BAB II

LANDASAN TEORI

2.1 Kecerdasan Buatan

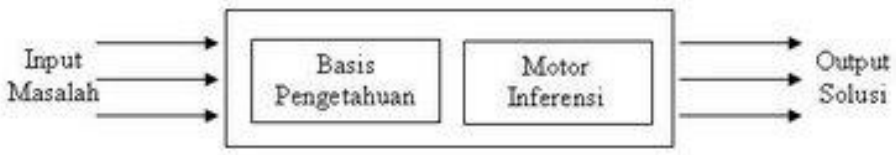
Secara umum Kecerdasan Buatan atau *Artificial Intelligence* merupakan salah satu bagian komputer yang membuat agar mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia [3]. Pada aal diciptakannya, komputer hanya difungsikan sebagai alat hitung saja. Namun seiring dengan perkembangan jaman, maka peran komputer semakin mendominasi kehidupan umat manusia. Komputer tidak lagi hanya digunakan sebagai alat hitung, lebih dari itu, komputer diharapkan untuk diberdayakan untuk mengerjakan segala sesuatu yang bisa dikerjakan oleh manusia.

Kecerdasan Buatan termasuk bidang ilmu yang relatif muda. Pada tahun 1950-an para ilmuwan dan peneliti mulai memikirkan bagaimana caranya agar mesin dapat melakukan pekerjaannya seperti yang bisa dikerjakan oleh manusia. Alan Turing, seorang matematikawan dari Inggris pertama kali mengusulkan adanya tes untuk melihat bisa tidaknya sebuah mesin dikatakan cerdas. Hasil test tersebut dikenal dengan *Turing Test*, dimana mesin tersebut menyamar seolah-olah sebagai seseorang di dalam suatu permainan yang mampu memberikan respon terhadap serangkaian pertanyaan yang diajukan.

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

sendiri dimuncilkan oleh seorang Professor dari *Masschusetts Institutr of Technology* yang bernama John McCarthy pada tahun 1956 pada Dartmouth Conference yang dihadiri oleh para peneliti *artificial intelligence*. Pada konferensi tersebut juga didefinisikan tujuan utama dari kecerdasan buatan, yaitu mengetahui dan memodelkan proses-proses berfikir manusia dan mendesain mesin agar dapat menirukan kelakuan manusia tersebut.

Kecerdasan buatan bertujuan untuk mengembangkan suatu program komputer yang dapat memecahkan suatu masalah tertentu, dengan cara yang dapat dianggap cerdas seperti layaknya manusia. Sehingga dengan perancangan kecerdasan buatan yang baik, diharapkan peran manusia dapat diminimalisasi dan dapat meringankan beban kerja manusia. *Artificial Intelligence* bekerja dengan menerima *input*, diproses dan kemudian menghasilkan *output* yang berupa suatu keputusan atau solusi. Konsep kecerdasan dapat dilihat pada gambar 2.1 dibawah ini.



Gambar 2.1 Konsep Kecerdasan Buatan di Komputer

Lingkup utama implementasi dari kecerdasan buatan saat ini ditemui pada bidang-bidang berikut :

a. Sistem Pakar (*Expert System*) : yaitu metode *artificial intelligence* yang berguna untuk meniru cara berfikir dan penalaran seorang ahli dalam mengambil keputusan berdasarkan situasi yang ada. Dengan

expert system seorang *user* biasa dapat melakukan konsultasi kepada komputer, seolah-olah *user* tersebut berkonsultasi dengan seorang ahli. Contohnya adalah program aplikasi yang mampu meniru seorang ahli medis dalam mendeteksi demam berdarah berdasarkan keluhan-keluhan pasiennya.

b. Pengenalan Ucapan (*Speech Recognition*) : yaitu metode *artificial intelligence* yang berguna untuk mengenali suara manusia dengan mencocokkan dengan acuan yang telah diatur sebelumnya. Contohnya adalah suara dari *user* dapat diterjemahkan menjadi sebuah perintah bagi komputer.
c. *Game Playing* : yaitu metode *artificial intelligence* yang meniru cara berfikir manusia dalam bermain *game.* Contohnya adalah program *Perfect Chessmate* yang mampu berfikir setara dengan seorang *grandmaster* catur.
d. *Computer Vision* : yaitu suatu metode *artificial intelligence* yang memungkinkan sebuah sistem komputer mengenali gambar sebagai inputnya. Contohnya adalah mengenali dan membaca tulisan yang ada pada gambar.
e. Pengolahan Bahasa Alami (*Natural Language Processing*) : adalah suatu metode *artificial intelligence* yang memungkinkan *user* dapat berkomunikasi dengan komputer dengan menggunakan bahasa sehari-hari.

f. Logika Fuzzy (*Fuzzy Logic*) : adalah metode *artificial intelligence* yang banyak terdapat pada alat-alat elektronik dan robotika. Dimana alat-alat elektronik tersebut mampu berfikir dan bertingkah laku sebagaimana layaknya manusia.

2.2 Sistem Pakar

Sistem pakar pertama kali dikembangkan oleh komunitas *Artificial Intelligence* pada pertengahan tahun 1960. Sistem pakar yang muncul pertama kali adalah *General-purpose Problem Solver* (GPS) yang dikembangkan oleh Newel dan Simon. Ketika hendak membuat suatu keputusan yang kompleks atau memecahkan masalah, seringkali kita meminta nasehat atau berkonsultasi dengan seorang pakar atau ahli.

