

PERTEMUAN 1

DEFINISI KECERDASAN BUATAN

PENGANTAR...

- ✓ Bisakah mesin berpikir ?
- ✓ Jika bisa bagaimana caranya?
- ✓ Dan jika tidak bisa, kenapa tidak ?
- ✓ Dan apa yang dikatakan sebagai pikiran (mind) ?



© Toons4Biz * www.ClipartOf.com/12906

Arti kecerdasan

Kemampuan untuk....

- belajar atau mengerti dari pengalaman,
- memahami pesan yang kontradiktif dan ambigu,
- menanggapi dengan cepat dan baik atas situasi yang baru,
- menggunakan penalaran dalam memecahkan masalah serta menyelesaiakannya dengan efektif

(Winston dan Pendergast, 1994)

Konsep Kecerdasan Buatan

Merupakan kawasan penelitian, aplikasi dan instruksi yang terkait dengan pemrograman komputer untuk melakukan sesuatu hal - yang dalam pandangan manusia adalah – cerdas (H. A. Simon [1987])

Sebuah studi tentang bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia (Rich and Knight [1991])

Mengapa Perlu AI?

- a. Hampir semua permasalahan dipecahkan dengan bantuan komputer.
- b. Masalah semakin komplek tidak mungkin manual lagi.
- c. Tidak ada keterbatasan hardware lagi.
- d. Keinginan manusia:
Komputer bertindak seperti manusia.

Detail Kecerdasan Buatan

a. Sudut Pandang Kecerdasan

Kecerdasan buatan mampu membuat mesin menjadi cerdas (berbuat seperti yang dilakukan manusia).

b. Sudut Pandang Penelitian

Kecerdasan buatan adalah studi bagaimana membuat komputer dapat melakukan sesuatu sebaik yang dilakukan manusia.

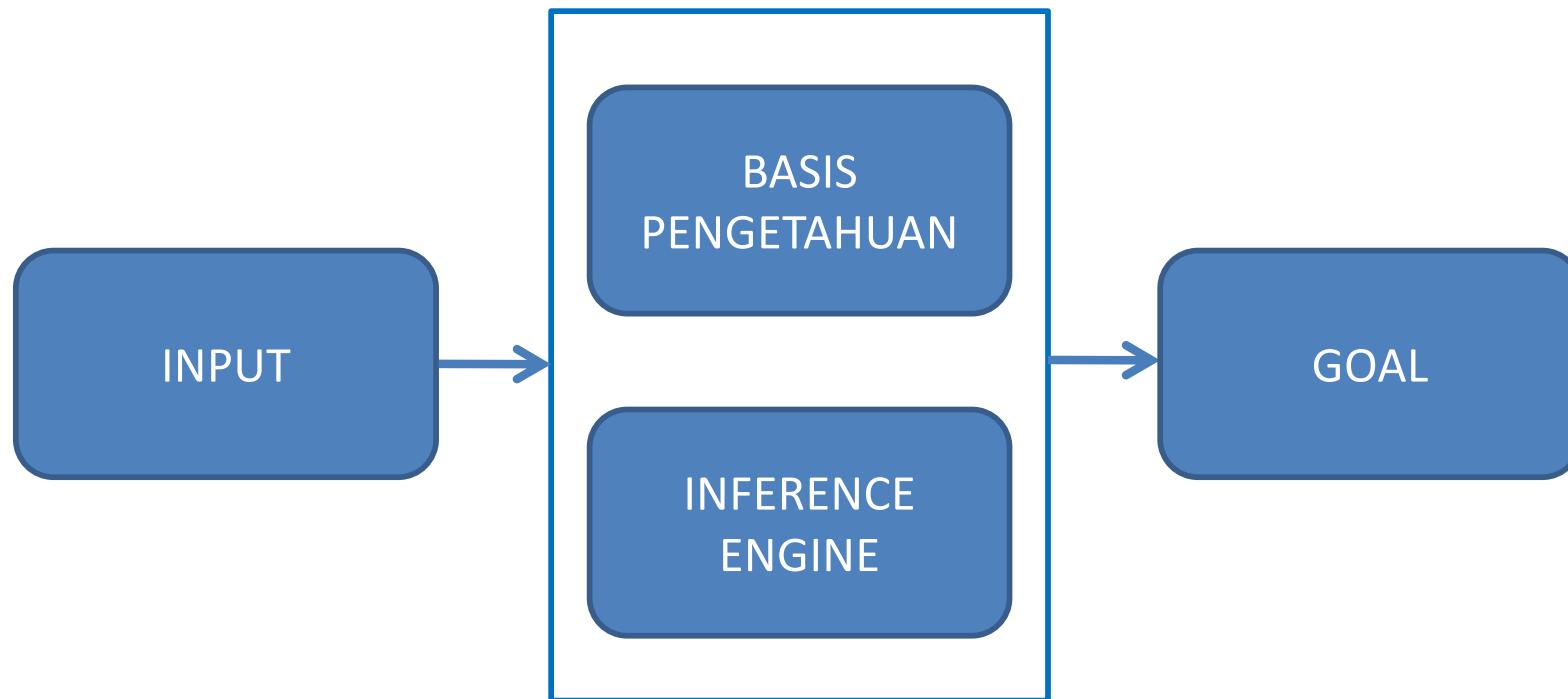
c. Sudut Pandang Bisnis

Kecerdasan buatan adalah kumpulan peralatan yang sangat powerful dan metodologis dalam menyelesaikan masalah bisnis

d. Sudut Pandang Pemrograman

Kecerdasan buatan meliputi studi tentang pemrograman simbolik, *problem solving*, dan pencarian (*searching*)

ARSITEKTUR INTELEGENSI



>> Basis Pengetahuan (*knowledge base*)

Berisi fakta-fakta, teori, pemikiran dan hubungan komponen satu dengan yang lainnya

>> Motor Inferensi (*inference engine*)

Kemampuan menarik kesimpulan berdasarkan pengalaman. Berkaitan dengan representasi dan duplikasi proses tersebut melalui mesin (misalnya, komputer dan robot).

Konsep Kecerdasan Buatan

a. Turing Test

Metode Pengujian Kecerdasan (Alan Turing).

Proses uji ini melibatkan seorang penanya (manusia) dan dua obyek yang ditanyai.

b. Pemrosesan Simbolik

Sifat penting dari AI adalah bahwa AI merupakan bagian dari ilmu komputer yang melakukan proses secara simbolik dan non-algoritmik dalam penyelesaian masalah.

c. Heuristic

Suatu strategi untuk melakukan proses pencarian (*search*) ruang problem secara efektif, yang memandu proses pencarian yang kita lakukan di sepanjang jalur yang memiliki kemungkinan sukses paling besar.

- d. Inferensi (Penarikan Kesimpulan) → AI mencoba membuat mesin memiliki kemampuan berpikir atau mempertimbangkan (*reasoning*), termasuk di dalamnya proses (*inferencing*) berdasarkan fakta-fakta dan aturan dengan menggunakan metode heuristik, dll
- e. Pencocokan Pola (*Pattern Matching*) → Berusaha untuk menjelaskan obyek, kejadian (*events*) atau proses, dalam hubungan logik atau komputasional

Tujuan Kecerdasan Buatan

- a. Membuat komputer lebih cerdas
- b. Mengerti tentang kecerdasan
- c. Membuat mesin lebih berguna

Perbedaan Kecerdasan Buatan dengan Kecerdasan Alami

- Lebih permanen
- Menawarkan kemudahan duplikasi dan penyebaran
- Lebih murah daripada kecerdasan alami
- Konsisten dan menyeluruh
- Dapat didokumentasikan
- Dapat mengeksekusi tugas tertentu lebih cepat daripada manusia
- Dapat menjalankan tugas tertentu lebih baik dari banyak atau kebanyakan orang.

Kelebihan Kecerdasan Alami dibanding AI

- Bersifat lebih kreatif
- Dapat melakukan proses pembelajaran secara langsung, sementara AI harus mendapatkan masukan berupa simbol dan representasi-representasi
- Menggunakan fokus yang luas sebagai referensi untuk pengambilan keputusan. Sebaliknya, AI menggunakan fokus yang sempit

Contoh Aplikasi AI

- Deep Blue mengalahkan Kasparov, juara dunia Catur.
- PEGASUS, suatu sistem memahami ucapan yang mampu menangani transaksi seperti mendapatkan informasi tiket udara termurah.
- MARVEL: suatu sistem pakar real-time memonitor arus data dari pesawat Voyager dan setiap anomali sinyal.
- Sistem robot mengemudikan sebuah mobil dengan kecepatan yang cepat pada jalan raya umum.

Definisi kecerdasan Buatan

- Suatu diagnostik sistem pakar sedang mengkoreksi hasil diagnosis pakar yang sudah punya reputasi.
- Agent pintar untuk bermacam-macam domain yang bertambah pada laju yang sangat tinggi .
- Subjek materi pakar mengajar suatu learning agent penalarannya dalam pusat penentuan gravitasi
- Aplikasi kecerdasan buatan dalam bidang Pertahanan dan Keamanan, **Face Recognition Software Deteksi Wajah**

Deep Blue



Deep Blue adalah sebuah [komputer catur](#) buatan [IBM](#). *Deep Blue* adalah komputer pertama yang memenangkan sebuah permainan catur melawan seorang juara dunia ([Garry Kasparov](#)) dalam waktu standar sebuah turnamen catur. Kemenangan pertamanya (dalam pertandingan atau babak pertama) terjadi pada [10 Februari 1996](#), dan merupakan permainan yang sangat terkenal. Namun Kasparov kemudian memenangkan 3 pertandingan lainnya dan memperoleh hasil [remis](#) pada 2 pertandingan selanjutnya, sehingga mengalahkan *Deep Blue* dengan hasil 4-2. *Deep Blue* lalu diupgrade lagi secara besar-besaran dan kembali bertanding melawan Kasparov pada Mei [1997](#). Dalam pertandingan enam babak tersebut *Deep Blue* menang dengan hasil 3,5-2,5. Babak terakhirnya berakhir pada [11 Mei](#).

Deep Blue menjadi komputer pertama yang mengalahkan juara dunia bertahan. Komputer ini saat ini sudah "dipensiunkan" dan dipajang di Museum Nasional Sejarah Amerika ([National Museum of American History](#)), Amerika Serikat.

Face Recognition Software



Contoh aplikasi kecerdasan buatan dalam bidang Pertahanan dan Keamanan ” : Face Recognition Software Deteksi Wajah. Seringkali kita bertemu dengan seseorang yang rasanya pernah kita kenal.

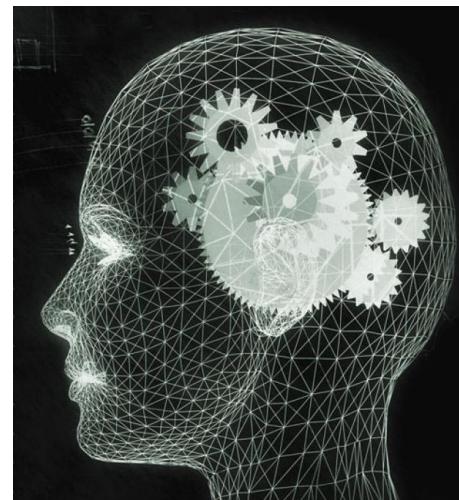
Setelah sedikit berbasa-basi, kita tahu bahwa ternyata dia adalah teman waktu SD dulu. Ini adalah bukti bahwa manusia memiliki kecerdasan untuk mengenali wajah. Kecerdasan inilah yang akan diadaptasi oleh komputer dengan nama **face recognition**.



THE END

PERTEMUAN 2

TEKNIK, MODEL & KRITERIA SUKSES, POTENSI
MANUSIA DALAM PEMROGRAMAN



2.1. Teknik AI

Para pengembang AI berpegang kepada prinsip bahwa akan ada teknik penyelesaian untuk setiap permasalahan yang berbeda, dimana teknik itu akan didasari kapada kemampuan untuk memanipulasi symbol (create, modification, reproduction dan destruction).

Hal lain yang harus dipahami adalah bahwa kecerdasan memerlukan pengetahuan, dimana pengetahuan memiliki karakteristik antara lain :

- sangat luas
- sulit didefinisikan dengan tepat
- selalu berubah
- dapat memiliki arti berbeda tergantung kapan digunakan

Beberapa teknik AI dapat dikategorikan secara umum kedalam beberapa kelompok, diantaranya :

1. **Search** (Pencarian)

menyediakan cara penyelesaian masalah untuk kasus dimana bila tidak ada lagi pendekatan langsung yang dapat digunakan maka pindahkan kerangka kerja kpd teknik langsung yang mungkin untuk dilekatkan.

2. **Use of Knowledge** (Penggunaan Pengetahuan)

menyediakan cara penyelesaian masalah yang lebih kompleks dengan mengekplorasi struktur dari objek yang terkait dengan masalah tsb.

3. **Abstraction**

menyediakan cara untuk memilah/memisahkan keterangan dan variasi yang penting dari sekian banyak yang tidak penting dimana akan mempercepat penyelesaian masalah.

2.2. Tingkatan Model

Sebelum kita membuat program AI ada baiknya kita tanya pada diri kita beberapa hal, seperti

“Apakah kita akan membuat program yang dapat melakukan hal secerdas yang dilakukan manusia ? ”

“ Apakah kita akan membuat program yang dapat melakukan sesuatu dgn cara yang sama dgn manusia ? ”

Atau kita akan membuat program yang dapat melakukan sesuatu yang lebih mudah dengan cara yg lebih mudah.

Jawaban dari pertanyaan diatas akan memberikan batasan dari pengembangan sistem / pembuatan program yang akan kita lakukan.

Usaha untuk membuat program yang menyajikan cara manusia menyelesaikan masalah dapat dibagi kedalam dua kelas, yaitu :

Program kelas pertama mencoba menyelesaikan masalah dengan cara yang tidak persis benar dengan definisi kita tentang AI. Program yang termasuk kelas ini menggunakan algoritma dan mekanisme yang mudah dan sederhana untuk dilakukan oleh komputer tetapi biasanya sulit dan tidak menarik untuk dilakuakan oleh manusia.

Contoh program dalam kelas ini adalah Elementary Perceiver and Memorizer (EPAM) [Feigenbaum, 1963] yang dapat mengingat pasangan terkait dari suku kata, dimana bila dimasukkan satu suku kata komputer tinggal mencarinya dalam memori kata mana yang mengandung suku kata yang dimaksud, yang pertama kali ditemukan maka itulah jawabannya, bagi manusia tidak semudah itu karena manusia selalu berfikir tentang arti dari kata yang dimaksud sesuai dengan konteks, sehingga masalah spt ini tidak menarik bagi manusia dan jarang dilakukan, namun hal semacam ini sering dilakukan dalam psychotest untuk mengetahui kemampuan mengingat seseorang.

Program kelas kedua berupaya memodelkan kemampuan manusia dalam melakukan sesuatu, yang berarti program pada kelas ini lebih mendekati kepada definisi tentang AI, yang berarti menjadi tidak mudah bagi komputer. Beberapa alasan dibuatnya model seperti ini al:

1. Untuk membuktikan teori psychology tentang kemampuan manusia. Contohnya adalah program PARRY yang ditulis Colby, 1975, yang mengeksploitasi perilaku paranoid manusia berdasarkan percakapan yang dilakukan, sehingga dengan menganalisa hasil percakapan, seorang psycholog dapat menyimpulkan apakah seseorang termasuk paranoid atau tidak.

* paranoid : gila karena ketakutan yang berlebihan

2. Untuk membuat komputer mengerti alasan manusia. Contohnya, membuat komputer dapat membaca/mengerti berita di koran dan menjawab pertanyaan spt "mengapa buruh mogok kerja?", program semacam ini harus dapat mensimulasi proses pengambilan alasan yang dilakukan manusia.

3. Untuk membuat manusia mengerti alasan komputer. Dalam banyak keadaan manusia enggan percaya pada output komputer kecuali dapat dimengerti bagaimana mesin mendapatkan hasil spt itu. Jika proses pengambilan alasan yg digunakan komputer sesuai dgn cara manusia maka akan lebih mudah untuk mendapatkan penjelasan yang dapat diterima.

4. Untuk mengeksploitasi pengetahuan apa yang dapat kita kumpulkan dari manusia.

Selama disepakati bahwa manusia memiliki kemampuan terbaik dalam menyelesaikan masalah, hal ini membuat banyak keinginan untuk melihat manusia sebagai petunjuk untuk menemukan cara untuk menyelesaikan masalah atau memproses suatu pekerjaan.

Hal ini juga akan memotivasi para pengembang AI untuk terus memproduksi mesin yang bertingkah laku cerdas dengan meniru manusia.

2.3. Kriteria Sukses

Satu pertanyaan terpenting yang harus dijawab pada tiap proyek penelitian ilmiah adalah " Bagaimana kita tahu kalau kita sudah berhasil ? ", begitu pula dalam AI. Bagaimana kita tahu mesin yang kita buat cerdas ? Menjawab pertanyaan itu sama sulitnya dengan menjawab pertanyaan "Apakah kecerdasan itu ?" tapi dapatkah kita melakukan sesuatu untuk memastikan kegiatan kita ?

Tahun 1950, Alan Turing memperkenalkan metode untuk menentukan apakah sebuah mesin dapat berpikir, yang kemudian dikenal dengan sebutan Turing Test.

Untuk melakukan test ini diperlukan 2 orang dan 1 mesin.

Satu orang bertindak sebagai penanya yang berada pada tempat terpisah dengan orang kedua dan mesin. Penanya dapat bertanya kepada orang kedua atau mesin dengan mengetikkan pertanyaannya dan menerima jawaban dalam bentuk ketikkan juga. Penanya tidak tahu yang mana orang yang mana mesin hanya si A dan si B, yang dilakukan oleh penanya dengan pertanyaan adalah menentukan mana yang orang, mana yang mesin. Tujuan dari test ini adalah mengelabui sifatnya sehingga menganggap mesin sebagai orang, caranya adalah membuat mesin tidak selalu menjawab benar dan menunda waktu menjawab.

Jika sang penanya akhirnya menyatakan mesin sebagai orang, maka dapat dikatakan mesin berhasil melewati test, dan dapat dinyakan bahwa mesin dapat berpikir.

Perlu waktu cukup lama dan perlu beberapa kali test dan perbaikan sampai akhirnya mesin ini dapat melewati test ini, namun hal yg menarik dari Turing Test ini adalah bahwa yang diperlukan oleh mesin untuk lulus test ini bukan jawaban yang benar atau tepat untuk tiap pertanyaan sehingga mesin tidak harus menjadi lebih cepat dan lebih benar dalam menjawab setiap pertanyaan untuk dinyatakan sebagai orang (cerdas) atau dapat berpikir.

2.4. POTENSI MANUSIA

1. Potensi Kecerdasan

Kecerdasan Spiritual

Kecerdasan Logika - Matematika

Kecerdasan Intrapersonal

Kecerdasan Musikal

Kecerdasan Natural

Kecerdasan Badan (Body) -Kinestetik

Kecerdasan Interpersonal

Kecerdasan Linguistik – Auditorial

Kecerdasan Spasial - Visual

2. Potensi Diri

Terdiri atas *empat elemen* yakni,

- Menerima diri
- Merumuskan Cita-cita
- Berinteraksi dengan lingkungan
- Mencari dan menciptakan pengalaman baru



3. Membangun Potensi



Tugas

- Cari contoh aplikasi kecerdasan buatan, dan jelaskan fungsi dari aplikasi tersebut di masyarakat !

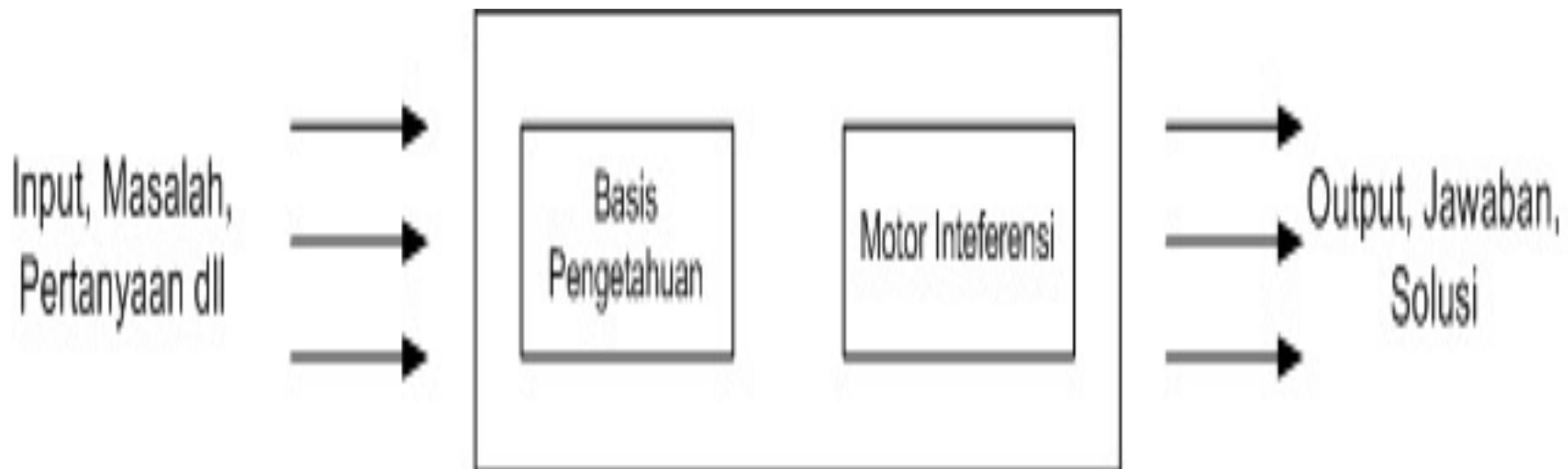


THE END

PERTEMUAN 3

MASALAH DAN METODE PEMECAHAN MASALAH

Sistem 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.

Untuk membuat system untuk menyelesaikan masalah terpisah, kita harus melakukan 4 hal sbb :

1. Mendefinisikan masalah dengan tepat, meliputi definisi yg tepat tentang keadaan awal dan keadaan akhir sebagai solusi yang dapat diterima.
2. Analisa masalah, beberapa fitur penting akan menentukan kelayakan dari beberapa teknik yang mungkin untuk menyelesaikan masalah.
3. Membatasi dan menghadirkan pengetahuan yang diperlukan untuk menyelesaikan masalah
4. Pilih teknik penyelesaian terbaik dan aplikasikan pada masalah.

