

Kecerdasan Buatan

Tujuan :

1. Mengetahui konsep dasar kecerdasan buatan, teknologi, dan area penerapan kecerdasan buatan.
2. Mampu merancang suatu sistem cerdas (*intelligent system*) dengan menggunakan suatu bahasa pemrograman tertentu.
3. Mempunyai dasar untuk mengembangkan ilmu pengetahuan yang dimiliki ke jenjang yang lebih tinggi.

Referensi :

1. Away, Gunaidi Abdia. *The Shortcut of Matlab Programming*. Penerbit Informatika
2. *Matlab User's Guide*. The Math Works, Inc
3. Negnevitsky, Michael. *Artificial Intelligence "A Guide To Intelligent Systems"*. Mc Graw Hill Company
4. Waterman, Donald A. *A Guide to Expert Systems*. Addison Wesley Publishing Company
5. Desiani, Anita dan Muhammad Arhami. *Konsep Kecerdasan Buatan*. Penerbit Andi
6. Pandjaitan, Lanny W. *Dasar – Dasar Komputasi Cerdas*. Penerbit Andi
7. Kusumadewi, Sri. *Membangun Jaringan Syaraf Tiruan Menggunakan Matlab dan Excel Link*. Penerbit Graha Ilmu
8. Arhami, Muhammad. *Konsep Dasar Sistem Pakar*. Penerbit Andi
9. Puspitaningrum, Diah. *Pengantar Jaringan Saraf Tiruan*. Penerbit Andi.
10. Website dan jurnal – jurnal ilmiah.

Pendahuluan

Sejak dekade akhir abad ke – 20, **kecerdasan tiruan** (*artificial intelligence*) telah menjadi topik yang sangat menarik untuk dikembangkan dalam berbagai disiplin ilmu. Namun, pada dasarnya **Artificial Intelligence** (AI) atau **kecerdasan buatan** merupakan cabang dari ilmu komputer yang *concern* dengan pengautomatisasi tingkah laku cerdas.

Beberapa pakar ilmu kecerdasan tiruan membuat bermacam – macam definisi, antara lain:

1. **Buchanan** dan **Shortliffe** (1985) menyatakan bahwa kecerdasan tiruan merupakan manipulasi simbol – simbol untuk menyelesaikan masalah.
2. **Waterman** (1986) mengungkapkan bahwa kecerdasan tiruan adalah bagian penting ilmu pengetahuan bidang komputer yang diperlukan untuk mengembangkan kecerdasan program – program komputer.
3. **Rich** (1981) mendefinisikan kecerdasan tiruan sebagai suatu studi bagaimana membuat komputer mengerjakan sesuatu sedemikian rupa sehingga pada saat itu orang merasa mendapatkan hasil yang lebih baik.
4. **Staugaard** dan **Marvin Minsky** memberikan pernyataan bahwa kecerdasan tiruan adalah suatu ilmu pengetahuan yang dapat membuat mesin melakukan sesuatu yang memerlukan kecerdasan apabila dikerjakan oleh manusia. Dengan kalimat lain, kecerdasan tiruan adalah : **suatu mekanisasi atau duplikasi proses berpikir manusia.**
5. **Shcildt** (1987) mengatakan bahwa suatu program kecerdasan tiruan akan menunjukkan perilaku program yang menyerupai perilaku manusia jika menghadapi persoalan yang sama.
6. **Charniak** dan **McDermott** (1985) menambahkan bahwa proses pembelajaran ada program kecerdasan tiruan menggunakan model komputasi.

Menurut **Russel** dan **Norwig**, suatu kecerdasan tiruan mempunyai dimensi :

- Peniruan perilaku
- Peniruan cara berpikir manusia

Secara umum AI dapat dibagi dalam empat kategori yaitu :

1. Sistem yang dapat berpikir seperti manusia “*Thinking Humanly*”
2. Sistem yang dapat bertindak laku seperti manusia “*Acting Humanly*”
3. Sistem yang dapat berpikir secara rasional “*Thinking Rationally*”
4. Sistem yang dapat bertindak laku secara rasional “*Acting rationally*”

Beberapa bidang terapan AI adalah :

1. Robotics
2. Perception

3. Common Sense Reasoning
4. Medical Diagnosis
5. EGINEERING
6. Scientific Analysis
7. Financial Analysis
8. Expert Tasks
9. Mathematics
10. Games
11. DII

Ada beberapa keuntungan kecerdasan buatan dibanding kecerdasan alamiah, yaitu :

1. Lebih permanen
2. Memberikan kemudahan dalam duplikasi dan penyebaran
3. Relatif lebih murah dari kecerdasan alamiah
4. Konsisten dan teliti
5. Dapat didokumentasi
6. Dapat mengerjakan beberapa task dengan lebih cepat dan lebih baik dibanding manusia.

Keuntungan kecerdasan alamiah dibanding kecerdasan buatan :

1. Bersifat lebih kreatif
2. Dapat melakukan proses pembelajaran secara langsung sementara AI harus mendapatkan masukan berupa simbol dan representasi
3. Fokus yang luas sebagai referensi untuk pengambilan keputusan, sebaliknya AI menggunakan fokus yang sempit.

Perbedaan Komputasi AI dengan Proses Komputasi Konvensional

Data yang diproses oleh komputer konvensional dapat dilihat pada tabel berikut ini.

Proses	Tugas
Kalkulasi	Mengerjakan operasi – operasi matematis seperti : +, -, x, :, atau mencari akar persamaan, menyelesaikan rumus / persamaan
Logika	Mengerjakan operasi logika seperti and, or, invert

Penyimpanan	Menyimpan data dan gambar pada file
Retrieve	Mengakses data yang disimpan pada file
Translate	Mengkonversi data dari satu bentuk ke bentuk lain
Sort	Memeriksa data dan menampilkan dalam urutan yang diinginkan
Edit	Melakukan perubahan, penambahan, dan penghapusan pada data
Monitor	Mengamati event eksternal dan internal serta melakukan tindakan jika kondisi tertentu tercapai
Kontrol	Memberikan perintah atau mengendalikan peralatan luar

Perbandingan antara karakteristik pemrograman konvensional dan karakteristik pemrograman deklaratif dapat dilihat pada tabel di bawah ini.

Program Konvensional	Program berbasis pengetahuan
Menggunakan data	Menggunakan pengetahuan
Menggabungkan data dan pengendalian	Membagi bagian pengetahuan dan pengendalian
Pemrosesan menggunakan algoritma	Pemrosesan menggunakan inferensi
Manipulasi basis data yang besar	Manipulasi basis pengetahuan yang besar
Kemampuan pemikiran tidak ada	Kemampuan pemikiran terbatas tetapi dapat ditingkatkan

Topologi Kecerdasan Tiruan

Suatu sistem yang cerdas umumnya mempunyai ciri khas yang menunjukkan kemampuan dalam hal :

1. Menyimpan informasi
2. Menggunakan informasi yang dimiliki untuk melakukan sesuatu pekerjaan dan menarik kesimpulan.
3. Beradaptasi dengan keadaan baru
4. Berkomunikasi dengan penggunanya

Sistem cerdas secara umum terdiri atas dua kategori, yaitu sistem yang mampu melakukan emulasi kepakaran seseorang (*Knowledge Based Expert System*) dan

sistem yang mampu melakukan komputasi secara cerdas berdasarkan komputasional (*computational intelligence*).

Sistem yang mempunyai kecerdasan dengan *knowledge based expert* dan *computational intelligence* dibedakan kemampuannya dalam hal, antara lain kemampuan mengenali pola, kemampuan beradaptasi dengan pengaruh lingkungan dan atau alam, mempunyai sifat toleransi terhadap adanya kesalahan sampai pada batas tertentu (*fault tolerant*), kecepatan dengan error yang kecil, sehingga sistem dapat menghasilkan kinerja mendekati kinerja manusia. *Defense Advance Research Project Agency* (DARPA) membagi kecerdasan tiruan menjadi tiga kategori, yaitu :

- Kecerdasan tiruan yang diprogram
- Sistem pakar
- Sistem yang mampu belajar

Dari penjelasan di atas, kita dapat mengatakan bahwa kecerdasan tiruan mempunyai dua dimensi, yaitu dimensi peniruan perilaku dan dimensi peniruan cara berpikir. Peniruan kecerdasan perilaku meliputi kecerdasan tiruan yang diprogram. Peniruan cara berpikir terdiri atas dua bagian, yaitu peniruan proses berpikir dan peniruan proses komputasi.

Topologi Kecerdasan Tiruan yang selengkapnya dapat dilihat pada tabel di bawah ini.

Kategori	Pendekatan Desain	Akuisisi Pengetahuan	Implementasi	Keunggulan
Kecerdasan Tiruan Diprogram (<i>Smart System</i>)	Pemodelan dan Pemrograman	Pemrograman	Perangkat keras yang dijalankan oleh perangkat lunak	Prosedur baku atau statis
Kecerdasan Tiruan berbasis Pengetahuan				
- Sistem Pakar (ES)	Emulasi kepakaran dalam menyelesaikan masalah	Observasi kepakaran	Perangkat lunak	Diagnosis
- Pemrograman Cerdas (<i>Intelligent Programming</i>)	Emulasi cara berpikir manusia	Algoritma atau proses berpikir	Perangkat lunak	Algoritma
Kecerdasan Tiruan Komputasional				
- JST	Pilihan arsitektur JST	Belajar (dengan dan tanpa pengawasan)	Perangkat lunak atau simulasi dan atau perangkat keras	Pengenalan Pola
- Fuzzy	Penerapan bahasa manusia ke dalam	Berbasis rancangan aturan (<i>Rule Based</i>)	Perangkat lunak dan atau perangkat keras	Pengendalian

- Algoritma genetik	mesin Emulasi evolusi dalam biologi	Mekanisme evolusi	Perangkat lunak	Optimasi
---------------------	---	-------------------	-----------------	----------

Soal :

1. Dalam pengembangan AI, apakah diperlukan kontribusi dari disiplin ilmu lain ?
Jelaskan jawaban Anda ?
2. Apakah karakteristik yang menonjol dari suatu sistem cerdas dibandingkan suatu sistem yang biasa ? Jelaskan jawaban Anda !
3. *Major Interest* dari AI adalah pengetahuan sedangkan Pemrograman konvensional adalah data dan informasi. Apakah perbedaan data dan informasi dengan pengetahuan ?

Jawaban :

1. Kontribusi dari disiplin ilmu lain mutlak diperlukan, misalnya dalam sistem pakar untuk pendeteksian suatu penyakit, maka kontribusi dari seorang pakar yaitu dalam hal ini dokter adalah sangat diperlukan.
2. Suatu sistem yang cerdas umumnya mempunyai ciri khas yang menunjukkan kemampuan dalam hal :
 - Menyimpan informasi
 - Menggunakan informasi yang dimiliki untuk melakukan sesuatu pekerjaan dan menarik kesimpulan.
 - Beradaptasi dengan keadaan baru
 - Berkomunikasi dengan penggunaanya
3. Data adalah fakta yang masih belum diolah, sedangkan informasi adalah data yang sudah mengalami proses pengolahan. Informasi hanya berguna sebatas pada jenis informasi yang dimaksud. Sedangkan informasi yang sudah diolah untuk menghasilkan suatu kesimpulan disebut sebagai pengetahuan. Perbedaan lainnya adalah pengetahuan akan memberikan nilai tambah yang lebih besar daripada sekedar informasi. Dengan pengetahuan, maka akan dapat digali suatu 'emas', 'berlian' yang berguna bagi organisasi kita. Salah satu contohnya adalah aplikasi *data mining* yang dapat kita gunakan untuk menghasilkan suatu pengetahuan dari suatu kumpulan data dan informasi.

Sebagai contoh, jika Anda mempunyai kartu kredit, sudah pasti Anda bakal sering menerima surat berisi brosur penawaran barang atau jasa. Jika Bank pemberi kartu kredit Anda mempunyai 1.000.000 nasabah, dan mengirimkan sebuah (hanya satu) penawaran dengan biaya pengiriman sebesar Rp. 1.000 per nasabah maka biaya yang dihabiskan adalah Rp. 1 Milyar !! jika Bank tersebut mengirimkan penawaran sekali sebulan yang berarti 12x dalam setahun maka anggaran yang dikeluarkan per tahunnya adalah Rp. 12 Milyar!! Dari dana Rp. 12 Milyar yang dikeluarkan, berapa persenkah konsumen yang benar – benar membeli ? mungkin hanya 10% - nya saja. Secara harfiah, berarti 90% dari dana tersebut terbuang sia – sia.

Persoalan di atas merupakan salah satu persoalan yang dapat diatasi oleh *data mining* dari sekian banyak potensi permasalahan yang ada. *Data mining* dapat menambang data transaksi belanja kartu kredit untuk melihat manakah pembeli – pembeli yang memang potensial untuk membeli produk tertentu. Mungkin tidak sampai presisi 10%, tapi bayangkan jika kita dapat menyaring 20% saja, tentunya 80% dana dapat digunakan untuk hal lainnya.

Teknik Dasar Pencarian

Pencarian atau pelacakan merupakan salah satu teknik menyelesaikan permasalahan AI. Keberhasilan suatu sistem salah satunya ditentukan oleh kesuksesan dalam pencarian dan pencocokan. Ada beberapa aplikasi yang menggunakan teknik pencarian ini, yaitu :

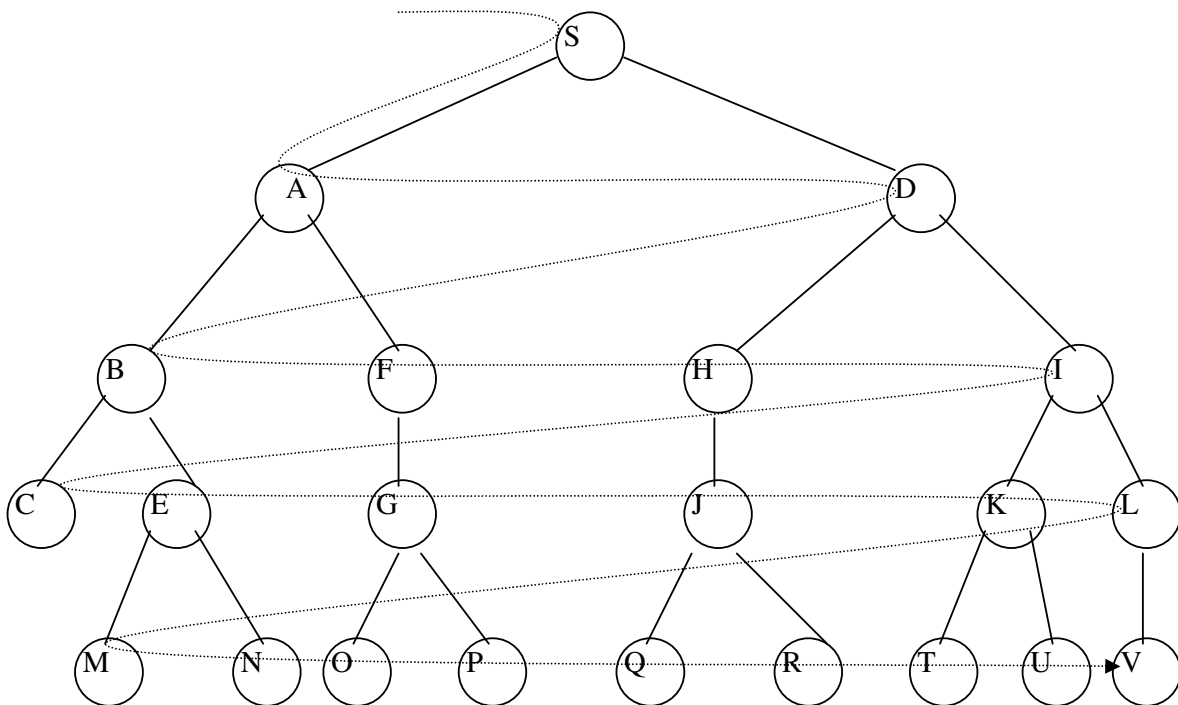
1. Papan *game* dan *puzzle* (Tic – tac – toe, catur, dan lain – lain)
2. Penjadwalan dan masalah *routing* (*travelling salesman problem*)
3. *Computer vision* dan pengenalan pola
4. Sistem pakar berbasis kaidah (*rule base expert system*)

Strategi Pencarian Mendalam

Pencarian boleh jadi merupakan hasil dari suatu solusi atau ruang keadaan yang mungkin telah dikunjungi semua, tetapi tanpa penyelesaian. Pencarian yang mendalam (*Exhaustive Search Strategy*) mungkin dilakukan dengan menggunakan strategi *Breadth First Search* atau *Depth First Search*). Kedua pencarian ini merupakan pencarian buta (*Blind Search*)

Breadth First Search

Prosedur *Breadth First Search* merupakan pencarian yang dilakukan dengan mengunjungi tiap – tiap node secara sistematis pada setiap level hingga keadaan tujuan (*goal state*) ditemukan. Atau dengan kata lain, penelusuran yang dilakukan adalah dengan mengunjungi node – node pada level yang sama hingga ditemukan *goal state* nya. Untuk lebih jelasnya, perhatikan ilustrasi dari *Breadth First Search* pada gambar berikut ini.



Pengimplementasian *Breadth First Search* dapat ditelusuri dengan menggunakan daftar (*List*) *open* dan *closed*, untuk menelusuri gerakan pencarian di dalam ruang keadaan, prosedur untuk *Breadth First Search* dapat dituliskan sebagai berikut (Luger : 2002) :

Prosedure *breadth first search*

Begin

Open := [Start];

Closed := [];

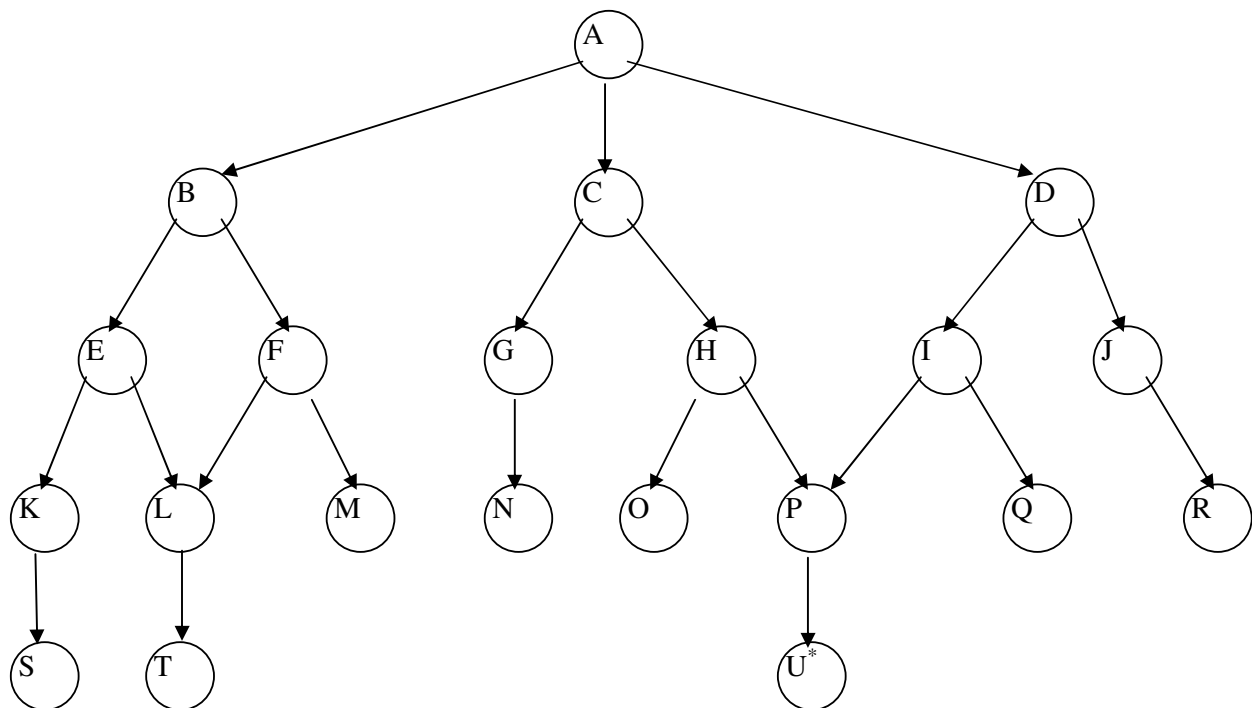
While Open ≠ [] do

Begin


```

Remove leftmost state from open, call it x
If X is a goal then return SUCCESS
Else
Begin
    Generate children of x;
    Put x on closed;
    Discard children of x is already open or closed;
    Put remaining children on right end of open;
End;
End;
Return FAIL
End.

```



Pada gambar di atas, U^* adalah node tujuannya (goal) sehingga bila ditelusuri menggunakan prosedur *Depth First Search*, diperoleh :

1. Open = [A]; Closed = []
2. Open = [B, C, D]; Closed = [A]

3. Open = [C, D, E, F]; Closed = [B, A]
4. Open = [D, E, F, G, H]; Closed = [C, B, A]
5. Open = [E, F, G, H, I, J]; Closed = [D, C, B, A]
6. Open = [F, G, H, I, J, K, L]; Closed = [E, D, C, B, A]
7. Open = [G, H, I, J, K, L, M]; Closed = [G, F, E, D, C, B, A]
8. dan seterusnya sampai U diperoleh atau open = []

Keuntungan :

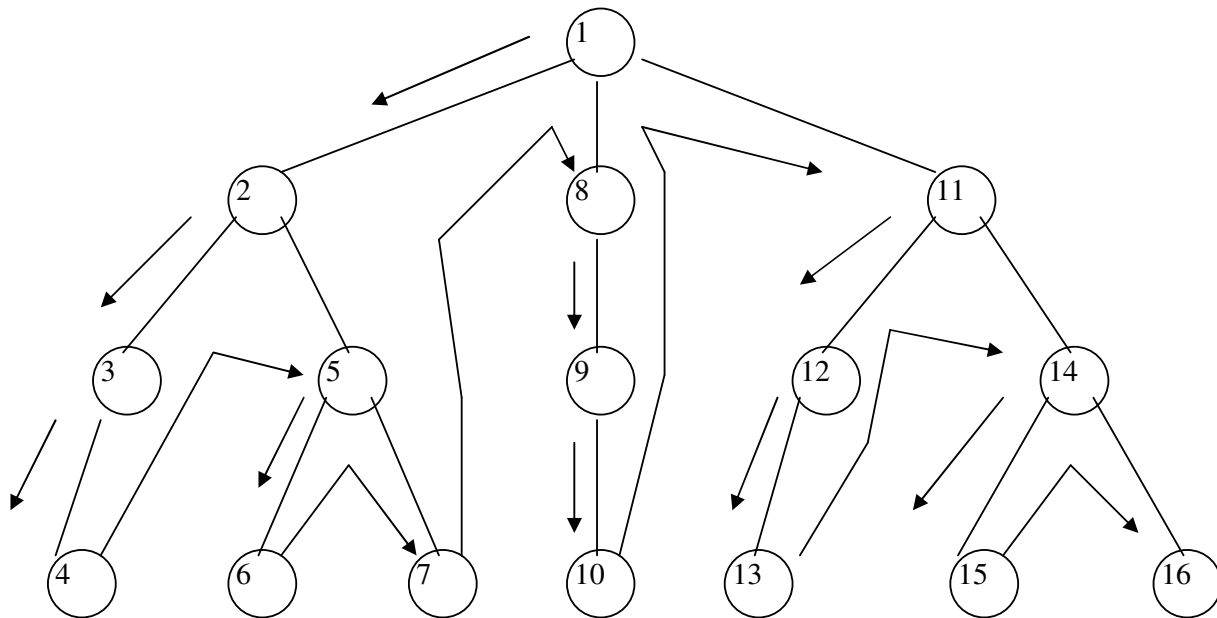
1. Tidak akan menemui jalan buntu dan jika ada satu solusi maka *Breadth First Search* akan menemukannya.
2. Jika ada lebih dari satu solusi maka solusi minimum akan ditemukan

Kerugian :

1. Membutuhkan memori yang besar, karena menyimpan semua node dalam satu pohon.
2. Membutuhkan sejumlah besar pekerjaan, khususnya bila solusi terpendek cukup panjang, karena jumlah node yang perlu diperiksa bertambah secara eksponensial terhadap panjang lintasan.

Depth First Search

Pencarian dengan metode ini dilakukan dari node awal secara mendalam hingga yang paling akhir (*dead end*) atau sampai ditemukan. Dengan kata lain, simpul cabang atau anak yang terlebih dahulu dikunjungi. Sebagai ilustrasinya dapat dilihat pada gambar berikut ini.



Prosedur *Depth First Search* dapat diimplementasikan dengan melakukan modifikasi proses *Breadth First Search* menjadi (Luger : 2002) :

Procedure depth first search

Begin

Open := [Start];

Closed := [];

While Open \neq [] do

Begin

Remove leftmost state from open, call it x

If X is a goal then return SUCCESS

Else

Begin

Generate children of x;

Put x on closed;

Discard children of x is already open or closed;

Put remaining children on left end of open;

End;

End;

Return FAIL

End.

Untuk memeriksa algoritma di atas, dapat dilihat bahwa keadaan – keadaan turunan (*descendant*) ditambahkan atau dihapuskan dari ujung kiri *open*. Untuk lebih jelasnya, dapat dilihat aplikasi algoritma tersebut untuk contoh sebelumnya seperti pada *Breadth First Search*. Langkah penelusuran adalah sebagai berikut.

1. Open = [A]; Closed = []
2. Open = [B, C, D]; Closed = [A]
3. Open = [E, F, C, D]; Closed = [B, A]
4. Open = [K, L, F, C, D]; Closed = [E, B, A]
5. Open = [S, L, F, C, D]; Closed = [K, E, B, A]
6. Open = [L, F, C, D]; Closed = [S, K, E, B, A]
7. Open = [T, F, C, D]; Closed = [L, S, K, E, B, A]
8. Open = [F, C, D]; Closed = [T, L, S, K, E, B, A]
9. Open = [M, C, D]; Closed = [F, T, L, S, K, E, B, A]
10. Open = [C, D]; Closed = [M, F, T, L, S, K, E, B, A]
11. Open = [G, H, D]; Closed = [C, M, F, T, L, S, K, E, B, A]
12. dan seterusnya sampai U diperoleh atau open = []

Studi Kasus :

Berikut ini adalah suatu studi kasus untuk penerapan pencarian dengan menggunakan metode *Depth First Search* dan *Breadth First Search* dalam pencarian solusi sistem pakar untuk mendeteksi kerusakan **handphone**.

