TUGAS AKHIR – TE 141599

PENENTUAN LETAK DAN KAPASITAS OPTIMAL BANK KAPASITOR PADA JARING TRANSMISI 150 KV SUMATERA UTARA MENGGUNAKAN *ARTIFICIAL BEE COLONY ALGORITHM*

Andita Noor Shafira
NRP 2214 105 004

Dosen Pembimbing
Prof. Dr. Ir. Adi Soeprijanto, M.T.
Ir. Sjamsul Anam, M.T.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016

FINAL PROJECT – **TE 141599**

OPTIMAL PLACEMENT AND SIZING OF CAPACITOR BANKS IN NORTH SUMATERA 150 KV TRANSMISSION SYSTEM USING ARTIFICIAL BEE COLONY ALGORITHM

Andita Noor Shafira
NRP 2214 105 004

Supervisor
Prof. Dr. Ir. Adi Soeprijanto, M.T.
Ir. Sjamsul Anam, M.T.

DEPARTMENT OF ELECTRICAL ENGINEERING
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2016

PENENTUAN LETAK DAN KAPASITAS OPTIMAL BANK KAPASITOR PADA JARING TRANSMISI 150 KV SUMATERA UTARA MENGGUNAKAN *ARTIFICIAL BEE COLONY ALGORITHM*

TUGAS AKHIR

Diajukan untuk Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Tenaga
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I,	Dosen Pembimbing II,
Prof. Dr. Ir. Adi Soeprijanto, M.T.	Ir. Sjamsul Anam, M.T.
NIP. 196404051990021001	NIP. 196307251990031002

SURABAYA
JUNI, 2016

PENENTUAN LETAK DAN KAPASITAS OPTIMAL BANK KAPASITOR PADA JARING TRANSMISI 150 KV SUMATERA UTARA MENGGUNAKAN *ARTIFICIAL BEE COLONY ALGORITHM*

Nama	**:**	**Andita Noor Shafira**
NRP	**:**	**2214 105 004**
Pembimbing 1	**:**	**Prof. Dr. Ir. Adi Soeprijanto, M.T.**
NIP	**:**	**196404051990021001**
Pembimbing 2	**:**	**Ir. Sjamsul Anam, M.T.**
NIP	**:**	**196307251990031002**

ABSTRAK

Listrik merupakan suatu kebutuhan mutlak yang harus dipenuhi untuk menjamin keberlangsungan hidup masyarakat masa kini. Kebutuhan ini terus meningkat seiring dengan pertumbuhan beban yang semakin bertambah dari tahun ke tahun. Pertumbuhan beban yang diikuti dengan peningkatan permintaan suplai daya reaktif akibat beban bersifat induktif meningkat menyebabkan perencanaan dan operasi dari sistem interkoneksi menjadi lebih kompleks sehingga kualitas sistem menjadi kurang dapat diandalkan. Aliran daya reaktif dapat menyebabkan drop tegangan dan kerugian daya dalam sistem transmisi. Untuk itu dilakukan penentuan letak dan kapasitas kapasitor shunt untuk mengurangi kerugian daya dengan menggunakan *Newton-Raphson* dan metode optimisasi *Artificial Bee Colony Algorithm*. Pada percobaan ini dilakukan pemasangan lima kapasitor dengan jumlah koloni sebesar 50 dan *Max Cycle Number* sebesar 150. Hasil simulasi menggunakan metode *Artificial Bee Colony Algorithm* menunjukkan bahwa pemasangan kapasitor pada Jaring Transmisi 150 kV Sumatera Utara dapat menurunkan kerugian daya aktif sebesar 8,37%.

Kata Kunci: Aliran Daya Reaktif, Kapasitor, *Artificial Bee Colony Algorithm*.

Halaman ini sengaja dikosongkan

OPTIMAL PLACEMENT AND SIZING OF CAPACITOR BANKS IN NORTH SUMATERA 150 KV TRANSMISSION SYSTEM USING ARTIFICIAL BEE COLONY ALGORITHM

Name	**:**	**Andita Noor Shafira**
Register Number	**:**	**2214 105 004**
***Supervisor* 1**	**:**	**Prof. Dr. Ir. Adi Soeprijanto, M.T.**
ID	**:**	**196404051990021001**
***Supervisor* 2**	**:**	**Ir. Sjamsul Anam, M.T.**
ID	**:**	**196307251990031002**

ABSTRACT

Electricity is an absolute necessity that must be met to ensure the survival of communities. This necessity has improved along with the growth of load from year to year. The growth of load which followed by high reactive power demand due to high inductive load causes the planning and operation of the interconnection system becomes more complex so that the quality of the system becomes less reliable. Reactive power flow can cause a voltage drop and line losses in the transmission system. For that reason, the determination of the location and capacity of shunt capacitor is needed to reduce power losses using Newton-Raphson and Artificial Bee Colony Algorithm optimization method. In this experiment was installed five capacitors with 50 colony sizes and 150 of Max Cycle Number. Simulation results using Artificial Bee Colony Algorithm method showed that the installation of capacitors at 150 kV Sumatera Utara Transmission System can reduce 8,37 % active power losses.

Keywords: *Reactive Power Flow, Capacitor Banks, Artificial Bee Colony Algorithm.*

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

INDEKS

Halaman ini sengaja dikosongkan

NOMENKLATUR

Simbol	Definisi	Satuan
I	Arus	(Ampere)
V	Tegangan	(Volt)
X_L	Reaktansi induktif	(Ω)
X_C	Reaktansi kapasitif	(Ω)
S	Daya total	(VA)
P	Daya aktif	(Watt)
Q	Daya reaktif	(Var)
V_S	Tegangan *line to neutral* sisi kirim	(Volt)
V_L	Tegangan *line to neutral* sisi terima	(Volt)
I_R	Arus	(Ampere)
I_X	Arus reaktif	(Ampere)
Z	Total impedansi seri	(Ω)
R	Resistansi	(Ω)
jX	Reaktansi	(Ω)
Q_C	Besar daya injeksi kapasitor	(Var)
Load	Beban	
I_{ij}	Arus injeksi	(Ampere)
I_{ji}	Arus injeksi	(Ampere)
V_i	Tegangan node	(Volt)
V_j	Tegangan node	(Volt)
I_1	Arus cabang	(Ampere)
S_{Lij}	Kerugian daya kompleks dari bus i ke bus j	(VA)
P_{Lij}	Kerugian daya aktif dari bus i ke bus j	(Watt)
g_{ij}	Konduktansi pada saluran dari bus i ke bus j	(Siemens)
θ	Selisih sudut tegangan antara bus i ke bus j	(derajat)
ΔP	Selisih injeksi netto daya aktif	(Watt)
ΔQ	Selisih injeksi netto daya reaktif	(Var)
$\Delta\delta$	Vektor koreksi sudut fasa - tegangan	(derajat)
$\Delta\|V$	Vektor koreksi magnitude tegangan	(Volt)

Halaman ini sengaja dikosongkan

Halaman ini sengaja dikosongkan

BAB I
PENDAHULUAN

1.1 Latar Belakang

Listrik merupakan suatu kebutuhan mutlak yang harus dipenuhi untuk menjamin keberlangsungan hidup masyarakat masa kini. Pemenuhan kebutuhan ini terus meningkat seiring dengan pertumbuhan beban yang semakin bertambah dari tahun ke tahun. Pertumbuhan beban ini diikuti dengan peningkatan permintaan suplai daya reaktif akibat beban yang bersifat induktif meningkat. Bila suatu jaring transmisi tidak memiliki sumber daya reaktif di daerah sekitar beban, maka semua kebutuhan beban reaktif dipikul oleh generator sehingga akan mengalir arus reaktif pada jaring transmisi yang mengakibatkan penurunan faktor daya, kerugian daya besar, dan jatuh tegangan pada ujung saluran meningkat.

Alternatif untuk mengurangi dampak dari arus reaktif yang meningkat adalah dengan melakukan kompensasi daya reaktif yang bertujuan untuk transportasi daya reaktif dan mengurangi kerugian daya. Salah satu langkah penyelesaian yang umum dilakukan adalah dengan penambahan kapasitor pada sistem. Kapasitor berguna sebagai sumber daya reaktif tambahan untuk mengkompensasi daya reaktif akibat pembebanan tersebut [1-2]. Dengan memasang *shunt capacitor* (kapasitor paralel), maka akan diperoleh keuntungan antara lain kerugian daya yang menurun, tegangan beban meningkat dan efisiensi peralatan di saluran transmisi yang meningkat pula sehingga memungkinkan untuk menambah beban tanpa menambah saluran baru.

Penentuan lokasi pemasangan kapasitor dan kapasitas yang optimal untuk dialokasikan pada jaring transmisi menjadi suatu permasalahan yang sering terjadi. Oleh sebab itu, digunakan salah satu metode optimisasi sebagai alat bantu. Terdapat dua metode optimisasi, yaitu metode deterministik seperti *Dynamic Programming*, *Simplex*, dan *Linear Programming* serta metode *undeterministik* seperti *Ant Colony Algorithm*, *Simulated Annealing*, *Genetic Algorithm*, serta *Artificial Bee Colony*.

Pada Tugas Akhir ini, metode yang diusulkan adalah *Artificial Bee Colony* (ABC) *Algorithm. Artificial Bee Colony* (ABC) *Algorithm* merupakan suatu algoritma yang dikenalkan oleh Karaboga pada tahun 2005 sebagai suatu teknik masalah optimasi numerik (Karaboga, 2005).

Algoritma ini dibuat berdasarkan teknik *metaheuristic* untuk mendapatkan hasil optimal dari suatu permasalahan yang telah diterapkan pada algoritma pendahulunya, seperti *Ant Colony Algorithm*, *Particle Swarm Optimization*, *Harmony Search*, dan lain sebagainya. Metode ini dikembangkan berdasarkan perilaku kecerdasan lebah madu dalam suatu koloninya dan performasinya dan dijadikan tolak ukur untuk menghitung nilai suatu fungsi optimisasi.

1.2 Permasalahan

Permasalahan yang akan dibahas dalam Tugas Akhir ini adalah bagaimana menentukan letak dan besar kapasitas optimal bank kapasitor yang akan dipasang karena adanya kerugian daya yang besar pada Jaring Transmisi 150 kV Sumatera Utara sehingga diperoleh kerugian daya seminimal mungkin.

1.3 Tujuan Penelitian

Tujuan dari penulisan Tugas Akhir ini adalah :

1. Dapat mensimulasikan pencarian lokasi dan ukuran bank kapasitor yang optimal pada Jaring Transmisi 150 kV Sumatera Utara menggunakan *Artificial Bee Colony* (ABC) *Algorithm*.
2. Mengetahui dampak dari pemasangan bank kapasitor pada Jaring Transmisi 150 kV Sumatera Utara.
3. Mengetahui perbedaan sebelum dilakukan pemasangan bank kapasitor dan setelah dilakukan pemasangan bank kapasitor pada sistem.
4. Memperbaiki kualitas daya, dalam hal ini diperoleh kerugian daya minimum pada Jaring Transmisi 150 kV Sumatera Utara.
5. Mengaplikasikan metode *Artificial Bee Colony* (ABC) *Algorithm* pada Jaring Transmisi 150 kV Sumatera Utara sebagai salah satu teknik komputasi untuk menyelesaikan permasalahan optimisasi.

1.4 Batasan Masalah

Batasan masalah dalam pengerjaan Tugas Akhir ini antara lain :

1. Hasil Tugas Akhir berupa simulasi dan analisis.
2. Sistem yang digunakan adalah sistem kelistrikan Sumatera Utara 150 kV dengan 29 bus.
3. Simulasi dilakukan dengan menggunakan MATLAB R2012a.

4. Metode yang dipakai untuk menyelesaikan permasalahan optimisasi adalah dengan menggunakan *Artificial Bee Colony* (ABC) *Algorithm*.
5. Sistem beroperasi dalam keadan normal, tidak dalam gangguan.
6. Fungsi objektif pada optimisasi yang dilakukan adalah mengurangi kerugian daya aktif pada Jaring Transmisi 150 kV Sumatera Utara.
7. Faktor harmonisa akibat pemasangan kapasitor diabaikan.
8. Faktor ekonomis tidak diperhitungkan.

1.5 Metode Penelitian

Penulisan dan penyusunan Tugas Akhir ini menggunakan metode penelitian sebagai berikut :

1. Studi Literatur
   Studi literatur sangat dibutuhkan dan perlu dilakukan untuk menunjang penguasaan materi yang mendukung dalam pengerjaan Tugas Akhir, seperti pencarian pustaka yang digunakan, pengumpulan data dan pemodelan sistem.
2. Pengumpulan Data
   Penulis melakukan pengambilan data kelistrikan Sumatera Utara 150 kV berupa data beban, data pembangkitan, data saluran dan *single line diagram* yang selanjutnya disimulasikan menggunakan *software* MATLAB.
3. Perencanaan Kapasitor
   Besar aliran daya reaktif menimbulkan kerugian pada jaring transmisi. Pemasangan kapasitor pada Jaring Transmisi Sumatera Utara 150 kV menyebabkan arus reaktif yang mengalir pada saluran berkurang sehingga kerugian daya dapat diminimalisirkan.
4. Simulasi
   Simulasi dilakukan dengan menggunakan *software* MATLAB R2012a dan program *Artificial Bee Colony* yang digunakan dalam optimisasi ditulis dalam M-file.
5. Analisis dan Perbandingan
   Menganalisis perbandingan antara sebelum dan sesudah dilakukan penempatan kapasitor menggunakan metode *Artificial Bee Colony* (ABC) *Algorithm.*

6. Penulisan Buku Tugas Akhir
   Penulisan laporan dilakukan sebagai penggambaran kesimpulan dari Tugas Akhir ini. Kesimpulan tersebut merupakan hasil analisis dan perbandingan. Selain itu juga akan diberikan saran sebagai masukan berkaitan dengan apa yang telah dilakukan.

1.6 Sistematika Penulisan

Adapun susunan sistematika laporan Tugas Akhir ini adalah sebagai berikut :

BAB 1 PENDAHULUAN

Bab ini berisi tentang latar belakang, permasalahan, tujuan penelitian, batasan masalah, metode penelitian, sistematika penulisan, dan relevansi.

BAB 2 ALIRAN DAYA REAKTIF PADA SISTEM TENAGA LISTRIK

Bab ini berisi tentang teori-teori yang berkaitan dengan topik penelitian seperti aliran daya reaktif, pengaruh faktor daya yang rendah, pengaruh kompensasi pada beban induktif, koreksi faktor daya, keuntungan perbaikan faktor daya, persamaan aliran daya, perhitungan rugi jaring transmisi dan metode *Newton-Raphson*.

BAB 3 *ARTIFICIAL BEE COLONY (ABC) ALGORITHM*

Bab ini berisi tentang penerapan *Artificial Bee Colony* (ABC) *Algorithm* pada proses optimisasi penentuan letak dan kapasitas kapasitor .

BAB 4 SIMULASI DAN ANALISIS

Bab ini berisi tentang hasil simulasi dan analisis dari penentuan letak dan kapasitas kapasitor menggunakan *Artificial Bee Colony* (ABC) *Algorithm.*

BAB 5 PENUTUP

Bab ini berisi tentang kesimpulan dan saran mengenai hasil penulisan laporan Tugas Akhir.

1.7 Relevansi

Hasil yang diperoleh dari Tugas Akhir ini diharapkan dapat menjadi referensi untuk pengembangan penelitian selanjutnya, terutama yang berkaitan dengan penerapan *Artificial Bee Colony* (ABC) *Algorithm* sebagai salah satu teknik komputasi dalam menyelesaikan permasalahan penentuan letak dan kapasitas kapasitor secara optimal.

BAB II
ALIRAN DAYA REAKTIF PADA SISTEM TENAGA LISTRIK

2.1 Pendahuluan

Ketergantungan masyarakat modern pada listrik yang semakin meningkat telah memaksa sistem tenaga listrik untuk beroperasi dengan tingkat keandalan yang sangat tinggi. Selain itu, kualitas daya listrik (*power quality*) sekarang ini menjadi perhatian utama para pengguna listrik sehingga memaksa penyedia listrik untuk mengembangkan dan menerapkan standar yang lebih ketat agar pelayanan yang diberikan kepada pelanggan atau pengguna listrik dapat maksimal.

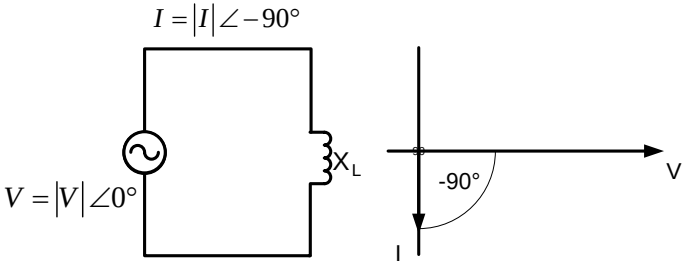
Pada saat ini, permintaan beban meningkat pesat seiring dengan kebutuhan akan penggunaan alat-alat listrik yang semakin meningkat pula. Penggunaan motor pada berbagai macam perlatan rumah tangga sampai pada industri skala besar membutuhkan daya reaktif yang sangat besar. Pengiriman daya reaktif dari penyedia listrik sampai ke pengguna akan mempengaruhi kemampuan dari saluran transmisi. Kapasitas dari saluran transmisi akan menurun seiring dengan peningkatan daya reaktif yang disalurkan. Aliran daya reaktif ini akan menyebabkan beberapa masalah, diantaranya *drop* tegangan pada sisi penerima dan kerugian daya pada saluran transmisi meningkat. Untuk mengatasi permasalahan kestabilan pada sistem tenaga listrik, dapat dilakukan dengan kompensasi daya reaktif. Penentuan lokasi pemasangan kapasitor dan kapasitas yang optimal untuk dialokasikan pada jaring transmisi menjadi suatu permasalahan yang sering terjadi. Optimisasi dapat dicapai dengan mempertimbangkan batas operasional dan kualitas daya yang dibutuhkan [1].

Berbagai usaha telah dilakukan untuk menentukan letak dan kapasitas kapasitor. Suatu metode yang digunakan untuk menyelesaikan permasalahan penentuan letak kapasitor sebagai kompensasi daya reaktif adalah dengan menggunakan program aplikasi [2]. Ji-Pyng Chiou dan kawan-kawan melakukan penelitian menggunakan VSHDE (*Variable Scaling Hybrid Differential Evolution*) untuk menentukan letak kapasitor pada jaring distribusi dalam skala besar [3,4]. Ahmed M. Azmy mengoptimalkan aliran daya untuk mengatur profil tegangan pada sistem terinterkoneksi [5]. S.K. Bhattacharya dan S.K. Goswami melakukan penempatan kapasitor mengunakan metode *Fuzzy* [6].

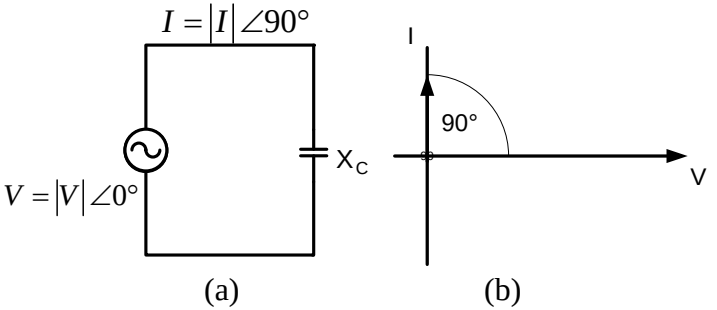
2.2 Daya Reaktif dalam Suatu Sistem

Daya aktif dari rangkaian AC diperoleh dari perkalian tegangan dan komponen arus yang sefase. Jika sebuah beban induktif murni dihubungkan dengan sumber tegangan (Volt) akan menghasilkan arus lagging, yaitu arus tertinggal atau terbelakang 90^o terhadap tegangan. Sebaliknya, jika sebuah beban kapasitif murni dihubungkan dengan sumber tegangan (Volt) akan menghasilkan arus leading, yaitu arus mendahului 90^o terhadap tegangan.

Gambar 2.1 dan 2.2 menunjukkan rangkaian dan diagram phasor antara arus terhadap tegangan suatu beban yang disuplai oleh sumber tegangan.