Mendefinisikan masalah sebagai pencarian ruang stata

Untuk membuat deskripsi formal dari permasalahan, harus dilakukan beberapa hal, diantaranya :

1. Definisikan ruang stata yang memuat semua konfigurasi yg mungkin dari objek yang terkait (dan mungkin beberapa yg tidak mungkin). Hal ini, tentu saja mungkin untuk mendefinisikan ruang stata dengan jumlah stata yang tidak terbatas.
2. Tentukan satu atau beberapa stata yang menyatakan keadaan awal dari masalah, disebut *initial states*.
3. Tentukan satu atau beberapa stata yang dapat diterima sebagai keadaan akhir (solusi), disebut *goal states*.

4. Tentukan sejumlah aturan yang menentukan aksi yang diperkenankan, hal ini mencakup beberapa hal sbb :

- a. Apa asumsi non stata yang ditampilkan dalam deskripsi masalah non formal ?

Akan menentukan yg termasuk dan tidak termasuk stata

- b. Seberapa luas aturan harus dibuat ?

Dapat menentukan aturan diberlakukan untuk stata mana saja

- c. Berapa banyak pekerjaan yang diperlukan untuk menyelesaikan masalah harus disusun dan dimasukkan ke dalam aturan ?

Akan menentukan jumlah baris aturan

Sistem Produksi

Sistem produksi terdiri dari :

1. Sejumlah aturan, dimana tiap aturan memiliki sis kiri yang menyatakan bentuk yang dapat digunakan dan sisikanan yang menyatakan hasil operasi jika aturan diaplikasikan.
2. Satu atau lebih pengetahuan/database yang berisi informasi apapun yang berkaitan dgn tiap masalah. Beberapa bagian database mungkin tetap sedangkan yang lain ditambahkan sesuai dengan masalah yang dihadapi. Informasi dalam database dapat tersusun dalam banyak cara berbeda.

3. Strategi kendali yang akan menentukan perintah dalam aturan yang mana yang akan dibandingkan dengan database dan cara menyelesaikan konflik yang muncul ketika ada lebih dari satu aturan yang cocok.
4. Penggunaan aturan

Strategi Kendali

Strategi kendali diperlukan untuk memutuskan aturan mana yang akan digunakan dalam lanjutan proses pencarian untuk mendapatkan penyelesaian masalah, hal ini akan semakin diperlukan jika terdapat lebih dari satu atau semakin banyak aturan yang mungkin untuk digunakan dalam tiap tahap proses. Pada akhirnya bagaimana keputusan dibuat akan mempengaruhi kecepatan dalam mendapatkan penyelesaian masalah.

Syarat suatu strategi kendali yang baik adalah :

1. Menggerakkan stata (menjalankan proses) menuju solusi
2. Harus sistematis

Kedua syarat itu akan melahirkan banyak metode pencarian

Contoh Kasus : Bejana Air

Diberikan dua bejana air, dengan kapasitas 4 liter dan 3 liter, yang keduanya tidak memiliki skala/ batas ukuran.

Ada sejumlah tak terbatas air yang dapat diisi ke dalam bejana. Bagaimana cara mengisikan tepat 2 liter air ke dalam bejana berukuran 4 liter?

Definisi ruang stata kasus Bejana Air

Definisi ruang stata untuk kasus bejana air dapat dinyatakan sebagai pasangan bilangan bulat

(x,y) ; $x=0,1,2,3$ atau 4 dan $y=0,1,2$ atau 3

Dimana x menyatakan volume air pada bejana 4 liter dan y menyatakan volume air pada bejana 3 liter.

Stata awal (initial state) : $(0,0)$ dm n kedua bejana kosong

Stata akhir (goal state) : $(2, n)$ dm n bejana pertama berisi 2 liter dan bejana kedua bisa berisi berapa saja.

Sistem Produksi Bejana Air

	Keadaan (yg mungkin)	Hasil (yg dilakukan)	Arti
1.	(x,y) If $x < 4$	$\rightarrow (4,y)$	Penuhi bejana 4 liter
2.	(x,y) If $y < 3$	$\rightarrow (x,3)$	Penuhi bejana 3 liter
3.	(x,y) If $x > 0$	$\rightarrow (x - d, y)$	Tuang air dari bejana 4 liter
4.	(x,y) If $y > 0$	$\rightarrow (x, y - d)$	Tuang air dari bejana 3 liter

	keadaan	Hasil	Arti
5.	(x,y) If $x > 0$	$\rightarrow (0,y)$	Kosongkan bejana 4 liter
6.	(x,y) If $y > 0$	$\rightarrow (x,0)$	Kosongkan bejana 3 liter
7.	(x,y) If $x+y \geq 4$ and $y > 0$	$\rightarrow (4, y-(4-x))$	Tuang air dari bejana 3 liter ke bejana 4 liter sampai bejana 4 liter penuh
8.	(x,y) If $x+y \geq 3$ and $x > 0$	$\rightarrow (x-(3-y), 3)$	Tuang air dari bejana 4 liter ke bejana 3 liter sampai bejana 3 liter penuh

	keadaan	hasil	Arti
9.	(x,y) If $x+y \leq 4$ and $y > 0$	$\rightarrow (x+y, 0)$	Tuang semua air dari bejana 3 liter ke bejana 4 liter
10.	(x,y) If $x+y \leq 3$ and $x > 0$	$\rightarrow (0, x+y)$	Tuang semua air dari bejana 4 liter ke bejana 3 liter
11.	$(0,2)$	$\rightarrow (2, 0)$	Tuang 2 liter air dari bejana 3 liter ke bejana 4 liter
12.	$(2,y)$	$\rightarrow (0, y)$	Buang 2 liter air dari bejana 4 liter

Penyelesaian Kasus Bejana Air

Operator	Isi Bejana 4 ltr	Isi Bejana 3 ltr
	0	0
2	0	3
9	3	0
2	3	3
7	4	2
5 atau 12	0	2
9 atau 11	2	0

Hasil Analisa Masalah

Permasalahan dapat dipecahkan dengan menggunakan aturan produksi yang dikombinasikan dengan pendekatan strategi kendali untuk menelusuri ruang masalah sampai ditemukan jalur dari initial state ke goal state. Sehingga proses pencarian menjadi penting untuk menyelesaikan masalah. Pencarian adalah mekanisme umum yang dapat digunakan ketika tidak ada metode langsung lain yang diketahui.

PERTEMUAN 4

SEARCH

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 perfomansi 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?**

Pencarian atau pelacakan merupakan salah satu teknik untuk menyelesaikan permasalahan dalam bidang kecerdasan buatan. Teknik dasar pencarian masalah memberikan suatu kunci bagi banyak sejarah penyelesaian yang penting dalam bidang kecerdasan buatan. Contoh beberapa aplikasi yang menggunakan teknik pencarian yaitu :

- Masalah *routing (travelling salesman problem)*
- Parsing bahasa dan interpretasinya
- Permainan
- logika pemrograman (pencarian fakta dan implikasinya)
- Pengenalan pola
- Sistem pakar berbasis kaidah (*rule based expert system*)

Teknik Pencarian

Pada dasarnya ada dua teknik pencarian yaitu yang biasanya digunakan, yaitu :

1. Pencarian buta (*blind search*)
2. Pencarian terbimbing (*heuristic search*)

Pencarian Buta

Pencarian buta merupakan sekumpulan prosedur yang digunakan dalam melacak ruang keadaan. Pencarian berlangsung sampai solusi terakhir ditemukan. Idenya adalah menguji seluruh kemungkinan yang ada untuk menemukan solusi.

Pendekatan ini kurang efisien dan merupakan pemaksaan (*brute force search*). Dalam memecahkan masalah yang sangat besar sejumlah keadaan baru muncul, sehingga alternatif yang perlu dipertimbangkan pun menjadi lebih banyak. Akibatnya diperlukan waktu yang lama untuk menemukan satu solusi.

Pencarian Heuristic

Kata heuristic berasal dari bahasa Yunani heuriskein dari kata dasar eureka atau heurika yang berarti mengungkap atau menemukan.

Dalam AI, heuristic diperkenalkan sebagai suatu teknik yang meningkatkan efisiensi proses pencarian, yang dimungkinkan dengan mengorbankan kelengkapan.

Heuristic seperti pemandu perjalanan, yang baik untuk tujuan pokok mencari arah yang secara umum menarik, tetapi bisa jadi tidak baik jika mempertimbangkan ketertarikan tiap orang berbeda untuk tiap objek berbeda.

Menggunakan heuristic kita berharap mendapatkan solusi yang baik dari masalah yang sulit

Satu contoh general-purpose heuristic yang baik yang berguna untuk banyak kombinasi masalah adalah *nearest neighbor heuristic*, yang bekerja dengan menyeleksi alternatif lokal terbaik pada tiap langkah.

Aplikasinya adalah dalam masalah *Travelling Salesman*, yang menggunakan beberapa prosedur berikut :

1. Pilih secara acak satu kota sebagai awal perjalanan
2. Untuk memilih kota berikut, lihat semua kota yang belum dikunjungi dan pilih yang terdekat lalu kunjungi.
3. Ulangi langkah 2 sampai semua kota dikunjungi.

Karakteristik Masalah

Pencarian heuristic adalah metode yang sangat umum yang dapat diterapkan dalam begitu banyak masalah, meliputi begitu banyak variasi teknik yang spesifik, dimana masing-masing efektif untuk penyelesaian masalah tertentu yang lebih spesifik. Untuk memilih metode mana (atau kombinasi metode mana) yang akan digunakan untuk menyelesaikan masalah, penting untuk menganalisa masalah pada beberapa dimensi kunci atau karakteristik, sebagai berikut :

- Dapatkah masalah disederhanakan kedalam kelompok terpisah yang lebih kecil atau subprogram yang lebih mudah ?
- Dapatkah satu tahap penyelesaian solusi diabaikan atau setidaknya tidak dilakukan jika terbukti tidak layak ?
- Apakah ruang lingkup masalah dapat diprediksi ?
- Dapatkah dinyatakan sebuah solusi yang baik untuk penyelesaian masalah tanpa membandingkannya dengan solusi lain yang mungkin ?
- Solusi yang diinginkan adalah sebuah statua atau jalur menuju statua ?

- Apakah sejumlah pengatahan mutlak diperlukan untuk menyelesaikan masalah atau pengetahuan hanya diperlukan untuk membatasi pencarian ?
- Dapatkah komputer yang diberikan permasalahan langsung memberikan solusi atau pemecahan masalah memerlukan interaksi antara komputer dan manusia ?

Teknik Search

- Arah search

Dapat dilakukan :

Maju, bermula dari keadaan awal (*start state*)

Mundur, diawali dari keadaan tujuan (*goal state*)

- Topologi proses *search*

Ada dua macam penggambaran problem, yaitu dalam bentuk :

1. Pohon (*tree*)

2. Graf (*graph*) :Graf berarah dan Graf tidak berarah

Metode Search

Beberapa metode search yang akan dipelajari :

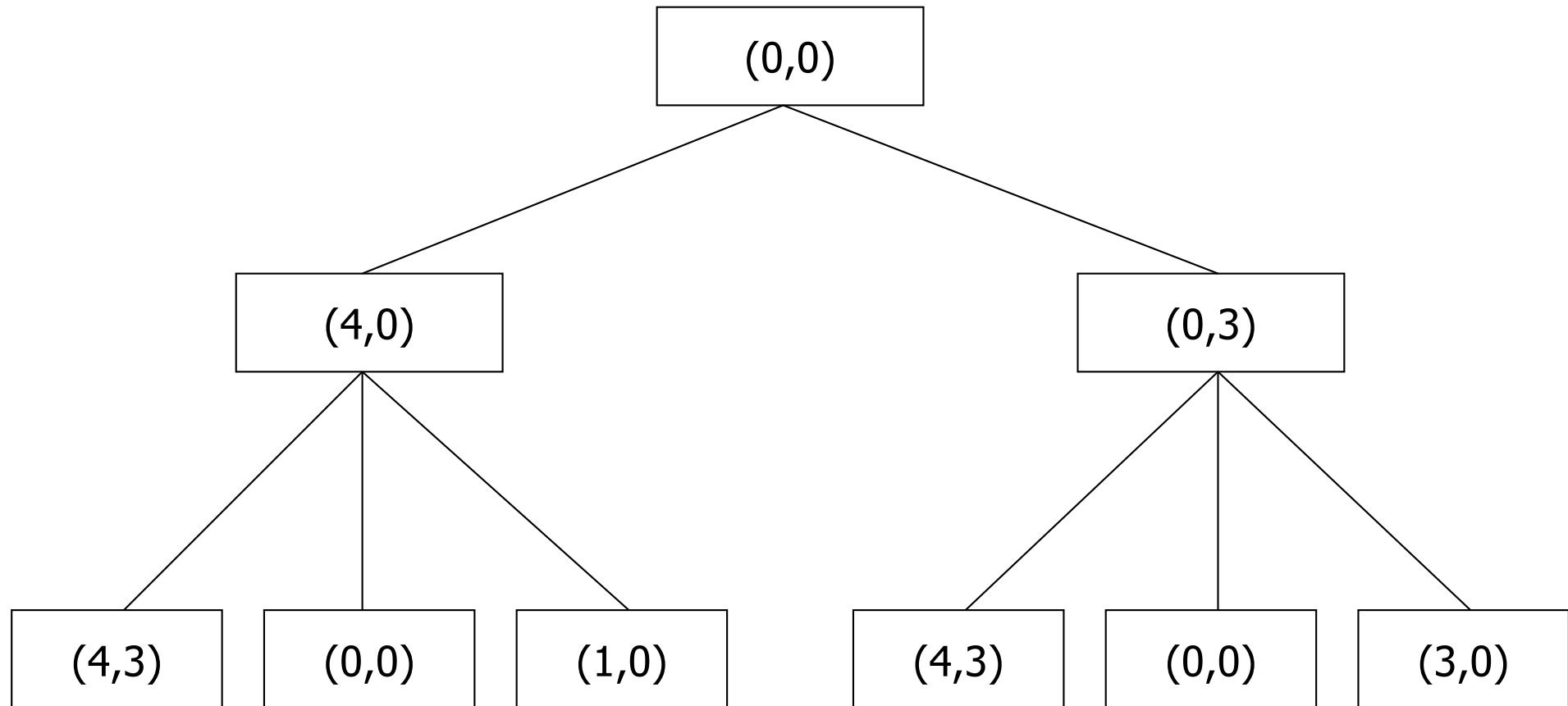
1. Breadth-First-Search
 2. Depth-First-Search
 3. Generate-and-Test
 4. Hill-Climbing
 5. Best-First-Search
- Pencarian buta (1,2), pencarian heuristic (3,4,5)

Breadth-First-Search

Algoritma Breadth-First-Search :

1. Bentuk variabel dengan nama NODE-LIST dan jadikan sebagai initial state.
2. Sampai goal state ditemukan atau NODE-LIST kosong, lakukan :
 - a. ambil elemen pertama dari NODE-LIST, sebut E.
jika NODE-LIST kosong, quit.
 - b. Untuk tiap cara dimana tiap aturan(fungsi) dapat cocok dengan stata di E, lakukan :
 - i. Gunakan aturan(fungsi) untuk menuju stata baru
 - ii. Jika stata baru adalah goal state, quit return stata ini
 - iii. Jika bukan, tambahkan stata baru di akhir NODE-LIST.

Breadth-First Search Tree untuk masalah bejana air



Kebaikan dan Keburukan Breadth-First-Search

Kebaikan Breadth-First-Search :

- Breadth-First-Search tidak akan terjebak untuk menelusuri satu jalur tertentu saja
- Jika solusi memang ada, maka dijamin Breadth-First-Search akan menemukannya.

Keburukan Breadth-First-Search :

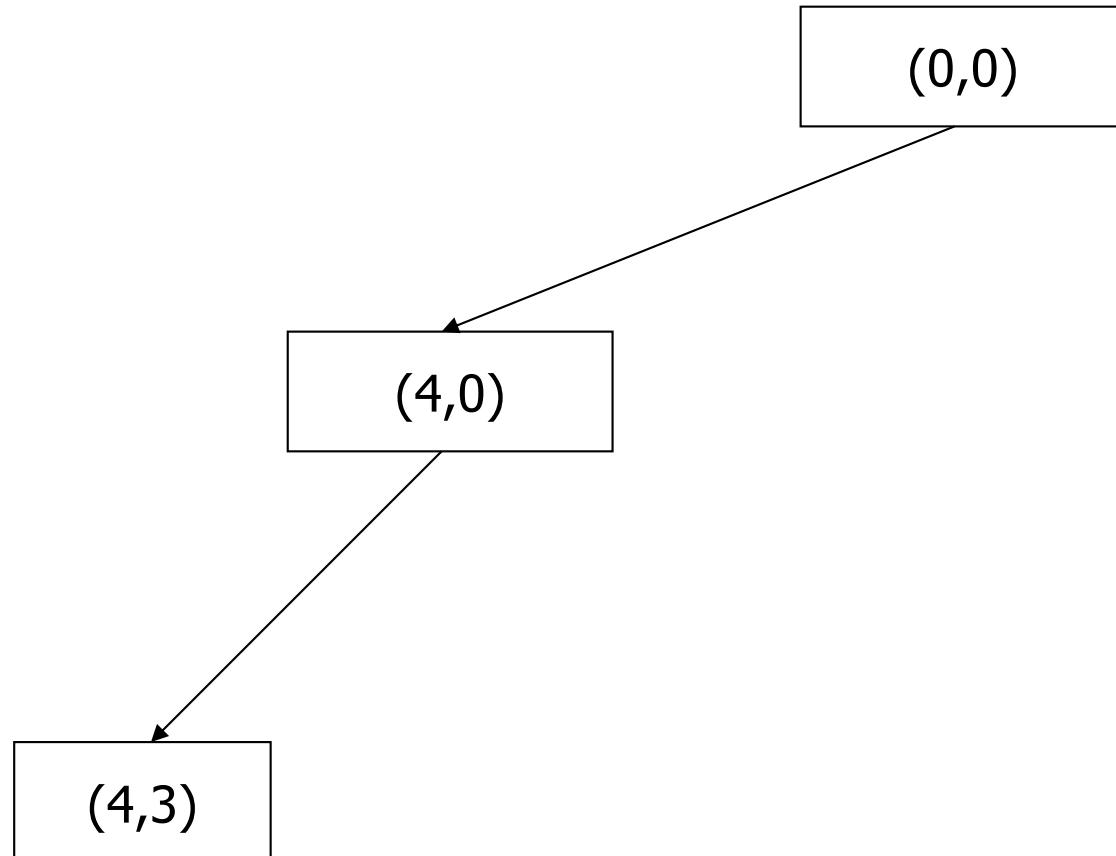
- Memerlukan memori lebih besar karena harus menyimpan semua simpul dari tree yang ditelusuri
- Harus menelusuri semua bagian tree pada level yang sama sebelum beralih ke level berikutnya.

Depth-First-Search

Algoritma Depth-First-Search :

1. Jika initial state adalah goal state, quit dan return success
2. Jika bukan, lakukan dibawah ini sampai dicapai sinyal success atau gagal
 - a. Tentukan successor, E dari initial state. Jika tidak ada lagi successor, maka sinyal gagal
 - b. Jalankan Depth-First-Search dengan E sebagai initial state
 - c. Jika success dihasilkan, sinyal success. Jika tidak maka ulangi langkah 2

Depth-First Search Tree untuk masalah bejana air



Kebaikan dan Keburukan Depth-First-Search

Kebaikan Depth-First-Search :

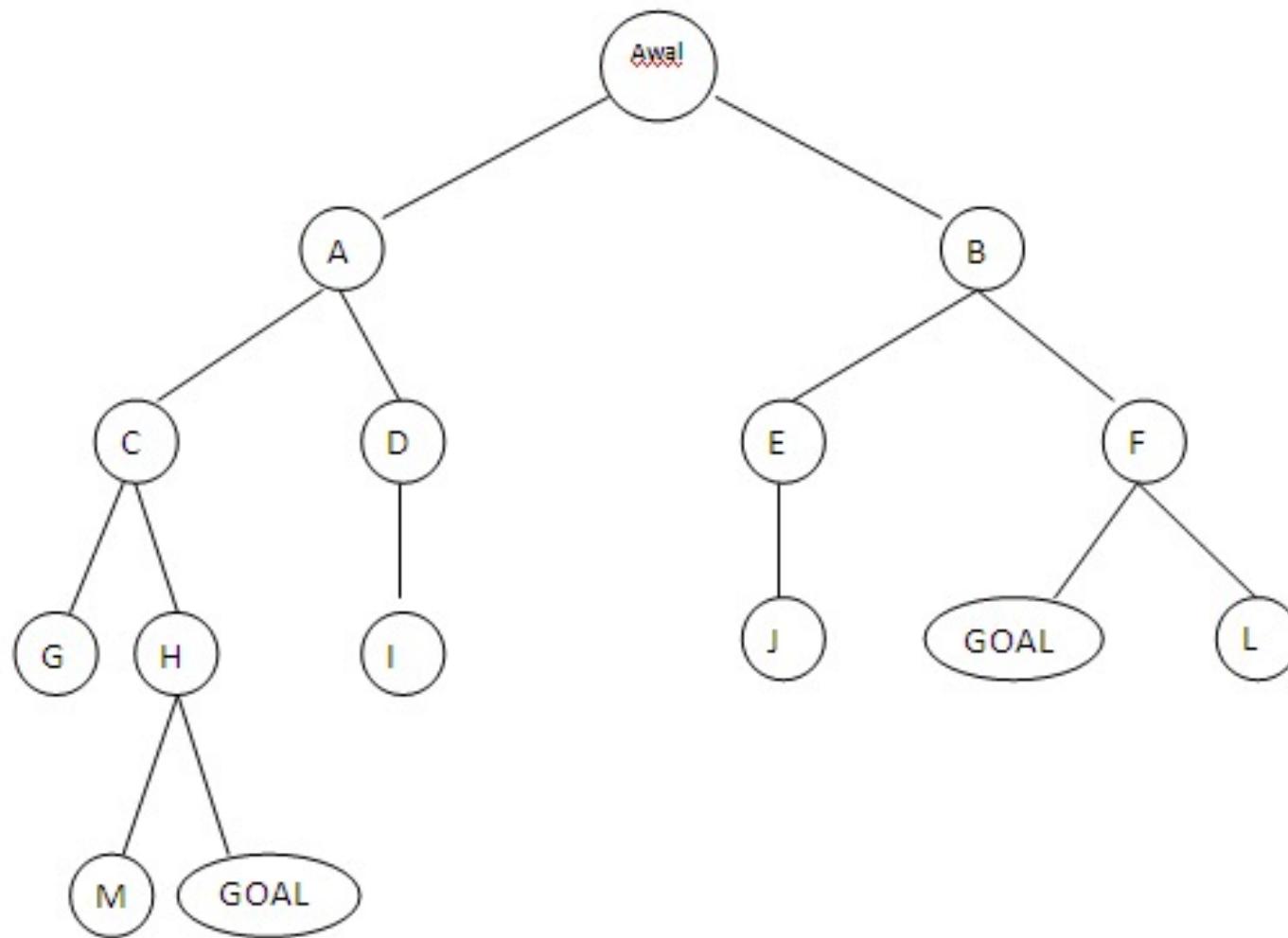
- Depth-First-Search memerlukan ruang memori lebih kecil karena hanya menyimpan simpul-simpul dari path/jalur yang sedang dikerjakan.
- Dapat menemukan solusi tanpa menelusuri terlalu banyak ruang search.