Sebagai catatan :

Untuk mempermudah kita dalam pencarian, maka pohon yang digunakan adalah berupa pohon biner yaitu suatu pohon yang tiap node masing – masing memiliki 2 *sub node*.

Hasil akhir dari aplikasi adalah perbandingan antara metode DFS dan BFS.

Tabel untuk Aplikasi ini terdiri dari 3 tabel yaitu sebagai berikut. (Sebagai catatan, *rule* yang ada hanya digunakan sebagai contoh, dan mungkin tidak sesuai dengan keadaan nyata)

1. Tabel Kerusakan

tblkerusakan	
Kode_Kerusakan	Nama_Kerusakan
M0000	Ponsel Terkena cairan
M0001	ID Power Amplifier Rusak
M0002	HP No Voice
M0003	IC RF HILANG
M0004	Ponsel Terkena Benturan
M0005	baterai Drop

2. Tabel Gejala

tblgejala		
Kode_Kerusakan	Kode_Gejala	Nama_Gejala
M0000	G0001	Audio Rusak
M0000	G0002	IC PS Rusak
M0000	G0003	Kerusakan Hardware
M0000	G0004	Kerusakan Software
M0000	G0005	Ponsel terkena Benturan
M0001	G0006	Ponsel terkena cairan
M0002	G0007	Ponsel terlalu Panas
M0003	G0008	Baterai lemah
M0004	G0009	Ponsel jatuh/ terkena benturan keras
M0004	G0010	IC PA lemah
M0004	G0011	Arus pendek (korsleting)
M0004	G0012	Kerusakan software/Kerusakan hardware
M0004	G0013	IC RF rusak
M0005	G0014	Kerusakan software
M0005	G0015	Kerusakan hardware
M0004	G0016	ANTENA RUSAK
M0004	G0017	RUSAK

3. Tabel Solusi

tblsolusi		
Kode_Gejala	Kode_Solusi	Nama_Solusi
G0001	G0019	IC PA LEMAH
G0002	S0001	Panaskan dengan solder agar kaki IC menempel dengan baik
G0003	S0002	Ganti baru
G0004	S0003	Ganti dengan komponen yang baru
G0005	S0004	Persiapan alat-alat yang diperlukan sesuai ponsel yang di upgrade
G0006	S0005	Buka program pengecekan untuk seri ponsel tertentu pada

tblsolusi		
Kode_Gejala	Kode_Solusi	Nama_Solusi
		komputer
G0007	S0011	- Gunakan Multitester, jika pada ponsel ada arus, lakukan pengecekan - Cek, apakah ada arus jika tidak, komponen dari h/w harus diganti
G0010	S0020	ganti
G0011	S0021	Lepaskan
G0012	S0022	Lepaskan
G0008	S0023	-Lepaskan baterai
G0013	S0024	ganti baru
G0014	S0025	ganti baru
G0015	S0026	lepaskan
G0016	S0027	BUKA
G0017	S0028	Cabut
G0009	S0030	cas

Adapun tampilan Form yang dihasilkan adalah sebagai berikut.

1. Form DFS

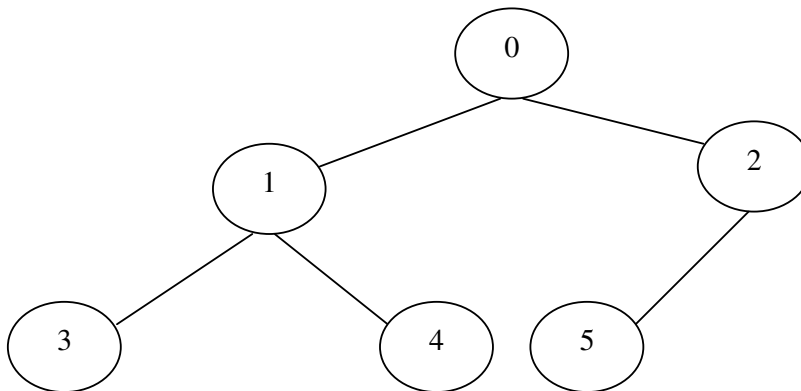
Berikut ini adalah tampilan Form pencarian secara DFS.

2. Form BFS

Berikut ini adalah tampilan Form pencarian secara BFS

Adapun langkah – langkah pengerjaannya adalah sebagai berikut.

1. Buatlah pohon biner, di mana jumlah nodenya berdasarkan jumlah kode kerusakan yang ada pada Tabel **kerusakan**. Kode kerusakan yang ada pada tabel kerusakan adalah **M0000**, **M0001**, **M0002**, **M0003**, **M0004**, dan **M0005**.



Maka bila pengerjaan dilakukan secara **DFS**, node – node yang terambil secara berturut – turut adalah :

Node **0** = **M0000**

Node **1** = **M0001**

Node **3** = **M0003**

Node **4** = **M0004**

Node 2 = **M0002**

Node 5 = **M0005**

Sedangkan bila pengerjaan dilakukan secara **BFS**, node – node yang diambil secara berturut – turut adalah sebagai berikut.

Node 0 = **M0000**

Node 1 = **M0001**

Node 2 = **M0002**

Node 3 = **M0003**

Node 4 = **M0004**

Node 5 = **M0005**

2. Sebagai contoh, bila yang diisikan oleh *user* adalah sebagai berikut.

Jenis Kerusakan	Ponsel Terkena Benturan
Jenis Gejala	Ponsel jatuh/ terkena benturan keras

3. Maka kita mulai dari Tabel Kerusakan, di mana **Ponsel Terkena Benturan** memiliki kode **M0004**.

- Pengerjaan Secara DFS

Proses 1, cari di kode M0000 dengan kode gejala G0001 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 2, cari di kode M0000 dengan kode gejala G0002 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 3, cari di kode M0000 dengan kode gejala G0003 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 4, cari di kode M0000 dengan kode gejala G0004 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 5, cari di kode M0000 dengan kode gejala G0005 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 6, cari di kode M0001 dengan kode gejala G0006 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 7, cari di kode M0003 dengan kode gejala G0008 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 8, cari di kode M0004 dengan kode gejala G0009 ternyata **ditemukan** jenis gejala ponsel jatuh/terkena benturan keras

Sehingga untuk mencari solusi, pencarian secara DFS membutuhkan **8 proses**.

- Pengerjaan secara **BFS**

Proses 1, cari di kode M0000 dengan kode gejala G0001 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 2, cari di kode M0000 dengan kode gejala G0002 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 3, cari di kode M0000 dengan kode gejala G0003 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 4, cari di kode M0000 dengan kode gejala G0004 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 5, cari di kode M0000 dengan kode gejala G0005 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 6, cari di kode M0001 dengan kode gejala G0006 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 7, cari di kode M0002 dengan kode gejala G0007 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 8, cari di kode M0003 dengan kode gejala G0008 ternyata jenis gejala yang ada **bukan** ponsel jatuh/terkena benturan keras

Proses 9, cari di kode M0004 dengan kode gejala G0009 ternyata **ditemukan** jenis gejala ponsel jatuh/terkena benturan keras.

Sehingga untuk mencari solusi, pencarian secara DFS membutuhkan **9 proses**.

4. Listing program yang selengkapnya dapat dilihat pada *file* yang disertakan

Diketahui suatu rute perjalanan yang harus dilalui oleh seorang sales di mana sales tersebut harus melalui setiap kota dan kembali lagi ke kota asal. Jumlah kota yang harus dilalui adalah 5 kota dengan masing – masing jarak $AB = 100$, $AC = 125$, $AD = 100$, $AE = 75$, $BC = 50$, $BD = 125$, $BE = 125$, $CD = 100$, $CE = 125$, dan $DE = 50$.

Tujuan masalah di atas adalah mencari jarak terpendek bagi sales untuk mengunjungi setiap kota. Tentukan lintasan terpendek dari permasalahan di atas jika diasumsikan bahwa sales tersebut mulai dari kota A dengan menggunakan Simple Hill Climbing dan Steepest Hill Climbing serta gambarkan grafnya.

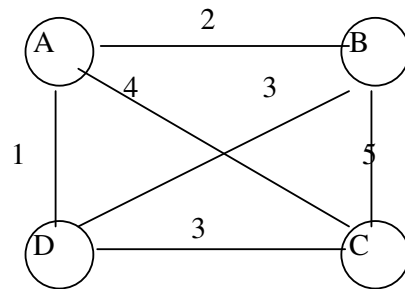
Pencarian Heuristik

Pencarian Heuristik adalah pencarian yang membutuhkan informasi. Biasanya digunakan untuk pencarian jarak terpendek. Sebagai contoh terdapat permasalahan Travelling Salesman Problem (TSP)

Diketahui suatu rute perjalanan yang harus dilalui oleh seorang sales di mana sales tersebut harus melalui setiap kota tepat sekali. Jumlah kota yang harus dilalui adalah 4 kota dengan masing – masing jarak $AB = 2$, $AC = 4$, $AD = 1$, $BC = 5$, $BD = 3$, dan $CD = 3$. Tujuan dari masalah di atas adalah mencari jarak (panjang lintasan) terpendek bagi sales untuk mengunjungi setiap kota. Tentukan lintasan terpendek dari permasalahan di atas.

Penyelesaian :

Permasalahan di atas sebaiknya digambarkan terlebih dahulu dalam bentuk graf, seperti terlihat pada gambar berikut ini.



Penyelesaian dengan Metode Generate and Test

Pada permasalahan ini terdapat 4 kota sehingga kombinasi lintasan yang ada adalah sebanyak $4! = 24$ kombinasi lintasan. Kombinasi lintasan yang menjadi solusi adalah lintasan yang menghasilkan panjang lintasan paling pendek di antara lintasan lainnya. Untuk mempermudah penghitungan panjang masing – masing lintasan, digunakan tabel di bawah ini.

No Pencarian	Lintasan	Panjang Lintasan	Lintasan yang Dipilih	Panjang Lintasan
1	ABCD	10	ABCD	10
2	ABDC	8	ABDC	8
3	ACBD	12	ABDC	8
4	ACDB	10	ABDC	8
5	ADBC	9	ABDC	8
6	ADCB	9	ABDC	8
7	BACD	9	ABDC	8
8	BADC	6	BADC	6
9	BCAD	10	BADC	6
10	BCDA	9	BADC	6
11	BDAC	8	BADC	6
12	BDCA	10	BADC	6
13	CABD	9	BADC	6
14	CADB	8	BADC	6
15	CBAD	8	BADC	6
16	CBDA	9	BADC	6
17	CDAB	6	BADC atau CADB	6
18	CDBA	8	BADC atau CADB	6
19	DABC	8	BADC atau CADB	6

20	DACB	10	BADC atau CADB	6
21	DBAC	9	BADC atau CADB	6
22	DBCA	12	BADC atau CADB	6
23	DCAB	9	BADC atau CADB	6
24	DCBA	10	BADC atau CADB	6

Dari tabel di atas, solusi pertama yang kita bangkitkan adalah ABCD (lintasan = 10), kemudian solusi kedua yang kita bangkitkan adalah ABDC (lintasan = 8). Ternyata, lintasan kedua menghasilkan jarak yang lebih pendek dari lintasan pertama sehingga dipilih lintasan ABDC = 8. Lakukan untuk langkah selanjutnya, kemudian bandingkan sehingga dari tabel di atas kita peroleh lintasan dengan panjang lintasan terpendek adalah BADC atau CDAB dengan panjang lintasan = 6. Solusi untuk permasalahan ini adalah BADC atau CDAB.

Kelemahan dari pencarian ini adalah perlu dibangkitkannya semua kemungkinan solusi yang ada sehingga apabila ditambahkan satu kota lagi untuk permasalahan TSP di atas menjadi 5 kota, kita akan memerlukan 120 kombinasi lintasan, kecuali bila diberikan beberapa kondisi tertentu misalnya kota awal bagi si sales telah ditentukan terlebih dahulu.

Hill Climbing

Metode ini hampir sama dengan metode *generate and test* dan merupakan salah satu variasi dari metode tersebut. Yang membedakan kedua metode ini adalah umpan balik (*feed back*) yang berasal dari prosedur pengujian digunakan untuk memutuskan arah gerak dalam pencarian.

Ada dua macam metode *hill climbing*, yaitu *simple hill climbing* dan *steepest ascent hill climbing*.

Simple Hill Climbing

Algoritma untuk *simple hill climbing* adalah sebagai berikut.

1. Mulai dari keadaan awal, lakukan pengujian. Jika merupakan tujuan maka berhenti. Jika sebaliknya, lanjutkan dengan keadaan sekarang sebagai keadaan awal.

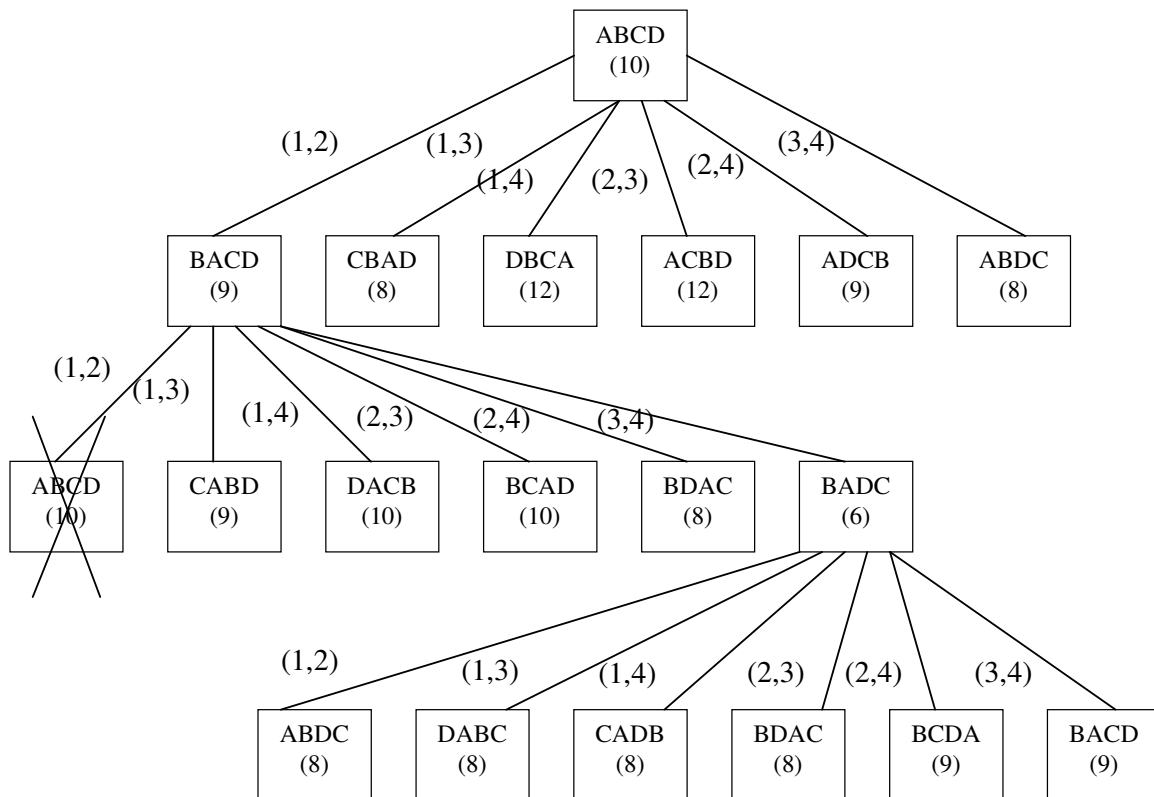
2. Ulangi langkah – langkah berikut hingga solusi ditemukan, atau sampai tidak ada operator baru yang akan diaplikasikan pada keadaan sekarang.
 - a. Pilih operator yang belum pernah digunakan, gunakan operator ini untuk mendapatkan keadaan yang baru.
 - b. Evaluasi keadaan yang baru tersebut.
 - i. Jika keadaan baru merupakan tujuan maka keluar
 - ii. Jika bukan tujuan, namun nilainya lebih baik daripada keadaan sekarang, maka jadikan keadaan baru tersebut menjadi keadaan sekarang.
 - iii. Jika keadaan baru tidak lebih baik daripada keadaan sekarang maka lanjutkan iterasi.

Sebagai ilustrasi pencarian solusi dengan *simple hill climbing*, kita gunakan contoh masalah TSP pada *generate and test*. Operator yang digunakan adalah operator yang bisa menghasilkan kombinasi lintasan kota yang berbeda – beda, yaitu dengan cara menukar posisi masing – masing kota. Untuk mempermudah penukaran posisi, kita cukup menukar posisi 2 kota, operator untuk kombinasi lintasan dengan menukar posisi 2 kota dapat dihitung dengan rumus $\frac{n!}{2!(n-2)!}$ sehingga operator yang kita peroleh untuk masalah ini adalah $\frac{4!}{2!(4-2)!}$ kombinasi operator, yaitu :

1. (1,2) : Menukar posisi kota kesatu dengan kota kedua
2. (1,3) : Menukar posisi kota kesatu dengan kota ketiga
3. (1,4) : Menukar posisi kota kesatu dengan kota keempat
4. (2,3) : Menukar posisi kota kedua dengan kota ketiga
5. (2,4) : Menukar posisi kota kedua dengan kota keempat
6. (3,4) : Menukar posisi kota ketiga dengan kota keempat

Pada pencarian dengan menggunakan *simple hill climbing*, penggunaan urutan operator harus konsisten, tidak boleh berbeda dalam setiap level. Urutan penggunaan operator juga sangat menentukan kecepatan dalam menemukan solusi.

Setelah urutan operator ditentukan, gunakan algoritma pengerjaan sesuai aturan metode *simple hill climbing*. Keadaan awal yang dibangkitkan, misalnya kita pilih ABCD, sesuai dengan gambar berikut ini.



Pencarian pada *simple hill climbing* mulai dilihat dari anak kiri. Apabila nilai heuristik anak kiri lebih baik maka dibuka untuk pencarian selanjutnya. Jika tidak, barulah kita dapat melihat tetangga dari anak kiri tersebut, dan seterusnya.

Pada gambar di atas, terlihat bahwa pada level satu BACD nilai heuristiknya lebih baik daripada ABCD ($ABCD = 10 < BACD = 9$) sehingga yang dibuka untuk pencarian selanjutnya adalah BACD tanpa harus mengecek nilai heuristik tiap node yang selevel BACD. Pada level dua untuk operator (1,2), lintasan yang ditemui adalah ABCD. Oleh karena lintasan tersebut telah dilalui dan dicek sebelumnya maka tidak perlu dilakukan pengecekan lagi. Kemudian, dipilih node CBAD yang ternyata nilai heuristiknya sama dengan BACD sehingga pengecekan dilanjutkan pada tetangga CABD yaitu DACB, yang ternyata nilai heuristiknya juga lebih jelek dari BACD ($BACD = 8 < DACB = 10$). Cek node tetangganya, yaitu BCAD, yang nilai heuristiknya juga lebih jelek dibandingkan BACD ($BACD = 8 < BCAD = 10$). Lakukan pengecekan untuk node selanjutnya, yaitu BDCA, yang ternyata nilai heuristiknya juga lebih jelek dibandingkan BACD ($BACD = 8 < BDCA = 10$). Pengecekan terakhir dilakukan pada node BADC yang ternyata nilai heuristiknya

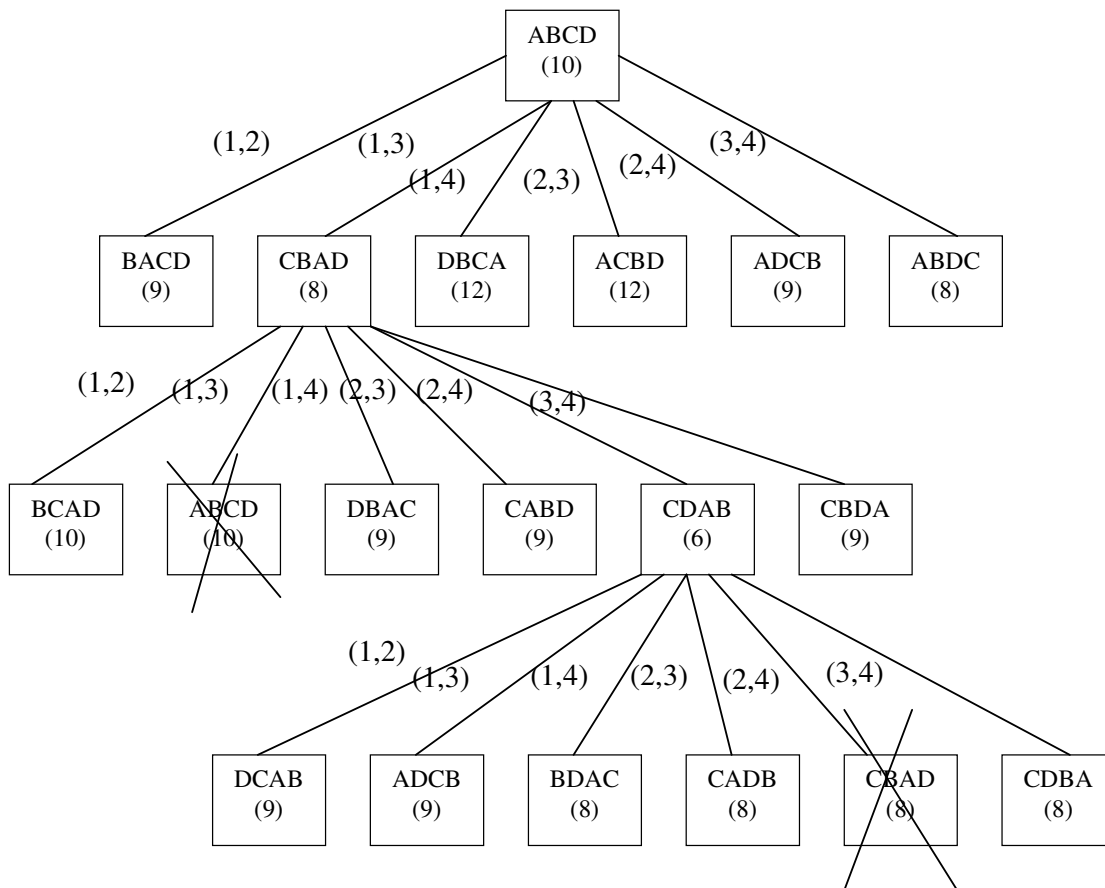
lebih baik dari BACD ($BACD = 8 > BADC = 6$) sehingga untuk pencarian selanjutnya, yang dibuka adalah node BADC.

Node yang dibuka untuk level tiga adalah BADC, setelah dieksplorasi dengan urutan operator yang telah ditentukan, secara berturut – turut diperoleh node berikut : node ABDC yang nilai heuristiknya lebih jelek dari BADC ($BADC = 6 < ABDC = 8$), DABC yang juga nilai heuristiknya lebih jelek dari BADC ($BADC = 6 < DABC = 8$), CADB yang nilai heuristiknya juga jelek ($BADC = 6 < CADB = 8$), BDAC yang nilai heuristiknya juga jelek ($BADC = 6 < BDAC = 8$), BCDA yang nilai heuristiknya juga lebih jelek dari BADC ($BADC = 6 < BCDA = 9$), selanjutnya pengecekan pada node terakhir BACD yang ternyata nilai heuristiknya juga lebih jelek dari BDAC ($BDAC = 6 < BCAD = 9$). Oleh karena pada level tiga tidak ada node yang menghasilkan nilai heuristik yang lebih baik dari BDAC maka pencarian solusi dihentikan. Solusi yang ditemukan dari metode *simple hill climbing* tersebut adalah BADC dengan panjang lintasan adalah 6. Berbeda apabila kita menggunakan metode *generate and test* di mana semua solusi dapat diperoleh yaitu BADC atau CDBA. Pada metode *simple hill climbing*, tidak semua solusi dapat kita temukan.

Penggunaan operator sangat mempengaruhi penemuan solusi pada metode *simple hill climbing*, begitu juga apabila penggunaan operator dibatasi misalnya untuk masalah TSP sebelumnya. Pada masalah TSP tersebut, diperlukan penggunaan 6 operator, tetapi kita batasi hanya menggunakan 4 operator. Dengan demikian, solusi yang dihasilkan belum tentu merupakan solusi optimal karena tidak semua kemungkinan solusi dilakukan pengecekan yang disebabkan oleh kurangnya penggunaan operator yang diperlukan untuk membangkitkan kemungkinan solusi yang ada.

Steepest Ascent Hill Climbing

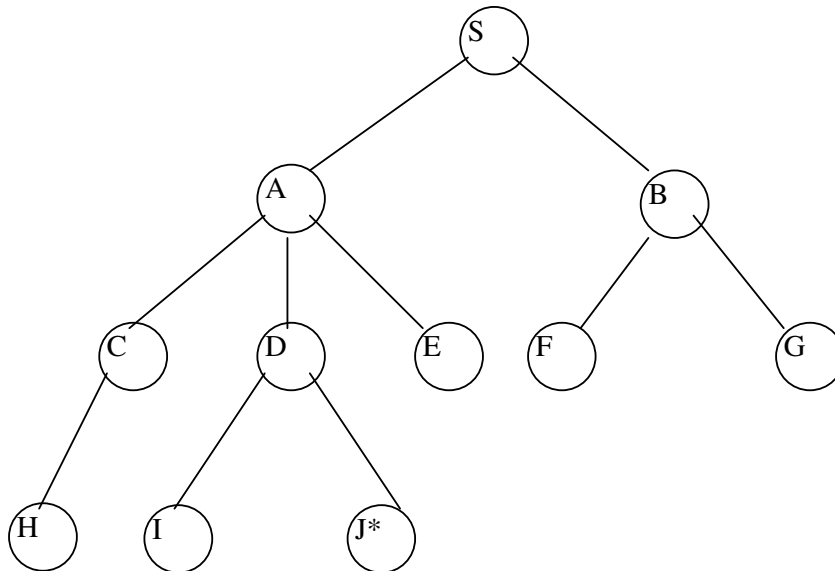
Steepest Ascent Hill Climbing hampir sama dengan *simple hill climbing*, dan yang membedakan keduanya adalah pada gerakan pencarian yang bila menemukan satu node tidak langsung berhenti tetapi dilanjutkan dengan mencari apakah ada node lain yang memiliki nilai heuristik yang lebih baik. Dalam hal ini, urutan penggunaan operator tidak menentukan penemuan solusi.