Gambar 2.1 Rangkaian dengan beban induktif
a) Rangkaian AC dengan beban elemen induktif
b) Diagram phasor rangkaian beban induktif



Gambar 2.2 Rangkaian dengan beban kapasitif
a) Rangkaian AC dengan beban elemen kapasitif
b) Diagram phasor rangkaian beban kapasitif

Daya listrik dibagi menjadi tiga elemen yang dapat diketahui masing-masing elemen dari daya tersebut, yaitu :

1. Daya total (S)) : $S = VI^*$ (VA) (2.1)
2. Daya aktif (P) : $P = VI\cos\theta$ (Watt) (2.2)
   Disebut sebagai daya nyata .
3. Daya reaktif (Q) : $Q = VI\sin\theta$ (Var) (2.3)
   Disebut sebagai daya semu, bernilai ”positif” bila beban induktif dan bernilai ”negatif” bila beban kapasitif.

Faktor daya dapat diketahui dari persamaan 2.4 berikut :

$$\text{Faktor daya} = \frac{\text{daya aktif (KW)}}{\text{daya komplek (KVA)}} = \cos\theta \quad (2.4)$$

Faktor daya dikatakan ”lagging” apabila beban induktif dan sedangkan ”leading” apabila beban kapasitif.

2.3 Pengaruh Faktor Daya yang Rendah

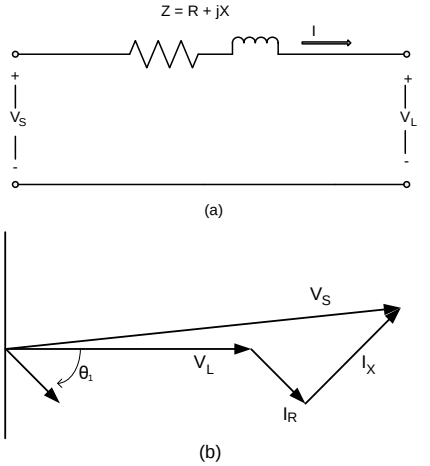
Faktor daya yang rendah memiliki beberapa pengaruh buruk yang tentu saja merupakan kerugian pada sistem tenaga listrik, diantaranya :

1. **Pusat pembangkit**
   Apabila generator menaikkan suplai daya reaktif, maka daya nyata yang dibangkitkan meningkat. Untuk itu, penguatan arus harus ditambah sedangkan penguatan arus dibatasi oleh rating generator. Penambahan arus penguatan menyebabkan kenaikan rugi besi generator yang dapat menurunkan efisiensi generator tersebut.
2. **Sistem transmisi**
   Faktor daya yang buruk berhubungan dengan arus reaktif yang mengalir pada sistem menyebabkan kapasitas dari peralatan pada sistem transmisi menurun seperti kabel, transformator daya, busbar dan peralatan lainnya.
3. **Konsumen**
   Dengan pertambahan beban induktif dapat menyebabkan pembangkit harus menaikkan suplai kapasitifnya sehingga suplai daya nyata bertambah yang menyebabkan instalasi pada konsumen seperti pengaman, kabel maupun peralatan lain harus ditingkatkan karena akan membatasi penggunaan beban pada kebutuhan daya aktif yang sama.

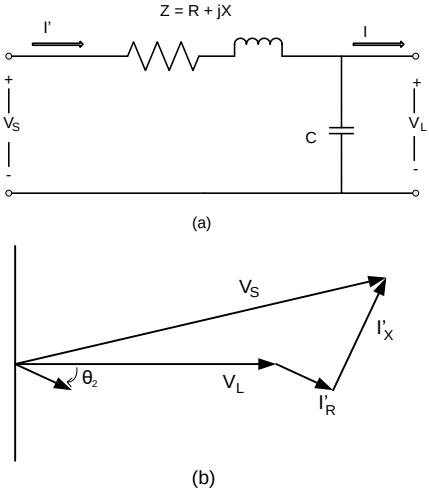
2.4 Pengaruh Kompensasi pada Beban Induktif

Kompensasi beban induktif dilakukan untuk meningkatkan kualitas daya, salah satunya yaitu kerugian daya seminimal mungkin. Dalam hal ini, aliran daya reaktif dapat dikontrol dengan cara memasang peralatan kompensasi paralel pada bus beban untuk menjaga keseimbangan yang tepat antara daya reaktif yang dihasilkan dan daya reaktif yang digunakan. Cara tersebut paling efektif dalam meningkatkan kemampuan transfer daya dari sistem dan meningkatkan stabilitas tegangan.

Kapasitor ini terhubung paralel pada jaring dengan tujuan untuk mengurangi kerugian daya pada jaring transmisi. Gambar 2.3 menunjukkan bahwa dengan menggunakan kapasitor, maka arus reaktif yang mengalir pada saluran dapat berkurang sehingga kerugian daya dapat diminimalisirkan.



Gambar 2.3 Sebelum dipasang kapasitor shunt
(a) Rangkaian ekivalen dari saluran
(b) Diagram vektor dari rangkaian ekivalen



Gambar 2.4 Setelah dipasang kapasitor shunt
(a) Rangkaian ekivalen dari saluran
(b) Diagram vektor dari rangkaian ekivalen

Dari Gambar 2.3 dan Gambar 2.4 diperoleh,

$$P_{Loss} = I^2 R \tag{2.5}$$

$$P_{Loss} = \left(\frac{S}{V}\right)^2 R \tag{2.6}$$

$$P_{Loss} = \left(\frac{\sqrt{P^2+Q^2}}{V}\right)^2 R \tag{2.7}$$

$$P_{Loss} = \frac{P^2+Q^2}{V^2} R \tag{2.8}$$

$$P_{Loss} = \frac{P^2+(Q^2-Q_C^2)}{V^2} R \tag{2.9}$$

$$P_{Loss} = I_{TURUN} R$$

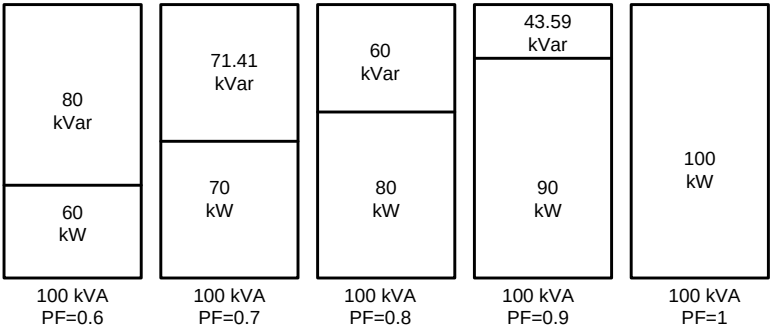
sehingga,

$$P_{Loss} = TURUN$$

2.5 Koreksi Faktor Daya

Pembangkitan daya reaktif pada perencanaan daya dan pensuplaiannya ke beban-beban yang berlokasi pada jarak yang jauh

adalah tidak ekonomis, tetapi dapat dengan mudah disediakan oleh kapasitor yang ditempatkan pada pusat beban.



Gambar 2.5 Ilustrasi perubahan daya akibat perubahan faktor daya

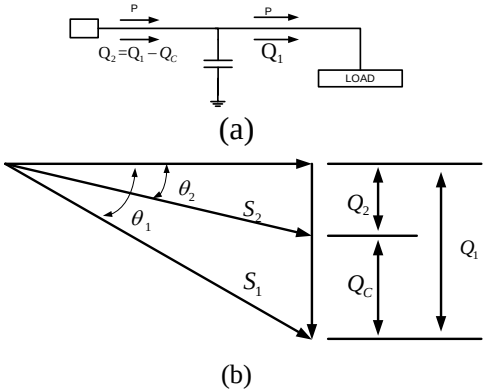
Dengan mengasumsikan bahwa beban disuplai dengan daya nyata (aktif) P, daya reaktif tertinggal (*lagging*) Q_1, dan daya semu S_1, maka rumus persamaan dari faktor daya tertinggal adalah sebagai berikut :

$$\cos\theta_1 = \frac{P}{S_1} = \frac{P}{(P^2 + Q_1^2)^{\frac{1}{2}}} \tag{2.10}$$

ketika kapasitor *shunt* Qc dipasang pada beban, faktor daya dapat ditingkatkan dari $\cos\theta_1$ menjadi $\cos\theta_2$ yang dijabarkan pada perumusan 2.11 sebagai berikut :

$$\cos\theta_2 = \frac{P}{S_2}$$

$$= \frac{P}{(P^2 + Q_2^2)^{\frac{1}{2}}} = \frac{P}{\left[P^2 + (Q_1 - Q_c)^2\right]^{\frac{1}{2}}} \tag{2.11}$$



Gambar 2.6 Ilustrasi koreksi faktor daya

Gambar 2.6 menunjukkan bahwa daya semu dan daya reaktif menurun dari S_1 kVA menjadi S_2 kVA dan dari Q_1 kvar menjadi Q_2 kvar. Tentu saja, penurunan hasil daya reaktif dalam penurunan arus total disebabkan oleh penurunan penyusutan daya sehingga koreksi faktor daya menghasilkan penghematan ekonomi dalam pengeluaran yang besar dan pengeluaran bahan bakar melalui pengurangan kapasitas kVA dan penurunan kerugian daya dalam semua perlengkapan diantara titik yang dipasang kapasitor dan rencana sumber daya, termasuk saluran distribusi, trafo di gardu induk dan saluran transmisi.

Keuntungan lain dari pemasangan kapasitor adalah perbaikan faktor daya dan pengurangan kVA yang mengalir pada jaring. Dengan memasang kapasitor maka akan mengurangi daya reaktif yang mengalir pada jaring sehingga dengan daya nyata yang sama, faktor daya akan menjadi lebih besar dan kVA akan berkurang. Dengan adanya perbaikan faktor daya, maka akan timbul pengurangan kVA yang mengalir pada jaring sehingga pada jaring tersebut dapat ditambahkan sejumlah kVA sebesar pengurangan kVA yang terjadi akibat pemasangan kapasitor. Dampak dari penambahan kVA pada suatu jaring, yaitu menambah jumlah beban yang dapat ditanggung oleh jaring tersebut. Hal ini merupakan suatu keuntungan, karena apabila ada tambahan beban pada daerah dimana jaring itu berada, maka daya listrik dapat dikirim melalui jaring tersebut tanpa perlu membangun jaring yang baru. Pada Tugas Akhir ini, suatu nilai optimum diperoleh apabila pemasangan kapasitor telah mencapai titik rugi saluran paling minimum [7].

2.6 Keuntungan Perbaikan Faktor Daya

Adapun keuntungan dari perbaikan faktor daya adalah sebagai berikut :

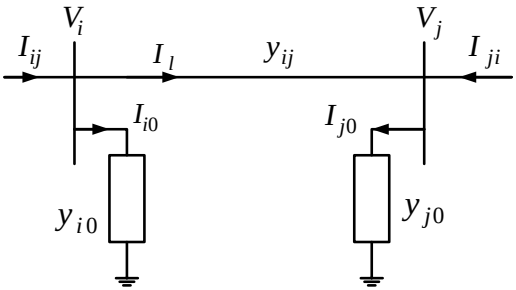
1. Pengaturan tegangan menjadi lebih baik.
2. Menaikkan tingkat level tegangan beban.
3. Menaikkan efisiensi peralatan.
4. Menaikkan faktor daya generator.
5. Mengurangi kerugian I^2R dan I^2X pada jaring transmisi karena pengurangan aliran daya reaktif .
6. Menaikkan kapasitas KW peralatan seperti transformator, motor, alternator dan kawat transmisi, sehingga mengurangi dampak *overload* dari peralatan tersebut.
7. Pengurangan KVA beban sehingga memperpanjang umur peralatan.
8. Mengurangi biaya yang harus dikeluarkan oleh pembangkit untuk memenuhi kebutuhan daya reaktif.

2.7 Persamaan Aliran Daya

Sebelum melakukan proses kompensasi daya reaktif pada sistem transmisi perlu mengetahui aliran daya pada sistem transmisi dan diketahui bahwa persamaan pada permasalahan analisis aliran daya adalah permasalahan *Algebraic Nonlinear Equation* yang harus diselesaikan menggunakan teknik iterasi. Ada beberapa teknik yang umum digunakan, namun pada Tugas Akhir ini metode yang digunakan untuk analisis aliran daya adalah metode *Newton-Rapshon* [8].

2.7.1 Perhitungan Kerugian Jaring Transmisi

Pemasangan kapasitor mempertimbangkan kerugian daya pada saluran sehingga perlu mengkalkulasi kerugian jaring transmisi dari perhitungan *load flow*.



Gambar 2.7 Pemodelan jaring transmisi untuk perhitungan aliran daya

Dengan memisalkan suatu saluran yang menghubungkan bus *i* dan *j* seperti ditunjukkan pada Gambar 2.7 maka arus pada saluran dapat diperoleh dari persamaan berikut,
dari bus *i* ke *j*

$$I_{ij} = I_l + I_{i0} = y_{ij}(V_i - V_j) + y_{i0}V_i \tag{2.12}$$

sedangkan dari bus *j* ke *i*

$$I_{ji} = -I_l + I_{j0} = y_{ij}(V_j - V_i) + y_{j0}V_j \tag{2.13}$$

Besar daya yang mengalir dan kerugian yang terjadi tiap saluran ditunjukkan dalam persamaan berikut,

$$S_{ij} = V_i I_{ij}^* = V_i(V_i^* - V_j^*)y_{ij}^* + V_i V_i^* y_{io}^* \tag{2.14}$$

$$S_{ji} = V_j I_{ji}^* = V_j(V_j^* - V_i^*)y_{ij}^* + V_j V_j^* y_{jo}^* \tag{2.15}$$

Kerugian daya kompleks pada jaring transmisi dari bus *i* ke *j* adalah penjumlahan persamaan yang dijelaskan dalam persamaan berikut,

$$S_{Lij} = S_{ij} + S_{ji} \tag{2.16}$$

Pada Tugas Akhir ini, kerugian daya yang diperhitungkan sebagai acuan dalam proses kompensasi (nilai fitness) adalah kerugian daya aktif. Dengan memperhitungkan konduktansi saluran dari bus *i* ke *j* dan tanpa memperhitungkan impedansi saluran antara bus ke beban diperoleh persamaan sebagai berikut :

$$P_{Lij} = g_{ij}\left[V_i^2 + V_j^2 - 2V_iV_j\cos\theta\right] \tag{2.17}$$

dengan g_{ij} adalah konduktansi pada saluran dari bus *i* ke *j* dan θ adalah selisih sudut tegangan antara bus *i* ke *j*.

2.7.2 Metode *Newton-Raphson*

Metode *Newton-Raphson* memiliki perhitungan yang lebih baik untuk aplikasi pada sistem yang besar dalam menyelesaikan persamaan dengan dua variabel atau lebih. Jumlah iterasi yang dibutuhkan untuk memperoleh pemecahan masalah ditentukan oleh besar sistem yang digunakan.

Besar arus pada tenaga listrik dan besar daya yang keluar dan daya yang masuk ke bus dapat diketahui dengan menggunakan persamaan sebagai berikut :

$$I_i = \sum_{j=i}^{N} Y_{ij} V_{ij} \tag{2.18}$$

Persamaan diatas bila ditulis dalam bentuk polar adalah :

$$I_i = \sum_{j=1}^{n} |Y_{ij}||V_j| \angle \theta_{ij} + \delta_j \tag{2.19}$$

Daya kompleks pada bus i adalah :

$$P_i - jQ_i = V_i^* I_i \tag{2.20}$$

Substitusi dari persamaan 2.19 untuk I_i kedalam persamaan 2.20 menghasilkan :

$$P_i - jQ_i = |V_i| \angle \delta_i \sum_{j=1}^{n} |Y_{ij}||V_j| \angle \theta_{ij} + \delta_j \tag{2.21}$$

Bagian riil dan imajiner dipisahkan sehingga bentuk persamaan tersebut menjadi :

$$P_i = \sum_{j=1}^{n} |V_i||V_j||Y_{ij}| \cos(\theta_{ij} - \delta_i + \delta_j) \tag{2.22}$$

$$Q_i = -\sum_{j=1}^{n} |V_i||V_j||Y_{ij}| \sin(\theta_{ij} - \delta_i + \delta_j) \tag{2.23}$$

Persamaan 2.22 dan 2.23 membentuk persamaan aljabar *non-linier* dengan variabel sendiri. Besar setiap variabel dinyatakan dalam satuan per unit, sedangkan untuk sudut fasa dinyatakan dalam satuan radian. Persamaan 2.22 dan 2.23 dikembangkan menjadi deret *Taylor*, dan dalam bentuk singkat, deret tersebut dapat ditulis sebagai berikut [7] :

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \begin{bmatrix} \Delta \delta \\ \Delta |V| \end{bmatrix} \tag{2.24}$$

Elemen matriks *Jacobian* ditentukan dengan (2n-2-m) x (2n-2-m) dengan n adalah jumlah bus pada sistem, sedangkan m adalah jumlah *voltage-controlled bus* (bus tegangan) sistem. J_1 diperoleh dari (n-1) x (n-1), J_2 diperoleh dari (n-1)x(n-1-m), J_3 diperoleh dari (n-1-m)x(n-1) dan J_4 diperoleh dari (n-1-m)x(n-1-m).

Untuk elemen diagonal dan diagonal luar $\mathbf{J_1}$ adalah :

$$\frac{\partial P_i}{\partial \delta_i} = \sum_{j \neq 1} |V_i||V_j||Y_{ij}|\sin(\theta_{ij} - \delta_i + \delta_j) \qquad (2.25)$$

$$\frac{\partial P_i}{\partial \delta_j} = -|V_i||V_j||Y_{ij}|\sin(\theta_{ij} - \delta_i + \delta_j) \qquad j \neq i \qquad (2.26)$$

Untuk elemen diagonal dan diagonal luar $\mathbf{J_2}$ adalah:

$$\frac{\partial P_i}{\partial |V_i|} = 2|V_i||Y_{ii}|\cos\theta_{ii} + \sum_{j \neq i} |V_j||Y_{ij}|\cos\left(\theta_{ij} - \delta_i + \delta j\right) \qquad (2.27)$$

$$\frac{\partial P_i}{\partial |V_j|} = |V_i||Y_{ij}|\cos\left(\theta_{ij} - \delta_i + \delta j\right) \qquad j \neq i \qquad (2.28)$$

Untuk elemen diagonal dan diagonal luar $\mathbf{J_3}$ adalah:

$$\frac{\partial Q_i}{\partial \delta_i} = \sum_{j \neq 1} |V_i||V_j||Y_{ij}|\cos(\theta_{ij} - \delta_i + \delta_j) \qquad (2.29)$$

$$\frac{\partial Q_i}{\partial \delta_j} = -|V_i||Y_{ij}|\cos\left(\theta_{ij} - \delta_i + \delta j\right) \qquad j \neq i \qquad (2.30)$$

Untuk elemen diagonal dan diagonal luar $\mathbf{J_4}$ adalah:

$$\frac{\partial Q_i}{\partial |V_i|} = -2|V_i||Y_{ii}|\sin\theta_{ii} - \sum_{j \neq i} |V_j||Y_{ij}|\sin\left(\theta_{ij} - \delta_i + \delta j\right) \qquad (2.31)$$

$$\frac{\partial Q_i}{\partial |V_j|} = -|V_i||Y_{ij}|\sin\left(\theta_{ij} - \delta_i + \delta j\right) \qquad j \neq i \qquad (2.32)$$

Harga dari $\Delta P_i^{(k)}$ dan $\Delta Q_i^{(k)}$ berbeda antara yang terjadwal dengan nilai perhitungan, dan disebut *power residual* diberikan dengan persamaan 2.33 - 2.34 :

$$\Delta P_i^{(k)} = P_i^{sch} - P_i^{(k)} \quad (2.33)$$

$$\Delta Q_i^{(k)} = Q_i^{sch} - Q_i^{(k)} \quad (2.34)$$

Perhitungan baru untuk sudut fasa dan tegangan pada bus adalah :

$$\delta_i^{(k+1)} = \delta_i^{(k)} + \Delta\delta_i^{(k)} \quad (2.35)$$

$$\left|V_i^{(k+1)}\right| = \left|V_i^{(k)}\right| + \Delta\left|V_i^{(k)}\right| \quad (2.36)$$

Prosedur penyelesaian studi aliran daya dengan metode *Newton-Raphson* adalah sebagai berikut :

1. Pada *bus* berbeda dimana harga P_i^{sch} dan Q_i^{sch} ditentukan. Besar tegangan dan susut fasa disamakan dengan nilai *slack bus* atau 1,0 dan 0,0, jadi $\left|V_i^{(0)}\right|$ =1,0 dan $\delta_i^{(0)}$ =0,0. Untuk *voltage regulated* bus, nilai $|Vi|$ dan P_i^{sch} ditentukan, sedangkan sudut fasa disamakan dengan sudut *slack bus*, jadi $\delta_i^{(0)} = 0$.
2. Hitung P_i^k dan Q_i^k pada bus beban dengan persamaan 2.22 dan persamaan 2.23, dan juga $\Delta P_i^{(k)}$ dan $\Delta Q_i^{(k)}$ dihitung dengan persamaan 2.33 dan persamaan 2.34.
3. Hitung P_i^k dan $\Delta P_i^{(k)}$ pada *voltage control bus* dengan persamaan 2.22 dan persamaan 2.31.
4. Hitung elemen-elemen matriks *Jacobian* J_1,J_2, J_3 dan J_4 dengan persamaan 2.24 sampai dengan persamaan 2.32.
5. Hitung harga-harga $\Delta\delta_i^{(k)}$ dengan persamaan 2.24.
6. Hitung harga-harga baru dari sudut fasa dan tegangan $\delta_i^{(k+1)}$ dan $\left|V_i^{(k+1)}\right|$ dengan persamaan 2.35 dan persamaan 2.36.
7. Proses ini berlangsung sampai :

$$\left|\Delta P_i^{(k)}\right| \le \varepsilon \quad (2.37)$$

$$\left|\Delta Q_i^{(k)}\right| \le \varepsilon \quad (2.38)$$

BAB III
ARTIFICIAL BEE COLONY (ABC) *ALGORITHM*

3.1 Pendahuluan

Kecerdasan *swarm* telah menjadi minat penelitian oleh banyak ilmuwan dari berbagai bidang dalam dekade terakhir ini. Bonabeau telah mendefinisikan kecerdasan *swarm* sebagai setiap upaya untuk merancang algoritma atau perangkat pemecahan masalah terdistribusi yang terinspirasi oleh perilaku kolektif dari koloni serangga sosial dan kumpulan hewan lain [9]. Istilah *swarm* digunakan secara umum untuk mengacu pada kumpulan terbatas dari setiap interaksi agen atau individu. Salah satu contoh dari *swarm* adalah kawanan lebah yang mengerubungi sarang mereka.