Keburukan Depth-First-Search :

- Ada kemungkinan terjebak pada satu jalur sampai terlalu jauh, bahkan selamanya, sebelum jalur tsb mendapatkan statua yang tidak lagi memiliki successor (buntu).
- Mungkin menemukan jalur panjang ke solusi pada satu bagian dari tree, sementara jalur terpendek tersedia pada bagian lain tree yang belum ditelusuri

Tambahan Contoh Kasus

Representasi masalah dalam tree



Menggunakan teknik *breadth-first search*

Node awal adalah awal dan tujuan adalah Goal, langkah-langkahnya :

1. Open := [Awal]; closed := []
2. Open:= [A,B]; closed:= [Awal]
3. Open:= [B,C,D]; closed:= [A,Awal]
4. Open:= [C,D,E,F]; closed:= [B,A,Awal]
5. Open:= [D,E,F,G,H]; closed:= [D,C,B,A,Awal]
6. Open:= [E,F,G,H,I]; closed:= [D,C,B,A,Awal]
7. Open:= [F,G,H,I,J]; closed:= [E,D,C,B,A,Awal]
8. Open:= [G,H,I,J,GOAL,L]; closed:= [F,E,D,C,B,A,Awal]
9. OPEN := [H,I,J,GOAL,L]; closed:= [G,F,E,D,C,B,A,Awal]
10. Open:= [I,J,GOAL,L]; closed:=[H,G,F,E,D,C,B,A,Awal]
11. Open:= [GOAL,L]; closed:=[I,J,H,G,F,E,D,C,B,A,Awal]
12. Open:= [L]; closed:=[GOAL,I,J,H,G,F,E,D,C,B,A,Awal]

Menggunakan teknik *depth-first search*

Node awal adalah awal dan tujuan adalah Goal, langkah-langkahnya :

1. Open := [Awal]; closed := []
2. Open := [A,B]; closed := [Awal]
3. Open := [C,D,B]; closed := [A,awal]
4. Open := [G,H,D,B]; closed := [C,A,awal]
5. Open := [H,D,B]; closed := [G,C,A,awal]
6. Open := [M,Goal,D,B]; closed := [G,C,A,awal]
7. Open := [Goal,D,B]; closed := [M,G,C,A,awal]
8. Open := [D,B]; closed := [Goal,M,G,C,A,awal]



THE END

PERTEMUAN 5

SEARCH BAGIAN 2

Generate-and-Test

Teknik Generate-and-Test adalah teknik yang paling mudah dibandingkan teknik search yang lain, namun relatif lebih lama dalam mendapatkan solusi.

Algoritma Generate-and-Test :

1. Bentuk solusi yang mungkin.

Untuk beberapa masalah, ini berarti membentuk poin terpisah dari area permasalahan. Pada masalah lain, ini berarti membentuk jalur dari stata awal.

2. Lakukan test untuk melihat apakah poin yang ditemui adalah solusi dengan membandingkan poin yang dipilih atau poin terakhir dari jalur yang dipilih dengan kumpulan stata tujuan
3. Jika solusi sudah ditemukan, quit. Jika belum kembali ke langkah 1.

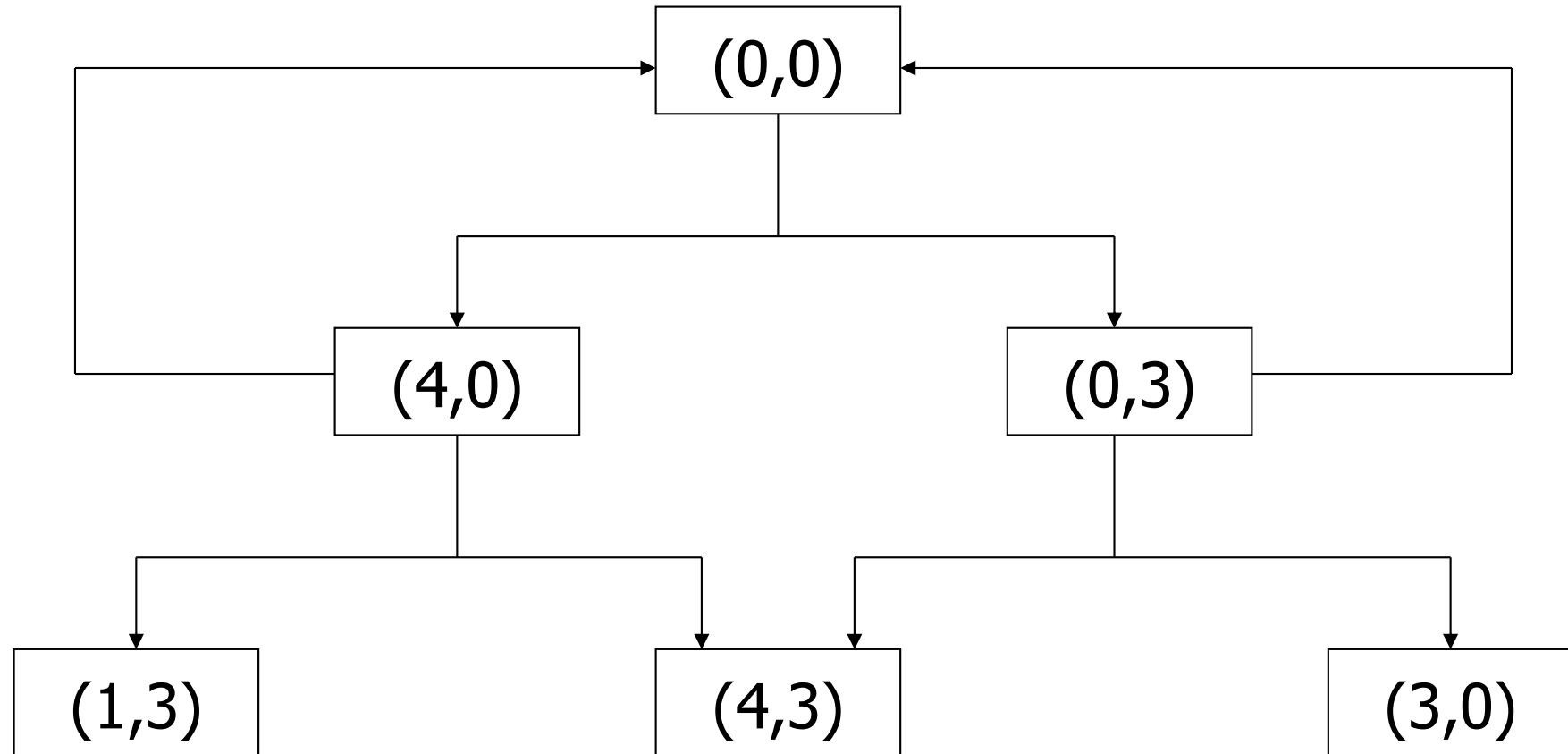
Kebaikan dan Keburukan Generate-and-Test

Jika penurunan solusi yang mungkin dilakukan secara sistematis, maka procedure diatas akan dapat menemukan solusi suatu saat, jika memang ada. Tapi sayangnya jika ruang permasalahan sangat luas maka saat ditemukannya solusi akan menjadi sangat lama.

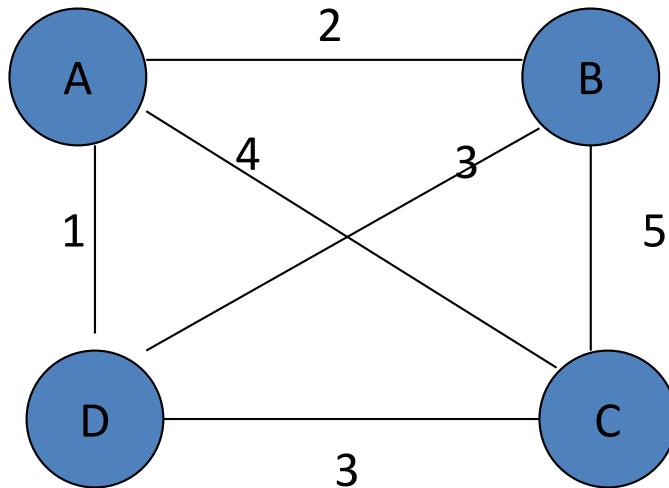
Cara terbaik menerapkan generate-and-test yang sistematis adalah pada tree dari depth-first search dengan backtracking, yaitu kembali ke stata sebelumnya bila ditemui stata yg sudah pernah di test atau memodifikasi prosedurnya untuk menelusuri stata pada bentuk graph.

Contoh Kasus 1

Search Graph untuk masalah bejana air



Contoh kasus TSP



Sebuah rute yang harus dilewati seorang sales dimana sales tersebut harus merlewati setiap kota tepat sekali. Terdapat 4 kota, dengan jarak masing-masing kota $AB=2$, $AC=4$, $AD=1$, $BC=5$, $BD=3$, $CD=3$. Tujuannya adalah mencari jarak terpendek bagi sales untuk mengunjungi semua kota sekali.

Penyelesaian menggunakan generate-test adalah dengan membangkitkan solusi-solusi yang mungkin ada sesuai permasalahan yang dihadapi oleh sales tersebut. Kombinasi abjad sebagai solusi yang mungkin adalah $n! = 4! = 24$. Tujuannya adalah mencari solusi dengan panjang terpendek.

No pencarian	Lintasan	Panjang Lintasan	Lintasan yang dipilih	Panjang lintasan
1	ABCD	10	ABCD	10
2	ABDC	8	ABDC	8
3	ACBD	12	ABDC	8
4	ACDB	10	ABDC	8
5	ADCB	9	ABDC	8
6	ADCB	9	ABDC	8
7	BACD	9	ABDC	8
8	BADC	6	BADC	6
9	BCAD	10	BADC	6
10	BCDA	9	BADC	6
11	BDAC	8	BADC	6
12	BDCA	10	BADC	6
13	CABD	9	BADC	6

14	CADB	8	BADC	6
15	CBAD	8	BADC	6
16	CBDA	9	BADC	6
17	CDAB	6	BADC/CDAB	6
18	CDBA	8	BADC/CDAB	6
19	DABC	8	BADC/CDAB	6
20	DACB	10	BADC/CDAB	6
21	DBAC	9	BADC/CDAB	6
22	DBCA	12	BADC/CDAB	6
23	DCAB	9	BADC/CDAB	6
24	DCBA	10	BADC/CDAB	6

Dari tabel diatas, solusi pertama yang dibangkitkan adalah $ABCD = 10$, solusi kedua $ABDC=8$. Ternyata solusi kedua menghasilkan jarak yang lebih pendek sehingga dipilih lintasan $ABDC=8$. Lakukan untuk langkah selanjutnya. Pada tabel didapat solusi terpendek adalah $BADC$ atau $CDBA$.

Kelemahan dari teknik ini pelru dibangkitkan semua kemungkinan yang ada sehingga apabila ditambahkan satu kota untuk permasalahan TSP ini diatas 5 kota. Maka akan diperlukan 120 kombinasi lintasan, kecuali diberikan kondisi tertentu misalnya kota awal bagi sales telah ditentukan.

Hill Climbing

Teknik Hill Climbing adalah pengembangan dari teknik Generate-and-Test, dengan penambahan adanya umpan balik dari prosedur test yang sudah digunakan untuk membantu memilih arah mana yang harus ditelusuri pada setiap area search.

Pada prosedur Generate-and-Test yang murni, fungsi test hanya ditanggapi dengan Ya atau Tidak. Tetapi pada Hill-Climbing fungsi test ditambahkan dengan fungsi heuristic atau fungsi objectif yang memungkinkan perkiraan seberapa dekat simpul yang ditelusuri terhadap goal state.

Hill-climbing sering kali digunakan jika fungsi heuristic yang baik tersedia untuk mengevaluasi statis, tapi ketika tidak ada lagi pengetahuan yang dapat digunakan.

Sebagai contoh, anda berada di suatu kota yang belum pernah anda kunjungi tanpa memiliki peta. Tujuannya menuju gedung tertinggi yang terlihat dari tempat anda berada.

Fungsi heuristic adalah hanya masalah jarak antara lokasi anda berada dengan letak gedung tertinggi dan bagaimana menemukan jarak yang terdekat atau cara tercepat menuju gedung tertinggi.

Penyelesaian masalah diatas dimulai dengan meninjau karakteristik masalah, apakah solusi yang pertama ditemukan dapat diterima sebagai solusi yang baik ? (mutlak atau relatif ?) Karena tidak ada peta dan tidak ada pengalaman memilih jalan (tidak ada pengetahuan) maka dipilih saja jalan yang arahnya menuju solusi sampai kita tiba di tujuan tanpa mengulangi atau mencoba lagi jalur yang lain dan kita terima itu sebagai solusi terbaik (dgn mengabaikan kemungkinan lain).

Jadi adalah masuk akal menerapkan hill-climbing ketika tidak ada alternatif yang dapat diterima untuk memilih atau menuju pada suatu stata.

Algoritma Simple Hill Climbing :

1. Evaluasi initial state. Jika ini goal state maka return dan keluar. Jika bukan maka lanjutkan dengan initial state sebagai current state.
2. Ulangi langkah berikut sampai menemukan solusi atau sampai tidak ada lagi operator yang dapat digunakan pada current state
 - a. Pilih operator yang belum digunakan pada current state dan gunakan untuk menghasilkan/menuju stata baru
 - b. Evaluasi stata baru.
 - i. Jika ini goal state maka return dan keluar
 - ii. Jika bukan goal state tetapi lebih baik dari current state maka jadikan stata baru sebagai current state
 - iii. Jika tidak lebih baik dari current state lanjutkan perulangan

Sebagai ilustrasi teknik pencarian *simple hill climbing* digunakan contoh masalah TSP pada masalah *generate and test*. Operator yang digunakan adalah operator yang dapat menghasilkan kombinasi lintasan kota yang berbeda-beda, yaitu dengan cara menukar posisi masing-masing kota. Untuk mempermudah penukaran posisi, kita cukup menukar posisi 2 kota, operator untuk kombinasi lintasan dengan menukar posisi 2 kota dapat dihitung dengan kalkulasi

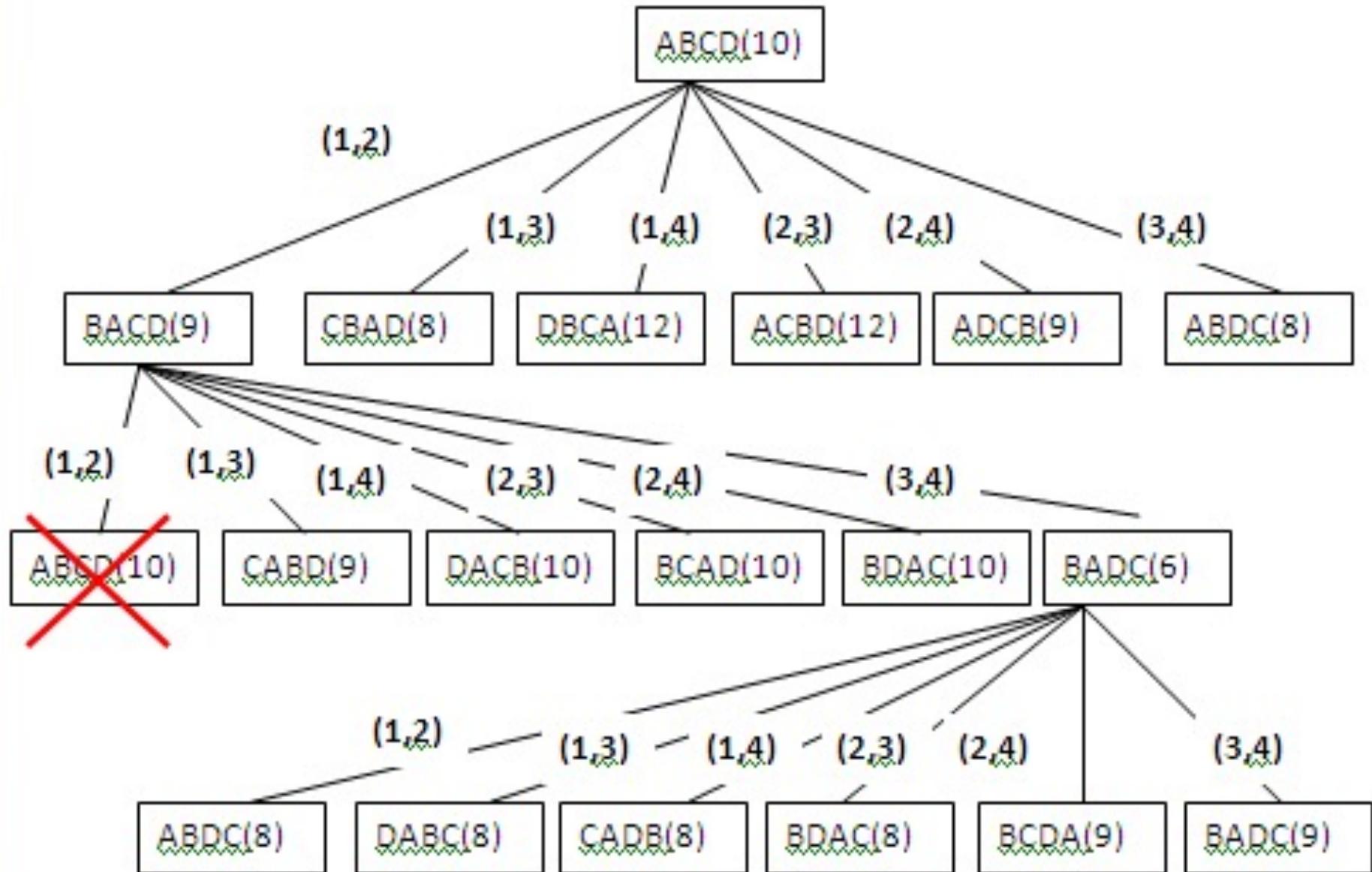
$$4!$$

$$2! (4-2)!$$

Yaitu :

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

Penggunaan pengurutan operator harus konsisten, tidak boleh berbeda tiap levelnya. urutan penggunaan operator juga sangat menentukan kecepatan dalam menemukan solusi.



Pencarian *simple hill climbing* dimulai dari anak kiri. Apabila nilai heuristik anak kiri lebih baik maka dibuka untuk pencarian selanjutnya. Jika tidak maka akan dilihat tetangga dari anak kiri tersebut, dan seterusnya.

Level 1 : (ABCD=10 > BACD =9) buka node BACD tanpa harus mencek node yang selevel dengan BACD.

Level 2 : node ABCD dilewati.

(BACD=9 = CABD=9) periksa node tetangga CABD
(BACD=9 < DABC=10) periksa node tetangga DABC
(BACD=9 < BCAD=10) periksa node tetangga BCAD
(BACD=9 < BDAC=10) periksa node tetangga BDAC
(BACD=9 > BADC=6) buka node BADC

Level 3 : (BADC=6 < ABDC=8) periksa tetangga ABDC

(BADC=6 < DABC=8) periksa tetangga DABC
(BADC=6 < CADB=8) periksa tetangga CADB
(BADC=6 < BDAC=8) periksa tetangga BDAC
(BADC=6 < BCDA=9) periksa tetangga BCDA
(BADC=6 < BADC=9) selesai.

Best-First-Search

Teknik Best-First-Search adalah teknik search yang menggabungkan kebaikan yang ada dari teknik Depth-First-Search dan Breadth-First-Search.

Tujuan menggabungkan dua teknik search ini adalah untuk menelusuri satu jalur saja pada satu saat, tapi dapat berpindah ketika jalur lain terlihat lebih menjanjikan dari jalur yang sedang ditelusuri. Untuk mendapatkan jalur yang menjanjikan adalah dengan memberikan skala prioritas pada setiap statua saat dihasilkan dengan fungsi heuristic.

Untuk menggunakan Best-First-Search, kita memerlukan dua daftar simpul, yaitu :

1. OPEN

berisi simpul yang dihasilkan dari fungsi heuristic tapi belum dievaluasi, memiliki antrian prioritas dimana elemen dengan prioritas tertinggi adalah yang memiliki nilai paling baik yang dihasilkan fungsi heuristic.

