BAB II

DASAR TEORI

2.1 Sistem Diagnosa Penyakit berbasis Perangkat Lunak

Kesehatan merupakan hal yang begitu penting bagi manusia. Ironisnya banyak sekali penyakit-penyakit yang pada akhirnya terlambat didiagnosa sehingga mencapai tahap kronis yang membuatnya sulit untuk ditangani. Padahal setiap penyakit sebelum mencapai tahap kronis umumnya menunjukkan gejala-gejala penyakit yang telah diderita oleh pasien tetapi masih dalam tahap yang ringan misalnya seperti sakit kepala, batuk atau nyeri pada sendi. Sayangnya karena ketidaktahuan pada penyakit tersebut maka biasanya seorang penderita akan mengabaikan gejala-gejala ringan tersebut yang berpotensi untuk menjadi fatal jika tidak diberi tindakan pengobatan.

Kemajuan teknologi mendorong setiap orang untuk terus memperbaharui sistem pelayanan kepada seorang pengguna jasa. Sistem pelayanan yang dimaksud disini akan membahas tentang tentang sistem pelayanan kesehatan. Sistem ini harus dapat memberikan kenyamanan kepada pasien dalam segala bidang. Hal yang perlu menjadi pertimbangan adalah bahwa pelayanan kesehatan konvensional

akan banyak menyita waktu seorang pasien. Meskipun pasien hanya ditanyai beberapa pertanyaan oleh seorang dokter mengenai gejala penyakitnya. Akan tetapi pasien tersebut tetap saja diharuskan untuk meluangkan waktunya untuk datang kerumah sakit, mengantri dan bertemu dokter. Belum lagi seorang pasien diharuskan untuk membayar biaya yang cukup mahal untuk bertemu dokter.

Sistem pelayanan kesehatan seperti ini sebenarnya dapat diganti dengan sistem baru yang berbasiskan perangkat lunak asalkan diketahui pola kerja seorang dokter dalam mendiagnosa suatu penyakit. Sehingga sistem ini akan sangat mudah dijangkau oleh setiap orang dalam pengaksesannya.[8]

2.2 Cara Diagnosa Dokter

Seorang dokter dalam mendiagnosa penyakit pasiennya akan melalui tiga tahapan yang akan dilakukannya :

1. Wawancara
2. Pengecekan Fisik
3. Pengecekan Laboratorium/Penunjang

Ketiga tahapan ini diurutkan berdasar pada metode seorang dokter dalam mendiagnosa suatu penyakit. Pada prakteknya, seorang dokter kemungkinan hanya melakukan tahapan wawancara untuk mendiagnosa penyakit dari seorang pasien. Karena pada tahap ini, merupakan keluhan dan gejala yang pertama dirasakan pasien saat ingin memeriksakan diri kedokter, sehingga dua tahapan

lainnya hanya dilakukan untuk menambah keyakinan dari seorang dokter untuk memvonis seorang pasien tentang penyakit yang dideritanya.

2.2.1 Wawancara

Pada tahapan ini, seorang dokter akan menanyakan seorang pasien tentang gejala yang dideritanya. Dimulai dari pertanyaan pertama tentang keluhan utama dari seorang pasien. Keluhan utama ini merupakan gejala yang diderita oleh seorang pasien yang menyebabkan mengapa pasien tersebut datang ke rumah sakit. Keluhan utama setiap orang dapat berbeda-beda walaupun penyakit yang diderita sama, tergantung dari tingkat ketahanan fisik seseorang terhadap suatu penyakit. Setelah diketahui keluhan utamanya maka selanjutnya dokter akan menanyakan tentang gejala-gejala yang diderita oleh seorang pasien. Gejala-gejala inilah yang akan dibandingkan dengan kriteria suatu penyakit dari gejala yang ada.

2.2.2 Pengecekan Fisik

Tahapan ini merupakan tahapan dimana seorang dokter melihat tanda-tanda fisik yang ada dari tubuh seorang pasien, mulai dari ujung kepala hingga ujung kaki. Pengecekan ini dilakukan dikarenakan ada beberapa penyakit yang memiliki ciri-ciri yang dapat dilihat pada tanda-tanda fisik dari seorang pasien. Contohnya, bintik merah yang timbul apabila seseorang menderita penyakit demam berdarah. Beda dengan pengecekan laboratorium, pengecekan fisik tidak menggunakan bantuan dari laboratorium untuk mendapatkan hasilnya.