Dari gambar di atas, solusi bagi masalah TSP tersebut adalah CDAB dengan panjang lintasan sebesar 6. Metode *Steepest Ascent Hill Climbing* tidak harus melihat naka kiri pertama kali, tetapi dengan mencari semua nilai heuristik yang lebih baik pada node – node selevel. Gambar di atas menunjukkan bahwa pada level satu, yang memiliki nilai heuristik yang lebih baik dari lintasan ABCD yang panjangnya 10 adalah lintasan CBAD dan ABDC dengan lintasan sama dengan 8. Oleh karena ada dua node yang nilainya sama dan lebih baik dibandingkan node tetangganya yang lain maka kita dapat memilih salah satu node yang ingin dibuka untuk melanjutkan pencarian solusi selanjutnya. Pada gambar, dipilih node CBAD untuk pencarian selanjutnya. Pada level dua, diperoleh node CDAB dengan panjang lintasan adalah 6 sebagai lintasan yang lebih baik nilai heuristiknya dibandingkan node lintasan sebelumnya dan juga bila dibandingkan dengan node – node tetangganya. Pada level tiga tidak ditemukan nilai heuristik yang lebih baik dibandingkan lintasan CDAB sehingga pencarian berhenti dan solusi yang diperoleh untuk lintasan terpendek adalah CDAB dengan panjang lintasan adalah 6.

Soal Latihan :

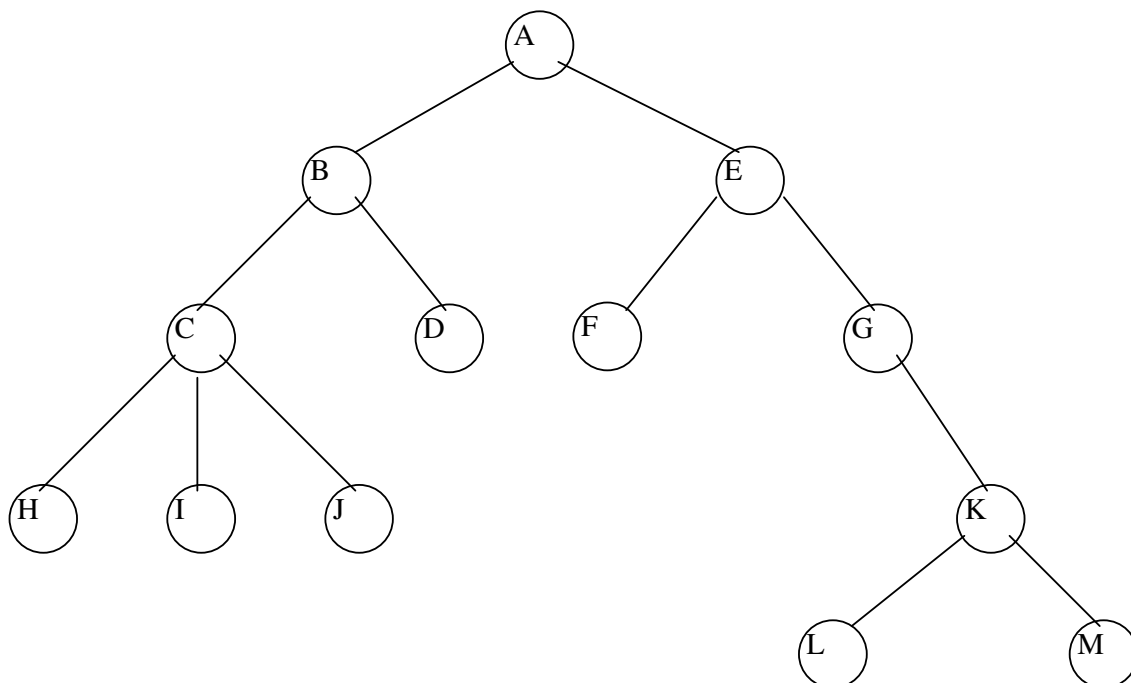
1. Apakah yang dimaksud dengan pencarian (*searching*) ?
2. Diketahui gambar pohon berikut ini :



J* adalah simpul tujuan (Goal) dari pohon di atas.

Implementasikan algoritma pencarian *Best First Search* dan *Depth First Search* untuk pohon di atas.

3. Diketahui gambar pohon berikut ini :



M^* adalah simpul tujuan (*goal*) dari pohon di atas.

Implementasikan algoritma pencarian *Best First Search* dan *Depth Fisrt Search* untuk pohon di atas.

4. Diketahui suatu rute perjalanan yang harus dilalui oleh seorang sales di mana sales tersebut harus melalui setiap kota dan kembali lagi ke kota asal. Jumlah kota yang harus dilalui adalah 5 kota dengan masing – masing jarak $AB = 100$, $AC = 125$, $AD = 100$, $AE = 75$, $BC = 50$, $BD = 125$, $BE = 125$, $CD = 100$, $CE = 125$, $DE = 50$.

Tujuan dari masalah di atas adalah mencari jarak (lintasan) terpendek bagi sales untuk mengunjungi setiap kota. Tentukan lintasan terpendek dari permasalahan di atas jika diasumsikan bahwa sales tersebut mulai dari kota A dengan menggunakan metode *Simple Hill Climbing* dan *Steepest Hill Climbing* serta gambarkan grafnya bagi permasalahan di atas.

Fuzzy Logic

Lofti Zadeh mengembangkan Logika Fuzzy pada tahun 1964. Dasar pemikirannya adalah tidak ada keadaan yang selalu “benar” dan “salah” atau “on” dan “off”. Tetapi ada bayangan di antara dua nilai ekstrem. Ada banyak permasalahan yang terkadang melibatkan sesuatu kondisi yang tidak secara tegas menyatakan benar ataupun salah, seperti sedang, tinggi, kaya, miskin, dsb. Di sini fungsi dan peranan dari Fuzzy Logic untuk menyelesaikan permasalahan yang dimaksud. Fuzzy Logic sering digunakan untuk menyajikan suatu himpunan yang batasannya tidak jelas, yang disebut juga sebagai Fuzzy Set. Contoh : himpunan mahasiswa yang tingginya sedang.

Alasan menggunakan Fuzzy :

1. Konsep logika *fuzzy* mudah dimengerti oleh manusia
2. Logika fuzzy sangat fleksibel
3. Memiliki toleransi terhadap data – data yang tidak tepat
4. Dapat membangun dan mengaplikasikan pengalaman – pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.

Aplikasi Logika *Fuzzy*

1. Tahun 1990 pertama kali mesin cuci dengan logika Fuzzy di Jepang (Matsushita Electric Industrial Company). Sistem Fuzzy digunakan untuk menentukan putaran yang tepat secara otomatis berdasarkan jenis dan banyaknya kotoran serta jumlah yang akan dicuci. Input yang digunakan : seberapa kotor, jenis kotoran, banyaknya yang dicuci. Mesin ini menggunakan sensor optik, mengeluarkan cahaya ke air dan mengukur bagaimana cahaya tersebut sampai ke ujung lainnya. Makin kotor, maka sinar yang sampai semakin redup. Sistem juga mampu menentukan jenis kotoran tersebut apakah daki / minyak.
2. Transmisi otomatis pada mobil Nissan, menghemat bensin 12 – 17%
3. Kereta bawah tanah Sendai mengontrol pemberhentian otomatis pada area tertentu.
4. Ilmu kedokteran dan biologi, seperti sistem diagnosa kanker.
5. Manajemen dan pengambilan keputusan misal tata letak pabrik berdasarkan logika Fuzzy, pembuatan *games* berdasarkan logika Fuzzy.
6. Ilmu lingkungan, misal kendali kualitas air, prediksi cuaca.
7. Teknik, misal perancangan Jarkom, prediksi adanya gempa bumi, dan lain – lain

Himpunan Tegas (*Crisp*)

Himpunan tegas adalah nilai keanggotaan suatu *item* x dalam suatu himpunan A , yang sering ditulis dengan $\mu_A[x]$, memiliki 2 kemungkinan, yaitu :

- 1, yang berarti bahwa item tersebut (x) anggota himpunan A
- 0, yang berarti bahwa item tersebut (x) bukan anggota himpunan A

Contoh :

- $S = [1, 2, 3, 4, 5, 6]$ adalah himpunan semesta
 $A = [1, 2, 3]$
 $B = [3, 4, 5]$

Jadi :

Nilai keanggotaan 2 pada himpunan A yaitu $\mu_A[2] = 1$, karena $2 \in A$

Nilai keanggotaan 3 pada himpunan A yaitu $\mu_A[3] = 1$, karena $3 \in A$

Nilai keanggotaan 4 pada himpunan A yaitu $\mu_A[4] = 0$, karena 4 bukan anggota himpunan A

Nilai keanggotaan 2 pada himpunan B yaitu $\mu_B[2] = 0$, karena 2 bukan merupakan anggota himpunan B

Nilai keanggotaan 3 pada himpunan B yaitu $\mu_B[3] = 1$, karena $3 \in B$

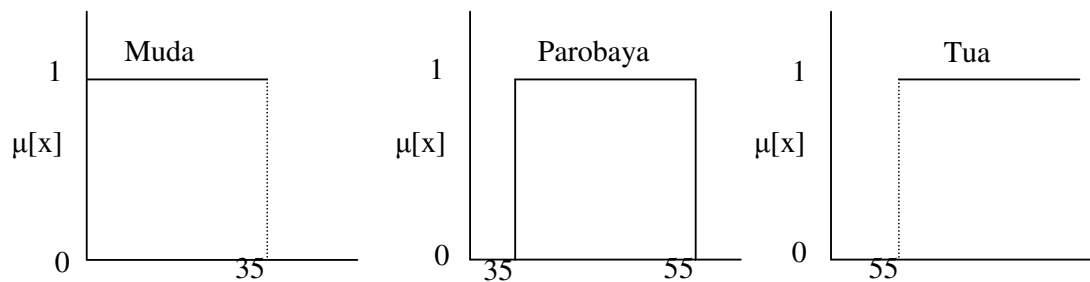
- Misal variabel umur dibagi menjadi 3 kategori, yaitu :

Muda, umur < 35 tahun

Parobaya $35 \leq \text{umur} \leq 55$ tahun

Tua umur > 55 tahun

Nilai keanggotaan secara grafis himpunan Muda, Parobaya, dan Tua, adalah sebagai berikut.



Usia 34 tahun, maka dikatakan Muda yaitu $\mu_{\text{Muda}}[34] = 1$

Usia 35 tahun, maka dikatakan Tidak Muda yaitu $\mu_{\text{Muda}}[35] = 0$

Usia 35 tahun, maka dikatakan Parobaya yaitu $\mu_{\text{Parobaya}}[35] = 1$

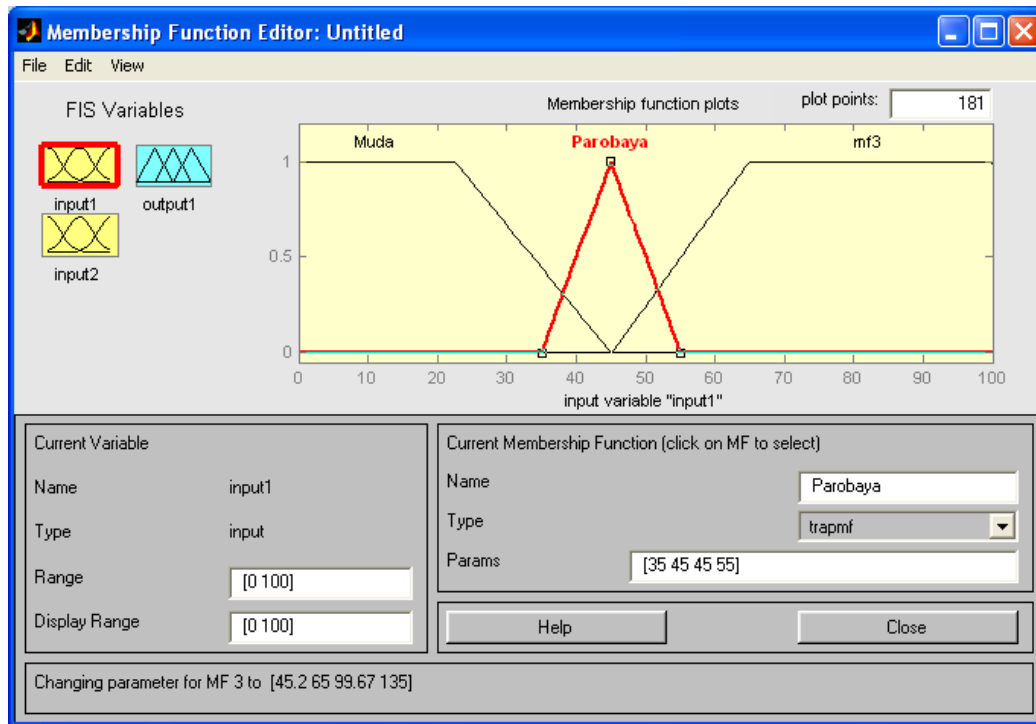
Usia 34 tahun, maka dikatakan Tidak Parobaya yaitu $\mu_{\text{Parobaya}}[34] = 0$

Usia 35 tahun kurang 1 hari, maka dikatakan Tidak Parobaya yaitu $\mu_{\text{Parobaya}}[35-1 \text{ hari}] = 0$

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

Himpunan Fuzzy

Himpunan Fuzzy digunakan untuk mengantisipasi hal tersebut di atas. Seseorang dapat masuk dalam 2 himpunan yang berbeda, Muda dan Parobaya, Parobaya dan Tua, dan sebagainya. Seberapa besar eksistensinya dalam himpunan tersebut dapat dilihat pada nilai keanggotaannya.



Usia 20 tahun termasuk dalam himpunan muda dengan $\mu_{Muda}[20] = 1$

Usia 40 tahun termasuk dalam himpunan Muda dengan $\mu_{Muda}[40] = 0.2$ dan termasuk juga ke dalam himpunan Parobaya dengan $\mu_{Parobaya}[40] = 0.4$

Components of Fuzzy Logic

Secara umum Fuzzy Logic dapat dinyatakan dalam 3 komponen atau 3 tahapan yaitu sebagai berikut.

1. Fuzzyfication : menentukan input dari suatu fuzzy dalam suatu bentuk linguistic data.
2. Inference Rule : merelasikan input parameter dan output parameternya.
3. Defuzzification : mengkonversi data linguistic yang sudah diperloeh ke dalam bentuk data nilai yang pasti.

Proses Fuzzifikasi

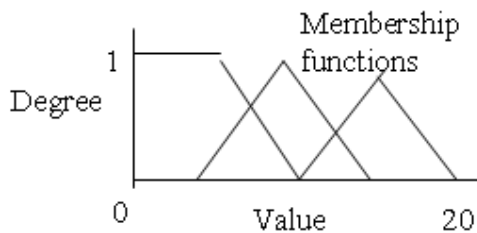
Proses fuzzifikasi dilakukan dengan menggunakan suatu *linguistic variable*. Yang dimaksud dengan linguistic variabel di sini adalah semua variabel yang digunakan di dalam if – then rules variable. Contoh : If demand = high and distance = short and

type of road = average then link = positive medium. Yang dimaksud dengan linguistic variabel pada kondisi if – then ini adalah **demand**, **distance**, **type of road**, dan **link**. Linguistic variabel di sini dapat dibagi ke dalam sejumlah possible value atau nilai yang mungkin, seperti yang ditunjukkan pada tabel di bawah ini.

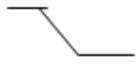



Linguistic Variable	Possible Values (Terms)
1. Demand	{ Vlow, Low, Medium, High, Vhigh }
2. Distance	{ Vlong, Long, Average, Short, Vshort }
3. Type of Road	{ Bad, Fair, Good }
4. Link	{ Zero, Positive Medium, Positive Small }

Membership Functions

Nilai – nilai yang ada pada suatu linguistic variabel sendiri dapat disajikan ke dalam bentuk suatu membership function. Membership Function adalah suatu bentuk kurva yang menggambarkan value yang ada pada suatu linguistic variabel. Untuk lebih jelasnya dapat dilihat pada gambar di bawah ini.



Standard Membership Functions

Shape	MF
	Z-type
	Λ -type
	Π -type
	S-type

Inference Rules

- ✦ Fuzzy inference rules relate the input linguistic variables to the output linguistic variable.
- ✦ Fuzzy inference step identifies the rules that apply to the current situation and computes the values of the output linguistic variable.
- ✦ The computation of the fuzzy inference consists of two components:
 1. Aggregation: computation of the IF part of the rules.
 2. Composition: computation of the THEN part of the rules.
- ✦ Example 3:

IF Demand = Medium AND Distance = Short THEN Link = Pos_Small

- Three operators are used in the majority of fuzzy logic applications:

Boolean Opr.	Fuzzy Logic Opr.
AND	$\mu_{A \wedge B} = \min(\mu_A, \mu_B)$
OR	$\mu_{A \vee B} = \max(\mu_A, \mu_B)$
NOT	$\mu_{\neg A} = 1 - \mu_A$

Composition

- Each rule defines an action to be taken in the THEN part.
- The degree to which the action is valid is given by the adequateness of the rule to the current situation.
- Example 4:
 - If Demand is High and Distance is Very Short and Type is BAD Then Link is Neg_Short
 - If Demand is High and Distance is Very Short and Type is FAIR Then Link is Pos_Medium
 - If Demand is High and Distance is Very Short and Type is GOOD Then Link is Pos_Medium

Defuzzification

- ⊕ Defuzzification is a reverse process of Fuzzification.
- ⊕ Defuzzification translates the value of a linguistic variable into real value called **crisp value**.
- ⊕ The objective is to derive the non fuzzy (crisp) value that best represents the fuzzy value of the linguistic output variable.
- ⊕ Common defuzzification methods:
 1. Center of Area/Gravity (COA/COG)
 2. Center of Maximum (COM)
 3. Mean of Maximum (MOM)

MOM Defuzzification Method

⊕ The Mean of Maxima (MOM) generates a crisp control action by averaging the support values which their membership values reach the maximum (Berenji, 1992).

⊕ MOM is calculated by

$$Z^* = \sum_{j=1}^l z_j / l$$

where l is the number of quantized z values which reach their maximum memberships.

COA Defuzzification Method

- ✦ This method determines the centre of the area of the combined membership functions.
- ✦ Assuming that a control action with a pointwise membership function μ_C has been produced, the Centre of Area (COA) method (Berenji, 1992) calculates:

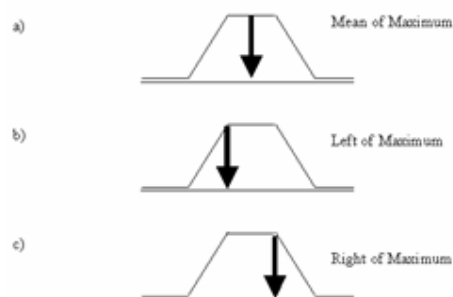
$$Z^* = \frac{\sum_{j=1}^q z_j \mu_C(z_j)}{\sum_{j=1}^q \mu_C(z_j)}$$

where:

- q = number of quantization levels of the output
- z_j = the amount of control output at the quantization level j
- $\mu_C(z_j)$ = membership value in C .

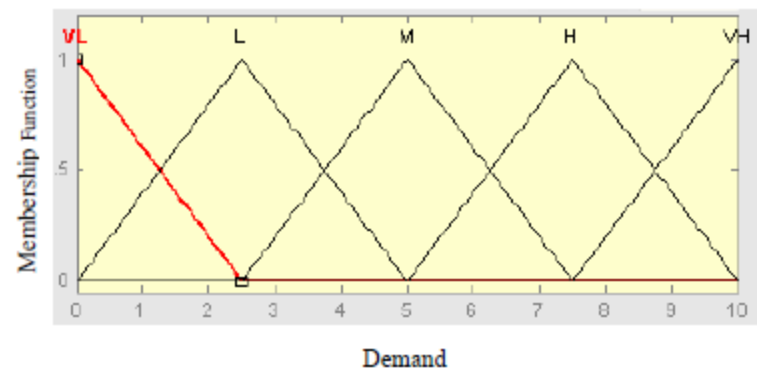
MOM Variants

- ✦ There are also variants of the MOM defuzzification method. They differ from MOM by the computation of the most typical value of a membership function.
- ✦ For Λ -type membership functions, the most typical value is uniquely defined.
- ✦ For Π -type membership functions, variants are possible (Figure 6).



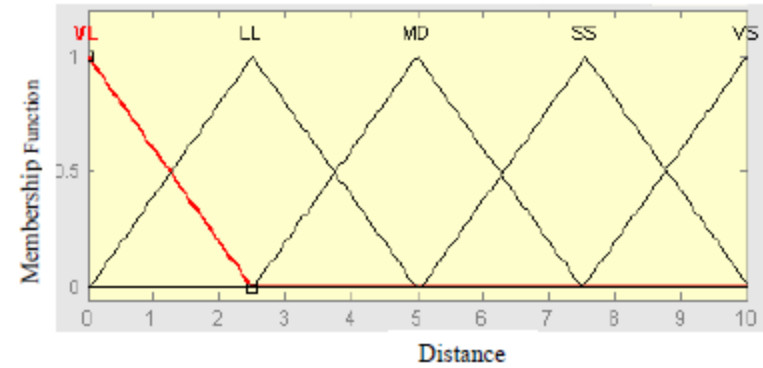
Contoh soal :

Diberikan fuzzy set sebagai berikut :



Legend:
VL = Very Low, L = Low, M = Medium, H = High, VH = Very High

Figure 7.8: Fuzzy Sets of Demand



Legend:
VL = Very Long, LL = Long, MD = Medium, SS = Short, VS = Very

Figure 7.10: Fuzzy Sets of Link Distance

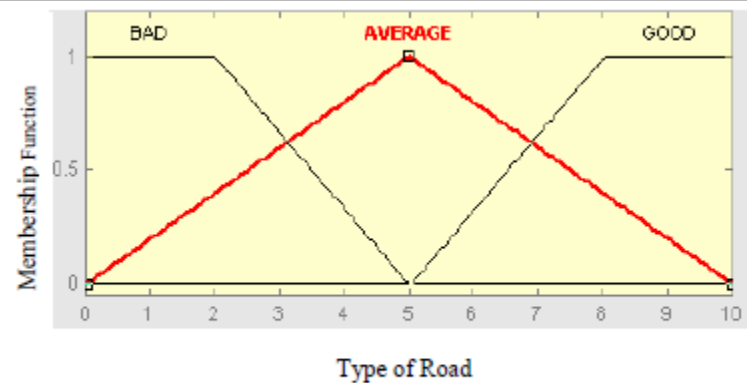
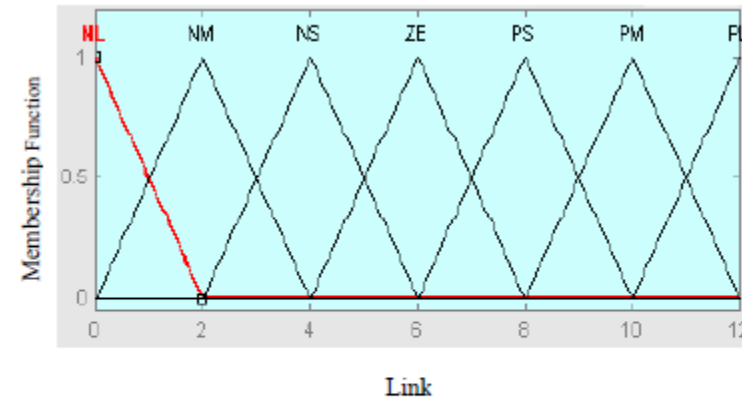


Figure 7.12: Fuzzy Sets of Type of Road



Legend:
NL = Negative Large, NM = Negative Medium, NS = Negative Small,
ZE = Zero

Figure 7.14: Link Membership Function

Adapun Rule yang ada adalah :

Rule 1 : If Demand = High and Distance = Short and Type of Road = Average then Link = Positive_Medium

Rule 2 : If Demand = High and Distance = Medium and Type of Road = Good then Link = Positive_Small

Rule 3 : If Demand = Medium and Distance = Very_Short and Type of Road = Good then Link = Zero

Berdasarkan Rule yang ada, lakukanlah proses Defuzzyfication !

Jika dihitung dengan menggunakan metode COA :

$$= \frac{8 \times 0 + 9 \times 0.4 + 10 \times 0.4 + 11 \times 0.4 + 12 \times 0}{0.4 + 0.4 + 0.4}$$

$$= 10$$

Jika dihitung dengan menggunakan metode MOM :

$$= \frac{9 \times 0.4 + 10 \times 0.4 + 11 \times 0.4}{0.4 + 0.4 + 0.4}$$

$$= 10$$

Implementasi Fuzzy Logic dengan Menggunakan Matlab

Sebagai contoh kasus, kita ingin menentukan berapa tip yang akan diberikan kepada pelayan di sebuah restoran.

Input :

1. *Quality of Service* [0 .. 10]
2. *Quality of The Food* [0..10]

Output :

Tips [0..30]

Knowledge : dari pelanggan dan pelayan yang sudah bertahun – tahun / berpengalaman.

Adapun aturan yang digunakan di sini adalah 3 yaitu sebagai berikut.

1. If the service is poor or the food is rancid, then tip is cheap
2. If the service is good, then tip is average
3. If the service is excellent or the food is delicious then tip is generous.

