

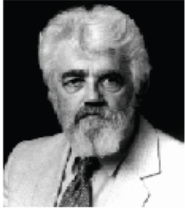
I. PENGANTAR KECERDASAN BUATAN

Pengampu : Idhawati Hestiningsih

DEFINISI

Kecerdasan buatan (Artificial Intelligence) :

Bagian dari ilmu komputer yang mempelajari bagaimana membuat mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia bahkan bisa lebih baik daripada yang dilakukan manusia.



Menurut John McCarthy, 1956, AI :

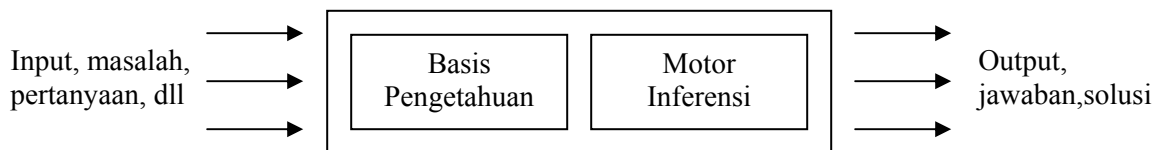
Untuk mengetahui dan memodelkan proses – proses berpikir manusia dan mendesain mesin agar dapat menirukan perilaku manusia.

Cerdas = memiliki pengetahuan + pengalaman, penalaran (bagaimana membuat keputusan & mengambil tindakan), moral yang baik

Agar mesin bisa cerdas (bertindak seperti & sebaik manusia) maka harus diberi bekal pengetahuan & mempunyai kemampuan untuk menalar.

2 bagian utama yg dibutuhkan untuk aplikasi kecerdasan buatan :

- a. basis pengetahuan (knowledge base): berisi fakta-fakta, teori, pemikiran & hubungan antara satu dengan lainnya.
- b. motor inferensi (inference engine) : kemampuan menarik kesimpulan berdasarkan pengetahuan



BEDA KECERDASAN BUATAN & KECERDASAN ALAMI

Kelebihan kecerdasan buatan :

1. Lebih bersifat permanen. Kecerdasan alami bisa berubah karena sifat manusia lupa. Kecerdasan buatan tidak berubah selama sistem komputer & program tidak mengubahnya.
2. Lebih mudah diduplikasi & disebar. Mentransfer pengetahuan manusia dari 1 orang ke orang lain membutuhkan proses yang sangat lama & keahlian tidak akan pernah dapat diduplikasi dengan lengkap. Jadi jika pengetahuan terletak pada suatu sistem komputer, pengetahuan tersebut dapat disalin dari komputer tersebut & dapat dipindahkan dengan mudah ke komputer yang lain.
3. Lebih murah. Menyediakan layanan komputer akan lebih mudah & murah dibandingkan mendatangkan seseorang untuk mengerjakan sejumlah pekerjaan dalam jangka waktu yang sangat lama.
4. Bersifat konsisten dan teliti karena kecerdasan buatan adalah bagian dari teknologi komputer sedangkan kecerdasan alami senantiasa berubah-ubah
5. Dapat didokumentasi. Keputusan yang dibuat komputer dapat didokumentasi dengan mudah dengan cara melacak setiap aktivitas dari sistem tersebut. Kecerdasan alami sangat sulit untuk direproduksi.
6. Dapat mengerjakan beberapa task lebih cepat dan lebih baik dibanding manusia

Kelebihan kecerdasan alami :

1. Kreatif : manusia memiliki kemampuan untuk menambah pengetahuan, sedangkan pada kecerdasan buatan untuk menambah pengetahuan harus dilakukan melalui sistem yang dibangun.
2. Memungkinkan orang untuk menggunakan pengalaman atau pembelajaran secara langsung. Sedangkan pada kecerdasan buatan harus mendapat masukan berupa input-input simbolik.
3. Pemikiran manusia dapat digunakan secara luas, sedangkan kecerdasan buatan sangat terbatas.

BEDA KECERDASAN BUATAN & PROGRAM KONVENSIONAL

	Kecerdasan buatan	Program konvensional
Fokus pemrosesan	Konsep simbolik / numerik (pengetahuan)	Data & informasi
Pencarian	Heuristik	Algoritma
Sifat input	Bisa tidak lengkap	Harus lengkap
Keterangan	Disediakan	Biasanya tidak disediakan
Struktur	Kontrol dipisahkan dari pengetahuan	Kontrol terintegrasi dengan informasi (data)
Sifat output	Kuantitatif	Kualitatif
Kemampuan menalar	Ya	Tidak

Program kecerdasan buatan dapat ditulis dalam semua bahasa komputer, baik dalam bahasa C, Pascal, Basic, dan bahasa pemrograman lainnya. Tetapi dalam perkembangan selanjutnya, dikembangkan bahasa pemrograman yang khusus untuk aplikasi kecerdasan buatan yaitu LISP dan PROLOG.

SEJARAH KECERDASAN BUATAN

Tahun 1950 – an Alan Turing, seorang pionir AI dan ahli matematika Inggris melakukan percobaan Turing (Turing Test) yaitu sebuah komputer melalui terminalnya ditempatkan pada jarak jauh. Di ujung yang satu ada terminal dengan software AI dan diujung lain ada sebuah terminal dengan seorang operator. Operator itu tidak mengetahui kalau di ujung terminal lain dipasang software AI. Mereka berkomunikasi dimana terminal di ujung memberikan respon terhadap serangkaian pertanyaan yang diajukan oleh operator. Dan sang operator itu mengira bahwa ia sedang berkomunikasi dengan operator lainnya yang berada pada terminal lain.

Turing beranggapan bahwa jika mesin dapat membuat seseorang percaya bahwa dirinya mampu berkomunikasi dengan orang lain, maka dapat dikatakan bahwa mesin tersebut cerdas (seperti layaknya manusia).

KECERDASAN BUATAN PADA APLIKASI KOMERSIAL

Lingkup utama kecerdasan buatan :

- 1. Sistem pakar (expert system) : komputer sebagai sarana untuk menyimpan pengetahuan para pakar sehingga komputer memiliki keahlian menyelesaikan permasalahan dengan meniru keahlian yang dimiliki pakar.

Diagnosa Penyakit THT

Apakah Anda demam (Y/T) ? y

Apakah Anda sakit kepala (Y/T) ? y

Apakah Anda merasa nyeri pada saat berbicara atau menelan (Y/T) ? y

Apakah Anda batuk (Y/T) ? y

Apakah Anda mengalami nyeri tenggorokan (Y/T) ? y

Apakah selaput lendir Anda berwarna merah dan bengkak (Y/T) ? y

Penyakit Anda adalah TONSILITIS

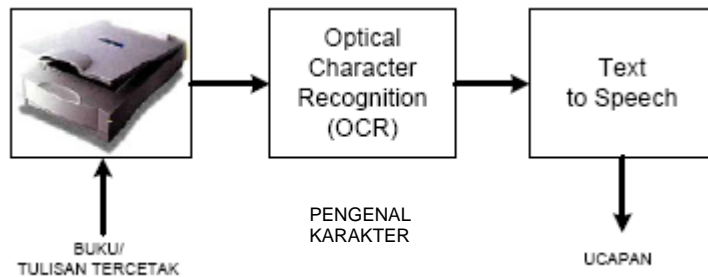
Ingin mengulang lagi (Y/T) ?

- 2. Pengolahan bahasa alami (natural language processing) : user dapat berkomunikasi dengan komputer menggunakan bahasa sehari-hari, misal bahasa inggris, bahasa indonesia, bahasa jawa, dll, contoh :
 - pengguna sistem dapat memberikan perintah dengan bahasa sehari-hari, misalnya, untuk menghapus semua file, pengguna cukup memberikan perintah **"komputer, tolong hapus semua file !"** maka sistem akan mentranslasikan perintah bahasa alami tersebut menjadi perintah bahasa formal yang dipahami oleh komputer, yaitu **"delete *.* <ENTER>"**.
 - Translator bahasa inggris ke bahasa indonesia begitu juga sebaliknya,dll, tetapi sistem ini tidak hanya sekedar kamus yang menerjemahkan kata per kata, tetapi juga mentranslasikan sintaks dari bahasa asal ke bahasa tujuan
 - Text summarization : suatu sistem yang dapat membuat ringkasan hal-hal penting dari suatu wacana yang diberikan.

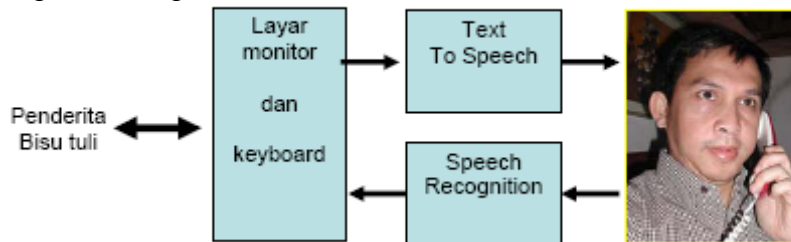
3. Pengenalan ucapan (speech recognition) : manusia dapat berkomunikasi dengan komputer menggunakan suara.

Contoh :

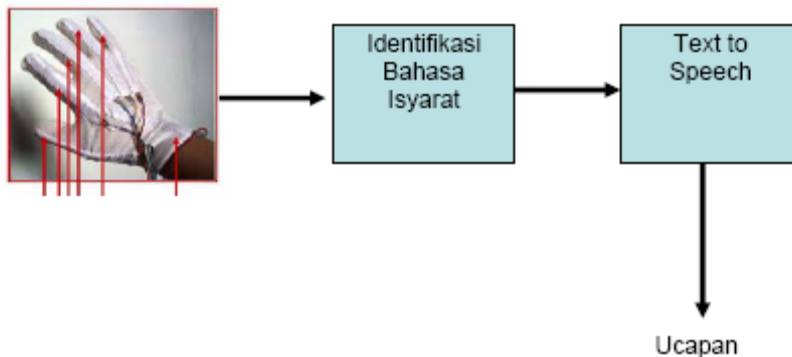
- memberikan instruksi ke komputer dengan suara
- alat bantu membaca untuk tunanetra, mempunyai masukan berupa teks tercetak (misalnya buku) dan mempunyai keluaran berupa ucapan dari teks tercetak yang diberikan.



- Telpn untuk penderita bisu-tuli



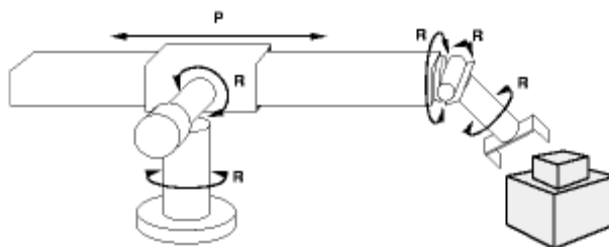
Alat untuk tuna wicara




- konversi dari SMS (*Short Message System*) ke ucapan sehingga pesan SMS dapat didengar. Dengan demikian memungkinkan untuk mendengar pesan SMS sambil melakukan aktivitas yang menyulitkan untuk membacanya, seperti mengendarai mobil.

4. Robotika & sistem sensor


- Sistem sensor pada mesin cuci yaitu menggunakan sensor optik, mengeluarkan cahaya ke air dan mengukur bagaimana cahaya tersebut sampai ke ujung lainnya. Makin kotor, maka sinar yang sampai makin redup. Sistem juga mampu menentukan jenis kotoran tersebut daki/minyak. Sistem juga bisa menentukan putaran yang tepat secara otomatis berdasarkan jenis dan banyaknya kotoran serta jumlah yang akan dicuci.
- Robotika



5. Computer vision : menginterpretasikan gambar atau objek-objek tampak melalui komputer



Pengenalan pola sidik jari seseorang

Designated Writing Area			Identity
Chinese:	English:	Numeral:	
	-----	9	9
	-----		No.1
			d=24.02
			s=0.650


Date from William Go

Total: 20 Done: 20 First: 20 Second: 0 Third: 0 Other: 0 Reject: 0

0 1 2 3 4 5 6 7 8 9

0 1 2 3 4 5 6 7 8 9

Pengenalan angka



(a) (b)

(c) (d)

Pengenalan/deteksi tanda tangan asli/palsu (tanda tangan yang dibuat oleh orang yang sama/berbeda)

6. Intelligent computer-aided instruction : komputer dapat digunakan sebagai tutor yang dapat melatih & mengajar
- Contoh : Learn to speak English



7. Game playing



1997, Deep Blue mengalahkan Garry Kasparov, the World Chess Champion

Deep Blue chess machine menggunakan komputer IBM, dibuat tahun 1990-an oleh Hsu, Campbell, Tan, Hoane, Brody, Benjamin

Deep Blue mampu mengevaluasi 200juta posisi bidak catur /detik



SOFT Computing

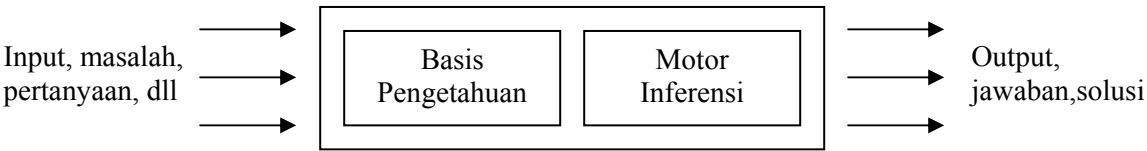
Soft computing merupakan inovasi baru dalam membangun sistem cerdas yaitu sistem yang memiliki keahlian seperti manusia pada domain tertentu, mampu beradaptasi dan belajar agar dapat bekerja lebih baik jika terjadi perubahan lingkungan. Soft computing mengeksplorasi adanya toleransi terhadap ketidaktepatan, ketidakpastian, dan kebenaran parsial untuk dapat diselesaikan dan dikendalikan dengan mudah agar sesuai dengan realita (Prof. Lotfi A Zadeh, 1992).