2.2.3 Pengecekan Laboratorium

Pengecekan laboratorium merupakan tahapan dimana seorang dokter memerlukan bantuan dari laboratorium untuk memeriksa seorang pasien tentang suatu penyakit yang dideritanya. Pengecekan ini meliputi : Pengambilan darah, tes urin, *rontgen*, dan lain-lain. Pengecekan laboratorium hanya dilakukan pada jenis-jenis penyakit tertentu yang sangat rumit dalam penentuan jenis penyakitnya jika hanya mengandalkan tahapan wawancara dan pengecekan fisik.[8]

2.3 Sistem Pakar

Sistem pakar adalah suatu program komputer yang mengandung pengetahuan dari satu atau lebih pakar manusia mengenai suatu bidang spesifik.[1] Jenis program ini pertama kali dikembangkan oleh periset kecerdasan buatan pada dasawarsa 1960-an dan 1970-an dan diterapkan secara komersial selama 1980-an. Bentuk umum sistem pakar adalah suatu perangkat lunak yang dibuat berdasarkan suatu set aturan yang menganalisis informasi (biasanya diberikan oleh oleh pengguna suatu sistem) mengenai suatu permasalahan yang spesifik serta analisis matematis dari masalah tersebut. Tergantung dari desainnya, sistem pakar juga mampu merekomendasikan suatu rangkaian tindakan pengguna untuk dapat menerapkan koreksi. Sistem ini memanfaatkan kemampuan penalaran untuk mencapai suatu kesimpulan. Tujuan dari sebuah sistem pakar adalah untuk mentransfer kepakaran/keahlian yang dimiliki seseorang pakar/ahli kedalam komputer, dan kepada orang lain (*nonexpert*).

Pengertian dari kata pakar/ahli itu sendiri adalah seorang individu yang memiliki kemampuan atau pemahaman yang berlebih pada suatu masalah atau bidang ilmu. Misalnya: seorang dokter, penasehat keuangan, pakar mesin mobil, dll. Dengan bersandar pada definisi tersebut maka seorang pakar/ahli memiliki kemampuan yang meliputi kemampuan untuk mengenali dan merumuskan masalah, menyelesaikan masalah dengan cepat dan tepat, menjelaskan solusi, belajar dari pengalaman, restrukturisasi pengetahuan, menentukan relevansi/hubungan, dan memahami batas kemampuan. Kepakaran/keahlian dari seseorang dapat diperoleh dengan memiliki pemahaman yang luas dari tugas atau pengetahuan spesifik yang diperoleh dari pelatihan, membaca dan pengalaman.

Alasan mendasar mengapa sistem pakar (*Expert System*) dikembangkan untuk menggantikan seorang pakar:[13]

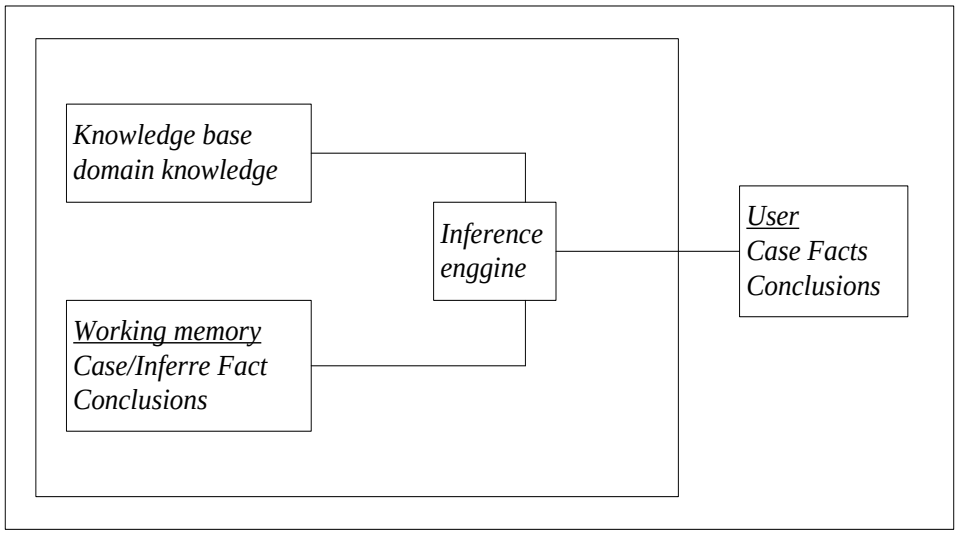
- Dapat menyediakan keahlian dari sebuah profesi setiap waktu dan diberbagai lokasi
- Secara otomatis mengerjakan tugas-tugas rutin yang membutuhkan seorang pakar.
- Seorang pakar dapat pensiun atau pergi.
- Biaya untuk menggunakan tenaga seorang pakar sangatlah mahal.
- Tenaga ahli dibutuhkan juga pada lingkungan yang tidak bersahabat.

Tabel 2.1. Perbandingan seseorang ahli/pakar (*Human Expert*) dengan sebuah sistem pakar (*Expert System*)[13]

Parameter Pembanding	Human Expert	Sistem Expert
Waktu Kesediaan	Hari Kerja	Setiap Saat
Letak Pengaksesan	Di Suatu Tempat Tertentu	Dimana Saja
Dapat Habis	Ya	Tidak
Performansi	Tidak Konsisten	Konsisten
Kecepatan	Tidak Konsisten	Konsisten
Biaya	Tinggi	Terjangkau

2.3.1 Struktur Dasar Sistem Pakar

Berikut dapat dilihat pada gambar 2.1 bagian-bagian yang terdapat pada sistem pakar beserta kegunaannya dalam memecahkan masalah.



