Desain dan Uji Komputasi Paralel Penentuan Nilai Penghalus (σ) Algoritma Jaringan Syaraf Probabilistik (*PNN*) untuk Klasifikasi Bunga Iris

Kudang Boro Seminar [1], Agus Buono[2] dan Teguh Pratama Januzir Sukin [3]

[1] *Staf Pengajar Departemen Teknik Pertanian, Fakultas Teknologi Pertanian IPB*
[2] *Staf Pengajar Departemen Ilmu Komputer, FMIPA IPB*
[3] *Alumni Departemen Ilmu Komputer, FMIPA IPB*

Abstrak

Probabilistic Neural Network (disingkat PNN) merupakan salah satu jaringan saraf tiruan yang banyak dikembangkan bagi kepentingan manusia. PNN memiliki tingkat akurasi klasifikasi yang cukup tinggi dan waktu pelatihan yang cukup singkat. Salah satu faktor yang mempengaruhi akurasi klasifikasi PNN adalah parameter penghalus (σ). Proses pencarian nilai σ yang optimum merupakan salah satu prapemrosesan yang harus dilakukan sebelum masuk ke PNN. Nilai parameter σ tidak dapat ditentukan secara langsung. Oleh sebab itu algoritma genetik dapat digunakan untuk mencari nilai σ yang optimum. Pada umumnya solusi dari algoritma genetik semakin baik apabila jumlah populasi yang dibangkitkan cukup besar dan proses alamiah evolusi sering terjadi. Hal ini tentunya membutuhkan banyak iterasi dan sumberdaya komputasi yang cukup besar. Permasalahan ini dapat diatasi dengan penerapan komputasi paralel pada algoritma genetik untuk menduga parameter penghalus (σ) dari model PNN yang optimum.

Penelitian ini bertujuan untuk menguji dan membandingkan kinerja dari algoritma PNN yang menggunakan komputasi paralel dengan yang menggunakan komputasi sequential (iteratif) untuk penentuan nilai penghalus (σ) yang diaplikasikan pada klasifikasi bunga Iris. Berbagai kriteria yang digunakan untuk uji analisis adalah akurasi klasifikasi PNN, waktu eksekusi total, rasio kebergantungan mesin, peningkatan kecepatan, dan efisiensi. Dari percobaan didapatkan kesimpulan bahwa secara umum kinerja algoritma paralel jauh lebih baik apabila dibandingkan dengan algoritma sekuensial (iteratif).

Kata kunci : *Komputasi Paralel, Probabilistic Neural Network (PNN), Nilai Penghalus (σ), Algoritma Genetik (GA), dan Klasifikasi Obyek.*

PENDAHULUAN

Latar Belakang

Dalam perkembangan komputasi yang pesat pada saat ini, telah banyak bidang kegiatan manusia yang diaplikasikan dengan menggunakan komputer. Beberapa fenomena komputasi yang cukup berkembang dengan cepat pada saat ini adalah mengenai komputasi paralel, algoritma genetik *(AG)* dan jaringan saraf tiruan *(JST)*.

Dengan adanya *JST*, suatu komputer dapat diprogram sedemikian rupa sehingga dapat melakukan pembelajaran sebagaimana layaknya manusia. Hal ini dimungkinkan disebabkan informasi yang ditangkap dan diolah oleh komputer disimulasikan dengan menggunakan sekumpulan neuron-neuron yang terkait satu sama lain mirip dengan pengolahan informasi pada jaringan saraf biologis makhluk hidup.

Salah satu *JST* yang banyak diteliti dan dikembangkan akhir-akhir ini oleh berbagai kalangan untuk berbagai kepentingan manusia adalah *Probabilistic Neural Network (PNN)*. Keunggulan yang dimiliki *PNN* adalah tingkat keakuratan yang cukup tinggi dan waktu pelatihannya yang cukup singkat.

Salah satu kendala utama dalam *PNN* adalah sulitnya mencari nilai parameter penghalus (σ). Proses pencarian nilai σ ini merupakan salah satu prapemrosesan sebelum masuk ke *PNN*. Bila kita mendapatkan nilai σ yang tepat maka *PNN* memiliki keakuratan sampai mendekati 100%. Tetapi tingkat keakuratan dari *PNN* bisa berkurang sampai dengan 50% apabila terjadi kesalahan dalam menentukan nilai parameter penghalus yang tepat.

Untuk mendapatkan parameter penghalus yang optimum, maka dapat digunakan *AG* yang merupakan suatu algoritma pencarian solusi kombinatorik yang bersifat acak sistematik. *AG* merupakan suatu teknik proses komputasi yang pada dasarnya meniru teori evolusi dari bidang ilmu Biologi.

Pada umumnya solusi yang didapatkan oleh *AG* semakin baik apabila jumlah populasi yang dibangkitkan cukup besar dan proses alamiah evolusi seperti mutasi dan rekombinasi sering terjadi. Keadaan ini tentunya membutuhkan banyak iterasi dan sumberdaya komputasi. Untuk mengatasinya maka digunakanlah komputasi paralel untuk menduga parameter penghalus (σ) dengan menggunakan *AG*. Diharapkan skenario paralel ini dapat meningkatkan kecepatan dan keakuratan penentuan nilai penghalus (σ) *PNN* yang optimum.

Tujuan

Penelitian ini bertujuan untuk:

1. Mendapatkan metode pencarian nilai σ secara paralel untuk diterapkan pada fungsi kernel *PNN*.
2. Mengetahui pengaruh nilai parameter σ persamaan kernel terhadap keakuratan klasifikasi *PNN* pada bunga *Iris*.
3. Membandingkan akurasi klasifikasi *PNN*, waktu total (kecepatan eksekusi program), rasio kebergantungan mesin *(Machine Dependent Ratio)*, peningkatan kecepatan *(Speed Up)*, dan efisiensi *(Efficiency)* antara pemrosesan *AG* sekuensial, *AG* paralel mode vektor baris, dan *AG* paralel mode vektor kolom.

Hasil dan Manfaat

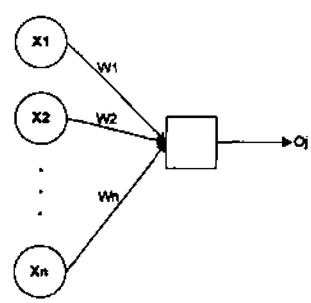
Hasil dari penelitian ini adalah kontribusi desain dan hasil uji komputasi paralel pencarian nilai penghalus (σ) algoritma *PNN* untuk aplikasinya pada proses klasifikasi obyek *(produk pertanian)*. Hasil penelitian ini dapat dijadikan acuan untuk mendisain sistem sortasi *(Seminar, Marimin, & Teguh 2002)* yang berbasis komputer di bidang agroindustri berskala menengah ke atas untuk sortasi produk secara masal dan komprehensif, mengarah pada pertanian presisi *(Seminar 2000)*.

