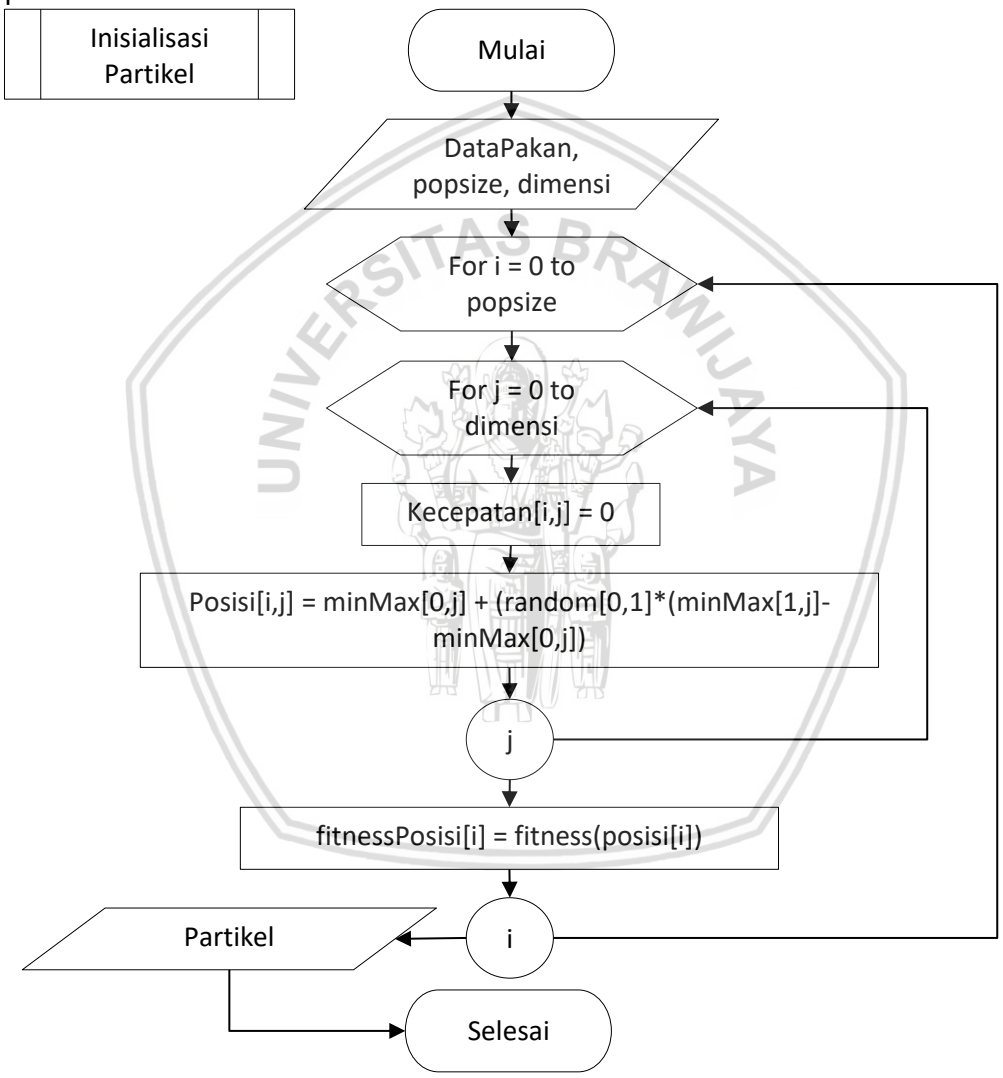
Berikut penjelasan dari diagram alir pada Gambar 4.1 :

1. Data bahan pakan yang tersedia dan inisialisasi sebagian algoritme PSO merupakan parameter awal dalam proses ini. Parameter ini mencakup batas atas dan batas atas posisi partikel (x_{max} dan x_{min}), w, c_1, c_2, nutrisi pakan, harga pakan, kebutuhan nutrisi burung. Setelah itu, masukan pengguna mengenai parameter algoritme PSO, yaitu banyaknya bahan pakan.
2. Kondisi jika iterasi = 0, maka melakukan proses inisialisasi awal partikel.
3. Jika bukan, maka dilakukan pembaruan posisi dan kecepatan partikel yang akan menghasilkan partikel yang lebih baik dari iterasi sebelumnya.
4. Melakukan perhitungan *fitness* agar partikel terbaik dari nilai *fitness* terbesar dapat ditemukan.
5. Melakukan pembaruan *pBest.*
6. Melakukan pembaruan *gBest.*

7. Kondisi jika iterasi masih kurang dari sama dengan maxIterasi, maka ulangi langkah 3 sampai langkah 6.
8. Jika tidak, maka solusi optimum untuk komposisi pakan burung *lovebird* telah ditemukan dari nilai *gBest*.

4.2.1 Inisialisasi Partikel

Panjang partikel atau dimensi dalam perhitungan PSO ditentukan pada tahap ini. Inisialisasi partikel ini merupakan tahap awal dalam perhitungan algoritme PSO atau bisa juga disebut iterasi ke-0. Gambar 4.2 ialah diagram alir inisialisasi partikel.

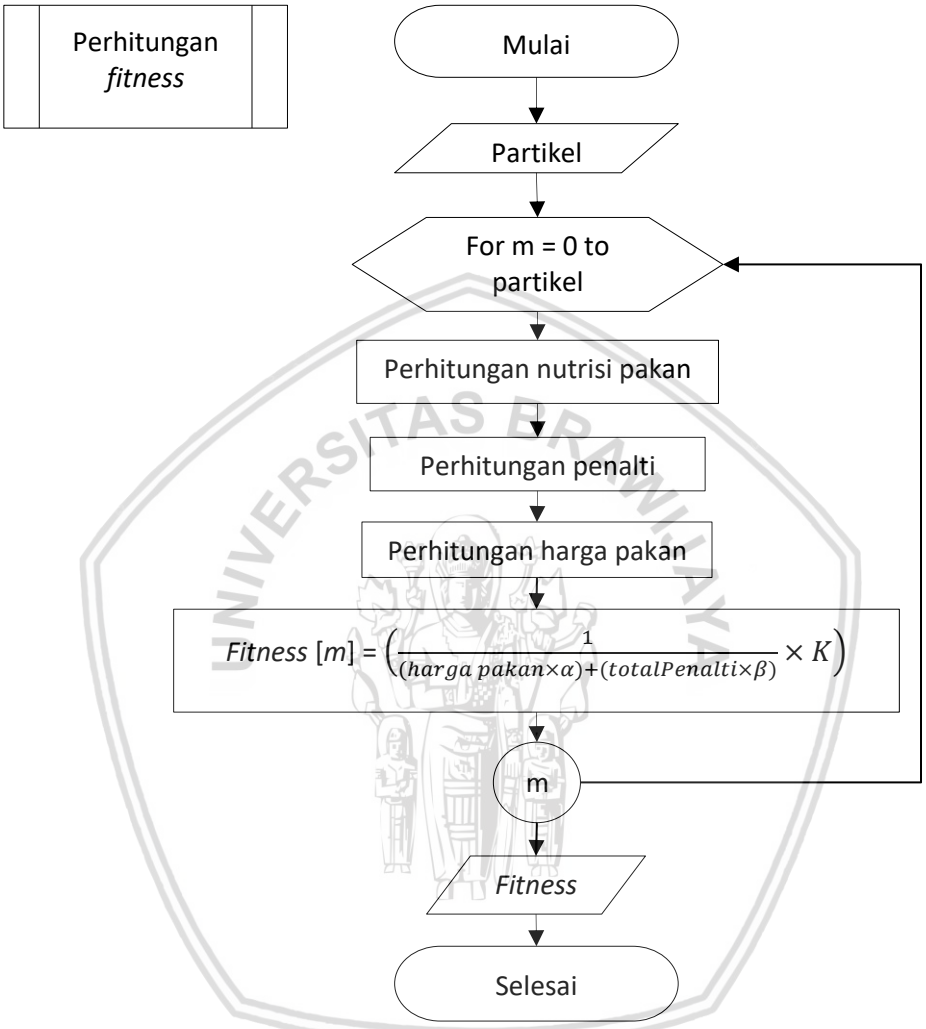


Gambar 4.2 Diagram Alir Inisialisasi Partikel

Pada proses inisialisasi partikel terdiri dari 2 proses inisialisasi yaitu inisialisasi kecepatan awal dan posisi awal. Di dalam proses ini terdapat 2 proses perulangan dimana pengulangan ini menggunakan *popsize* dan dimensi. Untuk kecepatan awal diisi dengan 0 dan posisi dilakukan proses perhitungan seperti pada Gambar 4.2. Lalu pemanggilan *method fitness* dengan parameter posisi adalah untuk mendapatkan nilai *pBest* tiap partikel.

4.2.2 Perhitungan Nilai *Fitness*

Solusi mana yang paling optimal diketahui dari nilai *fitness* yang dihasilkan tiap partikel. Pada penelitian ini, suatu partikel yang menghasilkan nilai *fitness* tertinggi diharapkan mampu memenuhi tujuan dari penelitian ini. Gambar 4.3 merupakan diagram alir perhitungan *fitness*.