Gambar 3.1 Contoh kawanan lebah

Saat ini telah berkembang suatu metode berdasarkan kecerdasan buatan yaitu *Artificial Bee Colony* (ABC) dan beberapa aplikasi yang menggunakan metode ini. Haiyan Quan dan Xinling Shi mendiskusikan berbagai macam permasalahan *Artificial Bee Colony* (ABC). Perbaikan dalam *Artificial Bee Colony* (ABC) disarankan dengan menganalisis beberapa fungsi oleh Haiyan Quan dan Xinling Shi [10]. Li-Pei, Wong Malcolm Yoke Hean Low dan Chin Soon Chong melakukan penelitian terhadap suatu kegiatan menggunakan pendekatan *Artificial Bee Colony* (ABC) dengan menyelesaikan suatu permasalahan perjalanan seorang sales berdasarkan pada analisis *Artificial Bee Colony* (ABC) *Algorithm* [11]. Pengembangan suatu desain filter IIR (*Infinite Input Respon*) oleh Nurhan Karaboga, dengan *Artificial Bee Colony* (ABC) *Algorithm* menjadi dasar pada pengembangan performansi dari filter IIR agar lebih optimal [12].

Pada penelitian Tugas Akhir ini, *Artificial Bee Colony* (ABC) *Algorithm* diusulkan untuk menyelesaikan permasalahan penentuan letak dan kapasitas kapasitor secara optimal pada jaring transmisi. *Artificial Bee Colony* (ABC) *Algorithm* merupakan kecerdasan buatan yang menirukan perilaku koloni lebah dalam mencari sumber *nectar* (sari bunga). Kemampuan koloni lebah ini dapat digunakan untuk mengenali suatu sistem transmisi interkoneksi, kemudian mempelajari dan membantu menyelesaikan permasalahan kompensasi daya reaktif sehingga dapat meningkatkan kestabilan tegangan dan mengurangi kerugian daya pada jaring transmisi.

Koloni lebah dalam menentukan sumber makanan terbagi menjadi tiga kelompok yaitu lebah pekerja, lebah penjelajah dan lebah pengintai. Lebah-lebah ini melakukan suatu fungsi untuk menentukan letak dan kapasitas suatu sumber *nectar* kemudian mengingat dan membandingkan dengan sumber lain. Pada akhir fungsi, dipilih suatu lokasi dengan sumber nektar yang paling optimal.

3.2 Pencarian Madu oleh Serangga Lebah [13]

Setiap sumber bunga memiliki keragaman dan perbedaan dalam hal kualitas, sehingga seseorang mungkin memiliki pemikiran bahwa keputusan tentang jumlah lebah yang harus dikirim ke setiap tempat sumber bunga dan lama lebah di sumber tersebut merupakan sebuah permasalahan dalam sebuah koloni yang ingin mencapai laju pengumpulan madu bunga (nektar) sebanyak mungkin. Berkat sistem kerja kawanan lebah yang sangat baik maka lebah mampu memecahkan permasalahan ini tanpa mengalami kesulitan.

Kawanan lebah yang berada di dalam sebuah sarang ada yang bertugas sebagai pengumpul nektar. Mereka memiliki tugas untuk berkeliling di antara bunga-bunga dan mengumpulkan nektar sebanyak mungkin. Ketika kembali ke sarang, mereka menyerahkan semua nektar yang mereka bawa kepada lebah-lebah yang bertugas menjaga sarang dan menyimpan bahan makanan. Lebah-lebah penyimpan ini kemudian menyimpan nektar di dalam sel-sel madu. Seekor lebah pengumpul nektar juga dibantu oleh lebah pengumpul *nectar* lain dalam menentukan kualitas sumber bunga yang ditemukannya. Lebah pengumpul nektar tersebut mengamati dan menunggu untuk bertemu dengan seekor lebah penyimpan makanan yang telah siap menerima nektar. Saat menunggu, lebah memiliki waktu tungggu yang berlangsung lama, sehingga sang lebah pengumpul nektar tersebut memahami hal ini adalah sebagai isyarat

bahwa sumber bunga yang dia temukan bukan dari mutu yang baik dan sumber bunga yang terbaik telah ditemukan oleh lebah-lebah yang lain. Sebaliknya, jika lebah pengumpul makanan disambut oleh sejumlah besar lebah-lebah penyimpan makanan maka semakin besar kemungkinan bahwa muatan nektar yang dia temukan memiliki kualitas yang baik.

Lebah yang memperoleh informasi ini memutuskan apakah sumber bunganya senilai dengan kerja keras yang akan dilakukan pada tingkat selanjutnya. Jika ya, maka lebah ini melakukan tarian getarnya agar dipahami oleh lebah-lebah lain. Lama tarian ini memperlihatkan seberapa besar keuntungan yang mungkin dapat diperoleh dari sumber bunga ini.

3.2.1 Penyimpanan Maksimum dengan Bahan Minimum

Sarang lebah dibangun dari dinding lilin lebah sarang madu dan disarang tersebut terdapat ratusan sel-sel kecil di tiap-tiap permukaan. Semua sel sarang lebah madu mempunyai ukuran yang sama tepat. Keajaiban teknik ini didapat dari kerja sama ribuan lebah. Lebah menggunakan sel tersebut untuk meletakkan makanan dan mengatur lebah muda.

Lebah madu menggunakan struktur heksagonal sebagai konstruksi pembangunan sarang mereka. Para ahli matematika memberikan alasan bahwa struktur heksagonal memiliki bentuk geometrik yang paling cocok untuk penggunaan maksimum dari suatu area. Jika sel sarang lebah dibentuk dengan bentuk lain seperti segiempat atau segitiga maka ada area yang tidak digunakan kemudian hanya sedikit madu yang akan disimpan dan semakin sedikit lebah yang mendapatkan manfaat dari penyimpaan madu tersebut.

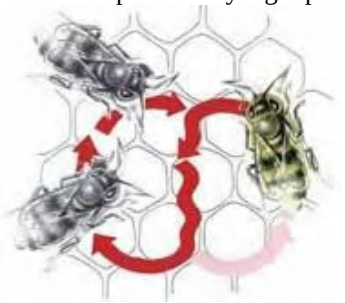
Untuk kedalaman sarang yang sama, sarang berbentuk sel segitiga atau segiempat akan menampung jumlah madu yang sama dengan sel heksagonal namun diantara semua bentuk geometrik, bentuk heksagonal mempunyai lingkaran yang paling dekat. Dengan demikian, jumlah lilin yang dibutuhkan untuk membangun sel sarang berbentuk heksagonal lebih sedikit daripada jumlah lilin yang dibutuhkan untuk sel segitiga atau segiempat. Sehingga dapat diambil kesimpulan bahwa sel-sel heksagonal membutuhkan jumlah minimal dari lilin dalam konstruksi ketika mereka menyimpan jumlah madu yang maksimal.

Selain dalam pembuatan diding sel, kawanan lebah ini juga membawa prinsip penyimpanan maksimum dalam pertimbangan ketika mereka membangun bagian bawah. Sarang dibuat dengan bentuk pipih

dengan dua baris sel yang saling membelakangi. Pada kasus ini, masalah dari titik pertemuan dari dua baris muncul. Membangun permukaan dasar sel-sel dengan mengkombinasikan tiga segiempat untuk menyelesaikan masalah ini. Ketika tiga sel dibangun di sisi sarang, permukaan dasar dari satu sel yang berada di sisi yang lain secara otomatis juga terbentuk. Karena permukaan bawah tersusun dari plat-plat lilin bujur sangkar maka bagian bawah sel-sel yang dibuat jadi bertambah dalam sehingga menambah volume sel dan juga jumlah madu yang disimpan.

3.2.2 Cara Lebah Membagi Informasi Sumber Makanan

Dalam pencarian sumber makanan, lebah menempuh jarak yang jauh dan menjelajahi area yang luas untuk menemukan sumber makanan. Lebah yang menemukan bunga kembali ke sarang untuk memberitahu letak sumber makanan yang telah ditemukan, lebah tersebut menari untuk mendeskripsikan lokasi bunga tersebut kepada lebah lain di dalam sarang. Tarian ini berarti ekspresi yang digunakan untuk menginformasikan kepada lebah yang lain tentang lokasi dari bunga. Tarian ini dilakukan berulang-ulang oleh lebah, termasuk untuk semua informasi mengenai kemiringan, arah, jarak dan detil lain dari sumber makanan yang membuat lebah lain mampu untuk mencapai lokasi yang diperoleh.



Gambar 3.2 Tarian lebah

Tarian ini membentuk angka “8” diulang-ulang secara konstan oleh lebah (**Gambar 3.2**). Lebah membentuk bagian tengah dari angka “8” dengan menggoyangkan ekornya dan berzig-zag. Sudut diantara zig-zag dan garis diantara matahari dan sarang memberi arah yang tepat dari sumber makanan.

3.2.3 Metode Penandaan Bunga

Lebah madu dapat mengetahui kalau bunga yang dia temui telah didatangi dan nektar yang ada telah diambil lebih dahulu oleh lebah lain maka dia segera meninggalkan lokasi tersebut dan langsung mencari lokasi yang lain. Dengan demikian, lebah ini dapat menghemat waktu dan tenaga. Hal ini terjadi karena lebah lain yang mendatangi letak bunga terlebih dahulu menandai dengan tetesan berbau khas. Begitu seekor lebah baru tiba mengunjungi bunga yang sama, dia mencium bau tersebut dan mengetahui bahwa bunga tersebut sudah tidak berguna dan kemudian langsung pergi ke bunga yang lain. Dengan begitu, lebah tidak akan membuang waktu dalam menemukan sumber makanan.

3.3 Pemakaian Konsep Lebah Untuk Optimisasi

Dua konsep dasar untuk kinerja kolektif *swarm* disajikan di bawah ini, yaitu organisasi diri (*self-organization*) dan pembagian kerja. Keduanya diperlukan sebagai properti untuk mendapatkan perilaku kecerdasan *swarm*, seperti halnya sistem pemecahan masalah terdistribusi (*distributed problem-solving*), yang mengatur dirinya sendiri dan beradaptasi dengan lingkungan tertentu.

1. Organisasi diri (*self-organization*) dapat didefinisikan sebagai seperangkat mekanisme dinamis, yang menghasilkan struktur di tingkat global dari sistem melalui interaksi diantara komponen tingkat rendah. Mekanisme ini menetapkan aturan dasar untuk interaksi antar komponen-komponen sistem. Aturan tersebut memastikan bahwa interaksi dijalankan atas dasar informasi lokal murni tanpa ada hubungannya dengan pola global. Bonabeau telah menandai empat sifat dasar organisasi yang mengandalkan diri sendiri, yaitu umpan balik positif (*positive feedback*), umpan balik negatif (*negative feedback*), fluktuasi, dan *multiple interactions* [20].
    i) Umpan balik positif (*positive feedback*) adalah perilaku sederhana dari "*rules of thumb*" yang mendorong terciptanya struktur yang sesuai. Rekrutmen dan penguatan, seperti halnya proses mengikuti jejak dalam beberapa spesies semut atau tarian pada spesies lebah, dapat ditunjukkan sebagai contoh dari umpan balik positif.
    ii) Umpan balik negatif (*negative feedback*) merupakan lawan dari umpan balik positif yang membantu untuk menstabilkan pola kolektif. Untuk menghindari kejenuhan

yang mungkin terjadi, berkenaan dengan jumlah pencari makan yang tersedia, sumber makanan yang telah habis, serta kerumunan atau kompetisi pada sumber makanan, maka mekanisme umpan balik negatif diperlukan.

iii) Fluktuasi, seperti perjalanan acak, error, pergantian tugas secara acak di antara kawanan individu, sangat penting untuk kreativitas dan inovasi. Keadaan tidak teratur (*randomness*) sering kali penting untuk struktur darurat karena memungkinkan penemuan solusi baru.

iv) Secara umum, organisasi diri (*self-organization*) memerlukan kerapatan (*density*) minimal individu yang saling toleran, memungkinkan mereka untuk memanfaatkan hasil dari aktivitasnya sendiri, serta aktivitas dari individu yang lain.

2. Di dalam *swarm*, ada tugas berbeda yang dilakukan secara bersamaan oleh individu-individu khusus. Fenomena semacam ini disebut sebagai pembagian kerja. Performa tugas simultan melalui kerja sama di antara individu-individu khusus tersebut diyakini lebih efisien daripada performa tugas secara berurutan oleh individu tanpa spesialisasi [14]. Pembagian kerja juga memungkinkan *swarm* untuk merespon perubahan kondisi dalam ruang pencarian.

3.4 Pemodelan Perilaku Kawanan Lebah Madu

Model minimal dari seleksi mencari makan, yang mengarah pada munculnya kecerdasan kolektif kawanan lebah madu, terdiri dari tiga komponen penting, yaitu sumber makanan, lebah pekerja dan lebah *unemployed*. Model tersebut mendefinisikan dua modus perilaku yang paling penting, yakni rekrutmen ke sumber nektar dan ditinggalkannya sumber [14].

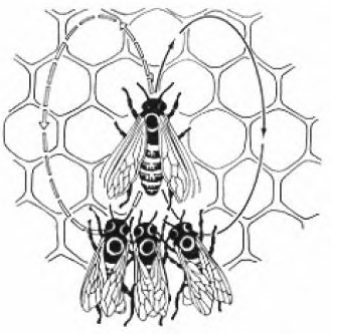
1. **Sumber makanan**

   Nilai sumber makanan tergantung pada banyak faktor, seperti jarak dekatnya ke sarang, kekayaan atau konsentrasi dari energi sumber makanan tersebut, dan tingkat kemudahan dalam pengambilan energi makanan. Untuk penyederhanaan, keuntungan (*profitability*) dari sumber makanan dapat diwakili dengan satu kuantitas [15].

2. **Lebah pekerja (*employed*)**
   Mereka dikaitkan dengan sumber makanan tertentu yang sedang mereka eksploitasi atau tempat mereka dipekerjakan. Mereka juga membawa informasi tentang letak dari sumber makanan, jarak dan arahnya dari sarang, *profitability* sumber makanan tersebut dan membagikan informasi ini dengan nilai *profitability* tertentu.
3. **Lebah *unemployed***
   Mereka secara terus-menerus keluar mencari sumber makanan untuk dieksploitasi. Ada dua jenis lebah *unemployed*, yaitu: lebah *scout*, yang bertugas mencari lingkungan di sekitar sarang untuk mendapatkan sumber makanan baru, dan lebah *onlooker*, yang bertugas menunggu di sarang dan mendapatkan sumber makanan melalui informasi yang dibagikan oleh lebah pekerja. Jumlah rata-rata lebah *scout* setara dengan sekitar 5 sampai 12% dari jumlah lebah keseluruhan [15].

Pertukaran informasi di antara lebah adalah kejadian yang paling penting dalam pembentukan pengetahuan kolektif. Saat memeriksa seluruh sarang, dimungkinkan untuk dapat membedakan beberapa bagian yang umumnya ada di semua sarang. Bagian paling penting dari sarang yang berkaitan dengan pertukaran informasi adalah *dancing area*. Komunikasi di antara lebah yang berkaitan dengan mutu sumber makanan terjadi di *dancing area*. Tarian lebah ini disebut dengan *waggle dance*.

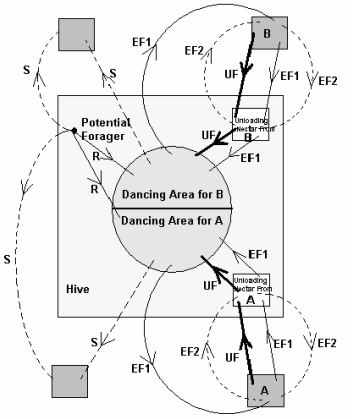
Karena informasi tentang semua sumber yang kaya makanan tersedia untuk lebah *onlooker* di *dancing area*, memungkinkan lebah tersebut untuk dapat menonton berbagai tarian lebah dan memutuskan untuk mempekerjakan dirinya pada sumber yang paling menguntungkan.



Gambar 3.3 *Waggle dance* dari lebah madu

Ada peluang yang lebih besar bagi lebah *onlooker* untuk memilih sumber-sumber makanan yang lebih menguntungkan, karena lebih banyak informasi yang beredar tentang sumber-sumber makanan yang lebih menguntungkan tersebut. Lebah pekerja membagi informasinya sesuai dengan probabilitas yang sebanding dengan *profitability* dari sumber makanan, dan membagi informasi ini melalui *waggle dance* dengan durasi yang lebih lama. Oleh karena itu, rekrutmen sebanding dengan *profitability* dari sumber makanan [15].

Supaya dapat memahami karakteristik perilaku dasar dari pencarian makan lebah dengan lebih baik, dapat dilihat pada Gambar 3.4.



Gambar 3.4 Perilaku pencarian makan lebah madu [9]

Diasumsikan bahwa ada dua sumber makanan yang ditemukan, yaitu A dan B. Pada mulanya, lebah pencari makan yang potensial akan mulai sebagai lebah *unemployed*. Lebah tersebut tidak memiliki pengetahuan tentang sumber makanan di sekitar sarang. Ada dua opsi pilihan untuk lebah tersebut, yaitu :

1. Bisa menjadi *scout* dan mulai mencari-cari sumber makanan di sekitar sarang secara spontan karena motivasi internal atau petunjuk eksternal yang mungkin (S pada Gambar 3.4).
2. Bisa menjadi rekrutan setelah menonton *waggle dance* dan mulai mencari sumber makanan (R pada Gambar 3.4).

Setelah menemukan sumber makanan, lebah tersebut menggunakan kemampuannya sendiri untuk mengingat lokasi sumber makanan dan kemudian mulai mengeksploitasinya dengan segera. Oleh karena itu, lebah tersebut akan menjadi "lebah pekerja". Lebah pekerja mengambil nektar dari sumber, lalu kembali ke sarang dan membongkar nektar pada tempat persediaan makanan. Setelah pembongkaran makanan, lebah pekerja tersebut memiliki tiga opsi sebagai berikut :

1. Menjadi pengikut tidak terikat (*uncommitted follower*) setelah meninggalkan sumber makanan (UF).
2. Melakukan *waggle dance* dan kemudian merekrut lebah lainnya sebelum kembali ke sumber makanan yang sama (EF1).
3. Meneruskan untuk mencari makan di sumber makanan semula tanpa merekrut lebah lainnya (EF2).

Penting untuk dicatat bahwa tidak semua lebah mulai mencari makan secara bersamaan. Percobaan yang telah dilakukan menegaskan bahwa lebah yang baru mulai mencari makan pada tingkat yang sebanding dengan perbedaan antara jumlah akhir lebah dan jumlah lebah yang sedang mencari makan.