Gambar 2.1. Cara kerja sistem pakar dalam memecahkan masalah.[14]

Knowledge Base

Bagian ini merupakan bagian dari sistem pakar yang berisi domain pengetahuan. Bagian ini berisi tentang pengetahuan yang dibutuhkan untuk memahami, merumuskan dan menyelesaikan suatu masalah.

Working Memory

Bagian ini merupakan bagian dari sistem pakar yang berisi fakta-fakta masalah yang ditemukan dalam suatu sesi. Bagian ini berisi fakta-fakta tentang suatu masalah yang ditemukan dalam proses konsultasi.

Inference engine

Pada bagian ini terdapat suatu pemroses (*Processor*) pada sistem pakar yang bertugas untuk mencocokkan fakta-fakta yang ada pada *Working Memory* dengan domain pengetahuan yang terdapat pada *Knowledge Base*, untuk menarik kesimpulan dari masalah yang dihadapi. Proses berfikir pada manusia dimodelkan dalam sistem pakar pada bagian ini.

2.3.2 Proses Pembuatan Sistem Pakar

Terdapat dua proses dalam rangka pembuatan sistem pakar :[13]

1. Knowledge Aqcuisistion

Pada proses ini terjadi kegiatan pengumpulan, pemindahan, dan perubahan dari kemampuan seorang pakar atau sumber pengetahuan kedalam program komputer.

2. Knowledge Engineering

Pada proses ini terjadi kegiatan pengembangan suatu sistem pakar yang dilakukan oleh pemrogaman yang dalam hal ini disebut *Knowledge Engineer*.

2.3.3 Komponen dalam Sistem Pakar

Ada tiga hal yang terlibat dalam keberadaan sistem pakar :[13]

1. Domain Expert

Merupakan bidang ahli, meliputi keahlian dari seseorang suatu bidang.

2. Knowledge Enggineer

Merupakan bagian dimana seorang pemrogram membangun suatu sistem pakar.

3. End-User

Merupakan bagian yang berisi pengguna awam yang akan menggunakan fasilitas dari sistem pakar tersebut.

2.3.4 Kategori Permasalahan Sistem Pakar

Terdapat beberapa permasalahan dalam sistem pakar secara umum :[13]

1. Interpretasi - membuat kesimpulan atau deskripsi dari sekumpulan data mentah.
2. Prediksi - memproyeksikan akibat-akibat yang dimungkinkan dari situasi-situasi tertentu.
3. Diagnosa - menentukan sebab malfungsi dalam situasi kompleks yang didasarkan pada gejala-gejala yang teramati.

4. Desain - menentukan konfigurasi komponen-komponen sistem yang cocok dengan tujuan-tujuan kinerja tertentu yang memenuhi kendala.
5. Perencanaan - merencanakan serangkaian tindakan yang akan dapat mencapai sejumlah tujuan dengan kondisi awal tertentu.
6. *Debugging* dan perbaikan - menentukan dan menginterpretasikan cara-cara untuk mengatasi malfungsi.
7. Instruksi - mendeteksi dan mengkoreksi defisiensi dalam pemahaman domain subyek.
8. Pengendalian - mengatur tingkah laku suatu *environment* yang kompleks.
9. Seleksi - mengidentifikasi pilihan terbaik dari sekumpulan (*list*) kemungkinan.
10. Simulasi - pemodelan interaksi antara komponen-komponen sistem.
11. Pengamatan - membandingkan hasil pengamatan dengan kondisi yang diharapkan.

2.4 Aplikasi-Aplikasi Pendukung

Untuk dapat menjalankan sistem diagnosa penyakit berbasis perangkat lunak dibutuhkan beberapa aplikasi pendukung seperti Amzi Prolog. Untuk bahasa pemrograman yang dipakai adalah bahasa pemrograman *Prolog* (*Program Logic*). Untuk lebih jelasnya, berikut ini adalah fungsi dari masing-masing komponen.

2.4.1 Sekilas Mengenai Prolog

Bahasa Prolog merupakan bahasa generasi kelima yang mendorong pemrograman kedalam dimensi baru. Bahasa Prolog dibangun atas dasar pemrograman alamiah dan logika. Untuk itulah lahir nama Prolog, yaitu singkatan dari **Programming in Logic**.

Prolog juga merupakan bahasa **deklaratif**. Artinya, jika kita memberi fakta dan aturan, prolog akan menyelesaikan problem secara deduktif; atau dari banyak fakta dan aturan kemudian diturunkan kesimpulan sebagai jawaban. Hal ini berbeda sekali dengan bahasa prosedural seperti Pascal atau Fortran. Dalam bahasa prosedural, pemrogram harus memberi perintah untuk memecahkan persoalan, langkah demi langkah; dengan kata lain, pemrogram harus tahu terlebih dahulu bagaimana memecahkan masalah itu. Bandingkanlah dengan pemrograman Prolog yang hanya perlu memberikan penjelasan masalah (fakta) dan aturan dasar untuk memecahkannya. Disini Prolog dibiarkan untuk menentukan sendiri bagaimana mencari jawaban. Dengan demikian program aplikasi penunjang kecerdasan buatan sepert basis pengetahuan, sistem pakar, antarmuka kebahasa alami, pemrograman simbolik serta sistem manajemen informasi yang cerdik, dan pengenalan citra, akan lebih mudah diwujudkan.[3]

2.4.2. Sejarah Singkat Prolog

Kelahiran Prolog diawali ketika Alain Colmearaurer dengan sekelompok peneliti menghadapi masalah penterjemahan bahasa dengan komputer di Montreal. Hal

ini terutama akibat hasil karya Chomsky dalam ilmu bahasa. Dia dan koleganya menelusuri cara mempertemukan bahasa alami dengan bahasa komputer.

