INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK ENERGI LISTRIK

ECONOMIC DISPATCH UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON

SKRIPSI

Disusun Oleh :
PURWANTO
NIM : 99.12.140

MARET 2006

[illegible]

[illegible]

[illegible]

LEMBAR PERSETUJUAN

ECONOMIC DISPATCH UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON

SKRIPSI

Disusun Guna Melengkapi dan Memenuhi Syarat-Syarat
Untuk Mencapai Gelar Sarjana Teknik Energi Listrik

Disusun Oleh :

PURWANTO
99.12.140

Mengetahui,	**Diperiksa dan Disetujui,**
Ketua Jurusan Teknik Elektro (S-1)	**Dosen Pembimbing**
Ir. F. Yudi Limpraptono, MT	**Ir. Yusuf Ismail Nakhoda, MT**
NIP.Y.103 95 00274	**NIP.Y.101 88 00189**

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK ENERGI LISTRIK
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

Lembar Persembahan

ASSALAMMU'ALAIKUM WR. WB
Kupersembahkan skripsi ini special untuk

Atas segala rahmat yang telah Engkau berikan kepadaku
Hanya Keridhoan-Mu-lah yang kuharapkan dalam hidup ini
Agar aku bisa menjalani perjalanan hidupku ini dengan penuh keikhlasan,
Kesabaran & ketabahan dalam keimanan & ketaqwaan

Nabi Besar Muhammad SAW karena beliaulah nabi yang diutus Allah SWT
Ke muka bumi untuk mengajak umat manusia dari zaman kegelapan
Menuju zaman yang penuh Barokah yakni Agama Islam
Semoga dengan Syafa'at beliau kita diberi keselamatan didunia &
diakhirat kelak

Ayahandaku tercinta, Suwardi Idja, Ibunda Sri Hartini, Adek2ku, Igun, Iwan, Adit dan Awin. Mbah Kakungku, Soedjipto , Mbah Putri, Almh. Suminah, Tante Yati, Om Mul dan anak2nya, Om Tono Sekeluarga, Om Yanto Sekeluarga, Tante Nday Sekeluarga, Om Yoso dan Tante Yeni serta tante Dede dan Om Adji serta yang tidak bisa Ipunk sebutkan satu-persatu.
Makasih banyak atas semua do'a, kasih sayang, pengorbanan, nasehat, dorongan & semangat yang diberikan kepada Ipunk, hingga akhirnya Ipunk bisa menyelesaikan kuliah ini tepat pada waktunya & dan semua berjalan dengan baik. Ipunk hanya bisa berdoa semoga Keluarga Besar Tercintaku, orang-orang yang selalu bersamaku, dalam suka & duka, selalu diberi kesehatan jasmani & rohani oleh Allah SWT & selalu dalam Lindungannya.
Sungguh suatu kebahagiaan tersendiri, karena Ipunk memiliki kalian semua.
Untuk Ayahanda Suwardi Idja & Ibunda Sri Hartini, semoga bahagia dan sehat selalu.
Untuk adek2ku, jangan berhenti belajar dan mencari ilmu sampai setinggi langit.
Om-om dan Tante-tanteku, maafin Ipunk, kalau selama ini Ipunk banyak ngerepotin & sampai detik ini belum banyak yang Ipunk persembahkan atas kebaikan semuanya.
Semoga Allah SWT masih memberikan waktu & kesempatan kepada Ipunk untuk berbagi kebahagiaan dengan kalian semuanya. Amien.

Mencintai...
Bukanlah Bagaimana Kamu Melupakan
Melainkan Bagaimana Kamu
Memaafkan
Bukanlah Bagaimana Kamu Mendengarkan
Melainkan Bagaimana Kamu
Mengerti
Bukanlah Apa Yang Kamu Lihat
Melainkan Apa yang Kamu
Rasakan
Bukanlah Bagaimana Kamu Melepaskan
Melainkan Bagaimana Kamu
Bertahan

Ingin kuungkapkan rasa Terima Kasih & Kasih Sayangku padamu, tapi kurasa goresan ini masih belum mampu mewakili apa yang ingin kuungkapkan, tapi walaupun begitu, ingin kucoba untuk melukiskannya lewat selembar kertas ini. Sayang, Walaupun kita baru kenal belum lama, tetapi kita udah saling mengerti satu sama yang lain. Maka dari itu, dari hatiku yang paling dalam ingin kuucapkan rasa syukurku, karena Allah SWT telah mempertemukan kita & memberikan kita kesabaran untuk menghadapi setiap cobaan yang dating. Aku sadar, aku tak akan bisa seperti yang sekarang ini bila tanpamu dihatiku. Terima kasih atas semua bantuan, doa, pengorbanan, kesabaran, penantian, perhatian, pengertian, rasa saying & cintamu kepadaku kekasihku, Maya Sari, S.Si. Hanya satu yang ingin kuucapkan untukmu, " I Always Love U, Forever...!" Semoga Alah SWT merestui apa yang telah kita rajut bersama. Dan tidak lupa ingin kuucapkan rasa terima kasih & sayangku buat keluarga di Purwosari. Terima kasih atas smua doa & support yang telah ibu berikan kepadaku. Semoga kita smua selalu dalam lindungan-Nya. Amien.

Dari Sayangmu
PURWANTO,ST.

INSTITUT TEKNOLOGI NASIONAL MALANG

LULUS

ABSTRAKSI

ECONOMIC DISPATCH UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON

(Purwanto, Nim. 99.12.140, Teknik Elektro/T.Energi Listrik)
(Dosen Pembimbing : Ir. Yusuf Ismail Nakhoda, MT.)

Kata Kunci : *Economic Dispatch, Emission Dispatch, Refined Genetic Algorithm, Lambda Iterative Technique.*

Dalam melayani kebutuhan daya listrik yang tidak tetap dari waktu ke waktu, sehingga menimbulkan suatu permasalahan yaitu bagaimana mengoperasikan suatu sistem tenaga listrik yang selalu dapat memenuhi permintaan daya pada setiap saat, dengan keandalan yang tinggi dan harga yang murah. Oleh karena itu pada suatu operasi pada beban tertentu, perhitungan ekonomis harus tetap merupakan suatu prioritas atau nilai yang harus diperhitungkan disamping hal-hal lain sehingga nantinya diperlukan suatu rencana operasi yang optimum dengan tetap memenuhi beberapa persyaratan pengoperasian sistem tenaga listrik yaitu antara lain : daya yang dibangkitkan cukup untuk memasok beban dan dapat mengetahui seberapa besar emisi yang dikeluarkan pada sistem pembangkitan listrik Paiton.

Skripsi ini menganalisis permasalahan *Economic Dispatch* atau pembebanan ekonomis dengan memperhatikan kendala emisi unit-unit pembangkit dalam melayani beban sistem selama periode waktu tertentu dengan menggunakan metode *Refined Genetic Algorithm*. Hasil dari analisa tersebut nantinya dapat digunakan sebagai salah satu acuan dalam operasi pembangkitan dan penyaluran daya yang ekonomis dan optimal, terutama mengenai biaya pembangkitan.

Analisa dilakukan dengan bantuan program komputer menggunakan bahasa pemrograman Delphi versi 7.0 dan telah sukses dicoba pada PLTU Paiton yang terdiri dari 6 unit pembangkit, dimana berhasil dilakukan penghematan biaya pembangkitan untuk hari Minggu, 8 Februari 2004 sebesar US $ 80,670 optimasi 3,82 %. Untuk hari Senin, 9 Februari 2004 sebesar US $ 133,326 optimasi 5,47 % dan hari Selasa, 10 Februari 2004 sebesar US $ 147,290 optimasi 5,94 % dengan menggunakan metode *Refined Genetic Algorithm*.

KATA PENGANTAR

Dengan memanjatkan puji syukur kehadirat Allah SWT, atas limpahan Rahmat dan Hidayah-Nya, sehingga penyusun dapat menyelesaikan skripsi ini dengan judul :

"*ECONOMIC DISPATCH* UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON"

Skripsi ini disusun sebagai salah satu persyaratan dalam menyelesaikan studi program strata satu (S-1) jurusan Teknik Elektro/Konsentrasi Teknik Energi Listrik, Fakultas Teknologi Industri, Institut Teknologi Nasional Malang

Sebelum dan selama penyusunan skripsi ini, penyusun telah banyak mendapatkan bantuan dan bimbingan dari berbagai pihak. Untuk itu pada kesempatan ini penyusun menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Bapak Ir. F.Yudi Limpraptono, MT, selaku Ketua Jurusan Teknik Elektro Fakultas Teknologi Industri Institut Teknologi Nasional Malang.
2. Ibu Ir. Mimien Mustikawati, selaku Sekretaris Jurusan Teknik Elektro Institut Teknologi Nasional Malang.
3. Bapak Ir. Yusuf Ismail Nakhoda, MT, selaku dosen pembimbing dalam penyusunan skripsi ini.
4. Bapak dan Ibu dosen jurusan Teknik Elektro Energi Listrik.
5. Ayah dan Ibu serta saudaraku, yang sangat berarti dalam hidup penyusun, dimana doa serta restu dan keridhaannya senantiasa penyusun harapkan.

6. Teman-teman Teknik Elektro Energi Listrik khususnya angkatan '99, terima kasih atas bantuan kalian semua. Serta semua pihak yang turut membantu penyusun menyelesaikan skripsi ini.

Penyusun menyadari sepenuhnya akan segala kekurangan yang ada dalam penyusunan skripsi ini, maka dengan kerendahan hati penyusun mengharapkan kritik dan saran demi penyempurnaan skripsi ini.

Akhirnya penyusun berharap semoga dalam skripsi ini dapat membantu serta bermanfaat bagi rekan-rekan mahasiswa khususnya pada jurusan Teknik Elektro Energi Listrik.

Malang, Maret 2006

Penyusun

DAFTAR ISI

HALAMAN JUDUL i

LEMBAR PERSETUJUAN ii

ABSTRAKSI iii

KATA PENGANTAR iv

DAFTAR ISI vi

DAFTAR GAMBAR x

DAFTAR TABEL xii

DAFTAR GRAFIK xvii

BAB I PENDAHULUAN 1

1.1. Latar Belakang Masalah 1

1.2. Rumusan Masalah 2

1.3. Tujuan Pemecahan Masalah 3

1.4. Batasan Masalah 3

1.5. Metodologi Pemecahan Masalah 3

1.6. Relevansi 4

1.7. Sistematika Penulisan 4

BAB II LANDASAN TEORI 6

2.1 Sistem Tenaga Listrik 6

2.2 Karakteristik Pembangkit Listrik 9

2.2.1. Karakteristik Masukan-Keluaran 10

2.2.2. Karakteristik Laju Tambahan Biaya Bahan Bakar 11

2.2.3. Fungsi Biaya Bahan Bakar 12

2.2.4. Karakteristik *Heat Rate* 12

2.3. Pembebanan Ekonomis Pembangkit Listrik 13

2.3.1. *Economic Dispatch* Dengan Mengabaikan Rugi-Rugi Transmisi 14

2.4. Kendala-Kendala Dalam Sistem Tenaga Listrik 15

2.4.1. Kendala Operasi Pada PLTU 15

2.4.2. Biaya Dan Pengendalian Pencemaran 17

2.4.3. Kendala Lingkungan 18

2.4.3.1. Oksida Nitrogen (NOx) 19

2.4.3.2. Strategi Pengendalian Emisi 20

BAB III APLIKASI *REFINED GENETIC ALGORITHM* UNIT PEMBANGKIT TERMAL DENGAN MEMPERHATIKAN KENDALA EMISI 23

3.1. Teori Dasar *Genetic Algorithm* 23

3.1.1. Konsep Dasar 23

3.1.2. Komponen-Komponen Utama *Genetic Algorithm* 24

3.1.2.1. Teknik Penyandian 24

3.1.2.2. Prosedur Inisialisasi 25

3.1.2.3. Fungsi Evaluasi 26

3.1.2.4. Seleksi 26

3.1.2.5. Operator Genetika 27

3.1.2.6. Penentuan Parameter 28

3.2. *Refined Genetic Algorithm* Dengan *Elitism* 29

3.3. Teori Dasar *Refined Genetic Algorithm* Dengan Memperhatikan Emisi 30

3.4. Formulasi Masalah *Economic Dispatch* Di Bawah Perencanaan Kompensasi Generasi 31

3.5. Implementasi *Refined Genetic Algorithm* Kaitannya dengan Minimum Emisi Dalam Optimasi Sistem Tenaga Listrik 32

BAB IV *ECONOMIC DISPATCH* UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON 35

4.1. Pendahuluan 35

4.2. Validasi Program 35

4.2.1. Data Validasi Referensi Jurnal IEEE 35

4.2.2. Hasil Validasi Referensi Jurnal IEEE Dengan Menggunakan Metode *Refined Genetic Algorithm* 36

4.3. Algoritma Program Pemecahan Masalah *Economic Dispatch* Pembangkit Termal Menggunakan Metode *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 38

4.3.1. Algoritma Program 38

4.3.2. Algoritma Fungsi *Fitness* 39

4.3.3. *Flowchart* Algoritma *Refined Genetic Algorithm* 40

4.3.4. *Flowchart* Fungsi *Fitness* 41

4.4. Data Pembangkit Termal Pada Sub Sistem Paiton 41

4.5. Data Pembebanan Pembangkit Termal Sub Sistem Paiton 43

4.6. Data Emisi NOx Pembangkit Termal Sub Sistem Paiton..... 45

4.7. Prosedur Pelaksanaan Program Perhitungan 46

4.8. Hasil Perhitungan Dan Analisa Data Antara PLTU Paiton dan *Refined Genetic Algorithm*..... 50

4.8.1. Perbandingan Daya, Emisi NOx dan Biaya Operasi Tiap Unit Pembangkit 50

4.8.2. Perbandingan Tingkat Optimum Daya 69

4.8.3. Perbandingan Tingkat Optimum Emisi NOx..... 71

4.8.4. Perbandingan Tingkat Optimum Biaya Operasi Pembangkitan..... 74

BAB V PENUTUP..... 81

5.1 Kesimpulan 81

5.2 Saran 82

DAFTAR PUSTAKA

LAMPIRAN

DAFTAR GAMBAR

Gambar 2.1. Elemen Pokok Sistem Tenaga Listrik 7

Gambar 2.2. Unit Boiler – Turbin – Generator 9

Gambar 2.3. Kurva karakteristik *Input-Output* Pembangkit Termal 11

Gambar 2.4. Kurva Karakteristik Laju Tambahan Biaya Bahan Bakar 12

Gambar 2.5. Kurva Karakteristik *Heat Rate* 13

Gambar 2.6. N Unit Pembangkit termis Melayani Beban P_D 14

Gambar 2.7. Biaya Sebagai Fungsi Dari Pengendalian Pencemaran 18

Gambar 2.8. Proses Penyebaran Emisi Dari Sumber Sampai Ke *Receptor* 19

Gambar 3.1. Representasi *String Bit* dan Pohon 25

Gambar 3.2. Seleksi Dengan Menggunakan Metode *Roulette Wheel* 27

Gambar 3.3. *Multi Point Crossover* 28

Gambar 3.4. Pembentukan *Next Generation* dalam *Refined Genetic Algorithm* 29

Gambar 4.1. Tampilan Parameter *Refined Genetic Algorithm* 37

Gambar 4.2. Tampilan Hasil Validasi Jurnal IEEE 37

Gambar 4.3. *Flowchart Refined Genetic Algorithm* 40

Gambar 4.4. *Flowchart* Fungsi *Fitness* 41

Gambar 4.5. Tampilan Utama Program 46

Gambar 4.6. Tampilan Masukan Data 47

Gambar 4.7. Tampilan Data Pembangkit 47

Gambar 4.8. Tampilan Data Pembebanan 48

Gambar 4.9. Tampilan Data PLTU Paiton 48

Gambar 4.10. Tampilan Parameter *Refined Genetic Algorithm* 49

Gambar 4.11. Tampilan Hasil Optimasi *Economic Dispatch* Yang Memperhatikan Kendala Emisi 49

DAFTAR TABEL

Tabel 2.1. Baku Mutu Emisi Untuk Pembangkit Listrik Tenaga Uap Berbahan Bakar Batubara 21

Tabel 2.2. Standar Emisi Udara 22

Tabel 3.1. *String* dan Nilai *Fitness* 27

Tabel 4.1. Data Beban Validasi 36

Tabel 4.2. Data Unit Pembangkit Validasi 36

Tabel 4.3. Perbandingan Data Referensi Jurnal Dengan Hasil Program 38

Tabel 4.4. Parameter Unit Pembangkit Termal 42

Tabel 4.5. Persamaan Biaya Bahan Bakar PLTU Paiton 43

Tabel 4.6. Parameter Emisi NOx PLTU Paiton 43

Tabel 4.7. Data Beban Sistem PLTU Paiton 44

Tabel 4.8. Data Emisi (NOx) PLTU Paiton 45

Tabel 4.9. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 1 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Minggu, 8 Februari 2004 51

Tabel 4.10. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 2 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Minggu, 8 Februari 2004 52

Tabel 4.11. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 5 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Minggu, 8 Februari 2004 53

Tabel 4.12. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 6 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Minggu, 8 Februari 2004 54

Tabel 4.13. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 7 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Minggu, 8 Februari 2004 55

Tabel 4.14. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 8 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Minggu, 8 Februari 2004 56

Tabel 4.15. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 1 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Senin, 9 Februari 2004 57

Tabel 4.16. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 2 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Senin, 9 Februari 2004 58

Tabel 4.17. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 5 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Senin, 9 Februari 2004 59

Tabel 4.18. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 6 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Senin, 9 Februari 2004 60

Tabel 4.19. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 7 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Senin, 9 Februari 2004 61

Tabel 4.20. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 8 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Senin, 9 Februari 2004 62

Tabel 4.21. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 1 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Selasa, 10 Februari 2004 63

Tabel 4.22. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 2 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Selasa, 10 Februari 2004 64

Tabel 4.23. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 5 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Selasa, 10 Februari 2004 65

Tabel 4.24. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 6 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Selasa, 10 Februari 2004 66

Tabel 4.25. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 7 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Selasa, 10 Februari 2004 67

Tabel 4.26. Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 8 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Selasa, 10 Februari 2004 68

Tabel 4.27. Perbandingan Daya Yang Dihasilkan Tiap 24 Jam Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 69

Tabel 4.28. Perbandingan Total Daya Yang Dihasilkan Tiap 72 Jam Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 70

Tabel 4.29. Perbandingan Total Emisi Yang Dihasilkan Tiap 24 Jam Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 72

Tabel 4.30. Perbandingan Total Emisi Tiap 72 Jam Tiap Unit Pembangkit Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 72

Tabel 4.31. Perbandingan Total Emisi (NOx) Perhari Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 73

Tabel 4.32. Perbandingan Biaya Tiap Jam Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 75

Tabel 4.33. Perbandingan Total Biaya Yang Dihasilkan Tiap 24 Jam Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 76

Tabel 4.34. Perbandingan Biaya Yang Dihasilkan Selama Tiga Hari Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 77

Tabel 4.35. Perbandingan Total Biaya Perhari Pada PLTU Paiton dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 77

Tabel 4.36. Perbandingan Total Biaya Perhari PLTU Paiton, *Refined Genetic Algorithm* Tanpa Kendala Emisi Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 79

DAFTAR GRAFIK

Grafik 4.1. Perbandingan Total Daya Tiap 72 Jam Tiap Unit Pembangkit Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 70

Grafik 4.2. Perbandingan Total *Output* Emisi (NOx) Perhari Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 73

Grafik 4.3. Perbandingan Total Biaya Perhari Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi 78

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Untuk mengatasi pemenuhan kebutuhan akan energi listrik dari para pelanggan yang ditanggung PLN Pembangkitan Jawa Bali, maka diperlukan perencanaan penggunaan pembangkit listrik yang ada secara efisien dan seoptimal mungkin dengan memperkirakan kemampuan dan kendala tiap unit pembangkit dalam mensuplai beban dasar atau sebagai pembangkit cadangan sebagai beban puncak.

Peningkatan konsumsi energi ini, tidak dibarengi dengan peningkatan kemampuan dalam penyediaan atau produksi energi, baik dalam investasi pembangkit baru maupun biaya operasi pembangkitan yang cenderung mahal. Sehingga diperlukan suatu optimalisasi dalam operasi pembangkitan energi listrik yang tersedia, agar dapat memenuhi keseimbangan daya dan biaya bahan bakar yang ekonomis dan optimal.

Pembangunan pembangkit tenaga listrik yang merupakan tuntutan jaman apabila menggunakan teknologi konvensional akan menimbulkan masalah lingkungan yang serius. Penyediaan tenaga listrik bukan saja memerlukan pusat pembangkit, tetapi juga sumber tenaga energi primer sebagai bahan bakarnya.

Dalam pemanfaatan bahan bakar fosil sebagai sumber energi primer untuk menghasilkan tenaga listrik, maka terjadi suatu proses yang mana selain menghasilkan tenaga listrik juga menghasilkan produk sampingan berupa emisi yang dapat mengganggu kelestarian lingkungan. Dampak lingkungan akibat

pemakaian bahan bakar fosil dalam sektor ketenagalistrikan akan naik sebanding dengan konsumsi energi listrik. Efek yang ditimbulkan secara umum dari pemakaian bahan bakar fosil adalah[1]:

a. hujan asam (SO_2 dan NO_2).

b. pemanasan global (CO_2).

c. penipisan lapisan ozon.

Atas dasar itulah skripsi ini akan mencoba membahas penggunaan metode *Refined Genetic Algorithm* dengan memperhatikan kendala emisi untuk menyelesaikan masalah optimasi. Metode ini diharapkan mampu menekan biaya operasi pada pembangkit tenaga listrik dan lebih ekonomis.

1.2. Rumusan Masalah

Salah satu permasalahan dalam mengoperasikan pembangkit tenaga listrik adalah bagaimana cara meminimumkan biaya pembangkit listrik dengan memperhitungkan biaya bahan bakar yang pada umumnya adalah biaya terbesar dalam perusahaan listrik, dengan melihat seberapa besar daya yang harus dihasilkan dengan memperhitungkan seberapa besar emisi yang akan dihasilkan. Salah satu metode yang dapat dipakai adalah metode *Refined Genetic Algorithm*. Oleh karena itu akan dikaji penggunaan metode *Refined Genetic Algorithm* dengan memperhatikan kendala emisi.

Berdasarkan gambaran permasalahan diatas, maka skripsi ini diberi judul :

***Economic Dispatch* Unit Pembangkit Termal Menggunakan Metode *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Pada PLTU Paiton.**

1.3. Tujuan Pemecahan Masalah

Berdasarkan permasalahan yang telah dikemukakan diatas, maka skripsi ini bertujuan memberikan analisis penerapan metode *Refined Genetic Algorithm* untuk unit pembangkit termal yang melayani kebutuhan beban yang berubah tiap jamnya, dan untuk mengetahui besarnya biaya total yang dapat ditekan dengan memperhatikan kendala emisi menggunakan *Economic Dispatch* (pembebanan ekonomis).

1.4. Batasan Masalah

Agar penelitian ini dapat dilakukan secara terarah dan tidak meluas, maka perlu adanya beberapa batasan masalah sebagai berikut:

a. Analisis dilakukan dengan asumsi bahwa sistem berada dalam operasi normal.
b. Tidak membahas masalah rugi-rugi dalam saluran transmisi.
c. Hanya memperhitungkan emisi NO_x.
d. Tidak mempertimbangkan efisiensi peralatan pengendali emisi.
e. Optimalisasi dilakukan hanya untuk pembangkit termal yang ada di PT. PLN UNIT PEMBANGKITAN JAWA BALI (PAITON 1&2), PT. YTL (PAITON 5&6) dan PT. EMOMI (PAITON 7&8).

1.5. Metodologi Pemecahan Masalah

Metodologi yang digunakan dalam pembahasan dilaksanakan dengan langkah-langkah sebagai berikut :

- Studi kepustakaan mengenai hal-hal yang berhubungan dengan pembahasan masalah.

- Pengumpulan data-data unit pembangkit termal, meliputi: Data parameter pembangkit termal, data pembebanan dan data emisi NOx di PLTU PAITON.
- Perhitungan *economic dispatch* unit pembangkit termal yang memperhatikan kendala emisi dengan menggunakan metode RGA dan dengan menggunakan bahasa pemprograman *Borland Delphi* versi 7.0.
- Membuat evaluasi, sehingga dapat disimpulkan apakah metode yang diterapkan efisien atau ekonomis.

1.6. Relevansi

Analisis ini dapat digunakan sebagai alternatif metode dalam menyelesaikan permasalahan *economic dispatch* unit pembangkit termal ysng memperhatikan kendala emisi dengan menggunakan metode *Refined Genetic Algorithm*.

1.7. Sistematika Penulisan

Adapun sistematika penulisan pada skripsi ini adalah sebagai berikut :

BAB I : PENDAHULUAN

Menguraikan latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, metodologi, sistematika penulisan dan relevansi.

BAB II : LANDASAN TEORI

Menguraikan pembahasan sistem tenaga listrik, karakteristik pembangkit, kendala lingkungan (emisi) dalam operasi sistem tenaga listrik, *economic dispatch* secara umum.

BAB III : APLIKASI *REFINED GENETIC ALGORITHM* UNIT PEMBANGKIT TERMAL DENGAN MEMPERHATIKAN KENDALA EMISI

Menguraikan teori dasar dari *Genetic Algorithm* dan implementasi *Refined Genetic Algorithm* kepermasalahan *economic dispatch* dengan kendala emisi.

BAB IV : *ECONOMIC DISPATCH* UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON

Menguraikan tentang hasil simulasi program dalam menyelesaikan masalah *economic dispatch* yang memperhitungkan kendala emisi serta hasil perhitungan, analisis dan evaluasinya.

BAB V : PENUTUP

Memuat intisari dari hasil pembahasan, yang berisikan kesimpulan dari analisa data yang telah dilakukan.

BAB II
LANDASAN TEORI

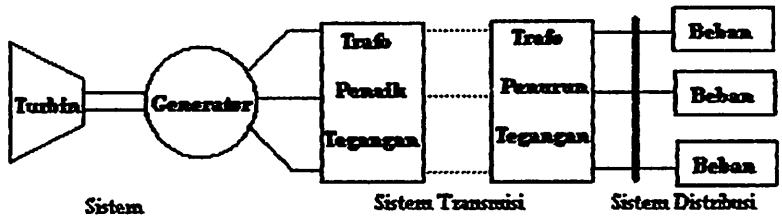
2.1. Sistem Tenaga Listrik

Karena berbagai persoalan teknis, tenaga listrik hanya dapat dibangkitkan pada lokasi tertentu. Mengingat pemakai tenaga listrik atau pelanggan tenaga listrik tersebar diberbagai tempat, sehingga diperlukan suatu manajemen operasi yang baik. Manajemen operasi tenaga listrik yang baik harus memikirkan bagaimana menyediakan tenaga listrik yang seekonomis mungkin dengan tetap memperhatikan mutu dan keandalan. Mutu dan keandalan diukur dengan frekuensi, tegangan dan jumlah gangguan. Masalah mutu tenaga listrik tidak semata-mata merupakan masalah operasi sistem tenaga listrik, tetapi erat kaitannya dengan pemeliharaan instalasi tenaga listrik dan juga masalah pengembangan sistem tenaga listrik, karena mengingat konsumsi tenaga listrik oleh pelanggan selalu berubah dari waktu ke waktu. Oleh karena itu, hasil-hasil operasi sistem tenaga listrik perlu dianalisa dan dievaluasi untuk menjadi masukan bagi pemeliharaan instalasi serta pengembangan sistem tenaga listrik. Mutu tenaga listrik yang baik merupakan kendala (pembatas) terhadap biaya pengadaan tenaga listrik yang serendah mungkin, maka kompromi antara keduanya merupakan masalah optimasi yang cukup komplek.

Tenaga listrik dibangkitkan di pusat-pusat listrik seperti PLTA, PLTD, PLTU, PLTG dan PLTGU, kemudian disalurkan melalui saluran transmisi setelah tegangannya dinaikkan terlebih dahulu oleh Transformator Penaik Tegangan yang terdapat di pusat listrik.

Setelah tenaga listrik disalurkan melalui transmisi, maka sampailah tenaga listrik tersebut di Gardu Induk (GI) yang untuk kemudian tegangannya diturunkan oleh Trafo Penurun Tegangan menjadi tegangan menengah atau tegangan distribusi primer.

Jaringan keluar dari gardu induk umumnya disebut jaringan distribusi, sedangkan antara pusat listrik dengan gardu induk disebut jaringan transmisi. Setelah disalurkan melalui jaringan distribusi primer maka tenaga listrik, kemudian diturunkan tegangannya oleh gardu distribusi menjadi tegangan distribusi sekunder atau tegangan rendah, dan baru kemudian disalurkan ke konsumen [4].



Gambar 2.1. Elemen Pokok Sistem Tenaga Listrik [4]

Dari uraian diatas kiranya dapat mengerti bahwa besar kecilnya tenaga listrik ditentukan sepenuhnya oleh konsumen, yaitu tergantung bagaimana para konsumen akan menggunakan peralatan listriknya, kemudian PT. PLN (Persero) harus mengimbangi kebutuhan tenaga listrik tersebut dalam arti selalu menyesuaikan daya listrik yang dibangkitkan dari waktu ke waktu.

Untuk memenuhi kebutuhan tenaga listrik bagi para konsumen, diperlukan berbagai peralatan listrik. Berbagai peralatan listrik ini dihubungkan satu sama

yang lain secara keseluruhan membentuk suatu sistem tenaga listrik. Sehingga yang disebut Sistem Tenaga Listrik disini adalah sekumpulan pusat listrik dan gardu induk yang satu dengan yang lain dihubungkan oleh jaringan transmisi sehingga merupakan sebuah kesatuan interkoneksi.

Karena daya listrik yang dibangkitkan harus sama dengan tenaga listrik yang dibutuhkan oleh konsumen, maka manajemen operasi sistem tenaga listrik harus mampu memperhatikan hal-hal sebagai berikut [4] :

a. Perkiraan beban.
b. Syarat-syarat pemeliharaan peralatan.
c. Keandalan yang diinginkan
d. Pengaturan dan penyaluran beban
e. Proses produksi tenaga listrik yang ekonomis

Kelima hal diatas masih harus sering dikaji ulang terhadap berbagai kendala seperti :

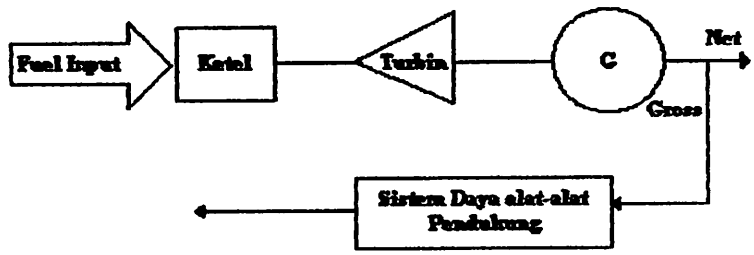
a. Aliran beban dalam jaringan
b. Daya hubung singkat dan gangguan yang sering menimpa peralatan
c. Stabilitas sistem
d. Penyediaan suku cadang dan dana

Dengan memperhatikan kendala-kendala diatas maka seringkali harus dilakukan pengaturan kembali terhadap rencana pemeliharaan dan alokasi beban. Makin besar sistem, makin banyak hal yang harus diamati dan dikoordinasi, sehingga diperlukan perencanaan, pelaksanaan, pengendalian dan evaluasi sistem yang cermat.

2.2. Karakteristik Pembangkit Listrik

Hal yang paling mendasar dalam optimasi ekonomi dari sebuah pembangkit listrik termal adalah dengan ditentukannya karakteristik masukan-keluaran (*input-output characteristic*) pusat listrik tersebut. Dalam mendefinisikan karakteristik masukan-keluaran, akan dibicarakan tentang *gross input* dan *net output* yang dihasilkan pusat listrik tersebut. *Gross input* pembangkit termal menyatakan jumlah keseluruhan bahan bakar yang diperlukan, sedangkan *net output* adalah daya nyata (*real power*) yang dihasilkan pembangkit listrik tersebut[2].

Tipe sebuah pembangkit listrik termal tampak pada gambar 2.2. Bagan tersebut terdiri atas sebuah ketel yang menghasilkan uap untuk menggerakkan turbin uap yang dikopel dengan sebuah generator listrik. Daya listrik dihasilkan tidak seluruhnya ke sistem tetapi sebagian kecil digunakan untuk mengoperasikan peralatan yang terdapat pada pusat listrik tersebut, seperti ketel, pompa, kompresor dan sebagainya, serta untuk mencatu peralatan kontrol, komunikasi, penerangan dan komputer.



Gambar 2.2. Unit Boiler – Turbin – Generator[2]

2.2.1. Karakteristik Masukan – Keluaran

Masukan sebuah pembangkit listrik termal umumnya dinyatakan sebagai banyaknya energi per satuan waktu dari bahan bakar yang diberikan ke ketel untuk menghasilkan daya listrik yang merupakan keluaran dari pusat listrik tersebut. Terdapat dua notasi yang umum digunakan [2] :

H dengan satuan [Mbtu/hour]

F dengan satuan [$US/hour],

Dimana $F = H \times \$US/Btu$, dari *$US/Btu* menyatakan harga bahan bakar per satuan energi yang dikandung oleh bahan bakar tersebut.

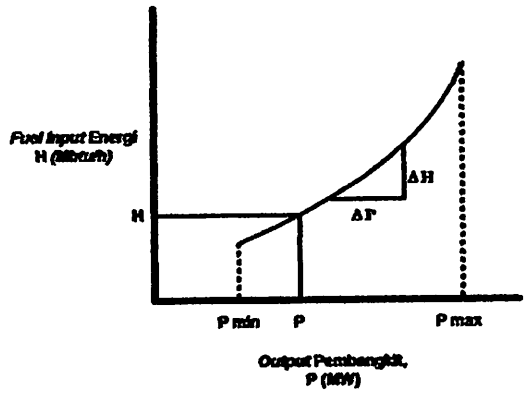
Sedangkan keluaran dari pembangkit listrik termal adalah daya nyata yang dihasilkan oleh generator dikurangi dengan daya nyata yang dipakai oleh pusat listrik tersebut. Notasi yang umum digunakan adalah :

P dengan satuan [MW]

Jadi dapat disimpulkan bahwa masukan pusat listrik merupakan fungsi terhadap keluarannya, maka hubungan tersebut dapat ditulis sebagai berikut :

$H = f(P)$ *[Mbtu/h] atau* $F = f(P)$ *[$US/h]*

Pembahasan selanjutnya akan berpedoman atas dasar fungsi biaya bahan bakar ($F = f(P)$ *[$US]*), sedangkan kurva dari karakteristik masukan-keluaran dari sebuah pembangkit tenaga termal yang telah diidealkan ditunjukkan pada gambar 2.3. Masukan adalah sebagai ordinat, yang berupa banyaknya energi yang diperlukan per satuan waktu [Mbtu/h] atau dapat juga merupakan biaya bahan bakar yang dikonsumsi per satuan waktu [$US/h]. Sedangkan keluaran adalah daya listrik [MW] yang dihasilkan blok tersebut untuk melayani beban sistem.



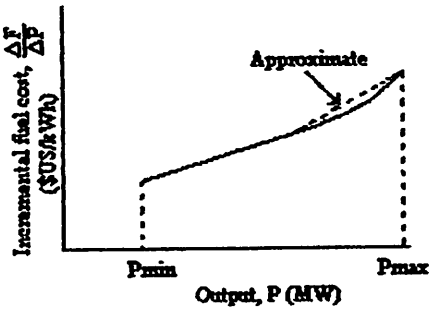
Gambar 2.3. Kurva Karakteristik Input – Output Pembangkit Termal [2]

Data yang diperlukan untuk menggambarkan diagram fungsi karakteristik masukan-keluaran dapat diperoleh dari perhitungan pada saat perencanaan atau tes yang telah dilakukan terhadap unit pembangkit yang bersangkutan.

2.2.2. Karakteristik Laju Tambahan Biaya Bahan Bakar

Karakteristik laju tambahan biaya bahan bakar atau *Incremental Fuel Cost Characteristic* adalah turunan pertama dari fungsi biaya bahan bakar F [$US/h] terhadap tingkat pembebanan P [MW] dari pusat listrik yang bersangkutan. Fungsi ini menunjukkan besarnya kenaikan atau penurunan biaya bahan bakar untuk setiap satu satuan perubahan beban.

Secara luas fungsi biaya bahan bakar akan digunakan untuk menentukan pembebanan ekonomis dari sebuah pembangkit listrik tenaga termal. Tampak pada gambar 2.4 adalah kurva laju tambahan biaya bahan bakar yang telah diidealkan dari sebuah pembangkit listrik termal [2].