3.5 *Artificial Bee Colony* (ABC) *Algorithm*

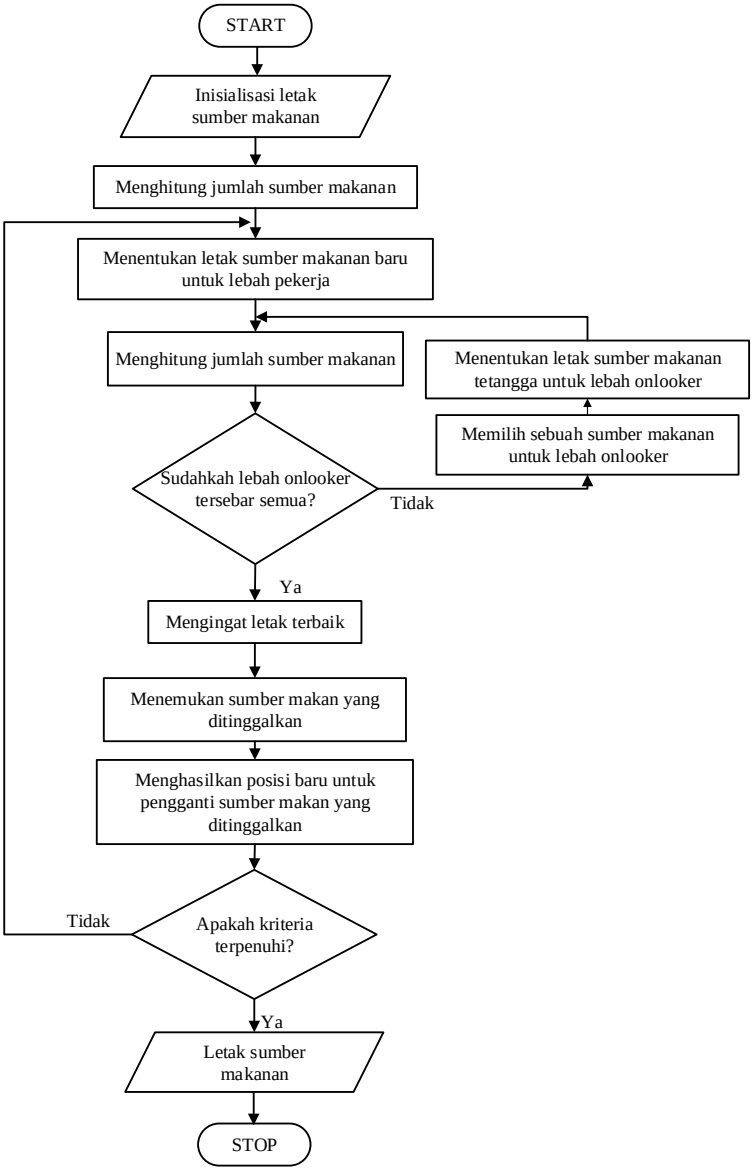
Artificial Bee Colony (ABC) *Algorithm* dikemukakan oleh karaboga untuk perhitungan numerik [11]. Dalam metode ini, perilaku cerdas tertentu dari sekawanan lebah madu berupa perilaku mencari makan ditinjau, dan sebuah algoritma baru dari koloni lebah buatan (*Artificial Bee Colony*) yang mensimulasikan perilaku lebah madu tersebut dijelaskan untuk memecahkan permasalahan optimisasi multidimensi dan multimodal. Dalam model ABC *algorithm*, koloni lebah

buatan terdiri dari tiga kelompok lebah, yaitu lebah pekerja, lebah *onlooker* dan lebah *scout*. Lebah yang menunggu di *dance area* untuk membuat keputusan dalam memilih sumber makanan, disebut sebagai lebah *onlooker*. Lebah yang pergi ke sumber makanan yang pernah dikunjungi sendiri sebelumnya diberi nama lebah pekerja. Sedangkan lebah yang melakukan pencarian secara acak disebut lebah *scout*. Separuh bagian pertama dari koloni terdiri dari lebah pekerja dan separuh bagian kedua mencakup lebah *onlooker*. Untuk setiap sumber makanan, hanya ada satu lebah pekerja. Dengan kata lain, jumlah lebah pekerja sama dengan jumlah sumber makanan di sekitar sarang. Lebah pekerja yang sumber makanannya telah habis akan menjadi lebah *scout*. Langkah-langkah utama dari ABC *Algorithm* dijelaskan di bawah ini :

1. Kirim lebah *scout* ke sumber makanan awal

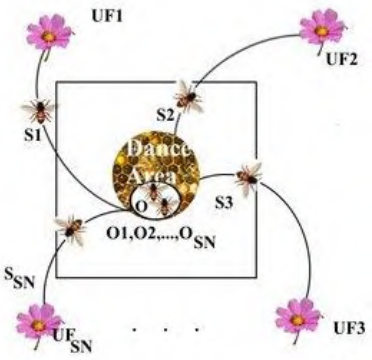
2. REPEAT
   Kirim lebah pekerja ke sumber makanan dan tentukan jumlah nektar (*Employed Bees Phase*).
   Tempatkan lebah *onlooker* pada sumber makanan berdasarkan jumlah nektarnya (*Onlooker Bees Phase*).
   Hentikan proses eksploitasi sumber-sumber makanan yang ditinggalkan oleh lebah pekerja.
   Kirim lebah *scout* ke daerah pencarian untuk menemukan sumber makanan baru secara acak (*Scout Bee Phase*).
   Mengingat sumber makanan terbaik yang telah ditemukan sejauh ini.

3. UNTIL (persyaratan terpenuhi)
   Untuk lebih mempermudah pemahaman tentang siklus dari koloni lebah ini dapat dilihat pada diagram alir pada Gambar 3.5 berikut.



Gambar 3.5 Diagram alir algoritma ABC

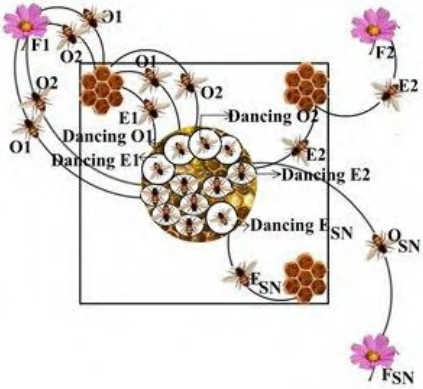
Setiap siklus (*cycle*) pencarian terdiri dari tiga langkah, yaitu mengirimkan lebah pekerja ke sumber makanan dan kemudian menghitung jumlah nektarnya, memilih sumber makanan untuk lebah *onlooker*, setelah informasi dibagikan oleh lebah pekerja, dan menentukan jumlah nektar dari sumber makanan tersebut, dan menentukan lebah *scout*, kemudian mengirimkan mereka ke sumber-sumber makanan lain yang mungkin [13]. Pada tahap inisialisasi, satu set posisi sumber makanan dipilih secara acak oleh lebah dan jumlah nektarnya ditentukan. Kemudian, lebah tersebut kembali ke sarang dan membagi informasi jumlah nektar dari sumber-sumber makanan tersebut dengan lebah *onlooker* yang sedang menunggu di *dance area* di dalam sarang. Setelah membagikan informasi, setiap lebah pekerja pergi ke daerah sumber makanan yang telah dikunjunginya sendiri pada siklus sebelumnya, karena sumber makanan tersebut ada dalam ingatannya, dan kemudian memilih sumber makanan yang baru melalui informasi visual di daerah sekitar sumber makanan yang sekarang.



Gambar 3.6 Tahap inisialisasi

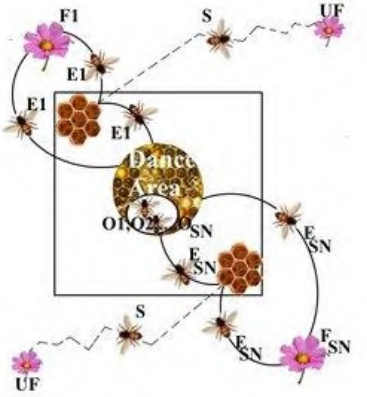
Pada tahap kedua, lebah *onlooker* memilih daerah sumber makanan berdasarkan informasi jumlah nektar yang didistribusikan oleh lebah pekerja di *dance area*. Ketika jumlah nektar sumber makanan meningkat, maka probabilitas sumber makanan tersebut akan dipilih oleh lebah *onlooker* juga meningkat. Setelah tiba di area yang dipilih, lebah *onlooker* memilih sumber makanan baru di lingkungan sekitar sumber makanan yang ada di memorinya sekarang tergantung pada informasi

visual, seperti dalam kasus lebah pekerja. Informasi visual tersebut didasarkan pada proses perbandingan posisi sumber makanan [15].



Gambar 3.7 Tahap lebah *onlooker*

Pada tahap ketiga, ketika nektar dari sebuah sumber makanan ditinggalkan oleh lebah pekerja, maka sumber makanan baru secara acak ditentukan oleh lebah *scout* untuk menggantikan sumber makanan yang telah ditinggalkan. Dalam pemodelan metode ini, untuk tiap *cycle*, paling banyak satu lebah *scout* pergi ke luar untuk mencari sumber makanan baru dan jumlah lebah pekerja sama dengan jumlah lebah *onlooker* [15].



Gambar 3.8 Tahap lebah pekerja dan lebah *scout*

Posisi sumber makanan melambangkan solusi yang mungkin dari masalah optimasi dan jumlah nektar sumber makanan dapat disamakan dengan kualitas (*fitness*) dari solusi yang terkait. Jumlah lebah pekerja atau lebah *onlooker* sama dengan jumlah solusi dalam populasi. Lebah *onlooker* ditempatkan pada sumber-sumber makanan dengan menggunakan metode *roulette wheel selection* [15]. Setiap koloni lebah mempunyai lebah *scout* yang merupakan penjelajah koloni [14]. Sebagai penjelajah, lebah *scout* tidak punya panduan saat mencari makanan. Mereka berhubungan dengan awal mula penemuan segala jenis sumber makanan. Sebagai akibat dari perilaku seperti itu, para lebah *scout* dicirikan dengan biaya pencarian yang rendah dan nilai rata-rata kualitas sumber makanan yang rendah. Terkadang, lebah *scout* dapat secara tidak sengaja menemukan sumber yang kaya makanan, yang sebelumnya sama sekali tidak dikenal. Dalam kasus lebah buatan (*Artificial Bee*), lebah *scout* buatan dapat ditugaskan untuk melakukan penemuan yang cepat dari kelompok solusi yang mungkin. Dalam metode ini, salah satu lebah pekerja diseleksi dan diklasifikasikan sebagai lebah *scout*. Proses seleksi dikendalikan oleh kontrol parameter yang disebut limit. Jika solusi yang melambangkan sumber makanan ini tidak dapat ditingkatkan melalui sejumlah percobaan (*trial*) yang telah ditetapkan, maka sumber makanan tersebut akan ditinggalkan oleh lebah pekerjanya dan lebah pekerja tersebut berubah menjadi lebah *scout*. Sejumlah percobaan untuk melepas sebuah sumber makanan tersebut sama dengan nilai dari "limit", yang merupakan parameter kontrol penting bagi ABC.

Pada proses pencarian yang sulit, proses eksplorasi dan eksploitasi harus dilakukan secara bersama-sama. Pada ABC *algorithm*, saat lebah *onlooker* dan lebah pekerja melakukan proses eksploitasi di ruang pencarian, lebah *scout* mengontrol proses eksplorasi. Dalam kasus lebah riil, tingkat rekrutmen melambangkan sebuah "ukuran" dari seberapa cepat kawanan lebah bisa menemukan dan mengeksploitasi sumber makanan yang baru diketahui. Proses rekrutmen buatan juga bisa melambangkan pengukuran dari kecepatan di mana solusi yang mungkin atau solusi optimal dari permasalahan optimisasi yang sulit dapat ditemukan. Kelangsungan hidup dan kemajuan kawanan lebah riil, tergantung pada penemuan yang cepat dan pemanfaatan yang efisien dari sumber-sumber makanan terbaik. Demikian juga solusi optimal dari permasalahan teknik yang sulit, berhubungan dengan penemuan yang relatif cepat dari "solusi yang baik", terutama untuk permasalahan yang perlu diselesaikan secara *real time*.

Secara lengkap metode ABC dapat dijelaskan sebagai berikut. Pada langkah pertama, ABC menghasilkan populasi awal yang didistribusikan secara random $P(C = 0)$ dari solusi *SN* (posisi sumber makanan), di mana *SN* menunjukkan ukuran populasi. Tiap solusi (sumber makanan) x_i ($i = 1, 2, \ldots, SN$) adalah sebuah vektor dimensi-*D*. Di sini, *D* adalah jumlah parameter optimisasi. Setelah inisialisasi, populasi dari posisi sumber makanan (solusi) dikenakan siklus yang berulang, $C = 1, 2, \ldots, MCN$, untuk proses pencarian lebah pekerja, lebah *onlooker* dan lebah *scout*. Lebah pekerja atau lebah *onlooker* buatan (*artificial*) secara probabilistik menghasilkan modifikasi posisi dari sumber makanan (solusi) dalam memorinya untuk menemukan sumber makanan baru dan mengetes jumlah nektar (*fitness value*) dari sumber makanan baru (solusi yang baru) tersebut. Dalam kasus lebah riil, produksi sumber makanan baru didasarkan pada proses perbandingan sumber makanan di suatu tempat, tergantung informasi yang dikumpulkan secara visual oleh lebah. Dalam pemodelan metode ini, produksi posisi sumber makanan baru juga didasarkan pada proses perbandingan posisi sumber makanan. Namun, lebah *artificial* tidak menggunakan informasi apa pun dalam perbandingan tersebut. Melainkan mereka secara acak memilih posisi sumber makanan dan menghasilkan modifikasi pada satu sumber makanan yang ada dalam memori mereka. Asalkan jumlah nektar sumber yang baru lebih tinggi dari yang sebelumnya, lebah akan mengingat posisi yang baru tersebut dan melupakan posisi yang lama. Jika tidak, dia tetap menyimpan posisi sebelumnya. Setelah semua lebah pekerja menyelesaikan proses pencarian, mereka membagi informasi nektar dari sumber-sumber makanan (solusi) dan informasi tentang posisi mereka dengan lebah *onlooker* di *dancing area*. Lebah *onlooker* mengevaluasi informasi yang diambil dari semua lebah pekerja dan memilih sumber makanan dengan probabilitas yang sesuai jumlah nektarnya. Seperti kasus lebah pekerja, lebah *onlooker* juga menghasilkan modifikasi pada posisi sumber makanan (solusi) dalam memorinya dan memeriksa jumlah nektar dari kandidat sumber makanan (solusi) yang baru. Asalkan nektarnya lebih tinggi dari sebelumnya, lebah akan mengingat posisi yang baru tersebut dan melupakan posisi yang lama.

Lebah *onlooker* memilih sumber makanan tergantung pada nilai probabilitas, p_i, yang dihitung melalui perumusan 3.1 berikut :

$$Pi = \frac{fit_i}{\sum_{i=1}^{s} fit_i} \tag{3.1}$$

dimana fit_i adalah nilai fitness dari solusi *i* yang dievaluasi oleh lebah pekerja, yang sebanding dengan jumlah nektar sumber makanan pada posisi *i* dan *SN* adalah jumlah sumber makanan yang sama dengan jumlah lebah pekerja (*BN*). Dengan cara ini, lebah pekerja menukar informasinya dengan lebah *onlooker*.

Untuk menghasilkan kandidat posisi makanan baru dari yang lama, ABC menggunakan rumusan 3.2 :

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - \varphi_{kj}) \tag{3.2}$$

dengan $k \in \{1, 2, \ldots, BN\}$ dan $j \in \{1, 2, \ldots, D\}$ adalah indeks yang dipilih secara random. Meskipun *k* ditentukan secara acak, namun harus berbeda dari *i*. $\varphi_{i,j}$ adalah nilai acak antara [0, 1], yang mengontrol produksi posisi sumber makanan 'tetangga' di sekitar $x_{i,j}$ dan modifikasi tersebut menyatakan perbandingan posisi makanan 'tetangga' secara visual oleh lebah. Persamaan 3.2 menunjukkan bahwa ketika perbedaan antara parameter $x_{i,j}$ dan $x_{k,j}$ berkurang, gangguan pada posisi $x_{i,j}$ juga berkurang. Oleh karena itu, ketika pencarian mendekati solusi optimal dalam ruang pencarian, panjang langkahnya secara adaptif dikurangi. Jika parameter yang dihasilkan oleh operasi ini melebihi batas yang telah ditentukan, maka parameter dapat diatur ke nilai batasnya.

Sumber makanan yang nektarnya ditinggalkan oleh lebah pekerja digantikan dengan sumber makanan baru oleh lebah *scout*. Jika sebuah posisi sumber makanan tidak dapat ditingkatkan lebih lanjut melalui sejumlah siklus (*cycle*) yang telah ditetapkan, atau yang disebut dengan *limit*, maka sumber makanan tersebut diasumsikan untuk ditinggalkan. Hal ini disimulasikan dengan menghasilkan posisi sumber makanan baru secara random untuk menggantikan posisi sumber makanan yang ditinggalkan. Diasumsikan bahwa sumber makanan yang ditinggalkan adalah x_i dan $j \in \{1, 2, \ldots, D\}$, maka lebah *scout* akan mencari sumber makanan baru untuk diganti dengan x_i. Operasi ini dilakukan dengan menggunakan persamaan 3.3 :

$$x_i^j = x_{\min}^j + rand[0,1](x_{\max}^j - x_{\min}^j) \tag{3.3}$$

Setelah masing-masing kandidat posisi sumber makanan $v_{i,j}$ diproduksi dan dievaluasi oleh lebah *artificial*, kinerjanya dibandingkan

dengan yang dari $x_{i,j.}$ Jika sumber makanan baru mempunyai nektar yang sama atau lebih baik daripada sumber yang lama, maka sumber yang lama tersebut akan digantikan dengan yang baru dalam memori. Jika tidak, yang lama dipertahankan. Dengan kata lain, mekanisme *greedy selection* digunakan sebagai operasi seleksi antara sumber makanan saat ini dan sumber makanan yang lama.

ABC *Algorithm* pada kenyataannya menggunakan empat proses seleksi yang berbeda, yaitu (1) proses seleksi global yang digunakan oleh lebah *onlooker artificial* untuk menemukan daerah yang menjanjikan seperti yang dijelaskan pada persamaan 3.1, (2) proses seleksi lokal yang dilakukan di suatu daerah oleh lebah pekerja dan lebah *onlooker artificial* berdasarkan informasi lokal (dalam kasus lebah riil, informasi ini termasuk warna, bentuk dan aroma bunga) untuk menentukan sumber makanan 'tetangga' di sekitar sumber yang ada dalam memori, sebagaimana didefinisikan dalam persamaan 3.2 (lebah tidak akan mampu mengidentifikasi jenis sumber nektar sampai mereka tiba di lokasi yang tepat dan membedakan sumber yang ada berdasarkan baunya), (3) proses seleksi lokal yang disebut dengan *greedy selection*, yang dilakukan oleh semua lebah, karena jika jumlah nektar dari kandidat sumber makanan baru lebih baik daripada sumber makanan yang ada sekarang, lebah akan melupakan posisi sumber makanan yang sekarang dan mengingat posisi kandidat sumber makanan baru. Jika tidak, lebah tetap menyimpan posisi sumber makanan yang sekarang dalam memori, (4) proses seleksi secara acak yang dilakukan oleh lebah *scout*.

Dengan memperhatikan penjelasan di atas, dapat diketahui bahwa terdapat tiga kontrol parameter yang digunakan dalam ABC dasar, yaitu jumlah sumber makanan, yang sama dengan jumlah lebah pekerja atau lebah *onlooker* (*SN*), nilai *limit*, dan jumlah *cycle* maksimum (*MCN*).