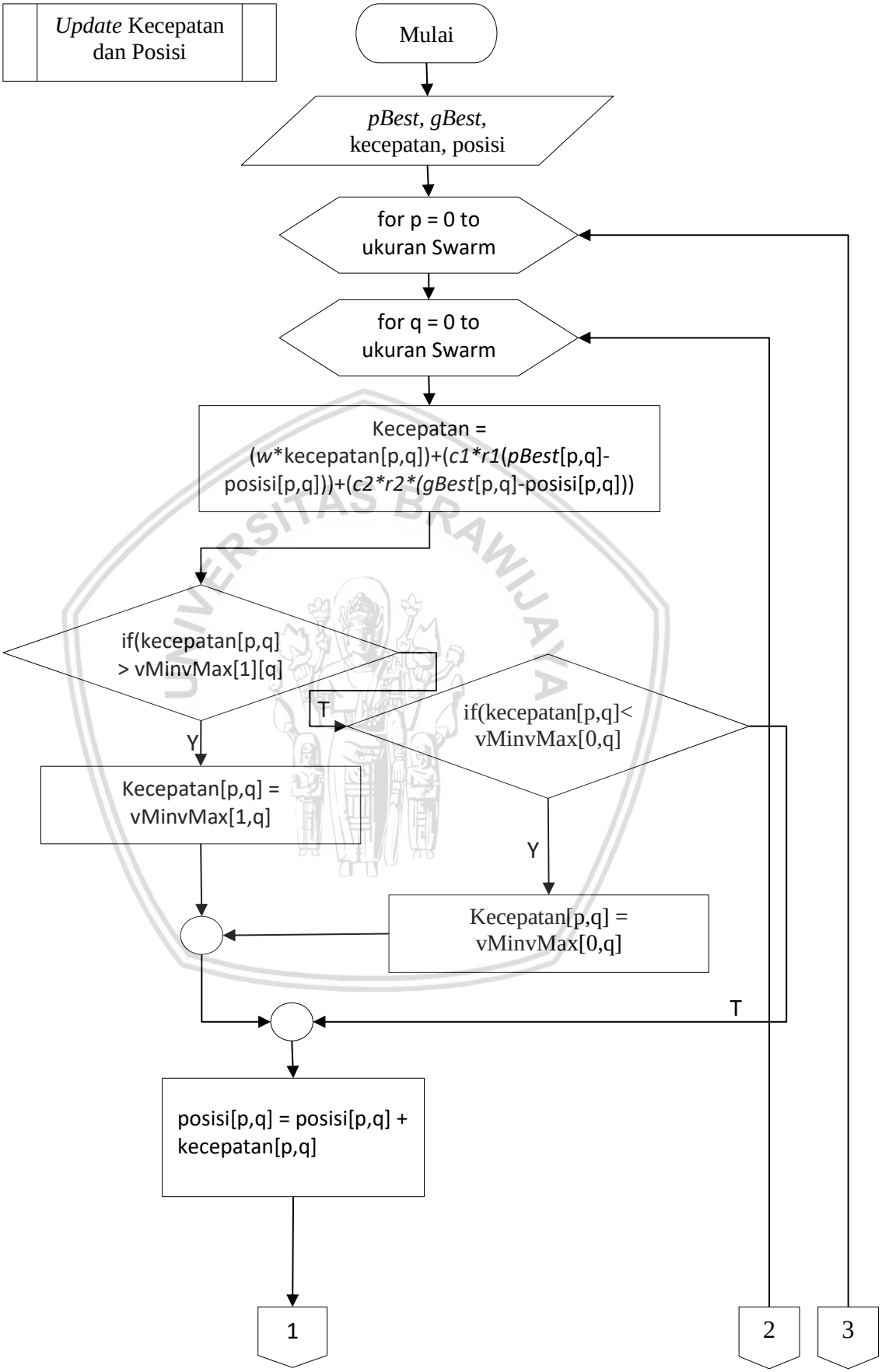


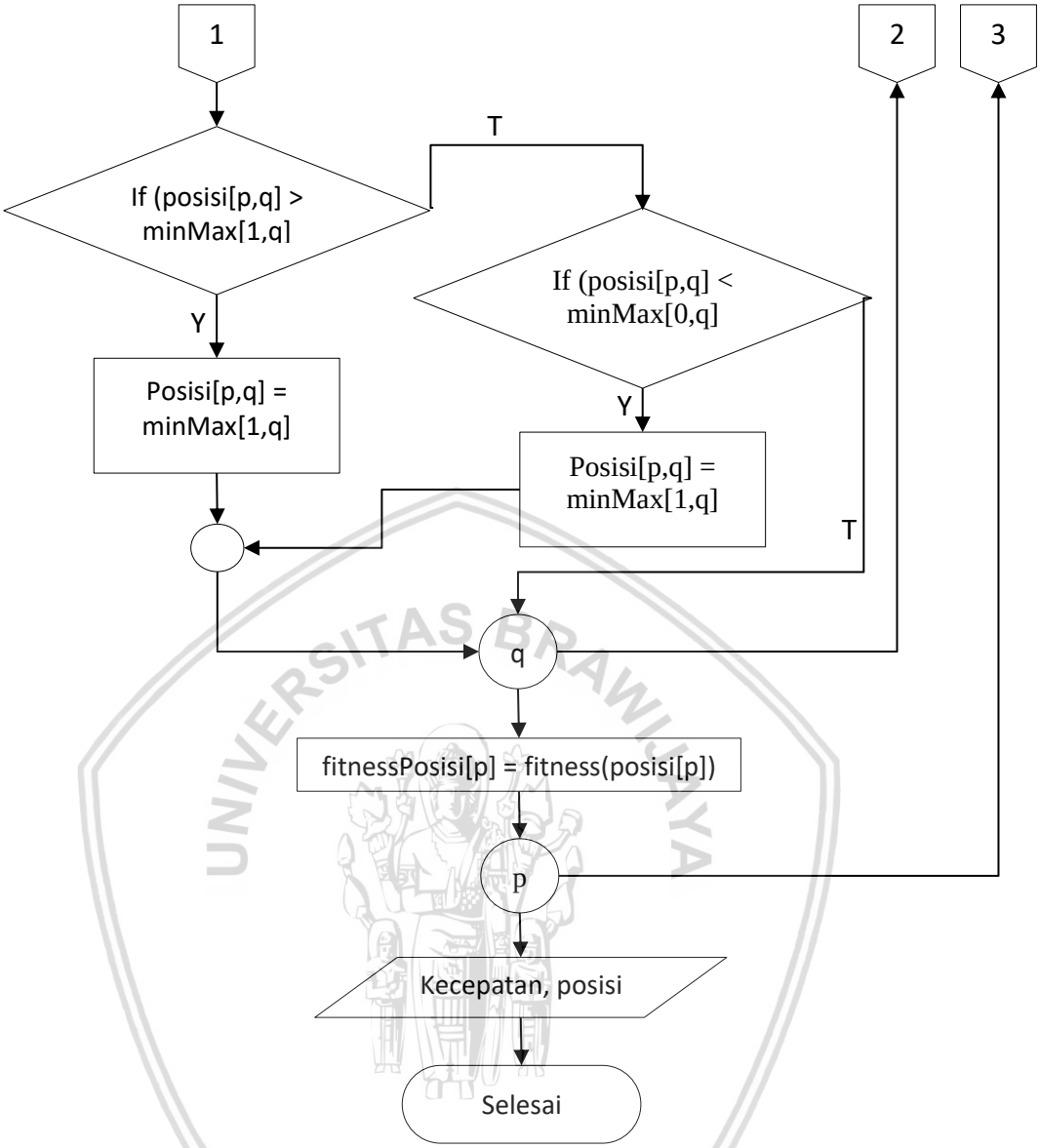
Gambar 4.3 Diagram Alir Perhitungan *Fitness*

Pada proses menghitung *fitness* terdiri dari beberapa perhitungan seperti perhitungan nutrisi pakan, perhitungan penalti, perhitungan harga pakan dan perhitungan *fitness* itu sendiri. Pada perhitungan *fitness* menggunakan hasil dari perhitungan pakan dan total penalti yang telah didapatkan.

4.2.3 *Update* Kecepatan dan Posisi

Pada iterasi 0 (t =0), nilai kecepatan diinisialisasi menjadi nol. Tetapi, pada iterasi selanjutnya, nilai posisi dan kecepatan dihitung mengacu pada Persamaan 2.10 sampai Persamaan 2.14. Gambar 4.4 ialah diagram alir proses *pembaruan* kecepatan dan posisi.


Update Kecepatan dan Posisi
Mulai
pBest, gBest, kecepatan, posisi
for p = 0 to ukuran Swarm
for q = 0 to ukuran Swarm
Kecepatan = (w*kecepatan[p,q])+(c1*r1(pBest[p,q]-posisi[p,q]))+(c2*r2*(gBest[p,q]-posisi[p,q]))
if(kecepatan[p,q] > vMinvMax[1][q]
Y
Kecepatan[p,q] = vMinvMax[1,q]
T
if(kecepatan[p,q]< vMinvMax[0,q]
Y
Kecepatan[p,q] = vMinvMax[0,q]
T
posisi[p,q] = posisi[p,q] + kecepatan[p,q]
1
2
3

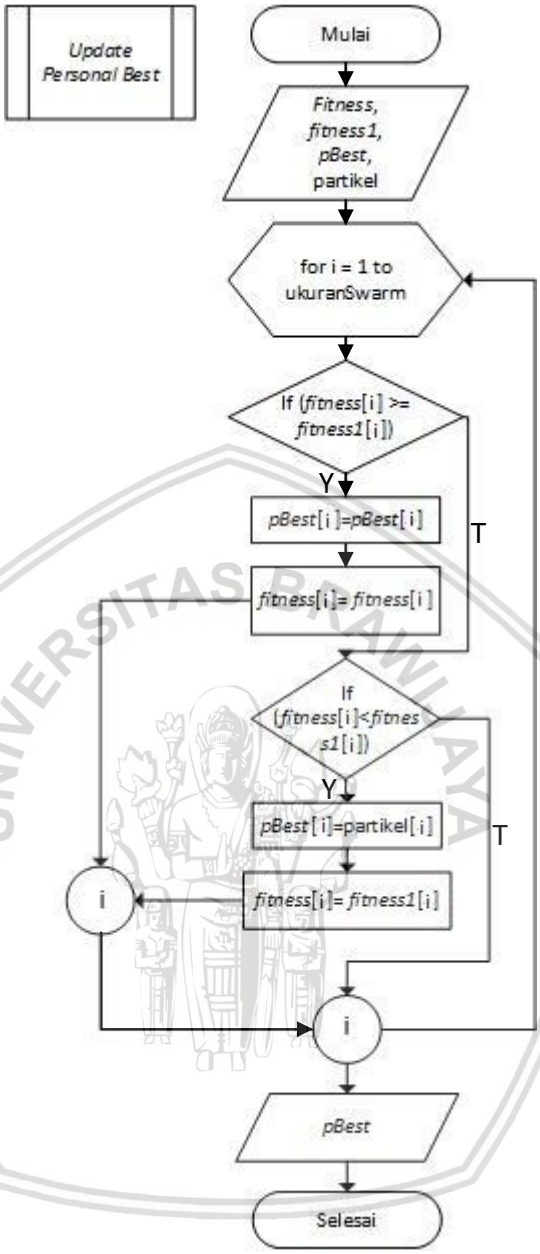


Gambar 4.4 Diagram Alir *Update* Kecepatan dan Posisi

Proses *update* kecepatan dan posisi ini dilakukan untuk memperbarui nilai kecepatan dan posisi. Proses awal dilakukan perhitungan kecepatan menggunakan Persamaan 2.10. Setelah itu dilakukan pengecekan apakah hasil perhitungan melebihi atau kurang dari batas yang telah dilakukan seperti pada Persamaan 2.11 dan Persamaan 2.12. Setelah nilai kecepatan yang baru didapatkan, dilakukan perhitungan posisi menggunakan Persamaan 2.13 dan juga dilakukan pengecekan menggunakan Persamaan 2.14.