Seorang pakar adalah seseorang yang mempunyai pengetahuan dan pengalaman spesifik dalam suatu bidang; misalnya pakar komputer, pakar uji tak merusak, pakar politik dan lain-lain. Semakin tidak terstruktur situasinya, semakin mengkhusus konsultasi yang dibutuhkan.

2.2.1 Pengertian Sistem Pakar

Sistem Pakar atau *expert system* adalah salah satu cabang dari *artificial intelligence* yang berusaha mengadopsi pengetahuan manusia ke komputer yang dirancang untuk memodelkan kemampuan menyelesaikan masalah seperti layaknya seorang pakar (*human expert*). Adapun beberapa definisi tentang sistem pakar (*expert system*) lainnya, adalah sebagai berikut :

a. Menurut Marimin [4], sistem pakar didefinisikan sebagai berikut :

   "*Sistem pakar adalah sistem perangkat lunak komputer yang menggunakan ilmu, fakta, dan teknik berpikir dalam pengambilan keputusan untuk menyelesaikan masalah-masalah yang biasanya hanya dapat diselesaikan oleh tenaga ahli dalam bidang yang bersangkutan".*

b. Menurut Sri [2], sistem pakar didefinisikan sebagai berikut :

   "Sistem pakar adalah suatu sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli".

c. Menurut Muhammad [3], sistem pakar didefinisikan sebagai berikut :

   "*Sistem pakar adalah salah satu cabang dari artificial intelligence yang membuat penggunaan secara luas knowledge yang khusus untuk menyelesaikan masalah tingkat manusia sebagai pakar".*

Seorang pakar adalah orang yang mempunyai keahlian dalam bidang tertentu, yaitu pakar yang mempunyai *knowledge* atau kemampuan khusus yang orang lain tidak mengetahui atau mampu dalam bidang yang dimilikinya [3]. Dengan sistem pakar ini, orang awam pun dapat menyelesaikan masalahnya atau hanya sekedar mencari suatu informasi berkualitas yang sebenarnya hanya dapat diperoleh dengan bantuan para ahli di bidangnya.

Sistem pakar ini juga akan dapat membantu aktivitas para pakar sebagai asisten yang berpengalaman dan mempunyai asisten yang berpengalaman dan mempunyai pengetahuan yang dibutuhkan. Dalam penyusunannya, sistem pakar mengkombinasikan kaidah-kaidah penarikan kesimpulan (*inference rules*) dengan

basis pengetahuan tertentu yang diberikan oleh satu atau lebih pakar dalam bidang tertentu. Kombinasi dari kedua hal tersebut disimpan dalam komputer, yang selanjutnya digunakan dalam proses pengambilan keputusan untuk penyelesaian masalah tertentu. Salah satu metode yang paling umum untuk merepresentasikan pengetahuan adalah dalam bentuk tipe aturan (*rule*) *IF…THEN* (Jika..maka).

2.2.2 Konsep Dasar Sistem Pakar

Menurut Efraim Tuban, konsep dasar dari suatu sistem pakar mengandung beberapa unsur, yaitu keahlian, pengalihan keahlian, inferensi, aturan dan kemampuan untuk menjelaskan. Umumnya sistem pakar adalah sistem berbasis aturan, yaitu pengetahuan yang terdiri dari aturan-aturan sebagai prosedur penyelesaian masalah. Pengetahuan tersebut digambarkan sebagai suatu urutan seri dari kaidah-kaidah yang sudah dibuat. Keahlian adalah suatu kelebihan penguasaan pengetahuan dibidang tertentu yang diperoleh dari pelatihan, membaca atau pengalaman. Seorang ahli adalah seseorang yang mampu menjelaskan sesuatu tanggapan, mempelajari hal-hal baru seputar topic permasalahan (*domain*), menyusun kembali pengetahuan jika dipandang perlu, memecah aturan-aturan jika dibutuhkan dan menentukan relevan tidaknya keahlian mereka.

Pengalihan keahlian dari para ahli ke komputer untuk kemudian dialihkan ladi ke orang lain yang bukan ahli, merupakan tujuan utama sistem pakar. Proses ini membutuhkan 4 aktifitas yaitu tambahan pengetahuan dari para ahli atau sumber-sumber lainnya, representasi pengetahuan ke komputer, inferensi

pengetahuan dan pengalihan pengetahuan ke *user*. Pengetahuan yang disimpan di komputer disebut dengan nama basis pengetahuan (*knowledge base*).

Salah satu fitur yang harus dimiliki oleh sistem pakar adalah kemampuan untuk menalar. Jika keahlian-keahlian sudah tersimpan sebagai basis pengetahuan dan sudah tersedia program yang mampu mengakses basis data, maka komputer harus dapat deprogram untuk membuat inferensi. Proses inferensi ini dikemas dalam bentuj motor inferensi (*inference engine*).

Kunci kekuatan sistem pakar adalah basis pengetahuan (*knowledge base*). Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi dan penyelesaian masalah yang terdiri dari heuristic dan sekumlah *rule* yang tersusun secara sistematis dan spesifik. *Knowledge base* tersebut dapat disimpan dalam sebuah basis data.

