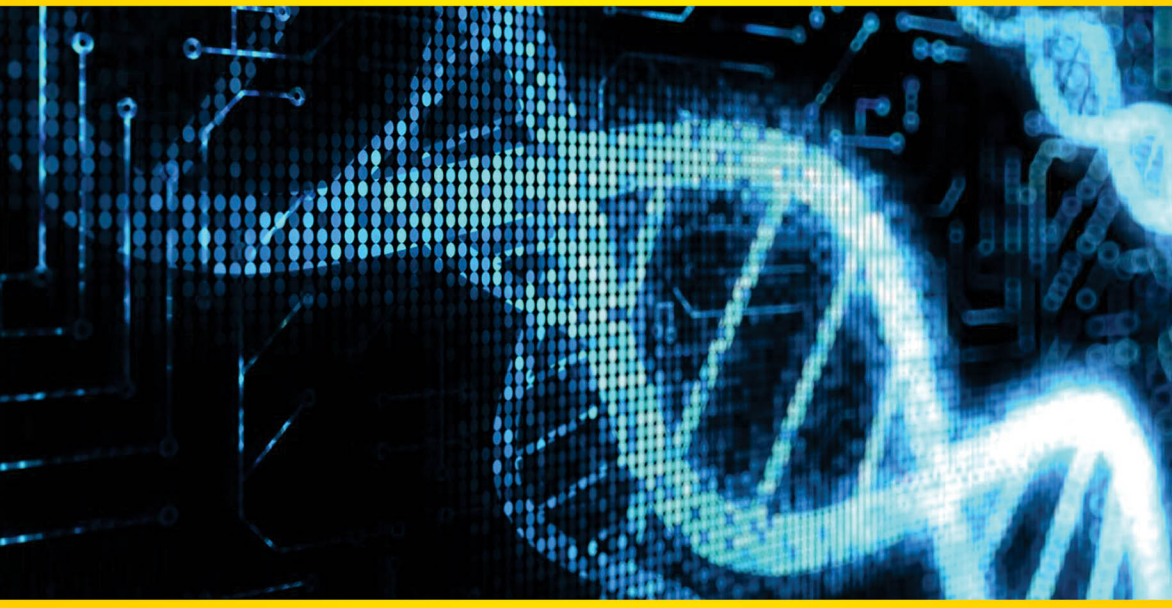
KECERDASAN KOMPUTASIONAL

Konsep dan Aplikasi



Eka Mala Sari Rochman
Aeri Rachmad

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN (D III)
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO
2017

MNC PUBLISHING

KECERDASAN KOMPUTASIONAL
Konsep dan Aplikasi

Eka Mala Sari Rochman, S.Kom., M.Kom
Aeri Rachmad, S.T., M.T

PROGRAM STUDI TEKNIK MULTIMEDIA DAN JARINGAN (DIII)
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO
2017

KECERDASAN KOMPUTASIONAL
Konsep dan Aplikasi

Penulis
Eka Mala Sari Rochman, S.Kom., M.Kom.
Aeri Rachmad, S.T., M.T.

Desain Cover & Penata Isi
Tim MNC Publishing

Cetakan I, Desember 2017

Diterbitkan oleh :
Media Nusa Creative
Anggota IKAPI (162/JTI/2015)
Bukit Cemara Tidar H5 No. 34, Malang
Telp. : 0341 – 563 149 / 0812.3334.0088
E-mail : mnc.publishing.kantor@gmail.com
Website : www.mncpublishing.com

x+108 hlmn ; 15.5x23 cm

ISBN : 978-602-6743-99-2

KATA PENGANTAR

Puji syukur kita panjatkan kehadirat Allah SWT, karena dengan karunia serta rahmatnya kami dapat menyelesaikan buku ajar "Kecerdasan Komputasional" ini. Materi dalam buku ini telah disesuaikan dengan Propram Pembelajaran (PP) dan Rencana Pembelajaran Semester (RPS) Fakultas Teknik. Secara garis besar buku ini, membahas tentang bagaimana menyelesaikan masalah menggunakan kecerdasan komputasional. Serta metode apa saja yang dapat digunakan untuk membantu menyelesaikan permasalahan.

Buku ini dapat dipergunakan oleh mahasiswa jurusan teknik informatika untuk lebih memahami tentang mata kuliah kecerdasan komputasional yang merupakan mata kuliah wajib pada jurusan Teknik Informatika.

Dengan selesainya buku ajar ini, tak lupa kami ucapkan terimakasih kepada Dekan Fakultas Teknik atas semua fasilitas yang telah disediakan demi kelancaran buku ajar ini. Terima kasih rekan-rekan dosen jurusan Teknik Informatika dan Teknik Multimedia Jaringan atas bantuan dan kerjasamanya. Besar harapan kami semoga buku ajar ini dapat bermanfaat, serta demi penyempurnaan buku ajar ini kami mohon saran dan kritik dari para pembaca.

Penulis

DAFTAR ISI

Kata Pengantar ..... **iii**
Daftar Isi ..... **v**
Daftar Gambar ..... **viii**
Daftar Tabel ..... **x**

BAB I Kecerdasan Komputasional **1**
1.1. Tujuan Instruksional Khusus 1
1.2. Materi Perkuliahan 1
A. Definisi Kecerdasan Komputasi 1
B. Metode dalam kecerdasan komputasi 3
1.3. Soal Latihan 4

BAB II Representasi Pengetahuan **5**
2.1. Tujuan Instruksional Khusus 5
2.2. Materi Perkuliahan 5
A. Representasi Pengetahuan 5
B. Logika Proposisi 7
C. Logika Predikat 11
D. Pohon 15
E. Jaringan Semantic 15
F. Frame 16
G. Naskah 17
H. Sistem Produksi 19
2.3. Soal Latihan 19

BAB III Sistem Pakar **21**
3.1. Tujuan Instruksional Khusus 21
3.2. Materi Perkuliahan 21

A. Definisi Sistem Pakar 21
B. Forward Chaining (Sistem Perantaian Maju) 25
C. Backward Chaining (Perantaian Balik) 28
3.3. Soal Latihan 30

BAB IV Logika Fuzzy 31
4.1. Tujuan Instruksional Khusus 31
4.2. Materi Perkuliahan 31
A. Definisi Logika Fuzzy 31
B. Himpunan Fuzzy 33
C. Fungsi Keanggotaan 35
D. Operator Dasar Operasi Himpunan Fuzzy 40
E. Metode Tsukamoto 41
F. Metode Mamdani 41
G. Metode Sugeno 44
4.3. Soal Latihan 44

BAB V Jaringan Saraf Tiruan 47
5.1. Tujuan Instruksional Khusus 47
5.2. Materi Perkuliahan 47
A. Jaringan Saraf Tiruan (JST) 47
B. Komponen Jaringan Saraf Tiruan (JST) 49
C. Arsitektur Jaringan Saraf Tiruan (JST) 50
D. Proses Pembelajaran 57
5.3. Soal Latihan 74

BAB VI Algoritma Genetika 75
6.1. Tujuan Instruksional Khusus 75
6.2. Materi Perkuliahan 75
A. Algoritma Genetika 75
B. Komponen-Komponen Utama Algoritma Genetika 77
C. Operator Genetika 79
D. Proses Algoritma Genetika 80

6.3. Soal Latihan 89

BAB VII Kecerdasan Koloni 91
7.1. Tujuan Instruksional Khusus 91
7.2. Materi Perkuliahan 91
A. Ant Colony (Koloni Semut) 91
B. Bee Colony (Koloni Lebah) 102
7.3. Soal Latihan 106

Daftar Pustaka 107

DAFTAR GAMBAR

Gambar 1. Proses Logika 6
Gambar 2. Resolusi pada Logika Proposisi dengan Pernyataan lengkap 11
Gambar 3. Struktur Pohon 15
Gambar 4. Contoh Jaringan Semantic 16
Gambar 5. *Frame* 17
Gambar 6. Contoh Pemetaan Input Output 32
Gambar 7. Himpunan Muda, Parobaya, Tua 34
Gambar 8. Representasi linear naik 35
Gambar 9. Representasi linear turun 36
Gambar 10. Kurva segitiga 36
Gambar 11. Kurva trapesium 37
Gambar 12. Kurva-S 38
Gambar 13. Kurva π 38
Gambar 14. Kurva β 39
Gambar 15. Kurva β 40
Gambar 16. Struktur neuron JST 49
Gambar 17. Jaringan syaraf dengan lapisan tunggal 51
Gambar 18. Jaringan syaraf dengan lapis banyak 52
Gambar 19. Jaringan syaraf dengan lapisan kompetitif 52
Gambar 20. Fungsi Aktivasi Undak Biner (Hard Limit) 53
Gambar 21. Fungsi Aktivasi Undak Biner (Threshold) 53
Gambar 22. Fungsi Aktivasi Bipolar (Symetric hard limit) 54
Gambar 23. Fungsi Aktivasi Bipolar (Threshold) 54
Gambar 24. Fungsi Aktivasi Linear (Identitas) 55
Gambar 25. Fungsi Aktivasi Saturating Linear 55
Gambar 26. Fungsi Aktivasi Symetric Saturating Linear 56
Gambar 27. Arsitektur jaringan Hebb 58

Gambar 28. Pembatasan linear dengan perceptron 59
Gambar 29. Arsitektur jaringan Backpropagation..... 62
Gambar 30. Arsitektur jaringan BAM 65
Gambar 31. Arsitektur *ELM* [1]..... 68
Gambar 32. SVM Menemukan *Hyperplane* terbaik yang meisahkan kedua kelas -1 dan +1 73
Gambar 33. Hyperplane terbentuk diantara class-1 dan +1 [6] 73
Gambar 34. Diagram alir algoritma genetika sederhana 82
Gambar 35. Contoh Kasus..... 97

DAFTAR TABEL

Tabel 1. Tabel Kebenaran 7
Tabel 2. Tabel Kebenaran Operator NOT 8
Tabel 3. Tabel Kebenaran Operator AND 8
Tabel 4. Tabel Kebenaran Operator OR 8
Tabel 5. Tabel Kebenaran Implikasi 8
Tabel 6. Tabel Kebenaran Ekuivalensi 9
Tabel 7. Jarak Antar kota d_{ij} 98
Tabel 8. Visibilitas antar kota 98
Tabel 9. Probabilitas kota untuk dikunjungi siklus ke-1 kasus 1 99
Tabel 10. Panjang jalur semut siklus ke-1 kasus 1 99
Tabel 11. Perubahan harga intensitas jejak kaki semut τ ij antarkota siklus ke-2 kasus 1 100
Tabel 12. Probabilitas kota untuk dikunjungi siklus ke-2 kasus 1 dengan τ ij telah diperbaharui 100
Tabel 13. Panjang jalur semut siklus ke-2 kasus 1 100
Tabel 14. Perubahan harga intensitas jejak kaki semut τ ij antarkota siklus ke-3 kasus 1 101
Tabel 15. Probabilitas kota untuk dikunjungi siklus ke-3 kasus 1 dengan τij telah diperbaharui 101
Tabel 16. Panjang jalur semut siklus ke-3 kasus 1 102
Tabel 17. Penentuan mengikuti wagle dance 105

BAB I.

KECERDASAN KOMPUTASIONAL

1.1. Tujuan Instruksional Khusus

- Mahasiswa mampu memahami karakteristik dan teknik pembelajaran berbagai tipe metode kecerdasan komputasional
- Mahasiswa mampu menerapkan beberapa metode Kecerdasan Komputasional dalam menyelesaikan sebuah permasalahan

1.2. Materi Perkuliahan

A. Definisi Kecerdasan Komputasi

Apakah Kecerdasan buatan itu? Kecerdasan buatan adalah pandangan manusia yang dianggap cerdas mengenai sebuah penelitian ataupun intruksi yang berkaitan dengan pemrograman komputer. Hal tersebut dikaitkan oleh kemampuan manusia yaitu semakin banyak bekal pengetahuan manusia maka akan lebih baik seseorang untuk menyelesaikan sebuah permasalahan. Manusia diberi akal sebagai penalaran untuk mengambil kesimpulan dari permasalahan-permasalahan yang ada, namun hal tersebut tidak dapat dijadikan sebagai bekal tanpa adanya ilmu pengetahuan. Sehingga manusia pandai karena memiliki pengetahuan dan pengalaman.

Kecerdasan buatan atau Artificial Intllegent adalah salah satu bagian ilmu dimana sebuah mesin (komputer) dapat melakukan apa yang dapat dikerjakan oleh manusia. Saat ini perkembangan teknologi semakin pesat, jadi kegunaan komputer tidak bukan sebuah alat untuk menghitung akan tetapi dapat membantu manusia

untuk menyelesaikan pekerjaannya dengan lebih baik dan cepat. Kecerdasan buatan dapat dimanfaatkan dalam berbagai bidang keilmuan seperti kesehatan, pertanian, kebutuhan rumah tangga dll.

Untuk menyelesaikan sebuah permasalahan sistem yang dibangun sarus mempertimbangkan beberapa hal, seperti mendefinisikan masalah, menganalisa, memperesentaikan pengetahuan serta memilih teknik yang benar agar hasil yang dicapai sesuai dengan yang diinginkan.

Kecerdasan Buatan terbagi ke dalam dua faham pemikiran yaitu Kecerdasan buatan Konvensional dan Kecerdasan Komputasional (CI, Computational Intelligence). Kecerdasan buatan yang konvensional lebih banyak memperlibatkan metode pembelajaran mesin yang telah diklasifikasikan.

Kecerdasan Komputasional dapat menyelesaikan permasalahan yang non-*algoritmizable* karena merupakan salah satu disiplin ilmu komputer. Kecerdasan Buatan merupakan bagian dari Kecerdasan Kompuasional yang lebih memfokuskan pada permasalahan dengan kognitif yang lebih tinggi. Sedangkan untuk permasalahan dengan kognitif yang lebih rendah ataupun yang berkaitan erat dengan persepsi maupun kontrol merupakan ranah Kecerdasan Komputasional.

Kecerdasan Komputasional (CI) merupakan pendekatan yang mana menerapkan cara berpikir manusia dalam sebuah ketidakpastian dengan berpikir dan belajar. Tujuan dari kecerdasan komputasional adalah membantu menyelesaikan permasalahan manusia. Dengan harapan sistem cerdas ini mempunyai kemampuan manusia dalam hal tertentu, serta dapat mengkondisikan pada saat terjadi perubahan pada lingkungan sehingga dapat menjelaskan hasil yang telah diperoleh.

Kecerdasan Komputasional terdiri dari tiga paradigma komputasi, yaitu: *Neural Network System (NN)*, *Fuzzy Logic (FL)*, *Probabilistic Reasoning* yang terdiri dari teori *Chaos*, *Belief Networks*, *Genetic Algorithm (GA)*. Semua metode ini telah dirumuskan sebelum

dimunculkannya paradigma tentang kecerdasan komputasi. Untuk perkembangan mengenai FL sudah dimulai sejak tahun 1965. Sejak tahun 1940-an konsep dasar mengenai *Neural Network System* sudah digali. John Holland pada pertengahan tahun 1970-an telah mencetuskan tentang adanya *Probabilistic Reasoning* juga serta dasar-dasar *Genetic Algorithm*. Sehingga Zadeh menyebut Kecedasan Komputasional merupakan reinkarnasi dan pembaharuan dari metode-metode tersebut.

B. Metode dalam kecerdasan komputasi

Ranah penelitian atau kajian dalam Kecerdasan Komputasional meliputi beberapa penelitian ini, antara lain :

- *Fuzzy System*
- *Pattern RecognitioImage*
- *Immage Processing*
- *Machine Learning*
- *Computer Vision*
- *Soft Computing*
- *Network Knowledge*
- *Image retrieval*
- *Decision Support System*
- *Expert System*
- *Robotic Graph*.

Metode-metode ini mempunyai kelebihan tersendiri yang mana jika diintegrasikan akan membentuk inti dari Kecedasan Komputasional. Melalui proses penggabungan pengetahuan manusia yang secara efektif, serta dapat menghasilkan atau mempelajari dengan sangat baik utamanya untuk keakurtan data serta hal ketidakpastian, kemudian pada lingkungan yang tidak dikenali atau tidak diketahui sebelumnya seperti sebuah prediksi kecerdasan komputasi dapat belajar untuk menciptkan kedinamisan data dan hasil.

Untuk menghadapi masalah komputasi dunia maka Kecedasan Komputasional bukan merupakan metode tunggal karena menggunakan proses penggabungan dari metode yang lain untuk mendapatkan hasil yang optimal. Penekanan pada *partnership* dengan hasil saling menguntungkan dari berbagai metode yang ada merupakan hal yang ditekankan pada kecerdasan komputasi, sehingga hal ini disebut sebagai Kecedasan Komputasional Hibrid.

Keunggulan dari kerjasama metode-metode itu menkankan keunggulan salah satu individual. Kekurangan dari satu metode tidak nampak karena telah ditutupi oleh kelebihan dari metode lainnya.. Setiap metode memiliki segi positif yang dapat digunakan, sehingga menutupi kekurangan dari metode yang lain. Metode *Fuzzy Logic*, *Neural Network System* dan *Probabilistic Reasoning* saling mendukung akan menguntungkan jika menggunakan kombinasi dari ketiganya dari pada digunakan secara individual.

Fuzzy-Genetic, *Neuro-Genetic*, dan sistem *Neuro-Fuzzy-Genetic* adalah contoh kombinasi tersebut. Kombinasi pada sistem *Neuro-Genetic*, untuk mengoptimalkan sebuah struktur dari *Neural Network System* menggunakan *Genetic Algorithm (GA)*. Jumlah dan variasi dari aplikasi *Fuzzy Logic* dan *Neural Network System* dalam beberapa tahun terakhir telah tumbuh cepat, seperti halnya pada sistem pendukung keputusan dan pasar finansial. Yang memiliki tujuan untuk menyelesaikan permasalahan pada kondisi nyata dengan mengambil keputusan, pemodelan, dan kontrol.

1.3. Soal Latihan

1. Jelaskan apa yang dimaksud kecerdasan komputasi!
2. sebutkan ranah penelitian kecerdasan komputasi!
3. Bagaimanakah peranan kecerdasan komputasional dalam kehidupan?

BAB II

REPRESENTASI PENGETAHUAN

2.1. Tujuan Instruksional Khusus

- Mahasiswa mampu mengetahui perlunya Representasi Pengetahuan dalam sebuah sistem Kecerdasan Komputasional
- Mahasiswa mampu mengetahui cara Representasi Pengetahuan sederhana

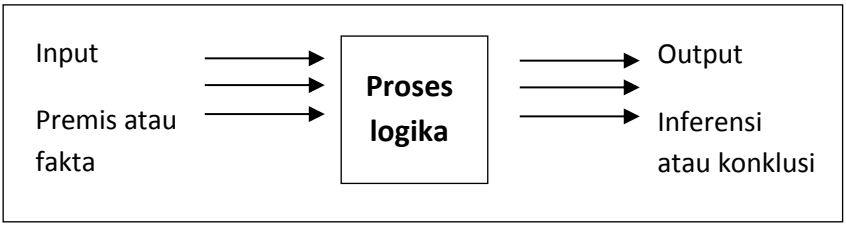
2.2. Materi Perkuliahan

A. Representasi Pengetahuan

Representasi pengetahuan adalah bagaimana mengorganisasi pengetahuan yang didapat dengan mengkodekan dan menyimpan dalam basis pengetahuan. Pada kecerdasan buatan, basis pengetahuan dan kemampuan untuk penalaran merupakan hal yang penting.

Logika merupakan salah satu bentuk (bahasa) representasi yang paling tua dan sederhana. Proses membentuk kesimpulan atau menarik inferensi berdasar fakta yang ada merupakan proses logika. Input dari proses logika adalah premis atau fakta yang diakui kebenarannya sehingga dapat dibentuk sebuah kesimpulan yang benar.

Aturan aljabar Boolean dapat digunakan untuk menggambarkan dan memanipulasi beberapa fakta pada proses logika.



Gambar 1. Proses Logika

Ada 2 penalaran yang dapat dilakukan untuk memperoleh konklusi :

1. *Penalaran deduktif,*

   Deduksi merupakan: penjelasan dari fakta yang sudah diketahui sebelumnya menjadi fakta yang tidak diketahui, dari hal-hal umum menuju ke hal hal spesifik, dari premis menuju ke kesimpulan logis.

   Contoh :

   Premis mayor : jika ban motor saya bocor, saya tidak akan berangkat kuliah
   Premis minor : Ban motor saya bocor
   Konklusi : hari ini saya tidak akan berangkat kuliah

2. *Penalaran induktif,*

   Induksi merupakan proses penjelasan dari fakta tertentu dan khusus yang nantinya menjadi sebuah kesimpulan secara umum.

   Contoh :

   Premis-1 : berhitung adalah pelajaran yang sulit
   Premis-2 : fisika adalah pelajaran yang sulit
   Premis-3 : kimia adalah pelajaran yang sulit
   Konklusi : matematika adalah pelajaran yang sulit

   Pada penalaran ini, muncul premis baru yang menyebabkan runtuhnya kesimpulan yang sudah dihasilkan. Seperti :

   Premis-4 : biologiQ adalah pelajaran yang sulit

; Karena biologi bukan merupakan bagian dari matematika. Jika kita menggunakan penalaran induktif, bisa saja mengakibatkan adanya *ketidakpastian*.

B. Logika Proposisi

Propositional connective digunakan pada propositional logic untuk membentuk sebuah statement sederhana atau statement yang kompleks, yang mana mekanismenya dapat menentukan kebenaran dari sebuah statement kompleks dari nilai kebenaran yang direpresentasikan oleh statement lain yang lebih sederhana

Proposisi adalah pernyataan yang dapat bernilai benar (B) atau salah (S). Operator logika menggabungkan proposisi yang lebih dari satu :

a. Konjungsi : ∧ (and)
b. Disjungsi : ∨ (or)
c. Negasi : ¬ (not)
d. Implikasi : → (if- then)
e. Ekuivalensi : ↔

p	q	$\sim p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$	$p \leftrightarrow q$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

Tabel 1. Tabel Kebenaran

a. Operator NOT

Digunakan untuk memberikan nilai negasi (lawan) dari pernyataan yang telah ada.

P	NOT (P)
B	S
S	B

Tabel 2. Tabel Kebenaran Operator NOT

b. Operator AND

Digunakan untuk mengkombinasikan 2 proposisi. Hasilnya bernilai benar jika kedua proposisi bernilai benar. Dan salah jika nilai kedua proposisi salah serta salah satu proposisinya bernilai salah.

P	Q	AND (P)
B	B	B
B	S	S
S	B	S
S	S	S

Tabel 3. Tabel Kebenaran Operator AND

c. Operator OR

Digunakan untuk mengkombinasikan 2 proposisi. Hasilnya akan bernilai benar bila salah satu bernilai benar dan salah jika kedua proposisi bernilai salah.

P	Q	AND (P)
B	B	B
B	S	B
S	B	B
S	S	S

Tabel 4. Tabel Kebenaran Operator OR

d. Implikasi

Akan bernilai **salah** jika P benar dan Q salah. Selain itu semuanya bernilai benar.

P	Q	AND (P)
B	B	B
B	S	S
S	B	B
S	S	B

Tabel 5. Tabel Kebenaran Implikasi

e. Ekuivalensi

Akan menghasilkan nilai **benar** jika P dan Q benar atau keduanya salah.

P	Q	AND (P)
B	B	B
B	S	S
S	B	S
S	S	B

Tabel 6. Tabel Kebenaran Ekuivalensi

Resolusi digunakan untuk melakukan inferensi pada logika proposisi. ***Resolusi*** merupakan aturan untuk melakukan inferensi yang berjalan secara ekektif dan efisien dalam bentuk khusus. Bentuk ini dikenal dengan *Conjunctive Normal Form (CNF)*.