Halaman ini sengaja dikosongkan

BAB IV
SIMULASI DAN ANALISIS

4.1 Implementasi *Artificial Bee Colony* (ABC) *Algorithm* pada Proses Optimisasi Penempatan dan Kapasitas Kapasitor

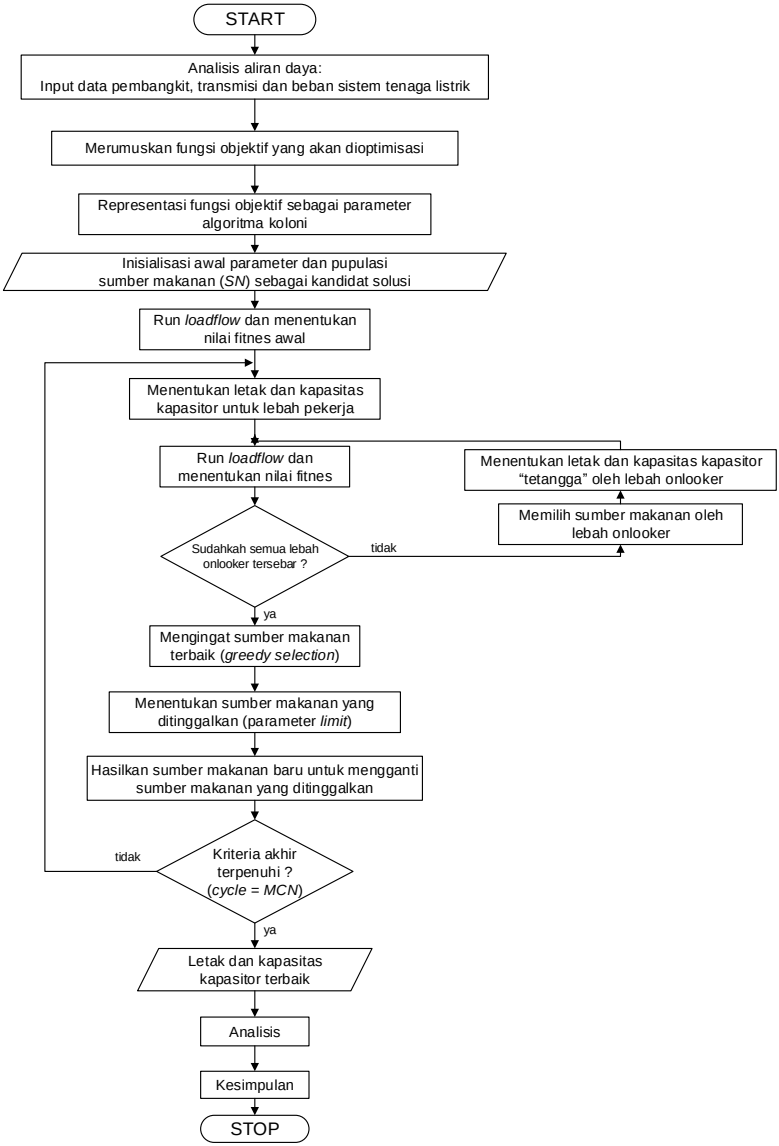
Sebelum proses optimisasi kapasitor dilakukan pada sistem transmisi Sumatera Utara 150 kV, maka parameter-parameter terkait yang ada pada proses optimisasi kapasitor harus direpresentasikan terlebih dahulu menjadi parameter-parameter ABC *Algorithm* sehingga pencarian secara acak oleh lebah dapat dilakukan.

Tabel 4.1 Representasi ABC *Algorithm* untuk optimisasi kapasitor

ABC *Algorithm*	Optimisasi kapasitor pada sistem transmisi
Jumlah lebah pekerja atau posisi sumber makanan	Kandidat bus sebagai posisi kapasitor dan kandidat kapasitas kapasitor yang akan dipasang
Dimensi	Jumlah kapasitor yang akan dipasang pada bus sistem transmisi
Fungsi obyektif	min $F = P_{loss}$
fitness	$\frac{1}{1+ fungsi\_objektif}$

Pada proses optimisasi, populasi lebah akan menentukan kandidat bus yang akan dipasang kapasitor dan kapasitas kapasitor terpasang untuk memperoleh nilai *fitness* yang mewakili nilai kerugian pada jaringan transmisi. Sedangkan dimensi mewakili jumlah kapasitor yang akan dipasang pada sistem tenaga listrik tersebut.

Kawanan lebah ini akan menyebar dan kemudian mencari sumber makanan secara acak. Setelah menemukan sumber makanan baru, lebah akan mengkalkulasi nektar dari setiap sumber makanan yang ditemukan. Hasil kalkulasi dari setiap sumber makanan yang ditemukan akan diseleksi dan diingat oleh lebah sehingga diperoleh sumber makanan terbaik. Hasil terbaik adalah kerugian daya aktif total minimum dari sekian banyak solusi yang dihasilkan pada saat pemasangan kapasitor. Proses optimisasi ini dapat ditunjukkan pada Gambar 4.1.



Gambar 4.1 Diagram alir implementasi ABC *Algorithm* untuk optimisasi kapasitor

4.2 Sistem Transmisi Sumatera Utara 150 kV

Data sistem transmisi Sumatera Utara 150 kV yang digunakan dalam Tugas Akhir ini adalah data tahun 2019 yang terdiri dari 29 bus, 46 saluran, dan 3 pusat pembangkit. Data-data saluran, beban dan generator diperlihatkan pada Tabel 4.2 dan Tabel 4.3.

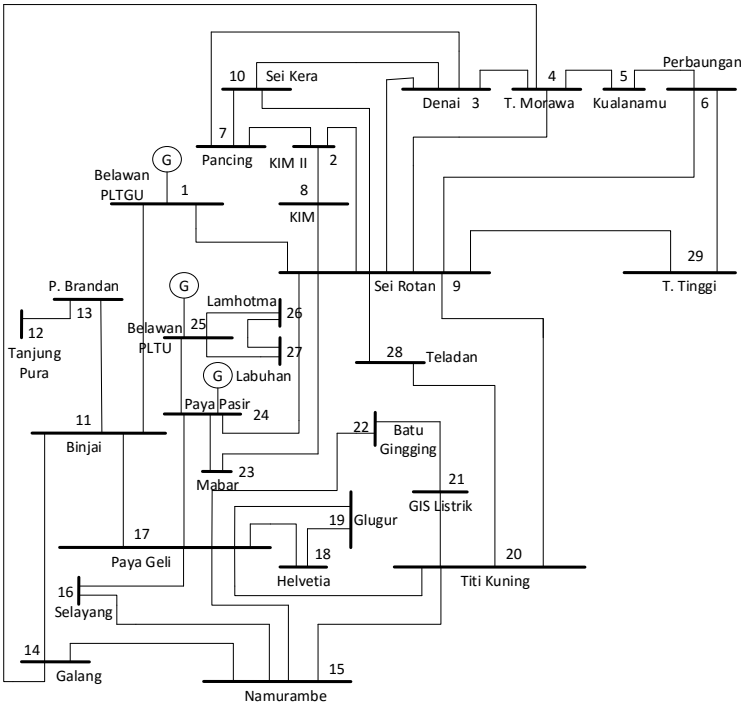
Penyelesaian analisis aliran daya dengan menggunakan metode *Newton-Raphson* didasarkan pada :

1. Base tegangan = 150 kV
2. Base daya = 1000 MVA
3. Akurasi = 0.0001
4. Akselerasi = 1.1
5. Maksimum iterasi = 50

Sedangkan bus-bus yang ada diklasifikasikan sebagai berikut :

a) *Slack* bus : Belawan PLTGU.
b) Bus generator : Paya Pasir dan Belawan PLTU.
c) Bus beban : Sei Kera, KIM II, Denai, T.Morawa, Kualanamu, Perbaungan, Pancing, KIM, Sei Rotan, T.Tinggi, Tanjung Pura, P. Brandan, Lamhotma, Labuhan, Binjai, Teladan, Mabar, Batu Gingging, GIS Listrik, Paya Geli, Selayang, Helvetia, Glugur, Titi Kuning, Galang, dan Namurambe.

S*ingle line diagram* sistem transmisi Sumatera Utara 150 kV dapat dilihat pada Gambar 4.2.



Gambar 4.2 *Single line diagram* sistem transmisi Sumatera Utara 150 kV

Tabel 4.2 Data beban dan generator pada sistem transmisi Sumatera Utara 150 kV

No. Bus	Nama Bus	Beban		Generator	
		P (MW)	Q (Mvar)	P (MW)	Q (Mvar)
1	Belawan PLTGU	0	0	-	-
2	KIM II	78,7	43,5	0	0
3	Denai	69,2	39,2	0	0
4	T.Morawa	69,2	39	0	0
5	Kualanamu	70,3	39,8	0	0
6	Perbaungan	38,1	20,7	0	0
7	Pancing	109,8	64	0	0
8	KIM	167,4	95,7	0	0
9	Sei Rotan	82,1	45,5	0	0
10	Sei Kera	59,9	32,1	0	0
11	Binjai	68,2	38	0	0
12	Tanjung Pura	46,6	27,3	0	0
13	P.Brandan	96,3	54,5	0	0
14	Galang	12,2	6,5	0	0
15	Namurambe	62,8	34,8	0	0
16	Selayang	95,2	54	0	0
17	Paya Geli	160,8	91,2	0	0
18	Helvetia	99,6	56,8	0	0
19	Glugur	68,1	36,6	0	0
20	Titi Kuning	98	54,9	0	0
21	GIS Listrik	119,2	69	0	0
22	Batu Gingging	160,2	92,8	0	0
23	Mabar	50,4	30,3	0	0
24	Paya Pasir	54,1	30,6	20	-
25	Belawan PLTU	0	0	339,2	-
26	Lamhotma	45,1	26,3	0	0

No. Bus	Nama Bus	Beban		Generator	
		P (MW)	Q (Mvar)	P (MW)	Q (Mvar)
27	Labuhan	42,2	24,4	0	0
28	Teladan	144,6	89,4	0	0
29	T.Tinggi	116,8	36,8	0	0

Data saluran pada sistem ditunjukkan pada Tabel 4.3 berikut ini :

Tabel 4.3 Data saluran transmisi Sumatera Utara 150 kV

Bus		R	X	1/2 B	Tap Setting
Dari	Ke	(pu)	(pu)	(p.u)	
1	9	0,02205	0,14798	0,05451	1
1	11	0,02881	0,19328	0,07120	1
2	7	0,01673	0,10751	0,04284	1
2	8	0,01683	0,08825	0,03242	1
2	9	0,08416	0,44128	0,01621	1
3	4	0,06578	0,20406	0,06937	1
3	7	0,02693	0,14121	0,05187	1
3	9	0,06749	0,20937	0,07117	1
4	5	0,02046	0,08033	0,02779	1
4	9	0,04578	0,14202	0,04828	1
4	14	0,13040	0,87476	0,03222	1
5	6	0,01469	0,05771	0,01996	1
6	9	0,21540	0,66819	0,02271	1
6	29	0,31777	0,98572	0,03351	1
8	9	0,10099	0,52954	0,01945	1
8	23	0,02959	0,13550	0,04788	1
9	20	0,02145	0,09826	0,03472	1
9	24	0,02959	0,13550	0,04788	1
9	29	0,31558	0,97895	0,03328	1
10	3	0,02693	0,14121	0,05187	1
10	7	0,02693	0,14121	0,05187	1
10	28	0,00748	0,03427	0,01211	1
11	12	0,03674	0,14428	0,04991	1
11	13	0,07468	0,29323	0,01014	1
11	14	0,07523	0,50467	0,01859	1
11	17	0,02046	0,08033	0,02779	1
12	13	0,03674	0,14428	0,04991	1
14	15	0,04012	0,26915	0,09915	1
15	16	0,01322	0,05194	0,01796	1

Bus		R	X	1/2 B	Tap Setting
Dari	Ke	(pu)	(pu)	(p.u)	
15	17	0,09265	0,33410	0,01165	1
15	20	0,06233	0,22478	0,07843	1
16	17	0,01322	0,05194	0,01796	1
17	18	0,00623	0,02856	0,01009	1
17	19	0,01487	0,06809	0,02406	1
17	20	0,15268	0,55058	0,01921	1
17	22	0,01469	0,05771	0,01996	1
17	24	0,02653	0,12151	0,04293	1
18	19	0,00623	0,02856	0,01009	1
20	21	0,01165	0,04576	0,01583	1
20	28	0,00623	0,02856	0,01009	1
21	22	0,00734	0,02885	0,09982	1
23	24	0,00739	0,03511	0,01197	1
24	25	0,00511	0,18468	0,01276	1
25	26	0,03539	0,10980	0,03733	1
25	27	0,01740	0,05398	0,01835	1
26	27	0,01888	0,05856	0,01991	1

4.3 Analisis Aliran Daya Sistem Transmisi Sumatera Utara 150 kV Sebelum Kompensasi

Untuk mengetahui kondisi awal dari sistem transmisi Sumatera Utara 150 kV maka dilakukan analisis aliran daya dengan menggunakan metode *Newton-Raphson*. Kondisi awal yang diperoleh akan dibandingkan dengan hasil analisis aliran daya setelah dilakukan kompensasi sehingga akan diketahui tingkat keberhasilan proses kompensasi dengan melihat total kerugian di saluran transmisi sebelum dan setelah kompensasi. Hasil analisis aliran daya ditunjukkan pada Tabel 4.4, dan kerugian daya pada masing-masing saluran direpresentasikan pada Table 4.5.

Tabel 4.4 Aliran daya sistem transmisi Sumut 150 kV sebelum pemasangan kapasitor

No. Bus	Tegangan (pu)	Sudut (derajat)	Beban		Pembangkitan	
			MW	MVar	MW	MVar
1	1,030	0.000	0	0	53,751	227,308
2	1,063	-12,079	78,7	43,5	0	0
3	1,090	-12,144	69,2	39,2	0	0
4	1,092	-11,877	69,2	39	0	0
5	1,093	-12,368	70,3	39,8	0	0
6	1,094	-12,474	38,1	20,7	0	0
7	1,080	-12,505	109,8	64	0	0
8	1,043	-11,691	167,4	95,7	0	0
9	1,069	-9,841	82,1	45,5	0	0
10	1,082	-12,423	59,9	32,1	0	0
11	1,073	-9,058	68,2	38	0	0
12	1,081	-9,905	46,6	27,3	0	0
13	1,078	-10,180	96,3	54,5	0	0
14	1,106	-11,451	12,2	6,5	0	0
15	1,081	-12,006	62,8	34,8	0	0
16	1,071	-11,835	95,2	54	0	0
17	1,062	-11,391	160,8	91,2	0	0
18	1,060	-11,568	99,6	56,8	0	0
19	1,061	-11,595	68,1	36,6	0	0
20	1,076	-11,997	98	54,9	0	0
21	1,072	-12,215	119,2	69	0	0

No. Bus	Tegangan (pu)	Sudut (derajat)	Beban		Pembangkitan	
			MW	MVar	MW	MVar
22	1,069	-12,129	160,2	92,8	0	0
23	1,009	-10,191	50,4	30,3	0	0
24	1,000	-9,705	54,1	30,6	20,000	-84,940
25	1,000	-9,188	0	0	339,200	-68,440
26	1,001	-9,435	45,1	26,3	0	0
27	1,000	-9,381	42,2	24,4	0	0
28	1,077	-12,307	144,6	89,4	0	0
29	1,083	-14,813	116,8	36,8	0	0
Total			**2285,1**	**1273,7**	**412,951**	**73,928**

Dari hasil analisis *load flow* dapat dilihat bahwa total daya aktif beban yaitu sebesar 2285,1 MW dan daya reaktif sebesar 1273,7 MW. Sedangkan pada data Tabel 4.5 diperoleh total kerugian daya aktif transmisi sebesar 106,437 MW dan total kerugian daya reaktif transmisi sebesar 3125,223 MVar.

Tabel 4.5 Kerugian daya pada saluran transmisi Sumatera Utara 150 kV sebelum kompensasi

Saluran		Kerugian Daya	
Dari	Ke	Aktif (MW)	Reaktif (MVar)
1	9	33,465	104,418
1	11	22,218	8,509
2	7	0,534	94,958
2	8	0,914	67,089
2	9	0,742	32,956
3	4	0,044	165,044
3	7	0,182	121,228
3	9	3,217	155,978
4	5	0,263	65,327
4	9	4,101	100,089
4	14	0,043	77,569
5	6	0,019	47,657
6	9	1,339	48,991
6	29	0,618	77,500
8	9	0,647	40,013

Saluran		Kerugian Daya	
Dari	Ke	Aktif (MW)	Reaktif (MVar)
8	23	2,875	87,678
9	20	3,541	63,676
9	24	7,424	68,644
9	29	2,657	68,872
10	3	0,124	121,691
10	7	0,006	121,228
10	28	0,166	27,465
11	12	0,517	113,788
11	13	0,376	21,978
11	14	0,910	38,060
11	17	6,025	39,691
12	13	0,064	116,015
14	15	0,397	234,558
15	16	0,573	39,329
15	17	0,393	25,336
15	20	0,033	182,355
16	17	0,667	38,211
17	18	0,094	22,290
17	19	0,049	53,961
17	20	0,153	43,342
17	22	0,977	41,458
17	24	8,122	54,128
18	19	0,002	22,683
20	21	0,165	35,871
20	28	0,259	22,198
21	22	0,116	228,269
23	24	0,959	19,765
24	25	0,331	23,245
25	26	0,052	74,569
25	27	0,063	36,522
26	27	0,005	39,858
Total		**106,437**	**3125,223**

4.4 Simulasi Penggunaan *Artificial Bee Colony* (ABC) *Algorithm* Pada Proses Kompensasi

Percobaan simulasi optimisasi algoritma *Artificial Bee Colony* (ABC) dalam proses kompensasi dilakukan pada sistem transmisi Sumatera Utara 150 kV. Penempatan kapasitor tidak dilakukan pada bus

generator dengan memiliki asumsi bahwa semua bus generator dianggap sudah mampu memenuhi kebutuhan daya reaktif beban di bus yang sama dan total kapasitas kapasitor yang terpasang pada sistem tidak dibatasi.

Untuk operasi yang efisien dan dapat diandalkan pada sebuah sistem tenaga, kontrol tegangan dan daya reaktif dengan memasang kapasitor shunt pada sistem tenaga listrik harus mencapai beberapa sasaran berikut :

1. Tegangan terminal semua peralatan dalam sistem berada pada batas yang dapat diterima.
2. Sistem harus aman beroperasi dalam jangka waktu yang lama.
3. Sistem harus diusahakan mampu mereduksi biaya pembangkitan listrik

Supaya sasaran-sasaran tersebut terpenuhi, maka performansi aliran daya pada sistem interkoneksi diusahakan memenuhi batasan batasan berikut ini :

1. Batasan tegangan yang diizinkan berada pada ± 5%.

   $V^{min} \leq V_i \leq V^{maks}$ untuk i=1,2,3…….N

   i = nomor bus

   V^{min} = 0.95 pu

   V^{maks} = 1.05 pu
2. Batasan operasi aman generator, suplai daya raktif generator harus harus berada pada range yang telah ditentukan.

   $Q^{min} \leq Q_i \leq Q^{maks}$ untuk i=1,2,3…….N
3. Fungsi obyektif yang digunakan untuk penentuan letak dan kapasitas kapasitor terpasang adalah:

   $F = \min \Sigma P_{loss}$

   $\min \Sigma P_{loss}$: Total kerugian daya aktif minimum pada saluran (MW)

   $$P_{loss} = \sum_{k=1}^{Nl} g_k [(t_k V_i)^2 + V_j^{\,2} - 2 t_k V_i V_j \cos\theta_{ij}] \qquad (4.1)$$

   g_k adalah konduktansi line *k* antara bus *i* dan *j*, t_k adalah tap ratio transformer line *k* dan *k* adalah nomor urutan line sesuai tabel 4.4.

4.4.1 Menentukan letak dan kapasitas lima kapasitor terpasang pada sistem

Percobaan ini merupakan percobaan untuk menentukan letak dan kapasitas kapasitor secara optimal menggunakan *Artificial Bee Colony* (ABC). Percobaan ini tidak hanya kapasitas saja yang dioptimisasi tetapi juga letak kapasitor itu sendiri karena letak atau lokasi penempatan kapasitor juga akan berpengaruh pada sistem terutama level tegangan masing-masing bus dan kerugian daya aktif pada tiap-tiap saluran pada sistem tenaga listrik.

Algoritma *Artificial Bee Colony* (ABC) yang sebagai metode optimisasi pada proses kompensasi yang disimulasikan menggunakan data parameter sebagai berikut,

a) *Colony size* : 50
b) *Maximum cycle* : 150
c) Dimensi : 5 (jumlah kapasitor yang akan dipasang)

Hasil yang diperoleh dari simulasi ABC yang dijalankan dapat dilihat data aliran daya setelah dilakukan kompensasi pada Tabel 4.6.