Metodologi-metodologi yang digunakan dalam Soft computing adalah :

1. Sistem Fuzzy (mengakomodasi ketidaktepatan) → Logika Fuzzy (fuzzy logic)
2. Jaringan Syaraf (menggunakan pembelajaran) → Jaringan Syaraf Tiruan (neurall network)
3. Probabilistic Reasoning (mengakomodasi ketidakpastian)
4. Evolutionary Computing (optimasi) → Algoritma Genetika

II. MASALAH DAN METODE PEMECAHAN MASALAH

Sistem yang menggunakan kecerdasan buatan akan memberikan output berupa solusi dari suatu masalah berdasarkan kumpulan pengetahuan yang ada.



Gambar sistem yang menggunakan kecerdasan buatan

Pada gambar, input yg diberikan pada sistem yg menggunakan kecerdasan buatan adalah berupa masalah. Sistem harus dilengkapi dengan sekumpulan pengetahuan yang ada pada basis pengetahuan. Sistem harus memiliki motor inferensi agar mampu mengambil kesimpulan berdasarkan fakta atau pengetahuan. Output yang diberikan berupa solusi masalah sebagai hasil dari inferensi.

Secara umum, untuk membangun suatu sistem yang mampu menyelesaikan masalah, perlu dipertimbangkan 4 hal :

- 1. Mendefinisikan masalah dengan tepat.
Pendefinisian ini mencakup spesifikasi yang tepat mengenai *keadaan awal* dan *solusi yang diharapkan*.
- 2. Menganalisis masalah tersebut serta mencari beberapa teknik penyelesaian masalah yang sesuai.
- 3. Merepresentasikan pengetahuan yang perlu untuk menyelesaikan masalah tersebut.
- 4. Memilih teknik penyelesaian masalah yang terbaik

MENDEFINISIKAN MASALAH SEBAGAI SUATU RUANG KEADAAN

Misalkan permasalahan yang dihadapi adalah permainan catur, maka harus ditentukan :

- 1. posisi awal pada papan catur
posisi awal setiap permainan catur selalu sama, yaitu semua bidak diletakkan di atas papan catur dalam 2 sisi, yaitu kubu putih dan kubu hitam.
- 2. aturan – aturan untuk melakukan gerakan
aturan – aturan ini sangat berguna untuk menentukan gerakan suatu bidak, yaitu melangkah dari satu keadaan ke keadaan lain. Misalkan untuk mempermudah menunjukkan posisi bidak, setiap kotak ditunjukkan dalam huruf (a,b,c,d,e,f,g,h) pada arah horisontal dan angka (1,2,3,4,5,6,7,8) pada arah vertikal. Suatu aturan untuk menggerakkan bidak dari posisi (e,2) ke (e,4) dapat ditunjukkan dengan aturan :
if bidak putih pada kotak(e,2),
and kotak(e,3) kosong,
and kotak(e,4) kosong
then gerakkan bidak dari (e,2) ke (e,4)
- 3. tujuan (goal)
tujuan yang ingin dicapai adalah posisi pada papan catur yang menunjukkan kemenangan seseorang terhadap lawannya. Kemenangan ini ditandai dengan posisi raja yang sudah tidak dapat bergerak lagi.

Contoh tersebut menunjukkan representasi masalah dalam Ruang Keadaan (State Space), yaitu suatu ruang yang berisi semua keadaan yang mungkin. Kita dapat memulai bermain catur dengan menempatkan diri pada keadaan awal, kemudian bergerak dari satu keadaan ke keadaan yang lain sesuai dengan aturan yang ada, dan mengakhiri permainan jika salah satu telah mencapai tujuan.

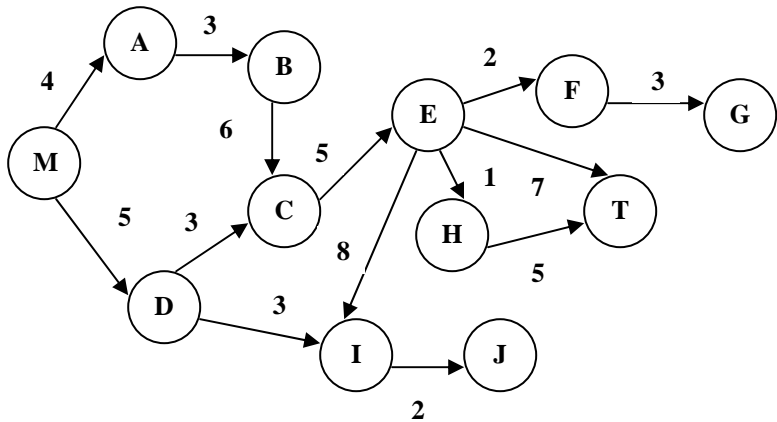
Jadi untuk *mendeskrripsikan masalah dengan baik* harus :

- 1. Mendefinisikan suatu ruang keadaan (state space)
- 2. Menetapkan satu atau lebih keadaan awal (initial state)
- 3. Menetapkan satu atau lebih tujuan (goal state)
- 4. Menetapkan kumpulan aturan

Ada beberapa cara untuk merepresentasikan Ruang Keadaan, antara lain :

GRAPH KEADAAN

Graph terdiri dari node-node yang menunjukkan keadaan yaitu keadaan awal dan keadaan baru yang akan dicapai dengan menggunakan operator. Node-node dalam graph keadaan saling dihubungkan dengan menggunakan arc (busur) yang diberi panah untuk menunjukkan arah dari suatu keadaan ke keadaan berikutnya.



Graph keadaan dengan node M menunjukkan keadaan awal, node T adalah tujuan. Ada 4 lintasan dari M ke T :

- M-A-B-C-E-T
- M-A-B-C-E-H-T
- M-D-C-E-T
- M-D-C-E-H-T

Lintasan buntu atau lintasan yang tidak sampai ke tujuan :

- M-A-B-C-E-F-G
- M-A-B-C-E-I-J
- M-D-C-E-F-G
- M-D-C-E-I-J
- M-D-I-J

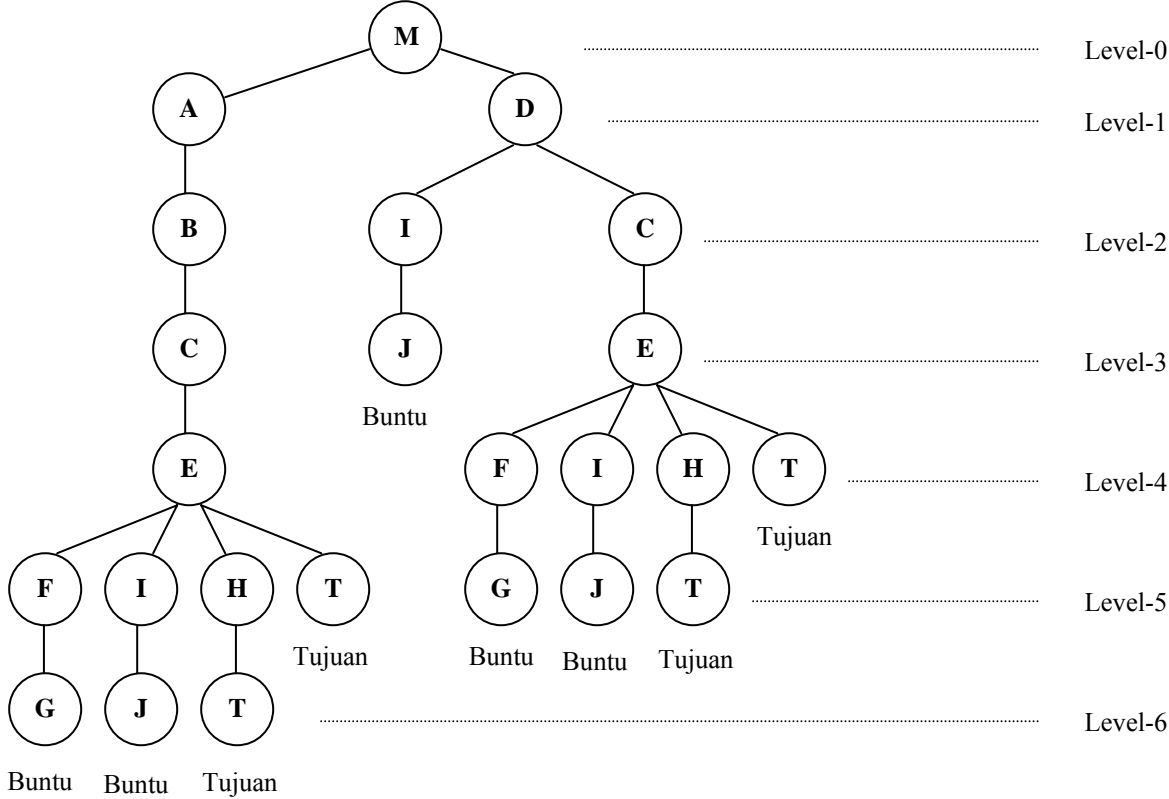
POHON PELACAKAN / PENCARIAN

Struktur pohon digunakan untuk menggambarkan keadaan secara hirarkis. Node yg terletak pada level-0 disebut 'akar'.

Node akar : menunjukkan keadaan awal & memiliki beberapa percabangan yang terdiri atas beberapa node yg disebut 'anak'.

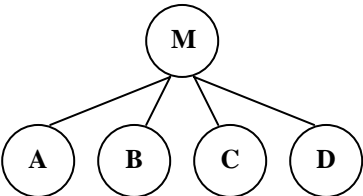
Node-node yg tidak memiliki anak disebut 'daun' menunjukkan akhir dari suatu pencarian, dapat berupa tujuan yang diharapkan (goal) atau jalan buntu (dead end).

Gambar berikut menunjukkan pohon pencarian untuk graph keadaan dengan 6 level.

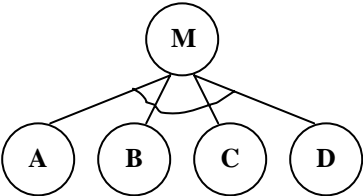


POHON AND/OR

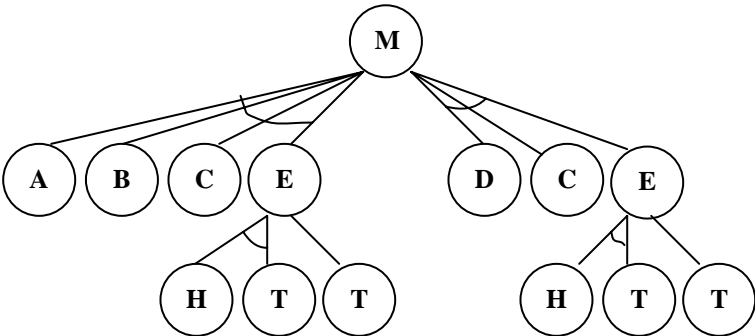
Masalah M dicari solusinya dengan 4 kemungkinan yaitu A OR B OR C OR D.



Masalah M hanya dapat diselesaikan dengan A AND B AND C AND D



Contoh : Dengan menggunakan pohon AND/OR tujuan yang dicapai pada pohon di Gambar sebelumnya bisa dipersingkat hanya sampai level-2 saja.



Contoh 1 : Masalah EMBER

Ada 2 ember masing-masing berkapasitas 4 galon (ember A) dan 3 galon (ember B). Ada pompa air yg akan digunakan untuk mengisi air pada ember tersebut. Bagaimana dapat mengisi tepat 2 galon air ke dalam ember berkapasitas 4 galon?