Usaha itu disusul dengan dikembangkannya sejumlah bahasa percobaan di Montreal dan Marseilles, diantaranya Tarzan yang merupakan bahasa untuk memanipulasi *tree*. di antara rekan Col Meraurer, Roussel tertarik pada segi deduksi pada sistem percobaannya dan menentukan kemngkinan penggunaan logika. Dari segi teori, pembangunan gagasan pemrograman dengan logika (*programming in logic*) yang pertama adalah Kowalski di Edinburgh. Kemudian didemonstrasikan secara eksperimental oleh Maarten Van Emden juga di Edinburgh, dan diwujudkan oleh Alain Colmeraurer di Universitas Marseiles, dengan berhasil membuat interpreter Prolog yang pertama bersama kelompoknya pada tahun 1972. Yang paling berpengaruh dalam implementasi Prolog ini adalah dihasilkannya kompiler Prolog pertama untuk komputer DEC-10 karya Warren dan rekan-rekannya pada tahun 1979. Secara implisit sistem ini memperlihatkan sintaks prolog baku (standard) dan efisiensinya membuktikan bahwa Prolog, disamping bahasa tingkat tinggi yang menarik. juga bahasa yang berdaya guna.[4] Karya David Warren juga memperlihatkan daya saingnya terhadap bahasa kecerdasan buatan (AI) lainnya (Lisp) dalam hal kecepatan.[6]

Akhirnya, dalam konfrensi tentang komputer generasi kelima di London yang diselenggarakan PL International, disajikan berbagai keberhasilan penggunaan Prolog yang memberi kesan bahwa Prolog adalah bahasa kecerdasan buatan yang lebih baik dari Lisp.[7]

2.4.3 Pemrograman Logika

Untuk mengenal pemrograman logika, terlebih dahulu kita singgung **logika predikat orde satu** (*first order predicat logic*), yang selanjutnya kita sebut **logika predikat** (kalkulus predikat). Logika predikat merupakan bagian dari komputasi logika yang juga mencakup aljabar Boole (logika proposisional). Dalam logika predikat, fakta dan aturan dinyatakan melalui predikat, seperti:

lelaki(beck)	(fakta)
menikah(Beck, Ewon)	(fakta)
∀ x ∀*y* [menikah (x,y) ^ lelaki (x) -- > ~ lelaki (y)	(aturan)
∀x ∃*y* [orang(x) -- > punyaibu(x,y)]	(aturan)

Kalimat pertama menyatakan beck seorang lelaki. Yang kedua menyatakan bahwa Beck menikah dengan Ewon. Kalimat ketiga mengatakan untuk setiap x dan untuk semua y, jika x menikah dengan y dan x lelaki, maka y bukan lelaki. Berikutnya, yang terakhir menyatakan bahwa untuk setiap x, ada y, jika x orang, maka y adalah ibu dari x.

Simbol predikat yang digunakan dalam contoh diatas adalah*: lelaki, menikah, orang punya ibu*. Simbol konstanta dalam contoh tadi adalah Beck dan Ewon, dengan operator ^ (dan), -- > (implikasi), ~ (negasi), ∀ (untuk setiap), ∃ (ada). Perhatikan bahwa ada dua simbol terakhir merupakan besaran logika (kwantor).

Fakta dan aturan dalam prolog tidak tertulis tepat seperti cara diatas tapi dalam bentuk yang terkadang disebut **bentuk klausa Kowalski**. Bentuk klausa ini

sebenarnya diturunkan dari **bentuk klausa Horn**. Yang dimaksud klausa disini adalah suatu hubungan disjungsi beberapa literal (fakta dan aturan), dan klausa Horn adalah klausa yang hanya terdiri dari (paling banyak) satu literal positif. Contoh klausa Horn dan penurunan menjadi klausa Prolog adalah (tanda **v** berarti atau /disjungsi):

~lelaki(X) v ~ menikah (X,Y) v perempuan (Y) (klausa Horn)

(*perempuan* merupakan satu-satunya literal positif) dengan menggunakan hokum De Morgan dan rumus:

~AvB = A-- >B

diperoleh:

~(lelaki(X) ^ menikah (X,Y)) v perempuan(Y) =
~lelaki(X) ^ menikah (X,Y) -- > perempuan(Y)

Yang dalam prolog ditulis sebagai:

Perempuan(Y) if lelaki(X) and menikah(X,Y)