Tabel 4.6 Aliran daya sistem transmisi Sumatera Utara 150 kV setelah pemasangan lima kapasitor dengan ABC

No. Bus	Tegangan (pu)	Sudut (derajat)	Beban		Pembangkitan		Injeksi Kapasitor (MVar)
			MW	MVar	MW	MVar	
1	1,030	0,000	0	0	2024	1066	0
2	1,104	-11,622	78,7	43,5	0	0	0
3	1,130	-11,676	69,2	39,2	0	0	91,291
4	1,128	-11,406	69,2	39	0	0	0
5	1,132	-11,844	70,3	39,8	0	0	0
6	1,135	-11,961	38,1	20,7	0	0	71,937
7	1,122	-11,974	109,8	64	0	0	53,793
8	1,086	-11,316	167,4	95,7	0	0	0
9	1,099	-9,603	82,1	45,5	0	0	0
10	1,119	-11,855	59,9	32,1	0	0	0
11	1,091	-8,774	68,2	38	0	0	0
12	1,098	-9,490	46,6	27,3	0	0	0
13	1,095	-9,720	96,3	54,5	0	0	0
14	1,132	-10,937	12,2	6,5	0	0	0
15	1,112	-11,434	62,8	34,8	0	0	0
16	1,102	-11,294	95,2	54	0	0	0
17	1,093	-10,923	160,8	91,2	0	0	0
18	1,092	-11,072	99,6	56,8	0	0	0

No. Bus	Tegangan (pu)	Sudut (derajat)	Beban		Pembangkitan		Injeksi Kapasitor (MVar)
			MW	MVar	MW	MVar	
19	1,092	-11,095	68,1	36,6	0	0	0
20	1,109	-11,455	98	54,9	0	0	0
21	1,105	-11,629	119,2	69	0	0	0
22	1,101	-11,553	160,2	92,8	0	0	0
23	1,058	-10,074	50,4	30,3	0	0	0
24	1,050	-9,651	54,1	30,6	20	1101,7	0
25	1,050	-9,182	0	0	339,2	320,45	57,094
26	1,054	-9,457	45,1	26,3	0	0	0
27	1,055	-9,430	42,2	24,4	0	0	97,965
28	1,112	-11,732	144,6	89,4	0	0	0
29	1,120	-13,767	116,8	36,8	0	0	0
Total			**2285,1**	**1273**	**2383,31**	**2488,5**	**372,080**

Dari percobaan ini diperoleh kerugian daya pada saluran transmisi yang dapat dilihat pada Tabel 4.7.

Sebelum dilakukan kompensasi diketahui total kerugian daya aktif pada saluran transmisi adalah sebesar 106,437 MW, namun terjadi penurunan total kerugian daya pada saluran transmisi sebesar 98,211 MW setelah dilakukan kompensasi.

Tabel 4.7 Kerugian daya pada saluran transmisi setelah kompensasi

Saluran		Kerugian Daya	
Dari	Ke	Aktif (MW)	Reaktif (MVar)
1	9	35,989	117,822
1	11	22,634	8,419
2	7	0,527	102,8
2	8	0,731	73,959
2	9	0,638	36,006
3	4	0,045	176,719
3	7	0,121	130,910
3	9	3,574	165,776
4	5	0,276	69,927
4	9	4,243	106,633
4	14	0,017	82,185
5	6	0,058	51,102
6	9	1,493	52,087
6	29	0,448	83,813
8	9	0,429	44,213

Saluran		Kerugian Daya	
Dari	**Ke**	**Aktif (MW)**	**Reaktif (Mvar)**
8	23	2,083	100,552
9	20	2,908	71,352
9	24	3,738	93,531
9	29	2,061	75,539
10	3	0,175	130,231
10	7	0,022	130,140
10	28	0,299	28,772
11	12	0,402	118,021
11	13	0,281	23,126
11	14	0,997	39,253
11	17	5,010	46,608
12	13	0,048	119,882
14	15	0,271	247,798
15	16	0,511	41,991
15	17	0,345	27,080
15	20	0,008	193,414
16	17	0,567	41,029
17	18	0,071	23,762
17	19	0,036	57,276
17	20	0,169	45,977
17	22	0,873	44,625
17	24	4,160	79,575
18	19	0,002	24,055
20	21	0,153	38,201
20	28	0,280	23,614
21	22	0,135	242,383
23	24	0,753	23,142
24	25	0,299	26,080
25	26	0,105	82,283
25	27	0,223	39,945
26	27	0,004	44,237
Total		**98,211**	**3390,203**

Dengan menggunakan optimisasi ABC pada percobaan ini yang pada proses pemasangan kapasitor menentukan lokasi dan kapasitas kapasitor yang optimal diperoleh penurunan kerugian daya aktif sebesar 8,37% menjadi 98,211 MW.

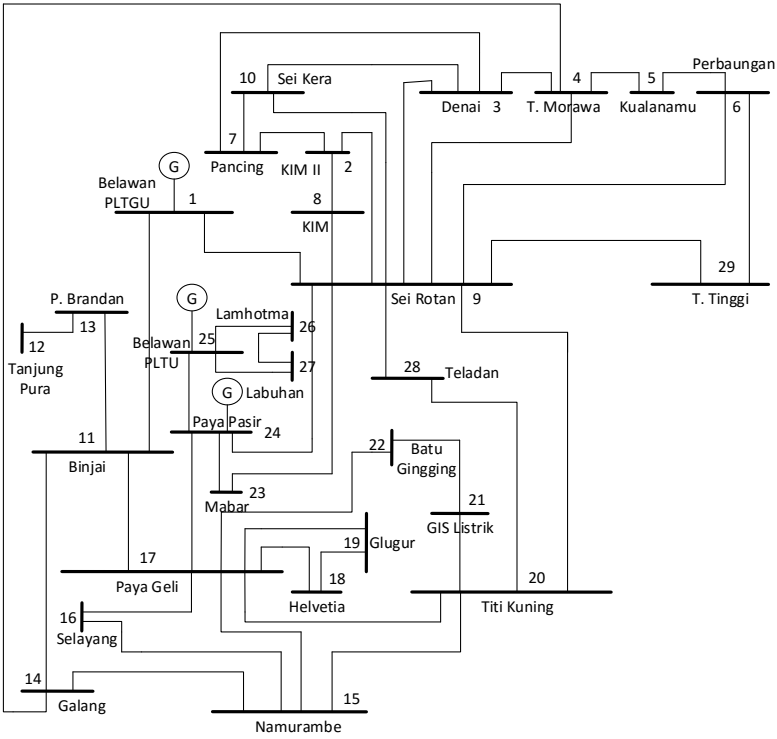
Tabel 4.8 Perbandingan kerugian daya pada saluran transmisi sebelum dan setelah dilakukan kompensasi

Saluran		Kerugian Daya Tanpa Kompensasi		Kerugian Daya Setelah Kompensasi	
Dari	**Ke**	**Aktif (MW**	**Reaktif (MVar)**	**Aktif (MW)**	**Reaktif (MVar)**
1	9	33,465	104,418	35,989	117,822
1	11	22,218	8,509	22,634	8,419
2	7	0,534	94,958	0,527	102,8
2	8	0,914	67,089	0,731	73,959
2	9	0,742	32,956	0,638	36,006
3	4	0,044	165,044	0,045	176,719
3	7	0,182	121,228	0,121	130,910
3	9	3,217	155,978	3,574	165,776
4	5	0,263	65,327	0,276	69,927
4	9	4,101	100,089	4,243	106,633
4	14	0,043	77,569	0,017	82,185
5	6	0,019	47,657	0,058	51,102
6	9	1,339	48,991	1,493	52,087
6	29	0,618	77,500	0,448	83,813
8	9	0,647	40,013	0,429	44,213
8	23	2,875	87,678	2,083	100,552
9	20	3,541	63,676	2,908	71,352
9	24	7,424	68,644	3,738	93,531
9	29	2,657	68,872	2,061	75,539
10	3	0,124	121,691	0,175	130,231
10	7	0,006	121,228	0,022	130,140
10	28	0,166	27,465	0,299	28,772
11	12	0,517	113,788	0,402	118,021
11	13	0,376	21,978	0,281	23,126
11	14	0,910	38,060	0,997	39,253
11	17	6,025	39,691	5,010	46,608
12	13	0,064	116,015	0,048	119,882
14	15	0,397	234,558	0,271	247,798
15	16	0,573	39,329	0,511	41,991
15	17	0,393	25,336	0,345	27,080
15	20	0,033	182,355	0,008	193,414
16	17	0,667	38,211	0,567	41,029
17	18	0,094	22,290	0,071	23,762
17	19	0,049	53,961	0,036	57,276
17	20	0,153	43,342	0,169	45,977

Saluran		Kerugian Daya Tanpa Kompensasi		Kerugian Daya Setelah Kompensasi	
Dari	**Ke**	**Aktif (MW**	**Reaktif (MVar)**	**Aktif (MW)**	**Reaktif (MVar)**
17	22	0,977	41,458	0,873	44,625
17	24	8,122	54,128	4,160	79,575
18	19	0,002	22,683	0,002	24,055
20	21	0,165	35,871	0,153	38,201
20	28	0,259	22,198	0,280	23,614
21	22	0,116	228,269	0,135	242,383
23	24	0,959	19,765	0,753	23,142
24	25	0,331	23,245	0,299	26,080
25	26	0,052	74,569	0,105	82,283
25	27	0,063	36,522	0,223	39,945
26	27	0,005	39,858	0,004	44,237
Total		**106,437**	**3125,223**	**98,211**	**3390,203**

LAMPIRAN

SINGLE LINE DIAGRAM
SISTEM KELISTRIKAN SUMATERA UTARA 150 KV 29 BUS



LISTING PROGRAM

```
clc
clear all
close all
basemva = 1000; accuracy = 0.0001; maxiter = 50; accel = 1.1;

%       Bus Bus  Voltage Angle   ---Load----  --------Generator-----
Injected
%       No  code Mag.   Degree  MW    Mvar       MW        Mvar     Qmin
Qmax      Mvar
busdata=[1   1    1.03    0.0      0.0    0.0      0.0        0.0        0     0       0
0         % Belawan PLTGU
        2   0    1.00    0.0     78.7    43.5     0.0        0.0        0     0       0
0         % KIM II
        3   0    1.00    0.0     69.2    39.2     0.0        0.0        0     0       0
0         % Denai
        4   0    1.00    0.0     69.2    39       0.0        0.0        0     0       0     0
% T.Morawa
        5   0    1.00    0.0     70.3    39.8     0.0        0.0        0     0       0
0         % Kualanamu
        6   0    1.00    0.0     38.1    20.7     0.0        0.0        0     0       0
0         % Perbaungan
        7   0    1.00    0.0    109.8    64       0.0        0.0        0     0       0
0         % Pancing
        8   0    1.00    0.0    167.4    95.7     0.0        0.0        0     0       0
0         % KIM
        9   0    1.00    0.0     82.1    45.5     0.0        0.0        0     0       0
0         % Sei Rotan
       10   0    1.00    0.0     59.9    32.1     0.0        0.0        0     0       0
0         % Sei Kera
       11   0    1.00    0.0     68.2    38       0.0        0.0        0     0       0
0         % Binjai
       12   0    1.00    0.0     46.6    27.3     0.0        0.0        0     0       0
0         % Tanjung Pura
       13   0    1.00    0.0     96.3    54.5     0.0        0.0        0     0       0
0         % P.Brandan
```

```
    14  0   1.00   0.0    12.2    6.5     0.0      0.0      0    0      0
0       % Galang
    15  0   1.00   0.0    62.8    34.8    0.0      0.0      0    0      0
0       % Namurambe
    16  0   1.00   0.0    95.2    54      0.0      0.0      0    0      0
0       % Selayang
    17  0   1.00   0.0    160.8   91.2    0.0      0.0      0    0      0
0       % Paya Geli
    18  0   1.00   0.0    99.6    56.8    0.0      0.0      0    0      0
0       % Helvetia
    19  0   1.00   0.0    68.1    36.6    0.0      0.0      0    0      0
0       % Glugur
    20  0   1.00   0.0    98      54.9    0.0      0.0      0    0      0
0       % Titi Kuning
    21  0   1.00   0.0    119.2   69      0.0      0.0      0    0      0
0       % GIS Listrik
    22  0   1.00   0.0    160.2   92.8    0.0      0.0      0    0      0
0       % Batu Gingging
    23  0   1.00   0.0    50.4    30.3    0.0      0.0      0    0      0
0       % Mabar
    24  2   1.00   0.0    54.1    30.6    20       0.0      0    20.349 0
0       % Paya Pasir
    25  2   1.00   0.0    0.0     0.0     339.2    0.0      0    298.336 0
0       % Belawan PLTU
    26  0   1.00   0.0    45.1    26.3    0.0      0.0      0    0      0
0       % Lamhotma
    27  0   1.00   0.0    42.2    24.4    0.0      0.0      0    0      0
0       % Labuhan
    28  0   1.00   0.0    144.6   89.4    0.0      0.0      0    0      0
0       % Teladan
    29  0   1.00   0.0    116.8   36.8    0.0      0.0      0    0      0
0];     % T.Tinggi
```

```
busGen=[1 24 25]
%       Linedata of Java Bali Transmission System
%       Bus   bus          R          X          1/2 B          = 1 for lines
%       nl    nr           p.u.       p.u.       p.u.           > 1 or < 1 tr. tap at
bus nl
linedata=[ 1    9        0.02205     0.14798     0.05451          1
           1   11        0.02881     0.19328     0.07120          1
           2    7        0.01673     0.10751     0.04284          1
           2    8        0.01683     0.08825     0.03242          1
           2    9        0.08416     0.44128     0.01621          1
           3    4        0.06578     0.20406     0.06937          1
           3    7        0.02693     0.14121     0.05187          1
           3    9        0.06749     0.20937     0.07117          1
           4    5        0.02046     0.08033     0.02779          1
           4    9        0.04578     0.14202     0.04828          1
           4   14        0.13040     0.87476     0.03222          1
           5    6        0.01469     0.05771     0.01996          1
           6    9        0.21540     0.66819     0.02271          1
           6   29        0.31777     0.98572     0.03351          1
           8    9        0.10099     0.52954     0.01945          1
           8   23        0.02959     0.13550     0.04788          1
           9   20        0.02145     0.09826     0.03472          1
           9   24        0.02959     0.13550     0.04788          1
           9   29        0.31558     0.97895     0.03328          1
          10    3        0.02693     0.14121     0.05187          1
          10    7        0.02693     0.14121     0.05187          1
          10   28        0.00748     0.03427     0.01211          1
          11   12        0.03674     0.14428     0.04991          1
          11   13        0.07468     0.29323     0.01014          1
          11   14        0.07523     0.50467     0.01859          1
          11   17        0.02046     0.08033     0.02779          1
          12   13        0.03674     0.14428     0.04991          1
          14   15        0.04012     0.26915     0.09915          1
          15   16        0.01322     0.05194     0.01796          1
          15   17        0.09265     0.33410     0.01165          1
          15   20        0.06233     0.22478     0.07843          1
          16   17        0.01322     0.05194     0.01796          1
          17   18        0.00623     0.02856     0.01009          1
          17   19        0.01487     0.06809     0.02406          1
```

```
        17   20      0.15268     0.55058     0.01921      1
        17   22      0.01469     0.05771     0.01996      1
        17   24      0.02653     0.12151     0.04293      1
        18   19      0.00623     0.02856     0.01009      1
        20   21      0.01165     0.04576     0.01583      1
        20   28      0.00623     0.02856     0.01009      1
        21   22      0.00734     0.02885     0.09982      1
        23   24      0.00739     0.03387     0.01197      1
        24   25      0.00511     0.03511     0.01276      1
        25   26      0.03539     0.10980     0.03733      1
        25   27      0.01740     0.05398     0.01835      1
        26   27      0.01888     0.05856     0.01991      1];
%=====================================================
======================
j=sqrt(-1); i = sqrt(-1);
nl = linedata(:,1); nr = linedata(:,2); R = linedata(:,3);
X = linedata(:,4); Bc = j*linedata(:,5); a = linedata(:, 6);
nbr=length(linedata(:,1)); nbus = max(max(nl), max(nr));
Z = R + j*X; y= ones(nbr,1)./Z;       %branch admittance
for n = 1:nbr
if a(n) <= 0  a(n) = 1; else end
Ybus=zeros(nbus,nbus);    % initialize Ybus to zero
          % formation of the off diagonal elements
for k=1:nbr;
     Ybus(nl(k),nr(k))=Ybus(nl(k),nr(k))-y(k)/a(k);
     Ybus(nr(k),nl(k))=Ybus(nl(k),nr(k));
  end
end
          % formation of the diagonal elements
for  n=1:nbus
   for k=1:nbr
      if nl(k)==n
      Ybus(n,n) = Ybus(n,n)+y(k)/(a(k)^2) + Bc(k);
      elseif nr(k)==n
      Ybus(n,n) = Ybus(n,n)+y(k) +Bc(k);
      else, end
   end
end
```

```
%====================================================
======================
% lfnewton
%   Power flow solution by Newton-Raphson method
%   Copyright (c) 1998 by  H. Saadat
ns=0; ng=0; Vm=0; delta=0; yload=0; deltad=0;
nbus = length(busdata(:,1));
for k=1:nbus
n=busdata(k,1);
kb(n)=busdata(k,2); Vm(n)=busdata(k,3); delta(n)=busdata(k, 4);
Pd(n)=busdata(k,5); Qd(n)=busdata(k,6); Pg(n)=busdata(k,7); Qg(n) =
busdata(k,8);
Qmin(n)=busdata(k, 9); Qmax(n)=busdata(k, 10);
Qsh(n)=busdata(k, 11);Qsvc(n)=busdata(k, 12);
   if Vm(n) <= 0  Vm(n) = 1.0; V(n) = 1 + j*0;
   else delta(n) = pi/180*delta(n);
       V(n) = Vm(n)*(cos(delta(n)) + j*sin(delta(n)));
       P(n)=(Pg(n)-Pd(n))/basemva;
       Q(n)=(Qg(n)-Qd(n)+Qsh(n)+Qsvc(n))/basemva;
       S(n) = P(n) + j*Q(n);
   end
end
%============================
for k=1:nbus
if kb(k) == 1, ns = ns+1; else, end
if kb(k) == 2 ng = ng+1; else, end
ngs(k) = ng;
nss(k) = ns;
end
Ym=abs(Ybus); t = angle(Ybus);
m=2*nbus-ng-2*ns;
maxerror = 1; converge=1;
iter = 0;
% Start of iterations
clear A  DC   J  DX
while maxerror >= accuracy  & iter <= maxiter & Vm<=1.05 % Test for
max. power mismatch
for i=1:m
for k=1:m
```

```
   A(i,k)=0;      %Initializing Jacobian matrix
end, end
iter = iter+1;
for n=1:nbus
nn=n-nss(n);
lm=nbus+n-ngs(n)-nss(n)-ns;
J11=0; J22=0; J33=0; J44=0;
  for i=1:nbr
    if nl(i) == n | nr(i) == n
      if nl(i) == n,  l = nr(i); end
      if nr(i) == n,  l = nl(i); end
      J11=J11+ Vm(n)*Vm(l)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));
      J33=J33+ Vm(n)*Vm(l)*Ym(n,l)*cos(t(n,l)- delta(n) + delta(l));
      if kb(n)~=1
      J22=J22+ Vm(l)*Ym(n,l)*cos(t(n,l)- delta(n) + delta(l));
      J44=J44+ Vm(l)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));
      else, end
      if kb(n) ~= 1  & kb(l) ~=1
      lk = nbus+l-ngs(l)-nss(l)-ns;
      ll = l -nss(l);
    % off diagonalelements of J1
      A(nn, ll) =-Vm(n)*Vm(l)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));
           if kb(l) == 0  % off diagonal elements of J2
           A(nn, lk) =Vm(n)*Ym(n,l)*cos(t(n,l)- delta(n) + delta(l));end
           if kb(n) == 0  % off diagonal elements of J3
           A(lm, ll) =-Vm(n)*Vm(l)*Ym(n,l)*cos(t(n,l)- delta(n)+delta(l));
end
           if kb(n) == 0 & kb(l) == 0  % off diagonal elements of  J4
           A(lm, lk) =-Vm(n)*Ym(n,l)*sin(t(n,l)- delta(n) + delta(l));end
      else end
    else  end
  end
  Pk = Vm(n)^2*Ym(n,n)*cos(t(n,n))+J33;
  Qk = -Vm(n)^2*Ym(n,n)*sin(t(n,n))-J11;
  if kb(n) == 1 P(n)=Pk; Q(n) = Qk; end % Swing bus P
   if kb(n) == 2  Q(n)=Qk;
      if Qmax(n) ~= 0
        Qgc = Q(n)*basemva + Qd(n) - Qsh(n) - Qsvc(n);
        if iter <= 7                % Between the 2th & 6th iterations
```

```
            if iter > 2                % the Mvar of generator buses are
              if Qgc  < Qmin(n),       % tested. If not within limits Vm(n)
              Vm(n) = Vm(n) + 0.01;    % is changed in steps of 0.01 pu to
              elseif Qgc  > Qmax(n),   % bring the generator Mvar within
              Vm(n) = Vm(n) - 0.01;end % the specified limits.
            else, end
          else,end
        else,end
      end
    if kb(n) ~= 1
      A(nn,nn) = J11;  %diagonal elements of J1
      DC(nn) = P(n)-Pk;
    end
    if kb(n) == 0
      A(nn,lm) = 2*Vm(n)*Ym(n,n)*cos(t(n,n))+J22;  %diagonal elements
of J2
      A(lm,nn)= J33;        %diagonal elements of J3
      A(lm,lm) =-2*Vm(n)*Ym(n,n)*sin(t(n,n))-J44;  %diagonal of
elements of J4
      DC(lm) = Q(n)-Qk;
    end
end
DX=A\DC';
for n=1:nbus
  nn=n-nss(n);
  lm=nbus+n-ngs(n)-nss(n)-ns;
    if kb(n) ~= 1
    delta(n) = delta(n)+DX(nn); end
    if kb(n) == 0
    Vm(n)=Vm(n)+DX(lm); end
 end
  maxerror=max(abs(DC));
     if iter == maxiter && maxerror > accuracy
   fprintf('\nWARNING: Iterative solution did not converged after ')
   fprintf('%g', iter), fprintf(' iterations.\n\n')
   fprintf('Press Enter to terminate the iterations and print the results \n')
   converge = 0; pause, else end

end
```

```
if converge ~= 1
  tech= ('                ITERATIVE SOLUTION DID NOT
CONVERGE'); else
  tech=('              Power Flow Solution by Newton-Raphson Method');
end
V = Vm.*cos(delta)+j*Vm.*sin(delta);
deltad=180/pi*delta;
i=sqrt(-1);
k=0;
for n = 1:nbus
    if kb(n) == 1
    k=k+1;
    S(n)= P(n)+j*Q(n);
    Pg(n) = P(n)*basemva + Pd(n);
    Qg(n) = Q(n)*basemva + Qd(n) - Qsh(n) - Qsvc(n);
    Pgg(k)=Pg(n);
    Qgg(k)=Qg(n);     %june 97
    elseif  kb(n) ==2
    k=k+1;
    S(n)=P(n)+j*Q(n);
    Qg(n) = Q(n)*basemva + Qd(n) - Qsh(n) - Qsvc(n);
    Pgg(k)=Pg(n);
    Qgg(k)=Qg(n);  % June 1997
 end
yload(n) = (Pd(n)- j*Qd(n)+j*Qsh(n)+j*Qsvc(n))/(basemva*Vm(n)^2);
end
busdata(:,3)=Vm'; busdata(:,4)=deltad';
Pgt = sum(Pg);  Qgt = sum(Qg); Pdt = sum(Pd); Qdt = sum(Qd); Qsht =
sum(Qsh); Qsvcht = sum(Qsvc);
%==================
% busout
disp(tech)
fprintf('                Maximum Power Mismatch = %g \n', maxerror)
fprintf('                    No. of Iterations = %g \n\n', iter)
head =['   Bus  Voltage  Angle    ------Load------    ---Generation---
Injected   Injected   '
     '   No.  Mag.    Degree    MW       Mvar      MW       Mvar      Mvar
SVC     '
```

```
    '                                                                    '];
disp(head)
for n=1:nbus
    fprintf(' %5g', n), fprintf(' %7.3f', Vm(n)),
    fprintf(' %8.3f', deltad(n)), fprintf(' %9.3f', Pd(n)),
    fprintf(' %9.3f', Qd(n)),  fprintf(' %9.3f', Pg(n)),
    fprintf(' %9.3f ', Qg(n)), fprintf(' %8.3f', Qsh(n)),fprintf(' %8.3f\n',
Qsvc(n))
end
   fprintf('     \n'), fprintf('    Total             ')
   fprintf(' %9.3f', Pdt), fprintf(' %9.3f', Qdt),
   fprintf(' %9.3f', Pgt), fprintf(' %9.3f', Qgt), fprintf(' %9.3f',
Qsht),fprintf(' %9.3f\n\n', Qsvcht),
%=================================================
======================
%lineflow
SLT = 0;
fprintf('\n')
fprintf('                        Line Flow and Losses \n\n')
fprintf('    --Line--  Power at bus & line flow    --Line loss--
Transformer\n')
fprintf('    from  to    MW      Mvar     MVA       MW      Mvar      tap\n')

for n = 1:nbus
busprt = 0;
  for L = 1:nbr;
     if busprt == 0
     fprintf('   \n'), fprintf('%6g', n), fprintf('      %9.3f', P(n)*basemva)
     fprintf('%9.3f', Q(n)*basemva), fprintf('%9.3f\n', abs(S(n)*basemva))

     busprt = 1;
     else, end
     if nl(L)==n      k = nr(L);
     In = (V(n) - a(L)*V(k))*y(L)/a(L)^2 + Bc(L)/a(L)^2*V(n);
     Ik = (V(k) - V(n)/a(L))*y(L) + Bc(L)*V(k);
     Snk = V(n)*conj(In)*basemva;
     Skn = V(k)*conj(Ik)*basemva;
     SL  = Snk + Skn;
     SLT = SLT + SL;
```

```
    elseif nr(L)==n  k = nl(L);
    In = (V(n) - V(k)/a(L))*y(L) + Bc(L)*V(n);
    Ik = (V(k) - a(L)*V(n))*y(L)/a(L)^2 + Bc(L)/a(L)^2*V(k);
    Snk = V(n)*conj(In)*basemva;
    Skn = V(k)*conj(Ik)*basemva;
    SL  = Snk + Skn;
    SLT = SLT + SL;
    else, end
     if nl(L)==n | nr(L)==n
     fprintf('%12g', k),
     fprintf('%9.3f', real(Snk)), fprintf('%9.3f', imag(Snk))
     fprintf('%9.3f', abs(Snk)),
     fprintf('%9.3f', real(SL)),
        if nl(L) ==n & a(L) ~= 1
        fprintf('%9.3f', imag(SL)), fprintf('%9.3f\n', a(L))
        else, fprintf('%9.3f\n', imag(SL))
        end
     else, end
 end
end
SLT = SLT/2;
fprintf('   \n'), fprintf('    Total loss                    ')
fprintf('%9.3f', real(SLT)), fprintf('%9.3f\n', imag(SLT))
```