Gambar 2.4. Kurva Karakteristik Laju Tambahan Biaya Bahan Bakar [2]

2.2.3. Fungsi Biaya Bahan Bakar

Biaya bahan bakar merupakan unsur biaya yang penting dalam operasi sistem pembangkitan termal. Fungsi biaya bahan bakar F_i (P_i) untuk tiap unit pembangkit terhadap daya keluaran diekspresikan dalam bentuk fungsi kuadrat, yang dapat dinyatakan sebagai berikut [2] :

$$F_i(P_{it}) = a_i + b_iP_{it} + ciPit^2 \quad \text{.....(2.1)}$$

dimana : a_i , b_i , c_i = konstanta persamaan dari unit ke-*i*

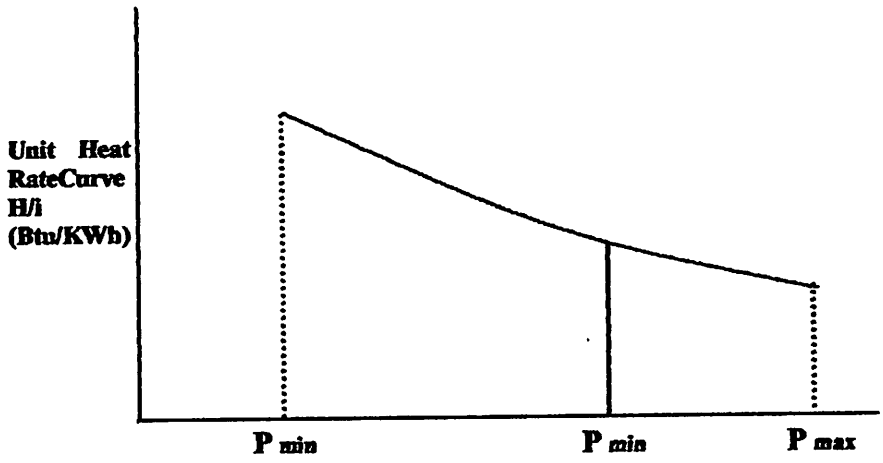
P_{it} = daya keluaran dari unit ke-*i* pada jam *t*

Dalam pengoperasian secara ekonomis adalah penting untuk mengetahui biaya bahan bakar yang digunakan untuk membangkitkan daya yang diperlukan, yaitu: jenis bahan bakar, nilai kalori dan biaya bahan bakar.

2.2.4. Karakteristik *Heat Rate*

Karakteristik *heat rate* merupakan karakteristik yang menunjukkan efisiensi dari mesin. Karakteristik *heat rate* sebuah unit pembangkit menunjukkan *input* kalor yang diberikan untuk menghasilkan energi 1 kilowatjam pada

megawatt *output* dari suatu unit [2]. Kurva dari karakteristik *heat rate* dapat dilihat pada gambar 2.5.



Gambar 2.5. Kurva Karakteristik *Heat Rate*[2]

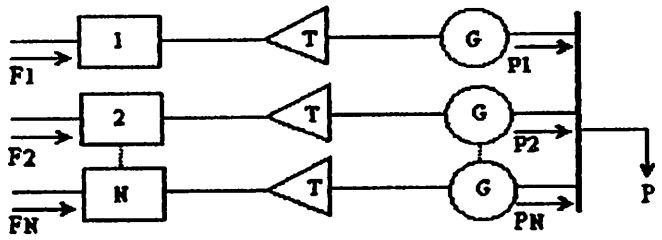
2.3. Pembebanan Ekonomis Pembangkit Listrik

Pembebanan ekonomis atau *economic dispatch* adalah pembangian pembebanan pada pembangkit-pembangkit yang ada dalam sistem tenaga listrik, secara optimal dan ekonomis pada harga beban tertentu. Komponen terbesar dari biaya pembangkitan adalah biaya bahan bakar[2]. Oleh sebab itu dengan dilakukannya *economic dispatch* berarti pula didapatkan biaya bahan bakar pembangkitan yang paling murah. Oleh karena beban yang harus ditanggung oleh sistem pembangkit selalu berubah setiap periode waktu tertentu, maka perhitungan *economic dispatch* ini dilakukan untuk setiap harga beban tertentu. Penyelesaian *economic dispatch* dapat dilakukan dengan beberapa cara yang akan dibahas pada sub bab di bawah ini.

2.3.1. *Economic Dispatch* Dengan Mengabaikan Rugi-Rugi Transmisi

Sistem tenaga listrik dengan mengabaikan rugi-rugi transmisi dapat dilihat pada gambar 2.5. Sistem ini memperlihatkan pembangkit termal yang terdiri atas N buah unit yang dihubungkan pada sebuah bus bar untuk melayani total beban sebesar P_D. Masukan untuk setiap unit ke-i adalah F_i yang menyatakan tingkat biaya bahan bakar dari masing-masing unit, dan keluaran dari masing-masing unit P_i adalah daya listrik yang dibangkitkan oleh tiap-tiap unit [2]. Total biaya dari sistem ini adalah penjumlahan dari biaya masing-masing unit. Kendala pokok dari operasi sistem ini adalah penjumlahan *output* daya harus sebanding dengan permintaan beban.

Biaya total F_T yang ditanggung sistem adalah jumlah biaya dari tiap-tiap unit pembangkit. Dan batasan yang paling penting dari pengoperasian pembangkit termal tersebut adalah daya listrik yang dihasilkan harus sama dengan besarnya beban konsumen.



Gambar 2.6. N Unit Pembangkit termis Melayani Beban P_D [2]

Yang menjadi permasalahan adalah meminimumkan total biaya F_T dengan memperhatikan pembatas φ bahwa daya dihasilkan unit pembangkit sama dengan yang diterima beban. Dengan catatan bahwa rugi-rugi transmisi diabaikan dan batas-batas operasi tidak secara tegas ditetapkan ketika merumuskan masalah ini.

Secara matematis pernyataan tersebut diatas dapat dinyatakan dengan persamaan berikut [2] :

$$F_T = F_1 + F_2 + \ldots + F_N$$

$$= \sum_{i=1}^{N} F_i(P_i) \quad \ldots\ldots (2.2)$$

Dan daya listrik yang dihasilkan oleh setiap unit untuk melayani beban total adalah :

$$P_D = \sum_{i=1}^{N} P_i \quad \ldots\ldots (2.3)$$

$$P_D - \sum_{i=1}^{N} P_i = \varphi = 0 \quad \ldots\ldots (2.4)$$

dimana P_D = kebutuhan beban, dan

P_i = jumlah daya yang dihasilkan

2.4. Kendala-Kendala Dalam Sistem Tenaga Listrik

2.4.1. Kendala Operasi Pada PLTU

Dari segi operasi PLTU paling banyak kendalanya, khususnya dalam kondisi dinamis. Hal ini disebabkan karena banyaknya komponen dari PLTU yang harus diatur. Dalam pengoperasiannya PLTU membutuhkan *starting time* yang cukup lama, disamping itu perubahan daya persatuan waktu sangat terbatas, yang mana kira-kira 5% per menit. Perubahan beban pada PLTU akan memaksa *gavernor* untuk melakukan penambahan atau pengurangan uap yang dialirkan ke turbin uap. Hal ini harus pula diikuti dengan penambahan atau pengurangan aliran air ke ketel, bahan bakar dan udara. Ini menunjukkan bahwa pengaturan beban unit PLTU akan menyangkut suatu sistem kontrol yang panjang sehingga sangat membatasi kemampuannya untuk menghadapi perubahan beban.

Mengingat kendala tersebut diatas, maka dalam pengoperasiannya PLTU harus memperhatikan hal-hal sebagai berikut [4]:

a. Beban maksimum

Dalam keadaan normal beban maksimum dari PLTU adalah sesuai dengan yang tercantum dalam buku spesifikasi teknis unit pembangkit. Dalam spesifikasi tersebut umumnya disebutkan beban maksimum untuk pembebanan yang *kontinyu* dan beberapa beban maksimum untuk waktu tertentu.

b. Beban minimum

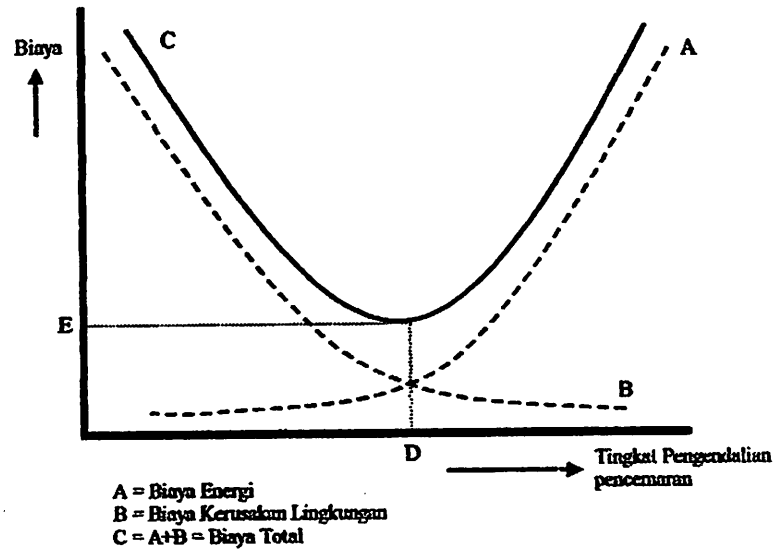
Beban minimum dari PLTU berkisar pada 25% daya maksimumnya. Pembatasan ini berhubungan dengan kontrol, karena pada beban rendah banyak yang hubungannya tidak linier sehingga menyulitkan kerjanya alat-alat kontrol. Misalnya hubungan antara suhu gas pembakaran dengan bahan bakar pada beban rendah tidak sama dengan beban tinggi. Pada PLTU campuran (*dual fuel firing*) bahan bakar minyak dan batubara, yang kalau bebannya kurang dari 25% tidak dapat beroperasi dengan menggunakan batubara, hal ini ada kaitannya dengan teknik pembakaran dalam ruang bakar ketel uap.

c. Kecepatan perubahan beban

Kecepatan perubahan beban pada unit PLTU harus menurut instruksi manual dari pabrik, karena perubahan beban akan memberikan dampak seperti yang telah diuraikan diatas. Kecepatan perubahan beban yang mampu dilakukan oleh unit PLTU tergantung pula pada posisi beban permulaan dalam kaitannya dengan sistem bahan bakar dan pengisian air ketel.

2.4.2. Biaya dan Pengendalian Pencemaran

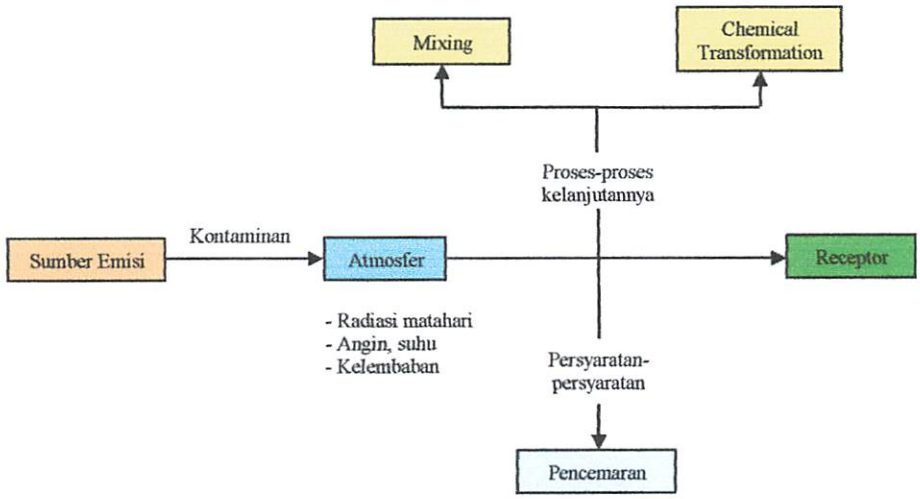
Penentuan biaya bahan bakar sebagai fungsi tingkat pengendalian pencemaran diperlihatkan pada gambar 2.7. Terlihat pada gambar tersebut terdapat tiga lengkung. Lengkung A merupakan biaya energi sebagai fungsi dari tingkat pengendalian pencemaran. Bilamana legislasi suatu negara dalam bidang pencemaran lingkungan hidup lunak atau kecil tingkat pengendaliannya, biaya untuk mengolah sumber daya energi juga kecil dan diperoleh harga energi yang rendah. Lebih ketat pengaturan pengendalian, lebih tinggi biaya bentuk energi yang bersangkutan. Lengkung B merupakan biaya kerusakan yang dialami oleh lingkungan hidup sebagai fungsi dari tingkat pencemaran. Jika tingkat pengaturan penjagaan kelestarian lingkungan di suatu negara rendah, maka banyak kerusakan yang akan dialami lingkungan hidup sebagai akibat pengelolaan suatu sumber energi. Dilain pihak, lebih tinggi tingkat pengendalian tersebut, lebih kecil kerusakan yang akan dialami oleh alam lingkungan. Kerusakan lingkungan menurut lengkung B dengan sendirinya dinyatakan sebagai biaya. Lengkung C merupakan penjumlahan dari lengkung A dan lengkung B. Pada lengkung C terdapat suatu minimum E pada tingkat pengendalian D [1].



Gambar 2.7. Biaya Sebagai Fungsi Dari Pengendalian Pencemaran[1]

2.4.3. Kendala Lingkungan

Penggunaan energi fosil banyak menimbulkan masalah lingkungan, dari tahap perolehannya, penyimpanannya dan penggunaannyauntuk membangkitkan energi listrik yang banyak menimbulkan limbah dengan beragam bentuk fisik dan komposisi kimia yang dapat mengganggu kelestarian lingkungan hidup. Dilain pihak, energi diperlukan oleh manusia dalam jumlah yang banyak dan pada saat yang sama, manusia memerlukan suatu lingkungan hidup yang baik untuk hidup secara sehat. Emisi merupakan salah satu dampak negatif dari pembakaran batubara. Emisi adalah segala bahan padat, cair dan gas yang masuk kedalam udara yang dalam kadar tertentu menimbulkan efek ketidaknyamanan. Pada dasarnya proses pencemaran terdiri atas tiga komponen yang saling berhubungan. Ketiga komponen itu ialah, sumber pencemar (emisi), media transmisi (udara) dan penerima (*receptor*) yang secara skematis dapat digambarkan sebagai berikut [1] :



Gambar 2.8. Proses Penyebaran Emisi Dari Sumber Sampai Ke *Receptor*[1]

2.4.3.1. Oksida Nitrogen (NO_x)

Oksida Nitrogen terdapat dalam bentuk: *Nitrogen Oksida* (NO), *Nitrous Oksida* (N_2O), *Nitrogen Dioksida* (NO_2), *Nitrogen Trioksida* (NO_3). Dari semua bentuk diatas, hanya NO dan NO_2 yang signifikan secara buatan membentuk oksida, sehingga oksida nitrogen (NO_x) sering diidentikkan dengan NO dan NO_2[1].

NO dihasilkan dari reaksi antara nitrogen dan oksigen pada temperatur tinggi dari pembakaran semua bahan bakar fosil. Pembentukannya tergantung dari temperatur pembakaran, lama waktu proses pembakaran, dan konsentrasi oksigen selama pembakaran. Walaupun hanya sebagian kecil NO yang lebih lanjut beroksidasi menjadi NO_2, namun di atmosfer NO secara cepat beroksidasi menjadi NO_2. Proses ini distimulasi oleh efek *photochemical* (kehadiran sinar *ultraviolet* matahari) dan materi organik dalam udara (katalis).

$$N_2 + O_2 \longrightarrow 2NO$$

$$2NO + O_2 \longrightarrow 2NO_2$$

Ada dua sumber nitrogen yang berkombinasi dengan oksigen untuk membentuk NOx. Sumber pertama adalah nitrogen dalam udara yang menghasilkan emisi yang disebut termal NOx. Sumber kedua adalah nitrogen dalam bahan bakar yang menghasilkan emisi yang disebut *fuel* NOx. Total NOx yang diproduksi selama pembakaran adalah penjumlahan termal NOx dan *fuel* NOx.

2.4.3.2. Strategi Pengendalian Emisi

Cara yang paling efisien dan secara langsung mengendalikan emisi hasil pembakaran batubara adalah dengan menggunakan jenis batubara bersih (*clean coal*) atau batubara berkadar nitrogen dan sulfur rendah.

Pengendalian terhadap partikulat dapat dilakukan dengan menggunakan *ESP* (*electrostatic precipitator*), *cyclone collector*, *fabric filter*, *bag-house collector*, *mechanical dust collector*, *scrubber* (*wet* atau *dry scrubber*). Dimana penggunaan alat-alat tersebut diatas dapat sejalan dengan alat pengendali emisi seperti *Flue Gas Desulfurization* (FGD) untuk mengendalikan SO_2, *Separate Over Fire Air* (SOFA) dan *Combine Over Fire Air* (COFA) untuk mengendalikan CO_2 dan NOx. Pengaturan operasi proses pembakaran juga dapat mengurangi NOx, yakni operasi pada *excess* udara yang rendah untuk mengurangi konsentrasi O_2 dan N_2, menginjeksikan air atau uap atau memasukkan kembali (*resilculating*) gas buang (*flue gas*) untuk menurunkan temperatur api (*flame*), pembakaran

bertingkat (*staged combustion*) untuk mengurangi *excess* udara dan temperatur pembakaran, dan pemilihan desain *burner* yang lebih maju dan rendah NOx [1].

Diperlukan suatu perangkat regulasi sebagai umpan balik terhadap sistem pengendalian manajemen polusi udara, salah satunya dengan menetapkan standar polusi udara atau baku mutu emisi khususnya dari sumber tidak bergerak (*stationary source*). Secara umum ada dua standar polusi udara, yakni: standar kualitas ambient udara atmosfer (*ambient air quality standart-AAQS*) dalam $\mu g / m^3$ dan standar sumber produksi (*source production standart-SPS*) dalam massa per unit waktu. Di Indonesia telah ditetapkan Keputusan Menteri Negara Lingkungan Hidup No. 13 Tahun 1995 tanggal 7 Maret 1995 Tentang: Baku Mutu Emisi Sumber Tidak Bergerak, seperti pada tabel 2.1.

Tabel 2.1.
Baku Mutu Emisi Untuk Pembangkit Listrik Tenaga Uap
Berbahan Bakar Batubara (Berlaku Efektif Tahun 2000)

No	Parameter	Batas Maksimum (mg/m^3)
1	Sulfur Oksida (SOx)	750
2	Nitrogen Oksida (NOx)	850
3	Total Partikel	150
4	Opasitas	20%

Sumber: Keputusan Menteri Lingkungan Hidup No. 13 Tahun 1995

Juga dikeluarkan baku mutu emisi dalam Keputusan Menteri Negara Kependudukan dan Lingkungan Hidup No. 02 Tahun 1988 tanggal 19 Januari 1988 yang diperuntukkan bagi *pilot project* dalam manajemen polusi udara pembangkit listrik tenaga uap berbahan bakar batubara yang dikelola pihak swasta seperti pada tabel 2.2.

Tabel 2.2. Standar Emisi Udara

No	Emission (kg/day)/ Flue Gas Temperature (°C)	70 C (U 5 & 6)	42 C (U 7 & 8)
1	SOx	5266	5064
2	NOx	36920	35500
3	Total Partikel	13361	12848
4	Opasitas	45760	44000

Sumber: Keputusan Menteri Lingkungan Hidup No. 02 Tahun 1988

BAB III

APLIKASI *REFINED GENETIC ALGORITHM* UNIT PEMBANGKIT TERMAL DENGAN MEMPERHATIKAN KENDALA EMISI

3.1. Teori Dasar *Genetic Algorithm*

3.1.1. Konsep Dasar

Genetic Algorithm merupakan metode adaptif yang bisi digunakan untuk memecahkan suatu pencarian nilai dalam sebuah masalah optimasi. Algoritma ini didasarkan pada proses genetic yang ada dalam makhluk hidup, yaitu perkembangan generasi dalam sebuah populasi yang alami, secara lambat laun mengikuti prinsip seleksi alam " siapa yang kuat, dia yang bertahan (*survive*)" [3]. Dengan meniru proses ini, *Genetic Algorithm* dapat digunakan untuk mencari solusi permasalahan-permasalahan dalam dunia nyata.

Genetic Algorithm ditemukan oleh John Holland pada awal tahun 1970 yang dilandasi oleh sifat-sifat evolusi alam [3]. Holland percaya bahwa ini sangat cocok digabungkan dalam sebuah algoritma komputer, menghasilkan sebuah teknik penyelesaian untuk permasalahan-permasalahan yang sulit dengan langkah alami, yaitu melalui *evolusi*. John Holland mulai bekerja dengan algoritma yang dibentuk dari *strings biner* 1 dan 0 yang disebut kromosom. Seperti halnya alam, Algoritma ini menyelesaikan permasalahan-permasalahan dengan menemukan kromosom-kromosom yang baik dengan memanipulasi materi dan sifat (gen) kromosom-kromosom. Algoritma ini tidak mengetahui tipe permasalahan yang akan diselesaikan. Hanya informasi yang telah diberikan dari evaluasi berupa nilai *fitness* setiap kromosom dengan nilai *fitness* terbaik yang bertahan hidup dan selalu diproduksi.

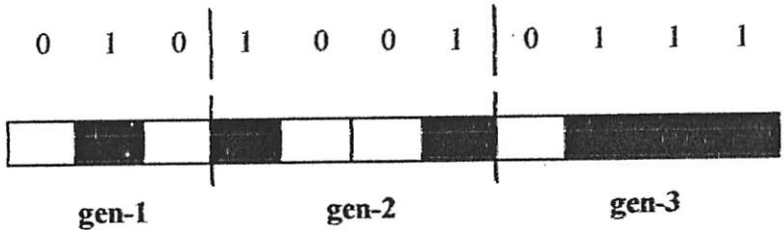
Sebelum *Genetic Algorithm* dijalankan, maka sebuah kode yang sesuai (representasi) untuk persoalan harus dirancang. Titik solusi dalam ruang permasalahan dikodekan dalam bentuk kromosom/*string* yang terdiri dari komponen genetik terkecil yaitu gen. Pemakaian bilangan seperti *integer*, *floating point* dan abjad sebagai *allele* (nilai gen) memungkinkan penerapan operator genetika yaitu proses produksi (*reproduction*), pindah silang (*crossover*), mutasi (*mutation*) untuk menciptakan himpunan titik-titik solusi [3]. Untuk memeriksa hasil optimasi, kita membutuhkan fungsi *fitness* yang menandakan gambaran hasil (*solution*) yang sudah dikodekan. Selama proses, induk harus digunakan untuk reproduksi, pindah silang dan mutasi untuk menciptakan keturunan (*offspring*). Jika *Genetic Algorithm* didesain dengan baik, populasi akan mengalami konvergensi dan akan mendapatkan sebuah solusi yang optimum.

3.1.2. Komponen-Komponen Utama *Genetic Algorithm*

3.1.2.1. Teknik Penyandian

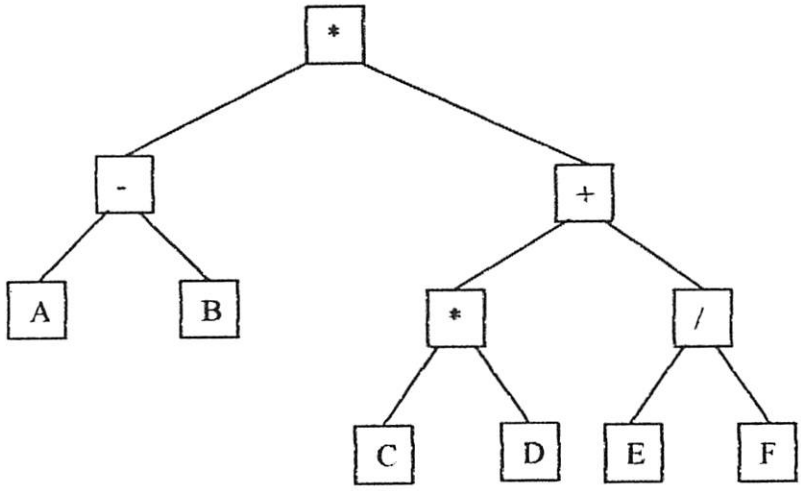
Teknik penyandian disini meliputi penyandian gen dari kromosom. Gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variabel. Gen dapat dipresentasikan dalam bentuk *string bit* dan pohon atau representasi lainnya yang dapat diimplementasikan untuk operator genetika [3]. Gambar 3.1 menunjukkan representasi *string bit* dan pohon.

String Biner



Pohon

(*(-(AB))(+(*(CD))(/(EF))))



Gambar 3.1. Representasi *String Bit* dan Pohon [3]

3.1.2.2. Prosedur Inisialisasi

Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut [3]. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

3.1.2.3. Fungsi Evaluasi

Ada dua hal yang harus dilakukan dalam melakukan evaluasi kromosom, yaitu evaluasi fungsi objektif (fungsi tujuan) dan konversi fungsi objektif ke dalam fungsi *fitness* [3]. Secara umum, fungsi *fitness* diturunkan dari fungsi objektif dengan nilai yang tidak negatif. Apabila ternyata fungsi objektif memiliki nilai negatif, maka perlu ditambahkan suatu konstanta c agar nilai *fitness* yang terbentuk menjadi tidak negatif.

3.1.2.4. Seleksi

Seleksi ini bertujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling *fit* dengan menggunakan metode seleksi dari induk yaitu *Roulette Whel Selection* [3]. Fungsi *fitness* meliputi kemampuan untuk membandingkan solusi dari satu generasi ke generasi yang lain. Nilai fungsi *fitness* [f(x)] berbanding lurus dengan kuadrat *string* kromosom (x^2) dan dirumuskan sebagai berikut :

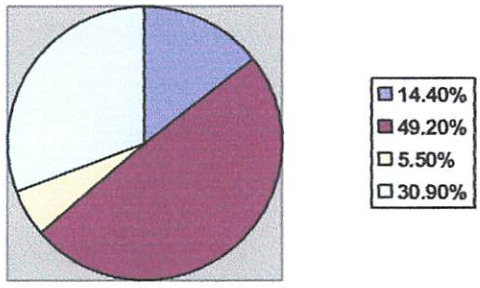
$$f(x) = x^2 \quad \text{.....} \quad (3.1)$$

Persamaan diatas dapat mengurangi genetika yang menyimpang dan akan menghasilkan kromosom yang sehat. Contoh penggambaran nilai *fitness* dapat ditunjukkan oleh tabel di bawah ini :

No	String	*Fitness*	% dari Jumlah
1	0 1 1 0 1	169	14.4
2	1 1 0 0 0	576	49.2
3	0 1 0 0 0	64	5.5
4	1 0 0 1 1	361	30.9
Jumlah		1170	100.0

Tabel 3.1. *String* dan Nilai *Fitness* [3]

Nilai fungsi *fitness* tidak dapat langsung dihubungkan dengan tujuannya, melainkan harus dirangking terlebih dahulu nilai tujuannya. Dengan cara ini, dapat ditentukan kromosom-kromosom mana yang layak digunakan dalam proses selanjutnya sehingga konvergensi awal dapat dihindari dan akan mempercepat penelitian ketika populasi mendekati konvergen.



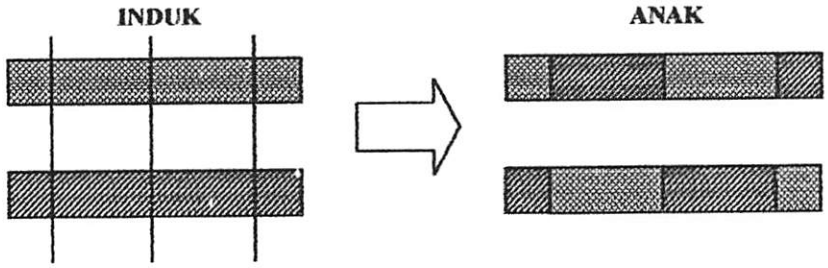
Gambar 3.2. Seleksi Dengan Menggunakan Metode *Roulette Wheel* [3]

3.1.2.5. Operator Genetika

Operator genetika yang akan digunakan untuk melakukan rekombinasi yaitu *crossover* banyak titik (*Multi Point Crossover*) untuk metode *Refined Genetic Algorithm*.

Multi Point Crossover [3]

Pada penyilangan banyak titik, m posisi penyilangan k_i (k = 1,2,...,n-1 ; i = 1,2,...,m) dengan n = panjang kromosom diseleksi secara random dan tidak diperbolehkan ada posisi yang sama, serta diurutkan naik. Variabel-variabel ditukar antar kromosom pada titik tersebut untuk menghasilkan anak (gambar 3.3).



Gambar 3.3. *Multi Point Crossover* [3]

Misalkan ada 2 kromosom dengan panjang 12 :

induk 1 : 0 1 1 1 0 0 1 0 1 1 1 0

induk 2 : 1 1 0 1 0 0 0 0 1 1 0 1

jumlah dan posisi titik penyilangan yang terpilih misalkan (m = 3) : 2, 6, 10; maka setelah penyilangan diperoleh kromosom-kromosom baru :

anak 1 : 0 1 | 0 1 0 0 | 1 0 1 1 | 0 1

anak 2 : 1 1 | 1 1 0 0 | 0 0 1 1 | 1 0

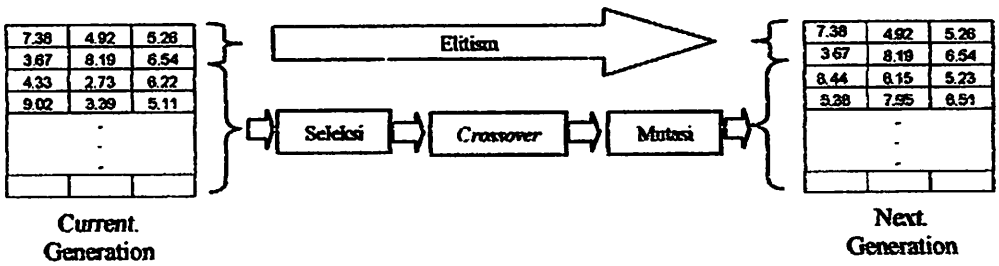
3.1.2.6. Penentuan Parameter

Yang dimaksud parameter yaitu parameter kontrol *genetic algorithm*, yaitu ukuran populasi (*popsize*), peluang *crossover* (Pc), dan peluang mutasi

(Pm)[3]. Pada metode *Refined Genetic Algorithm* peluang *crossover* dan peluang mutasi selalu diubah pada setiap generasi.

3.2. *Refined Genetic Algorithm* Dengan *Elitism*

Selama membuat populasi baru dengan *crossover* dan mutasi, kemungkinan akan terjadi kehilangan kromosom terbaik (*best / few best*). *Elitism* adalah metode yang pertama kali meng-*copy*-kan kromosom terbaik (*best / few best*) kedalam populasi baru [8]. Sisanya dikerjakan dengan cara biasa, yaitu melalui seleksi, *crossover* dan mutasi. *Elitism* dapat secara cepat meningkatkan performansi dari *Refined Genetic Algorithm* karena *elitism* menghindarkan hilangnya solusi terbaik (*best / few best*) yang telah ditemukan. Ilustrasi kerja operator ini dapat digambarkan seperti pada gambar 3.4.



Gambar 3.4. Pembentukan *Next Generation* dalam *Refined Genetic Algorithm*[8]

Hal ini juga membantu, menggambarkan metode yang mungkin untuk menghitung "titik berhenti (*stopping point*)" untuk program. Karena *elitisme* menyimpan rangkaian yang paling cocok dari tiap-tiap populasi, program-program mampu untuk secara cepat menemukan dan menyimpan solusi terbaik

pada masalah. Ketika program bertemu, program menghasilkan kriteria berhenti yang alami untuk program.

3.3. Teori Dasar *Refined Genetic Algorithm* Dengan Memperhatikan Emisi

Biaya produksi untuk pembangkit listrik utamanya terbentuk dari biaya bahan bakar, biaya penanganan batubara serta biaya pemeliharaan operasi. Pendekatan *Standar Economic Dispatch* (SED) telah berhasil diaplikasikan untuk meminimkan biaya produksi. Namun demikian strategi berbasis-ekonomi ini seringkali menghasilkan tingkat emisi NO_x yang tinggi. *Clean Air Act* 1990, mencerminkan semakin besarnya perhatian publik terhadap lingkungan, mengamanatkan bahwa dampak konvensional harus dimasukkan ke dalam sistem *costing* (penentuan biaya) mereka [8].

Batas-batas emisi diciptakan untuk memungkinkan memperdagangkan kelonggaran tersebut. Utilitas yang memiliki kelonggaran emisi NO_x diatas standar dapat menjual kelonggaran yang tidak mereka gunakan kepada utilitas yang tidak memenuhinya. Dengan memberikan fleksibilitas dalam memenuhi ketentuan penggunaan tersebut, maka ketentuan dalam Undang-Undang membolehkan pemenuhan didasarkan pada penggunaan rendah dari unit yang terpengaruh dengan mengorbankan kelonggaran emisi di akhir tahun. Ketentuan ini disebut dengan 'kompensasi pilihan generasi'. Kelonggaran emisi yang disampaikan ditentukan oleh jumlah penggunaan rendah dari unit yang dikalikan bobot tingkat emisi Mbtu pada unit ini.

3.4. Formulasi Masalah *Economic Dispatch* Di Bawah Perencanaan Kompensasi Generasi

Pilihan kompensasi generasi yang disediakan memungkinkan *underutilisasi* dari unit yang terpengaruh dengan mengorbankan penyerahan kelonggaran/batas-batas NOx didasarkan pada jumlah *underutilisasi* dari unit ini. Ketika utilitas tertarik dalam meminimkan biaya bersih, maka problem *Combined Economic Dispatch And Emission Dispatch* (CEED) dapat diekspresikan dengan persamaan [8] :

$$\text{minimal } \varnothing_t = \sum_{k=1}^{M} F_{tk} + g\sum_{k=1}^{M} E_{tk} \quad \text{.....(3.2)}$$

dimana:

F_{tk} : harga bahan bakar ($/Mbtu)

E_{tk} : tingkat emisi NOx (ton/Mbtu)

$\varnothing_t$: biaya operasi total dari sistem; (biaya bahan bakar sistem + kendala emisi)

g : faktor denda harga (*penalty factor*)

Ditujukan untuk :

1. Batas-batas kapasitas pembangkitan :

$$P_i^{min} \le P_{tk} \le P_i^{mak} \quad \text{.....(3.3)}$$

k = 1,2,....,M

2. Batas-batas permintaan :

$$P_{load} = \sum_{k=1}^{M} P_{tk} - P_{loss} \quad \text{.....(3.4)}$$

k = 1,2,....,M

Awalnya, rugi-rugi daya pada saluran transmisi diabaikan (P_{loss}).

dimana:

M : jumlah interval dispatch

k : indeks interval dispatch

i : indeks tiap unit

P_{load} : beban sistem (MW)

P_{ik} : daya yang dibangkitkan (MW)

P : vektor dengan komponen $P_{ik}, i \in \Phi k$, k = 1,2,....,M

$P_i^{min}(P_i^{max})$: limit generasi bawah (atas)(MW)

3.5. Implementasi *Refined Genetic Algorithm* Kaitannya dengan Minimum Emisi Dalam Optimasi Sistem Tenaga Listrik

Pergerakan kepada *Refined Genetic Algorithm* (RGA) khusus dicapai dengan menggunakan tiga operator primer : *reproduksi*, *crossover* (*persilangan*) dan *mutation*. Dalam sebuah contoh siklus eksekusi, operator reproduksi memilih sebuah *string* dari generasi sebelumnya didasarkan pada *string fitness* dan probabilitas perkembangbiakan pada generasi berikutnya. Reproduksi akan terus terjadi sampai generasi berikutnya terisi. Teknik seleksi turnamen dengan ukuran 2 digunakan dalam proses reproduksi. Operator *crossover* bekerja dalam konjungsi dengan reproduksi. Sebuah persilangan sederhana memilih posisi didalam dua *string* induk dan menukar informasi gen dari posisi tersebut pada akhir *string*. Operator mutasi mengubah secara *random* selama fase persilangan

reproduksi untuk meningkatkan keragaman ruang sample. Probabilitas dari persilangan dan kejadian mutasi dapat dipilih oleh *user* [8].

Seleksi parameter eksekusi adalah penting agar *Refined Genetic Algorithm* mampu menyelidiki seluruh ruang solusi. Populasi awal biasanya diciptakan dengan menggunakan generator angka *random* dan serangkaian generasi terbentuk dari informasi populasi awal.

Encoding

Skema *encoding* menggunakan *alfabet biner* [0,1] masing-masing kromosom/*string* berisi *encoding* dari solusi untuk unit generasi. Dalam memasukkan tekanan pada fungsi biaya sebagai bentuk pinalti, teknik pemetaan *non linier* digunakan dalam proses dekode [8].

Metode membaca kode (*decoding*) dirumuskan dalam persamaan dibawah :

$$value = bit_0 \times 2^0 + bit_1 \times 2^1 + ... + bit_i \times 2^i + ... + bit_{chrom\_length} \times 2^{chrom\_length}$$

Jika parameter yang dioptimalisasi termasuk ($P_{i\ mak}$, $P_{i\ min}$), nilai membaca kode dari parameter dihitung dengan persamaan berikut :

$$P_{ik} = P_{i\ min} + \frac{(value \times (P_{i\ mak} - P_{i\ min}))}{2^{chrom\_length} - 1} \quad \text{.....(3.5)}$$

P_{ik} adalah nilai dekode final untuk unit *output*. Ini digunakan dalam evaluasi *fitness*.

Fungsi *Fitness*

Karena masalah *CEED* adalah masalah minimisasi dan tekanan perlu dimasukkan dalam fungsi sasaran, maka dua langkah berikut dibutuhkan untuk mencapai evaluasi kesesuaian.