Gagasan yang melatar-belakangi pemrograman logika adalah menggantikan pemrograman dan komputasi dengan deskripsi logika suatu masalah dan mekanisme pembuktian untuk mendeduksi jawabannya. Pemikiran dasar dalam pemrograman logika membawa kita satu langkah kedepan, yakni komputasi adalah deduksi. Sebagai contoh, fungsi factorial biasa didefinisikan sebagai berikut:

```
faktorial(n) : = if n = 0 then 1
        else n* factorial(n-1);
```

jika sebagai pengganti program diatas kita tulis predikat dalam Prolog yang dtulis:

```
factorial (n,m)
```

yang secara logika benar jika m adalah faktorial n, dan dengan klausa:

```
faktorial(0,1).
faktorial(n,m) if n1 = n-1 and faktorial(n1,m1) and
                    m = n*m1.
```

Maka dengan **klausa logika** diatas kita dapat menghitung nilai faktorial. Inilah yang merupakan pokok pemrograman logika, yakni dengan klausa logika kita dapat melakukan perhitungan.[3]

2.4.4 Ciri Bahasa Prolog

Ciri yang menonjol dalam Prolog adalah disamping mencari jawaban secara logika terhadap pernyataan yang kita ajukan, juga dapat member jawaban lain atau member semua kemungkinan jawaban. Selain hanya berjalan dari awal sampai akhir program, Prolog dapat berjalan mundur dan mencari lebih dari satu cara penyelesaian tiap bagian masalah. Secara sistematik, cirri lain prolog yang menonjol adalah:[3]

a. Predikat

Prolog membutuhkan fakta-fakta yang terungkap dalam **relasi** dan **sifat** untuk mencari jawaban. Sebagai contoh:

```
Irham suka permen
Permen manis
```

Ini menggambarkan objek Irham dan permen dengan relasi suka, dan sifat manis pada permen. Seringkali, relasi bernilai benar jika memenuhi kondisi tertentu. Hubungan ini disebut **aturan** (rule).

Contoh aturan adalah:

```
Irham suka permen jika rasa permen asam
```

Dalam prolog, aturan dan fakta dinyatakan dalam kalimat yang disebut **klausa** (clause) seperti;

suka (Irham,permen).

Sintaks untuk menyatakan fakta seperti diatas disebut **logika predikat** (predicate logic). Untuk aturan diatas, kita tuliskan dalam Prolog:

Suka(Irham, permen) if rasa (permen, asam)

atau dapat pula berbentuk:

suka (Irham, permen) if asam (permen)

Selanjutnya, logika predikat ini kita sebut saja **predikat**, yang merupakan nama simbolik suatu relasi atau sifat, sedangkan objek yang suka dikaitkan disebut **argument**. Contoh predikat diatas adalah *suka* dengan argumen *Irham* dan *permen*.

b. Basis Data

Sebagai bahasa yang berorientasi pada fakta dan untuk menunjangn basis pengetahuan, Prolog menyediakan fasilitas basis data.

c. Deduksi

Prolog dapat melakukan deduksi (penarikan kesimpulan) seperti:

Diberikan fakta:

Dede suka Iik
Cece suka Rayan

dan aturan:

Celung suka Raya

d. Pemadanan dan Unifikasi

Dalam mencari jawaban atau menarik sesuatu kesimpulan (*reasoning*), Prolog melakukan **pemadanan**. Argumen yang pertama dipadankan dengan argumen pertama predikat lain yang sama, dan seterusnya. Tujuan melakukan pemadanan adalah agar dapat melakukan **unifikasi**, yakni penyatuan argument dalam suatu predikat dengan predikat lain yang sudah identik. Sebagai contoh, kita semua tahu bahwa bujur sangkar adalah suatu bangun segi empat, berisi sama dan bersudut 90 derajat. Andaikan kita diminta untuk memeriksa apakah suatu bangun berbentuk bujur sangkar. Pertama kita menduga bahwa suatu bangun berbentuk bujur sangkar. Jika kemudian ketiga syarat diatas terpenuhi, maka dugaan kita benar bahwa bangun yang kita periksa adalah bujur sangkar. Dalam Prolog, dugaan bahwa suatu bangun berbentuk bujur sangkar adalah unifikasi variabel, katakanlah X, dengan nilai bujur sangkar. Ini terjadi karena ada pemadanan antara bangun (X) dengan bangun (bujur sangkar) yang sudah identik, kemudian dilakukan unifikasi antara X dengan bujur sangkar. Dalam bahasa lain, pemberian nilai terhadap suatu variabel dilakukan dengan statemen pemberian (assignment statement) seperti y = 2. Ini bersifat permanen selama tidak ada statemen lain yang mengubahnya. Unifikasi bersifat sementara selama belum ada lacak balik dan hanya berlaku satu klausa, sebagaimana hanya berlaku dalam satu fungsi seperti dalam C.

e. Pembuktian Mundur

Dari contoh pembuktian bahwa suatu bangun berbentuk bujur sangkar, prolog menganggap suatu bangun berbentuk bujur sangkar terlebih dahulu, baru

kemudian dilakukan pembuktian. Cara ini disebut **pembuktian mundur** (backward chaining) atau disebut juga metoda ***top-down*** karena dari kenyataan pokok (bujur sangkar) kemasalah rinci (sudut dan sisi).