2. CLOSED

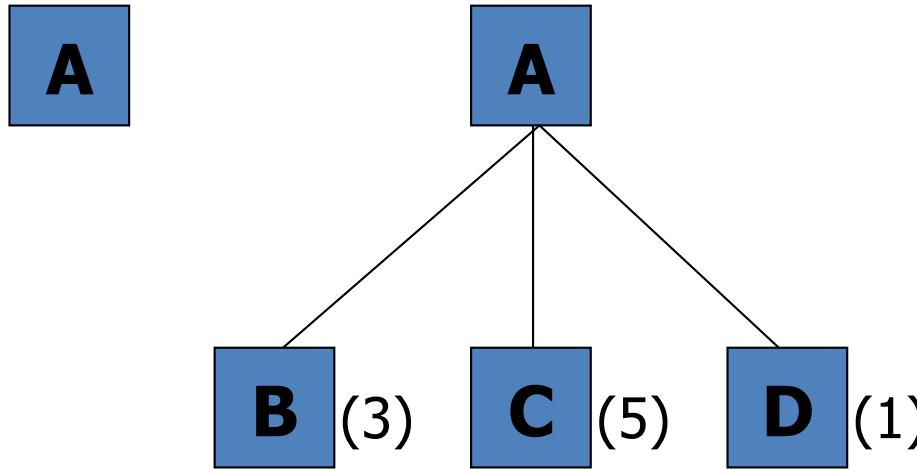
berisi simpul yang sudah dievaluasi. Kita perlu tetap menyimpan simpul-simpul ini dalam memori jika kita ingin melakukan search pada Graph, sehingga jika kita menemui suatu simpul kita bisa memeriksa apakah simpul ini sudah pernah dievaluasi atau belum

Algoritma Best-First-Search :

1. Mulai dengan OPEN hanya berisi initial state
2. Sampai goal ditemukan atau tidak ada lagi simpul yang tersisa dalam OPEN, lakukan :
 - a. Pilih simpul terbaik dalam OPEN
 - b. Telusuri successor-nya
 - c. Untuk tiap successor, lakukan :
 - i. Jika belum pernah ditelusuri sebelumnya, evaluasi simpul ini, tambahkan kedalam OPEN dan catat parentnya.
 - ii. Jika sudah pernah ditelusuri, ganti parent nya jika jalur baru lebih baik dari sebelumnya.

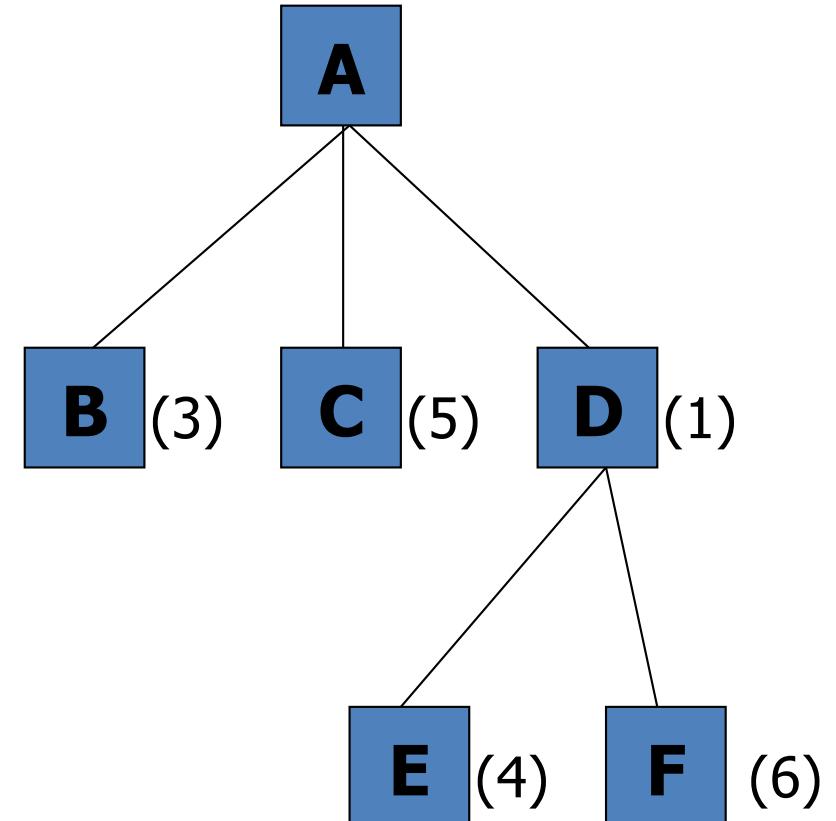
Contoh Best First Search

Step 1

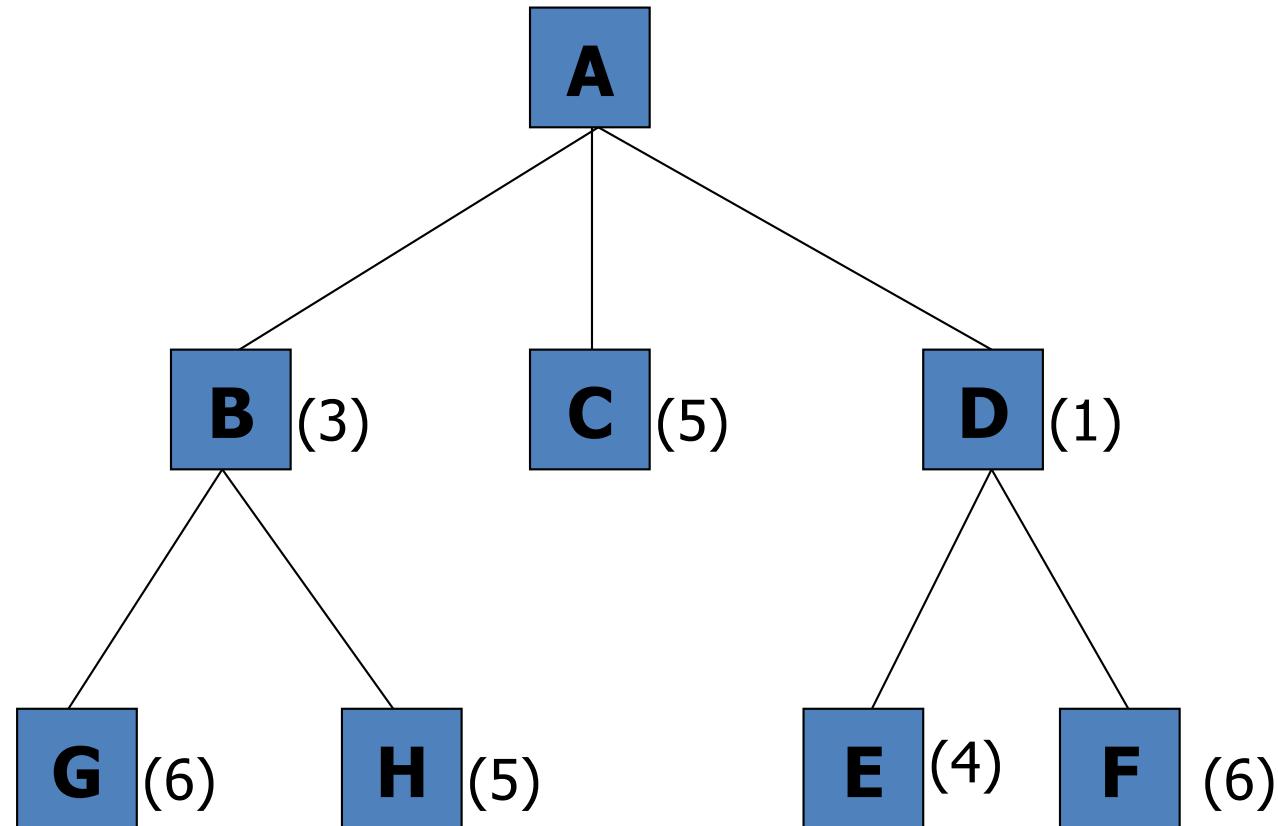


Step 2

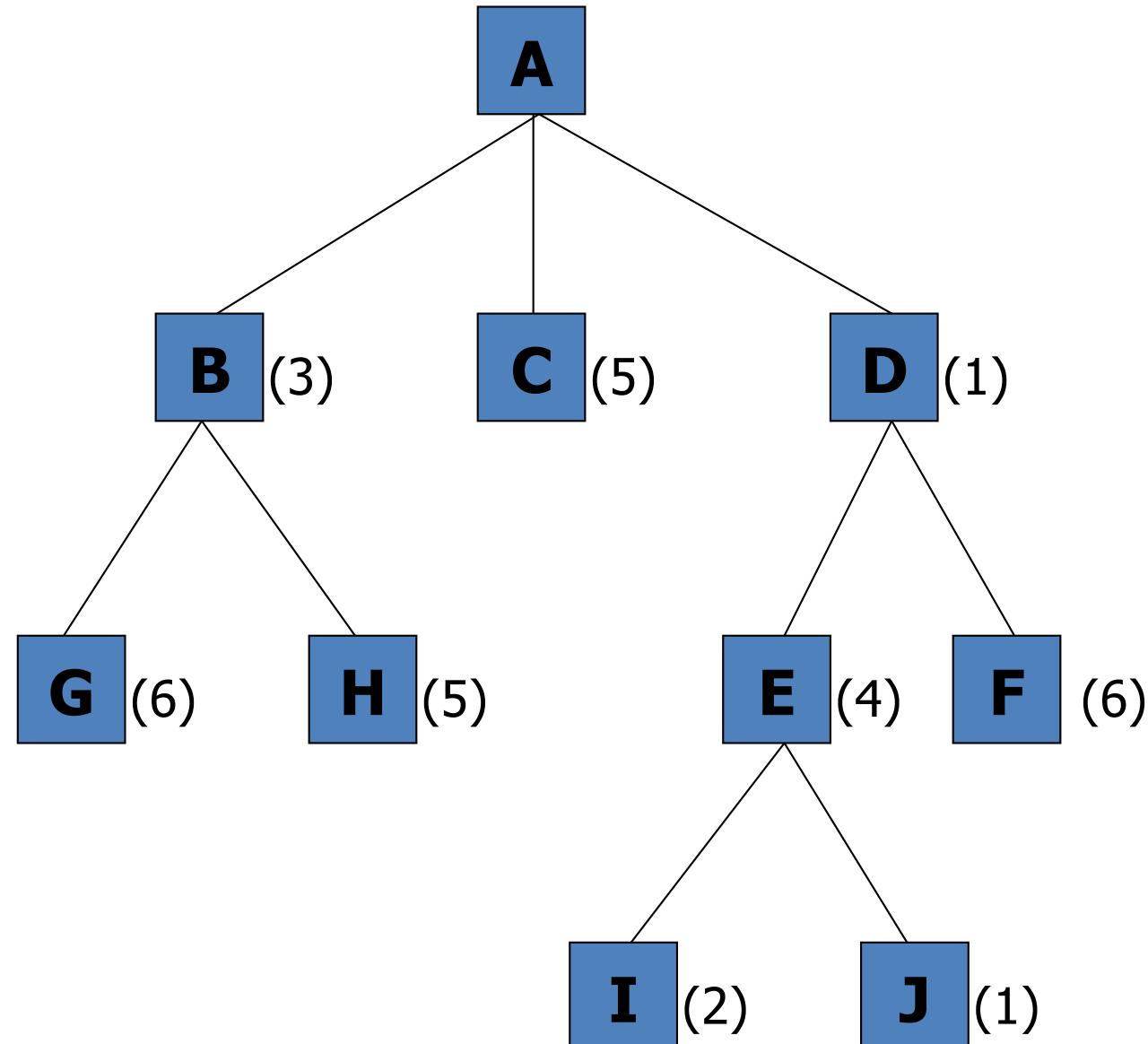
Step 3



Step 4



Step 5



Contoh lain *Best-First-Search*

Diketahui sebuah puzzle berukuran 3X3 yang berisi angka. Permasalahan adalah angka-angka dalam puzzle tersebut belum teratur.

Nilai awal puzzle :

1	2	
4	5	3
7	8	6

Goal :

1	2	3
4	5	6
7	8	

Nilai awal = {1,2,blank,4,5,3,7,8,6} Goal = {1,2,3,4,5,6,7,8,blank}

Nilai heuristic = $f(n) = g(n) + h(n)$

$g(n)$ = kedalaman dari pohon

$h(n)$ = jumlah angka yang masih salah posisi

Level 1

{1, 2, 3, 4, 5, blank, 7, 8, 6}

$$f(n) = 1+2 = 3$$

1	2	3
4	5	
7	8	6

1	2	
4	5	3
7	8	6

$$f(n) = h(n) + g(n)$$

{1, 2, blank, 4, 5, 3, 7, 8, 6}

$$f(n) = 0 + 3$$

{1, blank, 2, 4, 5, 3, 7, 8, 6}

$$f(n) = 1+4 = 5$$

1		2
4	5	3
7	8	6

Level 2

1	2	3
4	5	6
7	8	

1	2	3
4		5
7	8	6

1	2	
4	5	3
7	8	6

{1, 2, 3, 4, 5, 6, 7, 8, blank}

$$f(n) = 2+0 = 2$$

{1, 2, 3, 4, blank, 5, 7, 8, 6}

$$f(n) = 2+3 = 5$$

{1, 2, blank, 4, 5, 3, 7, 8, 6}

$$f(n) = 2+3=5$$

Latihan soal di rumah

Sebuah puzzle berukuran 3X3

Nilai awal :

2	8	3
1	6	4
7		5

Goal :

1	2	3
8		4
7	6	5

$$f(n) = g(n) + h(n)$$

$g(n)$ = kedalaman pohon

$h(n)$ = jumlah angka yang salah posisi



THE END

PERTEMUAN 6

LOGIC &
LEARNING METHOD

Logika dalam AI

Logika dalam AI digunakan sebagai suatu cara untuk menyampaikan fakta. Penyajian logika secara formal diperlukan karena akan menjadi suatu cara yang sangat disarankan untuk menurunkan/menjabarkan pengetahuan baru. Dengan logika formal kita dapat menyimpulkan bahwa suatu pernyataan baru adalah benar dengan membuktikan bahwa pernyataan itu diturunkan dari pernyataan-pernyataan lain yang sudah diketahui kebenarannya.

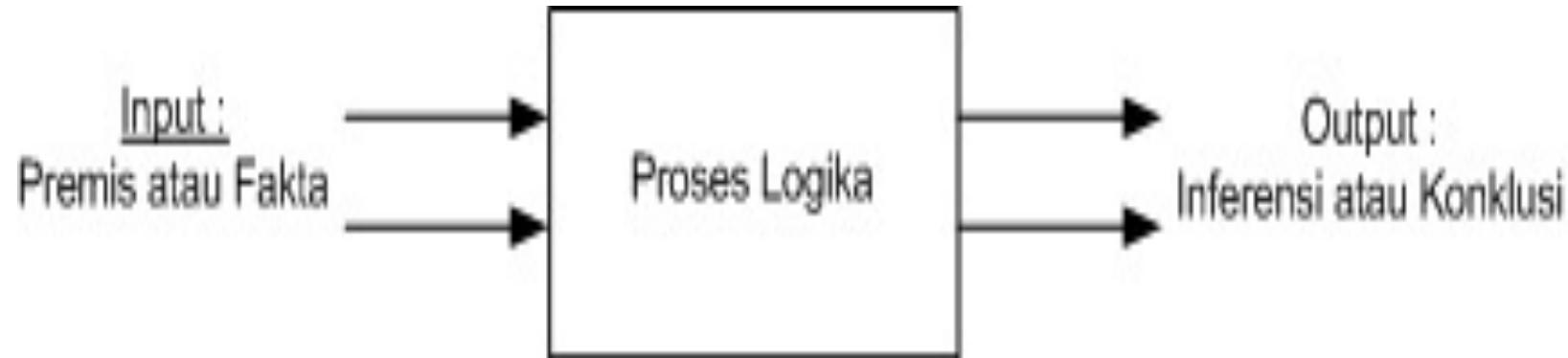
Contoh :

Jika : Matahari terbit dari Timur (benar)

Maka : Tidak mungkin matahari terbit dari Barat (benar)

Sejarah Singkat Logika

- Ahli logika pertama yang dikenal : **Aristotle** (384-322 BC), filsuf dan ahli ilmu alam Yunani. Aristotle telah mengembangkan banyak teori yang dikenal dengan *syllogistic* atau *classical logic*. *Syllogistic* pada dasarnya bertransaksi dengan penurunan kebenaran (atau yang bersifat salah) dari argumen seorang filsuf.
- *Symbolic logic* dimulai dengan **G.W. Leibniz** (1646-1717), tetapi dilupakan setelah ia meninggal, kemudian seluruh hal-hal tersebut dicakup kembali oleh : **George Boole** (1815-1864) dan logikanya dikenal dengan *Boolean Logic*. *Symbolic Logic* berinteraksi dengan konsep abstraksi ke dalam simbol-simbol dan interkoneksi simbol-simbol oleh operator tertentu.



□ 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.

Propotional Logic

Propotional logic digunakan sebagai cara menyajikan pengetahuan singkat/sederhana yang diperlukan dlm AI.

Dengan propotional logic kita akan dengan mudah menyajikan fakta dunia nyata sebagai proposisi logika yang disebut ***well-formed formulas (wff)***.

Simbol	Arti
\neg	not (negasi)
\wedge	And (konjungsi)
\vee	Or (disjungsi)
\rightarrow	Jika (implikasi)
\leftrightarrow	Jika dan hanya jika
:	Assigment(Equivalent)
\forall	for all (semua)

Contoh penyajian propotional logic :

it is raining : RAINING

it is sunny : SUNNY

if it is raining, then it is not sunny

RAINING → \neg SUNNY

Socrates is a man : SOCRATESMAN

Plato is a man : PLATOMAN

ditulis dengan cara yg lebih simple :

MAN(SOCRATES)

MAN(PLATO)

Proposisi : P dan Q yang direpresentasikan sebagai ekspresi logika dengan menggunakan logical connectives dalam suatu **tabel kebenaran**, berikut ini :

P	Q	$P \wedge Q$	$\neg P$	$\neg P \vee Q$	$P \rightarrow Q$	$P \leftrightarrow Q$	$(\neg P \vee Q) \therefore (P \rightarrow Q)$
T	T	T	F	T	T	T	T
T	F	F	F	F	F	F	T
F	T	F	T	T	T	F	T
F	F	F	T	T	T	T	T

Well Formed Formula (wff)

Berdasarkan tabel kebenaran di atas kita dapat membentuk wff berdasarkan pada aturan :

- Jika P adalah sebuah wff maka not $\neg P$ juga suatu wff
- Jika P dan Q adalah dua wff, maka berikut ini juga wff :

$$\begin{aligned} & \neg P \\ & P \wedge Q \\ & P \vee Q \\ & P \rightarrow Q \\ & P \leftrightarrow Q \end{aligned}$$

Contoh pembentukan wff

Jika diberikan 2 pernyataan bernilai benar

P : Hari Hujan

Q : Jalanan basah

Maka wff yang dapat dibentuk (benar) al:

Hari tidak hujan atau jalanan basah

Hari hujan dan jalanan basah

Jika hujan maka jalanan basah

Jika tidak hujan maka jalanan basah

Jalanan basah jika dan hanya jika hujan

Dan pernyataan berikut adalah salah :

Jika hujan maka jalanan tidak basah

Hujan dan jalanan tidak basah



Learning Method

Definisi

Learning Machine adalah suatu aplikasi dalam AI yang memiliki kemampuan beradaptasi dengan dunia luar dan dapat memanfaatkan informasi dari dunia luar untuk menambah pengetahuan dan meningkatkan kemampuannya.

Kata mesin digunakan untuk membedakan dengan manusia (mahluk hidup) yang secara alami memiliki kemampuan belajar.

Rote Learning

Metode learning ini menggunakan hasil penelusuran atau hasil perhitungan sebelumnya yang tersimpan dalam cache memori komputer untuk menentukan strategi ke langkah berikutnya.

Metode ini memiliki kemampuan untuk :

1. Mengorganisir penyimpanan informasi
adalah lebih cepat mengambil nilai yang sudah tersimpan daripada menghitung ulang
2. Generalisasi
hal ini akan mencegah terlalu besarnya informasi atau nilai yang disimpan

Learning by Taking Advice

Metode learning ini menggunakan advice tingkat tinggi (dalam bahasa manusia) untuk menghasilkan suatu aturan operasional.

Advice mana yang akan digunakan dari sekian banyak yang ada diproses/dipilih menggunakan operator-operator seperti : analisis kasus, pencocokan, dsb

Learning from example

Metode ini menggunakan semua contoh dari kasus-kasus yang pernah diselesaikan atau data contoh yang dimasukkan ke sistem.

Hal terpenting dari metode ini klasifikasi, untuk memilah atau mengklasifikasi contoh menjadi contoh positif dan contoh negatif.

Hasil dari metode ini adalah suatu deskripsi konsep.

Metode ini menggunakan Algoritma search untuk mengeliminasi contoh dan menghasilkan pohon keputusan

Learning in Problem Solving

Metode ini berusaha untuk memperbaiki pemecahan masalah dari pemecahan masalah yang sudah ada atau sudah pernah diaplikasikan.

Metode ini menggunakan solusi dari contoh masalah sebagai masukan dan akan menghasilkan penemuan cara baru untuk menyelesaikan masalah secara lebih efisien.

Metode ini menggunakan heuristic search seperti : generalisasi, learning berdasarkan penjelasan dan pertimbangan yang menyeluruh.

Discovery

Metode ini berusaha untuk menemukan pengetahuan-pengetahuan baru yang belum terungkap sebelumnya.

Metode ini menggunakan heuristic search yang berdasarkan kepada analogy, ketertarikan (minat) atau bahkan suatu misteri.

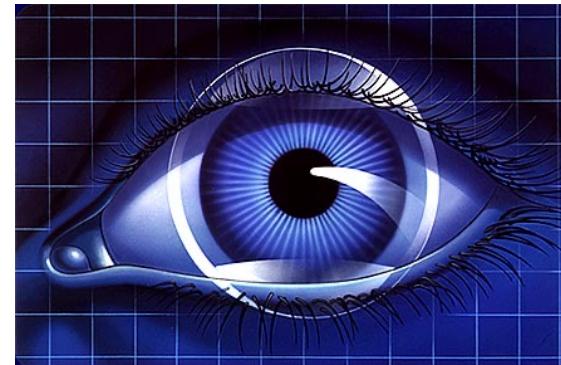
Hasil atau keluaran dari metode ini cenderung tidak diketahui atau sulit diperkirakan, karena biasanya berdasarkan informasi atau pengetahuan yang minim



THE END

PERTEMUAN 9

VISION



Pengertian Vision

Vision merupakan suatu aplikasi komputer, dimana didalamnya dapat mencakup navigasi robot, tugas manufaktur yang rumit, analisis citra satelit, pemrosesan citra medis, dsb.