TINJAUAN PUSTAKA

Jaringan Saraf Tiruan

Menurut *Fu (1994)*, Jaringan Saraf Tiruan *(JST)* merupakan suatu sistem pemrosesan informasi digital yang memiliki karakteristik-karakteristik seperti jaringan saraf pada makhluk hidup. Pada dasarnya, pemrosesan informasi pada *JST* mengacu pada pemrosesan informasi yang terjadi pada sel-sel saraf biologis, yaitu dengan pemancaran sinyal elektro kimia melalui serabut-serabut saraf *(neuron)*.

Setiap neuron menerima input *(x)* dari setiap neuron lain yang dikalikan dengan suatu nilai pembobotan *(W)* yang sesuai. Total penjumlahan akumulatif dari himpunan input terboboti dinamakan dengan level aktivasi. Level aktivasi inilah yang akan menentukan kemungkinan apakah suatu neuron dapat meneruskan sinyal ataukah tidak kepada neuron-neuron yang lain.



Gambar 1. *Bagan Model Aktivasi Sinyal Jaringan Saraf Tiruan (JST).*

$$LevelAktivasi = \sum_{i=1}^{n} X_i W_i \qquad (1)$$

Dengan

X_i : *Input ke-i*

W_i : *Bobot untuk input ke-i*

Sebelum dapat digunakan, *JST* harus diberikan pelatihan terlebih dahulu. Pelatihan ini diperlukan untuk menemukan nilai pembobotan yang tepat bagi *JST* agar keluarannya menjadi benar.

Pengklasifikasian Bayes

Menurut *Rish (2004)*, pengklasifikasian Bayes merupakan suatu metode klasifikasi yang menggunakan suatu fungsi diskriminan pada peluang posterior kelas. Fungsi klasifikasi yang digunakan oleh kelas i dapat dinotasikan dengan :

$$P(\omega_i)P(x \mid \omega_i) \tag{2}$$

Dengan

- $P(\omega_i)$: *Peluang kelas i*
- $P(x|\omega_i)$: *Peluang bersyarat x jika masuk ke dalam kelas i*
- X : *Vektor Input*
- ω_i : *Kelas i*

Pengklasifikasian *Bayes* banyak digunakan pada *PNN*. Dengan menggunakan metode *Bayes*, klasifikasi dapat dilakukan seoptimal mungkin dengan cara meminimisasi nilai kerugian yang terjadi bila terjadi kesalahan klasifikasi. Untuk mengklasifikasikan input x agar masuk ke dalam kelas *A*, maka harus dipenuhi syarat :

$$h_A c_A f_A(x) > h_B c_B f_B(x) \tag{3}$$

Dengan

- h_A : *Kemungkinan contoh terambil dari kelas A*
- h_B : *Kemungkinan contoh terambil dari kelas B*
- c_A : *Biaya yang dikorbankan bila terjadi kesalahan klasifikasi input A*
- c_B : *Biaya yang dikorbankan bila terjadi kesalahan klasifikasi input B*
- f_A : *Fungsi Kepekatan A*
- f_B : *Fungsi Kepekatan B*

Apabila syarat persamaan diatas tidak terpenuhi, maka input x dimasukkan kedalam kelas *B*.

Penduga Kepekatan Parzen

Menurut *Fu (1994)*, fungsi kepekatan yang digunakan untuk *PNN* yang berkaitan dengan data multivariat adalah fungsi kepekatan *Parzen*. Fungsi *Parzen* merupakan suatu prosedur non parametrik yang mensintesis Penduga *Probability Density Function (PDF) Gauss*. Fungsi *Parzen* akan memberikan keputusan klasifikasi setelah menghitung *PDF* untuk setiap kelas melalui pola pelatihan yang ada.

Pada fungsi *Parzen* terdapat fungsi pembobot yang disebut dengan fungsi *Kernel (K(x))*. Fungsi *Parzen* untuk data multivariat dapat dinotasikan dengan:

$$g(x) = \frac{1}{n\sigma^2}\sum_{i=1}^{n} K\left(\frac{x - x_i}{\sigma}\right) \tag{4}$$

Sedangkan fungsi *Kernel* yang digunakan adalah fungsi *Gauss* dinotasikan dengan :

$$K(x) = \frac{1}{\sigma(2\pi)^{\frac{d}{2}}}\exp\left(\frac{-x^2}{2\sigma^2}\right) \tag{5}$$

Maka kita mendapatkan fungsi kepekatan untuk kelas *A* sebagai berikut :

$$f_A(x) = \frac{1}{N_A\sigma^d(2\pi)^{\frac{d}{2}}}\sum_{i=1}^{N_A}\exp\left(\frac{-(x-x_{Ai})^T(x-x_{Ai})}{2\sigma^2}\right) \tag{6}$$

Jadi dapat dinotasikan :

$$P(\omega_A) = \frac{N_A}{N} \tag{7}$$

$$P(x \mid \omega_A) = \frac{1}{N_A\sigma^d(2\pi)^{\frac{d}{2}}}\sum_{i=1}^{N_A}\exp\left(\frac{-(x-x_{Ai})^T(x-x_{Ai})}{2\sigma^2}\right) \tag{8}$$

Sehingga

$$P(\omega_A)P(x \mid \omega_A) = \frac{1}{N\sigma^d(2\pi)^{\frac{d}{2}}}\sum_{i=1}^{N_A}\exp\left(\frac{-(x-x_{Ai})^T(x-x_{Ai})}{2\sigma^2}\right) \tag{9}$$

Dengan

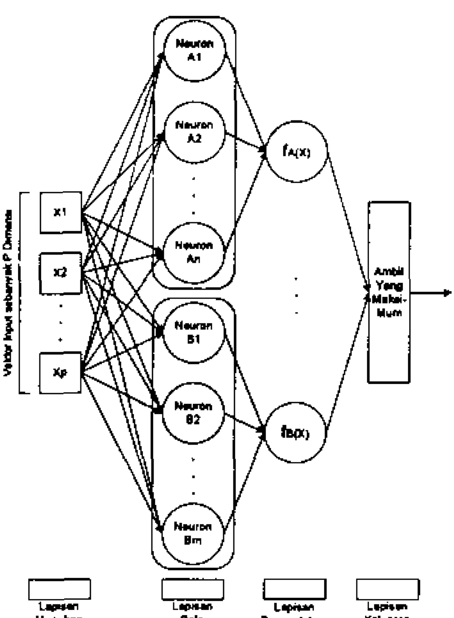
- $P(\omega_A)$: *Peluang kelas A*
- $P(x|\omega_A)$: *Peluang bersyarat x jika masuk ke dalam kelas A*
- x_{Ai} : *Pola pelatihan ke-i kelas A*
- d : *Dimensi vektor input*
- N_A : *Jumlah pola pelatihan kelas A*
- N : *Jumlah pola pelatihan seluruh kelas*
- σ : *Faktor penghalus*

Faktor Penghalus Sigma (σ)

Menurut *Lee, D. X, et. al. (2004)*, σ merupakan suatu nilai parameter yang berguna untuk menghaluskan fungsi kernel. Secara tidak langsung nilai σ berperan pula dalam menentukan ketepatan klasifikasi *PNN*. Nilai σ tidak dapat ditentukan secara langsung, akan tetapi bisa didapatkan melalui metode statistik maupun dari hasil coba-coba. Pada penelitian ini, nilai σ didapatkan melalui *AG*.