4.2.4 *Update Personal Best*

Pada inisialisasi awal, nilai *pBest* sama dengan nilai inisialisasi posisi. Namun pada iterasi selanjutnya, perbandingan nilai *fitness pBest* iterasi sebelumnya dan iterasi sekarang dapat menghasilkan nilai *pBest* yang baru. Gambar 4.5 adalah diagram alir untuk proses *update pBest*.



Gambar 4.5 Diagram Alir *Update Personal Best*

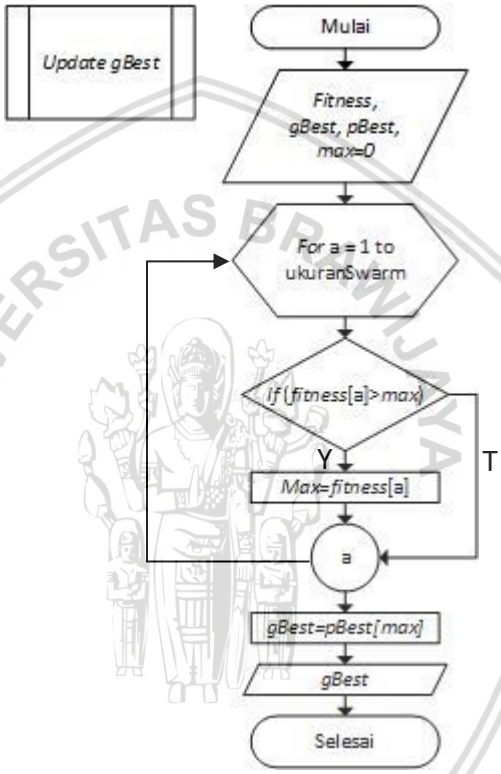
Pada proses *update pBest* ini pertama-tama sistem menerima beberapa masukkan seperti nilai *fitness* sebelumnya, *fitness* sekarang, *pBest* dan posisi partikel. Setelah itu, dilakukan *update pBest* dengan membandingkan *fitness* posisi sekarang dengan *fitness pBest* sebelumnya. Akhirnya nilai *pBest* yang baru didapatkan.

4.2.5 *Update Global Best*

Nilai *pBest* yang memiliki *fitness* terbesar adalah nilai *gBest*. Perbandingan nilai *fitness* pada *array pBest* dengan nilai *fitness* yang berada pada variabel *max* dapat

menghasilkan nilai *gBest*. Nilai *max* akan tergantikan apabila nilai *fitness* pada *array pBest* lebih besar, hal it uterus dilakukan sampai nilai *fitness* terbesar didapatkan yang akan menjadi *gBest*.

Nilai *gBest* didapatkan dari nilai *pBest* yang memiliki nilai *fitness* terbesar. Perbandingan antara nilai *fitness array personal best* dengan nilai *fitness* yang disimpan dalam *variable max* akan menghasilkan nilai *gBest.* Kemudian, apabila nilai *fitness* pada *array pBest* lebih besar, maka akan menggantikan nilai *max,* proses ini akan diteruskan sampai diperoleh nilai *fitness* terbesar. terus seperti itu sampai diperoleh nilai *fitness* terbesar, di mana nilai *fitness* terbesar disebut menjadi *gBest*. Gambar 4.6 ialah diagram alir *update gBest*.



Gambar 4.6 Diagram Alir *Update gBest*

Pada proses *update gBest* ini pertama-tama sistem menerima beberapa masukkan seperti nilai *fitness pBest*, *fitness gbest, pBest* dan inisialisasi variabel *max*. Setelah itu, dilakukan *update gBest* dengan mencari nilai *fitness pBest* terbesar dan dimasukkan ke dalam variabel *max*. Akhirnya nilai *gBest* yang baru didapatkan.

4.3 Perhitungan Manual

Sub bab ini akan menjelaskan tentang perhitungan manual untuk menyelesaikan contoh kasus. Di mana, proses penyelesaian mengacu pada alur algoritme PSO dalam sub bab 4.2. Diharapkan dengan keberadaan perhitungan manual ini, dapat mempermudah penafsiran mengenai penyelesaian masalah optimasi komposisi pakan burung *lovebird* sebelum diimplementasikan menjadi kode program.

Perhitungan manual pada penelitian ini memiliki beberapa parameter awal yang sudah ditentukan, antara lain:

- Jumlah iterasi = 2
- Ukuran *swarm* = 3
- Bobot inersia(ω) = 0,5
- Koefisien akselerasi 1 (c_1) = 1
- Koefisien akselerasi 2 (c_2) = 1
- Konstanta *k* = 10.000
- Konstanta α dan β = 20
- Nilai r_1 = 3 (dibuat konstan hanya untuk perhitungan manual)
- Nilai r_2 = 5 (dibuat konstan hanya untuk perhitungan manual)
- Jumlah dimensi = 6
- Maksimal dan minimal posisi (*interval*) = 20 & 0
- Batas atas dan batas bawah dapat dilihat pada Tabel 4.1.

Bahan	Batas Bawah	Batas Atas
Millet Putih	0	20
Millet Merah	0	15
Canary Seed	0	20
Jagung	0	20
Kuaci	0	5
Kangkung	0	20

4.3.1 Inisialisasi Partikel

Pada inisialisasi partikel iterasi ke-0 ini, yang akan dilakukan adalah memasukkan kecepatan dan posisi awal partikel sesuai dengan *popsize* dan dimensi yang sudah ditentukan. Nilai kecepatan awal diisi dengan 0 karena posisi partikel belum berpindah maka belum ada kecepatan. Nilai posisi awal didapatkan dari proses perhitungan menggunakan Persamaan 2.3. Untuk kecepatan dan posisi partikel, dapat dilihat pada Tabel 4.2 dan 4.3.

Tabel 4.1 Inisialisasi Kecepatan Awal Partikel

Inisialisasi Kecepatan Awal Partikel						
	Millet Putih	Millet Merah	*Canary Seed*	Jagung	Kuaci	Kangkung
v1(0)	0	0	0	0	0	0
v2(0)	0	0	0	0	0	0
v3(0)	0	0	0	0	0	0

Tabel 4.2 Inisialisasi Posisi Awal Partikel

Inisialisasi Posisi Awal Partikel						
	Millet Putih	Millet Merah	*Canary Seed*	Jagung	Kuaci	Kangkung
x1(0)	2	7	6	4	9	4
x2(0)	3	5	3	6	10	5
x3(0)	9	10	2	8	10	8

Berikut keterangan untuk Tabel 4.1 sampai Tabel 4.12:
V1(i) = Kecepatan partikel ke-1 pada iterasi ke-*i*
V2(i) = Kecepatan partikel ke-2 pada iterasi ke-*i*
V3(i) = Kecepatan partikel ke-3 pada iterasi ke-*i*
X1(i) = Posisi partikel ke-1 pada iterasi ke- *i*
X2(i) = Posisi partikel ke-2 pada iterasi ke- *i*
X3(i) = Posisi partikel ke-3 pada iterasi ke- *i*

4.3.2 Perhitungan Nilai *Fitness* Iterasi 0

Perhitungan nilai *fitness* mengacu kebutuhan nutrisi burung *lovebird*, yaitu ME, lemak, protein, karbohidrat, fosfor dan serat kasar. Nutrisi-nutrisi ini dihitung sampai didapatkan nilai penalti. Selain itu, nilai *fitness* juga dihitung berlandaskan harga bahan pakan. Kedua atribut tersebut bertolak belakang dengan optimasi, sehingga penulis berharap mampu menghasilkan nilai *fitness* terbesar.

Proses mendapatkan nilai *fitness* ini dibagi menjadi tiga langkah, seperti :

1. Menghitung nutrisi pakan
   Pertama-tama ditentukan terlebih dahulu bobot pakan yang akan diberikan setelah itu dilakukan normalisasi dengan menggunakan Persamaan 2.4. Setelah nutrisi pakan seluruh partikel sudah didapatkan, nutrisi pakan tersebut dijumlahkan. Berikut adalah sampel menghitung normalisasi bobot pakan untuk partikel pertama.

   Normalisasi bobot $pakan_1(\%)$ $= \frac{2}{32} \times 100\% = 6{,}25\%$

   Normalisasi bobot $pakan_2(\%)$ $= \frac{7}{32} \times 100\% = 21{,}875\%$

   Normalisasi bobot $pakan_3(\%)$ $= \frac{6}{32} \times 100\% = 18{,}75\%$

   Normalisasi bobot $pakan_4(\%)$ $= \frac{4}{32} \times 100\% = 12{,}5\%$

   Normalisasi bobot $pakan_5(\%)$ $= \frac{9}{32} \times 100\% = 28{,}125\%$

   Normalisasi bobot $pakan_6(\%)$ $= \frac{4}{32} \times 100\% = 12{,}5\%$

   Setelah bobot pakan diketahui, selanjutnya menghitung nutrisi pakan menggunakan Persamaan 2.1 dan Persamaan 2.2 dimana informasi kandungan nutrisi terdapat pada Lampiran A. Berikut contoh perhitungan nutrisi untuk bahan pakan pada partikel pertama dimensi pertama.