Vision (visi) dapat diartikan sebagai suatu cara/teknik untuk mentransfer sebuah citra menjadi informasi yang lebih berguna.

Citra (image) adalah apa yang ditangkap oleh suatu alat perekam gambar seperti kamera foto atau video.

Kamera video menyajikan komputer suatu citra yang direpresentasikan sebagai butiran (titik-titik) dua dimensi dengan tingkat intensitas (keabuan) yang berbeda-beda yang disebut **pixel**. Tiap pixel dapat memiliki informasi tunggal (seperti hitam/putih) atau banyak informasi (seperti nilai sebenarnya dari intensitas dan informasi warna).

Sebuah citra visual dapat terdiri dari ribuan pixel.

Beberapa hal yang dapat dilakukan terhadap citra (Tugas pengolahan citra)

1. *Signal Processing* (pemrosesan sinyal)

Mempertajam citra, baik untuk penglihatan manusia maupun sebagai input untuk proses yang lainnya.

2. *Measurement Analysis* (analisis ukuran)

Untuk citra yang berisi object tunggal, menentukan perluasan dua dimensi dari object yang digambarkan.

3. *Pattern Recognition* (pengenalan pola)

Untuk citra berisi object tunggal, mengklasifikasi object kedalam kategori dari beberapa kemungkinan yang ada.

4. *Image Understanding* (memahami citra)

Untuk citra yang berisi banyak object, mendapatkan lokasi object dalam citra, mengklasifikasikannya, dan membentuk gambar model tiga dimensi.

Dari semua tugas pengolahan citra, pemahaman citra adalah yang paling sulit, sehingga menjadi pembahasan pada banyak studi tentang AI. Beberapa masalah yang dihadapi dalam pengolahan citra adalah :

1. Suatu citra adalah dua dimensi (2-D), sementara semua object didunia ini adalah tiga dimensi (3-D), sehingga sebagian informasi akan hilang pada saat kita merekam object (3-D) ke dalam citra (2-D).
2. Suatu citra mungkin berisi beberapa object, dan tiap object dapat terdiri dari beberapa bagian lagi.
3. Nilai dari tiap pixel dipengaruhi oleh banyak fenomena yang berbeda, seperti warna object, sumber cahaya, sudut pengambilan gambar, jarak kamera dari object, polusi udara dsb. Dan sangat sulit untuk menghindari pengaruh tsb terhadap object.

Pada dasarnya, citra 2-D sangat membingungkan karena dari suatu citra 2-D kita dapat membentuk beberapa gambaran 3-D yang akan meningkatkan citra tsb.

Interpretasi Citra Tingkat Rendah

Pada interpretasi citra tingkat rendah digunakan beberapa hal

1. Pengetahuan komponen citra tingkat rendah seperti bayangan, texture, warna dan pantulan cahaya, untuk menentukan gambaran interpretasi yang paling mendekati dari suatu citra.
2. Menggunakan lebih dari satu citra untuk object yang sama yang akan berguna dalam menyusun struktur 3-D.
3. Stereo Vision, yaitu penggunaan dua atau lebih kamera untuk menghasilkan banyak sudut pandang yang simultan terhadap suatu object
4. Pengetahuan tentang bagaimana membuat efek bergerak pada suatu citra juga akan membantu interpretasi citra.

Interpretasi Citra Tingkat Tinggi

Dilakukan dengan Ekspektasi Top-Down, yang terdiri dari langkah-langkah :

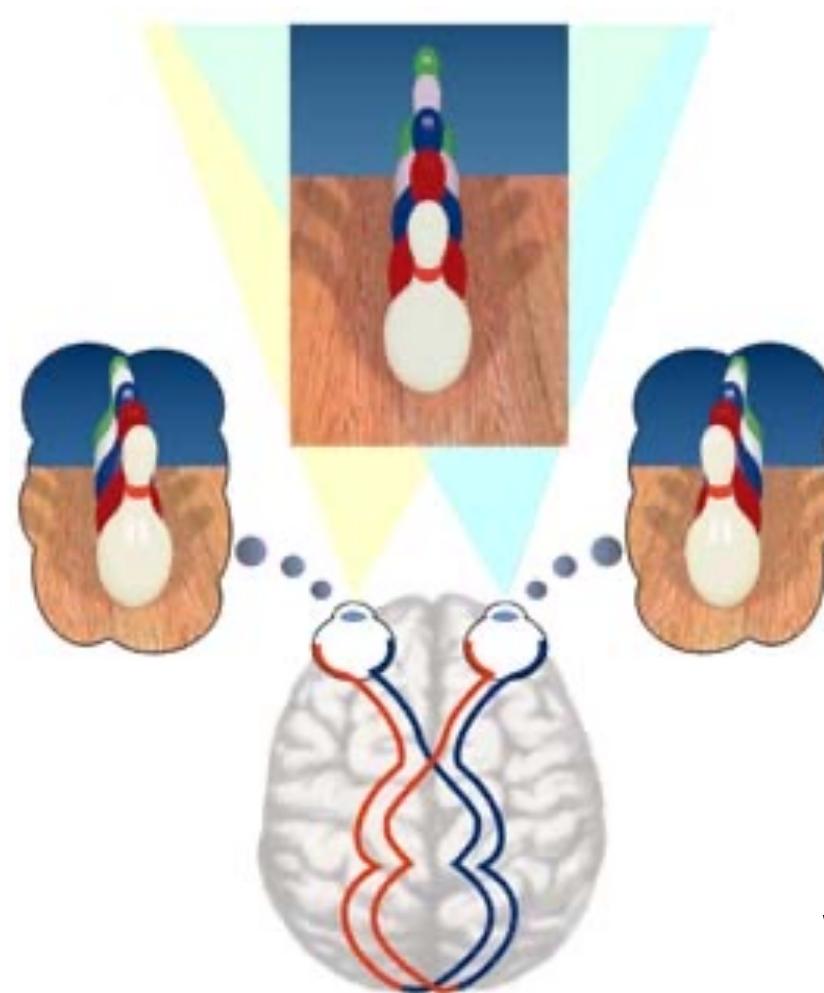
1. Mengkonversi sinyal analog video menjadi citra digital,
proses ini menghasilkan citra yang terdiri dari pixel-pixel
2. Deteksi tepian dan batas object

tepian dapat dideteksi dengan algoritma yang melihat sekumpulan pixel yang bernilai sama, batas dapat ditemukan dengan mengelompokkan pixel-pixel yang sejenis, proses ini akan menghasilkan garis-garis 2-D

3. Orientasi 3-D
akan menghasilkan bentuk-bentuk 2-D dari garis-garis 2-D
4. Pengelompokan permukaan
akan menghasilkan bentuk-bentuk solid 3-D terpisah

5. Pengelompokan bentuk
akan menghasilkan object 3-D yang utuh
6. Pencocokan
mencocokkan object 3-D yang didapat dengan basis
pengetahuan untuk mendapatkan interpretasi yang paling
mendekati

Contoh Stereo Vision pada mata manusia



www.vision3d.com



THE END

PERTEMUAN 10

**NATURAL LANGUAGE PROCESSING
(PEMROSES BAHASA ALAMI)**

Definisi

Pemroses bahasa alami (Natural Language Processing/NLP) adalah suatu aplikasi (program) dalam bidang AI yang dapat mengartikan suatu bahasa baik bahasa tulisan maupun bahasa lisian atau memproses masukan yang berupa bahasa menjadi suatu informasi atau pengetahuan

Yang menjadi pembahasan bukan bagaimana bahasa diinput atau dimasukkan kedalam program, tetapi lebih kepada bagaimana mengartikan suatu bahasa atau mengcopy / mengambil informasi/pengetahuan dari suatu bahasa.

Pembagian NLP

Masalah pemrosesan bahasa alami dibagi menjadi dua bagian besar, yaitu :

1. Pemrosesan Naskah Tertulis

menggunakan pengetahuan tentang leksikal, sintax, dan semantik

2. Pemrosesan Bahasa Lisan

menggunakan semua pengetahuan dari pemrosesan naskah tertulis ditambah pengetahuan tentang phonology.

Masalah dalam NLP

Beberapa masalah yang dihadapi dalam pemrosesan bahasa alami antara lain adalah :

1. Suatu kalimat sering kali tidak lengkap, artinya tidak memberi informasi yang jelas atau lengkap
2. Satu kalimat dapat memiliki lebih dari satu pengertian dalam konteks yang berbeda
3. Tidak ada program pemroses bahasa alami yang cukup lengkap karena bahasa selalu berkembang, kosa kata selalu bertambah.
4. Bisa terdapat lebih dari satu cara (lebih dari satu kalimat) untuk mengungkapkan hal(maksud) yang sama.

Tahapan Proses

Untuk memproses bahasa alami diperlukan 5 langkah sebagai berikut :

1. Analisis Morphology

Pada tahap ini dilakukan analisa untuk setiap kata dan komponen yang dimiliki tiap kata termasuk token non kata seperti spasi, tanda baca, tanda pemisah.

2. Analisis Sintax

Pada tahap ini sederetan kata disusun kedalam struktur yang memperlihatkan bagaimana hubungan satu kata dengan kata lainnya. Deretan kata akan ditolak bila tidak memenuhi aturan penyusunan kata yang ada

3. Analisis semantik

Pada tahap ini struktur deretan kata yang sudah terbentuk akan diberi arti. Dengan kata lain pemetaan dibuat antara struktur sintax dengan object yang berhubungan.

- **Penyatuan Arah (konteks)**

Pada tahap ini arti dari suatu kalimat disesuaikan dengan kalimat-kalimat lain, karena arti dari suatu kalimat biasanya berhubungan dengan kalimat sebelumnya dan kalimat sesudahnya.

5. Analisis Pragmatis

Struktur yang terbentuk menghasilkan interpretasi ulang dari apa yang sudah dikatakan atau ditulis sebelumnya dengan arti yang sebenarnya.

Grammar dan Parsers

- Grammar adalah suatu aturan yang menentukan bagaimana suatu kalimat dalam suatu bahasa dibentuk. Grammar berisi kumpulan sintax yang baku/benar dari suatu bahasa.

Contoh : Dalam bahasa Indonesia, suatu kalimat biasanya terdiri dari

Subject-Predikat-Object-Keterangan

- Parsers adalah suatu metode atau suatu program (sering disebut suatu mesin) yang dapat memproduksi/menghasilkan kalimat atau bahasa yang sesuai dengan Grammar yang sudah ditentukan atau diinginkan. Parsers juga dapat memeriksa apakah suatu kalimat yang dimasukkan sesuai dengan Grammar atau tidak.

Jenis Parsers

Parsers terdiri dari dua jenis, yaitu :

1. Top-Down Parsing

memulai proses parsing dari simbol start dan menggunakan aturan grammar sampai simbol-simbol terminal pada tree terhubung ke komponen kalimat yang di parsing

2. Bottom-Up Parsing

memulai proses parsing dari kalimat yang akan di parsing dan menggunakan aturan grammar secara terbalik untuk memproduksi kata menjadi terminal, terminal menjadi kalimat sampai tree/ pohon lengkap dan simbol start tercapai



THE END

PERTEMUAN 11

EXPERT SYSTEM

(SISTEM PAKAR)

Definisi Sistem Pakar

Sistem Pakar adalah suatu sistem yang menggabungkan pengetahuan dan penelusuran data untuk memecahkan masalah yang secara normal memerlukan keahlian seorang pakar.

Yang diperlukan untuk membangun sistem pakar adalah sejumlah pengetahuan dan suatu mekanisme untuk mengakses pengetahuan itu secara efisien (mekanisme inferensi) untuk memecahkan masalah.

Manfaat Sistem Pakar

Ada banyak manfaat yang dapat diperoleh dengan mengembangkan sistem pakar, antara lain :

1. Masyarakat awam non-pakar dapat memanfaatkan keahlian di dalam bidang tertentu tanpa kehadiran langsung seorang pakar
2. Meningkatkan produktivitas kerja, menambah efisiensi kerja dan hasil solusi kerja
3. Penghematan waktu dalam menyelesaikan masalah yg kompleks
4. Memberikan penyederhanaan solusi untuk kasus-kasus yang kompleks dan berulang-ulang
5. Pengetahuan dari seorang pakar dapat didokumentasikan tanpa ada batas waktu
6. Memungkinkan penggabungan berbagai bidang pengetahuan dari berbagai pakar untuk dikombinasikan

Perbandingan Pakar Manusia dan Sistem Pakar

Pakar Manusia	Sistem Pakar
Terbatas waktu karena manusia membutuhkan istirahat	Tidak terbatas waktu karena dapat digunakan kapanpun juga
Tempat akses bersifat lokal pada suatu tempat saja dimana pakar berada	Dapat digunakan diberbagai tempat
Pengetahuan bersifat variabel dan dapat berubah-ubah tergantung situasi	Pengetahuan bersifat konsisten
Kecepatan untuk menemukan solusi sifatnya bervariasi	Kecepatan untuk memberikan solusi konsisten dan lebih cepat daripada manusia
Biaya yang harus dibayar untuk konsultasi biasanya mahal	Biaya yang dikeluarkan lebih murah

Kelemahan Pengembangan Sistem Pakar

Beberapa kelemahan Sistem pakar diantaranya :

- Daya kerja dan produktifitas manusia menjadi berkurang karena semuanya dilakukan secara otomatis oleh sistem
- Pengembangan perangkat lunak sistem pakar lebih sulit dibandingkan perangkat lunak konvensional.

Perbandingan Perangkat Lunak Konvensional dengan Perangkat Lunak Sistem Pakar

Perangkat Lunak Konvensional	Perangkat Lunak Sistem Pakar
Fokus Pada Solusi	Fokus Pada Permasalahan
Pengembangan dapat dilakukan secara individu	Pengembangan dilakukan oleh tim kerja
Pengembangan secara sekuensial	Pengembangan secara iteratif

Ciri dan Karakteristik Sistem Pakar

1. Pengetahuan sistem pakar merupakan suatu konsep, bukan berbentuk numeris.
2. Informasi dalam sistem pakar tidak selalu lengkap, subyektif, tidak konsisten, subyek terus berubah dan tergantung pada kondisi lingkungan sehingga keputusan yang diambil bersifat tidak pasti dan tidak mutlak “ya” atau “tidak” tetapi menurut ukuran kebenaran tertentu
3. Kemungkinan solusi sistem pakar terhadap suatu permasalahan adalah bervariasi dan mempunyai banyak pilihan jawaban yang dapat diterima, semua faktor yang ditelusuri memiliki ruang masalah yang luas dan tidak pasti

4. Perubahan atau pengembangan pengetahuan dalam sistem pakar dapat terjadi setiap saat bahkan sepanjang waktu sehingga diperlukan kemudahan dalam modifikasi sistem untuk menampung jumlah pengetahuan yang semakin besar dan semakin bervariasi
5. Pandangan dan pendapat setiap pakar tidaklah selalu sama, yang oleh karena itu tidak ada jaminan bahwa solusi sistem pakar merupakan jawaban yang pasti benar. Setiap pakar akan memberikan pertimbangan-pertimbangan berdasarkan faktor subyektif.
6. Keputusan merupakan bagian terpenting dari sistem pakar. Sistem pakar harus memberikan solusi yang akurat berdasarkan masukan pengetahuan meskipun solusinya sulit sehingga fasilitas informasi sistem harus selalu diperlukan.

Bidang-Bidang Pengembangan Sistem Pakar

1. Kontrol

Contoh pengembangan banyak ditemukan dalam kasus pasien di rumah sakit, dimana dengan kemampuan sistem pakar dapat dilakukan kontrol terhadap cara pengobatan dan perawatan melalui sensor data atau kode alarm dan memberikan solusi terapi pengobatan yang tepat bagi pasien

2. Desain

Contoh sistem pakar desain adalah PEACE yang dibuat oleh Dinobas pada tahun 1980 untuk membantu desain pengembangan sirkuit elektronik.

3. Diagnosis

Merupakan pengembangan sistem pakar terbear seperti diagnosis penyakit, kerusakan mesin kendaraan, kerusakan komponen komputer dsb.

4. Instruksi

Merupakan pengembangan sistem pakar yang sangat berguna dalam bidang ilmu pengetahuan dan pendidikan, dimana sistem pakar dapat memberikan instruksi dan pengajaran tertentu terhadap suatu topik permasalahan

5. Interpretasi

Pengembangan bidang ini melakukan proses pemahaman terhadap situasi dari beberapa informasi yang direkam

6. Monitor

Bidang ini banyak digunakan militer, seperti penggunaan sensor radar kemudian kemudian menganalisisnya dan menentukan posisi obyek berdasarkan posisi radar tsb

7. Perencanaan

banyak digunakan dalam bidang bisnis dan keuangan suatu proyek.

8. Prediksi

sistem pakar ini mampu memprediksi kejadian dimasa mendatang berdasarkan informasi model dan permasalahan yang dihadapi

9. Seleksi

Sistem pakar dengan seleksi mengidentifikasi pilihan terbaik dari beberapa daftar pilihan kemungkinan solusi.

10. Simulasi

Sistem ini memproses operasi dari beberapa variasi kondisi yang ada dan menampilkannya dalam bentuk simulasi. Contohnya adalah program PLANT yang sudah menggabungkan antara prediksi dan simulasi, dimana program tersebut mampu menganalisis hama dengan berbagai kondisi suhu dan cuaca



THE END

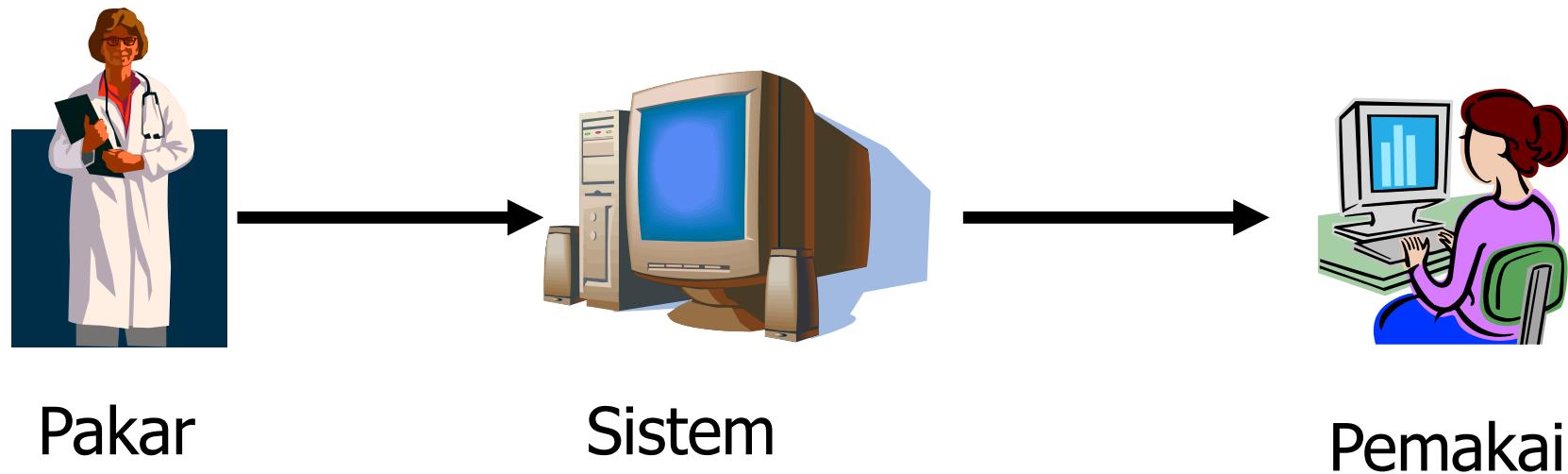
PERTEMUAN 12

EXPERT SYSTEM

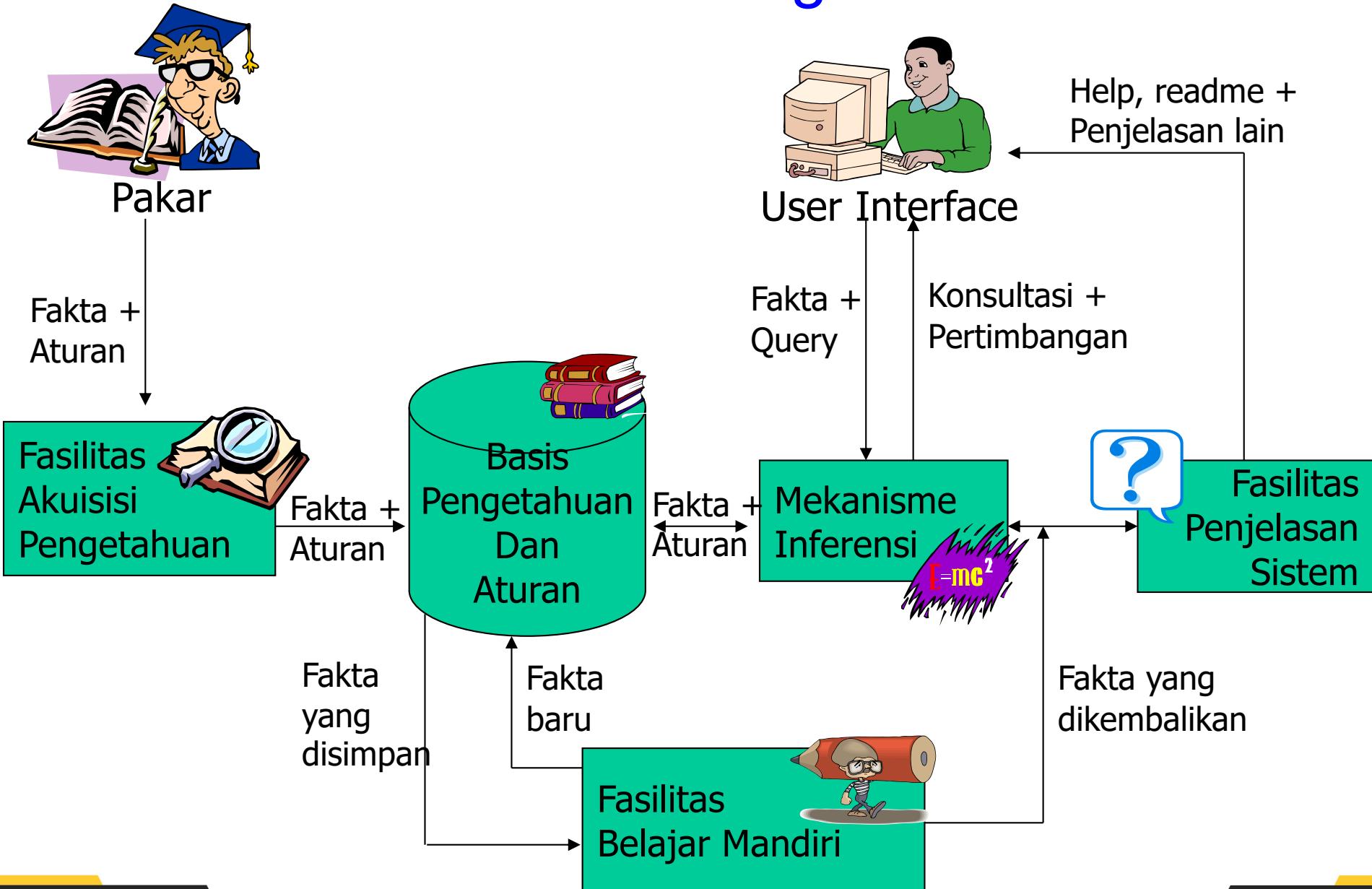
(SISTEM PAKAR, LANJUTAN)

Unsur Penting Pengembangan Sistem Pakar

Ada tiga unsur penting dalam pengembangan Sistem Pakar, yaitu :



Struktur Bagan Sistem Pakar



Komponen Sistem Pakar

Komponen Sistem Pakar terdiri dari :

1. Fasilitas Akuisisi pengetahuan
2. Basis Pengetahuan dan Basis Aturan
3. Mekanisme Inferensi
4. Fasilitas Belajar Mandiri
5. Fasilitas Penjelasan Sistem
6. Antarmuka Pemakai

1. Fasilitas Akuisisi Pengetahuan

Merupakan suatu proses untuk mengumpulkan data-data pengetahuan tentang suatu masalah dari pakar.