Service :

Type membership function : Triangular

Membership function : 3

1. Poor [1 2.5 4)
2. Good [4 6 8]
3. Excellent [8 9 10]

Food :

Type Membership Function : Trapezoid

Membership Function : 2

1. Rancid [1 2 3 5]
2. Delicious [5 6,5 8 10]

Output :

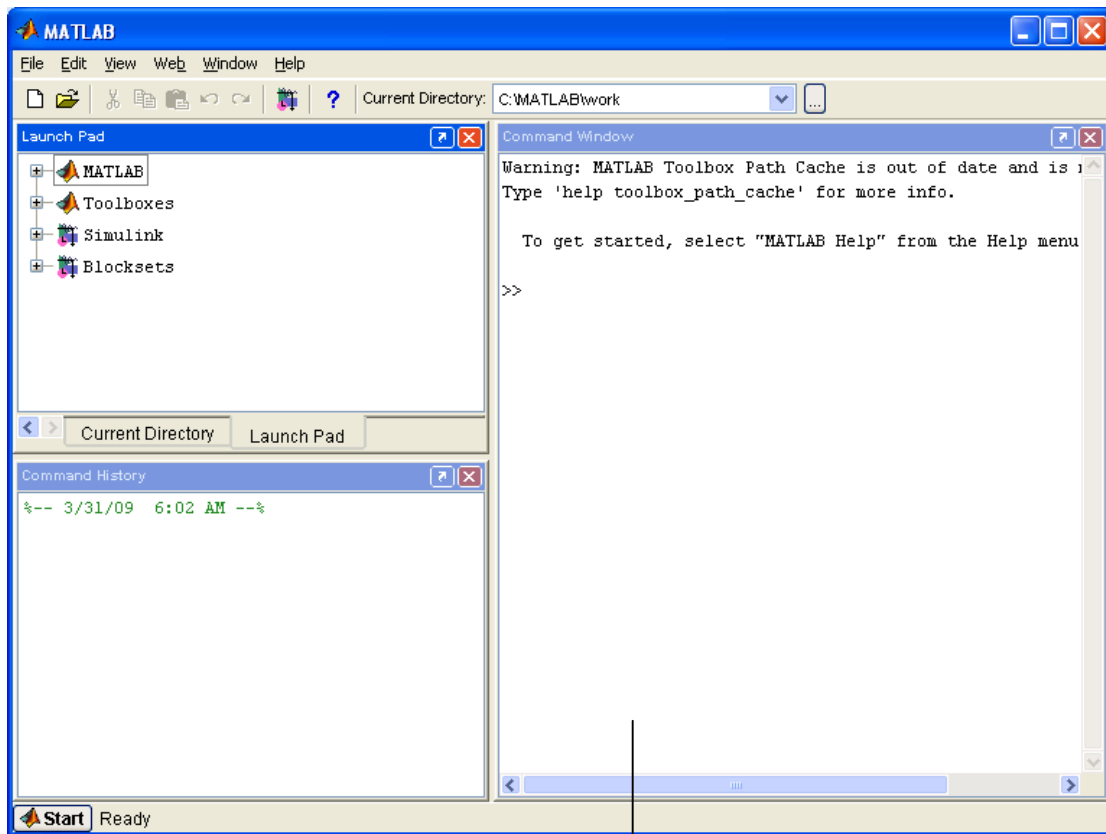
Type Membership Function : Triangular

Membership Function :

1. Cheap [1 5 10]
2. Average [10 15 20]
3. Generous [20 25 30]

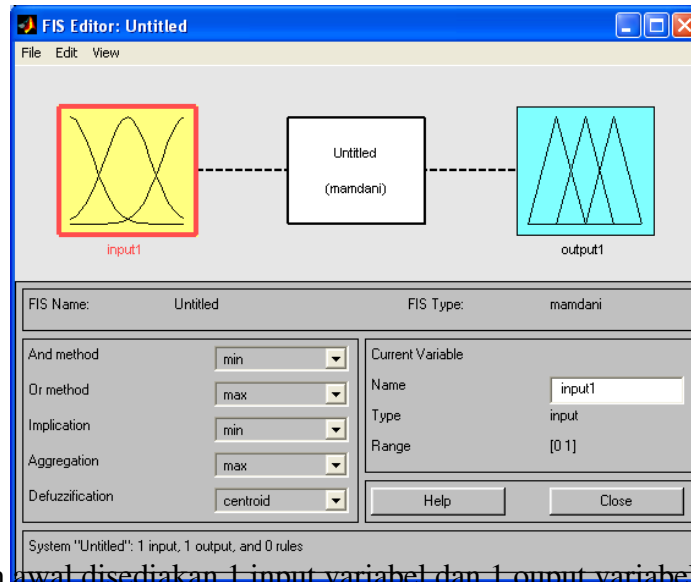
Langkah – langkah pemakaian Fuzzy Logic dengan menggunakan Matlab adalah sebagai berikut :

1. Jalankan program Matlab Anda, Anda akan menjumpai tampilan berikut.

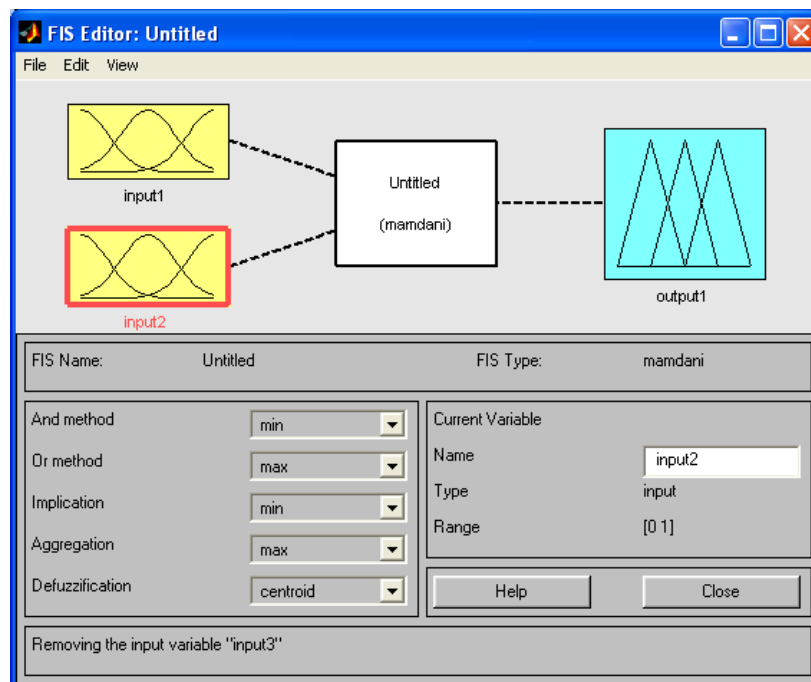


Command Window

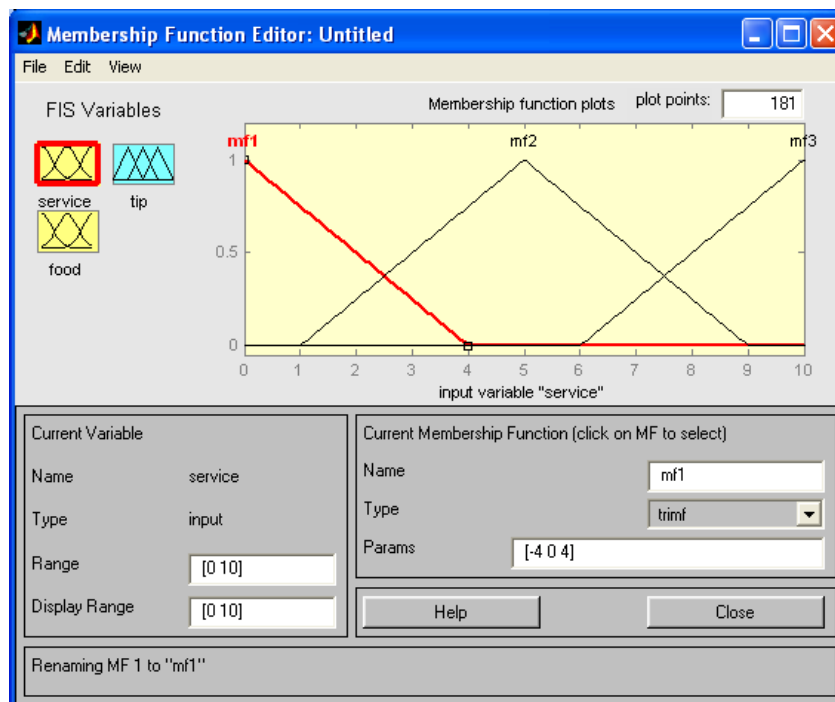
2. Pada Command Window ketikkan **Fuzzy** lalu tekan Enter, maka akan muncul tampilan **FIS Editor** berikut.



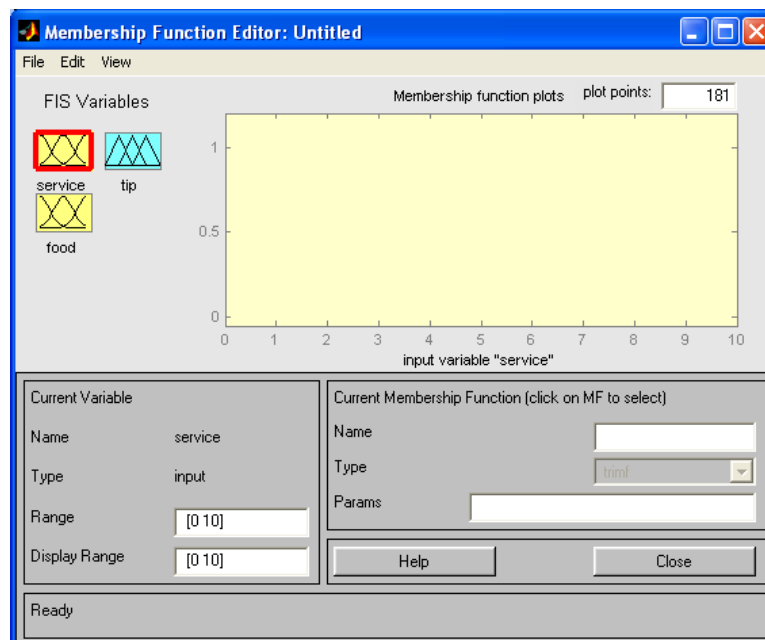
3. Pada keadaan awal disediakan 1 input variabel dan 1 output variabel.
4. Anda bisa menambahkan input variabel dengan cara klik menu **edit > Add Variabel > Input**. Sehingga jumlah input variabel yang kita dapat adalah berjumlah 2 buah seperti yang tampak pada gambar di bawah ini.



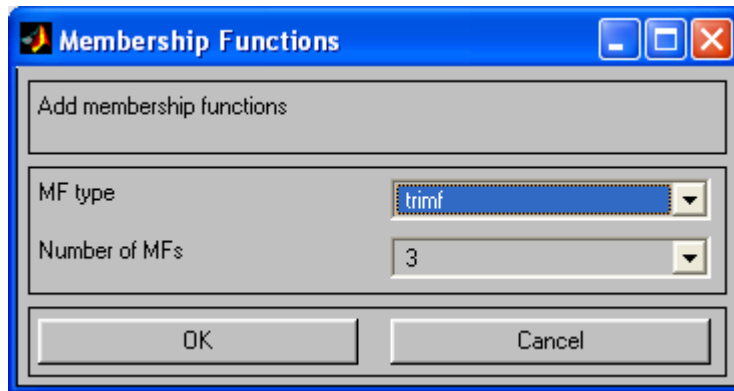
5. Ubah nama input 1 menjadi service, input 2 menjadi food, dan output 1 menjadi Tip. Kemudian lakukan penyesuaian terhadap input 1 dengan cara melakukan double click terhadap input 1 / service sehingga diperoleh tampilan berikut.



6. Pada soal, diketahui bahwa tipe membership function dari service adalah GaussMF, maka lakukanlah penyesuaian seperti permintaan yang ada pada soal. Caranya adalah sebagai berikut.
- a. Klik menu **Edit > Remove All MFS**, sehingga membership function yang ada sudah bersih, seperti yang tampak pada gambar berikut ini.

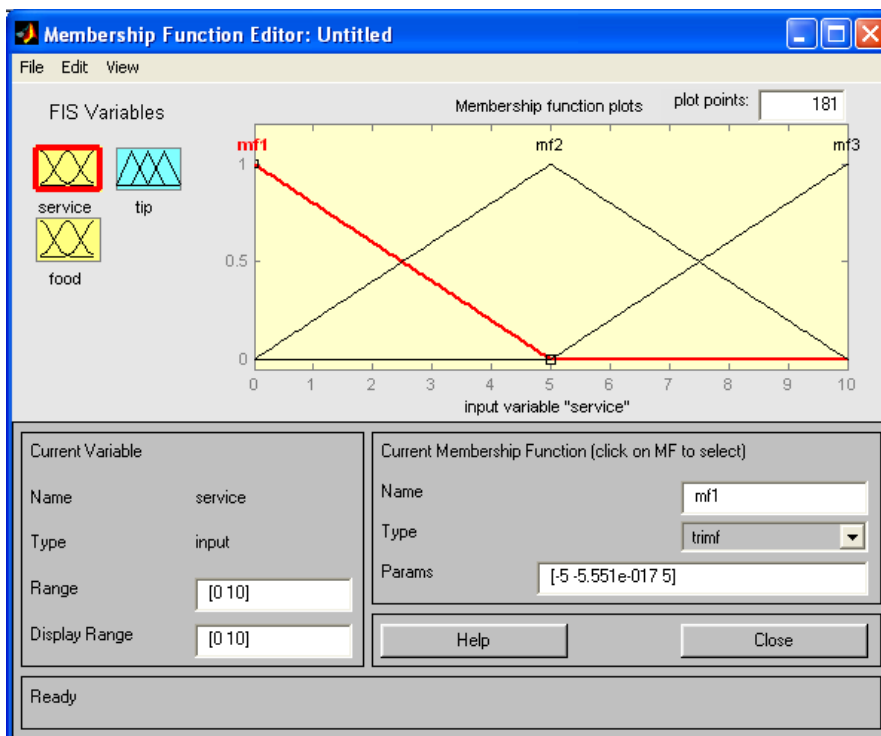


- b. Selanjutnya klik menu **Edit > Add MFS**, sehingga akan muncul tampilan berikut.

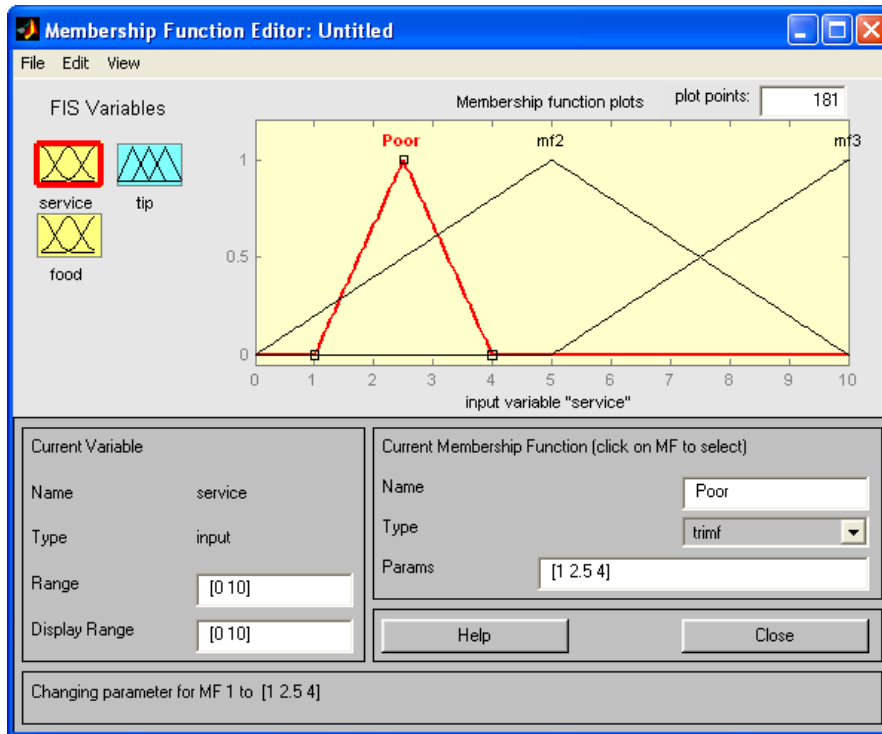


Pilih : MF Type : Trimf dan Number of MFS : 3

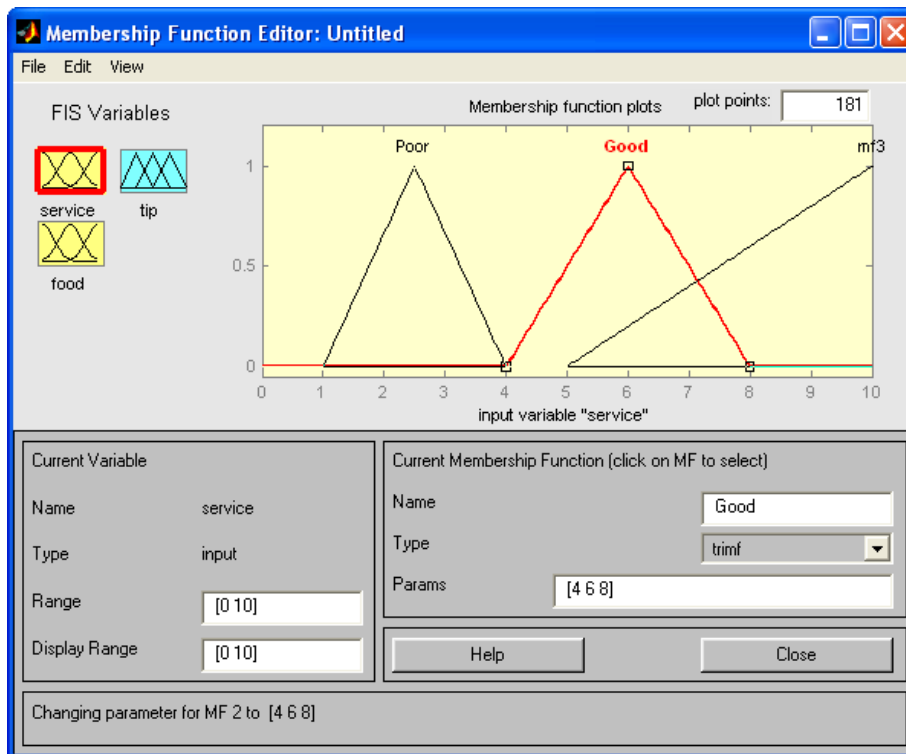
- c. Sehingga muncul tampilan berikut ini.



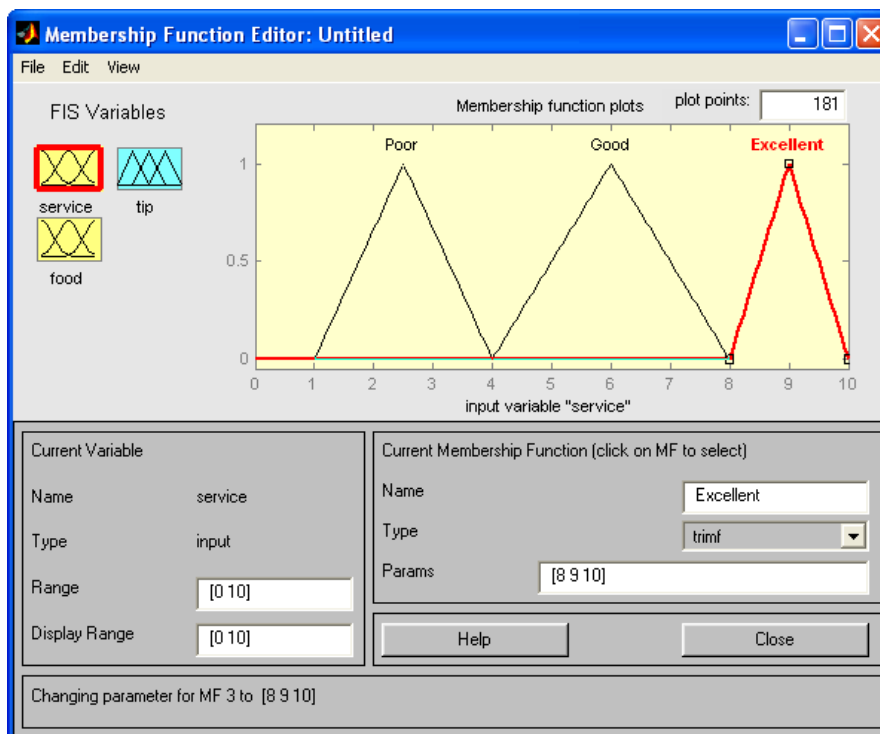
- d. Ubah Name : mf1 menjadi Poor params [1 2.5 4], sehingga muncul tampilan berikut ini.



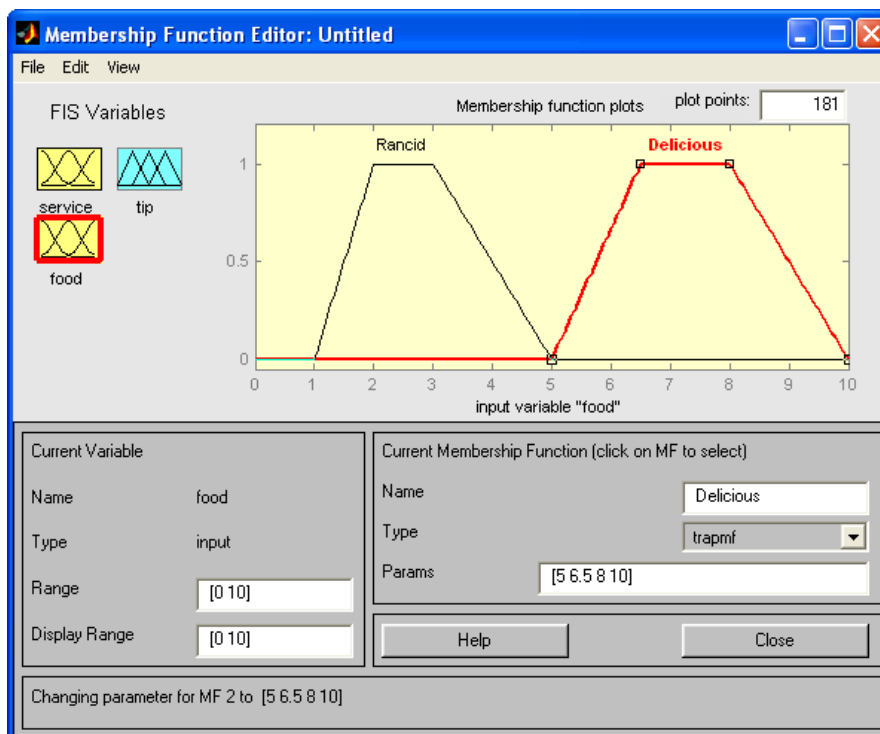
e. Lakukan hal yang sama, pada mf2 ubah menjadi Good[4 8], muncul :



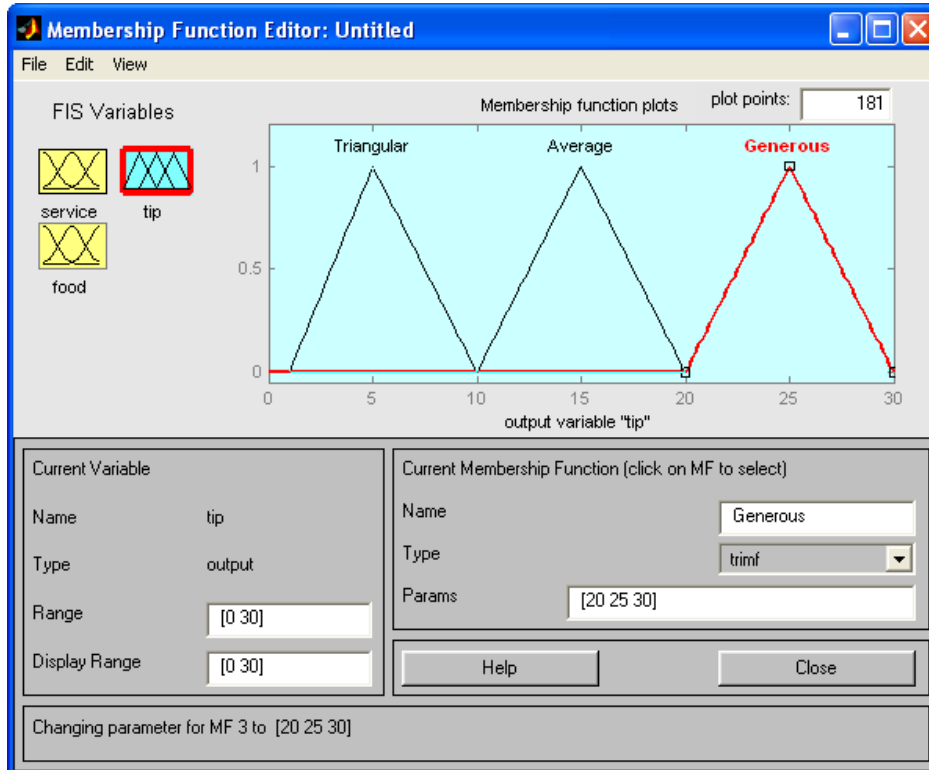
f. Ubah juga mf3 menjadi Excellent [8 9 10]



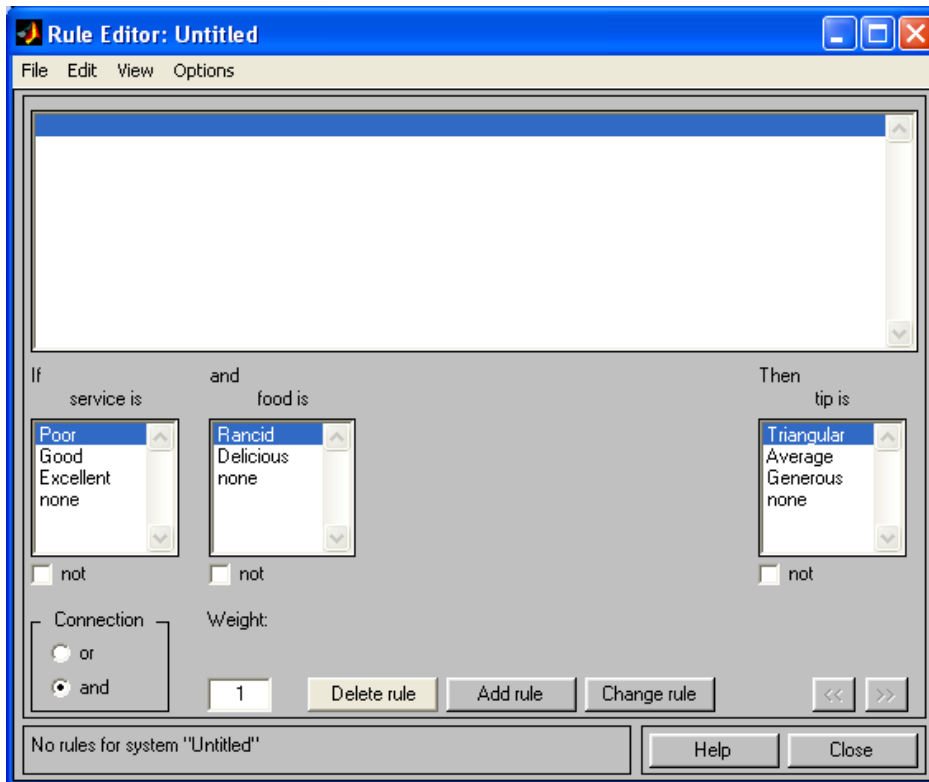
g. Lakukan hal yang sama untuk Food. Sehingga diperoleh tampilan berikut ini.