Pertama, fungsi objektif didefinisikan dengan memasukkan seluruh pertidaksamaan sebagai bentuk pinalti dari fungsi biaya awal. Fungsi-fungsi pinalti sederhana dipilih untuk menghasilkan [8] :

$$F_{obj} = \sum_{k=1}^{M} F_i(p_{ik}) + g\sum_{k=1}^{M} E_{ik}(p_{ik}) \quad \text{..... (3.6)}$$

Dimana g adalah faktor penalti dan P_{ik} adalah *deviasi output* unit dari limitnya. Dengan batas-batas permintaan:

$$P_{load} - \sum_{k=1}^{M} P_{ik} = 0$$

Pada langkah kedua, nilai biaya fungsi objektif ditransformasi ke dalam nilai kesesuaian. Ketika kesesuaian kromosom/*string* akan dibandingkan dengan seluruh kesesuaian *string* lainnya pada populasi sama, maka optimalitas ukuran tidak akan dibutuhkan. Untuk satu generasi populasi, jika F_{obj} = PCi adalah nilai biaya objektif untuk *string* i, dan PC_{max} dimana *Max* {PCi, i = 1,2, ... , N_{POP}}, kemudian *fitness* untuk populasi ini didefinisikan sebagai berikut:

$$FIT_i = PC_{max} - PC_i,\ i = 1,2, \ldots , N_{POP} \quad \text{..... (3.7)}$$

Ketika fungsi *fitness* adalah ukuran mentah dari nilai solusi, dan kita ingin memperoleh solusi sesuai optimal yang meminimkan biaya aslinya, maka seleksi fungsi pinalti yang sesuai adalah sesuatu yang sangat penting.

BAB IV

ECONOMIC DISPATCH UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON

4.1. Pendahuluan

Masalah *economic dispatch* akan menjadi semakin rumit ketika dampak emisi terhadap lingkungan harus diperhatikan. Karakteristik emisi dari masing-masing polutan adalah berbeda. Hal ini menambah kompleksitas masalah *economic dispatch* untuk mendapatkan biaya bahan bakar yang ekonomis dalam suatu operasi unit-unit pembangkit termal pada beban tertentu, dengan tetap mempertimbangkan emisi yang dijaga agar tidak melebihi batas-batas yang telah ditetapkan Pemerintah, sehingga dampak negatif emisi yang dihasilkan dari pembakaran bahan bakar pembangkit listrik dapat dikurangi.

4.2. Validasi Program

4.2.1. Data Validasi Referensi Jurnal IEEE

Program divalidasikan dengan menggunakan data pada referensi jurnal "*Application of Refined Genetic Algorithms to Combined Economic and Emission Dispatch*" IE (Journal)-EL, Vol 85, September 2004.

Sistem terdiri dari 3 unit pembangkit dan beban sistem. Parameter *Refined Genetic Algorithm* pada referensi jurnal yaitu: jumlah populasi sebesar 20, jumlah generasi sebesar 100, probabilitas mutasi 0.001.

Tabel 4.1. Data Beban Validasi

NO	Beban
	MW
1	400
2	500
3	700

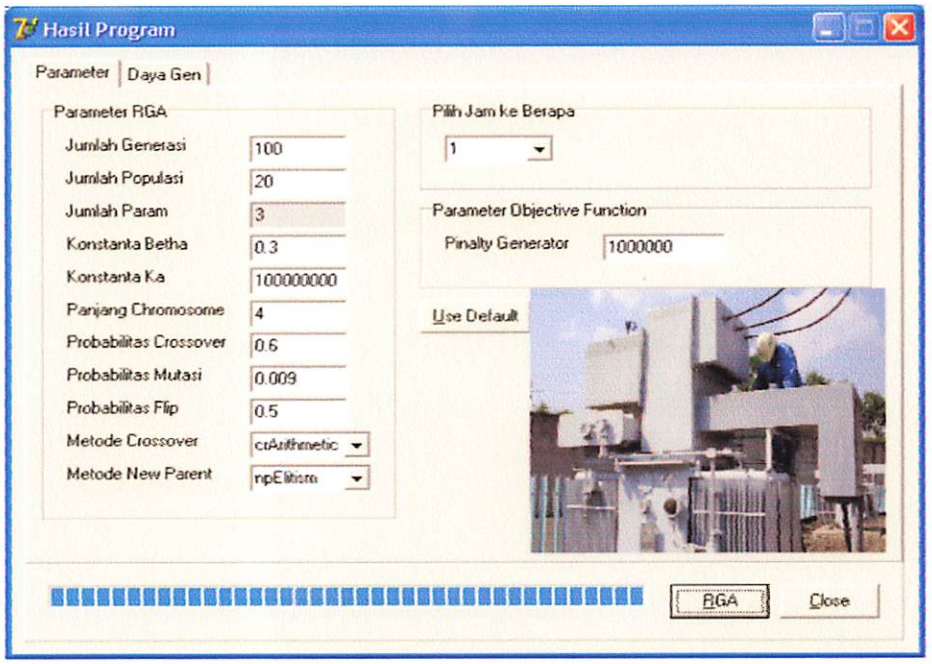
Tabel 4.2. Data Unit Pembangkit Validasi

No Unit	Daya Min. MW	Daya Max. MW	Tipe Pers	a2	a1	a0
1	35	210	Fuel	0.03546	38.30553	1243.53110
			NOx	0.00683	-0.54551	40.26690
2	130	325	Fuel	0.02111	36.32782	1658.56960
			Nox	0.00461	-0.51160	42.89553
3	125	315	Fuel	0.01799	38.27041	1356.65920
			Nox	0.00461	-0.51160	42.89553

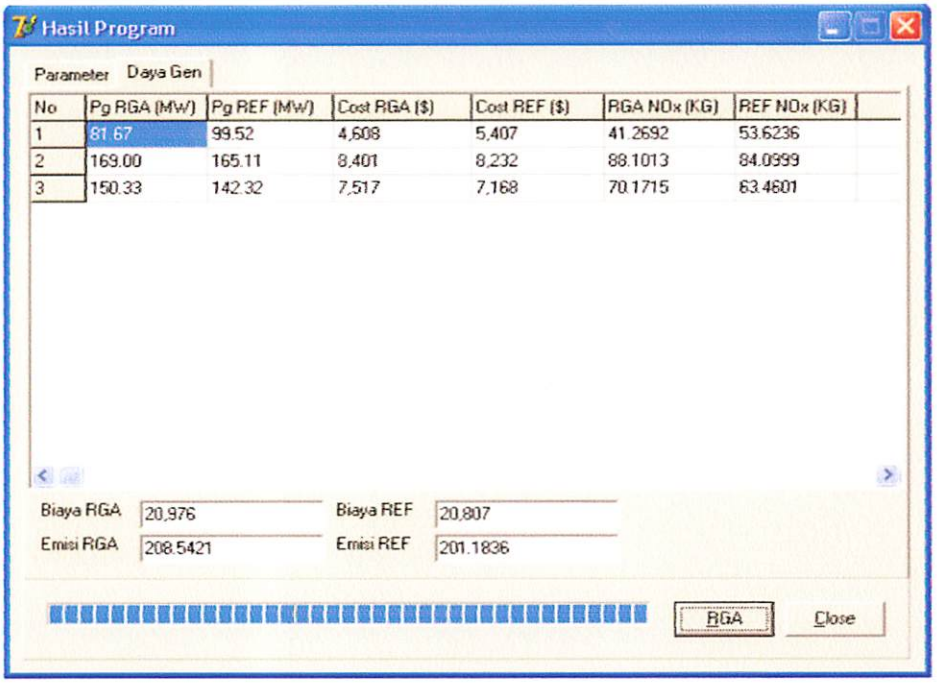
Sumber: "*Application of Refined Genetic Algorithms to Combined Economic and Emission Dispatch*" IE (Journal)-EL, Vol 85, September 2004.

4.2.2. Hasil Validasi Referensi Jurnal IEEE Dengan Menggunakan Metode *Refined Genetic Algorithm*

Program komputer menggunakan bahasa pemrograman *Borland Delphi* 7.0 dan diaplikasikan pada komputer dengan spesifikasi prosesor *AMD ATHLON* XP 2000 1,67 GHz, DDR SDRAM 256 MB dengan *Operation System* (*OS*) *Windows XP Professional*.



Gambar 4.1. Tampilan Parameter *Refined Genetic Algorithm*



Gambar 4.2. Tampilan Hasil Validasi Jurnal IEEE

Tabel 4.3.
Perbandingan Data Referensi Jurnal Dengan Hasil Program

Jam	Data Referensi			Hasil Program		
	Beban (MW)	NOx (KG)	Biaya ($)	Beban (MW)	NOx (KG)	Biaya ($)
1	400	201.184	20,807	400	208.542	20,976
2	500	311.162	25,489	500	313.277	25,437
3	700	651.421	35,469	700	635.324	34,864

Dari proses validasi diatas, didapatkan error rata-rata yaitu untuk output emisi NOx sebesar 0.005724 atau 0.57%, untuk biaya sebesar 0.0060042 atau 0.60%.

4.3. Algoritma Program Pemecahan Masalah *Economic Dispatch* Pembangkit Termal Menggunakan Metode *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi Pada PLTU Paiton

4.3.1. Algoritma Program

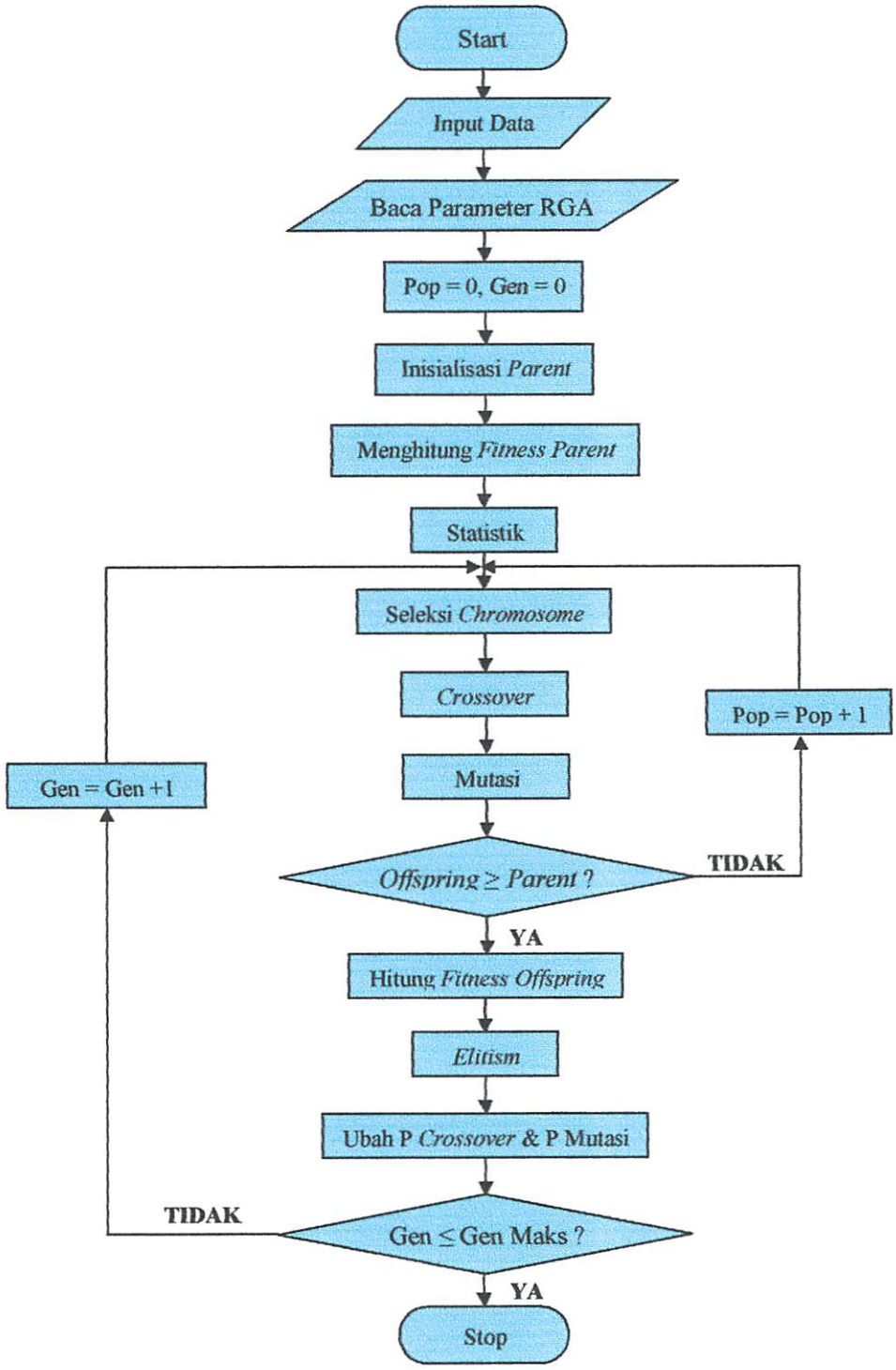
1. Masukkan inputan data parameter unit pembangkit termal Paiton berupa P_{maks}, P_{min}, koefisien biaya bahan bakar, koefisien emisi NOx dan data pembebanan tiap jam.
2. Masukkan input parameter *genetic algorithm* berupa Jumlah generasi, Jumlah populasi, Probabilitas *crossover*, Probabilitas mutasi, Konstanta *fitness*, *Penalty fitness*.
3. Menentukan populasi awal (Pop = 0) dan menentukan generasi awal (Gen = 0).
4. Menentukan inisialisasi *Parent*.
5. Menghitung *Fitness Parent*.

6. Melakukan proses Statistik.
7. Melakukan proses Seleksi *Chromosome*.
8. Melakukan proses *Crossover*.
9. Melakukan proses Mutasi.
10. Apakah *Offspring* lebih besar atau sama dengan jumlah *Parent*.
11. Jika "TIDAK", kembali proses Seleksi (langkah 7)
12. Jika "YA", menghitung nilai *Fitness Offspring*.
13. Melakukan proses Penambahan *Elitism*.
14. Mengubah P *Crossover* dan P Mutasi.
15. Apakah Generasi lebih kecil atau sama dengan maksimum Generasi.
16. Jika "TIDAK", Gen = Gen + 1, kembali ke proses Seleksi (langkah 7)
17. Jika "YA", perhitungan selesai.

4.3.2. Algoritma Fungsi *Fitness*

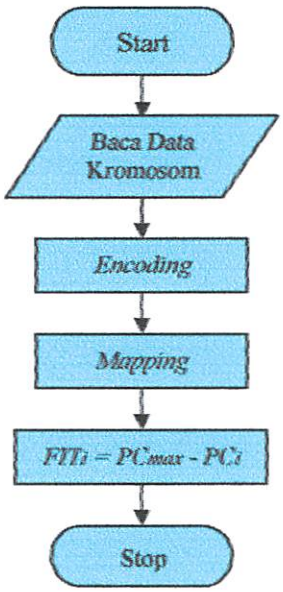
1. *Start*.
2. Baca Data Kromosom.
3. Melakukan proses *Encoding*.
4. Menghitung proses *Mapping*.
5. $FIT_i = PC_{max} - PC_i$.
6. *Stop*.

4.3.3. *Flowchart* Algoritma *Refined Genetic Algorithm*



Gambar 4.3. *Flowchart Refined Genetic Algorithm*

4.3.4. *Flowchart* Fungsi *Fitness*



Gambar 4.4. *Flowchart* Fungsi *Fitness*

4.4. Data Pembangkit Termal Pada Sub Sistem Paiton

Pembangkit termal yang ada di sub sistem Paiton ada 6 unit pembangkit listrik dengan bahan bakar batubara (PLTU). Adapun data-data lebih lengkapnya dapat dilihat pada tabel 4.4, untuk harga batubara Rp. 9000 per satu dolar Amerika.

Tabel 4.4.
Parameter Unit Pembangkit Termal

Nama Pembangkit	Bahan Bakar	Daya Terpasang (MW)	Pmin (MW)	Pmax (MW)	Koefisien Input-Output Pembangkit			Fuel Price (S/Mbtu)
					a_2	a_1	a_0	
PLTU Paiton 1	Coal	450	225	370	-0.0209	24.18767	-1985.53015	4.440
PLTU Paiton 2	Coal	450	225	370	-0.0184	22.46252	-1709.33811	4.355
PLTU Paiton 5	Coal	650	310	610	0.0029	5.93321	996.42342	2.799
PLTU Paiton 6	Coal	650	310	610	0.0018	7.21021	758.57413	2.820
PLTU Paiton 7	Coal	685	275	615	0.0117	-2.22107	3412.61128	4.186
PLTU Paiton 8	Coal	685	275	615	0.0021	7.04660	1107.90685	4.195

Catatan : Harga Batubara unit 1,2,7 dan 8 : 28.17 US $/ton
Harga Batubara unit 5 dan 6 : 19.75 US $/ton
Nilai tukar : 9000 Rp/US $

Dengan memasukkan data-data pada tabel 4.4. ke persamaan fungsi biaya bahan bakar, maka untuk unit pembangkit termal PLTU Paiton 1 diperoleh sebagai berikut :

$$Hi(Pi) = a_2 P_i^2 + a_1 P_i + a_0 \text{ (MBtu/jam)}$$

$$Fi(Pi) = Hi(Pi) \times (\$/\text{MBtu})$$

$$Fi(Pi) = \{(-0.0209).P_1^2 + (24.18767).P_1 + (-1985.53015)\} \times 4.44$$

$$= (-0.093).P_1^2 + (107.39326).P_1 + (-8815.75388)$$

Untuk persamaan biaya pembangkitan dari masing-masing unit pembangkit yang berdasarkan data-data diatas adalah sebagai berikut :

Tabel 4.5.
Persamaan Biaya Bahan Bakar PLTU Paiton

No	Nama Pembangkit	Persamaan Biaya Bahan Bakar (S/Jam)
1	PLTU Paiton Unit 1	$F = (-0.093) P^2 + (107.39326) P + (-8815.75388)$
2	PLTU Paiton Unit 2	$F = (-0.08) P^2 + (97.82428) P + (-7444.16747)$
3	PLTU Paiton Unit 5	$F = (0.008) P^2 + (16.60706) P + (2788.98915)$
4	PLTU Paiton Unit 6	$F = (0.005) P^2 + (20.33281) P + (2139.17904)$
5	PLTU Paiton Unit 7	$F = (0.049) P^2 + (-9.29738) P + (14285.19084)$
6	PLTU Paiton Unit 8	$F = (0.009) P^2 + (29.5605) P + (4647.66924)$

Untuk mengetahui seberapa besar efisiensi dari metode ini, maka dilakukan evaluasi dengan mengambil data unit PLTU Paiton sebagai bahan perbandingan.

Tabel 4.6.
Parameter Emisi NOx PLTU Paiton

Nama Pembangkit	Koefisien Emisi NOx			NOx Limit (Ton/h)
	a_2	a_1	a_0	
PLTU Paiton 1	0.000007	-0.003356031	0.404005948	2.2146
PLTU Paiton 2	0.000005	-0.002168669	0.245304934	2.2146
PLTU Paiton 5	0.000001	-0.000500016	0.317271496	1.5383
PLTU Paiton 6	0.000004	-0.002901878	0.881752405	1.5383
PLTU Paiton 7	0.000002	-0.001001355	0.413235015	1.4792
PLTU Paiton 8	0.000004	-0.003021271	0.838348526	1.4792

4.5. Data Pembebanan Pembangkit Termal Sub Sistem Paiton

Proses *Economic Dispatch* dengan metode *Refined Genetic Algorithm* yang memperhatikan kendala emisi bertujuan untuk membuat rencana pembagian beban suatu sistem tenaga listrik yang dapat memenuhi kebutuhan beban dengan biaya operasi yang ekonomis dan menghasilkan emisi dari hasil pembakaran bahan bakar yang tidak melampaui batas yang telah ditentukan.

Untuk mengetahui efisiensi metode ini, maka dilakukan evaluasi dengan mengambil data unit pembangkit termal dan beban yang ditanggung oleh PLTU Paiton (sub sistem Paiton) sebagai bahan perbandingan. Perhitungan dan analisa dilakukan pada karakteristik kurva beban yang berbeda, yakni :

- Tanggal 8 Februari 2004 adalah beban pada hari libur (Minggu).
- Tanggal 9 dan 10 Februari 2004 adalah beban pada hari kerja penuh (Senin dan Selasa).

Tabel 4.7.
Data Beban Sistem PLTU Paiton

JAM	Minggu 8 Februari 2004	Senin 9 Februari 2004	Selasa 10 Februari 2004
	Beban Sistem (MW)	Beban Sistem (MW)	Beban Sistem (MW)
00.00	2,568.705	2,509.61	2,510.54
01.00	2,447.279	2,300.00	2,390.21
02.00	2,419.617	2,168.72	2,930.14
03.00	2,335.950	2,137.54	2,375.93
04.00	2,293.225	2,136.38	2,362.72
05.00	2,287.977	2,208.71	2,394.08
06.00	2,131.770	2,103.91	2,220.61
07.00	1,824.395	1,955.76	2,039.15
08.00	1,837.199	2,234.28	2,339.06
09.00	1,835.547	2,671.79	2,625.30
10.00	1,836.043	2,771.94	2,779.04
11.00	1,827.974	2,770.70	2,773.78
12.00	1,795.155	2,652.41	2,625.92
13.00	1,833.428	2,584.09	2,580.73
14.00	1,827.045	2,757.04	2,764.43
15.00	1,835.399	2,709.13	2,786.33
16.00	1,915.738	2,746.23	2,776.95
17.00	2,137.235	2,774.73	2,785.83
18.00	2,567.636	3,004.53	2,816.35
19.00	2,940.322	3,167.18	3,097.60
20.00	2,969.909	3,158.37	3,159.44
21.00	2,964.424	3,169.66	3,168.49
22.00	2,922.564	2,948.93	2,977.80
23.00	2,625.662	2,700.97	2,709.82
TOTAL	**53,980.200**	**62,342.61**	**63,450.23**

Sumber : Data Total Pembebanan Unit Pembangkit Termal Sub PLTU Paiton
Jl. Raya Surabaya-Situbondo Km. 142, Probolinggo 67291

4.6. Data Emisi NOx Pembangkit Termal Sub Sistem Paiton

Untuk mengetahui seberapa besar emisi yang dihasilkan dari pembakaran yang dilakukan untuk menghasilkan energi listrik, maka dilakukan evaluasi dengan mengambil data emisi unit pembangkit termal yang terjadi di PLTU Paiton sebagai bahan perbandingan. Untuk data emisi sistem terdapat pada tabel 4.8.

Tabel 4.8.
Data Emisi (NOx) PLTU Paiton

JAM	Minggu 8 Februari 2004	Senin 9 Februari 2004	Selasa 10 Februari 2004
	NOx (TON)	NOx (TON)	NOx (TON)
00.00	1.8321	1.8535	1.9586
01.00	1.6867	1.6413	1.8674
02.00	1.6315	1.4773	1.6461
03.00	1.5090	1.3808	1.6135
04.00	1.4170	1.3380	1.5556
05.00	1.4307	1.3578	1.6181
06.00	1.4417	1.3837	1.5136
07.00	1.3775	1.5091	1.5944
08.00	1.3806	1.5623	1.5297
09.00	1.3470	1.6574	1.8463
10.00	1.3593	1.8912	1.9151
11.00	1.3218	1.9373	1.9626
12.00	1.3390	2.0119	1.9753
13.00	1.3716	2.0148	1.9570
14.00	1.3853	1.9550	1.9676
15.00	1.4381	1.8950	2.0635
16.00	1.5146	1.8931	2.0160
17.00	1.6388	1.9609	2.0732
18.00	1.6271	2.1134	2.0684
19.00	2.0160	2.2961	2.5563
20.00	2.1955	2.5482	2.8509
21.00	2.2568	2.6507	2.9900
22.00	2.2738	2.5752	2.7394
23.00	2.0844	2.0929	2.9290
TOTAL	**38.8759**	**44.9970**	**48.8075**

Sumber : Data Emisi NOx Unit Pembangkit Termal Sub PLTU Paiton
Jl. Raya Surabaya-Situbondo Km. 142, Probolinggo 67291

4.7. Prosedur Pelaksanaan Program Perhitungan

Prosedur menjalankan program perhitungan dilakukan setelah seluruh input data dimasukkan ke dalam program yang menggunakan bahasa pemrograman *Borland Delphi* 7.0. Prosedur jalannya program dapat dilakukan sebagai berikut :

1. Tampilan utama dari program *Economic Dispatch* unit pembangkit termal yang memperhatikan kendala emisi dengan menggunakan metode RGA pada sub sistem Paiton.

Gambar 4.5. Tampilan Utama Program

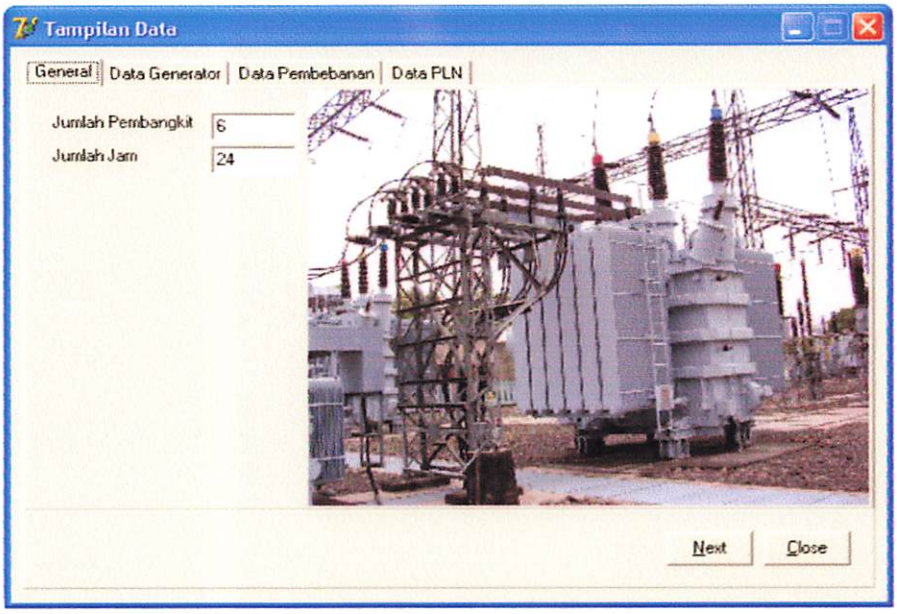
4.7. Prosedur Pelaksanaan Program Perhitungan

Prosedur menjalankan program perhitungan dilakukan setelah seluruh input data dimasukkan ke dalam program yang menggunakan bahasa pemrograman *Borland Delphi* 7.0. Prosedur jalannya program dapat dilakukan sebagai berikut :

1. Tampilan utama dari program *Economic Dispatch* unit pembangkit termal yang memperhatikan kendala emisi dengan menggunakan metode RGA pada sub sistem Paiton.

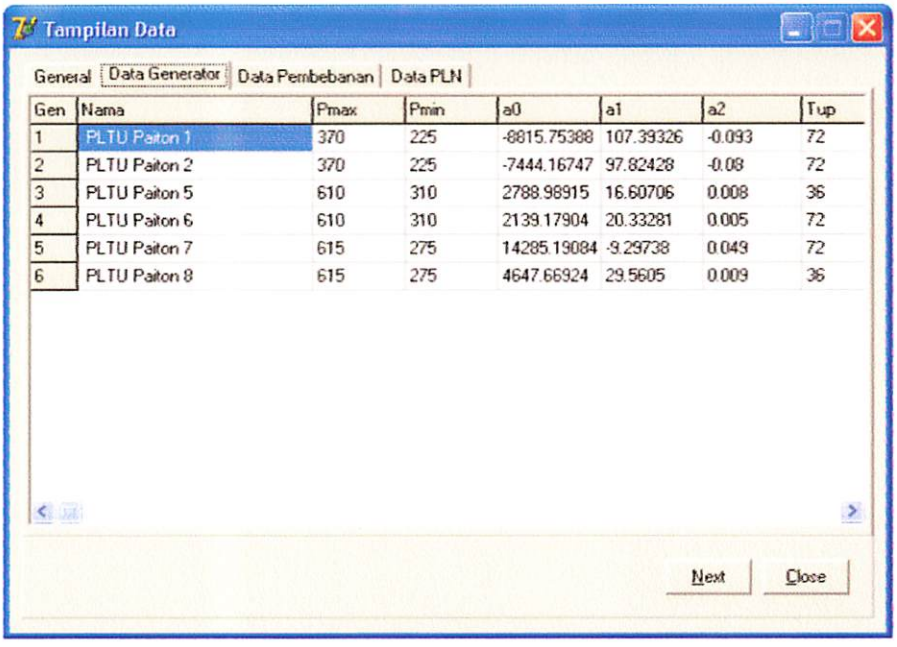
Gambar 4.5. Tampilan Utama Program

2. Tekan tombol *Open* untuk membuka data yang sudah tersimpan.



Gambar 4.6. Tampilan Masukan Data

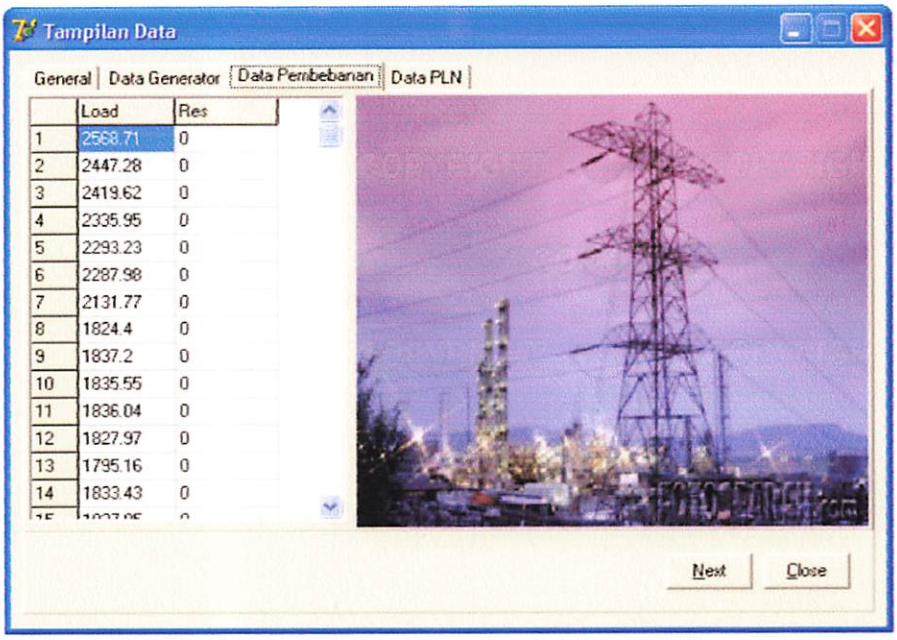
3. Kemudian tekan tombol Data Pembangkit.



Gen	Nama	Pmax	Pmin	a0	a1	a2	Tup
1	PLTU Paiton 1	370	225	-8815.75388	107.39326	-0.093	72
2	PLTU Paiton 2	370	225	-7444.16747	97.82428	-0.08	72
3	PLTU Paiton 5	610	310	2788.98915	16.60706	0.008	36
4	PLTU Paiton 6	610	310	2139.17904	20.33281	0.005	72
5	PLTU Paiton 7	615	275	14285.19084	-9.29738	0.049	72
6	PLTU Paiton 8	615	275	4647.66924	29.5605	0.009	36

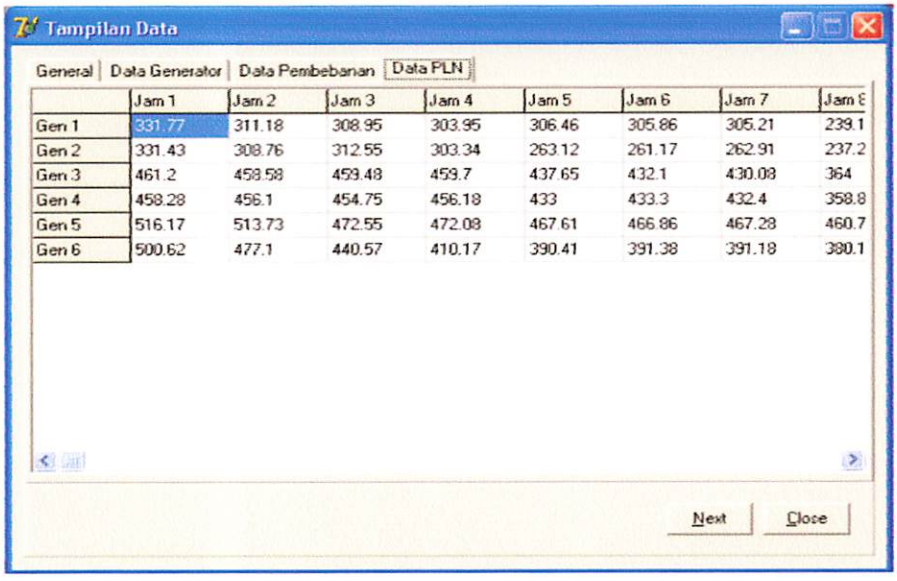
Gambar 4.7. Tampilan Data Pembangkit

4. Kemudian tekan tombol Data Pembebanan.



Gambar 4.8. Tampilan Data Pembebanan

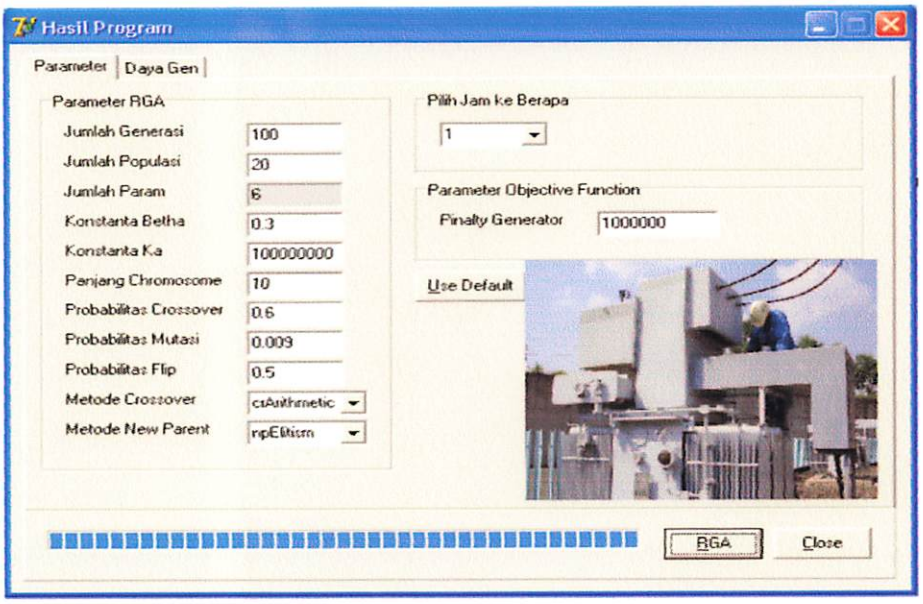
5. Kemudian tekan tombol Data PLN, kemudian tekan tombol *next*.



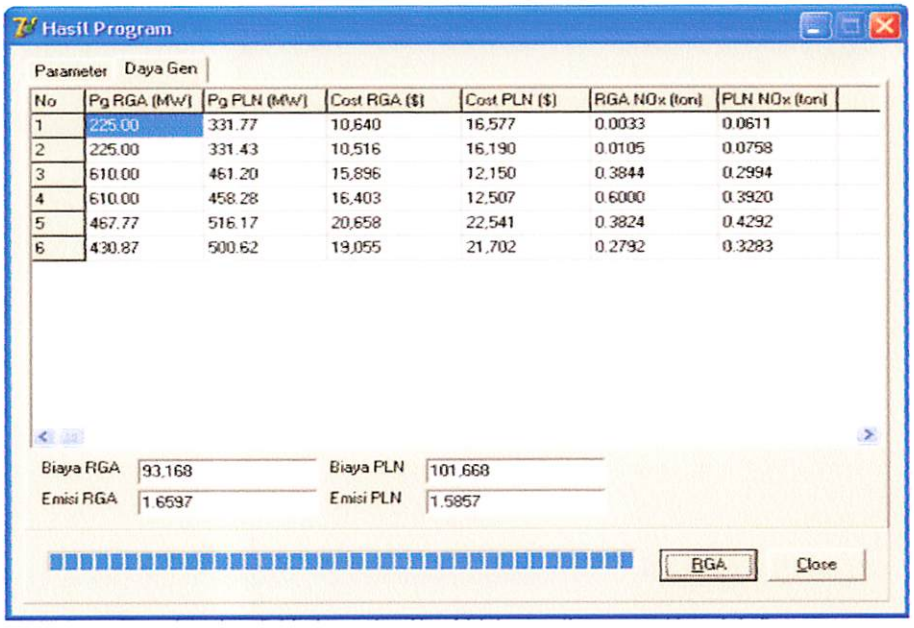
Gambar 4.9. Tampilan Data PLTU Paiton

6. Kemudian tekan tombol *default* untuk memasukkan parameter RGA.

Tekan tombol RGA, untuk menghitung optimasi *Economic Dispatch* yang memperhatikan kendala emisi.



Gambar 4.10. Tampilan Parameter *Refined Genetic Algorithm*



Gambar 4.11. Tampilan Hasil Optimasi *Economic Dispatch* Yang Memperhatikan Kendala Emisi

4.8. Hasil Perhitungan Dan Analisa Data Antara PLTU Paiton dan *Refined Genetic Algorithm*

4.8.1. Perbandingan Daya, Emisi NOx dan Biaya Operasi Tiap Unit Pembangkit

Setelah mendapatkan hasil yang optimal seperti yang sudah dijelaskan diatas, selanjutnya dilakukan perhitungan berapa besarnya daya yang dibangkitkan untuk mencukupi kebutuhan beban dengan biaya dan emisi (NOx) yang paling minimal pada unit-unit PLTU Paiton.Berikut ini adalah tabel-tabel yang merupakan perbandingan antara daya, emisi NOx dan biaya operasi per jam tiap unit pembangkit hasil perhitungan PLTU Paiton dan *Refined Genetic Algorithm* pada tabel 4.9 sampai dengan 4.26.