Probabilistic Neural Network (PNN)

Menurut *Rahmani (2004)*, *PNN* merupakan *JST* yang menggunakan teorema probabilitas klasik seperti pengklasifikasian *Bayes* dan penduga kepekatan *Parzen*. Proses yang dilakukan oleh *PNN* dapat berlangsung lebih cepat bila dibandingkan dengan *JST* Propagasi Balik. Hal ini terjadi disebabkan *PNN* hanya membutuhkan satu kali iterasi pelatihan bila dibandingkan dengan *JST* Propagasi Balik yang membutuhkan beberapa kali iterasi dalam proses pelatihannya.



Gambar 2. *Bagan Model Jaringan Saraf Tiruan Probabilistic Neural Network.*

Walaupun demikian, keakuratan dari klasifikasi *PNN* sangat ditentukan oleh nilai σ dan pola pelatihan yang diberikan. Bila nilai σ yang diterapkan pada *PNN* tepat, maka akurasi klasifikasi akan mendekati atau mencapai 100%. Bila nilai σ yang diterapkan tidak tepat maka akurasi klasifikasi *PNN* akan berkurang.

Demikian pula dengan pola pelatihan *PNN*. Apabila pola pelatihan dan data masukan pada satu kelas yang sama sangat berbeda jauh nilainya, maka *PNN* akan mengekstrapolasi data masukan tersebut. Hal inilah yang nantinya akan mengakibatkan akurasi klasifikasi *PNN* turun cukup drastis. Pada penelitian ini penentuan nilai σ yang optimum menggunakan algoritma genetik *(AG)*.

Pemrosesan Paralel

Menurut *Pacifico, et. al. (1998)*, Pemrosesan Paralel adalah penggunaan banyak prosesor yang saling bekerjasama satu sama lain untuk mencari suatu solusi tunggal dari suatu permasalahan. Pemrosesan paralel dapat digunakan untuk beberapa keperluan, diantaranya adalah untuk mempercepat waktu eksekusi dan mendistribusikan pencarian solusi dari permasalahan yang sangat kompleks.

Hukum Amdahl

Menurut *Amdahl (1967)*, peningkatan dari pemrosesan paralel tidak hanya bergantung pada banyaknya prosesor yang digunakan, akan tetapi lebih dipengaruhi oleh fraksi rasio antara intruksi sekuensial dengan keseluruhan instruksi pada suatu program.

$$SpeedUp = \frac{1}{\left(\frac{1-F}{N}\right)+F} \tag{10}$$

dimana

$$F = \frac{I_{Sek}}{I_{Sek} + I_{Par}} \tag{11}$$

Dengan

SpeedUp	: *Peningkatan Kecepatan*
F	: *Fraksi rasio intruksi sekuensial terhadap keseluruhan instruksi*
1-F	: *Fraksi rasio intruksi paralel terhadap keseluruhan instruksi*
I_{Sek}	: *Instruksi Sekuensial*
I_{Par}	: *Instruksi Paralel*
N	: *Banyaknya prosesor*

Hukum *Amdahl* menunjukkan bahwa peningkatan kinerja dari kecepatan *(Speed Up)* proses paralel dengan $F = 0{,}01$ hanya naik 10 kali lipat walaupun jumlah prosesor dinaikkan dari 10 buah menjadi 1000 buah *(sebanyak 100 kali lipat)*. Jadi nilai *F* pada program harus terus diperkecil untuk meningkatkan kecepatan pemrosesan paralel.

Bila hanya diketahui waktu pemrosesan intruksi paralel dan waktu pemrosesan instruksi sekuensial saja, maka perumusan *Speed Up* menjadi :

$$SpeedUp = \frac{T_{Sek} + T_{Par}}{T_{Sek} + \frac{T_{Par}}{N}} \qquad (12)$$

Dengan

SpeedUp : *Peningkatan Kecepatan*

T_{Par} : *Waktu yang dibutuhkan sebuah prosesor untuk mengeksekusi perintah paralel*

T_{Sek} : *Waktu yang dibutuhkan sebuah prosesor untuk mengeksekusi perintah sekuensial*

N : *Banyaknya prosesor*

Efisiensi (Efficiency)

Efisiensi adalah rasio antara *Speed Up* dengan banyaknya prosesor *(N)*.

$$Efficiency = \frac{SpeedUp}{N} \qquad (13)$$

Dengan

SpeedUp : *Peningkatan Kecepatan*

N : *Banyaknya Prosesor*

Bila nilai efisiensi semakin tinggi, maka kinerja program semakin baik. Dan bila sebaliknya maka kinerja dari program semakin buruk.

Machine Dependent Ratio (MDR)

MDR adalah suatu rasio antara waktu komunikasi dengan waktu kalkulasi. Semakin besar *MDR* maka waktu pemrosesan paralel semakin lambat. Demikian pula bila sebaliknya.

$$MDR = \frac{T_{Comm}}{T_{Calc}} \qquad (14)$$

Dengan

T_{Comm} : *Waktu yang dibutuhkan untuk mentransfer sebuah pesan antara dua buah simpul (node).*

T_{Calc} : *Waktu yang dibutuhkan untuk melakukan suatu operasi kalkulasi floating point.*

Salah satu strategi untuk memperkecil *MDR* ialah dengan memperbesar *overlaping* antara waktu komunikasi dan waktu kalkulasi pada setiap *node* komputer.

***Message Passing Interface* (MPI)**

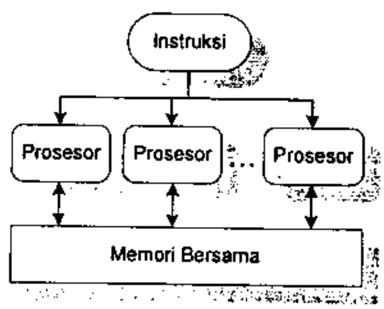
MPI digunakan untuk pemrosesan paralel pada arsitektur pemrograman *SIMD (Single Instruction Multiple Data)* dengan menggunakan pemrograman soket *(Socket Programming)*. Kelebihan dari *MPI* adalah adanya suatu mekanisme untuk sinkronisasi proses pada komputasi paralel. Fungsi sinkronisasi yang digunakan dalam penelitian ini adalah mekanisme *blocking* dalam pengiriman data *(MPI Isend)*, *blocking* dalam penerimaan data *(MPI Irecv)*, dan *blocking* dalam eksekusi *(MPI Barrier)*.

Pemrograman Soket

Menurut *Quinton (1997)*, Pemrograman Soket adalah pemrograman jaringan untuk mengembangkan sistem *Client Server* dengan menggunakan protokol *TCP/IP* atau *UDP/IP*. Pemrograman ini meliputi sistem pengalamatan jaringan yang terdiri atas *services*, *sockets* dan *port*.