Sedangkan yang menjadi pusat dalam pemrosesannya adalah *Inference engine,* yaitu suatu rancangan aplikasi yang berfungsi memberikan pertanyaan dan menerima *input* dari *user,* kemudian melakukan proses logika sesuai dengan *knowledge base* yang telah ada, untuk selanjutnya menghasilkan *output* berupa kesimpulan atau solusi dari permasalahan.

Untuk itu, langkah-langkah perancangan sistem pakar sebaiknya mengikuti urutan sebagai berikut :

1. Mengidentifikasi masalah dan kebutuhan
2. Menentukan permasalahan atau problema yang cocok
3. Mempertimbangkan alternatif.
4. Menghitung pengembalian investasi

5. Memilih alat pengembangan.
6. Merekayasa pengetahuan.
7. Merancang sistem.
8. Melengkapi pengembangan.
9. Menguji dan mencari kesalahan sistem.
10. Memelihara sistem.

2.2.3 Ciri-ciri Sistem Pakar

Sistem pakar memiliki ciri-ciri sebagai berikut [3] :

1. Terbatas pada bidang keahlian yang spesifik.
2. Dapat mengemukakan rangkaian alasan-alasan yang diberikan dengan cara yang dapat dipahami.
3. Dapat memberikan penalaran untuk data yang tidak lengkap atau tidak pasti.
4. Berdasarkan *rule* atau kaidah tertentu.
5. Dirancang untuk dikembangkan secara bertahap.
6. Keluaran bersifat anjuran atau nasehat.
7. Keluaran tergantung dari dialog dengan *user*.

2.2.4 Orang Yang Terlibat Dalam Sistem Pakar

Untuk memahami perancangan sistem pakar, perlu dipahami mengenai siapa saja yang berinteraksi dengan sistem. Orang-orang tersebut adalah :

1. Pakar (*domain expert*) adalah seseorang ahli yang dapat menyelesaikan masalah yang sedang diusahakan untuk dipecahkan oleh sistem.
2. Pembangun pengetahuan (*knowledge engineer*) merupakan seseorang yang menerjemahkan pengetahuan seorang pakar dalam bentuk deklaratif sehingga dapat digunakan untuk memecahkan masalah.
3. Pengguna (*user*) merupakan seseorang yang berkonsultasi dengan sistem untuk mendapatkan saran yang disediakan oleh pakar.

Pembangun sistem (*system engineer*) merupakan seseorang yang membuat antarmuka, merancang bentuk basis pengetahuan secara deklaratif dan mengimplementasikan mesin inferensi.

2.2.5 Kategori Masalah Sistem Pakar

Masalah-masalah yang dapat diselesaikan dengan sistem pakar adalah sebagai berikut :

1. Interpretasi, yaitu membuat kesimpulan atau deskripsi dari sekumpulan data mentah.
2. Proyeksi, yaitu memprediksi akibat-akibat yang dimungkinkan dari situasi-situasi tertentu, diantaranya peramalan, prediksi demografis, peramalan ekonomi, dan prediksi lalu lintas.
3. Diagnosis, yaitu menentukan malfungsi dalam situasi kompleks yang didasarkan pada gejala-gejala yang teramat, diantaranya medis, elektronis, mekanis dan diagnosis perangkat lunak.

4. Desain, yaitu menentukan konfigurasi komponen-komponen sistem yang cocok dengan tujuan-tujuan kinerja tertentu yang memenuhi kendala-kendala tertentu, diantaranya layout sirkuit dan perancangan bangunan.
5. Perencanaan, yaitu merencanakan serangkaian tindakan yang akan dapat mencapai sejumlah tujuan dengan kondisi awal tertentu, diantaranya perencanaan keuangan, komunikasi, militer, routing dan manajemen proyek.
6. *Monitoring*, yaitu membandingkan tingkah laku suatu sistem yang teramati dengan tingkah laku yang diharapkan darinya, diantaranya *Computer Aided Monitoring System.*
7. *Debigging* dan *repair,* yaitu menentukan dan mengimplementasikan cara untuk mengatasi manfungsi, diantaranya memberikan resep obat terhadap suatu kegagalan, diantaranya memberikan resep obat terhadap suatu kegagalan.
8. Instruksi, yaitu mendeteksi dan mengoreksi dalam pemahaman domain subjek, diantaranya melakukan instruksi untuk diagnosis, *debugging* dan perbaikan kerja.
9. Pengendalian, yaitu mengatur tingkah laku suatu *environtment* yang kompleks seperti control terhadap interpretasi-interpretasi, prediksiperbaikan dan monitoring kelakuan sistem.
10. Seleksi, mengidentifikasi pilihan terbaik dari sekumpulan (*list*) kemungkinan.