Yang ciri-cirinya sebagai berikut :

1. Setiap kalimat merupakan disjungsi literal
2. Semua kalimat terkonjungsi secara implicit

Untuk merubah kalimat ke bentuk CNF, ikuti langkah-langkah ini:

- Hilangkan implikasi dan ekuivalensi
  - $x \rightarrow y$ menjadi $\neg x \vee y$
  - $x \leftrightarrow y$ menjadi $(\neg x \vee y) \cap (\neg y \vee x)$
- Kurangi lingkup semua negasi menjadi satu saja
  - $\neg (\neg x)$ menjadi x
  - $\neg (x \vee y)$ menjadi $(\neg x \wedge \neg y)$
  - $\neg (x \wedge y)$ menjadi $(\neg x \vee \neg y)$
- Gunakan aturan assosiatif dan distributive untuk mengkonversi menjadi konjungsi dan disjungsi
  - Assosiatif : $(A \vee B) \vee C = A \vee (B \vee C)$
  - Distributif : $(A \wedge B) \vee C = (A \vee C) \wedge (B \vee C)$

  Bentuk satu kalimat yang terpisah untuk setiap konjungsi

Contoh :

Diketahui basis pengetahuan (fakta-fakta yang bernilai benar) sebagai berikut :

1. P
2. (P ∧ Q) → R
3. (S ∨ T) → Q
4. T

Maka tentukan kebenaran R.

Untuk membuktikannya dengan menggunakan resolusi, maka rubah kalimat menjadi bentuk CNF. Lalu tambahkan kontradiksi kedalam tujuannya, R menjadi ¬ R. Sehingga fakta-faktanya dapat disusun menjadi :

1. P
2. ¬ P ∨ ¬ Q ∨ R
3. ¬ S ∨ Q
4. ¬ T ∨ Q
5. T
6. ¬ R

Sehingga resolusi dapat dilakukan untuk membuktikan R.

Jika diterapkan pada kalimat :

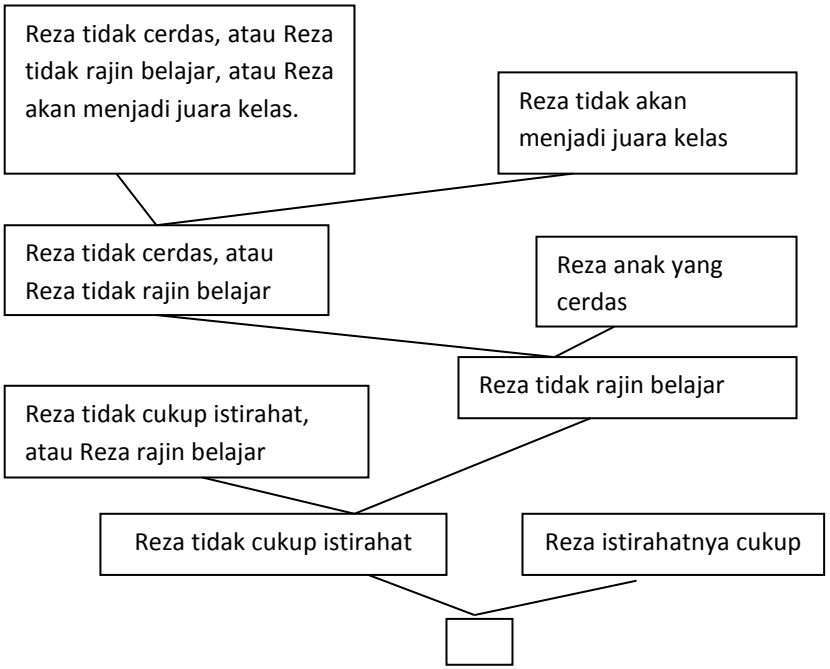
- P : Reza anak yang cerdas
- Q: Reza rajin belajar
- S : Reza akan menjadi juara kelas
- T : Reza makannya banyak
- T : Reza istirahatnya cukup

Sehingga kalimat yang terbentuk adalah :

- ✓ Reza anak yang cerdas
- ✓ Jika Reza anak yang cerdas dan Reza rajin belajar, maka Reza akan menjadi juara kelas
- ✓ Jika Reza makannya banyak atau Reza istirahatnya cukup, maka Reza rajin belajar
- ✓ Reza istirahatnya cukup

Setelah dikonversi ke bentuk CNF, akan diperoleh :

- Fakta ke-2 : Reza tidak cerdas atau Reza tidak rajin belajar atau Reza akan menjadi juara kelas
- Fakta ke-3 : Reza tidak makan banyak atau Reza rajin belajar
- Fakta ke-4 : Reza tidak cukup istirahat atau Reza rajin Belajar



Gambar 2. Resolusi pada Logika Proposisi dengan Pernyataan lengkap

C. Logika Predikat

Hal-hal yang tidak dapat direpresentasikan dengan menggunakan logika proposisi maka menggunakan logika predikat. Karena logika predikat memberi tambahan kemampuan dalam merepresentasikan pengetahuan dengan lebih cermat dan terperinci.

Penggertian kalkulus pada logika predikat berbeda dengan istilah kalkulus dalam matematika. ARGUMEN (atau objek) dan PREDIKAT (keterangan) merupakan dua buah bagian dari ebuah proposisi atau premis. Predikat adalah keterangan yang membuat argumen dan predikat. Predikat dapat berupa kata kerja atau bagian kata kerja pada satu kalimat. Kelebihan logika predikat lebih baik daripada menggunakan proportional logic kalkulus untuk kalimat yang lebih kompleks sehingga direpresentasikan lebih baik.

Misalkan diketahui fakta-fakta berikut :

- *Ari adalah seorang pria* : *A*
- *Firman adalah seorang pria* : *B*
- *Andy adalah seorang pria* : *C*
- *Farid adalah seorang pria* : *D*
- *Ridho adalah seorang pria* : *E*

Apabila kelima fakta-fakta diatas dinyatakan dengan proposisi, akan terjadi pemborosan karena sekalipun berada dalam proposisi yang berbeda namun pernyataannya sama pada predikat.

Dari contoh lima fakta yang ada dapat dituliskan :

pria (x)

x adalah vareabel yang didistribusikan dengan Ari, Firman, Andy, Farid dan Ridho serta pria yang lain.

ARGUMEN (atau objek) dan PREDIKAT (keterangan) merupakan dua buah bagian dari ebuah proposisi atau premis. Predikat adalah keterangan yang membuat argumen dan predikat..

Misalnya :

1. *Baju berwarna merah*

Dapat dinyatakan dalam bentuk logika predikat :

Berwarna (baju, merah)

Berwarna = predikat (keterangan)

Baju = argumen (objek)

Merah = argumen (objek)

2. *Mahasiswa berada di dalam kelas*

Dapat dinyatakan dalam bentuk logika predikat :

Didalam (mahasiswa, kelas)

Didalam = predikat (keterangan)

Mahasiswa = argumen (objek)

Kelas = argumen (objek)

3. *Johan suka Maria*

Dapat dinyatakan dalam bentuk logika predikat :

Suka (Johan, Maria)

4. *Johan suka Maria*

Ramon suka Maria

Misal : Johan = x, Maria = y, Ramon = z

Maka : suka (x,y) ∧ suka (z,y) → tidak suka (x,z)

Dibaca : Jika Johan suka Maria dan Ramon suka Maria maka Johan tidak suka Ramon

5. Ada beberapa pernyataan sebagai berikut :
   a. *Alfan adalah seorang mahasiswa*
   b. *Alfan masuk jurusan informatika*
   c. *Setiap mahasiswa informatika pasti mahasiswa teknik*
   d. *Kalkulus adalah matakuliah yang sulit*
   e. *Setiap mahasiswa teknik pasti akan suka kalkulus atau akan membencinya*
   f. *Setiap mahasiswa pasti akan suka terhadap suatu matakuliah*
   g. *Mahasiswa yang tidak pernah hadir pada matakuliah sulit, maka mereka pasti tidak suka terhadap matakuliah tersebut.*
   h. *Alfan tidak pernah hadir kuliah matakuliah kalkulus*

Kedelapan pernyataan tersebut dirubah ke bentuk logika predikat dengan menggunakan operator-operator : implikasi (→), NOT (¬), AND (∧), OR (∨), untuk setiap (∀), terdapat (∃). Sebagai berikut :

a. *Mahasiswa (Alfan)*
b. *Informatika (Alfan)*
c. *∀x : Informatika (x) → teknik (x)*
d. *Sulit (kalkulus)*
e. *∀x : Teknik (x) → suka (x, kalkulus) ∨ benci (x, kalkulus)*
f. *∀x : ∃y : suka (x,y)*
g. *∀x : ∃y : mahasiswa (x) ∧ sulit (y) ∧ ¬ hadir (x,y) → ¬ suka (x,y)*
h. *¬ hadir (Alfan, kalkulus)*

Andaikan menjawab pertanyaan ini :
"Apakah Alfan suka matakuliah kalkulus?"

Maka dari pernyataan ke-7 (**g)** kita buktikan Alfan tidak suka matakuliah kalkulus. Dengan menggunakan penalaran backward.

¬ suka (Alfan, kalkulus)

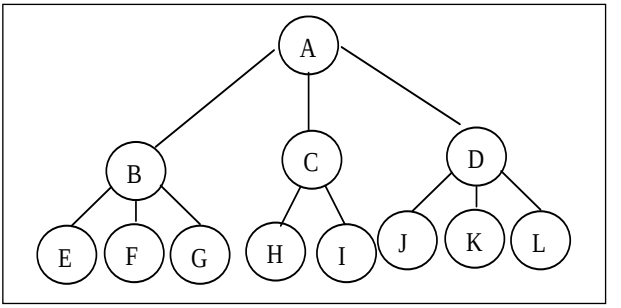
Sebagai berikut :

¬ suka (Alfan, kalkulus)
↑ (g, substitusi)
Mahasiswa (Alfan) ∧
Sulit (kalkulus) ∧
¬ hadir (Alfan, kalkulus)
↑ (a)
Sulit (kalkulus) ∧
¬ hadir (Alfan, kalkulus)
↑
¬ hadir (Alfan, kalkulus)
↑ (h)
□

Alfan tidak suka matakuliah kalkulus berdasarkan penalaran yang telah dijabarkan.

D. Pohon

Struktur grafik yang hirarki serta sebuah cara yang sederhana untuk menggambarkan hirarki dan list dari pengetahuan lainnya adalah pengertian dari Struktur pohon. Strukturnya terdiri dari node-node yang menunjukkan objek dan arc (busur) yang menghubungkan antar node.



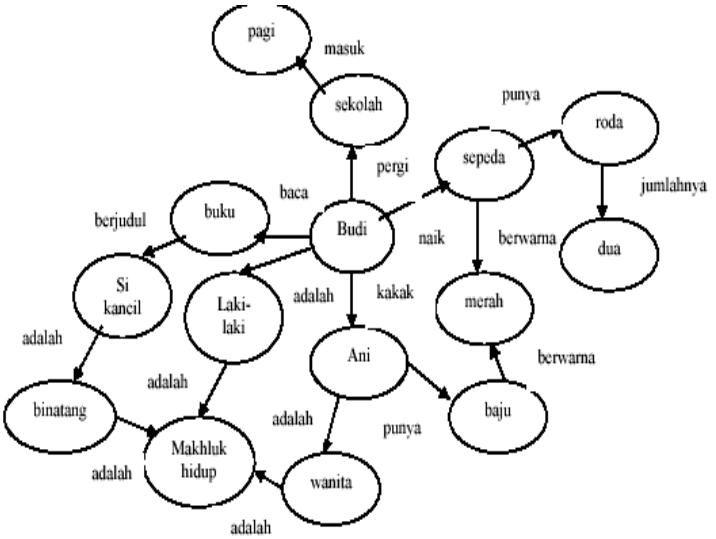
Gambar 3. Struktur Pohon

E. Jaringan Semantic

Jaringan semantic adalah salah satu bentuk representasi knowledge-base dalam bentuk diagram. Diagram tersebut terdiri atas *node* dan *arc*. *Node* merepresentasikan konsep, sedangkan *arc* merepresentasikan sebuah relasi.

Jaringan semantik adalah gambaran dari pengetahuan grafis yang menunjukkan hubungan antar objek yang terdiri dari lingkaran-lingkaran yang menunjukkan objek yang menginformasikan tentang objek tersebut. Objek dapat berupa peristiwa atau benda.

Sistem jaringan semantic selalu tergantung pada masalah yang akan dipecahkan. Jika masalah bersifat umum, maka membutuhkan sedikit rincian. Tetapi jika masalah melibatkan hal yang lain maka membutuhkan penjelasan yang lebih terperinci.



Gambar 4. Contoh Jaringan Semantic

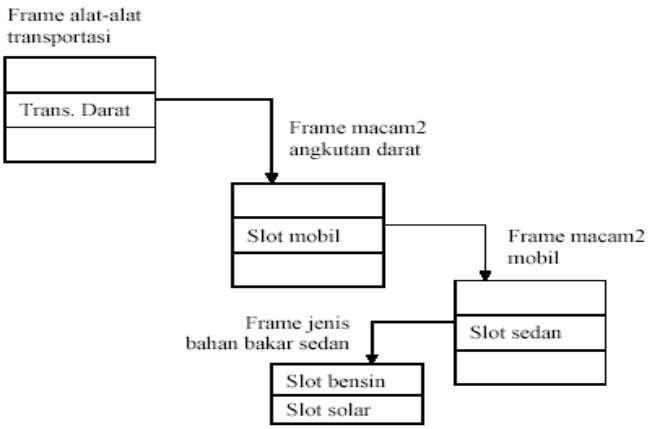
F. Frame

Frame merupakan kumpulan pengetahuan tentang suatu objek tertentu seperti peristiwa, lokasi, situasi dll. Dapat juga didefinisikan sebagai struktur data yang mengandung semua informasi pengetahuan yang relevan dari suatu objek. Pengetahuan tersebut diorganissai dalam struktur hirarkis khusus yang memungkinkan proses pengetahuan. Frame ini merupakan aplikasi dari pemrograman berorientasi objek dalam AI dan sistem pakar.

Pengetahuan dalam frame dibagi-bagi ke dalam slot atau atribut yang mendeskripsikan rincian dan karakteristik objek. Dengan frame, sangat mudah untuk melakukan inferensi tentang objek, peristiwa baru karena sudah tersedia basis pengetahuan yang ditarik dari pengalaman.

Pada gambar dibawah ini menunjukkan frame alat-alat transportasi. Frame tersebut mempunyai 3 slot yaitu alat transportasi udara, darat dan laut. Ada slot yang bernilai tetap dan bernilai tidak tetap. Slot yang bernilai tetap adalah jumlah roda pada sedan. Jenis

slot lainnya berisat procedural (slot yang memungkinkan penambahan informasi baru yang bisa ditambahkan pada aturan IF). Seperti informasi tentang kecepatan perjalanan, pengisian tangki bahan bakar atau pemakaian bahan bakar setiap km.



Gambar 5. *Frame*

G. Naskah

Naskah (*script*) adalah skema yang merepresentasikan pengetahuan berdasar karakteristikyang sudah dikenal sebagai pengalaman-pengalaman. Jadi naskah hampir sama dengan frame, hanya saja frame menggambarkan objek sedangkan naskah menggambarkan peristiwa.

Elemen-elemen pada naskah meliputi:

1) Kondisi input, kondisi sebelum peristiwa terjadi
2) Track, variasi pada naskah
3) Prop, selama peristiwa berlangsung pada objek pendukung
4) Role, dalam sebuah peristiwa terdapat peran yang dimainkan seseorang
5) Scene, dalam sebuah peristiwa terdapat adegan yang dimainkan
6) Hasil, kondisi setelah peristiwa naskah terjadi

Contoh : Naskah kejadian pada ujian akhir

- *Jalur (track)* : *ujian tertulis matakuliah kecerdasan Buatan*
- *Role (peran)* : *mahasiswa, pengawas*
- *Prop (pendukung)* : *lembar soal, lembar jawab, presensi, pena, dll.*
- *Kondisi input* : *mahasiswa terdaftar untuk mengikuti ujian.*

Adegan (scene) – 1 : persiapan pengawas

- *Pengawas menyiapkan lembar soal*
- *Pengawas menyiapkan lembar jawab*
- *Pengawas menyiapkan lembar presensi*

Adegan (scene) – 2 : mahasiswa masuk ruangan

- *Pengawas mempersilahkan mahasiswa masuk*
- *Pengawas membagikan lembar soal*
- *Pengawas membagikan lembar jawab*
- *Pengawas memimpin doa*

Adegan (scene) – 3 : mahasiswa mengerjakan soal ujian

- *Mahasiswa menulis identitas di lembar jawab*
- *Mahasiswa menandatangani lembar jawab*
- *Mahasiswa mengerjakan soall*
- *Mahasiswa mengecek jawaban*

Adegan (scene) – 4 : mahasiswa telah selesai ujian

- *Pengawas mempersilahkan mahasiswa keluar ruanagn*
- *Mahasiswa mengumpulkan kembali lembar jawab*
- *Mahasiswa keluar ruangan*

Adegan (scene) – 5 : pengawas mengemasi lembar jawab

- *Pengawas mengurutkan lembar jawab*
- *Pengawas mengecek lembar jawab dan presensi*
- *Pengawas meninggalkan ruangan*

Hasil :

- *Mahasiswa merasa senang dan lega*
- *Mahasiswa merasa kecewa*
- *Mahasiswa pusing*
- *Mahasiswa memaki-maki*
- *Mahasiswa sangat bersyukur*

H. Sistem Produksi

Representasi pengetahuan dengan sistem produksi, pada dasarnya berupa aplikasi aturan yang terdiri dari :

- ✓ Antecedent, bagian yang mengekspresikan situasi (IF)
- ✓ Konsekuen, menyatakan tindakan tertentu jika peristiwa bernilai benar (THEN)

Terdapat 2 metode penalaran yang digunakan jika pengetahuan direpresentasikan dengan aturan metode tersebut antara lain ***Forward Reasoning (penalaran maju)*** dan ***Backward Reasoning (penalaran mundur).*** Sedangkan komponen-komponen dalam sebuah sistem produksi antara lain :

- ✓ Ruang keadaan, berisi keadaan awal, tujuan serta aturan yang dipakai untuk mencapai tujuan
- ✓ Strategi kontrol, berfungsi mengarahkan proses pencarian dan mengontrol arah eksplorasi

2.3. Soal Latihan

Terdapat beberapa pernyataan, seperti dibawah ini :

1. Reza adalah seorang mahasiswa
2. Reza masuk Jurusan elektro
3. Setiap mahasiswa elektro pasti mahasiswa teknik
4. Aljabar linear adalah matakuliah yang sulit
5. Setiap mahasiswa teknik pasti akan suka aljabar linear atau akan membencinya
6. Setiap mahasiswa pasti akan suka terhadap suatu matakuliah

7. Mahasiswa yang tidak pernah hadir pada matakuliah sulit, mereka pasti tidak suka terhadap matakuliah tersebut
8. Reza tidak pernah hadir pada matakuliah aljabar linear

Coba masukkan kedelapan pernyataan diatas ke pernyataan predikat aljabar linear dengan menggunakan operator-operator logika. Serta buktikan pernyataan ini :
"Apakah Reza suka matakuliah aljabar linear?"

BAB III.

SISTEM PAKAR

3.1. Tujuan Instruksional Khusus

- Mahasiswa mampu mengetahui definisi sistem pakar
- Mahasiswa mampu mengetahui metode penelusuran Forward dan Backward Chaining

3.2. Materi Perkuliahan

A. Definisi Sistem Pakar

Ketika hendak membuat suatu keputusan yang komplek atau memecahkan masalah, seringkali kita meminta nasehat atau berkonsultasi dengan seorang pakar atau ahli. Seorang **pakar** adalah seseorang yang mempunyai pengetahuan dan pengalaman spesifik dalam suatu bidang; misalnya pakar komputer, pakar uji tak merusak, pakar politik dan lain-lain. Semakin tidak terstruktur situasinya, semakin mengkhusus (dan mahal) konsultasi yang dibutuhkan.

Sistem Pakar (*Expert System*) adalah usaha untuk menirukan seorang pakar. Biasanya Sistem Pakar berupa perangkat lunak pengambil keputusan yang mampu mencapai tingkat performa yang sebanding dengan seorang pakar dalam bidang mengatasi sebuah permasalahan. Ide dasarnya adalah: kepakaran ditransfer dari seorang pakar (atau sumber kepakaran yang lain) ke komputer, pengetahuan yang ada disimpan dalam komputer, dan pengguna dapat berkonsultasi pada komputer untuk suatu nasehat, kemudian komputer dapat menyimpulkan, mendeduksi, dll. seperti layaknya

seorang pakar. Hasilnya dijelaskan kepada pengguna sistem disertai alasan-alasannya.

Kepakaran (*expertise*) adalah sebuah pengetahuan yang didapat melalui rangkaian membaca, pelatihan dan pengalaman secara meluas (esktensif) dan spesifik. Sehingga dengan adanya pengetahuan dapat membuat pakar mengambil keputusan dengan lebih baik dan lebih cepat memecahkan problem yang kompleks daripada non-pakar. Sifat yang dimiliki epakaran adalah sifat berjenjang, yang mana pakar top memiliki pengetahuan lebih dibandingkan pakar yunior.

Mentransfer kepakaran dari seorang pakar ke komputer, kemudian ke orang lain (yang bukan pakar) hal ini merupakan Tujuan dari sebuah sistem pakar. Yang mana tercakup dalam sebuah rekayasa pengetahuan (*knowledge engineering*).

Manfaat Sistem Pakar

Sistem Pakar menjadi populer disebabkan oleh banyaknya kemampuan dan manfaat yang diberikan oleh Sistem Pakar, anatara lain :

a. Sistem Pakar dapat bekerja lebih cepat dari manusia sehingga dapat Meningkatkan output dan produktivitas.
b. Memberi nasehat yang konsisten dan mengurangi kesalahan dengan tujuan meningkatkan kualitas
c. Dapat membaca kepakaran yang sangat terbatas.
d. Pada lingkungan yang berbahaya dapat beroperasi.
e. Memudahkan akses ke pengetahuan.
f. Handal. Sistem Pakar tidak bosan, tidak mengalami kelelahan atau sakit, serta konsisten melihat semua detil dan tidak melewatkan informasi yang relevan berserta solusinya yang potensial.
g. Meningkatkan kapabilitas sistem terkomputerisasi yaitu Integrasi Sistem Pakar dengan sistem komputer lain menjadikan lebih efektif, dan lebih banyak mencakup aplikasi yang lain.

h. Walaupun dengan informasi yang tidak lengkap atau tidak pasti sistem pakar dapat bekerja. Pengguna dapat merespon dengan: "tidak tahu" atau "tidak yakin" pada satu atau lebih pertanyaan selama konsultasi, dan Sistem Pakar tetap akan memberikan jawabannya.
i. Terdapat pelatihan sehingga para pengguna pemula yang bekerja dengan Sistem Pakar dapat lebih berpengalaman. Karena fasilitas penjelas berfungsi seperti guru.
j. Meningkatkan dalam problem solving, karena mengambil sumber pengetahuan dari banyak pakar.
k. Perangkat mahal ditiadakan.
l. Fleksibel.

Keterbatasan Sistem Pakar

Sistem pakar memiliki keterbatasan, karena metodologi Sistem Pakar tidak selalu mudah, sederhana atau bahkan efektif. Berikut ini beberapa keterbatasan yang menghambat perkembangan pada Sistem Pakar :

a. Pengetahuan tidak selalu tersedia.
b. Kepakaran sulit diekstrak dari manusia.
c. Pendekatan oleh setiap pakar untuk sebuah masalah bisa berbeda, walaupun sama-sama benar.
d. Bagi seorang pakar sangat sulit untuk menjelaskan langkah menangani masalah
e. Pengguna mempunyai batas kognitif alami, sehingga bisa jadi tidak dapat memanfaatkan sistem secara maksimal.
f. Sistem Pakar bekerja baik pada bidang yang sempit.
g. Banyak pakar yang tidak memiliki jalan untuk mengecek apakah kesimpulan yang diambil benar dan masuk akal.
h. Istilah yang digunakan pakar dalam mengekspresikan fakta seringkali terbatas dan tidak mudah dimengerti.
i. Pengembangan Sistem Pakar seringkali membutuhkan perekayasa pengetahuan (knowledge engineer) yang langka dan mahal.

j. Kurangnya rasa percaya pengguna menghalangi pemakaian Sistem Pakar.
k. Transfer pengetahuan dapat bersifat subyektif dan bias.