Single Instruction Multiple Data (SIMD)

Menurut *Flynn (1966)*, *SIMD* adalah suatu klasifikasi arsitektur perangkat keras komputer yang menerapkan satu buah pengontrol *(controller)* pada beberapa unit elemen pemroses data untuk memroses beberapa aliran data secara bersamaan.



Gambar 3. *Bagan Model SIMD.*

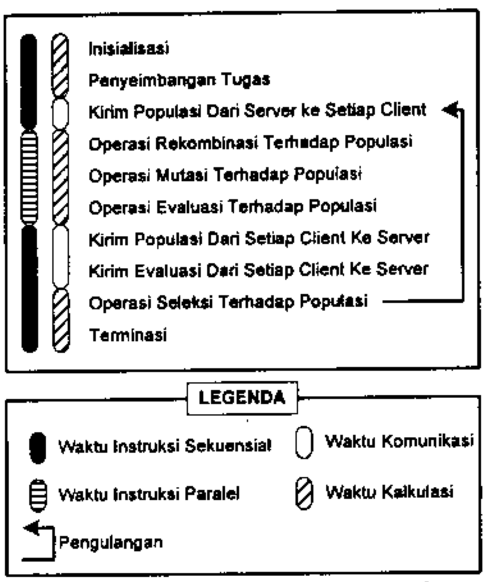
METODE PERANCANGAN DAN PENGUJIAN

Komputasi Data

Data-data yang dihitung dalam perancangan ini meliputi :

1. *Waktu Total:* waktu yang dibutuhkan sistem untuk menyelesaikan satu kali eksekusi secara keseluruhan.
2. *Waktu Komunikasi:* waktu yang dibutuhkan komputer *Server* untuk saling berkomunikasi dengan komputer-komputer *Client*.
3. *Waktu Kalkulasi:* waktu yang dibutuhkan setiap komputer untuk menyelesaikan seluruh proses eksekusi tanpa melakukan proses komunikasi.
4. *Waktu Instruksi Sekuensial:* waktu yang dibutuhkan komputer *Server* untuk melakukan proses-proses sekuensial.
5. *Waktu Instruksi Paralel:* waktu yang dibutuhkan oleh setiap komputer untuk melakukan instruksi yang dikerjakan secara paralel.
6. *Nilai Sigma (σ) terbaik*
7. *Tingkat Klasifikasi PNN terbaik*
8. *Rasio Kebergantungan Mesin*
9. *Peningkatan Kecepatan (Speed Up)*
10. *Efisiensi*

Fungsi waktu yang mencatat perubahan waktu pada program yang dibuat hanya memiliki tingkat ketelitian hingga satu detik. Artinya suatu proses komputasi yang berlangsung kurang dari satu detik, akan dihitung memiliki perubahan waktu sama dengan nol.



Gambar 4. Bagan Model Waktu Instruksi

Pada gambar diatas dapat dilihat bagaimana cara Program Utama mencatat waktu dari setiap proses yang terjadi. Waktu total bisa didapatkan dari penjumlahan waktu komunikasi ditambah waktu kalkulasi atau dari waktu instruksi sekuensial ditambah waktu instruksi paralel. Bila mode komputasi dilakukan secara sekuensial, maka waktu instruksi paralel dan waktu komunikasi sama dengan nol. Sehingga waktu total, waktu instruksi sekuensial, dan waktu kalkulasi akan selalu bernilai sama.

Berkas Data

Dalam melakukan percobaan, setiap inisialisasi dan data-data yang masuk maupun yang keluar direpresentasikan dalam bentuk file-file teks data. Secara umum, file-file data yang digunakan dalam percobaan ini dibagi ke dalam 4 kategori, yaitu :

1. File-file Inisialisasi *(*.INI)*
2. File-file Masukan *(*.TXT)*
3. File-file Keluaran *(*.TXT, *.STA)*
4. File-file Sementara *(*.TD7, *.TMP, *.DBL, *.OLD, *.MTR)*

Perancangan Nilai Batas Parameter

Batasan-batasan nilai parameter yang digunakan adalah sebagai berikut :

1. Operator mutasi yang digunakan dalam *AG* adalah operator mutasi satu titik dan operator rekombinasi yang digunakan dalam *AG* adalah operator rekombinasi satu titik.
2. Nilai peluang mutasi *AG* adalah 0,6 dan Nilai peluang rekombinasi *AG* adalah 0,35. Sedangkan banyaknya generasi yang dibangkitkan setiap eksekusi *AG* adalah 10 dan ketelitian pencarian nilai σ kernel dibagi dalam 9.007.199.254.740.991 interval. Nilai-nilai parameter ini selalu konstan untuk setiap percobaan dan perulangan.
3. *PNN* digunakan sebagai fungsi operator evaluasi bagi tiap individu *AG*. Akurasi pengklasifikasian *PNN* akan menentukan nilai ketahanan hidup bagi masing-masing individu yang dibangkitkan oleh *AG*.
4. Operator seleksi *AG* yang digunakan adalah *Roda Roulette (Roulette Whell)*.
5. Setiap individu dalam *AG* hanya mewakili satu buah variabel tunggal, yaitu nilai parameter σ kernel *PNN*.
6. Parameter σ kernel *PNN* yang digunakan pada penelitian ini bersifat homogen,

karena data pelatihan dan data masukan hanya terdiri atas satu tipe data kontinyu yaitu satuan cm.

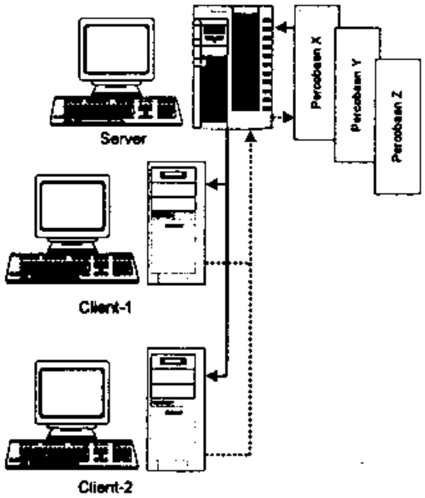
7. Data yang digunakan sebagai acuan pelatihan dan contoh masukan *PNN* berasal dari basis data bunga Iris yang dibuat oleh *R. A. Fisher*. Pengelompokan *PNN* dibagi atas 3 kelas: bunga *Iris Setosa*, *bunga Iris Versicolor*, dan bunga *Iris Virginica* masing-masing 50 vektor data.
8. Jumlah faktor perlakuan yang digunakan dalam analisis *MANOVA* terdiri atas 3 faktor, yaitu :
   a. Mode komputasi, terdiri dari Sekuensial, Paralel vektor baris, Paralel vektor kolom.
   b. Rasio jumlah vektor pelatihan dengan jumlah vektor contoh masukan *PNN*, terdiri dari :
      1) 135 vektor *(90%)* pelatihan dengan 15 vektor *(10%)* contoh masukan.
      2) 120 vektor *(80%)* pelatihan dengan 30 vektor *(20%)* contoh masukan.
      3) 105 vektor *(70%)* pelatihan dengan 45 vektor *(30%)* contoh masukan
      4) 90 vektor *(60%)* pelatihan dengan 60 vektor *(40%)* contoh masukan
      5) 75 vektor *(50%)* pelatihan dengan 75 vektor *(50%)* contoh masukan
      6) 60 vektor *(40%)* pelatihan dengan 90 vektor *(60%)* contoh masukan
      7) 45 vektor *(30%)* pelatihan dengan 105 vektor *(70%)* contoh masukan
      8) 30 vektor *(20%)* pelatihan dengan 120 vektor *(80%)* contoh masukan
      9) 15 vektor *(10%)* pelatihan dengan 135 vektor *(90%)* contoh masukan