Tabel 4.9.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 1 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Minggu, 8 Februari 2004

Jam	Daya PLTU Paiton UNIT 1 (MW)	Emisi PLTU Paiton UNIT 1 (TON)	Biaya PLTU Paiton UNIT 1 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	331.77	0.0611	16,577	225.00	0.0033	10,640
01.00	311.18	0.0375	15,597	225.00	0.0033	10,640
02.00	308.95	0.0353	15,487	225.00	0.0033	10,640
03.00	303.95	0.0306	15,235	225.00	0.0033	10,640
04.00	306.46	0.0329	15,362	225.14	0.0032	10,649
05.00	305.86	0.0324	15,331	225.00	0.0033	10,640
06.00	305.21	0.0318	15,299	225.00	0.0033	10,640
07.00	239.18	0.0018	11,550	225.00	0.0033	10,640
08.00	238.75	0.0018	11,523	225.00	0.0033	10,640
09.00	236.00	0.0019	11,349	225.00	0.0033	10,640
10.00	237.89	0.0018	11,469	225.00	0.0033	10,640
11.00	238.04	0.0018	11,478	225.00	0.0033	10,640
12.00	219.57	0.0046	10,281	225.00	0.0033	10,640
13.00	236.15	0.0018	11,359	225.00	0.0033	10,640
14.00	233.11	0.0021	11,165	225.00	0.0033	10,640
15.00	237.94	0.0018	11,472	225.00	0.0033	10,640
16.00	285.36	0.0163	14,257	225.00	0.0033	10,640
17.00	313.36	0.0397	15,705	225.14	0.0032	10,649
18.00	361.52	0.1056	17,854	225.00	0.0033	10,640
19.00	373.32	0.1267	18,315	225.00	0.0033	10,640
20.00	376.57	0.1329	18,437	225.14	0.0032	10,649
21.00	375.00	0.1299	18,379	225.00	0.0033	10,640
22.00	355.48	0.0956	17,608	225.28	0.0032	10,658
23.00	356.69	0.0975	17,658	225.00	0.0033	10,640
TOTAL	**7,087.32**	**1.0252**	**348,747**	**5,400.7**	**0.0788**	**255,405**

Tabel 4.10.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 2 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Minggu, 8 Februari 2004

Jam	Daya PLTU Paiton UNIT 2 (MW)	Emisi PLTU Paiton UNIT 2 (TON)	Biaya PLTU Paiton UNIT 2 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	331.43	0.0758	16,190	225.00	0.0105	10,516
01.00	308.76	0.0524	15,133	225.00	0.0105	10,516
02.00	312.55	0.0559	15,316	225.00	0.0105	10,516
03.00	303.34	0.0475	14,869	225.00	0.0105	10,516
04.00	263.12	0.0208	12,757	225.14	0.0105	10,525
05.00	261.17	0.0200	12,648	225.00	0.0105	10,516
06.00	262.91	0.0207	12,745	225.00	0.0105	10,516
07.00	237.20	0.0122	11,259	225.00	0.0105	10,516
08.00	247.57	0.0149	11,871	225.00	0.0105	10,516
09.00	246.27	0.0145	11,795	225.00	0.0105	10,516
10.00	246.65	0.0146	11,817	225.00	0.0105	10,516
11.00	237.90	0.0124	11,301	225.00	0.0105	10,516
12.00	224.07	0.0104	10,459	225.00	0.0105	10,516
13.00	247.35	0.0148	11,858	225.00	0.0105	10,516
14.00	242.22	0.0134	11.557	225.00	0.0105	10,516
15.00	246.16	0.0144	11,789	225.00	0.0105	10,516
16.00	282.69	0.0318	13,817	225.00	0.0105	10,516
17.00	323.34	0.0668	15,822	225.14	0.0105	10,525
18.00	365.31	0.1203	17,616	225.00	0.0105	10,516
19.00	377.92	0.1398	18,100	369.72	0.1270	17,788
20.00	376.33	0.1373	18,040	369.57	0.1267	17,782
21.00	372.82	0.1318	17,907	368.58	0.1252	17,744
22.00	349.00	0.0974	16,952	252.50	0.0165	12,156
23.00	332.71	0.0772	16,247	225.14	0.0105	10,525
TOTAL	**6,998.79**	**1.2171**	**326,320**	**5,860.79**	**0.6054**	**275,817**

Tabel 4.11.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 5 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Minggu, 8 Februari 2004

Jam	Daya PLTU Paiton UNIT 5 (MW)	Emisi PLTU Paiton UNIT 5 (TON)	Biaya PLTU Paiton UNIT 5 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	461.20	0.2994	12,150	610.00	0.3844	15,896
01.00	458.58	0.2983	12,087	609.71	0.3842	15,888
02.00	459.48	0.2986	12,109	610.00	0.3844	15,896
03.00	459.70	0.2987	12,114	606.77	0.3820	15,811
04.00	437.65	0.2900	11,589	594.75	0.3736	15,496
05.00	432.10	0.2879	11,459	596.22	0.3746	15,534
06.00	430.08	0.2872	11,411	554.87	0.3477	14,467
07.00	364.00	0.2678	9,894	419.68	0.2836	11,168
08.00	314.95	0.2590	8,813	429.94	0.2871	11,408
09.00	315.10	0.2590	8,816	433.17	0.2883	11,484
10.00	317.80	0.2594	8,875	447.24	0.2937	11,817
11.00	316.15	0.2591	8,839	418.50	0.2832	11,140
12.00	313.98	0.2589	8,792	414.99	0.2820	11,058
13.00	315.78	0.2591	8,831	418.80	0.2833	11,147
14.00	315.55	0.2591	8,826	449.88	0.2947	11,879
15.00	316.60	0.2592	8,849	432.87	0.2882	11,477
16.00	316.30	0.2592	8,842	452.23	0.2957	11,935
17.00	315.18	0.2590	8,818	557.80	0.3495	14,542
18.00	387.03	0.2735	10,415	610.00	0.3844	15,896
19.00	482.05	0.3086	12,653	610.00	0.3844	15,896
20.00	554.13	0.3473	14,448	610.00	0.3844	15,896
21.00	552.78	0.3464	14,414	610.00	0.3844	15,896
22.00	552.63	0.3463	14,410	610.00	0.3844	15,896
23.00	553.60	0.3469	14,434	610.00	0.3844	15,896
TOTAL	**9,742.36**	**6.8879**	**261,888**	**12,717.42**	**8.1666**	**333,419**

Tabel 4.12.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 6 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Minggu, 8 Februari 2004

Jam	Daya PLTU Paiton UNIT 6 (MW)	Emisi PLTU Paiton UNIT 6 (TON)	Biaya PLTU Paiton UNIT 6 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	458.28	0.3920	12,507	610.00	0.6000	16,403
01.00	456.10	0.3903	12,453	609.71	0.5994	16,395
02.00	454.75	0.3893	12,420	609.41	0.5988	16,387
03.00	456.18	0.3904	12,455	608.53	0.5971	16,364
04.00	433.00	0.3752	11,881	605.01	0.5902	16,271
05.00	433.30	0.3754	11,888	603.55	0.5874	16,232
06.00	432.40	0.3749	11,866	487.42	0.4176	13,238
07.00	358.83	0.3555	10,079	344.02	0.3568	9,726
08.00	314.65	0.3647	9.032	339.91	0.3575	9,628
09.00	315.93	0.3642	9,062	335.81	0.3583	9,531
10.00	316.60	0.3640	9,078	323.20	0.3617	9,233
11.00	315.55	0.3644	9,053	344.31	0.3568	9,733
12.00	316.83	0.3639	9,083	318.21	0.3634	9,116
13.00	317.20	0.3637	9,092	343.72	0.3569	9,719
14.00	315.70	0.3643	9,057	315.28	0.3645	9,047
15.00	316.38	0.3640	9,073	339.91	0.3575	9,628
16.00	316.75	0.3639	9,081	397.98	0.3604	11,023
17.00	314.05	0.3649	9,018	484.78	0.4150	13,171
18.00	384.48	0.3573	10,696	609.71	0.5994	16,395
19.00	489.25	0.4195	13,284	610.00	0.6000	16,403
20.00	553.75	0.5014	14,932	610.00	0.6000	16,403
21.00	553.60	0.5012	14,928	610.00	0.6000	16,403
22.00	554.20	0.5021	14,943	609.71	0.5994	16,395
23.00	554.80	0.5030	14,959	610.00	0.6000	16,403
TOTAL	**9,732.54**	**9.4695**	**260,897**	**11,680.18**	**11.5981**	**319,247**

Tabel 4.13.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 7 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Minggu, 8 Februari 2004

Jam	Daya PLTU Paiton UNIT 7 (MW)	Emisi PLTU Paiton UNIT 7 (TON)	Biaya PLTU Paiton UNIT 7 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	516.17	0.4292	22,541	467.77	0.3824	20,658
01.00	513.73	0.4266	22,441	447.83	0.3659	19,948
02.00	472.55	0.3867	20,834	422.57	0.3472	19,106
03.00	472.08	0.3862	20,816	386.01	0.3247	17,997
04.00	467.61	0.3823	20,652	367.06	0.3151	17,474
05.00	466.86	0.3817	20,625	362.74	0.3132	17,360
06.00	467.28	0.3820	20,640	364.40	0.3139	17,404
07.00	460.72	0.3764	20,403	335.82	0.3025	16,689
08.00	420.32	0.3457	19,034	342.47	0.3049	16,848
09.00	422.23	0.3470	19,095	341.47	0.3045	16,824
10.00	421.65	0.3466	19,077	340.47	0.3041	16,800
11.00	422.66	0.3473	19,109	340.14	0.3040	16,792
12.00	424.17	0.3483	19,158	336.82	0.3029	16,713
13.00	422.34	0.3471	19,099	345.79	0.3061	16,929
14.00	422.12	0.3469	19,092	336.82	0.3029	16,713
15.00	422.29	0.3470	19,097	337.48	0.3031	16,728
16.00	422.35	0.3471	19,099	340.47	0.3041	16,800
17.00	422.02	0.3468	19,088	369.39	0.3162	17,537
18.00	427.71	0.3508	19,272	476.74	0.3904	20,990
19.00	491.63	0.4043	21,558	523.27	0.4369	22,837
20.00	578.90	0.5038	25,324	542.55	0.4587	23,664
21.00	592.53	0.5221	25,980	538.56	0.4540	23,490
22.00	592.27	0.5217	25,967	610.68	0.5476	26,881
23.00	591.87	0.5212	25,948	479.73	0.3931	21,102
TOTAL	**11,334.05**	**9.4448**	**503,949**	**9,757.05**	**8.3984**	**454,284**

Tabel 4.14.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 8 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Minggu, 8 Februari 2004

Jam	Daya PLTU Paiton UNIT 8 (MW)	Emisi PLTU Paiton UNIT 8 (TON)	Biaya PLTU Paiton UNIT 8 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	500.62	0.3283	21,702	430.87	0.2792	19,055
01.00	477.10	0.3074	20,800	330.17	0.2769	15,389
02.00	440.57	0.2837	19,418	327.51	0.2779	15,294
03.00	410.17	0.2721	18,287	284.64	0.3025	13,791
04.00	390.41	0.2685	17,560	276.00	0.3092	13,492
05.00	391.38	0.2686	17,596	275.33	0.3097	13,469
06.00	391.18	0.2686	17,588	275.00	0.3100	13,457
07.00	380.11	0.2679	17,184	275.00	0.3100	13,457
08.00	298.10	0.2932	14,259	275.00	0.3100	13,457
09.00	297.63	0.2935	14,243	275.00	0.3100	13,457
10.00	297.23	0.2937	14,229	275.00	0.3100	13,457
11.00	297.14	0.2938	14,226	275.00	0.3100	13,457
12.00	297.07	0.2938	14,223	275.00	0.3100	13,457
13.00	296.20	0.2944	14,193	275.00	0.3100	13,457
14.00	296.56	0.2942	14,206	275.00	0.3100	13,457
15.00	296.44	0.2942	14,201	275.00	0.3100	13,457
16.00	295.90	0.2946	14,183	275.00	0.3100	13,457
17.00	296.44	0.2942	14,201	275.00	0.3100	13,457
18.00	301.33	0.2911	14,372	421.24	0.2754	18,697
19.00	377.88	0.2678	17,103	602.37	0.4698	25,720
20.00	502.31	0.3300	21,767	612.67	0.4888	26,137
21.00	518.10	0.3467	22,379	612.34	0.4881	26,123
22.00	517.52	0.3461	22,356	614.34	0.4919	26,204
23.00	517.82	0.3464	22,368	475.74	0.3063	20,748
TOTAL	**9,085.19**	**7.1328**	**412,644**	**8,563.22**	**7.9957**	**395,603**

Tabel 4.15.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 1 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Senin, 9 Februari 2004

Jam	Daya PLTU Paiton UNIT 1 (MW)	Emisi PLTU Paiton UNIT 1 (TON)	Biaya PLTU Paiton UNIT 1 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	325.97	0.0538	16,309	225.00	0.0033	10,640
01.00	334.35	0.0644	16,695	225.00	0.0033	10,640
02.00	314.08	0.0405	15,740	225.00	0.0033	10,640
03.00	313.18	0.0395	15,696	225.00	0.0033	10,640
04.00	306.53	0.0330	15,365	225.00	0.0033	10,640
05.00	306.25	0.0327	15,351	225.00	0.0033	10,640
06.00	309.69	0.0360	15,523	225.00	0.0033	10,640
07.00	303.06	0.0298	15,189	225.00	0.0033	10,640
08.00	303.77	0.0305	15,225	225.00	0.0033	10,640
09.00	342.20	0.0753	17,044	225.00	0.0033	10,640
10.00	355.85	0.0962	17,624	225.00	0.0033	10,640
11.00	381.30	0.1421	18,612	225.14	0.0032	10,649
12.00	376.90	0.1335	18,450	225.00	0.0033	10,640
13.00	380.11	0.1397	18,569	225.00	0.0033	10,640
14.00	377.47	0.1346	18,471	225.00	0.0033	10,640
15.00	380.33	0.1402	18,577	225.14	0.0032	10,649
16.00	375.97	0.1317	18,415	225.28	0.0032	10,658
17.00	379.81	0.1391	18,558	225.14	0.0032	10,649
18.00	375.99	0.1318	18,416	225.00	0.0033	10,640
19.00	374.53	0.1290	18,361	370.00	0.1206	18,188
20.00	375.36	0.1306	18,392	370.00	0.1206	18,188
21.00	372.44	0.1251	18,282	370.00	0.1206	18,188
22.00	377.46	0.1346	18,471	225.00	0.0033	10,640
23.00	366.46	0.1142	18,050	225.00	0.0033	10,640
TOTAL	**8,409.08**	**2.2579**	**415,322**	**5,835.7**	**0.4307**	**278,049**

Tabel 4.16.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 2 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Senin, 9 Februari 2004

Jam	Daya PLTU Paiton UNIT 2 (MW)	Emisi PLTU Paiton UNIT 2 (TON)	Biaya PLTU Paiton UNIT 2 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	337.05	0.0824	16,439	225.00	0.0105	10,516
01.00	339.44	0.0853	16,544	225.00	0.0105	10,516
02.00	312.45	0.0558	15,311	225.00	0.0105	10,516
03.00	313.45	0.0568	15,359	225.00	0.0105	10,516
04.00	313.56	0.0569	15,364	225.00	0.0105	10,516
05.00	314.69	0.0580	15,418	225.00	0.0105	10,516
06.00	314.69	0.0580	15,418	225.00	0.0105	10,516
07.00	304.73	0.0487	14,937	225.00	0.0105	10,516
08.00	302.61	0.0469	14,833	225.00	0.0105	10,516
09.00	327.17	0.0710	15,998	225.00	0.0105	10,516
10.00	349.83	0.0985	16,987	225.28	0.0105	10,534
11.00	372.95	0.1320	17,912	225.43	0.0105	10,543
12.00	375.51	0.1360	18,009	225.14	0.0105	10,525
13.00	374.00	0.1336	17,952	225.00	0.0105	10,516
14.00	376.29	0.1372	18,039	225.14	0.0105	10,525
15.00	369.78	0.1271	17,790	225.14	0.0105	10,525
16.00	369.59	0.1268	17,783	225.43	0.0105	10,543
17.00	379.98	0.1432	18,176	225.28	0.0105	10,534
18.00	376.50	0.1376	18,046	367.45	0.1235	17,700
19.00	376.28	0.1372	18,038	369.86	0.1272	17,793
20.00	371.05	0.1290	17,839	369.57	0.1267	17,782
21.00	372.21	0.1308	17,884	370.00	0.1274	17,799
22.00	381.64	0.1459	18,238	367.02	0.1229	17,683
23.00	369.64	0.1268	17,785	225.00	0.0105	10,516
TOTAL	**8,395.09**	**2.4615**	**406,099**	**6,120.74**	**0.8272**	**288,678**

Tabel 4.17.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 5 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Senin, 9 Februari 2004

Jam	Daya PLTU Paiton UNIT 5 (MW)	Emisi PLTU Paiton UNIT 5 (TON)	Biaya PLTU Paiton UNIT 5 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	504.93	0.3198	13,214	610.00	0.3844	15,896
01.00	462.25	0.2998	12,175	608.24	0.3831	15,850
02.00	465.78	0.3013	12,260	557.21	0.3491	14,527
03.00	407.20	0.2795	10,878	549.59	0.3445	14,332
04.00	402.48	0.2780	10,769	521.44	0.3284	13,624
05.00	404.20	0.2785	10,809	552.82	0.3465	14,414
06.00	430.50	0.2873	11,421	536.10	0.3366	13,991
07.00	381.40	0.2720	10,287	486.25	0.3106	12,756
08.00	328.45	0.2609	9,107	588.30	0.3692	15,328
09.00	408.85	0.2800	10,916	607.65	0.3827	15,834
10.00	458.58	0.2983	12,087	610.00	0.3844	15,896
11.00	452.50	0.2958	11,942	610.00	0.3844	15,896
12.00	454.90	0.2967	11,999	610.00	0.3844	15,896
13.00	405.93	0.2791	10,849	610.00	0.3844	15,896
14.00	412.45	0.2812	10,999	609.41	0.3839	15,881
15.00	457.23	0.2977	12,055	610.00	0.3844	15,896
16.00	457.68	0.2979	12,065	610.00	0.3844	15,896
17.00	457.38	0.2978	12,058	609.12	0.3837	15,873
18.00	476.43	0.3060	12,517	610.00	0.3844	15,896
19.00	580.45	0.3640	15,124	610.00	0.3844	15,896
20.00	604.23	0.3802	15,744	610.00	0.3844	15,896
21.00	600.63	0.3777	15,650	610.00	0.3844	15,896
22.00	601.07	0.3780	15,661	610.00	0.3844	15,896
23.00	532.45	0.3345	13,899	610.00	0.3844	15,896
TOTAL	**11,148.90**	**7.3420**	**294,485**	**14,156.13**	**8.9155**	**369,058**

Tabel 4.18.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 6 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Senin, 9 Februari 2004

Jam	Daya PLTU Paiton UNIT 6 (MW)	Emisi PLTU Paiton UNIT 6 (TON)	Biaya PLTU Paiton UNIT 6 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	507.70	0.4395	13,751	609.71	0.5994	16,395
01.00	460.00	0.3933	12,550	595.92	0.5730	16,032
02.00	360.70	0.3555	10,124	527.89	0.4645	14,266
03.00	358.45	0.3555	10,070	494.75	0.4252	13,423
04.00	356.88	0.3556	10,032	536.10	0.4757	14,477
05.00	358.15	0.3555	10,063	565.43	0.5198	15,234
06.00	385.75	0.3576	10,727	476.86	0.4075	12,972
07.00	325.98	0.3609	9,299	401.50	0.3615	11,109
08.00	305.73	0.3684	8,823	558.68	0.5090	15,059
09.00	409.08	0.3640	11,294	610.00	0.6000	16,403
10.00	557.58	0.5073	15,031	610.00	0.6000	16,403
11.00	555.55	0.5042	14,978	610.00	0.6000	16,403
12.00	554.73	0.5029	14,957	609.41	0.5988	16,387
13.00	503.95	0.4352	13,656	610.00	0.6000	16,403
14.00	506.28	0.4379	13,715	610.00	0.6000	16,403
15.00	555.33	0.5038	14,973	609.12	0.5983	16,379
16.00	505.15	0.4366	13,686	609.41	0.5988	16,387
17.00	554.35	0.5023	14,947	609.41	0.5988	16,387
18.00	531.93	0.4700	14,370	610.00	0.6000	16,403
19.00	579.48	0.5434	15,601	610.00	0.6000	16,403
20.00	600.93	0.5824	16,163	610.00	0.6000	16,403
21.00	603.93	0.5881	16,242	610.00	0.6000	16,403
22.00	601.67	0.5838	16,183	609.71	0.5994	16,395
23.00	529.45	0.4666	14,306	610.00	0.6000	16,403
TOTAL	**11,568.67**	**10.7703**	**315,541**	**13,913.9**	**13.3297**	**374,932**

Tabel 4.19.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 7 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Senin, 9 Februari 2004

Jam	Daya PLTU Paiton UNIT 7 (MW)	Emisi PLTU Paiton UNIT 7 (TON)	Biaya PLTU Paiton UNIT 7 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	492.22	0.4049	21,581	458.13	0.3742	20,310
01.00	472.60	0.3867	20,835	370.72	0.3169	17,573
02.00	441.67	0.3611	19,737	358.42	0.3113	17,248
03.00	423.67	0.3480	19,141	367.39	0.3153	17,483
04.00	423.38	0.3478	19,132	353.44	0.3092	17,120
05.00	423.82	0.3481	19,146	365.40	0.3144	17,430
06.00	435.27	0.3563	19,522	366.07	0.3147	17,448
07.00	466.26	0.3811	20,603	343.13	0.3051	16,864
08.00	423.01	0.3475	19,120	362.41	0.3130	17,351
09.00	438.42	0.3586	19,627	472.09	0.3862	20,816
10.00	515.88	0.4289	22,529	541.22	0.4571	23,606
11.00	523.58	0.4372	22,850	498.68	0.4112	21,834
12.00	524.27	0.4380	22,879	489.04	0.4018	21,457
13.00	515.16	0.4282	22,500	479.73	0.3931	21,102
14.00	472.92	0.3870	20,847	505.99	0.4186	22,126
15.00	516.58	0.4297	22,558	503.33	0.4159	22,019
16.00	518.34	0.4315	22,631	492.69	0.4054	21,599
17.00	507.17	0.4198	22,174	544.54	0.4610	23,752
18.00	523.72	0.4374	22,856	579.44	0.5045	25,350
19.00	561.25	0.4812	24,502	592.73	0.5224	25,990
20.00	615.88	0.5551	27,145	584.42	0.5111	25,588
21.00	592.63	0.5222	25,985	594.73	0.5251	26,087
22.00	591.30	0.5204	25,920	524.93	0.4387	22,907
23.00	548.08	0.4652	23,909	495.02	0.4076	21,690
TOTAL	**11,967.07**	**10.0219**	**527,729**	**11,243.69**	**9.5338**	**504,750**

Tabel 4.20.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 8 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Senin, 9 Februari 2004

Jam	Daya PLTU Paiton UNIT 8 (MW)	Emisi PLTU Paiton UNIT 8 (TON)	Biaya PLTU Paiton UNIT 8 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	533.12	0.3645	22,965	381.69	0.2679	17,242
01.00	440.97	0.2839	19,433	275.00	0.3100	13,457
02.00	405.32	0.2709	18,108	275.33	0.3097	13,469
03.00	352.77	0.2703	16,196	275.66	0.3085	13,480
04.00	334.73	0.2752	15,551	275.33	0.3097	13,469
05.00	332.09	0.2761	15,457	275.00	0.3100	13,457
06.00	331.81	0.2763	15,447	275.00	0.3100	13,457
07.00	322.49	0.2800	15,117	275.00	0.3100	13,457
08.00	292.19	0.2970	14,053	275.00	0.3100	13,457
09.00	308.56	0.2869	14,626	531.91	0.3630	22,918
10.00	434.08	0.2806	19,175	560.49	0.4016	24,044
11.00	486.07	0.3149	21,143	601.37	0.4680	25,679
12.00	484.39	0.3134	21,078	493.69	0.3217	21,435
13.00	473.27	0.3044	20,654	434.20	0.2806	19,180
14.00	438.69	0.2827	19,348	581.43	0.4339	24,878
15.00	477.81	0.3080	20,827	536.23	0.3684	23,087
16.00	482.40	0.3117	21,002	583.43	0.4372	24,958
17.00	467.54	0.3002	20,436	561.16	0.4025	24,070
18.00	490.17	0.3185	21,300	612.67	0.4888	26,137
19.00	532.55	0.3638	22,943	614.67	0.4925	26,218
20.00	599.73	0.4651	25,613	614.34	0.4919	26,204
21.00	616.53	0.4961	26,294	615.00	0.4932	26,231
22.00	616.52	0.4961	26,293	612.34	0.4881	26,123
23.00	602.85	0.4707	25,739	535.90	0.3680	23,074
TOTAL	**10,856.64**	**7.9073**	**478,798**	**11,071.84**	**9.0452**	**489,181**

Tabel 4.21.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 1 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Selasa, 10 Februari 2004

Jam	Daya PLTU Paiton UNIT 1 (MW)	Emisi PLTU Paiton UNIT 1 (TON)	Biaya PLTU Paiton UNIT 1 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	357.64	0.0991	17,697	225.00	0.0033	10,640
01.00	356.67	0.0975	17,657	225.00	0.0033	10,640
02.00	330.14	0.0590	16,503	225.43	0.0032	10,667
03.00	330.00	0.0588	16,496	225.00	0.0033	10,640
04.00	328.26	0.0566	16,416	225.00	0.0033	10,640
05.00	326.68	0.0547	16,343	225.00	0.0033	10,640
06.00	328.72	0.0572	16,437	225.00	0.0033	10,640
07.00	327.13	0.0552	16,363	225.00	0.0033	10,640
08.00	328.42	0.0568	16,423	225.00	0.0033	10,640
09.00	373.00	0.1261	18,303	225.00	0.0033	10,640
10.00	377.19	0.1341	18,461	225.00	0.0033	10,640
11.00	377.69	0.1350	18,479	225.00	0.0033	10,640
12.00	376.69	0.1331	18,442	225.00	0.0033	10,640
13.00	378.08	0.1358	18,494	225.00	0.0033	10,640
14.00	377.93	0.1355	18,488	225.00	0.0033	10,640
15.00	377.61	0.1349	18,476	225.00	0.0033	10,640
16.00	383.54	0.1466	18,693	225.00	0.0033	10,640
17.00	380.58	0.1407	18,586	225.00	0.0033	10,640
18.00	379.19	0.1379	18,535	225.00	0.0033	10,640
19.00	374.46	0.1288	18,358	296.01	0.0239	14,825
20.00	374.46	0.1288	18,358	370.00	0.1206	18,188
21.00	371.13	0.1226	18,232	370.00	0.1206	18,188
22.00	377.65	0.1349	18,478	225.00	0.0033	10,640
23.00	355.19	0.0951	17,596	225.00	0.0033	10,640
TOTAL	**8,648.03**	**2.5648**	**426,314**	**5,761.44**	**0.3343**	**274,668**

Tabel 4.22.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 2 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Selasa, 10 Februari 2004

Jam	Daya PLTU Paiton UNIT 2 (MW)	Emisi PLTU Paiton UNIT 2 (TON)	Biaya PLTU Paiton UNIT 2 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	352.69	0.1024	17,106	225.00	0.0105	10,516
01.00	352.71	0.1024	17,107	225.28	0.0105	10,534
02.00	337.53	0.0829	16,460	225.28	0.0105	10,534
03.00	329.85	0.0740	16,119	225.00	0.0105	10,516
04.00	326.99	0.0708	15,990	225.14	0.0105	10,525
05.00	332.00	0.0764	16,216	225.00	0.0105	10,516
06.00	334.67	0.0795	16,334	225.00	0.0105	10,516
07.00	331.71	0.0761	16,203	225.00	0.0105	10,516
08.00	333.58	0.0783	16,286	225.00	0.0105	10,516
09.00	372.00	0.1305	17,876	225.00	0.0105	10,516
10.00	371.47	0.1297	17,855	225.00	0.0105	10,516
11.00	369.95	0.1273	17,797	225.00	0.0105	10,516
12.00	370.35	0.1279	17,812	225.14	0.0105	10,525
13.00	374.69	0.1347	17,978	225.00	0.0105	10,516
14.00	373.97	0.1336	17,951	225.00	0.0105	10,516
15.00	373.31	0.1325	17,926	225.14	0.0105	10,525
16.00	376.05	0.1368	18,030	225.14	0.0105	10,525
17.00	375.73	0.1363	18,018	225.00	0.0105	10,516
18.00	380.64	0.1443	18,201	225.14	0.0105	10,525
19.00	373.47	0.1328	17,932	369.72	0.1270	17,788
20.00	378.24	0.1404	18,112	370.00	0.1274	17,799
21.00	378.06	0.1401	18,105	370.00	0.1274	17,799
22.00	377.65	0.1394	18,090	368.87	0.1257	17,755
23.00	353.78	0.1039	17,151	225.28	0.0105	10,534
TOTAL	**8,631.09**	**2.7330**	**416,655**	**5,980.13**	**0.7175**	**281,560**

Tabel 4.23.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 5 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Selasa, 10 Februari 2004

Jam	Daya PLTU Paiton UNIT 5 (MW)	Emisi PLTU Paiton UNIT 5 (TON)	Biaya PLTU Paiton UNIT 5 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	479.88	0.3076	12,601	609.41	0.3839	15,881
01.00	409.53	0.2802	10,932	609.41	0.3839	15,881
02.00	391.75	0.2749	10,523	608.83	0.3835	15,865
03.00	430.45	0.2873	11,420	608.24	0.3831	15,850
04.00	429.55	0.2870	11,399	607.65	0.3827	15,834
05.00	411.55	0.2809	10,979	609.41	0.3839	15,881
06.00	356.13	0.2660	9,718	606.19	0.3816	15,796
07.00	358.98	0.2666	9,782	523.49	0.3296	13,675
08.00	331.60	0.2614	9,176	609.41	0.3839	15,881
09.00	408.03	0.2797	10,897	610.00	0.3844	15,896
10.00	487.90	0.3114	12,796	610.00	0.3844	15,896
11.00	515.65	0.3253	13,480	610.00	0.3844	15,896
12.00	507.63	0.3211	13,281	610.00	0.3844	15,896
13.00	453.70	0.2963	11,970	609.71	0.3842	15,888
14.00	454.08	0.2964	11,979	610.00	0.3844	15,896
15.00	507.63	0.3211	13,281	610.00	0.3844	15,896
16.00	506.35	0.3205	13,249	610.00	0.3844	15,896
17.00	506.28	0.3204	13,247	610.00	0.3844	15,896
18.00	505.98	0.3203	13,240	610.00	0.3844	15,896
19.00	508.60	0.3216	13,305	610.00	0.3844	15,896
20.00	597.55	0.3756	15,569	610.00	0.3844	15,896
21.00	599.80	0.3771	15,628	610.00	0.3844	15,896
22.00	601.75	0.3785	15,679	610.00	0.3844	15,896
23.00	552.18	0.3461	14,398	610.00	0.3844	15,896
TOTAL	**11,312.48**	**7.4233**	**298,529**	**14,541.75**	**9.1619**	**378,976**

Tabel 4.24.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 6 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Selasa, 10 Februari 2004

Jam	Daya PLTU Paiton UNIT 6 (MW)	Emisi PLTU Paiton UNIT 6 (TON)	Biaya PLTU Paiton UNIT 6 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	484.53	0.4148	13,165	610.00	0.6000	16,403
01.00	458.73	0.3923	12,519	608.53	0.5971	16,364
02.00	466.53	0.3985	12,713	607.95	0.5960	16,348
03.00	433.30	0.3754	11,888	609.71	0.5994	16,395
04.00	432.55	0.3749	11,870	610.00	0.6000	16,403
05.00	464.65	0.3970	12,666	609.12	0.5983	16,379
06.00	507.33	0.4391	13,742	525.54	0.4615	14,206
07.00	365.05	0.3555	10,228	434.05	0.3758	11,907
08.00	331.30	0.3594	9,424	600.62	0.5818	16,155
09.00	411.78	0.3651	11,360	609.41	0.5988	16,387
10.00	481.23	0.4116	13,082	610.00	0.6000	16,403
11.00	509.65	0.4418	13,801	610.00	0.6000	16,403
12.00	506.13	0.4377	13,711	609.41	0.5988	16,387
13.00	455.95	0.3902	12,449	609.71	0.5994	16,395
14.00	457.90	0.3917	12,498	610.00	0.6000	16,403
15.00	505.45	0.4369	13,694	610.00	0.6000	16,403
16.00	506.65	0.4383	13,724	609.41	0.5988	16,387
17.00	504.63	0.4360	13,673	610.00	0.6000	16,403
18.00	504.63	0.4360	13,673	609.71	0.5994	16,395
19.00	503.95	0.4352	13,656	610.00	0.6000	16,403
20.00	604.45	0.5892	16,256	610.00	0.6000	16,403
21.00	602.65	0.5857	16,209	610.00	0.6000	16,403
22.00	604.98	0.5902	16,270	610.00	0.6000	16,403
23.00	555.10	0.5035	14,967	610.00	0.6000	16,403
TOTAL	**11,659.05**	**10.3960**	**317,238**	**14,363.17**	**14.0051**	**386,541**

Tabel 4.25.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 7 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Selasa, 10 Februari 2004

Jam	Daya PLTU Paiton UNIT 7 (MW)	Emisi PLTU Paiton UNIT 7 (TON)	Biaya PLTU Paiton UNIT 7 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	505.11	0.4177	22,091	468.43	0.3830	20,682
01.00	478.17	0.3917	21,043	440.18	0.3600	19,687
02.00	472.79	0.3869	20,842	440.18	0.3600	19,687
03.00	472.57	0.3867	20,834	430.54	0.3528	19,365
04.00	464.77	0.3799	20,549	406.95	0.3369	18,616
05.00	434.88	0.3560	19,509	439.18	0.3592	19,653
06.00	472.92	0.3870	20,847	362.08	0.3129	17,343
07.00	463.47	0.3787	20,502	356,76	0.3105	17,205
08.00	421.50	0.3465	19,072	403.29	0.3347	18,505
09.00	465.91	0.3808	20,590	494.02	0.4067	21,651
10.00	492.46	0.4051	21,590	538.89	0.4544	23,505
11.00	518.13	0.4313	22,622	508.65	0.4213	22,233
12.00	522.83	0.4364	22,818	465.77	0.3807	20,585
13.00	514.63	0.4276	22,478	468.43	0.3830	20,682
14.00	496.49	0.4091	21,748	522.61	0.4362	22,809
15.00	519.87	0.4332	22,695	544.54	0.4610	23,752
16.00	522.72	0.4363	22,814	535.57	0.4506	23,361
17.00	522.27	0.4358	22,795	556.51	0.4754	24,286
18.00	530.60	0.4450	23,147	533.24	0.4480	23,260
19.00	581.67	0.5075	25,456	596.72	0.5279	26,185
20.00	589.95	0.5186	25,854	584.42	0.5111	25,588
21.00	590.50	0.5193	25,881	593.40	0.5233	26,022
22.00	590.28	0.5190	25,870	550.52	0.4681	24,017
23.00	566.02	0.4872	24,721	521.28	0.4347	22,753
TOTAL	**12,210.47**	**10.2233**	**536,368**	**11,405.4**	**9.8924**	**521,432**

Tabel 4.26.
Perbandingan Daya, Emisi (NOx) Dan Biaya Yang Dihasilkan Tiap Jam Pada PLTU Paiton Unit 8 Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi
Selasa, 10 Februari 2004

Jam	Daya PLTU Paiton UNIT 8 (MW)	Emisi PLTU Paiton UNIT 8 (TON)	Biaya PLTU Paiton UNIT 8 ($)	Daya *Refined Genetic Algorithm* (MW)	Emisi *Refined Genetic Algorithm* (TON)	Biaya *Refined Genetic Algorithm* ($)
00.00	529.98	0.3607	22,842	372.71	0.2681	16,915
01.00	454.74	0.2916	19,951	281.65	0.3047	13,687
02.00	391.48	0.2686	17,599	282.31	0.3042	13,710
03.00	393.97	0.2689	17,691	277.33	0.3081	13,538
04.00	393.82	0.2689	17,685	287.96	0.3000	13,906
05.00	392.96	0.2688	17,654	286.30	0.3012	13,849
06.00	394.32	0.2690	17,703	276.66	0.3086	13,515
07.00	374.27	0.2679	16,972	275.00	0.3100	13,457
08.00	292.75	0.2967	14,073	275.66	0.3095	13,480
09.00	308.35	0.2871	14,618	461.78	0.2962	20,217
10.00	415.07	0.2734	18,468	570.13	0.4160	24,427
11.00	487.98	0.3165	21,216	595.06	0.4569	25,425
12.00	490.16	0.3185	21,299	490.70	0.3190	21,320
13.00	448.88	0.2881	19,730	442.84	0.2848	19,503
14.00	420.37	0.2751	18,664	571.79	0.4186	24,493
15.00	480.58	0.3102	20,932	571.79	0.4186	24,493
16.00	491.02	0.3192	21,332	571.79	0.4186	24,493
17.00	487.48	0.3161	21,197	559.50	0.4001	24,004
18.00	484.80	0.3138	21,094	613.34	0.4900	26,164
19.00	474.21	0.3051	20,689	615.00	0.4932	26,231
20.00	552.95	0.3908	23,745	615.00	0.4932	26,231
21.00	617.30	0.4976	26,325	615.00	0.4932	26,231
22.00	616.18	0.4954	26,279	613.34	0.4900	26,164
23.00	595.53	0.4577	25,444	518.28	0.3469	22,386
TOTAL	**10,989.13**	**7.7257**	**483,202**	**11,040.92**	**8.9497**	**487,839**

Dari tabel 4.9 hingga tabel 4.26, terdapat adanya pelanggaran operasi PLTU Paiton terhadap kendala operasi pembangkitan (P_{min} dan P_{mak}). Sebaliknya, hasil perhitungan *Refined Genetic Algorithm* tidak terdapat pelanggaran terhadap kendala operasi pembangkitan maupun kendala mutu emisi.

