Sistem Pakar

Buku Pegangan Kuliah

Jusak Irawan

Sekolah Tinggi
Manajemen Informatika &
Teknik Komputer Surabaya

Buku Pegangan Kuliah

SISTEM PAKAR

Jusak Irawan

Sekolah Tinggi Manajemen Informatika
& Teknik Komputer Surabaya (STIKOM)

August 2007

Contents

List of Figures

List of Tables

Chapter 0

Kontrak Perkuliahan

Nama Matakuliah	: Sistem Pakar
Kode Matakuliah	: 410103044
Jumlah SKS	: 3 sks
Manfaat matakuliah	: Membantu mahasiswa agar dapat merepresentasikan pengetahuan ke dalam bentuk yang dapat diolah oleh sistem pakar dan mendorong mahasiswa untuk dapat membuat aplikasi sistem pakar.
Deskripsi matakuliah	: Mata kuliah ini membahas tentang konsep dasar sistem pakar, representasi pengetahuan dengan menggunakan: propotitional logic, predicate calculus, sistem berbasis aturan, semantic network dan frame; representasi pengetahuan samar dengan menggunakan: bayesian, certainty factor dan sistem fuzzy; dan Jaringan Syaraf Tiruan.
Tujuan Instruksional Umum	: Setelah mengikuti matakuliah ini mahasiswa akan memahami model-model representasi pengetahuan, memiliki kemampuan untuk menarik kesimpulan (inference) dari fakta yang digambarkan dalam model-model representasi pengetahuan, serta mampu merancang dan membuat implementasi sistem pakar dengan menggunakan bahasa pemrograman.

Materi Perkuliahan

Materi perkuliahan meliputi:

- Pengantar sistem pakar dan kecerdasan buatan
- Arsitektur sistem pakar.
- Representasi pengetahuan dengan *Propotitional Logic* dan *Predicate Calculus*.
- Sistem berbasis aturan: Forward-chaining.
- Sistem berbasis aturan: Backward-chaining.
- Representasi pengetahuan dengan *Semantic Network* dan *Frame*.
- Certainty theory.
- Sistem Fuzzy.
- Jaringan Syaraf Tiruan (Artificial Neural Network).

Diagram Materi

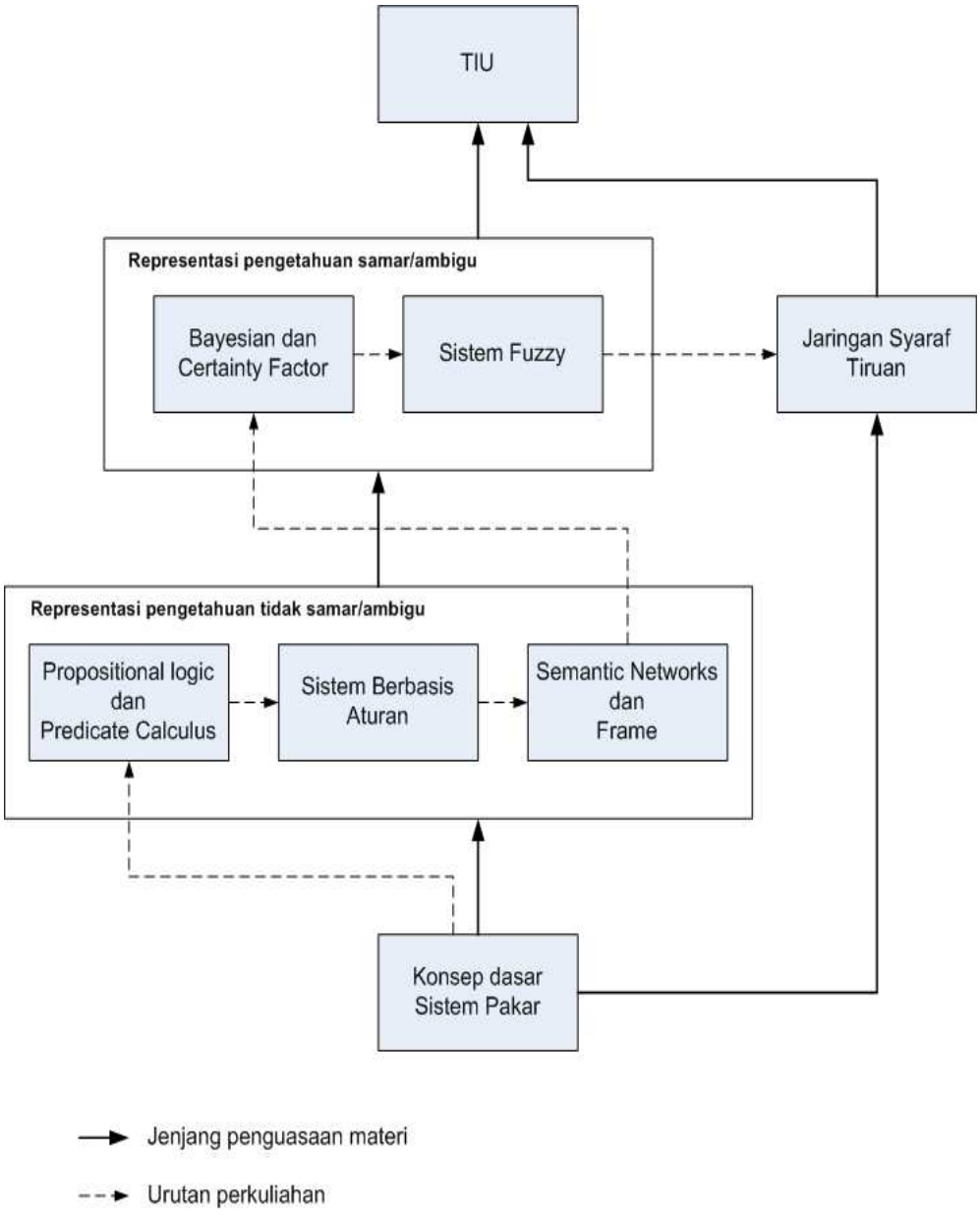

Figure 1: Diagram materi Sistem Pakar.

Kriteria Penilaian

Penilaian dilakukan dengan menggunakan kriteria yang berlaku di STIKOM, yaitu :

Nilai Huruf	**Nilai Numerik**	**Bobot**	**Kategori**
A+	95-100	4.0	Istimewa
A	90-94	3.75	Istimewa
A-	85-89	3.5	Sangat Baik
B+	80-84	3.25	Sangat Baik
B	75-79	3.0	Baik
B-	70-74	2.75	Baik
C+	65-69	2.5	Cukup Baik
C	60-64	2.25	Cukup Baik
C-	55-59	2.0	Agak Baik
D	41-54	1.0	Kurang Baik

Dalam menentukan nilai evaluasi akhir akan digunakan pembobotan sebagai berikut :

1. Tugas 30%
2. Ujian Tengah Semester 30%
3. Ujian Akhir Semester 40%

Jadwal Perkuliahan

Pertemuan ke-	Topik Bahasan	Bahan Acuan
1	Menjelaskan kontrak perkuliahan Konsep dasar Sistem Pakar Heuristic searching	Buku 2,4
2	Propositional logic Predicate Calculus	Buku 2,4
3	Sistem berbasis aturan: forward reasoning	Buku 2,4
4	Sistem berbasis aturan: backward reasoning	Buku 2,4
5	Membangun sistem berbasis aturan	Buku 1
6	Semantic Networks dan Frame	Buku 2,4
7,8	Bayesian dan Certainty factor	Buku 2,4
9,10,11	Sistem Fuzzy	Buku 5,6,8
12,13,14	Jaringan Syaraf Tiruan Presentasi tugas proyek	Buku 3,6,7,9

Bibliography

[1] Dologite, D.G., *Developing Knowledge-Based Systems using VP-EXPERT* Macmillan Publishing Company New York, USA, 1993

[2] Durkin, John., *Expert Systems: Design and Development,* Prentice Hall International, Inc., New Jersey, USA, 1994.

[3] Fausett, Laurene., *Fundamental of Neural Networks: Architectures, Algorithms and Applications,* Prentice Hall, Upper Sadle River, NJ, 1994.

[4] Gonzales, A.J. and Douglas, D.D., *The Engineering of Knowledge Based System: Theory and Practice,* Prentice Hall International, Inc., New Jersey, USA, 1993.

[5] Klir, George J., and Yuan, Bo, *Fuzzy Sets and Fuzzy Logic,* Prentice Hall International, Inc., New Jersey, USA, 1995.

[6] Kosko, Bart., *Neural Network and Fuzzy System,* Prentice Hall International, Inc., New Jersey, USA, 1992.

[7] Kumar, Satish., *Neural Network: A Classroom Approach,* Tata McGraw-Hill Companies, New Delhi, India, 2004.

[8] Mathworks, *Fuzzy Logic Toolbox for use in MATLAB:User's Guide,* The MathWorks, Inc., Natic, MA, USA, 2006.

[9] Mathworks, *Neural Network Toolbox for use in MATLAB:User's Guide,* The MathWorks, Inc., Natic, MA, USA, 2006.

Chapter 1

Pengantar Sistem Pakar

Tujuan Instruksional Khusus

- Mahasiswa mampu menjelaskan konsep dasar Sistem Pakar.
- Mahasiswa mampu memberi contoh aplikasi-aplikasi sistem pakar dalam sistem komputer modern.
- Mahasiswa memahami konsep Heuristic Searching.

1.1 Apa yang dimaksud dengan Sistem Pakar (Expert System)?

Note:

Sistem Pakar adalah sebuah program komputer yang mencoba meniru atau mensimulasikan pengetahuan (knowledge) dan ketrampilan (skill) dari seorang pakar pada area tertentu. Selanjutnya sistem ini akan mencoba memecahkan suatu permasalahan sesuai dengan kepakarannya.

- Sistem pakar merupakan salah satu aplikasi dari kecerdasan buatan (artificial intelligence).
- AI sendiri berakar dari keinginan manusia untuk membuat sebuah mesin cerdas.
- Dewasa ini sistem pakar telah diaplikasikan dalam banyak bidang, misalnya: industri manufaktur, pertanian, medis, militer de-es-be.

1.2 Sejarah Sistem Pakar

1943 Post E.L. membuktikan bahwa permasalahan-permasalahan komputasi dapat diselesaikan dengan aturan IF-THEN.

1961 General Problem Solver (GPS) oleh A. Newell and H. Simon. Adalah sebuah program yang dibangun untuk menyelesaikan permasalahan mulai dari *games* sampai *matematika integral*.

1969 DENDRAL. Dibangun di Stamford University atas permintaan NASA (Buchanan and Feigenbaum) untuk melakukan analisis kimiawi terhadap kondisi tanah di planet Mars.

1970s MCYN. Dibuat untuk diagnosa medis oleh Buchanan dan Shortliffe.

1982 R1/XCON adalah sistem pakar pertama yang dibuat oleh para peneliti di Carnegie Melon University (CMU).

1.3 Contoh Permasalahan

Contoh klasik permasalahan dalam sistem pakar adalah *masalah 2 ember air*. "Diberikan 2 ember air yang berkapasitas 8 litter dan 6 liter. Kita dapat mengisi satu ember dari ember lainnya dan proses penakaran

hanya dengan memakai 2 ember tersebut. Bagaimana kita bisa mendapatkan tepat 4 liter dalam ember 8 liter. Asumsikan tidak ada air yang hilang dalam proses penakaran".
Langkah penyelesaian:

1. Menentukan aksi-aksi (problem space) yang bisa mengubah kondisi pada kedua ember dalam bentuk *rule* atau *tree-diagram* seperti dalam Gambar 1.1 Contoh kemungkinan aksi-aksi:
   (a) Isi ember 8 liter.
   (b) Isi ember 6 liter.
   (c) Kosongkan ember 8 liter.
   (d) Kosongkan ember 6 liter.
   (e) Isikan seluruh air dalam ember 8 liter ke 6 liter.
   (f) Isikan seluruh air dalam ember 6 liter ke 8 liter.
   (g) Penuhi ember 8 liter dari 6 liter.
   (h) Penuhi ember 6 liter dari 8 liter.
2. Menentukan urutan aksi untuk menghasilkan solusi, seperti:
   $(0,0) \xrightarrow{b} (0,6) \xrightarrow{f} (6,0) \xrightarrow{b} (6,6) \xrightarrow{g} (8,4) \xrightarrow{c} (0,4) \xrightarrow{f} (4,0)$

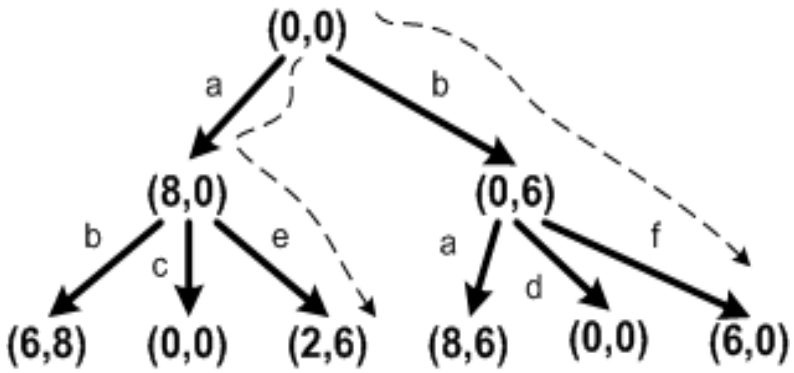

Figure 1.1: Diagram tree untuk "jug problem".

1.4 Sistem Berbasis Aturan

Sistem berbasis aturan (rule-based system) adalah sebuah program yang menggunakan aturan IF-THEN. Model ini berbeda dengan pemrograman konvensional, misalnya *rule* tidak harus berada pada urutan tertentu. Contoh dari sistem berbasis aturan adalah sbb:

IF	Sabtu OR Minggu	THEN	Nonton bioskop
IF	NOT (Sabtu OR Minggu)	THEN	Bekerja
IF	Nonton bioskop	THEN	Pergi keluar
IF	Bekerja	THEN	Pergi keluar
IF	NOT (Bisa pergi keluar)	THEN	Tiggal di rumah
IF	Cuaca baik	THEN	Bisa pergi keluar
IF	Hujan	THEN	Bawa Payung
IF	Hujan AND Bawa Payung	THEN	Bisa pergi keluar

1.5 Perbedaan Sistem Konvensional dan Sistem Pakar:

Table 1.1: Perbandingan Sistem Konvensional dan Sistem Pakar

Sistem Konvensional	Sistem Pakar
Fokus pada solusi	Fokus pada problem
Programmer bekerja sendiri	Team-work
Sequencial	iterative

1.6 Data, Informasi dan Pengetahuan (knowledge)

Note:

- *Data* merupakan hasil pengukuran atau catatan (record) tentang sebuah kejadian (mis. suhu, waktu, harga dsb). data dapat berupa angka, huruf, gambar, suara dsb.
- *Informasi* merupakan hasil olahan dari data sedemikian rupa sehingga karakteristik dari data tersebut dapat diuji, misalnya rata-rata, varian, distribusi dsb.
- *Pengetahuan* merupakan informasi yang diletakkan pada konteks/lingkungan tertentu, misalnya peta jawa, distribusi minyak di Indonesia, dsb.
- Data, informasi dan pengetahuaan dapat dikelompokkan ke dalam level abstraksi seperti ditunjukkan dalam Gambar 1.2 di bawah ini:

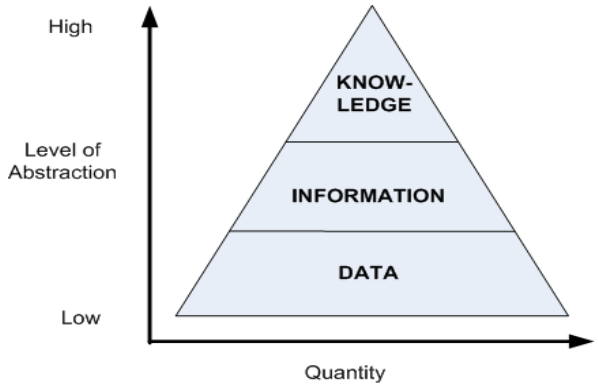

Figure 1.2: Diagram Data, Information dan Knowledge.

1.7 Struktur Sistem Pakar

- Secara umum struktur sebuah sistem pakar terdiri atas 3 komponen utama, yaitu: *knowledge base*, *working memory*, dan *inference engine*. Lihat Gambar 1.3.

- *Knowledge base (basis pengetahuan)* adalah bagian dari sebuah sistem pakar yang mengandung/menyimpan pengetahuan (domain knowledge). Knowledge base yang dikandung oleh sebuah sistem pakar berbeda antara satu dengan yang lain tergantung pada bidang kepakaran dari sistem yang dibangun. Misalnya, medical expert system akan memiliki basis pengetahuan tentang hal-hal yang berkaitan dengan medis. Knowledge base direpresentasikan dalam berbagai macam bentuk, salah satunya adalah dalam bentuk sistem berbasis aturan (ruled-based system).

- *Working memory* mengandung/menyimpan fakta-fakta yang ditemukan selama proses konsultasi dengan sistem pakar. Selama proses konsultasi, *user* memasukkan fakta-fakta yang dibutuhkan. Kemudian sistem akan mencari padanan tentang fakta tersebut dengan informasi yang ada dalam knowledge base untuk menghasilkan fakta baru. Sistem akan memasukkan fakta baru ini ke dalam working memory. Jadi working memory menyimpan informasi tentang fakta-fakta yang dimasukkan oleh user ataupun fakta baru hasil kesimpulan dari sistem.

- *Inference engine* bertugas mencari padanan antara fakta yang ada di dalam working memory dengan fakta-fakta tentang domain knowledge tertentu yang ada di dalam knowledge base, selanjutnya inference engine akan menarik/mengambil kesimpulan dari problem yang diajukan kepada sistem.

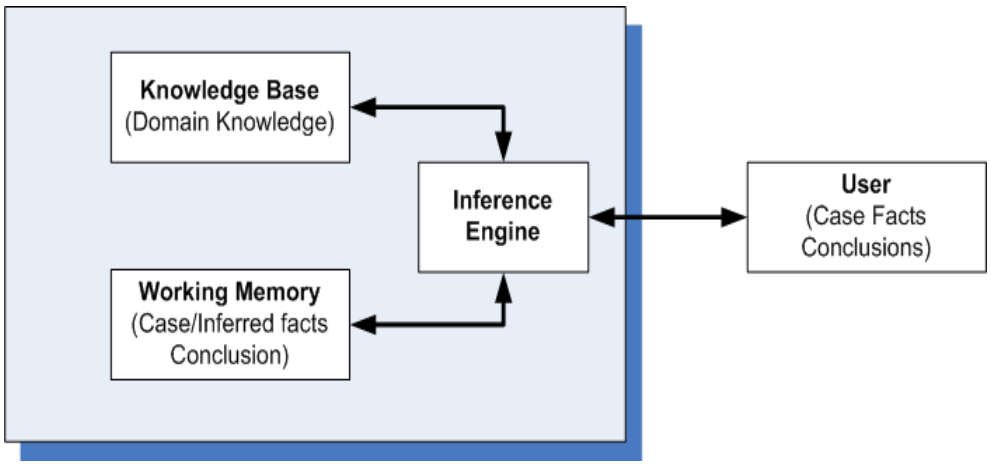

Figure 1.3: Struktur dasar sistem pakar.

1.8 *Heuristic Searching* Sebagai Dasar dari Artificial Intelligence (AI)

- Para peneliti awal kecerdasan buatan menitik beratkan pada penyelesaian masalah yang tidak menggunakan metoda komputasi konvensional. Hal ini disebabkan metoda pemecahan masalah konvensional tidak dapat lagi digunakan. Permasalahan pada sistem AI tidak memiliki algoritma tertentu. Kalaupun ada tentulah sangat kompleks.

- Karena itu haruslah ditemukan sebuah teknik baru yang mirip dengan cara yang digunakan oleh manusia untuk menyelesaikan masalah dan dapat diimplementasikan pada komputer.

- Salah satu metoda yang cukup terkenal adalah metoda *searching*.

- Searching dalam sebuah struktur data telah menjadi dasar bagi algoritma komputer, tetapi proses searching pada AI memiliki perbedaan. Metoda searching pada AI merupakan searching terhadap *problem space* bukan searching data (e.g., angka, karakter, string) tertentu. Proses searching ini berupa jalur yang menggambarkan keadaan awal sebuah masalah menuju kepada penyelesaian masalah

yang diinginkan (i.e., the solved problem). Jalur-jalur ini mengambarkan langkah-langkah penyelesaian masalah. Melalui proses searching menuju sebuah penyelesaian akan terbentuk sebuah *solution space*.

- Perhatikan contoh penyelesaian masalah komputer pada Gambar 1.4. Langkah pertama untuk mengetahui apakah komputer dapat digunakan atau tidak adalah men-switch ON. Selanjutnya dengan melakukan inspeksi terhadap kondisi lampu indikator kita dapat menentukan langkah berikutnya. Misalnya kondisi lampu OFF. Dengan melakukan searching terhadap problem space kita akan tiba pada sebuah penyelesaian masalah agar komputer dapat diaktifkan kembali.

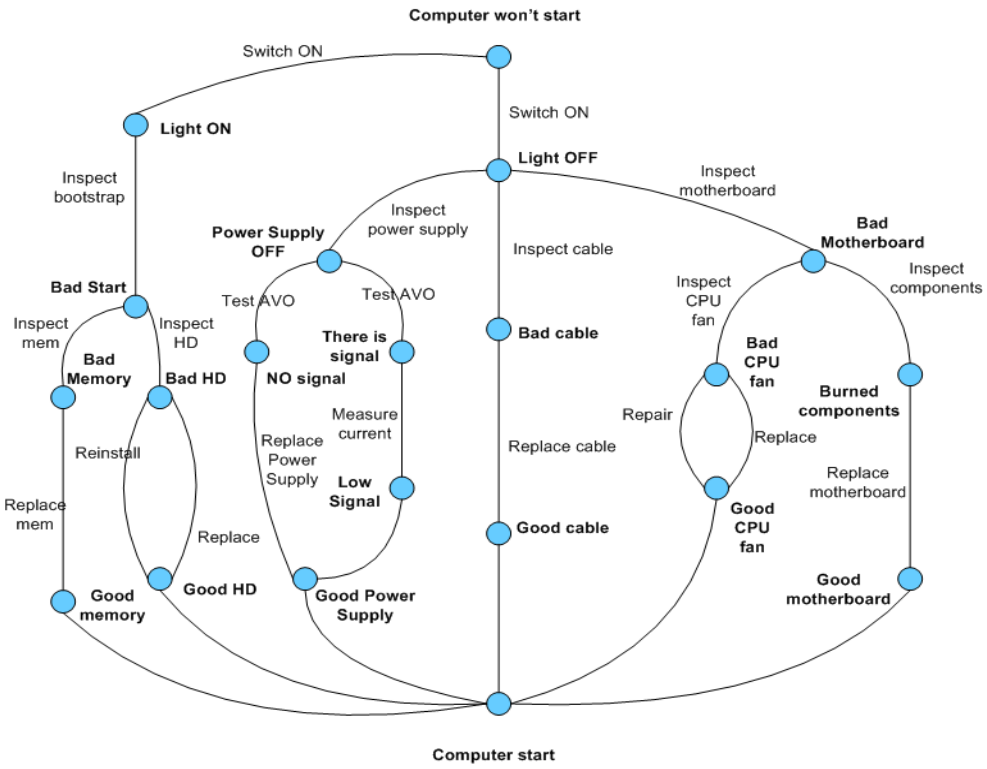

Figure 1.4: Representasi jaringan dari sebuah problem space.

1.8.1 Depth-first Search

Depth-first search (DFS) adalah proses searching sistematis buta yang melakukan ekpansi sebuah path (jalur) menuju penyelesaian masalah sebelum melakukan ekplorasi terhadap path yang lain. Proses searching mengikuti sebuah path tunggal sampai menemukan goal atau dead end. Apabila proses searching menemukan dead-end, DFS akan melakukan penelusuran balik ke node terakhir untuk melihat apakah node tersebut memiliki path cabang yang belum dieksplorasi. Apabila cabang ditemukan, DFS akan melakukan cabang tersebut. Apabila sudah tidak ada lagi cabang yang dapat dieksplorasi, DFS akan kembali ke node parent dan melakukan proses searching terhadap cabang yang belum dieksplorasi dari node parent sampai menemukan penyelesaian masalah. Urutan proses searching DFS ditunjukkan dalam Gambar 1.5 adalah: A, B, E, F, G, C, ...

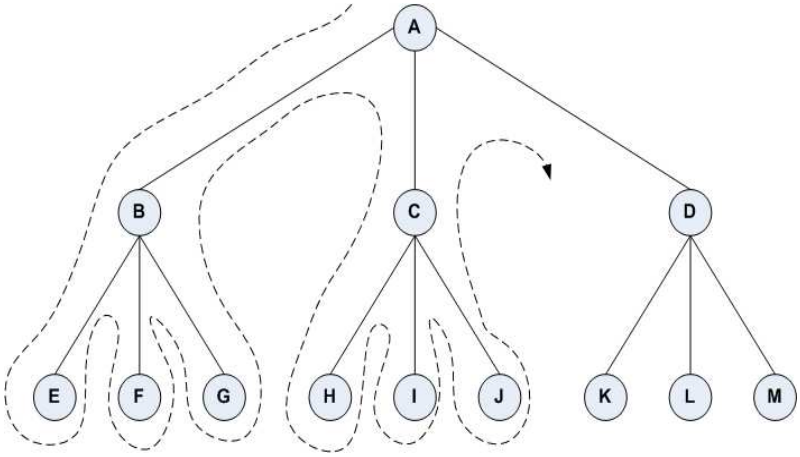

Figure 1.5: Diagram pohon dari DFS.

1.8.2 Breadth-first Search

Breadth-first search (BFS) melakukan proses searching pada semua node yang berada pada level atau hirarki yang sama terlebih dahulu sebelum

melanjutkan proses searching pada node di level berikutnya. urutan proses searching BFS ditunjukkan dalam Gambar 1.6 adalah: A,B,C,D,E,F, ...

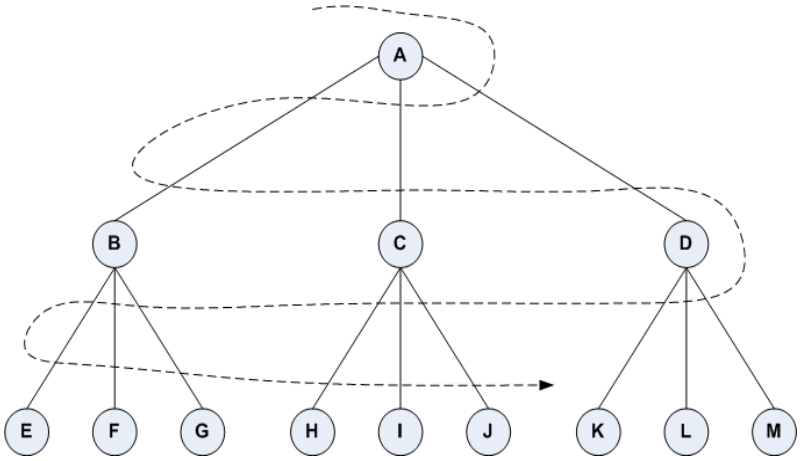

Figure 1.6: Diagram pohon dari BFS.

1.8.3 Hill-Climbing Search

Metoda Hill-climbing merupakan variasi dari depth-first search. Dengan metoda ini, eksplorasi terhadap keputusan dilakukan dengan cara depth-first search dengan mencari path yang bertujuan menurunkan cost untuk menuju kepada goal/keputusan. Sebagai contoh kita mencari arah menuju Tugu Pahlawan, setiap kali sampai dipersimpangan jalan kita berhenti dan mencari arah mana yang kira-kira akan mengurangi jarak menuju Tugu Pahlawan. Dengan cara demikian sebetulnya kita berasumsi bahwa secara umum arah tertentu semakin dekat ke Tugu Pahlawan.

1.8.4 Branch and Bound Search

Perhatikan Gambar 1.7 di bawah ini. Bagaimana menggunakan metoda *branch and bound* untuk mencari terpendek dari kota Semarang menuju kota Probolinggo?

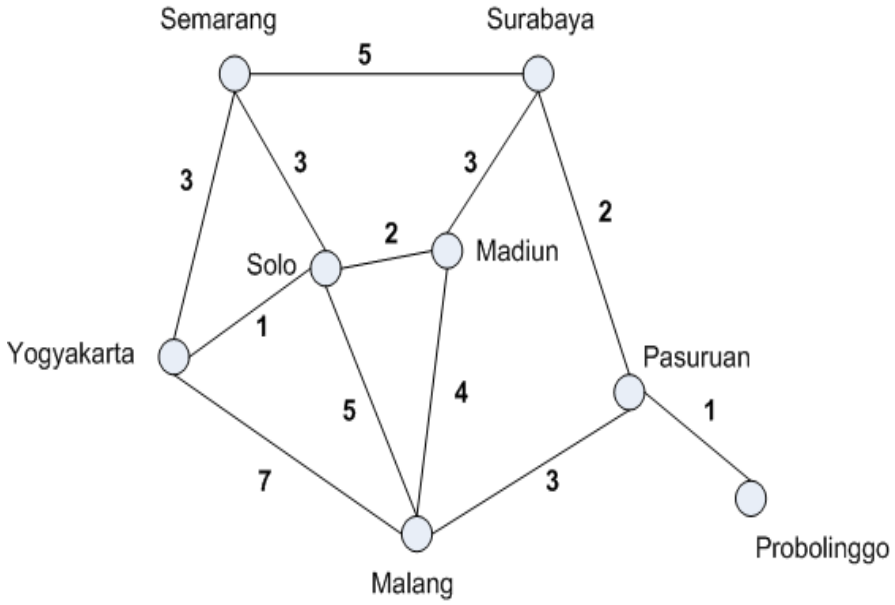

Figure 1.7: Diagram jaringan antar kota.

Dengan metoda branch and bound, proses searching ditunjukkan dalam Gambar 1.8 dan 1.9.

1.8.5 Best-First Search

Best-First Search melakukan proses searching dengan cara memberikan estimasi berapa jauh node asal dari solusi yang diinginkan. Dengan metoda ini, proses dilakukan dengan melakukan ekspansi terhadap setiap node yang memiliki estimasi terpendek.

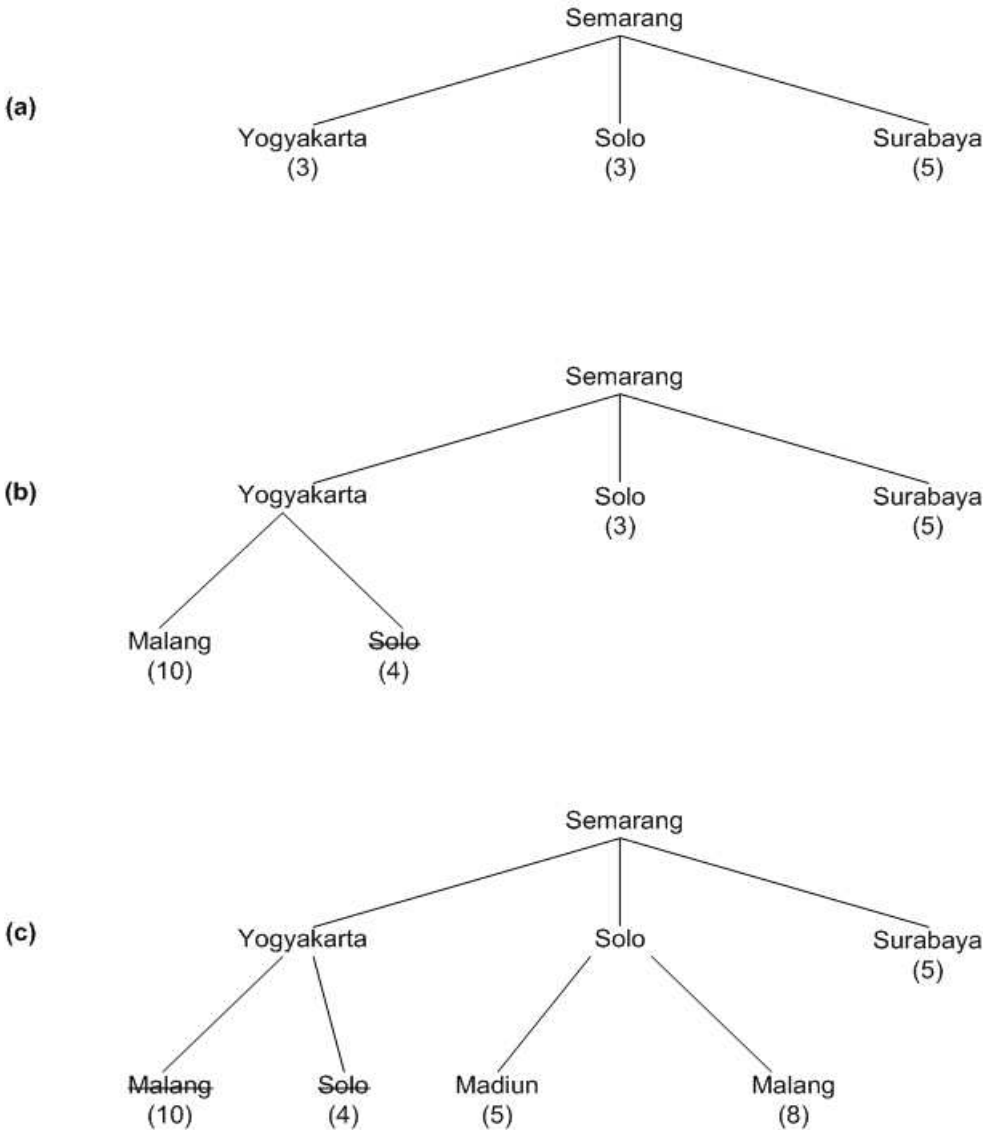

Figure 1.8: Proses searching dengan branch and bound.

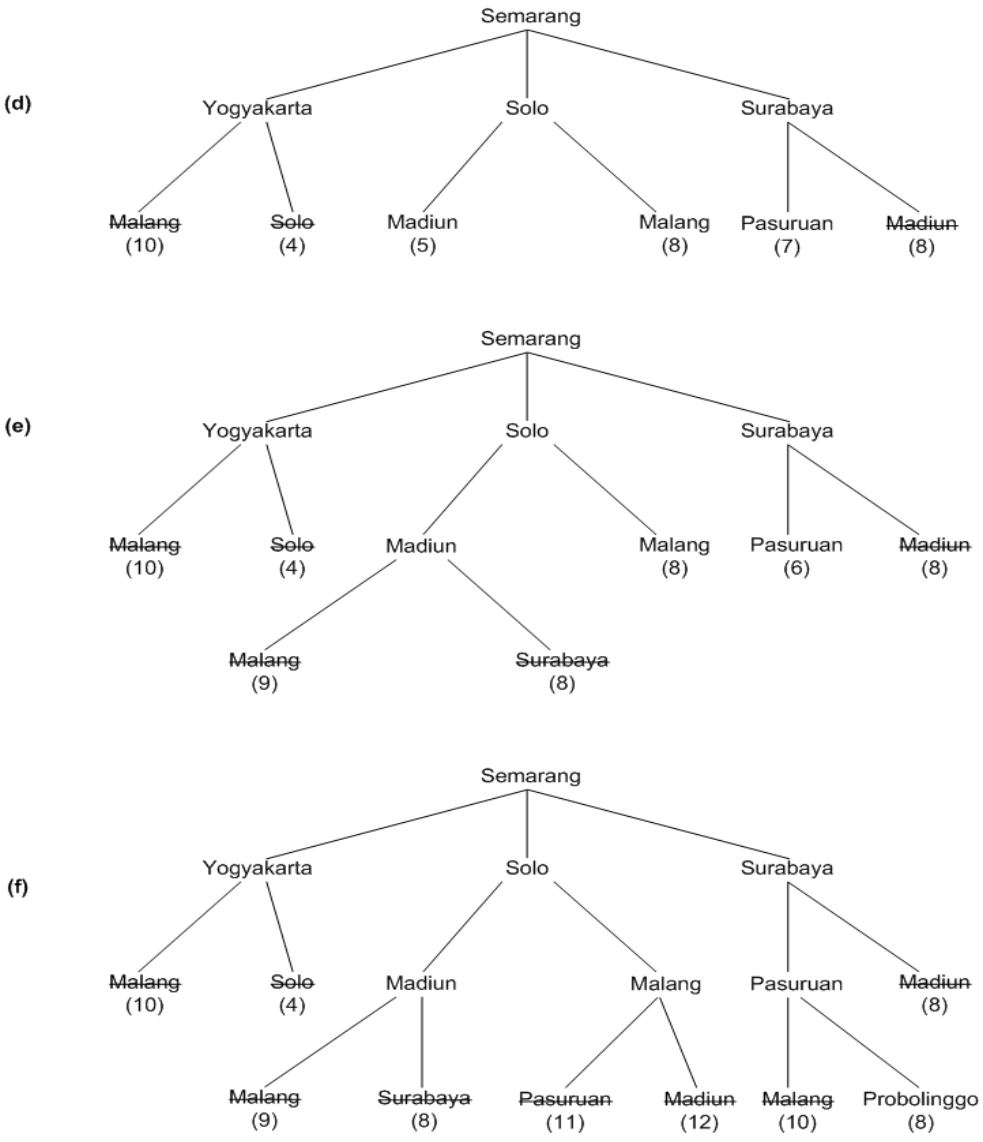

Figure 1.9: Proses searching dengan branch and bound (lanjutan).

1.8.6 A^* Search

A^* Search merupakan gabungan antara best-first dan branch and bound search. Misalkan kita memberikan estimasi setiap node terhadap solusi yang diinginkan. Maka proses searching untuk mencari jarak terpendek dilakukan dengan melakukan komputasi terhadap total estimasi:

> Estimasi total cost = Cost (perjalanan dari Semarang ke node sekarang) + Estimasi cost (perjalanan dari node sekarang ke Probolinggo).

Perhatikan diagram jaringan kota pada Gambar 1.7 yang sudah dilengkapi dengan estimasi setiap kota menuju node tujuan (probilinggo) seperti ditunjukkan dalam tabel di bawah ini:

Table 1.2: Estimasi Jarak

Semarang	8
Yogyakarta	9
Solo	7
Madiun	6
Surabaya	6
Malang	4
Pasuruan	2
Probolinggo	0

Dengan metoda A^*, proses searching ditunjukkan dalam Gambar 1.10.