   $Protein_1 = \frac{6{,}25}{100} \times 12\% = 0{,}75\%$

Karbohidrat$_1$ $= \frac{21{,}875}{100} \times 60{,}5\% = 3{,}78\%$

Lemak$_1$ $= \frac{18{,}75}{100} \times 4{,}5\% = 0{,}28\%$

Fosfor$_1$ $= \frac{12{,}5}{100} \times 0{,}6\% = 0{,}04\%$

Serat Kasar$_1$ $= \frac{28{,}125}{100} \times 2{,}4\% = 0{,}15\%$

ME $= \frac{12{,}5}{100} \times 3781\ Kkal/kg = 236{,}31\ Kkal/kg$

Perhitungan nutrisi ini dilakukan sampai dimensi terakhir. Hasil perhitungan nutrisi partikel 1 dapat dilihat pada Tabel 4.3 yang nantinya akan digunakan untuk menghitung penalti.

Tabel 4.3 Hasil Perhitungan Nutrisi

dimensi ke-	Protein	Karbohidrat	Lemak	Fosfor	Serat Kasar	ME
1	0,75	3,78	0,28	0,04	0,15	236,31
2	2,73	12,69	0,88	0,13	0,46	798,44
3	2,81	11,31	1,03	0,11	1,43	748,13
4	1,13	0,00	0,51	0,04	0,28	420,00
5	5,63	12,66	7,31	0,21	0,56	1448,44
6	0,38	6,75	0,04	6,25	0,24	362,50
Total	13,42	47,18	10,05	6,78	3,11	4013,81

2. Menghitung Penalti

Perbandingan antara Tabel 4.3 dengan kebutuhan nutrisi burung menggunakan Persamaan 2.5 dapat menghasilkan nilai penalti. Di mana nilai penalti dapat dilihat pada Tabel 4.4. Berikut adalah sampel menghitung penalti untuk partikel pertama.

Penalti protein = 13,2 < 20 maka 20 – 13,2 = 6,578125%

Penalti karbohidrat = 47,18 > 30 maka 0%

Penalti lemak = 10,05 < 27 maka 27-10,05 = 16,95%

Penalti fosfor = 6,78 < 22 maka 22-6,78 = 15,2225%

Penalti serat kasar = 3,11 < 8 maka 8-3,11 = 4,890625%

Penalti ME = 4013,81 > 3000 maka 0 *Kkal/kg*

Karena satuan ME masih *Kkal/kg*, maka harus dikonversi ke dalam persen. Berikut cara mengonversi *Kkal/kg* menjadi persen.

Penalti ME = $\frac{0\ Kkal/kg}{3000\ kkal/kg} \times 100\% = 0\%$

Setelah dikonversi, berikut adalah perhitungan total penalti nutrisi.

Penalti = 6,578125 + 0 + 16,95 + 15,2225 + 4,890625 + 0 = 43,64125%

Tabel 4.4 Penalti Nutrisi

Perhitungan Nutrisi							
	P. Pro	P.Kar	*P.Lem*	P.Fos	P.Ser	P.ME	Total Pinalti
x1(0)	6,578125	0	16,95	15,2225	4,890625	0	43,64125
x2(0)	7,109375	0	16,496875	13,69125	5,4	0	42,6975
x3(0)	8,10638298	0	18,77234043	13,01106383	5,646808511	0	45,53659574

3. Menghitung Harga Pakan (*cost*)

Menghitung biaya pakan menggunakan Persamaan 2.6 merupakan langkah terakhir sebelum menghitung nilai *fitness*. Berikut adalah sampel menghitung harga pakan tiap bahan pakan untuk partikel pertama.

$\text{Harga}_1 = \left(\frac{2}{100} \times 5\right) \times 8000 = 80.000$

$\text{Harga}_2 = \left(\frac{7}{100} \times 5\right) \times 8000 = 280.000$

$\text{Harga}_3 = \left(\frac{6}{100} \times 5\right) \times 8000 = 240.000$

$\text{Harga}_4 = \left(\frac{4}{100} \times 5\right) \times 1000 = 20.000$

$\text{Harga}_5 = \left(\frac{9}{100} \times 5\right) \times 18000 = 810.000$

$\text{Harga}_6 = \left(\frac{4}{100} \times 5\right) \times 3000 = 60.000$

Total Harga $= 80.000 + 280.000 + 240.000 + 20.000 + 810.000 + 60.000 = 1.490.000$

Setelah nilai penalti dan total harga didapatkan, kemudian Persamaan 2.7 digunakan untuk menghitung nilai *fitness*. Untuk mendapatkan nilai *fitness* pada partikel lain, tahap-tahap ini juga bisa digunakan. Tabel 4.5 merupakan hasil dari perhitungan nilai *fitness*.

Tabel 4.5 Hasil Nilai *Fitness* Partikel

Partikel ke-	Nilai Partikel						*Fitness*
x1(0)	2	7	6	4	9	4	11,45705374
x2(0)	3	5	3	6	10	5	11,71029095
x3(0)	9	10	2	8	10	8	10,98018224

4.3.3 Perhitungan Inisialisasi *Personal Best* dan *Global Best*

Inisialisasi awal *pBest* dan *gBest* sama seperti posisi awal PSO. Tabel 4.6 adalah hasil deklarasi nilai awal *pBest* dan *gBest*. Untuk tulisan yang berwarna merah merupakan *global best*.

Tabel 4.6 Inisialisasi *Personal Best* dan *Global Best*

Update Pbest dan Gbest							
	j1	j2	*j3*	j4	j5	j6	*fitness*
Pbest1(0)	2	7	6	4	9	4	11,45705374
Pbest2(0)	3	5	3	6	10	5	11,71029095
Pbest3(0)	9	10	2	8	10	8	10,98018224

4.3.4 Perhitungan Batas Kecepatan

Batas kecepatan ini diperlukan agar kecepatan mempunyai Batasan, yaitu batas atas dan batas bawah. Untuk menentukan batas atas dan batas bawah kecepatan dapat menggunakan Persamaan 2.8 untuk *vmax* dan Persamaan 2.9 untuk *vmin* dimana konstanta *k* merupakan nilai *random* dimana nilai yang didapatkan adalah 0,713276. Pada Tabel 4.7 merupakan batas kecepatan untuk iterasi ke-0.

Tabel 4.7 Penentuan Batas Kecepatan

Penentuan Batas Kecepatan			
Vmaxj1	7,132762093	Vminj1	-7,13276209
Vmaxj2	5,34957157	Vminj2	-5,34957157
Vmaxj3	7,132762093	Vminj3	-7,13276209
Vmaxj4	7,132762093	Vminj4	-7,13276209
Vmaxj5	1,783190523	Vminj5	-1,78319052
Vmaxj6	7,132762093	Vminj6	-7,13276209

4.3.5 Perhitungan *Update* Kecepatan dan Posisi

Untuk proses pembaruan posisi dan kecepatan pada iterasi 1 dan seterusnya, dapat menggunakan Persamaan 2.10 dan 2.14.

1. *Update* kecepatan
   Untuk mendapatkan nilai kecepatan yang baru, Persamaan 2.10 dapat digunakan. Tabel 4.8 adalah pembaruan dari kecepatan pada iterasi ke-1.

   Tabel 4.8 Hasil Perhitungan *Update* Kecepatan Iterasi ke-1

Update Kecepatan						
	Millet Putih	Millet Merah	*Canary Seed*	Jagung	Kuaci	Kangkung
v1(1)	48,5514547	23,55145474	28,55145474	38,55145474	13,55145474	38,55145474
v2(1)	43,5514547	33,55145474	43,55145474	28,55145474	8,551454739	33,55145474
v3(1)	13,5514547	8,551454739	48,55145474	18,55145474	8,551454739	18,55145474

   Setelah mendapatkan nilai kecepatan yang baru, lakukan pengecekan nilai tersebut menggunakan Persamaan 2.11 dan Persamaan 2.12. Tabel 4.9 merupakan hasil dari perbaikan kecepatan.

   Tabel 4.9 Hasil Perbaikan Kecepatan

Update Kecepatan						
	Millet Putih	Millet Merah	*Canary Seed*	Jagung	Kuaci	Kangkung
v1(1)	7,13276209	5,34957157	7,132762093	7,132762093	1,783190523	7,132762093
v2(1)	7,13276209	5,34957157	7,132762093	7,132762093	1,783190523	7,132762093
v3(1)	7,13276209	5,34957157	7,132762093	7,132762093	1,783190523	7,132762093

2. *Update* posisi
   Setelah kecepatan didapatkan, hal selanjutnya adalah meng-*update* posisi partikel. Pada *update* posisi ini, dapat digunakan juga untuk mendapatkan kandungan nutrisi, penalti kandungan, total harga dan juga *fitness*. Untuk mendapatkan nilai *update* posisi dapat menggunakan Persamaan 2.13, sedangkan untuk kandungan nutrisi, penalti kandungan, total harga dan

fitness Persamaan 2.4 sampai Persamaan 2.7 dapat digunakan. Tabel 4.10 merupakan hasil dari perhitungan.

Tabel 4.10 Hasil Perhitungan *Update* Posisi

Update Posisi			
	x1(1)	x2(1)	x3(1)
Millet Putih	9,13276209	10,13276209	16,13276209
Millet Merah	12,3495716	10,34957157	15
Canary Seed	13,1327621	10,13276209	9,132762093
Jagung	11,1327621	13,13276209	15,13276209
Kuaci	5	5	5
Kangkung	11,1327621	12,13276209	15,13276209
Pro	12,1636682	11,92490041	11,85506761
Kar	46,656039	44,42702165	45,53371816
Lem	5,52208528	5,423894512	5,023255765
Fos	9,44032249	10,38775572	10,44343064
Ser	3,28545587	3,038833146	2,802449457
ME	3675,58892	3639,560855	3610,022511
P. Pro	7,83633178	8,144932387	8,075099586
P.Kar	0	0	0
P.Lem	21,4779147	21,57610549	21,97674424
P.Fos	12,5596775	11,61224428	11,55656936
P.Ser	4,71454413	4,961166854	5,197550543
P.ME	0	0	0
Total Harga	20572,5907	19222,59072	23632,76209
Fitness	10,732272	10,81675006	10,66648753
Tot Pinalti	46,5884682	46,22461621	46,87579652

4.3.6 Perhitungan *Update Personal Best* dan *Global Best*

Pada iterasi ke-1 dan seterusnya, perbandingan nilai *fitness* mana yang terbesar antara nilai *fitness* pBest iterasi sebelumnya dengan nilai *fitness* posisi Sekarang dapat menghasilkan nilai *pBest* yang baru. Nilai *fitness* iterasi sebelumnya akan tetap digunakan apabila nilai *fitness pBest* iterasi sekarang dengan iterasi sebelumnya sama. Sedangkan nilai *gBest* didapatkan dari nilai *fitness pBest* terbesar. Untuk lebih jelasnya, dapat dilihat pada Tabel 4.11.