4.8.2. Perbandingan Tingkat Optimum Daya

Tabel 4.9 hingga tabel 4.26 diatas, juga dapat disederhanakan menjadi suatu tabel perbandingan total daya. Jika dibuat suatu perbandingan total daya tiap 24 jam dan 72 jam antara PLTU Paiton dan hasil perhitungan *Refined Genetic Algorithm*, maka pada hasil perhitungan *Refined Genetic Algorithm* terlihat adanya perubahan daya yang harus dibangkitkan oleh tiap-tiap unit pembangkit yakni sebagai berikut :

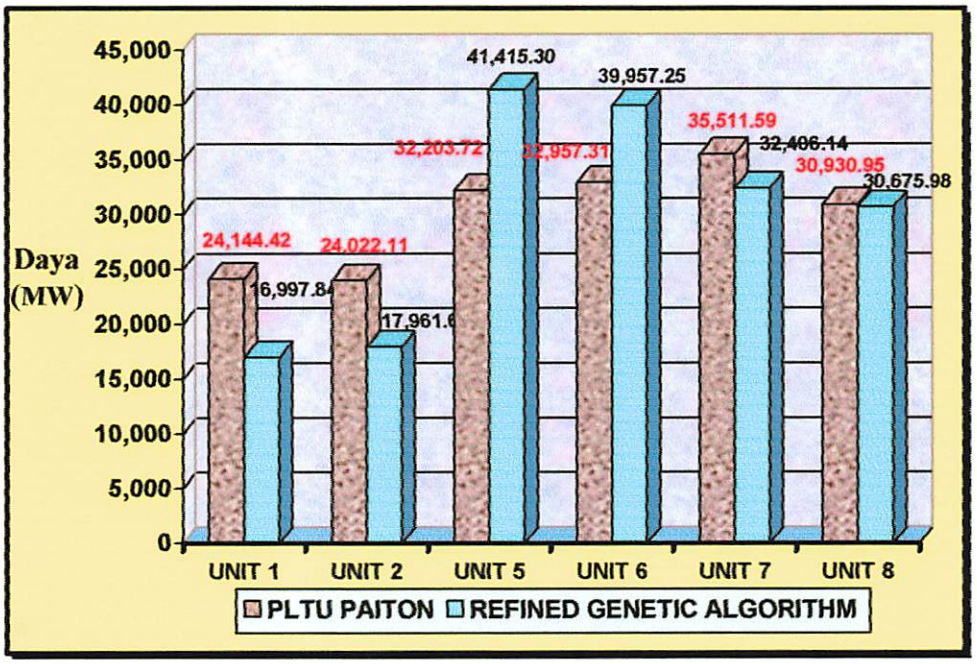
Tabel 4.27.
Perbandingan Daya Yang Dihasilkan Tiap 24 Jam Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi

Periode Waktu	PLTU Paiton	PLTU Paiton (MW)	*Refined Genetic Algorithm* (MW)
Minggu, 8 Februari 2004	UNIT 1	7,087.32	5,400.70
Minggu, 8 Februari 2004	UNIT 2	6,998.79	5,860.79
Minggu, 8 Februari 2004	UNIT 5	9,742.36	12,717.42
Minggu, 8 Februari 2004	UNIT 6	9,732.54	11,680.18
Minggu, 8 Februari 2004	UNIT 7	11,334.05	9,757.05
Minggu, 8 Februari 2004	UNIT 8	9,085.19	8,563.22
Senin, 9 Februari 2004	UNIT 1	8,409.08	5,835.70
Senin, 9 Februari 2004	UNIT 2	8,395.09	6,120.74
Senin, 9 Februari 2004	UNIT 5	11,148.90	14,156.13
Senin, 9 Februari 2004	UNIT 6	11,568.67	13,913.90
Senin, 9 Februari 2004	UNIT 7	11,967.07	11,243.69
Senin, 9 Februari 2004	UNIT 8	10,856.64	11,071.84
Selasa, 10 Februari 2004	UNIT 1	8,648.03	5,761.44
Selasa, 10 Februari 2004	UNIT 2	8,631.09	5,980.13
Selasa, 10 Februari 2004	UNIT 5	11,312.48	14,541.75
Selasa, 10 Februari 2004	UNIT 6	11,659.05	14,363.17
Selasa, 10 Februari 2004	UNIT 7	12,210.47	11,405.40
Selasa, 10 Februari 2004	UNIT 8	10,989.13	11,040.92

Tabel 4.28.
Perbandingan Total Daya Yang Dihasilkan Tiap 72 Jam
Pada PLTU Paiton Dan *Refined Genetic Algorithm*
Dengan Memperhatikan Kendala Emisi

PLTU Paiton	PLTU Paiton (MW)	*Refined Genetic Algorithm* (MW)
UNIT 1	24,144.42	16,997.84
UNIT 2	24,022.11	17,961.66
UNIT 5	32,203.72	41,415.30
UNIT 6	32,957.31	39,957.25
UNIT 7	35,511.59	32,406.14
UNIT 8	30,930.95	30,675.98

Grafik 4.1
Perbandingan Total Daya Tiap 72 Jam Tiap Unit Pembangkit
Pada PLTU Paiton Dan *Refined Genetic Algorithm*
Dengan Memperhatikan Kendala Emisi



Untuk mencapai fungsi tujuan yaitu mendapatkan biaya operasi yang murah dan pemenuhan permintaan beban, *Refined Genetic Algorithm* akan melakukan optimasi terhadap komposisi output daya dari tiap pembangkit dan memberikan prioritas pembangkitan terhadap unit-unit pembangkitan yang memiliki biaya bahan bakar (biaya operasi) yang lebih murah.

Dari tabel 4.27, 4.28 dan grafik 4.1 terlihat bahwa pada unit 1 dan 2 terjadi pengurangan daya yang harus dibangkitkan, untuk unit 1 dari 24,144.42 MW menjadi 16,997.84 MW dan untuk unit 2 dari 24,022.11 MW menjadi 17,961.66 MW. Ini dilakukan untuk mengurangi biaya bahan bakar unit 1 dan unit 2 yang lebih mahal dari unit-unit pembangkit lain.

Unit 5 dan unit 6 diprioritaskan untuk menaikkan daya yang harus dibangkitkan, karena biaya bahan bakar unit 5 dan unit 6 lebih murah dari unit-unit pembangkit lain. Untuk unit 5 terjadi penambahan daya yang harus dibangkitkan, dari 32,203.72 MW menjadi 41,415.30 MW. Sedangkan untuk unit 6 dari 32,957.31 MW menjadi 39,957.25 MW.

Unit 7 dan unit 8 terjadi pengurangan daya yang harus dibangkitkan, untuk unit 7 dari 35,511.59 MW menjadi 32,406.14 MW dan untuk unit 8 dari 30,930.95 MW menjadi 30,675.98 MW. Ini dilakukan untuk mengurangi biaya bahan bakar unit 7 dan unit 8 yang relatif lebih mahal dari unit 5 dan unit 6.

4.8.3. Perbandingan Tingkat Optimum Emisi NOx

Dari tabel 4.9 hingga tabel 4.26, dapat dibuat suatu perbandingan total output emisi NOx tiap 24 jam dan 72 jam antara PLTU Paiton dan hasil perhitungan *Refined Genetic Algorithm*, maka pada hasil perhitungan *Refined*

Genetic Algorithm terlihat adanya penambahan atau maksimalisasi output emisi namun masih dibawah batas baku mutu emisi yang telah ditetapkan.

Tabel 4.29.
Perbandingan Total Emisi Yang Dihasilkan Tiap 24 Jam Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi

Periode Waktu	PLTU Paiton	PLTU Paiton (Ton/hari)	Refined Genetic Algorithm (Ton/hari)	NOx Limit (Ton/hari)
Minggu, 8 Februari 2004	UNIT 1	1.0252	0.0788	53.1504
Minggu, 8 Februari 2004	UNIT 2	1.2171	0.6054	53.1504
Minggu, 8 Februari 2004	UNIT 5	6.8879	8.1666	36.9200
Minggu, 8 Februari 2004	UNIT 6	9.4695	11.5981	36.9200
Minggu, 8 Februari 2004	UNIT 7	9.4448	8.3984	35.5000
Minggu, 8 Februari 2004	UNIT 8	7.1328	7.9957	35.5000
Senin, 9 Februari 2004	UNIT 1	2.2579	0.4307	53.1504
Senin, 9 Februari 2004	UNIT 2	2.4615	0.8272	53.1504
Senin, 9 Februari 2004	UNIT 5	7.3420	8.9155	36.9200
Senin, 9 Februari 2004	UNIT 6	10.7703	13.3297	36.9200
Senin, 9 Februari 2004	UNIT 7	10.0219	9.5338	35.5000
Senin, 9 Februari 2004	UNIT 8	7.9073	9.0452	35.5000
Selasa, 10 Februari 2004	UNIT 1	2.5648	0.3343	53.1504
Selasa, 10 Februari 2004	UNIT 2	2.7330	0.7175	53.1504
Selasa, 10 Februari 2004	UNIT 5	7.4233	9.1619	36.9200
Selasa, 10 Februari 2004	UNIT 6	10.3960	14.0051	36.9200
Selasa, 10 Februari 2004	UNIT 7	10.2233	9.8924	35.5000
Selasa, 10 Februari 2004	UNIT 8	7.7257	8.9497	35.5000

Tabel 4.30.
Perbandingan Total Emisi Tiap 72 Jam Tiap Unit Pembangkit Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi

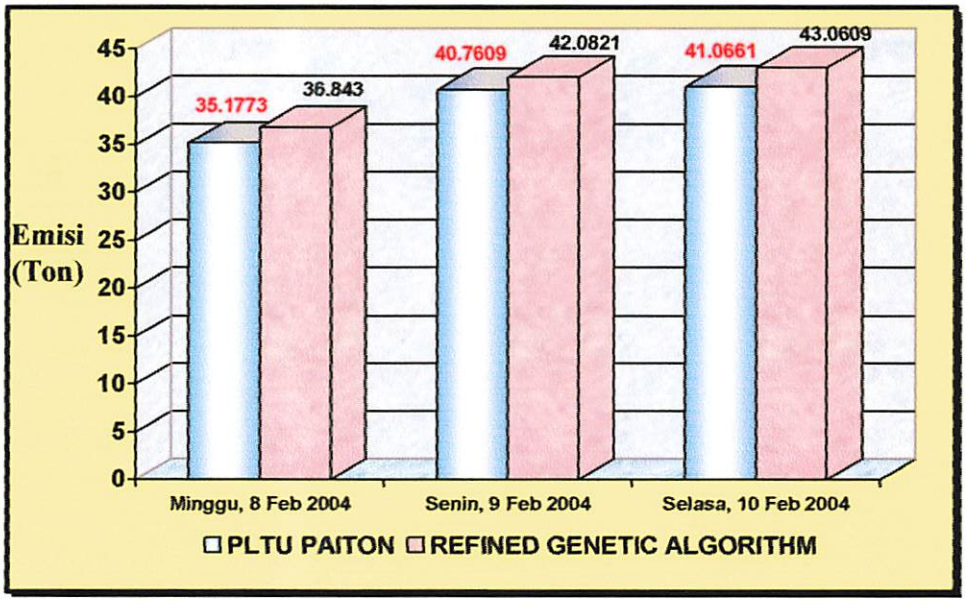
PLTU Paiton	PLTU Paiton (Ton/tiga hari)	*Refined Genetic Algorithm* (Ton/tiga hari)
UNIT 1	5.8479	0.8438
UNIT 2	6.4116	2.1501
UNIT 5	21.6532	26.2440
UNIT 6	30.6358	38.9329
UNIT 7	29.6900	27.8246
UNIT 8	22.7658	25.9906

Dari tabel perbandingan emisi untuk tiap periode jam selama 24 jam, dapat dibuat perbandingan emisi total untuk periode perhari antara PLTU Paiton dengan *Refined Genetic Algorithm*, perbandingan tersebut dapat dilihat pada tabel 4.31.

Tabel 4.31.
Perbandingan Total Emisi (NOx) Perhari Pada
PLTU Paiton Dan *Refined Genetic Algorithm*
Dengan Memperhatikan Kendala Emisi

Periode Waktu	PLTU Paiton (Ton/hari)	*Refined Genetic Algorithm* (Ton/hari)
Minggu, 8 Februari 2004	35.1773	36.843
Senin, 9 Februari 2004	40.7609	42.0821
Selasa, 10 Februari 2004	41.0661	43.0609
Total	**117.0043**	**121.986**

Grafik 4.2.
Perbandingan Total Output Emisi (NOx) Perhari Pada
PLTU Paiton Dan *Refined Genetic Algorithm* Dengan
Memperhatikan Kendala Emisi



Pada tabel 4.29, 4.30, 4.31 dan grafik 4.2 juga terjadi penambahan atau maksimalisasi output emisi yang dilakukan *Refined Genetic Algorithm* untuk mencapai fungsi tujuan yaitu biaya operasi yang lebih murah dan pemenuhan permintaan beban. Penambahan output emisi NOx juga sangat mempengaruhi karakteristik emisi sebagai fungsi dari daya masing-masing unit pembangkit yang beragam.

Pengurangan daya yang dibangkitkan pada unit 1, 2 dan 7, berakibat penurunan output emisi NOx unit-unit tersebut. Sedangkan penambahan daya yang dibangkitkan pada unit 5 dan 6 memiliki kecenderungan (*trend*) menambah output emisi NOx yang dihasilkan. Pada unit 8, pengurangan daya yang dibangkitkan dari 30,930.95 MW menjadi 30,675.98 MW tidak otomatis mengurangi output emisi NOx sebaliknya menambah output emisi NOx dari 22.7658 ton/3 hari menjadi 25.9906 ton/3 hari.

Dari analisis perhitungan, didapatkan penambahan output emisi NOx masih dibawah baku mutu emisi yang telah ditentukan yaitu 753.4224 Ton/72 jam untuk NOx.

4.8.4. Perbandingan Tingkat Optimum Biaya Operasi Pembangkitan

Dari tabel 4.9 hingga tabel 4.26, dapat dibuat suatu perbandingan biaya operasi tiap jam antara PLTU Paiton dan hasil perhitungan *Refined Genetic Algorithm* seperti pada tabel dibawah ini, maka pada hasil perhitungan *Refined Genetic Algorithm* terlihat adanya pengurangan biaya operasi yakni sebagai berikut :

Tabel 4.32.
Perbandingan Biaya Tiap Jam Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi

Jam	Minggu 8-02-2004		Senin 9-02-2004		Selasa 10-02-2004	
	PLTU Paiton ($)	*Refined Genetic Algorithm* ($)	PLTU Paiton ($)	*Refined Genetic Algorithm* ($)	PLTU Paiton ($)	*Refined Genetic Algorithm* ($)
00.00	101,667	93,168	104,259	90,999	105,502	91,037
01.00	98,511	88,776	98,232	84,068	99,209	86,793
02.00	95,584	87,839	91,280	80,666	94,640	86,811
03.00	93,776	85,119	87,340	79,874	94,448	86,304
04.00	89,801	83,907	86,213	79,846	93,909	85,924
05.00	89,547	83,751	86,244	81,691	93,367	86,918
06.00	89,549	79,722	88,058	79,024	94,781	82,016
07.00	80,369	72,196	85,432	75,342	90,050	77,400
08.00	65,509	72,497	81,161	82,351	84,454	85,177
09.00	74,360	72,452	89,505	97,127	93,644	95,307
10.00	74,545	72,463	103,433	101,123	102,252	101,387
11.00	74,006	72,278	107,437	101,004	107,395	101,113
12.00	71,996	71,500	107,372	96,340	107,363	95,353
13.00	74,432	72,408	104,180	93,737	103,099	93,624
14.00	62,358	72,252	101,419	100,453	101,328	100,757
15.00	74,481	72,446	106,780	98,555	107,004	101,709
16.00	79,279	74,371	105,582	100,041	107,842	101,302
17.00	82,652	79,881	106,349	101,265	107,516	101,745
18.00	90,225	93,134	107,505	112,126	107,890	102,880
19.00	101,013	109,284	114,569	120,488	109,396	117,328
20.00	112,948	110,531	120,833	120,061	117,894	120,105
21.00	113,987	110,296	120,337	120,604	120,380	120,539
22.00	112,236	108,190	120,766	109,644	120,666	110,875
23.00	111,614	95,314	113,688	98,219	114,277	98,612
TOTAL	**2,114,445**	**2,033,775**	**2,437,974**	**2,304,648**	**2,478,306**	**2,331,016**

Bila dibuat suatu perbandingan biaya tiap pembangkit untuk waktu 24 jam pada tabel 4.33, maka terlihat adanya naik atau turunnya biaya yang dihasilkan dari hasil proses *Refined Genetic Algorithm*.

Tabel 4.33.
Perbandingan Total Biaya Yang Dihasilkan Tiap 24 Jam Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi

Periode Waktu	PLTU Paiton	PLTU Paiton ($)	*Refined Genetic Algorithm* ($)
Minggu, 8 Februari 2004	UNIT 1	348,747	255,405
Minggu, 8 Februari 2004	UNIT 2	326,320	275,817
Minggu, 8 Februari 2004	UNIT 5	261,888	333,419
Minggu, 8 Februari 2004	UNIT 6	260,897	319,247
Minggu, 8 Februari 2004	UNIT 7	503,949	454,284
Minggu, 8 Februari 2004	UNIT 8	412,644	395,603
Senin, 9 Februari 2004	UNIT 1	415,322	278,049
Senin, 9 Februari 2004	UNIT 2	406,099	288,678
Senin, 9 Februari 2004	UNIT 5	294,485	369,058
Senin, 9 Februari 2004	UNIT 6	315,541	374,932
Senin, 9 Februari 2004	UNIT 7	527,729	504,750
Senin, 9 Februari 2004	UNIT 8	478,798	489,181
Selasa, 10 Februari 2004	UNIT 1	426,314	274,668
Selasa, 10 Februari 2004	UNIT 2	416,655	281,560
Selasa, 10 Februari 2004	UNIT 5	298,529	378,976
Selasa, 10 Februari 2004	UNIT 6	317,238	386,541
Selasa, 10 Februari 2004	UNIT 7	536,368	521,432
Selasa, 10 Februari 2004	UNIT 8	483,202	487,839

Bila total biaya operasi selama 72 jam untuk tiap-tiap pembangkit dihitung dengan kedua cara, yaitu PLTU Paiton dan metode *Refined Genetic Algorithm* dengan memperhatikan kendala emisi dapat dilihat pada tabel 4.34.

Tabel 4.34.
Perbandingan Biaya Yang Dihasilkan Selama Tiga Hari
Pada PLTU Paiton Dan *Refined Genetic Algorithm*
Dengan Memperhatikan Kendala Emisi

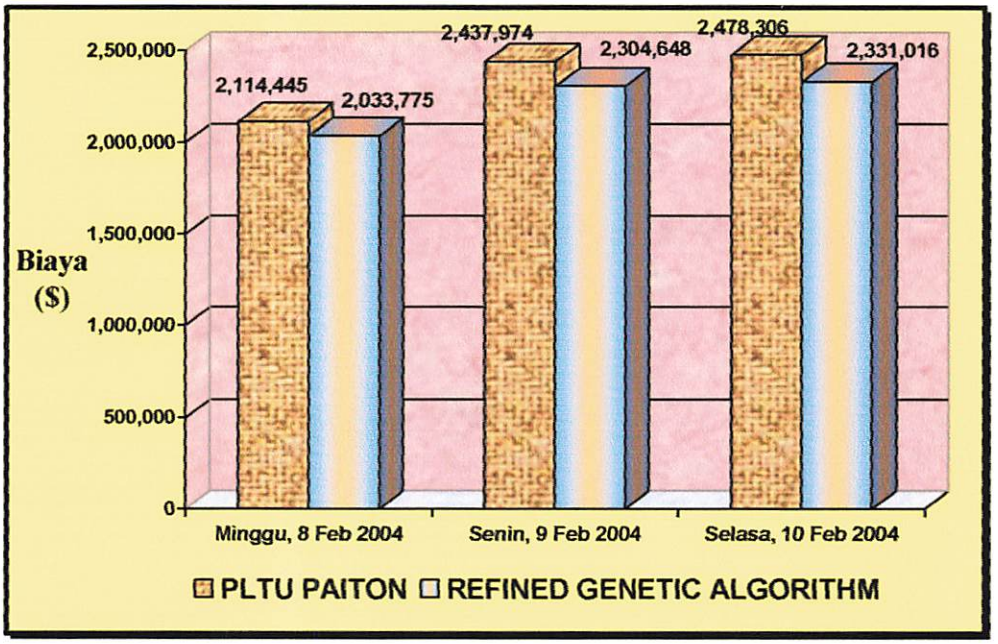
PLTU Paiton	PLTU Paiton ($)	*Refined Genetic Algorithm* ($)
UNIT 1	1,190,383	808,122
UNIT 2	1,149,074	846,055
UNIT 5	854,902	1,081,453
UNIT 6	893,676	1,080,720
UNIT 7	1,568,046	1,480,466
UNIT 8	1,374,644	1,372,623

Dari tabel diatas, dapat dibuat tabel perbandingan total biaya seperti pada tabel 4.35.

Tabel 4.35.
Perbandingan Total Biaya Perhari Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi

Periode Waktu	PLTU Paiton ($/hari)	*Refined Genetic Algorithm* ($/hari)
Minggu, 8 Februari 2004	2,114,445	2,033,775
Senin, 9 Februari 2004	2,437,974	2,304,648
Selasa, 10 Februari 2004	2,478,306	2,331,016
Total	**7,030,725**	**6,669,439**

Grafik 4.3.
Perbandingan Total Biaya Perhari Pada PLTU Paiton Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi



Tabel 4.32, 4.33, 4.34, 4.35 dan grafik 4.3 menunjukkan bahwa dari total biaya per hari operasional yang dikeluarkan oleh PLTU Paiton dengan hasil optimasi menggunakan metode *Refined Genetic Algorithm* dengan memperhatikan kendala emisi, terdapat perbedaan yang relatif besar, dimana hasil yang diperoleh pada optimasi dengan menggunakan metode *Refined Genetic Algorithm* menghasilkan penghematan yang relatif besar. Adapun besar persentase keuntungan yang diperoleh adalah :

$$Profit\ \% = \frac{Total\ Biaya\ Operasi\ PLTU - Total\ Biaya\ Operasi\ RGA}{Total\ Biaya\ Operasi\ PLTU}\ x\ 100\%$$

Pada hari Minggu, 8 Februari 2004, total biaya operasi PLTU Paiton adalah US $ 2,114,445. Sedangkan total biaya operasi dengan metode *Refined Genetic Algorithm* adalah US $ 2,033,775 atau terdapat penghematan biaya sebesar 3.82 %.

Pada hari Senin, 9 Februari 2004, total biaya operasi PLTU Paiton adalah US $ 2,437,974. Sedangkan total biaya operasi dengan metode *Refined Genetic Algorithm* adalah US $ 2,304,648, atau terdapat penghematan biaya sebesar 5.47 %.

Pada hari Selasa, 10 Februari 2004, total biaya operasi PLTU Paiton adalah US $ 2,478,306. Sedangkan total biaya operasi dengan metode *Refined Genetic Algorithm* adalah US $ 2,331,016 atau terdapat penghematan biaya sebesar 5.94 %.

Jika dibuat suatu perbandingan total biaya operasi tiap 24 jam antara PLTU Paiton, hasil perhitungan RGA Standar dan hasil perhitungan RGA dengan kendala emisi, maka pada hasil perhitungan RGA terlihat adanya optimasi biaya yakni sebagai berikut:

Tabel 4.36.
Perbandingan Total Biaya Perhari PLTU Paiton, *Refined Genetic Algorithm* Tanpa Kendala Emisi Dan *Refined Genetic Algorithm* Dengan Memperhatikan Kendala Emisi

Periode Waktu	PLTU Paiton ($ / hari)	*Refined Genetic Algorithm* Dengan Kendala Emisi ($ / hari)	*Refined Genetic Algorithm* Tanpa Kendala Emisi ($ / hari)
Minggu, 8 Feb 2004	2,114,445	2,033,775	2,033,775
Senin, 9 Feb 2004	2,437,974	2,304,648	2,304,648
Selasa, 10 Feb 2004	2,478,306	2,331,016	2,331,016
Total	7,030,725	6,669,439	6,669,439

Perbandingan antara total biaya operasi perhari antara metode RGA tanpa kendala emisi dan metode RGA yang memperhatikan kendala emisi pada tabel 4.36 menunjukkan nilai yang sama. Hal ini disebabkan karena input data yang diambil selama tiga hari tidak terdapat adanya pelanggaran baku mutu emisi yang cukup berarti. Apabila terjadi kelebihan emisi, maka terjadi pelanggaran baku mutu emisi. Sehingga berpengaruh terhadap total biaya. Kasus RGA yang memperhatikan kendala emisi pada skripsi ini, pencarian solusi optimum lebih dipengaruhi oleh kendala operasi pembangkitan (P_{min} dan P_{mak}) dan kendala keseimbangan daya.

BAB V
PENUTUP

5.1. Kesimpulan

Dari hasil perhitungan dan analisa data hasil perhitungan terhadap penggunaan metode *Refined Genetic Algorithm* dengan memperhatikan kendala emisi pada optimasi PLTU Paiton pada tanggal 8, 9 dan 10 Februari 2004, maka dapat diambil kesimpulan sebagai berikut :

1. Penyelesaian masalah *economic dispatch* yang memperhatikan kendala emisi dengan menggunakan metode *Refined Genetic Algorithm* pada sub sistem Paiton menghasilkan biaya operasi pembangkitan yang lebih optimum. Total biaya operasi hasil perhitungan PLTU Paiton selama 72 jam (3 hari) adalah US $ 7,030,725. Sedangkan total biaya operasi dengan metode *Refined Genetic Algorithm* adalah US $ 6,669,439. Selisih total biayanya sekitar US $ 361,286 atau terdapat penghematan biaya sebesar 5.14 %.
2. Terjadi penambahan output emisi NOx selama 72 jam (3 hari) PLTU Paiton menghasilkan output emisi sebesar 117.0043 ton, sedangkan proses metode *Refined Genetic Algorithm* menghasilkan output emisi sebesar 121.986 ton. Terdapat penambahan output emisi NOx 4.982 ton atau sekitar 4.25 %.
3. Dari analisis perhitungan, didapatkan penambahan output emisi NOx masih dibawah baku mutu emisi yang telah ditentukan, yaitu 753.4224 Ton/72 jam..

5.2. Saran

1. Sebagai studi lebih lanjut, *economic dispatch* yang menggunakan metode *Refined Genetic Algorithm* dengan memperhatikan kendala emisi ini dapat dikembangkan lagi terhadap sistem jaringan tenaga listrik yang lebih luas dan memperhitungkan rugi-rugi transmisi, penambahan masalah seperti memasukkan masalah emisi COx, SOx serta jenis batubara yang digunakan, partikulat sebagai pertimbangan kendala emisi dan menambah jumlah pembangkit yang akan dioptimasi.
2. Perlu adanya perbandingan dengan metode lain mengenai cara pencarian kurva fungsi atau karakteristik emisi, sehingga diketahui metode mana yang menghasilkan nilai fungsi hampiran yang lebih mendekati nilai sejatinya.

DAFTAR PUSTAKA

[1]. Abdul Kadir, Prof. Ir., *"Energi Sumber Daya, Inovasi, Tenaga Listrik Dan Potensi Ekonomi"*, UI Press, 1995.

[2]. Allen. J. Wood dan W.F. Bruce, *"Power Generation, Operation and Control"* , John Wiley and Sons, 1984.

[3]. David E. Goldberg, *"Genetic Algorithms in Search, Optimization, and Machine Learning,"* The University of Alabama, Addison-Wesley Publishing Company, INC.

[4]. Djiteng Marsudi, Ir., *"Operasi Sistem Tenaga Listrik"*, diterbitkan oleh Balai Penerbit & Humas ISTN, Bhumi Srengseng Indah, PS. Minggu, Jakarta Selatan.

[5]. G. B. Sheble and K. Brittig, *"Refined Genetic Algorithm : Economic Dispatch Example"*, IEEE Transactions on Power System, vol: 10, no: 1, November 1995, pp 117-124.

[6]. H Ma, A.A. El-Keib and R.E. Smith., *"A Genetic Algorithm – Based Approach To Economic Dispatch Of Power Systems"*, IEEE Transaction On Power System, 1994.

[7]. J W Lamont and E V Obessis. *"Emission Dispatch Models and Algorithms for the* 1990*'s"* IEEE Transactions on Power System, vol 10, no: 2, May 1995, pp 941-947.

[8]. M Sudhakaran, S M R Slochanal, R Sreeram, dan N Chandrasekhar, *"Application of Refined Genetic Algorithm to Combined Economic and Emission Dispatch"*, IE (I) Journal-EL, vol: 85, September 2004.

[9]. Zuhal, Prof. Dr. Ir., *"Ketenagalistrikan Indonesia"*, PT. Ganesa Prima, April 1995.

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK ENERGI LISTRIK

LEMBAR BIMBINGAN SKRIPSI

1.	Nama	: PURWANTO
2.	N.I.M	: 99.12.140
3.	Jurusan	: Teknik Elektro S-1
4.	Konsentrasi	: Teknik Energi Listrik
5.	Judul Skripsi	: *ECONOMIC DISPATCH* UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON
6.	Tanggal Mengajukan Skripsi	: 22 Juli 2005
7.	Tanggal Menyelesaikan Skripsi	: 27 Maret 2006
8.	Dosen Pembimbing	: Ir.Yusuf Ismail Nakhoda, MT.
9.	Telah dievaluasi dengan nilai	: 87 (Delapan Puluh Tujuh)

Malang, April 2006

Mengetahui,	Disetujui,
Ketua Jurusan Teknik Elektro S-1	Dosen Pembimbing
Ir. F. Yudi Limpraptono, MT.	Ir. Yusuf Ismail Nakhoda, MT.
NIP.Y. 103 95 00274	NIP.Y. 101 88 00189

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

FORMULIR BIMBINGAN SKRIPSI

Nama : PURWANTO
Nim : 99.12.140
Masa Bimbingan : 22 JANUARI 2006 s/d 22 JULI 2006
Judul Skripsi : ***ECONOMIC DISPATCH*** **UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE** ***REFINED GENETIC ALGORITHM*** **DENGAN MEMPERHATIKAN KENDALA EMISI PADA PLTU PAITON**

No.	Tanggal	Uraian	Parap Pembimbing
1.	07-01-2006	Perbaiki sistematika penulisan pada Bab I, II dan III.	
2.	14-01-2006	Konsultasi Bab IV serta revisi Bab III	
3.	18-01-2006	Bab IV :Cek uji validasi, apakah sesuai dengan hasil pada jurnal.	
4.	27-01-2006	Bab IV :Periksa kembali selisih biaya program, apakah sudah optimal.	
5.	02-02-2006	Perbaiki dan periksa lagi masalah selisih total biaya tiap harinya.	
6.	08-02-2006	- Siapkan makalah untuk seminar hasil. - ACC makalah seminar hasil.	
7.	11-02-2006	Cek ulang abstraksi serta kesimpulan.	
8.	14-02-2006	Siapkan skripsi yang sudah fix sebagai persiapan untuk komprehensif.	
9.	16-02-2006	ACC skripsi.	
10.			

Malang, 16 Februari 2006
Dosen Pembimbing,

Ir. Yusuf Ismail Nakhoda, MTP
Nip. P. 1018800189

Form.S-4b

LAMPIRAN

LAMPIRAN

LISTING PROGRAM *BORLAND DELPHI* 7.0

ECONOMIC DISPATCH UNIT PEMBANGKIT TERMAL MENGGUNAKAN METODE *REFINED GENETIC ALGORITHM* DENGAN MEMPERHATIKAN KENDALA EMISI