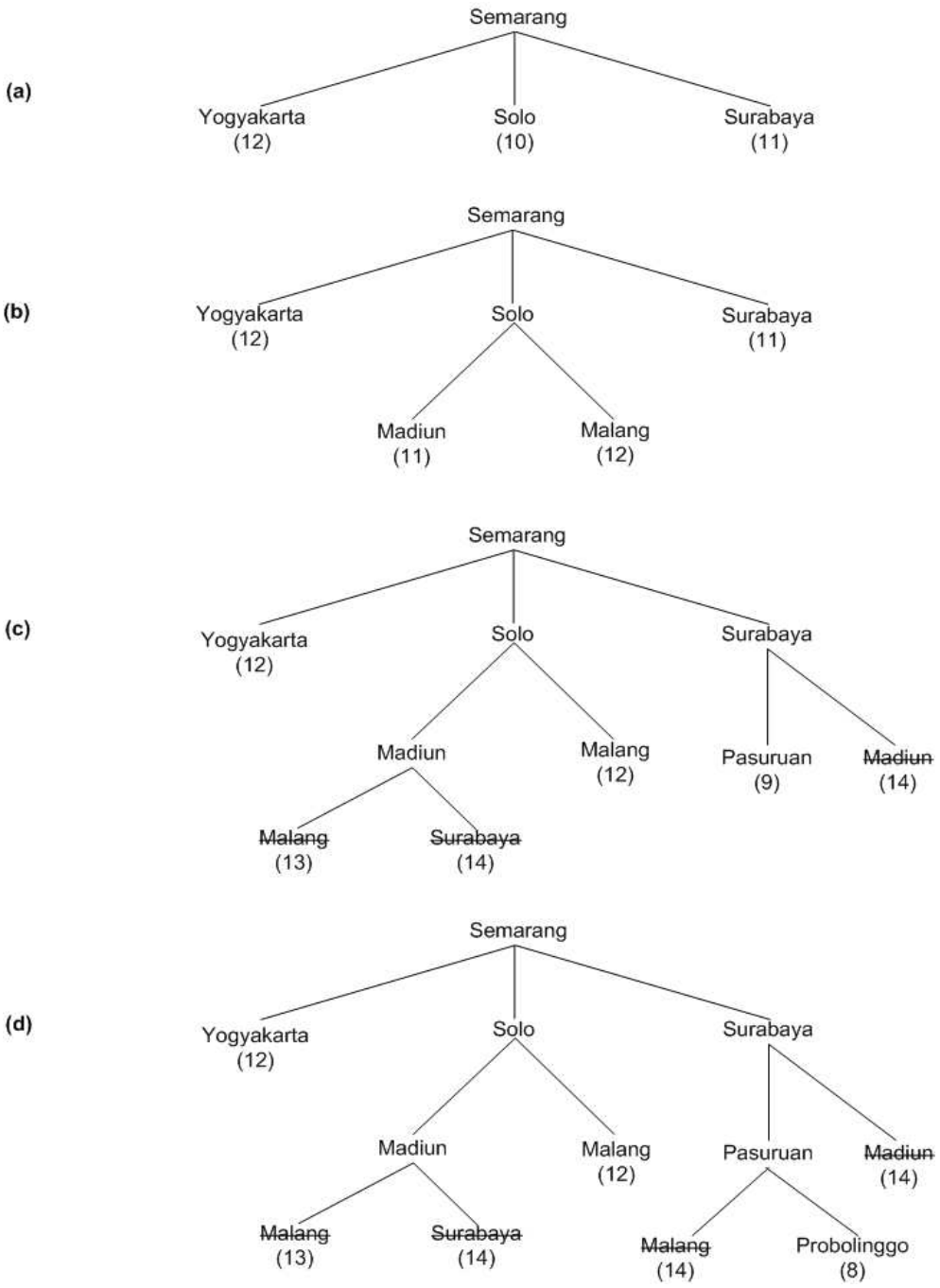

Figure 1.10: Proses searching dengan A^* search.

Chapter 2

Propositional Logic dan Predicate Calculus

Tujuan Instruksional Khusus

- Mahasiswa mampu merepresentasikan knowledge ke dalam bentuk *propositional logic* dan *predicate calculus*.
- Mahasiswa mampu menganalisis permasalahan yang direpresentasikan dalam bentuk *propositional logic* dan *predicate calculus*.

2.1 Propositional Logic

Note:

- *Propositional logic* merupakan salah satu bentuk (bahasa) representasi logika yang paling tua dan paling sederhana.
- Dengan cara ini beberapa fakta dapat digambarkan dan dimanipulasi dengan menggunakan aturan-aturan aljabar Boolean.
- Propositional logic membentuk statement sederhana atau statement yang kompleks dengan menggunakan *propositional connec-*

tive, dimana mekanisme ini menentukan kebenaran dari sebuah statement kompleks dari nilai kebenaran yang direpresentasikan oleh statement lain yang lebih sederhana.

Beberapa operator penghubung dasar yang seringkali dipakai dalam propositional logic ditunjukkan dalam Tabel 2.1 sedangkan tabel kebenaran untuk masing-masing operator dapat dilihat pada Tabel 2.2.

Table 2.1: Operator Penghubung

English Name	Connective Name	Symbol
Conjunction	AND	$\wedge$
Disjunction	OR	$\vee$
Negation	Not	$\sim$
Material Implication	If-Then	$\rightarrow$
Material equivalence	Equals	$\leftrightarrow$

Table 2.2: Tabel Kebenaran

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

Pemahaman antara operator penghubung dan tabel kebenaran dapat dijelaskan dengan menggunakan kalimat sederhana (kecuali operator implikasi material). Misalnya, seseorang sedang memegang dua buah benda, pensil dan penghapus. Lalu orang tersebut mengatakan: "saya sedang memegang *pensil* dan *penghapus*". Maka kita tahu bahwa pernyataan tersebut adalah BENAR (TRUE). Jika kemudian orang tersebut

mengatakan: "saya sedang memegang *pensil* dan *tinta*", maka kita tahu bahwa pernyataan tersebut SALAH (FALSE). Tetapi jika ia mengubah pernyataan menjadi: "saya sedang memegang *pensil* atau *tinta*", maka pernyataan tersebut adalah BENAR (TRUE).

Satu-satunya kaitan antara operator dan tabel kebenaran yang tidak dapat dijelaskan dengan menggunakan kalimat sederhana adalah implikasi material. Tetapi bukan berarti nilai dari tabel kebenaran tidak benar, karena tabel kebenaran implikasi material telah teruji benar dalam aljabar boolean. Simaklah kutipan berikut:

> "Material implication as you and many others have noted elsewhere is not the same as what people are talking about in ordinary speech when they say that one thing is implied by another".

2.1.1 Arti Dari Operator Penghubung

Hubungan variabel dengan operator penghubung dalam propositional logic dapat diartikan seperti dalam Tabel 2.3 di bawah ini.

Contoh 1:
Tentukan bentuk propositional logic dari kalimat ini: *Jika Pluto mengitari matahari, maka Pluto adalah planet, jika tidak demikian maka pluto bukan planet.*

pm . . . Pluto mengitari matahari
pp . . . Pluto adalah planet

Table 2.3: Operator penghubung dan artinya

Operator	Arti
$p \wedge q$	p dan q adalah sahih
	p dan q keduanya sahih
	p dan q adalah sahih pada saat bersamaan
$p \vee q$	p atau q adalah sahih
	p dan/atau q adalah sahih
	paling tidak satu dari p dan p adalah sahih
$p \rightarrow q$	q adalah sahih, jika p sahih
	jika p sahih, demikian juga q adalah sahih
	jika p sahih, maka q juga sahih
	dari p mengikuti q
	p adalah syarat cukup untuk q
	q adalah syarat perlu untuk p
$p \leftrightarrow q$	p sama dengan q
	p benar-benar sahih jika q adalah sahih
	p hanya sahih jika q adalah sahih
	p adalah syarat cukup dan perlu untuk q
	p adalah sahih jika dan hanya jika q sahih

Kalimat di atas dapat ditranslasikan ke dalam bentuk yang lain: *Hanya jika Pluto mengitari matahari, maka Pluto adalah planet.* Sehingga berdasarkan Tabel 2.3, kalimat tersebut dapat diubah ke dalam bentuk propositional logic:

$pm \leftrightarrow pp$

Contoh 2:

Tentukan bentuk propositional logic dari kalimat ini: *If Romeo jatuh*

cinta AND Juliet menerima cintanya, THEN Cupid sedang beraksi[1].

Contoh 3:

Tentukan bentuk *propositional logic* dari kalimat ini: *barangsiapa memahami aturan perkuliahan atau memiliki buku pedoman dan melanggar aturan tersebut dengan sengaja atau tidak akan mendapat hukuman* [2].

2.2 Predicate Calculus

- Kalkulus predikat, disebut juga logika predikat memberi tambahan kemampuan untuk merepresentasikan pengetahuan dengan lebih cermat dan rinci.
- Istilah kalkulus disini berbeda dengan istilah kalkulus dalam bidang matematika.
- Suatu proposisi atau premis dibagi menjadi dua bagian, yaitu ARGUMEN (atau objek) dan PREDIKAT (keterangan).
- Argumen adalah individu atau objek yang membuat keterangan.
- Predikat adalah keterangan yang membuat argumen dan predikat.
- Dalam suatu kalimat, predikat bisa berupa kata kerja atau bagian kata kerja.
- Representasi pengetahuan dengan menggunakan *predicate calculus* merupakan dasar bagi penulisan bahasa pemrograman PROLOG.

[1] r(romeo jatuh cinta),j(juliet menerima cinta),c(cupid sedang beraksi), propositional logic: $r \wedge j \rightarrow c$

[2] map(memahami aturan perkuliahan),mbp(memiliki buku pedoman),ms(melanggar sengaja),mts(melanggar tidak sengaja),mh(mendapat hukuman), propositional logic: $(map \vee mbp) \wedge (ms \vee mts) \rightarrow mh$

Misalnya sebuah proposisi:

Rumput berwarna hijau.

Dapat dinyatakan dalam bentuk *predicate calculus*:

`berwarna(rumput, hijau)`

Seperti terlihat dalam contoh di atas, dengan menggunakan *predicate calculus* statemen/kalimat yang lebih kompleks dapat direpresentasikan lebih baik daripada menggunakan propositional logic.

Beberapa contoh lain:

Proposition : Manusia menjelajah Mars

Predicate calculus : `Jelajah(manusia, mars)`

Proposition : Jono menyukai Rebeca

Predicate calculus : `suka(jono, rebeca)`

Proposition : Rebeca cantik

Predicate calculus : `cantik(rebeca)`

2.2.1 Variabel

- Dalam *predicate calculus* huruf dapat digunakan untuk menggantikan argumen.
- Simbol-simbol juga bisa digunakan untuk merancang beberapa objek atau individu. Contoh: x = Jono, y = Rebeca, maka pernyataan Jono menyukai Rebeca dapat ditulis dalam bentuk *predicate calculus*: `suka(x,y)`.
- Dalam beberapa hal variabel dibutuhkan agar pengetahuan dapat diekspresikan dalam kalkulus predikat sehingga nantinya dapat dimanipulasi dengan mudah dalam proses inferensi.

2.2.2 Fungsi

- *Predicate calculus* memperbolehkan penggunaan simbol untuk mewakili fungsi-fungsi.
  Contoh: `ayah(Jono)=Santoso`, `ibu(Rebeca)=Rini`.

- Fungsi juga dapat digunakan bersamaan dengan predikat.
  Contoh:
  `teman(ayah(Jono),ibu(Rebeca))` = `teman(Santoso,Rini)`

2.2.3 Operator

Predicate calculus menggunakan operator yang sama seperti operator-operator yang berlaku pada *propositional logic*.
Contoh:
Diketahui dua buah statement sebagai berikut:
`suka(Jono,Rebeca)`
`suka(Dani,Rebeca)`

Pada 2 predikat diatas, terdapat dua orang menyukai Rebeca. Untuk memberikan pernyataan adanya kecemburuan di antara mereka, maka:
Jika `suka(x,y)` AND `suka(z,y)`, maka TIDAK `suka(x,z)`.
atau
`suka(x,y)` $\wedge$ `suka(z,y)` $\rightarrow \sim$ `suka(x,z)`

Dalam predicate calculus di atas, pengetahuan yang tersirat adalah : Jika dua orang pria menyukai wanita yang sama, maka kedua pria itu pasti tidak saling suka (saling membenci).

2.3 Quantifier

- Dalam bagian terdahulu, sebuah obyek atau argumen dapat diwakili oleh sebuah variabel, akan tetapi variabel yag telah dibicarakan hanya mewakili sebuah obyek atau individu atau argumen. Bagaimana representasi dapat dilakukan apabila terdapat beberapa obyek? Atau dengan kata lain, bagaimana kuantitas dari sebuh obyek dapat dinyatakan?

- Variabel dapat dikuantitaskan dengan dua cara, yaitu:
  - Ukuran kuantitas universal $\forall$, yang berarti untuk semua.
  - Ukuran kuantitas eksistensial $\exists$, yang berarti ada beberapa.

Contoh 1:

Proposisi: *Semua planet tata-surya mengelilingi matahari.*

Dapat diekspresikan ke dalam bentuk:

$$\forall \texttt{X}, [\texttt{planet-tata-surya(X)} \rightarrow \texttt{mengelilingi(X,matahari)}].$$

Contoh 2:

Proposisi: *Asteroid mengelilingi beberapa planet.*

Dapat diekspresikan ke dalam bentuk:

$$\exists \texttt{Y}, [\texttt{planet(Y)} \wedge \texttt{mengelilingi(asteroid,Y)}].$$

Contoh 3:

Proposisi: *Jika rata-rata nilai dari mahasiswa lebih besar dari 80%, maka mahasiswa akan mendapat nilai huruf A.*

Dapat diekspresikan ke dalam bentuk:

$$\forall \texttt{Nama}, \forall \texttt{X}, [\texttt{mahasiswa(Nama)} \wedge \texttt{rata-nilai(Nama,X)} \wedge \texttt{mendapat(X,80)} \rightarrow \texttt{nilai-huruf(Nama, "A")}].$$

Latihan:

Rubahlah proposisi di bawah ini ke dalam bentuk *predicate calculus*:

- Garfield adalah seekor kucing.
- Garfield adalah tokoh kartun.
- Semua kucing adalah binatang.
- Setiap orang menyukai seseorang.
- Semua kucing menyukai atau membenci anjing.
- Seseorang hanya mencoba melukai seseorang yang mereka tidak suka.
- Garfield mencoba melukai anjing Rover.

2.4 Model-Model Inferensi

Note:

Inti dasar dari *predicate calculus* sebenarnya adalah kemampuan untuk melakukan inferensi logis. Pada proses inferensi kebenaran baru dapat diturunkan dari aksioma-aksioma yang sudah ada. Konsep ini sebenarnya merupakan dasar dari sistem berbasis pengetahuan yang akan kita bicarakan pada Bab selanjutnya. Terdapat beberapa model inferensi yang secara umum digunakan dalam persoalan-persoalan logika, antara lain:

Modus Ponens

Seperti dijelaskan di atas, melakukan proses inferensi berarti juga menurunkan fakta baru dari beberapa fakta yang sudah ada. Modus Ponens melakukan inferensi dengan mengikuti aturan sebagai berikut:

> Jika pernyataan p dan $(p \rightarrow q)$ adalah benar,
> maka dapat ditarik kesimpulan bahwa q adalah benar.

Modus Ponens merupakan dasar bagi sistem berbasis aturan (rule-based system). Sebagai contoh perhatikan pernyataan di bawah ini:

`Jika seseorang rajin belajar maka ia bisa menjadi sarjana`

Jika representasikan dalam bentuk *predicate calculus*, menjadi:

$\forall$ X, [`rajin-belajar(X)` $\rightarrow$ `jadi-sarjana(X)`]

Apabila sebuah fakta (pernyataan) ditemukan dalam database seperti:

`rajin-belajar(alex)`

maka melalui Modus Ponens, dapat ditarik kesimpulan:

`jadi-sarjana(alex)`

Modus Tolens

Model inferensi yang lain disebut sebagai Modus Tolens yang dinyatakan dengan rumusan:

> Jika $(p \rightarrow q)$ adalah benar,
> dan q tidak benar, maka p tidak benar.

Sebagai contoh, dengan menggunakan pernyataan pada contoh terdahulu ditemukan sebuah fakta sebagai berikut:

$\sim$ `jadi-sarjana(alex)`

maka dengan menggunakan Modus Tolens dapat ditarik kesimpulan:

$\sim$ `rajin-belajar(alex)`

2.5 Automated Reasoning

Ada tiga macam metoda reasoning yang secara umum digunakan yaitu: Deduksi (Deduction), Abduksi (Abduction) dan Induksi (Induction).

Deduksi

Deduksi didefinisikan sebagai: reasoning dari fakta yang sudah diketahui menuju fakta yang belum diketahui, dari hal-hal umum menuju ke hal-hal spesifik, dari premis menuju ke kesimpulan logis.

> Jika obyek A lebih besar dari beberapa obyek B
> dan obyek B lebih besar daripada obyek C,
> maka obyek A lebih besar daripada obyek C.

Definisi di atas dapat dinyatakan dalam bentuk *predicate calculus* sebagai:

$$\forall \text{ A}, \forall \text{ B}, \forall \text{ C}, [\texttt{lebih-besar(A,B)} \wedge \texttt{lebih-besar(B,C)} \rightarrow \texttt{lebih-besar(A,C)}]$$

Misalkan di dalam sebuah knowledge-based ditemukan fakta-fakta sebagai berikut:

```
lebih-besar(bumi,merkurius)
lebih-besar(yupiter,bumi)
```

maka dengan menggunakan reasoning deduktif dapat ditarik suatu kesimpulan:

```
lebih-besar(yupiter,merkurius)
```

Abduksi

Abduksi adalah metoda reasoning yang sering dipakai untuk memberikan/menghasilkan penjelasan terhadap fakta. Berbeda dengan metoda deduksi, pada metoda

ini tidak ada jaminan bahwa kesimpulan yang didapat selalu benar. Sebagai contoh, sebuah aturan seperti pada contoh terdahulu dituliskan sebagai berikut:

$$\forall \text{ X, } [\texttt{rajin-belajar(X)} \rightarrow \texttt{jadi-sarjana(X)}]$$

Misalkan didapati bahwa Alex telah diwisuda, maka bentuk *predicate calculus* nya adalah:

```
jadi-sarjana(alex)
```

Dengan menggunakan abduksi dapat disimpulkan bahwa:

```
rajin-belajar(alex)
```

Tetapi tidak ada jaminan bahwa kesimpulan tersebut benar. Menjadi sarjana tidak selalu berarti rajin belajar.

Induksi

Induksi berarti proses reasoning dari fakta-fakta khusus atau kasus-kasus individual menuju ke kesimpulan secara general. Sebagai contoh:

```
P(a) adalah benar
P(b) adalah benar
```

maka dengan induksi dapat disimpulkan bahwa:

$$\forall \texttt{ X, P(X) adalah benar}$$

Kembali pada contoh terdahulu, misalnya setelah melakukan observasi berulang-berulang ternyata kita menemukan bahwa hanya mahasiswa yang belajar dengan rajin menjadi sarjana, maka dengan induksi dapat ditarik kesimpulan bahwa:

$$\forall \text{ X, } [\texttt{jadi-sarjana(X)} \rightarrow \texttt{rajin-belajar(X)}]$$

2.6 Soal-Soal Latihan

1. Diberikan fakta-fakta dari PROLOG sebagai berikut :

   job(smith,clerk)

   job(dell, stock-person)

   job(jones, clerk)

   job(putnam, assistant-manager)

   job(fishback, clerk)

   job(adams, stock-person)

   job(philips, manager)

   job(stevens, vice-president)

   job(johnson, president)

   boss(clerk, assistant-manager)

   boss(stock-person, assistant-manager)

   boss(assistant-manager, manager)

   boss(manager, vice-president)

   boss(vice-president, president)

   Tentukan respons yang diberikan oleh PROLOG terhadap Query berikut

   (a) ?− job(philips, X), boss(X, Y), job(Z, Y).

   (b) ?− boss(stock-person, X); boss(clerk, X).

   (c) Jawab kembali kedua pertanyaan di atas apabila ditambahkan Rule : Boss(X, Z) :- boss(X, Y), boss(Y, Z)

   Note: dalam PROLOG operator AND (konjungsi) direpresentasikan dengan tanda komma (,), sedangkan operator OR (disjungsi) direpresentasikan dengan tanda titik-koma (;). Sementara itu tanda := merupakan representasi dari implikasi material dan tanda $\sim$ adalah

representasi dari negasi.

2. Rubahlah fakta-fakta di bawah ini ke dalam bentuk predicate calculus menggunakan hubungan: `meninggal(X)`, `sex(X,Y)`, `mengenal(X,Y)`, `membenci(X,Y)`, `korban(X)`, `pembunuh(X)`. Lalu dengan menggunakan metoda inferensi tentukan siapa pembunuh dalam kasus ini.

   - Korban meninggal.
   - Korban adalah perempuan.
   - Jono dan Suryo mengenal korban.
   - Korban mengenal Toni dan Jono.
   - Si pembunuh mengenal korban.
   - Susi adalah korban.
   - Jono membenci Susi.
   - Suryo membenci Toni.
   - Toni membenci Jono.
   - Korban mengenal seseorang yang membenci pembunuh tersebut.

Chapter 3

Sistem Berbasis Aturan

Tujuan Instruksional Khusus

- Mahasiswa mampu menjelaskan representasi pengetahuan dalam bentuk Sistem Berbasis Aturan (Ruled-based System).
- Mahasiswa memahami model inferensi *forward chaining* dan *backward chaining* dalam sistem berbasis aturan.

3.1 Pendahuluan

Sistem berbasis aturan (rule-based system) menggunakan Modus Ponens sebagai dasar untuk memanipulasi aturan, yaitu:

`fakta A benar, dan`
`operasi A` $\rightarrow$ `B benar,`
`maka fakta B adalah benar`

Dengan menggunakan teknik searching yang telah didefinisikan dalam

Bab 1 dan penyesuaian pola (pattern matching), sistem berbasis aturan melakukan proses reasoning mulai dari fakta awal sampai menuju pada kesimpulan. Dalam proses ini mungkin akan dihasilkan fakta-fakta baru menuju pada penyelesaian masalah. Jadi dapat disimpulkan bahwa proses penyelesaian masalah pada sistem berbasis aturan adalah menciptakan sederet fakta-fakta baru yang merupakan hasil dari sederetan proses inferensi sehingga membentuk semacam jalur antara definisi masalah menuju pada solusi masalah. Deretan proses inferensi tersebut adalah *inference chain*.

Sebagai contoh, sebuah sistem peramal cuaca dibangun dengan sistem berbasis pengetahuan untuk mengetahui keadaan cuaca pada 12 sampai 24 jam ke depan.

```
RULE 1:  IF    suhu udara sekitar di atas 32^o C
         THEN  cuaca adalah panas
RULE 2:  IF    kelembaban udara relatif di atas 65%
         THEN  udara sangat lembab
RULE 3:  IF    cuaca panas dan udara sangat lembab
         THEN  sangat mungkin terjadi badai
```

- Jika hanya rule 1 (tanpa rule 2 dan rule 3), sistem berbasis pengetahuan tidak berarti apa2.
- Karena itu sebuah sistem berbasis pengetahuan harus terdiri atas sekelompok aturan yang membentuk rangakaian aturan *rule chain*.
- *Fakta* didefisinikan sebagai statemen yang dianggap benar. Contoh: `Suhu udara di sekitar adalah` 35^o `C` dan `kelembaban udara relatif 70%` adalah fakta.
- Maka proses inferensi melihat fakta-fakta dari premis pada Rule 1 dan Rule 2 sebagai dasar untuk menghasilkan fakta baru: `Cuaca panas` dan `Udara lembab`.

- Selanjutnya proses inferensi melihat bahwa kedua fakta ini sesuai dengan premis pada Rule 3, maka akan dihasilkan fakta baru lagi: `Sangat mungkin terjadi badai`.

3.2 Proses Reasoning

Note:

- Proses reasoning dari sebuah sistem berbasis aturan adalah tahapan proses mulai dari sekumpulan fakta menuju solusi, jawaban dan kesimpulan.

- Terdapat dua macam cara yang dapat digunakan untuk menghasilkan suatu kesimpulan, yaitu:
  - *Forward Chaining (data driven):* kesimpulan dihasilkan dari seperangkat data yang diketahui.
  - *Backward Chaining (goal driven):* memilih beberapa kesimpulan yang mungkin dan mencoba membuktikan kesimpulan tersebut dari bukti-bukti yang ada.

Sebagai contoh, seperangkat rule di bawah ini akan digunakan untuk menjelaskan proses forward chaining dan backward chaining.

Kasus 1: Identifikasi Binatang

```
Rule 1:  If    Menyusui = tidak and
               Kaki > 4 and
               Warna bulu = polos
         Then  Binatang = belalang

Rule 2:  If    Menyusui = ya and
               kaki = 4 and
               berdarah = panas
         Then  Kelas = mamalia

Rule 3:  If    Makanan = binatang
         Then  Kategori = carnivora

Rule 4:  If    Makanan = tumbuhan
         Then  Kategori = herbivora

Rule 5:  If    Kelas = mamalia and
               Warna bulu = berbintik and
               Tinggi > 80 centimeter
         Then  Binatang = Harimau

Rule 6:  If    Kelas = mamalia and
               Warna bulu = polos and
               Tinggi > 80 centimeter
         Then  Binatang = Kuda
```

```
Rule 7:   If    Kelas = mamalia and
                Kategori = herbivora and
                Warna bulu = bergaris and
                Tinggi > 80 centimeter
          Then  Binatang = Zebra

Rule 8:   If    Kelas = mamalia and
                Kategori = carnivora and
                Warna bulu = polos or berbintik or bergaris and
                Tinggi < 80 centimeter
          Then  Binatang = Kucing

Rule 9:   If    Kaki = 2 and
                Berdarah = panas and
                Warna Bulu = berbintik or
                Menyusui = tidak
          Then  Binatang = elang

Rule 10:  If    Berdarah = dingin and
                Tinggi < 80 centimeter
          Then  Binatang = salmon
```

Pengetahuan yang telah didefinisikan di atas dapat juga direpresentasikan dalam bentuk diagram pohon *and-or tree*, seperti di tunjukkan dalam Gambar 3.1. Data yang diberikan oleh user diletakkan pada bagian kiri gambar, sedangkan kesimpulan berada pada bagian kanan gambar. Beberapa data dihubungkan dengan kesimpulan memiliki tanda penghubung lengkung mewakili hubungan AND, sedangkan tanpa penghubung lengkung mewakili OR. Sedangkan, *intermediate facts* atau parameter

diletakkan di tengah-tengah di antara data dan kesimpulan.

Sekarang muncul pertanyaan, dari fakta-fakta yang telah tersedia di atas bagaimana sistem pakar melakukan proses *tracing* dari satu fakta ke fakta berikutnya dengan menggunakan rule yang telah ditentukan sehingga menghasilkan suatu kesimpulan? Proses ini dikenal dengan istilah *reasoning*. Dalam sistem berbasis aturan dikenal adanya dua macam proses reasoning, yaitu *forward reasoning* dan *backward reasoning*. Sekarang marilah kita bicarakan masing-masing proses reasoning tersebut dengan lebih detail, diawali dengan pembahasan tentang *forward reasoning*.

3.2.1 Forward Reasoning

Note:

Dalam forward reasoning, proses inferensi dimulai dari seperangkat data yang ada menuju ke kesimpulan. Pada proses ini akan dilakukan pengecekan terhadap setiap rule untuk melihat apakah data yang sedang diobservasi tersebut memenuhi premis dari rule tersebut. Apabila memenuhi, maka rule akan dieksekusi untuk menghasilkan fakta baru yang mungkin akan digunakan oleh rule yang lain. Proses pengecekan rule ini disebut sebagai *rule interpretation*. Pada sistem berbasis pengetahuan, rule interpretation (interpretasi rule) dilakukan oleh inference engine. Proses interpretasi rule ini merupakan proses berulang seperti terlihat dalam Gambar 3.2

Fungsi masing-masing step untuk gambar di atas dijelaskan sebagai berikut:

1. *Matching.* Pada step ini, setiap rule yang ada pada basis pengetahuan dibandingkan dengan fakta-fakta yang diketahui untuk mencari rule mana yang memenuhi (istilah 'memenuhi' berarti: situasi, premis, atau antecedent bernilai benar).

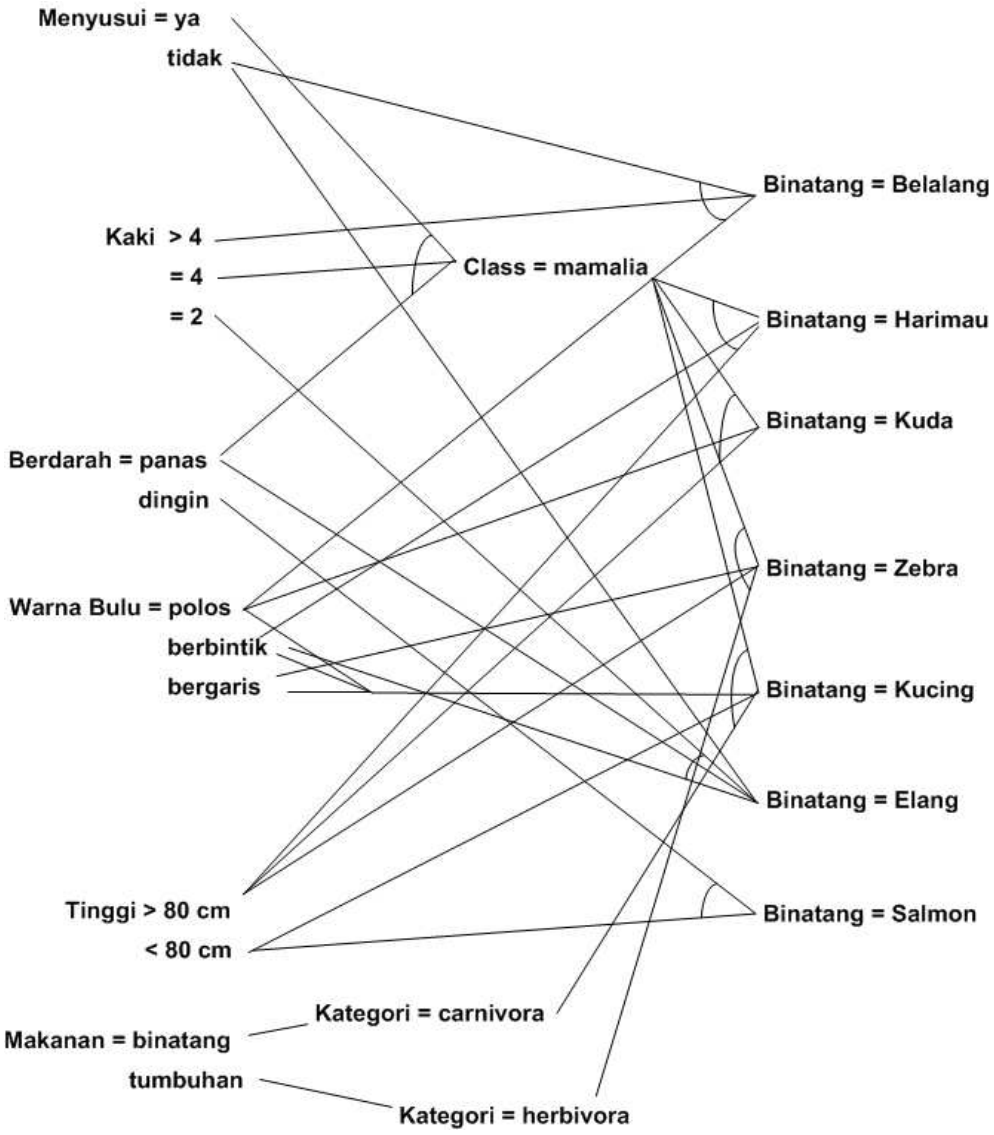

Figure 3.1: Representasi *and-or tree* untuk identifikasi binatang.

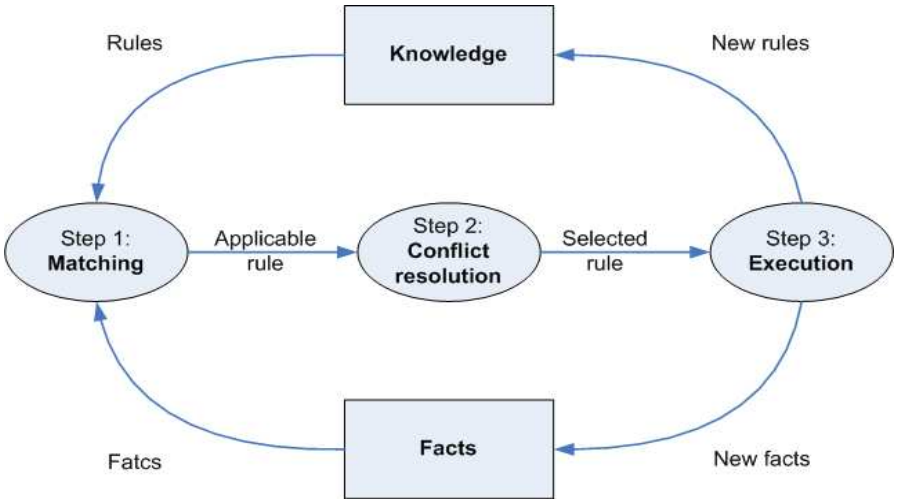

Figure 3.2: Proses inferensi forward reasoning.

2. *Conflict Resolution.* Pada langkah pertama sangat mungkin dihasilkan suatu kondisi dimana beberapa rule dipenuhi. Conflict Resolution bertugas untuk mencari rule mana yang memiliki prioritas tertinggi yang berpotensi untuk dieksekusi.

3. *Execution.* Langkah terakhir dari proses *forward reasoning* adalah eksekusi (firing) dari rule. Proses ini menghasilkan dua kemungkinan, yaitu: fakta baru diturunkan dan ditambahkah *fact base* atau rule baru dihasilkan dan ditambahkan ke *knowledge base*.

Sebagai contoh penyelesaian Kasus 1 di atas dengan menggunakan forward reasoning dimulai dengan memasukkan fakta-fakta berikut:

Database:

`Menyusui = ya`

`Jumlah kaki = 4`

`Berdarah = panas`

`Warna bulu = berbintik`

`Tinggi = 20 cm`

`Makanan = binatang`

Hasil penelusuran dari proses eksekusi ditunjukkan dalam Tabel 3.1.

Table 3.1: Hasil Penelusuran Dengan Forward Reasoning

Execution cycle	Applicable rules	Selected rule	Derived fact
1	2, 3	2	Kelas = mamalia
2	2, 3	3	Kategori = carnivora
3	2, 3, 7	7	Binatang = kucing
4	2, 3, 7	−−	−−

Sebagai latihan, jika fakta-fakta baru dimasukkan seperti dibawah ini, tentukan bagaimana hasil penelusuran sistem pakar dengan menggunakan forward reasoning!

Database:

```
Menyusui = ya
Jumlah kaki = 4
Berdarah = panas
Warna bulu = polos
Tinggi = 2,2 meter
```

Kasus 2: Flood Identification

Untuk memberikan gambaran tentang penerapan sistem berbasis aturan secara nyata, berikut ini adalah sebuah contoh tentang penggunaan forward reasoning dalam membantu menyelesaikan permasalahan banjir seperti ditulis oleh Gonzales (1993).

Penduduk dari sebuah daerah bernama Low Lands di Florida, US, menemukan bahwa desa dimana mereka berharap dapat menghabiskan masa tuanya berada di daerah banjir luapan dari sungai Suwannee. Banjir selalu terjadi pada setiap musim semi. Karena itu mereka mengundang seorang pakar Sistem Pakar bernama Dr. Paul Peters untuk membangun sebuah sistem berbasis pengetahuan untuk memberitahu mereka mengetahui apabila banjir akan datang dan apakah perlu dilakukan evakuasi.

Dr. Peters menemukan bahwa diperlukan 10 buah paramater dan 18 rule untuk menyelesaikan masalah ini. Paramater dan rule tersebut ditunjukkan sebagai berikut:

Paramaters	**Possible Values**
month	any month of the year
upstream precipitation	none, light, heavy
weather forecast	sunny, cloudy, stormy
river height	measurement in feet
season	dry, wet
local rain	none, light rain, heavy rain
river change	none, lower, higher
river level	low, normal, high
flood warning	yes, no
evacuation order	yes, no

```
R1  If    month = may ··· oct
    Then  season = wet

R2  If    month = nov ··· apr
    Then  season = dry

R3  If    upstream = none AND
          season = dry
    Then  change = lower

R4  If    upstream = none AND
          season = wet
    Then  change = none

R5  If    upstream = light
    Then  change = none

R6  If    upstream = heavy
    Then  change = higher

R7  If    level = low
    Then  flood = no AND evac = no

R8  If    change = none | lower AND
          level = normal | lower
    Then  flood = no AND evac = no

R9  If    change = higher AND
          level = normal
          rain = heavy
    Then  flood = yes (CF = 0.4) AND evac = no
```

```
R10  If    change = higher AND
           level = normal
           rain = light
     Then  flood = no AND evac = no

R11  If    change = higher AND
           level = high
           rain = none | light
     Then  flood = yes (CF = 0.5) AND
           evac = yes (CF = 0.2)

R12  If    change = higher AND
           level = high
           rain = heavy
     Then  flood = yes AND
           evac = yes (CF = 0.8)

R13  If    height < 10
     Then  level = low

R14  If    height ≥ 10 AND ≤ 16
     Then  level = normal

R15  If    height > 16
     Then  level = high

R16  If    forecast = sunny
     Then  rain = none
```

```
R17  If    forecast = cloudy
     Then  rain = light

R18  If    forecast = stormy
     Then  rain = heavy
```

Sistem berbasis aturan di atas dapat digambarkan dalam bentuk *inference network* seperti terlihat dalam Gambar 3.3.

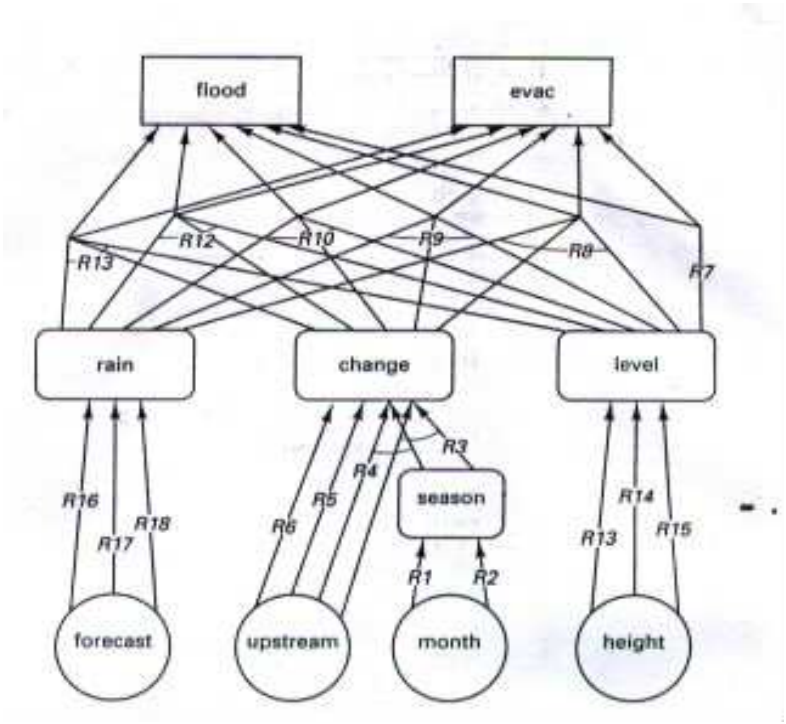

Figure 3.3: Inference network dari kasus 2.