Tabel 4.11 Perbandingan *Fitness* Iterasi 0 dan Iterasi 1

Partikel ke-	*Fitness pBest* iterasi 0	*Fitness pBest* iterasi 1
1	11,45705374	10,73227202
2	11,71029095	10,81675006
3	10,98018224	10,66648753

Dengan melihat Tabel 4.12, nilai *pBest* terbaru bisa didapatkan.

Tabel 4.12 Hasil *Update Personal Best*

Update Pbest dan Gbest							
	j1	j2	*j3*	j4	j5	j6	*fitness*
Pbest1(0)	2	7	6	4	9	4	11,45705374
Pbest2(0)	3	5	3	6	10	5	11,71029095
Pbest3(0)	9	10	2	8	10	8	10,98018224

Sama seperti sebelumnya, tulisan yang berwarna merah merupakan *global best*.

Solusi permasalahan penentuan komposisi pakan burung *lovebird* didapatkan dari bobot pakan tertinggi dari nilai *gBest* tersebut. Tabel 4.3 merupakan hasil *decode* komposisi pakan.

Tabel 4.13 Hasil *Decode* Komposisi Pakan

No.	Nama Bahan Pakan	Kandungan Nutrisi						Total Harga
		ME	Protein	Lemak	Serat Kasar	Karbohidrat	Fosfor	
1	Millet Putih	3781	12	4,5	2,4	60,5	0,6	576356,4
2	Millet Merah	3650	12,5	4	2,1	58	0,6	
3	Kangkung	2900	3	0,3	1,9	54	50	
4	Jagung	3360	9	4,1	2,2	0,02	0,29	
5	Kuaci	5150	20	26	2	45	0,76	

BAB 5 IMPLEMENTASI

Bab ini berisikan tentang implementasi sistem, di mana terdapat implementasi algoritme PSO.

5.1 Implementasi Algoritme *Particle Swarm Optimization* (PSO)

Pada implementasi algoritme ini memuat hasil dari perancangan sistem penelitian ini dalam bentuk kode program. Tahapan yang akan dibahas adalah inisialisasi kecepatan awal, inisialisasi posisi awal, inisialisasi *pBest* dan *gBest*, perhitungan *fitness*, penentuan batas kecepatan, *update* kecepatan, *update* posisi serta *update pBest* dan *gBest*.

5.1.1 Implementasi Inisialisasi Kecepatan Awal

Implementasi inisialisasi kecepatan awal merupakan tahap awal dalam proses PSO yaitu iterasi ke-0, di mana kecepatan awal partikel bernilai nol untuk seluruh partikel. Kode Program 5.1 adalah implementasi kecepatan awal.

```
1  public static void inisialisasiKecepatanAwal(int popsize, int
2     dimensi) {
3          for (int i = 0; i < popsize; i++) {
4               for (int j = 0; j < dimensi; j++) {
5                    kecepatan[i][j] = 0;
6               }
7          }
8     }
```

Kode Program 5.1 Implementasi Inisialisasi Kecepatan Awal

Penjelasan dari Kode Program 5.1 implementasi inisialisasi kecepatan awal sebagai berikut :

Baris 1-2 Deklarasi *method* inisialisasiKecepatanAwal dan parameternya.

Baris 3-4 Deklarasi perulangan untuk menginisialisasi kecepatan awal tiap partikel.

Baris 5 Proses inisialisasi kecepatan awal partikel yang nilainya berupa 0.

5.1.2 Implementasi Inisialisasi Posisi Awal

Implementasi inisialisasi posisi awal merupakan salah satu tahap awal dalam proses PSO. Kode Program implementasi inisialisasi posisi awal.

```
1  public static void inisialisasiPosisiAwal(int popsize, int
2  dimensi) {
3            for (int i = 0; i < popsize; i++) {
4                 for (int j = 0; j < dimensi; j++) {
5                      posisi[i][j]      =      minMax[0][j]      +
6  (rand.nextDouble() * (minMax[1][j]-minMax[0][j]));
7                 }
8                 fitnessPosisi[i] = fitness(posisi[i]);
9            }
10    }
```

Kode Program 5.2 Implementasi Inisialisasi Posisi Awal

Penjelasan Kode Program 5.2 adalah sebagai berikut :

Baris 1-2 Deklarasi *method* inisialisasiPosisiAwal dan parameternya.

Baris 3-4 Deklarasi perulangan untuk menginisialisasi posisi tiap *popsize* dan tiap dimensi.

Baris 5-6 Proses perhitungan untuk inisialisasi posisi awal menggunakan Persamaan 2.1.

Baris 8 Inisialisasi variabel fitnessPosisi yang memanggil *method fitness*.

5.1.3 Implementasi Perhitungan Nilai *Fitness*

Terdapat empat tahapan dalam menginisialisasi *fitness* awal yaitu dengan cara menghitung nutrisi pakan, menghitung penalti, menghitung harga pakan dan yang terakhir barulah menghitung nilai *fitness*-nya. Kode Program 5.3 ialah implementasi perhitungan nilai *fitness.*

```
1  public static double fitness(double pakan[]) {
2            double pengaliA = 20;
3            double pengaliB = 20;
4            double nutrisi[] = new double[nutrisiBahan.length];
5            Arrays.fill(nutrisi, 0);
6            double pinalti[] = new double[nutrisi.length];
7            Arrays.fill(pinalti, 0);
8            double pinalti1 = 0;
9            double hargaPakan = 0;
10           double totalPinalti = 0;
11           double sum = 0;
12           double fitness = 0;
13           double jumlahNutrisi = 0;
14           int iterasi = 0;
15           for (int j = 0; j < pakan.length; j++) {
16               sum += pakan[j];
17           }
18          for (int i = 0; i < nutrisi.length; i++) {
19             for (int j = 0; j < pakan.length; j++) {
20                 nutrisi[i]   +=   ((pakan[j]   /   sum)   *
21 nutrisiBahan[i][j]);
22             }
23             if (nutrisi[i] >= nutrisiBurung[i]) {
24                 pinalti[i] = 0;
25                 if (i == 5) {
26                pinalti1 = (pinalti[i]/nutrisiBurung[5])*100;
27                 }
28             } else {
29                 pinalti[i] = nutrisiBurung[i] - nutrisi[i];
30                 if (i == 5) {
31                pinalti1 = (pinalti[i]/nutrisiBurung[5])*100;
32                 }
33                }
34             totalPinalti += pinalti[i] + pinalti1;
35            }
36            for (int i = 0; i < nutrisi.length; i++) {
37                if (i == 0) {
38                    jumlahNutrisi += nutrisi[i];
39                } else if (i % 6 == 0) {
```

```
40                      iterasi++;
41                      jumlahNutrisi += nutrisi[i];
42                  } else {
43                      jumlahNutrisi += nutrisi[i];
44                  }
45              }
46          for (int i = 0; i < pakan.length; i++) {
47              hargaPakan += pakan[i] / 100 * hargaBahan[i] *
48  kebPakanBurungPerHari;
49              }
50              hrgPakan = hargaPakan;
51          fitness = ((1 / (hargaPakan * pengaliA)) + (1 /
52  (totalPinalti * pengaliB)) * 10000);
53          return fitness;
54  }
```

Kode Program 5.3 Implementasi Perhitungan Nilai *Fitness*

Penjelasan dari Kode Program 5.3 adalah sebagai berikut :

Baris 1 Deklarasi *method fitness* dan parameternya.

Baris 2-14 Deklarasi variabel – variabel yang nantinya akan dipakai di dalam *method* ini.

Baris 15-17 Proses perulangan untuk menjumlahkan nilai – nilai yang terdapat di dalam *array*.

Baris 18-22 Proses perulangan untuk menghitung nutrisi sesuai dengan Persamaan 2.2.

Baris 23-33 Deklarasi kondisi apabila nutrisi yang telah dihitung lebih dari atau kurang dari nutrisi burung seharusnya maka terkena penalti sesuai dengan Persamaan 2.3.

Baris 34 Menginisialisasi variabel totalPinalti.

Baris 36-45 Deklarasi kondisi untuk mendapatkan nilai jumlahNutrisi.

Baris 46-49 Proses perhitungan total harga pakan sesuai dengan Persamaan 2.4.

Baris 50 Inisialisasi nilai variabel hrgPakan.

Baris 51-52 Proses perhitungan nilai *fitness* sesuai dengan Persamaan 2.5.

5.1.4 Implementasi Inisialisasi *pBest* dan *gBest* Awal

Inisialisasi nilai *pBest* awal didapatkan dari posisi partikel pada iterasi ke-0. Sedangkan untuk inisialisasi nilai *gBest* awal didapatkan dari nilai *fitness* terbesar dari semua partikel yang ada pada *swarm*. Kode Program 5.4 merupakan implementasi inisialisasi *pBest* dan *gBest* awal.

```
1   public static void inisialisasiPBestGbestAwal() {
2           for (int i = 0; i < pBest.length; i++) {
3               for (int j = 0; j < pBest[0].length; j++) {
4                   pBest[i][j] = posisi[i][j];
5               }
6           }
```

```
7           for (int i = 0; i < popsize; i++) {
8               fitnesspBest[i] = fitnessPosisi[i];
9               hargaPakan[i] = hrgPakan;
10          }
11          double max = 0;
12          for (int i = 0; i < pBest.length; i++) {
13              if (fitnesspBest[i] >= max) {
14                  idGbestTerbesar = i;
15                  max = fitnesspBest[i];
16                  gBest = pBest[i];
17                  gBest1[0] = max;
18              }
19          }
20      }
```

Kode Program 5.4 Implementasi Inisialisasi *pBest* dan *gBest*

Penjelasan Kode Program 5.4 adalah sebagai berikut :

Baris 1 Deklarasi *method* inisialisasiPBestGbestAwal

Baris 2-6 Proses perulangan untuk mendapatkan nilai *pBest*.