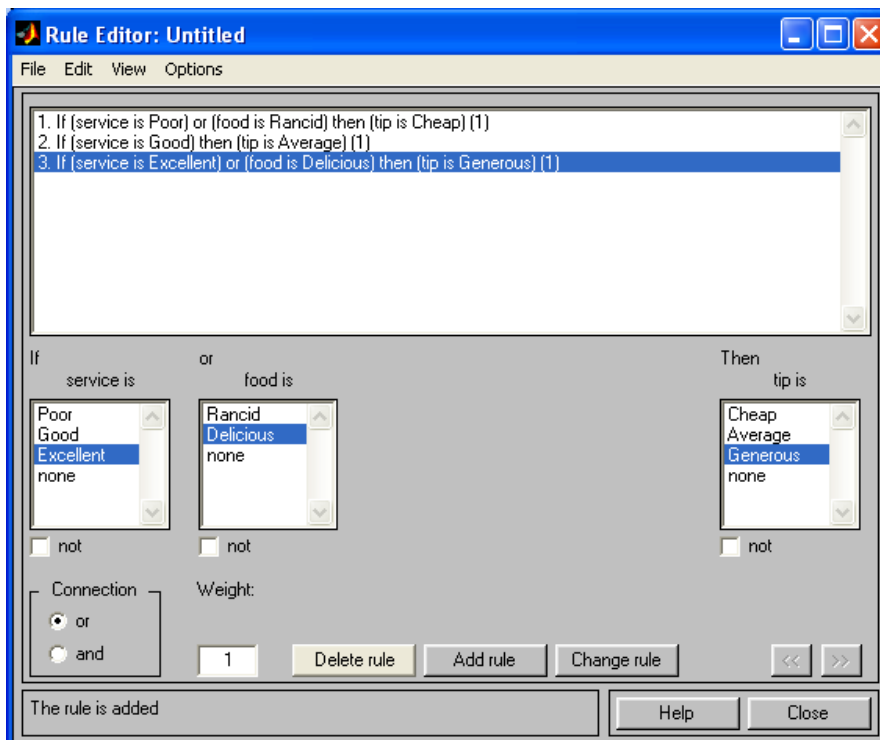
h. Lakukanlah hal yang sama untuk tips sehingga diperoleh tampilan berikut ini.



7. Kemudian masukkan *rule* yang ada dengan cara klik menu **Edit > Rules**. Sehingga akan muncul tampilan berikut ini.

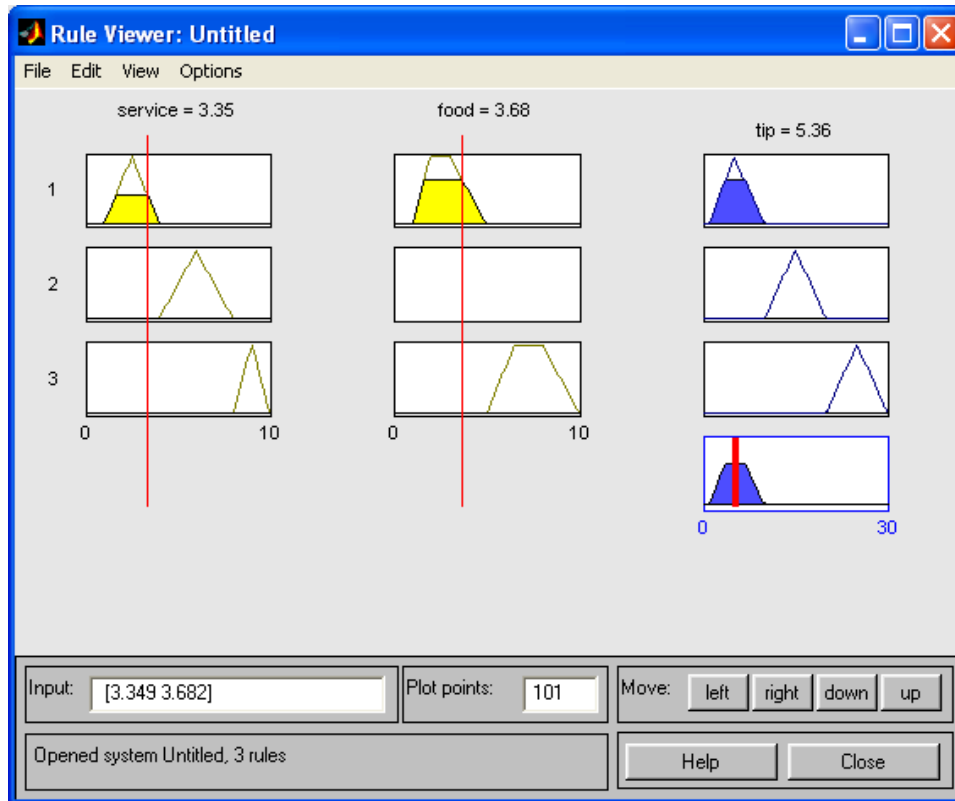


8. Masukkan *rule – rule* yang diketahui sehingga akan muncul tampilan berikut ini.



9. Setelah selesai klik tombol Close

10. Untuk melihat hasilnya klik menu **View > Rules**. Sehingga akan muncul tampilan berikut ini.



Studi Kasus :

Untuk studi kasus ini maka kita akan membuat aplikasi peramalan cuaca berbasis Logika Fuzzy. Informasi cuaca merupakan hal penting yang mendukung kelancaran kegiatan manusia. Peramalan cuaca berbasis Logika Fuzzy dilakukan dengan memberikan masukan berupa suhu, tekanan, dan kelembapan relatif. Kemudian, hasilnya adalah masukan peramalan cuaca yang berupa suhu rata – rata, kelembapan relatif rata – rata, serta keadaan cuaca.

Fungsi keanggotaan (*membership function*) himpunan Fuzzy untuk masing – masing masukan adalah :

1. Fungsi Keanggotaan masukan suhu (temperatur) : *very low*, *low*, *medium*, *high*, dan *very high*.

2. Fungsi keanggotaan masukan tekanan (*pressure*) : *very low*, *low*, *medium*, *high*, dan *very high*.
3. Fungsi keanggotaan masukan kelembaban relatif (*relative humidity*) : *very low*, *low*, *medium*, *high*, dan *very high*.

Fungsi keanggotaan himpunan fuzzy untuk masing – masing keluaran adalah :

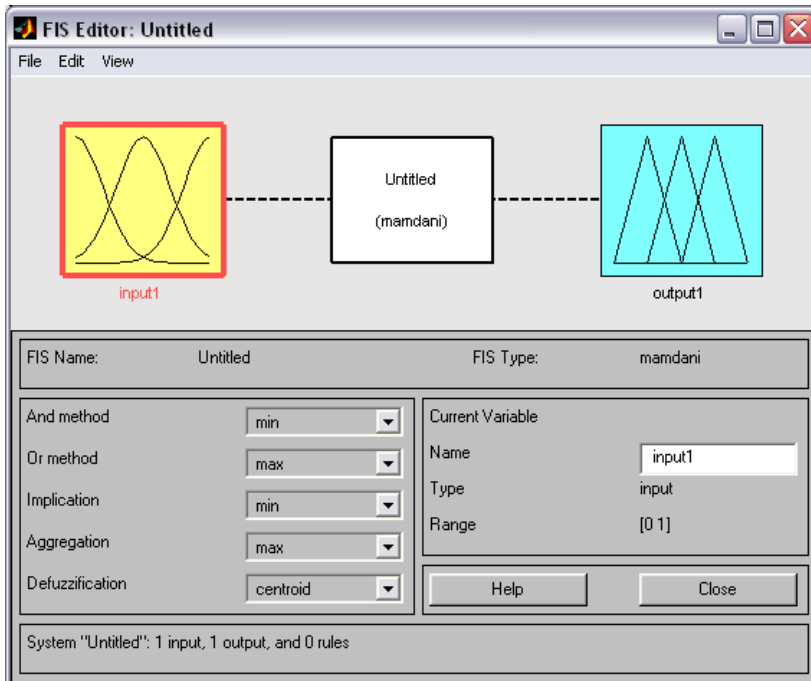
1. Fungsi keanggotaan keluaran suhu rata – rata : *very low*, *low*, *medium*, *high*, dan *very high*.
2. Fungsi keanggotaan keluaran kelembaban relatif rata – rata : *very low*, *low*, *medium*, *high*, dan *very high*.
3. Fungsi keanggotaan keluaran keadaan cuaca : hujan, hujan berawan, berawan, cerah berawan, dan cerah.

Peramalan cuaca memberikan masukan suhu, tekanan, dan kelembaban relatif suatu daerah dengan batasan – batasannya, yaitu :

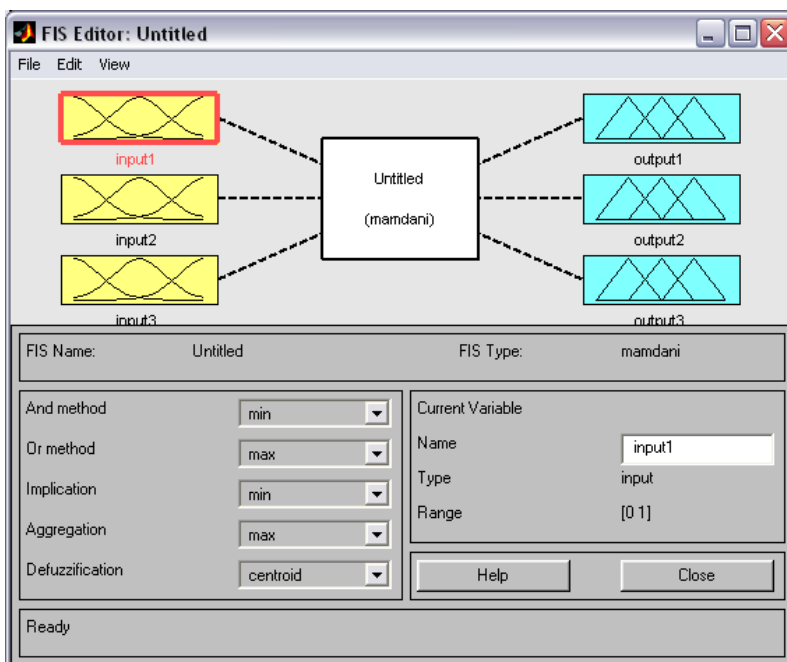
1. Batasan suhu, yaitu berada di antara 24.3°C sampai dengan 31°C untuk mendefinisikan suhu udara saat ini.
2. Batasan tekanan udara, yaitu berada di antara 1006 mBar sampai dengan 1014.6 mBar untuk mendefinisikan tekanan udara saat ini.
3. Batasan kelembaban relatif, yaitu berada di antara 62% sampai dengan 97% untuk mendefinisikan tingkat kelembaban udara relatif saat ini.

Adapun langkah – langkah pembuatannya adalah sebagai berikut.

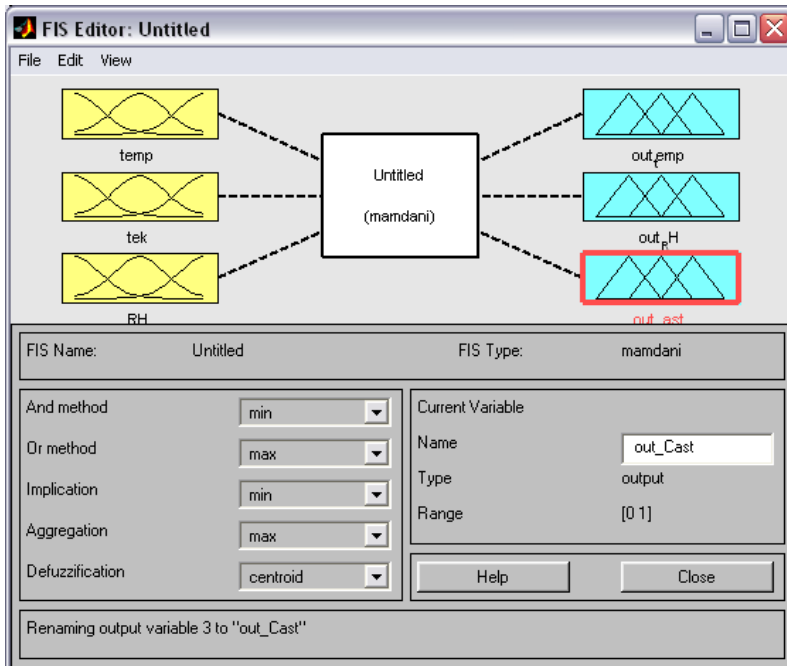
1. Jalankan Matlab, pada *command Window* ketikkan **fuzzy**
2. Maka Anda akan menjumpai tampilan berikut.



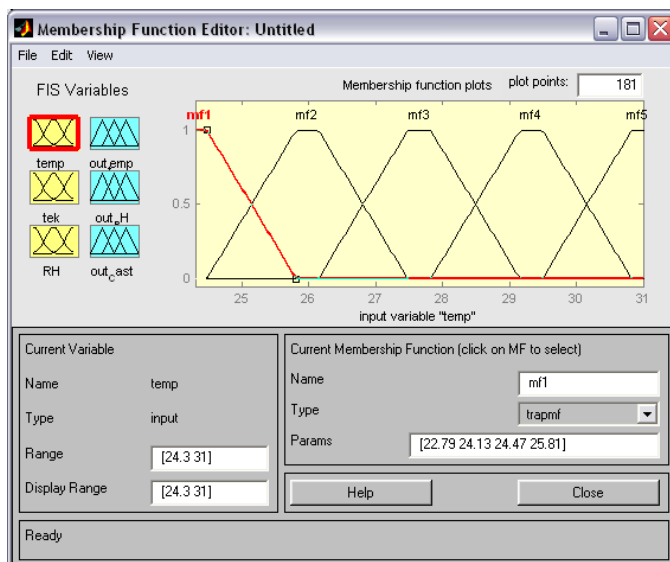
3. Pada keadaan awal disediakan 1 input variabel dan 1 output variabel.
4. Anda bisa menambahkan input variabel dengan cara klik menu **edit > Add Variabel > Input**. Sehingga jumlah input variabel yang kita dapat adalah berjumlah 3 buah.
5. Anda juga perlu menambahkan output variabel sehingga berjumlah 3 buah.
6. Hasil proses 4 dan 5 seperti yang terlihat pada gambar di bawah ini.



7. Ganti Nama **input1** menjadi **temp**, **input2** menjadi **tek**, dan **input 3** menjadi **RH**. Ganti juga **output1** menjadi **out_temp**, **ouput2** menjadi **out_rh**, dan **output3** menjadi **out_cast**. Sehingga akan diperoleh hasil seperti gambar berikut.



8. Lakukan penyesuaian pada **temp** dengan cara **double click** pada **temp**.
9. Kemudian setelah itu klik menu **edit > Remove All MFS**
10. Kemudian setelah itu klik menu **edit > Add MFS**. Pilih **Trap MF** dan **number of MFS = 5**. Maka akan muncul tampilan berikut.



Lakukan pengubahan pada **MF1** ubah menjadi **Very Low** dengan params [22.8 24.1 25.9 28.3]

Lakukan pengubahan pada **MF2** ubah menjadi **Low** dengan params [25.9 28.07 28.1 28.8]

Lakukan pengubahan pada **MF3** ubah menjadi **Medium** dengan params [28.1 28.9 28.9 29.5]

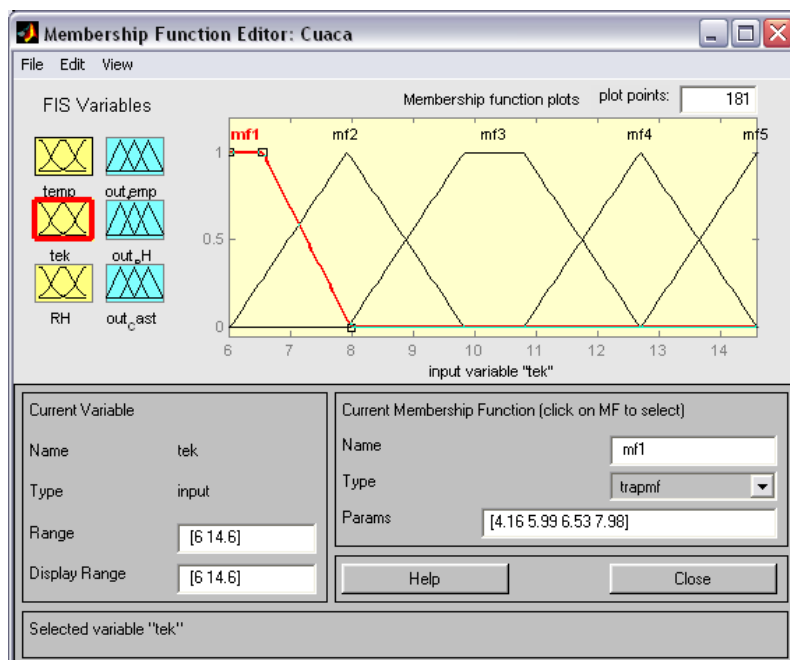
Lakukan pengubahan pada **MF4** ubah menjadi **High** dengan params [28.9 29.1 29.13 29.4]

Lakukan pengubahan pada **MF5** ubah menjadi **Very High** dengan params [29.1 29.3 31 32.4]

11. Lakukan penyesuaian pada **tek** dengan cara **double click** pada **tek**.

12. Kemudian setelah itu klik menu **edit > Remove All MFS**

13. Kemudian setelah itu klik menu **edit > Add MFS**. Pilih **Trap MF** dan **number of MFS = 5**. Maka akan muncul tampilan berikut.



Lakukan pengubahan pada **MF1** ubah menjadi **Very Low** dengan params [4.16 5.99 9.54 10]

Lakukan pengubahan pada **MF2** ubah menjadi **Low** dengan params [9.54 10 10 10.2]

Lakukan pengubahan pada **MF3** ubah menjadi **Medium** dengan params [10 10.5 10.5 10.6]

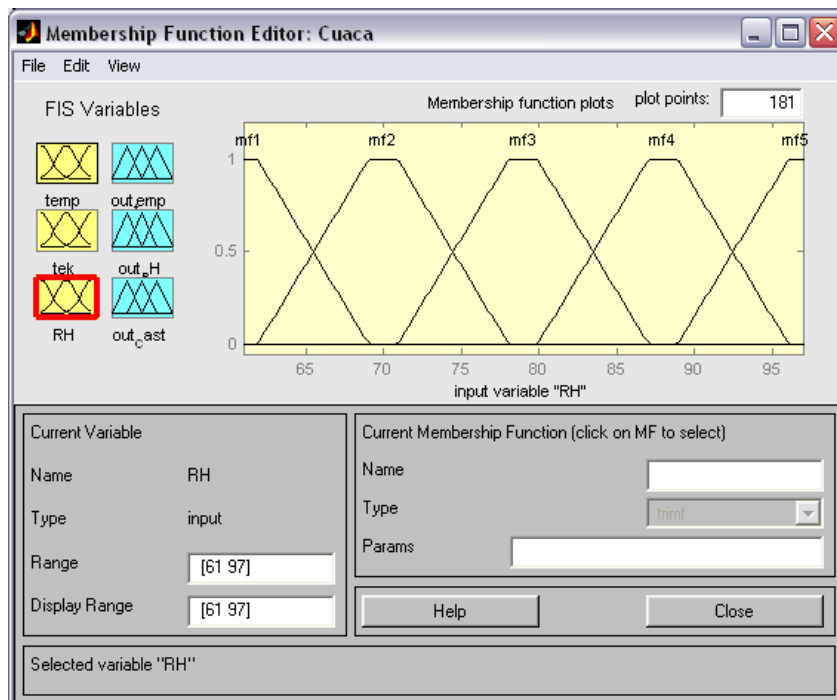
Lakukan pengubahan pada **MF4** ubah menjadi **High** dengan params [10.3 10.7 10.7 10.9]

Lakukan pengubahan pada **MF5** ubah menjadi **Very High** dengan params [10.4 10.9 14.6 16.5]

14. Lakukan penyesuaian pada **RH** dengan cara **double click** pada **RH**.

15. Kemudian setelah itu klik menu **edit > Remove All MFS**

16. Kemudian setelah itu klik menu **edit > Add MFS**. Pilih **Trap MF** dan **number of MFS = 5**. Maka akan muncul tampilan berikut.



Lakukan pengubahan pada **MF1** ubah menjadi **Very Low** dengan params [52.9 60.1 70.1 73.8]

Lakukan pengubahan pada **MF2** ubah menjadi **Low** dengan params [70.1 73.7 73.8 78.2]

Lakukan pengubahan pada **MF3** ubah menjadi **Medium** dengan params [73.8 78.3 78.4 81.9]

Lakukan pengubahan pada **MF4** ubah menjadi **High** dengan params [78.3 81.9 82.1 90.1]

Lakukan pengubahan pada **MF5** ubah menjadi **Very High** dengan params [82 90.1 96.8 104]

17. Lakukan penyesuaian pada **out_temp** dengan cara **double click** pada **out_Temp**.

18. Kemudian setelah itu klik menu **edit > Remove All MFS**

19. Kemudian setelah itu klik menu **edit > Add MFS**. Pilih **Trap MF** dan **number of MFS = 5**.

Lakukan pengubahan pada **MF1** ubah menjadi **Very Low** dengan params [22.8 24.1 25.9 28.3]

Lakukan pengubahan pada **MF2** ubah menjadi **Low** dengan params [25.9 28.07 28.1 28.8]

Lakukan pengubahan pada **MF3** ubah menjadi **Medium** dengan params [28.1 28.9 28.9 29.5]

Lakukan pengubahan pada **MF4** ubah menjadi **High** dengan params [28.9 29.1 29.13 29.4]

Lakukan pengubahan pada **MF5** ubah menjadi **Very High** dengan params [29.1 29.3 31 32.4]

20. Lakukan penyesuaian pada **out_RH** dengan cara **double click** pada **out_RH**.

21. Kemudian setelah itu klik menu **edit > Remove All MFS**

22. Kemudian setelah itu klik menu **edit > Add MFS**. Pilih **Trap MF** dan **number of MFS = 5**.

Lakukan pengubahan pada **MF1** ubah menjadi **Very Low** dengan params [52.9 60.1 70.1 73.8]

Lakukan pengubahan pada **MF2** ubah menjadi **Low** dengan params [70.1 73.7 73.8 78.2]

Lakukan pengubahan pada **MF3** ubah menjadi **Medium** dengan params [73.8 78.3 78.4 81.9]

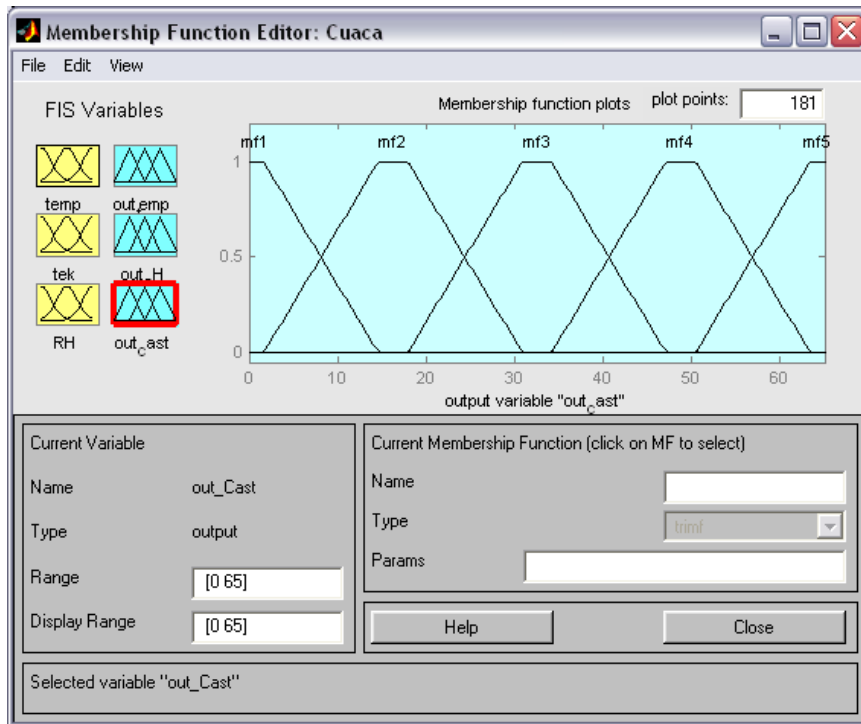
Lakukan pengubahan pada **MF4** ubah menjadi **High** dengan params [78.3 81.9 82.1 90.1]

Lakukan pengubahan pada **MF5** ubah menjadi **Very High** dengan params [82 90.1 96.8 104]

23. Lakukan penyesuaian pada **out_cast** dengan cara **double click** pada **out_cast**.

24. Kemudian setelah itu klik menu **edit > Remove All MFS**

25. Kemudian setelah itu klik menu **edit > Add MFS**. Pilih **Trap MF** dan **number of MFS = 5**. Sehingga akan muncul tampilan berikut.