Lebih lanjut Dr. Peter melakukan identifikasi untuk setiap rule sebagai berikut:

1. Memiliki nomor rule.
2. Memiliki parameter yang digunakan dalam premise (upstream element).
3. Memiliki parameter yang digunakan dalam kesimpulan (downstream

element).

4. Memiliki seperangkat premise.

5. Memiliki seperangkat kesimpulan.

Dan identifikasi untuk setiap paramater:

1. Nama dari parameter tersebut.

2. Seperangkat rule yang menurunkan nilai untuk parameter ini (disebut set-by).

3. Seperangkat rule yang menggunakan parameter ini untuk bagian premise (disebut premise-for).

4. Nilai dari parameter ini yang berasosiasi dengan confidence fator (CF).

Sebagai contoh rule R1 dan parameter *change* dideskripsikan sebagai:

Rule:	R1
Upstream-elements:	month
Downstream-elements:	season
Premises:	month = may $\cdots$ oct
Conclusions:	season = wet
Parameter:	change
Set-by:	(R3 R4 R5 R6)
Premise-for:	(R8 R9 R10 R11 R12)
Values:	$--$

Berdasarkan Gambar 3.3, dapat dilihat bahwa terdapat beberapa parameter sebagai input data dan beberapa parameter untuk menarik kesimpulan, yaitu:

```
Input data:   (month upstream forecast height)
Conclusions:  (flood evac)
```

Misalkan, dari beberapa sensor yang telah terpasang diketahui data sebagai berikut:

```
month = may
upstream = light
forecast = cloudy
height = 15
```

Dengan memadankan fakta yang ada dengan rule-based dan Gambar 3.3, antrian rule yang akan diproses adalah sbb:

```
Q = (R1 R2 R3 R4 R5 R6 R13 14 R15 R16 R17 R18)
```

Rule pertama yang harus diuji adalah premise dari R1, apakah match dengan salah satu fakta. Seperti terlihat dalam R1, nilai `may` dan parameter `month` terpenuhi, maka R1 dieksekusi dan menghasilkan `season=wet`. Parameter `season` terpakai dalam premise R3 dan R4, tapi karena kesua rule tersebut sudah ada di dalam antrian Q, maka R3 dan R4 tidak perlu ditambahkan dalam Q. Selanjutnya R1 dikeluarkan dari Q dan pengujian dilanjutkan menuju R2. R2, R3 dan R4 tidak memenuhi, maka ketiga rule tersebut dikeluarkan dari Q, pengujian dilanjutkan ke R5. R5 memenuhi dan menghasilkan `change=none`. Nilai dari dari parameter change digunakan oleh R8, R9, R10, R11, R12 yang tidak termasuk dalam, maka update terhadap Q menghasilkan:

```
Q = (R6 R13 14 R15 R16 R17 R18 R8 R9 R10 R11 12)
```

dan diketahui fakta-fakta:

```
month = may
upstream = light
forecast = cloudy
height = 15
season = wet
change = none
```

Proses ini dilanjutkan sampai Q menjadi kosong. Setelah Q kosong, diketahui fakta-fakta sebagai berikut:

```
month = may
upstream = light
forecast = cloudy
height = 15
season = wet
change = none
level = normal
rain = light
flood = no
evac = no
```

3.2.2 Backward Reasoning

Note:

Sekarang mari kita beralih dari forward reasoning ke backward reasoning. Mekanisme inferensi pada backward reasoning berbeda dengan forward reasoning. Walaupun kedua proses melibatkan pengujian terhadap masing-masing rule, backward reasoning mulai dari konklusi yang diharapkan menuju fakta-fakta yang mendukung konklusi tersebut. Misalnya, pada saat awal fakta yang diketahui dalam database kosong, seperti:

Fakta yang diketahui: ( )

Dengan mengambil contoh pada Kasus 1: Identifikasi Binatang, didapatkan bahwa goal yang diharapkan adalah:

Goal: `(Binatang)`

Dengan melihat semua rule yang ada pada knowledge-based didapatkan bahwa rule yang dapat menurunkan goal tersebut adalah: 1,5,6,7,8,9,dan

10. Eksekusi dimulai dari rule 1. Premise pertama dari rule tersebut `(menyusui=tidak)` diuji dan ternyata tidak ada nilai `menyusui` ditemukan dalam database. Karena tidak ada rule yang dapat menurunkan nilai dari premise `menyusui`, maka inference engine bertanya kepada user:

`Apakah termasuk binatang menyusui?`

Selanjutnya user memberi respon dengan menambahkan fakta pada database:

Fakta yang diketahui:

`( (Menyusui = ya) )`

Sampai disini, evaluasi menunjukkan bahwa premise pada rule 1 tidak sesuai dengan fakta pada database, karena itu rule 1 tidak dapat diproses. Evaluasi dilanjutkan menuju rule 5. Premise pertama adalah `(Kelas=mamalia)`. Proses pencarian terhadap rule menemukan bahwa rule 2 mampu menurunkan nilai untuk `kelas`, karena itu untuk sementara proses pencarian untuk solusi `binatang` dihentikan dan menambahkan parameter dibawah ini pada stack:

Goal: `(kelas binatang)`

Sekarang evaluasi dilanjutkan untuk rule 2. Premise pertama pada rule 2 adalah `menyusui`, karena `menyusui` ada di dalam database dengan nilai `ya`, pengujian dilanjutkan ke premise berikutnya. Nilai `kaki` tidak ada dalam database, maka *inference engine* bertanya kepada user:

`Berapa jumlah kaki binatang tersebut?`

User memberikan response dengan nilai 4, maka sekarang update terhadap database menghasilkan:

Fakta yang diketahui:

`( (Menyusui = ya) )`

`( (Kaki = 4) )`

Premise ketiga dari rule 2 menunjuk pada sebuah fakta baru lagi yang berkaitan dengan suhu badan dari binatang. Dalam istilah biologi, berdasarkan suhu badannya binatang dibedakan atas binatang berdarah panas dan binatang berdarah dingin. Karena fakta tentang suhu badan ini belum ada dalam database, maka *inference engine* akan bertanya kepada user.

```
Binatang tersebut berdarah panas atau dingin?
```

User memberikan response dengan nilai panas, maka update terhadap database menghasilkan kumpulan fakta-fakta baru sebagai berikut:

Fakta yang diketahui:

```
( (Menyusui = ya) )
( (Kaki = 4) )
( (Berdarah = panas) )
```

Karena semua fakta pada premise dari rule 2 telah terpenuhi, maka sekarang rule 2 firing dan memberikan keluaran (goal) dengan nilai mamalia. Sehingga fakta baru lagi dimasukkan ke dalam database menjadi:

Fakta yang diketahui:

```
( (Menyusui = ya) )
( (Kaki = 4) )
( (Berdarah = panas) )
( (Kelas = mamalia) )
```

Setelah goal kelas memperoleh jawaban, sekarang sistem kembali ke rule 5 dengan goal: binatang.

Premise pertama pada rule 5 berpadanan dengan kelas=mamalia seperti terdapat dalam database. Evaluasi dilanjutkan ke premise berikutnya. Fakta dari premise ini ternyata tidak ditemukan dalam database, sehingga *inference engine* akan bertanya kepada user:

```
Bagaimana warna bulu dari binatang tersebut?
```

User menjawab dengan nilai berintik. Sehingga fakta baru lagi ditambahkan pada database menjadi seperti berikut:

Fakta yang diketahui:

```
( (Menyusui = ya) )
( (Kaki = 4) )
( (Berdarah = panas) )
( (Kelas = mamalia) )
( (Warna bulu = berbintik) )
```

Masih pada rule 5, premise terakhir diuji. Fakta dari premise ini ternyata tidak ditemukan dalam database, sehingga *inference engine* akan bertanya kepada user:

```
Berapa tinggi binatang tersebut?
```

Maka user meresponse dengan nilai tinggi = 20 cm. Fakta ini selanjutnya dimasukkan kembali ke dalam database menghasilkan daftar fakta sebagai berikut:

Fakta yang diketahui:

```
( (Menyusui = ya) )
( (Kaki = 4) )
( (Berdarah = panas) )
( (Kelas = mamalia) )
( (Warna bulu = berbintik) )
( (Tinggi = 20 cm) )
```

Karena fakta ini tidak memenuhi premise terakhir dari rule 5, maka rule 5 tidak dikerjakan. Berlanjut ke rule 6. Seperti terlihat bahwa premise kedua dari rule 6 tidak memenuhi fakta warna bulu yang ada dalam database, maka rule 6 tidak dikerjakan. Berlanjut ke rule 7.

Premise pertama pada rule 7 terpenuhi. Maka evaluasi dilanjutkan ke premise berikutnya. Premise ini memunculkan fakta baru: kategori

yang tidak ada dalam database. Dari penelusuran yang dilakukan, sistem mendapati bahwa fakta ini dapat dihasilkan oleh rule 3 dan 4. Untuk sementara goal binatang digeser masuk ke dalam stack, goal yang baru adalah:

Goal: `(kategori binatang)`

Evaluasi terhadap rule 3 menghasilkan fakta baru: makanan. Karena makanan tidak ada dalam database, sistem akan bertanya kepada user:

`Apakah makanan dari binatang tersebut?`

User menjawab dengan nilai makanan adalah binatang. Sehingga rule 3 firing dan menghasilkan nilai keluaran dari goal adalah carnivora. Sedangkan isi dari database sekarang diupdate menjadi seperti berikut:

Fakta yang diketahui:

```
( (Menyusui = ya) )
( (Kaki = 4) )
( (Berdarah = panas) )
( (Kelas = mamalia) )
( (Warna bulu = berbintik) )
( (Tinggi = 20 cm) )
( (Kategori = carnivora) )
```

Karena goal kategori sudah terjawab, goal ini dibuang dari stack. Proses dilanjutkan kembali ke goal mula-mula. Seperti terlihat, premise kedua dari rule 7 tidak terpenuhi. Maka rule 7 tidak dikerjakan. Dilanjutkan dengan rule 8. Nampaknya semua premise terpenuhi oleh fakta-fakta yang ada dalam database. Menghasilkan keluaran binatang yang dimaksud adalah kucing.

Karena `binatang` telah diketahui, maka `binatang` dikeluarkan dari goal. Karena itu stack sekarang menjadi kosong, proses backward reasoning berhenti sampai di sini. Secara keseluruhan urutan proses backward

reasoning dari identifikasi binatang ini dapat digambarkan dalam Gambar 3.4.

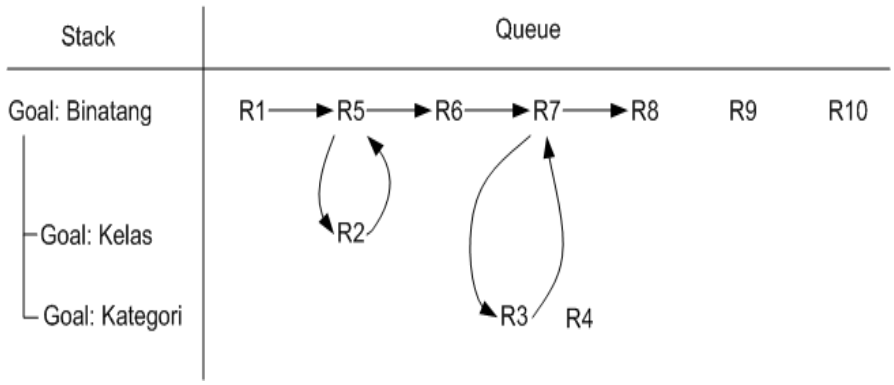

Figure 3.4: Urutan backward reasoning untuk kasus identifikasi binatang.

Kasus 2: Beverage and Main Course Identification

Sebagai contoh kasus ke-2 untuk backward reasoning, mari kita perhatikan Beverage and Main Course Identification. Contoh ini diambil dari Gonzales(1993).

Perhatikan rule di bawah ini:

Parameter-parameter yang digunakan adalah:

guest-age: positive integer between 15 and 100

alcohol-indicated: yes/no

meal: formal/informal

drink: wine/beer/soda

guest: boss/neighbor/friend

dinner: fish/veal/read-meat/pultry/pizza

day: moday/tuesday/ . . . /sunday

Sistem dimulai dengan:

Known Fact Base: ()

Misalnya goal yang diharapkan adalah:

Goals: `(drink wine-type dinner)`

Tentukan hasil akhir dari database jika saat ini adalah Tuesday, kita mengundang neighbor berumur 30 th untuk casual meal.

R1	If	guest-age < 21
	Then	alcohol-indecated = no
R2	If	guest-age $\geq$ 21
	Then	alcohol-indecated = yes
R3	If	alcohol-indecated = yes AND
		meal = formal
	Then	drink = wine
R4	If	alcohol-indecated = yes AND
		guest = boss
	Then	drink = wine
R5	If	alcohol-indecated = yes AND
		guest = neighbor
	Then	drink = beer
R6	If	drink = wine AND
		dinner = fish
	Then	wine-type = white
R7	If	drink = wine AND
		dinner = red-meat
	Then	wine-type = red
R8	If	guest = boss AND
		day = friday
	Then	dinner = fish

R9	If	guest = boss AND
		day <> friday
	Then	dinner = red-meat
R10	If	guest-age < 21
	Then	dinner = pizza
R11	If	guest-age $\geq$ 21
	Then	dinner = fish
R12	If	alcohol not indicated
	Then	drink = soda

Chapter 4

Membangun KBS dengan Sistem Berbasis Aturan

Tujuan Instruksional Khusus

- Mahasiswa mampu membuat dan menganalisis rancangan *dependency diagram* untuk kasus yang akan diselesaikan dengan sistem berbasis aturan.
- Mahasiswa mampu menyusun sistem berbasis aturan untuk kasus yang akan diselesaikan dalam sistem pakar.

4.1 Pendahuluan

Note:

Bab ini membahas langkah demi langkah membangun sebuah knowledge-based system (KBS) dengan menggunakan sistem berbasis aturan yang telah dibahas dalam Bab. 3. Sebagai contoh permasalahan akan diambil kasus pada Health Maintenance Organization (HMO) seperti didokumentasikan dalam Dologite(1994).

HMO adalah sebuah organisasi yang memberikan layanan kesehatan bagi anggotanya, misalnya layanan pengobatan, layanan panggilan am-

bulan dsb. Setiap anggota telah membayar semua biaya secara pre-paid.

Untuk menjamin bahwa masalah kesehatan serius akan mendapatkan prioritas layanan, seorang manajer telah menempatkan seseorang untuk melakukan screening awal terhadap pasien. Screening dilakukan dengan cara berkonsultasi dengan Sistem Pakar untuk menentukan status dan jenis layanan yang tepat bagi pasien.

4.2 Langkah-langkah Membangun KBS

Note:

LANGKAH 1: Isolasi area bagi KBS

Untuk membatasi permasalahan pada sistem pakar yang akan dibangun harus diberikan batasan organisasi dan juga layanan yang dapat diberikan oleh sistem. Sebagai contoh, untuk sistem HMO, batasan struktur organisasi dan layanan ditunjukkan dalam Gambar 4.1

LANGKAH 2: Target Keputusan

Setelah permasalahn dibatsi, langkah selanjutnya adalah menentukan target keputusan bagi sistem pakar. Pasien pada umumnya membutuhkan bantuan untuk kasus penyakit yang baru diderita (new case) atau penanganan berkelanjutan dari penyakit yang sudah lama diderita (follow-up case). Atau mungkin juga pasien yang lain hanya membutuhkan informasi atau layanan lain, sedangkan mereka yang bukan merupakan anggota akan diarahkan untuk ikut serta dalam keanggotaan HMO ini. Karena itu dapat ditentukan 3 hal yang menjadi faktor utama yang menentukan target keputusan (Lihat Gambar 4.2), yaitu:

- *HMO status:* Bagaimana status keanggotaan dari pasien? Deklarasi keanggotaan dari pasien akan diikuti dengan validasi nomor id.
- *Reason:* Apa alasan datang ke HMO? Apakah new case, follow-up case, information seeking atau other visit?

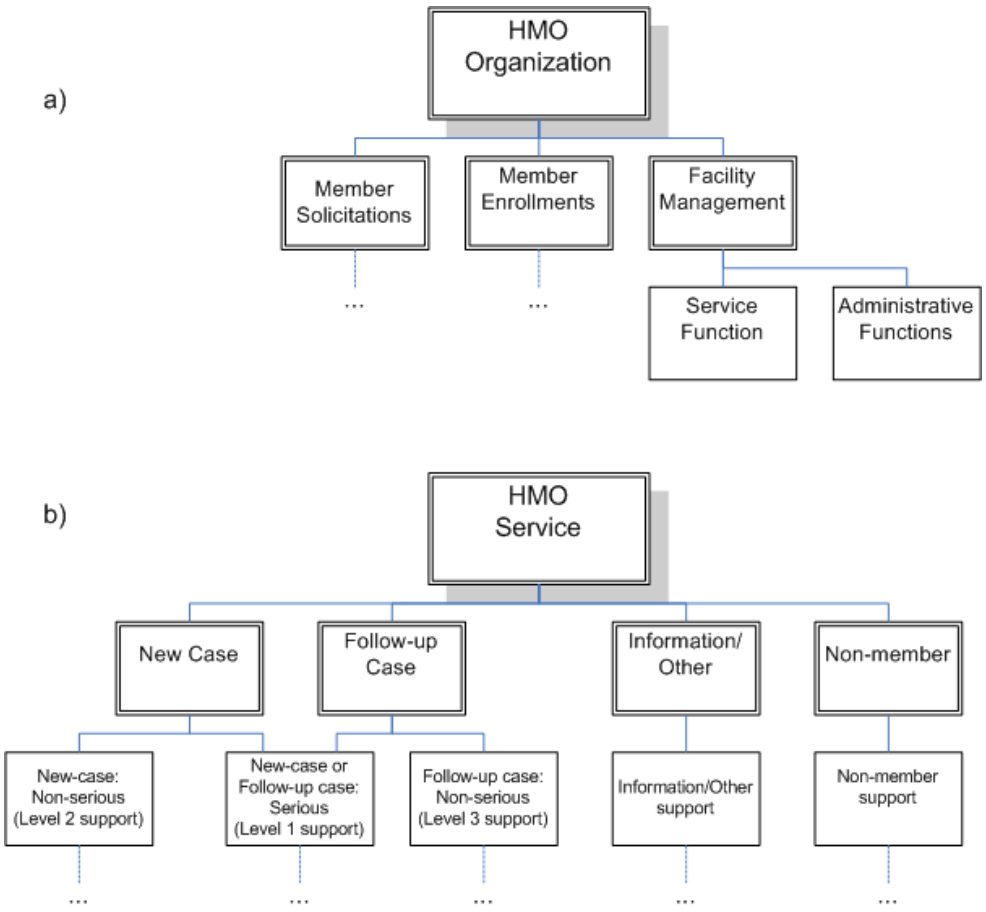

Figure 4.1: Blok diagram organisasi dan layanan HMO

- *Problem:* Bagaimana keseriusan dari kondisi pasien sekarang? Dalam hal ini dapat diidentikasi dari temperature dan symptom yang lain.

LANGKAH 3: Membuat Dependency Diagram (Diagram Ketergantungan)

Dependency Diagram dibuat seperti ditunjukkan dalam Gambar 4.3.

LANGKAH 4: Membuat Tabel Keputusan

Tabel keputusan diturunkan dari dependency diagram pada Gambar 4.3. Karena dalam gambar tersebut terdapat tiga segitiga, maka akan terdapat 3 tabel keputusan. Tabel keputusan untuk Set 1 (Rule 1-5) adalah sebagai berikut:

Note:

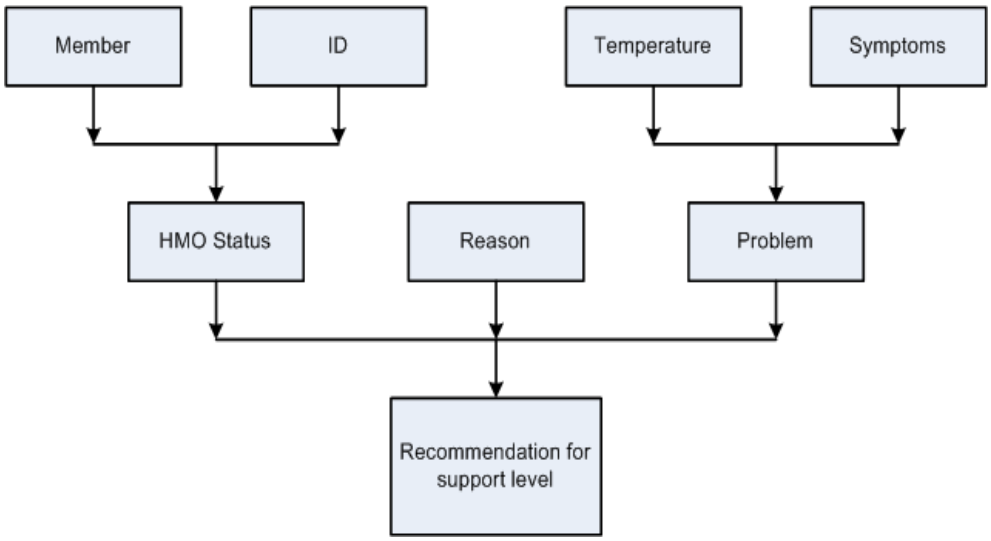

Figure 4.2: Blok diagram target keputusan HMO

Plan

		Number of values
Conditions:	Member-status(ok,not-ok)	= 2
	Reason(new-case, follow-up-case, information-other)	= 3
	Problem(serious,non-serious)	= 2
	Row = 2 × 3 ×2 = 12	

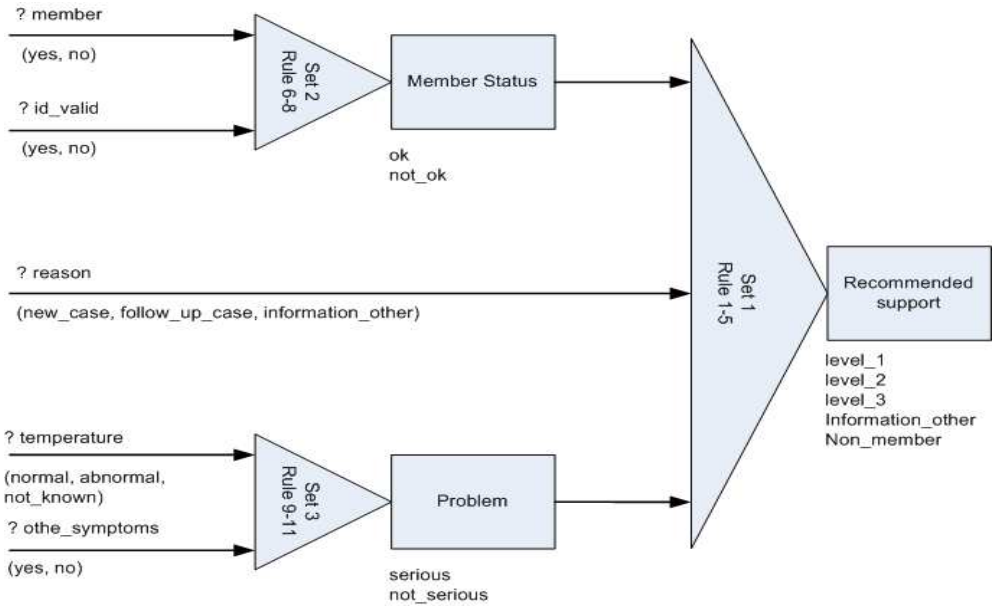

Figure 4.3: Dependency diagram HMO

Tabel Keputusan Set 1

Rule	Member Status	Reason	Problem	Concluding Recommendation for support level
A 1	ok	new-case	serious	level-1
A 2	ok	new-case	non-serious	level-2
A 3	ok	follow-up-case	serious	level-1
A 4	ok	follow-up-case	non-serious	level-3
A 5	ok	information-other	serious	information-other
A 6	ok	information-other	non-serious	information-other
A 7	not-ok	new-case	serious	non-member
A 8	not-ok	new-case	non-serious	non-member
A 9	not-ok	follow-up-case	serious	non-member
A 10	not-ok	follow-up-case	non-serious	non-member
A 11	not-ok	information-other	serious	non-member
A 12	not-ok	information-other	non-serious	non-member

Penyederhanaan Tabel Keputusan Set 1

Rule	Member Status	Reason	Problem	Concluding Recommendation for support level
B 1	ok	new-case	serious	level-1
B 2	ok	new-case	non-serious	level-2
B 3	ok	follow-up-case	serious	level-1
B 4	ok	follow-up-case	non-serious	level-3
B 5	ok	information-other	–	information-other
B 6	not-ok	–	–	non-member

LANGKAH 5: Menulis IF-THEN Rule

Selanjutnya, berdasarkan tabel keputusan yang telah direduksi dapat diturunkan sistem berbasis aturan seperti ditunjukkan di bawah ini:

Rule 1	If	member-status = ok AND
		reason = new-case OR
		reason = follow-up-case AND
		problem = serious
	Then	support = level-1
Rule 2	If	member-status = ok AND
		reason = new-case AND
		problem = non-serious
	Then	support = level-2
Rule 3	If	member-status = ok AND
		reason = follow-up-case AND
		problem = non-serious
	Then	support = level-3
Rule 4	If	member-status = ok AND
		reason = information-other
	Then	support = information-other
Rule 5	If	member-status = not-ok
	Then	support = non-member

LATIHAN

Berdasarkan contoh di atas, buatlah tabel keputusan dan IF-THEN rule untuk Set 2 dan Set 3!

4.3 SOAL LATIHAN

Setelah lulus dari SMA, Julaikah memiliki keinginan untuk melanjutkan studi ke perguruan tinggi di bidang komputer. Namun karena bidang

komputer memiliki beberapa disiplin ilmu, maka Julaikah harus berkonsultasi dengan sistem pakar untuk menentukan pilihan yang tepat bagi dia. Buatlah sebuah sistem pakar untuk membantu Julaikah menentukan pilihannya dengan kriteria sebagai berikut:

- Disiplin ilmu yang direkomendasikan meliputi: Programmer Komputer, Ilmu Komputer (Computer Science), Teknisi Komputer atau bidang lain selain komputer.
- Tiga hal utama sebagai penentu keputusan adalah:
    1. Tes Aptitute (ok, not-ok) yang meliputi: tes kemampuan matematika (Ya, Tidak) dan tes kemampuan programming (Ya, Tidak).
    2. Tes Minat (Bagus, sedang, rendah) yang meliputi: minat pada komputer (Ya, Tidak), minat pada kemampuan reparasi (Ya, Tidak) dan minat pada pemecahan masalah (Ya, Tidak).
    3. Kemampuan finansial (Ya, Tidak).

Maka berdasarkan kriteria di atas tentukan rule-rule dengan menggunakan sistem berbasis aturan (rule-based system), dengan urutan proses (dianjurkan) sebagai berikut:

1. Tentukan Dependency Diagram untuk kasus di atas.
2. Tentukan Tabel Keputusan untuk kasus di atas.
3. Tentukan Rule-rule untuk menyelesaikan kasus di atas.

Chapter 5

Semantic Networks dan Frame

Tujuan Instruksional Khusus

- Mahasiswa mampu menginterpretasikan representasi pengetahuan dalam bentuk *Semantic Networks*.
- Mahasiswa mampu menginterpretasikan representasi pengetahuan dalam bentuk *Frame*.

5.1 Semantic Networks

Note:

- Semantic (associative) networks adalah salah satu bentuk representasi knowledge-base dalam bentuk diagram.
- Diagram tersebut terdiri atas *node* dan *arc*. *Node* merepresentasikan sebuah konsep, sedangkan *arc* merepresentasikan sebuah relasi.
- Sebagai contoh perhatikan node di bawah ini:

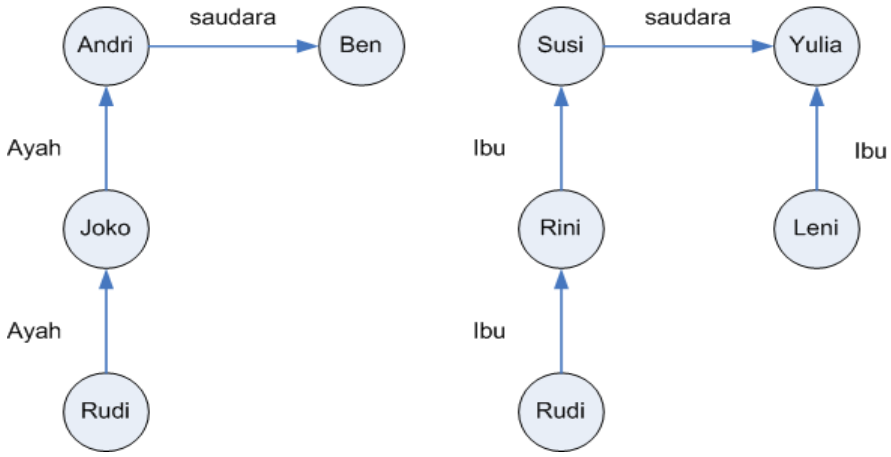

Figure 5.1: Contoh Semantic Networks

- Contoh di atas adalah sebuah semantic networks yang mengilustrasikan sebuah hubungan keluarga. Jika diketahui bahwa Rudi saat ini berumur 12 tahun, Joko berumur 40 tahun, Andri berumur 64 tahun dan Ben berumur 66 tahun. Berapakah umur dari Leni saat ini?

- Diagram di atas dapat dikonversikan ke dalam bentuk predicate calculus sebagai berikut:

  ```
  ayah(joko,rudi)
  ayah(andri,joko)
  saudara(ben,andri)
  ibu(rini,rudi)
  ibu(susi,rini)
  sudara(yulia,susi)
  ibu(yulia,leni)
  ```

- Semantic Networks juga dapat digunakan untuk menggambarkan sebuah hubungan/relasi tunggal dari beberapa konsep. Perhatikan contoh pada gambar 5.2. Gambar tersebut merepresentasikan adanya hubungan saudara di antara Jim, Joko dan Joni.

- Hubungan antar node yang tidak bersifat biner dapat direpresen-

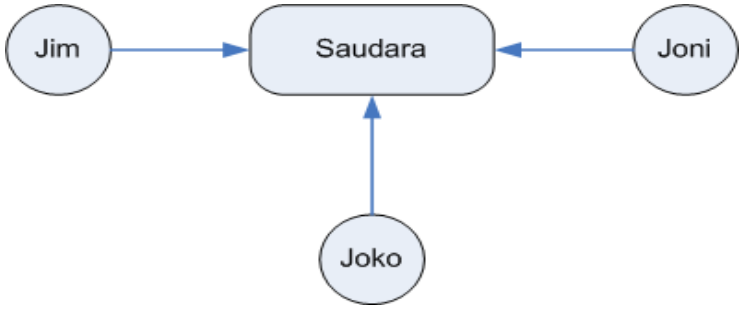

Figure 5.2: Semantic Networks menggambarkan hubungan saudara

tasikan dengan merubah hubungan tersebut menjadi obyek. Semantic networks dengan bentuk semacam ini disebut sebagai *reification*. Perhatikan contoh dalam Gambar dibawah ini: Semantic

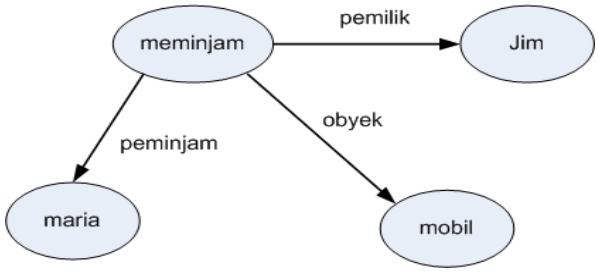

Figure 5.3: Semantic Networks dengan reification

networks tersebut dapat direpresentasikan dalam bentuk predicate calculus yang terdiri atas 3 hal yaitu: pemilik, peminjam dan obyek menjadi: `meminjam(maria,jim,mobil)`.

- Pada kebanyakan semantic networks terdapat kesulitan yang umum, yaitu bagaimana membedakan antara individual dan class? Untuk menggambarkan hal itu, node dapat direpresentasikan dalam dua hal yaitu: node sebagai individual dan node sebagai class. Perhatikan contoh di bawah ini:

Pada gambar tersebut terlihat bahwa node *animal* dan node *birds* adalah class, sedangkan node *wing* adalah sebuah individu.

Representasi pengetahuan dengan menggunakan Semantic Networks masih memiliki beberapa kelemahan, antara lain:

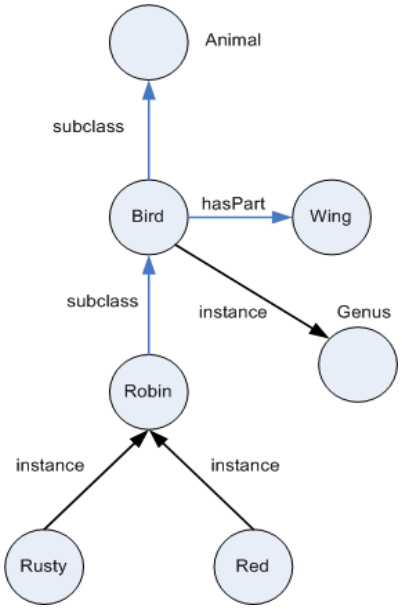

Figure 5.4: Semantic Networks dengan subclass

- memungkinkan terjadinya interpretasi yang berbeda-beda pada semantic networks yang akan membawa pada kesalahan dalam proses pengambilan kesimpulan.

- Relasi yang menghubungkan antar node tidak dapat mengandung semua informasi, misalnya pada Gambar 5.1, relasi *ayah* atau *ibu* tidak menggambarkan apakah relas tersebut merupakan sub-class atau anggota.

5.2 Frame

Note:

- Pada era 70 dan 80 an, semantic networks berubah bentuk menjadi model representasi *frame*.

- Sebuah *frame* memiliki seperangkat *slot*. Sebagai contoh perhatikan diagram struktur keluarga di bawah ini:

  Gambar 5.5 dapat direpresentasikan dalam bentuk frame sebagai

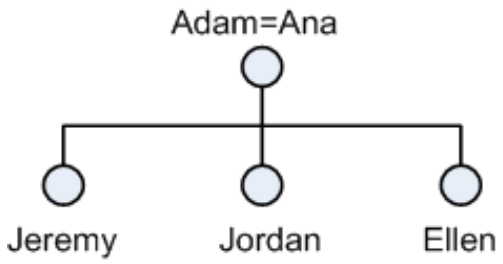

Figure 5.5: Frame dari keluarga Adam

berikut:

```
Frame Adam:
    sex:    Laki-laki
    teman-hidup:  Ana
    anak:   (Jeremy Jordan Ellen)
```

Dalam frame tersebut, frame Adam memiliki 3 buah slot yaitu, sex, teman-hidup dan anak. Sedangkan slot anak memiliki tiga nilai, yaitu: Jeremy, Jordan dan Ellen. Seperti terlihat dalam gambar, kita juga dapat membuat frame lain yang merupakan sub-class dari Frame Adam. Setidaknya ada 7 frame sub-class yang menjelaskan masing-masing individu, yaitu: Adam, Ana, Jeremy, Jordan, Ellen, Laki-laki, and Perempuan.

- Frame Adam dapat dikonversi ke dalam bentuk predicate calculus menjadi:

  ```
  sex(Adam,Laki-laki)
  teman-hidup(Adam,Ana)
  anak(Adam,Jeremy)
  anak(Adam,Jordan)
  anak(Adam,Ellen)
  ```

- Sebuah *slot* merupakan relasi ke frame yang lain atau ke sebuah nilai.

- Sebuah *slot* memiliki satu atau lebih *facet*. Perhatikan contoh frame Binatang pada Gambar 5.6 yang menggambarkan hubungan antar slot untuk membentuk sebuah frame, dan masing-masing slot memiliki beberapa facet.
- *Facet* memiliki nilai yang berasosiasi dengan masing-masing slot.
- Nilai (value) tidak hanya terbatas pada sebuah facet, mungkin juga pada kasus-kasus tertentu sebuah slot di dalam sebuah frame memiliki beberapa nilai.
- Contoh Frame Binatang yang terbentuk atas slot dan facet ditunjukkan seperti dalam Gambar 5.6.

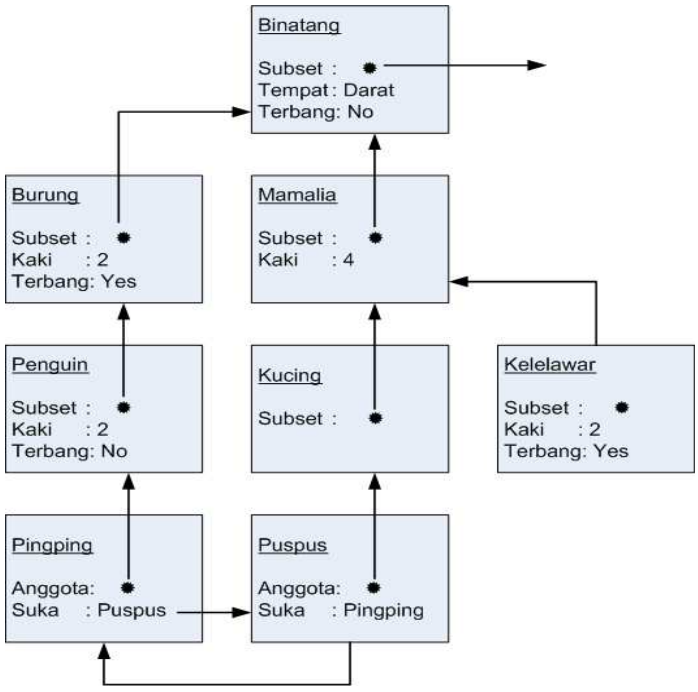

Figure 5.6: Struktur dari Frame Binatang

- Frame juga dapat digunakan untuk merepresentasikan sebuah aksi beserta konsekuensinya. Untuk menggambarkan hal tersebut digunakan dua buah frame, yaitu *Action Frame* dan *State-Change Frame*.

- Sebagai contoh perhatikanlah contoh frame yang merepresentasikan sebuah aksi dalam Gambar 5.8. Dalam bentuk kalimat frame ini berarti: "Mengantongi hadiah 500 ribu membuat Toni senang".

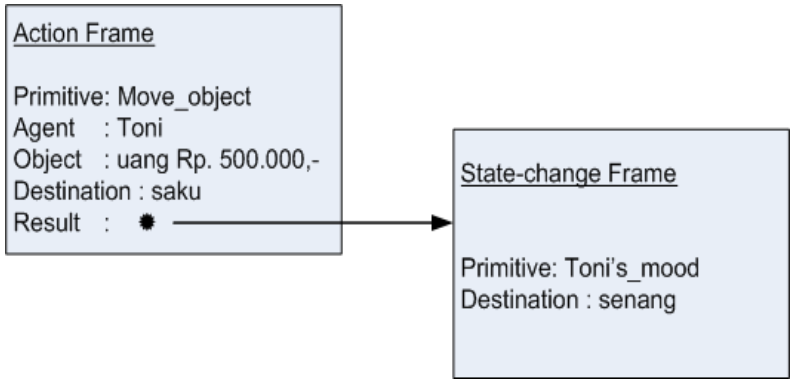

Figure 5.7: Contoh Action Frame

Seperti terlihat dalam gambar, *state-change frame* merupakan hasil action dari frame pertama.