Baris 7-9 Proses perulangan untuk mendapatkan nilai *fitness pBest* dan harga terbaik.

Baris 10-17 Proses pencarian *gBest*.

5.1.5 Implementasi Penentuan Batas Kecepatan

Penentuan batas kecepatan ini berguna untuk menentukan batas minimal dan batas maksimal dari kecepatan pada iterasi selanjutnya. Apabila setelah dilakukan *update* kecepatan, nilai kecepatan kurang dari atau lebih dari batas yang telah ditentukan maka akan dilakukan perbaikan kecepatan. Implementasi penentuan batas kecepatan dapat dilihat pada Kode Program 5.5.

```
1   public static void VminVmax() {
2           double k = 0.4;
3           for (int i = 0; i < vMinvMax.length; i++) {
4               for (int j = 0; j < vMinvMax[0].length; j++) {
5                   if (i == 0) {
6                       vMinvMax[i][j]   =   (-(k   *   ((double)
7   (minMax[1][j]-minMax[0][j]) / 2)));
8                   } else {
9                       vMinvMax[i][j]    =    (k    *    ((double)
10  (minMax[1][j]-minMax[0][j]) / 2));
11                  }
12              }
13          }
14      }
```

Kode Program 5.5 Implementasi Penentuan Batas Kecepatan

Penjelasan dari Kode Program 5.5 adalah sebagai berikut :

Baris 1 Deklarasi *method* VminVmax.

Baris 2 Deklarasi variabel yang akan digunakan dalam *method* ini.

Baris 3-4 Proses perulangan untuk membaca batas kecepatan yang telah ditentukan.

Baris 5-10 Proses perhitungan untuk menentukan batas kecepatan sesuai dengan Persamaan 2.6 dan Persamaan 2.7.

5.1.6 Implementasi *Update* Kecepatan

Implementasi *update* kecepatan adalah proses dalam PSO untuk memperbarui kecepatan partikel pada setiap iterasinya. Untuk mendapatkan nilai kecepatan partikel yang baru, perlu dilakukan proses perhitungan Persamaan 2.8. Implementasi *update* kecepatan dapat dilihat pada Kode Program 5.6.

```
1  public static void updateKecepatan(double r1, double r2) {
2          for (int i = 0; i < kecepatan.length; i++) {
3              for (int j = 0; j < kecepatan[0].length; j++) {
4                  kecepatan[i][j] = (w * kecepatan[i][j]) + (c1
5  * r1 * (pBest[i][j]-posisi[i][j])) + (c2 * r2 * (gBest1[0]-
6  posisi[i][j]));
7                  if (kecepatan[i][j] > vMinvMax[1][j]) {
8                      kecepatan[i][j] = vMinvMax[1][j];
9                  } else if (kecepatan[i][j] < vMinvMax[0][j])
10 {
11                     kecepatan[i][j] = vMinvMax[0][j];
12                 }
13             }
14         }
15     }
```

Kode Program 5.6 Implementasi *Update* Kecepatan

Penjelasan dari Kode Program 5.6 adalah sebagai berikut :

Baris 1 Deklarasi *method update*Kecepatan dengan parameternya.

Baris 2-6 Proses perhitungan *update* kecepatan menggunakan Persamaan 2.8.

Baris 7-8 Pengecekan apabila hasil dari proses perhitungan melebihi dari batas kecepatan yang telah di tentukan dan hasilnya dengan menggunakan Persamaan 2.9.

Baris 9-11 Pengecekan apabila hasil dari proses perhitungan kurang dari batas kecepatan yang telah di tentukan dan hasilnya dengan menggunakan Persamaan 2.10.

5.1.7 Implementasi *Update* Posisi

Implementasi *update* posisi merupakan proses dalam PSO untuk memperbarui posisi partikel setiap iterasinya. Untuk mendapatkan nilai posisi partikel yang baru, perlu dilakukan proses perhitungan Persamaan 2.11. Implementasi *update* kecepatan dapat dilihat pada Kode Program 5.7.

```
1   public static void updatePosisi() {
2           for (int i = 0; i < posisi.length; i++) {
3               for (int j = 0; j < posisi[0].length; j++) {
4                   posisi[i][j]      =      posisi[i][j]      +
5   kecepatan[i][j];
6                   if (posisi[i][j] > minMax[1][j]) {
7                       posisi[i][j] = minMax[1][j];
8                   } else if (posisi[i][j] < minMax[0][j]) {
9                       posisi[i][j] = minMax[0][j];
10                  }
11              }
12              fitnessPosisi[i] = fitness(posisi[i]);
13          }
14      }
```

Kode Program 5.7 Implementasi *Update* Posisi

Penjelasan dari Kode Program 5.7 adalah sebagai berikut :

Baris 1 Deklarasi *method update*Posisi.

Baris 2-5 Proses perhitungan nilai posisi yang baru menggunakan Persamaan 2.11.

Baris 6-7 Pengecekan apabila nilai posisi melebihi dari batas atas yang telah ditentukan dan hasilnya.

Baris 8-10 Pengecekan apabila nilai posisi kurang dari batas bawah yang telah ditentukan dan hasilnya.

Baris 13 Inisialisasi variabel *fitness*Posisi dengan memanggil *method fitness.*

5.1.8 Implementasi *Update pBest* dan *gBest*

Implementasi *update pBest* dan *gBest* merupakan proses untuk memperbarui nilai *pBest* dan *gBest* pada setiap iterasi. Untuk memperbarui nilai *pBest* dan *gBest*, maka dilakukan perbandingan antara nilai *fitness* iterasi sekarang dengan nilai *fitness pBest* iterasi sebelumnya. Apabila nilai sudah di bandingkan, maka nilai *fitness* terbesarlah yang dijadikan nilai *pBest* yang baru. Tetapi, apabila nilai *fitness* pada iterasi sekarang dan iterasi sebelumnya sama, maka nilai *fitness* pada iterasi sebelumnya yang akan digunakan sebagai nilai *fitness pBest* yang baru. Kode Program 5.8 ialah implementasi *update pBest* dan *gBest.*

```
1   public static void updatePBestGbest() {
2           for (int i = 0; i < popsize; i++) {
3               if (fitnessPosisi[i] > fitnesspBest[i]) {
4                   for (int j = 0; j < pBest[0].length; j++) {
5                       pBest[i][j] = posisi[i][j];
6                   }
7               }
8               fitnesspBest[i] = fitness(pBest[i]);
9               hargaPakan[i] = hrgPakan;
10          }
11          double max = 0;
12          for (int i = 0; i < popsize; i++) {
13              if (fitnesspBest[i] >= max) {
14                  idGbestTerbesar = i;
15                  max = fitnesspBest[i];
16                  gBest = pBest[i];
17                  gBest1[0] = max;
18              }
19          }
20      }
```

Kode Program 5.8 Implementasi *pBest* dan *gBest*

Penjelasan dari Kode Program 5.7 adalah sebagai berikut :

Baris 1 Deklarasi *method updatePBestGbest*.

Baris 2-7 Proses perbandingan antara nilai *fitness* iterasi sekarang dengan nilai *fitness pBest* iterasi sebelumnya. Apabila nilai sudah di bandingkan, maka nilai *fitness* terbesarlah yang dijadikan nilai *pBest* yang baru. Tetapi, apabila nilai *fitness* pada iterasi sekarang dan iterasi sebelumnya sama, maka nilai *fitness* pada iterasi sebelumnya yang akan digunakan sebagai nilai *fitness pBest* yang baru.

Baris 8-9 Proses perulangan untuk mendapatkan nilai *fitness pBest* dan harga terbaik.

Baris 11-17 Proses pencarian *gBest*.

BAB 6 PENGUJIAN DAN ANALISIS

Bab ini membahas tentang tahap pengujian serta analisis algoritme yang telah diterapkan pada bab sebelumnya. Beberapa pengujian dilakukan untuk mengetahui apakah parameter algoritme yang digunakan adalah parameter yang terbaik. Terdapat tiga pengujian yang akan dilakukan pada penelitian ini.

6.1 Pengujian dan Analisis Jumlah Partikel

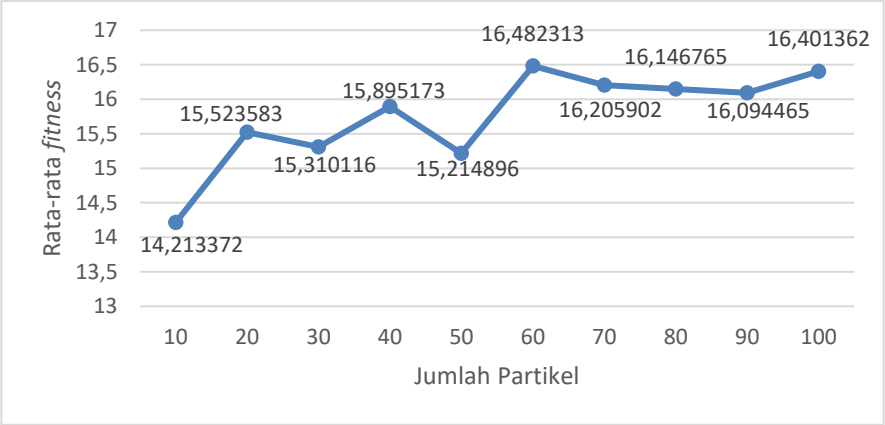
Pada sub bab ini akan dilakukan pengujian terhadap banyaknya partikel yang akan digunakan dalam penelitian ini. Tujuan dari pengujian yang dilakukan ini adalah untuk mengetahui pengaruh banyaknya partikel terhadap nilai *fitness* yang dihasilkan sehingga parameter yang terbaik didapatkan. Data pakan burung *lovebird* yang digunakan dalam pengujian ini sebanyak 17 jenis pakan yaitu millet putih, millet merah, *canary seed,* jagung, kuaci, kangkung, biji kedelai, juwawut, biji oat, kacang hijau, kacang merah, kecambah, tepung tulang, sawi putih, gabah padi, premil dan tepung telur. Parameter awal algoritme PSO yang digunakan antara lain koefisien akselerasi (c_1, c_2) masing – masing 1, bobot inersia (w) 0.5, nilai *random* (r_1, r_2) masing-masing nilai acak antara 0 dan 1 serta tiap iterasi berubah-ubah, batas bawah (x_{min}) masing-masing 0 dan batas atas (x_{max}) berurutan dengan jenis pakan 20, 15, 20, 20, 5, 20, 25, 4, 18 ,25, 5, 5, 21, 2, 30, 4, 23, jumlah iterasi (t_{max}) sebanyak 50 dan koefisien k 0,6.