Lakukan pengubahan pada **MF1** ubah menjadi **Hujan** dengan params [-2.67 30 30.2 36]

Lakukan pengubahan pada **MF2** ubah menjadi **Berawan** dengan params [33.1 36 36.2 40.5]

Lakukan pengubahan pada **MF3** ubah menjadi **CerahBerawan** dengan params [36 41.9 42 43.6]

Lakukan pengubahan pada **MF4** ubah menjadi **HujanBerawan** dengan params [40.5 43.4 43.8 46.51]

Lakukan pengubahan pada **MF5** ubah menjadi **Cerah** dengan params [42 46.5 46.7 65.2]

26. Adapun rule yang perlu dimasukkan adalah sebagai berikut.

Fuzzy Rule Base						
No	IF			Then		
	RH	Tek	Temp	Out_RH	Out_Temp	Out_Cast
1		<i>Very Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
2	<i>Medium</i>		<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
3	<i>High</i>		<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
4	<i>Very low</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
5	<i>Low</i>	<i>Low</i>	<i>Very low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
6	<i>Very high</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
7		<i>Medium</i>	<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
8	<i>Very low</i>	<i>High</i>	<i>Very low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
9	<i>Low</i>	<i>High</i>	<i>Very low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
10	<i>Very high</i>	<i>High</i>	<i>Very low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
11	<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
12	<i>Low</i>	<i>Very high</i>	<i>Very low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
13	<i>Very high</i>	<i>Very high</i>	<i>Very low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
14	<i>Very low</i>		<i>Low</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
15	<i>Low</i>	<i>Very low</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
16	<i>Medium</i>	<i>Very low</i>	<i>Low</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
17	<i>High</i>	<i>Very low</i>	<i>Low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
18	<i>Very high</i>	<i>Very low</i>	<i>Low</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
19	<i>Low</i>	<i>Low</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
20	<i>Medium</i>	<i>Low</i>	<i>Low</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
21	<i>High</i>	<i>Low</i>	<i>Low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
22	<i>Very high</i>	<i>Low</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
23	<i>Low</i>	<i>Medium</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
24	<i>Medium</i>	<i>Medium</i>	<i>Low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
25	<i>High</i>	<i>Medium</i>	<i>Low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
26	<i>Very high</i>	<i>Medium</i>	<i>Low</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
27	<i>Low</i>	<i>High</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>

28	<i>Medium</i>	<i>High</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
29	<i>High</i>	<i>High</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
30	<i>Very high</i>	<i>High</i>	<i>Low</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
31		<i>Very high</i>	<i>Low</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
32	<i>Very low</i>	<i>Very low</i>	<i>Medium</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
33	<i>Low</i>		<i>Medium</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
34	<i>Medium</i>	<i>Very low</i>	<i>Medium</i>	<i>Medium</i>	<i>High</i>	<i>Hujan berawan</i>
35	<i>High</i>	<i>Very low</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
36	<i>Very high</i>	<i>Very low</i>	<i>Medium</i>	<i>High</i>	<i>Low</i>	<i>Berawan</i>
37	<i>Very low</i>	<i>Low</i>	<i>Medium</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
38	<i>Medium</i>	<i>Low</i>	<i>Medium</i>	<i>High</i>	<i>Low</i>	<i>Berawan</i>
39	<i>High</i>	<i>Low</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
40	<i>Very high</i>	<i>Low</i>	<i>Medium</i>	<i>High</i>	<i>Low</i>	<i>Berawan</i>
41	<i>Very low</i>	<i>Medium</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
42	<i>Medium</i>	<i>Medium</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
43	<i>High</i>	<i>Medium</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
44	<i>Very high</i>	<i>Medium</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
45		<i>High</i>	<i>Medium</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
46	<i>Very low</i>	<i>Very high</i>	<i>Medium</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
47	<i>Medium</i>	<i>Very high</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
48	<i>High</i>	<i>Very high</i>	<i>Medium</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
49	<i>Very high</i>	<i>Very high</i>	<i>Medium</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
50	<i>Very low</i>	<i>Very low</i>	<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
51	<i>Low</i>		<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
52	<i>Medium</i>	<i>Very low</i>	<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
53	<i>High</i>	<i>Very low</i>	<i>High</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
54		<i>Low</i>	<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
55	<i>Very high</i>		<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
56	<i>Very low</i>	<i>Medium</i>	<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
57	<i>Medium</i>	<i>Medium</i>	<i>High</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>

58	<i>High</i>	<i>Medium</i>	<i>High</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
59		<i>High</i>	<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
60	<i>Very low</i>	<i>Very high</i>	<i>High</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
61	<i>Medium</i>	<i>Very high</i>	<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
62	<i>High</i>	<i>Very high</i>	<i>High</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
63	<i>Very low</i>	<i>Very low</i>	<i>Very high</i>	<i>Medium</i>	<i>High</i>	<i>Hujan berawan</i>
64	<i>Low</i>		<i>Very high</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
65	<i>Medium</i>	<i>Very low</i>	<i>Very high</i>	<i>Medium</i>	<i>High</i>	<i>Hujan berawan</i>
66	<i>High</i>		<i>Very high</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
67	<i>Very high</i>	<i>Very low</i>	<i>Very high</i>	<i>Medium</i>	<i>High</i>	<i>Hujan berawan</i>
68	<i>Very low</i>	<i>Low</i>	<i>Very high</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
69	<i>Medium</i>	<i>Low</i>	<i>Very high</i>	<i>Medium</i>	<i>High</i>	<i>Hujan berawan</i>
70	<i>Very high</i>	<i>Low</i>	<i>Very high</i>	<i>High</i>	<i>Low</i>	<i>Berawan</i>
71	<i>Very low</i>	<i>Medium</i>	<i>Very high</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
72	<i>Medium</i>	<i>Medium</i>	<i>Very high</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
73	<i>Very high</i>	<i>Medium</i>	<i>Very high</i>	<i>Very high</i>	<i>Very low</i>	<i>Hujan</i>
74	<i>Very low</i>	<i>High</i>	<i>Very high</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
75	<i>Medium</i>	<i>High</i>	<i>Very high</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
76	<i>Very high</i>	<i>High</i>	<i>Very high</i>	<i>Very low</i>	<i>Very high</i>	<i>Cerah</i>
77	<i>Very low</i>	<i>Very high</i>	<i>Very high</i>	<i>Low</i>	<i>Medium</i>	<i>Cerah berawan</i>
78	<i>Medium</i>	<i>Very high</i>	<i>Very high</i>	<i>Medium</i>	<i>High</i>	<i>Hujan berawan</i>
79	<i>Very high</i>	<i>Very high</i>	<i>Very high</i>	<i>Very high</i>	<i>Very low</i>	<i>hujan</i>

Membangun Fuzzy Logic dari Command

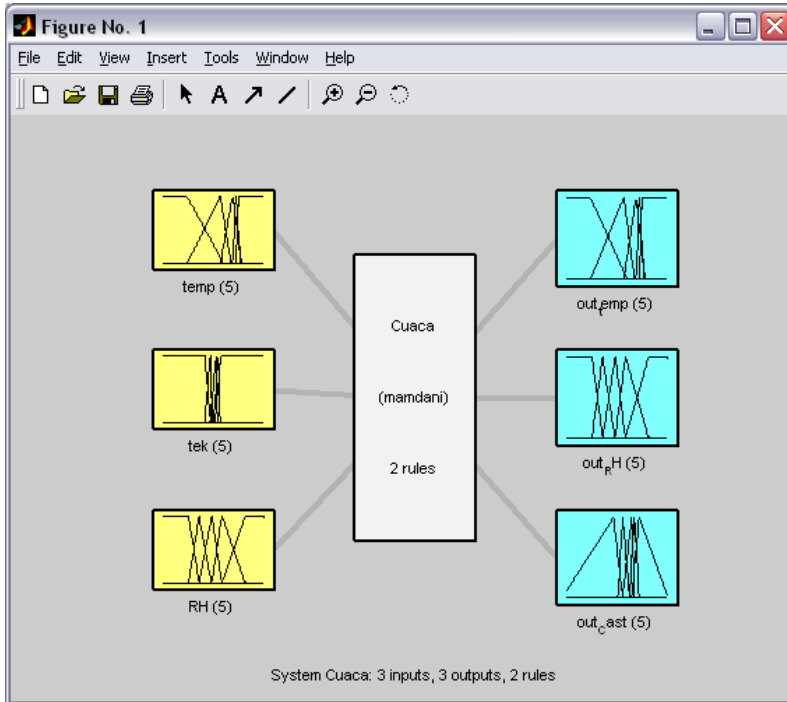
Fungsi Penampil FIS

GUI Fuzzy Logic Toolbox yang dibahas ini memungkinkan Anda untuk melakukan tiga hal sekaligus: merancang, memodifikasi, dan melihat FIS. Ada tiga fungsi Matlab yang dikhususkan untuk melihat FIS. Tiga perintah tersebut adalah **plotfis**, **plotmf**, dan **gensurf**. Sebagai contohnya kita akan melihat **fis** untuk aplikasi peramala cuaca yang telah kita buat sebelumnya.

Caranya adalah ketikkan perintah berikut.

```
a=readfis('cuaca')
Plotfis(a)
```

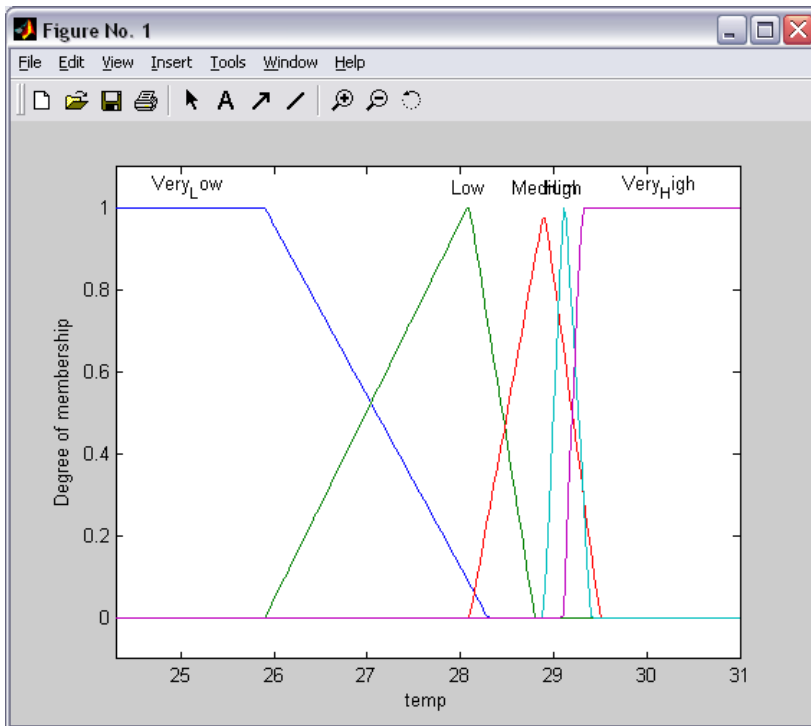
Maka akan muncul tampilan FIS yang telah kita buat sebelumnya, yaitu sebagai berikut.



Perintah **plotmf** digunakan untuk menampilkan variabel **input** dan **output** yang kita inputkan sebelumnya. Sebagai contoh bila kita ingin melihat variabel **input** I yaitu **temp**, maka cukup ketikkan perintah berikut.

```
a=readfis('cuaca')
plotmf(a,'input',1)
```

Maka akan muncul tampilan berikut.



Untuk melihat *output* I yaitu **out_temp** caranya sama yaitu sebagai berikut.

```
a=readfis('cuaca')
plotmf(a,'output',1)
```

Fungsi Membaca Informasi FIS

Untuk membaca informasi FIS dapat digunakan perintah sebagai berikut.

```
a=readfis('cuaca')
```

Maka hasilnya adalah sebagai berikut.

```
name: 'Cuaca'
type: 'mamdani'
andMethod: 'min'
orMethod: 'max'
defuzzMethod: 'centroid'
impMethod: 'min'
aggMethod: 'max'
input: [1x3 struct]
output: [1x3 struct]
rule: [1x2 struct]
```

Misalkan Anda ingin tahu tipe dari FIS yang digunakan, maka ketikkan perintah berikut.

```
a=readfis('cuaca')
a.type
```

Response Matlab :

```
ans =
Mamdani
```

Contoh lainnya:

```
a.input
```

Hasilnya adalah :

```
ans =

mamdani

>> a.input

ans =
```

Membangun FIS melalui Command Lines

Tanpa GUI pun, Fuzzy Logic Toolbox memungkinkan Anda membangun FIS dengan mudah, yaitu dengan menggunakan Command Lines atau baris – baris perintah yang tersimpan dalam sebuah M-File

Untuk membangun sebuah FIS baru dengan Command Lines, mulailah dengan perintah berikut.

```
a = newfis('pcuaca')
```

Maka hasilnya adalah sebagai berikut.

```
a =
    name: 'pcuaca'
    type: 'mamdani'
    andMethod: 'min'
    orMethod: 'max'
    defuzzMethod: 'centroid'
    impMethod: 'min'
    aggMethod: 'max'
    input: []
    output: []
    rule: []
```

Perintah **newfis** harus diberi argumen (masukan) nama FIS yang diinginkan, yang dalam hal ini **pcuaca**. Keluaran **newfis** adalah sebuah struktur FIS. Karena jumlah input, output, dan rule belum ditentukan, maka Anda lihat tanda kurung siku kosong pada komponen input, output, dan rule, yang menunjukkan bahwa ketiganya adalah array kosong. Secara default, **newfis** memilih tipe Mamdani, operasi AND dilakukan dengan fungsi **min**, operasi OR dengan fungsi **max**, defuzzifikasi dengan fungsi **centroid**, implikasi dengan fungsi **min**, dan agregasi dengan fungsi **max**.

Anda bisa membentuk FIS sesuai dengan keinginan kita dengan menggunakan perintah :

```
a=newfis(fisName,fisType,andMethod,orMethod,impMethod, ...
        aggMethod,defuzzMethod)
```

Sebagai contoh, yang kita masukkan adalah :

```
a=newfis('pcuaca','Mamdani','min','max','min', ...
        'max','mom')
```

Maka nantinya proses defuzzifikasi akan dilakukan dengan fungsi **mom**.

Sekarang akan kita desain ulang FIS sebagaimana sebelumnya, namun sekarang semuanya dilakukan dengan Command Lines. Ada dua cara yang bisa dipakai. Cara pertama adalah dengan menggunakan sintaks akses langsung komponen variabel struktur Matlab. Cara kedua adalah dengan menggunakan sintaks bawaan Fuzzy Logic Toolbox.

Sintaks Variabel Struktur Matlab

Untuk mendefinisikan variabel – variabel input

```
a.input(1).name='temp'
a.input(2).name='tek'
a.input(3).name='RH'
```

Yang berarti variabel input pertama FIS dilabeli dengan **temp**, variabel input kedua dengan **tek**, dan variabel input ketiga dengan nama **RH**.

Untuk mendefinisikan variabel – variabel output.

```
a.output(1).name='out_temp'
a.output(2).name='out_RH'
a.output(3).name='out_cast'
```

Untuk mendefinisikan rentang harga dari ketiga variabel input dan output tersebut, ketikkan :

```
a.input(1).range=[24.31 30]
a.input(2).range=[6 14.6]
a.input(3).range=[61 97]

a.output(1).range=[24.31 30]
a.output(2).range=[61 97]
a.output(3).range=[0 65]
```

Variabel **temp** dibagi menjadi 5 fungsi keanggotaan yang dilabeli dengan **very low**, **low**, **medium**, **high**, dan **very high**. Masing – masing fungsi keanggotaan diinginkan bertipe trapezoid(**trapmf**) dengan parameter masing – masing [22.8 24.1 25.9 28.3], [25.9 28.07 28.1 28.8], [28.1 28.9 28.9 29.5], [28.9 29.1 29.13 29.4], dan [29.1 29.3 31 32.4]. Untuk memasukkan ke 5 label dalam FIS, dilakukan :

```
a.input(1).mf(1).name='Very_Low'
a.input(1).mf(1).type='trapmf'
a.input(1).mf(1).params=[22.8 24.1 25.9 28.3]

a.input(1).mf(2).name='Low'
```



```
a.input(1).mf(2).type='trapmf'
a.input(1).mf(2).params=[25.9 28.07 28.1 28.8]
```

```
a.input(1).mf(3).name='Medium'
a.input(1).mf(3).type='trapmf'
a.input(1).mf(3).params=[28.1 28.9 28.9 29.5]
```

```
a.input(1).mf(4).name='High'
a.input(1).mf(4).type='trapmf'
a.input(1).mf(4).params=[28.9 29.1 29.13 29.4]
```

```
a.input(1).mf(5).name='Very_High'
a.input(1).mf(5).type='trapmf'
a.input(1).mf(5).params=[29.1 29.3 31 32.4]
```

```
a.input(2).mf(1).name='Very_Low'
a.input(2).mf(1).type='trapmf'
a.input(2).mf(1).params=[4.16 5.99 9.54 10]
```

```
a.input(2).mf(2).name='Low'
a.input(2).mf(2).type='trapmf'
a.input(2).mf(2).params=[9.54 10 10 10.2]
```

```
a.input(2).mf(3).name='Medium'
a.input(2).mf(3).type='trapmf'
a.input(2).mf(3).params=[10 10.5 10.5 10.6]
```

```
a.input(2).mf(4).name='High'
a.input(2).mf(4).type='trapmf'
a.input(2).mf(4).params=[10.3 10.7 10.7 10.9]
```

```
a.input(2).mf(5).name='Very_High'
a.input(2).mf(5).type='trapmf'
a.input(2).mf(5).params=[10.4 10.9 14.6 16.5]
```

```
a.input(3).mf(1).name='Very_Low'
a.input(3).mf(1).type='trapmf'
a.input(3).mf(1).params=[52.9 60.1 70.1 73.8]
```

```

a.input(3).mf(2).name='Low'
a.input(3).mf(2).type='trapmf'
a.input(3).mf(2).params=[70.1 73.7 73.8 78.2]

a.input(3).mf(3).name='Medium'
a.input(3).mf(3).type='trapmf'
a.input(3).mf(3).params=[73.8 78.3 78.4 81.9]

a.input(3).mf(4).name='High'
a.input(3).mf(4).type='trapmf'
a.input(3).mf(4).params=[78.3 81.9 82.1 90.1]

a.input(3).mf(5).name='Very_High'
a.input(3).mf(5).type='trapmf'
a.input(3).mf(5).params=[82 90.1 96.8 104]

a.output(1).mf(1).name='Very_Low'
a.output(1).mf(1).type='trapmf'
a.output(1).mf(1).params=[22.8 24.1 25.9 28.3]

a.output(1).mf(2).name='Low'
a.output(1).mf(2).type='trapmf'
a.output(1).mf(2).params=[25.9 28.07 28.1 28.8]

a.output(1).mf(3).name='Medium'
a.output(1).mf(3).type='trapmf'
a.output(1).mf(3).params=[28.1 28.9 28.9 29.5]

a.output(1).mf(4).name='High'
a.output(1).mf(4).type='trapmf'
a.output(1).mf(4).params=[28.9 29.1 29.13 29.4]

a.output(1).mf(5).name='Very_High'
a.output(1).mf(5).type='trapmf'
a.output(1).mf(5).params=[29.1 29.3 31 32.4]

a.output(2).mf(1).name='Very_Low'

```

```
a.output(2).mf(1).type='trapmf'
a.output(2).mf(1).params=[52.9 60.1 70.1 73.8]
```

```
a.output(2).mf(2).name='Low'
a.output(2).mf(2).type='trapmf'
a.output(2).mf(2).params=[70.1 73.7 73.8 78.2]
```

```
a.output(2).mf(3).name='Medium'
a.output(2).mf(3).type='trapmf'
a.output(2).mf(3).params=[73.8 78.3 78.4 81.9]
```

```
a.output(2).mf(4).name='High'
a.output(2).mf(4).type='trapmf'
a.output(2).mf(4).params=[78.3 81.9 82.1 90.1]
```

```
a.output(2).mf(5).name='Very_High'
a.output(2).mf(5).type='trapmf'
a.output(2).mf(5).params=[82 90.1 96.8 104]
```

```
a.output(3).mf(1).name='Hujan'
a.output(3).mf(1).type='trapmf'
a.output(3).mf(1).params=[-2.67 30 30.2 36]
```

```
a.output(3).mf(2).name='Berawan'
a.output(3).mf(2).type='trapmf'
a.output(3).mf(2).params=[33.1 36 36.2 40.5]
```

```
a.output(3).mf(3).name='Cerah_Berawan'
a.output(3).mf(3).type='trapmf'
a.output(3).mf(3).params=[36 41.9 42 43.6]
```

```
a.output(3).mf(4).name='Hujan_Berawan'
a.output(3).mf(4).type='trapmf'
a.output(3).mf(4).params=[40.5 43.4 43.8 46.5]
```

```
a.output(3).mf(5).name='Cerah'
a.output(3).mf(5).type='trapmf'
a.output(3).mf(5).params=[42 46.5 46.7 65.2]
```

Sampai tahap ini kita telah mempunyai FIS dengan masukan dan keluaran berikut fungsi – fungsi keanggotaannya, namun masih belum diberitahu If – Then Rule yang harus diikuti. Tahap berikutnya adalah mendefinisikan If – Then Rules. Sebelum membahas pendefinisian If – Then Rule dengan Command Lines, perlu disampaikan dulu sistem pengodean dari tiap variabel dan fungsi keanggotaan.

Tiap variabel input atau output mempunyai sebuah bilangan indeks, demikian juga fungsi keanggotaannya. Dalam rancang bangun FIS dalam Fuzzy Logic Toolbox, sebuah If – Then Rule dibangun dengan pernyataan seperti ini :

Operator And dan Or masing – masing dikodekan dengan 1 dan 2.

Sekarang ambil contoh salah satu rule di dalam FIS yang sudah dibahas sebelumnya :

If Temp = very low and tek = very low then Out_temp = very low and out_rh = very high and out_cast = hujan

Rule ini dimasukkan dalam FIS dengan command Line Berikut :

```
a.rule(1).antecedent=[1 1 0];
a.rule(1).connection=1;
a.rule(1).consequent=[1 5 1];
a.rule(1).weight=1;
```

Angka 1 dalam rule(1) menunjukkan indeks rule (rule urutan beberapa). Kolom pertama dalam antecedent = [1 1 0] menunjukkan bahwa temp= very low, dimana very low berindeks 1. Kolom kedua diasosiasikan dengan input kedua yaitu tek, yaitu tek = very low, di mana very low berindeks 1. Kolom ketiga diasosiasikan dengan input ketiga yaitu RH, yaitu 0 menunjukkan bahwa RH tidak digunakan. Operator yang dipakai dalam rule adalah And (berindeks 1). Oleh karenanya kita set connection = 1. Output dari rule adalah out_temp = very low. Karena fungsi keanggotaan very low berindeks 1. Out RH=very high, juga berindeks 5, dan out_cast = hujan berindeks 1. Oleh karena itu kita set consequent = [1]. Bobot rule adalah 1, maka kita set weight = 1.

Rule kedua berbunyi :

If temp = very low and RH = medium then out_temp = very low and out_rh = very high and out_cast = hujan

Dimasukkan dalam FIS dengan command lines berikut.

```
a.rule(1).antecedent=[1 0 3];
a.rule(1).connection=1;
a.rule(1).consequent=[1 5 1];
a.rule(1).weight=1;
```

Cara yang sama dapat dilakukan untuk rule – rule lainnya.

Sintaks Fuzzy Logic Toolbox

Pembangunan FIS menggunakan sintaks bawaan Fuzzy Logic Toolbox jauh lebih ringkas. Di sini kita akan menggunakan contoh kasus yang sama seperti sebelumnya, yaitu membangun FIS mulai dari awal lagi. Ketikkan baris – baris perintah berikut:

```
a=newfis('pcuaca')
a=addvar(a,'input',1,'Temp',[24.31 30])
a=addvar(a,'input',2,'Tek',[6 14.6])
a=addvar(a,'input',3,'RH',[61 97])

a=addvar(a,'output',1,'out_Temp',[24.31 30])
a=addvar(a,'output',2,'out_RH',[61 97])
a=addvar(a,'output',3,'out_cast',[0 65])
```

di mana **addvar** adalah sebuah fungsi yang melakukan penambahan sebuah variabel baru, memberi nomor indeks, memberi label, serta menentukan rentang harga variabel.

Dengan baris – baris perintah di atas akan dihasilkan sebuah FIS dengan variabel input pertama adalah **temp**, variabel input kedua adalah **tek**, dan variabel input ketiga adalah **RH**. Juga akan dihasilkan variabel output pertama adalah **out_temp**, variabel output kedua adalah **out_rh**, dan variabel output ketiga adalah **out_cast**.

Untuk mendefinisikan fungsi – fungsi keanggotaan variabel input **temp** berikut dengan tipe dan parameternya, ketikkan baris – baris perintah berikut.

```
a=addmf(a,'input',1,'very_low','trapmf',[22.8 24.1 25.9 28.3]);
a=addmf(a,'input',1,'low','trapmf',[25.9 28.07 28.1 28.8]);
a=addmf(a,'input',1,'medium','trapmf',[28.1 28.9 28.9 29.5]);
a=addmf(a,'input',1,'high','trapmf',[28.9 29.1 29.13 29.4]);
a=addmf(a,'input',1,'very_high','trapmf',[29.1 29.3 31 32.4]);
```

di mana **addmf** adalah sebuah fungsi yang melakukan penambahan sebuah fungsi keanggotaan untuk sebuah variabel FIS yang sudah didefinisikan sebelumnya dengan fungsi **addvar**. Meskipun tidak menyebut nama/label variabelnya, sintaks **addmf(a,'input',1,...)** berarti menambahkan sebuah fungsi keanggotaan untuk variabel berindeks 1, yaitu **temp**.

Kita lihat bahwa sekarang pendefinisian variabel input **temp**, fungsi – fungsi keanggotaan, tipe – tipe fungsi keanggotaan, dan parameternya bisa dilakukan dengan lebih ringkas dan cepat.

Dengan cara yang sama Anda bisa menambahkan untuk variabel input yang lain maupun untuk variabel **output**.

Sampai tahap ini input dan output FIS sudah didefinisikan dengan lengkap, tinggal mendefinisikan If – Then rules. Fuzzy Logic Toolbox menyediakan sebuah fungsi **addrule** (<variabel struktur FIS>, <rule list>) yang akan menambahkan keseluruhan rule yang diinginkan ke dalam FIS dengan sekali perintah.

Untuk memudahkan, kita tulis ulang rule yang perlu kita masukkan :

Rule #1 : If Temp = very low and tek = very low then Out_temp = very low and out_rh = very high and out_cast = hujan

Rule #2 : If temp = very low and RH = medium then out_temp = very low and out_rh = very high and out_cast = hujan

Untuk memasukkan semua rule di atas dengan fungsi **addrule**, ketikkan berikut :

```
ruleList=[ ...
1  1  0  1  5  1  1  1
1  0  3  1  5  1  1  1];
a=addrule(a,ruleList);
```

Dengan disusun seperti di atas, tiap baris **ruleList** merupakan kode untuk tiap rule yang terdiri dari 8 kolom. Baris pertama kolom pertama berisi kode fungsi keanggotaan dari variabel input **temp**. Dalam hal ini diisi 1 (indeks untuk fungsi keanggotaan very_low).

Baris pertama kolom kedua berisi kode fungsi keanggotaan dari variabel input **tek**. Dalam hal ini diisi 1 (bilangan indeks untuk fungsi keanggotaan very low). Baris pertama kolom ketiga berisi kode fungsi keanggotaan dari variabel input **RH**. Dalam hal ini diisi 0 karena variabel **RH** tidak diikuti.