Komponen Sistem Pakar

Basis Pengetahuan, berisi pengetahuan yang dibutuhkan untuk memahami, memformulasi, dan memecahkan masalah. Basis pengetahuan tersusun atas 2 elemen dasar:

1. Fakta, terdiri dari: situasi, kondisi, dan kenyataan dari permasalahan yang ada, serta teori.
2. Aturan, mengarahkan penggunaan pengetahuan dalam memecahkan masalah yang spesifik pada bidang tertentu atau khusus

Mesin Inferensi *(Inference Engine),* adalah otak dari Sistem Pakar. Yang dikenal sebagai penerjemah aturan (rule interpreter). Komponen ini berupa program komputer yang menyediakan suatu metodologi untuk memikirkan (reasoning) dan memformulasi kesimpulan. Kerja mesin inferensi meliputi:

1. Menentukan aturan mana akan dipakai
2. Menyajikan pertanyaan kepada pemakai, ketika diperlukan.
3. Menambahkan jawaban ke dalam memori Sistem Pakar.
4. Menyimpulkan fakta baru dari sebuah aturan
5. Menambahkan fakta ke dalam memori.

Pemilihan Masalah

Pembuatan Sistem Pakar membutuhkan waktu dan biaya yang banyak. Tujuan dari dibuatnya beberapa pedoman untuk menentukan apakah Sistem Pakar cocok untuk memecahkan suatu problem adalah untuk menghindari kegagalan dan kerugian yang besar:

a. Biaya, diperlukan untuk pembangunan Sistem Pakar ditentukan oleh kebutuhan.

b. Pakar tidak mudah ditemui untuk semua situasi di mana dibutuhkan. Jika pakar pengetahuan tersebut terdapat di mana saja dan kapan saja, maka pembangunan Sistem Pakar menjadi kurang berharga.
c. Problem yang ada dapat diselesaikan dengan teknik penalaran simbolik, dan tidak membutuhkan kemampuan fisik.
d. Problem harus terstruktur dengan baik dan tidak membutuhkan terlalu banyak pengetahuan awam (common sense).
e. Problem tidak mudah diselesaikan dengan metode komputasi yang lebih tradisionil. Jika terdapat penyelesaian secara algoritmis yang bagus untuk sebuah problem, maka tidak perlu adanya Sistem Pakar.
f. Ada seorang pakar yang dapat memberikan penjelasan tentang kepakarannya dan bisa bekerjasama. Pakar yang dihubungi benar-benar memiliki keinginan kuat untuk berpartisipasi serta tidak merasa pekerjaannya terancam menjadi sangat penting.
g. Problem mempunyai sekup yang tepat. Yaitu problem yang membutuhkan kepakaran yang sangat khusus namun hanya membutuhkan seorang pakar untuk dapat menyelesaikannya dalam waktu yang relatif singkat (misalnya paling lama 1 jam).

B. Forward Chaining (Sistem Perantaian Maju)

Pada sistem forward chaining, fakta-fakta sistem tersimpan dalam memori kerja yang secara terus menerus diperbarui. aturan kondisi-aksi adalah aturan dalam sistem yang merepresentasikan aksi apa saja yang harus diambil jika terdapat suatu kondisi khusus pada item-item dalam memori kerja. Kondisi tersebut dapat berupa pola yang cocok dengan item pada memori kerja, sedangkan aksi dapat berupa penambahan atau penghapusan item pada memori kerja.

Proses siklus mengenal-beraksi (recognise-act) merupakan Aktivitas sistem. Dengan mengawali proses sistem yaitu mencari semua aturan yang kondisinya ada pada memori kerja, kemudian memilih salah satunya sehingga dapat menjalankan aksi yang sesuai

dengan aturan. Sebuah strategi tetap yang disebut strategi penyelesain konflik dibuat didasarkan pada pemilihan aturan yang akan dijalankan (fire). Sehingga sksi tersebut dapat menghasilkan sebuah memori kerja baru, siklus diulangi sampai tidak ada aturan yang dapat dipicu (fire), atau goal (tujuan) yang dikehendaki sudah tercapai.

Sebagai contoh, terdapat beberapa aturan sederhana berikut ini (untuk menyatakan sebuah variabel maka menggunakan kata yang diawali huruf kapital. Namun pada sistem lain, menggunakan cara lain, misalnya saja dapat menggunakan awalan yang berupa tanda baca ? atau ^):

1. JIKA (mengajar X) DAN (mengoreksi_tugas X) MAKA TAMBAH (terlalu_banyak_bekerja X)
2. JIKA (bulan Juni) MAKA TAMBAH (mengajar Rindu)
3. JIKA (bulan Juni) MAKA TAMBAH (mengoreksi_tugas rindu)
4. JIKA (terlalu_banyak_bekerja X) ATAU (lelah X) MAKA TAMBAH (mood_kurang_baik X)
5. JIKA (mood_kurang_baik X) MAKA HAPUS (bahagia X)
6. JIKA (mengajar X) MAKA HAPUS (meneliti X)

dengan asumsi, pada awalnya memiliki memori kerja yang berisi fakta berikut:

(bulan Juni)
(bahagia Rindu)
(meneliti Rindu)

Pada tahap awal sistem Pakar akan memeriksa semua aturan untuk mengenali aturan mana yang dapat memicu aksi, yaitu aturan 2 dan 3. Kemudian sistem akan memilih salah satu di antara kedua aturan dengan menggunakan strategi penyelesaian konflik. Untuk 2 aturan yang terpilih, maka fakta (mengajar Rindu) akan ditambahkan ke dalam memori kerja. Keadaan memori kerja sekarang menjadi:

(mengajar Rindu)
(bulan Juni)
(bahagia Rindu)

(meneliti Rindu)

Proses pengulangan siklus terjadi kembali, kali ini pada aturan 3 dan 6 yang kondisinya terpenuhi. Aturan 3 terpilih dan terpicu, maka fakta (mengoreksi_tugas Rindu) akan ditambahkan ke dalam memori kerja. Kemudian isi dari siklus ketiga, aturan 1 terpicu, sehingga variabel X akan berisi *(bound to)* Rindu, dan fakta (terlalu_banyak_bekerja Rindu) ditambahkan, sehingga isi memori kerja menjadi:

(terlalu_banyak_bekerja Rindu)
(mengoreksi_tugas Rindu)
(mengajar Rindu)
(bulan Juni)
(bahagia Rindu)
(meneliti Rindu)

Ketika Aturan 4 dan 6 dapat diterapkan. Seperti halnya aturan 4 yang terpicu, sehingga fakta (mood_kurang_baik Rindu) ditambahkan. Pada siklus selanjutnya, aturan 5 yang terpilih dan dipicu, sehingga fakta (bahagia Rindu) dihapus dari memori kerja. Lalu aturan 6 akan terpicu sehingga fakta (meneliti Rindu) dihapus dari memori kerja menjadi:

(mood_kurang_baik Rindu)
(terlalu_banyak_bekerja Rindu)
(mengoreksi_tugas Rindu)
(mengajar Rindu)
(bulan Juni)

Urutan aturan yang dipicu bisa jadi sangat vital, utamanya jika aturan yang ada mengakibatkan terhapusnya item dari memori kerja. Pada kasus berikut ini: seandainya terdapat aturan tambahan pada kumpulan aturan di atas, yaitu:

7. JIKA (bahagia X) MAKA TAMBAH (memberi_nilai_bagus X)

Jika aturan 7 ini terpicu sebelum (bahagia Rindu) dihapus dari memori, maka Sistem Pakar akan memberi kesimpulan bahwa saya

akan memberi nilai bagus. Namun jika aturan 5 yang terpicu lebih dahulu, maka aturan 7 tidak dapat dijalankan (artinya saya tidak akan memberi nilai bagus).

C. Backward Chaining (Perantaian Balik)

Kita telah mengetahui bagaimana sebuah sistem berbasis aturan dapat digunakan untuk mendapatkan kesimpulan yang baru dari data yang ada, ataupun menambah kesimpulan ini ke dalam memori kerja. Pendekatan ini dapat bermanfaat ketika kita mengetahui semua fakta pada awalnya, namun tidak dapat menebak kesimpulan yang bisa diambil. Sebuah Perantaian maju menjadi tidak efisien disebabkan karena kita telah mengetahui kesimpulan yang seharusnya, atau mempunyai beberapa hipotesis yang spesifik.

Sebagai contoh, jika kita ingin mengetahui apakah sekarang saya dalam keadaan mood yang baik, kemungkinan kita akan berulangkali memicu aturan-aturan dan memperbarui memori kerja untuk mendapat kesimpulan yang terjadi pada bulan Juni, atau apa yang terjadi jika saya mengajar, yang sebenarnya tidak perlu terlalu kita pusingkan. Yang terpenting adalah bagaimana cara untuk menarik kesimpulan yang relevan dengan tujuan atau goal.

Hal tersebut dapat dikerjakan dengan metode perantaian balik dari pernyataan goal (atau sebuah hipotesis yang menarik). Jika diberi sebuah goal yang akan dibuktikan, maka sistem akan memeriksa apakah goal yang ada cocok dengan fakta-fakta awal yang dimiliki. Jika ya, maka goal telah terbukti atau terpenuhi. Jika tidak, maka sistem akan mencari aturan-aturan yang konklusinya (aksinya) cocok dengan goal. Salah satu aturan tersebut akan dipilih, kemudian sistem akan mencoba membuktikan fakta-fakta prakondisi aturan menggunakan prosedur yang sama, yaitu goal baru yang harus dibuktikan dengan menset prakondisi.

Harus kita perhatikan bahwa pada perantaian balik, sistem tidak perlu memperbarui memori kerja, namun memerlukan

pencatatatan goal apa saja yang dibuktikan untuk dapat membuktikan goal utama (hipotesis).

Dalam hal ini, dapat menggunakan aturan-aturan yang sama untuk perantaian maju dan balik dengan sedikit memodifikasi aturan. Bagian MAKA dalam aturan biasanya tidak diekspresikan sebagai suatu aksi untuk dijalankan (misalnya TAMBAH atau HAPUS), tetapi suatu keadaan yang bernilai benar jika premisnya (bagian JIKA) bernilai benar semua aturan ini berlaku pada perantaian balik. Sehingga aturan-aturan di atas dapat diubah menjadi:

1. JIKA (mengajar X) DAN (mengoreksi_tugas X) MAKA (terlalu_banyak_bekerja X)
2. JIKA (bulan Juni) MAKA (mengajar Rindu)
3. JIKA (bulan Juni) MAKA (mengoreksi_tugas Rindu)
4. JIKA (terlalu_banyak_bekerja X) ATAU (lelah X) MAKA (mood_kurang_baik X)
5. JIKA (mood_kurang_baik X) MAKA TIDAK BENAR (bahagia X)
   dengan fakta awal :
   (bulan Juni)
   (meneliti Rindu)

Apabila kita hendak membuktikan apakah mood sedang kurang baik. Harus kita periksa terlebih dahulu apakah goal cocok dengan fakta awal. Jika tidak ada fakta awal yang menyatakan demikian, maka langkah kedua adalah mencari aturan mana yang mempunyai kesimpulan (mood_kurang_baik Rindu).

Aturan yang cocok adalah aturan 4 dengan variabel X diisi dengan *(bound to)* Rindu. Dengan demikian kita harus membuktikan bahwa prakondisi aturan ini, (terlalu_banyak_bekerja Rindu) atau (lelah Rindu), salah satunya adalah benar (karena memakai ATAU). Kemudian periksa aturan manakah yang membuktikan bahwa adalah (terlalu_banyak_bekerja Rindu) benar, ternyata aturan 1, sehingga prakondisinya, (mengajar X) dan (mengoreksi_tugas X), dua-duanya benar (karena memakai DAN).

Ternyata menurut aturan 2 dan 3, keduanya bernilai bernilai benar jika (bulan Juni) adalah benar. Hal ini sudah sesuai dengan fakta awal, maka keduanya bernilai benar. Karena semua goal sudah terpenuhi maka goal utama (hipotesis) bahwa mood saya sedang kurang baik adalah benar (terpenuhi).

Stack (tumpukan) digunakan untuk mencatat semua goal yang harus dipenuhi/dibuktikan. Jika setiap kali terdapat aturan yang konklusinya cocok dengan goal yang sedang dibuktikan, maka fakta-fakta prakondisi dari aturan itu diletakkan (push) ke dalam stack sebagai goal baru. Dan setiap kali goal pada tumpukan teratas terpenuhi, maka goal tersebut diambil (pop) dari tumpukan. Demikian seterusnya sampai goal utama (yang terdapat pada tumpukan terbawah) sudah terpenuhi atau tidak ada goal lagi di dalam stack.

3.3. Soal Latihan

Sistem Pakar sederhana, yang bertujuan untuk mencari apa yang salah sehingga mesin mobil tidak mau menyala, dengan memberikan gejala-gejala yang teramati. Kita anggap Sistem Pakar memiliki aturan-aturan ini, antara lain:

1. JIKA mesin_mendapatkan_bensin DAN starter_dapat_dihidupkan MAKA ada_masalah_dengan_pengapian
2. JIKA TIDAK BENAR starter_dapat_dihidupkan DAN TIDAK BENAR lampu_menyala MAKA ada_masalah_dengan_aki
3. JIKA TIDAK BENAR starter_dapat_dihidupkan DAN lampu_menyala MAKA ada_masalah_dengan_starter
4. JIKA ada_bensin_dalam_tangki_bahan_bakar MAKA mesin_mendapatkan_bensin

Terdapat 3 masalah yang mungkin terjadi, yaitu: ada_masalah_dengan_pengapian, ada_masalah_dengan_aki dan ada_masalah_dengan_starter. Dengan sistem terarah-tujuan (goal-driven), buktikan keberadaan setiap masalah!

BAB IV.

LOGIKA FUZZY

4.1. Tujuan Instruksional Khusus

- Mahasiswa mampu menjelaskan Logika Fuzzy

4.2. Materi Perkuliahan

A. Definisi Logika Fuzzy

Logika fuzzy adalah suatu cara untuk memetakan ruang input ke dalam ruang output. Dalam logika fuzzy sebuah nilai dapat bernilai benar dan salah secara bersamaan namun berapa besar kebenaran dan kesalahan sebuah nilai tergantung kepada bobot keanggotaan yang dimiliki. Beberapa alasan kenapa menggunakan logika fuzzy, antara lain :

1. Konsep logika Fuzzy mudah dimengerti
2. Logika fuzzy cukup fleksibel
3. Logika fuzzy mampu memodelkan fungsi-fungsi nonlinier yang kompleks
4. Logika fuzzy dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung
5. Logika fuzzy bisa bekerjasama dengan teknik kendali konvensional
6. Logika fuzzy didasarkan pada bahasa alami

Manusia cenderung menggunakan bahasa dalam bentuk sesuatu yang dapat dipahami secara umum, bukan dalam bentuk

bahasa matematika yang mementingkan akurasi. Misalkan, kita mengatakan:

"*Benda itu sangat berat*" daripada "*Benda itu beratnya 1500 kg*".

Representasi fakta semacam di atas menggunakan istilah yang ambigu atau samar (fuzzy). Misalnya, kata *sangat* berat dapat memiliki arti berbeda-beda, seberapa berat?.

Dengan teori Fuzzy sets kita dapat merepresentasikan dan menangani masalah ketidakpastian dalam hal ini bisa berarti keraguan, ketidaktepatan, kekuranglengkapan informasi dan kebenaran yang bersifat sebagian. Fuzzy system adalah suatu sistem yang menggunakan *himpunan fuzzy* untuk memetakan suatu inputan menjadi ouput tertentu (black box).

Gambar 4.1. Contoh Pemetaan Input Output

Aplikasi logika fuzzy sudah banyak sekali digunakan dalam berbagai bidang kehidupan, diantaranya :

- 1950, Pertama kali diperkenalkan mesin cuci dengan menggunakan logika fuzzy oleh sebuah perusahaan jepang
- Pada kedokteran, logika fuzzy diterapkan untuk mendiagnosis penyakit
- Pada bidang ekonomi, logika fuzzy digunakan untuk pemasaran.
- Pada bidang psikologi, logika fuzzy digunakan untuk menganalisa kelakuan masyarakat, pencegahan dan investigasi criminal.
- Pada bidang ilmu lingkungan, untuk kendali kualitas air serta perkiraan cuaca
- Klasifikasi dan pencocokan pola.

- Pada perancangan jaringan computer, dapat digunakan untuk prediksi adanya gempa bumi. Serta berbagai bidang yang lain

B. Himpunan Fuzzy

Dalam logika fuzzy dikenal himpunan fuzzy (fuzzy set) yang merupakan pengelompokan sesuatu berdasarkan vareabel bahasa dan dinyatakan dalam sebuah fungsi keanggotaan. Fungsi keanggotaan pada logika fuzzy bernilai dari 0 sampai 1.
Yang sering ditulis μA[x], yaitu :

- Satu (1), yaitu suatu item menjadi anggota dalam suatu himpunan
- Nol (0), suatu item tidak menjadi anggota dalam suatu himpunan.

Ada beberapa hal yang perlu diketahui untuk memahami logika fuzzy, yaitu :

- Vareable fuzzy, vareabel yang akan dibahas pada sistem fuzzy
- Himpunan fuzzy, suatu grup yang mewakili kondisi tertentu pada vareabel fuzzy
- Semesta pembicaraan, keseluruhan nilai yang dioperasikan dalam vareabel fuzzy
- Domain, keseluruhan nilai yang diperbolehkan pada semesta pembicaraan dan bisa dioperasikan pada himpunan fuzzy

Karena kemiripan antara keanggotaan fuzzy dengan probabilitas menimbulkan kerancuan. Keduanya mempunyai nilai pada interval [0,1], tetapi interpretasi nilainya berbeda antara kadua kasus tersebut. Keanggotaan fuzzy memberikan ukuran untuk sebuah keputusan, pada probabilitas jika terjadi keseringan hasilnya bernilai benar dalam jangka yang panjang maka terindikasi sebuah proporsi.
Himpunan fuzzy memiliki dua atribut, antara lain:

1. Linguistik, penamaan sebuah grup yang mewakili keadaan tertentu
2. Numeris, nilai yang menunjukkan ukuran dari vareabel

Contoh :

1. Jika diketahui :

   S = {1,2,3,4,5,6} adalah semesta pembicaraan

   A = {1,2,3}

   B = {3,4,5}

   Dapat dikatakan:

   - Nilai keanggotaan 2 pada himpunan A, μA[2] = 1, Karena 2 ε A
   - Nilai keanggotaan 3 pada himpunan A, μA[3] = 1, Karena 3 ε A
   - Nilai keanggotaan 4 pada himpunan A, μA[4] = 0, Karena 4 ε A ⁄
   - Nilai keanggotaan 2 pada himpunan A, μA[2] = 0, Karena 2 ε A ⁄
   - Nilai keanggotaan 3 pada himpunan B, μB[3] = 1, Karena 3 ε B

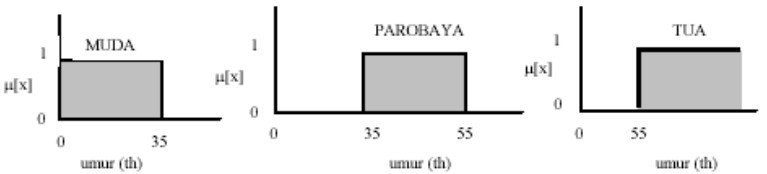
2. Ada tiga kategori vareabel umur, antar lain :

   Muda umur < 35 tahun

   Parobaya 35 ≤ umur ≤ 55 tahun

   Tua umur ≥ 55 tahun

   Secara grafis dapat kita lihat :



Gambar 4.2. Himpunan Muda, Parobaya, Tua

Usia 34 tahun maka dikatakan Muda → μMuda [34] = 1
Usia 35 tahun maka dikatakan Tidak Muda → μMuda [35] = 0
Usia 34 tahun maka dikatakan Parobaya → μParobaya [35] =1
Usia 34 tahun maka dikatakan Tidak Parobaya → μParobaya [34] = 0
Usia 35 tahun kurang 1 hari maka dikatakan Tidak Parobaya → μParobaya [35 th – 1 hr] = 0

Himpunan crisp untuk menyatakan umur tidak bisa adil karena adanya perubahan kecil saja pada sebuah nilai dapat mengakibatkan perbedaan kategori yang cukup signifikan.

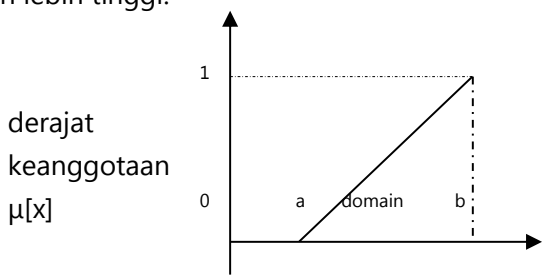
C. Fungsi Keanggotaan

Fungsi keanggotaan adalah kurva yang menunjukkan pemetaan titik-titik input data kedalam nilai keanggotaan dan mempunyai interval 0 sampai 1. Untuk mendapakan nilai keanggotaan dapat dilakukan dengan pendekatan fungsi. Beberapa fungsi yang dapt digunakan:

a. Representasi Linear

Bentuk ini merupakan bentuk yang paling sederhana dan merupakan pilihan yang baik untuk mendekati konsep yang kurang jelas. Pemetaan input ke derajat keanggotaan digambarkan dengan sebuah garis lurus.

Pada representasi linear naik, kenaikan himpunan pada nilai domain mempunyai derajat kenaggotaan nol (0) bergerak ke arah kanan menuju ke nilai domain yang mempunyai derajat keanggotaan lebih tinggi.

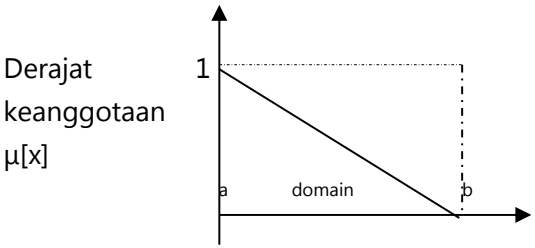


Gambar 4.3. Representasi linear naik

Fungsi keanggotaan :

$$\mu[x] = \begin{cases} 0; & x \le a \\ (x-a) / (b-a); & a \le x \le b \\ 1; & x \ge b \end{cases}$$

Pada representasi linear turun merupakan kebalikan dari representasi linear naik. Yaitu garis lurus yang berasal dari nilai domain yang derajat keanggotaan tertinggi pada sisi kiri, bergerak menurun menuju domain yang nilai derajat keanggotaannya lebih rendah.

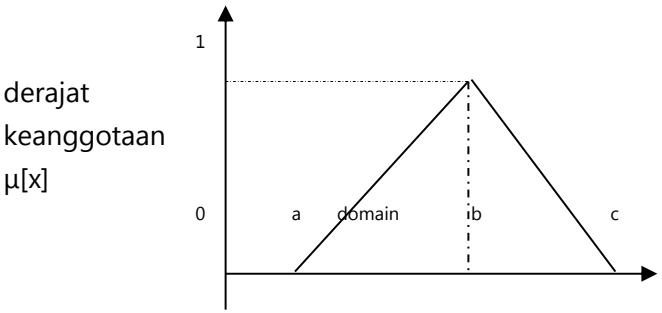


Gambar 6. Representasi linear turun

Fungsi keanggotaan :

$$\mu[x] = \begin{cases} (b-x) / (b-a) & a \le x \le b \\ 0; & x \ge b \end{cases}$$

b. Representasi kurva segitiga. Merupakan gabungan dari dua garis linier.

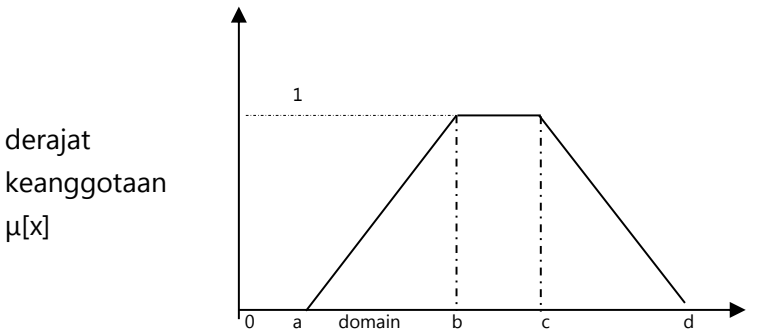


Gambar 7. Kurva segitiga

Fungsi keanggotaan :

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge c \\ (x-a) / (b-a); & a \le x \le b \\ (c-x) / (c-b); & b \le x \le c \end{cases}$$

c. Representasi kurva trapesium
Pada dasarnya sama dengan kurva segitiga hanya beberapa titik mempunyai keanggotaan 1.