Bahan pengetahuan dapat diperoleh dengan berbagai cara, seperti dari buku, jurnal ilmiah, pakar di bidangnya, laporan, literatur dsb. Sumber pengetahuan tsb dijadikan dokumentasi untuk dipelajari, diolah dan diorganisir secara terstruktur menjadi basis pengetahuan.

2. Basis Pengetahuan dan Basis Aturan

Ada beberapa cara merepresentasikan data menjadi basis pengetahuan, seperti dalam bentuk atribut, aturan-aturan, jaringan semantik, frame dan logika. Semua bentuk representasi data tsb bertujuan untuk menyederhanakan data sehingga mudah dimengerti dan mengefektifkan proses pengembangan program.

Dalam pemrograman visual umumnya disediakan sarana untuk mengembangkan tabel-tabel penyimpanan data yang terangkum dalam sebuah database.

3. Mekanisme Inferensi

Adalah bagian sistem pakar yang melakukan penalaran dengan menggunakan isi daftar aturan berdasarkan urutan dan pola tertentu. Selama proses konsultasi antara sistem dan pemakai, mekanisme inferensi menguji aturan satu persatu sampai kondisi aturan itu benar.

Secara umum ada dua teknik utama yang digunakan dalam mekanisme inferensi untuk pengujian aturan, yaitu penalaran maju (*forward reasoning*) dan penalaran mundur (*reverse reasoning*)

4. Fasilitas Belajar Mandiri

Fasilitas ini memungkinkan sistem untuk mengembangkan dirinya sendiri dengan memilah atau mengelompokan kembali fakta yang sudah ada, memasukkan fakta-fakta baru kedalam basis pengetahuan yang merupakan hasil penurunan (iterasi) dari fakta-fakta sebelumnya dan dapat mengembalikan fakta ke pada mekanisme inferensi sehingga dapat dimintakan fakta lainnya dari pemakai melalui antarmuka pemakai

5. Fasilitas Penjelasan Sistem

Merupakan bagian komponen sistem pakar yang memberikan penjelasan tentang bagaimana program dijalankan, apa yang harus dijelaskan kepada pemakai tentang suatu masalah, memberikan rekomendasi kepada pemakai, mengakomodasi kesalahan pemakai dan menjelaskan bagaimana suatu masalah terjadi.

Dalam sistem pakar, fasilitas penjelasan sistem sebaiknya diintegrasikan ke dalam tabel basis pengetahuan dan basis aturan karena hal ini lebih memudahkan perancangan sistem

6. Antarmuka Pemakai

Komponen ini memberikan fasilitas komunikasi antara pemakai dan sistem, memberikan berbagai fasilitas informasi dan berbagai keterangan yang bertujuan untuk membantu mengarahkan alur penelusuran masalah sampai ditemukan solusi.

Syarat utama membangun antarmuka pemakai adalah kemudahan dalam menjalankan sistem, tampilan yang interaktif, komunikatif dan mudah bagi pemakai

Tahap Pengembangan Sistem Pakar

Sistem Pakar dikembangkan dengan mengikuti 6 tahap seperti berikut ini :

1. Identifikasi
2. Konseptualisasi
3. Formalisasi
4. Implementasi
5. Evaluasi
6. Pengembangan Sistem



THE END

PERTEMUAN 13



ROBOTIC

Definisi

Robot

Istilah ini pertama kali diperkenalkan ke dalam bahasa Inggris oleh penulis dan dramawan asal Chechnya, [Karel Capek](#), yang memiliki arti pekerja dalam aksi panggungnya yang berjul [R.U.R \(Rossum's Universal Robots\)](#) pada [tahun 1921](#).

Robotik

adalah ilmu yang mematerikan kecerdasan/intelegensia terhadap energi, artinya pengendalian secara cerdas terhadap gerakan yang terkoordinasi secara nyata.

Istilah ini diperkenalkan oleh [Isaac Asimov](#) untuk pertama kalinya dalam cerita pendeknya yang berjudul [Runaround yang terbit tahun 1942](#)

Karel capek

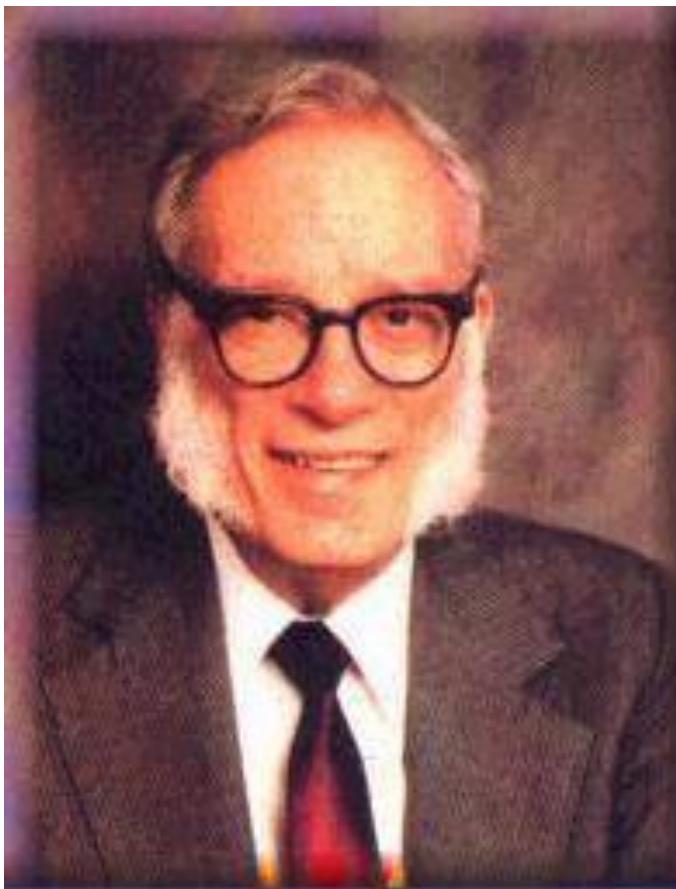
(1890 – 1938)



- Lahir di Svatonovice, Bohemia
- Anak seorang Dokter
- Mempelajari filosofi di Paris, berlin dan Praha
- Pernah bekerja sebagai wartawan di yang menentang Nazi dan menyerukan demokrasi di Czechoslovakia
- Sebagai penulis, menghasilkan drama R.U.R dan novel fantasy a.l : War with the Newts

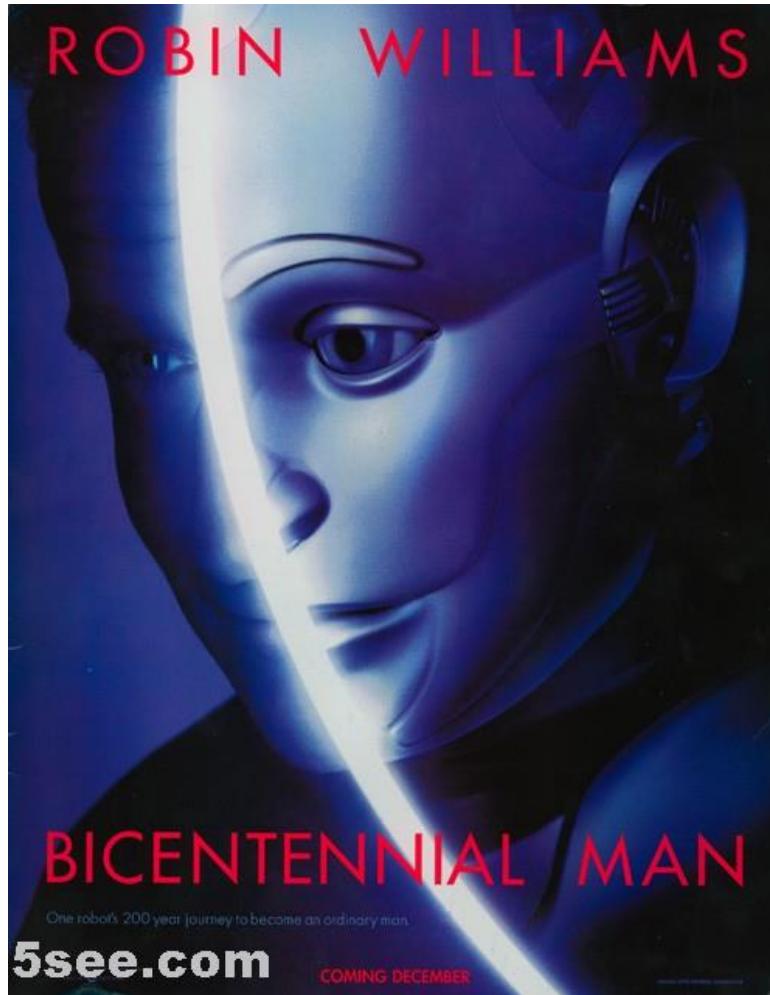
Isaac Asimov

(1920 – 1992)



- Lahir di Petrovichi, Rusia
- Pindah ke AS sejak usia 3 thn (1923)
- Mendapat gelar PhD (S3) bidang kimia dari Columbia University
- Sebagai penulis fiksi ilmiah menghasilkan 500 judul buku al : Foundation, Foundation and Empire, Second Foundation, The caves of Steel, The Naked Sun, **Bicentennial Man**, dan cerita pendek **I, Robot**.
- Menambahkan istilah "**Robotics**" dalam kamus bahasa Inggris yang diperkenalkannya dalam cerita pendek berjudul **Runaround** (1942)
- Menjadi Konsultan untuk film Star Trek : The Motion Picture (1979)

Karya Asimov yang sudah difilmkan



Filosofi Pembuatan Robot

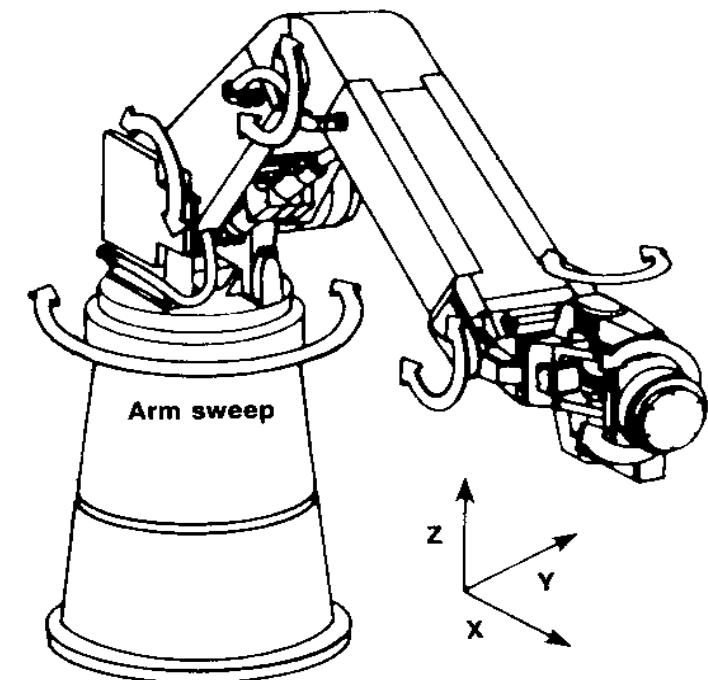
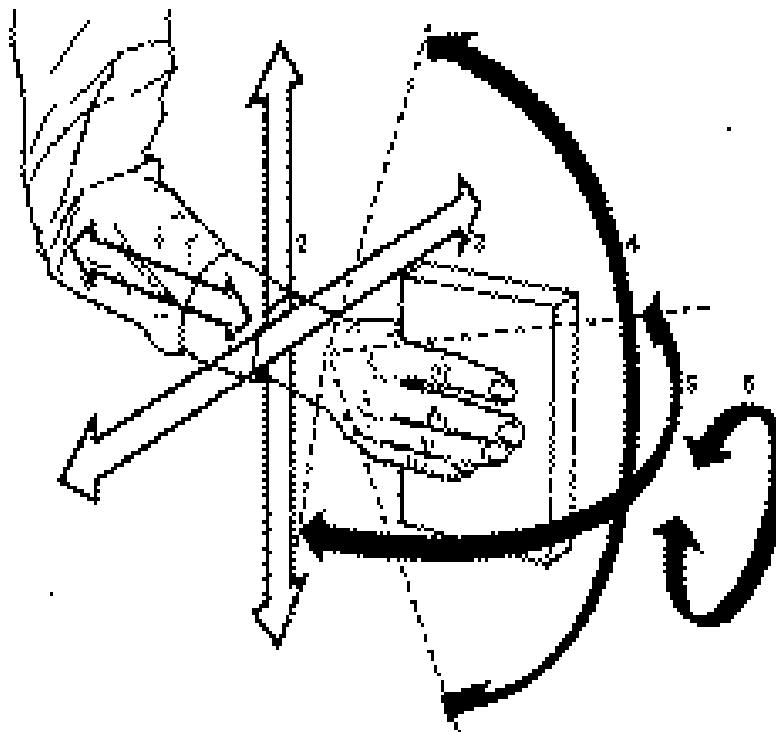
Isaac Asimov dalam cerita pendeknya yang berjudul I, Robot mencetuskan suatu filosofi agar pembuatan robot sejak awal dan sampai masa yang akan datang diharapkan dapat memenuhi tiga aturan yang dikenal sebagai Asimov's Tree Laws of Robotic :

1. Robot tidak boleh menyakiti manusia
2. Robot harus mematuhi manusia selama tidak bertentangan dengan hukum 1
3. Robot harus dapat melindungi dirinya selama tidak bertentangan dengan hukum 1 dan 2

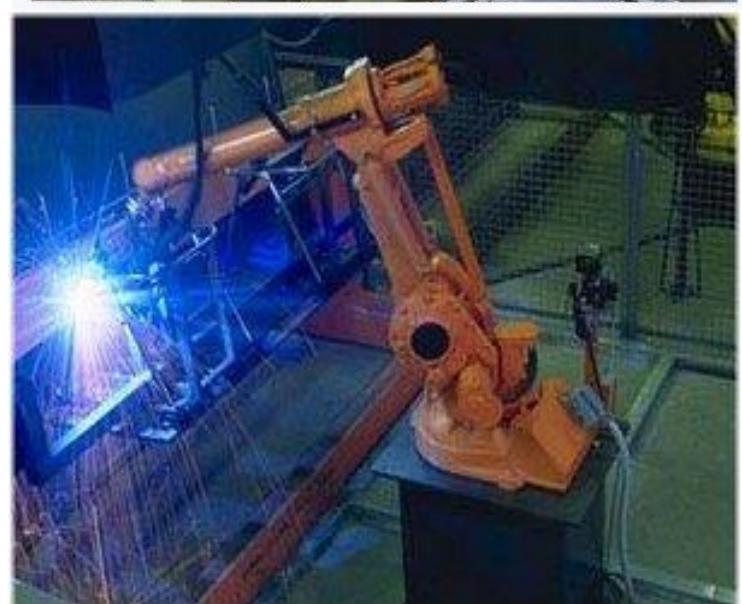
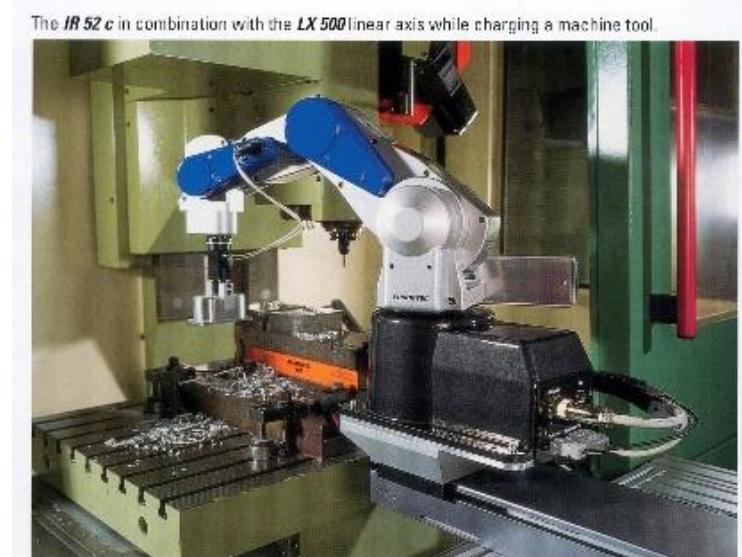
Tipe Robot

1. Robot Industri
 bentuknya sudah fix dan
membutuhkan tempat yang tetap
2. Robot Autonomous / Personal
 bentuknya lebih fleksibel dan tidak
membutuhkan tempat yang tetap
(mandiri)

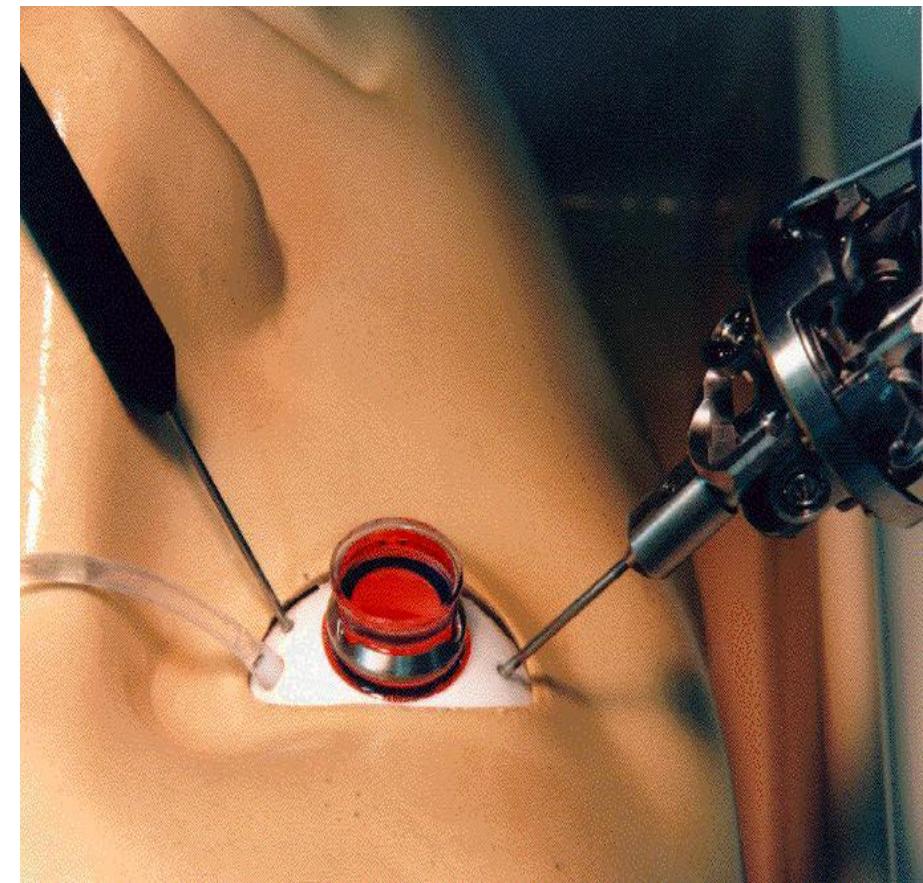
Kaidah Robot Industri



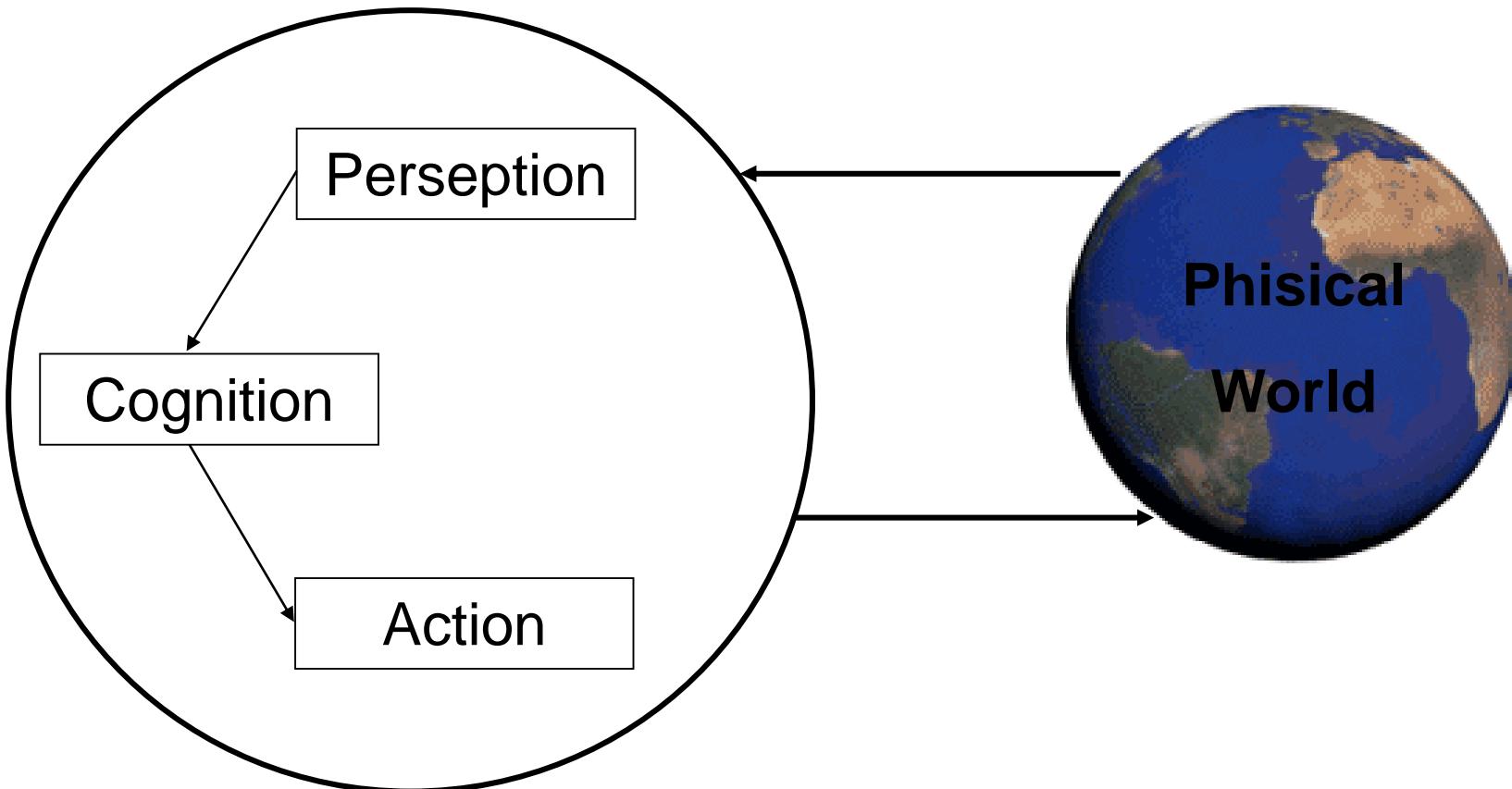
Contoh Robot Industri



Tangan Robot Untuk Operasi (Pembedahan)