   c. Populasi *AG*, terdiri dari :
      1) 1000 individu
      2) 5000 individu

Nilai respon yang akan diuji oleh *MANOVA* adalah nilai akurasi klasifikasi *PNN*, waktu total, rasio kebergantungan mesin, peningkatan kecepatan, dan efisiensi.

Perancangan Model Komunikasi

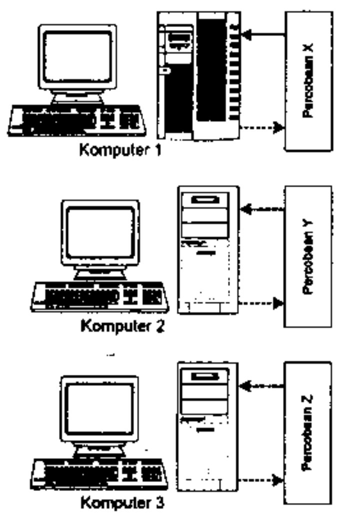
Dalam mode komputasi paralel, setiap komputer *Client* hanya berkomunikasi dengan komputer *Server* dan antar sesama komputer *Client* tidak berkomunikasi satu sama lain. Komunikasi hanya terjadi pada saat proses inisialisasi, pengimbangan tugas, pengiriman populasi dari *Server* ke *Client*, penerimaan populasi dari *Client* ke *Server*, dan penerimaan evaluasi dari *Client* ke *Server* ***(Gambar 5)***.



Gambar 5 *Bagan Percobaan Komputasi Paralel.*

Proses-proses kalkulasi dilakukan oleh komputer *Server* dan setiap komputer *Client*. Dengan kata lain komputer *Server* tidak hanya melakukan pengontrolan proses pada setiap komputer *Client* saja, akan tetapi ikut pula memproses data yang sudah terbagi-bagi bersama-sama dengan komputer-komputer *Client*.

Pada proses sekuensial ***(Gambar 6)*** komputer bekerja memproses pekerjaannya sendiri-sendiri tanpa adanya komunikasi satu



Gambar 6 *Bagan Percobaan Komputasi Sekuensial*

sama lain. Disebabkan oleh lambatnya waktu yang diperlukan komputasi sekuensial untuk melakukan suatu operasi *AG*, maka setiap komputer yang identik mengerjakan percobaan yang berbeda-beda. Diharapkan dengan dilakukannya mekanisme ini dapat memberikan hasil keluaran beberapa buah percobaan yang berbeda-beda dalam waktu yang jauh lebih singkat.

Prosedur Percobaan

Proses *AG* sekuensial terlihat pada bagan dibawah ini :

Gambar 7. *Bagan Kompleksitas Waktu Algoritma Genetik Sekuensial*

$$T_{Total} = T_{In} + U\left(T_{Re} + T_{Mu} + T_{Ev} + T_{Se}\right) + T_{So} \tag{15}$$

T_{In} + T_{So}	=	Biaya Rutin *(C_R)*
T_{Re}	≈	Populasi AG *(P)*
T_{Ev}	≈	Populasi AG *(P)* * Jumlah Data Masukan PNN *(N_M)* * Jumlah Data Pelatihan PNN *(N_P)*
T_{Mu}	≈	Populasi AG *(P)* * Jumlah Bit yang diperlukan untuk merepresentasikan satu buah individu *(R)*
T_{Se}	≈	Populasi AG *(P)* * *(Populasi AG (P)+3)*

$$T_{Total} = C_R + U\left[P + (PR) + (PN_M N_P) + (P(P+3))\right]$$

$$T_{Total} = C_R + UP\left[P + R + N_M N_P + 4\right] \tag{16}$$

Analisa AG sekuensuial:

1. *Kasus Terbaik*

   Kasus terbaik terjadi jika *Bit* yang digunakan dalam representasi kromoson *AG* adalah satu buah, jumlah populasi dua individu, dan kondisi terminasi dapat terpenuhi dalam satu kali iterasi.

$$T_{Total} = C_R + 2N_M N_P + 14$$

Kompleksitas $= \Omega(N_M N_P)$

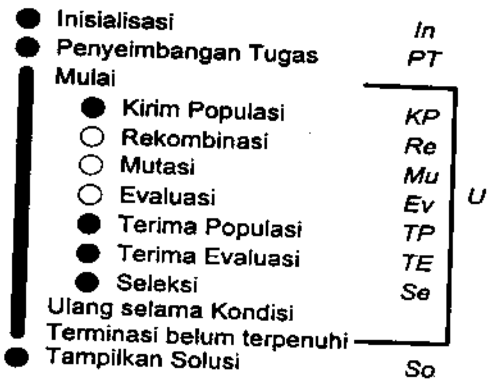
2. *Kasus Terburuk*

   Kasus terburuk terjadi apabila banyak *Bit* yang digunakan dalam representasi kromoson *AG* adalah 53 buah, jumlah populasi adalah N_{Pop} individu, kali kondisi terminasi dapat terpenuhi dalam N_{Itr} kali iterasi.

$$T_{Total} = C_R + \left(N_{Itr} N_{Pop} (N_{Pop} + N_M N_P + 57)\right)$$

Kompleksitas =

$$O\left(N_{Itr} N_{Pop} \left(N_{Pop} + N_M N_P\right)\right)$$



Gambar 8. *Bagan Kompleksitas Waktu Algoritma Genetik Paralel*

$$T_{Total} = T_{In} + T_{PT} + U\left(\begin{array}{c} T_{KP} + \left(\dfrac{T_{Re} + T_{Mu} + T_{Ev}}{N_{Pro}}\right) \\ + T_{TP} + T_{TE} + T_{Se} \end{array}\right) + T_{So} \tag{17}$$

$T_{In} + T_{So}$	=	Biaya Rutin (C_R)
T_{PT}	=	Biaya Komunikasi Paralelisme 1 (C_{P1})
T_{KP}+T_{TP}+ T_{TE}	=	Biaya Komunikasi Paralelisme 2 (C_{P2})
T_{Ev}	≈	Populasi AG (P) * Jumlah Data Masukan PNN (N_M) * Jumlah Data Pelatihan PNN (N_P)
T_{KP}, T_{Re}, T_{TP}, T_{TE}	≈	Populasi AG (P)
T_{Mu}	≈	Populasi AG (P) * Jumlah Bit yang diperlukan untuk merepresentasikan satu buah individu (R)
T_{Se}	≈	Populasi AG (P) * (Populasi AG (P)+3)
N_{Pro}	=	Banyaknya Prosesor