Penyelesaian :

- 1. Identifikasi ruang keadaan (state space)
Permasalahan ini dapat digambarkan sebagai himpunan pasangan bilangan bulat :
x = jumlah air yg diisikan ke ember 4 galon (ember A)
y = jumlah air yg diisikan ke ember 3 galon (ember B)
Ruang keadaan = (x,y) sedemikian hingga $x \in \{0,1,2,3,4\}$ dan $y \in \{0,1,2,3\}$
- 2. Keadaan awal & tujuan
Keadaan awal : kedua ember kosong = (0,0)
Tujuan : ember 4 galon berisi 2 galon air = (2,n) dengan sembarang n
- 3. Keadaan ember
Keadaan ember bisa digambarkan sebagai berikut :

Keadaan awal	Tujuan ∇				
<div><div></div><div></div><div></div><div></div></div> <div>(0,0)</div>	(1,0)	(2,0)	(3,0)	(4,0)	
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)	
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)	
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)	

4. Aturan-aturan
- Diasumsikan kita dapat mengisi ember air itu dari pompa air, membuang air dari ember ke luar, menuangkan air dari ember yang satu ke ember yang lain.
- Kita buat beberapa aturan-aturan yang dapat digambarkan sebagai berikut :

Aturan ke-	Jika	Maka
1	(x,y) $x < 4$	$(4,y)$ Isi ember A
2	(x,y) $y < 3$	$(x,3)$ Isi ember B
3	(x,y) $x > 0$	$(x - d,y)$ Tuang sebagian air keluar dari ember A
4	(x,y) $y > 0$	$(x,y - d)$ Tuang sebagian air keluar dari ember B
5	(x,y) $x > 0$	$(0,y)$ Kosongkan ember A dengan membuang airnya
6	(x,y) $y > 0$	$(x,0)$ Kosongkan ember B dengan membuang airnya
7	(x,y) $x+y \geq 4$ dan $y > 0$	$(4,y - (4 - x))$ Tuang air dari ember B ke ember A sampai ember A penuh
8	(x,y) $x+y \geq 3$ dan $x > 0$	$(x - (3 - y),3)$ Tuang air dari ember A ke ember B sampai ember B penuh
9	(x,y) $x+y \leq 4$ dan $y > 0$	$(x+y,0)$ Tuang seluruh air dari ember B ke ember A
10	(x,y) $x+y \leq 3$ dan $x > 0$	$(0,x+y)$ Tuang seluruh air dari ember A ke ember B
11	$(0,2)$	$(2,0)$ Tuang 2 galon air dari ember B ke ember A

5. Representasi ruang keadaan dengan pohon pelacakan
- Pencarian suatu solusi dapat dilukiskan dengan menggunakan pohon. Tiap-tiap node menunjukkan satu keadaan. Jalur dari parent ke child ,menunjukkan 1 operasi. Tiap node memiliki node child yg menunjukkan keadaan yg dapat dicapai oleh parent.

Solusi yg ditemukan :

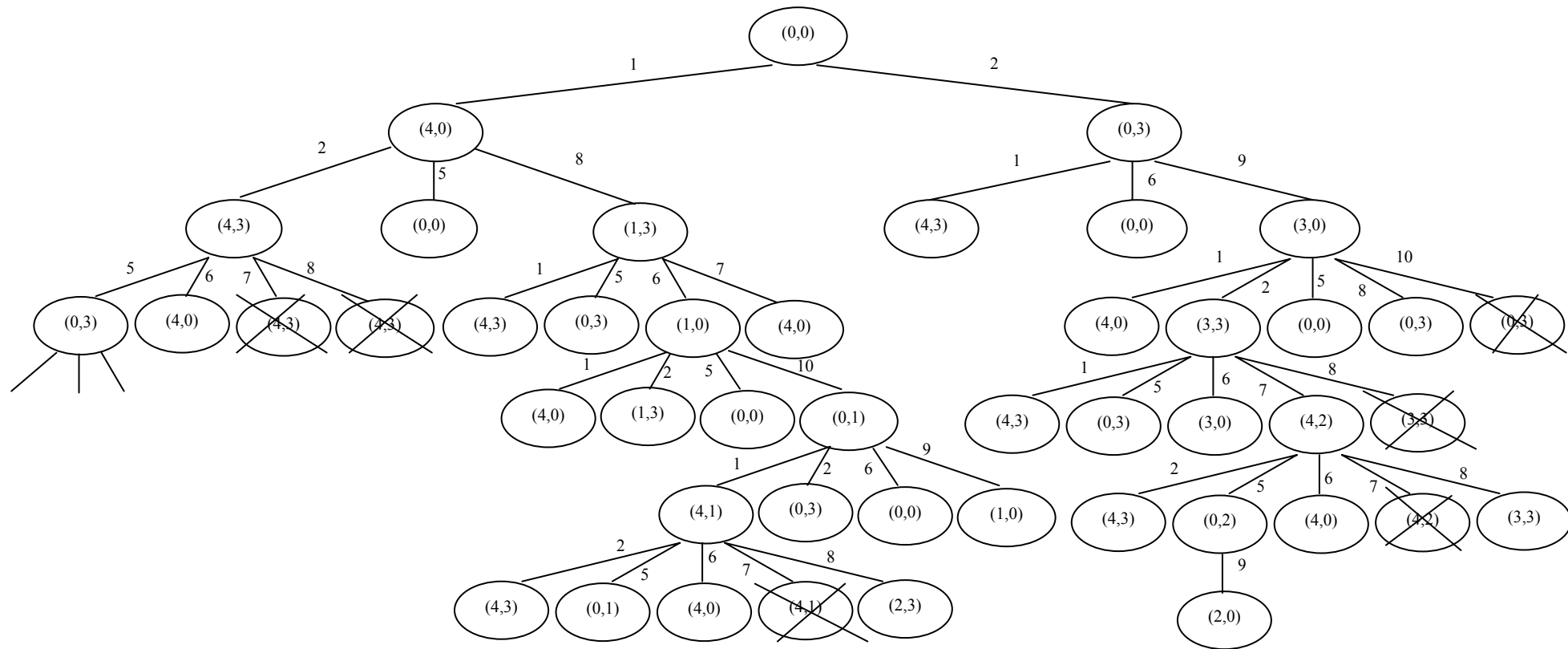
Solusi 1

Isi ember A	Isi ember B	Aturan yg dipakai
0	0	1
4	0	8
1	3	6
1	0	10
0	1	1
4	1	8
2	3	Solusi

Solusi 2

Isi ember A	Isi ember B	Aturan yg dipakai
0	0	2
0	3	9
3	0	2
3	3	7
4	2	5
0	2	9
2	0	Solusi

Representasi ruang keadaan untuk kasus EMBER



Contoh 2 : Masalah PETANI,KAMBING,SERIGALA,SAYURAN,PERAHU

Seorang petani akan menyeberangkan seekor kambing,seekor serigala,sayuran dengan sebuah perahu yg melalui sungai. Perahu hanya bisa memuat petani & satu penumpang yg lain (kambing, serigala, atau sayuran). Jika ditinggalkan petani tersebut, maka sayuran dimakan kambing dan kambing akan dimakan serigala.

Penyelesaian :

1. Identifikasi ruang keadaan
- Permasalahan ini dapat dilambangkan dengan (jumlah kambing,jumlah serigala,jumlah sayuran,jumlah perahu).
- Contoh : daerah asal (0,1,1,1) = daerah asal tidak ada kambing,ada serigala, ada sayuran,ada perahu
2. Keadaan awal & tujuan
- Keadaan awal, pada kedua daerah :
- daerah asal = (1,1,1,1)
- daerah seberang = (0,0,0,0)
- Keadaan tujuan, pada kedua daerah :
- daerah asal = (0,0,0,0)
- daerah seberang = (1,1,1,1)

3. Aturan-aturan

Aturan ke-	Aturan
1	Kambing menyeberang
2	Sayuran menyeberang
3	Serigala menyeberang
4	Kambing kembali
5	Sayuran kembali
6	Serigala kembali
7	Perahu kembali

4. Solusi yg ditemukan

Daerah asal	Daerah seberang	Aturan yg dipakai
(1,1,1,1)	(0,0,0,0)	1
(0,1,1,0)	(1,0,0,1)	7
(0,1,1,1)	(1,0,0,0)	3
(0,0,1,0)	(1,1,0,1)	4
(1,0,1,1)	(0,1,0,0)	2
(1,0,0,0)	(0,1,1,1)	7
(1,0,0,1)	(0,1,1,0)	1
(0,0,0,0)	(1,1,1,1)	Solusi

METODE PELACAKAN/PENCARIAN

Hal penting dalam menentukan keberhasilan sistem cerdas adalah kesuksesan dalam pencarian.

Pencarian = suatu proses mencari solusi dari suatu permasalahan melalui sekumpulan kemungkinan ruang keadaan (state space). Ruang keadaan = merupakan suatu ruang yang berisi semua keadaan yang mungkin.

Untuk mengukur performansi metode pencarian, terdapat empat kriteria yang dapat digunakan :

- Completeness : apakah metode tersebut **menjamin penemuan solusi** jika solusinya memang ada?
- Time complexity : berapa lama **waktu** yang diperlukan?
- Space complexity : berapa banyak **memori** yang diperlukan
- Optimality : apakah metode tersebut menjamin menemukan **solusi yang terbaik** jika terdapat beberapa solusi berbeda?

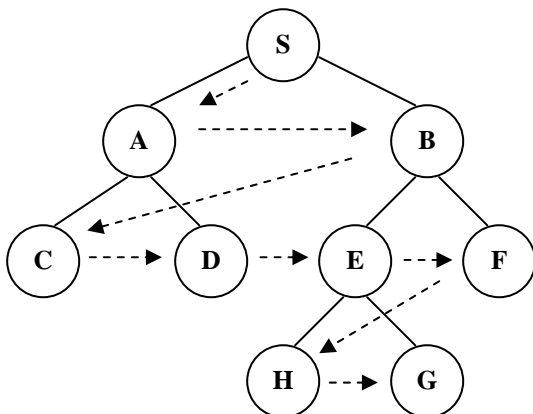
Teknik pencarian :

- A. Pencarian buta (blind search) : tidak ada informasi awal yang digunakan dalam proses pencarian
 1. Pencarian melebar pertama (Breadth – First Search)
 2. Pencarian mendalam pertama (Depth – First Search)
- B. Pencarian terbimbing (heuristic search) : adanya informasi awal yang digunakan dalam proses pencarian
 1. Pendakian Bukit (Hill Climbing)
 2. Pencarian Terbaik Pertama (Best First Search)

A. Pencarian Buta (blind search)

1. Breadth – First Search

Semua node pada level n akan dikunjungi terlebih dahulu sebelum mengunjungi node-node pada level n+1. Pencarian dimulai dari node akar terus ke level 1 dari kiri ke kanan, kemudian berpindah ke level berikutnya dari kiri ke kanan hingga solusi ditemukan.



Keuntungan :

- tidak akan menemui jalan buntu, menjamin ditemukannya solusi (jika solusinya memang ada) dan solusi yang ditemukan pasti yang paling baik
- jika ada 1 solusi, maka breadth – first search akan ditemukannya, jika ada lebih dari 1 solusi, maka solusi minimum akan ditemukan.
- Kesimpulan : **complete** dan **optimal**

Kelemahan :

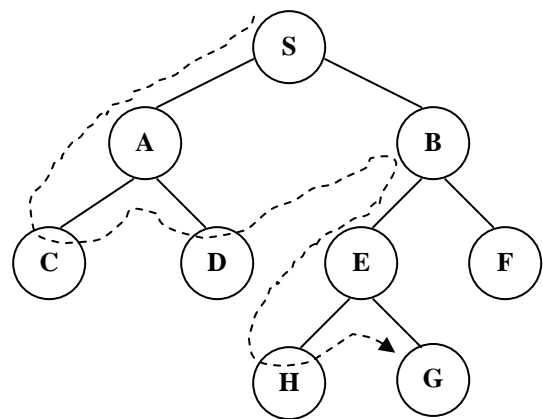
- membutuhkan **memori yang banyak**, karena harus menyimpan semua simpul yang pernah dibangkitkan. Hal ini harus dilakukan agar BFS dapat melakukan penelusuran simpul-simpul sampai di level bawah
- membutuhkan **waktu yang cukup lama**

2. Depth – First Search

Pencarian dilakukan pada suatu simpul dalam setiap level dari yang paling kiri.

Jika pada level yang paling dalam tidak ditemukan solusi, maka pencarian dilanjutkan pada simpul sebelah kanan dan simpul yang kiri dapat dihapus dari memori.

Jika pada level yang paling dalam tidak ditemukan solusi, maka pencarian dilanjutkan pada level sebelumnya. Demikian seterusnya sampai ditemukan solusi.



- Keuntungan :
- membutuhkan **memori relatif kecil**, karena hanya node-node pada lintasan yang aktif saja yang disimpan
 - Secara kebetulan, akan menemukan solusi tanpa harus menguji lebih banyak lagi dalam ruang keadaan, jadi jika solusi yang dicari berada pada level yang dalam dan paling kiri, maka DFS akan menemukannya dengan cepat → **waktu cepat**
- Kelemahan :
- Memungkinkan tidak ditemukannya tujuan yang diharapkan, karena jika pohon yang dibangkitkan mempunyai level yang sangat dalam (tak terhingga) → **tidak complete** karena tidak ada jaminan menemukan solusi
 - Hanya mendapat 1 solusi pada setiap pencarian, karena jika terdapat lebih dari satu solusi yang sama tetapi berada pada level yang berbeda, maka DFS tidak menjamin untuk menemukan solusi yang paling baik → **tidak optimal**.

B. Heuristic Search

Pencarian buta tidak selalu dapat diterapkan dengan baik, hal ini disebabkan waktu aksesnya yang cukup lama & besarnya memori yang diperlukan. Untuk masalah dengan ruang masalah yang besar, teknik pencarian buta bukan metode yang baik karena keterbatasan kecepatan komputer dan memori.