11. Simulasi, yaitu pemodelan interaksi diantara komponen-komponen sistem.

2.2.6 Keuntungan Sistem Pakar

Sistem pakar (*expert system*) merupakan paket perangkat lunak atau paket program komputer yang bertujuan sebagai penyedia nasehat dan sarana bantu dalam memecahkan masalah di bidang-bidang spesialisasi tertentu seperti sains, perekayasa, matematika, kedokteran, pendidikan dan sebagainya. Sistem pakar merupakan subset dari *Artificial Intelligence.*

Berikut ini merupakan keuntungan bila menggunakan sistem pakar, diantaranya adalah :

1. Menjadikan pengetahuan dan nasihat lebih mudah didapat.
2. Mengingkatkan *output* dan produktivitas.
3. Meningkatkan menyelesaian masalah, menerusi paduan pakar, penerangan, sistem pakar khas.
4. Meningkatkan reliabilitas.
5. Menyimpan pengetahuan dan keahian pakar.
6. Memberikan jawaban yang tepat
7. Merupakan panduan yang *intelligence* (cerdas).
8. Dapat bekerja dengan informasi yang kurang lengkap dan mengandung ketidakpastian.
9. *Intelligence database* (basis data cerdas), bahwa sistem pakar dapat digunakan untuk mengakses basis data dengan cara cerdas.

2.2.7 Keunggulan Sistem Pakar

Sistem pakar sebegai sebagai program komputer memiliki keunggulan bila dibandingkan dengan sistem konvensional. Berikut ini merupakan beberapa keunggulan sistem pakar , diantaranya adalah dapat :

1. Menghimpun data dalam jumlah yang sangat besar.
2. Menyimpan data tersebut untuk jangka waktu yang panjang dalam suatu bentuk tertentu.
3. Mengerjakan perhitungan secara cepat dan tepatdan tanpa jemu mencari kembali data yang tersimpan dengan kecepatan tinggi.

Terdapat perbedaan yang yang cukup signifikan dari konsep pengembangan perangkat lunak bila dibandingkan antara sistem pakar dengan sistem konvensional, diantaranya adalah sebagai berikut [3] :

Tabel 2.1 Perbandingan Sistem Konvensional dengan Sistem Pakar

Sistem Konvensional	Sistem Pakar
Informasi dan pemrosesannya biasanya jadi satu dengan program	Basis pengetahuan merupakan bagian terpisah dari mekanisme inferensi
Tidak ada penjelasan mengapa data input diperlukan, kesimpulan dihasilkan	Penjelasan bagian dari sistem pakar
Eksekusi dilakuan dalam basis algoritma, tahap demi tahap	Eksekusi dilakukan pada keseluruhan basis pengetahuan
Tujuan utamanya adalah efisiensi	Tujuan utamanya adalah efektivitas

2.2.8 Kelemahan Sistem Pakar

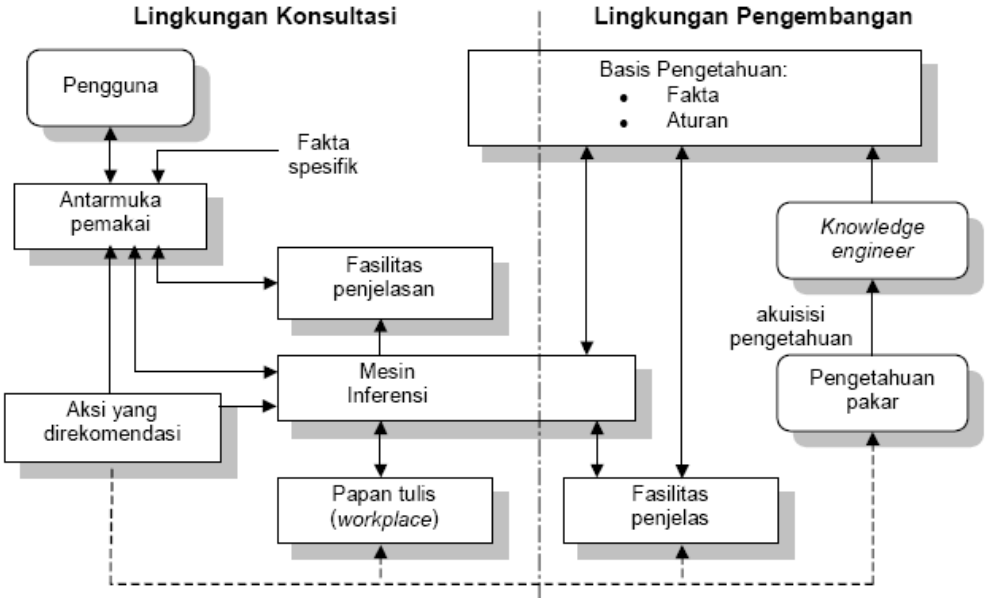
Selain keuntungan dan keunggulan tersebut, sistem pakar halnya sistem lainnya, juga memiliki kelemahan, diantaranya adalah :

a. Pengetahuan yang hendak diambil tidak selalu tersedia.
b. Kepakaran sangat sulit diekstrak dari manusia.
c. Pendekatan oleh setiap pakar untuk suatu situasi atau problem bisa berbeda-beda, meskipun sama-sama benar.
d. Adalah sangat sulit bagi seorang pakar untuk mengabstraksi atau menjelaskan langkah mereka dalam menangani masalah.
e. Pengguna Sistem Pakar mempunyai batas kognitif alami, sehingga mungkin tidak bisa memanfaatkan sistem secara maksimal.
f. Sistem Pakar bekerja baik untuk suatu bidang yang sempit.
g. Banyak pakar yang tidak mempunyai jalan untuk mencek apakah kesimpulan mereka benar dan masuk akal.
h. Boleh jadi sistem tidak dapat membuat keputusan, karena memungkinkan penarikan kesimpulan dengan gejala yang berbenturan.
i. Sistem pakar tidaklah 100% menguntungkan, walaupun seorang tetap tidak sempurna atau tidak selalu benar. Oleh karena itu perlu diuji ulang secara teliti sebelum digunakan.