- Slot *Primitive* dapat terdiri atas salah satu dari 15 macam primitive, yaitu:

Move-body-part	Expel	Ingest
Move-object	Propel	Speak
Hear	Feel	See
Smell	Move-possession	Conclude
Move-concept	Think-about	Do

- Secara lebih detail *Action Frame* juga dapat dibagi-bagi lagi menjadi beberapa sub-aksi seperti ditunjukkan dalam Gambar 5.7.

- Seperti terlihat dalam gambar tersebut, *Action Frame* memiliki 3 buah *sub-action frame* yang masing-masing menjelaskan detail aksi dari *action frame*.

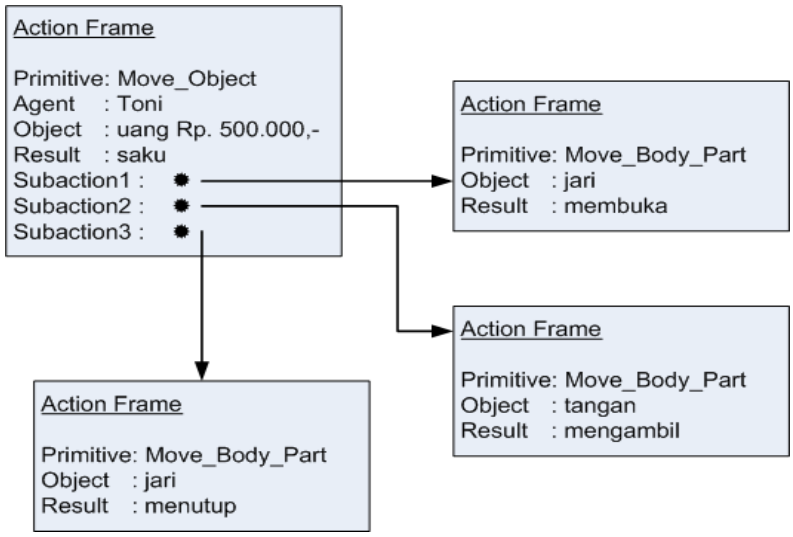

Figure 5.8: Contoh Sub-Action Frame

- Dari beberapa contoh di atas, representasi pengetahuan dengan *Action Frame* harus tunduk pada atura-aturan sebagai berikut:
  - Sebuah *Action Frame* harus memiliki sebuah slot *primitive* dengan mengambil salah satu dari 15 primitive yang telah didefinisikan.
  - Sebuah *State-Change Frame* mengandung slot *object* yang berisi nilai sesuai dengan kebutuhan aplikasi.
  - Sebuah *Action Frame* dapat dihubungkan kepada satu atau beberapa *action frame* lain melalui slot *subaction*.
  - *Action Frame* dan *State-Change Frame* dapat dihubungkan dengan menggunakan slot *result*.
  - Slot-slot yang lain beserta nilainya tergantung pada kebutuhan aplikasi.

5.3 SOAL LATIHAN

Diberikan satu set fakta sebagai berikut:

ayah(Suryo,Arman)

ibu(Susi,Lusi)

istri(Sari,Suryo)

suami(Joko,Susi)

istri(Maria,Arman)

anak(Doni, Arman)

ayah(Arman,Haris)

anak(Ari,Susi)

anak(Susi,Suryo)

Berdasarkan fakta-fakta di atas jawablah pertanyaan di bawah ini:

1. Gambarkan semantic networks diagram dari silsilah keluarga tersebut!
2. Apa isi dari variabel X untuk ekpresi : paman(X,Ari)?
3. Apa isi dari variabel X untuk ekpresi : ibu(X,Susi)?
4. Apa isi dari variabel Y untuk ekpresi : saudara-kandung(Lusi,Y)?
5. Apa isi dari variabel Y untuk ekpresi : ipar(Arman,Y)?
6. Apa isi dari variabel X untuk ekpresi : nenek(X,Doni)?
7. Apa isi dari variabel Y untuk ekpresi : menantu(Sari,Y)?

Chapter 6

Uncertainty Management

Tujuan Instruksional Khusus

- Mahasiswa memahami pendekatan Bayesian sebagai dasar interpretasi fakta yang memiliki derajad ketidak-pastian tertentu.
- Mahasiswa mampu membuat interpretasi fakta dengan menggunakan metoda certainty factor.
- Mahasiswa mampu mendefinisikan dan menyelesaikan (menarik kesimpulan) permasalahan yang mengandung fakta dengan derajad ketidakpastian tertentu.

6.1 Pendahuluan

- Dalam pembicaraan sistem intelligent, dalam banyak hal kita berhadapan dengan data yang bersifat ambigu, samar dan tidak pasti. Sebagai contoh, pada sebuah fakta: `Angin bertiup kencang`, terkandung ketidak-pastian terhadap berapa tingkat kekencangan dari tiupan angin. Dalam bahasa sehari-hari, kita sering menjumpai

fakta-fakta yang samar/ambigu/tidak pasti seperti pada contoh di atas.

Note:

- Karena itu dalam hal representasi knowledge dibutuhkan juga suatu cara agar derajad ketidakpastian dari sebuah fakta dapat terwakili dengan baik. Representasi pengetahuan semacam inilah yang akan dibahas dalam *uncertainty management*.

- Setidaknya terdapat tiga isue yang harus diselesaikan dalam pembicaraan *uncertainty management*, yaitu:

  1. Bagaimana merepresentasikan data yang tidak pasti (uncertain data)?
  2. Bagaimana mengkombinasikan dua atau lebih data yang tidak pasti?
  3. Bagaimana mengambil kesimpulan (inferensi) menggunakan data yang tidak pasti?

6.2 Pendekatan Bayesian

Note:

- Bayes' Rule merupakan teknik tertua dan paling baik untuk menggambarkan ketidakpastian. Bayes' Rule dibangun berdasarkan teori probabilitas klasik.

- Misalkan x_i adalah beberapa *event*, koleksi dari semua event yang disebut *sample space* didefinisikan sebagai himpunan X (huruf kapital), yang mana:

$$X = \{x_1, x_2, \ldots, x_n\}$$

- Probabilitas dari event x_i terjadi dinotasikan sebagai $p(x)$.

- Setiap fungsi probabilitas, p,harus memenuhi tiga kondisi di bawah ini:

  1. Probabilitas dari sembarang event x_i adalah positif. Probabilitas sebuah event mungkin 0 (event tidak akan terjadi) atau mungkin 1 (event pasti terjadi) atau mungkin sembarang nilai antara 0 dan 1.
  2. Jumlah total probabilitas untuk seluruh sample space adalah satu (1).
  3. Jika satu set event $x_i, x_2, \ldots, x_k$ adalah *mutually exclusive*, maka probabilitas bahwa paling tidak satu dari event tersebut terjadi adalah jumlah dari semua probabilitas dari masing-masing elemen.

- Misalkan kita memiliki dua buah event x dan y dari sebuah sample space, kemungkinan/probabilitas bahwa event y terjadi jika event x terjadi, disebut sebagai *conditional probability* dan ditulis sebagai $p(y|x)$. Probabilitas keduanya terjadi disebut sebagai *joint probability* dan dinotasikan sebagai $p(x \wedge y)$. Menurut Bayes' rule, conditional probility didefinisikan sebagai:

$$p(y|x) = \frac{p(x|y) * p(y)}{p(x)} \tag{6.1}$$

  dalam bentuk yang lain, Bayes' rule juga dapat ditulis sebagai:

$$p(y|x) = \frac{p(x|y) * p(y)}{p(x|y) * p(y) + p(x| \sim y) * p(\sim y)}. \tag{6.2}$$

6.2.1 Bayes' Rule dan Sistem Berbasis Pengetahuan

Note:

Seperti pada pembahasan dalam Bab 3, sistem berbasis pengetahuan dapat direpresentasikan dalam format IF-THEN dengan:

`IF` X `adalah benar`
`THEN` Y `dapat disimpulkan dengan probabilitas` p

Artinya, apabila hasil observasi kita menunjukkan bahwa X adalah benar, maka dapat disimpulkan bahwa Y ada dengan probabilitas tertentu. Sebagai contoh:

`IF` `Seseorang sedang marah`
`THEN` `Seseorang akan meninggalkan rumah (0.75)`

Akan tetapi jika observasi dilakukan terhadap Y tanpa mengetahui apapun yang terjadi dengan X, kesimpulan apa yang dapat ditarik? Bayes' rule mendefinisikan bagaimana kita dapat menurunkan probabilitas dari X. Y seringkali juga disebut sebagai evidence (disimbolkan dengan E) dan X disebut sebagai hypothesis (disimbolkan dengan H), maka persamaan Bayes' rule menjadi:

$$p(H|E) = \frac{p(E|H) * p(H)}{p(E)} \tag{6.3}$$

atau

$$p(H|E) = \frac{p(E|H) * p(H)}{p(E|H) * p(H) + p(x| \sim H) * p(\sim H)}. \tag{6.4}$$

Sekarang marilah kita hitung kemungkinan bahwa `Joko sedang marah` jika diketahui bahwa `ia meninggalkan rumah`.

- Persamaan 6.3 menunjukkan bahwa probabilitas bahwa Joko sedang marah jika diketahui bahwa ia sedang meninggalkan rumah adalah:

perbandingan antara probabilitas bahwa Joko marah dan meninggalkan rumah dengan probabilitas bahwa ia meninggalkan rumah.

- Probabilitas bahwa Joko meninggalkan rumah adalah jumlah antara conditional probability bahwa ia meninggalkan rumah jika ia marah dan conditional probability bahwa ia meninggalkan rumah jika ia tidak marah. Dengan kalimat lain item ini berarti probabilitas bahwa ia meninggalkan rumah tidak peduli apakah Joko marah atau tidak.

Misalkan diketahui data-data sebagai berikut:

$$\begin{aligned} p(H) &= p(\text{Joko sedang marah}) \\ &= 0.2 \\ p(E|H) &= p(\text{Joko meninggalkan rumah}|\text{Joko sedang marah}) \\ &= 0.75 \\ p(E|\sim H) &= p(\text{Joko meninggalkan rumah}|\text{Joko tidak sedang marah}) \\ &= 0.2 \end{aligned}$$

maka

$$\begin{aligned} p(E) &= p(\text{Joko meninggalkan rumah}) \\ &= (0.75)(0.2) + (0.2)(0.8) \\ &= 0.31 \end{aligned}$$

dan

$$\begin{aligned} p(H|E) &= p(\text{Joko sedang marah}|\text{Joko meninggalkan rumah}) \\ &= \frac{(0.75) * (0.2)}{(0.31)} \\ &= 0.48387 \end{aligned}$$

Dengan kata lain, probabilitas bahwa Joko sedang marah jika diketahui bahwa ia meninggalkan rumah adalah sekitar 0.5. Dengan cara

yang sama probabilitas bahwa ia sedang marah jika Joko tidak meninggalkan rumah adalah:

$$\begin{aligned} p(H|\sim E) &= \frac{p(\sim E|H) * p(H)}{p(\sim E)} \\ &= \frac{(1-0.75) * (0.2)}{(1-0.31)} \\ &= 0.07246 \end{aligned}$$

Jadi dengan mengetahui bahwa Joko meninggalkan rumah, meningkatkan probabilitas bahwa ia sedang marah kira-kira 2.5 kali. Sedangkan mengetahui bahwa ia tidak meninggalkan rumah menurunkan probabilitas bahwa ia sedang marah sekitar 3 kali.

6.2.2 Propagasi Kepercayaan

Note:

Sebagaimana dibicarakan dalam sub-Bab sebelumnya, Bayes' rule hanya mempertimbangkan satu hypothesis dan satu evidence. Sebenarnya Bayes' rule dapat digeneralisasi untuk kasus dimana terdapat m hypothesis dan n evidence yang biasanya ditemui dalam kehidupan sehari-hari. Maka persamaan Bayes' rule menjadi:

$$\begin{aligned} p(H_i|E_1E_2\ldots E_n) &= \frac{p(E_1E_2\ldots E_n|H_i) * p(H_i)}{p(E_1E_2\ldots E_3)} \\ &= \frac{p(E_1|H_i) * p(E_2|H_i) * p(E_n|H_i) * p(H_i)}{\sum_{k=1}^{m} p(E_1|H_k) * p(E_2|H_k) * \ldots * p(E_n|H_k) * p(H_k)} \end{aligned} \quad (6.5)$$

Persamaan di atas disebut sebagai *posterior probability* hypothesis H_i dari observasi terhadap evidence $E_1, E_2, \ldots, E_n$.

Untuk memberikan ilustrasi bagaimana kepercayaan dipropagasikan dalam Bayes' rule, perhatikan contoh dalam Tabel 6.1. Tabel ini menjelaskan bahwa terdapat tiga mutually exclusive hypothesis, yaitu: $H1$,

Manager Lapindo melakukan kesalahan pengeboran, $H2$, Manager Lapindo tidak mempunyai konsultan profesional, dan H_3, Manager Lapindo terkena getah akibat bencana alam. Juga terdapat dua evidence yang saling bebas, yaitu: $E1$, Lumpur terus mengalir dan E_2, Patahan bor tertinggal dalam perut bumi, yang mendukung ketiga hypothesis.

Table 6.1: Contoh Kasus Propagasi Kepercayaan

	$i = 1$ (kesalahan)	$i = 2$ (tidak ada konsultan)	$i = 3$ (bencana alam)
$p(H_i)$	0.4	0.6	0.1
$p(E_1\|H_i)$	0.8	0.4	0.3
$p(E_2\|H_i)$	0.9	0.6	0.0

Jika observasi dilakukan terhadap E_1 (i.e., Lumpur terus mengalir), maka dengan menggunakan persamaan 6.5 kita dapat menghitung posterior probability dari masing-masing hypothesis sebagai berikut:

$$\begin{aligned} p(H_1|E_1) &= \frac{0.8 * 0.4}{0.8 * 0.4 + 0.4 * 0.6 + 0.3 * 0.1} = 0.54 \\ p(H_2|E_1) &= \frac{0.4 * 0.6}{0.8 * 0.4 + 0.4 * 0.6 + 0.3 * 0.1} = 0.41 \\ p(H_3|E_1) &= \frac{0.3 * 0.1}{0.8 * 0.4 + 0.4 * 0.6 + 0.3 * 0.1} = 0.05 \end{aligned}$$

Perhatikan bahwa kepercayaan pada hypothesis H_2 dan H_3 menurun sedang tingkat kepercayaan pada hypothesis H_1 naik setelah observasi terhadap E_1. Jika obeservasi sekarang juga dilakukan terhadap E_2, maka posterior probability dapat dihitung dengan:

$$\begin{aligned} p(H_1|E_1E_2) &= \frac{0.8 * 0.9 * 0.4}{0.8 * 0.9 * 0.4 + 0.4 * 0.6 * 0.6 + 0.3 * 0.0 * 0.1} = 0.67 \\ p(H_2|E_1E_2) &= \frac{0.4 * 0.6 * 0.6}{0.8 * 0.9 * 0.4 + 0.4 * 0.6 * 0.6 + 0.3 * 0.0 * 0.1} = 0.33 \\ p(H_3|E_1E_2) &= \frac{0.3 * 0.0 * 0.1}{0.8 * 0.9 * 0.4 + 0.4 * 0.6 * 0.6 + 0.3 * 0.0 * 0.1} = 0.0 \end{aligned}$$

Pada contoh di atas hypothesis H_3 bukan merupakan hypothesis yang valid, sedangkan hypothesis H_1 sekarang dianggap lebih memungkinkan walaupun pada awalnya H_2 berada diperingkat pertama.

LATIHAN

Misalkan diketahui fakta sebagai berikut:

(a) Probabilitas bahwa kita akan melihat buaya di sungai Jagir adalah 0.7.

(b) Probabilitas bahwa banyak itik di sungai Jagir jika kita melihat buaya adalah 0.05.

(c) Probabilitas bahwa banyak itik di sungai Jagir jika kita tidak melihat buaya di sungai adalah 0.2.

Berapa probabilitas kita melihat buaya jika terdapat beberapa itik di sungai Jagir[1]?

6.3 Certainty Factor

Note:

Pengetahuan di dalam sistem pakar yang direpresentasikan dengan menggunakan Certainty Facto (CF) diekspresikan dalam seperangkat aturan yang memiliki format:

```
IF   EVIDENCE
THEN HYPOTHESIS CF(RULE)
```

dimana Evidence adalah satu atau beberapa fakta yang diketahui untuk mendukung Hypothesis dan CF(RULE) adalah certainty factor untuk Hypothesis jika evidence diketahui.

[1] $p(H)$=p(Kita melihat buaya);$p(E)$=p(Kita melihat itik di sungai Jagir); maka $p(H|E)$=0.368.

Seperti dalam pembahasan terhahulu, probabilitas dari suatu hypothesis terjadi jika diketahui/diberikan beberapa evidence disebut sebagai conditional probability dan disimbulkan sebagai $p(H|E)$. Jika $p(H|E)$ lebih besar dari probabilitas sebelumnya, yaitu: $p(H|E) > p(H)$, maka berarti keyakinan pada hypothesis meningkat. Sebaliknya jika $p(H|E)$ lebih kecil dari dari probabilitas sebelumnya, yaitu: $p(H|E) < p(H)$, maka keyakinan pada hypothesis akan menurun.

Ukuran yang menunjukkan peningkatan keyakinan pada suatu hypothesis berdasarkan evidence yang ada disebut sebagai *measure of belief (MB)*. Sedangkan ukuran yang menunjukkan penurunan keyakinan pada suatu hypothesis berdasarkan evidence yang ada disebut sebagai *measure of disbelief (MD)*.

Nilai dari MB dan MD dibatasi sedemikian sehingga:

$$0 \leq MB \leq 1$$
$$0 \leq MD \leq 1$$

Ukuran MB secara formal didefinisikan sebagai:

$$\text{MB}(H,E) = \begin{cases} 1 & \text{if } P(H) = 1 \\ \frac{max[P(H|E),P(H)]-P(H)}{1-P(H)} & \text{otherwise} \end{cases} \tag{6.6}$$

Sedangkan MD didefinisikan sebagai:

$$\text{MD}(H,E) = \begin{cases} 1 & \text{if } P(H) = 0 \\ \frac{P(H)-min[P(H|E),P(H)]}{P(H)} & \text{otherwise.} \end{cases} \tag{6.7}$$

Karena dalam proses observasi kepercayaan dapat bertambah atau berkurang, maka diperlukan ukuran ketiga untuk mengkombinasikan MB dan MD, yaitu: *Certainty Factor*. Certainty Factor didefiniskan sebagai:

$$\text{CF}(H,E) = \frac{\text{MB}(H,E) - \text{MD}(H,E)}{1 - min(\text{MB}(H,E), \text{MD}(H,E))} \tag{6.8}$$

Dimana nilai dar CF dibatasi oleh:

$$-1 \leq CF \leq 1$$

Nilai 1 berarti sangat benar, nilai 0 berarti tidak diketahui dan nilai -1 berarti sangat salah. Nilai CF negatif menunjuk pada derajad ketidak-percayaan sedang nilai CF positif menunjuk pada derajad kepercayaan.

6.3.1 Propagasi Keyakinan untuk Rule dengan Satu Premise

Note:

Yang dimaksud dengan propagasi keyakinan/kepercayaan adalah proses menentukan derajad kepercayaan pada kesimpulan pada kondisi dimana fakta/bukti/evidence yang ada tidak pasti (uncertain). Untuk rule dengan satu premise CF(H,E) didapatkan dengan rumusan:

$$\text{CF}(H, E) = \text{CF}(E) * \text{CF}(\text{RULE}) \tag{6.9}$$

Propagasi Keyakinan untuk Rule dengan Beberapa Premise

Note:

Pada rule dengan beberapa premise terdapat dua macam penghubung yang biasa digunakan untuk menghubungkan premise-premise tersebut: konjungsi dan disjungsi.

Rule dengan Konjungsi

Pada rule dengan konjungsi, pendekatan yang digunakan adalah sebagai berikut:

$$\begin{gathered}\text{IF } E_1 \text{ AND } E_2 \text{ AND } \ldots \text{ THEN } H \quad \text{CF(RULE)} \\ \text{CF}(H, E_1 \text{ AND } E_2 \text{ AND } \ldots) = min\{\text{CF}(E_i)\} * \text{CF}(\text{RULE})\end{gathered} \tag{6.10}$$

Fungsi 'min' akan mengembalikan nilai paling kecil dari 1 set evidence yang ada.

Perhatikan contoh dibawah ini:

IF	Suhu udara rata-rata turun
AND	Hembusan angin semakin kencang
THEN	Musim hujan akan segera datang. (CF=0.8)

Asumsikan bahwa derajad kepercayaan kita pada premise pertama adalah:

$$\text{CF(Suhu udara rata-rata turun)} = 1.0$$

dan derajad kepercayaan pada premise kedua:

$$\text{CF(Hembusan angin semakin kencang)} = 0.7$$

Maka derajad kepercayaan bahwa 'Musim hujan akan datang' dapat dihitung:

CF(Musim hujan akan datang *jika*
suhu udara rata-rata turun *AND*
hembusan angin semakin kencang) $= min\{1.0, 0.7\} * 0.8 = 0.56$

Berarti bahwa: Musim hujan *mungkin* akan datang.

Rule dengan disjungsi

Pada rule dengan disjungsi, pendekatan yang digunakan adalah sebagai berikut:

$$\text{IF } E_1 \text{ OR } E_2 \text{ OR } \ldots \text{ THEN } H \quad \text{CF(RULE)}$$
$$\text{CF}(H, E_1 \text{ OR } E_2 \text{ OR } \ldots) = max\{\text{CF}(E_i)\} * \text{CF(RULE)} \qquad (6.11)$$

Fungsi 'max' akan mengembalikan nilai paling besar dari 1 set evidence yang ada.

Contoh:

IF	Suhu udara rata-rata turun
OR	Hembusan angin semakin kencang
THEN	Musim hujan akan datang. (CF=0.9)

Maka derajad kepercayaan bahwa 'Musim hujan akan datang' adalah:

CF(Musim hujan akan datang *jika*
suhu udara rata-rata turun *OR*
hembusan angin semakin kencang) $= max\{1.0, 0.7\} * 0.9 = 0.9$

Berarti bahwa: Musim hujan *hampir pasti* akan datang.

LATIHAN

Bagaimana bentuk certainty factor dari hypothesis untuk rule seperti ditunjukkan di bawah ini:

IF	E_1
AND	E_2
OR	E_3
AND	E_4
THEN	H CF(RULE).

6.3.2 Rule dengan konklusi yang sama

Dalam proses eksekusi rule, mungkin sekali terjadi bahwa beberapa rule dapat menghasilkan hypothesis atau kesimpulan yang sama. Karena itu harus ada mekanisme untuk mengkombinasikan beberapa hypothesis tersebut untuk menjadi satu buah hypothesis saja. Persamaan untuk

menggabungkan dua buah CF adalah sebagai berikut:

$$\mathrm{CF}_{\mathrm{kombinasi}}(\mathrm{CF}_{\mathrm{lama}}, \mathrm{CF}_{\mathrm{baru}}) = \begin{cases} \mathrm{CF}_{\mathrm{lama}} + \mathrm{CF}_{\mathrm{baru}}(1 - \mathrm{CF}_{\mathrm{lama}}) & \text{both } > 0 \\ \mathrm{CF}_{\mathrm{lama}} + \mathrm{CF}_{\mathrm{baru}}(1 + \mathrm{CF}_{\mathrm{lama}}) & \text{both } < 0 \\ \frac{\mathrm{CF}_{\mathrm{lama}} + \mathrm{CF}_{\mathrm{baru}}}{1 - min(|\mathrm{CF}_{\mathrm{lama}}|, |\mathrm{CF}_{\mathrm{baru}}|)} & \text{one } < 0 \end{cases} \tag{6.12}$$

Untuk menjelaskan bagaimana keyakinan dipropagasikan dalam Certainty Factor, maka dalam bagian ini akan diberikan dua contoh kasus yang diselesaikan dengan model Certainty Factor.

Contoh 1:

Contoh pertama adalah berkaitan dengan proses keputusan di dalam sebuah pengadilan dimana seseorang telah dituduh terlibat dalam pembunuhan tingkat pertama (hypothesis). Contoh ini diambil dari Gonzales (1993). Berdasarkan fakta-fakta yang ada (evidence) hakim harus memutuskan apakah orang tersebut bersalah. Pada awal proses peradilan, hakim harus menjunjung tinggi asas praduga tak bersalah, karena itu pada certainty factor dari 'bersalah' bernilai 0 (CF=0). Perhatikan rule-rule dibawah ini:

R1	IF	Sidik jari tertuduh ada pada senjata pembunuh,
	THEN	Tertuduh bersalah. (CF = 0.75)
R2	IF	Tertuduh memiliki motif,
	THEN	Tertuduh bersalah melakukan kejahatan. (CF = 0.6)
R3	IF	Tertuduh memiliki alibi,
	THEN	Tertuduh tidak bersalah. (CF = -0.8)

Dalam proses peradilan diketahui fakta-fakta sebagai berikut:

- Sidik jari tertuduh ada pada senjata pembunuh (CF = 0.9).
- Tertuduh memiliki motif (CF=0.5).
- Tertuduh memiliki alibi (CF=0.95).

Penyelesaian untuk kasus di atas adalah sebagai berikut:

STEP 0

Dengan menjunjung asas praduga tak bersalah, pada tahap awal hakim akan mengasumsikan bahwa "tertuduh bersalah" memiliki CF=0, seperti ditunjukkan dalam Gambar 6.1.

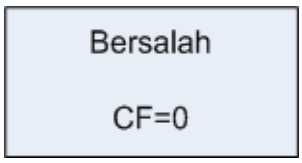

Figure 6.1: Tertuduh bersalah, CF=0

STEP 1

Diketahui bahwa premise dari R1 memiliki evidence dengan nilai CF=0.9. Maka hasil propagasi keyakinan yang memberi pengaruh pada bagian hypothesis adalah:

$$\begin{aligned} CF_{kombinasi1} &= CF_{R1} * CF_{evid1} \\ &= 0.75 * 0.9 \\ &= 0.675 \end{aligned} \tag{6.13}$$

Karena pada saat awal kita asumsikan bahwa nilai 'bersalah' adalah 0, maka CF_{revisi} dapat dicari dengan:

$$\begin{aligned} CF_{revisi} &= CF_{lama} + CF_{baru} * (1 - CF_{lama}) \\ &= 0.0 + 0.675 * (1 - 0) \\ &= 0.675 \end{aligned} \tag{6.14}$$

Hasil propagasi kepercayaan R1 menyebabkan nilai CF sekarang berubah menjadi 0.675. Ditunjukkan dalam Gambar 6.2. Yang berarti: dengan adanya R1 meningkatkan kepercayaan bahwa tertuduh bersalah. Tetapi hakim tidak akan langsung mengetokkan palu tanda bersalah sebelum bukti-bukti yang lain diuji.

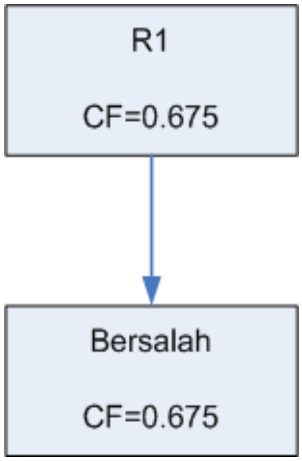

Figure 6.2: Tertuduh bersalah, CF=0.675

STEP 2

Diketahui bahwa premise dari R2 memiliki evidence dengan nilai CF=0.5. Maka hasil propagasi keyakinan yang memberi pengaruh pada bagian hypothesis dari R2 adalah:

$$\begin{aligned} CF_{kombinasi2} &= CF_{R2} * CF_{evid2} \\ &= 0.6 * 0.5 \\ &= 0.30 \end{aligned} \tag{6.15}$$

Pada step 1 kita dapatkan dengan CF=0.675, maka selanjutnya tingkat keyakinan dipropagasikan dengan adanya evidence kedua menjadi:

$$\begin{aligned} CF_{revisi} &= CF_{lama} + CF_{baru} * (1 - CF_{lama}) \\ &= 0.675 + 0.3 * (1 - 0.675) \\ &= 0.7725 \end{aligned} \tag{6.16}$$

Hasil kombinasi antara R1 dan R2 ternyata semakin meningkatkan kepercayaan bahwa tertuduh memang bersalah. Perhatikan Gambar 6.3.

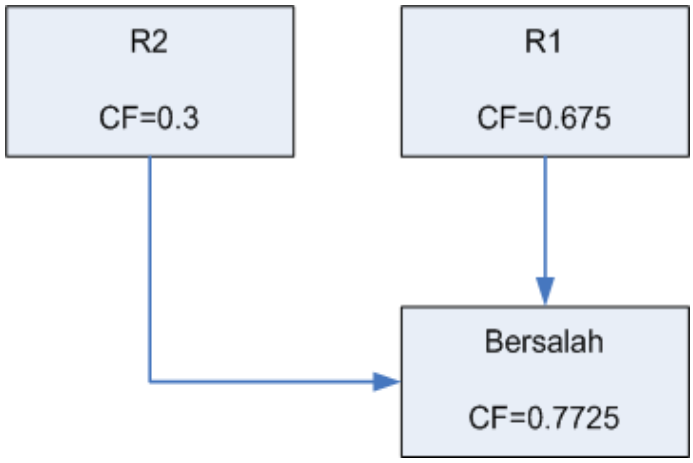

Figure 6.3: Tertuduh bersalah, CF=0.7725

STEP 3

Diketahui bahwa premise dari R3 memiliki evidence dengan nilai CF=0.95. Maka hasil propagasi keyakinan yang memberi pengaruh pada bagian hypothesis dari R3 adalah:

$$\begin{aligned} \mathrm{CF}_{\mathrm{kombinasi3}} &= \mathrm{CF}_{\mathrm{R3}} * \mathrm{CF}_{\mathrm{evid3}} \\ &= 0.95 * (-0.80) \\ &= -0.76 \end{aligned} \tag{6.17}$$

Hasil kombinasi terakhir memberikan nilai:

$$\begin{aligned} \mathrm{CF}_{\mathrm{revisi}} &= \frac{\mathrm{CF}_{\mathrm{lama}} + \mathrm{CF}_{\mathrm{baru}}}{1 - min(|\mathrm{CF}_{\mathrm{lama}}|, |\mathrm{CF}_{\mathrm{baru}}|)} \\ &= \frac{0.7725 - 0.76}{1 - 0.76} \\ &= 0.052 \end{aligned} \tag{6.18}$$

Karena itu hasil akhir dari proses peradilan: hakim tidak dapat memutuskan bahwa tertuduh bersalah. Bukti lain dibutuhkan untuk menentukan tertuduh bersalah atau tidak. Perhatikan Gambar 6.4.

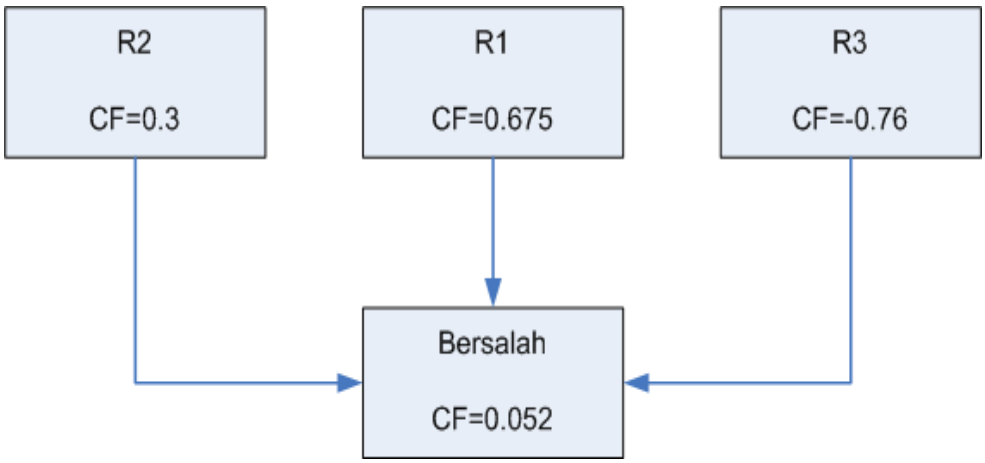

Figure 6.4: Tertuduh bersalah, CF=0.052

Contoh 2:

Problem berikutnya adalah menentukan apakah saya seharusnya pergi bermain bola atau tidak. Kita asumsikan bahwa hypothesis adalah: "Saya seharusnya tidak pergi bermain bola" dan penyelesaian dilakukan dengan metoda backward reasoning. Rule-rule yang digunakan adalah sebagai berikut:

R1	IF	Cuaca kelihatan mendung,	E1
	OR	Saya dalam suasana hati tidak enak,	E2
	THEN	Saya seharusnya tidak pergi bermain bola. (CF = 0.9)	H1
R2	IF	Saya percaya akan hujan,	E3
	THEN	Cuaca kelihatan mendung. (CF = 0.8)	E1
R3	IF	Saya percaya akan hujan,	E3
	AND	Ramalan cuaca mengatakan akan hujan,	E4
	THEN	Saya dalam suasana hati tidak enak. (CF = 0.9)	E2
R4	IF	Ramalan cuaca mengatakan akan hujan,	E4
	THEN	Cuaca kelihatan mendung. (CF = 0.7)	E1
R5	IF	Cuaca kelihatan mendung,	E1
	THEN	Saya dalam suasana hati tidak enak. (CF = 0.95)	E2

Dan diketahui fakta-fakta sebagai berikut:

- Saya percaya akan hujan (CF=0.95).
- Ramalan cuaca mengatakan akan hujan (CF=0.85).

Penyelesaian untuk kasus di atas adalah sebagai berikut:

STEP 1

Perhatikan bahwa premise pertama pada R1 (yang disimbolkan dengan E1) merupakan konklusi dari R2 dan R4. Sistem akan mengerjakan R2 terlebih dahulu karena R2 memiliki nilai CF lebih besar daripada R4. Maka:

$$\begin{aligned} \text{CF(E1,E3)} &= \text{CF}_{\text{R2}} * \text{CF}_{\text{E3}} \\ &= 0.8 * 0.95 \\ &= 0.76 \end{aligned} \tag{6.19}$$

Setelah itu kita kerjakan R4 sehingga:

$$\begin{aligned} \text{CF(E1,E4)} &= \text{CF}_{\text{R4}} * \text{CF}_{\text{E4}} \\ &= 0.7 * 0.85 \\ &= 0.60 \end{aligned} \tag{6.20}$$

Sekarang kita memiliki 2 buah fakta baru yang memberikan konfirmasi tentang E1 (Cuaca kelihatan mendung), kombinasi dari kedua buah fakta tersebut adalah:

$$\begin{aligned} \text{CF(E1)} &= \text{CF(E1,E3)} + \text{CF(E1,E4)} * (1 - \text{CF(E1,E3)}) \\ &= 0.76 + 0.6 * (1 - 0.76) \\ &= 0.9 \end{aligned} \tag{6.21}$$

STEP 2

Perhatikan bahwa premise kedua pada R1 (yang disimbolkan dengan E2) merupakan konklusi dari R3 dan R5. Sistem akan mengerjakan R5

terlebih dahulu karena R5 memiliki nilai CF lebih besar daripada R3. Maka:

$$\begin{aligned} \text{CF(E2,E1)} &= \text{CF}_{\text{R5}} * \text{CF}_{\text{E1}} \\ &= 0.95 * 0.9 \\ &= 0.86 \end{aligned} \tag{6.22}$$

Selanjutnya sistem akan mengerjakan R3 sehingga:

$$\begin{aligned} \text{CF(E1,E3 AND E4)} &= min\{\text{CF}_{\text{E3}}, \text{CF}_{\text{E4}}\} * \text{CF}_{\text{R3}} \\ &= min\{0.85, 0.85\} * 0.9 \\ &= 0.77 \end{aligned} \tag{6.23}$$

Sekarang kita memiliki 2 buah fakta baru yang memberikan konfirmasi tentang E2 (Saya dalam suasana hati tidak enak), kombinasi dari kedua buah fakta tersebut adalah:

$$\begin{aligned} \text{CF(E2)} &= \text{CF(E2,E1)} + \text{CF(E2,E3 AND E4)} * (1 - \text{CF(E2,E1)}) \\ &= 0.86 + 0.77 * (1 - 0.86) \\ &= 0.97 \end{aligned} \tag{6.24}$$

STEP 3

Kembali ke R1, maka nilai CF untuk H1 jika diberikan E1 OR E2 adalah:

$$\begin{aligned} \text{CF(H1,E1 OR E2)} &= max\{\text{CF}_{\text{E1}}, \text{CF}_{\text{E2}}\} * \text{CF}_{\text{R1}} \\ &= max\{0.9, 0.97\} * 0.9 \\ &= 0.87 \end{aligned} \tag{6.25}$$

Yang berarti bahwa: Saya seharusnya tidak pergi bermain bola.

6.4 SOAL LATIHAN:

Dengan menggunakan seperangkat rule dan fakta dibawah ini, hitunglah kemungkinan pencuri mobil dari Tim. Mike atau John yang layak di-

tuduh sebagai pencuri?

Diketahui fakta-fakta sebagai berikut:

- Mobil dari Mike mogok (berarti dia membutuhkan transportasi) (CF=1.0).
- Mobil dari John tidak mogok (berarti dia tidak membutuhkan transportasi) (CF=1.0).
- Sidik jari dari Mike ada pada mobil (CF=1.0).
- Sidik jari dari John tidak ada pada mobil (CF=1.0).
- Sidik jari dari Mike tidak ada pada kunci (CF=1.0).
- Sidik jari dari John ada pada kunci (CF=1.0).
- Kunci mobil dari Tim tertinggal dalam mobil (CF=1.0).
- Mike tidak menyukai Tim (CF=0.6).
- John menyukai Tim (CF=0.8).
- Mike sedang melihat televisi ketika pencurian terjadi (berarti dia memiliki alibi) (CF=0.85).
- John sedang tidur ketika pencurian terjadi (berarti dia memiliki alibi) (CF=0.2)

R1	IF	Tertuduh memiliki motif,
	AND	Tertuduh memiliki kesempatan
	THEN	Tertuduh bersalah karena melakukan kejahatan. (CF = 0.6)
R2	IF	Tertuduh memiliki alibi,
	THEN	Tertuduh bersalah. (CF = -0.8)
R3	IF	Sidik jari dari tertuduh ditemukan pada mobil,
	THEN	Tertuduh bersalah. (CF = 0.4)
R4	IF	Kunci tertinggal di dalam mobil,
	THEN	Tertuduh memiliki kesempatan. (CF = 0.9)
R5	IF	Tertuduh tidak menyukai Tim,
	THEN	Tertuduh memiliki motif. (CF = 0.5)
R6	IF	Tertuduh membutuhkan transportasi,
	THEN	Tertuduh memiliki motif. (CF = 0.9)
R7	IF	Sidik jari dari tertuduh ditemukan pada kunci,
	THEN	Tertuduh bersalah. (CF = 0.7)

Chapter 7

Sistem Fuzzy

Tujuan Instruksional Khusus

- Mahasiswa mampu memformulasikan permasalahan yang mengandung fakta dengan derajad ketidakpastian tertentu ke dalam pendekatan Sistem Fuzzy.
- Mahasiswa mampu melakukan perhitungan secara manual dan analisis untuk melakukan inferensi dan defuzzifikasi pada Sistem Fuzzy.
- Mahasiswa mampu merancang perangkat lunak untuk penyelesaian masalah dengan menggunakan *Fuzzy System*.