Maka

$$T_{Total} = \begin{array}{c} C_R + C_{P1} + \\ U\left(C_{P2} + \left(\dfrac{P + (PR) + PN_M N_P}{N_{Pro}}\right) + (P(P+3))\right) \end{array}$$

$$T_{Total} = C_R + C_{P1} + UC_{P2} + UP\left(\left(\frac{R + N_M N_P + 1}{N_{Pro}}\right) + P + 3\right) \quad (21)$$

Analisis AG paralel :

1. *Kasus terbaik*

Kasus terbaik terjadi apabila banyak Bit yang digunakan dalam kromosom *AG* adalah satu buah, jumlah populasi adalah $2N_{Pro}$ individu, kondisi terminasi dapat terpenuhi dalam satu kali iterasi, dan biaya paralelisme *(C_{P1} dan C_{P2})* ≈0.

$$T_{Total} = C_R + 4N_{Pro}^{\ 2} + 6N_{Pro} + 2N_M N_P + 4$$

$$\text{Kompleksitas} = \Omega\left(2N_{Pro}^{\ 2} + N_M N_P\right)$$

2. *Kasus Terburuk*

Kasus terburuk terjadi apabila banyak Bit yang digunakan dalam representasi kromosom *AG* adalah 53 buah, jumlah populasi adalah N_{Pop} individu, kondisi terminasi dapat terpenuhi dalam N_{Itr} kali iterasi, dan biaya paralelisme *(C_{P1} dan C_{P2})* sangat tinggi.

$$T_{Total} = C_R + C_{P1} + N_{Itr}\left(C_{P2} + \frac{N_{Pop}(N_M N_P + 54)}{N_{Pro}} + N_{Pop}^{\ 2} + 3N_{Pop}\right)$$

Kompleksitas=

$$O\left(N_{Itr}\left(C_{P2} + \frac{N_{Pop}(N_M N_P)}{N_{Pro}} + N_{Pop}^{\ 2}\right)\right)$$

Kode semu dibawah ini menunjukkan metode AG sekuensial yang digunakan dalam penelitian ini.

Algoritma Genetika Sekuensial

1. Buat Populasi Inisial
2. Selama Kriteria Terminasi belum tercapai, lakukan :
   a. Lakukan proses Rekombinasi pada populasi
   b. Lakukan proses Mutasi pada populasi
   c. Lakukan proses Evaluasi pada populasi
   d. Lakukan proses Seleksi pada populasi
3. Tampilkan Solusi

Gambar 9. *Kode Semu Algoritma Genetik Sekuensial*

Sedangkan kode semu dibawah ini menunjukkan metode *AG* paralel yang digunakan dalam penelitian ini.

Algoritma Genetika Paralel

Pada Komputer Server

1. Buat Populasi Inisial
2. Seimbangkan tugas untuk setiap prosesor dengan asumsi semua prosesor memiliki spesifikasi yang sama.
3. Selama Kriteria Terminasi belum tercapai, lakukan :
   a. Kirim sub-sub populasi kepada tiap komputer yang ada

Pada Komputer Server dan Client

   b. Lakukan proses Rekombinasi pada tiap sub populasi
   c. Lakukan proses Mutasi pada tiap sub populasi
   d. Lakukan proses Evaluasi pada tiap sub populasi
   e. Kumpulkan sub-sub populasi menjadi satu populasi beserta hasil evaluasi pada Komputer Server

Pada Komputer Server

   f. Lakukan proses Seleksi pada populasi
4. Tampilkan Solusi

Gambar 10. *Algoritma Genetik Paralel.*

Seleksi Algoritma Genetika Paralel

Pada Komputer Server

1. i = 0
2. Baca nilai evaluasi dari file "HasilEvaluasi"
3. Cari Total Kecocokkan (F) dengan menggunakan rumus :

$$F = \sum_{i=1}^{Populasi} Evaluasi \ (V_i)$$

4. i = 0
5. Selama i <> Jumlah Populasi, lakukan :
   a. Cari Peluang tiap individu (P_i) dengan menggunakan rumus :

$$P_i = \frac{Evaluasi \ (V_i)}{F}$$

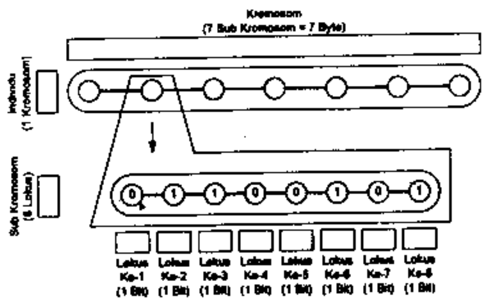
   b. Cari Peluang Kumulatif tiap individu (Q_i) dengan menggunakan rumus :

$$Q_i = \sum_{j=1}^{i} P_j$$

   c. i = i + 1
6. Bangkitkan bilangan acak [0..1] (r_i) untuk setiap individu (V_i) populasi
7. i=0
8. Selama i <> Jumlah Populasi, lakukan :
   a. Jika ($r_i < q_1$) → Pilih Individu ke-1 (V_1)
   Selainnya → Pilih V_j yang memenuhi $q_{j-1} < r_i < q_j$ dan $2 \le j \le$ Jumlah Populasi
   b. i = i + 1

Gambar 11. *Algoritma Seleksi Algoritma Genetik Paralel.*

Secara umum, representasi tipe data untuk satu individu yang terdiri atas satu buah kromosom pada *AG* dapat dilihat pada *gambar* dibawah ini.



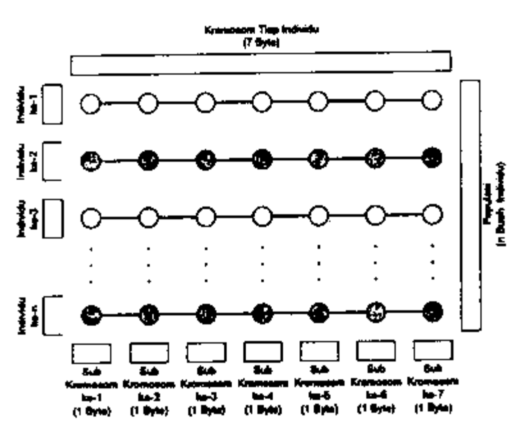
Gambar 12. B Representasi data Kromosom, Sub Kromosom, dan Lokus.

Pendefinisian tipe data kromosom hanya menggunakan 7 Byte *(56 Bit)* saja.

```
struct ModelSubKromosom
  {
  unsigned char Lokus8 : 1;
  unsigned char Lokus7 : 1;
  unsigned char Lokus6 : 1;
  unsigned char Lokus5 : 1;
  unsigned char Lokus4 : 1;
  unsigned char Lokus3 : 1;
  unsigned char Lokus2 : 1;
  unsigned char Lokus1 : 1;
  };
```

Gambar 13. Struktur Data Sub Kromosom dalam bahasa C++.