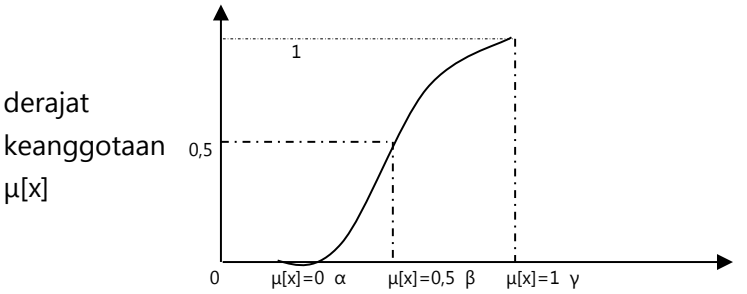
Gambar 8. Kurva trapesium

Fungsi keanggotaan :

$$\mu[x] = \begin{cases} 0; & x \le a \text{ atau } x \ge d \\ (x-a) / (b-a); & a \le x \le b \\ 1; & b \le x \le c \\ (d-x) / (d-c); & c \le x \le d \end{cases}$$

d. Representasi kurva bentuk bahu
   Daerah yang terletak di tengah-tengah suatu vareabel dan dalam bentuk segitiga, sisi kanan dan kirinya akan naik turun.

e. Representasi kurva-S
   Kurva pertumbuhan dan penyusutan merupakan kurva-S atau sigmoid yang berhubungan dengan kenaikan dan penurunan permukaan secara tak linier. Kurva ini didefinisikan dengan 3 parameter antara lain:
   1. nilai keanggotaan nol (ά)
   2. nilai keanggotaan lengkap (γ)
   3. titik infleksi (β)



Gambar 9. Kurva-S

Fungsi keanggotaan :

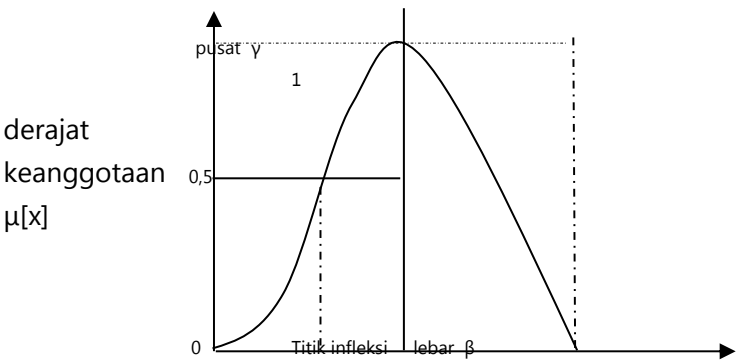
$$S(x;\alpha,\beta,\gamma) = \begin{cases} 0; & x \le \alpha \\ 2((x-\alpha)/(\gamma-\alpha))^2; & \alpha \le x \le \beta \\ 1-2((\gamma-x)/(\gamma-\alpha))^2; & \beta \le x \le \gamma \\ 1; & x \ge \gamma \end{cases}$$

f. Representasi kurva bentuk lonceng

Kurva ini terbagi 3 kelas yaitu : himpunan fuzzy π, beta dan gauss. Perbedaanya terletak pada gradien masing-masing.

1. Kurva π

   Kurva ini berbentuk lonceng derajat keanggotaannya 1 yang terletak pada pusat dengan domain (γ), dan lebar kurva (β).



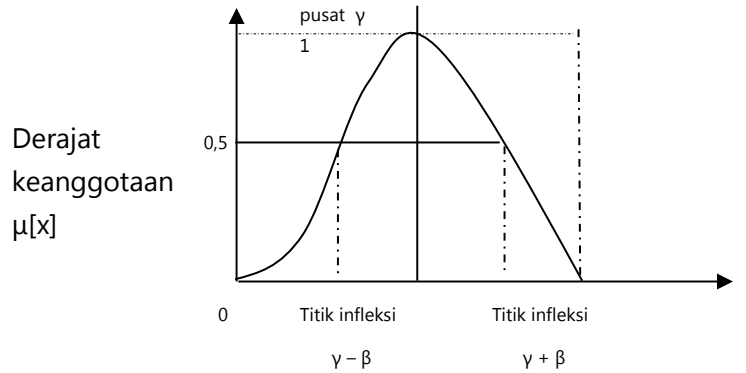
Gambar 10. Kurva π

Fungsi keanggotaan :

$$\pi(x, \beta, \gamma) = \begin{cases} S\left(x; \gamma - \gamma - \frac{\beta}{2}, \gamma\right) & x \le \gamma \\ 1\text{-}S\left(x; \gamma, \gamma + \frac{\beta}{2}, \gamma + \beta\right) & x > \gamma \end{cases}$$

2. Kurva β

Kurva ini juga berbentuk lonceng namun lebih rapat. Kurva ini didefinisikan dengan 2 parameter, yaitu nilai pada domain yang menunjukkan pusat kurva (γ), dan setengah lebar kurva (β).
Perbedaan yang mencolok antara kurva π dan kurva β ini adalah pada kurva β fungsi keanggotaannya akan bernilai nol (0) hanya jika nilai (β) sangat besar.



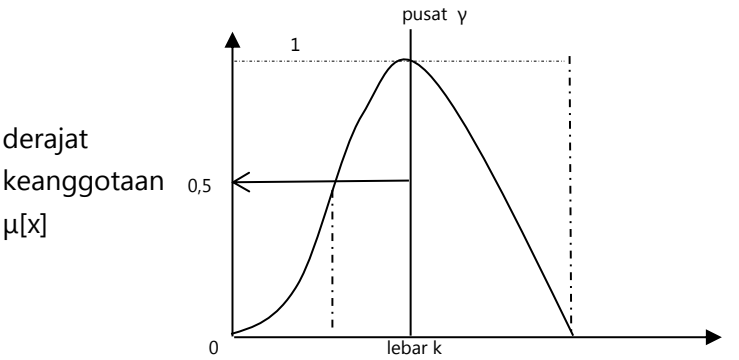
Gambar 11. Kurva β

Fungsi keanggotaan :

$$\beta(x, \beta, \gamma) = \frac{1}{1 + \left(\frac{x - \gamma}{\beta}\right)}$$

3. Kurva Gauss

Kurva GAUSS juga menggunakan π untuk menunjukkan nilai domain pada pusat kurva dan k yang menunjukkan lebar kurva.



Gambar 12. Kurva β

Fungsi keanggotaan :

$G(x; k, \gamma) = e^{-k(y-x)2}$

k menunjukkan lebar kurva.

g. Koordinat keanggotaan

Himpunan fuzzy berisi urutan pasangan berurutan yang mempunyai nilai domain dan kebenaran nilai keanggotaan dalam bentuk :

Skalar (i) / Drajat (i)

Skalar adalah nilai dari domain himpunan fuzzy

Drajat adalah derajat keanggotaan himpunan fuzzy

D. Operator Dasar Operasi Himpunan Fuzzy

Operasi yang digunakan untuk menkombinasi dan memodifikasi himpunan fuzzy ada beberapa, hal tersebut sama dengan yang digunakan paa himpunan konvensional. Nilai keanggotaan hasil dari operasi dari 2 himpunan dikenal dengan *free strength*. Zadeh menciptakan 3 buah operasi dasar, yaitu:

1. Operator AND

   Operator AND berhubungan dengan interseksi pada himpunan.

   $\mu A \cap B = \min(\mu A[x], \mu B[y])$

2. Operator OR
   Operator OR berhubungan dengan operasi union pada himpunan
   $\mu A \cup B = \max(\mu A[x], \mu B[y])$
3. Operator NOT
   Operator NOT berhubungan dengan operasi komplemen pada himpunan
   $\mu A' = 1 - \mu A[x]$

E. Metode Tsukamoto

Pada metode ini, aturan yang berbentuk IF-THEN harus direpresentasikan dengan sebuah fungsi keanggotaan yang monoton. Hasilnya adalah berupa output inferensi dari setiap aturan dipertegas berdasarkan α-predikat dengan menggunakan rata-rata berbobot.

Misalkan ada 2 vareabel input, Var–1 (x) dan Var–2 (y) serta 1 vareabel output Var–3 (z), yang mana Var–1 (x) dibagi menjadi himpunan A_1 dan A_2. Var–2 (y) juga terbagi atas himpunan B_1 dan B_2. Var–3 (z) juga terbagi atas himpunan yang kedua himpunan ini harus MONOTON yaitu C_1 dan C_2. ada dua atuaran yang digunakan :

[R1] IF (x is A_1) and (y is B_1) THEN (z is C_1)

[R2] IF (x is A_2) and (y is B_2) THEN (z is C_2)

F. Metode Mamdani

Metode ini dikenal juga sebagai metode max-min. Pertama kali diperkenalkan oleh Ebrahim Mamdani (1975). Ada 4 tahap untuk menghasilkan output, yaitu :

1. Pembentukan himpunan fuzzy. Pada vareabel input dan output dibagi menjadi satu atau lebih himpunan fuzzy.
2. Aplikasi fungsi aplikasi. Fungsi implikasi yang digunakan adalah Min.

3. Komposisi aturan. Ada 3 metode yang digunakan dalam melakukan inferensi sistem fuzzy, yaitu :
   - ***max,*** solusi yang didapat yaitu dengan mengambil nilai maksimu aturan lalu memakainya untuk memperbaiki daerah fuzzy dan mengaplikasikan pada output dengan operator OR. Bila semua proposisi sudah dievaluasi, maka output akan berisi himpunan fuzzy yang merefleksikan konstribusi dari setiap proposisi.

     $\mu_{sf}[x_i] = \max(\mu_{sf}[x_i], \mu_{kf}[x_i])$

     dengan :

     $\mu_{sf}[x_i]$: nilai keanggotaan solusi fuzzy sampai aturan ke-i

     $\mu_{kf}[x_i]$: nilai keanggotaan konsekuen fuzzy aturan ke-i

   - ***additive,*** solusi yang didapat dengan melakukan *bounded-sum* ke semua daerah fuzzy.

     $\mu_{sf}[x_i] = \min(1, \mu_{sf}[x_i] + \mu_{kf}[x_i])$

     dengan :

     $\mu_{sf}[x_i]$: nilai keanggotaan solusi fuzzy sampai aturan ke-i

     $\mu_{kf}[x_i]$: nilai keanggotaan konsekuen fuzzy aturan ke-i

   - ***probabilistik OR (probor),*** solusi yang didapat dengan melakukan *product* pada semua output.

     $\mu_{sf}[x_i] = (\mu_{sf}[x_i] + \mu_{kf}[x_i]) - (\mu_{sf}[x_i] * \mu_{kf}[x_i])$

     dengan :

     $\mu_{sf}[x_i]$: nilai keanggotaan solusi fuzzy sampai aturan ke-i

     $\mu_{kf}[x_i]$: nilai keanggotaan konsekuen fuzzy aturan ke-i

4. Penegasan (defuzzy)

   Inputannya berupa himpunan fuzzy yang didapat dari komposisi aturan fuzzy, untuk output yang dihasilkan adalah bilangan pada domain himpunan fuzzy itu sendiri. Jadi jika himpunan fuzzy diberi sebuah nilai range tertentu maka harus dapat diambil nilai crisp sebagai outputnya.

Beberapa metode yang dipakai antara lain:

- Metode Centroid (composite Moment), solusi tegasnya didapat dengan mengambil titik pusat (z*) di daerah fuzzy

$$z^* = \frac{\int_z z\mu(z)dz}{\int_z \mu(z)dz} \qquad \textit{untuk vareabel kontinu,}$$

$$z^* = \frac{\sum_{j=1}^{n} z_j\mu(z_j)}{\sum_{j=1}^{n} \mu(z_j)} \qquad \textit{untuk vareabel deskret,}$$

- Metode Bisektor, solusi tegasnya didapat dengan mengambil nilai pada domain yang mempunyai nilai setengah dai jumlah total nilai keanggotaan di daerah fuzzy.

  Secara umum :

$$zp \text{ sedemikian sehingga } \int_{K1}^{p} \mu(z)\,dz = \int^{\kappa n} \mu(z)\,dz$$

- Metode Mean of Maximum (MOM), solusi tegasnya didapat dengan mengambil nilai rata-rata domain yang mempunyai nilai keanggotaan maksimum
- Metode Largest of Maximum (LOM), solusi tegasnya didapat dengan mengambil nilai terbesar dari domain yang mempunyai nilai keanggotaan maksimum
- Metode Smallest of Maximum (SOM), solusi tegasnya didapat dengan mengambil nilai terkecil dari domain yang mempunyai nilai keanggotaan minimum

G. Metode Sugeno

Diperkenalkan oleh Takagi-Sugeno Kang pada tahun 1985. Penalaran pada metode ini mirip dengan metode Mamdani, akan tetapi output sistem yang dihasilkan berupa konstanta atau persamaan linier bukan berupa himpunan fuzzy.

Dua buah model bentuk metode ini adalah :

a. Model Fuzzy Sugeno Orde Nol
   Bentuknya secara umum :
   IF $(x_1$ *is* $A_1) \cdot (x_2$ *is* $A_2) \cdot (x_3$ *is* $A_3) \cdot \ldots \cdot (x_n$ *is* $A_n)$ *THEN* $z = k$
   A_1 adalah himpunan fuzzy ke-I sebagai anteseden
   K adalah konstanta (tegas) sebagai konsekuen

b. Model Fuzzy Sugeno Orde satu
   IF $(x_1$ *is* $A_1) \cdot (x_2$ *is* $A_2) \cdot \ldots \cdot (x_n$ *is* $A_n)$ *THEN* $z = p_1 * x_1 + \ldots p_1 * x_n + q$
   A_1 adalah himpunan fuzzy ke-I sebagai anteseden
   k adalah konstanta (tegas) sebagai konsekuen
   q juga merupakan konstanta dalam konsekuen

4.3. Soal Latihan

1. Sistem fuzzy sangat luas digunakan dalam berbagai bidang. Berikan contoh aplikasi fuzzy pada beberapa bidang (min 3 bidang)!
2. Apakah perbedaan logika fuzzy dengan logika tegas?
3. Apakah yang dimaksud dengan defuzzyfikasi?
4. Jika nilai keanggotaan 32 tahun pada himpunan muda adalah 0,6 (μmuda[27]=0,9). Nilai keanggotaan Rp. 3.000.000,- pada himpunan penghasilan tinggi adalah 0,4 (μ Gaji Tinggi $[3x10^6]$=0,4). Maka α-predikat untuk usia muda dan berpenghasilan tinggi adalah? (gunakan operator AND, OR dan NOT)
5. Sebuah perusahaan makanan kaleng jenis Sedap, pada 1 bulan terakhir permintaan terbesar adalah 5000 kemasan/hari,

permintaan terkecil 1000 kemasan/hari. Persediaan barang di gudang terbanyak 600 kemasan/hari dan terkecil mencapai 100 kemasan/hari. Dengan semua keterbatasan sampai saat ini perusahaan baru mampu meproduksi barang maksimum 7000 kemasan/hari. Untuk efesiensi mesin dan SDM perhari maka perusahaan diharapkan paling tidak memproduksi sebanyak 2000 kemasan/hari. Berapa kemasankah yang harus diproduksi jika jumlah permintaan sebanyak 4000 kemasan dan persediaan digudang masih 300 kemasan. Apabila proses produksi perusahaan tersebut menggunakan 4 aturan fuzzy, sbb:

R1 : IF permintaan turun AND persediaan banyak THEN produksi berkurang

R2 : IF permintaan turun AND persediaan sedikit THEN produksi berkurang

R3 : IF permintaan naik AND persediaan banyak THEN produksi bertambah

R4 : IF permintaan naik AND persediaan sedikit THEN produksi bertambah

BAB V.

JARINGAN SARAF TIRUAN

5.1. Tujuan Instruksional Khusus

- Mahasiswa mampu menjelaskan Jaringan Syaraf Tiruan
- Mahasiswa mampu menerapkan metode Jaringan saraf tiruan untuk memecahkan masalah maupun prediksi

5.2. Materi Perkuliahan

A. Jaringan Saraf Tiruan (JST)

Jaringan syaraf merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia. Dikatakan sebagai buatan karena diimplementasikan menggunakan program komputer untuk menyelesaikan sejumlah proses perhitungan saat pembelajaran berlangsung. Pada otak manusia berisi berjuta sel syaraf yang bertugas untuk memproses informasi. Tiap sel bekerja seperti seperti sebuah processor sederhana dan saling beriteraksi untuk mendukung kemampuan kerja otak manusia.

JST dapat digambarkan sebagai model matematis dan komputasi untuk fungsi klasifikasi data, cluster dan regresi non parametik atau sebuah simulasi dari koneksi model syaraf biologi. Model syaraf ditunjukkan dengan kemampuan dalam emulasi, analisa, prediksi dan asosiasi. Berdasarkan kemampuan yang dimiliki, JST dapat digunakan untuk belajar dan maenghasilkan aturan dan operasi dan untuk menghasilkan output yang sempurna dari input yang dimasukkan dan membuat prediksi tentang kemungkinan

output yang akan muncul atau menyimpan karakteristik dari input yang disimpan. Jaringan syaraf telah mengalami perkembangan sejak ditemukannya, yaitu :

1. 1940, para ilmuwan mengemukakan bahwa psikologi otak sama dengan pemrosesan pada komputer
2. 1943, McCulloch dan Pits merancang model formal untuk perhitungan dasar neuron
3. 1949, Hebb menyatakan bahwa informasi bisa disimpan dalam koneksi-koneksi. Dan juga mengusulkan tentang skema pembelajaran untuk memperbaiki koneksi antar neuron
4. 1954, Farley dan Clark mengatur ulang model relasi adaptif stimulus-respon dalam jaringan random
5. 1958, Rosenblatt mengembangkan konsep dasar perceptron untuk klasifikasi pola
6. 1960, Widrow dan Hoff mengembangkan ADALINE untuk kendali adptif. Serta pencocokan pola yang dilatif dengan pembelajaran LMS
7. 1974, Werbos mengenalkan algoritma *backpropagation* untuk melatih perceptron dengan banyak lapisan
8. 1975, Little dan Shaw menggambarkan jaringan syaraf dengan menggunakan model probabilistic
9. 1982, Kohonen mengembangkan metode pembelajaran jaringan syaraf yang tak terawasi (unsupervised learning). Grossberg dan Carpenter mengenalkan sejumlah arsitektur jaringan yaitu : Adaptive Resonance Theory (ART), ART2 dan ART3. Pada tahun itu juga dikembangkan jaringan syaraf recurrent yang dapat digunakan untuk menyimpan informasi dan optimasi oleh Hopfield.
10. 1985, dikembangkannya algoritma pembelajaran dengan mesin Boltzman menggunakan model jaringan syaraf probabilistik
11. 1988, mulai dikembangkan fungsi radial basis

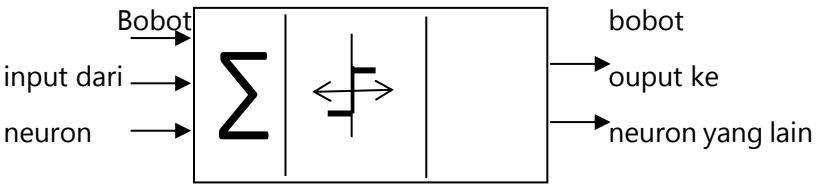
B. Komponen Jaringan Saraf Tiruan (JST)

Sama seperti otak manusia, jaringan syaraf terdiri beberapa neuron yang saling berhubungan. Neuron akan mentransformasikan informasi yang diterima melalui sambungan menuju neuron yang lainnya. Jaringan syaraf tiruan atau JST adalah sebuah pemroses sistem informasi yang memiliki karakteristik performasi tertentu sebagaimana pada jaringan syaraf biologis. Jadi dapat diasumsikan bahwa JST dibentuk dari generalisasi model matematika jaringan syaraf biologis yaitu:

1. Pemrosesan informasi terjadi pada suatu bentuk prosesor sederhana yang disebut neuron
2. Sinyal dilewatkan antat neuron melalui sebuah koneksi tertentu
3. Setiap koneksi berasosiasi dengan bobot tertentu
4. Setiap neuron merupakan fungsi aktifasi tertentu pada masing-masing sinyal input untuk menentukan sinyal keluaran.

Dengan begitu karakteristik JST didasarkan pada

- pola koneksi antar neuron yang disebut Arsitektur JST
- metode untuk menentukan perubahan nilai bobot, yang disebut algoritma JST
- fungsi aktifasi.



Gambar 13. Struktur neuron JST

Pada struktur diatas, cara kerja neuron buatan sama dengan neuron biologis. Input (berupa informasi) akan dikirim ke neuron dengan bobot kedatangan. Input akan diproses oleh Fungsi perambatan yang akan menjumlahkan semua nilai bobot yang masuk. Hasilnya akan dibandingkan dengan nilai ambang (*threshold*)

melalui *fungsi aktivasi* setiap neuron. Jika input melewati nilai ambang tertentu, maka neuron tersebut akan diaktifkan. Namun jika tidak maka neuron tidak diaktifkan. Bila neuron tersebut aktif, maka neuron tersebut akan mengirimkan output melalui bobot-bobot output ke semua neuron yang berhubungan dengannya. Pada JST, neuron dikumpulkan pada lapisan-lapisan yang disebut *layers* yang saling dihubungkan antara lapisan sebelum dan lapisan sesudahnya kecuali lapisan input dan lapisan output. Informasi yang disalurkan ke setiap lapisan dilewatkan melalui lapisan yang lain yang disebut sebagai lapisan tersembunyi atau *hidden layer*.

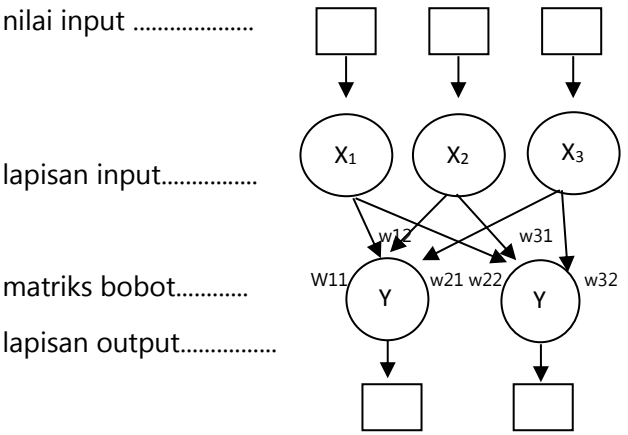
Beberapa jaringan syaraf ada yang memiliki lapisan tersembunyi tatapi ada juga yang tidak mamilikinya. Selain itu juga terdapat jaringan syaraf yang mana neuron-neuronnya tersusun dalam bentuk matriks.

C. Arsitektur Jaringan Saraf Tiruan (JST)

Neuron-neuron dikelompokkan pada lapisan-lapisan yang umumnya letak neuron tersebut pada lapisan yang sama sehingga mempunyai keadaaan yang sama. Faktor terpenting untuk menentukan kelakuan sebuah neuron adalah *fungsi aktivasi* dan *pola bobotnya*.

Ada beberapa macam arsitektur jaringan syaraf, antara lain:

1. **Jaringan dengan lapisan tunggal (*singgle layer net*)**
   Jaringan ini hanya mempunyai satu lapisan saja, dengan bobot terhubung. Yaitu saat menerima input akan secara langsung diolah menjadi output tanpa melalui lapisan tersembunyi terlebih dahulu.