7.1 Pendahuluan

Note:

- Manusia cenderung menggunakan bahasa dalam bentuk sesuatu yang dapat dipahami secara umum, bukan dalam bentuk bahasa matematika yang mementingkan akurasi. Misalkan, kita mengatakan: "Benda itu *sangat* berat" ketimbang "Benda itu beratnya 1500 kg."

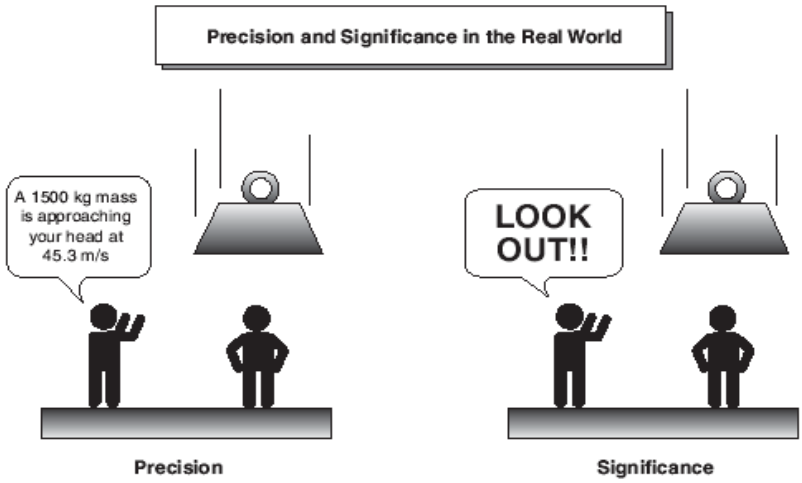

Figure 7.1: Ilustrasi fuzzy.

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

- Fuzzy system adalah suatu sistem yang menggunakan *himpunan fuzzy* untuk memetakan suatu inputan menjadi ouput tertentu (black box). Misalnya, jika anda mengetahui seberapa layanan pada restaurant tersebut, anda dapat menentukan berapa jumlah tip yang layak diberikan kepada pelayan. Perhatikan gambar di bawah ini:

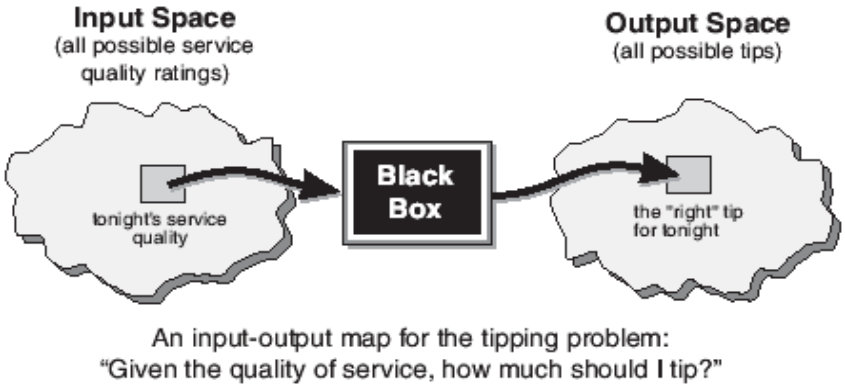

Figure 7.2: Pemetaan input output dengan fuzzy.

7.2 Variabel Linguistik (Linguistic Variable)

Note:

Istilah-istilah yang merepresentasikan fakta yang samar seperti pada contoh diatas disebut sebagai *variabel linguistik*. Tabel 7.1 menunjukkan contoh-contoh lain dari variabel linguistik beserta nilai tipikal yang mungkin.

Table 7.1: Contoh variabel linguistik

Variabel Linguistik	*Nilai Tipikal*
suhu	panas, dingin
ketinggian	pendek, cukup, tinggi
kelajuan	sangat lambat, lambat, cepat

Dalam sistem pakar fuzzy (fuzzy expert system), variabel linguistik digunakan pada aturan-aturan fuzzy (fuzzy rules). Perhatikan contoh di bawah ini:

R1	IF	Kelajuan rendah
	THEN	Buat akselerasi menjadi tinggi

R2	IF	Suhu udara rendah
	AND	Tekanan cukup
	THEN	Buat kelajuan menjadi rendah

Jangkauan (range) nilai yang mungkin dalam variabel linguistik disebut sebagai *universe of discourse*. Sebagai contoh "kelajuan" dalam R1 dapat memiliki range antara 0 sampai 200 km/jam. Karena "kelajuan rendah" menempati sebagian segmen dari universe of discourse.

7.3 Himpunan Fuzzy (Fuzzy Set)

Note:

Himpunan fuzzy berbeda dengan himpunan klasik. Himpunan klasik memiliki batasan yang jelas (crisp set), karena itu keanggotaan dari himpunan klasik dapat dinyatakan hanya dalam dua macam yaitu: menjadi anggota himpunan atau tidak. Sedangkan pada himpunan fuzzy, keanggotaan suatu elemen pada suatu himpunan lebih lanjut dinyatakan dengan derajad keanggotaannya. Perhatikan Gambar 7.3.

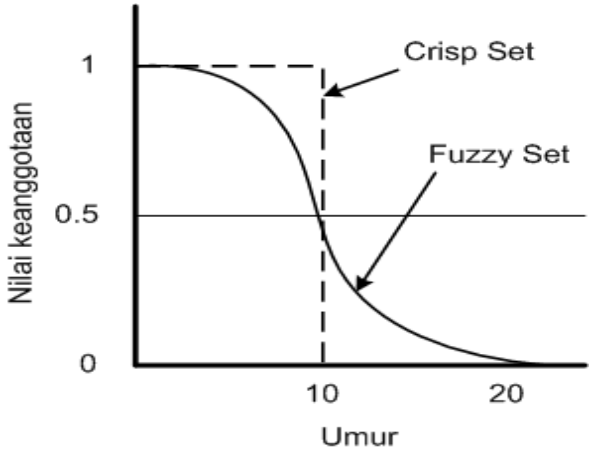

Figure 7.3: Ilustrasi fuzzy dan crisp set.

Gambar 7.3 merupakan himpunan orang muda. Representasi dengan crisp set menyatakan bahwa jika seseorang berumur dibawah 10 th maka ia merupakan himpunan orang muda, jika tidak maka ia tergolong tua. Sebaliknya dengan menggunakan fuzzy set, himpunan orang muda ditentukan oleh derajad keanggotaannya. Misalnya, seseorang berumur 2 th merupakan himpunan orang muda dengan nilai keanggotaan 0.95, atau dengan kata lain *sangat muda*, orang berumur 10 th merupakan himpunan orang muda dengan nilai keanggotaan 0.5, sedangkan orang berumur 18 th merupakan himpunan orang muda dengan nilai keanggotaan 0.2, atau dengan kata lain *kurang muda*.

Kurva pada Gambar 7.3 juga dapat ditafsirkan sebagai mapping dari input berupa umur seseorang menuju ke output berupa *derajad kemu-*

daan seseorang. Secara khusus kurva semacam ini disebut sebagai *fungsi keanggotaan (membership function)*.

Contoh lain yang menggambarkan fuzzy set adalah himpunan fuzzy tentang musim di belahan bumi utara, seperti ditunjukkan dalam Gambar 7.4.

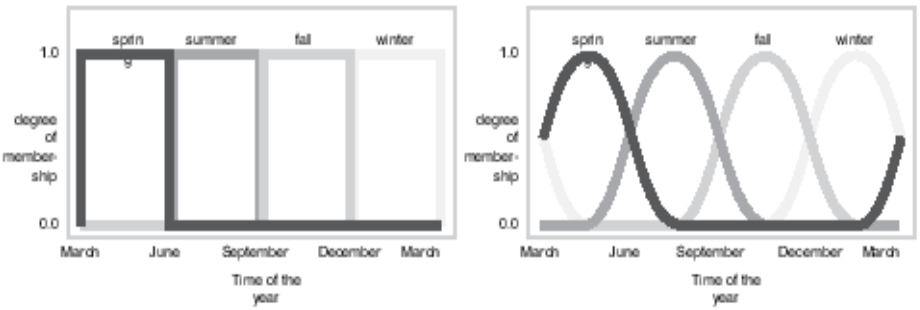

Figure 7.4: Fuzzy set musim di belahan bumi utara.

Definisi secara astronomis tentang musim adalah seperti ditunjukkan pada gambar sebelah kiri dimana seharusnya terdapat perbedaan drastis pada setiap musim, misalnya spring persis dimulai pada bulan March dan selesai pada bulan June. Akan tetapi setiap negara di belahan bumi utara mungkin merasakan perbedaan musim tersebut berbeda-beda, misalnya spring mungkin dimulai pada minggu kedua atau ketiga pada bulan March dan berakhir pada minggu pertama atau kedua pada bulan June, seperti digambarkan pada bagian kanan Gambar 7.4. Negara lain mungkin mengalami situasi yang berbeda.

Gambar 7.4 juga menunjukkan bahwa dalam fuzzy set musim di belahan bumi utara terdapat beberapa fuzzy subset, yaitu: spring, summer, fall dan winter. Perhatikan juga bahwa minggu kedua April (perkiraan pada gambar) merupakan anggota dari *spring* dengan nilai keanggotaan 0.8 dan merupakan anggota dari *summer* dengan nilai keanggotaan 0.2. Jadi, suatu input dapat dianggap memiliki keanggotaan parsial pada beberapa fungsi keanggotaan. Implikasi dari hal ini akan dibahas pada

sub-bab berikutnya.

Dengan berlandaskan contoh-contoh di atas, definisi akurat dari fuzzy set adalah sebagai berikut:

> *Fuzzy Set:* Jika X adalah universe of discourse, elemen dari X dinotasikan sebagai x. Sebuah himpunan fuzzy A dari X dikarakteristikkan oleh fungsi keanggotaan $\mu_A(x)$ yang mengasosiasikan setiap elemen x dengan nilai dari derajad keanggotaan dalam A.

7.4 Fungsi Keanggotaan (Membership Function)

Note:

Membership Function (MF) adalah kurva yang memetakan setiap titik pada inputan (universe of discourse) ke sebuah nilai keanggotaan (atau derajad keanggotaan) yang memiliki nilai antara 0 dan 1 yang didefinisikan secara matematis oleh persamaan:

$$\mu_A(x) : X \rightarrow [0, 1] \tag{7.1}$$

Setiap elemen x dipetakan pada sebuah nilai keanggotaan oleh MF. Nilai ini merupakan derajad keanggotaan dari x pada himpunan fuzzy A.

$$\mu_A(x) = \text{Degree}(x \in A) \tag{7.2}$$

Dimana nilai keangotaan dari x dibatasi oleh:

$$0 \leq \mu_A(x) \leq 1$$

Sebagai contoh pemetaan elemen dari x (dalam hal ini adalah ketinggian seseorang) oleh MF ke nilai keanggotaannya, ditunjukkan dalam Gambar 7.5.

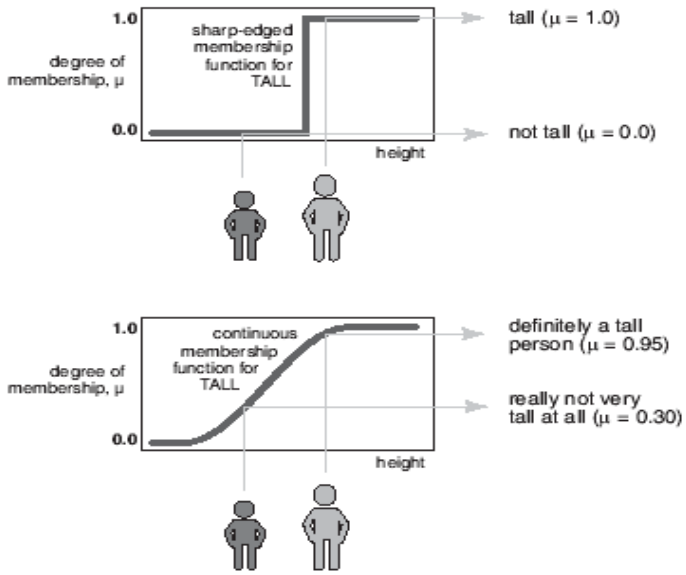

Figure 7.5: Pemetaan ketinggian seseorang oleh MF.

Nilai keanggotaan hasil pemetaan elemen x oleh suatu fungsi keanggotaan dapat memiliki nilai yang berbeda-beda tergantung pada jenis dari fungsi keanggotaan. Fungsi keanggotaan yang umum digunakan adalah: fungsi segitiga, fungsi trapesium, fungsi gaussian, fungsi bell dan fungsi sigmoid. Bentuk dari masing-masing fungsi keanggotaan adalah sebagai berikut:

1. *Fungsi segitiga.* Fungsi keanggotaan berbentuk segitiga didefinisikan oleh 3 parameter a, b, c dengan persamaan:

$$\text{Segitiga}(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \tag{7.3}$$

   Fungsi segitiga dengan parameter: segitiga(x;0.2,0.6,0.8) ditunjukkan dalam Gambar 7.6.

2. *Fungsi Trapesium.* Fungsi keanggotaan berbentuk trapesium didefinisikan oleh 4 parameter a, b, c, d dengan persamaan:

$$\text{Trapesium}(x; a, b, c, d) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \tag{7.4}$$

   Fungsi segitiga dengan parameter: trapesium(x;0.1,0.2,0.6,0.95) ditunjukkan dalam Gambar 7.7.

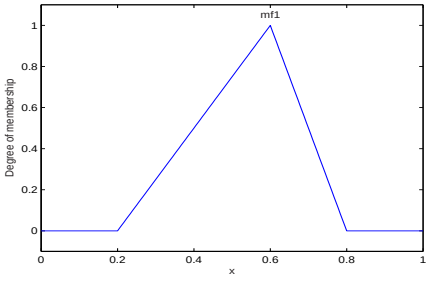

Figure 7.6: Fungsi keanggotaan segitiga (triangle).

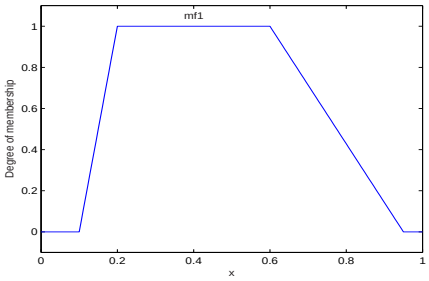

Figure 7.7: Fungsi keanggotaan trapesium (trapezoidal).

3. *Fungsi Gaussian.* Fungsi keanggotaan berbentuk gaussian didefinisikan oleh 2 parameter σ, dan c dengan persamaan:

$$\text{Gaussian}(x; \sigma, c) = e^{-\left(\frac{x-c}{\sigma}\right)^2} \tag{7.5}$$

Fungsi Gaussian dengan parameter: gaussian(x;0.15,0.5) ditunjukkan dalam Gambar 7.8.

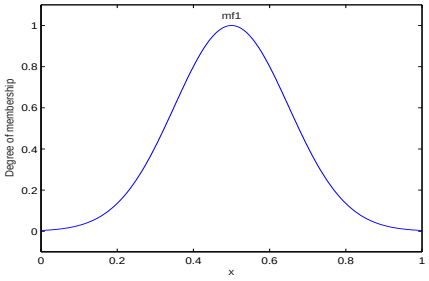

Figure 7.8: Fungsi keanggotaan Gaussian. σ= standar deviasi, c=pusat.

4. *Fungsi Bell.* Fungsi keanggotaan berbentuk bell didefinisikan oleh

3 parameter a, b dan c dengan persamaan:

$$\text{Bell}(x; a, b, c) = \frac{1}{1 + \left|\frac{x-c}{a}\right|^{2b}} \tag{7.6}$$

Fungsi Bell dengan parameter: bell(x;0.25,2.5,0.5) ditunjukkan dalam Gambar 7.9. Sedangkan parameter a, b dan c yang menspesi-

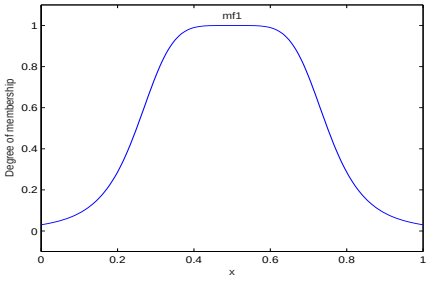

Figure 7.9: Fungsi keanggotaan Bell.

fikasikan fungsi Bell ditunjukkan dalam Gambar 7.10.

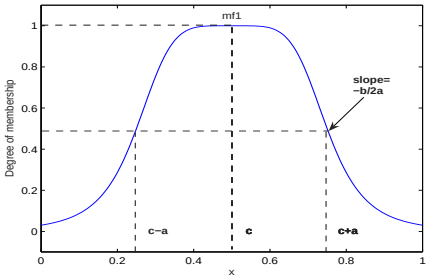

Figure 7.10: Letak parameter a,b dan c pada fungsi keanggotaan Bell.

5. *Fungsi Sigmoid.* Fungsi keanggotaan Sigmoid didefinisikan oleh 2 parameter a dan c dengan persamaan:

$$\text{Sigmoid}(x; a, c) = \frac{1}{1 + e^{-a(x-c)}} \tag{7.7}$$

Jika nilai $a > 0$, maka fungsi sigmoid akan membuka ke kanan, sedang jika $a < 0$ maka fungsi sigmoid akan membuka ke kiri. Fungsi Sigmoid membuka ke kanan dengan parameter: sigmoid(x;12,0.25) ditunjukkan dalam Gambar 7.11. Sedangkan fungsi Sigmoid mem-

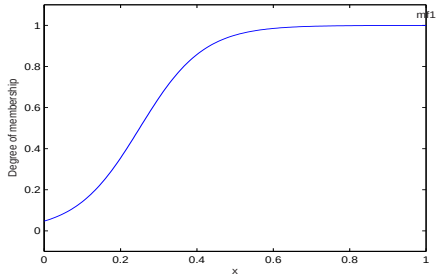

Figure 7.11: Fungsi keanggotaan Sigmoid membuka ke kanan.

buka ke kiri dengan parameter: sigmoid(x;-12,0.75) ditunjukkan dalam Gambar 7.12.

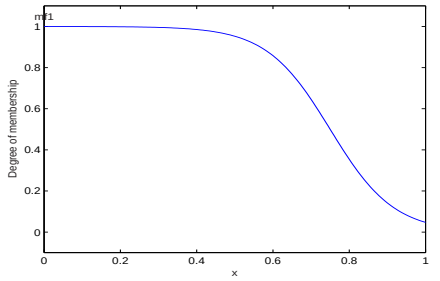

Figure 7.12: Fungsi keanggotaan Sigmoid membuka ke kiri.

7.5 Representasi Himpunan Fuzzy

Seperti dijelaskan di atas bahwa jika kita memiliki universe of discourse X dengan himpunan fuzzy A di dalamnya, maka himpunan elemen dari X yang dinyatakan sebagai $\{x_1, x_2, \ldots, x_n\}$ dapat dipetakan oleh fungsi keanggotaan $\mu_A(x)$ ke dalam nilai keanggotaan masing-masing dengan nilai [0,1]. Misalkan sebuah himpunan fuzzy dinyatakan sebagai:

$$A = (a_1, a_2, \ldots, a_n) \tag{7.8}$$

dimana

$$a_i = \mu_A(x_i). \tag{7.9}$$

Dengan cara yang lebih kompak, himpunan fuzzy A dapat direpresentasikan sebagai:

$$A = (a_1/x_1, a_2/x_2, \ldots, a_n/x_n) \tag{7.10}$$

dimana simbol "/" merupakan pemetaan dari input x_i ke suatu nilai keanggotaan a_i. Misalnya:

$$\text{TINGGI} = (0/125, 0.25/140, 0.7/150, 1/170, 1/190)$$

7.6 Operasi Himpunan Fuzzy

Note:

Operasi-operasi pada himpunan fuzzy pada dasarnya mirip dengan operasi pada himpunan klasik. Sebagai contoh perhatikan operasi himpunan fuzzy pada Gambar 7.13, dimana operasi logika AND diganti dengan *min* sedangkan operasi logika OR diganti dengan *max*. Terlihat bahwa himpunan fuzzy (yang hanya memiliki anggota 0 dan 1) memberikan hasil yang sama dengan operasi pada himpunan klasik. Dengan demikian, operasi ini dapat diperluas untuk himpunan fuzzy yang memiliki nilai keanggotaan [0,1].

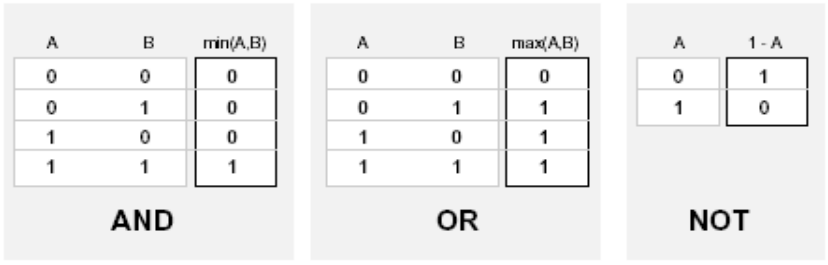

Figure 7.13: Operasi himpunan fuzzy.

AND

Operasi AND antara dua buah himpunan fuzzy A dan B akan meng-

hasilkan interseksi antara A dan B pada X yang didefinisikan sebagai:

$$\begin{aligned}\mu_{A\wedge B}(X) &= \min(\mu_A(x), \mu_B(x)) \quad \text{untuk semua} x \in X \\ &= \mu_A(x) \wedge \mu_B(x) \\ &= \mu_A(x) \cap \mu_B(x). \end{aligned} \tag{7.11}$$

Sebagai contoh perhatikan dua buah himpunan fuzzy di bawah ini:

$$\begin{aligned}\text{TINGGI} &= (0/125, 0.2/140, 0.5/150, 0.8/170, 1/190) \\ \text{PENDEK} &= (1/125, 0.8/140, 0.5/150, 0.2/170, 0/190)\end{aligned}$$

hasil operasi AND pada kedua himpunan fuzzy di atas:

$$\mu_{\text{TINGGI}} \wedge \mu_{\text{PENDEK}}(x) = (0/125, 0.2/140, 0.5/150, 0.2/170, 0/190)$$

OR

Operasi OR antara dua buah himpunan fuzzy A dan B akan menghasilkan gabungan antara A dan B pada X yang didefinisikan sebagai:

$$\begin{aligned}\mu_{A\vee B}(X) &= \max(\mu_A(x), \mu_B(x)) \quad \text{untuk semua} x \in X \\ &= \mu_A(x) \vee \mu_B(x) \\ &= \mu_A(x) \cup \mu_B(x). \end{aligned} \tag{7.12}$$

Kembali pada contoh di atas, maka hasil operasi OR pada himpunan fuzzy TINGGI dan PENDEK adalah:

$$\mu_{\text{TINGGI}} \vee \mu_{\text{PENDEK}}(x) = (1/125, 0.8/140, 0.5/150, 0.8/170, 1/190)$$

NOT

Operasi NOT pada himpunan fuzzy A akan memberikan hasil komplemen dari A, yaitu:

$$\mu_{\sim A}(x) = 1 - \mu_A(x) \tag{7.13}$$

Maka operasi NOT untuk himpunan fuzzy TINGGI pada contoh terdahulu akan menghasilkan himpunan komplemen dari TINGGI:

$$\mu_{\text{NOT TINGGI}}(x) = (1/125, 0.8/140, 0.5/150, 0.2/170, 0/190)$$

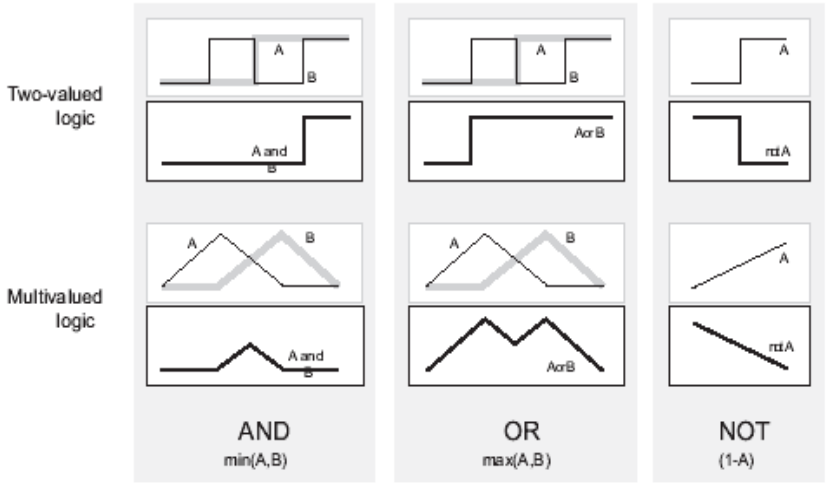

Figure 7.14: Operasi himpunan fuzzy dengan diagram.

Representasi operasi himpunan fuzzy dalam bentuk diagram diperlihatkan pada Gambar 7.14.

Contoh

1. Himpunan fuzzy tentang suhu udara di Surabaya (dengan universe of discourse antara 20 sampai 40 derajad celcius) dispesifikasikan ke dalam tiga subset yaitu: rendah, sedang dan tinggi. Fungsi keanggotaan dari masing-masing subset dengan parameternya ditentukan pada tabel di bawah. Gambarkan himpunan fuzzy tersebut!

Subset	Fungsi Keanggotaan	Parameter
rendah	Trapesium	a=20, b=20, c=25, d=30
sedang	Segitiga	a=25, b=30, c=35
tinggi	Trapesium	a=30, b=35, c=40, d=40

2. Jika diketahui dua buah himpunan fuzzy SUHU dan KELEMBA-

BAN seperti dibawah ini:

$$\begin{aligned} \text{SUHU} &= (0/20, 0.1/22, 0.2/24, 0.8/30, 1/35) \\ \text{KELEMBABAN} &= (0/10, 0.5/40, 0.5/50, 0.5/60, 0/90) \end{aligned}$$

Bagaimanakah hasil operasi AND, OR dan NOT untuk kedua himpunan fuzzy di atas?

7.7 Fuzzy Inference

Note:

Seperti pada halnya sistem berbasis aturan, sistem fuzzy juga memiliki dua buah proposisi. Kedua buah proposisi tersebut dihubungkan oleh fuzzy rule seperti berikut:

```
IF X is A THEN Y is B
```

dimana A dan B adalah himpunan fuzzy yang memiliki universe of discourse terletak pada X.

Fuzzy inference dalam hal ini mencoba untuk menentukan derajad kepercayaan dari konklusi (Y is B) jika diketahui fakta/bukti/premise tertentu (Y is A). Secara umum terdapat dua macam sistem inferensi fuzzy yang digunakan secara luas di dunia, yaitu *Mamdani* dan *Sugeno*. Sistem inferensi fuzzy yang digunakan pada seluruh bab ini adalah sistem Mamdani. Apabila mahasiswa tertarik untuk mempelajari sistem Sugeno dipersilahkan mengacu pada buku-buku literatur.

Berdasarkan operasi antar himpunan fuzzy, metode inferensi fuzzy juga dapat dibagi lagi menjadi dua macam, yaitu: *Max-Min inference* dan *Max-Product inference*.

7.7.1 Max-Min Inference

Max-Min Inference melakukan proses inferensi dengan cara melakukan operasi AND pada kedua himpunan fuzzy (A AND B). Dirumuskan sebagai berikut:

$$\begin{aligned} B_{\text{inf}} &= \mu_A(x_k) \wedge \mu_B(y) \\ &= \min(\mu_A(x_k), \mu_B(y)). \end{aligned} \tag{7.14}$$

Sebagai contoh, perhatikan fuzzy rule di bawah ini:

```
IF Inflasi normal THEN Suku bunga bank medium
```

dengan fungsi keanggotaan sebagai berikut:

$$\begin{aligned} \text{Inflasi normal} &= (0/0.3, 0.5/0.4, 1/0.50, 0.5/0.6, 0/0.7) \\ \text{Bunga bank medium} &= (0/10, 0.6/20, 1/30, 0.6/40, 0/50) \end{aligned}$$

Diketahui bahwa pada saat ini angka inflasi di suatu negara saat ini adalah 40%, sehingga fungsi keanggotaan akan memetakan nilai tersebut menjadi derajad keanggotaan dengan nilai $\mu_A = 0.5$, maka nilai dari B_{inf} adalah:

$$\begin{aligned} B_{inf} &= [\min(0.5, 0), \min(0.5, 0.6), \min(0.5, 1), \min(0.5, 0.6), \min(0.5, 0)] \\ &= (0, 0.5, 0.5, 0.5, 0) \end{aligned}$$

Proses Max-Min inference dapat digambarkan seperti dalam Gambar 7.15.

7.7.2 Max-Product Inference

Max-Product Inference melakukan proses inferensi dengan cara melakukan operasi perkalian sebagai berikut:

$$B_{\text{inf}} = \mu_A * \mu_B(y). \tag{7.15}$$

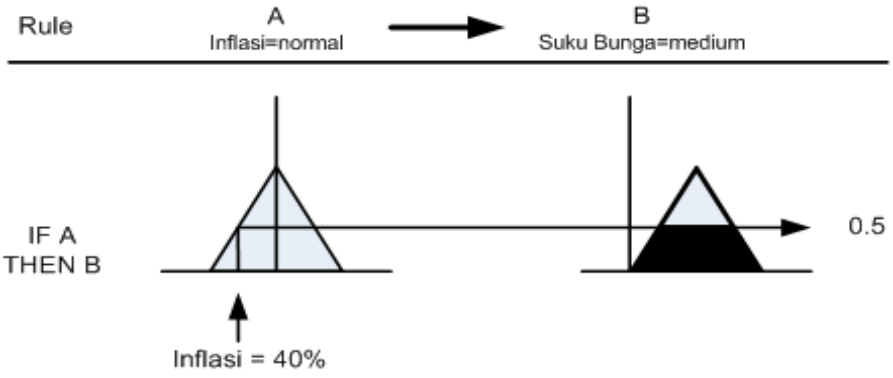

Figure 7.15: Max-Min Inference.

Kembali pada contoh diatas, maka hasil Max-Product inference adalah sebagai berikut:

$$\begin{aligned} B_{inf} &= \mu_A * (0/10, 0.6/20, 1/30, 0.6/40, 0/50) \\ &= 0.5 * (0, 0.6, 1, 0.6, 0) \\ &= (0, 0.3, 0.5, 0.3, 0) \end{aligned}$$

Proses Max-Product inference dapat digambarkan seperti dalam Gambar 7.16.

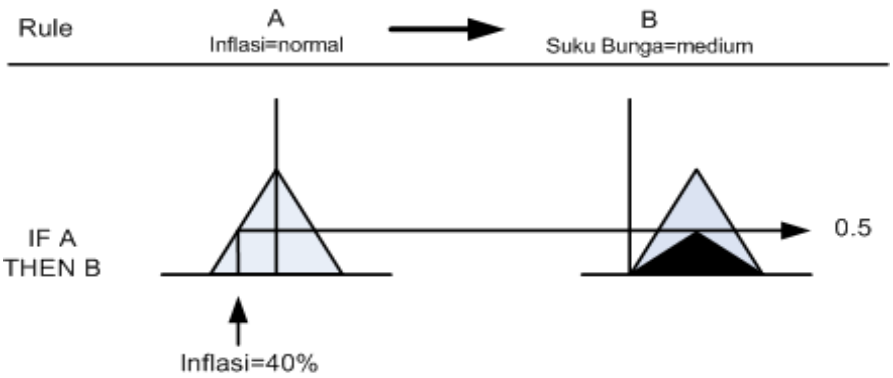

Figure 7.16: Max-Product Inference.

7.8 Fuzzy Inference dengan Beberapa Rule

Note:

Pada sub-bab terdahulu, kita hanya membicarakan proses inferensi pada fuzzy rule yang hanya memiliki sebuah rule. Pada kebanyakan situasi nyata, sistem fuzzy dapat diperluas hingga memiliki beberapa rule seperti pada contoh berikut ini:

R1	IF	Inflasi normal
	OR	Suku bunga rendah
	THEN	Investasi rendah

R2	IF	Inflasi normal
	AND	Suku bunga normal
	THEN	Investasi medium

Maka proses inferensi dari kedua rule dan kedua premise tersebut dapat ditunjukkan dalam Gambar 7.17 untuk Max-min inference dan Gambar 7.18 untuk Max-product inference.

Dari Gambar 7.17 dan Gambar 7.18 terlihat jelas bahwa apabila terdapat beberapa rule yang berperan dalam pembentukan sistem fuzzy (pada kenyataannya memang demikian), maka untuk mendapatkan satu nilai output terdapat proses penggabungan himpunan fuzzy (hasil dari inferensi) rule-rule tersebut. Proses ini dikenal sebagai *agregasi*.

7.9 If-Then Rules

Note:

Sebagaimana halnya pada sistem berbasis aturan (rule-based system), fuzzy system juga menggunakan aturan If-Then untuk memformulasikan 'conditional statement'.

Sebuah aturan If-Then dalam fuzzy memiliki bentuk:

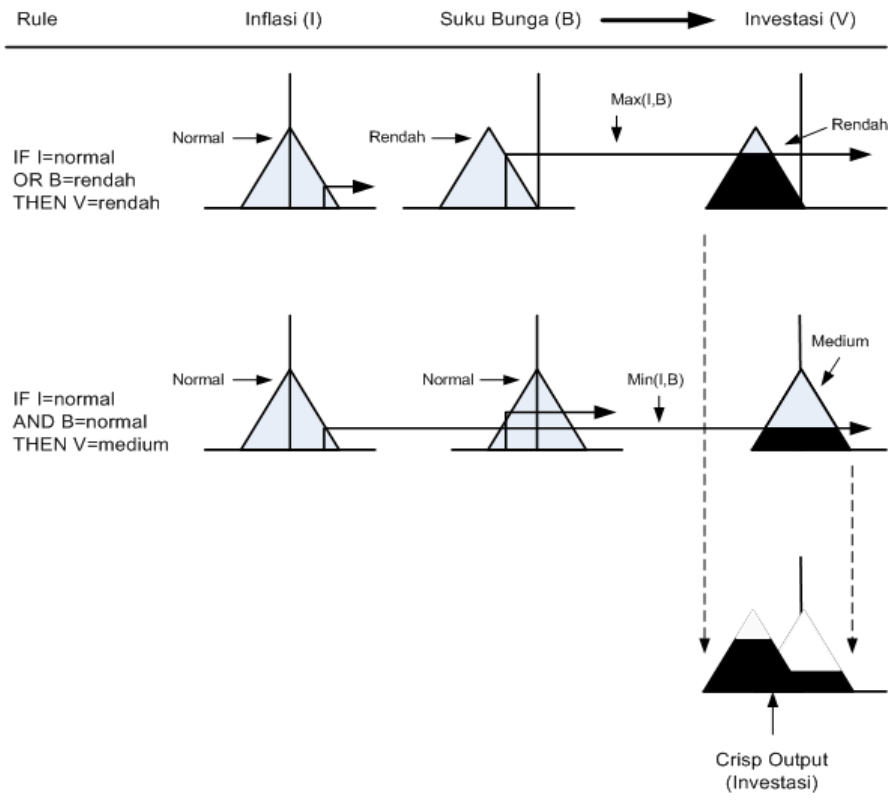

Figure 7.17: Max-min Inference.

```
If x adalah A Then Y adalah B
```

dimana A dan B adalah nilai linguistik yang didefinisikan oleh himpunan fuzzy pada universe of discourse X dan Y. Bagian If (x adalah A) disebut sebagai *antecedent* atau *premise*, sedangkan bagian Then (y adalah B) disebut sebagai *consequent* atau *conclusion*. Contoh:

```
If service is good Then tip is average
```

Bagian antecedent juga dapat memiliki beberapa bagian, misalnya:

```
If service is excellent or food is delicious
Then tip is generous
```

Interpretasi dari If-Then rule dapat dikategorikan ke dalam tiga proses, yaitu:

1. *Fuzzifikasi input:* mentransformasi semua antecedent ke dalam derajad keanggotaan yang memiliki nilai antara 0 dan 1.

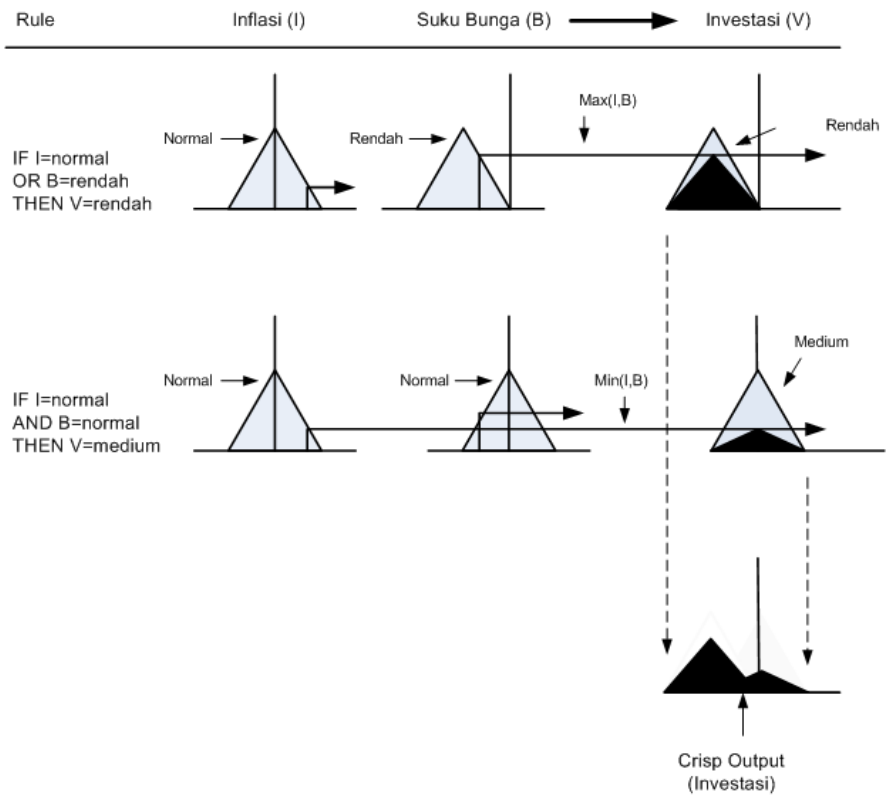

Figure 7.18: Max-product Inference.

2. *Mengaplikasikan operator fuzzy jika terdapat beberapa input:* jika terdapat beberapa bagian antecedent, maka operasikan operator-operator fuzzy untuk menghasilkan satu nilai antecedent yang terletak antara 0 dan 1.