Dengan struktur sub-kromosom seperti di atas, data populasi dan individu direpresentasikan dalam bentuk matriks seperti disajikan pada **Gambar 14.**



Gambar 14. Representasi data Populasi dan Individu dalam bentuk matriks.

Tipe data vektor *MPI* dapat digunakan untuk melempar paket-paket data yang berisi data-data sub populasi yang digunakan pada *AG* paralel. Matriks populasi dapat dipecah menjadi dua metode pelemparan vektor yaitu: pelemparan vektor data baris dan pelemparan vektor data kolom.

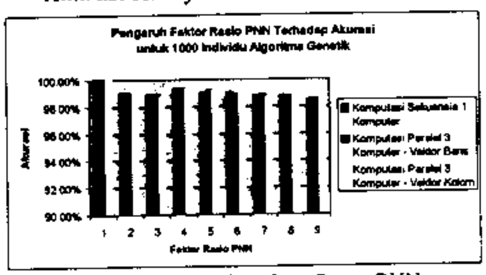
Implementasi

Implementasi sistem dilakukan dengan menggunakan perangkat lunak *Microsoft Visual C++ 6.0, Microsoft Visual Basic 6.0,* dan *Library MPICH* untuk *Microsoft Windows NT* pada komputer dengan spesifikasi prosesor *Intel Pentium 4 1,7 GHz, RAM 128 MB,* kapasitas *Hardisk Maxtor 20 GB,* dan Kartu Jaringan *Realtek RTL8139 Family PCI Fast Ethernet 10 Mbps.* Sedangkan sistem operasi yang digunakan adalah *Microsoft Windows XP* Professional. Untuk *Concentrator* jaringan digunakan Hub *FlexHub/18 16 Port 10 Base-T with BNC-AUI.*

HASIL DAN PEMBAHASAN

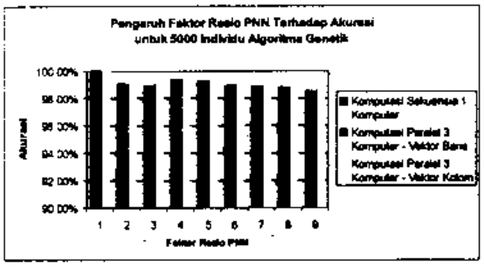
Pengamatan Grafik
Pengaruh Faktor Rasio PNN

1. *Pengaruh Faktor Rasio PNN terhadap Akurasi Klasifikasi PNN*



Gambar 15. Pengaruh Faktor Rasio PNN Terhadap Akurasi Klasifikasi PNN (1000 Individu).

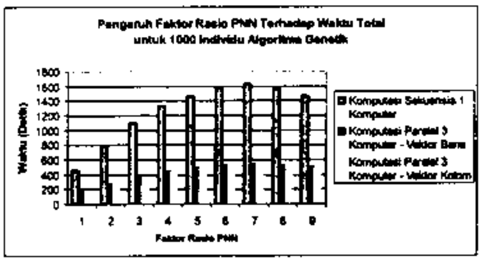
Dari **Gambar 15-16** terlihat bahwa akurasi klasifikasi *PNN* semakin meningkat sesuai dengan kenaikan jumlah populasi. Akurasi klasifikasi *PNN* tertinggi terjadi pada saat faktor rasio data masukan dan pelatihan *PNN* 10% : 90%. Sedangkan akurasi terendah terjadi pada saat faktor rasio data masukan dan pelatihan 90% :10%



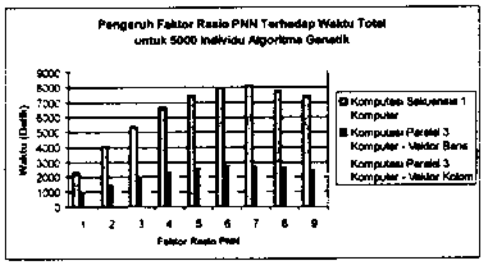
Gambar 16. Pengaruh Faktor Rasio Terhadap Akurasi Klasifikasi PNN (5000 Individu).

2. Pengaruh Faktor Rasio PNN terhadap Waktu Total

Dari **Gambar 17-18** terlihat bahwa waktu total semakin meningkat sesuai dengan kenaikan jumlah populasi. Walaupun demikian setiap grafik menunjukkan pola keteraturan yang tetap, yaitu berbentuk gelombang yang naik kemudian turun kembali



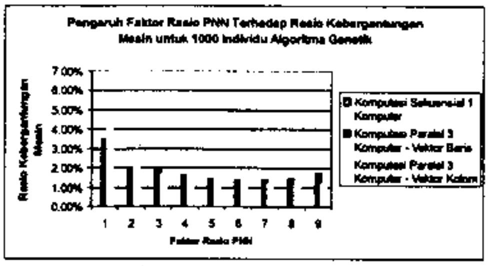
Gambar 17. Pengaruh Faktor Rasio PNN Terhadap Waktu Total (1000 Individu)



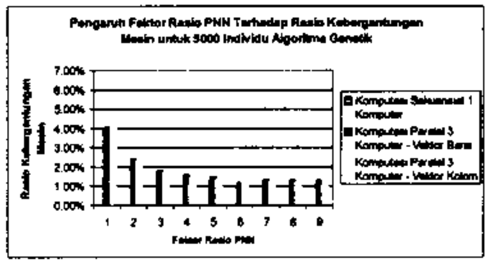
Gambar 18. Pengaruh Faktor Rasio PNN Terhadap Waktu Total (5000 Individu).

Pada grafik terlihat bahwa waktu total komputasi sekuensial selalu lebih besar bila dibandingkan dengan komputasi paralel. Dari pola grafik terlihat bahwa selisih waktu kalkulasi antara komputasi paralel mode vektor baris dan mode vektor kolom tidak signifikan.

3. Pengaruh Faktor Rasio PNN terhadap Rasio Kebergantungan Mesin

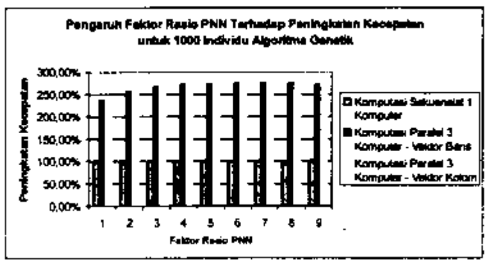


Gambar 19. Pengaruh Faktor Rasio PNN Terhadap Rasio Kebergantungan Mesin (1000 Individu).

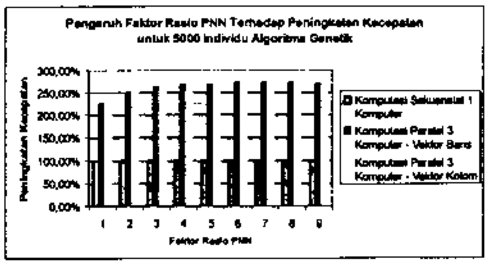


Gambar 20. Pengaruh Faktor Rasio PNN Terhadap Rasio Kebergantungan Mesin (5000 Individu).