PROGRAM *BEE COLONY ALGORITHM*

```
%/* ABC algorithm coded using MATLAB language */

%/* Artificial Bee Colony (ABC) is one of the most recently defined
algorithms by Dervis Karaboga in 2005, motivated by the intelligent
behavior of honey bees. */

%/* Reference Papers*/

%/*D. Karaboga, AN IDEA BASED ON HONEY BEE SWARM FOR
NUMERICAL OPTIMIZATION,TECHNICAL REPORT-TR06, Erciyes
University, Engineering Faculty, Computer Engineering Department
2005.*/
```

```
%/*D. Karaboga, B. Basturk, A powerful and Efficient Algorithm for
Numerical Function Optimization: Artificial Bee Colony (ABC)
Algorithm, Journal of Global Optimization, Volume:39, Issue:3,pp:459-
171, November 2007,ISSN:0925-5001 , doi: 10.1007/s10898-007-9149-x
*/

%/*D. Karaboga, B. Basturk, On The Performance Of Artificial Bee
Colony (ABC) Algorithm, Applied Soft Computing,Volume 8, Issue 1,
January 2008, Pages 687-697. */

%/*D. Karaboga, B. Akay, A Comparative Study of Artificial Bee
Colony Algorithm,  Applied Mathematics and Computation, 214, 108-
132, 2009. */

%/*Copyright © 2009 Erciyes University, Intelligent Systems Research
Group, The Dept. of Computer Engineering*/

%/*Contact:
%Dervis Karaboga (karaboga@erciyes.edu.tr )
%Bahriye Basturk Akay (bahriye@erciyes.edu.tr)
%*/
close all
clc
clear all
NewRaph_DITA;
D1=5;% jumlah kapasitor
D2=D1;%Bus
D=D1+D2;
NP=50 %/* The number of colony size (employed bees+onlooker bees)*/
FoodNumber=NP/2; %/*The number of food sources equals the half of
the colony size*/
limit=FoodNumber*D; %/*A food source which could not be improved
through "limit" trials is abandoned by its employed bee*/
maxCycle=150 %/*The number of cycles for foraging {a stopping
criteria}*/
runtime=1;%/*Algorithm can be run many times in order to see its
robustness*/
```

```
%Foods [FoodNumber][D]; /*Foods is the population of food sources. Each row of Foods matrix is a vector holding D parameters to be optimized. The number of rows of Foods matrix equals to the FoodNumber*/
%ObjVal[FoodNumber];  /*f is a vector holding objective function values associated with food sources */
%Fitness[FoodNumber]; /*fitness is a vector holding fitness (quality) values associated with food sources*/
%trial[FoodNumber]; /*trial is a vector holding trial numbers through which solutions can not be improved*/
%prob[FoodNumber]; /*prob is a vector holding probabilities of food sources (solutions) to be chosen*/
%solution [D]; /*New solution (neighbour) produced by v_{ij}=x_{ij}+\phi_{ij}*(x_{kj}-x_{ij}) j is a randomly chosen parameter and k is a randomlu chosen solution different from i*/
%ObjValSol; /*Objective function value of new solution*/
%FitnessSol; /*Fitness value of new solution*/
%neighbour, param2change; /*param2change corrresponds to j, neighbour corresponds to k in equation v_{ij}=x_{ij}+\phi_{ij}*(x_{kj}-x_{ij})*/
%GlobalMin; /*Optimum solution obtained by ABC algorithm*/
%GlobalParams[D]; /*Parameters of the optimum solution*/
%GlobalMins[runtime]; /*GlobalMins holds the GlobalMin of each run in multiple runs*/

GlobalMins=zeros(1,runtime);
for r=1:runtime
% /*All food sources are initialized */
%/*Variables are initialized in the range [lb,ub]. If each parameter has different range, use arrays lb[j], ub[j] instead of lb and ub */
ub=100;lb=50;
Range = repmat((ub-lb),[FoodNumber D1]);
Lower = repmat(lb, [FoodNumber D1]);
FoodsQ  = rand(FoodNumber,D1) .* Range + Lower  %random sizing kapasitor

% x=ran(busgen)
ubus=2; lbus=29;
% Range = repmat((ubus-lbus),[FoodNumber D2]);
```

```
% Lower = repmat(lbus, [FoodNumber D2]);
Range = lbus-ubus;
Lower = ubus;
for i=1:D2
   for j=1:FoodNumber
      a  = rand(1)* Range + Lower ;
      FoodsBus(j,i)=round(a) %random placement capacitor
      l=i-1;
      for k=1:l
        if FoodsBus(j,i)==FoodsBus(j,k)
              %disp('cek');
              FoodsBus(j,i)  = round(rand(1)* Range) + Lower ;
              %FoodsBus(j,i)= 0
              k=1;
        end
      end
   end

end
FoodsBus
for iabc=1:FoodNumber
   if iabc >1
      jabc=iabc-1;
   else
      jabc=iabc;
   end
   busdata(FoodsBus(jabc),12)=0;
   busdata(FoodsBus(iabc),12)=FoodsQ(iabc);
   total_loss2=loadflow(busdata,linedata,0);
   ObjVal(iabc)=total_loss2;                    %
ObjVal=feval(objfun,Foods);
   Fitness(iabc)=calculateFitness(ObjVal(iabc));
end
%reset trial counters
busdata(:,12)=0;
trial=zeros(1,FoodNumber);

%/*The best food source is memorized*/
BestInd=find(ObjVal==min(ObjVal));
```

```
BestInd=BestInd(end);
GlobalMin=ObjVal(BestInd);
GlobalParamsQ=FoodsQ(BestInd,:);
GlobalParamsBus=FoodsBus(BestInd,:);
inisialisasi=[GlobalParamsQ GlobalParamsBus]
busdata(GlobalParamsBus,12)=GlobalParamsQ;
total_loss2=loadflow(busdata,linedata,0);

%=================================================
====
% Inisialisasi Grafik Konvergensi
%=================================================
====
hfig = figure(1);
hold on
title('Convergence of Artificial Bee Colony Algorithm Graphic');
set(hfig, 'position', [50,40,600,300]);
set(hfig, 'DoubleBuffer', 'on');
hbestplot1 = plot(1:maxCycle,zeros(1,maxCycle));
xlabel('Cycles of Bees');
ylabel('Fitness Function');
hold off
drawnow;
iter=1;
while ((iter <= maxCycle)),
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% EMPLOYED BEE PHASE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
%disp('=============================================
=== EMPLOYED BEE PHASE
====================================')
for iabc=1:(FoodNumber)

      Param2Change=fix(rand*D1)+1  ;          %/*The parameter to be
changed is determined randomly*/
```

```
        neighbour=fix(rand*(FoodNumber))+1;     %/*A randomly chosen
solution is used in producing a mutant solution of the solution i*/

        %/*Randomly selected solution must be different from the solution
i*/
           while(neighbour==iabc)
              neighbour=fix(rand*(FoodNumber))+1;
           end
        solQ=FoodsQ(iabc,:);
        solBus=FoodsBus(iabc,:);

        %  /*v_{ij}=x_{ij}+\phi_{ij}*(x_{kj}-x_{ij}) */

solQ(Param2Change)=FoodsQ(iabc,Param2Change)+(FoodsQ(iabc,Para
m2Change)-FoodsQ(neighbour,Param2Change))*(rand-0.5)*2;

solBus(Param2Change)=FoodsBus(iabc,Param2Change)+(FoodsBus(iabc
,Param2Change)-FoodsBus(neighbour,Param2Change))*(rand-0.5)*2;
        solBus(Param2Change)=round(solBus(Param2Change));

        %  /*if generated parameter value is out of boundaries, it is shifted
onto the boundaries*/
        if solQ(Param2Change)<lb
            solQ(Param2Change)=lb;
        end
        if solQ(Param2Change)>ub
            solQ(Param2Change)=ub;
        end
        if solBus(Param2Change)<ubus
            solBus(Param2Change)=ubus;
        end
        if solBus(Param2Change)>lbus
            solBus(Param2Change)=lbus;
        end

        EMPLOYED=[solQ solBus];
        % evaluate new solution
        busdata(:,12)=0;
```

```
        busdata(solBus(:,:),12)=solQ(:,:);
        total_loss2=loadflow(busdata,linedata,0);
        ObjValSol(iabc)=total_loss2;  % ObjValSol=feval(objfun,sol);
        FitnessSol(iabc)=calculateFitness(ObjValSol(iabc));

        if (FitnessSol(iabc)>Fitness(iabc))%(iabc)) %/*If the mutant solution
is better than the current solution iabc, replace the solution with the
mutant and reset the trial counter of solution i*/
            FoodsQ(iabc,:)=solQ;
            FoodsBus(iabc,:)=solBus;
            Fitness(iabc)=FitnessSol(iabc);
            ObjVal(iabc)=ObjValSol(iabc);
            trial(iabc)=0;
            else
            trial(iabc)=trial(iabc)+1; %/*if the solution i can not be improved,
increase its trial counter*/
        end;
end
%%%%%%%%%%%%%%%%%%%%%%%% CalculateProbabilities
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%/* A food source is chosen with the probability which is proportioal to
its quality*/
%/*Different schemes can be used to calculate the probability values*/
%/*For example prob(iabc)=fitness(iabc)/sum(fitness)*/
%/*or in a way used in the metot below
prob(iabc)=a*fitness(iabc)/max(fitness)+b*/
%/*probability values are calculated by using fitness values and
normalized by dividing maximum fitness value*/

prob=(0.9.*Fitness./max(Fitness))+0.1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% ONLOOKER BEE PHASE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%
```

```
%disp('=============================================
=== ONLOOKER BEE PHASE
====================================')
iabc=1;
t=0;
while(t<FoodNumber)
   if(rand<prob)              %(iabc)
      t=t+1;

      %/*The parameter to be changed is determined randomly*/
      Param2Change=fix(rand*D1)+1  ;

      %/*A randomly chosen solution is used in producing a mutant
solution of the solution i*/
      neighbour=fix(rand*(FoodNumber))+1;

      %/*Randomly selected solution must be different from the solution
i*/
         while(neighbour==iabc)
            neighbour=fix(rand*(FoodNumber))+1;
         end
      solQ=FoodsQ(iabc,:);
      solBus=FoodsBus(iabc,:);

      %  /*v_{ij}=x_{ij}+\phi_{ij}*(x_{kj}-x_{ij}) */

solQ(Param2Change)=FoodsQ(iabc,Param2Change)+(FoodsQ(iabc,Para
m2Change)-FoodsQ(neighbour,Param2Change))*(rand-0.5)*2;

solBus(Param2Change)=FoodsBus(iabc,Param2Change)+(FoodsBus(iabc
,Param2Change)-FoodsBus(neighbour,Param2Change))*(rand-0.5)*2;
      solBus(Param2Change)=round(solBus(Param2Change));

      %  /*if generated parameter value is out of boundaries, it is shifted
onto the boundaries*/
      if solQ(Param2Change)<lb
          solQ(Param2Change)=lb;
      end
```

```
        if solQ(Param2Change)>ub
            solQ(Param2Change)=ub;
        end
        if solBus(Param2Change)<ubus
            solBus(Param2Change)=ubus;
        end
        if solBus(Param2Change)>lbus
            solBus(Param2Change)=lbus;
        end
        ONLOOKER=[solQ solBus];

        % evaluate new solution
        busdata(:,12)=0;
        busdata(solBus(:,:),12)=solQ(:,:);
        total_loss2=loadflow(busdata,linedata,0);
        ObjValSol(iabc)=total_loss2;  % ObjValSol=feval(objfun,sol);
        FitnessSol(iabc)=calculateFitness(ObjValSol(iabc));

        if (FitnessSol(iabc)>Fitness(iabc))%(iabc)) %/*If the mutant solution
is better than the current solution iabc, replace the solution with the
mutant and reset the trial counter of solution i*/
            FoodsQ(iabc,:)=solQ;
            FoodsBus(iabc,:)=solBus;
            Fitness(iabc)=FitnessSol(iabc);
            ObjVal(iabc)=ObjValSol(iabc);
            trial(iabc)=0;
        else
            trial(iabc)=trial(iabc)+1; %/*if the solution i can not be improved,
increase its trial counter*/
        end;
    end
    iabc=iabc+1;
    if (iabc==(FoodNumber)+1)
        iabc=1;
    end;
end

%/*The best food source is memorized*/
```

```
   ind=find(ObjVal==min(ObjVal));
   ind=ind(end);
   if (ObjVal(ind)<GlobalMin)
      GlobalMin=ObjVal(ind);
      GlobalParamsQ=FoodsQ(ind,:);
      GlobalParamsBus=FoodsBus(ind,:);
   end;
   Mx(iter)=min(GlobalMin);

ONLOOKER=[GlobalParamsQ GlobalParamsBus];

%%%%%%%%%%%%%%%%%%%%%%%% SCOUT BEE PHASE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%
%disp('=========================================
=== SCOUT BEE PHASE
====================================')

ind=find(trial==max(trial));
ind=ind(end);
if (trial(ind)>limit)
   Bas(ind)=0;
   solQ=(ub-lb).*rand(1,D1)+lb;
%     solBus=(lbus-ubus).*rand(1,D2)+ubus;
%     solBus=round(solBus);

   SCOUT=[solQ solBus];
   busdata(solBus(:,:),12)=solQ(:,:);
   %data_Fix1;
   total_loss2=loadflow(busdata,linedata,0);
   ObjValSol((ind))=total_loss2;%
   %ObjValSol=feval(objfun,sol);
   FitnessSol((ind))=calculateFitness(ObjValSol(ind));

   FoodsQ(iabc,:)=solQ;
   FoodsBus(iabc,:)=solBus;
   Fitness(ind)=FitnessSol(ind);
   ObjVal(ind)=ObjValSol(ind);
```

```
end;

% fprintf('Iter=%d ObjVal=%g\n',iter,GlobalMin);
plotvectorabc=get(hbestplot1,'Ydata');
plotvectorabc(iter)=Mx(iter);
set(hbestplot1,'Ydata',plotvectorabc);
drawnow
% fprintf('Mean =%g
Std=%g\n',mean(GlobalMins(:,end)),std(GlobalMins(:,end)));
iter=iter+1;;

end % End of ABC

GlobalMins(r)=GlobalMin;
end; %end of runs
busdata(:,12)=0;
GlobalParamsQakhir=GlobalParamsQ
GlobalParamsBusakhir=GlobalParamsBus
busdata(GlobalParamsBus,12)=GlobalParamsQ;
total_loss2=loadflow(busdata,linedata,1);
```