Metode heuristic search diharapkan bisa menyelesaikan permasalahan yang lebih besar.

Metode heuristic search menggunakan suatu fungsi yang menghitung biaya perkiraan (estimasi) dari suatu simpul tertentu menuju ke simpul tujuan → disebut **fungsi heuristic**

Aplikasi yang menggunakan fungsi heuristic : Google, Deep Blue Chess Machine

Misal kasus 8-puzzle. Ada 4 operator yang dapat digunakan untuk menggerakkan dari satu keadaan ke keadaan yang baru

- 1. Ubin kosong digeser ke kiri
- 2. Ubin kosong digeser ke kanan
- 3. Ubin kosong digeser ke bawah
- 4. Ubin kosong digeser ke atas

Langkah awal

Keadaan awal

Tujuan

1	2	3
7	8	4
6		5

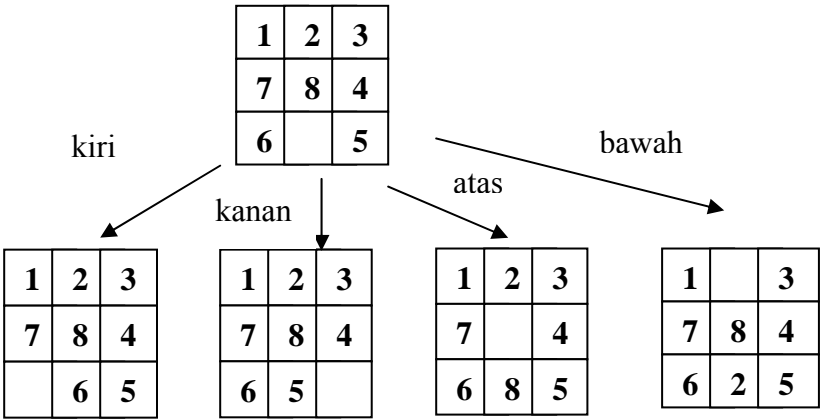


1	2	3
8		4
7	6	5

Tujuan

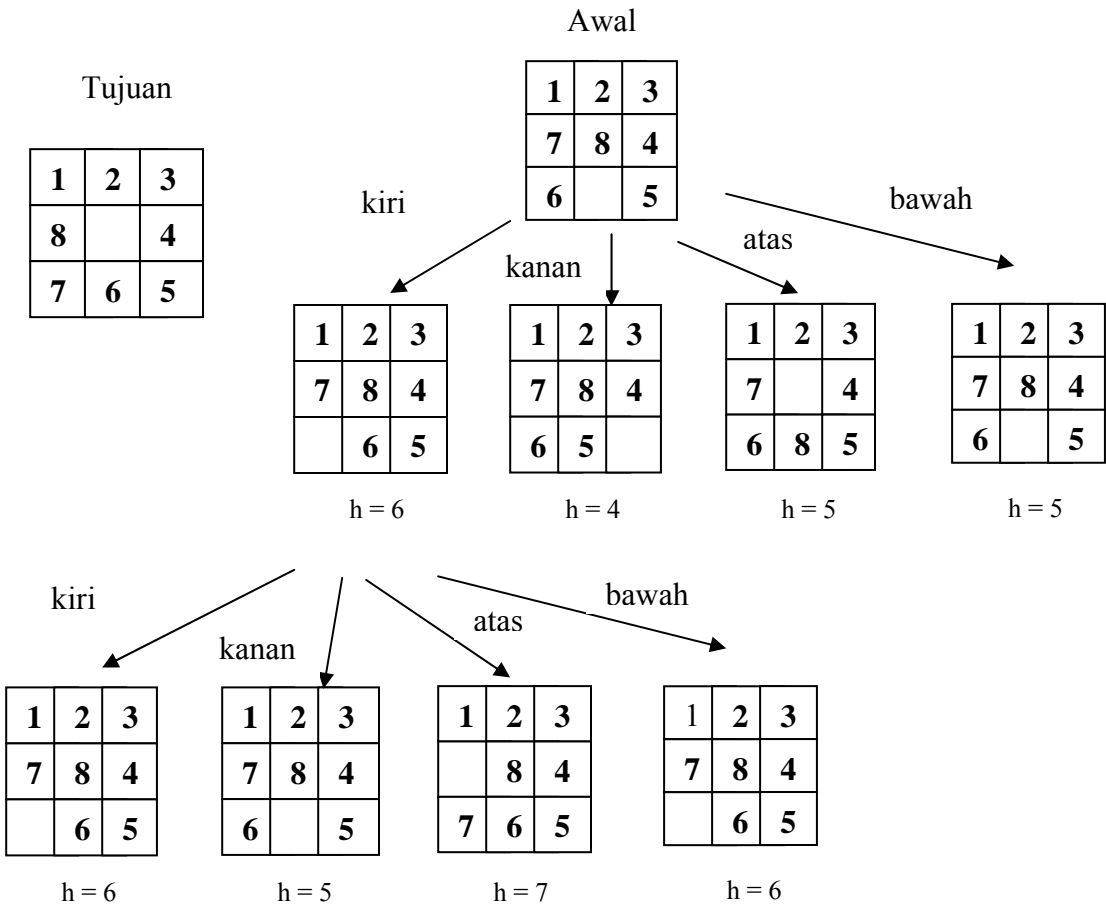
1	2	3
8		4
7	6	5

Awal

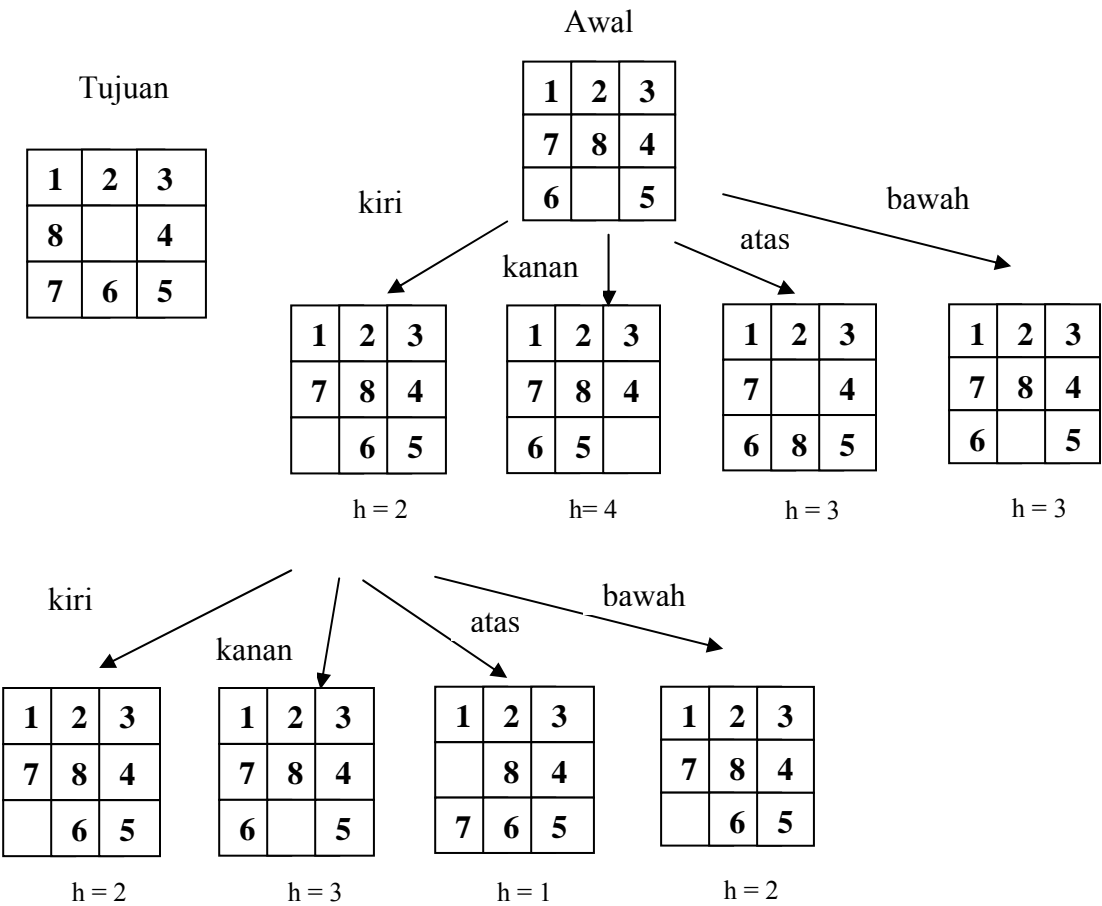


Pada pencarian heuristik perlu diberikan informasi khusus, yaitu :

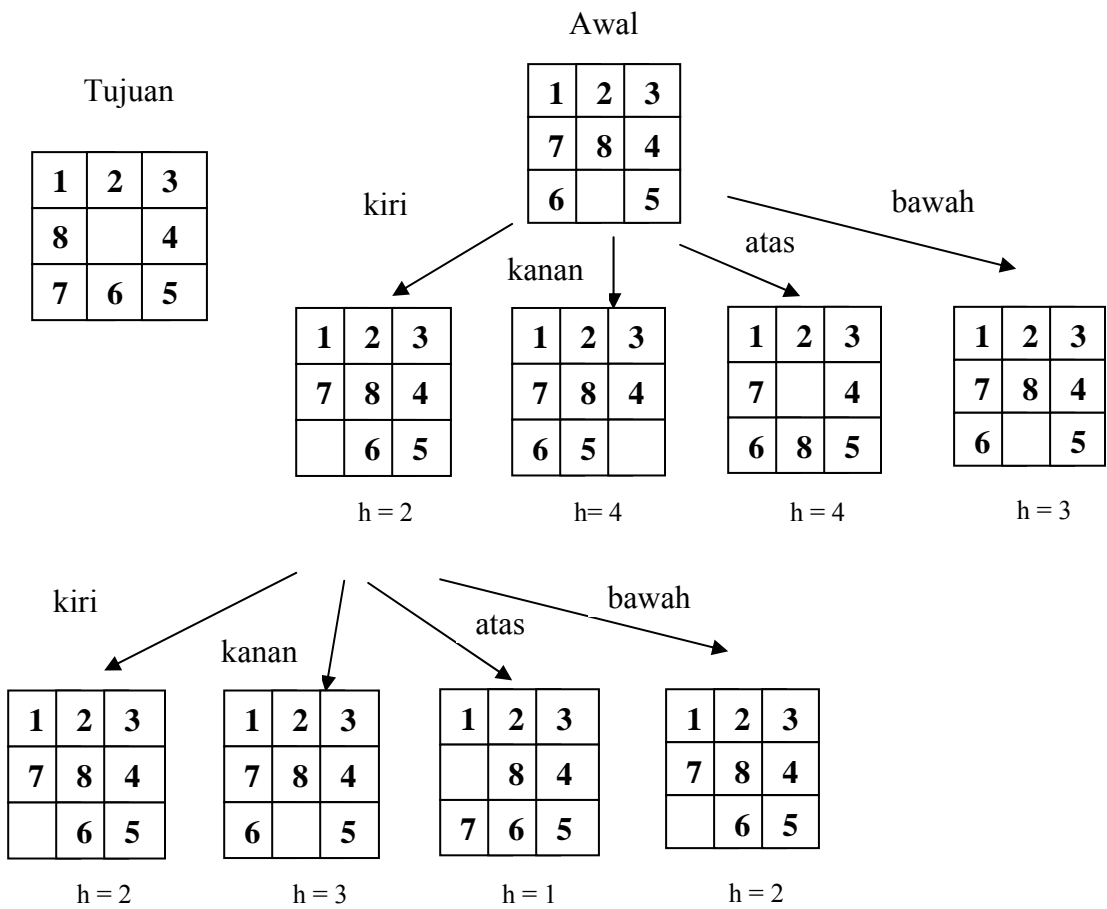
- ♦ Untuk jumlah ubin yang menempati posisi yang benar
Jumlah yang lebih tinggi adalah yang lebih diharapkan (lebih baik)



- ♦ Untuk jumlah ubin yang menempati posisi yang salah
Jumlah yang lebih kecil adalah yang diharapkan (lebih baik)



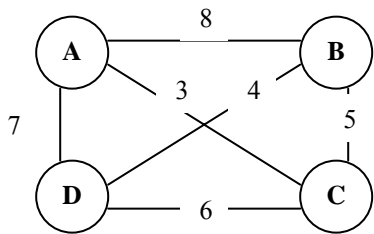
- ◆ Menghitung total gerakan yang diperlukan untuk mencapai tujuan
Jumlah yang lebih kecil adalah yang diharapkan (lebih baik)



1. Hill Climbing

Contoh : Traveling Salesman Problem (TSP)

Seorang salesman ingin mengunjungi n kota. Jarak antara tiap-tiap kota sudah diketahui. Kita ingin mengetahui rute terpendek dimana setiap kota hanya boleh dikunjungi tepat 1 kali. Misal ada 4 kota dengan jarak antara tiap-tiap kota seperti berikut ini :



Solusi – solusi yang mungkin dengan menyusun kota-kota dalam urutan abjad, misal :