3. *Implikasi:* proses implikasi diterapkan untuk menghasilkan nilai keluaran (inferensi). Bagian consequent dari If-Then rule ini memetakan semua himpunan fuzzy pada keluaran.

Ketiga proses tersebut di atas ditunjukkan dalam Gambar 7.19. Terlihat pada bagian akhir dari gambar bahwa proses inferensi menggunakan metoda Max-min inference.

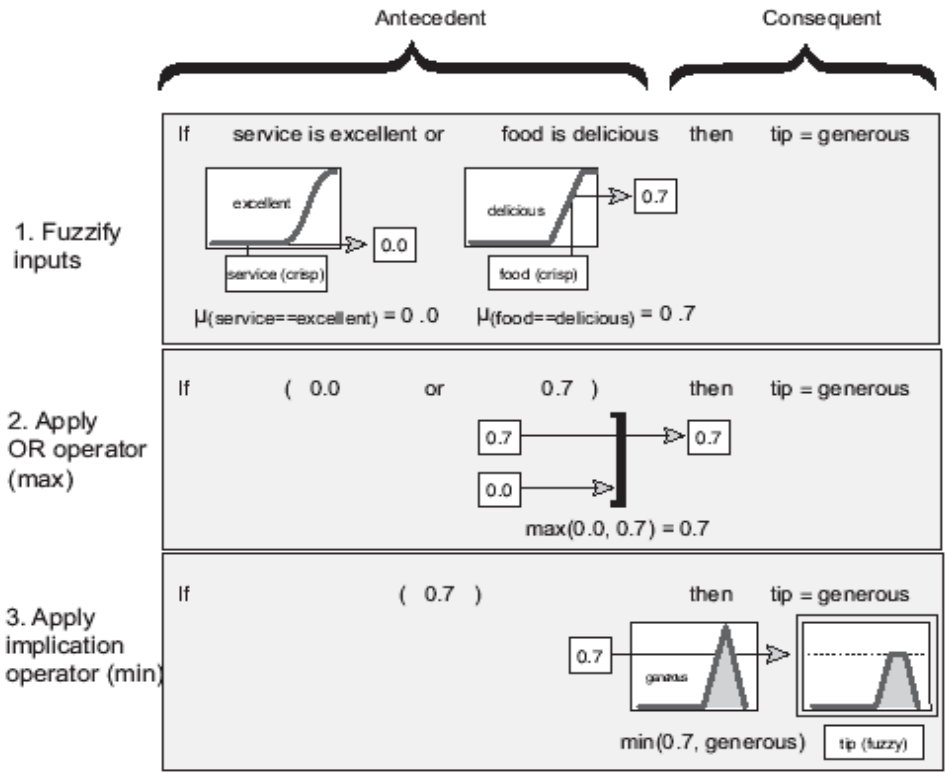

Figure 7.19: Fuzzifikasi, operasi fuzzy dan implikasi.

7.10 Defuzzifikasi

Note:

Proses terakhir yang harus dilakukan dari sebuah sistem fuzzy adalah proses defuzzifikasi, yaitu proses untuk mentransformasikan kembali dari himpunan fuzzy pada bagian konklusi menjadi sebuah bilangan keluaran (crisp output).

Proses defuzzifikasi dilakukan dengan berbagai macam metode. Misalnya, maximum defuzzification, centroid (center of gravity) defuzzification, weighted average defuzzification, dll. Pada bab ini hanya akan dibahas dua metode pertama yang telah disebutkan di atas.

7.10.1 Maximum Defuzzification

Defuzzification dengan menggunakan metoda ini dilakukan dengan cara mengambil nilai maximum dari fungsi keanggotaan. Secara aljabar, maximum defuzzification dapat dirumuskan sebagai:

$$\mu_A(x^*) \geq \mu_A(x) \quad \text{untuk semua } x \in X \tag{7.16}$$

dimana x^* adalah nilai yang terdefuzzifikasi. Perhatikan Gambar 7.20. Pada gambar terlihat bahwa metode ini lebih lanjut terdiri atas tiga macam yaitu: smallest of maxima, mean of maxima dan largest of maxima tergantung pada posisi dimana nilai x^* diambil.

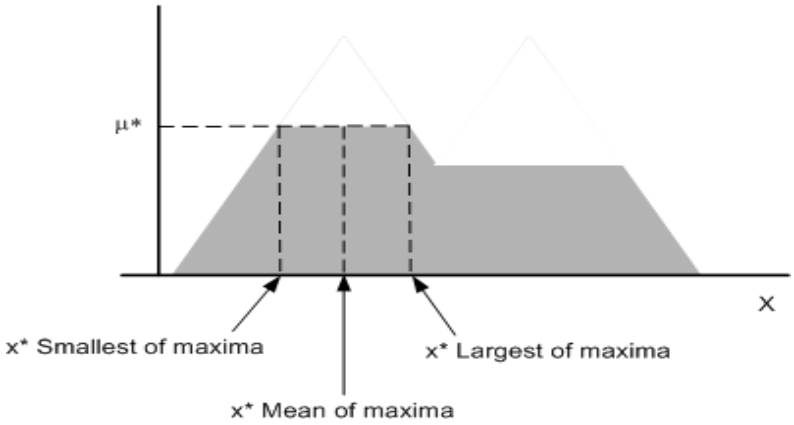

Figure 7.20: Maximum defuzzification.

Metode maximum deffuzzification sangat mudah diimplementasikan, tetapi kurang akurat.

7.10.2 Centroid Defuzzification

Metode centroid defuzzification adalah metode defuzzifikasi yang paling umum digunakan dan memberikan hasil yang sangat akurat. Metode ini dirumuskan sebagai berikut:

$$x^* = \frac{\sum_{i=1}^{N} x_i \mu_A(x_i)}{\sum_{i=1}^{N} \mu_A(x_i)} \tag{7.17}$$

dimana x^* adalah nilai yang terdefuzzifikasi, N adalah jumlah sample yang diambil, x_i nilai keluaran ke-i, dan $\mu_A(x_i)$ adalah fungsi keanggotaan teragregasi.

Perhatikan contoh pada Gambar 7.21 di bawah ini:

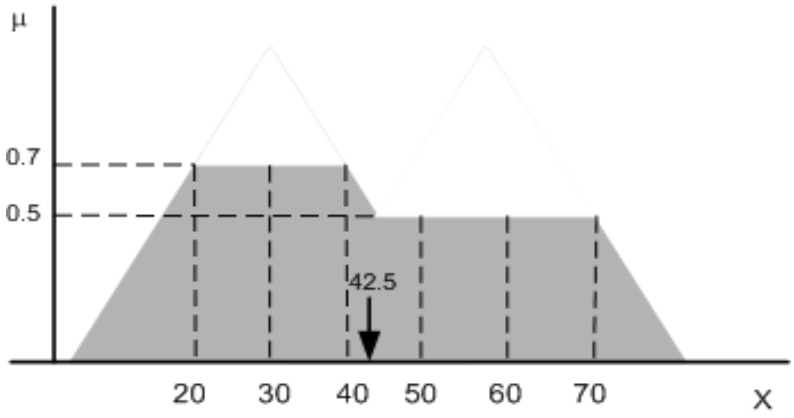

Figure 7.21: Centroid defuzzification.

Berdasarkan dari gambar di atas, nilai dari x^* dapat dihitung:

$$\begin{aligned} x^* &= \frac{0.7 \times (20 + 30 + 40) + 0.5 \times (50 + 60 + 70)}{0.7 + 0.7 + 0.7 + 0.5 + 0.5 + 0.5} \\ &= \frac{153}{3.6} \\ &= 42.5 \end{aligned} \tag{7.18}$$

CONTOH KASUS 1

Pada contoh kasus pertama diberikan kondisi sebagai berikut. Untuk menentukan tip bagi pelayan restaurant digunakan dua buah inputan, yaitu service dan food. Dengan kata lain, jumlah tip yang akan diberikan kepada pelayan ditentukan dari kondisi service dan food. Perhatikan diagram pada Gambar 7.22 di bawah ini.

Terlihat pada gambar, input service dan food masing-masing memiliki range 0-10. Sedangkan output tip memiliki range antara 0-25%.

Langkah-langkah penyelesaian untuk contoh kasus di atas adalah sebagai berikut:

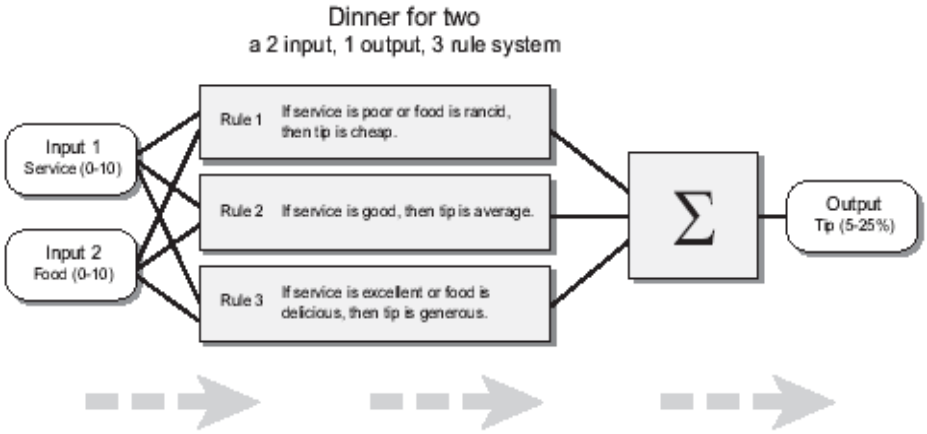

Figure 7.22: Contoh kasus 1: Menentukan tip.

1. *Fuzzifikasi input.* Proses pertama dalam sistem fuzzy adalah mentransformasi nilai crisp (mentah) menjadi fungsi keanggotaan melalui fungsi keanggotaan. Hal ini berarti bahwa fungsi keanggotaan untuk setiap himpunan fuzzy harus ditentukan terlebih dahulu. Selain itu, Sistem fuzzy memiliki 3 buah rule yang didefinisikan sebagai berikut:

   R1 IF Service is poor
   OR Food is rancid
   THEN Tip is cheap

   R2 IF Service is good
   THEN tip is average

   R3 IF Service is excellent
   OR Food is delicious
   THEN Tip is generous

   Misalkan sekarang kita tentukan bahwa kualitas dari food sangat enak sehingga kita memberikan nilai 8 untuk input makanan. Dengan menggunakan fungsi keanggotaan, nilai crisp utnuk input makanan tersebut akan ditransformasi menjadi nilai keanggotaan,

dalam hal ini kita dapatkan $\mu = 0.7$. Lihat Gambar 7.23.

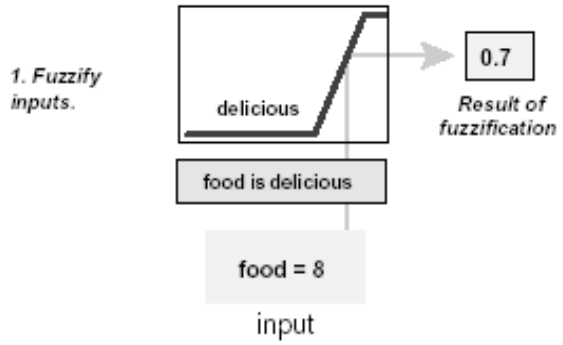

Figure 7.23: Fuzzifikasi input food untuk contoh kasus 1.

2. *Operator fuzzy.* Jika terdapat dua atau lebih premis pada setiap rule yang terlibat maka harus digunakan operasi fuzzy untuk setiap premise pada rule tersebut. Sebagai contoh pada Gambar 7.24 di bawah ini. Karena nilai keanggotaan service adalah dan nilai keanggotaan dari food adalah 0.7, maka hasil operasi OR atau *max* adalah 0.7.

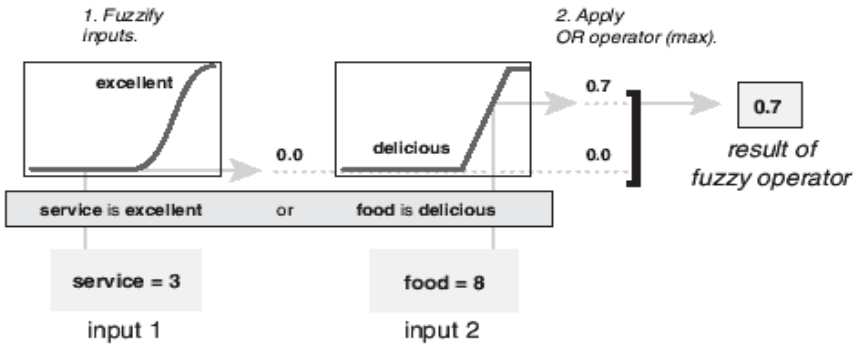

Figure 7.24: Operasi OR pada kasus 1.

3. *Inferensi fuzzy (Implikasi).* Proses selanjutnya adalah proses inferensi (implikasi) jika diketahui fakta (premise) untuk menghasilkan konklusi. Dalam contoh ini digunakan model inferensi Max-min. Hasil dari inferensi untuk rule pertama ditunjukkan pada Gambar 7.25.

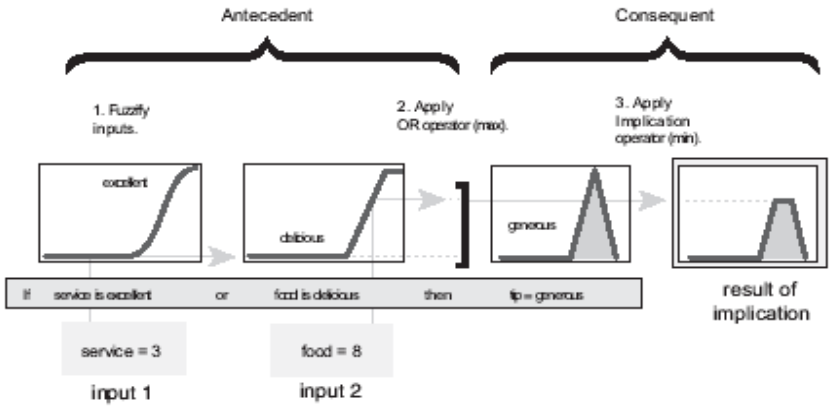

Figure 7.25: Inferensi fuzzy pada kasus 1.

4. *Agregasi semua keluaran.* Karena pada contoh kasus ini terdapat 3 buah rule terlibat dalam pengambilan keputusan, maka perlu dilakukan agregasi terhadap keluaran dari 3 buah rule tersebut. Proses agregasi ditunjukkan dalam Gambar 7.26. Agregasi dapat dilakukan dengan menggabungkan beberapa fungsi keanggotaan pada keluaran sistem fuzzy yang memiliki nilai keanggotaan maksimum (disebut metoda agregasi *Max*). Atau alternatif kedua dengan cara menjumlahkan semua fungsi keanggotaan pada keluaran sistem fuzzy (disebut metoda agregasi *Sum*). Seperti terlihat dalam Gambar, metoda agregasi yang digunakan adalah metoda Max.

5. *Defuzzifikasi.* Proses terakhir yang menjadi keluaran bagi sistem fuzzy adalah nilai mentah/crisp pada bagian keluaran. Dalam contoh kasus ini adalah nilai tip yang harus diberikan kepada pelayan restaurant. Proses mengtransformasi nilai fuzzy set ke dalam crisp set adalah defuzzifikasi (kebalikan dari proses fuzzifikasi). Dengan menggunakan metoda centroid, keluaran dari defuzzifikasi seperti ditunjukkan dalam Gambar 7.27. Yang berarti bahwa jumlah tip yang harus diberikan adalah 16.7%.

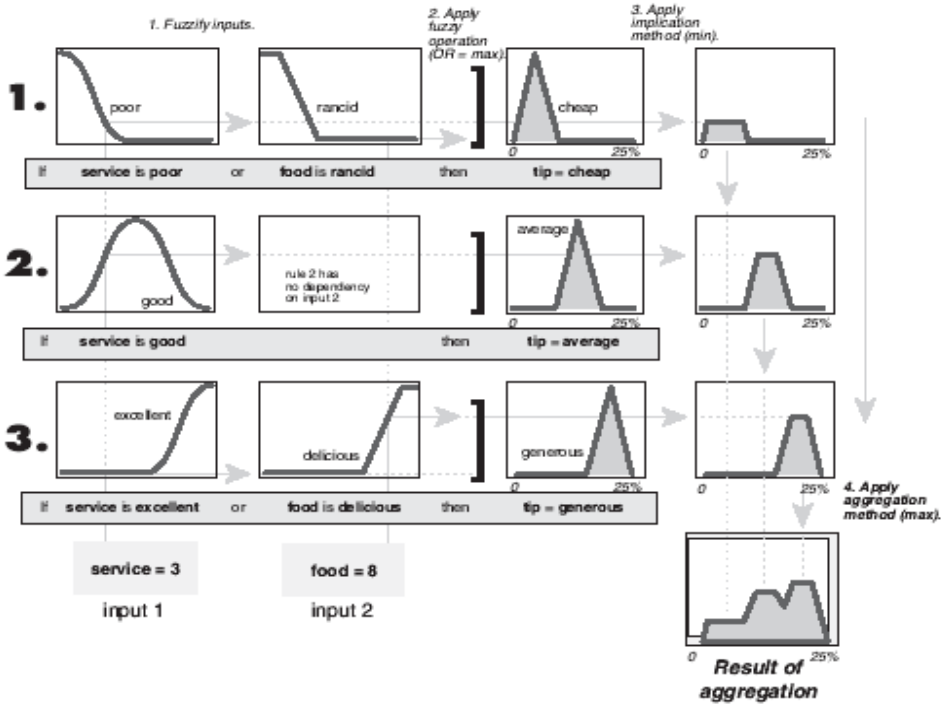

Figure 7.26: Proses agregasi pada kasus 1.

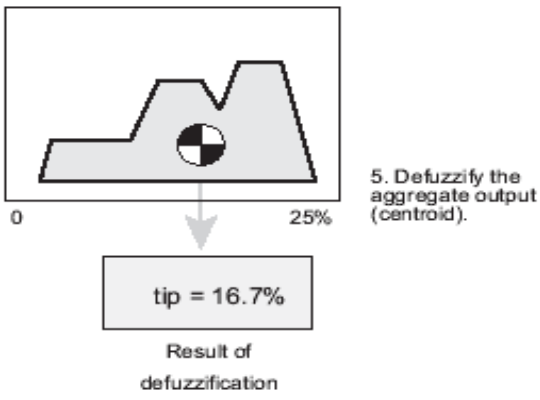

Figure 7.27: Defuzzifikasi pada kasus 1.

CONTOH KASUS 2

Sebuah perusahaan menggunakan sistem fuzzy untuk menentukan nilai kinerja dari karyawan. Masukan yang digunakan adalah Attitude dan nilai IQ. Adapun himpunan fuzzy dari masing-masing input dan output adalah sebagai berikut:

attitude(range: 0 - 10)

Subset	MF	Parameter
Buruk	segitiga	a=0; b=2.5; c=5
Normal	segitiga	a=2.5; b=5; c=7.5
Baik	segitiga	a=5; b=7.5; c=10

IQ(range: 80 - 160)

Subset	MF	Parameter
Rendah	segitiga	a=80; b=100; c=120
Sedang	segitiga	a=100; b=120; c=140
Tinggi	segitiga	a=120; b=140; c=160

nilai kinerja(range: 0 - 10)

Subset	MF	Parameter
Rendah	segitiga	a=0; b=2.5; c=5
Sedang	segitiga	a=2.5; b=5; c=7.5
Tinggi	segitiga	a=5; b=7.5; c=10

Diketahui rule-rule sebagai berikut:

R1	If	attitude buruk
	AND	IQ sedang
	THEN	nilai-kinerja rendah

R2	If	attitude normal
	AND	IQ rendah
	THEN	nilai-kinerja sedang

Jika seorang karyawan diketahui memiliki nilai attitude=4 dan nilai IQ=110, berapakah nilai kinerja kayawan tersebut (dengan menggunakan metoda deffuzifikasi centroid)?

CONTOH KASUS 3

Pada kasus ke-2 akan dilakukan kontrol terhadap Sprinkle System (Penyiram tanaman). Lihat Gambar 7.28. Sebagai inputan 1 digunakan suhu udara disekitar taman yang memiliki universe of discourse $18 - 40°C$, inputan 2 digunakan kelembaban tanah dengan universe of discourse 0-100%. Sedang output dari sistem ini adalah lama putaran dari sprinkle dengan universe of discourse 10 - 110 menit. Diagram sprinkle system yang digenerate dengan menggunakan MATLAB ditunjukkan dalam Gambar 7.29.

Proses awal dimulai dari desain terhadap himpunan fuzzy dari kedua input: suhu udara dan kelembaban tanah, serta himpunan fuzzy dari output yaitu lama putaran. Himpunan fuzzy dari input suhu udara ditunjukkan dalam Gambar 7.30. Spesifikasi parameter dari himpunan fuzzy suhu ditunjukkan dalam Tabel 7.2 di bawah ini:

Figure 7.28: Sprinkle system.

Table 7.2: Parameter dari himpunan fuzzy: suhu

MF-ke	MF	MF label	Parameter
1	Bell	Sangat Rendah	a=2.75; b=2.5; c=18
2	Bell	Rendah	a=2.75; b=2.5; c=23.5
3	Bell	Normal	a=2.75; b=2.5; c=29
4	Bell	Tinggi	a=2.75; b=2.5; c=34.5
5	Bell	Sangat Tinggi	a=2.75; b=2.5; c=40

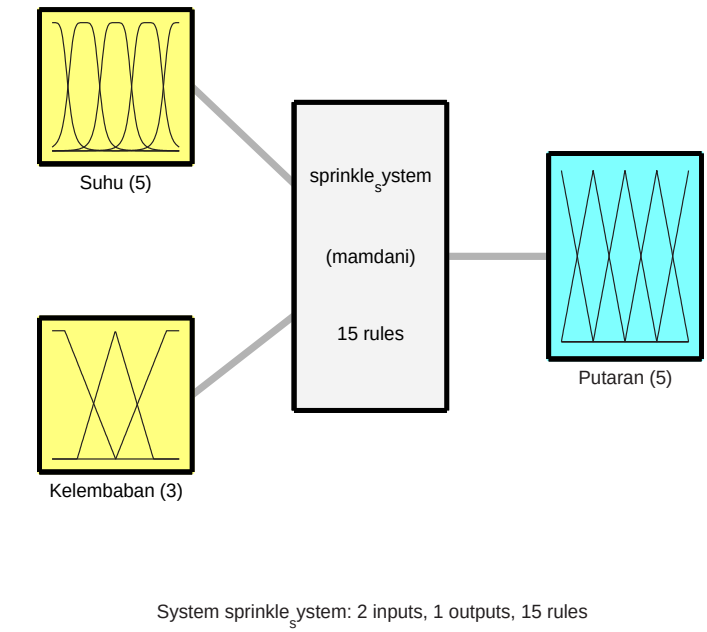

Figure 7.29: Diagram sprinkle system.

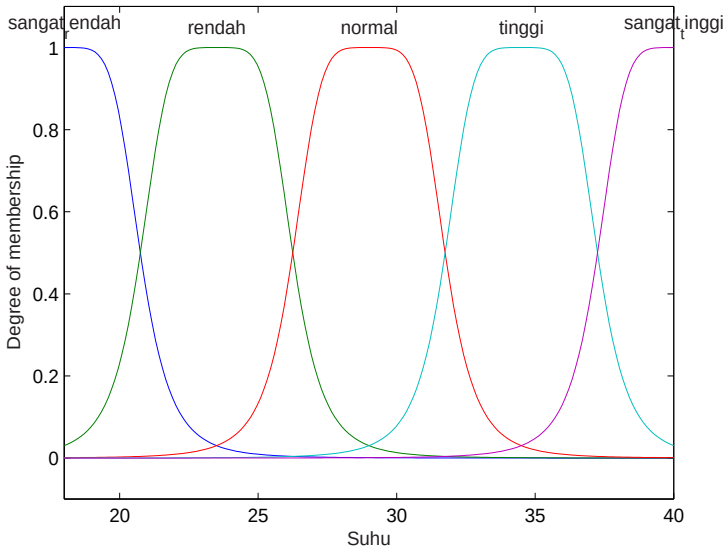

Figure 7.30: Himpunan fuzzy dari input ke-1: suhu.

Himpunan fuzzy dari input kelembaban tanah ditunjukkan dalam Gambar 7.31. Spesifikasi parameter dari himpunan fuzzy kelembaban tanah ditunjukkan dalam Tabel 7.3 di bawah ini:

Table 7.3: Parameter dari himpunan fuzzy: kelembaban

MF-ke	MF	MF label	Parameter
1	Trapesium	Rendah	a=0; b=0; c=10; d=50
2	Segitiga	Normal	a=20; b=50; c=80
3	Trapesium	Tinggi	a=50; b=90; c=100; d=100

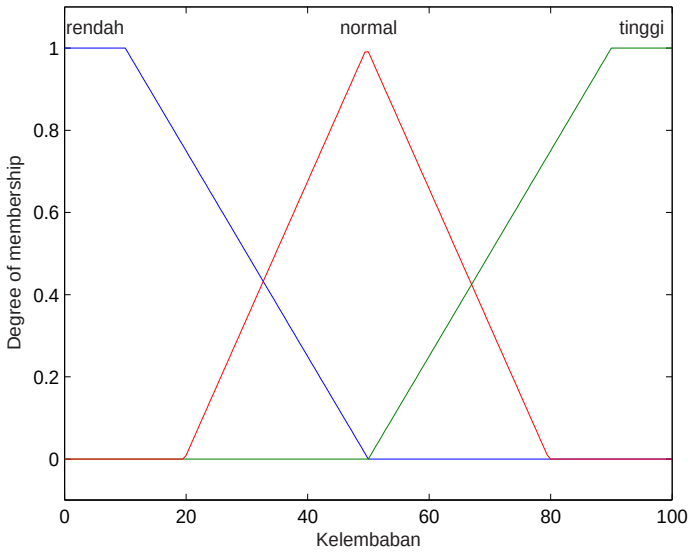

Figure 7.31: Himpunan fuzzy dari input ke-2: kelembaban tanah.

Himpunan fuzzy dari output lama putaran ditunjukkan dalam Gambar 7.32. Spesifikasi parameter dari himpunan fuzzy lama putaran ditunjukkan dalam Tabel 7.4 di bawah ini:

Table 7.4: Parameter dari himpunan fuzzy: lama putaran

MF-ke	MF	MF label	Parameter
1	Segitiga	Sangat Sebentar	a=-15; b=10; c=35
2	Segitiga	Sebentar	a=10; b=35; c=60
3	Segitiga	Normal	a=35; b=60; c=85
4	Segitiga	Lama	a=60; b=85; c=110
5	Segitiga	Lama Sekali	a=85; b=110; c=135

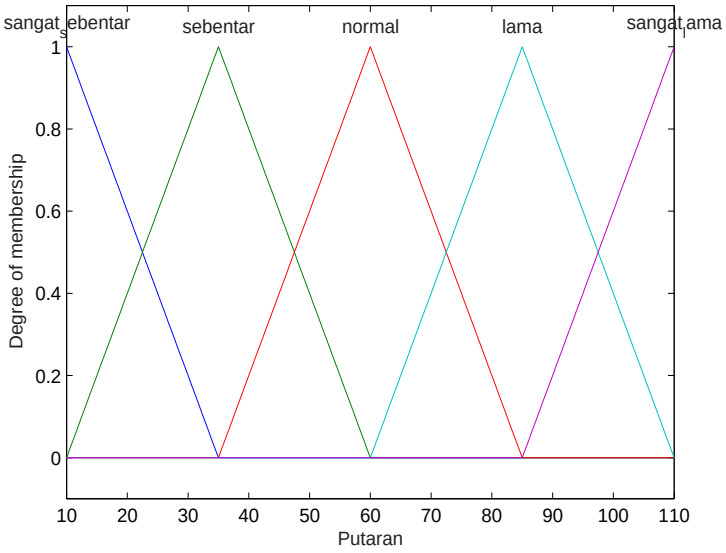

Figure 7.32: Himpunan fuzzy dari ouput: lama putaran sprinkle.

Setelah melakukan desain terhadap himpunan fuzzy dari input dan output, proses paling krusial selanjutnya adalah menentukan rule-rule yang bertugas untuk memetakan input ke output. Desain terhadap rule dapat dibuat dengan menggunakan tabel input-output seperti ditunjukkan dalam Tabel 7.5.

Table 7.5: tabel Input-Ouput dari Sprinkle System

Suhu						
		SR	R	N	T	ST
Kelembaban	R	Sb	Sb	L	SL	SL
	N	SSb	Sb	N	L	SL
	T	SSb	SSb	N	N	L

Keterangan:

SR: Sangat rendah

R: Rendah

N: Normal

T: Tinggi

ST: Sangat Tinggi

Sb: Sebentar

SSb: Sangat Sebentar

L: Lama

SL: Sangat Lama

Berdasarkan tabel input-output di atas maka jumlah rule yang harus dibuat adalah sebanyak 15 rule sebagai berikut:

R1	If	suhu = sangat-rendah AND
		kelembaban = rendah
	Then	putaran = sebentar
R2	If	suhu = rendah AND
		kelembaban = rendah
	Then	putaran = sebentar
R3	If	suhu = normal AND
		kelembaban = rendah
	Then	putaran = lama
R4	If	suhu = tinggi AND
		kelembaban = rendah
	Then	putaran = sangat-lama
R5	If	suhu = sangat-tinggi AND
		kelembaban = rendah
	Then	putaran = sangat-lama
R6	If	suhu = sangat-rendah AND
		kelembaban = normal
	Then	putaran = sangat-sebentar
R7	If	suhu = rendah AND
		kelembaban = normal
	Then	putaran = sebentar
R8	If	suhu = normal AND
		kelembaban = normal
	Then	putaran = normal
R9	If	suhu = tinggi AND
		kelembaban = normal
	Then	putaran = lama
R10	If	suhu = sangat-tinggi AND
		kelembaban = normal
	Then	putaran = sangat-lama

R11	If	suhu = sangat-rendah AND
		kelembaban = tinggi
	Then	putaran = sangat-sebentar
R12	If	suhu = rendah AND
		kelembaban = tinggi
	Then	putaran = sangat-sebentar
R13	If	suhu = normal AND
		kelembaban = tinggi
	Then	putaran = normal
R14	If	suhu = tinggi AND
		kelembaban = tinggi
	Then	putaran = normal
R15	If	suhu = sangat-tinggi AND
		kelembaban = tinggi
	Then	putaran = lama

Sekarang mari kita coba fuzzy sprinkle system yang telah kita bangun tersebut dengan memberikan inputan berupa: suhu = 25°C dan kelembaban tanah = 30%. Maka dengan memperhatikan Gambar 7.33 yang telah digenerate oleh MATLAB dengan dengan menggunakan defuzzifikasi centroid, didapatkan bahwa sprinkle system akan berputar selama 47.7 menit.

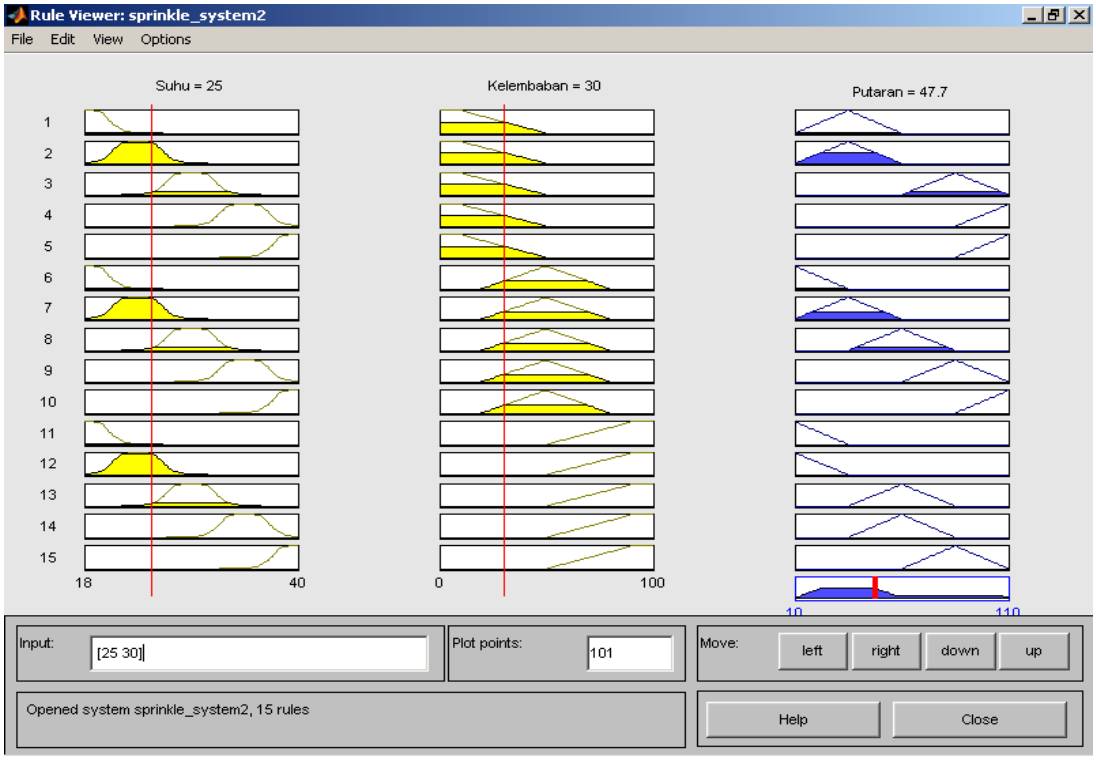

Figure 7.33: Representasi rule dan proses inferensi dari sprinkle system.

CONTOH KASUS 4

Sebuah washing machine menggunakan sistem fuzzy untuk menentukan berapa lama mesin tersebut harus berputar. Lihat Gambar 7.34. Masukan yang digunakan adalah berat-beban (0-10kg) dan tingkat-kekotoran-air (0-80%). Adapun himpunan fuzzy dari masing-masing input dan output adalah sebagai berikut:

Figure 7.34: Washing machine.

berat-beban(range: 0 - 10)

Subset	MF	Parameter
Ringan	trapesium	a=0; b=0; c=2; d=5
Normal	trapesium	a=2; b=4; c=6; d=8
Berat	trapesium	a=5; b=8; c=10; d=10

kekotoran-air(range: 0 - 80)

Subset	MF	Parameter
Bersih	segitiga	a=0; b=20; c=40
Normal	segitiga	a=20; b=40; c=60
kotor	segitiga	a=40; b=60; c=80

lama-putaran(range: 0 - 40)

Subset	MF	Parameter
Sebentar	segitiga	a=0; b=10; c=20
Normal	segitiga	a=10; b=20; c=30
Lama	segitiga	a=20; b=30; c=40

Diketahui rule-rule sebagai berikut:

R1 If berat-beban = ringan
AND kekotoran-air = normal
THEN lama-putaran = normal

R2 If berat-beban = normal
AND kekotoran-air = kotor
THEN lama-putaran = lama

R3 If berat-beban = normal
AND kekotoran-air = normal
THEN lama-putaran = normal

R4 If berat-beban = ringan
AND kekotoran-air = kotor
THEN lama-putaran = lama

Jika berat pakaian yang dimasukkan ke dalam washing machine pada saat itu adalah 3 kg dan tingkat kekotoran air adalah 55%, berapa lama mesin tersebut akan bekerja (dengan menggunakan metoda deffuzifikasi centroid)?

7.11 Fuzzy C-Means

Fuzzy C-Means (FCM) adalah salah satu bentuk implementasi sistem fuzzy untuk kasus analisa cluster (clustering). *Clustering* merupakan salah satu topik penting yang digunakan dalam pengenalan pola, tetapi dalam banyak hal *clustering* juga dapat ditemukan untuk aplikasi-aplikasi sistem informasi atau sistem pendukung keputusan. FCM dibangun oleh J.C. Bezdek pada tahun 1981 (Klir, 1995).

FCM melakukan proses pengklasifikasian data dengan berdasarkan fungsi oyektif:

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^m \|x_i - c_j\|^2, \quad 1 \le m \le \infty \tag{7.19}$$

dimana m adalah sembarang bilangan real lebih besar dari 1, u_{ij} adalah derajad keanggotaan dari x_i pada cluster ke-j, c_j adalah pusat (center) dari cluster dan $\| * \|$ adalah jarak Euclidean antara data dengan pusat cluster.

Proses pengelompokan dilakukan dengan menggunakan iterasi optimasi terhadap fungsi obyektif yang telah didefinisikan di atas. Proses update terhadap pusat cluster dilakukan dengan persamaan:

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^m \cdot x_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{7.20}$$

Sedangkan update terhadap derajad keanggotaan dilakukan dengan persamaan:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|}\right)^{\frac{2}{m-1}}} \tag{7.21}$$

Iterasi akan berhenti pada saat nilai dari $\max_{ij}\{\|u_{ij}^{k+1} - u_{ij}^k\|\} < \varepsilon$, dimana ε adalah nilai error yang terletak antara 0 dan 1, sedangkan k adalah langkah iterasi. Algoritma FCM dilakukan dengan urutan sebagai berikut:

STEP 1. Inisialisasu matrik U yang berisi nilai dari derajad keanggotaan u_{ij}. Inisialisasi U disimbolkan dengan $U^{(0)}$.

STEP 2. Pada langkah ke-k lakukan update terhadap nilai pusat dari cluster $C_{(k)} = [c_j]$.

STEP 3. Pada langkah ke-k lakukan update terhadap nilai dari derajad keanggotaan $U_{(k)}$.

STEP 4. Jika $\|U^{(k+1)} - U^{(k)}\| < \varepsilon$, maka berhenti, jika tidak kembali ke STEP 2.

7.12 Aplikasi Sistem Fuzzy

Note:

Sistem fuzzy sangat luas digunakan untuk berbagai macam aplikasi, mulai dari bidang kontrol (engineering) sampai bidang-bidang yang terkait dengan sistem informasi. Di bawah ini adalah beberapa contoh aplikasi sistem fuzzy:

- Bidang kontrol: pengendali gerakan robot, pengendali mesin cuci, pengendali suhu udara dalam sebuh ruangan, dsb.
- Sistem Pengambil Keputusan (decision support system).
- Peramalan (forecasting).
- Geographical information system (GIS).
- Data Mining.
- Aplikasi pengenalan pola.
- E-commerce: risk analysis pada e-commerce.
- Speech and character recognition.

Dalam perkembangan lebih lanjut, sistem fuzzy tidak berdiri sendiri melainkan digabungkan dengan Jaringan Syaraf Tiruan (JST). Salah satu

contoh penggabungan sistem fuzzy dan JST adalah ANFIS (Adaptive-Network-Based Fuzzy Inference System).