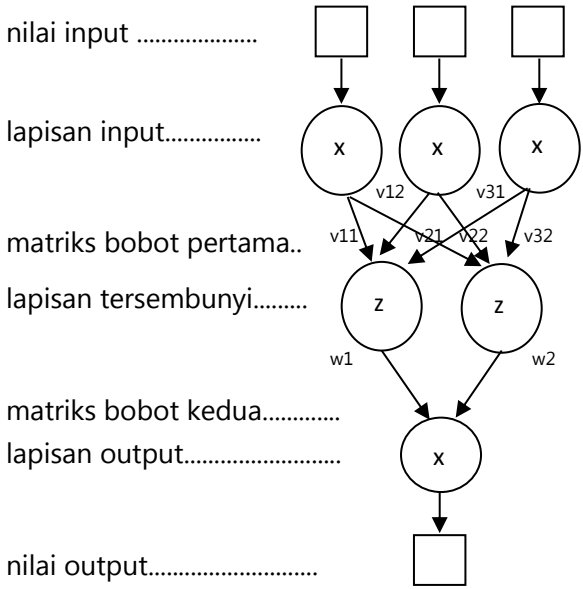


Gambar 14. Jaringan syaraf dengan lapisan tunggal

Pada arsitektur diatas, lapisan input mempunyai 3 neuron yaitu X_1, X_2, X_3. Sedangkan lapisan outputnya mempunyai 2 neuron yaitu Y_1, Y_2. Semua neuron tersebut saling berhubungan yang besar hubungan tersebut ditentukan oleh bobot yang bersesuaian. Semua unit input akan dihubungkan dengan setiap unit output.

2. Jaringan dengan banyak lapisan (*multi layer net*)

Jaringan ini mampu untuk menyelesaikan permasalahan yang lebih rumit tentunya dengan pembelajaran yang cukup rumit juga. Pada kasus lain, pembelajaran dengan jaringan lapisan banyak ini lebih sukses untuk menyelesaikan permasalahan.

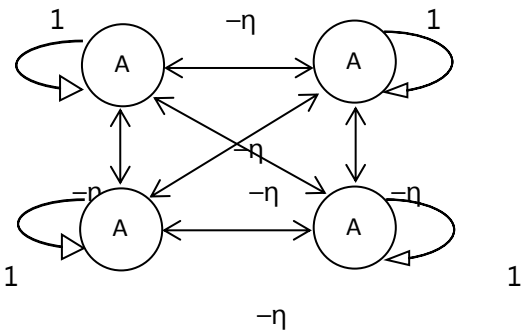


Gambar 15. Jaringan syaraf dengan lapis banyak

Pada arsitektur diatas, jaringan ini memiliki 1 atau lebih lapisan (lapisan termbunyi) yang terletak diantara lapisan input dan lapisan ouptut. Ada lapisan bobot yang terletak pada 2 lapisan yang bersebelahan

3. Jaringan dengan lapisan kompetitif

Pada dasarnya, hubungan antar neuron tidak ditunjukkan secara arsitektur pada beberapa jaringan syaraf. Jaringan ini memiliki bobot –η



Gambar 16. Jaringan syaraf dengan lapisan kompetitif

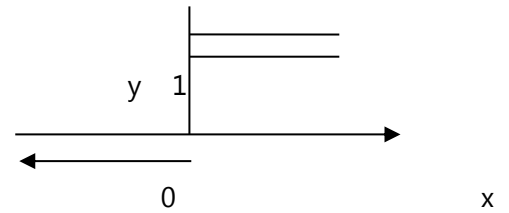
Suatu karakteristik dari JST adalah kesempurnaan dalam pembelajaran. Cara pembelajarannya adalah dari proses pelatihan yan diberikan kepadanya menunjukkan kesamaan dengan perkembangan intelektual manusia. tapikemampuan belajar jaringan syaraf tiruan bersifat terbatas sehingga tidak dapat melakukan segalanya.

Ada beberapa macam fungsi aktivasi yang sering digunakan dalam JST, antara lain:

a) Fungsi Undak Biner (*Hard Limit*)

Jaringan lapis tunggal menggunakan fungsi ini untuk mengkonversikan input dari suatu variabel yang bernilai kontinue ke suatu output biner (0 atau 1)

$$y = \begin{cases} 0, & \text{jika } x \le 0 \\ 1, & \text{jika } x > 0 \end{cases}$$

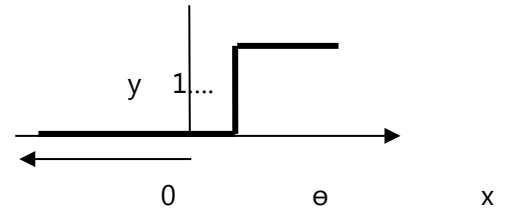


Gambar 17. Fungsi Aktivasi Undak Biner (Hard Limit)

b) Fungsi Undak Biner (Threshold)

Fungsi ini menggunakan nilai ambang yang disebut dengan nilai ambang atau Threshold (nilainya 0).

$$y = \begin{cases} 0, & \text{jika } x < 0 \\ 1, & \text{jika } x \ge 0 \end{cases}$$

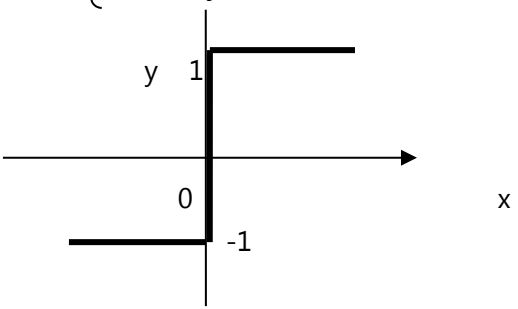


Gambar 18. Fungsi Aktivasi Undak Biner (Threshold)

c) Fungsi Bipolar

Fungsi ini mirip dengan fungsi undak biner, tetapi outputnya berupa 1, 0 atau -1.

$$y = \begin{cases} 1, & \text{jika } x > 0 \\ 0, & \text{jika } x = 0 \\ 1, & \text{jika } x < 0 \end{cases}$$

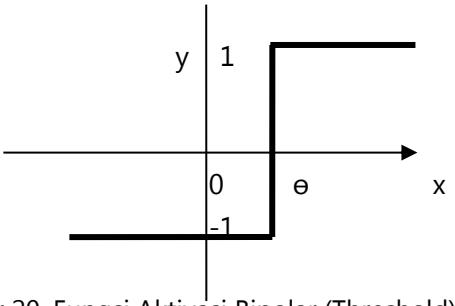


Gambar 19. Fungsi Aktivasi Bipolar (Symetric hard limit)

d) Fungsi Bipolar (Threshold)

Fungsi ini juga sama dengan dengan fungsi undak biner dengan threshold hanya outputnya berupa 1, 0 atau -1

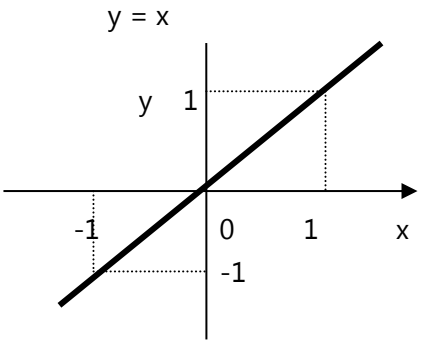
$$y = \begin{cases} 1, & \text{jika } x \geq \theta \\ -1, & \text{jika } x < \theta \end{cases}$$



Gambar 20. Fungsi Aktivasi Bipolar (Threshold)

e) Fungsi Linear

Nilai output fungsi ini adalah sama dengan nilai inputnya

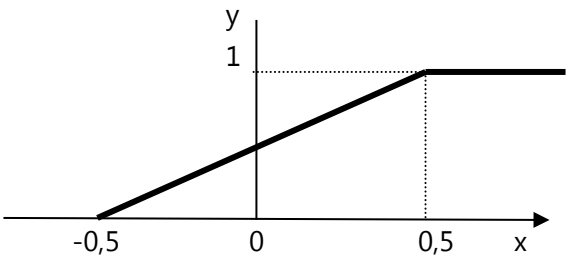


Gambar 21. Fungsi Aktivasi Linear (Identitas)

f) Fungsi Saturating Linear

Nilai outputnya akan bernilai 0 jika inputnya kurang dari -½ , dan bernilai 1 jika inputnya lebih dari ½

$$y = \begin{cases} 1, & \text{jika } x \geq 0,5 \\ x + 0,5, & \text{jika } -0,5 \leq x \leq 0,5 \\ 0, & \text{jika } x \leq -0,5 \end{cases}$$

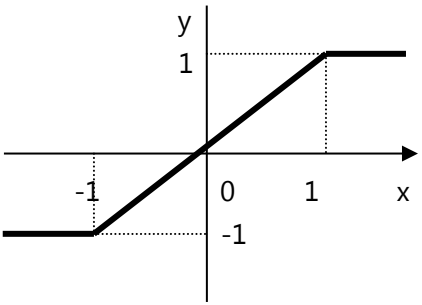


Gambar 22. Fungsi Aktivasi Saturating Linear

g) Fungsi Symetric Saturating Linear

Nilai outputnya akan bernilai -1 jika inputnya kurang dari -1, akan bernilai 1 jika inputnya lebih dari 1

$$y = \begin{cases} 1, & \text{jika } x \geq 1 \\ x, & \text{jika } -1 \leq x \leq 1 \\ 0, & \text{jika } x \leq -1 \end{cases}$$



Gambar 23. Fungsi Aktivasi Symetric Saturating Linear

h) Fungsi Sigmoid Biner

Fungsi ini secara khusus dapat menguntungkan dalam penggunaan jaringan syaraf tiruan yang dilatih dengan metode *backpropagation,* karena hubungan yang sederhana diantara nilai fungsi pada suatu titik mengurangi beban perhitungan selama pelatihan. Fungsi ini seringkali dgunaka untuk jaringan syaraf yang membutuhkan nilai output yang terletak pada interval 0 sampai 1. Nilai rangenya berkisar dari 0 sampai 1.

Dengan rumus :

$$y = f(x) = \frac{1}{1 + e^{-\sigma x}}$$

Dengan $f(x) = \sigma f(x) [1 - f(x)]$

i) Fungsi Sigmoid Bipolar

Fungsi ini diskalakan mempunyai jangkauan yang cocok dengan sebuah permasalahan tertentu. Nilai rangenya berkisar antara 1 sampai -1

Dengan rumus:

$$y = f(x) = \frac{1 - e^{-x}}{1 + e^{-x}}$$

Dengan $f'(x) = \frac{\sigma}{2} [1 + f(x)] [1 - f(x)]$

D. Proses Pembelajaran

Jaringan syaraf mensimulasikan kemampuan otak manusia untuk pembelajaran. Tidak seperti model biologis, jaringan syaraf memiliki struktur yang tidak bisa dirubah. Pada JST juga tersusun atas neuron dan dendrit sama seperti susunan otak manusia. setiap neuron mempunyai nilai tertentu yang menunjukkan koneksi antar neuron yang disebut sebagai bobot. Nilai bobot akan bertambah jika informasi yang diberikan neuron tersebut telah tersampaikan. Jika tidak tersampaikan maka nilai bobot akan dikurangi.

Pada proses pembelajaran dilakukan pada nilai input yang berbeda, nantinya akan dihasilkan sebuah nilai yang seimbang karena nilai bobot berubah secara dinamis. Jika sudah menghasilkan nilai tersebut, maka hal itu menunjukkan bahwa setiap input telah berhubungan dengan outputnya.

Ada dua macam proses pembelajaran pada JST yaitu **pembelajaran terawasi** dan **pembelajaran yang tak terawasi**.

1. Pembelajaran Terawasi (Supervised Learning)

Disebut terawasi karena output yang diharapkan telah diketahui sebelumnya. Saat proses pembelajaran, satu pola input diberikan ke satu neuron pada lapisan input yang akan dirambatkan sepanjang jaringan syaraf sampai pada lapisan output. Yang hasilnya dicocokkan dengan pola output target. Jika tidak cocok maka akan muncul eror yang berarti membutuhkan pembelajaran lebih banyak lagi.

a. Hebb Rule

Merupakan metode pembelajaran yang paling sederhana. Pembelajarannya yaitu memperbaiki nilai bobot ketika ada 2 neuron yang terhubung pada kondisi hidup pada saat yang sama maka bobot keduanya dinaikkan.

Perbaikan bobotnya adalah bobot yang diperoleh dari penjumlahan bobot sebelumnya dan perubahan bobot.

Dapat dirumuskan :

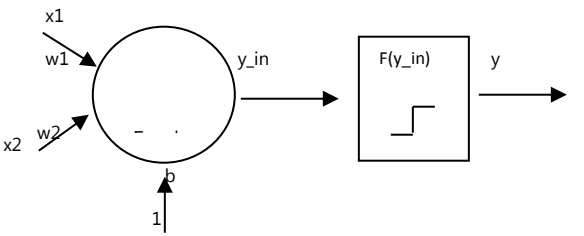
w_i *(baru)* = w_i *(lama)* + x_i * *y*

Dengan :

w_i = bobot data input ke-i

x_i = input data ke-i

y = output data



Gambar 24. Arsitektur jaringan Hebb

Algoritma :

1. inisialisasi semua bobot:

   w_{ij} = 0, dengan i=1,2,3,… n; dan j = 1,2,3,…m

2. untuk setiap pasangan input-output (s-t), ikuti langkah berikut :
   a. set input dengan nilai sama dengan vector input

      $x_i = s_i$ (i = 1,2,..n)

   b. set output dengan nilai sama dengan vector output

      $y_i = t_i$ (i = 1,2,..m)

   c. perbaiki bobot

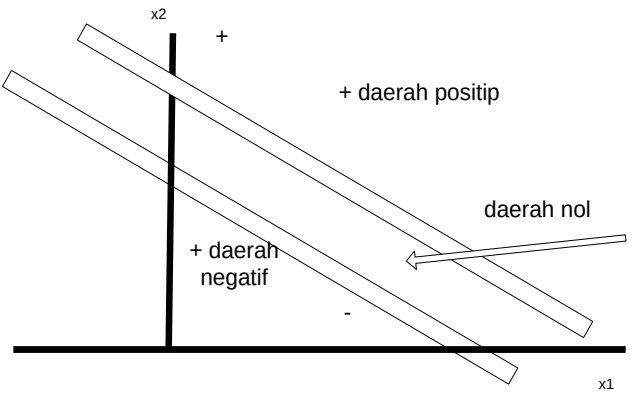
      w_{ij} *(baru)* = w_{ij} *(lama)* + x_i * *y*

      dengan nilai bias = 1

b. Perceptron

Biasanya digunakan untuk mengklasifikasikan suatu tipe pola tertentu yang dikenal dengan pemisahan cara linear. Perceptron merupakan jaringan lapis tunggal yang mempunyai bobot bisa diatur dan mempunyai ambang batas. Procedure pelatihannya dapat membuat output dari bobot menjadi konvergen. Perceptron memiliki 3 lapisan dari neuron-neuronnya yaitu unit sensor, asosiasi dan unit respon. Ketiga

unit ini menggunakan fungsi aktifasi yang berbeda, untuk unit sensor dan asosiasi menggunakan fungsi aktivasi biner sedangkan unit respon menggunakan fungsi aktivasi bipolar. Algoritma yang digunakan berfungsi mengatur parameter-parameter bebasnya melalui pembelajaran. Nilai threshold (Θ) pada fungsi aktivasi adalah non negatif



Gambar 25. Pembatasan linear dengan perceptron

Garis pemisah antara daerah positif dan daerah nol memiliki pertidaksamaan:

$$w_1x_1 + w_2x_2 + b > \Theta$$

garis pemisah antara daerah negative dengan daerah nol memiliki pertidaksamaan:

$$w_1x_1 + w_2x_2 + b < -\Theta$$

Algoritma :

1. Inisialisasi semua bobot dan bias (set semua bobot dan bias dengan nol)

   Set learning rate : α ($0 < \alpha \leq 1$) atau set semua dengan 1 agar sederhana
2. Selama kondisi bernilai false, ikuti langkah dibawah ini:
   a. Untuk setiap pasangan pembelajaran s-t, lakukan:
      - Set input dengan nilai yang sama dengan vector input

$$x_i = s_j$$

i. Hitung respon untuk unit output

$$y\_in = b + \sum_i x_i w_i$$

$$y = \begin{cases} 1, & \text{jika } y\_in > \Theta \\ 0, & \text{jika } -\Theta \leq y\_in \leq \Theta \\ -1, & \text{jika } y\_in < \Theta \end{cases}$$

ii. Perbaiki bobot dan bias jika terjadi error

Jika $y \neq t$ maka :

w_i (baru) = w_i (lama) + $\alpha * t * x_i$

b(baru) = b(lama) + $\alpha * t$

Jika tidak, maka :

w_i (baru) = w_i (lama)

b(baru) = b(lama)

b. Tes kondisi berhenti : jika tidak terjadi perubahan bobot pada (i) maka kondisi berhenti TRUE, jika masih terjadi perubahan maka FALSE

Algoritma ini dapat dipakai untuk input biner maupun bipolar dengan Θ tertentu juga bias yang dapat diatur. Bobot yang diperbaiki merupakan bobot yang berhubungan dengan input yang aktif serta bobot yang tidak menghasilkan nilai y yang benar.

Tujuan jaringan ini adalah mengklasifikasikan pola input kepada suatu kelas terterntu. Bila nilai output +1 maka dapat dikatakan bahwa input adalah anggota kelas tertentu. Namun jika menghasilkan -1 maka input bukan anggota dari kelas tertentu.

Nilai threshold untuk fungsi aktivasi untuk unit output selalu berharga positif Θ. Bentuknya diatur sedemikian rupa sehingga selalu menghasilkan keputusan daerah positif dan daerah negative.

c. Delta Rule

Delta rule mengubah bobot yang menghubungkan antara jaringan input ke unit output (y_in) dengan target (t) yang berfungsi meminimalkan error selama pelatihan.

Perbaikan bobotnya adalah :

$$\Delta w_i = \alpha (t - y\_in) * x_i$$

Dengan :

x = vektor input

y_in = input jaringan ke unit output y $\rightarrow y\_in = \sum_{i=1}^{n} x_i * w_i$

t = target output

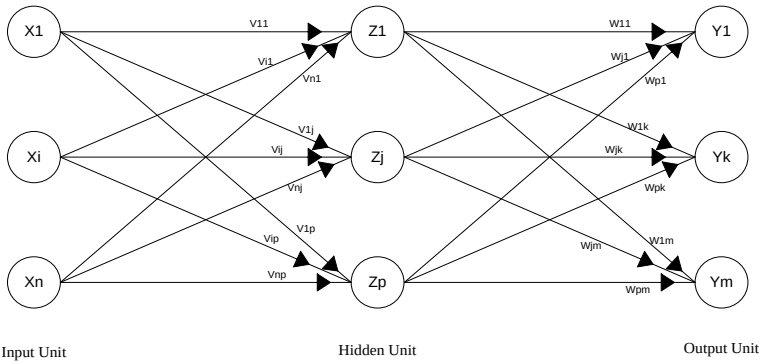
Nilai w yang baru diperoleh dari nilai w lama ditambah dengan Δw

$w_i = w_i + \Delta w_i$

d. Backpropagation

Adalah algoritma pembelajaran yang biasa digunakan perceptron banyak lapisan untuk merubah nilai bobot yang terhubung dengan neuron pada lapisan tersembunyi. Algoritmanya memakai eror output untuk merubah nilai bobot dengan arah mundur (*backward*). Untuk menghasilkan eror output tersebut harus melalui perambatan maju (*forward*) dengan terlebih dahulu mengaktifkan fungsi aktivasi sigmoid.

$$f(x) = \frac{1}{1 + e^{-x}}$$



Gambar 26. Arsitektur jaringan Backpropagation

Keterangan:

X_i Unit *input* ke-i

Z_j *Hidden* unit ke-j

Y_k Unit *output* ke-k

V_{ij} Bobot antara unit *input* ke-i dengan hidden unit ke-j

W_{jk} Bobot antara hidden unit ke-j dengan unit *output* ke-k

Pada dasarnya, pembelajaran atau pelatihan dengan metode *Backpropagation* terdiri atas tiga langkah utama, yaitu:

1. Data dimasukkan ke unit *input* (*feedforward*).
2. Perhitungan dan propagasi balik dari *error*.
3. Pembaharuan (*adjusment*) bobot.

Untuk selengkapnya, langkah-langkah pelatihan adalah sebagai berikut :

Step 1 Inisialisasi nilai bobot

Nilai bobot dapat diset dengan sembarang angka (acak) antara -0.5 dan 0.5

Umpan maju (*feedforward*)

Step 2 Setiap unit *input* (X_i, i = 1,...,n) menerima sinyal *input* dan menyebarkannya pada seluruh *hidden* unit melalui Vij.

Step 3 Setiap *hidden* unit (Z_j, j = 1,...,p) menghitung sinyal-sinyal *input* yang sudah berbobot dan menggunakan fungsi aktivasi yang telah ditentukan untuk menentukan sinyal *output* dari *hidden* unit tersebut.

Step 4 Setiap unit *output* (Y_k, k = 1,...,m) menghitung sinyal-sinyal *input* yang sudah berbobot dan menggunakan fungsi aktivasi yang telah ditentukan untuk menentukan sinyal *output* dari unit *output* tersebut lalu mengirim sinyal *output* ini ke seluruh unit pada unit *output*.

Propagasi *error* (*backpropagation of error*)

Step 5 Setiap unit *output* (Y_k, k = 1,...,m) memiliki *target output* yang sesuai dengan *input training*. Hitung kesalahan antara target *output* dengan *output* yang dihasilkan jaringan sehingga menghasilkan faktor koreksi *error* (δ_k). Faktor koreksi *error* digunakan untuk menghitung koreksi *error* (ΛW_{jk}) untuk memperbaharui W_{jk} dan dikirimkan ke *hidden* unit.

Step 6 Setiap *hidden* unit (Z_j, j = 1,...,p) menghitung bobot yang dikirimkan *output* unit.dan menghasilkan faktor koreksi *error* (δ_j). Faktor koreksi *error* digunakan untuk menghitung koreksi *error* (ΛV_{ij}) untuk memperbaharui V_{ij} dan dikirimkan ke unit *input*.

Pembaharuan bobot (*adjustment*)

Step 7 Setiap unit *output* (Y_k, k = 1,...,m) memperbaharui bobotnya dari setiap *hidden* unit

$$W_{jk}(\text{baru}) = W_{jk}(\text{lama}) + \Lambda W_{jk}$$

Demikian pula setiap hidden unit (Zj, j = 1,...,p) memperbaharui bobotnya dari setiap unit *input*

$$V_{ij}(\text{baru}) = V_{ij}(\text{lama}) + \Lambda V_{ij}$$

Step 8 Memeriksa *stop condition*

Untuk menentukan *stopping condition* dengan cara membatasi error. Digunakan metode *Mean Absolute Persentage Error*

(MAPE) untuk menghitung rata-rata *error* antara *output* yang dikehendaki (target *output*) dengan *output* yang dihasilkan oleh jaringan.

Setelah pelatihan selesai, jaringan akan dapat mengenali pola-pola data yang dilatihkan. Cara mendapatkan *output* adalah dengan cara melakukan proses umpan maju (*step* 2 sampai *step* 4).

e. Heteroassociative Memory

Merupakan jaringan yang bobotnya telah ditentukan sehingga jaringan dapat menyimpan kumpulan pengelomokan pola. Masing-masing kelompok merupakan pasangan vector (s(p), t(p)) dengan p = 1,2,..P. Setiap vector s(p) mempunyai n komponen, dan setiap vector y(p) memiliki m komponen. Algoritma yang biasa digunakan oleh jaringan Heteroassociative memory ini adalah Hebb dan delta rule. Output yang dihasilkan adalah vektor output yang sesuai dengan vektor input yang merupakan salah satu vektor s(p) atau vector lain diluar s(p).