A – B – C – D : dengan panjang lintasan (=19)

A – B – D – C : (=18)

A – C – B – D : (=12)

A – C – D – B : (=13)

dst

a. Metode simple hill climbing

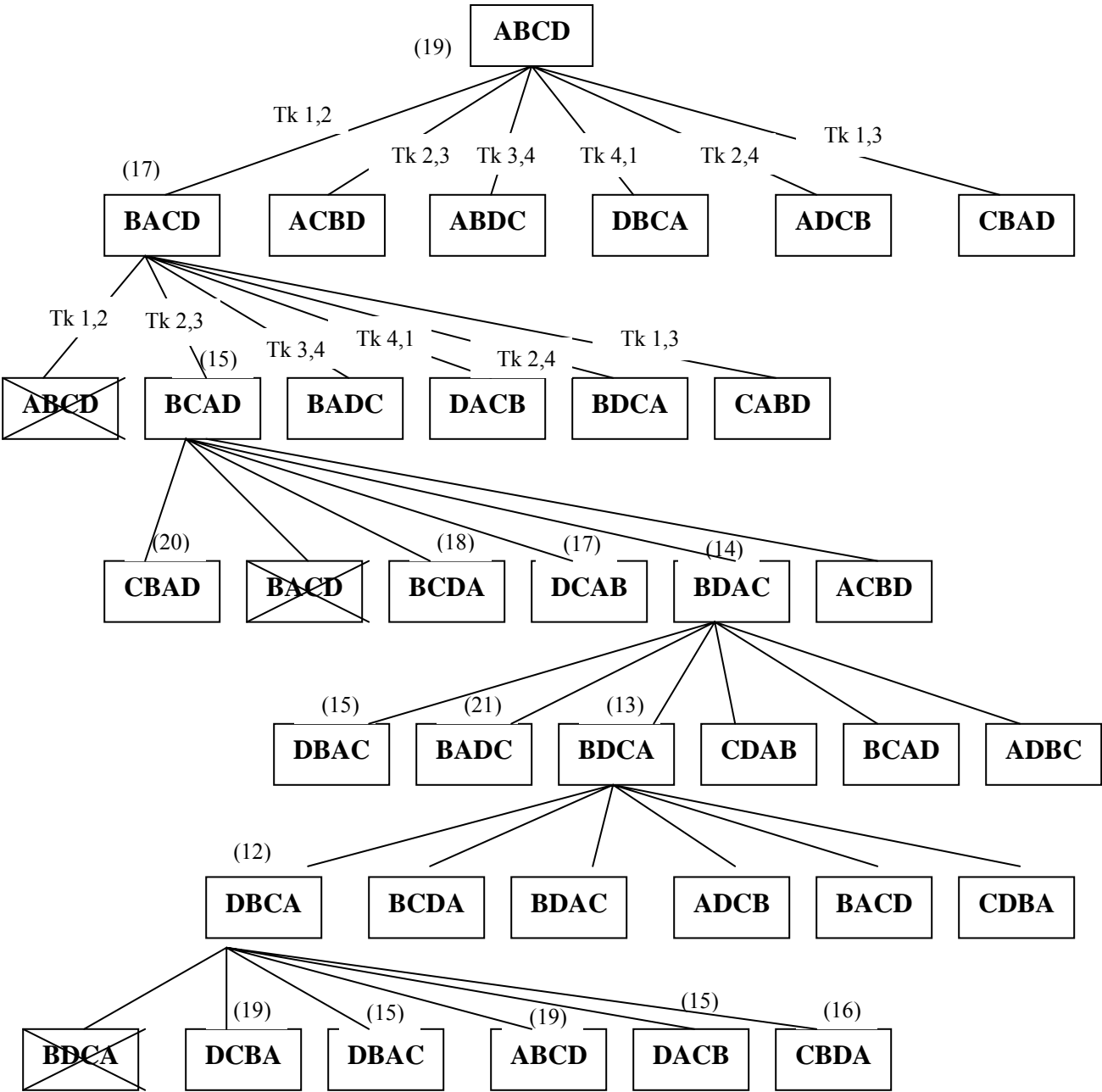
Ruang keadaan berisi semua kemungkinan lintasan yang mungkin. Operator digunakan untuk menukar posisi kota-kota yang bersebelahan. Fungsi heuristik yang digunakan adalah panjang lintasan yang terjadi.

Operator yang akan digunakan adalah menukar urutan posisi 2 kota dalam 1 lintasan. Bila ada n kota, dan ingin mencari kombinasi lintasan dengan menukar posisi urutan 2 kota, maka akan didapat sebanyak :

$$\frac{n!}{2!(n-2)!} = \frac{4!}{2!(4-2)!} = 6 \text{ kombinasi}$$

Keenam kompbikasi ini akan dipakai semuanya sebagai operator, yaitu :

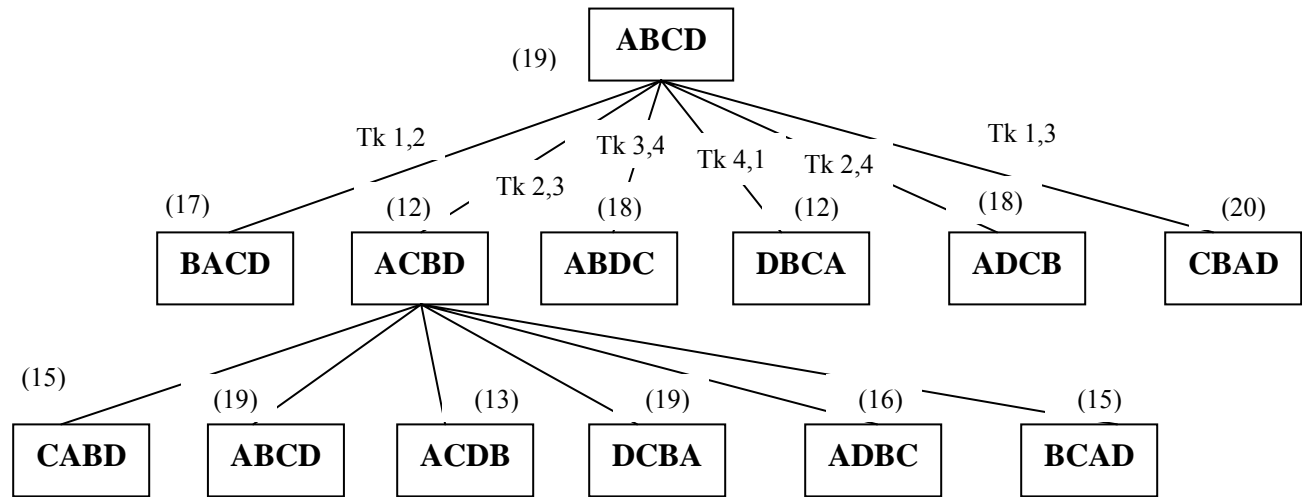
- Tukar 1,2 = menukar urutan posisi kota ke – 1 dengan kota ke – 2
- Tukar 2,3 = menukar urutan posisi kota ke – 2 dengan kota ke – 3
- Tukar 3,4 = menukar urutan posisi kota ke – 3 dengan kota ke – 4
- Tukar 4,1 = menukar urutan posisi kota ke – 4 dengan kota ke – 1
- Tukar 2,4 = menukar urutan posisi kota ke – 2 dengan kota ke – 4
- Tukar 1,3 = menukar urutan posisi kota ke – 1 dengan kota ke – 3



Keadaan awal, lintasan ABCD (=19).
Level pertama, hill climbing mengunjungi BACD (=17), BACD (=17) < ABCD (=19), sehingga BACD menjadi pilihan selanjutnya dengan operator Tukar 1,2
Level kedua, mengunjungi ABCD, karena operator Tukar 1,2 sudah dipakai BACD, maka pilih node lain yaitu BCAD (=15), BCAD (=15) < BACD (=17)
Level ketiga, mengunjungi CBAD (=20), CBAD (=20) > BCAD (=15), maka pilih node lain yaitu BCDA (=18), pilih node lain yaitu DCAB (=17), pilih node lain yaitu BDAC (=14), BDAC (=14) < BCAD (=15)
Level keempat, mengunjungi DBAC (=15), DBAC(=15) > BDAC (=14), maka pilih node lain yaitu BADC (=21), pilih node lain yaitu BDCA (=13), BDCA (=13) < BDAC (=14)
Level kelima, mengunjungi DBCA (=12), DBCA (=12) < BDCA (=13)
Level keenam, mengunjungi BDCA, karena operator Tukar 1,2 sudah dipakai DBCA, maka pilih node lain yaitu DCBA, pilih DBAC, pilih ABCD, pilih DACB, pilih CBDA
Karena sudah tidak ada node yang memiliki nilai heuristik yang lebih kecil dibanding nilai heuristik DBCA, maka **node DBCA (=12) adalah lintasan terpendek (SOLUSI)**

b. Metode steepest – ascent hill climbing

Steepest – ascent hill climbing hampir sama dengan simple – ascent hill climbing, hanya saja gerakan pencarian tidak dimulai dari kiri, tetapi berdasarkan nilai heuristik terbaik.



Keadaan awal, lintasan ABCD (=19).

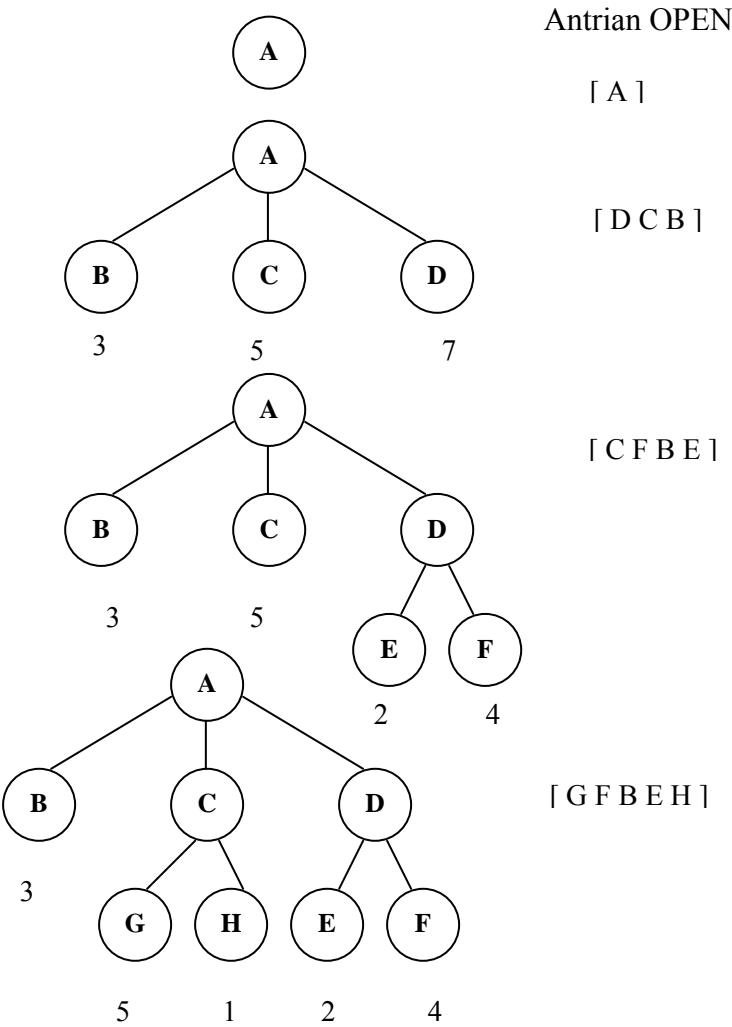
Level pertama, hill climbing memilih nilai heuristik terbaik yaitu ACBD (=12) sehingga ACBD menjadi pilihan selanjutnya.

Level kedua, hill climbing memilih nilai heuristik terbaik, karena nilai heuristik lebih besar dibanding ACBD, maka hasil yang diperoleh lintasannya tetap ACBD (=12)

2. Best First Search

Metode best first search merupakan kombinasi dari metode depth first search dan breadth first search dengan mengambil kelebihan dari kedua metode tersebut. Hill climbing tidak diperbolehkan untuk kembali ke node pada lebih rendah meskipun node tersebut memiliki nilai heuristik lebih baik. Pada best first search, pencarian diperbolehkan mengunjungi node di lebih rendah, jika ternyata node di level lebih tinggi memiliki nilai heuristik lebih buruk. Untuk mengimplementasikan metode ini, dibutuhkan 2 antrian yang berisi node-node, yaitu :

OPEN : berisi node-node yang sudah dibangkitkan, sudah memiliki fungsi heuristik namun belum diuji. Umumnya berupa antrian berprioritas yang berisi elemen-elemen dengan nilai heuristik tertinggi.
CLOSED : berisi node-node yang sudah diuji.



Diasumsikan node dengan nilai yang lebih besar memiliki nilai evaluasi yang lebih baik. Pada keadaan awal, antrian berisi A. Pengujian dilakukan di level pertama, node D memiliki nilai terbaik, sehingga menempati antrian pertama, disusul dengan C dan B. Node D memiliki cabang E dan F yang masing-masing bernilai 2 & 4. Dengan demikian C merupakan pilihan terbaik dengan menempati antrian pertama. Demikian seterusnya.