4. Percobaan Pengaruh Faktor Rasio PNN terhadap Peningkatan Kecepatan



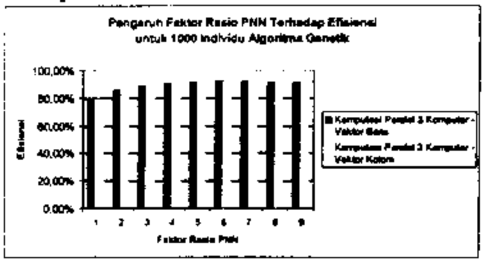
Gambar 21. Pengaruh Faktor Rasio PNN Terhadap Kecepatan (1000 Individu)



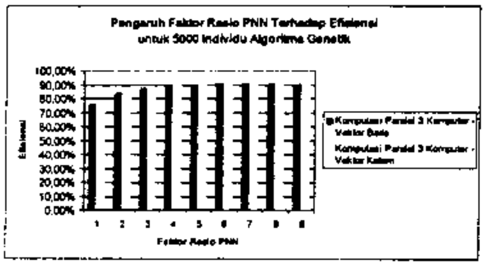
Gambar 22. Pengaruh Faktor Rasio PNN Terhadap Kecepatan (5000 Individu)

Dari **Gambar 21-22** terlihat bahwa kecepatan semakin meningkat dengan semakin bertambahnya data masukkan *PNN* dan semakin kecilnya data pelatihan *PNN*.

5. *Pengaruh Faktor Rasio PNN terhadap Efisiensi*



Gambar 23. *Pengaruh Faktor Rasio PNN Terhadap Efisiensi (1000 Individu)*



Gambar 24. *Pengaruh Faktor Rasio PNN Terhadap Efisiensi (5000 Individu)*

Dari **Gambar 23-24** terlihat bahwa efisiensi paralelisme meningkat dengan bertambahnya data masukan *PNN* dan semakin kecilnya data pelatihan *PNN*.

Analisis data

Tabel 1. *Rataan Akurasi Klasifikasi PNN, Waktu Total, Rasio Ketergantungan Mesin, Peningkatan Kecepatan, dan Efisiensi*

		Mode Komputasi		
		Sekuen-sial	Paralel Vektor Baris	Paralel Vektor Kolom
Rataan	Akurasi Klasifikasi PNN	98,60%	98,72%	98,69%
	Waktu Total	1717,39	662,77	658,59
	Rasio Kebergan-tungan Mesin	0,00%	2,02%	1,84%
	Peningkatan Kecepatan	100%	223,58%	224,22%
	Efisiensi	-	74,53%	74,74%

Tabel 2. *Ragam Akurasi Klasifikasi PNN, Waktu Total, Rasio Ketergantungan Mesin, Peningkatan Kecepatan, dan Efisiensi .*

		Mode Komputasi		
		Sekuen-sial	Paralel Vektor Baris	Paralel Vektor Kolom
Ragam	Akurasi Klasifikasi PNN	2,397 E-04	2,043 E-04	2,123 E-04
	Waktu Total	6734646	880340,6	861880,1
	Rasio Kebergan-tungan Mesin	0,00 E+00	9,982 E-04	8,125 E-04
	Peningkatan Kecepatan	0,000	0,282	0,279
	Efisiensi	-	3,182 E-02	3,095 E-02

Tabel 3. *Pengaruh Faktor Perlakuan dan Interaksinya Terhadap Variabel Respon Berdasarkan Uji MANOVA (0.95)*

		Respon			
		Akurasi Klasifikasi PNN	Waktu Total	Rasio Kebergantungan Mesin	Peningkatan Kecepatan
Faktor Perlakuan dan Interaksi	Mode Komputasi	-	√	√	√
	Rasio PNN	√	√	√	√
	Populasi AG	√	√	√	√
	Interaksi Mode Komputasi & Rasio PNN	-	√	√	√
	Interaksi Mode Komputasi & Populasi AG	-	√	√	√
	Interaksi Rasio PNN & Populasi AG	-	√	-	√
	Interaksi Mode Komputasi, Rasio PNN, & Populasi AG	-	√	√	√

KESIMPULAN DAN SARAN

Desain dan uji komputasi paralel telah membuktikan kinerja yang lebih unggul dibandingkan dengan komputasi sekuensial. Akurasi klasifikasi *PNN* terbukti menjadi lebih baik apabila nilai σ yang diberikan dalam *PNN* tepat, mencapai 91,11% - 100%.

Dari uji *MANOVA* terlihat bahwa mode komputasi, rasio *PNN*, populasi *AG* dan interaksinya berpengaruh nyata pada Waktu Total, Akurasi Kalsifikasi *PNN*, Rasio Ketergantungan Mesin (*MDR*), dan Efisiensi. Perbedaan kinerja komputasi paralel mode vektor baris dan mode vektor kolom tidak signifikan.

Pada penelitian ini hanya digunakan data bunga *IRIS* dan parameter penghalus (σ) yang homogen pada *PNN*. Untuk penelitian selanjutnya dapat digunakan pemodelan *PNN* yang lebih kompleks dengan melibatkan data dan parameter penghalus (σ) yang heterogen.

DAFTAR PUSTAKA

Flynn. 1966. Tanpa Judul. http//csep1.phy ornl.gov/ca/node11.html [11-12-2002]

Fu, Limin. 1994. *Neural Networks in Computer Intelligence*. ISBN 0-07-113319-4. Mc Graw-Hill. Singapura.

Lee, D. X, G. Thoma, & H. Weschler. 2004. *"Classification of Binary Document Images into Texture or Non-textual Data Blocks Using Neural Network Models"*. http://archive.nlm.nih.gov/pubs/doc_class/m v.php. [09-08-2004]

Pacifico, Mark & Mike Merrill. 1998. Tanpa Judul. http://www.cs.umd.edu/class/fall2001/cmsc4 11/projects/parallel2/proposal.html. [11-12-2002]

Quinton, Reggers. 1997. Tanpa Judul. http://www.uwo.ca/its/doc/courses/notes/soc ket/#INTRO. [11-8-2003]

Rahmani. 2004. *Probabilistic Neural Networks*. http://courses.iust.ac.ir/ ~rahmani/NN/pnn.htm. [09-08-2004]

Rish, I. 2004. *"An empirical study of the naive Bayes classifier"*. http://www.intellektik. informatik.tu-darmstadt.de/~tom/IJCAI01/ Rish.pdf. [09-08-2004]

Seminar, K..B. 2000. *Precision Agriculture: Paradigma dan Aplikasi*. Agrimedia ISSN 0853-8468 Vol.6 No. 1.

Seminar, Marimin, & Teguh. 2002. Aplikasi jaringan syaraf tiruan dan analisis komponen utama untuk sortasi . *Buletin Keteknikan Pertanian, ISSN 0216-3365. Vol 17, No. 2,* hal 39-52.