Baris pertama kolom keempat berisi kode fungsi keanggotaan dari variabel output **out_temp**. Dalam hal ini diisi 1 (bilangan indeks untuk fungsi keanggotaan very_low). Baris pertama kolom kelima berisi kode fungsi keanggotaan dari variabel output **out_RH**. Dalam hal ini diisi 5 (bilangan indeks untuk fungsi keanggotaan very_high).

Baris pertama kolom keenam berisi kode fungsi keanggotaan dari variabel output **out_cast**. Dalam hal ini diisi 5 (bilangan indeks untuk fungsi keanggotaan hujan). Baris pertama kolom ketujuh berisi bobot dari rule yang dalam hal ini adalah 1. Baris pertama kolom kedelapan berisi kode jenis operasi di dalam antecedent dari rule pertama. Dalam hal ini diisi dengan 1, yang berarti jenis operasi dalam antecedent rule pertama adalah And.

Sistem pengkodean untuk rule – rule lainnya adalah sama seperti cara di atas. Sampai disini pembangunan FIS dengan Fuzzy Logic Toolbox Command Lines telah selesai. Apabila ingin memodifikasi atau menghapus variabel FIS atau fungsi keanggotaannya, Fuzzy Logic Toolbox telah menyediakan perintah – perintahnya. Untuk menghapus sebuah variabel FIS, gunakan perintah:

```
a = rmvar (a,'varType', varindex)
```

Di mana **varType** adalah tipe variabel: input atau output, **varindex** adalah bilangan indeks variabel.

Dengan sintaks seperti di atas, setelah sebuah variabel FIS dihapus, FIS terbaru diisikan lagi ke variabel **a**.

```
a=rmmf(a,varType,varIndex,'mf',mfIndex,warningDlgEnabled)
```

Di mana **warningDlgEnabled** (diset true atau false) adalah opsi untuk memunculkan kotak dialog konfirmasi

Kode program selengkapnya adalah sebagai berikut.

```
a = newfis('coba.fis')
a.input(1).name='temp'
a.input(2).name='tek'
a.input(3).name='RH'

a.output(1).name='out_temp'
a.output(2).name='out_RH'
a.output(3).name='out_cast'

a.input(1).range=[24.31 31]
a.input(2).range=[6 14.6]
a.input(3).range=[61 97]

a.output(1).range=[24.31 30]
a.output(2).range=[61 97]
a.output(3).range=[0 65]

a.input(1).mf(1).name='Very_Low'
a.input(1).mf(1).type='trapmf'
a.input(1).mf(1).params=[22.8 24.1 25.9 28.3]

a.input(1).mf(2).name='Low'
a.input(1).mf(2).type='trapmf'
```



```
a.input(1).mf(2).params=[25.9 28.07 28.1 28.8]
```

```
a.input(1).mf(3).name='Medium'
```

```
a.input(1).mf(3).type='trapmf'
```

```
a.input(1).mf(3).params=[28.1 28.9 28.9 29.5]
```

```
a.input(1).mf(4).name='High'
```

```
a.input(1).mf(4).type='trapmf'
```

```
a.input(1).mf(4).params=[28.9 29.1 29.13 29.4]
```

```
a.input(1).mf(5).name='Very_High'
```

```
a.input(1).mf(5).type='trapmf'
```

```
a.input(1).mf(5).params=[29.1 29.3 31 32.4]
```

```
a.input(2).mf(1).name='Very_Low'
```

```
a.input(2).mf(1).type='trapmf'
```

```
a.input(2).mf(1).params=[4.16 5.99 9.54 10]
```

```
a.input(2).mf(2).name='Low'
```

```
a.input(2).mf(2).type='trapmf'
```

```
a.input(2).mf(2).params=[9.54 10 10 10.2]
```

```
a.input(2).mf(3).name='Medium'
```

```
a.input(2).mf(3).type='trapmf'
```

```
a.input(2).mf(3).params=[10 10.5 10.5 10.6]
```

```
a.input(2).mf(4).name='High'
```

```
a.input(2).mf(4).type='trapmf'
```

```
a.input(2).mf(4).params=[10.3 10.7 10.7 10.9]
```

```
a.input(2).mf(5).name='Very_High'
```

```
a.input(2).mf(5).type='trapmf'
```

```
a.input(2).mf(5).params=[10.4 10.9 14.6 16.5]
```

```
a.input(3).mf(1).name='Very_Low'
```

```
a.input(3).mf(1).type='trapmf'
```

```
a.input(3).mf(1).params=[52.9 60.1 70.1 73.8]
```

```

a.input(3).mf(2).name='Low'
a.input(3).mf(2).type='trapmf'
a.input(3).mf(2).params=[70.1 73.7 73.8 78.2]

a.input(3).mf(3).name='Medium'
a.input(3).mf(3).type='trapmf'
a.input(3).mf(3).params=[73.8 78.3 78.4 81.9]

a.input(3).mf(4).name='High'
a.input(3).mf(4).type='trapmf'
a.input(3).mf(4).params=[78.3 81.9 82.1 90.1]

a.input(3).mf(5).name='Very_High'
a.input(3).mf(5).type='trapmf'
a.input(3).mf(5).params=[82 90.1 96.8 104]

a.output(1).mf(1).name='Very_Low'
a.output(1).mf(1).type='trapmf'
a.output(1).mf(1).params=[22.8 24.1 25.9 28.3]

a.output(1).mf(2).name='Low'
a.output(1).mf(2).type='trapmf'
a.output(1).mf(2).params=[25.9 28.07 28.1 28.8]

a.output(1).mf(3).name='Medium'
a.output(1).mf(3).type='trapmf'
a.output(1).mf(3).params=[28.1 28.9 28.9 29.5]

a.output(1).mf(4).name='High'
a.output(1).mf(4).type='trapmf'
a.output(1).mf(4).params=[28.9 29.1 29.13 29.4]

a.output(1).mf(5).name='Very_High'
a.output(1).mf(5).type='trapmf'
a.output(1).mf(5).params=[29.1 29.3 31 32.4]

a.output(2).mf(1).name='Very_Low'
a.output(2).mf(1).type='trapmf'

```

```
a.output(2).mf(1).params=[52.9 60.1 70.1 73.8]
```

```
a.output(2).mf(2).name='Low'
```

```
a.output(2).mf(2).type='trapmf'
```

```
a.output(2).mf(2).params=[70.1 73.7 73.8 78.2]
```

```
a.output(2).mf(3).name='Medium'
```

```
a.output(2).mf(3).type='trapmf'
```

```
a.output(2).mf(3).params=[73.8 78.3 78.4 81.9]
```

```
a.output(2).mf(4).name='High'
```

```
a.output(2).mf(4).type='trapmf'
```

```
a.output(2).mf(4).params=[78.3 81.9 82.1 90.1]
```

```
a.output(2).mf(5).name='Very_High'
```

```
a.output(2).mf(5).type='trapmf'
```

```
a.output(2).mf(5).params=[82 90.1 96.8 104]
```

```
a.output(3).mf(1).name='Hujan'
```

```
a.output(3).mf(1).type='trapmf'
```

```
a.output(3).mf(1).params=[-2.67 30 30.2 36]
```

```
a.output(3).mf(2).name='Berawan'
```

```
a.output(3).mf(2).type='trapmf'
```

```
a.output(3).mf(2).params=[33.1 36 36.2 40.5]
```

```
a.output(3).mf(3).name='Cerah_Berawan'
```

```
a.output(3).mf(3).type='trapmf'
```

```
a.output(3).mf(3).params=[36 41.9 42 43.6]
```

```
a.output(3).mf(4).name='Hujan_Berawan'
```

```
a.output(3).mf(4).type='trapmf'
```

```
a.output(3).mf(4).params=[40.5 43.4 43.8 46.5]
```

```
a.output(3).mf(5).name='Cerah'
```

```
a.output(3).mf(5).type='trapmf'
```

```
a.output(3).mf(5).params=[42 46.5 46.7 65.2]
```

```

a.rule(1).antecedent=[1 1 0];
a.rule(1).connection=1;
a.rule(1).consequent=[1 5 1];
a.rule(1).weight=1;

a.rule(2).antecedent=[1 0 3];
a.rule(2).connection=1;
a.rule(2).consequent=[1 5 1];
a.rule(2).weight=1;
writefis(a, 'my_file')

```

Simpan program di atas dengan nama **‘mcuaca’**

Kemudian berikut program untuk menjalankannya.

```

temp = input ('Data Temp = ');
tek= input ('Data Tek = ');
rh=input('Data RH =');
a=readfis('my_file')
a=evalfis([temp,tek,rh],a)

```

Simpan program di atas dengan nama **‘inputdata’**

Fuzzy Logic dengan Visual Basic

Misalkan akan dirancang aplikasi Fuzzy Logic untuk menentukan kecepatan kestabilan suhu di malam hari.

Dengan paramater input terdiri dari :

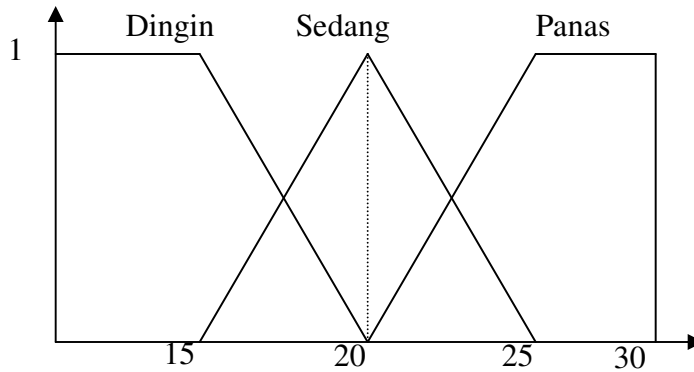
1. Suhu
2. Kecepatan Angin
3. Kelembapan Udara

Sedangkan output adalah : Kecepatan kestabilan suhu

Adapun *membership function* dari masing – masing parameter adalah sebagai berikut.

1. Suhu

Adapun *membership function* dari suhu adalah terdiri dari : **Dingin**, **Sedang**, dan **Panas**. Gambar dari *membership function* dapat dilihat pada gambar di bawah ini.



Nilai Keanggotaan dari masing – masing *membership function* dapat dihitung dengan persamaan sebagai berikut.

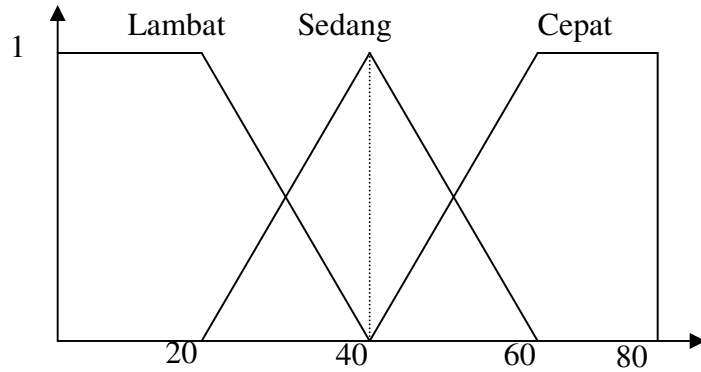
$$\mu_{\text{dingin}} [x : 0, 15, 20] \begin{cases} 0 & \text{untuk } x < 0 \\ 1 & \text{untuk } 0 \leq x \leq 15 \\ (20-x) / (20-15) & \text{untuk } 15 < x < 20 \\ 0 & \text{untuk } x = 20 \end{cases}$$

$$\mu_{\text{Sedang}} [x : 15, 20, 25] \begin{cases} 0 & \text{untuk } x < 15 \\ (x-15) / (20-15) & \text{untuk } 15 \leq x \leq 20 \\ (25-x) / (25-20) & \text{untuk } 20 < x < 25 \\ 0 & \text{untuk } x = 25 \end{cases}$$

$$\mu_{\text{Panas}} [x : 20, 25, 30] \begin{cases} 0 & \text{untuk } x < 20 \\ (x-20) / (25-20) & \text{untuk } 20 \leq x \leq 25 \\ 1 & \text{untuk } 25 < x < 30 \\ 1 & \text{untuk } x = 30 \end{cases}$$

2. Kecepatan Angin

Adapun *membership function* dari Kecepatan Angin adalah terdiri dari : **Lambat**, **Sedang**, dan **Cepat**. Gambar dari *membership function* dapat dilihat pada gambar di bawah ini.



Nilai Keanggotaan dari masing – masing *membership function* dapat dihitung dengan persamaan sebagai berikut.

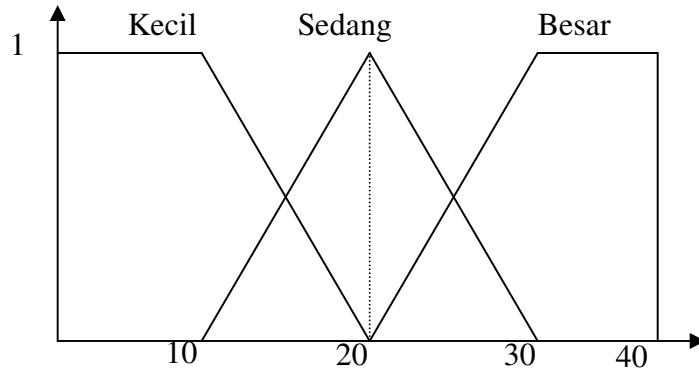
$$\mu_{\text{Lambat}} [x : 0, 20, 40] \begin{cases} 0 & \text{untuk } x < 0 \\ 1 & \text{untuk } 0 \leq x \leq 20 \\ (40-x) / (40-20) & \text{untuk } 20 < x < 40 \\ 0 & \text{untuk } x = 40 \end{cases}$$

$$\mu_{\text{Sedang}} [x : 20, 40, 60] \begin{cases} 0 & \text{untuk } x < 20 \\ (x-20) / (40-20) & \text{untuk } 20 \leq x \leq 40 \\ (60-x) / (60-40) & \text{untuk } 40 < x < 60 \\ 0 & \text{untuk } x = 60 \end{cases}$$

$$\mu_{\text{Cepat}} [x : 40, 60, 80] \begin{cases} 0 & \text{untuk } x < 40 \\ (x-40) / (60-40) & \text{untuk } 40 \leq x \leq 60 \\ 1 & \text{untuk } 60 < x < 80 \\ 1 & \text{untuk } x = 80 \end{cases}$$

3. Kelembapan Udara

Adapun *membership function* dari Kecepatan Angin adalah terdiri dari : **Kecil**, **Sedang**, dan **Besar**. Gambar dari *membership function* dapat dilihat pada gambar di bawah ini.



Nilai Keanggotaan dari masing – masing *membership function* dapat dihitung dengan persamaan sebagai berikut.

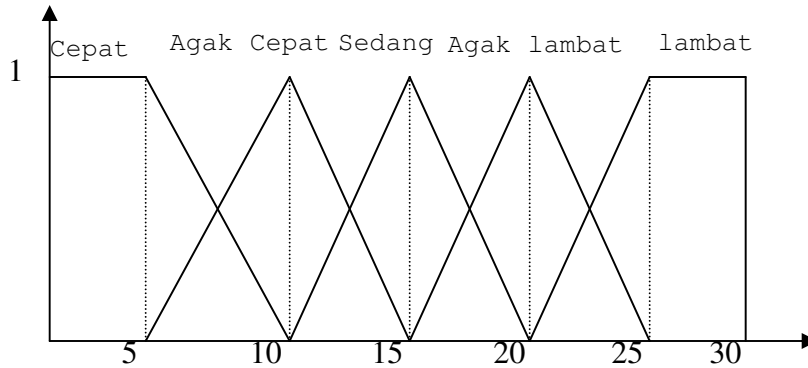
$$\mu_{\text{Kecil}} [x : 0, 10, 20] \begin{cases} 0 & \text{untuk } x < 0 \\ 1 & \text{untuk } 0 \leq x \leq 10 \\ (20-x) / (20-10) & \text{untuk } 10 < x < 20 \\ 0 & \text{untuk } x = 20 \end{cases}$$

$$\mu_{\text{Sedang}} [x : 10, 20, 30] \begin{cases} 0 & \text{untuk } x < 10 \\ (x-10) / (20-10) & \text{untuk } 10 \leq x \leq 20 \\ (30-x) / (30-20) & \text{untuk } 20 < x < 30 \\ 0 & \text{untuk } x = 30 \end{cases}$$

$$\mu_{\text{Besar}} [x : 20, 30, 40] \begin{cases} 0 & \text{untuk } x < 20 \\ (x-20) / (30-20) & \text{untuk } 20 \leq x \leq 30 \\ 1 & \text{untuk } 30 < x < 40 \\ 1 & \text{untuk } x = 40 \end{cases}$$

4. Kestabilan Suhu

Adapun *membership function* dari Kestabilan suhu adalah terdiri dari : **Cepat**, **Agak cepat**, **Sedang**, **Agak lambat**, dan **Lambat**. Gambar dari *membership function* dapat dilihat pada gambar di bawah ini.



Nilai Keanggotaan dari masing – masing *membership function* dapat dihitung dengan persamaan sebagai berikut.

$$\mu_{\text{Cepat}} [x : 0, 5, 10] \begin{cases} 0 & \text{untuk } x < 0 \\ 1 & \text{untuk } 0 \leq x \leq 5 \\ (10-x) / (10-5) & \text{untuk } 5 < x \leq 10 \\ 0 & \text{untuk } x = 10 \end{cases}$$

$$\mu_{\text{Agak cepat}} [x : 5, 10, 15] \begin{cases} 0 & \text{untuk } x < 5 \\ (x-5) / (10-5) & \text{untuk } 5 \leq x \leq 10 \\ (15-x) / (15-10) & \text{untuk } 10 < x < 15 \\ 0 & \text{untuk } x = 15 \end{cases}$$

$$\mu_{\text{Sedang}} [x : 10, 15, 20] \begin{cases} 0 & \text{untuk } x < 10 \\ (x-10) / (15-10) & \text{untuk } 10 \leq x \leq 15 \\ (20-x) / (20-15) & \text{untuk } 15 < x < 20 \\ 0 & \text{untuk } x = 20 \end{cases}$$

$$\begin{array}{l}
 \mu_{\text{Agak lambat}} \\
 [x : 10, 15, 20]
 \end{array}
 \left\{
 \begin{array}{ll}
 0 & \text{untuk } x < 15 \\
 (x-15) / (20-15) & \text{untuk } 15 \leq x \leq 20 \\
 (25-x) / (25-20) & \text{untuk } 20 < x < 25 \\
 0 & \text{untuk } x = 25
 \end{array}
 \right.$$

$$\begin{array}{l}
 \mu_{\text{Lambat}} \\
 [x : 20, 25, 30]
 \end{array}
 \left\{
 \begin{array}{ll}
 0 & \text{untuk } x < 20 \\
 (x-20) / (30-20) & \text{untuk } 20 \leq x \leq 25 \\
 1 & \text{untuk } 25 < x < 30 \\
 1 & \text{untuk } x = 30
 \end{array}
 \right.$$

Adapun kumpulan aturan untuk proses inferensi adalah sebagai berikut.

1. Jika derajat suhu adalah panas dan kecepatan angin adalah lambat dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah cepat.
2. Jika derajat suhu adalah panas dan kecepatan angin adalah lambat dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah cepat.
3. Jika derajat suhu adalah panas dan kecepatan angin adalah lambat dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah cepat.
4. Jika derajat suhu adalah panas dan kecepatan angin adalah sedang dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah cepat.
5. Jika derajat suhu adalah panas dan kecepatan angin adalah sedang dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah cepat.
6. Jika derajat suhu adalah panas dan kecepatan angin adalah sedang dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah cepat.
7. Jika derajat suhu adalah panas dan kecepatan angin adalah cepat dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah agak cepat.

8. Jika derajat suhu adalah panas dan kecepatan angin adalah cepat dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah agak cepat.
9. Jika derajat suhu adalah panas dan kecepatan angin adalah cepat dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah agak cepat.
10. Jika derajat suhu adalah sedang dan kecepatan angin adalah lambat dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah agak cepat.
11. Jika derajat suhu adalah sedang dan kecepatan angin adalah lambat dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah agak cepat.
12. Jika derajat suhu adalah sedang dan kecepatan angin adalah lambat dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah agak cepat.
13. Jika derajat suhu adalah sedang dan kecepatan angin adalah sedang dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah sedang.
14. Jika derajat suhu adalah sedang dan kecepatan angin adalah sedang dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah sedang.
15. Jika derajat suhu adalah sedang dan kecepatan angin adalah sedang dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah sedang.
16. Jika derajat suhu adalah sedang dan kecepatan angin adalah cepat dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah agak lambat.
17. Jika derajat suhu adalah sedang dan kecepatan angin adalah cepat dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah agak lambat.
18. Jika derajat suhu adalah sedang dan kecepatan angin adalah cepat dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah agak lambat.
19. Jika derajat suhu adalah dingin dan kecepatan angin adalah lambat dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah agak lambat.

20. Jika derajat suhu adalah dingin dan kecepatan angin adalah lambat dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah agak lambat.
21. Jika derajat suhu adalah dingin dan kecepatan angin adalah lambat dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah agak lambat.
22. Jika derajat suhu adalah dingin dan kecepatan angin adalah sedang dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah lambat.
23. Jika derajat suhu adalah dingin dan kecepatan angin adalah sedang dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah lambat.
24. Jika derajat suhu adalah dingin dan kecepatan angin adalah sedang dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah lambat.
25. Jika derajat suhu adalah dingin dan kecepatan angin adalah cepat dan kelembapan udara adalah besar maka waktu kestabilan suhu adalah lambat.
26. Jika derajat suhu adalah dingin dan kecepatan angin adalah cepat dan kelembapan udara adalah sedang maka waktu kestabilan suhu adalah lambat.
27. Jika derajat suhu adalah dingin dan kecepatan angin adalah cepat dan kelembapan udara adalah kecil maka waktu kestabilan suhu adalah lambat.

Pada saat program dijalankan maka pengguna akan diminta untuk memberikan masukan derajat suhu, kecepatan angin, dan kelembapan udara.

Misalkan data yang dimasukkan adalah sebagai berikut.

1. Suhu = 15°C
2. Kecepatan angin = 30 m/jam
3. Kelembapan udara = 28%

Menghitung Nilai Keanggotaan untuk Suhu

Derajat suhu = 15°C , akan menghasilkan :

- a. Himpunan Fuzzy dingin, dengan nilai keanggotaan 1
- b. Himpunan Fuzzy sedang, dengan nilai keanggotaan 0

Untuk mencari Nilai keanggotaan dari suhu 15°C dari Fuzzy Dingin adalah dengan memasukkan ke rumus :

$$\begin{aligned}\text{Nilai Keanggotaan} &= (20-x) / (20-15) \\ &= (20-15) / (20-15) = 1\end{aligned}$$

Untuk mencari Nilai keanggotaan dari suhu 15°C dari Fuzzy Sedang adalah dengan memasukkan ke rumus :

$$\begin{aligned}\text{Nilai Keanggotaan} &= (x-15) / (20-15) \\ &= (15-15) / (20-15) = 0\end{aligned}$$

Menghitung Nilai Keanggotaan untuk Kecepatan Angin

Kecepatan Angin = 30 M/Jam, akan menghasilkan :

- a. Himpunan Fuzzy Lambat, dengan nilai keanggotaan 0.5
- b. Himpunan Fuzzy Sedang, dengan nilai keanggotaan 0.5

Untuk mencari Nilai keanggotaan dari kecepatan angin 30 M/Jam dari Fuzzy Lambat adalah dengan memasukkan ke rumus :

$$\begin{aligned}\text{Nilai Keanggotaan} &= (40-x) / (40-20) \\ &= (40-30) / 20 = 0.5\end{aligned}$$

Untuk mencari Nilai keanggotaan dari kecepatan angin 30 M/Jam dari Fuzzy Sedang adalah dengan memasukkan ke rumus :

$$\begin{aligned}\text{Nilai Keanggotaan} &= (x-20) / (40-20) \\ &= (30-20) / 20 = 0.5\end{aligned}$$

Menghitung Nilai Keanggotaan untuk Kelembapan Udara

Kelembapan Udara = 28%, akan menghasilkan :

- a. Himpunan Fuzzy Sedang, dengan nilai keanggotaan 0.2
- b. Himpunan Fuzzy Besar, dengan nilai keanggotaan 0.8

Untuk mencari Nilai keanggotaan dari kelembapan udara 28% dari Fuzzy Sedang adalah dengan memasukkan ke rumus :

$$\begin{aligned}\text{Nilai Keanggotaan} &= (30-x) / (30-20) \\ &= (30-28) / 10 = 0.2\end{aligned}$$

Untuk mencari Nilai keanggotaan dari kelembapan udara 28% dari Fuzzy Besar adalah dengan memasukkan ke rumus :

$$\begin{aligned}\text{Nilai Keanggotaan} &= (x-20) / (30-20) \\ &= (28-20) / 10 = 0.8\end{aligned}$$

Dengan memakai Logika Penghubung And, dan setelah dicocokkan dengan semua *rule* yang ada, maka akan diperoleh rule yang mungkin adalah :

1. Dingin – Lambat – Sedang = Agak Lambat
2. Dingin – Lambat – Besar = Agak Lambat
3. Dingin – Sedang – Sedang = Lambat
4. Dingin – Sedang – Besar = Lambat
5. Sedang – Lambat – Sedang = Agak Cepat
6. Sedang – Lambat – Besar = Agak Cepat
7. Sedang – Sedang – Sedang = Sedang
8. Sedang –S edang – Besar = Sedang

Nilai α yang dihasilkan :

$$\alpha_1 = \min (1, 0.5, 0.2) = 0.2$$

$$\alpha_2 = \min (1, 0.5, 0.8) = 0.5$$

$$\alpha_3 = \min (1, 0.5, 0.2) = 0.2$$

$$\alpha_4 = \min (1, 0.5, 0.8) = 0.5$$

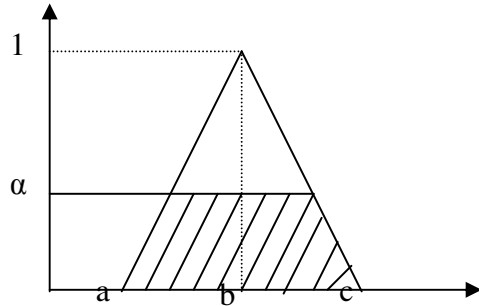
$$\alpha_5 = \min (0, 0.5, 0.2) = 0$$

$$\alpha_6 = \min (0, 0.5, 0.8) = 0$$

$$\alpha_7 = \min (0, 0.5, 0.2) = 0$$

$$\alpha_8 = \min (1, 0.5, 0.8) = 0$$

Cara Penggunaan Sigma :