III. REPRESENTASI PENGETAHUAN

- Dua bagian dasar sistem kecerdasan buatan (menurut Turban) :
 - Basis pengetahuan :
Berisi fakta tentang objek-objek dalam domain yang dipilih dan hubungan diantara domain-domain tersebut
 - Inference Engine :
Merupakan sekumpulan prosedur yang digunakan untuk menguji basis pengetahuan dalam menjawab suatu pertanyaan, menyelesaikan masalah, atau membuat keputusan
- Basis pengetahuan berisi struktur data yang dapat dimanipulasi oleh suatu sistem inferensi yang menggunakan pencarian dan teknik pencocokan pola pada basis pengetahuan yang bermanfaat untuk menjawab pertanyaan, menggambarkan kesimpulan atau bentuk lainnya sebagai suatu fungsi kecerdasan
- Karakteristik representasi pengetahuan
 1. Dapat diprogram dengan bahasa komputer dan disimpan dalam memori
 2. Fakta dan pengetahuan lain yang terkandung didalamnya dapat digunakan untuk melakukan penalaran
- Dalam menyelesaikan masalah harus dibutuhkan pengetahuan yang cukup dan sistem juga harus memiliki kemampuan untuk menalar. Basis pengetahuan dan kemampuan untuk melakukan penalaran merupakan bagian terpenting dari sistem yang menggunakan kecerdasan buatan.

3.1 LOGIKA

Logika adalah bentuk representasi pengetahuan yang paling tua. Proses logika adalah proses membentuk kesimpulan atau menarik suatu inferensi berdasarkan fakta yang telah ada. Input dari proses logika berupa premis atau fakta-fakta yang diakui kebenarannya sehingga dengan melakukan penalaran pada proses logika dapat dibentuk suatu inferensi atau kesimpulan yang benar juga.



Ada 2 penalaran yang dapat dilakukan untuk mendapat konklusi :

1. Penalaran deduktif : dimulai dari prinsip umum untuk mendapatkan konklusi yang lebih khusus.
Contoh :
Premis mayor : Jika hujan turun saya tidak akan berangkat kuliah
Premis minor : Hari ini hujan turun
Konklusi : Hari ini saya tidak akan berangkat kuliah
2. Penalaran induktif : dimulai dari fakta-fakta khusus untuk mendapatkan kesimpulan umum.
Contoh :
Premis -1 : Aljabar adalah pelajaran yang sulit
Premis -2 : Geometri adalah pelajaran yang sulit
Premis -3 : Kalkulus adalah pelajaran yang sulit
Konklusi : Matematika adalah pelajaran yang sulit

Munculnya premis baru bisa mengakibatkan gugurnya konklusi yang sudah diperoleh, misal :
Premis -4 : Kinematika adalah pelajaran yang sulit
Premis tersebut menyebabkan konklusi : “Matematika adalah pelajaran yang sulit”, menjadi salah, karena Kinematika bukan merupakan bagian dari Matematika, sehingga bila menggunakan penalaran induktif sangat dimungkinkan adanya ketidakpastian.

3.1.1 Logika Proposisi

Proposisi adalah suatu pernyataan yang dapat bernilai Benar atau Salah. Simbol-simbol seperti P dan Q menunjukkan proposisi. Dua atau lebih proposisi dapat digabungkan dengan menggunakan operator logika :

- a. Konjungsi : \wedge (and)
- b. Disjungsi : \vee (or)
- c. Negasi : \neg (not)
- d. Implikasi : \rightarrow (if then)
- e. Ekuivalensi : \leftrightarrow (if and only if)

Not

P	not P
B	S
S	B

And, Or, If – Then, If – and – only – if

P	Q	P and Q	P or Q	if P then Q	P if and only if Q
B	B	B	B	B	B
B	S	S	B	S	S
S	B	S	B	B	S
S	S	S	S	B	B

Untuk melakukan inferensi pada logika proposisi dapat dilakukan dengan menggunakan resolusi. Resolusi adalah suatu aturan untuk melakukan inferensi yang dapat berjalan secara efisien dalam suatu bentuk khusus yaitu *conjunctive normal form* (CNF), ciri – cirinya :

- setiap kalimat merupakan disjungsi literal
- semua kalimat terkonjungsi secara implisit

Langkah-langkah untuk mengubah suatu kalimat (konversi) ke bentuk CNF :

- Hilangkan implikasi dan ekuivalensi
 - $x \rightarrow y$ menjadi $\neg x \vee y$
 - $x \leftrightarrow y$ menjadi $(\neg x \vee y) \wedge (\neg y \vee x)$
- Kurangi lingkup semua negasi menjadi satu negasi saja
 - $\neg(\neg x)$ menjadi x
 - $\neg(x \vee y)$ menjadi $(\neg x \wedge \neg y)$
 - $\neg(x \wedge y)$ menjadi $(\neg x \vee \neg y)$
- Gunakan aturan asosiatif dan distributif untuk mengkonversi menjadi conjunction of disjunction
 - Asosiatif : $(A \vee B) \vee C$ menjadi $A \vee (B \vee C)$
 - Distributif : $(A \wedge B) \vee C$ menjadi $(A \vee C) \wedge (B \vee C)$
- Buat satu kalimat terpisah untuk tiap-tiap konjungsi

Contoh :

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

1. P
2. $(P \wedge Q) \rightarrow R$
3. $(S \vee T) \rightarrow Q$
4. T

Tentukan kebenaran R.

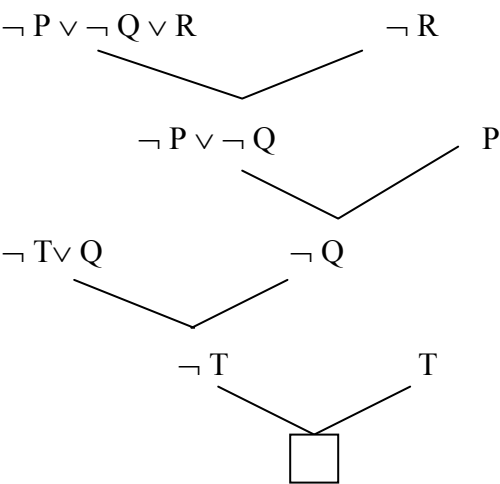
Untuk membuktikan kebenaran R dengan menggunakan resolusi,maka ubah dulu menjadi bentuk CNF.

Kalimat	Langkah-langkah	CNF
1. P	Sudah merupakan bentuk CNF	P
2. $(P \wedge Q) \rightarrow R$	<ul style="list-style-type: none">■ Menghilangkan implikasi : $\neg (P \wedge Q) \vee R$■ Mengurangi lingkup negasi : $(\neg P \vee \neg Q) \vee R$■ Gunakan asosiatif : $\neg P \vee \neg Q \vee R$	$\neg P \vee \neg Q \vee R$
3. $(S \vee T) \rightarrow Q$	<ul style="list-style-type: none">■ Menghilangkan implikasi : $\neg (S \vee T) \vee Q$■ Mengurangi lingkup negasi : $(\neg S \wedge \neg T) \vee Q$■ Gunakan distributif : $(\neg S \vee Q) \wedge (\neg T \vee Q)$	$(\neg S \vee Q)$ $(\neg T \vee Q)$
4. T	Sudah merupakan bentuk CNF	T

Kemudian kita tambahkan kontradiksi pada tujuannya, R menjadi $\neg R$ sehingga fakta-fakta (dalam bentuk CNF) dapat disusun menjadi :

- 1. P
- 2. $\neg P \vee \neg Q \vee R$
- 3. $\neg S \vee Q$
- 4. $\neg T \vee Q$
- 5. T
- 6. $\neg R$

Sehingga resolusi dapat dilakukan untuk membuktikan kebenaran R, sebagai berikut :



Contoh bila diterapkan dalam kalimat :

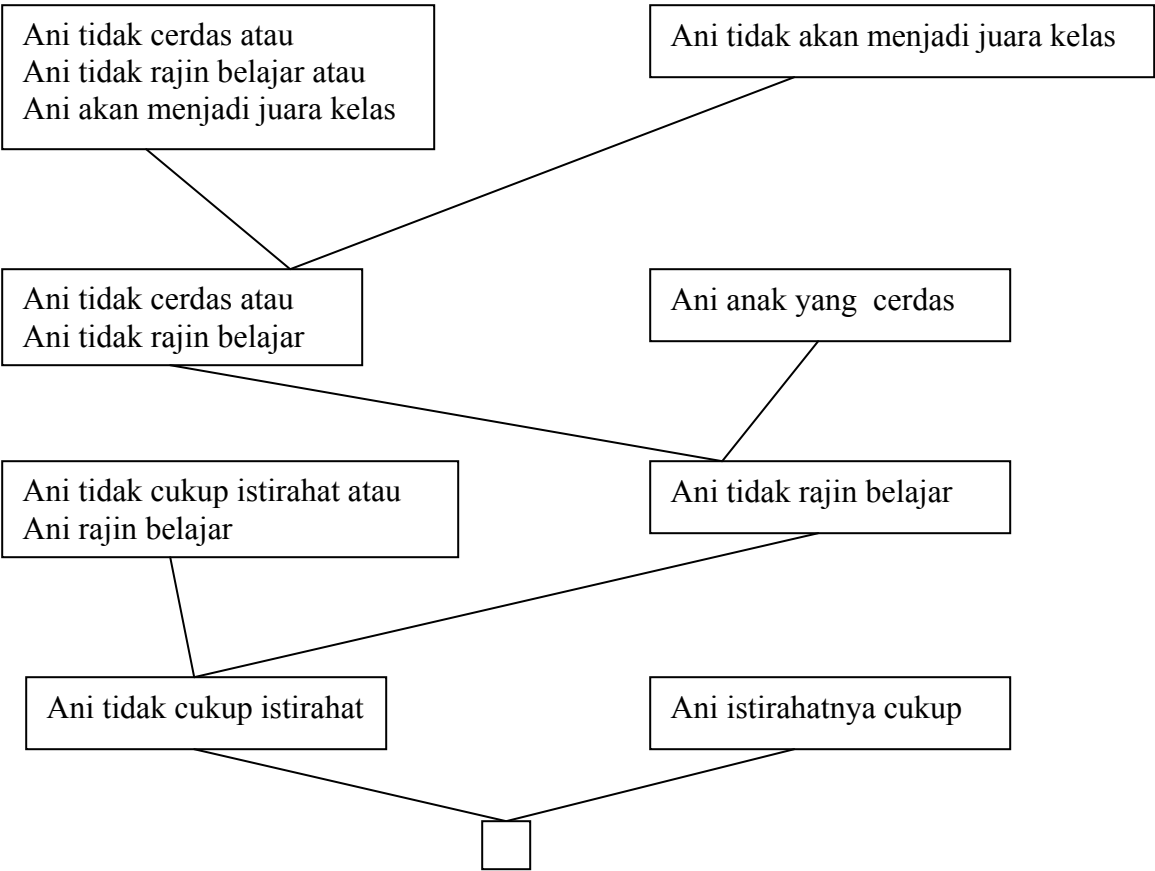
- P : Ani anak yang cerdas
- Q : Ani rajin belajar
- R : Ani akan menjadi juara kelas
- S : Ani makannya banyak
- T : Ani istirahatnya cukup

Kalimat yang terbentuk :

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

Setelah dilakukan konversi ke bentuk CNF, didapat :

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



3.1.2 Logika Predikat

Representasi Fakta Sederhana

Misal diketahui fakta-fakta sebagai berikut :

- Andi adalah seorang laki-laki : A
- Ali adalah seorang laki-laki : B
- Amir adalah seorang laki-laki : C
- Anto adalah seorang laki-laki : D
- Agus adalah seorang laki-laki : E

Jika kelima fakta tersebut dinyatakan dengan menggunakan proposisi, maka akan terjadi pemborosan, dimana beberapa pernyataan dengan predikat yang sama akan dibuat dalam proposisi yang berbeda.

Logika predikat digunakan untuk merepresentasikan hal-hal yang tidak dapat direpresentasikan dengan menggunakan logika proposisi. Pada logika predikat kita dapat merepresentasikan fakta-fakta sebagai suatu pernyataan yang disebut dengan *wff (well – formed formula)*. Logika predikat merupakan dasar bagi bahasa AI seperti bahasa pemrograman PROLOG

Pada contoh diatas, dapat dituliskan :

laki-laki(x)

dimana x adalah variabel yang disubstitusikan dengan Andi, Ali, Amir, Anto, Agus, dan laki-laki yang lain.

Dalam logika predikat, suatu proposisi atau premis dibagi menjadi 2 bagian, yaitu argumen (objek) dan predikat (keterangan). Argumen adalah individu atau objek yang membuat keterangan. Predikat adalah keterangan yang membuat argumen dan predikat.