Algoritma :

1. Inisialisasi bobot dengan menggunakan Hebb rule atau delta rule
2. Untuk setiap vector input, maka :
   a. Set input dengan nilai sama dengan vector input
   b. Hitung input jaringan ke unit output :

$$y\_in_j = \sum_i x_j * w_{ij}$$

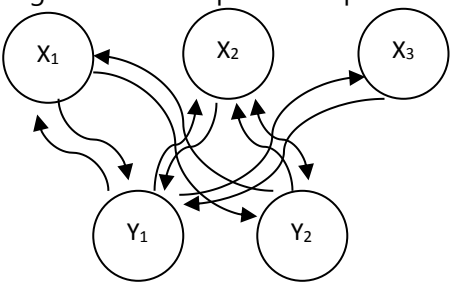
   c. Tentukan aktivasi dari setiap unit output

$$y = \begin{cases} 1, & \text{jika } y\_in_j > 0 \\ 0, & \text{jika } y\_in_j = 0 \\ -1, & \text{jika } y\_in_j < 0 \end{cases}$$

Untuk target bipolar

f. Bidirectional Associative Memory (BAM)

BAM adalah model jaringan yeng mempunayi dua lapisan yang terhubung dengan lapisan lainnya. Pada model ini bisa terjadi hubungan timbal balik antara lapisan input dan outputnya. Matriks bobot yang menghubungkan neuron dari output ke input sama dengan transpose matrik bobot yang menghubungkan neuron input ke output.



Gambar 27. Arsitektur jaringan BAM

Pada arsitektur tersebut terdapat 3 neuron pada lapisan input dan 2 neuron pada lapisan output.

Ada 2 macam BAM, yaitu :

1. BAM Diskret

   Ada 2 tipe dat yaitu iner dan bipolar. Matriks bobot awal dibuat sedemikian rupa sehingga dapat menyimpan pasangan vector input dan vector output.

   Matriks bobot pada vector input biner:

$$W_{ij} = \sum_{p} (2 * s_i(p) - 1)(2 * t_i(p) - 1)$$

   untuk fungsi aktivasi:

   pada lapisan input :

$$x_i = \begin{cases} 1, & \text{jika } x\_in_i > 0 \\ x_i, & \text{jika } x\_in_i = 0 \\ 0, & \text{jika } x\_in_i < 0 \end{cases}$$

Untuk lapisan output :

$$y_j = \begin{cases} 1, & \text{jika } y\_in_j > 0 \\ y_j, & \text{jika } y\_in_j = 0 \\ 0, & \text{jika } y\_in_j < 0 \end{cases}$$

Matriks bobot pada vector input bipolar:

$$W_{ij} = \sum_p (s_i(p) * t_i(p))$$

untuk fungsi aktivasi pada lapisan input :

$$x_i = \begin{cases} 1, & \text{jika } x\_in_i > \Theta \\ x_i, & \text{jika } x\_in_i = \Theta \\ 0, & \text{jika } x\_in_i < \Theta \end{cases}$$

Untuk lapisan output :

$$y_j = \begin{cases} 1, & \text{jika } y\_in_j > \Theta \\ y_j, & \text{jika } y\_in_j = \Theta \\ 0, & \text{jika } y\_in_j < \Theta \end{cases}$$

Input hasil olahan sama dengan nilai thresholdnya, maka fungsi aktivasi akan menghasilkan nilai sama dengan nilai sebelumnya.

2. BAM Kontinu

   BAM kontinu akan mentranformasikan input secara lebih halus dan kontinu ke lapisan outputdengan nilai range 0,1. Fungsi aktivasi yang digunakan adalah fungsi aktivasi sigmoid.

g. Learning Vector Quantization (LVQ)

LVQ merupakan metode melakukan pembelajaran pada lapisan kompetitif yang terawasi yang mana tiap lapisannya secara otomatis mengklasifikasikan vektor input. Lapisan kompetitif akan secara otomatis belajar untuk mengklasifikasikan vektor-vektor input. Kelas-kelas yang didapatkan sebagai hasil dari lapisan kompetitif ini hanya bergantung pada jarak antara

vektor-vektor input. Apabila dua buah input nilainya mendekati sama maka lapisan kompetitif akan meletakkannya pada kelas yang sama.

h. Extreme Learning Machine (ELM)

Extreme Learning Machine merupakan metode pembelajaran baru dari jaringan syaraf tiruan. Metode ini pertama kali diperkenalkan oleh Huang (2004). *ELM* merupakan jaringan syaraf tiruan feedforward dengan single *hidden* layer atau biasa disebut dengan *Single Hidden Layer Feedforward neural Networks (SLFNs)* Metode pembelajaran *ELM* dibuat untuk mengatasi kelemahan-kelemahan dari jaringan syaraf tiruan *i* terutama dalam hal *learning speed.* Huang *et al* mengemukaan dua alasan mengapa JST feedforward lain mempunyai *leraning speed* rendah, yaitu :

1. menggunakan *slow gradient based learning algorithm* untuk melakukan *training*.
2. semua parameter pada jaringan ditentukan secara *iterative* dengan menggunakan metode pembelajaran tersebut.

Pada *ELM* parameter-parameter seperti *input weight* dan *hidden bias* dipilih secara *random*, sehingga *ELM* memiliki *learning speed* yang cepat dan mampu menghasilkan *good generalization performance.* Metode *ELM* mempunyai model matematis yang berbeda dari jaringan syaraf tiruan *feedforward*. Model matematis dari *ELM* lebih sederhana dan efektif. Berikut model matrematis dari *ELM*.

Untuk *N* jumlah sample yang berbeda (Xi , ti)

$$Xi = [\, Xi_1 , Xi_2 \ldots., Xi_n \,]^T \in R^n,$$
$$Xt = [\, Xt_1 , Xt_2 \ldots., Xt_n \,]^T \in R^n,$$

Standart *SLFNs* dengan jumlah *hidden* nodes sebanyak *N* dan *activation function g* (*x*) dapat digambarkan secara matematis sebagai berikut :

$$\sum_{i=1}^{N} \beta igi\,(xj) = \sum_{i=1}^{N} \beta ig\,(W_{i} \bullet X_{j} + b_{i}) = o_{j}$$

Dimana :

J = 1,2,..., *N*

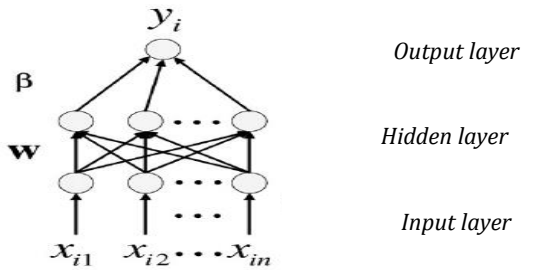
$w_i = (w_{i1}, w_{i2}, \ldots, w_{in})^T$ = merupakan vektor dari weight yang menghubungkan ***i*** th *hidden* nodes dan input nodes.

$\beta_i = (\beta_{i1}, \beta_{i2}, \ldots, \beta_{in})^T$ = merupakan *weight vector* yang menghubungkan ***i*** th *hidden* dan *output nodes.*

b_i = *threshold* dari *i* th *hidden nodes.*

$w_i x_j$ = merupakan *inner* produk dari w_i dan x_j

Konfigurasi sederhana algoritma *ELM* dapat dijelaskan pada gambar dibawah ini:



Gambar 28. Arsitektur *ELM* [1]

Proses Extreme Learning Machine

- Langkah-langkah *ELM* dapat dirinci sebagai berikut :
  1. Pembagian data *training* dan *testing*
  2. *Training ELM*
  3. *Testing ELM*
  4. Analisa hasil peramalan

Pembagian data menjadi data *training dan testing*. Proses tersebut *testing* mutlak diperlukan pada proses peramalan dengan *ELM*. Adapun proses pertama yang harus adalah proses pelatihan tujuan dari proses ini adalah untuk mendapatkan *input weight , bias* dan *output weight* dengan tingkat kesalahan yang rendah. Selanjutnya proses *training* untuk mengembangkan model dari *ELM*, Oleh karena itu data dibagi menjadi dua yaitu *data training* dan *data testing*. Menurut Zhang (1997) beberapa peneliti membagi data *training* dan *testing* dengan komposisi sebagai berikut :

- Data *training* sebanyak 80% dari total data.
- Data *testing* 20% dari total data.

Menentukan fungsi aktivasi dan jumlah *hidden neuron* Pada proses training jumlah *hidden neuron* dan fungsi aktivasi dari *ELM* harus ditentukan terlebih dahulu. Pada tugas akhir ini dilakukan uji coba dengan menggunakan fungsi aktivasi *log sigmoid* atau *tan sigmoid* karena kedua fungsi tersebut yang paling sering digunakan pada permasalahan *forecasting*, serta fungsi transfer *purelin* karena data yang diramalkan bersifat *stasioner*. Hal ini mengacu pada tulisan Zhang (1997) bahwa fungsi *transfer linier* memiliki kelemahan pada pola data yang memiliki trend. Untuk jumlah *hidden neuron* menurut Sun et al (2008) *ELM* menghasilkan *output* peramalan yang stabil dengan jumlah *hidden neuron* 0-30. Namun jika *output* yang didapatkan dari *ELM* kurang optimal, maka akan digunakan *alternative* fungsi *transfer* yang lain atau merubah jumlah *hidden neuron*.

Menghitung *input weight, bias of hidden neuron* dan *weight* o*utput* dari proses pelatihan *ELM* adalah *input* dan *output weight* serta *bias* dari *hidden neuron* dengan tingkat kesalahan rendah yang diukur dengan *MSE* dan *MAPE*. *Input weight* ditentukan secara *random*, sedangkan *output weight* merupakan *invers* dari *matrix hidden layer* dan *output*. Secara matematis dapat ditulis sebagai berikut :

$$H = (w_i,.....,w_N, b_i,....., b_N, x_i,....., x_N) \quad (3.3)$$

$$g(w_1 \bullet x_1 + b_1) \quad ..... \quad g(w_N \bullet x_1 + b_N)$$
$$\vdots$$
$$= g(w_1 \bullet x_N + b_1) ..... \quad g(w_N \bullet x_N + b_N)$$

$$\boldsymbol{\beta} = \begin{matrix} \boldsymbol{\beta}^T \\ 1 \\ \vdots \\ \boldsymbol{\beta}^T \\ N \end{matrix}$$

$$\boldsymbol{T} = \begin{matrix} t^T \\ 1 \\ \vdots \\ t^T \\ N \end{matrix}$$

$$\boldsymbol{\beta} = \boldsymbol{H}^{\tau}\boldsymbol{T}$$

H diatas adalah *hidden layer output matrix* $\boldsymbol{g}$ $(\mathbf{w_i} \bullet \mathbf{x_i} + \mathbf{b_1})$ menunjukkan *output* dari *hidden neuron* yang berhubungan dengan *input* i x . β merupakan *matrix* dari *output weight* dan T *matrix* dari target atau output . Pada *ELM input weight dan hidden bias* ditentukan secara acak, maka output weight yang berhubungan dengan *hidden layer*.

Dalam proses testing *ELM* Berdasarkan *input weight* dan *output weight* yang didapatkan dari proses *training*, maka tahap selanjutnya adalah melakukan peramalan dengan *ELM*. Data yang digunakan adalah data *testing* sebanyak 20% dari data. Pada tahap ini data *input* dinormalisasi terlebih dahulu dengan *range* dan rumus normalisasi yang sama dengan data *training*. Secara otomatis *output* dari proses ini juga harus melalui proses denormalisasi.[1].

i. Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi - fungsi linier dalam sebuah ruang fitur (feature space) berdimensi tinggi, dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan learning bias yang berasal dari teori pembelajaran statistik. Prinsip dasar SVM adalah pengklasifikasi linier, dan selanjutnya dikembangkan agar dapat bekerja pada permasalahan nonlinier. dengan memasukkan konsep kernel trick pada ruang kerja berdimensi tinggi. Perkembangan ini memberikan minat penelitian di bidang pengenalan pola untuk investigasi potensi kemampuan SVM secara teoritis maupun dari segi aplikasi [9].

Linearly separable data merupakan data yang dapat dipisahkan secara linear. Misalkan xi∈{ xn, ..., x1 } adalah dataset dan yi ∈{+1,-1} adalah label kelas dari data xi. Fungsi yang digunakan untuk memisahkan kelas adalah dengan menggunakan fungsi linear, dimana fungsi tersebut didefenisikan sebagai berikut :

$$g(x) = sign\ (f(x))$$

$$\text{dengan } f(x)=(w^T x+b)$$

dimana,

w= normal bidang

b= posisi bidang relatif terhadap pusat koordinat

Untuk pencarian bidang pemisah terbaik dengan nilai margin terbesar dapat dirumuskan menjadi masalah optimasi constraintSVM untuk kasus klasifikasi

Linear dalam primal space, yaitu :

$$min\ \frac{1}{2}|w|^2$$

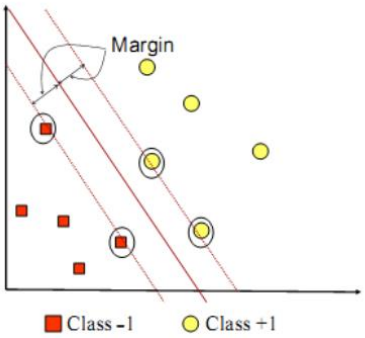
$$s.t.\ y_i(x_i.w + b) - 1 \geq 0$$

dimana,
xi= data input
yi= output dari xi

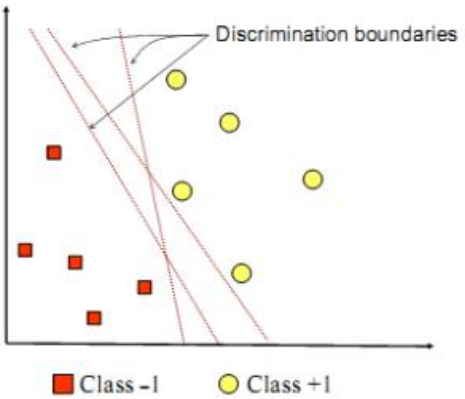
Pada persamaan optimasi constraintuntuk meminimalkan fungsi objektif 1/2||w||2 atau memaksimalkan wTw yaitu dengan memperhatikan pembatas yi(xi· w + b)≥ 1. Bila output data yi= +1, maka pembatas menjadi (xi.w + b) ≥ 1, sebaliknya yi = -1, maka pembatas menjadi (xi.w + b) ≤ 1. Dalam beberapa kasus, terdapat beberapa data yang tidak dapat diklasifikasikan secara benar (infeasible), maka dapat dinyatakanmelalui persamaan berikut,

$$min\ \frac{1}{2}|w|^2 + C\left(\sum_{i=1}^{n} \xi^i\right)$$
$$s.t.\ y_i(w.x_i + b) \geq 1 - \xi_i$$

Nilai C (Complexity) adalah nilai yang dipilih sebelum dilakukan optimasi dengan proses Quadratic Programming. Nilai C memiliki rentang antara nol sampai positif tak hingga (0 < C < ∞). Tujuan adanya nilai C (Complexity) adalah untuk meminimalkan error dan memperkecil nilai slack variabel. Jika nilai C mendekati nol, maka lebar margin pada bidang pembatas menjadi maksimum dan jumlah data yang dilatih yang berada dalam margin atau yang ada posisi yang salah tidak akan dipedulikan. Hal ini berarti akan mengurangi tingkat akurasi pada proses training, sehingga mengakibatkan data uji tidak dapat diklasifikasikan dengan baik.



Gambar 29. SVM Menemukan *Hyperplane* terbaik yang meisahkan kedua kelas -1 dan +1



Gambar 30. Hyperplane terbentuk diantara class-1 dan +1 [6]

Hyperplane pemisah terbaik antara kedua class dapat ditemukan dengan mengukur margin hyperplane tsb. dan mencari titik maksimalnya. Margin adalah jarak antara hyperplane tersebut dengan pattern terdekat dari masing-masing class. Pattern yang paling dekat ini disebut sebagai support vector. Garis solid pada gambar menunjukkan hyperplane yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua class, sedangkan titik merah dan kuning yang berada dalam lingkaran hitam adalah support vector.

Usaha untuk mencari lokasi hyperplane ini merupakan inti dari proses pembelajaran pada SVM.

E. Pembelajaran Tak Terawasi (Unsupervised Learning)

Pada metode ini tidak memerlukan target output karena tidak dapat ditentukan hasil yang seperti apakah yang diharapkan selama proses pembelajaran. Nilai bobot tergantung pada nilai input yang diberikan. Pembelajaran ini cocok untuk pengelompokan pola.

Jaringan Kohonen merupakan JST dengan metode pembelajaran tak terawasi. Ditemukan oleh Prof. Teuvo Kohonen pada 1982. Jaringan ini, semua neuron yang ada pada tiap lapisan akan menyusun dirinya berdasarkan nilai inputnya. Selama proses ini, cluster yang memiliki bobot cocok dengan pola input akan terpilih sebagai hasil. Sehingga neuron yang terpilih tersebut memperbaiki bobot neuron yang lain.

5.3. Soal Latihan

1. Bagaimanakah arsitektur dari jaringan lapis tunggal?
2. Bagaimanakah algoritma dari backpropagation?
3. Bagaimanakah algoritma dari JST jaringan kohonen?
4. Bandingkan perbedaan jaringan syaraf biologis dengan jaringan syaraf tiruan
5. Jelaskan konsep dasar dalam JST untuk memecahkan permasalahan!
6. Buat rangkuman tentang JST!
7. Sebutkan contoh aplikasi yang menggunakan JST!

BAB VI.

ALGORITMA GENETIKA

6.1. Tujuan Instruksional Khusus

- Mahasiswa mampu menjelaskan Algoritma Genetika
- Mahasiswa mampu menerapkan algoritma genetika pada pengoptimalan sebuah masalah

6.2. Materi Perkuliahan

A. Algoritma Genetika

Algoritma genetika merupakan algoritma pencarian dengan heuristic berdasarkan mekanisme evolusi biologis. Adanya keberagaman dalam kromosom sangat mempengaruhi produksi dan kemampuan makhluk hidup untuk bertahan hidup.

Dapat pula dikatakan bahwa Algoritma genetik adalah algoritma pencarian yang berdasarkan pada mekanisme sistem natural yakni genetik dan seleksi alam. Dalam aplikasi algoritma genetik, variabel solusi dikodekan kedalam struktur string yang merepresentasikan barisan gen, yang merupakan karakteristik dari solusi problem. Ada beberapa kondisi yang mempengaruhi proses evolusi, yaitu :

1. Kemampuan reproduksi
2. Populasi makhluk hidup
3. Keanekaragaman makhluk hidup dalam sebuah populasi
4. Perbedaaan kemampuan untuk bertahan

Hal-hal yang harus dilakukan untuk menggunakan Algoritma genetika yaitu :

1. Mendefinisikan individu, dimana individu menyatakan sebuah solusi dari permasalahan yang akan dipecahkan
2. Mendefinisikan nilai fitness yang menjadi ukuran baik tidaknya solusi atau individu
3. Menentukan proses pembangkitan populasi awal.
4. Menentukan proses seleksi
5. Menentukan proses perkawinan silang dan mutasi gennya.

Algoritma genetika dikembangkan oleh John Holland dari Universitas Michigan pada tahun 1975 yang menyatakan bahwa setiap masalah yang berbentuk adaptasi dapat berbentuk alami maupun buatan dan diformulasikan dalam terminologi genetika. Sedangkan pendapat Goldberg pada tahun 1989, Setelah beberapa generasi maka algoritma genetik akan konvergen pada kromosom terbaik, yang diharapkan merupakan solusi optimal.

Lain halnya dengan Gen dan Cheng pada tahun 1997 yang mengemukakan bahwa, Kromosom-kromosom berevolusi dalam suatu proses iterasi yang berkelanjutan yang disebut generasi. Pada setiap generasi, kromosom dievaluasi berdasarkan suatu fungsi evaluasi.

Teknik pencariannya dilakukan bersamaan dengan berapa besar solusi yang bisa dicapai dan dikenali, hal ini disebut dengan populasi. Kromosom adalah istilah untuk individu yang ada pada satu populasi. Dari populasi akan membentuk sebuah generasi. Untuk mengukur jumlah kromosom pada sebuah populasi digunakan fitness. Yang nantinya akan menunjukkan kualitas kromosom tersebut. Generasi berikutnya disebut sebagai offspring ysng terbentuk oleh 2 buah kromosom dengan hasil penyilangan.

Setelah melalui proses beberapa generasi, maka algoritma genetika akan konvergen ke kromosom yang mempunyai nilai

terbaik. Cara merepresentasikan permasalahan dalam kromosom merupakan suatu hal yang penting dalam algoritma genetik

Ada beberapa model representasi kromosom yang dapat dipergunakan untuk menyelesaikan suatu masalah, salah satunya adalah *operation-based representation*. Prinsip dari *Operation-Base Representation* adalah semua operasi pada suatu pekerjaan akan dikodekan dengan simbol yang sama, kemudian diinterpretasikan menurut urutannya dalam sequence

B. Komponen-Komponen Utama Algoritma Genetika

Pada teknik penyandian ini meliputi penyandian gen dari kromosom, satu gen mewakili satu buah vareabel. Gen ini adalah bagian dari kromosom itu sendiri.

Gen dapat direpresentasian dalam berbagai macam bentuk seperti string, bit, pohon, array, bilangan real, daftar aturan, elemen permutasi elemen program dan elemen lain yang bisa diimplementasikan dalam algoritma genetika.

Prosedure Inisialisasi

Sebelum melakukan inisialisasi terhadap kromosom dilakukan, maka yang terlebih dahulu adalah menentukan ukuran populasi. Ukuran populasi ini bergantung pada masalah yang akan diselesaikan serta jenis operator genetika yang akan diimplementasikan. Inisialisasi dilakukan secara acak tetapi tetap memperhatikan domain solusi dan kendala yang ada.

Fungsi Evaluasi

Fungsi evaluasi merupakan dasar untuk proses seleksi. Langkah-langkahnya yaitu string dikonversi ke parameter fungsi, fungsi objective-nya dievaluasi, kemudian mengkonvert fungsi objektif tersebut ke dalam *fitness*.

Yaitu yang perlu dilakukan saat melakukan proses evaluasi adalah melakukan evaluasi objektif dan mengkonversi fungsi objektif

ke dalam fungsi fitness. Nilai dari fungsi fitness adalah positif. Jika nilai fungsi objektif adalan negatif maka harus ditambah satu konstanta agar nilai fungsi fitness tidak berupa negatif.

Output dari fungsi *fitness* dipergunakan sebagai dasar untuk menseleksi individu pada generasi berikutnya

Seleksi

Tujuan dari adanya seleksi adalah memberi peluang yang besar bagi anggota populasi untuk melakukan reproduksi. Ada beberapa macam metode seleksi yaitu:

- *Rank Based fitness assigment*¸ populasi diurutkan menurut nilai objektif
- *Roulette wheel selection,* individu dipetakan dalam segmen garis secara terurut hingga memiliki ukuran yang sama dengan fitnessnya
- *Stochostic universal sampling,* memiliki nilai bias 0 dan penyebaran yang minimum. Untuk pengurutan fitnessnya sama seperti roulette wheel selection
- *Local selection,* individu yang berada pada konstrain tertentu disebut dengan nama lokal
- *Truncation selection,* pada seleksi pemotongan ini nampak seperti buatan
- *Tournament selection*, pada seleksi ini ditetapkan suatu nilai untuk individu yang dipilih secara acak dari populasi.

Seleksi ini menentukan individu mana yang akan dipilih untuk dirokombinasi dan bagaimana membentuk offspring dari individu tersebut.

Ada beberapa definisi yang bisa digunakan untuk perbandingan dengan beberapa metode yang akan digunakan, yaitu:

1. Selective pressure.
   Probabilitas dari individu terbaik dibandingkan dengan rata-rata probabilitas dari semua individu yang telah diseleksi

2. Bias
   Perbedaan absolut antara fitness individu dengan probabilitas repropduksi
3. Spread
   Range nilai kemungkinan offspring
4. Loss of diversity
   Proporsi dari individu dalam populasi yang tidak terseleksi
5. Selection intensity
   Nilai fitness rata-rata yang diinginkan setelah di seleksi
6. Selection variance
   Variansi yang diinginkan setelah proses seleksi

C. Operator Genetika

Operator genetik dipergunakan untuk mengkombinasi (modifikasi) individu dalam aliran populasi guna mencetak individu pada generasi berikutnya.