Pengujian dilakukan sebanyak 10 kali. Indikator keberhasilan dari pengujian ini ialah rata-rata nilai *fitness*, di mana komposisi pakan akan memiliki rata-rata *fitness* terbesar. Tabel 6.1 merupakan hasil dari pengujian ini.

Tabel 6.1 Hasil Pengujian Jumlah Partikel

Jumlah Partikel	Nilai *fitness* percobaan ke-*i*					Rata-rata *fitness*
	1	2	3	...	10	
10	15,225450	15,618696	12,944994	...	12,636372	14,213372
20	16,382427	14,708431	13,969340	...	15,008713	15,523583
30	15,512033	15,514896	15,130564	...	17,367563	15,310116
40	13,394298	17,234211	15,039431	...	15,688830	15,895173
50	16,207303	14,755897	14,350584	...	14,931142	15,214896
60	**16,228019**	**17,242899**	**16,611698**	**...**	**16,674727**	**16,482313**
70	16,610414	19,143356	16,391402	...	14,133428	16,205902
80	17,465722	16,460384	17,228853	...	16,165624	16,146765
90	15,832510	17,225469	15,201666	...	18,747518	16,094465
100	17,690363	13,181464	16,785101	...	16,016324	16,401362

Untuk hasil pengujian ini yang lebih lengkap dapat dilihat pada Lampiran C.1. Gambar 6.1 merupakan grafik hasil rata-rata *fitness* tiap percobaan.



Gambar 6.1 Grafik Pengujian Jumlah Partikel

Dari Gambar 6.1 dapat dilihat bahwa pada uji coba jumlah partikel sebanyak 10, 20, 50 dan 60 terjadi kenaikan rata-rata *fitness* yang paling besar di antara yang lain yaitu sekitar 1,2 sampai 1,3. Hal ini dapat terjadi karena adanya keberagaman nilai partikel, di mana dalam proses inisialisasi terdapat nilai acak seperti pada Persamaan 2.1. Dapat dikatakan sebaiknya jumlah partikel tidak terlalu sedikit untuk mendapatkan hasil optimasi yang sesuai atau terlalu banyak karena dapat memengaruhi proses eksekusi algoritme PSO yang berulang – ulang. Menurut pengujian partikel ini, didapatkan banyaknya partikel yang optimal sebanyak 60 dengan nilai rata – rata *fitness* sebesar 16,482313.

6.2 Pengujian dan Analisis Nilai Koefisien *k*

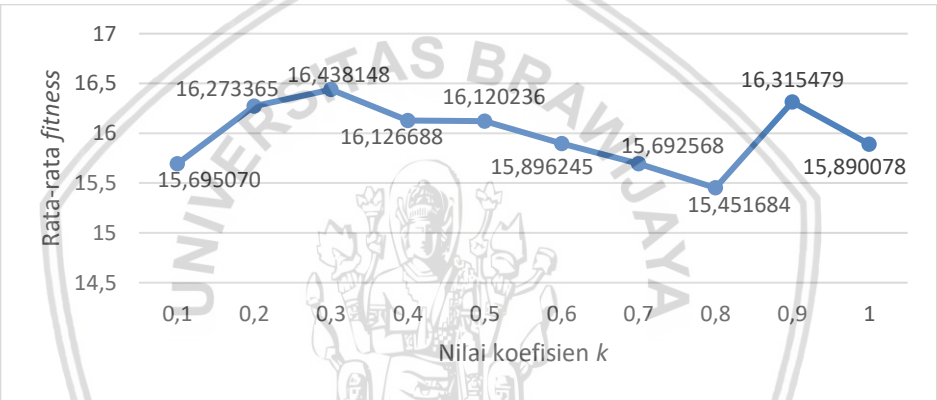
Pada sub bab ini akan dilakukan pengujian nilai koefisien *k* yang akan menentukan nilai batas bawah dan batas atas kecepatan (v_{min}, v_{max}) sesuai dengan Persamaan 2.6 dan Persamaan 2.7. Dengan dilakukannya pengujian ini, diharapkan pengaruh perubahan nilai koefisien *k* untuk menentukan batas bawah kecepatan dan batas atas kecepatan (v_{min}, v_{max}) terhadap nilai *fitness* dapat terlihat sehingga parameter yang optimal didapatkan. Data pakan burung *lovebird* yang digunakan untuk pengujian ini sebanyak 17 jenis pakan yaitu millet putih, millet merah, *canary seed,* jagung, kuaci, kangkung, biji kedelai, juwawut, biji oat, kacang hijau, kacang merah, kecambah, tepung tulang, sawi putih, gabah padi, premil dan tepung telur. Parameter awal algoritme PSO yang digunakan antara lain koefisien akselerasi (c_1, c_2) masing – masing 1, bobot inersia (w) 0.5, nilai *random* (r_1, r_2) masing-masing nilai acak antara 0 dan 1 serta tiap iterasi berubah-ubah, batas bawah (x_{min}) masing – masing 0 dan batas atas (x_{max}) berurutan dengan jenis pakan 20, 15, 20, 20, 5, 20, 25, 4, 18 ,25, 5, 5, 21, 2, 30, 4, 23, koefisien k 0,6, menggunakan jumlah partikel sesuai dengan hasil dari pengujian jumlah partikel di atas yaitu sebanyak 60 partikel.

Pengujian dilakukan sebanyak 10 kali. Indikator keberhasilan dari pengujian ini adalah rata-rata nilai *fitness*, di mana komposisi pakan yang terbaik akan memiliki rata-rata *fitness* terbesar. Tabel 6.2 ialah hasil dari pengujian ini.

Tabel 6.2 Hasil Pengujian Nilai Koefisien *k*

Nilai *k*	Nilai *fitness* percobaan ke-*i*					Rata-rata *fitness*
	1	2	3	...	10	
0,1	16,758602	15,859722	14,992380	...	15,029591	15,695070
0,2	14,259609	15,572009	15,555407	...	16,081858	16,273365
0,3	**16,150516**	**23,108255**	**15,552598**	**...**	**15,542630**	**16,438148**
0,4	15,993009	17,356172	16,324652	...	16,356201	16,126688
0,5	15,658511	15,438497	15,786712	...	16,379898	16,120236
0,6	17,044310	16,153659	15,597787	...	16,666214	15,896245
0,7	15,499363	15,897973	17,009397	...	13,787994	15,692568
0,8	14,133380	14,739242	15,371500	...	15,727876	15,451684
0,9	16,116731	16,779714	15,481428	...	17,661866	16,315479
1	16,058999	15,893780	16,924822	...	15,685308	15,890078

Untuk hasil ini yang lebih lengkap dapat dilihat pada Lampiran C.2. Gambar 6.2 merupakan grafik hasil rata-rata *fitness* tiap percobaan.



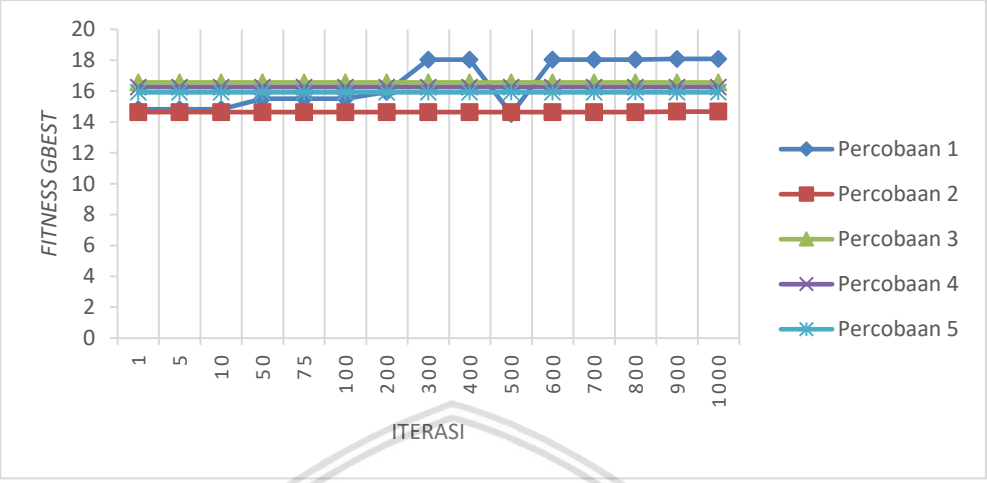
Gambar 6.2 Grafik Uji Coba Nilai *k*

Berdasarkan Gambar 6.2, dapat dilihat bahwa nilai rata-rata *fitness* pada tiap kali percobaan mengalami perubahan yang tidak terlalu signifikan. Perbedaan rata-rata *fitness* antara satu percobaan dengan yang lainnya sekitar 0,06 sampai 0,9. Dari hasil tersebut maka dapat disimpulkan bahwa nilai koefisien *k* tidak terlalu berpengaruh secara signifikan terhadap nilai *fitness* yang dihasilkan karena nilai koefisien *k* hanya untuk menentukan nilai batas kecepatan (v_{min}, v_{max}). Dari pengujian ini didapatkan hasil bahwa nilai koefisien *k* tidak terlalu memengaruhi nilai *fitness* yang dihasilkan dan nilai optimum untuk nilai koefisien *k* adalah 0,3 dengan rata – rata *fitness* sebesar 16,438148.

6.3 Pengujian dan Analisis Konvergensi

Pada sub bab ini dilakukan pengujian konvergensi yang dapat menentukan kelayakan solusi yang akan dihasilkan oleh program pada penelitian ini. Data yang digunakan untuk pengujian ini adalah data yang digunakan pada pengujian sebelumnya kecuali nilai jumlah partikel dan nilai koefisien *k* yang didapatkan dari hasil terbaik pada pengujian jumlah partikel dan pengujian nilai koefisien *k* dan iterasi yang dilakukan sebanyak 1000 kali.

Pengujian ini dilakukan sebanyak 5 kali percobaan. Untuk melihat hasil dari pengujian ini dapat dilihat pada Lampiran C.3.