Contoh :

1. Jika besok tidak hujan, Tommy pergi ke gunung
 $\neg \text{cuaca}(\text{hujan}, \text{besok}) \rightarrow \text{pergi}(\text{tommy}, \text{gunung})$
2. Diana adalah nenek dari ibu Amir
 $\text{nenek}(\text{Diana}, \text{ibu}(\text{Amir}))$
3. Mahasiswa berada di dalam kelas
 $\text{didalam}(\text{mahasiswa}, \text{kelas})$

Dari contoh diatas dapat dijabarkan sebagai berikut :

di dalam = predikat (keterangan)

mahasiswa = argumen (objek)

kelas = argumen (objek)

4. Johan suka Maria
 $\text{suka}(\text{johan}, \text{maria})$
5. Pintu terbuka
 $\text{Buka}(\text{pintu})$
6. Johan suka Maria
 Ramon suka Maria
 Misal : Johan = x, Maria = y, Ramon = z
 Maka : $\text{suka}(x, y) \wedge \text{suka}(z, y) \rightarrow \text{tidak suka}(x, z)$
 Dibaca : Jika Johan suka Maria dan Ramon suka Maria, maka Johan tidak suka Ramon

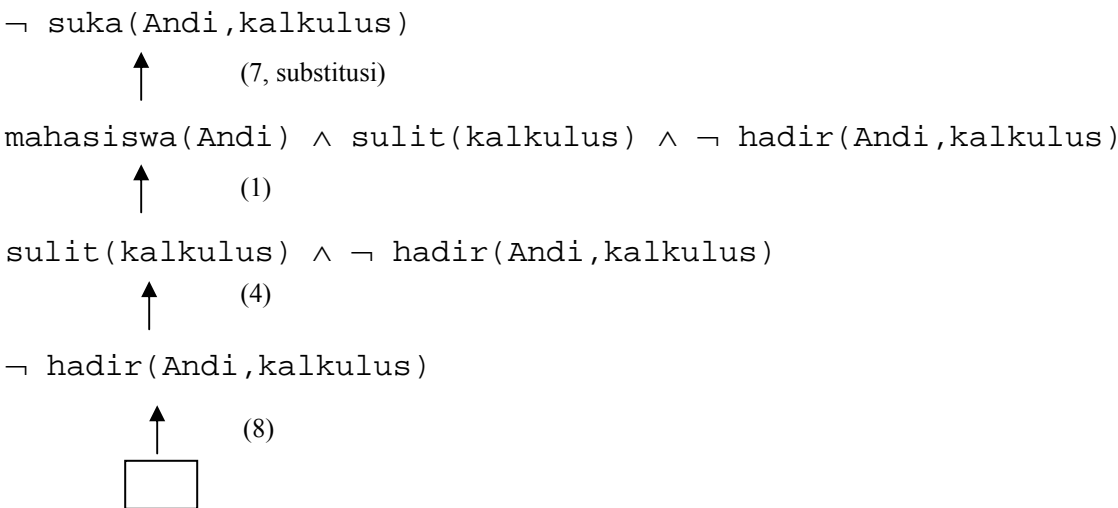
Misal terdapat pernyataan sebagai berikut :

1. Andi adalah seorang mahasiswa
2. Andi masuk jurusan Elektro
3. Setiap mahasiswa elektro pasti mahasiswa teknik
4. Kalkulus adalah matakuliah yang sulit
5. Setiap mahasiswa teknik pasti akan suka kalkulus atau akan membencinya
6. Setiap mahasiswa pasti akan suka terhadap suatu matakuliah
7. Mahasiswa yang tidak pernah hadir pada kuliah matakuliah sulit, maka mereka pasti tidak suka terhadap matakuliah tersebut.
8. Andi tidak pernah hadir kuliah matakuliah kalkulus

Kedelapan pernyataan diatas dapat dibawa ke bentuk logika predikat dengan menggunakan operator-operator : \rightarrow , \neg , \wedge , \vee , \forall (untuk setiap), \exists (terdapat), sebagai berikut :

- 1. mahasiswa(Andi)
- 2. elektro(Andi)
- 3. $\forall x : \text{elektro}(x) \rightarrow \text{teknik}(x)$
- 4. sulit(kalkulus)
- 5. $\forall x : \text{teknik}(x) \rightarrow \text{suka}(x,\text{kalkulus}) \vee \text{benci}(x,\text{kalkulus})$
- 6. $\forall x : \exists y : \text{suka}(x,y)$
- 7. $\forall x : \forall y : \text{mahasiswa}(x) \wedge \text{sulit}(y) \wedge \neg \text{hadir}(x,y) \rightarrow \neg \text{suka}(x,y)$
- 8. $\neg \text{hadir}(\text{Andi},\text{kalkulus})$

Andaikan kita akan menjawab pertanyaan :
"Apakah Andi suka matakuliah kalkulus?"
Maka dari pernyataan ke-7 kita akan membuktikan bahwa Andi tidak suka dengan matakuliah kalkulus.
Dengan menggunakan penalaran backward, bisa dibuktikan bahwa :
 $\neg \text{suka}(\text{Andi},\text{kalkulus})$
Sebagai berikut :



Dari penalaran tersebut dapat dibuktikan bahwa Andi tidak suka dengan matakuliah kalkulus.

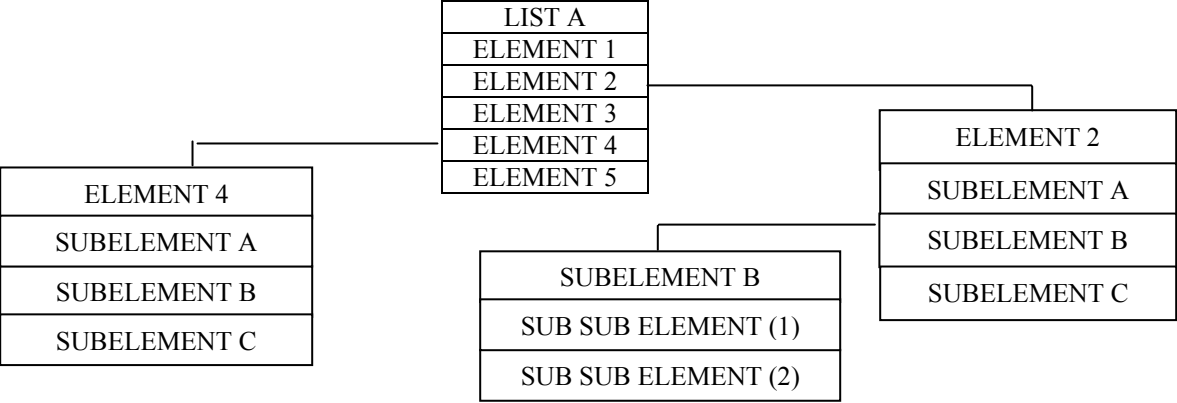
3.2 LIST dan TREE

List dan Tree merupakan struktur sederhana yang digunakan dalam representasi hirarki pengetahuan.

LIST

Adalah daftar dari rangkaian materi yang terkait. Hal ini bisa merupakan suatu daftar (list) nama orang yang anda kenal, barang-barang yang akan dibeli dari toko Serba Ada, hal-hal yang akan dikerjakan minggu ini, atau produk-produk berbagai jenis barang dalam katalog, dll.

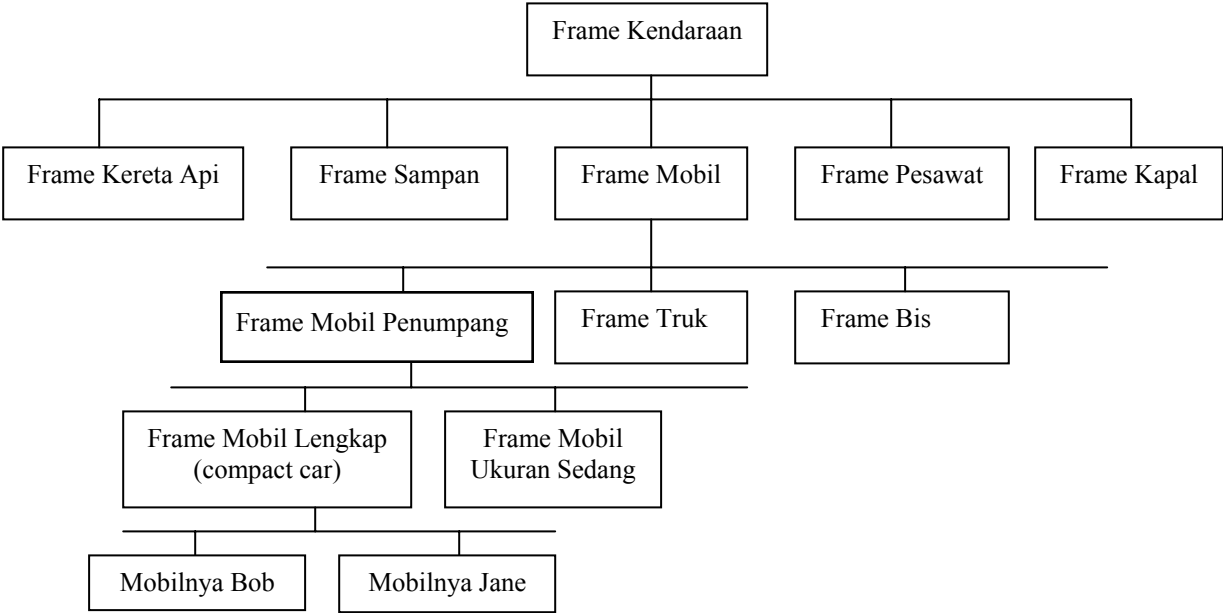
List biasanya digunakan untuk merepresentasikan hirarki pengetahuan dimana objek dikelompokkan, dikategorikan atau digabungkan sesuai dengan urutan atau hubungannya. Objek dibagi dalam kelompok atau jenis yang sama. Kemudian hubungan ditampilkan dengan menghubungkan satu sama lain.



Frame Mobil
Class : Transportasi
Nama pabrik : Audi
Negara : Jerman
Model : 5000 Turbo
Tipe : Sedan
Bobot : 3300 lb
Ukuran dasar roda : 105,8 inchi
Jumlah pintu : 4 (default)
Transmisi : 3-speed otomatis
Jumlah roda : 4 (default)
Mesin : (referensi kerangka mesin) <ul style="list-style-type: none">• Tipe : in-line, overhead cam• Jumlah silinder : 5
Akselerasi 0-60 : 40,4 detik ¼ mil : 17,1 detik, 85 mph Jarak gas : rata-rata 22 mpg
Frame Mesin
Kaliber silinder : 3,19 inci
Tak silinder : 3,4 inci
Rasio kompresi : 7,8 : 1
Sistem bahan bakar : injeksi dengan pertukaran turbo
Tenaga : 140 HP
Torsi : 160/ft/LB

HIRARKI FRAME

Kebanyakan sistem AI menggunakan kumpulan frame yang saling terkait satu dengan lainnya bersama-sama. Gambar di bawah ini menunjukkan hirarki frame kendaraan, terdiri dari 5 frame yaitu frame kereta api, frame sampan, frame mobil, frame pesawat, frame kapal. Masing-masing frame masih dapat dipecah lagi menjadi beberapa frame yang rinci, misal frame mobil terdiri dari frame penumpang mobil, frame truk, frame bis.



Susunan hirarki dari frame memungkinkan pewarisan frame. Akar dari tree terletak di puncak, dimana level tertinggi dari abstraksi disajikan. Frame pada bagian dasar (bawah) disebut daun dari tree. Hirarki memungkinkan pewarisan sifat-sifat. Setiap frame biasanya mewarisi sifat-sifat dari frame dengan level yang lebih tinggi. Pewarisan merupakan mekanisme untuk membentuk pengetahuan, yang menyediakan nilai slot, dari frame ke frame.