Kelemahan-kelemahan dari sistem pakar tersebut bukanlah sama sekali tidak bisa diatasi, tetapi dengan terus melakukan perbaikan sistem dan perancangan konsep penelusuran yang digunakan berdasarkan pengalaman yang telah ada maka hal itu diyakini akan dapat teratasi, walaupun dalam waktu yang panjang dan terus menerus.

2.2.9 Struktur Sistem Pakar

Sistem pakar terdiri dari dua bagian utama yaitu *development environment* dan *consultation environment. Development environment* digunakan untuk memasukkan pengetahuan pakar ke dalam lingkungan sistem pakar, sedangkan *consultation environment* digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar. Komponen-komponen sistem pakar dalam kedua bagian tersebut dapat dilihat pada gambar 2.2 berikut.



Gambar 2.2 Struktur Sistem Pakar

Komponen-komponen yang terdapat dalam sistem pakar adalah seperti yang terdapat pada Gambar 2.2, yaitu *User Interface* (antarmuka pengguna), basis pengetahuan, akuisisi pengetahuan, mesin inferensi, *workplace,* fasilitas penjelasan dan perbaikan pengetahuan.

2.2.9.1 Antarmuka Pengguna (*Knowledge Base*)

User interface merupakan mekanisme yang digunakan oleh pengguna dan sistem pakar untuk berkomunikasi. Antarmuka menerima informasi dari pemakai dan mengubahnya kedalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai.

Menurut McLeod (1995), pada bagia ini terjadi dialig antara program dan pemakai, yang memungkinkan sistem pakar menerima instruksi dan informasi (*input*) dari pemakai, juga memberikan informasi (*output*) kepada pemakai. Sistem pakar mengatur komunikasi antara pengguna dan komputer. Komunikasi ini paling baik berupa bahasa alami, biasanya disajikan dalam bentuk tanya-jawab dan kadang ditampilkan dalam bentuk gambar/grafik. Antarmuka yang lebih canggih dilengkapi dengan percakapan (voice communication).

2.2.9.2 Basis Pengetahuan (*Knowledge Base*)

Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi dan penyelesaian masalah. Komponen sistem pakar ini disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang objek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

Ada dua bentuk pendekatan berbasis pengetahuan yang sangat umum digunakan, yaitu [2]:

1. Penalaran berbasis aturan (*Rule-Based Reasoning*)

   Pada penalaran berbasis aturan, pengetahuan direpresentasikan dengan menggunakan aturan berbentuk *IF-THEN*. Bentuk ini digunakan apabila kita memliki sejumlah pengetahuan pakar pada suatu permasalahan tertentu, dan pakar dapat menyelesaikan masalah tersebut secara berurutan. Disamping itu, bentuk ini juga digunakan apabila dibutuhkan penjelasan tantang langkah-langkah mencapai solusi.

2. Penalaran berbasis kasus (*Case-Based Reasoning*)

   Apada panalaran berbasis kasus, basis pengetahuan akan berisi solusi-solusi yang telah dicapai sebelumnya, kemudian akan diturunkan suatu solusi untuk keadaan yang terjadi sekarang (fakta yang ada). Bentuk ini digunakan apabila *user* menginginkan unuk tahu lebih banyak lagi pada kasus-kasus yang hamper sama.

Dalam studi kasus pada sistem berbasis pengetahuan terdapat beberapa karakteristik yang dibangun untuk membantu arsitekturnya. Prinsip tersebut meliputi :

1. Pengetahuan merupakan kunci kekuatan sistem pakar.
2. Oengetahuan sering tidak pasri dan tidak lengkap.
3. Pengetahuan sering miskin spesifikasi.
4. Amatir menjadi ahli secara bertahap.
5. Sistem pakar harus fleksibel.
6. Sistem pakar harus transparan.

Sejarah penelitian di bidan *artificial intelligence* telah menunjukkan berulang kali bahwa pengetahuan adalah kunci untuk setiap sistem cerdas (*intelligence system*).

2.2.9.3 Akuisisi Pengetahuan (*Knowledge Acquisition*)

Akuisisi pengetahuan adalah akumulasi, transfer dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan ke dalam komputer. Dalam tahap ini *knowledge engineer* berusaha menyerap pengetahuan untuk selanjutnya ditransfer ke dalam basis pengetahuan. Pengetahuan diperoleh dari pakar, dilengkapi dengan buku, basis data, laporan penelitian dan pengalaman pemakai.