LISTING PROGRAM

REFINED GENETIC ALGORITHM

unit uAbout;

```
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms,
  Dialogs;
type
  TfrmAbout = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmAbout: TfrmAbout;
implementation
{$R *.dfm}
end.
```

unit uGABin;

```
interface
uses uUtils,uGenVar,uGenetic,SysUtils,uFitness2,uHasil;
type
  TGABin=class(TGenetic)
  private
    FPcross,FPmutasi,FPflip:double;
    FCrossType:TCrossType;
    FNewParent:TNewParent;
  protected
    function Mutasi(const rAllele:boolean):boolean;
  public
    constructor Create(const
rMaxGen,rPopSize,rLength:integer;
            const rKa,rPcross,rPmutasi,rPflip:double;
            const rCrossType:TCrossType;
            const rNewParent:TNewParent);
    property Pcross:double read FPcross write FPcross;
    property Pmutasi:double read FPmutasi write FPmutasi;
    property Pflip:double read FPflip write FPflip;
    property CrossType:TCrossType read FCrossType write
FCrossType;
    property NewParent:TNewParent read FNewParent write
FNewParent;
  end;
  TGABin2=class(TGABin)
  private
    FNParam:integer;
    FBatas:TBatasArr1;
    FParent,FChild:TPopBin2;
    FBestIndi:TIndiBin2;
    function getBatas:TBatasArr1;
    procedure setBatas(const rBatas:TBatasArr1);
    function DecodePgenToChrom(const rPgen:dArr1):bArr2;
    function getIndividu(const rIndi:TIndiBin2):TIndiBin2;
    procedure SwapIndi(var rIndi1,rIndi2:TIndiBin2);
    function DecodeFitness(var rChrom:bArr2):double;
    function FindIndiMax:TIndiBin2;
    procedure InitParent;
    procedure Statistik;
    function Seleksi:integer;
    procedure Crossover(const rParent1,rParent2:bArr2;
          var rChild1,rChild2:bArr2);
    procedure ArithmeticCrossover(const
rParent1,rParent2:bArr2;
          var rChild1,rChild2:bArr2);
    procedure Generasi;
    procedure FindNewParent;
    procedure doHitung;
    function getBestChrom:bArr2;
    function getParent:TPopDouble1;
  public
    constructor Create(const
rMaxGen,rPopSize,rLength,rNParam:integer;
            const rKa,rPcross,rPmutasi,rPflip:double;
            const rCrossType:TCrossType;
            const rNewParent:TNewParent;
            const rBatas:TBatasArr1);
    function DecodeChromToPgen(const
rChrom:bArr2):dArr1;
    property Batas:TBatasArr1 read getBatas write setBatas;
    property BestChrom:bArr2 read getBestChrom;
    property Parent:TPopDouble1 read getParent;
  end;
implementation
//constructor
constructor TGABin.Create(const
rMaxGen,rPopSize,rLength:integer;
        const rKa,rPcross,rPmutasi,rPflip:double;
        const rCrossType:TCrossType;
        const rNewParent:TNewParent);
begin
  inherited Create(rMaxGen,rPopSize,rLength,rKa);
  FPcross:=rPcross;
  FPmutasi:=rPmutasi;
  FPflip:=rPflip;
  FCrossType:=rCrossType;
  FNewParent:=rNewParent;
end;
//data proccessing
function TGABin.Mutasi(const rAllele:boolean):boolean;
begin
  if FRandom.NextBoolean(FPMutasi)=true then
  begin
    result:=not rAllele;
  end
  else
  begin
    result:=rAllele;
  end;
end;

//constructor
```

```
constructor TGABin2.Create(const
rMaxGen,rPopSize,rLength,rNParam:integer;
        const rKa,rPcross,rPmutasi,rPflip:double;
        const rCrossType:TCrossType;
        const rNewParent:TNewParent;
        const rBatas:TBatasArr1);
var i:integer;
begin
 inherited
Create(rMaxGen,rPopSize,rLength,rKa,rPcross,rPmutasi,
       rPflip,rCrossType,rNewParent);
 FNParam:=rNParam;
 if (high(rBatas)+1)<>rNParam then
 begin
  raise Exception.Create('Jumlah Param dan Batas tidak
sama!');
 end;
 SetLength(FBatas,FNParam);
 for i:=0 to FNParam-1 do
 begin
  FBatas[i].min:=rBatas[i].min;
  FBatas[i].max:=rBatas[i].max;
 end;
end;
//data accessing
function TGABin2.getBatas:TBatasArr1;
var i:integer;
begin
 SetLength(result,FNParam);
 for i:=0 to FNParam-1 do
 begin
  result[i].min:=FBatas[i].min;
  result[i].max:=FBatas[i].max;
 end;
end;
procedure TGABin2.setBatas(const rBatas:TBatasArr1);
var i:integer;
begin
 if FNParam<>(high(rBatas)+1) then
 begin
  raise Exception.Create('Indeks Matrik Batas dan Length
tidak sama!');
 end;
 SetLength(FBatas,Length);
 for i:=0 to FNParam-1 do
 begin
  FBatas[i].min:=rBatas[i].min;
  FBatas[i].max:=rBatas[i].max;
 end;
end;
//data proccessing
function TGABin2.DecodePgenToChrom(const
rPgen:dArr1):bArr2;
var i:integer;
   tmp:dArr1;
begin
 SetLength(tmp,FNParam);
 for i:=0 to FNParam-1 do
 begin
tmp[i]:=GetRealToBatas(rPgen[i],FBatas[i].min,FBatas[i].m
ax);
 end;
 result:=DecodeFloat2ToBinBase0(Length,tmp);
end;

function TGABin2.DecodeChromToPgen(const
rChrom:bArr2):dArr1;
var i:integer;
   tmp:dArr1;
begin
 tmp:=DecodeBinToFloat2Base0(rChrom);
 SetLength(result,FNParam);
 for i:=0 to FNParam-1 do
 begin
result[i]:=GetBatasToReal(tmp[i],FBatas[i].min,FBatas[i].ma
x);
 end;
end;
function TGABin2.getIndividu(const
rIndi:TIndiBin2):TIndiBin2;
var i,j:integer;
begin
 SetLength(result.chrom,FNParam,Length);
 for i:=0 to FNParam-1 do
 begin
  for j:=0 to Length-1 do
  begin
   result.chrom[i,j]:=rIndi.chrom[i,j];
  end;
 end;
 result.fitness:=rIndi.fitness;
end;
procedure TGABin2.SwapIndi(var rIndi1,rIndi2:TIndiBin2);
var tmpIndi:TIndiBin2;
begin
 tmpIndi:=getIndividu(rIndi1);
 rIndi1:=getIndividu(rIndi2);
 rIndi2:=getIndividu(tmpIndi);
end;
function TGABin2.DecodeFitness(var
rChrom:bArr2):double;
var Pgen:dArr1;
begin
 Pgen:=DecodeChromToPgen(rChrom);
 result:=gFitness2.doCalcCost(Pgen);
 rChrom:=DecodePgenToChrom(Pgen);
end;
function TGABin2.FindIndiMax:TIndiBin2;
var i:integer;
begin
 result:=getIndividu(FParent[0]);
 for i:=1 to PopSize-1 do
 begin
  if result.fitness<FParent[i].fitness then
  begin
   result:=getIndividu(FParent[i]);
  end;
 end;
end;
procedure TGABin2.InitParent;
var i,j,k:integer;
begin
 SetLength(FParent,PopSize);
 SetLength(FChild,PopSize);
 SetLength(FMin,MaxGen);
 SetLength(FAvg,MaxGen);
 SetLength(FMax,MaxGen);
 for i:=0 to PopSize-1 do
 begin
```

```
    SetLength(FParent[i].chrom,FNParam,Length);
    for j:=0 to FNParam-1 do
    begin
     for k:=0 to Length-1 do
     begin
      FParent[i].chrom[j,k]:=FRandom.NextBoolean(Pflip);
     end;
    end;
    FParent[i].fitness:=Ka/DecodeFitness(FParent[i].chrom);
   end;
  end;
  procedure TGABin2.Statistik;
  var i:integer;
  begin
   SumFitness:=FParent[0].fitness;
   Min1:=FParent[0].fitness;
   Max1:=FParent[0].fitness;
   for i:=1 to PopSize-1 do
   begin
    if Min1>FParent[i].fitness then
    begin
     Min1:=FParent[i].fitness;
    end;
    if Max1<FParent[i].fitness then
    begin
     Max1:=FParent[i].fitness;
    end;
    SumFitness:=SumFitness+FParent[i].fitness;
   end;
   Avg1:=SumFitness/PopSize;
  end;
  function TGABin2.Seleksi:integer;
  var i:integer;
     randFit,partSum:double;
  begin
   i:=0;
   partSum:=0.0;
   randFit:=FRandom.NextDouble*SumFitness;
   repeat
    inc(i);
    partSum:=partSum+FParent[i-1].fitness;
   until (partSum>randFit) or (i=PopSize);
   result:=i-1;
  end;
  procedure TGABin2.Crossover(const
  rParent1,rParent2:bArr2;
         var rChild1,rChild2:bArr2);
  var i,j,point1,point2,sa:integer;
  begin
   SetLength(rChild1,FNParam,Length);
   SetLength(rChild2,FNParam,Length);
   if FRandom.NextBoolean(Pcross)=true then
   begin
    if CrossType=crOne then
    begin
     point1:=FRandom.NextInt(0,FNParam*Length-2);
     sa:=0;
     for i:=0 to FNParam-1 do
     begin
      for j:=0 to Length-1 do
      begin
       if sa<=point1 then
       begin
        rChild1[i,j]:=Mutasi(rParent2[i,j]);
        rChild2[i,j]:=Mutasi(rParent1[i,j]);
       end
       else
       begin
        rChild1[i,j]:=Mutasi(rParent1[i,j]);
        rChild2[i,j]:=Mutasi(rParent2[i,j]);
       end;
       inc(sa);
      end;
     end;
    end
    else if CrossType=crTwo then
    begin
     point1:=FRandom.NextInt(0,FNParam*Length-2);
     repeat
      point2:=FRandom.NextInt(0,FNParam*Length-2);
     until point2<>point1;
     if point2<point1 then
     begin
      Swap(point1,point2);
     end;
     sa:=0;
     for i:=0 to FNParam-1 do
     begin
      for j:=0 to Length-1 do
      begin
       if sa<=point1 then
       begin
        rChild1[i,j]:=Mutasi(rParent2[i,j]);
        rChild2[i,j]:=Mutasi(rParent1[i,j]);
       end
       else if (sa>point1) and (sa<=point2) then
       begin
        rChild1[i,j]:=Mutasi(rParent1[i,j]);
        rChild2[i,j]:=Mutasi(rParent2[i,j]);
       end
       else
       begin
        rChild1[i,j]:=Mutasi(rParent2[i,j]);
        rChild2[i,j]:=Mutasi(rParent1[i,j]);
       end;
       inc(sa);
      end;
     end;
    end
    else if CrossType=crMulti then
    begin
     for i:=0 to FNParam-1 do
     begin
      for j:=0 to Length-1 do
      begin
       if FRandom.NextBoolean(Pcross)=true then
       begin
        rChild1[i,j]:=Mutasi(rParent2[i,j]);
        rChild2[i,j]:=Mutasi(rParent1[i,j]);
       end
       else
       begin
        rChild1[i,j]:=Mutasi(rParent1[i,j]);
        rChild2[i,j]:=Mutasi(rParent2[i,j]);
       end;
      end;
     end;
    end
```

```
    else if CrossType=crArithmetic then
    begin
      ArithmeticCrossover(rParent1,rParent2,rChild1,rChild2);
    end;
   end
   else
   begin
    for i:=0 to FNParam-1 do
    begin
     for j:=0 to Length-1 do
     begin
      rChild1[i,j]:=Mutasi(rParent1[i,j]);
      rChild2[i,j]:=Mutasi(rParent2[i,j]);
     end,
    end;
   end;
end;
procedure TGABin2.ArithmeticCrossover(const
rParent1,rParent2:bArr2;
       var rChild1,rChild2:bArr2);
var i,j:integer;
   rand:double;
   X1,X2,X1a,X2a:dArr1;
begin
  X1:=DecodeBinToFloat2Base0(rParent1);
  X2:=DecodeBinToFloat2Base0(rParent2);
  for i:=0 to FNParam-1 do
  begin
X1[i]:=GetBatasToReal(X1[i],FBatas[i].min,FBatas[i].max);
X2[i]:=GetBatasToReal(X2[i],FBatas[i].min,FBatas[i].max);
  end;
  SetLength(X1a,FNParam);
  SetLength(X2a,FNParam);
  for i:=0 to FNParam-1 do
  begin
   rand:=random;
   X1a[i]:=rand*X1[i]+(1-rand)*X2[i];
   if X1a[i]>FBatas[i].max then X1a[i]:=FBatas[i].max;
   if X1a[i]<FBatas[i].min then X1a[i]:=FBatas[i].min;
   X2a[i]:=(1-rand)*X1[i]+rand*X2[i];
   if X2a[i]>FBatas[i].max then X2a[i]:=FBatas[i].max;
   if X2a[i]<FBatas[i].min then X2a[i]:=FBatas[i].min;
  end;
  for i:=0 to FNParam-1 do
  begin
X1a[i]:=GetRealToBatas(X1a[i],FBatas[i].min,FBatas[i].max
);
X2a[i]:=GetRealToBatas(X2a[i],FBatas[i].min,FBatas[i].max
);
  end;
  rChild1:=DecodeFloat2ToBinBase0(Length,X1a);
  rChild2:=DecodeFloat2ToBinBase0(Length,X2a);
  for i:=0 to FNParam-1 do
  begin
   for j:=0 to Length-1 do
   begin
    rChild1[i,j]:=mutasi(rChild1[i,j]);
    rChild2[i,j]:=mutasi(rChild2[i,j]);
   end;
  end;
end;
procedure TGABin2.Generasi;
var i,mate1,mate2:integer;
begin
  i:=0;
  repeat
   mate1:=Seleksi;
   mate2:=Seleksi;
   Crossover(FParent[mate1].chrom,FParent[mate2].chrom,
   FChild[i].chrom,FChild[i+1].chrom);
   FChild[i].fitness:=Ka/DecodeFitness(FChild[i].chrom);
FChild[i+1].fitness:=Ka/DecodeFitness(FChild[i+1].chrom);
   i:=i+2;
  until i>PopSize-1;
end;
procedure TGABin2.FindNewParent;
var i,sa,j:integer;
   tmpPop:TPopBin2;
begin
  if NewParent=npStandart then
  begin
   for i:=0 to PopSize-1 do
   begin
    FParent[i]:=getIndividu(FChild[i]);
   end;
  end
  else if NewParent=npReplikasi then
  begin
   SetLength(tmpPop,PopSize);
   for i:=0 to PopSize-1 do
   begin
    repeat
     sa:=FRandom.NextInt(0,PopSize-1);
    until sa<>i;
    if FChild[i].fitness>FParent[sa].fitness then
    begin
     tmpPop[i]:=getIndividu(FChild[i]);
    end
    else
    begin
     tmpPop[i]:=getIndividu(FParent[sa]);
    end;
   end;
   for i:=0 to PopSize-1 do
   begin
    FParent[i]:=getIndividu(tmpPop[i]);
   end;
  end
  else if NewParent=npElitism then
  begin
   SetLength(tmpPop,2*PopSize);
   for i:=0 to PopSize-1 do
   begin
    tmpPop[i]:=getIndividu(FParent[i]);
    tmpPop[PopSize+i]:=getIndividu(FChild[i]);
   end;
   for i:=0 to 2*PopSize-2 do
   begin
    for j:=i to 2*PopSize-1 do
    begin
     if tmpPop[i].fitness<tmpPop[j].fitness then
     begin
      SwapIndi(tmpPop[i],tmpPop[j]);
     end;
    end;
   end;
   for i:=0 to PopSize-1 do
   begin
```

```
      FParent[i]:=getIndividu(tmpPop[i]);
    end;
  end;
end;
procedure TGABin2.doHitung;
var gen:integer;
   TempIndi:TIndiBin2;
begin
 InitParent;
 Statistik;
 FBestIndi:=FindIndiMax;
 gen:=1;
 repeat
  Generasi;
  FindNewParent;
  Statistik;
  TempIndi:=FindIndiMax;
  if FBestIndi.fitness<TempIndi.fitness then
  begin
   FBestIndi:=GetIndividu(TempIndi);
  end;
  FMin[gen-1]:=Min1;
  FAvg[gen-1]:=Avg1;
  FMax[gen-1]:=Max1;
  frmHasil.pbIterasi.StepBy(1);
  inc(gen);
 until (gen>MaxGen);
end;
function TGABin2.getBestChrom:bArr2;
var i,j:integer;
begin
 doHitung;
 SetLength(result,FNParam,Length);
 for i:=0 to FNParam-1 do
 begin
  for j:=0 to Length-1 do
  begin
   result[i,j]:=FBestIndi.chrom[i,j];
  end;
 end;
end;
function TGABin2.getParent:TPopDouble1;
var i,j:integer;
  Pgen:dArr1;
begin
 SetLength(result,PopSize);
 SetLength(Pgen,FNParam);
 for i:=0 to PopSize-1 do
 begin
  SetLength(result[i].chrom,FNParam);
  Pgen:=DecodeChromToPgen(FParent[i].chrom);
  for j:=0 to FNParam-1 do
  begin
   result[i].chrom[j]:=Pgen[j];
  end;
  result[i].fitness:=FParent[i].fitness;
 end;
end;
end.
```

unit uGenetic;

```
interface
uses uUtils,uRandom;
type
 TGenetic=class
 private
  FMaxGen,FPopSize,FLength:integer;
  FMin1,FAvg1,FMax1,FSumFitness,FKa:double;
  function getMin:dArr1;
  function getAvg:dArr1;
  function getMax:dArr1;
 protected
  FMin,FAvg,FMax:dArr1;
  FRandom:TRandomu;
 public
  constructor Create(const
rMaxGen,rPopSize,rLength:integer;
          const rKa:double);
  destructor Destroy;override;
  property MaxGen:integer read FMaxGen write FMaxGen;
  property PopSize:integer read FPopSize write FPopSize;
  property Length:integer read FLength write FLength;
  property Ka:double read FKa write FKa;
  property Min1:double read FMin1 write FMin1;
  property Avg1:double read FAvg1 write FAvg1;
  property Max1:double read FMax1 write FMax1;
  property SumFitness:double read FSumFitness write
FSumFitness;
  property Min:dArr1 read getMin;
  property Avg:dArr1 read getAvg;
  property Max:dArr1 read getMax;
 end;
implementation
//constructor
constructor TGenetic.Create(const
rMaxGen,rPopSize,rLength:integer;
       const rKa:double);
begin
 inherited Create;
 FMaxGen:=rMaxGen;
 FPopSize:=rPopSize;
 FLength:=rLength;
 FKa:=rKa;
 FRandom:=TRandomu.Create;
end;
//destructor
destructor TGenetic.Destroy;
begin
 try
  FRandom.Free;
 finally
  inherited Destroy;
 end;
end;
function TGenetic.getAvg:dArr1;
var i:integer;
begin
 SetLength(result,FMaxGen);
 for i:=0 to FMaxGen-1 do
 begin
  result[i]:=FMin[i];
 end;
end;
```

```
function TGenetic.getMax:dArr1;
var i:integer;
begin
  SetLength(result,FMaxGen);
  for i:=0 to FMaxGen-1 do
  begin
    result[i]:=FAvg[i];
  end;
end;
function TGenetic.getMin: dArr1;
var i:integer;
begin
  SetLength(result,FMaxGen);
  for i:=0 to FMaxGen-1 do
  begin
    result[i]:=FMax[i];
  end;
end;
end.
```

unit uRGA;

```
interface
uses uUtils,uGenVar,uGenetic,SysUtils,uFitness2,uHasil;
type
  TRGA=class(TGenetic)
  private
    FBetha:double;
  public
    constructor Create(const
rMaxGen,rPopSize,rLength:integer;
            const rKa,rBetha:double);
    property Betha:double read FBetha write FBetha;
  end;
  TRGA1=class(TRGA)
  private
    FBatas:TBatasArr1;
    FParent,FChild:TPopDouble1;
    FBestIndi:TIndiDouble1;
    function getBatas:TBatasArr1;
    procedure setBatas(const rBatas:TBatasArr1);
    function getParent:TPopDouble1;
    procedure setParent(const rParent:TPopDouble1);
    function getIndividu(const
rIndi:TIndiDouble1):TIndiDouble1;
    function FindIndiMax:TIndiDouble1;
    function FindFitnessMax:double;
    procedure SwapIndi(var rIndi1,rIndi2:TIndiDouble1);
    procedure InitParent;
    procedure Statistik;
    procedure Generasi;
    procedure Kompetisi;
    procedure doHitung;
    function getBestChrom:dArr1;
  public
    constructor Create(const
rMaxGen,rPopSize,rLength:integer;
            const rKa,rBetha:double;
            const rBatas:TBatasArr1);overload;
    constructor Create(const
rMaxGen,rPopSize,rLength:integer;
            const rKa,rBetha:double;
            const rBatas:TBatasArr1;
            const rParent:TPopDouble1);overload;
    property Batas:TBatasArr1 read getBatas write setBatas;
    property BestChrom:dArr1 read getBestChrom;
    property Parent:TPopDouble1 read getParent write
setParent;
  end;
implementation
//constructor
constructor TRGA.Create(const
rMaxGen,rPopSize,rLength:integer;
        const rKa,rBetha:double);
begin
  inherited Create(rMaxGen,rPopSize,rLength,rKa);
  FBetha:=rBetha;
end;
//constructor
constructor TRGA1.Create(const
rMaxGen,rPopSize,rLength:integer;
        const rKa,rBetha:double;
        const rBatas:TBatasArr1);
var i:integer;
begin
  if (high(rBatas)+1)<>rLength then
  begin
    raise Exception.Create('Length dan Batas arraynya tidak
sama!');
  end;
  inherited Create(rMaxGen,rPopSize,rLength,rKa,rBetha);
  SetLength(FBatas,Length);
  for i:=0 to Length-1 do
  begin
    FBatas[i].min:=rBatas[i].min;
    FBatas[i].max:=rBatas[i].max;
  end;
  InitParent;
end;
constructor TRGA1.Create(const
rMaxGen,rPopSize,rLength:integer;
        const rKa,rBetha:double;
        const rBatas:TBatasArr1;
        const rParent:TPopDouble1);
var i:integer;
begin
  if (high(rBatas)+1)<>rLength then
  begin
    raise Exception.Create('Length dan Batas arraynya tidak
sama!');
  end;
  inherited Create(rMaxGen,rPopSize,rLength,rKa,rBetha);
  SetLength(FBatas,Length);
  for i:=0 to Length-1 do
  begin
    FBatas[i].min:=rBatas[i].min;
    FBatas[i].max:=rBatas[i].max;
  end;
  if (high(rParent)+1)<>PopSize then
  begin
    raise Exception.Create('Indeks matrik tidak sama!');
  end;
  SetLength(FParent,PopSize);
  for i:=0 to PopSize-1 do
  begin
    FParent[i]:=getIndividu(rParent[i]);
  end;
  SetLength(FChild,PopSize);
```

```
    SetLength(FMin,MaxGen);
    SetLength(FAvg,MaxGen);
    SetLength(FMax,MaxGen);
    for i:=0 to PopSize-1 do
    begin
      SetLength(FChild[i].chrom,Length);
    end;
    SetLength(FBestIndi.chrom,Length);
  end;
  //data accessing
  function TRGA1.getBatas:TBatasArr1;
  var i:integer;
  begin
    SetLength(result,Length);
    for i:=0 to Length-1 do
    begin
      result[i].min:=FBatas[i].min;
      result[i].max:=FBatas[i].max;
    end;
  end;
  procedure TRGA1.setBatas(const rBatas:TBatasArr1);
  var i:integer;
  begin
    if (high(rBatas)+1)<>Length then
    begin
      raise Exception.Create('Length dan Batas arraynya tidak
  sama!');
    end;
    SetLength(FBatas,Length);
    for i:=0 to Length-1 do
    begin
      FBatas[i].min:=rBatas[i].min;
      FBatas[i].max:=rBatas[i].max;
    end;
  end;
  function TRGA1.getParent:TPopDouble1;
  var i:integer;
  begin
    SetLength(result,PopSize);
    for i:=0 to PopSize-1 do
    begin
      result[i]:=getIndividu(FParent[i]);
    end;
  end;

  procedure TRGA1.setParent(const rParent:TPopDouble1);
  var i:integer;
  begin
    if (high(rParent)+1)<>PopSize then
    begin
      raise Exception.Create('Indeks matrik tidak sama!');
    end;
    SetLength(FParent,PopSize);
    for i:=0 to PopSize-1 do
    begin
      FParent[i]:=getIndividu(rParent[i]);
    end;
  end;
  //data proccessing
  function TRGA1.getIndividu(const
  rIndi:TIndiDouble1):TIndiDouble1;
  var i:integer;
  begin
    SetLength(result.chrom,Length);
    for i:=0 to Length-1 do
    begin
      result.chrom[i]:=rIndi.chrom[i];
    end;
    result.fitness:=rIndi.fitness;
  end;
  function TRGA1.FindIndiMax:TIndiDouble1;
  var i:integer;
  begin
    result:=getIndividu(FParent[0]);
    for i:=1 to PopSize-1 do
    begin
      if result.fitness<FParent[i].fitness then
      begin
        result:=getIndividu(FParent[i]);
      end;
    end;
  end;
  function TRGA1.FindFitnessMax:double;
  var i:integer;
  begin
    result:=FParent[0].fitness;
    for i:=1 to PopSize-1 do
    begin
      if result<FParent[i].fitness then
      begin
        result:=FParent[i].fitness;
      end;
    end;
  end;
  procedure TRGA1.SwapIndi(var
  rIndi1,rIndi2:TIndiDouble1);
  var tmpIndi:TIndiDouble1;
  begin
    tmpIndi:=getIndividu(rIndi1);
    rIndi1:=getIndividu(rIndi2);
    rIndi2:=getIndividu(tmpIndi);
  end;
  procedure TRGA1.InitParent;
  var i,j:integer;
  begin
    SetLength(FParent,PopSize);
    SetLength(FChild,PopSize);
    SetLength(FMin,MaxGen);
    SetLength(FAvg,MaxGen);
    SetLength(FMax,MaxGen);
    for i:=0 to PopSize-1 do
    begin
      SetLength(FParent[i].chrom,Length);
      SetLength(FChild[i].chrom,Length);
    end;
    SetLength(FBestIndi.chrom,Length);
    for i:=0 to PopSize-1 do
    begin
      for j:=0 to Length-1 do
      begin
  FParent[i].chrom[j]:=GetBatasToReal(FRandom.NextDouble
  ,FBatas[j].min,
                      FBatas[j].max);
      end;
  FParent[i].fitness:=Ka/gFitness2.doCalcCost(FParent[i].chro
  m);
    end;
  end;
```

```
procedure TRGA1.Statistik;
var i:integer;
begin
  Min1:=FParent[0].fitness;
  Max1:=FParent[0].fitness;
  SumFitness:=FParent[0].fitness;
  for i:=1 to PopSize-1 do
  begin
    if Min1>FParent[i].fitness then
    begin
      Min1:=FParent[i].fitness;
    end;
    if Max1<FParent[i].fitness then
    begin
      Max1:=FParent[i].fitness;
    end;
    SumFitness:=SumFitness+FParent[i].fitness;
  end;
  Avg1:=SumFitness/PopSize;
end;
procedure TRGA1.Generasi;
var i,j:integer;
    Fmax,fitRGA,fitES:double;
    tho:double;
    chromRGA,chromES:dArr1;
begin
  Fmax:=FindFitnessMax;
  SetLength(chromRGA,Length);
  SetLength(chromES,Length);
  for i:=0 to PopSize-1 do
  begin
    SetLength(FChild[i].chrom,Length);
    for j:=0 to Length-1 do
    begin
      //tho:=Betha*(FBatas[j].max-
FBatas[j].min)*FMax/FParent[i].fitness;
      tho:=(FBatas[j].max-FBatas[j].min)*((Fmax-
FParent[i].fitness)/
          Fmax+Betha);
chromRGA[j]:=FParent[i].chrom[j]+FRandom.NextGaussian
(0,sqr(tho));
      if chromRGA[j]>FBatas[j].max then
      begin
        chromEP[j]:=FBatas[j].max;
      end;
      if chromRGA[j]<FBatas[j].min then
      begin
        chromRGA[j]:=FBatas[j].min;
      end;
    end;
    fitRGA:=Ka/gFitness2.doCalcCost(chromEP);
    for j:=0 to Length-1 do
    begin
chromES[j]:=FParent[i].chrom[j]+FRandom.NextGaussian(0,
Betha);
      if chromES[j]>FBatas[j].max then
      begin
        chromES[j]:=FBatas[j].max;
      end;
      if chromES[j]<FBatas[j].min then
      begin
        chromES[j]:=FBatas[j].min;
      end;
    end;
    fitES:=Ka/gFitness2.doCalcCost(chromES);
    if fitRGA>fitES then
    begin
      for j:=0 to Length-1 do
      begin
        FChild[i].chrom[j]:=chromRGA[j];
      end;
      FChild[i].fitness:=fitRGA;
    end
    else
    begin
      for j:=0 to Length-1 do
      begin
        FChild[i].chrom[j]:=chromES[j];
      end;
      FChild[i].fitness:=fitES;
    end;
  end;
end;

procedure TRGA1.Kompetisi;
var i,j,sa,Ntmp:integer;
    tmpPop:TPopDouble1;
    sort:iArr1;
begin
  Ntmp:=2*PopSize;
  SetLength(tmpPop,Ntmp);
  SetLength(sort,Ntmp);
  for i:=0 to PopSize-1 do
  begin
    tmpPop[i]:=getIndividu(FParent[i]);
    tmpPop[PopSize+i]:=getIndividu(FChild[i]);
    sort[i]:=0;
    sort[PopSize+i]:=0;
  end;
  for i:=0 to Ntmp-1 do
  begin
    for j:=0 to Ntmp-2 do
    begin
      repeat
        sa:=FRandom.NextInt(0,(Ntmp-1));
      until sa<>i;
      if tmpPop[i].fitness>tmpPop[sa].fitness then
      begin
        inc(sort[i]);
      end;
    end;
  end;
  for i:=0 to Ntmp-2 do
  begin
    for j:=i to Ntmp-1 do
    begin
      if sort[i]<sort[j] then
      begin
        Swap(sort[i],sort[j]);
        SwapIndi(tmpPop[i],tmpPop[j]);
      end;
    end;
  end;
  for i:=0 to PopSize-1 do
  begin
    FParent[i]:=getIndividu(tmpPop[i]);
  end;
end;
```

```
procedure TRGA1.doHitung;
var gen:integer;
   TempIndi:TIndiDouble1;
begin
 Statistik;
 FBestIndi:=FindIndiMax;
 gen:=1;
 repeat
  Generasi;
  Kompetisi;
  Statistik;
  TempIndi:=FindIndiMax;
  if FBestIndi.fitness<TempIndi.fitness then
  begin
   FBestIndi:=GetIndividu(TempIndi);
  end;
  FMin[gen-1]:=Min1;
  FAvg[gen-1]:=Avg1;
  FMax[gen-1]:=Max1;
  frmHasil.pbIterasi2.StepBy(1);
  inc(gen);
 until (gen>MaxGen);
end;
function TRGA1.getBestChrom:dArr1;
var i:integer;
begin
 doHitung;
 SetLength(result,Length);
 for i:=0 to Length-1 do
 begin
  result[i]:=FBestIndi.chrom[i];
 end;
end;
end.
```

unit uRandom;

```
interface
type
 TRandomu = class
 private
  FNextGaussian:double;
  procedure GetGaussian(var dv1,dv2:double);
 public
  constructor Create;
  function NextInt(const dmax:integer):integer;overload;
  function NextInt(const
dmin,dmax:integer):integer;overload;
  function NextDouble:double;overload;
  function NextDouble(const
dmin,dmax:double):double;overload;
  function NextBoolean:boolean;overload;
  function NextBoolean(const
dflip:double):boolean;overload;
  function NextGaussian:double;overload;
  function NextGaussian(const
dmean,dvariance:double):double;overload;
  function NextCauchy(const
dmean,dvariance:double):double;
 end;
implementation
constructor TRandomu.Create;
begin
 inherited Create;
  FNextGaussian:=0;
end;
function TRandomu.NextInt(const dmax:integer):integer;
begin
 result:=round(random*dmax);
end;
function TRandomu.NextInt(const
dmin,dmax:integer):integer;
begin
 result:=round(dmin+(dmax-dmin)*random);
end;
function TRandomu.NextDouble:double;
begin
 result:=random;
end;
function TRandomu.NextDouble(const
dmin,dmax:double):double;
begin
 result:=dmin+(dmax-dmin)*random;
end;
function TRandomu.NextBoolean:boolean;
begin
 result:=false;
 if random<=0.5 then
 begin
  result:=true;
 end;
end;
function TRandomu.NextBoolean(const
dflip:double):boolean;
begin
 result:=false;
 if random<=dflip then
 begin
  result:=true;
 end;
end;
procedure TRandomu.GetGaussian(var dv1,dv2:double);
var s,multiplier:double;
begin
 repeat
  dv1:=2*random-1;
  dv2:=2*random-1;
  s:=dv1*dv1+dv2*dv2;
 until s<=1;
 multiplier:=sqrt((-2*Ln(s))/s);
 dv1:=dv1*multiplier;
 dv2:=dv2*multiplier;
end;
function TRandomu.NextGaussian:double;
begin
 if FNextGaussian<>0 then
 begin
  result:=FNextGaussian;
  FNextGaussian:=0;
 end
 else
 begin
  GetGaussian(result,FNextGaussian);
 end;
end;
function TRandomu.NextGaussian(const dmean,
      dvariance:double):double;
var gauss:double;
```

```
 begin
  if FNextGaussian<>0 then
  begin
   result:=dmean+sqrt(dvariance)*FNextGaussian;
   FNextGaussian:=0;
  end
  else
  begin
   GetGaussian(gauss,FNextGaussian);
   result:=dmean+sqrt(dvariance)*gauss;
  end;
 end;
 function TRandomu.NextCauchy(const
 dmean,dvariance:double):double;
 var v1,v2:double;
 begin
  v1:=NextGaussian(dmean,dvariance);
  v2:=NextGaussian(dmean,dvariance);
  if abs(v2)>0.001 then
  begin
   result:=v1/v2;
  end
  else
  begin
   result:=0;
  end;
 end;
 end.
```

unit uComplex;

```
interface
uses uUtils;
type
 TComplex=class
 private
  FReal,FImag:double;
 public
  constructor Create;overload;
  constructor Create(const aReal:double);overload;
  constructor Create(const aReal,aImag:double);overload;
  constructor Create(const aComplex:TComplex);overload;
  function GetAbs:double;
  function GetAngleRad:double;
  function GetAngleDeg:double;
  function Add(const aReal:double):TComplex;overload;
  function Add(const
aComplex:TComplex):TComplex;overload;
  function Subtract(const
aReal:double):TComplex;overload;
  function Subtract(const
aComplex:TComplex):TComplex;overload;
  function Multiply(const
aReal:double):TComplex;overload;
  function Multiply(const
aComplex:TComplex):TComplex;overload;
  function Divide(const aReal:double):TComplex;overload;
  function Divide(const
aComplex:TComplex):TComplex;overload;
  function Conj:TComplex;
  function Negative:TComplex;
  function toStringI(const rLen:integer):string;
  function toStringJ(const rLen:integer):string;
  property Real:double read FReal write FReal;
  property Imag:double read FImag write FImag;
 end;
 CArr1=array of TComplex;
 CArr2=array of array of TComplex;
implementation
{ TComplex }
//constructor
constructor TComplex.Create;
begin
 inherited Create;
 FReal:=0.0;
 FImag:=0.0;
end;
constructor TComplex.Create(const aReal:double);
begin
 inherited Create;
 FReal:=aReal;
 FImag:=0.0;
end;

constructor TComplex.Create(const aReal,aImag:double);
begin
 inherited Create;
 FReal:=aReal;
 FImag:=aImag;
end;
constructor TComplex.Create(const aComplex:TComplex);
begin
 inherited Create;
 FReal:=aComplex.FReal;
 FImag:=aComplex.FImag;
end;
//data operation
function TComplex.Add(const aReal:double):TComplex;
begin
 result:=TComplex.Create((FReal+aReal),FImag);
end;
function TComplex.Add(const
aComplex:TComplex):TComplex;
begin
result:=TComplex.Create((FReal+aComplex.FReal),(FImag+
aComplex.FImag));
end;
function TComplex.Subtract(const aReal:double):TComplex;
begin
 result:=TComplex.Create((FReal-aReal),FImag);
end;
function TComplex.Subtract(const
aComplex:TComplex).TComplex;
begin
 result:=TComplex.Create((FReal-aComplex.FReal),(FImag-
aComplex.FImag));
end;
function TComplex.Multiply(const aReal:double):TComplex;
begin
 result:=TComplex.Create((aReal*FReal),(aReal*FImag));
end;
function TComplex.Multiply(const
aComplex:TComplex):TComplex;
begin
 result:=TComplex.Create((FReal*aComplex.FReal-
FImag*aComplex.FImag),
      (FReal*aComplex.FImag+FImag*aComplex.FReal));
end;
```

```
function TComplex.Divide(const aReal:double):TComplex;
begin
  result:=TComplex.Create((FReal/aReal),(FImag/aReal));
end;
function TComplex.Divide(const
aComplex:TComplex):TComplex;
var denote:double;
begin
  denote:=sqr(aComplex.FReal)+sqr(aComplex.FImag);
result:=TComplex.Create(((FReal*aComplex.FReal+FImag*
aComplex.FImag)/denote),
        ((FImag*aComplex.FReal-
FReal*aComplex.FImag)/denote));
end;
function TComplex.GetAbs:double;
begin
  result:=sqrt(sqr(FReal)+sqr(FImag));
end;
function TComplex.GetAngleRad:double;
begin
  result:=arctan(FImag/FReal);
end;
function TComplex.GetAngleDeg:double;
var phi:double;
begin
  phi:=4*arctan(1);
  result:=GetAngleRad*180/phi;
end;
function TComplex.Conj:TComplex;
begin
  result:=TComplex.Create(FReal,-FImag);
end;
function TComplex.Negative:TComplex;
begin
  result:=TComplex.Create((FReal*-1),(FImag*-1));
end;
function TComplex.toStringI(const rLen:integer):string;
begin
  if FImag<0 then
  begin
    result:=RealToStr(FReal,rLen)+'-
'+RealToStr(FImag,rLen)+'i';
  end
  else
  begin
result:=RealToStr(FReal,rLen)+'+'+RealToStr(FImag,rLen)+'
i';
  end;
end;
function TComplex.toStringJ(const rLen:integer):string;
begin
  if FImag<0 then
  begin
    result:=RealToStr(FReal,rLen)+'-
j'+RealToStr(FImag,rLen);
  end
  else
  begin
    result:=RealToStr(FReal,rLen)+'+
j'+RealToStr(FImag,rLen);
  end;
end;
end.
```