Didalam hirarki diatas, masing-masing frame dirinci hubungannya seperti hubungan antara frame orangtua (parent frame) dan anak (child frame)

Parent Frame		Child Frame	
Nama : Compact Car		Nama : Mobilnya Jane	
Slot	Facets	Slot	Facets
Pemilik	Cek daftar registrasi	Pemilik	Jane
Warna	Daftar per manufaktur	Warna	Biru
No silinder		No silinder	6
Range	4 atau 6		
Jika dibutuhkan	Tanya pemilik		
Buatan		Buatan	Honda
Daftar range	Semua manufaktur	Model	Accord
Jika dibutuhkan	Tanya pemilik		
Model	Gunakan hubungan frame		
Model (tahun)		Model (tahun)	1992
Range	1950 – 2001		
Jika dibutuhkan	Tanya pemilik		

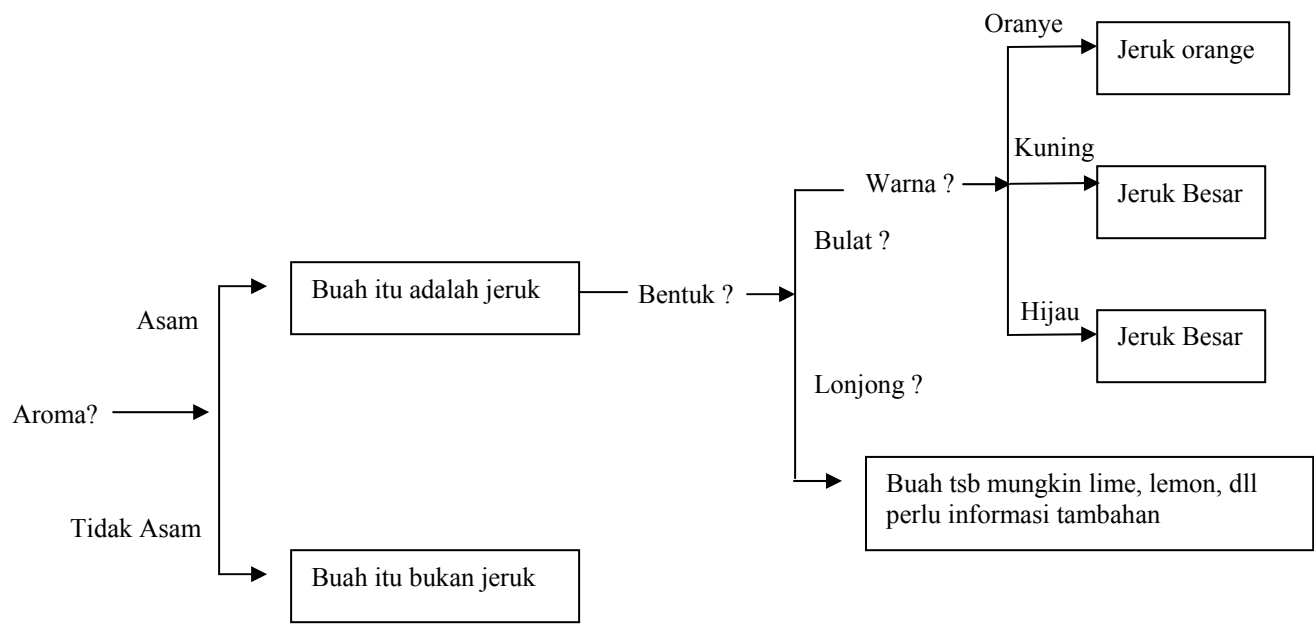
3.4. TABEL KEPUTUSAN (TABEL KEPUTUSAN)

- Pengetahuan diorganisasikan dalam format spreadsheet, menggunakan baris dan kolom.
- Tabel dibagi 2 bagian, pertama sebuah list dari atribut dibuat dan untuk setiap atribut semua nilai yang mungkin ditampilkan. Kemudian sebuah list kesimpulan dirumuskan
- Pengetahuan dalam tabel diperoleh dari proses akuisisi pengetahuan.

Atribut								
Bentuk	Bulat	Bulat	Bulat	Bulat	Lonjong	Lonjong	Lonjong	Lonjong
Aroma	Asam	Asam	Manis	Manis	Manis	Manis	Asam	Manis
Warna	Kuning	Oranye	Kuning	Merah	Kuning	Kuning	Oranye	Hijau
Rasa	Asam	Manis	Manis	Manis	Manis	Manis	Asam	Manis
Kulit	Kasar	Kasar	Halus	Halus	Halus	Halus	Halus	Halus
Kesimpulan								
Anggur	X							
Jeruk		X						
Apel			X	X				
Pisang					X			
Pir						X		X

3.5. POHON KEPUTUSAN (DECISION TREE)

Keuntungan utama representasi pengetahuan dengan pohon keputusan adalah dapat menyederhanakan proses akuisisi pengetahuan dan dapat dengan mudah dikonversikan ke bentuk aturan (rule)



3.6. NASKAH (SCRIPT)

Script adalah skema representasi pengetahuan yang sama dengan frame, yaitu merepresentasikan pengetahuan berdasarkan karakteristik yang sudah dikenal sebagai pengalaman-pengalaman.

Perbedaannya, frame menggambarkan objek, sedangkan script menggambarkan urutan peristiwa.

Dalam menggambarkan urutan peristiwa, script menggunakan slot yang berisi informasi tentang orang, objek, dan tindakan-tindakan yang terjadi dalam suatu peristiwa.

Elemen script meliputi :

1. Kondisi input, yaitu kondisi yang harus dipenuhi sebelum terjadi atau berlaku suatu peristiwa dalam script
2. Track, yaitu variasi yang mungkin terjadi dalam suatu script
3. Prop, berisi objek-objek pendukung yang digunakan selama peristiwa terjadi
4. Role, yaitu peran yang dimainkan oleh seseorang dalam peristiwa
5. Scene, yaitu adegan yang dimainkan yang menjadi bagian dari suatu peristiwa
6. Hasil, yaitu kondisi yang ada setelah urutan peristiwa dalam script terjadi.

Berikut ini adalah contoh script kejadian yang ada di “Ujian Akhir”

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

Adegan (scene) -1 : Persiapan pengawas

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

Adegan-2 : Mahasiswa masuk ruangan

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

Adegan - 3 : Mahasiswa mengerjakan soal ujian

- Mahasiswa menuliskan identitas di lembar jawab
- Mahasiswa menandatangani lembar jawab
- Mahasiswa mengerjakan soal
- Mahasiswa mengecek jawaban

Adegan - 4 : Mahasiswa telah selesai ujian

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

Adegan - 5 : Mahasiswa mengemasi lembar jawab

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

Hasil :

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

3.7 SISTEM PRODUKSI (ATURAN PRODUKSI/PRODUCTION RULES)

Representasi pengetahuan dengan sistem produksi berupa aplikasi aturan (rule) yang berupa :

1. Antecedent, yaitu bagian yang mengekspresikan situasi atau premis (pernyataan berawalan IF)
2. Konsekuen, yaitu bagian yang menyatakan suatu tindakan tertentu atau konklusi yang diterapkan jika suatu situasi atau premis bernilai benar (pernyataan berawalan THEN)

Konsekuensi atau konklusi yang dinyatakan pada bagian THEN baru dinyatakan benar, jika bagian IF pada sistem tersebut juga benar atau sesuai dengan aturan tertentu.

Contoh :

IF lalulintas pagi ini padat
THEN saya naik sepeda motor saja

Aturan dapat ditulis dalam beberapa bentuk :

1. IF premis THEN kesimpulan
Jika pendapatan tinggi MAKA pajak yang harus dibayar juga tinggi
2. Kesimpulan IF premis
Pajak yang harus dibayar tinggi JIKA pendapatan tinggi
3. Inclusion of ELSE
IF pendapatan tinggi OR pengeluaran tinggi, THEN pajak yang harus dibayar tinggi
ELSE pajak yang harus dibayar rendah
4. Aturan yang lebih kompleks
IF rating kredit tinggi AND gaji lebih besar dari \$30,000 OR aset lebih dari \$75,000 AND sejarah pembayaran tidak miskin THEN pinjaman diatas \$ 10,000 disetujui dan daftar pinjaman masuk kategori "B"

Apabila pengetahuan direpresentasikan dengan aturan, maka ada 2 metode penalaran yang dapat digunakan :

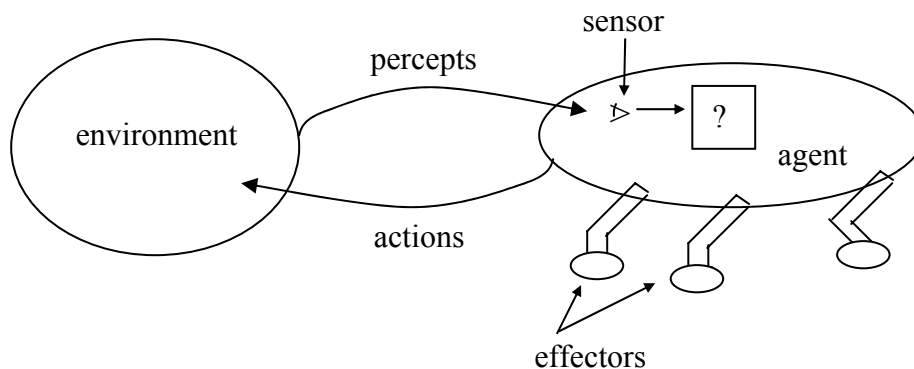
1. Forward Reasoning (penalaran maju)
Pelacakan dimulai dari keadaan awal (informasi atau fakta yang ada) dan kemudian dicoba untuk mencocokkan dengan tujuan yang diharapkan
2. Backward Reasoning (penalaran mundur)
Penalaran dimulai dari tujuan atau hipotesa, baru dicocokkan dengan keadaan awal atau fakta-fakta yang ada.

Ada beberapa faktor yang mempengaruhi pemilihan backward atau forward dalam memilih metode penalaran :

- banyaknya keadaan awal dan tujuan. Jika jumlah keadaan awal lebih kecil daripada tujuan, maka digunakan penalaran forward. Sebaliknya jika jumlah tujuan lebih banyak daripada keadaan awal, maka dipilih penalaran backward
- rata-rata jumlah node yang dapat diraih langsung dari suatu node. Lebih baik dipilih yang jumlah node tiap cabangnya lebih sedikit
- apakah program butuh menanyai user untuk melakukan justifikasi terhadap proses penalaran? Jika ya, maka alangkah baiknya jika dipilih arah yang lebih memudahkan user
- bentuk kejadian yang akan memicu penyelesaian masalah. Jika kejadian itu berupa fakta baru, maka lebih baik dipilih penalaran forward. Namun jika kejadian itu berupa query, maka lebih baik digunakan penalaran backward.

IV. INTELLIGENT AGENT

- Menurut Okamoto & Takaoka (1997):
Agent dapat dipandang sebagai sebuah objek yang mempunyai tujuan dan bersifat autonomous (memberdayakan resourcenya sendiri) untuk memecahkan suatu permasalahan melalui interaksi seperti kolaborasi, kompetisi, negosiasi, dsb
- Agent = sesuatu yang seolah-olah merasakan sesuatu dari lingkungannya melalui sensor dan memberikan aksi balasan kepada lingkungan tersebut melalui effector.
Multi agent = kumpulan dari beberapa agent yang berada pada lingkungan yang sama
- Human agent = agent yang dibuat menyerupai manusia memiliki mata, telinga dan organ lain sebagai sensor, serta tangan, kaki, mulut, dan bagian tubuh lain sebagai effector.
- Sebuah agent robot menggunakan kamera dan sinar infrared dalam range tertentu sebagai sensor dan berbagai motor (mesin) sebagai effector
- Berikut diagram interaksi agent dengan lingkungan melalui sensor dan effector



- Struktur intelligent agent :
Agent = arsitektur + program

Tujuan utama kecerdasan buatan adalah mendesain “program” bagi agent yang berfungsi mengimplementasikan pemetaan persepsi ke aksi dari suatu atau beberapa agent.

Program tersebut akan berjalan pada beberapa computing device yang disebut “arsitektur”.

Arsitektur = hardware yang digunakan untuk tujuan tertentu, misal : kamera, filtering audio input, dll

- Sebelum membuat agent harus mendesain/merancang gambaran tentang agent tersebut, yaitu :
 - *agent type*
 - *percepts and actions* yang ada pada agent
 - *goal / performance measure* yang akan agent capai
 - *environment* tempat agent tersebut beroperasi
- Contoh : agent untuk otomasi pengemudi taksi

Agent type	: pengemudi taksi
Percepts	: kamera, speedometer, accelerometer, GPS, microphone/keyboard, sensor infrared / sonar
Actions	: menyetir, menambah kecepatan, menurunkan kecepatan, mengerem, bicara dengan penumpang
Goals	: keamanan, kecepatan, kenyamanan perjalanan, keuntungan maksimum
Environment	: jalan, lalu lintas lainnya, konsumen pejalan kaki

Penjelasan :

Taksi otomatis harus mengetahui tempatnya berada, apa saja yang ada di jalanan dan berapa kecepatan yang dijalankan oleh taksi. Informasi tersebut diperoleh dari percept yang disediakan oleh satu atau lebih kamera, speedometer. Untuk mengontrol kendaraan menggunakan accelerometer khususnya pada saat tikungan. GPS untuk menentukan posisi, sensor infrared/sonar untuk mendeteksi jarak antara satu kendaraan dengan yang lain dan hambatan yang ada. Microphone/keyboard untuk penumpang yang akan memberitahu kemana tujuan mereka.

Action kurang lebih sama dengan pengemudi manusia yaitu pengontrolan pada saat menyetir, mengerem, mempercepat/melambatkan laju kendaraan. Adanya layar atau suara sintetik/sarana komunikasi lain untuk berbicara/menyampaikan informasi bagi penumpang

Beberapa hal dapat dijadikan sebagai ukuran keberhasilan taksi otomatis tersebut (performance measure/goal) yaitu :

- keberhasilan mencapai tempat tujuan penumpang
- minimalisasi konsumsi dan penggunaan bahan bakar
- minimalisasi biaya dan waktu perjalanan
- minimalisasi pelanggaran lalu lintas dan tabrakan dengan pengemudi lain
- maksimum keamanan
- kenyamanan yang diberikan kepada penumpang
- keuntungan maksimal

Taksi akan digunakan dalam lingkungan seperti apa? Misal :

- beroperasi pada lalulintas dalam kota atau jalan raya lintas?
- bisa digunakan di daerah bersalju atau tidak?
- Posisi pengemudi di kanan atau di kiri atau fleksibel di posisi manapun