Kaidah Personal Robot



Sebuah Rancangan Untuk Sebuah Autonomous Robot

Klasifikasi Robot Personal (1)

(Exhibited at ROBODEX2003)

	classification	exclusive	general	usage, features	examples
1	all-purpose utility		O	use tools and life environment for people	ASHIMO, HRP-2
2	3K Robot 3K = dangerous, heavy, filthy		O	search mine, rescue, nuclear reactor, space work, human security safe robot, limit work robot	MARS- I
3	serious utility (immediate safety)		O	nursing, guarding	C4
4	non-serious utility (non-immediate safety)		O	receptionist, help, care-taker, cleaning	BANRYU T73S, FLATTHRU, HOSPI, ASKA, U-chan, Care power assist suit, Intelligent white cane

Klasifikasi Robot Personal (2)

(Exhibited at ROBODEX2003)

	classification	exclusive	general	usage, features	examples
5	home robot		<input checked="" type="radio"/>		MARON-1, Apri Alpha
6	communication robot	<input checked="" type="radio"/>	<input checked="" type="radio"/>	communicate with people (with language / without language)	Wakamaru, ifbot, ROBOVIE, Kismet, Leonardo, WE-4R, WAMO E B A - 2 R I, SAKURA
7	entertainment robot		<input checked="" type="radio"/>	pleasure, life-partner, characterized robot	Q R I O , A I B O , Monsieur II - P , DORAEMON THE ROBOT
8	invisible robot	<input checked="" type="radio"/>	<input checked="" type="radio"/>	room area or environment itself become robot	
9	toy		<input checked="" type="radio"/>		CAM-09, Dr. Robot

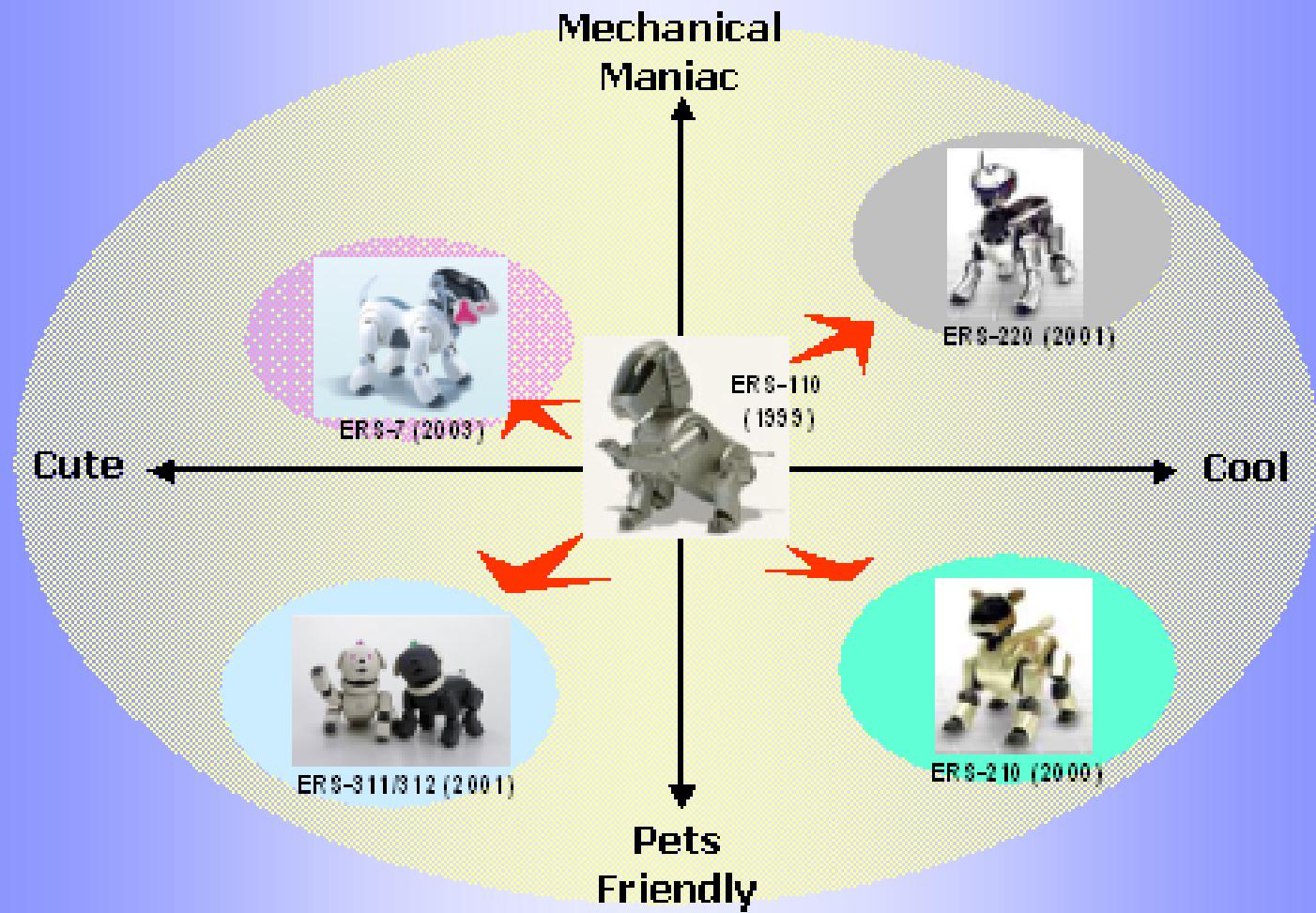
AIBO (Artificial Intelligence Robot)



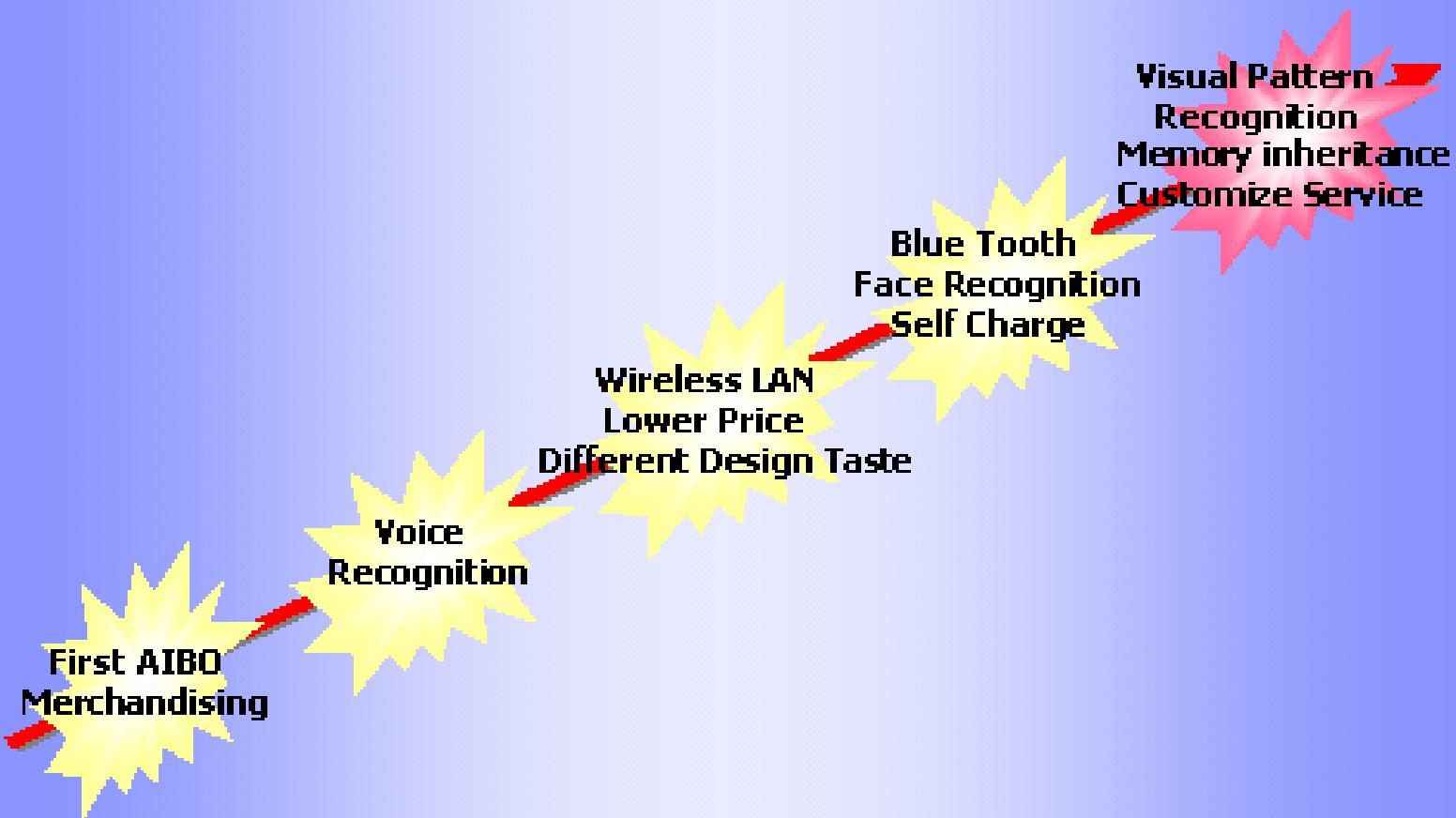
Robot berbentuk anjing buatan Sony, Jepang, yang dapat berprilaku dan berinteraksi seperti layaknya seekor anjing hidup.

Dipasarkan pertama kali Juni 1999 via internet dan terjual 3000 unit dalam 20 menit di Jepang.

Pengembangan AIBO



Evolusi Features AIBO



1999

2000

2001

2002

2003

Varian AIBO 2005 ERS-7M2 dan ERS-7M3



ASIMO

(Advanced Step in Innovative Mobility)



Robot berbentuk manusia buatan Honda yang memiliki keseimbangan yang sangat baik, sehingga mampu melakukan banyak aktifitas seperti membawa barang, menaiki tangga, dan memanjat tebing selain berjalan dan berlari

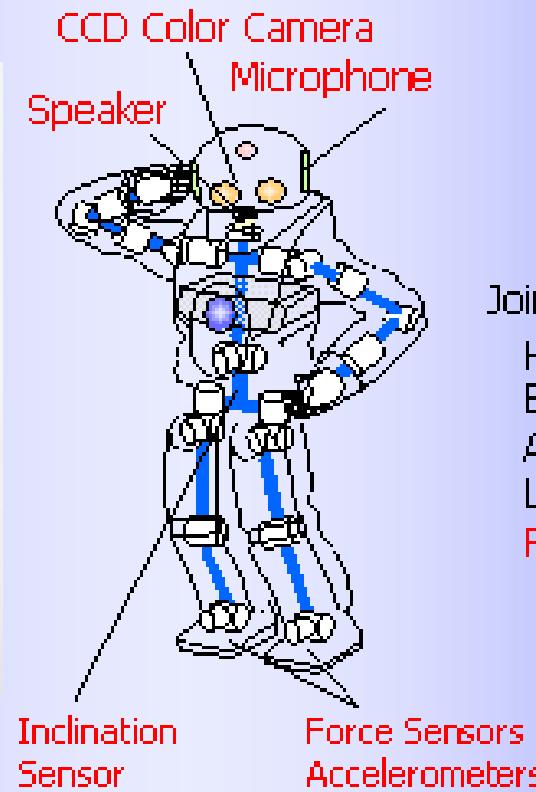
world.honda.com/asimo

QARIO



www.sony.net/SonyInfo/Qrio

Configurasi dasar QRIO



Joints: **38 DOF**

Head: **4 DOF**

Body: **2 DOF**

Arms: **5 DOF ×2**

Legs: **6 DOF ×2**

Fingers: **5 DOF ×2**



DOF = Degrees of Freedom

Multi-modal Human Interaction

- 1) Detect and Recognize a Face and Voice
 - Memorize and Recognize up to 10 Individual Faces
- 2) Continuous Speech Recognition
 - about 20,000 words
- 3) Unknown Vocabulary Acquisition
- 4) Conversation Control
 - based on short-term and long-term Memory
 - conversation using memorized words
 - conversation using memorized scenarios
- 5) Emotionally Expressive Speech

QRIO: Communication Technology

- Individual Detection, Identification, and Learning
 - Face and Voice detection, identification, and learning
- Large Vocabulary Continuous Speech Recognition
 - More than 20,000 words
- Unknown Word Acquisition
- Entertainment Dialogue
 - e.g. Reuse of acquired information
- Speech & Singing-Voice Synthesis
 - Speech Synthesis with emotion
 - Singing a song





Real-time Balancing / Walking



External Force Adaptive Control



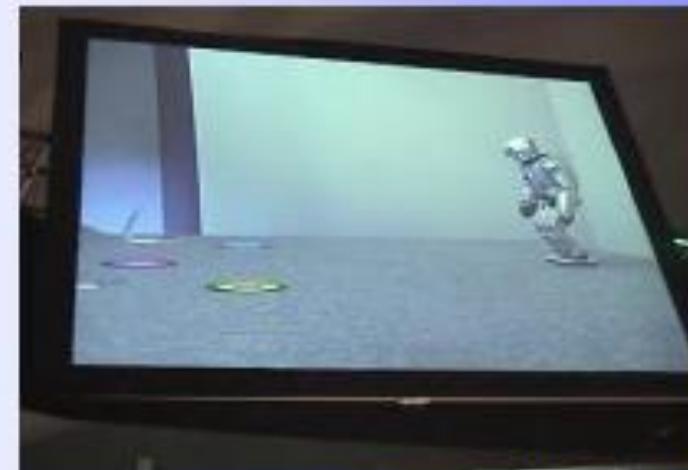
Fall Down / Recover Motion Control



Whole Body Motion Control



Sound Source Direction Estimation + Motion Detection



Simultaneous Localization And Map Building (SLAM) + Realtime Path Planning



Interaction using information from network



Imitation

THE END

PERTEMUAN 14

NEURAL NETWORK

Definisi Neural Network

Neural network (jaringan syaraf) adalah sistem pengolahan informasi yang didasari filosofi struktur perilaku syaraf mahluk hidup.

Struktur jaringan syaraf untuk mempelajari bagaimana menghasilkan keluaran yang diinginkan pada saat diberikan sekumpulan masukan. Hal ini dikenal dengan *input-output mapping*.

Menurut Haykin :

Jaringan syaraf tiruan (JST) didefinisikan sebagai prosesor tersebar pararel yang sangat besar dan memiliki kecenderungan untuk menyimpan pengetahuan yang bersifat pengalaman dan membuatnya siap untuk digunakan.

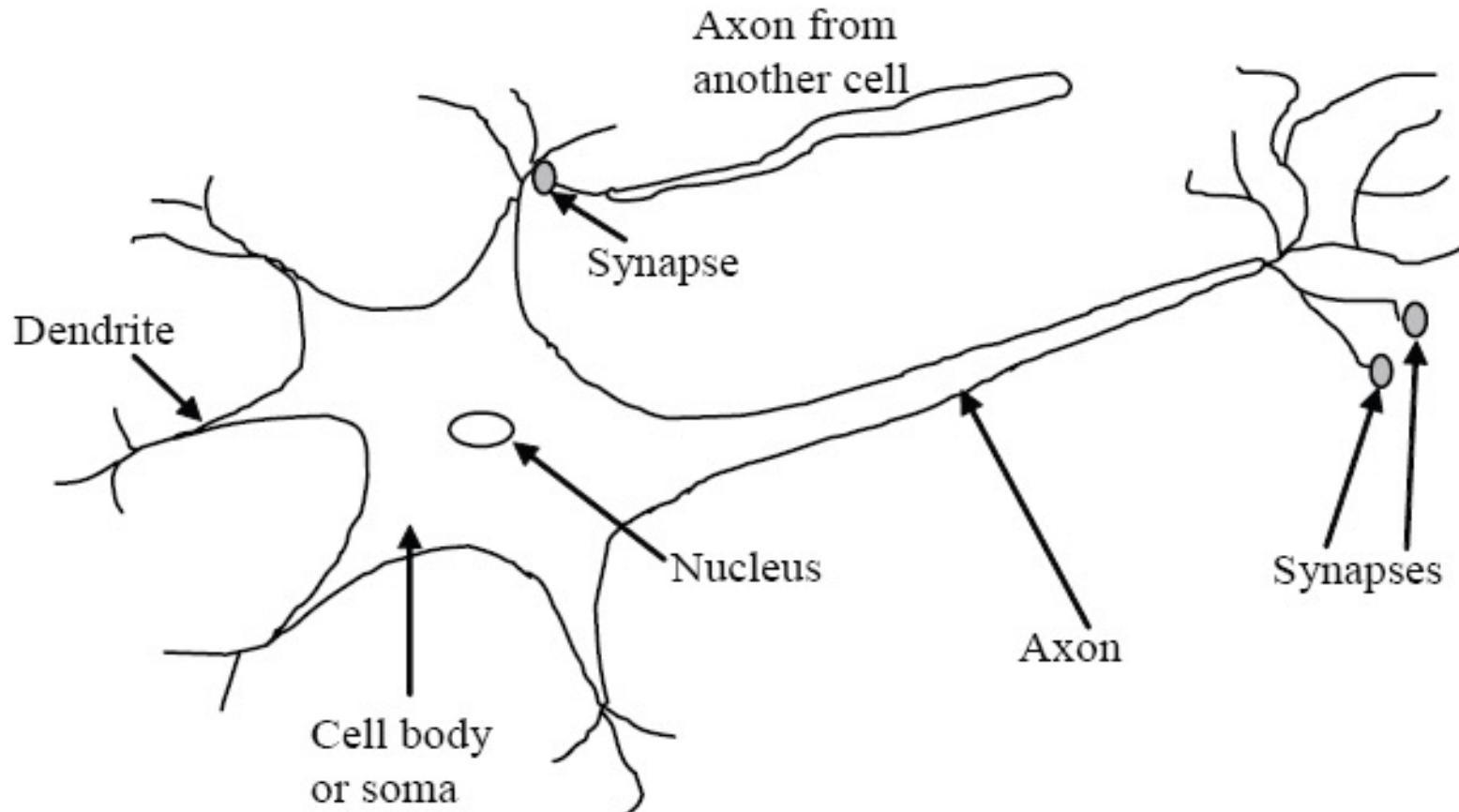
Proses ini dilakukan secara internal, yaitu dengan memerintahkan sistem untuk mengidentifikasi hubungan antar masukan kemudian mempelajarinya.

Proses eksternal, sistem bisa menggunakan umpan balik eksternal atau tanggapan yang diinginkan, untuk membentuk prilaku jaringan yang disebut sebagai *Supervised Learning*.

Jaringan Saraf Riil

Jaringan syaraf riil terdiri dari :

1. *Sinapsis*
2. *Dendrit*
3. *Axon*
4. Cell Body (kumpulan cell)



Sebuah saraf terdiri dari sebuah badan sel, satu axon, dan beberapa dendrit. Dendrit menerima masukan dari axon saraf lain yang memperlihatkan gairah atau kalangan sinopses. Saraf sesungguhnya mempunyai lebih banyak dendrit.