f. Lacak Balik (*backtracking*)

Dalam mencari jawaban, Prolog melacak suatu fakt atau aturan dengan melakukan pemadanan (*matching*) berturut-turut. Jika sepadan, akan terjadi unifikasi dan dilanjutkan pembuktian yang selanjutnya dilakukan pemadanan serupa. Katakanlah dalam pembuktian suatu bentuk seperti diatas, kita menemui **kegagalan**, misalnya, karena sudut-sudutnya tidak sama dengan 90 derajat. Yang pertama kita lakukan adalah mengubah anggapan bahwa suatu bangun berbentuk bujur sangkar. Langkah berikutnya adalah mencari anggapan baru dengan definisi baru, misalnya bentuk trapezium. Dalam prolog, kejadian seperti ini disebut sebagai **lacak balik** (*backtracking*). Jadi, jika kemudian menemui kegagalan, artinya hipotesa pertama tidak dapat digunakan harus dibatalkan. Kemudian Prolog melakukan pelacakan mundur (lacak balik) untuk mencari hipotesa baru. Lacak balik akan berhenti jika ditemui jawaban atau kehabisa fakta untuk menyimpulkan jawaban.

g. Rekursi

Prolog merupakan program yang paling banyak memanfaatkan rekursi, baik dalam aturan maupun data. Rekursi berarti cara memecahkan masalah dengan menguraikan masalah tadi menjadi lebih kecil tetapi memiliki struktur atau model yang sama. Dalam bahasa procedural, kita kenal rekursi sebagai suatu prosedur

yang memanggil dirinya sendir dengan memperkecil permasalahan. Akan tetapi tidak semua bahasa mampu melakukan rekursi, misalnya: Fortran dan Cobol.

Contoh bentuk rekursi yang sederhana adalah faktorial. Untuk mencari faktorial 4 misalnya, dicari faktorial 3 untuk dikalikan dengan 4. Faktorial 3 diperoleh dari faktorial 2 kali 3 dan seterusnya sehingga sampai pada faktorial yang diketahui nilainya. Jelas bahwa disini ada usaha memperkecil permasalahan, sehingga masalah yang lebih dasar dapat dipecahkan melalui masalah yang lebih kecil dan mudah dengan bentuk yang serupa. Contoh data rekursif adalah list, yakni sekumpulan data sejenis dalam satu besaran data. List terdiri dari kepala (unsur pertama) dan list ekor yang lebih kecil, dan seterusnya.

h. Pengolahan simbol

Prolog ditekankan juga pada pengolahan informasi simbolik seperti pada contoh diatas, juga untuk pengolahan struktur data dinamis dan membuat struktur data baru. Selain itu prolog juga dapat memanipulasi simbol seperti dalam pemecahan persamaan matematik, diferensial dan integral, dan lain sebagainya.

i. Meta Programming

Dalam bahasa lain, apabila program dijalankan maka algoritma pemrograman tidak dapat diubah lagi, baik dalam interpreter apalagi melalui *compiler*. Pengubahan algoritma hanya dapat dilakukan pada program sumber melalui suatu editor. Dengan prolog, kita dapat membuat program yang dapat diubah prosedurnya atau dalam hal ini logikanya pada saat program dijalankan.

j. Paralel

Meskipun kebanyakan implementasi Prolog sampai sekarang ini belum bersifat paralel, Prolog merupakan bahasa yang berstruktur paralel dari sifatnya. Ini karena beberapa klausa (yang sama) dapat dieksekusi secara bersamaan (*concurrent*). Paralelisme demikian disebut **Or-Paralellism**. Dalam klausa sendiri, beberapa sub-klausa dapat dieksekusi secara paralel, dan disebut **And-Parallelism**. Hal ini membuat Prolog sebagai calon kuat bahasa komputer masa depan dengan arsitektur paralel.[3]

2.4.5 Bahasa Deklaratif

Pokok perbedaan prolog dari bahasa lain adalah karena bersifat deskriptif atau deklaratif, sedangkan bahasa lain bersifat procedural atau **imperatif**. Artinya, Prolog hanya membutuhkan deklarasi atau uraian masalah sedangkan yang lain memerlukan perintah. Dalam memecahkan masalah, prolog melakukan deduksi (dari sekumpulan fakta ditarik kesimpulan). Disini kewajiban kita adalah memberi perincian masalah secara benar, bukannya membuat sekumpulan perintah untuk dikerjakan komputer.

Perbedaaan deklaratif dan prosedural dapat diterangkan dengan perumpamaan seperti kita naik taksi. Secara deklaratif, kita hanya meminta supir taksi untuk membawa kita ke suatu tempat tujuan, tidak peduli jalan mana saja yang harus ditempuh. Penentuan jalan yang ditempuh diserahkan kepada supir. Dalam

pengertian prosedural, kita yang menentukan jalan mana saja yang harus ditempuh dan boleh saja supir taksi tidak tahu kemana tujuan kita sebenarnya.