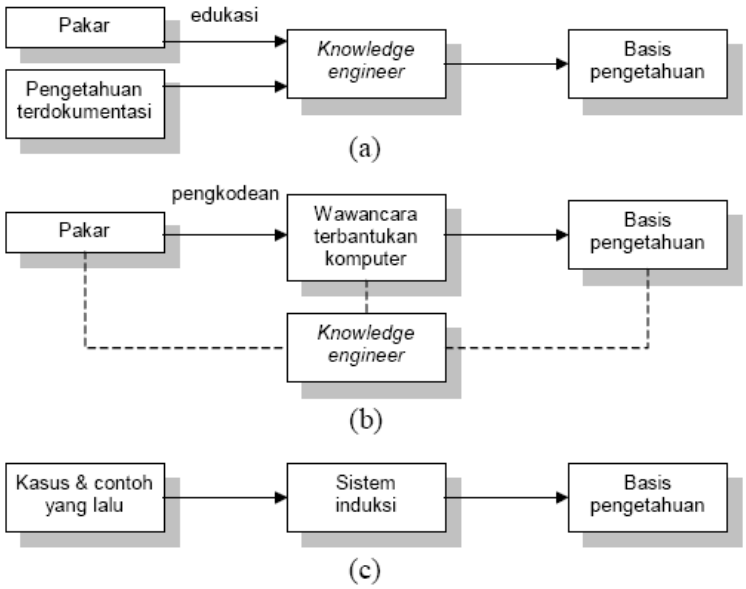
Dalam proses akuisisi pengetahuan, seorang perekayasa pengetahuan menjembatani antara pakar dengan basis pengetahuan. Perekayasa pengetahuan mendapatkan pengetahuan dari pakar, mengolahnya bersama pakar tersebut, dan menaruhnya dalam basis pengetahuan, dengan format tertentu. Pengambilan pengetahuan dari pakar dapat dilakukan secara :

1. Manual, dimana perekayasa pengetahuan (*knowledge engineer*) mendapatkan pengetahuan dari pakar (melalui wawancara) atau sumber lain, kemudian mengkodekannya kedalam basis pengetahuan. Proses ini biasanya berlangsung lambat, mahal, serta kadangkala tidak akurat.
2. Semi-otomatik, dimana terdapat peran komputer untuk mendukung pakar dengan mengijinkannya membangun basis pengetahuan dengan sedikit

bantuan dari perekayasa pengetahuan, atau membantu perekayasa pengetahuan sehingga kerjanya menjadi lebih efisien dan efektif.

3. Otomatik, dimana peran pakar, perekayasa pengetahuan dan pembangun basis pengetahuan (*system builder*) digabung. Misalnya dapat dilakukan oleh *system analys* seperti pada metode induksi.

Metode akuisisi pengetahuan tersebut dapat dilihat pada gambar 2.3.



Gambar 2.3 Metode Akuisisi Pengetahuan

2.2.9.4 Mesin Inferensi (*Inference Engine*)

Mesin inferensi merupakan otak dalam suatu sistem pakar. Komponen ini mengandung mekanisme pola piker dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi adalah program komputer yang memberikan metodologi untuk penalaran tantang informasi yang ada dalam

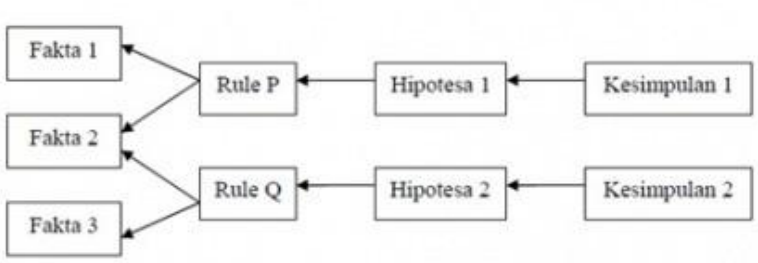
basis pengetahuan dan dalam *workplace*, dan untuk memformulasikan kesimpulan.

Kebanyakan sistem pakar berbasis aturan menggunakan strategi inferensi yang dinamakan modus ponen. Berdasarkan strategi ini, jika terdapat aturan "*IF* A *THEN* B", dan jika diketahuin bahwa A benar, maka dapet disimpulkan B juga benar.

Terdapat dua pendekatan untuk mengontrol inferensi dalam sistem pakar berbasis aturan, yaitu pelacakan ke belakang (*backward chaining*) dan pelacakan de depan (*forward chaining*).

2.2.8.4.1 Pelacakan ke Belakang (*Backward Chaining*)

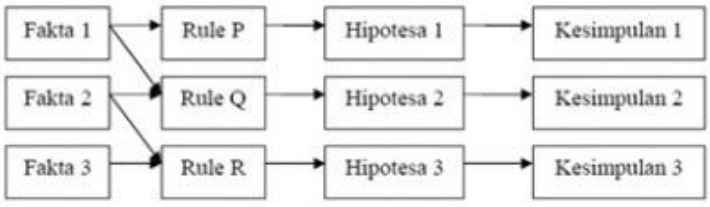
Pelacakan ke belakang adalah pendekatan yang dimotori tujuan (*goal-driven*). Dalam pendekatan ini pelacakan dimulai dari tujuan, selanjutnya dicari aturan yang memiliki tujuan tersebut untuk kesimpulannya. Selannjutnya proses pelacakan menggunakan premis untuk aturan tersebut sebagai tujuan baru dan mencari aturan lain dengan tujuan baru sebagai kesimpulannya. Proses berlanjut sampai semua kemungkinan ditemukan. Metode pelacakan *backward chaining* dapat dilihat pada gambar 2.4.