CALCULATING FITNESS

```
function fFitness=calculateFitness(fObjV)
fFitness=zeros(size(fObjV));
ind=find(fObjV>=0);
fFitness(ind)=1./(fObjV(ind)+1);
ind=find(fObjV<0);
fFitness(ind)=1+abs(fObjV(ind));
```

LOAD FLOW

```
%    File : lineflow for loadflow
%    Program latihan penggunaan MATLAB dalam
sistem tenaga listrik
%   This program is used in conjunction with
lfgauss or lf Newton
```

```
%   for the computation of line flow and line
losses.
%
%   Copyright (c) 1998 H. Saadat
function TLoss=loadflow(busdata,linedata,CETAK)
basemva = 1000; accuracy = 0.0001; maxiter = 50;
accel = 1.1;
% lfybus
j=sqrt(-1); i = sqrt(-1);
nl = linedata(:,1); nr = linedata(:,2); R =
linedata(:,3);
X = linedata(:,4); Bc = j*linedata(:,5); a =
linedata(:, 6);
nbr=length(linedata(:,1)); nbus = max(max(nl),
max(nr));
Z = R + j*X; y= ones(nbr,1)./Z;         %branch
admittance
for n = 1:nbr
if a(n) <= 0  a(n) = 1; else end
Ybus=zeros(nbus,nbus);     % initialize Ybus to
zero
               % formation of the off diagonal
elements
for k=1:nbr;
       Ybus(nl(k),nr(k))=Ybus(nl(k),nr(k))-
y(k)/a(k);
       Ybus(nr(k),nl(k))=Ybus(nl(k),nr(k));
    end
end
              % formation of the diagonal
elements
for  n=1:nbus
     for k=1:nbr
         if nl(k)==n
         Ybus(n,n) = Ybus(n,n)+y(k)/(a(k)^2) +
Bc(k);
         elseif nr(k)==n
         Ybus(n,n) = Ybus(n,n)+y(k) +Bc(k);
         else, end
     end
```

```
end
%=================================================
==========================
% lfnewton
%   Power flow solution by Newton-Raphson method
%   Copyright (c) 1998 by  H. Saadat
ns=0; ng=0; Vm=0; delta=0; yload=0; deltad=0;
nbus = length(busdata(:,1));
for k=1:nbus
n=busdata(k,1);
kb(n)=busdata(k,2); Vm(n)=busdata(k,3);
delta(n)=busdata(k, 4);
Pd(n)=busdata(k,5); Qd(n)=busdata(k,6);
Pg(n)=busdata(k,7); Qg(n) = busdata(k,8);
Qmin(n)=busdata(k, 9); Qmax(n)=busdata(k, 10);
Qsh(n)=busdata(k, 11);Qsvc(n)=busdata(k, 12);
    if Vm(n) <= 0  Vm(n) = 1.0; V(n) = 1 + j*0;
    else delta(n) = pi/180*delta(n);
         V(n) = Vm(n)*(cos(delta(n)) +
j*sin(delta(n)));
         P(n)=(Pg(n)-Pd(n))/basemva;
         Q(n)=(Qg(n)-
Qd(n)+Qsh(n)+Qsvc(n))/basemva;
         S(n) = P(n) + j*Q(n);
    end
end
%============================
for k=1:nbus
if kb(k) == 1, ns = ns+1; else, end
if kb(k) == 2 ng = ng+1; else, end
ngs(k) = ng;
nss(k) = ns;
end
Ym=abs(Ybus); t = angle(Ybus);
m=2*nbus-ng-2*ns;
maxerror = 1; converge=1;
iter = 0;
% Start of iterations
clear A  DC   J  DX
```

```
while maxerror >= accuracy & iter <= maxiter %
Test for max. power mismatch
for i=1:m
for k=1:m
   A(i,k)=0;       %Initializing Jacobian matrix
end, end
iter = iter+1;
for n=1:nbus
nn=n-nss(n);
lm=nbus+n-ngs(n)-nss(n)-ns;
J11=0; J22=0; J33=0; J44=0;
   for i=1:nbr
     if nl(i) == n | nr(i) == n
        if nl(i) == n,  l = nr(i); end
        if nr(i) == n,  l = nl(i); end
        J11=J11+ Vm(n)*Vm(l)*Ym(n,l)*sin(t(n,l)-
delta(n) + delta(l));
        J33=J33+ Vm(n)*Vm(l)*Ym(n,l)*cos(t(n,l)-
delta(n) + delta(l));
        if kb(n)~=1
        J22=J22+ Vm(l)*Ym(n,l)*cos(t(n,l)-
delta(n) + delta(l));
        J44=J44+ Vm(l)*Ym(n,l)*sin(t(n,l)-
delta(n) + delta(l));
        else, end
        if kb(n) ~= 1  & kb(l) ~=1
        lk = nbus+l-ngs(l)-nss(l)-ns;
        ll = l -nss(l);
      % off diagonalelements of J1
        A(nn, ll) =-
Vm(n)*Vm(l)*Ym(n,l)*sin(t(n,l)- delta(n) +
delta(l));
              if kb(l) == 0  % off diagonal
elements of J2
              A(nn, lk)
=Vm(n)*Ym(n,l)*cos(t(n,l)- delta(n) +
delta(l));end
              if kb(n) == 0  % off diagonal
elements of J3
```

```
                A(lm, ll) =-
Vm(n)*Vm(l)*Ym(n,l)*cos(t(n,l)-
delta(n)+delta(l)); end
                if kb(n) == 0 & kb(l) == 0  % off
diagonal elements of  J4
                A(lm, lk) =-
Vm(n)*Ym(n,l)*sin(t(n,l)- delta(n) +
delta(l));end
          else end
       else , end
    end
    Pk = Vm(n)^2*Ym(n,n)*cos(t(n,n))+J33;
    Qk = -Vm(n)^2*Ym(n,n)*sin(t(n,n))-J11;
    if kb(n) == 1 P(n)=Pk; Q(n) = Qk; end % Swing
bus P
      if kb(n) == 2  Q(n)=Qk;
           if Qmax(n) ~= 0
             Qgc = Q(n)*basemva + Qd(n) - Qsh(n) -
Qsvc(n);
             if iter <= 7                    %
Between the 2th & 6th iterations
                if iter > 2                  % the
Mvar of generator buses are
                  if Qgc  < Qmin(n),         %
tested. If not within limits Vm(n)
                  Vm(n) = Vm(n) + 0.01;      % is
changed in steps of 0.01 pu to
                  elseif Qgc  > Qmax(n),     % bring
the generator Mvar within
                  Vm(n) = Vm(n) - 0.01;end % the
specified limits.
                else, end
             else,end
           else,end
      end
    if kb(n) ~= 1
      A(nn,nn) = J11;  %diagonal elements of J1
      DC(nn) = P(n)-Pk;
    end
    if kb(n) == 0
```

```
      A(nn,lm) = 2*Vm(n)*Ym(n,n)*cos(t(n,n))+J22;
%diagonal elements of J2
      A(lm,nn)= J33;         %diagonal elements of
J3
      A(lm,lm) =-2*Vm(n)*Ym(n,n)*sin(t(n,n))-J44;
%diagonal of elements of J4
      DC(lm) = Q(n)-Qk;
    end
end
DX=A\DC';
for n=1:nbus
  nn=n-nss(n);
  lm=nbus+n-ngs(n)-nss(n)-ns;
    if kb(n) ~= 1
    delta(n) = delta(n)+DX(nn); end
    if kb(n) == 0
    Vm(n)=Vm(n)+DX(lm); end
 end
  maxerror=max(abs(DC));
     if iter == maxiter & maxerror > accuracy
   fprintf('\nWARNING: Iterative solution did not
converged after ')
   fprintf('%g', iter), fprintf('
iterations.\n\n')
   fprintf('Press Enter to terminate the
iterations and print the results \n')
   converge = 0; pause, else, end

end

if converge ~= 1
   tech= ('                         ITERATIVE
SOLUTION DID NOT CONVERGE'); else,
   tech=('                       Power Flow Solution
by Newton-Raphson Method');
end
V = Vm.*cos(delta)+j*Vm.*sin(delta);
deltad=180/pi*delta;
i=sqrt(-1);
```

```
k=0;
for n = 1:nbus
     if kb(n) == 1
     k=k+1;
     S(n)= P(n)+j*Q(n);
     Pg(n) = P(n)*basemva + Pd(n);
     Qg(n) = Q(n)*basemva + Qd(n) - Qsh(n) -
Qsvc(n);
     Pgg(k)=Pg(n);
     Qgg(k)=Qg(n);
     elseif  kb(n) ==2
     k=k+1;
     S(n)=P(n)+j*Q(n);
     Qg(n) = Q(n)*basemva + Qd(n) - Qsh(n) -
Qsvc(n);
     Pgg(k)=Pg(n);
     Qgg(k)=Qg(n);
  end
yload(n) = (Pd(n)-
j*Qd(n)+j*Qsh(n)+j*Qsvc(n))/(basemva*Vm(n)^2);
end
busdata(:,3)=Vm'; busdata(:,4)=deltad';
Pgt = sum(Pg);  Qgt = sum(Qg); Pdt = sum(Pd); Qdt
= sum(Qd); Qsht = sum(Qsh); Qsvcht = sum(Qsvc);

if CETAK

disp(tech)
fprintf('                        Maximum Power
Mismatch = %g \n', maxerror)
fprintf('                                  No. of
Iterations = %g \n\n', iter)
head =['    Bus  Voltage  Angle    ------Load----
--    ---Generation---   Injected   Injected    '
       '    No.  Mag.     Degree     MW
Mvar       MW       Mvar       Mvar      MVAR
'
       '
'];
```

```
disp(head)
for n=1:nbus
     fprintf(' %5g', n), fprintf(' %7.3f',
Vm(n)),
     fprintf(' %8.3f', deltad(n)), fprintf('
%9.3f', Pd(n)),
     fprintf(' %9.3f', Qd(n)),  fprintf(' %9.3f',
Pg(n)),
     fprintf(' %9.3f ', Qg(n)), fprintf(' %8.3f',
Qsh(n)),fprintf(' %8.3f\n', Qsvc(n))
end
    fprintf('      \n'), fprintf('    Total
')
    fprintf(' %9.3f', Pdt), fprintf(' %9.3f',
Qdt),
    fprintf(' %9.3f', Pgt), fprintf(' %9.3f',
Qgt), fprintf(' %9.3f', Qsht),fprintf('
%9.3f\n\n', Qsvcht),
%================================================
==========================

end

SLT = 0;
if CETAK
 fprintf('\n')
 fprintf('                                Line Flow
and Losses \n\n')
 fprintf('     --Line--  Power at bus & line flow
--Line loss--  Transformer\n')
 fprintf('     from  to    MW      Mvar     MVA
MW      Mvar      tap\n')
end

for n = 1:nbus
busprt = 0;
   for L = 1:nbr;
       if CETAK
       if busprt == 0
```

```
        fprintf('   \n'), fprintf('%6g', n),
fprintf('      %9.3f', P(n)*basemva)
        fprintf('%9.3f', Q(n)*basemva),
fprintf('%9.3f\n', abs(S(n)*basemva))

        busprt = 1;
        else, end
        end
        if nl(L)==n      k = nr(L);
        In = (V(n) - a(L)*V(k))*y(L)/a(L)^2 +
Bc(L)/a(L)^2*V(n);
        Ik = (V(k) - V(n)/a(L))*y(L) + Bc(L)*V(k);
        Snk = V(n)*conj(In)*basemva;
        Skn = V(k)*conj(Ik)*basemva;
        SL  = Snk + Skn;
        SLT = SLT + SL;
        elseif nr(L)==n  k = nl(L);
        In = (V(n) - V(k)/a(L))*y(L) + Bc(L)*V(n);
        Ik = (V(k) - a(L)*V(n))*y(L)/a(L)^2 +
Bc(L)/a(L)^2*V(k);
        Snk = V(n)*conj(In)*basemva;
        Skn = V(k)*conj(Ik)*basemva;
        SL  = Snk + Skn;
        SLT = SLT + SL;
        else, end
        if CETAK
          if nl(L)==n | nr(L)==n
          fprintf('%12g', k),
          fprintf('%9.3f', real(Snk)),
fprintf('%9.3f', imag(Snk))
          fprintf('%9.3f', abs(Snk)),
          fprintf('%9.3f', real(SL)),
              if nl(L) ==n & a(L) ~= 1
              fprintf('%9.3f', imag(SL)),
fprintf('%9.3f\n', a(L))
              else, fprintf('%9.3f\n', imag(SL))
              end
          else, end
        end
```

```
  end
end
SLT = SLT/2;
if CETAK
 fprintf('    \n'), fprintf('     Total loss
')
 fprintf('%9.3f', real(SLT)), fprintf('%9.3f\n',
imag(SLT))
end
TLoss=real(SLT);
clear Ik In SL SLT Skn Snk
```

BAB V
PENUTUP

5.1 Kesimpulan

Artificial Bee Colony (ABC) *Algorithm* sebagai metode yang diusulkan untuk penyelesaian permasalahan peletakan (*placement*) dan penentuan ukuran (*sizing*) *shunt capacitor* yang optimal pada jaringan transmisi sistem tenaga listrik menunjukan hasil yang memuaskan. Kerugian daya aktif digunakan sebagai fungsi objektif.

Setelah dilakukan percobaan kompensasi menentukan letak dan kapasitas kapasitor optimal di sistem transmisi Sumatera Utara 150 kV dapat ditarik kesimpulan bahwa terjadi penurunan kerugian daya aktif sebesar 8,37 % yaitu dari 106,437 MW menjadi 98,211 MW.

5.2 Saran

Dari hasil yang diperoleh pada Tugas Akhir ini ada berbagai saran untuk membantu dalam pengembangan penelitian selanjutnya agar diperoleh hasil yang lebih baik. Saran-saran yang dibutuhkan untuk pengembangan penelitian selanjutnya antara lain :

1. Kompensator yang digunakan tidak hanya dari *shunt capacitor* tipe *fixed capacitor*, tetapi juga melibatkan tipe-tipe yang lain (seperti *switched capacitor*).
2. Menggunakan metode penyelesaian permasalahan kompensasi daya reaktif yang lain, seperti *Genetic Algorithm* (GA), *Particle Swarm Optimization* (PSO), *Ant Colony Algorithm*, dan sebagainya.
3. Memperhitungkan faktor ekonomis seperti biaya pemasangan kapasitor.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] Mohammad A. S. Masoum, Marjan Ladjevardi, Akbar Jafarian and Ewald F. Fuchs, "*Optimal Placement, Replacement and Sizing of Capacitor Banks in Distorted Distribution Networks by Genetic Algorithms",* IEEE Transaction on Power Delivery, Vol. 19, No. 4, Oktober 2004.

[2] Ngakan Putu Satriya Utama, *"Memperbaiki Profil Tegangan Di Sistem Distribusi Primer dengan Kapasitor Shunt"*, Teknologi Elektro, 45 Vol. 7, No. 1 Januari -Juni 2008.

[3] Ji-Pyng Chiou, Chung-Fu Chang and Ching-Tzong Su, *"Ant Direction Hybrid Differential Evolution for Solving Large Capacitor Placement Problems"*, IEEE Transaction On Power Systems, Vol. 19, No. 4, November 2004.

[4] Ji-Pyng Chiou, Chung-Fu Chang and Ching-Tzong Su, "*Capacitor Placement in Large-Scale Distribution Systems Using Variable Scaling Hybrid Differential Evolution*", Electrical Power and Energy Systems, Vol. 28, Desember 2006.

[5] Ahmed M. Azmy, "*Optimal Power Flow to Manage Voltage Profiles in Interconnected Networks Using Expert Systems*", IEEE Transaction on Power Systems, Vol. 22, No. 4, November 2007.

[6] S.K. Bhattacharya, S.K. Goswami*, "A New Fuzzy Based Solution of The Capacitor Placement Problem in Radial Distribution System"*, Expert Systems with Applications, Vol. 36, 2009.

[7] Robandi, Imam. "*Desain Sistem Tenaga Modern*", ANDI, Yogyakarta, 2006.

[8] Saadat, Hadi. "*Power System Analysis*", McGraw-Hill, Singapore, 2004.

[9] Karaboga, D., "*A Comparative Study of Artificial Bee Colony Algorithm*", Applied Mathematics and Computation 214, Erciyes University, The Department of Computer Engineering, 2009.

[10] Haiyan Quan, Xinling Shi, *"On the Analysis of Performance of the Improved Artificial-Bee-Colony Algorithm"***.** Fourth International Conference on Natural Computation, 2008.

[11] Li-Pei Wong, Malcolm Yoke Hean Low and Chin Soon Chong, "*A Bee Colony Optimization Algorithm for Traveling Salesman*

Problem", Second Asia International Conference on Modelling & Simulation, Vol. 27, No. 4, Oktober 2008.

[12] Nurhan Karaboga. "*A New Design Method Based on Artificial Bee Colony Algorithm for Digital IIR Filters*", Journal of the Franklin Institute November 2008.

[13] http://us1.harunyahya.com/Detail/T/EDCRFV/productId/15049/THE_MIRACLE_OF_THE_HONEYBEE.

[14] Tereshko V., "*Reaction-Diffusion Model of A Honey Bee Colony's Foraging Behaviour*", Lecture Notes in Computer Science, Vol. 1917, Springer-Verlag: Berlin, p. 807-816, 2000.

[15] V. Tereshko, A. Loengarov, "*Collective Decision-Making in Honey Bee Foraging Dynamics*", Computing and Information Systems Journal, ISSN 1352-9404, Vol. 9, No. 3, Oktober 2005.

[16] Isnaini Laili Izzati, "*Economic Dispatch Optimization For 500 Kv Jawa Bali Electrical Power System Using Bacterial Foraging Optimization*", Final Project, Department of Electrical Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia 2010.

[17] Juningtijastuti, "*Optimization of Parameter and Location of UPFC For Transmission Loss Reduction Using Bacterial Foraging Algorithm*", Master Thesis, Department of Electrical Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia 2010.

[18] Ellithy, K, A. Al-Hinai, dan A. Moosa. 2008. "*Optimal Shunt Capacitors Allocation in Distribution Networks Using Genetic Algorithm- Practical Case Study*". Intenational Journal of Innovations in Energy Systems and Power, Vol. 3, No. 1 (April 2008).

[19] Sulistyo, Danang. "*Penentuan Letak dan Kapasitas Bank Kapasitor Secara Optimal Pada Jaring Transmisi Menggunakan Bee Colony Algorithm",* Final Project, Department of Electrical Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia 2010.

[20] Karaboga, D., "*An Idea Based On Honey Bee Swarm For Numerical Optimization*", Technical Report-TR06, Erciyes University, The Department of Computer Engineering, 2005.

BIOGRAFI PENULIS

ANDITA NOOR SHAFIRA atau biasa dipanggil Dita, merupakan anak pertama dari tiga bersaudara yang lahir di Jakarta pada tanggal 19 Agustus 1993. Penulis memulai jenjang pendidikan di TK Purwarini, Jakarta dan melanjutkan pendidikannya di SDN Makasar 09 Pagi Jakarta tahun 1999. Setelah menamatkan pendidikan sekolah dasar pada tahun 2005, penulis melanjutkan pendidikan di SMP Negeri 20 Jakarta dan kemudian di SMA Negeri 67 Jakarta pada tahun 2008. Setelah menempuh jenjang SLTA, tahun 2011 penulis diterima sebagai mahasiswa Program D-III Teknik Elektro, Sekolah Vokasi, Universitas Gajah Mada, Yogyakarta dengan konsentrasi Arus Kuat dan berhasil mendapatkan gelar A.Md pada tanggal 20 Agustus 2014. Pada tahun yang sama, penulis melanjutkan pendidikan S1 Lintas Jalur di Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember, Surabaya dan mengambil bidang studi Teknik Sistem Tenaga. Penulis pernah melakukan Kerja Praktek di Pupuk Kujang Cikampek tahun 2013 dan pada awal 2014 melakukan Tugas Akhir di Kangean Energy Indonesia Ltd. selama 2 bulan. Putri pertama dari pasangan Ir. Muhammad Edi Miraza dan Sugiarti Utami ini aktif dalam berbagai kegiatan diantaranya member Society of Petroleum Engineers ITS Student Chapter, member Society of Exploration Geophysicist ITS Student Chapter, member Laboratorium Simulasi Sistem Tenaga Listrik ITS. Adapun kritik, masukan, dan saran yang membangun diharapkan oleh penulis dengan langsung menghubungi anditanoorshafir@gmail.com