Sebagai bukti bahwa Prolog merupakan bahasa deklaratif adalah:

1. Jika kita ingin menerangkan bahwa A adalah orang tua B, dalam Prolog cukup ditulis:

   Orangtua(A,B).

2. Jika kita menerangkan bahwa A adalah leluhur Z, maka harus didefinisikan terlebih dahulu logikanya yang dalam bahasa Indonesia berbunyi sebagai berikut:

   A adalah leluhur Z jika A orang tua Z atau
   A adalah leluhur Z jika orang tua B dan B leluhur Z.

Aturan seperti ini dalam prolog ditulis sebagai aturan rekursif seperti berikut:

leluhur(A,Z) if orangtua(A,Z).
leluhur(A,Z) if orangtua(A,B) and leluhur(B,Z).

Hanya dengan menuliskan aturan seperti ini, kita dapat menentukan siapa leluhur Z, siapa keturunan A atau membuktikan kebenaran bahwa A leluhur Z dengan memberikan data hubungan (fakta) orang tua secukupnya seperti diatas tanpa harus merinci cara mencari jawaban atau cara membuktikannya.

Aturan seperti diatas diturunkan berdasarkan logika, sehingga pemrograman prolog, seperti namanya, disebut pemrograman logika (logic programming). Dari contoh diatas jelas bahwa Prolog adalah bahasa berjenis *Apa* yakni *Apa yang harus dipecahkan* atau disebut juga **Goal Oriented**. Sedangkan yang lain berjenis *Bagaimana* , yakni *bagaimana memecahkan masalah*. Dalam hal terakhir ini

komputer hanya sebagai pembantu untuk mempercepat penyelesaiian masalah. Dengan adanya Prolog, diharapkan bahwa komputerlah yang berfungsi sebagai pemecah masalah (*Intelligence*) atau komputer yang harus *memahami* masalah yang harus dipecahkan. Dengan demikian, pengenalan prolog sangat penting bagi mahasiswa ilmu komputer (*Computer Science*) karena tidak ada jalan lain kecuali mengetahui bagaimana gagasan pemrograman berjenis apa tersebut. Demikian menurut P.H. Winston, seorang pakar Lisp dari tempat kelahiran Lisp (MIT).[5]

2.4.6 Prolog dan Proyek Komputer Generasi Kelima

Prolog lahir di Eropa, tapi yang tidak juga kecil peranannya dalam membesarkan prolog adalah Proyek Komputer Generasi Kelima yang dicanangkan Jepang. Memang sejauh ini perkembangan teknologi komputer dari satu generasi ke generasi berikutnya hanyalah berkisar pada kemajuan peralatannya dari tabung hampa ke transistor, ke IC dan sekarang ke LSI. Gejala ini mengatakan bahwa dalam filosofi desain dan sasaran utama pemamfaatan komputer, tidak ada perubahan besar.

Dengan komputer generasi kelima, diharapkan bahwa perubahan generasi tidak saja melibatkan perubahan perangkat keras seperti penggunaan VLSI, tapi juga dalam perubahan filosofi desain dan bisang penerapannya. Demikian pernyataan Lembaga Komputer Generasi Baru Jepang (ICOT).

Lebih jauh ditegaskan bahwa perubahan tersebut melibatkan perubahan dari komputer numeric ke komputer yang dapat menafsirkan arti informasi dan dapat

memecahkan masalah.[6] Dari cita-cita besar ini, memang tidak salah untuk mengambil *Prolog* sebagai bahasa inti untuk komputer generasi kelima tersebut, selain dari sifat paralelnya.

2.4.7 Turbo Prolog dan Prolog Tradisional

Di awal kelahirannya, Prolog masih merupakan bahasa murni yang didasarkan pada keindahan logika. Prolog masih berkembang lambat dan hanya digunakan di kalangan peneliti di perguruan tinggi terkenal karena mereka menyadari arti pentingnya Prolog. Akhirnya mereka menemukan adanya kebutuhan untuk penggunaan praktis dan keampuhan pada prolog untuk bidang penelitian yang luas. Diawal tahun 1980-an, dunia industry menyadari kemajuan prolog terhadap bahasa lain. Disini muncul kebutuhan untuk memiliki prolog yang cepat dari bahasa lain, memori lebih kecil dan karakteristik praktis lainnya.

Dengan campur tangan dunia industri, muncul dua kubu pemrogram prolog dengan filosofi yang berbeda. Yang pertama tetap yakin bahwa Prolog adalah bahasa riset dan dipihak lain yakin bahwa Prolog akan menjadi bahasa yang ampuh untuk pengembangan program terapan. Turbo Prolog merupakan hasil perkembangan ini. Turbo Prolog menggabungkan keampuhan Prolog tradisional dengan kecepatan dan keampuhan bahasa lainnya. Perbedaan utama antara Turbo Prolog dengan Prolog tradisional adalah:[3]

1. Turbo Prolog merupakan kompiler Prolog bertype yakni semua objek dan relasi/relasi harus dideklarasikandalam program.

2. Dalam Turbo Prolog, metaprogramming bukan merupakan fasilitas jadi yang siap pakai tapi dapat dimodelkan.

   Menurut pembuatnya, alas an penting yang melatarbelakanginhal tersebut adalah:

   1. Peningkatan kecepatan yang sangat besar.
   2. Daya baca program lebih besar.
   3. Waktu untuk pengembangan program lebih pendek.
   4. Deteksi galat (error) lebih besar.
   5. Pemeliharaan program lebih murah.