Gambar 2.4 Proses Pelacakan *Backward Chaining*

2.2.8.4.1 Pelacakan ke Depan (*Forward Chaining*)

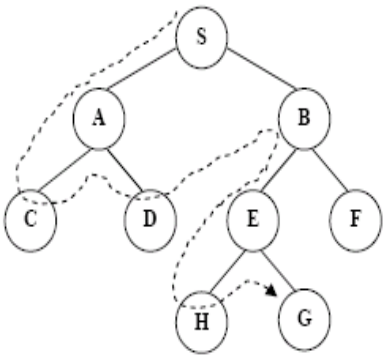
Pelacakan ke depan adalah pendekatan yang dimotori data (*data-driven*). Dalam pendekatan ini pelacakan dimulai dari informasi masukan, dan selanjutnya mencoba menggambarkan kesimpulan. Pelacakan ke depan mencari fakta yang sesuai dengan bagian *IF* dari aturan *IF-THEN*. Metode pelacakan *forward chaining* dapat dilihat pada gambar 2.5.



Gambar 2.5 Proses Pelacakan *Forward Chaining*

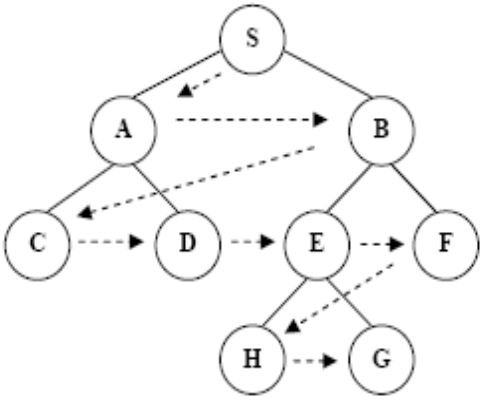
Kedua metode inferensi tersebut dipengaruhi oleh tiga macam yang penelusuran, yaitu *Depth-first search, Breadth-first search* dan *Best-firsth search.*

a. *Depth-first search,* melakukan penelusuran kaidah secara mendalam dari simpul akar bergerak menurun ke tingkat dalam yang berurutan. Penelusuran *depth-first search* dapat dilihat pada gambar 2.6.



Gambar 2.6 Diagram Alir Teknik Penelusuran *Depth First Search*

b. *Breadth-first search,* bergerak dari simpul akar, simpul yang ada pada setiap tingkat diuji sebelum pindah ke tingkat selanjutnya. Penelusuran *breadth-first search* dapat dilihat pada gambar 2.7.



Gambar 2.7 Diagram Alir Teknik Penelusuran *Breadth First Search*

c. *Best-first search,* bekerja berdasarkan kombinasi kedua metode sebelumnya.

2.2.9.5 Workplace

Workplace merupakan area dari sekumpulan memori kerja (*working memory*). *Workplace* digunakan untuk merekam hasil-hasil antara dan kesimpulan yang dicapai. Ada 3 tipe keputusan yang dapat direkam, yaitu :

1. Rencana : bagaimana menghadapi masalah.
2. Agenda : aksi-aksi yang potensial yang sedang menunggu untuk dieksekusi.

Solusi : calon aksi yang akan dibangkitkan.

2.2.9.6 Fasilitas Penjelasan

Fasilitas penjelasan adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar. Komponen ini menggambarkan penalaran sistem kepada pemakai (*user*). Fasilitas penjelasan dapat menjelaskan perilaku sistem pakar dengan menjawab pertanyaan-pertanyaan sebagai berikut :

1. Mengapa pertanyaan tertentu ditanyakan oleh sistem pakar?
2. Bagaimana kesimpulan tertentu diperoleh?
3. Mengapa alternative tertentu ditolak?
4. Apa rencana untuk memperoleh penyelesaian?

2.2.9.7 Perbaikan Pengetahuan

Pakar memiliki kemampuan untuk menganalisis dan meningkatkan kinerjanya serta kemampuan untuk belajar dari kinerjanya. Kemampuan tersebut adalah penting dalam pembelajaran terkomputerisasi, sehingga program akan mampu menganalisis penyebab kesiksesan dan kegagalan yang dialaminya.

2.2.10 Logika Fuzzy

Konsep logika *fuzzy* pertama kali diperkenalkan oleh Profesor Lotfi A. Zadeh dari Universitas Califirnia, pada bulan Juni 1965. Logika *fuzzy* merupakan generalisasi dari logika klasik yang hanya memiliki dua nilai keanggotaan, yaitu 0 dan 1. Dalam logika *fuzzy*, nilai kebenaran suatu pernyataan berkisar dari sepenuhnya benar sampai sepenuhnya salah, atau dalam rentang 0 hingga 1.