unit uUtils;

```
interface
uses SysUtils;
type
  TSort=(asc,dec);
  TBatas=record
    min,max:double;
  end;
  TBatasArr1=array of TBatas;
  TBatasArr2=array of array of TBatas;
  dArr1=array of double;
  dArr2=array of array of double;
  iArr1=array of integer;
  iArr2=array of array of integer;
  bArr1=array of boolean;
  bArr2=array of array of boolean;
  sArr1=array of String;
  TAlleleTCSC=record
    Lokasi,TypeAlat,Setting:double;
  end;
  TChromTCSC1=array of TAlleleTCSC;
  TAlleleUpfc=record
    Status:boolean;
    TypeAlat,Tap,Sudut:double;
  end;
  TChromUpfc1=array of TAlleleUpfc;
  function RealToStr(Num:double;Pecahan:byte):String;
  function StrToReal(Huruf:string):double;
  function Pangkat(Val,pangkat:double):double;
  procedure Swap(var X,Y:byte);overload;
  procedure Swap(var X,Y:integer);overload;
  procedure Swap(var X,Y:word);overload;
  procedure Swap(var X,Y:double);overload;
  procedure Swap(var X,Y:extended);overload;
  procedure Swap(var X,Y:string);overload;
  procedure Swap(var X,Y:boolean);overload;
  procedure BubleSort(var aData:dArr1;const
aType:TSort);overload;
  procedure BubleSort(var aData:iArr1;const
aType:TSort);overload;
  procedure BubleSort(var aData:sArr1;const
aType:TSort);overload;
  function DecodeBinToFloat1(const aData:bArr1):double;
  function DecodeBinToFloat2(const aData:bArr2):dArr1;
  function DecodeBinToFloat2Base0(const
aData:bArr2):dArr1;
  function DecodeFloat1ToBin(const aLength:integer;
        const aData:double):bArr1;
  function DecodeFloat2ToBin(const aLength:integer;
        const aData:dArr1):bArr2;
  function DecodeFloat2ToBinBase0(const aLength:integer;
        const aData:dArr1):bArr2;
  function GetBatas(const aValue,aMin,aMax:double):double;
  function GetFlip(const aFlip:double):boolean;
  function GetBatasToReal(const
aValue,aMin,aMax:double):double;
  function GetRealToBatas(const
aValue,aMin,aMax:double):double;
  function ArrayBase0ToBase1(const
rData:bArr1):bArr1;overload;
  function ArrayBase0ToBase1(const
rData:bArr2):bArr2;overload;
```

```
  function ArrayBase0ToBase1(const
rData:dArr1):dArr1;overload;
  function ArrayBase0ToBase1(const
rData:dArr2):dArr2;overload;
  function ArrayBase1ToBase0(const
rData:bArr1):bArr1;overload;
  function ArrayBase1ToBase0(const
rData:bArr2):bArr2;overload;
  function ArrayBase1ToBase0(const
rData:dArr1):dArr1;overload;
  function ArrayBase1ToBase0(const
rData:dArr2):dArr2;overload;
implementation
function RealToStr(Num:double;Pecahan:byte):String;
var Hasil:String;
   le:byte;
begin
  le:=sizeof(Num);
  Str(Num:le:Pecahan,Hasil);
  Result:=Hasil;
end;
function Pangkat(Val,pangkat:double):double;
begin
  Result:=exp(Pangkat*ln(Val));
end;
function StrToReal(Huruf:string):double;
var Temp:double;
   Code:integer;
begin
  val(Huruf,Temp,Code);
  Result:=Temp;
end;
procedure Swap(var X,Y:byte);
var tmp:byte;
begin
  tmp:=X;
  X:=Y;
  Y:=tmp;
end;
procedure Swap(var X,Y:integer);
var tmp:integer;
begin
  tmp:=X;
  X:=Y;
  Y:=tmp;
end;
procedure Swap(var X,Y:word);
var tmp:word;
begin
  tmp:=X;
  X:=Y;
  Y:=tmp;
end;
procedure Swap(var X,Y:double);
var tmp:double;
begin
  tmp:=X;
  X:=Y;
  Y:=tmp;
end;
procedure Swap(var X,Y:extended);
var tmp:extended;
begin
  tmp:=X;
  X:=Y;
  Y:=tmp;
end;
procedure Swap(var X,Y:string);
var tmp:string;
begin
  tmp:=X;
  X:=Y;
  Y:=tmp;
end;
procedure Swap(var X,Y:boolean);
var tmp:boolean;
begin
  tmp:=X;
  X:=Y;
  Y:=tmp;
end;
procedure BubleSort(var aData:dArr1;const aType:TSort);
var i,j:integer;
begin
  for i:=1 to (high(aData)-1) do
  begin
   for j:=i to high(aData) do
   begin
    if aType=asc then
    begin
     if aData[i]>aData[j] then
     begin
      Swap(aData[i],aData[j]);
     end;
    end
    else if aType=dec then
    begin
     if aData[i]<aData[j] then
     begin
      Swap(aData[i],aData[j]);
     end;
    end;
   end;
  end;
end;
procedure BubleSort(var aData:iArr1;const aType:TSort);
var i,j:integer;
begin
  for i:=1 to (high(aData)-1) do
  begin
   for j:=i to high(aData) do
   begin
    if aType=asc then
    begin
     if aData[i]>aData[j] then
     begin
      Swap(aData[i],aData[j]);
     end;
    end
    else if aType=dec then
    begin
     if aData[i]<aData[j] then
     begin
      Swap(aData[i],aData[j]);
     end;
    end;
   end;
  end;
end;
```

```
  end;
  procedure BubleSort(var aData:sArr1;const aType:TSort);
  var i,j:integer;
  begin
   for i:=1 to (high(aData)-1) do
   begin
    for j:=i to high(aData) do
    begin
     if aType=asc then
     begin
      if aData[i]>aData[j] then
      begin
       Swap(aData[i],aData[j]);
      end;
     end
     else if aType=dec then
     begin
      if aData[i]<aData[j] then
      begin
       Swap(aData[i],aData[j]);
      end;
     end;
    end;
   end;
 end;
 function DecodeBinToFloat1(const aData:bArr1):double;
 var i:integer;
    powerof2,sa:double;
 begin
  result:=0;
  powerof2:=1;
  sa:=pangkat(2,high(aData))-1;
  for i:=high(aData) downto 1 do
  begin
   if aData[i]=true then
   begin
    result:=result+powerof2;
   end;
   powerof2:=powerof2*2;
  end;
  result:=result/sa;
 end;
 function DecodeBinToFloat2(const aData:bArr2):dArr1;
 var i,j,rows,cols:integer;
    Data:bArr1;
 begin
  rows:=high(aData);
  cols:=high(aData[0]);
  SetLength(Data,cols+1);
  SetLength(result,rows+1);
  for i:=1 to rows do
  begin
   for j:=1 to cols do
   begin
    Data[j]:=aData[i,j];
   end;
   result[i]:=DecodeBinToFloat1(Data);
  end;
 end;
 function DecodeBinToFloat2Base0(const
 aData:bArr2):dArr1;
 var tmpData:bArr2;
    tmp:dArr1;
 begin
  tmpData:=ArrayBase0ToBase1(aData);
  tmp:=DecodeBinToFloat2(tmpData);
  result:=ArrayBase1ToBase0(tmp);
 end;
 function DecodeFloat1ToBin(const aLength:integer;
       const aData:double):bArr1;
 var i,Sa,cek,value:integer;
 begin
  Sa:=round(pangkat(2,aLength))-1;
  value:=round(aData*Sa);
  SetLength(result,aLength+1);
  for i:=1 to aLength do
  begin
   result[i]:=false;
  end;
  i:=aLength;
  Repeat
   Cek:=value mod 2;
   if Cek=1 then
   begin
    result[i]:=true;
   end
   else
   begin
    result[i]:=false;
   end;
   value:=value div 2;
   i:=i-1;
  until value=0;
 end;
 function DecodeFloat2ToBin(const aLength:integer;
       const aData:dArr1):bArr2;
 var i,j,NData:integer;
    chrom1:bArr1;
 begin
  NData:=high(aData);
  SetLength(result,NData+1,aLength+1);
  for i:=1 to NData do
  begin
   if (aData[i]>1) or (aData[i]<0) then
   begin
    if aData[i]>1 then aData[i]:=1;
    if aData[i]<0 then aData[i]:=0;
    //raise Exception.Create('Data Salah harus diantara 0 dan
 1 ya!');
   end;
   chrom1:=DecodeFloat1ToBin(aLength,aData[i]);
   for j:=1 to aLength do
   begin
    result[i,j]:=chrom1[j];
   end;
  end;
 end;
 function DecodeFloat2ToBinBase0(const aLength:integer;
       const aData:dArr1):bArr2;
 var tmpData:dArr1;
    tmpChrom:bArr2;
 begin
  tmpData:=ArrayBase0ToBase1(aData);
  tmpChrom:=DecodeFloat2ToBin(aLength,tmpData);
  result:=ArrayBase1ToBase0(tmpChrom);
 end;
 function GetBatas(const aValue,aMin,aMax:double):double;
 begin
```

```
  if aValue>1.0 then raise Exception.Create('Value tidak
boleh lebih dari 1');
  if aValue<0.0 then raise Exception.Create('Value tidak
boleh kurang dari 0');
  result:=aMin+aValue*(aMax-aMin);
end;
function GetFlip(const aFlip:double):boolean;
begin
  result:=false;
  if random<=aFlip then result:=true;
end;
function GetBatasToReal(const
aValue,aMin,aMax:double):double;
begin
  result:=aMin+aValue*(aMax-aMin);
end;
function GetRealToBatas(const
aValue,aMin,aMax:double):double;
begin
  result:=(aValue-aMin)/(aMax-aMin);
end;
function ArrayBase0ToBase1(const
rData:bArr1):bArr1;overload;
var i,rows:integer;
begin
  rows:=high(rData)+1;
  SetLength(result,rows+1);
  for i:=0 to rows-1 do
  begin
    result[i+1]:=rData[i];
  end;
end;
function ArrayBase0ToBase1(const
rData:bArr2):bArr2;overload;
var i,j,rows,cols:integer;
begin
  rows:=high(rData)+1;
  cols:=high(rData[0])+1;
  SetLength(result,rows+1,cols+1);
  for i:=0 to rows-1 do
  begin
    for j:=0 to cols-1 do
    begin
      result[i+1,j+1]:=rData[i,j];
    end;
  end;
end;
function ArrayBase0ToBase1(const
rData:dArr1):dArr1;overload;
var i,rows:integer;
begin
  rows:=high(rData)+1;
  SetLength(result,rows+1);
  for i:=0 to rows-1 do
  begin
    result[i+1]:=rData[i];
  end;
end;
function ArrayBase0ToBase1(const
rData:dArr2):dArr2;overload;
var i,j,rows,cols:integer;
begin
  rows:=high(rData)+1;
  cols:=high(rData[0])+1;
  SetLength(result,rows+1,cols+1);
  for i:=0 to rows-1 do
  begin
    for j:=0 to cols-1 do
    begin
      result[i+1,j+1]:=rData[i,j];
    end;
  end;
end;
function ArrayBase1ToBase0(const
rData:bArr1):bArr1;overload;
var i,rows:integer;
begin
  rows:=high(rData);
  SetLength(result,rows);
  for i:=0 to rows-1 do
  begin
    result[i]:=rData[i+1];
  end;
end;
function ArrayBase1ToBase0(const
rData:bArr2):bArr2;overload;
var i,j,rows,cols:integer;
begin
  rows:=high(rData);
  cols:=high(rData[0]);
  SetLength(result,rows,cols);
  for i:=0 to rows-1 do
  begin
    for j:=0 to cols-1 do
    begin
      result[i,j]:=rData[i+1,j+1];
    end;
  end;
end;
function ArrayBase1ToBase0(const
rData:dArr1):dArr1;overload;
var i,rows:integer;
begin
  rows:=high(rData);
  SetLength(result,rows);
  for i:=0 to rows-1 do
  begin
    result[i]:=rData[i+1];
  end;
end;
function ArrayBase1ToBase0(const
rData:dArr2):dArr2;overload;
var i,j,rows,cols:integer;
begin
  rows:=high(rData);
  cols:=high(rData[0]);
  SetLength(result,rows,cols);
  for i:=0 to rows-1 do
  begin
    for j:=0 to cols-1 do
    begin
      result[i,j]:=rData[i+1,j+1];
    end;
  end;
end;
end.
```

unit uObjFunc;

```
interface
uses uUtils,uGenerator,SysUtils,uMatrix;
type
 TObjFunc=class
 private
  FNgen,FNjam:integer;
  FBeban,FRes,FAFLC:dArr1;
  FSortAFLC:iArr1;
  FPLN:dArr2;
  FGen:TGenArr;
  function GetBeban:dArr1;
  function GetRes:dArr1;
  function GetPLN:dArr2;
  function GetGen:TGenArr;
  procedure SetGen(const rGen:TGenArr);
  procedure SetBeban(const rBeban:dArr1);
  procedure SetRes(const rRes:dArr1);
  procedure SetPLN(const rPLN:dArr2);
  function isON(const rFlip:double):boolean;
  function isServe(const rJam:integer;const
rChrom:bArr1):boolean;
  function isRampRate(const rJam:integer;const
rPL:dArr2):boolean;
  function FindAFLC(const ri:integer):integer;
  procedure RepairAFLC(var rChrom:bArr2);
  function CreateChromBase:bArr2;
  function CreateChromONOFF:bArr2;
  function HitungEcoDis(const rJam:integer;
       const rChrom1:bArr1):dArr1;
  function GetSortAFLC:iArr1;
  function GetSortChrom(const rRank:integer):bArr1;
  function HitungCostGen(const rPL:dArr2):dArr2;
  function HitungCostSUC(const rPL:dArr2):dArr2;
  function GantiChrom(const rChrom:bArr2):bArr2;
  function GetSwap(const rChrom:bArr2):bArr2;
  function doCariGreyZone(const rChrom:bArr2):bArr2;
 public
  constructor Create;overload;
  constructor Create(const rBeban,rRes:dArr1;
        const rPLN:dArr2;
        const rGen:TGenArr);overload;
  function getRandomChrom(const rFlip:double):bArr2;
  function getConstructSolution(const rFlip:double):bArr2;
  procedure setLocalSearch(var rChrom:bArr2);
  procedure doHitungChrom(const rChrom:bArr2;
       var rCostTotal:double);overload;
  procedure doHitungChrom(const rChrom:bArr2;
       var rPL:dArr2;
       var rCostPerJam:dArr1;
       var rCostTotal:double);overload;
  procedure doHitungPLN(
       var rCostPerJam:dArr1;
       var rCostTotal:double);
  procedure doHitungPL(const rPL:dArr2;
       var rCostPerJam:dArr1;
       var rCostTotal:double);
  procedure doExecute(var rChrom:bArr2;
       var rPL:dArr2;
       var rCostPerJam:dArr1;
       var rCostTotal:double);
  function HitungEcoDisGrad(const rIterasi,rJam:integer;
       const rAlpha:double;
       const rChrom1:bArr1):dArr1;
  destructor Destroy;override;
  property Ngen:integer read FNgen write FNgen;
  property Njam:integer read FNjam write FNjam;
  property Gen:TGenArr read GetGen write SetGen;
  property Beban:dArr1 read GetBeban write SetBeban;
  property PLN:dArr2 read GetPLN write SetPLN;
  property Res:dArr1 read GetRes write SetRes;
  property SortAFLC:iArr1 read GetSortAFLC;
 end;
var gObjFunc:TObjFunc;
implementation
//constructor
constructor TObjFunc.Create;
begin
 inherited Create;
 FNgen:=0;
 FNjam:=0;
end;
constructor TObjFunc.Create(const rBeban,rRes:dArr1;
       const rPLN:dArr2;
       const rGen:TGenArr);
var i,j,Neek:integer;
begin
 inherited Create;
 FNgen:=high(rGen);
 FNjam:=high(rBeban);
 Neek:=high(rRes);
 if FNjam<>Neek then raise Exception.Create('Dimensi
matrik tidak sama!');
 SetLength(FGen,FNgen+1);
 SetLength(FBeban,FNjam+1);
 SetLength(FRes,FNjam+1);
 SetLength(FAFLC,FNgen+1);
 for i:=1 to FNgen do
 begin
  FGen[i]:=TPembangkit.Create(rGen[i]);
  FAFLC[i]:=FGen[i].AFLC;
 end;
 for i:=1 to FNjam do
 begin
  FBeban[i]:=rBeban[i];
  FRes[i]:=rRes[i];
 end;
 SetLength(FPLN,FNgen+1,FNjam+1);
 for i:=1 to FNgen do
 begin
  for j:=1 to FNjam do
  begin
   FPLN[i,j]:=rPLN[i,j];
  end;
 end;
 FSortAFLC:=GetSortAFLC;
end;
//data accessing
function TObjFunc.GetBeban:dArr1;
var i:integer;
begin
 SetLength(result,FNjam+1);
 for i:=1 to FNjam do
 begin
  result[i]:=FBeban[i];
 end;
end;
```

```
function TObjFunc.GetRes:dArr1;
var i:integer;
begin
  SetLength(result,FNjam+1);
  for i:=1 to FNjam do
  begin
    result[i]:=FRes[i];
  end;
end;
function TObjFunc.GetPLN:dArr2;
var i,j:integer;
begin
  SetLength(result,FNgen+1,FNjam+1);
  for i:=1 to FNgen do
  begin
    for j:=1 to FNjam do
    begin
      result[i,j]:=FPLN[i,j];
    end;
  end;
end;
function TObjFunc.GetGen:TGenArr;
var i:integer;
begin
  SetLength(result,FNgen+1);
  for i:=1 to FNgen do
  begin
    result[i]:=TPembangkit.Create(FGen[i]);
  end;
end;
procedure TObjFunc.SetGen(const rGen:TGenArr);
var i:integer;
begin
  FNgen:=high(rGen);
  SetLength(FGen,FNgen+1);
  SetLength(FAFLC,FNgen+1);
  for i:=1 to FNgen do
  begin
    FGen[i]:=TPembangkit.Create(rGen[i]);
    FAFLC[i]:=FGen[i].AFLC;
  end;
  FSortAFLC:=GetSortAFLC;
end;
procedure TObjFunc.SetBeban(const rBeban:dArr1);
var i,Ncek:integer;
begin
  if FNjam>0 then
  begin
    Ncek:=high(rBeban);
    if FNjam<>Ncek then raise Exceptio.Create('Dimensi
matrik tidak sama!');
  end
  else
  begin
    FNjam:=high(rBeban);
  end;
  SetLength(FBeban,FNjam+1);
  for i:=1 to FNjam do
  begin
    FBeban[i]:=rBeban[i];
  end;
end;
procedure TObjFunc.SetRes(const rRes:dArr1);
var i,Ncek:integer;
begin
  if FNjam>0 then
  begin
    Ncek:=high(rRes);
    if FNjam<>Ncek then raise Exception.Create('Dimensi
matrik tidak sama!');
  end
  else
  begin
    FNjam:=high(rRes);
  end;
  SetLength(FRes,FNjam+1);
  for i:=1 to FNjam do
  begin
    FRes[i]:=rRes[i];
  end;
end;
procedure TObjFunc.SetPLN(const rPLN:dArr2);
var i,j,Ncek:integer;
begin
  if FNgen>0 then
  begin
    Ncek:=high(rPLN);
    if FNgen<>Ncek then raise Exception.Create('Dimensi
matrik tidak sama!');
  end
  else
  begin
    FNgen:=high(rPLN);
  end;
  if FNjam>0 then
  begin
    Ncek:=high(rPLN[0]);
    if FNjam<>Ncek then raise Exception.Create('Dimensi
matrik tidak sama!');
  end
  else
  begin
    FNjam:=high(rPLN[0]);
  end;
  SetLength(FPLN,FNgen+1,FNjam+1);
  for i:=1 to FNgen do
  begin
    for j:=1 to FNjam do
    begin
      FPLN[i,j]:=rPLN[i,j];
    end;
  end;
end;
//data processing
function TObjFunc.isON(const rFlip:double):boolean;
begin
  result:=false;
  if random<=rFlip then result:=true;
end;
function TObjFunc.isServe(const rJam:integer;const
rChrom:bArr1):boolean;
var i:integer;
    load,sBebanMin,sBebanMax:double;
begin
  result:=true;
  sBebanMin:=0;
  sBebanMax:=0;
  for i:=1 to FNgen do
```

```
  begin
   if rChrom[i]=true then
   begin
    sBebanMin:=sBebanMin+FGen[i].Pmin;
    sBebanMax:=sBebanMax+FGen[i].Pmax;
   end;
  end;
  load:=FBeban[rJam]+FRes[rJam];
  if load<sBebanMin then result:=false;
  if load>sBebanMax then result:=false;
end;
function TObjFunc.isRampRate(const rJam:integer;const
rPL:dArr2):boolean;
var i:integer;
   delta:double;
begin
 result:=true;
 for i:=1 to FNgen do
 begin
  if rJam>1 then
  begin
   delta:=rPL[i,rJam]-rPL[i,rJam-1];
   if delta>0 then
   begin
    if delta>FGen[i].Ramp then
    begin
     result:=false;
     break;
    end;
   end;
  end;
 end;
end;
function TObjFunc.HitungEcoDis(const rJam:integer;
      const rChrom1:bArr1):dArr1;
var i,j:integer;
   Status:bArr1;
   LoadCek,Pa,Pb,Lmd,LoadSplit:double;
   aBeban,diffa2,diffa1,Cek,tes:double;
begin
 SetLength(Status,FNgen+1);
 for i:=1 to FNgen do
 begin
  Status[i]:=rChrom1[i];
  FGen[i].Daya:=0;
 end;
 aBeban:=FBeban[rJam];
 LoadCek:=aBeban;
 LoadSplit:=aBeban;
 for i:=1 to 15 do
 begin
  Pa:=0;
  Pb:=0;
  for j:=1 to FNgen do
  begin
   if Status[j] then
   begin
    diffa2:=FGen[j].a2*2;
    diffa1:=FGen[j].a1;
    Pa:=Pa+1/diffa2;
    Pb:=Pb+diffa1/diffa2;
   end;
  end;
  if Pa<>0 then
  begin
   Lmd:=(LoadSplit+Pb)/Pa;
  end
  else
  begin
   Lmd:=LoadSplit+Pb;
  end;
  Cek:=0;
  for j:=1 to FNgen do
  begin
   if Status[j] then
   begin
    diffa2:=2*FGen[j].a2;
    diffa1:=FGen[j].a1;
    FGen[j].Daya:=(Lmd-diffa1)/diffa2;
    if FGen[j].Daya<FGen[j].Pmin then
    begin
     FGen[j].Daya:=FGen[j].Pmin;
    end;
    if FGen[j].Daya>FGen[j].Pmax then
    begin
     FGen[j].Daya:=FGen[j].Pmax;
    end;
   end;
   Cek:=Cek+FGen[j].Daya;
  end;
  tes:=LoadCek-Cek;
  if (tes<0.0001) and (tes>-0.0001) then
  begin
   break;
  end
  else if tes>0 then
  begin
   for j:=1 to fNgen do
   begin
    if Status[j] then
    begin
     if FGen[j].Daya=FGen[j].PMax then
     begin
      Status[j]:=false;
      LoadSplit:=LoadSplit-FGen[j].Daya;
      if LoadSplit<0 then
      begin
       LoadSplit:=LoadSplit+fGen[j].Daya;
       Status[j]:=true;
      end;
     end;
    end;
   end;
  end
  else if tes<0 then
  begin
   for j:=1 to fNgen do
   begin
    if Status[j] then
    begin
     if FGen[j].Daya=FGen[j].Pmin then
     begin
      Status[j]:=false;
      LoadSplit:=LoadSplit-FGen[j].Daya;
      if LoadSplit<0 then
      begin
       LoadSplit:=LoadSplit+FGen[j].Daya;
       Status[j]:=true;
```

```
        end;
       end;
      end;
     end;
    end;
   end;
   SetLength(result,FNgen+1);
   for i:=1 to FNgen do
   begin
    result[i]:=0;
    if rChrom1[i] then
    begin
     result[i]:=FGen[i].Daya;
    end;
   end;
end;
function TObjFunc.HitungEcoDisGrad(const
rIterasi,rJam:integer;
      const rAlpha:double;
      const rChrom1:bArr1):dArr1;
function CariChrom1(const rJam:integer):bArr1;
var i:integer;
begin
  SetLength(result,FNgen+1);
  for i:=1 to FNgen do
  begin
   if FPLN[i,rJam]<>0 then
   begin
    result[i]:=true;
   end
   else
   begin
    result[i]:=false;
   end;
  end;
end;
function InitDaya(const rJam:integer;
      const rChrom1:bArr1;
      var rNgenON:integer):dArr1;
var i,sa:integer;
   sBebanMin,dBeban,tmp:double;
   Pgen:dArr1;
begin
  SetLength(Pgen,FNgen+1);
  sBebanMin:=0;
  rNgenON:=0;
  for i:=1 to FNgen do
  begin
   sa:=FSortAFLC[i];
   Pgen[sa]:=0;
   if rChrom1[sa]=true then
   begin
    inc(rNgenON);
    Pgen[sa]:=FGen[sa].Pmin;
    sBebanMin:=sBebanMin+Pgen[sa];
   end;
  end;
  dBeban:=FBeban[rJam]-sBebanMin;
  for i:=1 to FNgen do
  begin
   sa:=FSortAFLC[i];
   if rChrom1[sa]=true then
   begin
    if dBeban>0 then
    begin
     tmp:=Pgen[sa];
     Pgen[sa]:=Pgen[sa]+dBeban;
     if Pgen[sa]>FGen[sa].Pmax then
     begin
      Pgen[sa]:=FGen[sa].Pmax;
     end;
     dBeban:=dBeban-(Pgen[sa]-tmp);
    end;
   end;
  end;
  SetLength(result,rNgenON+1);
  sa:=0;
  for i:=1 to FNgen do
  begin
   if rChrom1[i]=true then
   begin
    inc(sa);
    result[sa]:=Pgen[i];
   end;
  end;
end;
function CariAveLamda(const rPgen:dArr1;
      const rChrom1:bArr1):double;
var i,sa:integer;
   sLamda:double;
begin
  sLamda:=0;
  sa:=0;
  for i:=1 to FNgen do
  begin
   if rChrom1[i]=true then
   begin
    inc(sa);
    sLamda:=sLamda+FGen[i].GetLamda(rPgen[sa]);
   end;
  end;
  result:=sLamda/sa;
end;
function CariPgen(const rChrom:bArr1;
      const rLamda:double):dArr1;
var i:integer;
begin
  SetLength(result,FNgen+2);
  for i:=1 to FNgen do
  begin
   if rChrom[i]=true then
   begin
    result[i]:=FGen[i].GetDaya(rLamda);
   end;
  end;
end;
function CaridL(const rJam:integer;
      const rChrom1:bArr1;
      const rPgen:dArr1;
      const rLamda:double):dArr1;
var i,sa:integer;
   minLamda,maxLamda,sumPgen:double;
begin
  SetLength(result,high(rPgen)+2);
  sumPgen:=0;
  sa:=0;
  for i:=1 to FNgen do
  begin
```

```
    if rChrom1[i]=true then
    begin
     inc(sa);
     minLamda:=FGen[i].GetLamda(FGen[i].Pmin);
     maxLamda:=FGen[i].GetLamda(FGen[i].Pmax);
     sumPgen:=sumPgen+rPgen[sa];
     if (rLamda<minLamda) or (rLamda>maxLamda) then
     //if (maxLamda<rLamda) then
     begin
      result[sa]:=0;
     end
     else
     begin
      result[sa]:=FGen[i].GetLamda(rPgen[sa])-rLamda;
     end;
    end;
  end;
  result[sa+1]:=FBeban[rJam]-sumPgen;
end;
function CariMaxdL(const rdL:dArr1):double;
var i:integer;
begin
  result:=rdL[1];
  for i:=2 to high(rdL) do
  begin
   if result<abs(rdL[i]) then
   begin
    result:=abs(rdL[i]);
   end;
  end;
end;
function MatrixHessian(const rChrom1:bArr1):dArr2;
var i,j,sa,sb,NgenON:integer;
begin
  NgenON:=0;
  for i:=1 to FNgen do
  begin
   if rChrom1[i]=true then
   begin
    inc(NgenON);
   end;
  end;
  SetLength(result,NgenON+2,NgenON+2);
  sa:=0;
  for i:=1 to FNgen do
  begin
   if rChrom1[i]=true then
   begin
    inc(sa);
    sb:=0;
    for j:=1 to FNgen do
    begin
     if rChrom1[j]=true then
     begin
      inc(sb);
      if sb=sa then
      begin
       result[sa,sb]:=2*FGen[i].a2;
      end
      else
      begin
       result[sa,sb]:=0;
      end;
     end;
    end;
   end;
  end;
  sa:=0;
  for i:=1 to FNgen do
  begin
   if rChrom1[i]=true then
   begin
    inc(sa);
    result[NgenON+1,sa]:=-1;
    result[sa,NgenON+1]:=-1;
   end;
  end;
  result[NgenON+1,NgenON+1]:=0;
end;
function BalikKeSemula(const rPgen:dArr1;
      const rChrom1:bArr1):dArr1;
var i,sa:integer;
begin
  SetLength(result,FNgen+1);
  sa:=0;
  for i:=1 to FNgen do
  begin
   if rChrom1[i]=true then
   begin
    inc(sa);
    result[i]:=rPgen[sa];
   end;
  end;
end;
procedure UpdateLamda(const rdL:dArr1;
      const rChrom1:bArr1;
      const rHessian:dArr2;
      var rPgen:dArr1;
      var rLamda:double);
var i,sa:integer;
   invHessian:dArr2;
   dX:dArr1;
begin
  invHessian:=MatrixInvers(rHessian);
  invHessian:=MatrixNegative(invHessian);
  dX:=MatrixMul(invHessian,rdL);
  sa:=0;
  for i:=1 to FNgen do
  begin
   if rChrom1[i]=true then
   begin
    inc(sa);
    rPgen[sa]:=rPgen[sa]+dX[sa];
    if rPgen[sa]>FGen[i].Pmax then
rPgen[sa]:=FGen[i].Pmax;
    if rPgen[sa]<FGen[i].Pmin then
rPgen[sa]:=FGen[i].Pmin;
   end;
  end;
  rLamda:=rLamda+dX[FNgen+1];
end;
var i,NgenON:integer;
   status:bArr1;
   Hessian:dArr2;
   Pgen,dL:dArr1;
   sPmax,sPmin,Lamda,Alpha,error,max:double;
begin
  SetLength(status,FNgen+1);
```

```
  for i:=1 to FNgen do
  begin
   status[i]:=rChrom1[i];
  end;
  sPmax:=0;
  sPmin:=0;
  for i:=1 to FNgen do
  begin
   if status[i]=true then
   begin
    sPmin:=sPmin+FGen[i].Pmin;
    sPmax:=sPmax+FGen[i].Pmax;
   end;
  end;
  SetLength(result,FNgen+1);
  if sPmin>FBeban[rJam] then
  begin
   for i:=1 to FNgen do
   begin
    if status[i]=true then
    begin
     result[i]:=FGen[i].Pmin;
    end;
   end;
  end
  else if sPmax<FBeban[rJam] then
  begin
   for i:=1 to FNgen do
   begin
    if status[i]=true then
    begin
     result[i]:=FGen[i].Pmax;
    end;
   end;
  end
  else
  begin
   error:=rAlpha;
   Pgen:=InitDaya(rJam,rChrom1,NgenON);
   Lamda:=CariAveLamda(Pgen,rChrom1);
   for i:=1 to rIterasi do
   begin
    dL:=CaridL(rJam,rChrom1,Pgen,Lamda);
    max:=cariMaxdL(dL);
    if max<error then break;
    Hessian:=MatrixHessian(rChrom1);
    UpdateLamda(dL,rChrom1,Hessian,Pgen,Lamda);
   end;
   result:=BalikKeSemula(Pgen,rChrom1);
  end;
end;
function TObjFunc.HitungCostGen(const rPL:dArr2):dArr2;
var i,j:integer;
begin
  SetLength(result,FNgen+1,FNjam+1);
  for i:=1 to FNgen do
  begin
   for j:=1 to FNjam do
   begin
    result[i,j]:=FGen[i].GetBiaya(rPL[i,j]);
   end;
  end;
end;
function TObjFunc.HitungCostSUC(const rPL:dArr2):dArr2;
var i,j,init,tcold:integer;
begin
  SetLength(result,FNgen+1,FNjam+1);
  for i:=1 to FNgen do
  begin
   init:=FGen[i].InitSt;
   tcold:=FGen[i].Tdown+FGen[i].Tcold;
   for j:=1 to FNjam do
   begin
    result[i,j]:=0;
    if rPL[i,j]<>0 then
    begin
     if init>0 then
     begin
      init:=init+1;
     end
     else if init<0 then
     begin
      if abs(init)<=tcold then
      begin
       result[i,j]:=FGen[i].Sh;
      end
      else
      begin
       result[i,j]:=FGen[i].Sc;
      end;
      init:=1;
     end;
    end
    else if rPL[i,j]=0 then
    begin
     if init>0 then
     begin
      init:=-1;
     end
     else if init<0 then
     begin
      init:=init-1;
     end;
    end;
   end;
  end;
end;
procedure TObjFunc.doHitungChrom(const rChrom:bArr2;
        var rCostTotal:double);
var i,j:integer;
   PLa:dArr1;
   PL,CostGen,CostSUC:dArr2;
   chrom1:bArr1;
begin
  SetLength(PL,FNgen+1,FNjam+1);
  SetLength(chrom1,FNgen+1);
  SetLength(PLa,FNgen+1);
  for i:=1 to FNjam do
  begin
   for j:=1 to FNgen do
   begin
    chrom1[j]:=rChrom[j,i];
   end;
   PLa:=HitungEcoDis(i,chrom1);
   for j:=1 to FNgen do
   begin
    PL[j,i]:=PLa[j];
   end;
```

```
    end;
    CostGen:=HitungCostGen(PL);
    CostSUC:=HitungCostSUC(PL);
    rCostTotal:=0;
    for i:=1 to FNjam do
    begin
     for j:=1 to FNgen do
     begin
      rCostTotal:=rCostTotal+CostGen[j,i]+CostSUC[j,i];
     end;
    end;
  end;
  procedure TObjFunc.doHitungChrom(const rChrom:bArr2;
          var rPL:dArr2;
          var rCostPerJam:dArr1;
          var rCostTotal:double);
  var i,j:integer;
     PLa:dArr1;
     CostGen,CostSUC:dArr2;
     chrom1:bArr1;
  begin
    SetLength(rPL,FNgen+1,FNjam+1);
    SetLength(rCostPerJam,FNjam+1);
    SetLength(chrom1,FNgen+1);
    SetLength(PLa,FNgen+1);
    for i:=1 to FNjam do
    begin
     for j:=1 to FNgen do
     begin
      chrom1[j]:=rChrom[j,i];
     end;
     PLa:=HitungEcoDis(i,chrom1);
     for j:=1 to FNgen do
     begin
      rPL[j,i]:=PLa[j];
     end;
    end;
    CostGen:=HitungCostGen(rPL);
    CostSUC:=HitungCostSUC(rPL);
    SetLength(rCostPerJam,FNjam+1);
    rCostTotal:=0;
    for i:=1 to FNjam do
    begin
     rCostPerJam[i]:=0;
     for j:=1 to FNgen do
     begin

  rCostPerJam[i]:=rCostPerJam[i]+CostGen[j,i]+CostSUC[j,i];
     end;
     rCostTotal:=rCostTotal+rCostPerJam[i];
    end;
  end;
  procedure TObjFunc.doHitungPL(const rPL:dArr2;
          var rCostPerJam:dArr1;
          var rCostTotal:double);
  var i,j:integer;
     CostGen,CostSUC:dArr2;
  begin
    SetLength(rCostPerJam,FNjam+1);
    CostGen:=HitungCostGen(rPL);
    CostSUC:=HitungCostSUC(rPL);
    rCostTotal:=0;
    for i:=1 to FNjam do
    begin
     rCostPerJam[i]:=0;
     for j:=1 to FNgen do
     begin
  rCostPerJam[i]:=rCostPerJam[i]+CostGen[j,i]+CostSUC[j,i];
     end;
     rCostTotal:=rCostTotal+rCostPerJam[i];
    end;
  end;
  procedure TObjFunc.doHitungPLN(var rCostPerJam:dArr1;
          var rCostTotal:double);
  var i,j:integer;
     CostGen,CostSUC:dArr2;
  begin
    SetLength(rCostPerJam,FNjam+1);
    CostGen:=HitungCostGen(FPLN);
    CostSUC:=HitungCostSUC(FPLN);
    rCostTotal:=0;
    for i:=1 to FNjam do
    begin
     rCostPerJam[i]:=0;
     for j:=1 to FNgen do
     begin
  rCostPerJam[i]:=rCostPerJam[i]+CostGen[j,i]+CostSUC[j,i];
     end;
     rCostTotal:=rCostTotal+rCostPerJam[i];
    end;
  end;
  //data output
  function TObjFunc.GetSortAFLC:iArr1;
  var i,j,tmp:integer;
     tmpAFLC:double;
  begin
    SetLength(result,FNgen+1);
    for i:=1 to FNgen do
    begin
     result[i]:=i;
    end;
    for i:=1 to fNgen-1 do
    begin
     for j:=i to fNgen do
     begin
      if FAFLC[i]>FAFLC[j] then
      begin
       tmpAFLC:=FAFLC[i];
       FAFLC[i]:=FAFLC[j];
       FAFLC[j]:=tmpAFLC;
       tmp:=result[i];
       result[i]:=result[j];
       result[j]:=tmp;
      end;
     end;
    end;
    for i:=1 to fNgen do
    begin
     fAFLC[i]:=fGen[i].AFLC;
    end;
  end;
  function TObjFunc.GetSortChrom(const
  rRank:integer):bArr1;
  var i:integer;
  begin
    SetLength(result,FNgen+1);
    for i:=1 to FNgen do
    begin
```

```
    result[i]:=false;
   end;
   for i:=1 to rRank do
   begin
    result[FSortAFLC[i]]:=true;
   end;
 end;
 function TObjFunc.FindAFLC(const ri:integer):integer;
 var i:integer;
 begin
  result:=1;
  for i:=1 to FNgen do
  begin
   if FSortAFLC[i]=ri then
   begin
    result:=i;
    break;
   end;
  end;
 end;
 procedure TObjFunc.RepairAFLC(var rChrom:bArr2);
 var i,j,pos,k:integer;
 begin
  for i:=1 to FNjam do
  begin
   for j:=1 to Ngen do
   begin
    if rChrom[j,i]=true then
    begin
     pos:=FindAFLC(j);
     for k:=1 to pos do
     begin
      if rChrom[FSortAFLC[k],i]=false then
      begin
       rChrom[FSortAFLC[k],i]:=true;
       rChrom[j,i]:=false;
       break;
      end;
     end;
    end;
   end;
  end;
 end;
 function TObjFunc.CreateChromBase:bArr2;
 var i,j,k:integer;
   chrom1:bArr1;
   serve:boolean;
 begin
  SetLength(result,FNgen+1,FNjam+1);
  SetLength(chrom1,FNgen+1);
  for i:=1 to FNjam do
  begin
   for j:=1 to FNgen do
   begin
    chrom1:=GetSortChrom(j);
    serve:=isServe(i,chrom1);
    if serve=true then
    begin
     for k:=1 to FNgen do
     begin
      result[k,i]:=chrom1[k];
     end;
     break;
    end;
   end;
  end;
 end;
 function TObjFunc.CreateChromONOFF:bArr2;
 var i,j,k:integer;
   chrom1:bArr1;
   serve:boolean;
 begin
  SetLength(result,FNgen+1,FNjam+1);
  SetLength(chrom1,FNgen+1);
  for i:=1 to FNjam do
  begin
   for j:=FNgen downto 1 do
   begin
    chrom1:=GetSortChrom(j);
    serve:=isServe(i,chrom1);
    if serve=true then
    begin
     for k:=1 to FNgen do
     begin
      result[k,i]:=chrom1[k];
     end;
     break;
    end;
   end;
  end;
 end;
 function TObjFunc.getRandomChrom(const
 rFlip:double):bArr2;
 var i,j:integer;
   chromBase,chromONOFF:bArr2;
 begin
  chromBase:=CreateChromBase;
  chromONOFF:=CreateChromONOFF;
  SetLength(result,Ngen+1,Njam+1);
  for i:=1 to Ngen do
  begin
   for j:=1 to Njam do
   begin
    if chromBase[i,j]=true then
    begin
     result[i,j]:=true;
    end
    else
    begin
     if chromONOFF[i,j]=true then
     begin
      if isON(rFlip)=true then
      begin
       result[i,j]:=true;
      end
      else
      begin
       result[i,j]:=false;
      end;
     end;
    end;
   end;
  end;
  RepairAFLC(result);
  result:=GetSwap(result);
 end;
 function TObjFunc.getConstructSolution(const
 rFlip:double):bArr2;
```

```
  var i,j:integer;
    chromBase,chromONOFF:bArr2;
  begin
   chromBase:=CreateChromBase;
   chromONOFF:=CreateChromONOFF;
   SetLength(result,Ngen+1,Njam+1);
   for i:=1 to Ngen do
   begin
    for j:=1 to Njam do
    begin
     if chromBase[i,j]=true then
     begin
      result[i,j]:=true;
     end
     else
     begin
      if chromONOFF[i,j]=true then
      begin
       if isON(rFlip)=true then
       begin
        result[i,j]:=true;
       end
       else
       begin
        result[i,j]:=false;
       end;
      end;
     end;
    end;
   end;
   RepairAFLC(result);
   result:=GetSwap(result);
  end;
  procedure TObjFunc.setLocalSearch(var rChrom:bArr2);
  var i,j:integer;
    CostBest,CostCek:double;
    greyChrom:bArr2;
  begin
   doHitungChrom(rChrom,CostBest);
   greyChrom:=doCariGreyZone(rChrom);
   for i:=1 to FNgen do
   begin
    for j:=1 to FNjam do
    begin
     if greyChrom[i,j]=true then
     begin
      rChrom[i,j-1]:=true;
      doHitungChrom(rChrom,CostCek);
      if CostBest>CostCek then
      begin
       CostBest:=CostCek;
      end
      else
      begin
       rChrom[i,j-1]:=false;
      end;
     end;
    end;
   end;
  end;
  function TObjFunc.GetSwap(const rChrom:bArr2):bArr2;
  var i,j,k,init,pos:integer;
  begin
   SetLength(result,FNgen+1,FNjam+1);
   for i:=1 to FNgen do
   begin
    init:=FGen[i].InitSt;
    for j:=1 to FNjam do
    begin
     result[i,j]:=rChrom[i,j];
     if result[i,j]=true then
     begin
      if init<0 then
      begin
       if abs(init)>=FGen[i].Tdown then
       begin
        init:=1;
       end
       else
       begin
        pos:=j+init;
        if pos<1 then
        begin
         pos:=1;
        end;
        for k:=pos to j-1 do
        begin
         result[i,k]:=true;
        end;
       end;
      end
      else if init>0 then
      begin
       init:=init+1;
      end;
     end
     else if result[i,j]=false then
     begin
      if init<0 then
      begin
       init:=init-1;
      end
      else if init>0 then
      begin
       if init>=FGen[i].Tup then
       begin
        init:=-1;
       end
       else
       begin
        result[i,j]:=true;
        init:=init+1;
       end;
      end;
     end;
    end;
   end;
  end;
  //destructor
  destructor TObjFunc.Destroy;
  var i:integer;
  begin
   try
    for i:=1 to FNgen do
    begin
     FGen[i].Free;
    end;
   finally
```

```
    inherited Destroy;
  end;
end;
function TObjFunc.GantiChrom(const rChrom:bArr2):bArr2;
var i,j:integer;
begin
  SetLength(result,FNgen+1,FNjam+1);
  for i:=1 to FNgen do
  begin
    for j:=1 to FNjam do
    begin
      result[i,j]:=rChrom[i,j];
    end;
  end;
end;
function TObjFunc.doCariGreyZone(const
rChrom:bArr2):bArr2;
var i,j,init,tcold:integer;
begin
  SetLength(result,FNgen+1,FNjam+1);
  for i:=1 to FNgen do
  begin
    for j:=1 to FNjam do
    begin
      result[i,j]:=false;
    end;
  end;
  for i:=1 to FNgen do
  begin
    init:=FGen[i].InitSt;
    tcold:=FGen[i].Tdown+FGen[i].Tcold;
    for j:=1 to FNjam do
    begin
      if rChrom[i,j]=true then
      begin
        if init<0 then
        begin
          if abs(init)=(tcold+1) then
          begin
            result[i,j]:=true;
          end;
          init:=1;
        end
        else if init>0 then
        begin
          init:=init+1;
        end;
      end
      else if rChrom[i,j]=false then
      begin
        if init<0 then
        begin
          init:=init-1;
        end
        else if init>0 then
        begin
          init:=-1;
        end;
      end;
    end;
  end;
end;
procedure TObjFunc.doExecute(var rChrom:bArr2;
      var rPL:dArr2;
      var rCostPerJam:dArr1;
      var rCostTotal:double);
var i,j:integer;
    CostBest,CostCek:double;
    chromBase,greyChrom:bArr2;
begin
  chromBase:=CreateChromBase;
  rChrom:=GetSwap(chromBase);
  doHitungChrom(rChrom,CostBest);
  greyChrom:=doCariGreyZone(rChrom);
  for i:=1 to FNgen do
  begin
    for j:=1 to FNjam do
    begin
      if greyChrom[i,j]=true then
      begin
        rChrom[i,j-1]:=true;
        doHitungChrom(rChrom,CostCek);
        if CostBest>CostCek then
        begin
          CostBest:=CostCek;
        end
        else
        begin
          rChrom[i,j-1]:=false;
        end;
      end;
    end;
  end;
  doHitungChrom(rChrom,rPL,rCostPerJam,rCostTotal);
end;
end.
```