Gambar 6.3 Grafik Uji Coba Konvergensi

Berdasarkan Gambar 6.3, dari keseluruhan percobaan terlihat bahwa konvergensi sudah didapatkan pada iterasi ke 600. Hal ini disebabkan oleh inisialisasi awal partikel yang dibangkitkan secara *random* dan banyaknya bahan pakan yang di proses yaitu sebanyak 18 bahan pakan sehingga membutuhkan waktu lebih untuk mencapai konvergen. Konvergen adalah ketika keragaman populasi semakin berkurang yang disebabkan oleh proses pembaruan yang terus menerus dan selisih nilai *fitness* yang dihasilkan dari iterasi satu ke yang lainnya memiliki selisih sejumlah 0. Nilai konvergen yang dicapai pada uji coba ini termasuk pada nilai global optimum karena lebih memfokuskan pada nilai *gBest* setiap iterasi tanpa memperhitungkan faktor *time variant*.

6.4 Analisis Global

Analisis global akan menggunakan parameter-parameter terbaik yang sebelumnya didapatkan dari pengujian sebelumnya. Parameter yang digunakan adalah:

- Bobot inersia (w) : 0,5
- Nilai c_1 & c_2 : 1
- Nilai r_1 & r_2 : *random*[0,1]
- Jumlah partikel : 60
- Nilai Koefisien k : 0,3
- Jumlah iterasi : 600

Analisis global ini dilakukan dengan cara menggunakan parameter optimal di atas yang akan digunakan untuk menguji sistem dengan membandingkan hasilnya dengan data sampel perhitungan kebutuhan nutrisi aktual yang didapatkan dari

peternak. Berikut adalah daftar pakan dengan kebutuhan nutrisi yang dapat dilihat pada Tabel 6.3.

Tabel 6.3 Data Perhitungan Nutrisi Manual

No.	Nama Bahan Pakan	Kandungan Nutrisi						Total Harga
		ME (kkal/kg)	Protein (%)	Lemak (%)	Serat Kasar (%)	Karbohidrat (%)	Fosfor (%)	
1	Millet Putih	94525	3000	1125	600	15125	150	2120
2	Millet Merah	328,5	11,25	3,6	1,89	52,2	0,54	
3	Kangkung	2900	30	3	19	540	500	
4	Jagung	1680	45	20,5	11	0,1	1,45	
5	Kuaci	10300	400	52	4	90	15,2	

Langkah selanjutnya dalam analisis global adalah melakukan pengujian sistem dengan menggunakan parameter terbaik sekaligus dengan data aktual sebagai perhitungannya. Berikut hasil perhitungan nutrisi yang dihasilkan oleh sistem dapat dilihat pada Tabel 6.4.

Tabel 6.4 Data Perhitungan Nutrisi Sistem

No.	Nama Bahan Pakan	Kandungan Nutrisi						Total Harga
		ME (kkal/kg)	Protein (%)	Lemak (%)	Serat Kasar (%)	Karbohidrat (%)	Fosfor (%)	
1	Millet Putih	8540,03	271,04	101,64	54,20808	1366,49535	13,55	7660
2	Millet Merah	4,8652	0,1666	0,0533	0,02799156	0,77310027	0,008	
3	Kangkung	286,072	2,9594	0,2959	1,87426431	53,2685646	49,32	
4	Jagung	5,26425	0,141	0,0642	0,03446828	0,00031335	0,005	
5	Kuaci	224,179	8,706	1,1318	0,08705968	1,9588428	0,331	

Tabel 6.5 menunjukkan selisih antara perhitungan manual dengan perhitungan dari sistem.

Tabel 6.5 Selisih Kandungan Nutrisi

No	Nama Bahan Pakan	Selisih ME (%)	Selisih Protein (%)	Selisih Lemak (%)	Selisih Serat Kasar (%)	Selisih Karbohidrat (%)	Selisih Fosfor (%)	Selisih Harga
1	Millet Putih	2274,13	2728,96	1023,36	545,79	13758,50	136,45	-5540
2	Millet Merah	8,87	11,08	3,55	1,86	51,43	0,53	
3	Kangkung	90,14	27,04	2,70	17,13	486,73	450,68	
4	Jagung	49,84	44,86	20,44	10,97	0,10	1,45	
5	Kuaci	195,65	391,29	50,87	3,91	88,04	14,87	
Rata-Rata Selisih		523,73	640,65	220,18	115,93	2876,96	120,79	-5540

Berdasarkan pada Tabel 6.5, dihasilkan rata-rata selisih kebutuhan ME, protein, lemak, serat kasar, karbohidrat dan fosfor. Rata-rata selisih nilai kebutuhan ME, protein, lemak, serat kasar, karbohidrat dan fosfor yang dihasilkan sistem masing-masing sebesar 523.73%, 640.65%, 220.18%, 115.93%, 2876.96%, dan 120.79% dan selisih harga Rp 5.540,00.

BAB 7 PENUTUP

Bab terakhir ini merupakan tahap penarikan kesimpulan serta saran yang dapat digunakan sebagai referensi penelitian selanjutnya.

7.1 Kesimpulan

1. Dengan digunakannya algoritme *Particle Swarm Optimization* (PSO), Permasalahan optimasi komposisi pakan burung *lovebird* terselesaikan. Penerapan ini di mulai dengan inisialisasi panjang dimensi sebanyak bahan pakan yang digunakan. Setelah panjang dimensi ditentukan, maka dilakukan perhitungan inisialisasi awal yang akan menghitung kecepatan, posisi, *fitness, pBest* dan *gBest* awal. Setelah inisialisasi awal selesai, maka dilakukan proses pembaruan nilai dengan meng-*update* nilai kecepatan, posisi, *fitness, pBest* dan gBest. Proses *update* ini terus dilakukan sebanyak jumlah iterasi yang telah ditentukan.
2. Berdasarkan pengujian yang telah dilakukan, didapatkan parameter terbaik, yaitu:
   - Jumlah partikel sebanyak 60 dengan rata-rata *fitness* sebesar 16,482313.
   - Nilai koefisien *k* sebesar 0,3 dengan rata-rata *fitness* sebesar 16,438148.
   - Jumlah iterasi sebanyak 600 berdasarkan pengujian konvergensi.
3. Untuk mendapatkan solusi permasalahan optimasi komposisi dilakukan perhitungan *fitness* yang di peroleh dari perhitungan bobot pakan, total harga pakan dan total penalti.
4. Hasil analisis global didapatkan rata-rata selisih harga antara data dari peternak dengan sistem sekitar Rp 5.540,00.

7.2 Saran

1. Dapat ditambahkan perhitungan untuk kebutuhan nutrisi kategori burung muda dan burung pertumbuhan.
2. Algoritme PSO adaptif seperti *Time Variant Inertia Weight* (TVIW) atau *Time Variant Acceleration Coefficient* (TVAC) dapat diterapkan pada penelitian selanjutnya untuk melihat pengaruh nilai bobot inertia dan nilai koefisien akselerasi yang beragam terhadap kualitas solusi yang dihasilkan serta populasi partikel dapat menyesuaikan dengan kondisi populasi dan jumlah iterasi yang dijalankan sehingga tidak mudah terjebak pada konvergensi dini.

DAFTAR PUSTAKA

Achmad, M., 2018. *Pakan Burung* [Wawancara] (February 2018).

Afrianto, E. & Leviawaty, E., 2005. *Pakan Ikan Pembuatan, Penyimpanan, Pengujian, Pengembangan.* Yogyakarta: Kanisius.

Barker, R. T. K. J. S. B. S. C. R. k. S. C. M., 2012. *Preliminary investigation of employee's dog presence on stress and organizational perceptions.* [Online] Available at: https://www.researchgate.net/publication/243973613_Preliminary_investigation_of_employee%27s_dog_presence_on_stress_and_organizational_perceptions
[Diakses 18 January 2018].

Cholissodin, I. & Riyandani, E., 2016. *Swarm Intelligence (Teori & Case Study).* Malang: s.n.

Darmawan, D. I., Cholissodin, I. & Dewi, C., 2017. Optimasi Formulasi Pakan pada Proses Budidaya Ikan Bandeng Menggunakan Particle Swarm Optimization (PSO). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 2(2), pp. 776-784.

Erny, 2013. Optimasi Pola Penyusunan Barang dalam Peti Kemas Menggunakan Algoritme Particle Swarm Optimization. *Jurnal Matematika Komputasi.*

Esmin, A. A. A., Coelho, R. A. & Matwin, S., 2015. A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. *The Artificial Intelligence Review,* I(44), pp. 23-45.

Hamiyanti, A. A., Achmanu, Muharlien & Putra, A., 2011. Pengaruh Jumlah Telur Terhadap Bobot Telur, Lama Mengeram, Fertilitas Serta Daya Tetas Burung Kenari. *Jurnal Ternak Terpadu,* 12(1), pp. 95-101.

Khaqqo, A., Cholissodin, I. & Widodo, A. W., 2016. Optimasi Komposisi Pakan Sapi Perah Menggunakan Algoritma Particle Swarm Optimization (PSO). *Jurnal Teknologi Informasi dan Ilmu Komputer,* 8(2).

Marginingtyas, E., Mahmudy, W. F. & Indriati, 2015. Penentuan Komposisi Pakan Ternak untuk Memenuhi Kebutuhan Nutrisi Ayam Petelur dengan Biaya Minimum Menggunakan Algoritma Genetika. *Jurnal Teknologi Informasi dan Ilmu Komputer,* 5(2).

Natsir, M. H., 2018. *Pakan Burung* [Wawancara] (February 2018).

Permana, K. E. & Hashim, S. Z. M., 2010. *Fuzzy Membership Generation Using Particle Swarm Optimization: International Journal Open Problems Computation Math No 3.* s.l., s.n.

Wardhany, B. A. K., Cholissodin, I. & Santoso, E., 2017. Penentuan Komposisi Pakan Ternak untuk Memenuhi Kebutuhan Nutrisi Ayam Petelur dengan Biaya Minimum Menggunakan Particle Swarm Optimization (PSO). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 1(12), pp. 1642-1651.

Zhang, Y. et al., 2014. *Color production in blue and green feather barbs of the rosy-faced lovebird.* China, Science Direct, pp. 130-137.