7.13 SOAL LATIHAN

Penentuan keputusan investasi dalam bentuk saham dibuat dengan menggunakan system fuzzy. Input bagi system fuzzy adalah "suku bunga" tabungan dan "inflasi". Sedangkan keluaran sistem fuzzy adalah "jumlah investasi" dalam bentuk uang yang harus dikeluarkan. Himpunan fuzzy untuk masing-masing variabel di atas adalah sebagai berikut:

Suku-bunga(range: 0 - 15 %)

Subset	MF	Parameter
Sangat-rendah	trapesium	a=0; b=0; c=2; d=5
Rendah	segitiga	a=3; b=5; c=7
Sedang	segitiga	a=5; b=7.5; c=10
Tinggi	segitiga	a=8; b=10; c=12
Sangat-tinggi	trapesium	a=10; b=13; c=15; d=15

Inflasi(range: 0 - 10 %)

Subset	MF	Parameter
Sangat-rendah	trapesium	a=0; b=0; c=3; d=4
Rendah	bell	a=0.5; b=2; c=4.5
Sedang	bell	a=0.5; b=2; c=5.5
Tinggi	bell	a=0.5; b=2; c=6.5
Sangat-tinggi	trapesium	a=7; b=8; c=10; d=10

Investasi(range: 10 - 100 jt rupiah)

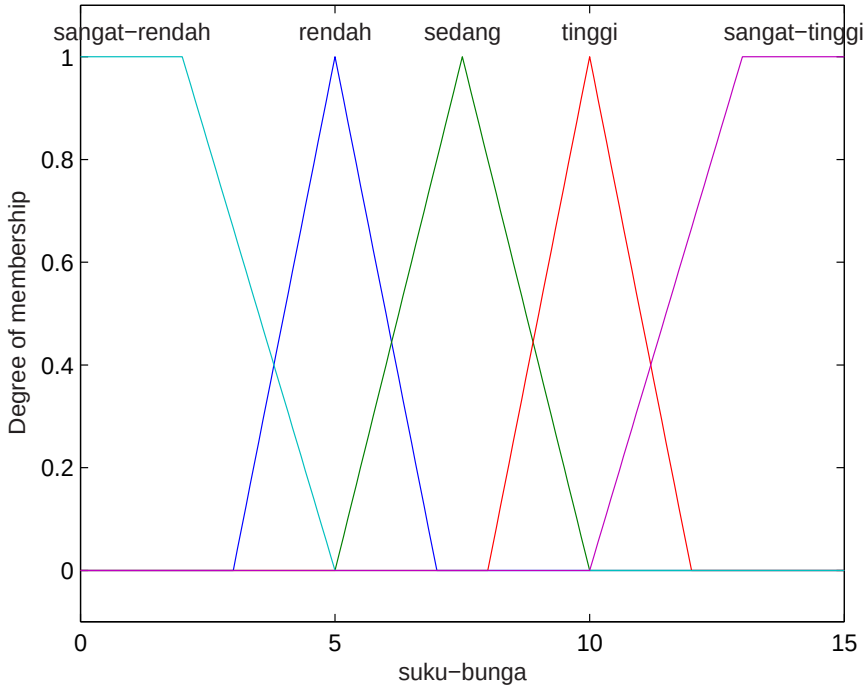

Figure 7.35: Himpunan fuzzy untuk input Suku-bunga.

Subset	MF	Parameter
Sangat-kecil	trapesium	a=0; b=0; c=20; d=30
Cukup-kecil	segitiga	a=20; b=30; c=40
Kecil	segitiga	a=30; b=40; c=50
Sedang-kiri	segitiga	a=40; b=50; c=60
Sedang-kanan	segitiga	a=50; b=60; c=70
Besar	segitiga	a=60; b=70; c=80
Cukup-besar	segitiga	a=70; b=80; c=90
Sangat-besar	trapesium	a=80; b=90; c=100; d=100

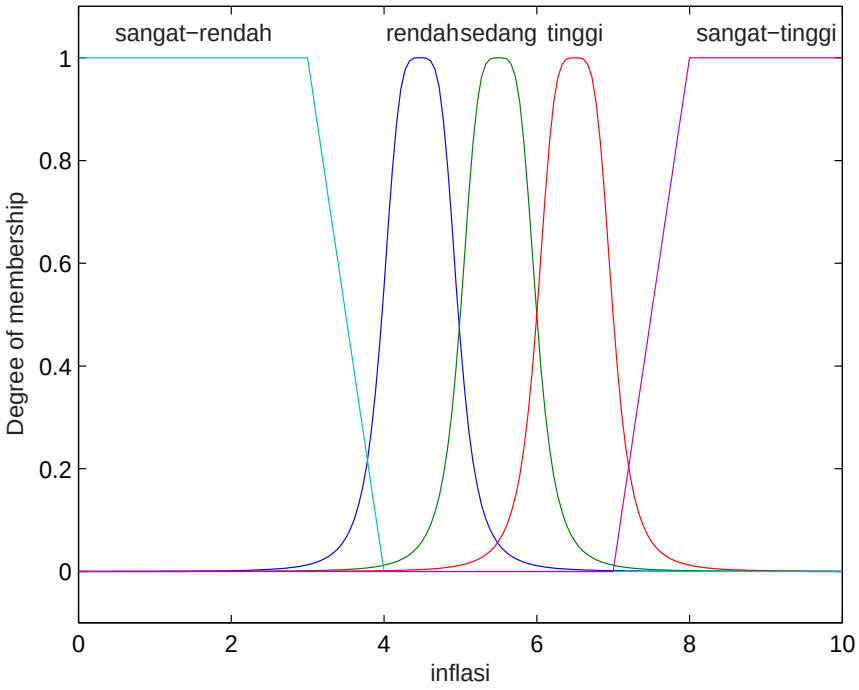

Figure 7.36: Himpunan fuzzy untuk input Inflasi.

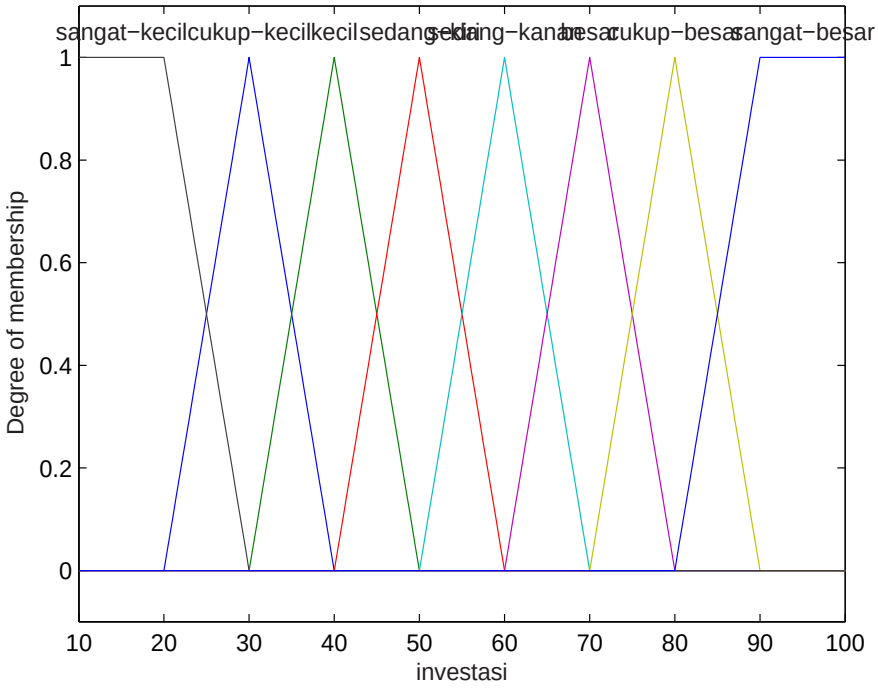

Figure 7.37: Himpunan fuzzy untuk output Investasi.

Berdasakan pada himpunan fuzzy input dan output di atas, maka tentukan:

1. Tentukan semua rule-rule yang akan digunakan untuk memetakan input ke output dari sistem fuzzy tersebut !

2. Lakukan uji coba sistem fuzzy tersebut dengan mengambil 1 buah nilai input suku-bunga pada range 3 - 12% dan 1 buah input inflasi pada range 4 - 7%, tentukan jumlah investasi untuk kasus yang anda uji!

Chapter 8

Jaringan Syaraf Tiruan (Artificial Neural Network)

Tujuan Instruksional Khusus

- Mahasiswa memahami berbagai jenis arsitektur dan algoritma yang digunakan dalam Jaringan Syaraf Tiruan.
- Perbandingan antara jaringan syaraf biologis dan Jaringan Syaraf Tiruan.
- Mahasiswa mampu menjelaskan konsep belajar dalam Jaringan Syaraf Tiruan.
- Mahasiswa mampu merencanakan aplikasi sistem Jaringan Syaraf Tiruan untuk menyelesaikan permasalahan-permasalahan time-series.

8.1 Pendahuluan

Note:

Artificial Neural Network (Jaringan Syaraf Tiruan - JST) adalah sebuah sistem pemroses informasi yang memiliki karakteristik performasi tertentu sebagaimana halnya pada jaringan syaraf biologis. (Fauset, hal 3). JST dibentuk dari generalisasi model matematik jaringan syaraf biologis, berdasarkan asumsi bahwa :

1. Pemrosesan informasi terjadi pada suatu bentuk prosesor sederhana yang disebut neuron.
2. Sinyal dilewatkan antar neuron melalui sebuah koneksi tertentu.
3. Masing-masing koneksi berasosiasi dengan bobot (weight) tertentu.
4. Masing-masing neuron menerapkan fungsi aktifasi tertentu pada masing-masing jumlah sinyal input untuk menentukan sinyal keluaran.

Dengan demikian karakteristik suatu JST didasarkan pada :

1. Pola koneksi antar neuron, yang selanjutnya disebut sebagai arsitektur JST.
2. Metode untuk menentukan perubahan nilai bobot, yang selanjutnya disebut sebagai metode belajar atau algoritma JST.
3. Fungsi aktifasi.

Jaringan Syaraf Tiruan dapat dianggap sebagai semacam kotak hitam (black box) yang memetakan masukan (input) ke suatu keluaran (output) tertentu. Lihat Gambar 8.1 Kita tidak pernah tahu isi dari kotak hitam tersebut. Sebagai suatu model adaptif yang memiliki kemampuan belajar, JST mampu membuat generalisasi dan menyimpan hasil proses belajar tersebut untuk menghasilkan keluaran(output) sebagaimana kita harapkan atau tidak kita harapkan tetapi tetap sesuai dengan kaidah-kaidah JST.

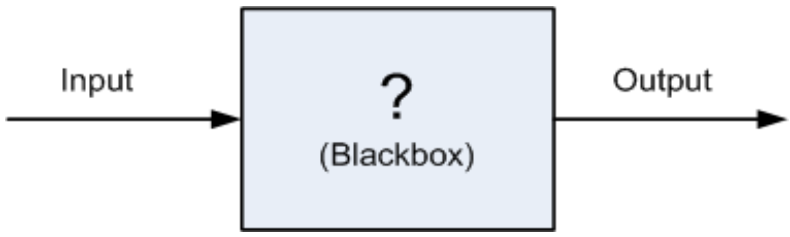

Figure 8.1: ANN sebagai sebuah blackbox.

Secara umum berdasarkan algoritma belajar, JST dapat dibedakan atas :

1. Supervised learning, yaitu metode belajar JST dengan pengawasan, dalam hal ini membandingkan suatu masukan dengan suatu output tertentu yang sudah ditentukan. Contoh : Multilayer-Perceptron (MLP), Back-Error Propagation (BEP) dan Radial Basis Function (RBF).

2. Unsupervised learning, yaitu metode belajar JST tanpa pengawasan, dalam hal ini tidak diperlukan adanya suatu keluaran sebagai acuan. Contoh : Kohonen - Self-Organizing Map (SOM), Learning Vector Quantization (LVQ), Adaptive Resonance Theory (ART).

Klasifikasi JST yang lain dibuat oleh Kohonen sebagai berikut :

1. Signal Transfer Network. *"In signal transfer networks, the input signal is transformed into an output signal. The signal traverses the network and undergoes a signal transformation of some kind. The network has usually a set of pre-defined basis functions, which are parametrized. The learning in these networks corresponds to changing parameters of these basis functions".* Contoh : Multilayer-Perceptron (MLP), Back-Error Propagation (BEP) dan Radial Basis Function (RBF)

2. State Transition Network. *"In state transition networks the dynamic behavior of the network is essential. Given an input, the*

network converges to a stable state, which, hopefully, is a solution to a problem presented to it". Contoh : Jaringan Hopfield, Boltzman Machine.

3. Competitive Learning Network. *"In competitive learning networks, or self-organizing networks, all the neurons of the network receive the same input. The cells have lateral competition and the one with most activity "wins". Learning is based on the concept of winner neurons".* Contoh : Self-Organizing Map.

JST yang disebut sebagai arsitektur komputer generasi ke VI memiliki beberapa perbedaan dengan arsitektur komputer Von-Neuman sebagaimana terdapat dalam Tabel 1.1.

8.2 Model Sel Syaraf

Note:

Sub-Bab ini diawali dengan pembahasan dasar fisiologis sel syaraf secara umum yang akan dijadikan dasar pembentukan model sel syaraf. Model sel syaraf ini merupakan pembentuk utama Jaringan Syaraf Tiruan yang disusun berdasarkan pilihan arsitektur dan konsep belajar.

8.2.1 Tinjauan Fisiologis Sel Syaraf

Sistem syaraf biologis merupakan susunan sel-sel syaraf yang disebut neuron. Tidak kurang dari 100 milyar neuron mendukung fungsi otak manusia dewasa. Setiap neuron berhubungan dengan neuron lainnya melalui sambungan khusus yang disebut dengan sinapsis. Sebuah neuron dapat berhubungan dengan beberapa hingga 10000 neuron lain, sehingga masing-masing neuron dapat mengirimkan impuls listrik kepada sel tujuan sampai sekitar 10000 sel.

Table 8.1: perbedaan Digital Computer dan Neural Network

	Digital Computer	*Neural Network*
⟲	Deductive Reasoning, rule dibuat untuk data masukan sehingga menghasilkan keluaran tertentu.	Inductive Reasoning, diberikan masukan dan keluaran (pola belajar) dan JST menyusun rule.
⟲	Proses komputasi terpusat, sinkronous dan serial.	Proses komputasi adalah kolektif, asinkronous dan paralel.
⟲	Memori terpaket dengan model pengalamatan lokasi.	Memori terdistribusi dengan model pengalamatan isi memori.
⟲	Tidak bersifat kebal terhadap kesalahan (fault tolerance).	Kebal terhadap kesalahan (fault tolerance).
⟲	Cepat, dalam ukuran jutaan per detik.	Lambat, dalam ukuran ribuan per detik.
⟲	Eksak.	Tidak eksak.
⟲	Konektifitas statis.	Konektifitas dinamis.
⟲	Dapat digunakan apabila rule telah ditentukan dengan masukan yang akurat.	Dapat digunakan apabila rule tidak diketahui atau kompleks, masukan bias atau parsial.

Sel syaraf (neuron). Gambar 8.2 memperlihatkan sebuah sel syaraf atau disebut dengan neuron, secara garis besar terdiri atas :

- Badan sel, di dalamnya terdapat inti (nucleus) dan,
- Benang-benang perpanjangan sel, yang terdiri atas satu akson dan beberapa dendrit.

Membran sel syaraf memisahkan plasma yang ada di dalam sel dengan cairan diluarnya. Karena sifatnya yang permeabilitas, membran mampu melewatkan ion-ion tertentu. Ion sodium (Na+) dipompa keluar sel,

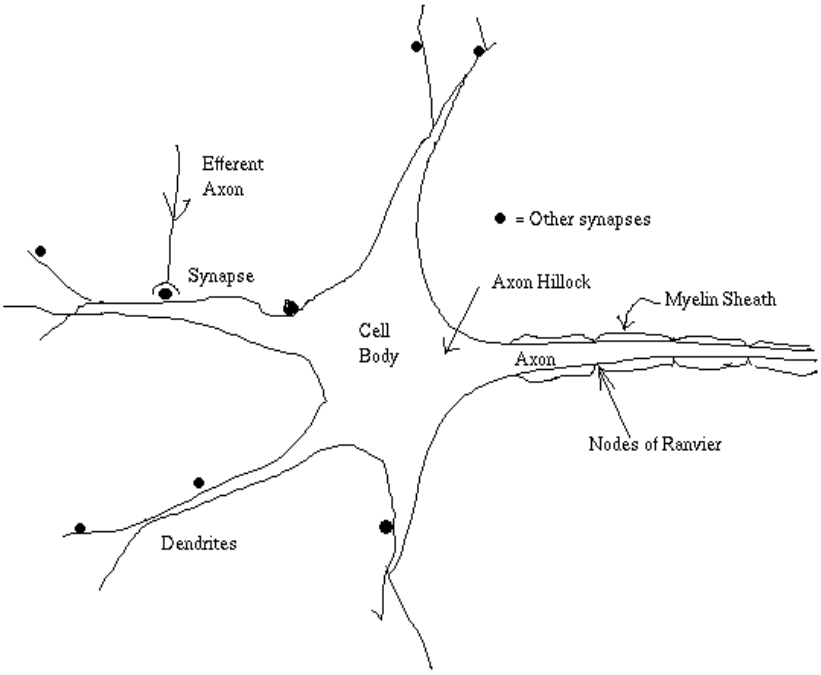

Figure 8.2: Fisiologi sel syaraf.

sedangkan ion Potasium (K+) dipompa ke dalam sel. Sehingga di dalam sel terdapat ion Potasium dengan konsentrasi lebih tinggi dibandingkan dengan ion sodium dan di luar sel ion Sodium mempunyai konsentrasi lebih tinggi dibanding ion Potasium. Keseimbangan ini menyebabkan adanya perbedaan potensial pada membran sel sekitar 70 mV, dengan tegangan di dalam sel lebih negatif. Tegangan ini disebut tegangan rehat (Resting Potential). (Skapura,1992 : hal :9)

Pada saat Neuron menerima sinyal informasi melalui dendrit atau hubungan lainnya, tegangan rehat akan terganggu. Depolarisasi (tegangan menjadi lebih positif) terjadi pada daerah pemicu (axon hillock) sehingga dihasilkan sebuah tegangan aksi yang menjalar melalui akson. Lihat Gambar 8.3. Kemudian neuron akan kembali pada keadaan semula, dengan tegangan rehat sebesar minus 70 mV.

Potensial aksi yang dipropagasikan sepanjang benang akson berujung pada cabang-cabang kontak sinaptik. Suatu pembungkus, disebut

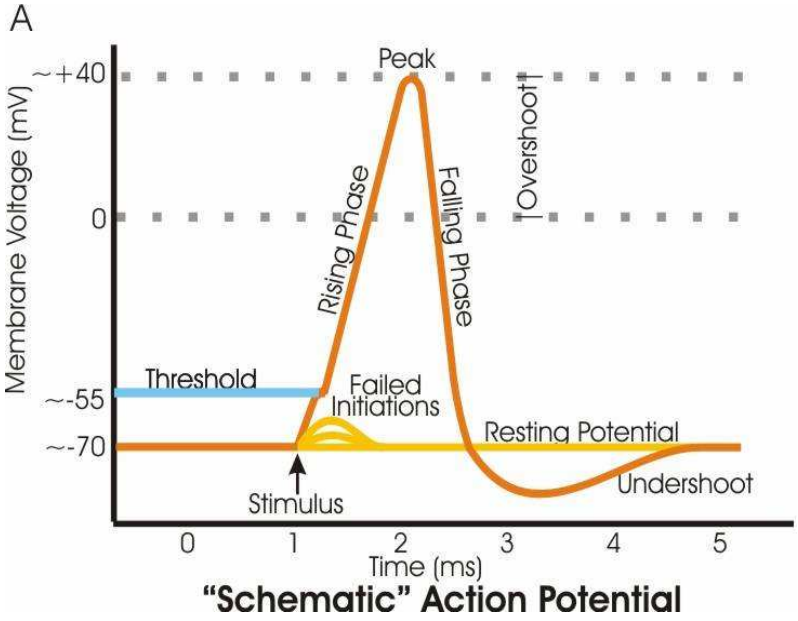

Figure 8.3: Depolarisasi. (*source: wikipedia*)

myelin, digunakan untuk mengisolasi akson dengan cairan di luar sel, sehingga mempercepat proses propagasi. Dalam jarak tertentu, di sepanjang akson yang terbungkus myelin terdapat celah yang disebut Titik Ranvier (Node Of Ranvier). Celah ini berkelakuan sebagai penguat ulang (repeater), yang akan memperbaharui sinyal yang melewatinya.

8.2.2 Sinapsis

Note:

Sinapsis adalah daerah sambungan khusus antar neuron. Diperkirakan dalam otak manusia terdapat sekitar 1014 sinapsis, yang dipakai dalam komunikasi antar sel. Dalam sistem syaraf, pola interkoneksi sel ke sel beserta fenomena komunikasi antar unit pemroses tersebut menentukan kemampuan komputasi jaringan secara keseluruhan. Dengan kata lain sinapsis merupakan bagian penting dalam komputasi neural (neurocomputing). (Dayhoff,1990: hal 136)

Proses transmisi sinyal dalam sinapsis adalah sebagai berikut : Masuknya potensial aksi menyebabkan depolarisasi pada membran sinap-

sis awal (presinaptic/pemancar), permeabilitas ion kalsium naik. Ion - ion Ca^{2+} masuk ke dalam bouton (terminal pemancar pada ujung-ujung akson) dan memicu pembentukan gelembung yang berisi sejumlah Neurotransmitter. Neurotransmitter adalah suatu kuanta biologis yang merupakan unit terkecil yang tersusun atas molekul-molekul transmitter dengan energi yang mampu mempengaruhi sifat listrik membran sinapsis akhir (postsinaptic/penerima). Gelembung bergerak ke arah celah dan didifusikan sehingga tersebar di dalam celah. Sebagian Neurotransmiiter mencapai membran sinapsis akhir dan menempel pada tempat penerima.

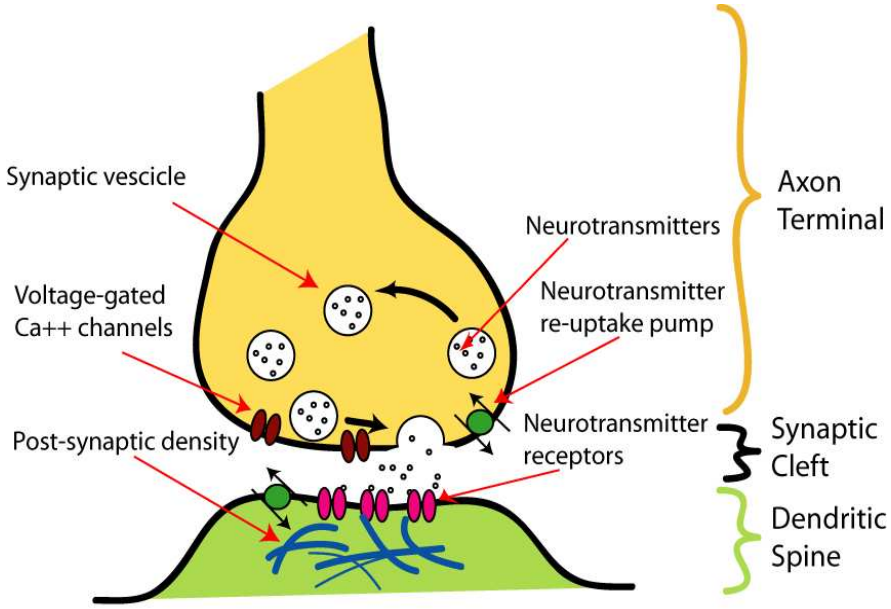

Figure 8.4: Ilustrasi pelepasan neurotransmitter pada sinapsis.(*source: wikipedia*)

Secara keseluruhan dapat dikatakan, penyebaran sinyal rangsang dari sel sinapsis awal ke sinapsis akhir dimodulasi oleh ion kalsium maupun enzim. Tingkat modulasi mempengaruhi tingkat keterpengaruhan sel sinapsis akhir terhadap rangsang yang dikeluarkan oleh sel sinapsis awal. Mekanisme modulasi neural inilah yang diduga digunakan dalam penyesuaian ‘kekuatanŠ hubungan sinapsis pada saat belajar.

8.2.3 Operasi Listrik pada Neuron

Note:

Setiap neuron menggunakan beberapa tipe sinyal listrik untuk melakukan fungsinya. Dalam berkomunikasi, impuls listrik merupakan bahasa umum yang dimengerti oleh sistem syaraf.

Catu daya dan potensial rehat dihasilkan oleh membran sel dengan mengatur perbedaan konsentrasi di dalam dan di luar sel. Sedangkan potensial aktifasi dihasilkan di daerah akson. Potensial ini merupakan hasil jumlah sinyal pengaruh yang diterima dari semua sinapsis melalui cabang-cabang dendrit setiap waktu. Potensial sinapsis akhir yang dihasilkan pada bagian penerima merupakan masukan bagi neuron untuk diolah lebih lanjut.

Bagian penerima ini merupakan transduser yang mengubah energi rangsang menjadi energi listrik. Energi rangsang yang diterima dapat bersifat membangkitkan (Excitatory Postsinaptic Potential / EPSP) atau menghambat (Inhibitory Postsinaptic Potential / IPSP). Misalkan pulsa pertama yang masuk ke sinapsis akan menghasilkan EPSP. Jika pulsa kedua muncul saat EPSP pertama belum hilang, maka kedua pengaruh tersebut akan dijumlahkan. Penjumlahan urutan energi rangsang pada sinapsis yang sama disebut penjumlahan temporal. (Akibat dari penjumlahan temporal ini, sinapsis juga berfungsi menerjemahkan kode-kode pulsa yang dipancarkan oleh akson). Selain secara temporal, semua potensial sinapsis akhir yang diterima dari berbagai sinapsis dalam neuron yang sama, akan dijumlahkan dan dihasilkan potensial aktifasi. Sifat penjumlahan ini disebut penjumlahan spasial.

8.3 Pemodelan Sel Syaraf

Note:

Seperti dijelaskan di awal bahwa penyebaran sinyal rangsang dari sel sinapsis awal ke sinapsis akhir dimodulasi oleh ion kalsium maupun en-

zim. Hasilnya diumpankan ke dendrit dan badan sel untuk dijumlahkan, sehingga menghasilkan potensial aktifasi, A. Besar potensial aktifasi menentukan tingkat perkembangan potensial aksi yang dikeluarkan sel.

Tabel yang menunjukkan ringkasan bagian - bagian sel yang diperlukan dalam pemrosesan sinyal terlihat dalam Tabel 8.2.

Table 8.2: Ringkasan bagian - bagian sel yang diperlukan dalam pemrosesan sinyal

Bagian	Masukan dan Keluaran	Aktifitas
Sinapsis	Input: potensial aksi, X; Output: potensial postsinaptik, V_p	Modulasi dan penerjemahan kode potensial aksi
Dendrit & badan sel	Input: potensial postsinaptik; Output: potensial aktifasi (A)	Penjumlahan parsial dari berbagai sinapsis.
Akson	Input: potensial aktifasi (A); Output: potensial aksi	Fungsi aktifasi dan pengkodean

8.3.1 Pemodelan Sinapsis dan Badan Sel

Variabel bobot, W_{ij}, diambil untuk memodelkan pengaruh modulasi ion Ca^{2+} dan enzim pada sinapsis. Variabel W_{ij} disebut sebagai bobot koneksi yang menghubungkan sel j ke sel i. Dengan variabel tersebut, fungsi dari sinapsis, V_p, dapat dimodelkan sebagai fungsi linier seperti:

$$V_p = X_j.W_{ij} \tag{8.1}$$

dimana X_j adalah potensial aksi (masukan).

Selanjutnya dalam dendrit, semua potensial sinapsis akhir yang dihasilkan akan dijumlahkan dan menghasilkan potensial aktifasi untuk sel i:

$$net_i = \sum_{j=1}^{n} X_j.W_{ij} \tag{8.2}$$

8.3.2 Pemodelan Akson dan Potensial Aksi

Note:

Seperti dijelaskan di atas bahwa di dalam sebuah neuron terjadi pertukaran antara ion potasium $K+$ dan ion sodium Ca^{2+} melalui membran neuron. Pada saat gerbang sodium terbuka, maka ion sodium bergerak ke dalam neuron yang selanjutnya mendorong terjadinya depolarisasi. Ion sodium akan terus bertambah sampai mencapai ekuilibrium pada +60mV. Akan tetapi sebelum mencapai titik tersebut, gerbang sodium mulai menutup.

Pada saat gerbang sodium mulai menutup, gerbang potasium mulai membuka mengakibatkan aliran potasium keluar dari neuron. Hal ini menyebabkan tegangan dari membran kembali menurun ke titik resting potential.

Dalam bentuk matematis, perbandingan antara gerbang yang terbuka N_o dan gerbang yang tertutup N_c dapat dimodelkan dalam bentuk matematis:

$$N_o/N_c = e^{-\frac{E_t}{KT}} \tag{8.3}$$

dimana E_t adalah energi transisi yang dibutuhkan agar semua gerbang terbuka, K konstanta .... dan T adalah suhu dalam Kelvin. Jika $N = N_o + N_c$, maka γ, yaitu perbandingan antara jumlah gerbang terbuka dan jumlah semua gerbang terbuka atau tertutup (N_o/N) adalah:

$$\gamma = \frac{N_o}{N_c} = \frac{1}{1 + \frac{1}{K}e^{-\frac{E_p}{KT}}} \tag{8.4}$$

Persamaan di atas adalah suatu bentuk fungsi sigmoid. Pemberian tegangan, V , yang besar positif, menyebabkan γ bersaturasi pada 1, artinya

semua gerbang terbuka. Tegangan, V , yang besar negatif, menyebabkan γ bersaturasi pada 0, artinya semua gerbang tertutup. Bentuk fungsi sigmoid dapat dilihat dalam Gambar 8.5.

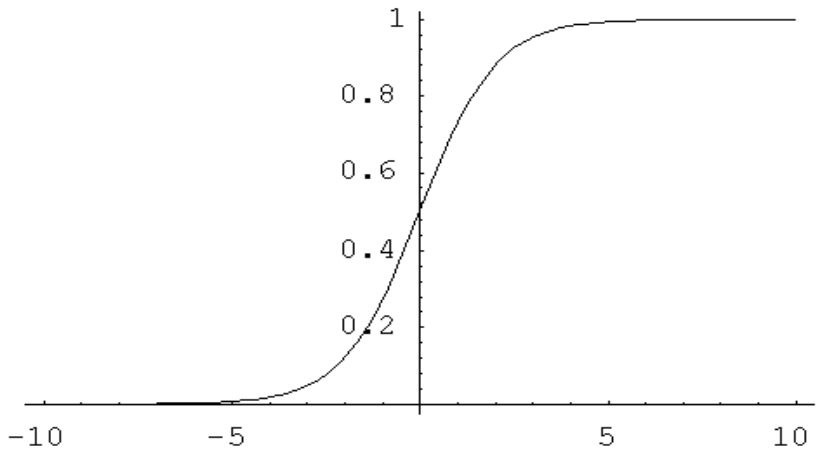

Figure 8.5: Fungsi aktifasi sigmoid

Jumlah gerbang yang terbuka menentukan potensial aksi yang dihasilkan. Dengan mengabaikan pengkodean pada akson, mekanisme inisialisasi (fungsi aktifasi) dapat dimodelkan dengan fungsi sigmoid sebagai berikut:

$$f(net_i) = \frac{1}{1 + e^{-net_i}} \tag{8.5}$$

Secara keseluruhan model neuroan pada Jaringan Syaraf Tiruan yang diturunkan dari model neuron yang sesungguhnya ditunjukkan dalam Gambar 8.6.

8.4 Interpretasi Terhadap Bias (Threshold)

Note:

Dalam JST, *bias* adalah suatu variabel bobot yang terhubung pada suatu masukan yang selalu bernilai 1 (biasanya diberi index 0). Dengan

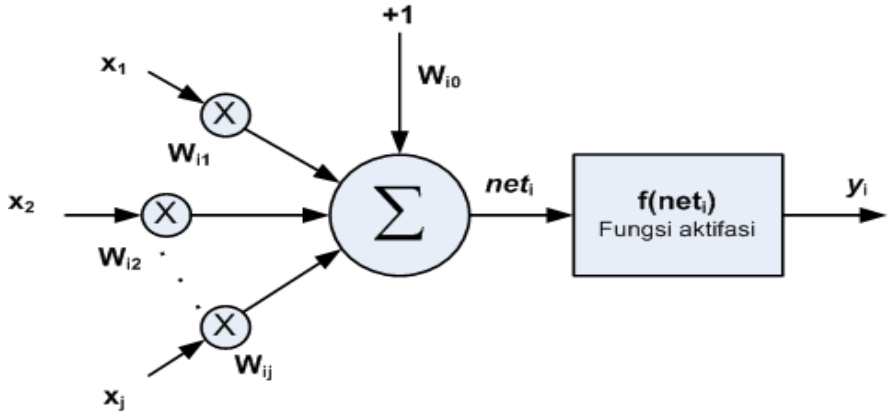

Figure 8.6: Model 1 sel syaraf tiruan

penambahan bias, maka persamaan potensial aktifasi di atas menjadi:

$$net_i = \sum_{j=0}^{n} X_j.W_{ij} \tag{8.6}$$

$$= \sum_{j=1}^{n} X_j.W_{ij} + W_{i0} \tag{8.7}$$

$$= q_i + \theta_i \tag{8.8}$$

dimana q_i merepresentasikan aktifasi dari semua sumber (dendrit) eksternal dari neuron, sedangkan θ_i adalah threshold internal.

Apa pengaruh variabel bias terhadap jaringan neuron secara keseluruhan? Variabel bias menggeser nilai threshold sedemikian rupa sehingga aktifasi dari neuron juga bergeser. Sebagai contoh perhatikan Gambar 8.7. Gambar tersebut merupakan fungsi aktifasi *Binary Threshold*, jika jumlah masukan, X, kurang dari 0 maka neuron *tidak fire*, sebaliknya jika lebih atau sama dengan 0, maka neuron akan *fire*. Tetapi apabila kita berikan, misalnya, nilai threshold 3, maka threshold akan mem-*bias* (menggeser) fungsi aktifasi sedemikian sehingga neuron sekarang akan fire jika jumlah inputan lebih besar dari -3.

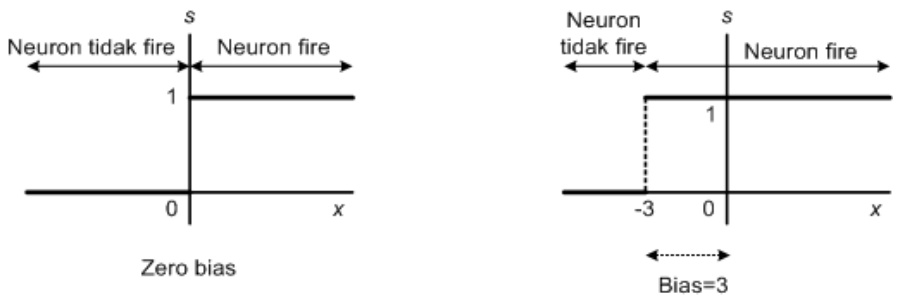

Figure 8.7: Fungsi aktifasi Binary Threshold dengan bias = 3.

8.5 Fungsi Aktifasi

Note:

Fungsi aktifasi mentransformasikan semua sinyal masukan (hasil penjumlahan dari dendrit) ke suatu nilai tertentu yang disebut sebagai potensial aksi. Atau dengan kata lain, fungsi aktifasi mentransformasikan nilai aktifasi yang tak terbatas (infinite) menjadi terbatas (finite) dalam range tertentu. Dalam JST terdapat bermacam-macam model fungsi aktifasi, antara lain: *binary dan bipolar threshold, linear threshold, binary dan bipolar sigmoidal, dan gaussian.* Tetapi seperti dijelaskan di atas, fungsi sigmoid mirip dengan keadaan neuron yang sesungguhnya. Karena itu fungsi sigmoid umum dipakai dalam model-model JST.

Fungsi Aktifasi Binary Threshold

Fungsi aktifasi binary threshold dirumuskan dengan:

$$f(net_i) = \begin{cases} 1 & net_i \geq 0 \\ 0 & net_i < 0. \end{cases} \tag{8.9}$$

Fungsi aktifasi binary threshold dengan $\theta = 0$ dan $\theta = 3$ ditunjukkan dalam Gambar 8.8.

Fungsi Aktifasi Bipolar Threshold

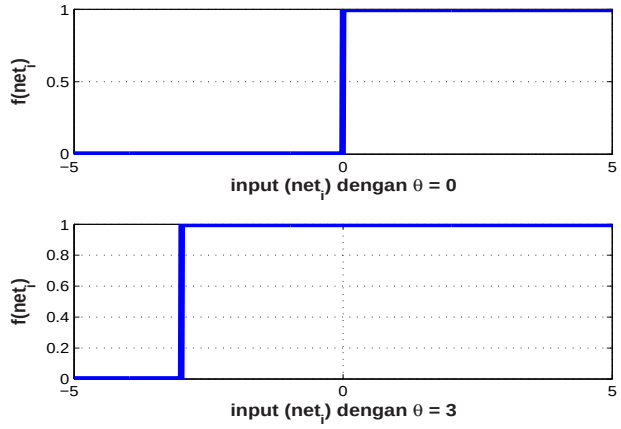

Figure 8.8: Fungsi aktifasi Binary Threshold.

Fungsi aktifasi bipolar threshold dirumuskan dengan:

$$f(net_i) = \begin{cases} 1 & net_i \geq 0 \\ -1 & net_i < 0. \end{cases} \tag{8.10}$$

Fungsi aktifasi bipolar threshold dengan $\theta = 0$ dan $\theta = -2$ ditunjukkan dalam Gambar 8.9. Seperti terlihat dalam gambar, fungsi aktifasi ini memiliki rentang nilai $f(net_i)$ antara -1 dan 1.

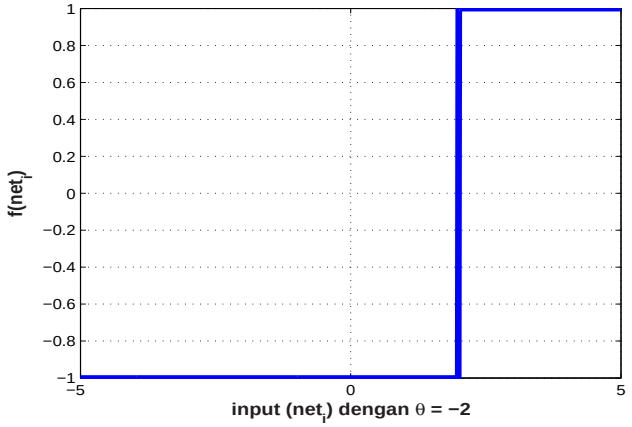

Figure 8.9: Fungsi aktifasi Bipolar Threshold.

Fungsi Aktifasi Linear Threshold

Fungsi aktifasi linear threshold dirumuskan dengan:

$$f(net_i) = \begin{cases} 0 & net_i \leq 0 \\ \alpha_i \; net_i & net_i 0 < net_i < net_m. \\ 1 & net_i \geq net_m \end{cases} \tag{8.11}$$

dimana $\alpha_i = 1/net_m$ adalah slope/gradient/kemiringan dari fungsi aktifasi pada $0 < net_i < net_m$. Fungsi aktifasi linear threshold dengan $\theta = 0$ dan $\theta = -1$ ditunjukkan dalam Gambar 8.10.