Operator genetika ada 2 macam yaitu:

a) Operator untuk rekombinasi, Terdiri dari :

- **Rekombinasi bernilai real**
  - *Rekombinasi diskret*, akan menukar nilai variabel antar kromosom induk
  - *Rekombinasi intermediate*, adalah metode kombinasi yang hanya digunakan untuk vareabel real

  Anak dihasilkan menurut aturan :

  Anak = induk1 + alpha (induk2 – induk1)

  Alpha adalah skala yang dipilih secara acak.
  - *Rekombinasi garis*, nilai alpha untuk semua vareabel sama karena metode ini hampir sama dengan rekombinasi menengah
  - *Rekombinasi garis yang diperluas*
- **Rekombinasi bernilai biner** (*crossover*)
  - *Crossover satu titik,* posisi penyilangan dengan panjang kromosom diseleksi secara acak.

- *Crossover banyak titik,* posisi penyilangan dengan panjang kromosom diseleksi secara acak tatapi tidak boleh ada posisi yang sama dan harus diurutkan naik.
- *Crossover seragam,* sebuah lokasi mempunyai potensi untuk penyilangan. Induk yang akan memberi sebuah bit akan dipilih acak dengan probabilitas yang sama

➢ **Rekombinasi bernilai biner dengan permutasi**
Pada penyilangan ini, kromosom anak diambil dengan cara memilih sub barisan dari satu induk dengan menjaga urutannya.

b) Mutasi

Mutasi ini berperan menggantikan gen yang hilang dari populasi akibat seleksi yang memungkinkan untuk memunculkan kembali gen yang tidak muncul pada inisialisasi populasi.

➢ Mutasi bernilai real, ukuran langkahnya sulit ditentukan bila ukurannya kecil sering mengalami kesulitan tetapi ukuran yang lebih besar dapat berjalan lebih cepat
➢ Mutasi bernilai biner, untuk mendapatkan mutasi biner yaitu mengganti satu atau beberapa nilai gen dari kromosom.

Penentuan Parameter

Parameter yang digunakan adalah parameter kontrol algoritma genetika, antara lain ukuran populasi, peluang crossover, dan peluang mutasi. Nilainya ditentukan sesuai dengan masalah yang dihadapi.

D. Proses Algoritma Genetika

Algoritma genetika sederhana dapat ditulis secara sederhana:

1. Generasi awal=0
2. Inisialisasi populasi awal secara acak
3. Evaluasi nilai fitness pada tiap individu

4. Melakukan tahap berikut sampai mendapatkan nilai generasi maksimum :
   - generasi = generasi + 1
   - menyeleksi populasi untuk memperoleh kandidat induk
   - crossover pada generasi
   - mutasi pada generasi
   - evaluasi tiap fitness pada setiap individu di generasi
   - buat populasi baru: p(generasi)= {p(generasi-1) yang bertahan, p'(generasi)}

Metode-metode pada algoritma genetika sederhana antara lain:

a. Seleksi dengan Roda Roulette

Metode ini paling sering digunakan, karena seleksi bertujuan memberi peluang besar pada tiap organisme untuk bereproduksi.

Algoritmanya:

1. Hitung total fitness (F)
2. Hitung finess relatif
3. Hitung fitness komulatif
4. Pilih induk yang jadi kandidat untu crossover dengan membangkitkan random

b. Crossover

Penyilangan dilakukan jika dua buah kromosom menghasilkan kromosom anak (*offspring*) yang mewarisi sebagian sifat kromosom induk. Bentuk ini sering digunakan jika kromosom berbentuk string biner. Untuk menentukan posisi persilangan maka pilih bilangan secara acak.

Bagian terpenting dari crossover adalah sebuah peluang crossover, karena menunjukkan rasio dari anak yang dihasilkan oleh setiap generasi pada sebuah populasi.

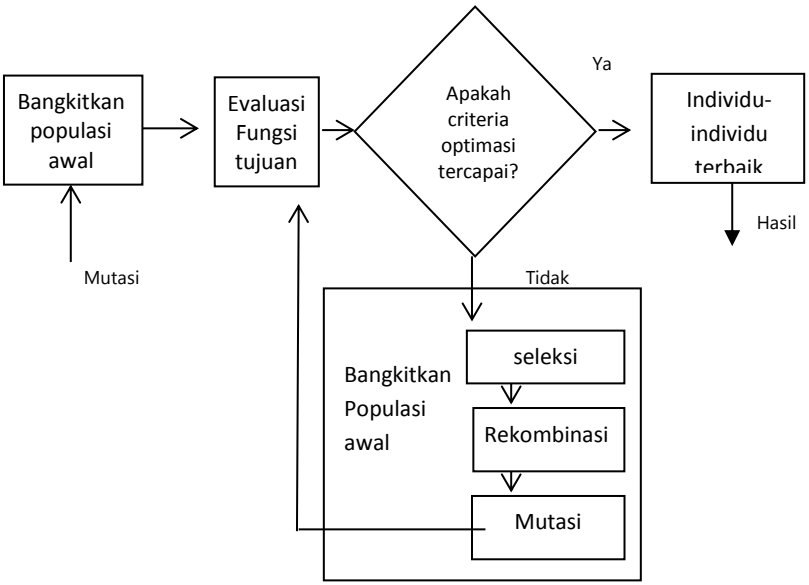
c. Mutasi

Mutasi menciptakan individu baru dengan melakukan modifikasi satu atau lebih gen dalam individu yang sama. Sehingga mutasi akan meningkatkan variasi populasi.

Mutasi merubah secara acak nilai bit pada posisi tertentu. Kemudian mengganti nilai bit 1 dengan 0, atau sebaliknya. Pada mutasi akan mungkin muncul kromosom baru yang belum muncul pada populasi awal.

Ada bagian yang juga penting pada mutasi, yaitu peluang crossover karena menunjukkan jumlah total gen pada populasi yang mengalami mutasi.

Untuk menentukan mutasi, harus terlebih dahulu menghitung jumlah total gen pada populasinya. Kemudian bangkitkan bilangan random yang menentukan posisi mana yang akan dimutasi.



Gambar 31. Diagram alir algoritma genetika sederhana

Contoh penyelesaian kasus menggunakan Algoritma Genetika pada penjadwalan mata kuliah :

Dibawah ini adalah bagaimana menyelesaikan permasalahan penjawalan dengan menggunakan metode GA secara manual.

- Pengkodean setiap data (Mata Kuliah, Ruang, Kelas, dan Waktu)

Kode Mata Kuliah

Kode Mata Kuliah	Mata Kuliah
UNG 110	Bahasa Inggris
TIF 115	Basis Data I
TIF 118	Metode Numerik

Kode Kelas

Kode Kelas	Kelas	Id Dosen
K1A	A	1
K1B	B	2
K1C	C	2
K2A	A	3
K2B	B	3
K2C	C	4
K3A	A	5
K3B	B	5
K3C	C	6

Kode Ruang

Kode Ruang	Ruang
R01	F108
R02	F201
R03	F303
R04	F403
R05	F401
R06	F402

Kode Waktu

Index Waktu	Hari	Waktu
T6	Jum'at	07.30-08.30
T7	Jum'at	09.00-11.00
T5	Kamis	07.00-09.30
T4	Rabu	09.30-12.00
T3	Rabu	07.00-09.30
T2	Selasa	09.30-12.00
T1	Selasa	07.00-09.30

- Menginisiasi kromosom

  Semua pengkodean di atas di inisiasikan secara random

TIF115 K2A R1T2	TIF118 K3A R3T2	UNG110 K1A R1T2
TIF115 K2B R5T2	TIF118 K3B R4T1	UNG110 K113 R3T4
TIF115 K2C R3T2	TIF118 K3C R2T2	UNG110 K1C R1T4
UNG110 K1B R3T6	TIF118 K3B R1T6	TIF115 K2B R1T6
TIF118 K3B R4T1	TIF115 K2C R3T2	UNG110 K1A R1T4
TIF115 K2A R2T3	TIF118 K3A R1T3	UNG110 K1B R1T5
TIF118 K3B R4T4	TIF115 K2C R4T4	UNG110 K1A R3T1
UNG110 K1B R2T2	TIF118 K3A R1T6	TIF115 K2C R3T4
TIF115 K2B R3T2	TIF118 K3B R5T1	UNG110 K1A R3T3

- Menghitung nilai fitnes

  Batasan yg digunakan :

  - Dosen tidak boleh dijadwalkan lebih dari dari satu kali pada waktu yang bersamaan
  - Satu kelas dan ruang tidak boleh dijadwalkan lebih dari satu kali pada waktu yang bersamaan

$$F = \frac{1}{1 + (\sum BD + \sum BK + \sum BR + \sum \boldsymbol{BT})}$$

Keterangan :
BD = Banyaknya bentrok dosen & mata kuliah
BK = Banyaknya bentrok kelas perkuliahan
BR = Banyaknya bentrok ruang yang digunakan
BT = Banyaknya *bentrok waktu yang digunakan*

Nilai Fitnes
Fitnes 2=1/1+(0+0+0+0)=0
Fitnes 1=1/1+(0+0+1+1)=0,33
Fitnes 3=1/1+(0+0+0+0)=1
Fitnes 4=1/1+(0+0+1+1)=0,33
Fitnes 5=1/1+(0+0+0+0)=1
Fitnes 6=1/1+(0+0+0+0)=1
Fitnes 7=1/1+(0+0+1+1)=0,33
Fitnes 8=1/1+(0+0+0+0)=1
Fitnes 9=1/1+(0+0+1+1)=0,33

- Seleksi

Total Nilai Fitnes

Kromosom	Nilai Fitnes
1	0,33
2	1
3	1
4	0,33
5	1
6	1
7	0,33
8	1
9	0,33
Total nilai fitnes	6,32

Probabilitas tiap kromosom

Kromosom	Probabilitas
1	0,33/6,32=0,05
2	1/6,32=0,16
3	1/6,32=0,16
4	0,33/6,32=0,05
5	1/6,32=0,16
6	1/6,32=0,16
7	0,33/6,32=0,05
8	1/6,32=0,16
9	0,33/6,32=0,05
Total Probabilitas	1

Menempatkan masing-masing kromosom pada interval nilai [0-1]

Kromosom	Interval nilai
1	0-0,05
2	0,06-0,21
3	0,22-0,37
4	0,38-0,42
5	0,43-0,58
6	0,59-0,74
7	0,75-0,79
8	0,80-0,95
9	0,96-1

Hasil seleksi adalah [0,2 , 0,25 , 0,15 ,0,15 , 0,01 , 0,75 , 0,09, 0,5 , 0,75]

Jika R[K] < C[K] maka kromosom 1 sebagai induk

Syarat C[K-1] < R<C[K] maka krokomosom ke-k sebagai induk

Ket : R[K] = bilangan random antara [0-1]

C[K] = comulative dari probabilitas

kromosom populasi baru hasil seleksi adalah :

TIF115 K2B R5 T2	TIF118 K3B R4 T1	UNG110 K1B R3 T4
TIF115 K2C R3 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF115 K2C R3 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF118 K3B R4 T1	TIF115 K2C R3 T2	UNG110 K1A R1 T4
TIF115 K2A R1 T2	TIF118 K3A R3 T2	UNG110 K1A R1 T2
TIF118 K3B R4 T4	TIF115 K2C R4 T4	UNG110 K1A R3 T1
TIF115 K2B R5 T2	TIF118 K3B R4 T1	UNG110 K1B R3 T4
TIF115 K2B R3 T2	TIF118 K3B R5 T1	UNG110 K1A R3 T2
TIF115 K2A R1 T2	TIF118 K3A R3 T2	UNG110 K1A R1 T2

- Pindah Silang (Cross Over)

  Bilangan Random : [0,15 , 0,09 , 0,75 ,0,9 ,0,25 , 0,67 , 0,87, 0,37 ,0,55]

Kromosom 1 >< kromosom 2
Kromosom 2 >< kromosom 5
Kromosom 5 >< kromosom 8
Kromosom 8 >< kromosom 1

  Posisi Cut-Point Crossover dipilih menggunakan bilangan acak

  Misal : C[1] = 1,C[2]=1,C[3]=2,C[4]=1

1	Kromosom 1 >< Kromosom 2 TIF115 K2B R5 T2 TIF118 K3C R2 T2 UNG110 K1C R1 T4
2	Kromosom 2 >< Kromosom 5 TIF115 K2C R3 T2 TIF118 K3A R3 T2 UNG110 K1A R1 T2
3	Kromosom 5 >< Kromosom 8 TIF115 K2B R3 T2 TIF118 K3A R3 T2 UNG110 K1A R3 T2
4	Kromosom 8 >< Kromosom 1 TIF115 K2B R3 T2 TIF118 K3B R4 T1 UNG110 K1B R3 T2

Populasi kromosom setelah proses crossover

TIF115 K2B R5 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF115 K2C R3 T2	TIF118 K3A R3 T2	UNG110 K1A R1 T4
TIF115 K2C R3 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF118 K3B R4 T1	TIF115 K2C R3 T2	UNG110 K1A R1 T4
TIF115 K2A R3 T2	TIF118 K3A R3 T2	UNG110 K1A R3 T2
TIF118 K3B R4 T4	TIF115 K2C R4 T4	UNG110 K1A R3 T1
TIF115 K2B R5 T2	TIF118 K3B R4 T1	UNG110 K1B R3 T4
TIF115 K2B R3 T2	TIF118 K3B R5 T1	UNG110 K1B R3 T2
TIF115 K2A R1 T2	TIF118 K3A R3 T2	UNG110 K1A R1 T2

- Mutasi

  Pmuk umumnya diset antara [0-1] = 0,1

Total gen = jumlah gen x jumlah kromosom = 3x9 = 27
Mutasi yang terjadi = 0,1 x 27 = 2,7
Gen yang dimutasi = 3

Posisi gen yang terpilih 13 14 15, maka setelah dimutasi maka **menghasilkan Kromosom baru :**

Sebelum mutasi = TIF115 K2B R3T2 TIF118 K3A R3 T2 UNG110 K1A R3 T2
Setelah mutasi = TIF115 K2B R1 T1 TIF188 K3A R2 T1 UNG110 K1A R2 T2
Kromosom 5 = TIF115 K2B R1 T1 TIF188 K3A R2 T1 UNG110 K1A R2 T2

Pada iterasi 1 masih terdapat kesalahan / pembentrokan maka perhitungan akan berlangsung kembali ke iterasi 2. Pada iterasi kedua perhitungan di mulai dari :

- Menghitung Nilai Fitness

Fitnes 1=1/1+(0+0+0+0)=1
Fitnes 2=1/1+(0+0+1+1)=0,33
Fitnes 3=1/1+(0+0+0+0)=1
Fitnes 4=1/1+(0+0+0+0)=1
Fitnes 5=1/1+(0+0+0+0)=1
Fitnes 6=1/1+(0+0+1+1)=0,33
Fitnes 7=1/1+(0+0+0+0)=1
Fitnes 8=1/1+(0+0+1+1)=0,33
Fitnes 9=1/1+(0+0+1+1)=0,33

- Seleksi

Kromosom	Nilai Fitnes
1	1
2	0,33
3	1
4	1
5	1
6	0,33
7	1
8	0,33
9	0,33
Total nilai fitnes	6,32

Kromosom	Probabilitas
1	1/6,32=0,16
2	0,33/6,32=0,05
3	1/6,32=0,16
4	1/6,32=0,16
5	1/6,32=0,16
6	0,33/6,32=0,05
7	1/6,32=0,16
8	0,33/6,32=0,05
9	0,33/6,32=0,05
Total Probabilitas	1

Menempatkan masing-masing kromosom pada interval nilai [0-1]

Kromosom	Interval nilai
1	0-0,16
2	0,17-0,21
3	0,22-0,37
4	0,38-0,53
5	0,54-0,69
6	0,70-0,74
7	0,75-0,9
8	0,91-0,95
9	0,96-1

Bilangan random : [0,14 , 0,25 , 0,34 , 0,75 , 0,15 , 2,20 , 0,9 , 0,05 , 0,17]

kromosom populasi baru hasil seleksi adalah :

TIF115 K2B R5 T2	TIF118 K3B R4 T1	UNG110 K1C R1 T4
TIF115 K2C R3 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF118 K3B R4 T1	TIF118 K3C R2 T2	UNG110 K1A R1 T4
TIF115 K2B R1 T1	TIF115 K2C R3 T2	UNG110 K1A R2 T2
TIF115 K2B R5 T2	TIF118 K3A R3 T2	UNG110 K1C R1 T4
TIF115 K2C R3 T2	TIF115 K2C R4 T4	UNG110 K1A R1 T2
TIF115 K2C R3 T2	TIF118 K3B R4 T1	UNG110 K1C R1 T4
TIF115 K2B R5 T2	TIF118 K3B R5 T1	UNG110 K1C R1 T4
TIF115 K2C R3 T2	TIF118 K3A R3 T2	UNG110 K1A R1 T4

- Pindah Silang (Cross Over)

  Bilangan Random :[0,17 , 0,8 , 0,4 , 0,35 ,0,84 , 0,64 , 0,16 , 0,9 ,0,77]

Kromosom 1 >< kromosom 2
Kromosom 2 >< kromosom 5
Kromosom 5 >< kromosom 8
Kromosom 8 >< kromosom 1

  Misal : C[1]=2, C[2]=3, C[3]=2, C[4]=3

Kromosom 1 >< Kromosom 3
TIF118 K3B R4 T1 TIF118 K3C R2 T2 UNG110 K1A R1 T4
Kromosom 3 >< Kromosom 4
TIF115 K22 R1 T1 TIF118 K3A R2 T1 UNG110 K1A R1 T4
Kromosom 4 >< Kromosom 7
TIF115 K2C R3 T2 TIF118 K3A R2 T1 UNG110 K1C R1 T4
Kromosom 7 >< Kromosom 1
TIF115 K2B R5 T2 TIF118 K3C R2 T2 UNG110 K1C R1 T4

  Populasi kromosom setelah mengalami proses cross-over :

TIF118 K3B R4 T1	TIF118 K3C R2 T2	UNG110 K1A R1 T4
TIF115 K2C R3 T2	TIF118 K3A R2 T2	UNG110 K1C R1 T4
TIF115 K2B R1 T1	TIF118 K3A R2 T1	UNG110 K1A R1 T4
TIF115 K2C R3 T2	TIF118 K2A R2 T1	UNG110 K1C R1 T4
TIF115 K2B R5 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF118 K2C R3 T2	TIF118 K3A R3 T2	UNG110 K1A R1 T4
TIF115 K2B R5 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF115 K2B R5 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF115 K2C R3 T2	TIF118 K3A R3 T2	UNG110 K1A R1 T4

- Mutasi

  Pmuk = 0,1

  Total Gen = 3 x 9 = 27

  Probabilitas = 0,1 x 27 = 2,7 = 3

  Posisi gen yang terpilih 16 18 25, maka setelah dimutasi menghasilkan kromosom baru

Pmut = 0,1
Total gen = jumlah gen x jumlah kromosom = 3x9 = 27
Mutasi yang terjadi = 0,1 x 2,7 = 2,7
Gen yang dimutasi = 3
Posis GEN yang terpilih 16,18,25
Hasil kromosom baru
Sebelum mutasi = TIF115 K2C R3T2 TIF118 K3A R3 T2 UNG110 K1A R1 T4
Setelah mutasi = TIF115 K2C R1 T7 TIF188 K3A R3 T2 UNG110 K1A R1 T6
Sebelum mutasi = TIF115 K2C R3T2 TIF118 K3A R3 T2 UNG110 K1A R1 T4
Setelah mutasi = TIF115 K2C R2 T2 TIF188 K3A R3 T2 UNG110 K1A R1 T4

  Mendapatkan Kromosom Baru :

TIF118 K3B R4 T1	TIF118 K3C R2 T2	UNG110 K1A R1 T4
TIF115 K2C R3 T2	TIF118 K3A_R2 T2	UNG110 K1C R1 T4
TIF115 K2B R1 T1	TIF118 K3A R2 T1	UNG110 K1A R1 T4
TIF115 K2C R3 T2	TIF118 K2A R2 T1	UNG110 K1C R1 T4
TIF115 K2B R5 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF118 K2C R1 T7	TIF118 K3A R3 T2	UNG110 K1A R1 T6
TIF115 K2B R5 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF115 K2B R5 T2	TIF118 K3C R2 T2	UNG110 K1C R1 T4
TIF115 K2C R2 T2	TIF118 K3A_R3 T2	UNG110 K1A R1 T4

  Perhitungan berhenti di iterasi kedua, dikarenakan hasil dari kromosom di atas sudah tidak mengalami pembentrokan.

6.3. Soal Latihan

Bagaimanakah algoritma dan bentuk diagram alir dari penciptaan sebuah gen baru dari algoritma genetika?

BAB VII.

KECERDASAN BERKOLONI

7.1. Tujuan Instruksional Khusus

- Mahasiswa mampu menjelaskan Algoritma koloni diantaranya Ant colony, Bee colony, Particle Swarm Optimization
- Mahasiswa mampu menerapkan algoritma koloni untuk memecahkan permasalahan

7.2. Materi Perkuliahan

Swarm Intelligence merupakan sebuah metode penyelesaian masalah yang memanfaatkan perilaku sekumpulan agen yang saling bekerja sama dengan mengutamakan kecerdasan berkelompok. Dengan semakin banyaknya agen dan terkumpulnya kecerdasan agen-agen dalam kelompok akan menyebabkan terkumpulnya kecerdasan kelompok yang luar biasa. Beberapa algoritma yang termasuk dalam swarm intelligence diantaranya *Particle Swarm Optimization*, *Ant Colony Optimization* dan *Artificial Bee Colony*.

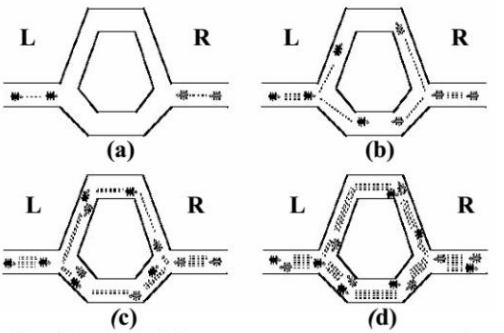
A. Ant Colony (Koloni Semut)

Dalam kehidupan, sering dilakukan perjalanan dari satu tempat atau kota ke tempat yang lain dengan mempertimbangkan efisiensi, waktu dan biaya sehingga diperlukan ketepatan dalam menentukan jalur terpendek antar suatu kota. Hasil penentuan jalur terpendek akan menjadi pertimbangan dalam pengambilan keputusan untuk menununjukkan jalur yang akan ditempuh. Hasil yang didapatkan juga membutuhkan kecepatan dan keakuratan

dengan bantuan komputer. Secara umum, pencarian jalur terpendek dapat dibagi menjadi dua metode, yaitu metode konvensional dan metode heuristik.

Metode konvensional cenderung lebih mudah dipahami daripada metode heuristik, tetapi jika dibandingkan, hasil yang diperoleh dari metode heuristik lebih variatif dan waktu perhitungan yang diperlukan lebih singkat . Metode heuristik terdiri dari beberapa macam algortima yang biasa digunakan. Salah satunya adalah algoritma semut (Ant Colony, Antco). Antco diambil dari perilaku koloni semut dalam pencarian jalur terpendek antara sarang dan sumber makanan.

Antco diadopsi dari perilaku koloni semut yang dikenal sebagai sistem semut (Dorigo, 1996). Secara alamiah koloni semut mampu menemukan rute terpendek dalam perjalanan dari sarang ke tempat-tempat sumber makanan. Koloni semut dapat menemukan rute terpendek antara sarang dan sumber makanan berdasarkan jejak kaki pada lintasan yang telah dilalui. Semakin banyak semut yang melewati suatu lintasan maka akan semakin jelas bekas jejak kakinya, hal ini akan menyebabkan lintasan yang dilalui semut dalam jumlah sedikit semakin lama akan semakin berkurang kepadatan semut yang melewatinya, atau bahkan akan tidak dilewati sama sekali dan sebaliknya. Mengingat prinsip algoritma yang didasarkan pada perilaku koloni semut dalam menemukan jarak perjalanan paling pendek tersebut, Algoritma Semut sangat tepat digunakan untuk diterapkan dalam penyelesaian masalah optimasi, salah satunya adalah untuk menentukan jalur terpendek.



Gambar 1. Perjalanan semut menemukan sumber makanan.