program EcoRGAEv;

```
uses
  Forms,
  uAbout in 'uAbout.pas' {frmAbout},
  uComplex in 'uComplex.pas',
  uMatrix in 'uMatrix.pas',
  uMenu in 'uMenu.pas' {frmMenu},
  uUtils in 'uUtils.pas',
  uGenerator in 'GenThermal\uGenerator.pas',
  uHasil in 'GenThermal\uHasil.pas' {frmHasil},
  uInputGen in 'GenThermal\uInputGen.pas' {frmInput},
  uObjFunc in 'GenThermal\uObjFunc.pas',
  uFitness2 in 'Khusus\uFitness2.pas',
  uGABin in 'Khusus\uGABin.pas',
  uGenetic in 'Khusus\uGenetic.pas',
  uGenVar in 'Khusus\uGenVar.pas',
  uRandom in 'Khusus\uRandom.pas';
{$R *.res}
begin
  Application.Initialize;
  Application.CreateForm(TfrmMenu, frmMenu);
  Application.CreateForm(TfrmHasil, frmHasil);
  Application.CreateForm(TfrmInput, frmInput);
  Application.CreateForm(TfrmAbout, frmAbout);
  Application.Run;
end.
```

unit uFitness2;

```
interface
uses uUtils,uGenerator,uRandom;
type
 TFitness2=class
 private
  FNgen:integer;
  FLoad,FPinGen:double;
  FGen:TGenArr;
  function getGen:TGenArr;
  procedure setGen(const rGen:TGenArr);
  function getBatasGen:TBatasArr1;
 public
  constructor Create(const rPinGen:double;
          var rGen:TGenArr;
          const rLoad:double);
  destructor Destroy;override;
  function getRandomGen:dArr1;
  function doCalcCost(var rChrom:dArr1):double;
  function doRecombination(var
rChrom1,rChrom2:dArr1):dArr1;
  function doMutasi(const rNo:integer;
       const rPgen,rCk,rCmin,rBetha:double;
       var rRand:TRandomu):double;
  property Ngen:integer read FNgen write FNgen;
  property Load:double read FLoad write FLoad;
  property Gen:TGenArr read getGen write SetGen;
  property PinGen:double read FPinGen write FPinGen;
  property BatasGen:TBatasArr1 read getBatasGen;
 end;
var gFitness2:TFitness2;
implementation
//constructor
constructor TFitness2.Create(const rPinGen:double;
       var rGen:TGenArr;
       const rLoad:double);
var i:integer;
begin
 inherited Create;
 FNgen:=high(rGen)+1;
 SetLength(FGen,FNgen);
 for i:=0 to FNgen-1 do
 begin
  FGen[i]:=TPembangkit.Create(rGen[i]);
 end;
 FLoad:=rLoad;
 FPinGen:=rPinGen;
end;
//destructor
destructor TFitness2.Destroy;
var i:integer;
begin
 try
  for i:=0 to FNgen-1 do
  begin
   FGen[i].Free;
  end;
 finally
  inherited Destroy;
 end;
end;
//data accessing
function TFitness2.getGen:TGenArr;
var i:integer;
begin
 SetLength(result,FNgen);
 for i:=0 to FNgen-1 do
 begin
  result[i]:=TPembangkit.Create(FGen[i]);
 end;
end;
procedure TFitness2.setGen(const rGen:TGenArr);
var i:integer;
begin
 FNgen:=high(rGen);
 SetLength(FGen,FNgen);
 for i:=0 to FNgen-1 do
 begin
  FGen[i]:=TPembangkit.Create(rGen[i]);
 end;
end;
//data proccessing
function TFitness2.getBatasGen:TBatasArr1;
var i:integer;
begin
 SetLength(result,FNgen);
 for i:=0 to FNgen-1 do
 begin
  result[i].min:=FGen[i].Pmin;
  result[i].max:=FGen[i].Pmax;
 end;
end;
function TFitness2.getRandomGen:dArr1;
var i,pos:integer;
   sumGen:double;
begin
 SetLength(result,FNgen);
 for i:=0 to FNgen-1 do
 begin
  result[i]:=FGen[i].Pmin+random*(FGen[i].Pmax-
FGen[i].Pmin);
 end;
 pos:=round(1+random*(FNgen-1));
 pos:=pos-1;
 sumGen:=0;
 for i:=0 to FNgen-1 do
 begin
  if i<>pos then
  begin
   sumGen:=sumGen+result[i];
  end;
 end;
 result[pos]:=FLoad-sumGen;
 if result[pos]>FGen[pos].Pmax then
result[pos]:=FGen[pos].Pmax;
 if result[pos]<FGen[pos].Pmin then
result[pos]:=FGen[pos].Pmin;
end;
function TFitness2.doCalcCost(var rChrom:dArr1):double;
var i,pos:integer;
   sumGen,pinalty,sumNOx:double;
begin
 pos:=round(1+random*(FNgen-1));
 pos:=pos-1;
 sumGen:=0;
 for i:=0 to FNgen-1 do
 begin
  if i<>pos then
  begin
```

```
      sumGen:=sumGen+rChrom[i];
     end;
    end;
    rChrom[pos]:=FLoad-sumGen;
    pinalty:=0;
    if rChrom[pos]>FGen[pos].Pmax then
    begin
     pinalty:=rChrom[pos]-FGen[pos].Pmax;
     rChrom[pos]:=FGen[pos].Pmax;
    end;
    if rChrom[pos]<FGen[pos].Pmin then
    begin
     pinalty:=FGen[pos].Pmin-rChrom[pos];
     rChrom[pos]:=FGen[pos].Pmin;
    end;
    result:=0;
    sumNOx:=0;
    for i:=0 to FNgen-1 do
    begin
     result:=result+FGen[i].GetBiaya(rChrom[i]);
     sumNOx:=sumNOx+FGen[i].GetEmisiNOx(rChrom[i]);
    end;
    result:=result+sumNOx+FPinGen*pinalty;
  end;
  function TFitness2.doRecombination(var
  rChrom1,rChrom2:dArr1):dArr1;
  var i,pos:integer;
     sumGen:double;
  begin
   SetLength(result,FNgen);
   for i:=0 to FNgen-1 do
   begin
    result[i]:=rChrom1[i]+random*(rChrom2[i]-rChrom1[i]);
    if result[i]>FGen[i].Pmax then result[i]:=FGen[i].Pmax;
    if result[i]<FGen[i].Pmin then result[i]:=FGen[i].Pmin;
   end;
   pos:=round(1+random*(FNgen-1));
   pos:=pos-1;
   sumGen:=0;
   for i:=0 to FNgen-1 do
   begin
    if i<>pos then
    begin
     sumGen:=sumGen+result[i];
    end;
   end;
   result[pos]:=FLoad-sumGen;
   if result[pos]>FGen[pos].Pmax then
  result[pos]:=FGen[pos].Pmax;
   if result[pos]<FGen[pos].Pmin then
  result[pos]:=FGen[pos].Pmin;
  end;
  function TFitness2.doMutasi(const rNo:integer;
        const rPgen,rCk,rCmin,rBetha:double;
        var rRand:TRandomu):double;
  var tho,dPgen:double;
  begin
   tho:=rCk/rCmin*(FGen[rNo].Pmax-
  FGen[rNo].Pmin)*rBetha;
   dPgen:=rRand.NextGaussian(0,tho);
   result:=rPgen+dPgen;
   if result>FGen[rNo].Pmax then result:=FGen[rNo].Pmax;
   if result<FGen[rNo].Pmin then result:=FGen[rNo].Pmin;
  end;
```

```
  end.
```

unit uGenerator;

```
  interface
  type
   TPembangkit=class
   private
    FNama:string;
  FPmax,FPmin,Fa2,Fa1,Fa0,FSh,FSc,FDaya,FRamp,Fac2,Fae
  1,Fac0:double;
    FTup,FTdown,FTcold,FInitSt:integer;
    function GetAFLC:double;
   public
    constructor Create;overload;
    constructor Create(const rNama:string;
          const
  rPmin,rPmax,ra2,ra1,ra0,rSh,rSc,rRamp:double;
          const
  rTup,rTdown,rTcold,rInitSt:integer);overload;
    constructor Create(const rNama:string;
          const rPmin,rPmax,ra2,ra1,ra0,rSh,rSc,rRamp,
          rae2,rae1,rae0:double;
          const
  rTup,rTdown,rTcold,rInitSt:integer);overload;
    constructor Create(const
  rPembangkit:TPembangkit);overload;
    procedure Assign(const rPembangkit:TPembangkit);
    function GetBiaya(const rDaya:double):double;
    function GetEmisiNOx(const rDaya:double):double;
    function GetDaya(const rLamda:double):double;
    function GetLamda(const rDaya:double):double;
    property Nama:string read FNama write FNama;
    property Pmax:double read FPmax write FPmax;
    property Pmin:double read FPmin write FPmin;
    property a2:double read Fa2 write Fa2;
    property a1:double read Fa1 write Fa1;
    property a0:double read Fa0 write Fa0;
    property ac2:double read Fac2 write Fac2;
    property ae1:double read Fae1 write Fae1;
    property ae0:double read Fae0 write Fae0;
    property Sh:double read FSh write FSh;
    property Sc:double read FSc write FSc;
    property Ramp:double read FRamp write FRamp;
    property Tup:integer read FTup write FTup;
    property Tdown:integer read FTdown write FTdown;
    property Tcold:integer read FTcold write FTcold;
    property InitSt:integer read FInitSt write FInitSt;
    property Daya:double read FDaya write FDaya;
    property AFLC:double read GetAFLC;
   end;
   TGenArr=array of TPembangkit;
  implementation
  //constructor
  constructor TPembangkit.Create;
  begin
   inherited Create;
  end;
  constructor TPembangkit.Create(const rNama:string;
        const rPmin,rPmax,ra2,ra1,ra0,rSh,rSc,rRamp:double;
        const rTup,rTdown,rTcold,rInitSt:integer);
  begin
   inherited Create;
   FNama:=rNama;
```

```
    FPmin:=rPmin;
    FPmax:=rPmax;
    Fa2:=ra2;
    Fa1:=ra1;
    Fa0:=ra0;
    Fae2:=0;
    Fae1:=0;
    Fae0:=0;
    FSh:=rSh;
    FSc:=rSc;
    FRamp:=rRamp;
    FTup:=rTup;
    FTdown:=rTdown;
    FTcold:=rTcold;
    FInitSt:=rInitSt;
  end;
  constructor TPembangkit.Create(const rNama:string;
          const rPmin,rPmax,ra2,ra1,ra0,rSh,rSc,rRamp,
          rae2,rae1,rae0:double;
          const rTup,rTdown,rTcold,rInitSt:integer);
  begin
    inherited Create;
    FNama:=rNama;
    FPmin:=rPmin;
    FPmax:=rPmax;
    Fa2:=ra2;
    Fa1:=ra1;
    Fa0:=ra0,
    Fae2:=rae2;
    Fae1:=rae1;
    Fae0:=rae0;
    FSh:=rSh;
    FSc:=rSc;
    FRamp:=rRamp;
    FTup:=rTup;
    FTdown:=rTdown;
    FTcold:=rTcold;
    FInitSt:=rInitSt;
  end;
  constructor TPembangkit.Create(const
  rPembangkit:TPembangkit);
  begin
    inherited Create;
    FNama:=rPembangkit.Nama;
    FPmin:=rPembangkit.Pmin;
    FPmax:=rPembangkit.Pmax;
    Fa2:=rPembangkit.a2;
    Fa1:=rPembangkit.a1;
    Fa0:=rPembangkit.a0;
    Fae2:=rPembangkit.ae2;
    Fae1:=rPembangkit.ae1;
    Fae0:=rPembangkit.ae0;
    FSh:=rPembangkit.Sh;
    FSc:=rPembangkit.Sc;
    FRamp:=rPembangkit.Ramp;
    FTup:=rPembangkit.Tup;
    FTdown:=rPembangkit.Tdown;
    FTcold:=rPembangkit.Tcold;
    FInitSt:=rPembangkit.InitSt;
  end;
  function TPembangkit.GetAFLC:double;
  begin
    Result:=fa0/fPmax+fa1+fa2*fPmax;
  end;
  procedure TPembangkit.Assign(const
  rPembangkit:TPembangkit);
  begin
    FNama:=rPembangkit.Nama;
    FPmin:=rPembangkit.Pmin;
    FPmax:=rPembangkit.Pmax;
    Fa2:=rPembangkit.a2;
    Fa1:=rPembangkit.a1;
    Fa0:=rPembangkit.a0;
    FSh:=rPembangkit.Sh;
    FSc:=rPembangkit.Sc;
    FRamp:=rPembangkit.Ramp;
    FTup:=rPembangkit.Tup;
    FTdown:=rPembangkit.Tdown;
    FTcold:=rPembangkit.Tcold;
    FInitSt:=rPembangkit.InitSt;
  end;
  //data operation
  function TPembangkit.GetBiaya(const rDaya:double):double;
  begin
    result:=0;
    if rDaya<>0 then
    begin
      result:=Fa2*sqr(rDaya)+Fa1*rDaya+Fa0;
    end;
  end;
  function TPembangkit.GetDaya(const
  rLamda:double):double;
  begin
    result:=(rLamda-Fa1)/(2*Fa2);
    if result>FPmax then result:=FPmax;
    if result<FPmin then result:=FPmin;
  end;

  function TPembangkit.GetEmisiNOx(const
  rDaya:double):double;
  begin
    result:=0;
    if rDaya<>0 then
    begin
      result:=Fae2*sqr(rDaya)+Fae1*rDaya+Fae0;
    end;
  end;
  function TPembangkit.GetLamda(const
  rDaya:double):double;
  begin
    result:=2*Fa2*rDaya+Fa1;
  end;
  end.
```

unit uInputGen;

```
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, ExtCtrls, Grids, jpeg;
type
  TfrmInput = class(TForm)
    PageControl1: TPageControl;
    Panel1: TPanel;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
```

```
    btnClose: TButton;
    btnNext: TButton;
    SaveDialog1: TSaveDialog;
    Label1: TLabel;
    Label2: TLabel;
    edtNGen: TEdit;
    edtNjam: TEdit;
    fgGen. TStringGrid;
    fgLoad: TStringGrid;
    TabSheet4: TTabSheet;
    fgPLN: TStringGrid;
    Image1: TImage;
    Image2: TImage;
    procedure btnCloseClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure edtNGenChange(Sender: TObject);
    procedure edtNjamChange(Sender: TObject);
    procedure btnNextClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmInput: TfrmInput;
implementation
uses uObjFunc, uHasil;
{$R *.dfm}
procedure TfrmInput.btnCloseClick(Sender: TObject);
begin
  Close;
end;
procedure TfrmInput.FormCreate(Sender: TObject);
begin
  fgGen.Cells[0,0] :='Gen';
  fgGen.Cells[1,0] :='Nama';
  fgGen.Cells[2,0] :='Pmax';
  fgGen.Cells[3,0] :='Pmin';
  fgGen.Cells[4,0] :='a0';
  fgGen.Cells[5,0] :='a1';
  fgGen.Cells[6,0] :='a2';
  fgGen.Cells[7,0] :='Tup';
  fgGen.Cells[8,0] :='Tdown';
  fgGen.Cells[9,0] :='Sh';
  fgGen.Cells[10,0]:='Sc';
  fgGen.Cells[11,0]:='Tcold';
  fgGen.Cells[12,0]:='InitSt';
  fgGen.Cells[13,0]:='Ramp Rate';
  fgGen.Cells[14,0]:='ae2';
  fgGen.Cells[15,0]:='ae1';
  fgGen.Cells[16,0]:='ae0';
  fgLoad.Cells[0,0]:='';
  fgLoad.Cells[1,0]:='Load';
  fgLoad.Cells[2,0]:='Res';
end;
procedure TfrmInput.edtNGenChange(Sender: TObject);
var i:integer;
begin
  if edtNgen.Text='' then
  begin
    fgGen.RowCount:=2;
    fgPLN.RowCount:=2;
  end
  else
  begin
    fgGen.RowCount:=StrToInt(edtNgen.Text)+1;
    fgPLN.RowCount:=StrToInt(edtNgen.Text)+1;
    for i:=1 to StrToInt(edtNger.Text) do
    begin
      fgGen.Cells[0,i]:=IntToStr(i);
      fgPLN.Cells[0,i]:='Gen '+IntToStr(i);
    end;
  end;
end;
procedure TfrmInput.edtNjamChange(Sender: TObject);
var i:integer;
begin
  if edtNjam.Text='' then
  begin
    fgLoad.RowCount:=2;
    fgPLN.ColCount:=2;
  end
  else
  begin
    fgLoad.RowCount:=StrToInt(edtNjam.Text)+1;
    fgPLN.ColCount:=StrToInt(edtNjam.Text)+1;
    for i:=1 to StrToInt(edtNjam.Text) do
    begin
      fgLoad.Cells[0,i]:=IntToStr(i);
      fgPLN.Cells[i,0]:='Jam '+IntToStr(i);
    end;
  end;
end;
procedure TfrmInput.btnNextClick(Sender: TObject);
var input:TextFile;
    NamaFile,Nama:string;
    Pmin,Pmax,a2,a1,a0,Sh,Sc,Ramp,Load,Res:double;
    i,j,Tup,Tdown,Tcold,InitSt,Ngen,Njam:integer;
begin
  if btnNext.Caption='&Save' then
  begin
    if SaveDialog1.Execute then
    begin
      NamaFile:=SaveDialog1.FileName;
      AssignFile(input,NamaFile+'.txt');
      Reset(input);
      Ngen:=StrToInt(edtNgen.Text);
      Njam:=StrToInt(edtNjam.Text);
      Writeln(input,Ngen);
      Writeln(input,Njam);
      for i:=1 to Ngen do
      begin
        Nama:=fgGen.Cells[1,i];
        Pmax:=StrToFloat(fgGen.Cells[2,i]);
        Pmin:=StrToFloat(fgGen.Cells[3,i]);
        a0:=StrToFloat(fgGen.Cells[4,i]);
        a1:=StrToFloat(fgGen.Cells[5,i]);
        a2:=StrToFloat(fgGen.Cells[6,i]);
        Tup:=StrToInt(fgGen.Cells[7,i]);
        Tdown:=StrToInt(fgGen.Cells[8,i]);
        Sh:=StrToFloat(fgGen.Cells[9,i]);
        Sc:=StrToFloat(fgGen.Cells[10,i]);
        Tcold:=StrToInt(fgGen.Cells[11,i]);
        InitSt:=StrToInt(fgGen.Cells[12,i]);
        Ramp:=StrToFloat(fgGen.Cells[13,i]);
        Writeln(input,Pmax:7:0,' ',Pmin:7:0,' ',
        a0:9:4,' ',a1:9:4,' ',a2:9:5,' ',Tup,' ',Tdown,' ',
        Sh:7:0,' ',Sc:7:0,' ',tcold,' ',InitSt,' ',Ramp:7:0,' ',Nama);
```

```
      end;
      for i:=1 to Njam do
      begin
       Load:=StrToFloat(fgLoad.Cells[1,i]);
       Res:=StrToFloat(fgLoad.Cells[2,i]);
       Writeln(input,Load,Res);
      end;
      fgPLN.RowCount:=Ngen+1;
      fgPLN.ColCount:=Njam+1;
      for i:=1 to Ngen do
      begin
       for j:=1 to Njam do
       begin
        Load:=StrToFloat(fgPLN.Cells[j,i]);
        Write(input,Load:7:2,' ');
       end;
       Writeln(input,'');
      end;
      CloseFile(input);
     end;
    end
    else if btnNext.Caption='&Next' then
    begin
     frmHasil.edtLength.Text:=IntToStr(gObjFunc.Ngen);
     frmHasil.cmbJam.Items.Clear;
     for i:=1 to gObjFunc.Njam do
     begin
      frmHasil.cmbJam.AddItem(IntToStr(i),nil);
     end;
     frmHasil.cmbJam.Text:='1';
     frmHasil.Show;
    end;
  end;
  end.
```

unit uHasil;

```
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms,
  Dialogs, ExtCtrls, TeEngine, Series, TeeProcs, Chart, Grids,
ComCtrls,
  StdCtrls, jpeg;
type
  TfrmHasil = class(TForm)
    TabSheet5: TTabSheet;
    TabSheet6: TTabSheet;
    Panel1: TPanel;
    btnClose: TButton;
    btnHitungRGA: TButton;
    TabSheet4: TPageControl;
    pbIterasi: TProgressBar;
    GroupBox1: TGroupBox;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label11: TLabel;
    edtMaxGen: TEdit;
    edtPopSize: TEdit;
    edtNParam: TEdit;
    edtBetha: TEdit;
    edtKa: TEdit;
    btnUseDefault: TButton;
    GroupBox2: TGroupBox;
    cmbJam: TComboBox;
    fgDaya: TStringGrid;
    GroupBox3: TGroupBox;
    Label1: TLabel;
    edtPinGen: TEdit;
    Label8: TLabel;
    Label9: TLabel;
    edtCostRGA: TEdit;
    edtCostPLN: TEdit;
    Label17: TLabel;
    Label7: TLabel;
    Label10: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    edtLength: TEdit;
    edtPCross: TEdit;
    edtPMutasi: TEdit;
    edtPFlip: TEdit;
    cmbCrossMethod: TComboBox;
    cmbNewParentMethod: TComboBox;
    Label16: TLabel;
    edtEmisiRGA: TEdit;
    Label18: TLabel;
    edtEmisiPLN: TEdit;
    Label19: TLabel;
    Panel2: TPanel;
    Image1: TImage;
    procedure btnCloseClick(Sender: TObject);
    procedure btnUseDefaultClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure btnHitungRGAClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmHasil: TfrmHasil;
implementation
uses uObjFunc, uUtils, uGenerator, uFitness2, uGenVar,
uGABin;
{$R *.dfm}
procedure TfrmHasil.btnCloseClick(Sender: TObject);
begin
  Close;
end;
procedure TfrmHasil.btnUseDefaultClick(Sender: TObject);
begin
  edtMaxGen.Text:='100';
  edtPopSize.Text:='20';
  edtBetha.Text:='0.3';
  edtKa.Text:='1000000000';
  edtLength.Text:='10';
  edtPCross.Text:='0.6';
  edtPMutasi.Text:='0.009';
  edtPFlip.Text:='0.5';
  edtPinGen.Text:='1000000';
  btnHitungRGA.Enabled:=true;
end;
```

```
procedure TfrmHasil.FormCreate(Sender: TObject);
begin
  fgDaya.Cells[0,0]:='No';
  fgDaya.Cells[1,0]:='Pg RGA (MW)';
  fgDaya.Cells[2,0]:='Pg PLN (MW)';
  fgDaya.Cells[3,0]:='Cost RGA ($)';
  fgDaya.Cells[4,0]:='Cost PLN ($)';
  fgDaya.Cells[5,0]:='RGA NOx (ton)';
  fgDaya.Cells[6,0]:='PLN NOx (ton)';
  cmbCrossMethod.Text:='crArithmetic';
  cmbNewParentMethod.Text:='npElitism';
end;
procedure TfrmHasil.btnHitungRGAClick(Sender: TObject);
var gas:TGABin2;
    i,sa,jam,MaxGen,PopSize,Length,NParam:integer;
    Load,Betha,Ka,PCross,PMutasi,PFlip,PCrossDC:double;
sumRGA,sumPLN,sumNOxPLN,sumNOxRGA,pinGen:dou
ble;
    PLN:dArr2;
    BestChrom:bArr2;
    LoadAsli,BestPgen,Min,Avg,Max:dArr1;
    Gen,GenAsli:TGenArr;
    BatasGen:TBatasArr1;
    CrossMethod:TCrossType;
    NewParent:TNewParent;
    Parent:TPopDouble1;
begin
  PLN:=gObjFunc.PLN;
  LoadAsli:=gObjFunc.Beban;
  GenAsli:=gObjFunc.Gen;
  jam:=StrToInt(cmbJam.Text);
  sa:=0;
  for i:=1 to high(PLN) do
  begin
    if PLN[i,jam]<>0 then
    begin
      inc(sa);
    end;
  end;
  SetLength(Gen,sa);
  sa:=0;
  for i:=1 to gObjFunc.Ngen do
  begin
    if PLN[i,jam]<>0 then
    begin
      Gen[sa]:=TPembangkit.Create(GenAsli[i]);
      inc(sa);
    end;
  end;
  NParam:=sa;
  edtNParam.Text:=IntToStr(NParam);
  Load:=LoadAsli[jam];
  PinGen:=StrToFloat(edtPinGen.Text);
  gFitness2:=TFitness2.Create(PinGen,Gen,Load);
  BatasGen:=gFitness2.BatasGen;
  for i:=1 to high(GenAsli) do
  begin
    GenAsli[i].Free;
  end;
  MaxGen:=StrToInt(edtMaxGen.Text);
  pbIterasi.Max:=MaxGen;
  PopSize:=StrToInt(edtPopSize.Text);
  Betha:=StrToFloat(edtBetha.Text);
  Ka:=StrToFloat(edtKa.Text);
  PCross:=StrToFloat(edtPCross.Text);
  PMutasi:=StrToFloat(edtPMutasi.Text);
  PFlip:=StrToFloat(edtPFlip.Text);
  Length:=StrToInt(edtLength.Text);
  CrossMethod:=crOne;
  if cmbCrossMethod.Text='crOne' then
  begin
    CrossMethod:=crOne;
  end
  else if cmbCrossMethod.Text='crTwo' then
  begin
    CrossMethod:=crTwo;
  end
  else if cmbCrossMethod.Text='crMulti' then
  begin
    CrossMethod:=crMulti;
  end
  else if cmbCrossMethod.Text='crArithmetic' then
  begin
    CrossMethod:=crArithmetic;
  end;
  if cmbNewParentMethod.Text='npElitism' then
  begin
    NewParent:=npElitism;
  end;
 gas:=TGABin2.Create(MaxGen,PopSize,Length,NParam,Ka,
PCross,PMutasi,PFlip,
      CrossMethod,NewParent,BatasGen);
  BestChrom:=gas.BestChrom;
  BestPgen:=gas.DecodeChromToPgen(BestChrom);
  Min:=gas.Min;
  Avg:=gas.Avg;
  Max:=gas.Max;
  gas.Free;
  for i:=0 to high(Min) do
  fgDaya.RowCount:=high(BestPgen)+2;
  sa:=0;
  sumRGA:=0;
  sumPLN:=0;
  sumNOxPLN:=0;
  sumNOxRGA:=0;
  for i:=1 to gObjFunc.Ngen do
  begin
    if PLN[i,jam]<>0 then
    begin
      fgDaya.Cells[0,sa+1]:=IntToStr(sa+1);
fgDaya.Cells[1,sa+1]:=FormatFloat('#,##0.00',BestPgen[sa]);

fgDaya.Cells[2,sa+1]:=FormatFloat('#,##0.00',PLN[i,jam]);

fgDaya.Cells[3,sa+1]:=FormatFloat('#,##0',Gen[sa].GetBiaya
(BestPgen[sa]));
fgDaya.Cells[4,sa+1]:=FormatFloat('#,##0',Gen[sa].GetBiaya
(PLN[i,jam]));
fgDaya.Cells[5,sa+1]:=FormatFloat('#,##0.0000',Gen[sa].Get
EmisiNOx(BestPgen[sa]));
fgDaya.Cells[6,sa+1]:=FormatFloat('#,##0.0000',Gen[sa].Get
EmisiNOx(PLN[i,jam]));
      sumRGA:=sumRGA+Gen[sa].GetBiaya(BestPgen[sa]);
      sumPLN:=sumPLN+Gen[sa].GetBiaya(PLN[i,jam]);
sumNOxRGA:=sumNOxRGA+Gen[sa].GetEmisiNOx(BestP
gen[sa]);
sumNOxPLN:=sumNOxPLN+Gen[sa].GetEmisiNOx(PLN[i,
jam]);
```

```
    inc(sa);
   end;
  end;
  edtCostRGA.Text:=FormatFloat('#,##0',sumIGA);
  edtCostPLN.Text:=FormatFloat('#,##0',sumPLN);
edtEmisiRGA.Text:=FormatFloat('#,##0.0000',sumNOxRGA
);
edtEmisiPLN.Text:=FormatFloat('#,##0.0000',sumNOxPLN)
;
  for i:=0 to high(Gen) do
  begin
   Gen[i].Free;
  end;
  gFitness2.Free;
end;
end.
```

unit uMenu;

```
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms,
  Dialogs, ComCtrls, StdCtrls, ExtCtrls, jpeg;
type
  TfrmMenu = class(TForm)
    Panel1: TPanel;
    btnNew: TButton;
    btnOpen: TButton;
    btnExit: TButton;
    StatusBar1: TStatusBar;
    Panel2: TPanel;
    OpenDialog1: TOpenDialog;
    Image3: TImage;
    procedure btnExitClick(Sender: TObject);
    procedure btnNewClick(Sender: TObject);
    procedure btnOpenClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frmMenu: TfrmMenu;
implementation
uses uComplex, uUtils, uInputGen, uObjFunc, uGenerator;
{$R *.dfm}
procedure TfrmMenu.btnExitClick(Sender: TObject);
begin
  gObjFunc.Free;
  Application.Terminate;
end;
procedure TfrmMenu.btnNewClick(Sender: TObject);
begin
  frmInput.Caption:='Input Data';
  frmInput.btnNext.Caption:='&Save';
  frmInput.Show;
end;
procedure TfrmMenu.btnOpenClick(Sender: TObject);
var NamaFile,Nama:string;
    output:TextFile;
Pmin,Pmax,a2,a1,a0,Sh,Sc,Ramp,Load,Res,ae2,ae1,ae0:doub
le;
    i,j,Ngen,Njam,Tup,Tdown,Tcold,InitSt:integer;
    aLoad,aRes:dArr1;
    aPLN:dArr2;
    aGen:TGenArr;
begin
  try
    if OpenDialog1.Execute then
    begin
      NamaFile:=OpenDialog1.FileName;
      AssignFile(output,NamaFile);
      Reset(output);
      Readln(output,Ngen);
      Readln(output,Njam);
      frmInput.edtNGen.Text:=IntToStr(Ngen);
      frmInput.edtNjam.Text:=IntToStr(Njam);
      frmInput.fgGen.RowCount:=Ngen+1;
      SetLength(aGen,Ngen+1);
      for i:=1 to Ngen do
      begin
Readln(output,Pmax,Pmin,a0,a1,a2,Tup,Tdown,Sh,Sc,tcold,I
nitSt,
        Ramp,ae0,ae1,ae2,Nama);
aGen[i]:=TPembangkit.Create(Nama,Pmin,Pmax,a2,a1,a0,Sh
,Sc,Ramp,
            ae2,ae1,ae0,Tup,Tdown,Tcold,InitSt);
        frmInput.fgGen.Cells[1,i]:=Nama;
        frmInput.fgGen.Cells[2,i]:=FloatToStr(Pmax);
        frmInput.fgGen.Cells[3,i]:=FloatToStr(Pmin);
        frmInput.fgGen.Cells[4,i]:=FloatToStr(a0);
        frmInput.fgGen.Cells[5,i]:=FloatToStr(a1);
        frmInput.fgGen.Cells[6,i]:=FloatToStr(a2);
        frmInput.fgGen.Cells[7,i]:=IntToStr(Tup);
        frmInput.fgGen.Cells[8,i]:=IntToStr(Tdown);
        frmInput.fgGen.Cells[9,i]:=FloatToStr(Sh);
        frmInput.fgGen.Cells[10,i]:=FloatToStr(Sc);
        frmInput.fgGen.Cells[11,i]:=IntToStr(Tcold);
        frmInput.fgGen.Cells[12,i]:=IntToStr(InitSt);
        frmInput.fgGen.Cells[13,i]:=FloatToStr(Ramp);
        frmInput.fgGen.Cells[14,i]:=FloatToStr(ae2);
        frmInput.fgGen.Cells[15,i]:=FloatToStr(ae1);
        frmInput.fgGen.Cells[16,i]:=FloatToStr(ae0);
      end;
      frmInput.fgLoad.RowCount:=Njam+1;
      SetLength(aLoad,Njam+1);
      SetLength(aRes,Njam+1);
      for i:=1 to Njam do
      begin
        Readln(output,Load,Res);
        aLoad[i]:=Load;
        aRes[i]:=Res;
        frmInput.fgLoad.Cells[1,i]:=FloatToStr(Load);
        frmInput.fgLoad.Cells[2,i]:=FloatToStr(Res);
      end;
      frmInput.fgPLN.RowCount:=Ngen+1;
      frmInput.fgPLN.ColCount:=Njam+1;
      SetLength(aPLN,Ngen+1,Njam+1);
      for i:=1 to Ngen do
      begin
        for j:=1 to Njam do
        begin
          Read(output,Load);
          aPLN[i,j]:=Load;
          frmInput.fgPLN.Cells[j,i]:=FloatToStr(Load);
        end;
        Readln(output);
```

```
    end;
    CloseFile(output);
    gObjFunc:=TObjFunc.Create(aLoad,aRes,aPLN,aGen);
    for i:=1 to Ngen do
    begin
      aGen[i].Free;
    end;
    frmInput.Caption:='Tampilan Data';
    frmInput.btnNext.Caption:='&Next';
    frmInput.Show;
   end;
  except
   MessageDlg('File Corrupt atau Error
Program!',mtWarning,[mbOK],0);
  end;
end;
end.
```