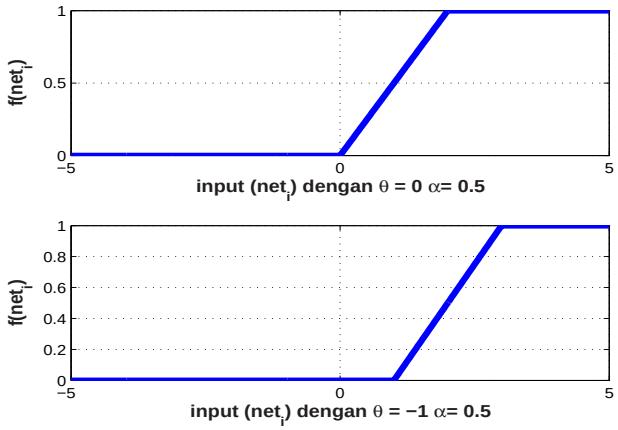

Figure 8.10: Fungsi aktifasi Linear Threshold.

Fungsi Aktifasi Binary Sigmoid

Fungsi aktifasi telah dijelaskan pada pemodelan neuron. Tetapi untuk memudahkan akan dituliskan kembali di sini. Fungsi sigmoid dirumuskan dengan:

$$f(net_i) = \frac{1}{1 + e^{-\lambda net_i}} \tag{8.12}$$

dimana lambda adalah faktor penguatan. Fungsi aktifasi sigmoid dengan berbagai macam nilai λ ditunjukkan dalam Gambar 8.11.

Fungsi Aktifasi Bipolar Sigmoid

Fungsi aktifasi bipolar sigmoid dirumuskan dengan:

$$f(net_i) = 1 - \frac{2}{1 + e^{\lambda net_i}} \tag{8.13}$$

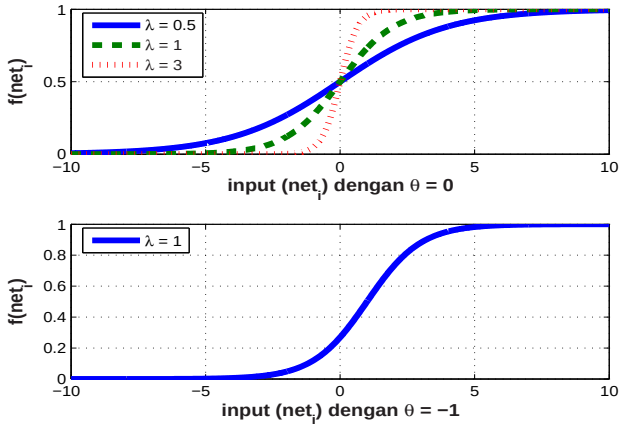

Figure 8.11: Fungsi aktifasi Binary Sigmoid.

dimana lambda adalah faktor penguatan. Fungsi aktifasi bipolar sigmoid rentang nilai $f(net_i)$ yang terletak antara -1 dan 1. Fungsi ini ditunjukkan dalam Gambar 8.12.

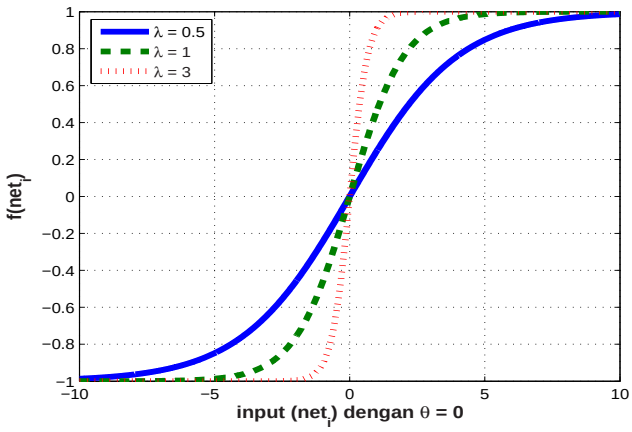

Figure 8.12: Fungsi aktifasi Bipolar Sigmoid.

Fungsi Aktifasi Gaussian

Fungsi aktifasi gaussian digunakan utamanya pada jaringan Radial Basis Function (RBF). Fungsi aktifasi ini dapat dirumuskan sebagai berikut:

$$f(net_i) = e^{-\frac{(net_i - c)^2}{2\sigma^2}} \tag{8.14}$$

dimana c adalah titik tengah dari fungsi gaussian dan σ adalah faktor

regangan (standar deviasi). Fungsi aktifasi ini dengan berbagai nilai σ ditunjukkan dalam Gambar 8.13.

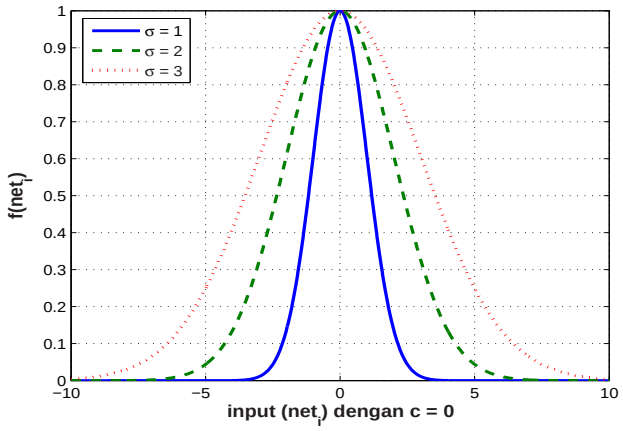

Figure 8.13: Fungsi aktifasi Gaussian.

8.6 Perceptron

Note:

Perceptron yang dikembangkan oleh Rosenblatt (1962) dan Minsky & Papert (1969,1988) merupakan pengembangan dari Hebb Rule dengan kemampuan belajar yang lebih baik. Dengan proses pelatihan berulang-ulang (iterasi), perceptron akan menghasilkan nilai bobot yang benar. Yaitu, nilai bobot yang menghasilkan keluaran benar untuk setiap setiap pelatihan pola input.

Salah satu bentuk Perceptron yang paling sederhana adalah perceptron dengan satu layer di mana nilai bobot dan bias dapat diupdate/dilatih untuk menghasilkan keluaran yang diinginkan. Salah satu bentuk training yang digunakan adalah *perceptron learning rule*. Salah satu hal yang membuat perceptron menjadi menarik adalah kemampuannya untuk belajar mengikuti data training yang diberikan dengan menggunakan varibel bobot yang diinisialisasi secara acak (random). Perceptron biasanya cocok digunakan untuk menyelesaikan masalah yang

berkaitan dengan klasifikasi pola. Selain mudah, perceptron juga akan memberikan keputusan yang cepat.

Bagian ini secara khusus akan membicarakan arsitektur dan algoritma perceptron sebagai dasar sistem JST yang lebih kompleks.

Secara khusus, perceptron menggunakan fungsi aktifasi yang mirip dengan fungsi aktifasi bipolar threshold dengan sedikit modifikasi. Fungsi aktifasi tersebut dirumuskan sebagai berikut:

$$f(net_i) = \begin{cases} 1 & net_i > \theta \\ 0 & -\theta \leq net_i < \theta \\ -1 & net_i < -\theta. \end{cases} \tag{8.15}$$

dimana θ adalah nilai threshold dari fungsi aktifasi. Bobot interkoneksi dari node input ke node berikutnya diupdate (disesuaikan) dengan menggunakan proses belajar perceptron. Untuk setiap pelatihan input, perceptron akan menghitung respon keluaran. Kemudian perceptron menghitung apakah nilai target telah tercapai. Jika nilai target tercapai maka proses belajar dianggap selesai. Proses penyesuaian (updating) bobot dilakukan sesuai dengan persamaan berikut:

$$W_i(new) = W_i(old) + \alpha.\ t.\ X_i \tag{8.16}$$

yang mana t adalah target (+1 atau -1) dan α adalah konstanta kecepatan belajar (learning rate).

8.6.1 Arsitektur Perceptron

Perceptron hanya memiliki satu layer dengan beberapa neuron saja. Karena itu perceptron hanya memiliki satu set bobot yang menghubungkan node input ke node output. Arsitektur dari perceptron ditunjukkan dalam Gambar 8.18.

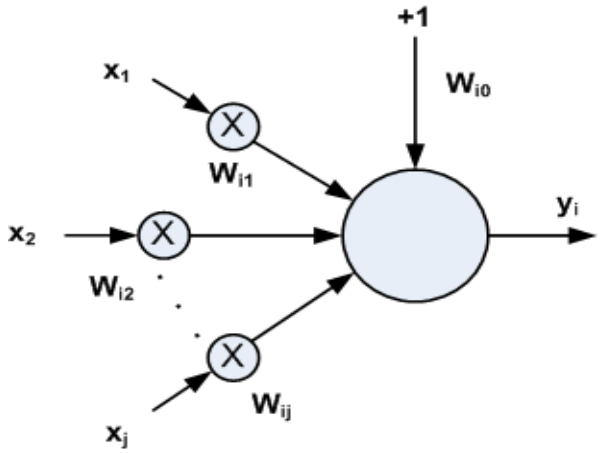

Figure 8.14: Arsitektur Perceptron.

Tujuan jaringan ini adalah meng-klasifikasikan pola input kepada suatu kelas tertentu. Apabila unit keluaran menghasilkan nilai +1, maka dikatakan bahwa pola input merupakan anggota dari suatu kelas tertentu. Apabila unit keluaran menghasilkan nilai -1, maka dikatakan bahwa pola input bukan merupakan anggota dari suatu kelas tertentu. Proses pelatihan dilakukan dengan menggunakan algoritma di bawah ini.

8.6.2 Algoritma Perceptron

Note:

- **Langkah 0.** Inisialisasi bobot dan bias. (Sederhananya, inisialisasi bobot dan bias dengan 0).
- **Langkah 1.** Jika kondisi berhenti (stopping condition) tidak dipenuhi kerjakan Langkah 2 - 6.
- **Langkah 2.** Untuk setiap pasangan pelatihan **s:t** kerjakan langkah 3-5. Dimana **s** adalah set data input, sedangkan **t** adalah set data target (contoh).
- **Langkah 3**. Lakukan setting untuk semua unit input :

$$X_i = s_i$$

- **Langkah 4.** Hitung tanggapan pada unit keluaran dari fungsi aktifasi.

- **Langkah 5.** Update bobot dan bias jika terdapat kesalahan (error) pada pola ini :

  Jika y tidak sama dengan t,

$$\begin{aligned} W_i(new) &= W_i(old) + \alpha.\, t.\, X_i \\ b(new) &= b(old) + \alpha.\, t \end{aligned}$$

  Else

$$\begin{aligned} W_i(new) &= W_i(old) \\ b(new) &= b(old) \end{aligned}$$

- **Langkah 6.** Uji kondisi berhenti : jika tidak terdapat perubahan bobot pada langkah 2, proses pelatihan telah selesai. Sebaliknya, jika masih terdapat perubahan bobot, lanjutkan pada proses pelatihan berikutnya.

Dapat disimpulkan bahwa proses updating bobot terjadi jika beberapa kondisi di bawah ini dipenuhi :

- Sinyal masukan pada unit input tidak sama dengan 0.
- Masih terdapat selisih / kesalahan pada unit output (y) dibandingkan terhadap target (t).

Nilai threshold fungsi aktifasi untuk unit output selalu berharga positif θ. Bentuk fungsi aktifasi tersebut memang diatur sedemikian rupa sehingga selalu menghasilkan keputusan daerah positif dan daerah negatif. (tidak mungkin tidak menghasilkan keputusan)

Sebagai contoh untuk kasus di mana perceptron memiliki dua buah input saja, maka *daerah positif* dipisahkan oleh sebuah garis dengan pertidaksamaan :

$$W_1.X_1 + W_2.X_2 + b > \theta \tag{8.17}$$

Sedangkan *daerah negatif* dipisahkan oleh sebuah garis dengan pertidaksamaan :

$$W_1.X_1 + W_2.X_2 + b < -\theta \tag{8.18}$$

CONTOH

Sebuah Perceptron dengan dua buah input akan digunakan untuk mengklasifikasikan fungsi AND pada kasus dua dimensi. Input adalah biner sedangkan target adalah bipolar. Pasangan pola **s:t** dari fungsi AND ditunjukkan dalam tabel di bawah ini:

Note:

Pola	x_b	x_1	x_2	t
1	1	1	1	1
2	1	1	0	-1
3	1	0	1	-1
4	1	0	0	-1

Asumsikan bahwa semua variabel bobot, W, dan bias , b, diinisialisasi dengan nilai 0. Konstanta percepatan $\alpha = 1$ dan $\theta = 0.2$. Lakukan proses pembelajaran terhadap perceptron tersebut sehingga perceptron mampu melakukan klasifikasi fungsi AND dengan baik!

Proses pembelajaran

Iterasi 1

Pola pertama

Input			Net	Out	Target	W_1	W_2	b
x_1	x_2	x_b				0	0	0
1	1	1	0	0	1	1	1	1

Berdasarkan hasil pada iterasi 1 dan pola input pertama didapatkan persamaan garis:

$$x_1 + x_2 + 1 = 0.2$$

dan

$$x_1 + x_2 + 1 = -0.2$$

Gambar 8.15 menunjukkan respon perceptron untuk input pertama tersebut.

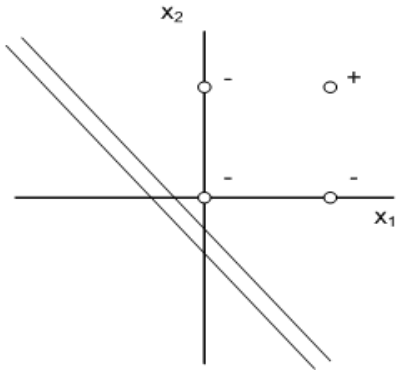

Figure 8.15: Perceptron untuk fungsi AND: pola input pertama.

Seperti terlihat dalam gambar, kedua garis belum dapat memilah/memisahkan antara daerah positif dan negatif dari fungsi AND. Karena itu pembelajaran/training dilanjutkan dengan mengambil pasangan pola input dan target kedua.

Pola kedua

Input			Net	Out	Target	W_1	W_2	b
x_1	x_2	x_b				1	1	1
1	0	1	2	1	-1	0	1	0

Berdasarkan hasil pada iterasi 1 dan pola input kedua didapatkan persamaan garis:

$$x_2 = 0.2$$

dan

$$x_2 = -0.2$$

Gambar 8.16 menunjukkan respon perceptron untuk input kedua.

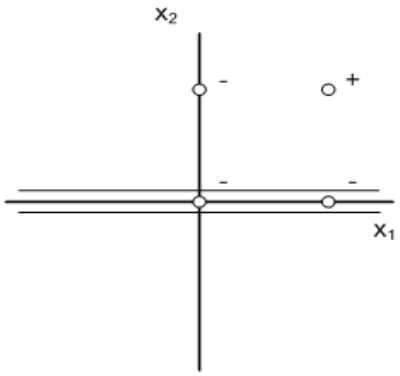

Figure 8.16: Perceptron untuk fungsi AND: pola input kedua.

Sama halnya seperti pada pola input pertama, pembelajaran masih belum menghasilkan kondisi yang diinginkan.

Pola ketiga

Input			Net	Out	Target	W_1	W_2	b
x_1	x_2	x_b				0	1	0
0	1	1	1	1	-1	0	0	-1

Sampai di sini terlihat bahwa komponen bobot tidak positif, maka tanggapan perceptron akan negatif atau nol. (tidak fire)

Pola keempat

Input			Net	Out	Target	W_1	W_2	b
x_1	x_2	x_b				0	0	-1
0	0	1	-1	-1	-1	0	0	-1

sekali lagi, tanggapan perceptron negatif.

Terlihat bahwa sampai iterasi 1 selesai, keluaran dari perceptron masih belum benar, maka proses pelatihan dilakukan untuk iterasi kedua dan seterusnya. Secara lengkap, keseluruhan proses pelatihan ditunjukkan dalam tabel di bawah ini:

Table 8.3: Proses pelatihan perceptron untuk fungsi AND

Iterasi 2								
Input			Net	Out	Target	W_1	W_2	b
x_1	x_2	x_b				0	0	-1
1	1	1	-1	-1	1	1	1	0
1	0	1	1	1	-1	0	1	-1
0	1	1	0	0	-1	0	0	-2
0	0	1	-2	-1	-1	0	0	-2

Iterasi 3								
Input			Net	Out	Target	W_1	W_2	b
x_1	x_2	x_b				0	0	-2
1	1	1	-2	-1	1	1	1	-1
1	0	1	0	0	-1	0	1	-2
0	1	1	-1	-1	-1	0	1	-2
0	0	1	-2	-1	-1	0	1	-2
Iterasi 4								
1	1	1	-1	-1	1	1	2	-1
1	0	1	0	0	-1	0	2	-2
0	1	1	0	0	-1	0	1	-3
0	0	1	-3	-1	-1	0	1	-3
Iterasi 5								
1	1	1	-2	-1	1	1	2	-2
1	0	1	-1	-1	-1	1	2	-2
0	1	1	0	0	-1	1	1	-3
0	0	1	-3	-1	-1	1	1	-3

Iterasi 6								
Input			Net	Out	Target	W_1	W_2	b
1	1	1	-1	-1	1	2	2	-2
1	0	1	0	0	-1	1	2	-3
0	1	1	-1	-1	-1	1	2	-3
0	0	1	-3	-1	-1	1	2	-3
Iterasi 7								
1	1	1	0	0	1	2	3	-2
1	0	1	0	0	-1	1	3	-3
0	1	1	0	0	-1	1	2	-4
0	0	1	-4	-1	-1	1	2	-4
Iterasi 8								
x_1	x_2	x_b				1	2	-4
1	1	1	-1	-1	1	2	3	-3
1	0	1	-1	-1	-1	2	3	-3
0	1	1	0	0	-1	2	2	-4
0	0	1	-4	-1	-1	2	2	-4
Iterasi 9								
1	1	1	0	0	1	3	3	-3
1	0	1	0	0	-1	2	3	-4
0	1	1	-1	-1	-1	2	3	-4
0	0	1	-4	-1	-1	2	3	-4
Iterasi 10								
1	1	1	1	1	1	2	3	-4
1	0	1	-2	-1	-1	2	3	-4
0	1	1	-1	-1	-1	2	3	-4
0	0	1	-4	-1	-1	2	3	-4

Sampai proses pelatihan putaran kesepuluh tidak lagi terjadi perubahan bobot, dan nilai ouput telah sama dengan nilai target. Dengan

demikian keseluruhan proses pelatihan telah selesai.

Persamaan garis pada putaran kesepuluh adalah :

$$2x_1 + 3x_2 - 4 > 0.2 \tag{8.19}$$

dan

$$2x_1 + 3x_2 - 4 < -0.2 \tag{8.20}$$

Respons dari perceptron sampai putaran kesepuluh ditunjukkan dalam Gambar 8.17. Seperti terlihat dalam gambar, sekarang keluaran dari perceptron telah berhasil memisahkan daerah positif dan negatif dari fungsi AND.

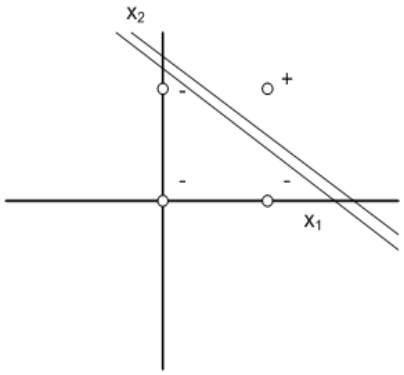

Figure 8.17: Perceptron untuk fungsi AND: iterasi kesepuluh.

Sampai di sini proses pembelajaran telah selesai. Hasil dari proses pembelajaran ini adalah satu set varibel bobot yang bernilai: $W_1 = 2$,$W_2 = 3$ dan $b = -4$. Variabel bobot ini disimpan dalam memori sebagai semacam ingatan bagi perceptron. Sekarang mari kita uji cobakan hasil pembelajaran tersebut dengan inputan biner. Konfigurasi dari perceptron setelah proses pembelajaran adalah sebagai berikut:

Jika nilai $x_1 = 1$ dan $x_2 = 0$ kita inputkan, maka perceptron akan memberikan respons nilai $y = -1$. Jika inputan sekarang dirubah menjadi $x_1 = 1$ dan $x_2 = 1$ maka repons dari perceptron adalah $y = 1$. Dengan demikian perceptron telah dapat mengenali fungsi AND dengan baik. Atau dengan kata lain: *perceptron telah berhasil meniru fungsi*

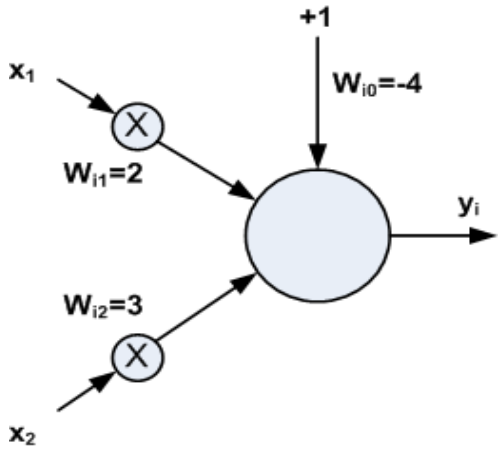

Figure 8.18: Perceptron untuk fungsi AND: uji coba.

AND. Masih ingat penjelasan tentang blackbox ? Hasil pelatihan perceptron inilah yang kita isikan ke dalam blackbox untuk fungsi AND.

8.7 Back Error Propagation (Propagasi Balik Kesalahan)

Note:

David E Rummelhart dan McClelland telah melakukan penelitian tentang proses belajar yang digunakan untuk Jaringan berlapis banyak (Multilayer Perceptron - MLP). Kaidah belajar ini disebut sebagai Propagasi Balik Kesalahan (Back Error Propagation / BEP)

Kesalahan lokal tiap sel dilihat sebagai bagian yang berkontribusi dalam menghasilkan kesalahan total pada lapis keluaran. Apabila kesalahan pada lapis keluaran dapat dipropagasikan kembali masuk ke lapis dalam, maka kesalahan lokal sel-sel pada lapis tersebut dapat dihitung.

Arsitektur dari Multilayer Perceptron ditunjukkan dalam Gambar 8.19.

Seperti terlihat dalam gambar, MLP terdiri atas 3 lapisan, yaitu: *lapisan input*, *lapisan hidden* dan *lapisan output*. Jumlah node pada

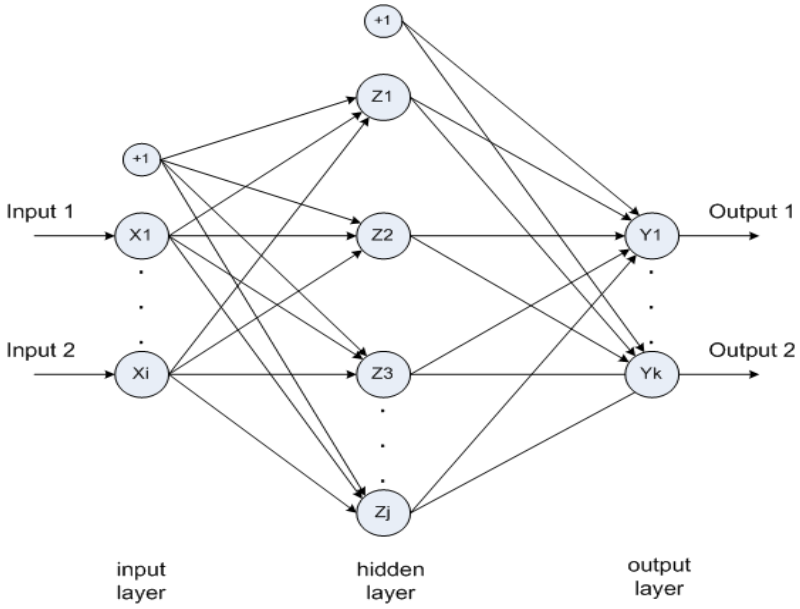

Figure 8.19: Arsitektur Multilayer Perceptron.

lapisan input mengikuti jumlah input dari aplikasi yang akan diajarkan pada JST, demikian pula jumlah node pada lapisan output. Sedangkan lapisan hidden mungkin sekali terdiri atas beberapa lapisan lagi (tidak hanya 1 lapis) terdiri atas kompleksitas dari aplikasi. Namun secara umum masih belum ada rumusan yang tepat untuk menentukan jumlah node yang dibutuhkan pada lapisan hidden.

Operasi algoritma Back Error Propagation pada jaringan multilayer perceptron secara umum terdiri atas dua hal, yaitu: propagasi maju dan propagasi balik. Pada propagasi maju dilakukan perhitungan untuk menentukan respon keluaran dari jaringan. Sedangkan propagasi balik digunakan untuk melakukan mencari nilai kesalahan antara target dengan keluaran aktual dari jaringan serta melakukan update terhadap setiap bobot pada semua lapisan (dengan demikian BEP merupakan supervised learning).

8.7.1 Propagasi maju

Note:

Maka sesuai dengan arsitektur di atas perhitungan pada propagasi maju adalah sebagai berikut:

Pada lapis dalam (hidden layer). Masukan dari setiap node (neuron) pada lapis dalam berasal dari penjumlahan node input sebagai berikut:

$$net_j^h = \sum_{i=1}^{N_i} W_{ji}^h \, X_i + \theta_j^h. \tag{8.21}$$

Sehingga keluaran dari fungsi aktifasi sigmoid pada hidden layer adalah:

$$H_j = f_j^h(net_j^h). \tag{8.22}$$

Pada lapis keluaran (output layer). Masukan dari setiap node (neuron) pada lapis keluaran berasal dari penjumlahan node pada lapis dalam sebagai berikut:

$$net_k^o = \sum_{j=1}^{N_j} W_{kj}^o \, H_j + \theta_k^o. \tag{8.23}$$

Sehingga keluaran dari fungsi aktifasi sigmoid pada hidden layer adalah:

$$O_k = f_k^o(net_k^o). \tag{8.24}$$

Sampai di sini proses propagasi maju telah selesai. Selanjutnya hasil keluaran pada output layer dibandingkan dengan target, apabila terdapat kesalahan (selisih antara target dan keluaran aktual) maka nilai kesalahan tersebut dipropagasikan balik dengan tujuan untuk mengupdate seluruh bobot yang ada pada multilayer perceptron.

8.7.2 Propagasi balik

Note:

Pada lapis keluaran, kesalahan pada sebuah node keluaran didefinisikan sebagai :

$$\delta_k = Y_k - O_k, \tag{8.25}$$

dimana Y_k adalah nilai target/acuan atau nilai keluaran yang diinginkan dan O_k adalah keluaran aktual dari multilayer perceptron.

Maka error square pada lapis keluaran adalah:

$$\begin{aligned} E &= \frac{1}{2}\sum_{k=1}^{N_k}\delta_j^2 \\ &= \frac{1}{2}\sum_{k=1}^{N_k}(Y_k - O_k)^2 \end{aligned} \tag{8.26}$$

Proses updating dari nilai bobot dimulai dengan cara meminimalkan turunan dari error tersebut di atas terhadap variabel bobot pada node keluaran:

$$\frac{\partial E}{\partial W_{kj}^o} = -(Y_k - O_k)\frac{\partial f_k^o}{\partial (net_k^o)} \cdot \frac{\partial (net_k^o)}{\partial W_{kj}^o}. \tag{8.27}$$

Sisi kanan dari persamaan di atas dapat dijabarkan sebagai berikut:

$$\frac{\partial (net_k^o)}{\partial W_{kj}^o} = \left[\frac{\partial}{\partial W_{kj}^o}\sum_{j=1}^{N_j} W_{kj}^o\ H_j + \theta_k^o\right], \tag{8.28}$$

$$= H_j \tag{8.29}$$

Dengan demikian persamaan pada 8.27 dapat ditulis kembali menjadi:

$$-\frac{\partial E}{\partial W_{kj}^o} = (Y_k - O_k) f_k^{o'}(net_k^o) \cdot H_j. \tag{8.30}$$

Dari di atas dapat diketahui bahwa syarat agar persamaan perubahan bobot dapat dipecahkan, fungsi aktifasi sel harus bersifat dapat diturunkan (diflerentiable), contoh: fungsi sigmoid.

Persamaan di atas juga berarti bahwa parameter jaringan yaitu bobot-bobot koneksi dari sel j ke sel i, W_{ij}, harus diubah sebanding dengan negatif gradien fungsi kesalahan terhadap perubahan bobot:

$$\triangle W_{kj}^o = -\eta \cdot \frac{\partial E}{\partial W_{kj}^o}, \tag{8.31}$$

dimana $\triangle W_{kj}^o$ adalah perubahan bobot node j ke node k, dan η adalah konstanta kecepatan belajar ($0 \leq \eta \leq 1$).

Dengan demikian persamaan di atas dapat dimodifikasi menjadi:

$$\triangle W_{kj}^{o} = \eta(Y_k - O_k) f_k^{o'}(net_k^o) \cdot H_j. \tag{8.32}$$

Selanjutnya bobot yang baru pada lapis keluaran dapat diupdate dengan persamaan :

$$W_{kj}^{o}(new) = W_{kj}^{o}(old) + \triangle W_{kj}^{o} \tag{8.33}$$

Untuk menyederhanakan Persamaan 8.32 diambil konstanta baru, ∂_k^o, yaitu :

$$\begin{aligned} \partial_k^o &= (Y_k - O_k) f_k^{o'}(net_k^o) \\ &= \delta_k f_k^{o'}(net_k^o). \end{aligned} \tag{8.34}$$

Sehingga persamaan 8.33 dapat ditulis kembali menjadi:

$$W_{kj}^{o}(new) = W_{kj}^{o}(old) + \eta \partial_k^o \cdot H_j. \tag{8.35}$$

Lebih lanjut, kesalahan (error) dipropagasikan ke lapisan dalam untuk mengupdate nilai bobot pada lapisan dalam. Persamaan penyesuaian bobot pada lapis dalam adalah :

$$W_{ji}^{h}(new) = W_{ji}^{h}(old) + \eta \partial_j^h \cdot X_i, \tag{8.36}$$

dimana nilai dari ∂_j^h pada persamaan di atas adalah:

$$\partial_j^h = f_j^{h'}(net_j^h) \sum_{k=1}^{N_k} \partial_k^o \cdot W_{kj}^{o}. \tag{8.37}$$

Secara ringkas, algoritma dari Back Error Propagation (BEP) yang diimplementasikan pada multilayer perceptron dapat disarikan sebagai berikut:

Langkah 0 Inisialisasi bobot (weights) dengan bilangan acak.

Langkah 1 Jika kondisi berhenti tidak terpenuhi, lakukan 2 - 9.

Langkah 2 Untuk masing-masing pasangan pelatihan, lakukan langkah 3 - 8.

Propagasi Maju

Langkah 3 Masing-masing unit input menerima sinyal input X_i dan menyebarkan ke semua unit pada lapis di depannya.

Langkah 4 Pada masing-masing unit lapis dalam jumlahkan dengan :

$$net_j^h = \sum_{i=1}^{N_i} W_{ji}^h \, X_i + \theta_j^h.$$

Hitung fungsi aktifasi pada setiap unit lapis dalam:

$$H_j = f_j^h(net_j^h).$$

Langkah 5 Pada masing-masing unit lapis output jumlahkan dengan :

$$net_k^o = \sum_{j=1}^{N_j} W_{kj}^o \, H_j + \theta_k^o.$$

Hitung fungsi aktifasi pada setiap unit lapis keluaran:

$$O_k = f_k^o(net_k^o).$$

Propagasi Balik Kesalahan

Langkah 6 Hitung Error pada masing-masing unit lapis output:

$$\partial_k^o = (Y_k - O_k) f_k^{o'}(net_k^o)$$

Hitung kenaikan nilai bobot :

$$\triangle W_{kj}^o = \eta \partial_k^o net_j^h$$

Hitung kenaikan nilai bobot unit bias :

$$\triangle W_{0j}^o = \eta \partial_k^o$$

dimana $f_k^{o'} = f_k^o(net_k^o) \lfloor 1 - f_k^o(net_k^o \rfloor$

Langkah 7 Hitung informasi kesalahan pada masing-masing unit lapis dalam :

$$\partial_j^h = f_j^{h'}(net_j^h) \sum_{k=1}^{N_j} \partial_k^o \cdot W_{kj}^o.$$

Hitung kenaikan nilai bobot:

$$\triangle W_{ji}^h = \eta \partial_j^h \cdot X_i$$

Hitung kenaikan nilai bobot unit bias:

$$\triangle W_{0i}^h = \eta \partial_j^h$$

dimana $f_j^{h'} = f_j^h(net_j^h)\lfloor 1 - f_j^h(net_j^h \rfloor$

Langkah 8 Lakukan updating semua bobot antara lapis dalam dan lapis output :

$$W_{kj}^o(new) = W_{kj}^o(old) + \triangle W_{kj}^o$$

Lakukan updating semua bobot antara lapis input dan lapis dalam :

$$W_{ji}^h(new) = W_{ji}^h(old) + \triangle W_{ji}^h$$

Langkah 9 Uji kondisi berhenti :

$$E = \frac{1}{2} \sum_{k=1}^{N_k} (Y_k - O_k)^2$$

Proses pelatihan dihentikan apabila kondisi berhenti telah sesuai dengan ketelitian yang kita harapkan.

8.8 Jaringan Syaraf Tiruan untuk Aplikasi Time-Series

Karena penelitian-penelitian terhadap aplikasi JST telah berkembang cukup luas, maka dalam literatur ada terdapat banyak sekali arsitektur JST yang secara khusus dipergunakan untuk menyelesaikan permasalahan-permasalahan time-series. Salah satu model yang akan dibahas di sini adalah JST dengan arsitektur Elman (Fausett, 1994). Arsitektur dari jaringan Elman ditunjukkan dalam Gambar 8.20.

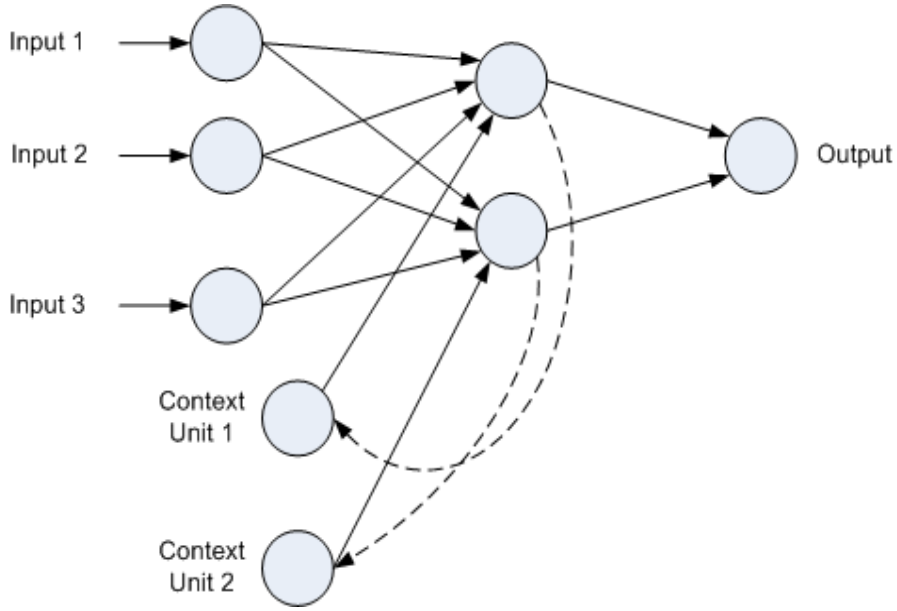

Figure 8.20: JST dengan Arsitektur Elman.

Pada dasarnya proses pelatihan/pembelajaran/training pada jaringan Elman tidak berbeda dengan proses training yang menggunakan back-error propagation. Seluruh nilai dari bobot (weight) diupdate dengan menggunakan algoritma back-error propagation. Namun, seperti terlihat dalam gambar, jaringan Elman memiliki umpan balik yang menghubungkan keluaran dari hidden layer dengan context unit. Pada saat t, context unit akan menerima input yang berasal dari keluaran hidden layer pada saat $t-1$, memproses dan mengirimkan kembali ke hidden layer untuk dijum-

lahkan dengan keluaran dari input layer.

8.9 Contoh aplikasi dengan JST

Note:

Beberapa contoh aplikasi yang telah diimplementasikan dengan menggunakan Jaringan Syaraf Tiruan antara lain:

- Investment analysis: JST digunakan untuk memprediksikan pergerakan harga saham, valas dll.
- Signature analysis: digunakan untuk mengenali tanda-tangan, misalnya pada bank, dengan cara membandingkan bentuk tanda-tangan yang dibuat dengan yang tersimpan. Aplikasi ini secara luas digunakan di USA dan merupakan aplikasi neural network pertama yang diimplementasikan dalam bentuk chip.
- Process control: sebagian besar didalam industri tidak dapat ditentukan hanya dengan menggunakan algoritma-algoritma komputasi. Newcastle University Chemical Engineering Department bekerjasama dengan beberapa industri (seperti Zeneca and BP) untuk membuat aplikasi ini.
- Monitoring: neural network digunakan dalam monitoring, contohnya kondisi dari mesin pesawat terbang dimonitor dengan cara memonitor level getaran dari mesin sebagai peringatan awal apabila terjadi kerusakan. British Rail juga menggunakan aplikasi yang sama untuk memonitor mesin diesel.
- Marketing: neural network dapat digunakan untuk membuat pola tingkah laku customer terhadap produk-produk tertentu, pengaruh dari sebuah produk terhadap kondisi geografis dsb. Dengan demikian

neural-network akan sangat membantu proses marketing dalam hal menentukan strategi-strategi bisnis baru.

8.10 SOAL LATIHAN

1. Lakukan proses pelatihan perceptron untuk fungsi AND seperti di atas dengan kombinasi input dan target bipolar!

2. Perceptron dengan 1 buah neuron akan digunakan untuk memodelkan fungsi dari pembuka pintu air bendungan. Ketinggian air saat itu akan menjadi inputan bagi neuron dan aksi dari pembukaan pintu merupakan keluaran. Jika ketinggian air di atas 100cm maka pintu bendungan terbuka (+1), jika ketinggian air dibawah 100cm atau 50cm maka pintu tertutup (-1). Dalam hal ini, ketinggian air di atas 100cm disimbolkan sebagai X3, ketinggian dibawah 100cm disimbolkan sebagai X2 dan ketinggian di bawah 50 cm disimbolkan sebagai X1. Aksi dari buka tutup pintu disimbolkan sebagai Y. Tabel keputusan untuk pengatur pintu air tersebut ditunjukkan sebagai berikut:

Pola	X_1	X_2	X_3	X_b	Y
1	-1	-1	1	1	1
2	-1	1	-1	1	-1
3	1	-1	-1	1	-1

Parameter-parameter bagi perceptron ditentukan sebagai berikut: $\alpha = 1$; threshold(θ)=0,1; inisialisasi semua bobot dan bias = 0.

Lakukan proses pembelajaran perceptron tersebut untuk menentukan konfigurasi nilai bobot akhir, sehingga setelah pelatihan ini perceptron akan mengatur buka dan tutup pintu bendungan secara otomatis.