LS = Luas Segitiga Utuh

$$Q1 = a + (LS * \alpha)$$

$$P1 = c - (LS * \alpha)$$

$$Q = Q1 - a$$

$$P = P1 - a$$

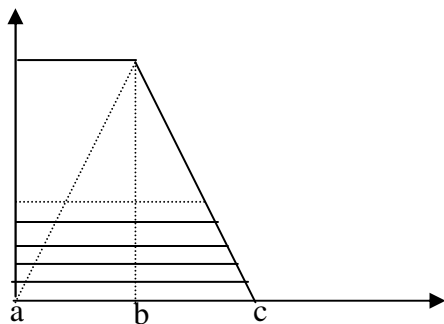
$$AL = P - Q$$

$$T = 1 - \alpha$$

$$\text{Luas} = 0.5 * AL * T \text{ (Luas yang diarsir)}$$

$$\text{LuasSegi} = LS - \text{Luas}$$

$$M = b * \text{luas segi}$$



LS = Luas Segitiga Utuh

$$Q1 = a + (LS * \alpha)$$

$$P1 = c - (LS * \alpha)$$

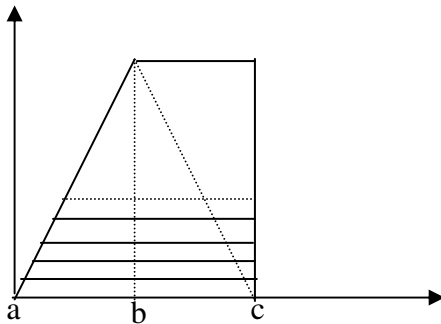
$$Q = Q1 - a$$

$$P = P1 - a$$

$$AL = c - a$$

$$\text{Luas} = 0.5 * (AL + p) * \alpha$$

$$M = b * \text{luas}$$



LS = Luas Segitiga Utuh

$$Q1 = a + (LS * \alpha)$$

$$P1 = c - (LS * \alpha)$$

$$Q = Q1 - a$$

$$P = P1 - a$$

$$AL = c - a$$

$$\text{Luas} = 0.5 * (AL + p) * \alpha$$

$$M = b * \text{luas}$$

Perhitungan Menggunakan Rumus Sigma

Momen untuk $\alpha_1 = 0.2$ dan Agak Lambat

$$Q1 = 15 + (5 * 0.2) = 16$$

$$P1 = 25 - (5 * 0.2) = 24$$

$$Q = 16 - 15 = 1$$

$$P = 24 - 15 = 9$$

$$AL = 9 - 1 = 8$$

$$T = 1 - 0.2 = 0.8$$

$$\text{Luas} = 0.5 * 8 * 0.8 = 3.2$$

$$\text{LuasSegi} = 5 - 3.2 = 1.8$$

$$M = 20 * 1.8 = 36$$

Momen untuk $\alpha_2 = 0.5$ dan Agak Lambat

$$Q1 = 15 + (5 * 0.5) = 17.5$$

$$P1 = 25 - (5 * 0.5) = 22.5$$

$$Q = 17.5 - 15 = 2.5$$

$$P = 22.5 - 15 = 7.5$$

$$AL = 7.5 - 2.5 = 5$$

$$T = 1 - 0.5 = 0.5$$

$$\text{Luas} = 0.5 * 5 * 0.5 = 1.25$$

$$\text{LuasSegi} = 5 - 1.25 = 3.75$$

$$M = 20 * 3.75 = 75$$

Momen untuk $\alpha_3 = 0.2$ dan Lambat

$$Q1 = 20 + (5 * 0.2) = 21$$

$$P1 = 30 - (5 * 0.2) = 29$$

$$Q = 21 - 20 = 1$$

$$P = 29 - 20 = 9$$

$$\text{Luas} = 0.5 * ((10 - q) + 10) * \alpha_3$$

$$= 0.5 * (9 + 10) * 0.2$$

$$= 1.9$$

$$M = 25 * \text{luas} = 25 * 1.9 = 47.5$$

Momen untuk $\alpha_4 = 0.5$ dan Lambat

$$Q1 = 20 + (5 * 0.5) = 22.5$$

$$P1 = 30 - (5 * 0.5) = 27.5$$

$$Q = 22.5 - 20 = 2.5$$

$$P = 27.5 - 20 = 7.5$$

$$\begin{aligned}
 \text{Luas} &= 0.5 * ((10-q)+10) * \alpha_4 \\
 &= 0.5 * ((10-2.5)+10) * 0.5 \\
 &= 4.375 \\
 M &= 25 * 4.375 = 109.375
 \end{aligned}$$

Momen untuk $\alpha_5 = 0$ dan Agak Cepat

$$\begin{aligned}
 Q1 &= 5 + (5 * 0) = 5 \\
 P1 &= 15 - (5 * 0) = 15 \\
 Q &= 5 - 5 = 0 \\
 P &= 15 - 5 = 10 \\
 AL &= 10 - 0 = 10 \\
 T &= 1 - 0 = 1 \\
 \text{Luas} &= 0.5 * 10 * 1 = 5 \\
 \text{LuasSegi} &= 5 - 5 = 0 \\
 M &= 10 * 0 = 0
 \end{aligned}$$

Momen untuk $\alpha_6 = 0$ dan Agak Cepat

$$\begin{aligned}
 Q1 &= 5 + (5 * 0) = 5 \\
 P1 &= 15 - (5 * 0) = 15 \\
 Q &= 5 - 5 = 0 \\
 P &= 15 - 5 = 10 \\
 AL &= 10 - 0 = 10 \\
 T &= 1 - 0 = 1 \\
 \text{Luas} &= 0.5 * 10 * 1 = 5 \\
 \text{LuasSegi} &= 5 - 5 = 0 \\
 M &= 10 * 0 = 0
 \end{aligned}$$

Momen untuk $\alpha_7 = 0$ dan Sedang

$$Q1 = 10 + (5 * 0) = 10$$

$$P1 = 20 - (5 * 0) = 20$$

$$Q = 10 - 10 = 0$$

$$P = 20 - 10 = 10$$

$$AL = 10 - 0 = 10$$

$$T = 1 - 0 = 1$$

$$Luas = 0.5 * 10 * 1 = 5$$

$$LuasSegi = 5 - 5 = 0$$

$$M = 10 * 0 = 0$$

Momen untuk $\alpha_8 = 0$ dan Sedang

$$Q1 = 10 + (5 * 0) = 10$$

$$P1 = 20 - (5 * 0) = 20$$

$$Q = 10 - 10 = 0$$

$$P = 20 - 10 = 10$$

$$AL = 10 - 0 = 10$$

$$T = 1 - 0 = 1$$

$$Luas = 0.5 * 10 * 1 = 5$$

$$LuasSegi = 5 - 5 = 0$$

$$M = 10 * 0 = 0$$

$$C1 = 0.2 * 36 = 7.2$$

$$C2 = 0.5 * 75 = 37.5$$

$$C3 = 0.2 * 47.5 = 9.5$$

$$C4 = 0.5 * 109.375 = 54.6875$$

$$C5 = 0$$

$$C6 = 0$$

$$C7 = 0$$

$$C8 = 0$$

$$\Sigma C = 108.8875$$

$$D1 = 0.2 * 1.8 = 0.36$$

$$D2 = 0.5 * 3.75 = 1.875$$

$$D3 = 0.2 * 1.9 = 0.38$$

$$D4 = 0.5 * 4.375 = 2.1875$$

$$D5 = 0$$

$$D6 = 0$$

$$D7 = 0$$

$$D8 = 0$$

$$\Sigma D = 4.8025$$

$$W = \frac{\Sigma C}{\Sigma D} = 22.67$$

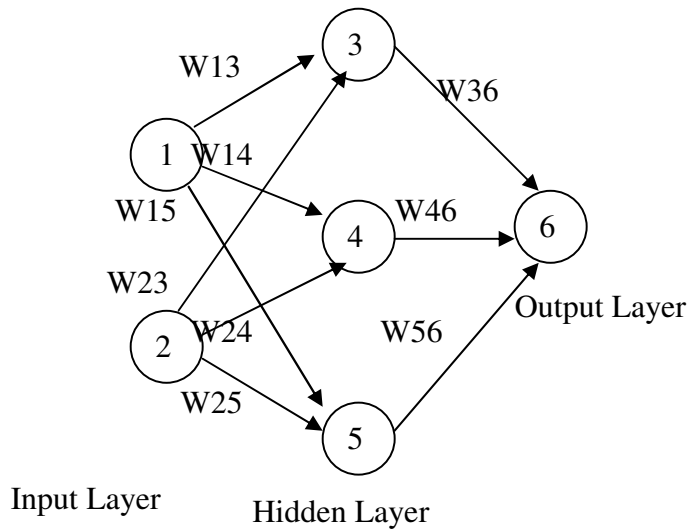
Jaringan Saraf Tiruan

Jaringan saraf tiruan merupakan suatu sistem pemrosesan informasi yang memiliki karakteristik – karakteristik menyerupai jaringan saraf biologi (Fauset, 1994). Hal yang sama diutarakan oleh Simon Haykin, yang menyatakan bahwa JST adalah sebuah mesin yang dirancang untuk memodelkan cara otak manusia mengerjakan fungsi atau tugas – tugas tertentu. Mesin ini memiliki kemampuan menyimpan pengetahuan berdasarkan pengalaman dan menjadikan simpanan pengetahuan yang dimiliki menjadi bermanfaat.

Berikut ini adalah hubungan antara konsep biologi dengan jaringan saraf tiruan menurut Medsker dan Liebowitz (1994) dalam Turban (2001) yang digambarkan pada tabel berikut ini.

Biologi	Jaringan Saraf Tiruan
Soma	Node (Simpul)
Dendrite	Input
Axon	Output
Synapse	Weight (bobot)
Slow Speed	Fast Speed
Terdiri dari banyak Neuron (10^9)	Beberapa Neuron

Berikut ini adalah gambar arsitektur Jaringan Saraf Tiruan



1. Lapisan Masukan (*Input Layer*)

Lapisan masukan merupakan lapisan yang terdiri dari beberapa neuron yang akan menerima sinyal dari luar dan kemudian meneruskan ke neuron – neuron lain dalam jaringan.

2. Lapisan Tersembunyi (*Hidden Layer*)

Lapisan tersembunyi merupakan tiruan dari sel – sel saraf konektor pada jaringan saraf biologis. Lapisan tersembunyi berfungsi meningkatkan kemampuan jaringan dalam memecahkan masalah. Konsekuensi dari adanya lapisan ini adalah pelatihan menjadi makin sulit atau lama.

3. Lapisan Keluaran (*Output Layer*)

Lapisan keluaran berfungsi menyalurkan sinyal – sinyal keluaran hasil pemrosesan jaringan. Lapisan ini juga terdiri dari sejumlah neuron. Lapisan keluaran merupakan tiruan dari sel – sel saraf motor pada jaringan saraf biologis.

Istilah – istilah Jaringan Saraf Tiruan

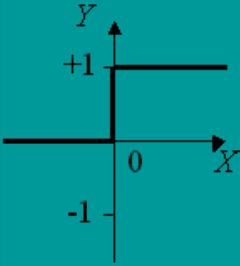
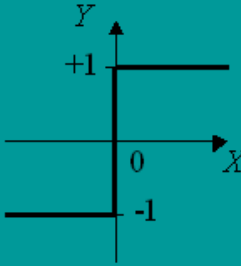
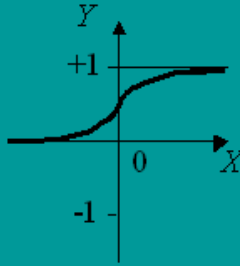
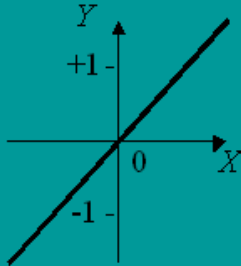
Berikut ini adalah beberapa istilah Jaringan Saraf yang sering ditemui :

1. Neuron atau Node atau Unit : Sel saraf tiruan yang merupakan elemen pengolahan jaringan saraf tiruan. Setiap neuron menerima data input, memproses input tersebut, dan mengirimkan hasilnya berupa sebuah Output.

2. Jaringan : Kumpulan neuron yang saling terhubung dan membentuk lapisan.
3. Input atau Masukan : Berkoresponden dengan sebuah atribut tunggal dari sebuah pola atau data lain dari dunia luar. Sinyal – sinyal input ini kemudian diteruskan ke lapisan selanjutnya.
4. Output atau Keluaran : Solusi atau hasil pemahaman jaringan terhadap data input. Tujuan pembangunan jaringan saraf tiruan sendiri adalah untuk mengetahui nilai output.
5. Lapisan Tersembunyi (Hidden Layer) : Lapisan yang tidak secara langsung berinteraksi dengan dunia luar. Lapisan ini memperluas kemampuan jaringan saraf tiruan dalam menghadapi masalah – masalah yang kompleks.
6. Bobot : Bobot dalam jaringan saraf tiruan merupakan nilai matematis dari koneksi, yang mentransfer data dari satu lapisan ke lapisan lainnya. Bobot ini digunakan untuk mengatur jaringan sehingga jaringan saraf tiruan bisa menghasilkan output yang diinginkan sekaligus bertujuan membuat jaringan tersebut belajar.
7. Summation Function : Fungsi yang digunakan untuk mencari rata – rata bobot dari semua elemen input. Yang sederhana adalah dengan mengalikan setiap nilai input (X_j) dengan bobotnya (W_{ij}) dan menjumlahkannya (disebut penjumlahan berbobot, atau Y_i).

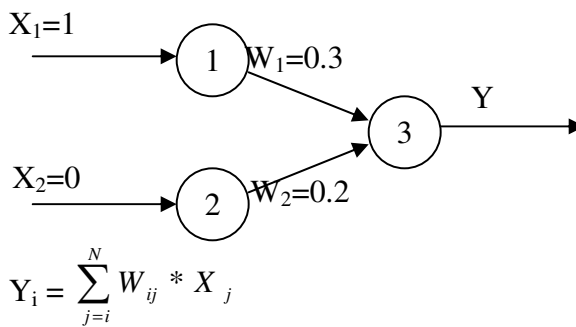
$$Y_i = \sum_{j=1}^N W_{ij} * X_j$$

8. Fungsi Aktivasi atau Fungsi Transfer : Fungsi yang menggambarkan hubungan antara tingkat aktivasi internal (*summation function*). Beberapa fungsi aktivasi yang sering digunakan adalah : Step Function, Sigmoid Function, Sign Function, dan Linear Function.

Step function	Sign function	Sigmoid function	Linear function
 $Y^{step} = \begin{cases} 1, & \text{if } X \geq 0 \\ 0, & \text{if } X < 0 \end{cases}$	 $Y^{sign} = \begin{cases} +1, & \text{if } X \geq 0 \\ -1, & \text{if } X < 0 \end{cases}$	 $Y^{sigmoid} = \frac{1}{1+e^{-X}}$	 $Y^{linear} = X$

Solat :

Diketahui gambar berikut ini :



Dengan menggunakan fungsi aktivasi Step :

$$\begin{aligned}
 Y &= X_1 * W_1 + X_2 * W_2 \\
 &= 1 * 0,3 + 0 * 0,2 \\
 &= 0,3 \geq 0
 \end{aligned}$$

$$Y = 1$$

Dengan menggunakan fungsi aktivasi Sign :

$$\begin{aligned}
 Y &= X_1 * W_1 + X_2 * W_2 \\
 &= 1 * 0,3 + 0 * 0,2 \\
 &= 0,3 \geq 0
 \end{aligned}$$

$$Y = +1$$

Dengan menggunakan fungsi aktivasi Linear :

$$\begin{aligned} Y &= X_1 * W_1 + X_2 * W_2 \\ &= 1 * 0,3 + 0 * 0,2 \\ &= 0,3 \end{aligned}$$

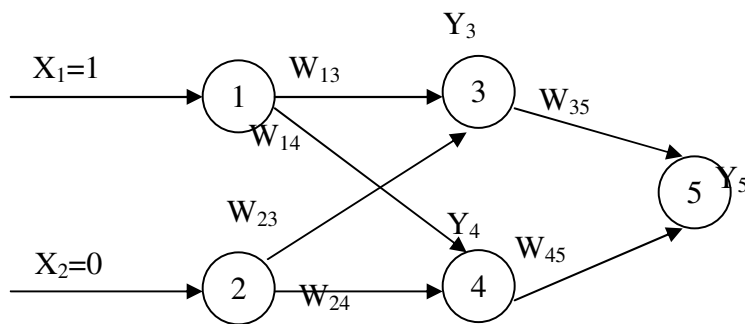
$$Y = 0,3$$

Dengan menggunakan fungsi aktivasi Linear :

$$Y = \frac{1}{1 + e^{-0,3}} = \dots\dots$$

Solat :

Hitung keluaran neuron berikut :



$$W_{13} = 0,3$$

$$W_{24} = 0,2$$

$$W_{14} = 0,5$$

$$W_{35} = 0,1$$

$$W_{23} = 0,2$$

$$W_{45} = 0,4$$

Menggunakan Fungsi Aktivasi Step :

$$\begin{aligned} Y_3 &= X_1 * W_{13} + X_2 * W_{23} \\ &= 1 * 0,3 + 0 * 0,2 \\ &= 0,3 \geq 0 \end{aligned}$$

$$Y_3 = 1$$

$$\begin{aligned} Y_4 &= X_1 * W_{14} + X_2 * W_{24} \\ &= 1 * 0,5 + 0 * 0,2 \end{aligned}$$

$$= 0,5 \geq 0$$

$$Y_4 = 1$$

$$Y_5 = Y_3 * W_{35} + Y_4 * W_{45}$$

$$= 1 * 0,1 + 1 * 0,4$$

$$= 0,5 \geq 0$$

$$Y_5 = 1$$

Perceptron

Frank Rosenblatt bersama – sama dengan beberapa peneliti lain (1958, 1959, 1962) memperkenalkan dan mengembangkan sebuah kelompok besar jaringan saraf tiruan yang disebut dengan Perceptron.

Adapun tahapan – tahapan dalam membangun sebuah Jaringan Saraf Tiruan dengan algoritma Perceptron adalah sebagai berikut.

Step 1 : Inisialisasi

Dilakukan dengan memberikan nilai awal terhadap nilai input, output yang diharapkan (Y_d), Weight, Learning Rate (α), Threshold (θ)

Output ada 2 :

1. Output yang diharapkan = target
2. Output aktual = hasil proses

Step 2 : Aktivasi

Menghitung output aktual dengan menggunakan nilai – nilai yang telah diberikan pada Step 1.

$$Y(P) = \text{FA} \left(\sum_{i=1}^n X_i(P) \cdot W_i(P) - \theta \right)$$

Ket : P = iterasi

Step 3 : Weight Training

Mengupdate nilai weight dengan rumus :

$$W_i(P+1) = W_i(P) + \Delta W_i(P)$$

Di mana :

$$\Delta W_i(P) = \alpha \cdot X_i(P) \cdot e(P)$$

Keterangan :

α = Learning Rate

e = Error

$e = Y_d - \text{Actual}$

Step 4 : Iterasi

Proses pengulangan sampai error minimal atau output yang diharapkan = atau mendekati output aktual.

Berikut ini adalah contoh penerapan algoritma Perceptron untuk mengenali Logika And.

Untuk Epoch 1

Input		YD	Weight		Actual	Error	Final Weight	
X1	X2		W1	W2			W1	W2
0	0	0	0,3	-0,1				
0	1	0						
1	0	0						
1	1	1						

$$\theta = 0,2$$

$$\alpha = 0,1$$

Step 2 : Activation

$$\begin{aligned}
 Y(1) &= \text{Step}(X1(1) * W1(1) + X2(1) * W2(1) - \theta) \\
 &= \text{Step}(0 * 0,3 + 0 * -0,1 - 0,2) \\
 &= \text{Step}(0 - 0 - 0,2) \\
 &= \text{Step}(-0,2) \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 \text{Error} &= YD - \text{Actual} \\
 &= 0 - 0
 \end{aligned}$$

Step 3 : Weight Training

$$W_i(P+1) = W_i(P) + \Delta W_i(P)$$

Di mana :

$$\Delta W_i(P) = \alpha \cdot X_i(P) \cdot e(P)$$

$$\begin{aligned}
 \Delta W1(1) &= \alpha \cdot X1(1) \cdot e(1) \\
 &= 0,1 * 0 * 0 \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 \Delta W2(1) &= \alpha \cdot X2(1) \cdot e(1) \\
 &= 0,1 * 0 * 0 \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 W1(2) &= W1(1) + \Delta W1(1) \\
 &= 0,3 + 0 \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 W2(2) &= W2(1) + \Delta W2(1) \\
 &= -0,1 + 0 \\
 &= -0,1
 \end{aligned}$$

Sehingga diperoleh :

Input		YD	Weight		Actual	Error	Final Weight	
X1	X2		W1	W2			W1	W2
0	0	0	0,3	-0,1	0	0	0,3	-0,1
0	1	0	0,3	-0,1				
1	0	0						
1	1	1						

Step 2 : Activation

$$\begin{aligned}
 Y(2) &= \text{Step}(X1(2) * W1(2) + X2(2) * W2(2) - \theta) \\
 &= \text{Step}(0 * 0,3 + 1 * -0,1 - 0,2) \\
 &= \text{Step}(-0,1 - 0,2) \\
 &= \text{Step}(-0,3) \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 \text{Error} &= YD - \text{Actual} \\
 &= 0 - 0
 \end{aligned}$$

Step 3 : Weight Training

$$W_i(P+1) = W_i(P) + \Delta W_i(P)$$

Di mana :

$$\Delta W_i(P) = \alpha \cdot X_i(P) \cdot e(P)$$

$$\begin{aligned}
 \Delta W1(2) &= \alpha \cdot X1(2) \cdot e(2) \\
 &= 0,1 * 0 * 0 \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 \Delta W2(2) &= \alpha \cdot X2(2) \cdot e(2) \\
 &= 0,1 * 1 * 0 \\
 &= 0
 \end{aligned}$$

$$\begin{aligned}
 W1(3) &= W1(2) + \Delta W1(2) \\
 &= 0,3 + 0 \\
 &= 0,3
 \end{aligned}$$

$$\begin{aligned}
 W2(3) &= W2(2) + \Delta W2(2) \\
 &= -0,1 + 0 \\
 &= -0,1
 \end{aligned}$$

Sehingga Diperoleh :

Input		YD	Weight		Actual	Error	Final Weight	
X1	X2		W1	W2			W1	W2
0	0	0	0,3	-0,1	0	0	0,3	-0,1
0	1	0	0,3	-0,1	0	0	0,3	-0,1
1	0	0	0,3	-0,1				
1	1	1						

Step 2 : Activation

$$\begin{aligned}
 Y(3) &= \text{Step}(X1(3) * W1(3) + X2(3) * W2(3) - \theta) \\
 &= \text{Step}(1 * 0,3 + 0 * -0,1 - 0,2) \\
 &= \text{Step}(0,3 - 0,2) \\
 &= \text{Step}(0,1) \\
 &= 1
 \end{aligned}$$

$$\begin{aligned}
 \text{Error} &= YD - \text{Actual} \\
 &= 0 - 1 \\
 &= -1
 \end{aligned}$$

Step 3 : Weight Training

$$W_i(P+1) = W_i(P) + \Delta W_i(P)$$

Di mana :

$$\Delta W_i(P) = \alpha \cdot X_i(P) \cdot e(P)$$

$$\Delta W1(3) = \alpha \cdot X1(3) \cdot e(3)$$

$$= 0.1 \cdot 1 \cdot -1$$

$$= -0,1$$

$$\Delta W2(3) = \alpha \cdot X2(3) \cdot e(3)$$

$$= 0.1 \cdot 0 \cdot -1$$

$$= 0$$

$$W1(4) = W1(3) + \Delta W1(3)$$

$$= 0.3 + (-0,1)$$

$$= 0,2$$

$$W2(4) = W2(3) + \Delta W2(3)$$

$$= -0,1 + 0$$

$$= -0,1$$

Sehingga Diperoleh :

Input		YD	Weight		Actual	Error	Final Weight	
X1	X2		W1	W2			W1	W2
0	0	0	0,3	-0,1	0	0	0,3	-0,1
0	1	0	0,3	-0,1	0	0	0,3	-0,1
1	0	0	0,3	-0,1	1	-1	0,2	-0,1
1	1	1	0,2	-0,1				

Step 2 : Activation

$$Y(4) = \text{Step}(X1(4) \cdot W1(4) + X2(4) \cdot W2(4) - \theta)$$

$$= \text{Step}(1 \cdot 0,2 + 1 \cdot -0,1 - 0,2)$$

$$= \text{Step}(0,2 - 0,1 - 0,2)$$

$$= \text{Step } (-0,1)$$

$$= 0$$

$$\text{Error} = YD - \text{Actual}$$

$$= 1 - 0$$

$$= 1$$

Step 3 : Weight Training

$$W_i(P+1) = W_i(P) + \Delta W_i(P)$$

Di mana :

$$\Delta W_i(P) = \alpha \cdot X_i(P) \cdot e(P)$$

$$\Delta W1(4) = \alpha \cdot X1(4) \cdot e(4)$$

$$= 0.1 * 1 * 1$$

$$= 0,1$$

$$\Delta W2(4) = \alpha \cdot X2(4) \cdot e(4)$$

$$= 0.1 * 1 * 1$$

$$= 0,1$$

$$W1(5) = W1(4) + \Delta W1(4)$$

$$= 0,2 + 0,1$$

$$= 0,3$$

$$W2(5) = W2(4) + \Delta W2(4)$$

$$= -0,1 + 0,1$$

$$= 0$$

Sehingga diperoleh :

Input		YD	Weight		Actual	Error	Final Weight	
X1	X2		W1	W2			W1	W2
0	0	0	0,3	-0,1	0	0	0,3	-0,1

0	1	0	0,3	-0,1	0	0	0,3	-0,1
1	0	0	0,3	-0,1	1	-1	0,2	-0,1
1	1	1	0,2	-0,1	0	1	0,3	0

Terlihat bahwa pada epoch / proses yang ke – 1 hasil yang diinginkan masih belum tercapai, sehingga perlu dilakukan ke epoch yang ke – 2 yang caranya sama dengan epoch yang ke – 1 dan bila masih belum mencapai hasil yang diinginkan maka bisa dilanjutkan ke epoch yang ke – 3, dan seterusnya sehingga dicapai hasil yang diinginkan.