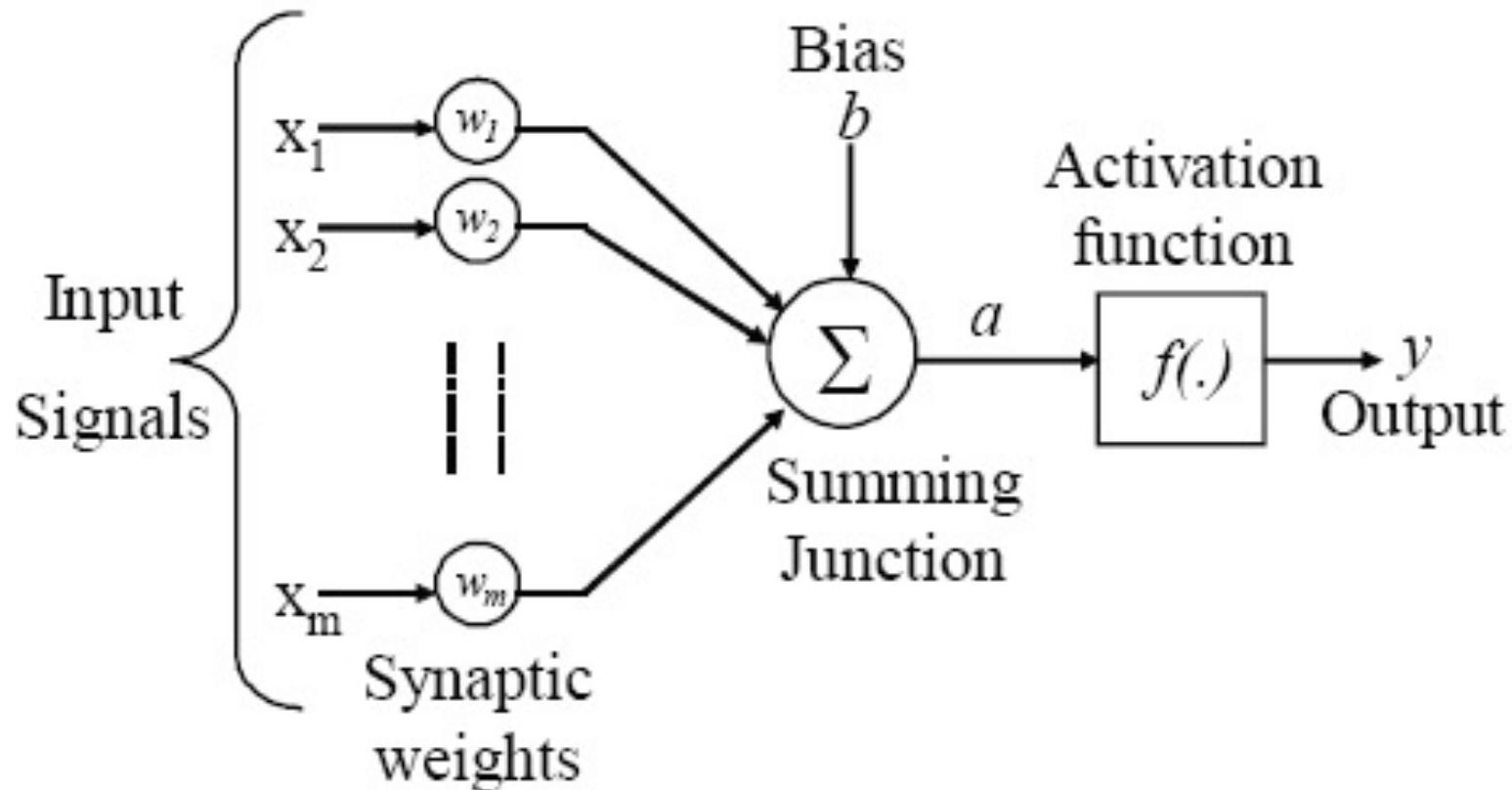
Jaringan Syaraf Tiruan

Jaringan syaraf tiruan terdiri dari :

1. Pengali
2. Penambah
3. Selisih

Jaringan Syaraf Tiruan (JST) menyerupai otak manusia dalam dua hal yaitu :

1. Pengetahuan yang diperoleh JST melalui proses belajar (*learning*)
2. Kekuatan hubungan antara neuron yang dikenal dengan *synaptic weights* digunakan untuk menyimpan pengetahuan.



Syaraf yang ditiru. Masukan dari saraf lain dikalikan dengan bobot dan kemudian diisi bersama. Jumlah itu yang kemudian dibandingkan dengan sebuah tingkat *threshold*. Jika jumlah di atas threshold, *outputnya* 1, kalau sebaliknya *output* 0.

(lihat gambar slide sebelum)

Input bagi neuron direpresentasikan oleh vektor x dan w sebagai bobot syaraf. Fungsi aktivasi merupakan jumlah dari perkalian x dengan w .

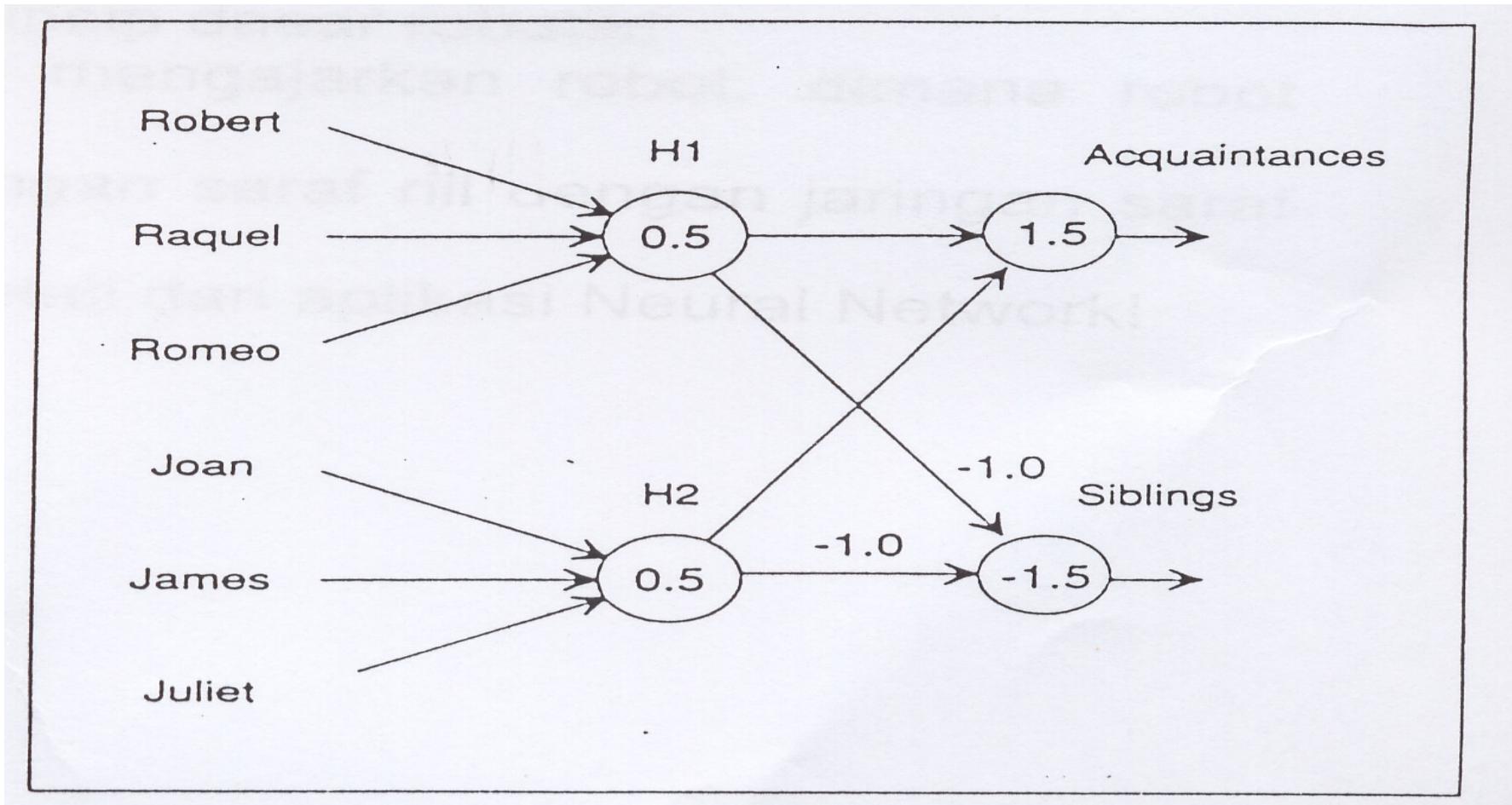
$$y = f\left(\sum_i w_i x_i\right) = f(\mathbf{w} \cdot \mathbf{x}) = f(\mathbf{w}^T \mathbf{x})$$

Bila w antara dua neuron bernilai positif maka input neuron memberikan efek *excitatory* atau menguatkan.

Sebaliknya maka input neuron memberikan *inhibitory* atau meredam.

$$f(t) = \begin{cases} 1 & \text{if } t > 0 \\ 0 & \text{if } t \leq 0 \end{cases}$$

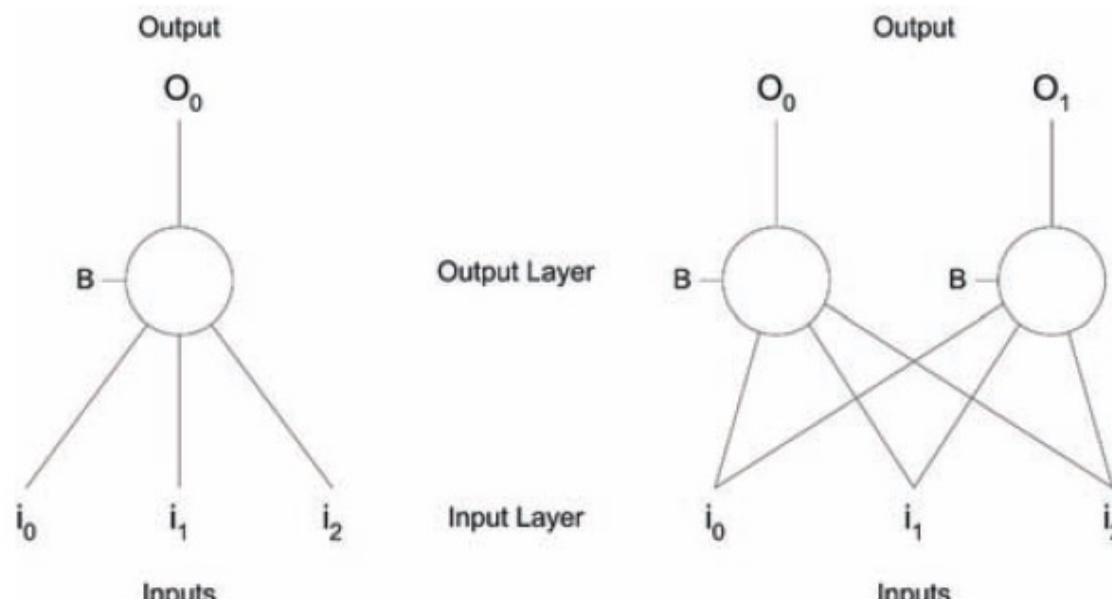
Contoh Jaringan Syaraf



Jaringan syaraf yang mengenali saudara kandung dan kenalan. Semuanya kecuali dua yang menunjukkan bobot ada kala 1.0. *threshold* adalah ditunjukkan di bagian dalam simpul tersebut.

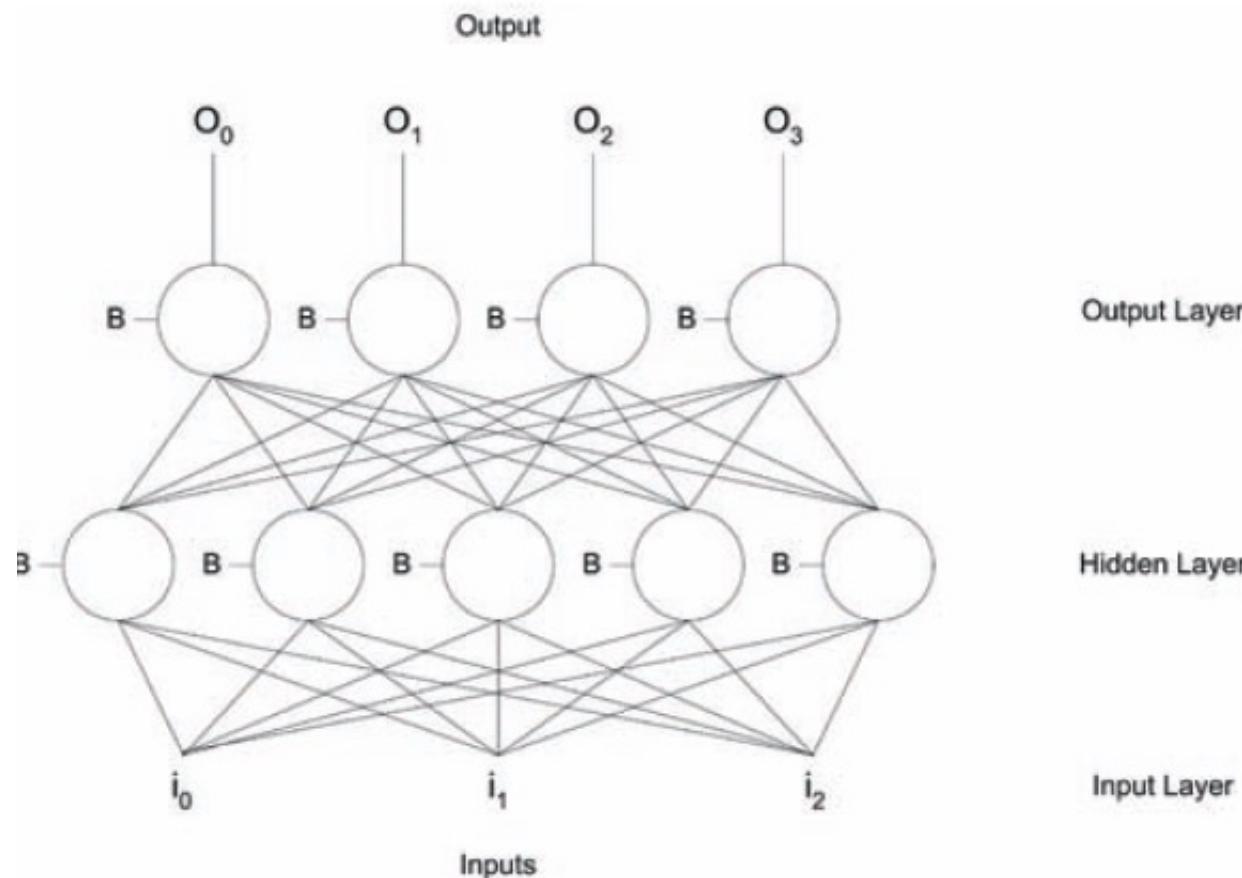
Single Perceptron Neural Network

Dapat dilihat pada gambar dibawah, Jaringan syaraf *Single perceptron (SLP)* ini terdiri atas lapisan input dan lapisan output. SLP ini merupakan model yang sederhana. Biasa digunakan untuk mengemulasikan fungsi logika NOT, OR, AND, NOR, NAND.



Multi Layer Perceptron (MLP)

JST *Multi Layer Perceptron*, terdiri dari lapisan input, lapisan tersembunyi (*hidden*) dan lapisan output.



Bila SLP digunakan untuk fungsi sederhana, MLP digunakan untuk fungsi yang lebih rumit seperti XOR. Jumlah lapisan pada Lapisan tersembunyi bisa sangat banyak.

Implementasi JST

1. Arsitektur

Pola koneksi antara neuron disebut Arsitektur JST. Suatu JST biasanya terdiri dari lapisan input, 1 atau 2 lapisan tersembunyi dan output. Dapat berupa SLP atau MLP. Berapa jumlah dan nilai input , *weight*, output.

2. Metode belajar

Setelah arsitektur telah selesai, JST perlu Belajar(*learning*) atau dilatih(*train*). JST Belajar dengan cara mengubah-ubah nilai *weight* sehingga output dari JST sesuai

Ada dua jenis metode pembelajaran :

1. Supervised Learning

Metode pembelajaran ini merupakan metode belajar dari contoh yang benar. Dalam metode ini JST tidak belajar sendiri tetapi diajarkan melalui contoh-contoh tersebut.

Contoh algoritma : *Backpropagation, Least-Mean-Squared*

2. Unsupervised Learning

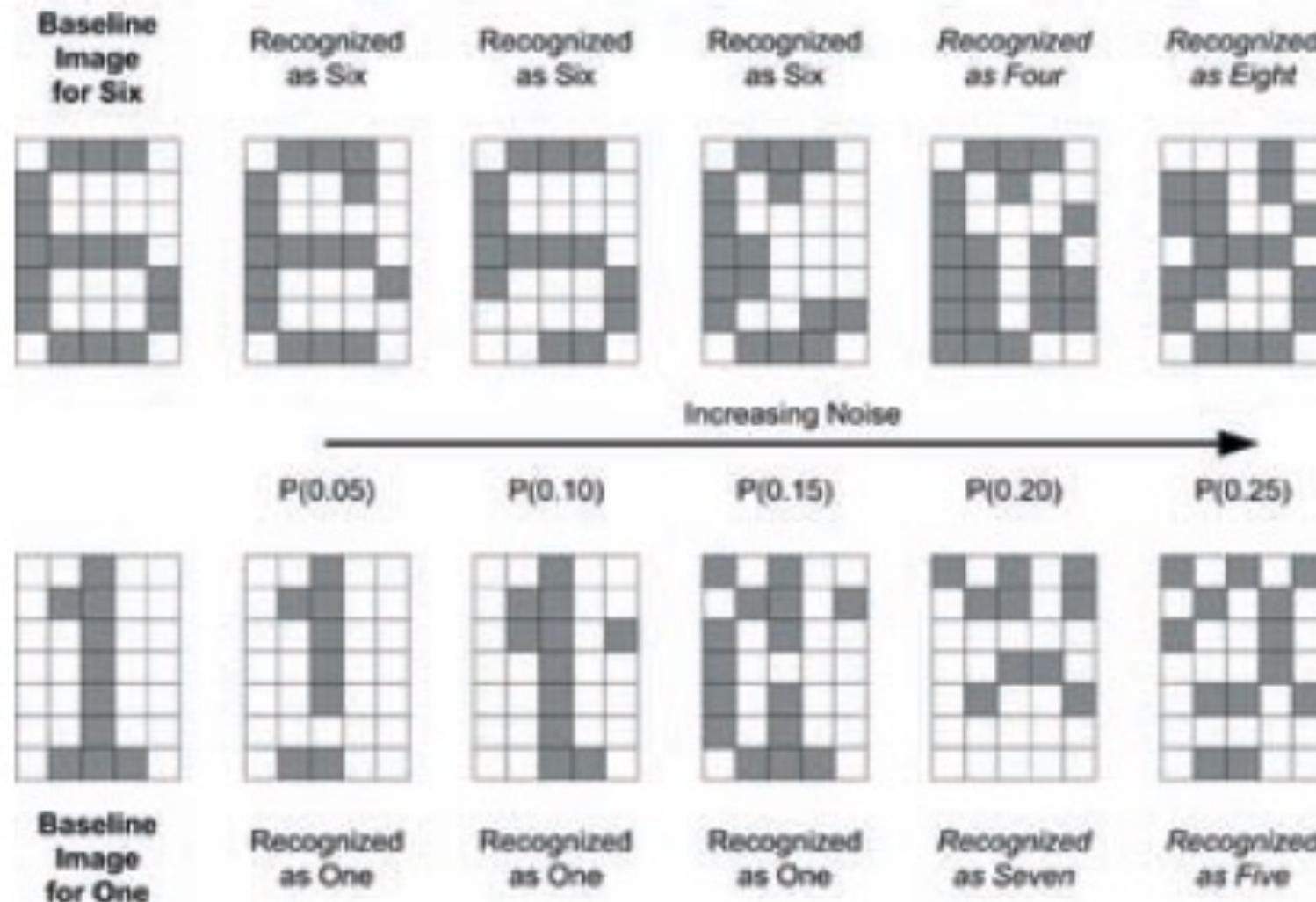
Pada metode ini JST tidak diberikan contoh-contoh yang benar. Tetapi mengandalkan analisa JST mengenali kesamaan dan perbedaan antara data-data input. Contoh algoritma : k-means Clustering, Kohonen, ART.

Contoh Implementasi JST dalam aplikasi dunia nyata :

1. Aproksimasi fungsi (*function approximation*), analisa regresi (*regression analysis*), prediksi berkala (*time series prediction*).
2. Klasifikasi, pengenalan pola. Misalnya pengenalan pola tulisan tangan, pengenalan pola huruf abjad.
3. Pemrosesan data seperti *clustering*, *filtering*.
4. Game.

Berikut ini akan diberikan sekilas bagaimanakah representasi input untuk pengenalan pola angka dalam JST

Representasi input JST dalam pengenalan pola angka (salah satu contoh aplikasi sederhana JST untuk pengenalan pola)



Dapat dilihat pada gambar sebelumnya, input yang berupa angka 6 direpresentasikan sebagai matriks berukuran 7×5 . Untuk sel yang berwarna hitam akan direpresentasi sebagai biner 1 dan sel berwarna putih akan direpresentasikan sebagai biner 0. Maka matrik untuk representasi angka enam pertama adalah

0	1	1	1	0
1	0	0	0	0
1	0	0	0	0
1	1	1	1	0
1	0	0	0	1
1	0	0	0	1
0	1	1	1	0

Pola pada matriks disamping ini dijadikan sebagai target atau baseline angka 6. Pada gambar slide sebelum, dapat dilihat input (berupa angka enam) dapat saja mengalami noise sehingga bila sangat parah akan menyebabkan JST tidak mengenali input sebagai enam tetapi angka lain yang lebih mirip.

Tentukan nilai *weight* harus ditentukan misalnya antara -0.5 s/d 0.5. nilai threshold misal antara 0.95 s/d1.0 terdekat dengan sebaliknya.

Setelah arsitektur selesai, JST kemudian dilatih. Algoritma belajar JST dapat dipilih diantara beberapa algoritma pembelajaran yang ada. Tujuannya yang dinginkan dari proses belajar adalah meminimalkan error dari kesalahan output yang sangat jauh dari output target (output yang dinginkan). Proses ini bisa sangat lama dalam pengimplementasiannya.

JST dapat dibuat dengan menggunakan bahasa pemrograman seperti C++, Visual basic, pascal , Matlab.

Atau JST dapat saja merupakan algoritma yang disisipkan ke dalam software lain seperti penggunaan algoritma JST dalam WEKA (untuk data mining), penggunaan jst dalam game.



THE END