2.4.8 Penerapan Prolog

Prolog sangat membantu dalam program untuk menirukan cara berfikir manusia dengan mesin (komputer) karena Prolog memiliki sifat atau cirri yang sesuai dengan metoda pemrograman kecerdasan buatan. Dengan kata lain Prolog sangat besar manfaatnya dalam kecerdasan buatan yang meliputi berbagai bidang:

a. *Sistem pakar*: Secara sederhana, sistem pakar adalah suatu program yang ditulis untuk dapat menirukan keahlian seorang pakar (dalam bidang keahlian yang sangat khusus) dalam menjawab persoalan, seperti mendiagnosa infeksi bakteri (MYCIN), mengidentifikasi suatu jenis binatang, untuk konsultasi, analisa, prediksi dan sebagainya. Sebagai bahasa yang berdasarkan pada pemrograman logika, Prolog lebih menguntungkan, baik segi representasi pengetahuan maupun cara pencarian/penarikan kesimpulan.

b. *Permainan*: Sekarang banyak beredar program untuk memecahkan masalah permainan seperti untuk bermain poker, bridge, catur dan sebagainya. Sebagai contoh, program catur Mephisto III yang pernah bermain seri dengan juara dunia Anatoly Karpov. Dalam catur misalnya, pemain harus mencari segala kemungkinan langkah terbaik untuk memenangkan permainan. Namun jika kita menjajagi semua kemungkinan langkah dan akibatnya, tentu akan lama. Ivan Bratko telah banyak mengupas tentang ***Advice Languanges*** yang dapat memberi saran untuk langkah berikutnya secara deklaratif. Saran-saran ini dinyatakan dalam sekumpulan goal dan cara mencapai goal tersebut.

c. *Searching atau pencarian jawaban*: Program untuk menyederhanakan rumus matematik, mencari jejak, atau membuktikan suatu kebenaran. Dalam hal ini masalah yang dihadapi biasanya adalah tidak diketahuinya algoritma pemecahannya. Sebagai bahasa deklaratif, Prolog tampil sebagai calon terbaik.

d. *Pengolahan bahasa alami*: Ini mencakup penterjemahan antar-bahasa, humor, percakapan dengan mesin, anlisa kalimat dan sebagainya. Kelebihan Prolog dalam hal ini adalah bahwa program untuk pengolahan bahasa akan lebih mudah diwujudkan dan lebih jelas secara logika sehingga mudah dalam pemeliharaan dan pengembangan programnya.

e. *Robot*: Ini mencakup bidang yang lebih luas seperti pengendalian gerak, percakapan robot (SHRDLU), sistem pakar robotik (ARGOS II). Prolog dipicu dengan suatu sasaran (***goal-driven***) sedangkan pemrograman dalam

robot pada umumnya lebih mudah jika diwujudkan secara goal-driven, seperti untuk mengambil kubus dengan bagian goal: *gerakkan manipulator kearah kubus.*

f. *Belajar* atau *Machine Learning*: Program yang dibuat agar komputer dapat mengambil pelajaran (pengalaman), misalnya *Hacker*. Program yang demikian, umumnya selalu memperbaiki atau menambah data untuk peningkatan performansi dalam hal kecerdikan memberi jawaban. Prolog didukung dengan fasilitas untuk dapat mengubah fakta dan aturan pada saat program berjalan, yang sesuai dengan cara kita belajar.

g. *Pengenalan Citra* (Pattern Recognition): Ini mengusahakan agar komputer dapat mengenal bentuk yang agak berbeda seperti membaca tulisan tangan. Ini bermanfaat misalnya dalam komputer **palmtop** yang tidak memerlukan keyboard. Dalam pengolahan visual, nampaknya lebih menguntungkan jika memanfaatkan paralelisme, deklaratif dan pemadanan Prolog.

Meskipun Prolog merupakan bahasa deklaratif, dapat pula ditafsirkan dari segi procedural, sehingga dapat mencakup bidang penerapan yang lebih luas. Selain penerapan seperti diatas, Prolog khususnya Turbo Prolog, dapat pula digunakan untuk:[3]

- Pengaturan dan pemantauan proses dalam industri.
- Mewujudkan program basis data yang canggih.

- Membuat program aplikasi untuk bisnis.
- Menulis (membuat) kompiler.
- Merencanakan dan penjadwalan projek, kegiatan, kurikulum dan sebagainya.
- Pendidikan seperti sejarah, hokum ilmu komputer, logika matematik, kedokteran dan sebagainya.
- Pemodelan dan simulasi