Koloni semut dapat menemukan jalur terpendek antara sarang dan sumber makanan berdasarkan jejak kaki pada lintasan yang telah dilewati. Semakin banyak semut yang melewati suatu lintasan maka semakin jelas bekas jejak kakinya. Hal ini menyebabkan lintasan yang dilalui semut dalam jumlah sedikit semakin lama semakin berkurang kepadatan semut yang melewatinya, atau bahkan akan tidak dilewati sama sekali. Sebaliknya lintasan yang dilalui semut dalam jumlah banyak semakin lama akan semakin bertambah kepadatan semut yang melewatinya atau bahkan semua semut melewati lintasan tersebut.

Gambar 1.a menujukkan perjalanan semut dalam menemukan jalur terpendek dari sarang ke sumber makanan, terdapat dua kelompok semut yang melakukan perjalanan. Kelompok semut L berangkat dari arah kiri ke kanan dan kelompok semut R berangkat dari kanan ke kiri. Kedua kelompok berangkat dari titik yang sama dan dalam posisi pengambilan keputusan jalan sebelah mana yang akan diambil. Kelompok L membagi dua kelompok lagi. Sebagian melewati jalan atas dan sebagian melewati jalan bawah. Hal ini juga berlaku pada kelompok R. Gambar 1.b dan Gambar 1.c menunjukkan bahwa kelompok semut berjalan pada kecepatan yang sama dengan meninggalkan feromon atau jejak kaki di jalan yang telah dilalui. Feromon yang ditinggalkan oleh kumpulan semut yang melewati jalan atas telah mengalami banyak penguapan karena

semut yang melewati jalan atas berjumlah lebih sedikit dibandingkan jalan yang di bawah. Hal ini disebabkan jarak yang ditempuh lebih panjang dibandingkan jalan bawah. Sedangkan feromon yang berada pada bagian bawah penguapannya cenderung lebih lama. Karena semut yang melewati jalan bawah lebih banyak daripada semut yang melewati jalan atas. Gambar 1.d menunjukkan bahwa semut-semut yang lain pada akhirnya memutuskan untuk melewati jalan bawah karena feromon yang ditinggalkan masih banyak, sedangkan feromon pada jalan atas sudah banyak menguap sehingga semut-semut tidak memilih jalan atas. Semakin banyak semut yang melewati jalan maka semakin banyak semut yang mengikutinya, semakin sedikit semut yang melewati jalan, maka feromon yang ditinggalkan semakin berkurang bahkan hilang. Dari sinilah kemudian terpilihlah jalur terpendek antara sarang dan sumber makanan.

Dalam algoritma semut, diperlukan beberapa variabel dan langkah-langkah untuk menentukan jalur terpendek, yaitu :

Langkah 1 :

a. Inisialisasi harga parameter-parameter algoritma.
   Parameter-parameter yang di inisialisasikan adalah:
   1. Intensitas jejak semut antar kota dan perubahannya (τij)
   2. Banyak kota (n) termasuk x dan y (koordinat) atau dij (jarak antar kota)
   3. Penentuan kota berangkat dan kota tujuan
   4. Tetapan siklus-semut (Q)
   5. Tetapan pengendali intensitas jejak semut (α)
   6. Tetapan pengendali visibilitas (β)
   7. Visibilitas antar kota = 1/dij (ηij)
   8. Jumlah semut (m)
   9. Tetapan penguapan jejak semut (ρ)
   10. Jumlah siklus maksimum (NCmax) bersifat tetap selama algoritma dijalankan, sedangkan τij akan selalu diperbaharui harganya pada setiap siklus algoritma mulai dari siklus

pertama (NC=1) sampai tercapai jumlah siklus maksimum (NC=NCmax) atau sampai terjadi konvergensi.

b. Inisialisasi kota pertama setiap semut. Setelah inisialisasi τij dilakukan, kemudian m semut ditempatkan pada kota pertama yang telah ditentukan.

Langkah 2 :

Pengisian kota pertama ke dalam tabu list. Hasil inisialisasi kota pertama semut pada langkah 1 harus diisikan sebagai elemen pertama tabu list. Hasil dari langkah ini adalah terisinya elemen pertama tabu list setiap semut dengan indeks kota pertama.

Langkah 3 :

Penyusunan jalur kunjungan setiap semut ke setiap kota. Koloni semut yang sudah terdistribusi ke kota pertama akan mulai melakukan perjalanan dari kota pertama sebagai kota asal dan salah satu kotakota lainnya sebagai kota tujuan. Kemudian dari kota kedua, masing-masing koloni semut akan dari kota-kota yang tidak terdapat pada tabuk sebagai kota tujuan selanjutnya. Perjalanan koloni semut berlangsung terus menerus hingga mencapai kota yang telah ditentukan. Jika s menyatakan indeks urutan kunjungan, kota asal dinyatakan sebagai tabuk(s) dan kota-kota lainnya dinyatakan sebagai {N-tabuk}, maka untuk menentukan kota tujuan digunakan persamaan probabilitas kota untuk dikunjungi sebagai berikut,

$$p_{ij}^{k} = \frac{\left[\tau_{ij}\right]^{\alpha} \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{k' \in \{N-tabu_k\}} \left[\tau_{ik'}\right]^{\alpha} \cdot \left[\eta_{ik'}\right]^{\beta}} \quad \text{untuk } j \in \{N\text{-}tabu_k\}$$

$$p_{ij}^{k} = 0, \text{ untuk j lainnya}$$

dengan i sebagai indeks kota asal dan j sebagai indeks kota tujuan.

Langkah 4 :

a. Perhitungan panjang jalur setiap semut. Perhitungan panjang jalur tertutup (length closed tour) atau Lk setiap semut dilakukan setelah satu siklus diselesaikan oleh semua semut. Perhitungan dilakukan berdasarkan tabuk masing-masing dengan persamaan berikut:

$$L_k = d_{tabu_k(n),tabu_k(1)} + \sum_{s=1}^{n-1} d_{tabu_k(s),tabu_k(s+1)}$$

dengan dij adalah jarak antara kota i ke kota j yang dihitung berdasarkan persamaan:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

b. Pencarian rute terpendek. Setelah Lk setiap semut dihitung, akan diperoleh harga minimal panjang jalur tertutup setiap siklus atau LminNC dan harga minimal panjang jalur tertutup secara keseluruhan adalah atau Lmin.

c. Perhitungan perubahan harga intensitas jejak kaki semut antar kota. Koloni semut akan meninggalkan jejak-jejak kaki pada lintasan antar kota yang dilaluinya. Adanya penguapan dan perbedaan jumlah semut yang lewat, menyebabkan kemungkinan terjadinya perubahan harga intensitas jejak kaki semut antar kota. Persamaan perubahannya adalah:

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

dengan $\Delta\tau k\ ij$ adalah perubahan harga intensitas jejak kaki semut antar kota setiap semut yang dihitung berdasarkan persamaan

$$\Delta\tau_{ij}^{k} = \frac{Q}{L_k}$$

untuk (i,j) ∈ kota asal dan kota tujuan dalam $tabu_k$

$$\Delta\tau_{ij}^{k} = 0 \text{ , untuk (i,j) lainnya}$$

Langkah 5 :

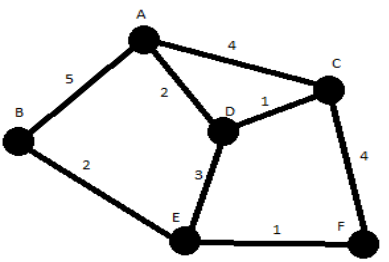
a. Perhitungan harga intensitas jejak kaki semut antar kota untuk siklus selanjutnya. Harga intensitas jejak kaki semut antar kota pada semua lintasan antar kota ada kemungkinan berubah karena adanya penguapan dan perbedaan jumlah semut yang melewati. Untuk siklus selanjutnya, semut yang akan melewati lintasan tersebut harga intensitasnya telah berubah. Harga intensitas jejak kaki semut antar kota untuk siklus selanjutnya dihitung dengan persamaan:

$$\tau_{ij} = \rho \cdot \tau_{ij} + \Delta\tau_{ij}$$

b. Atur ulang harga perubahan intensitas jejak kaki semut antar kota. Untuk siklus selanjutnya perubahan harga intensitas jejak semut antar kota perlu diatur kembali agar memiliki nilai sama dengan nol.

Langkah 6 :

Pengosongan tabu list, dan ulangi langkah dua jika diperlukan. Tabu list perlu dikosongkan untuk diisi lagi dengan urutan kota yang baru pada siklus selanjutnya, jika jumlah siklus maksimum belum tercapai atau belum terjadi konvergensi Algoritma diulang lagi dari langkah dua dengan harga parameter intensitas jejak kaki semut antar kota yang sudah diperbaharui.



Gambar 32. Contoh Kasus

Jarak AntarKota						
	A	**B**	**C**	**D**	**E**	**F**
A	0	5	4	2	0	0
B	5	0	0	0	2	0
C	4	0	0	1	0	4
D	2	0	1	0	3	0
E	0	2	0	3	0	1

Tabel 7. Jarak Antar kota d_{ij}

Dari jarak kota yang telah diketahui dapat dihitung visibilitas antarkota sebagai berikut:

$$n = \frac{1}{d_{iy}}$$

Menghitung visibilitas antarkota						
	A	**B**	**C**	**D**	**E**	**F**
A	0	0,2	0,25	0,5	0	0
B	0,2	0	0	0	0,5	0
C	0,25	0	0	1	0	0,25
D	0,5	0	1	0	0,333333333	0
E	0	0,5	0	0,333333333	0	1
F	0	0	0,25	0	1	0

Tabel 8. Visibilitas antar kota

Parameter-parameter yang digunakan adalah sebagai berikut :

α	**β**	**m**	**n**	**Q**	**ρ**	**τij**	**NCmax**
1	1	3	5	1	0,5	0,1	3
kota awal = D							
kota tujuan = B							

Dari intensitas jejak semut τ_{ij} yang telah ditetapkan dan perhitungan visibilitas antarkota maka dapat diketahui probabilitas kota untuk dikunjungi sebagai berikut:

Probabilitas kota untuk dikunjungi siklus ke-1 kasus 1						
	A	**B**	**C**	**D**	**E**	**F**
A	0	0,10909	0,13636	0,27273	0	0
B	0,10909	0	0	0	0, 0,27273	0
C	0,13636	0	0	0,545455	0	0,13636
D	0,27273	0	0,545454	0	0,181812	0
E	0	0,27273	0	0,181812	0	0, 54545
F	0	0	0,13636	0	0,54545	0

Tabel 9. Probabilitas kota untuk dikunjungi siklus ke-1 kasus 1

Sehingga didapat panjang jalur semut pada siklus ke-1 sebagai berikut :

Jalur semut siklus 1		
Semut	**Jalur**	**Panjang jalur**
1	D - A - B	7
2	D - E - B	5
3	D - C - F - E - B	8

Tabel 10. Panjang jalur semut siklus ke-1 kasus 1

Dari perjalan semut pada siklus ke-1 maka akan terjadi perubahan harga intensitas jejak kaki semut τ ij antarkota untuk siklus selanjutnya sebagai berikut :

Perubahan intensitas jejak kaki semut τ ij antar kota ke 2							
	τ *ij* (awal)	P_{ij}^{k}	$\Delta\tau_{ij}^{1}$	$\Delta\tau_{ij}^{2}$	$\Delta\tau_{ij}^{3}$	$\Delta\tau_{ij}$	τ_{NEW}
A,B	0,1	0,10909	0,142857	0	0	0,142857	0,192857
C,F	0,1	0,13636	0	0	0,125	0,125	0,175
D,A	0,1	0,27273	0,142857	0	0	0,142857	0,192857
D,C	0,1	0,54545	0	0	0,125	0,125	0,175
D,E	0,1	0,18182	0,2	0	0	0,2	0,25
F,E	0,1	0,54545	0	0	0,125	0,125	0,175
E,B	0,1	0,27273	0	0,2	0,125	0,325	0,375

A,C	0,1	0,13636	0	0	0	0	0,05
A,D	0,1	0,27273	0	0	0	0	0,05
B,A	0,1	0,10909	0	0	0	0	0,05
C,A	0,1	0,13636	0	0	0	0	0,05
C,D	0,1	0,54545	0	0	0	0	0,05
E,D	0,1	0,18182	0	0	0	0	0,05
E,F	0,1	0,54545	0	0	0	0	0,05
B,E	0,1	0,27273	0	0	0	0	0,05
F,C	0,1	0,13636	0	0	0	0	0,05

Tabel 11. Perubahan harga intensitas jejak kaki semut τ ij antarkota siklus ke-2 kasus 1

Mengitung probabilitas kota untuk dikunjungi siklus 2						
	A	**B**	**C**	**D**	**E**	**F**
A	0	0,054545	0,13636	0,52597	0	0
B	0,210389	0	0	0	1,02272	0
C	0,054545	0	0	0,47727	0	0,13636
D	0,054545	0	0,13636	0	0,13636	0
E	0	0,054545	0	0,68181	0	0,47727
F	0	0	0,47727	0	0,13636	0

Tabel 12. Probabilitas kota untuk dikunjungi siklus ke-2 kasus 1 dengan τ ij telah diperbaharui

panjang jalur semut sikllus 2		
semut	**jalur**	**panjang jalur**
1	D - A - B	7
2	D - E - B	5
3	D - C - A - B	10

Tabel 13. Panjang jalur semut siklus ke-2 kasus 1

Perubahan intensitas jejak kaki semut τ ij antar kota ke 3							
	τ *ij* (awal)	P_{ij}^{k}	$\Delta\tau_{ij}^{1}$	$\Delta\tau_{ij}^{2}$	$\Delta\tau_{ij}^{3}$	$\Delta\tau_{ij}$	τ_{NEW}
A,B	0,1	0,21031	0,14285	0	0,1	0,24285	0,29285
C,F	0,1	0,47727	0	0	0	0	0,05
D,A	0,1	0,52597	0,14285	0	0	0,14285	0,19285
D,C	0,1	0,47727	0	0	0,1	0,1	0,15
D,E	0,1	0,68181	0	0,2	0	0,2	0,25
F,E	0,1	0,47727	0	0	0	0	0,05
E,B	0,1	1,02272	0	0,2	0	0,2	0,25
A,C	0,1	0,05454	0	0	0	0	0,05
A,D	0,1	0,05454	0	0	0	0	0,05
B,A	0,1	0,05454	0	0	0	0	0,05
C,A	0,1	0,13636	0,1	0	0	0,1	0,15
C,D	0,1	0,13636	0	0	0	0	0,05
E,D	0,1	0,13636	0	0	0	0	0,05
E,F	0,1	0,13636	0	0	0	0	0,05
B,E	0,1	0,05454	0	0	0	0	0,05
F,C	0,1	0,13636	0	0	0	0	0,05

Tabel 14. Perubahan harga intensitas jejak kaki semut τ ij antarkota siklus ke-3 kasus 1

Mengitung probabilitas kota untuk dikunjungi siklus 3						
	A	**B**	**C**	**D**	**E**	**F**
A	0	0,05454	0,16363	0,21038	0	0
B	0,31948	0	0	0	0,27272	0
C	0,05454	0	0	0,16363	0	0,054545
D	0,05454	0	0,05454	0	0,05454	0
E	0	0,05454	0	0,27272	0	0,054545
F	0	0	0,05454	0	0,05454	0

Tabel 15. Probabilitas kota untuk dikunjungi siklus ke-3 kasus 1 dengan τij telah diperbaharui

Panjang jalur semut siklus 3		
Semut	**Jalur**	**Panjang jalur**
1	D - A - B	7
2	D - E - B	5
3	D - E - B	5

Tabel 16. Panjang jalur semut siklus ke-3 kasus 1

Dari ke-3 siklus yang telah dilakukan akan diperoleh jalur terpendek dari kota asal D ke kota tujuan B yaitu D – E – B dengan panjang jalur 5. Karena siklus yang diinginkan hanya 3 siklus maka langkah selanjutnya tidak dikerjakan lagi karena sudah terjadi konvergensi.

B. Bee Colony (Koloni Lebah)

Algoritma koloni lebah adalah *swarm intelligence* yang terinspirasi perilaku sosial koloni lebah madu, dan digunakan untuk menyelesaikan berbagai persoalan komputasi. Lebah merupakan serangga sosial yang sangat terorganisir. Kelangsungan hidup sebuah koloni tergantung dari setiap individu yang lain. lebah menggunakan segregasi sistematis untuk memastikan kelanjutan keberadaan koloninya. Mereka juga melakukan berbagai macam tugas seperti mencari makanan, reproduksi, mengurus lebah yang masih muda, patroli dan membangun sarang. Dari jumlah lebah yang banyak dalam sebuah koloni, mencari makan merupakan kegiatan utama yang dilakukan oleh koloni lebah untuk menjamin pasokan sumber makanan bagi koloni lainnya. Perilaku yang dilakukan oleh koloni lebah masih menjadi misteri sampai Von Frisch [19] menterjemahkan bahasa isyarat yang ada pada tarian lebah. Tarian lebah ini digunakan sebagai alat komunikasi yang digunakan oleh koloni. Misalnya setelah lebah kembali dari sarangnya, ia akan melakukan tarian yang

disebut dengan waggle dance. Dengan menggunakan tarian ini lebah yang lain dapat saling berbagi informasi mengenai sumber makanan berupa jarak dan arah untuk menuju ke sumber makanan yang telah ditemukan. Sebagai algoritma *metaheuristik*, berbagai teknik Optimasi Lebah, seperti Optimasi Koloni Lebah (*Bee Colony Optimization*), Koloni Lebah Tiruan (*Artificial Bee Colony Optimization*) dan Optimasi Persilangan Madu-Lebah (*Honey-Bee Mating Optimization*), menerapkan perilaku-perilaku lebah dalam menemukan rute terpendek di antara bunga-bunga yang ada.

Algoritma *Bee Colony Optimization* pertama kali diusulkan oleh Lucic dan Teodorovic pada tahun 2001 dan digunakan untuk menyelesaikan masalah *Traveling Salesman Problem* (TSP). Algoritma *Bee Colony Optimization* (*BCO*) adalah algoritma yang berbasis populasi, yaitu populasi lebah-lebah buatan mencari solusi optimal. lebah-lebah mewakili sekumpulan agen yang bersama sama menyelesaikan masalah-masalah optimasi kombinatorial yang kompleks. Setiap lebah buatan menghasilkan satu solusi untuk permasalahan yang akan diselesaikan.

Menurut Teodorovic, Algoritma ini terdiri dari dua tahap yang diiterasikan secara bergantian yaitu:

1. ***Forward Pass*:** Semua lebah buatan mengeksplorasi ruang pencarian dan melakukan pergerakan yang telah ditentukan, yang membangun dan/atau meningkatkan solusi untuk menghasilkan solusi baru. Setelah solusi parsial baru diperoleh, lebah akan kembali lagi ke sarang dan memulai tahap kedua
2. ***Backward Pass***: semua lebah buatan saling berbagi informasi mengenai solusi-solusi yang telah ditemukan. Dalam tahap ini setiap lebah memutuskan dengan probabilitas tertentu, apakah akan mengabaikan solusi parsial yang telah dibuat dan menjadi *follower* dengan mengikuti solusi lebah lain, atau menjadi *recruiter* dengan melakukan waggle dance dan merekrut lebah-lebah lain di sarang sebelum kembali ke solusi parsial yang telah dibuat. (Lebah dengan nilai fungsi obyektif lebih tinggi memiliki peluang

lebih besar untuk meneruskan jalur eksplorasinya sendiri). Setiap lebah *follower* memilih solusi baru dari lebah *recruiter* yang akan diikuti secara *Roullete wheel* dimana lebah dengan solusi yang lebih baik memiliki peluang yang lebih tinggi untuk dipilih.

Tahap *forward pass* dan *backward pass* dilakukan secara iteratif dan akan berhenti setelah mencapai kondisi yang telah ditentukan. kondisi-kondisi dimaksud antara lain seperti jumlah total tahap *forward* dan *backward pass*, maksimum jumlah total iterasi atau tahap *forward* dan *backward pass* tanpa perbaikan fungsi obyektif dan sebagainya.

Seleksi Roulette Wheel

Seleksi *Roulette Wheel* merupakan skema seleksi yang paling sederhana dan merupakan algoritma stokastik yang meliputi langkah-langkah sebagai berikut. Setiap individu dalam populasi dipetakan pada beberapa segmen dimana segmen untuk setiap individu berukuran sama dengan nilai profitabilitasnya. Sebuah bilangan acak kemudian dibangkitkan dan individu yang segmennya mengandung bilangan acak tersebut akan dipilih.

Proses tersebut diiterasikan sampai jumlah individu yang dikehendaki diperoleh (*mating population*). Teknik ini dapat dianalogikan seperti sebuah roda roulette (*Roulette Wheel*) dengan ukuran setiap potongannya proporsional dengan profitabilitasnya (*fitness*).

Waggle Dance

Jika seekor lebah menari, tarian lebah akan berlangsung selama beberapa durasi. Durasi tarian *Di* dari lebah *i* ditentukan oleh fungsi linear berikut.

$$Di = K.\frac{Pfi}{Pfcolony}$$

Durasi diukur selama iterasi pada algoritma dieksekusi. Menurut fungsi linear, jika lebah *i* memiliki Pfi yang lebih tinggi, maka akan diberikan kesempatan untuk menari lagi (*dance* muncul dalam iterasi lebih). Jika tidak, maka tarian tesrsebut untuk jangka pendek. Pfi melambangkan skor profitabilitas lebah *i*. Pfi suatu simpul pada jalur dapat ditafsirkan sebagai kuantitas nektar yang dikumpulkan oleh lebah.ke *i*. Lebah akan mengumpulkan jumlah *nectar* yang lebih banyak bila melakukann perjalanan yang panjang dengan rute lebih pendek. Dengan demikian, Pfi didefinisikan berbanding terbalik dengan panjang tur *Li*.

$$Pfi = \frac{1}{Li}$$

P$f$colony menunjukkan koloni lebah dengan rata-rata profitabilitas dan diperbarui setelah masing- masing lebah menyeselaikan tur. K didefinisikan sebagai skala faktor yang mengendalikan besarnya durasi.

$$Pfcolony = \frac{1}{NBee}\sum_{i=1}^{NBee} Pfi$$

Kemungkinan *Ri* mengikuti path yang biasa menurut profitability rating dari lebah dan koloni pada dasarnya seperti yang diperlihatkan pada tabel 17

Profitability Scores	***Pfollow / Ri***
Pfi < 0.95 P$f$colony	0.60
0.95 P$f$colony ≤ Pfi < 0.975 P$f$colony	0.20
0.975 P$f$colony ≤ Pfi < 0.99 P$f$colony	0.02
0.99P$f$colony ≤ Pfi	0.00

Tabel 17. Penentuan mengikuti *wagle dance*

Lebah lebih menyukai mengobservasi secara random dan mengikuti waggle dance dalam *dance floor* jika *profitability rating* nya lebih rendah dalam koloni. Dalam kasus ekstrim, dimana *Ri* adalah

nol, maka lebah akan memelihara *path*-nya sendiri. Lebah lebih suka melakukan pencarian secara random dan mengikuti *waggle dance* jika *profitability rating*-nya rendah bila dibandingkan dengan rata-rata *profitability* koloninya.

7.3. Soal Latihan

Lakukan algotritma bee colony untuk memecahkan permasalahan pada sebuah rubik dengan ubin 3x3. Buat alur algoritma dan solusinya!

DAFTAR PUSTAKA

1. Andri Kristanto. *Jaringan Syaraf Tiruan,* Gava Media.Jogjakarta. 2004
2. Saludin Muis. *Jaringan Syaraf Tiruan Sebagai Alat Bantu Peramalan Saham.* Graha Ilmu. Jogjakarta. 2006
3. Sri Kusumadewi. *Artificial Intelligence*. Graha Ilmu. Jogjakarta. 2003
4. Sri Kusumadewi, Hari Purnomo. *Logika Fuzzy,* Graha Ilmu. 2004
5. Jong Jek Siang, Jaringan Syaraf Tiruan & Pemrogramanny Menggunakan Matlab, Jogjakarta, Andi Offset. 2009
6. Eka Mala Sari R, Handout Kecerdasan Komputasional, Universitas Trunojoyo Madura, 2016