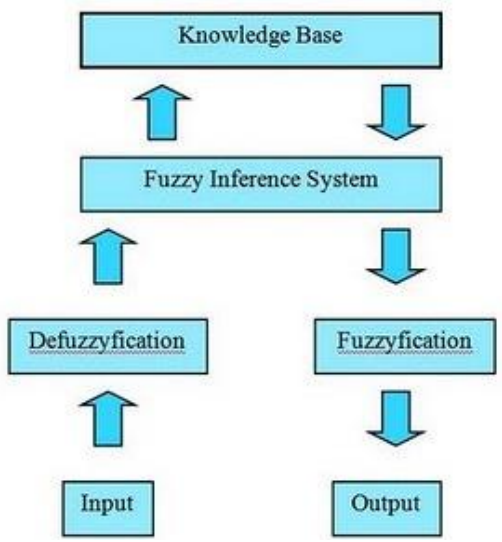
Logika *fuzzy* digunakan untuk menerjemahkan suatu besaran yang diekspresikan menggunakan bahasa (*linguistic*), misalkan besaran kecepatan laju kendaraan yang diekspresikan dengan pelan, agak cepat, cepat dan sangat cepat. Dengan teori himpunan *fuzzy*, suatu objek dapat menjadi anggota dari banyak himpunan dengan derejat keanggotaan yang berbeda dalam masing-masing himpunan.

Konsep ini berbeda dengan teori himpunan klasik (*crisp*). Teori himpunan klasik tergantung pada logika dua nilai (*two-valued logic*) untuk menentukan apakah sebuah objek merupakan suatu himpunan atau bukan.

Secara umum dalam sistem logika *fuzzy* terdapat empat buah elemen dasar, yaitu :

1. Basis kaidah (*rule base*), yang berisi aturan-aturan secara linguistik yang bersumber dari para pakar.
2. Suatu mekanisme pengambilan keputusan (*inference engine*), yang memperagakan bagaimana para pakar mengambil suatu keputusan dengan menerapkan pengetahuan (*knowledge*).
3. Proses fuzzifikasi (*fuzzification*), yang mengubah besaran tegas (*crisp*) ke besaran *fuzzy*.
4. Proses defuzzifikasi (*defuzzification*), yang mengubah besaran *fuzzy* hasil dari *inference engine*, menjadi besaran tegas (*crisp*).

Adapun skema dasar logika *fuzzy* dapat dilihat pada gambar 2.8.



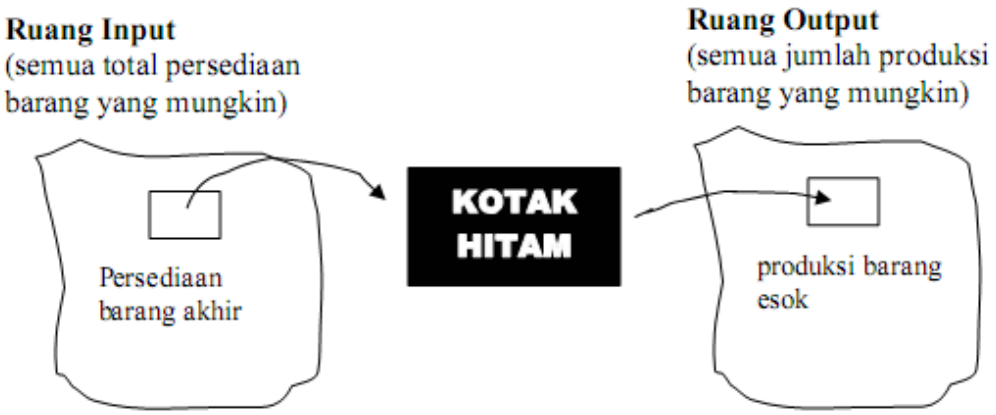
Gambar 2.8 Skema Dasar Logika *Fuzzy*

Logika fuzzy adalah suatu cara yang tepat untuk memetakan suatu ruang *input* ke dalam suatu ruang *output*. Contoh pemetaan *input-output* dalam logika *fuzzy* bisa dilihat pada gambar 2.9.



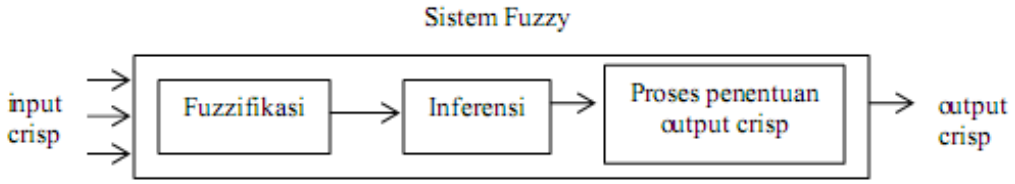
Gambar 2.9 Contoh Pemetaan *Input-Output* Logika *Fuzzy*

Antara *input* dan *output* terdapat suatu kotak hitam yang harus memetakan *input* dan *output* yang sesuai. Contohnya pada pemetaan *input-output* pada masalah produksi, dimana diberikan data persediaan barang untuk mengetahui berapa jumlah barang yang harus diproduksi. Pemetaannya dapat dilihat pada gambar 2.10.



Gambar 2.10 Contoh Pemetaan *Input-Output* Logika *Fuzzy*

Menurut Havinga *et al* (1999), terdapat empat tahap dalam pembangunan sistem *fuzzy*, yaitu fuzzifikasi, inferensi, komposisi dan defuzzifikasi. Tahap pembangunan sistem *fuzzy* dapat dilihat pada gambar 2.11.



Gambar 2.11 Tahap Pembangunan Sistem *Fuzzy*

2.2.10.1 Fuzzifikasi

Dalam fuzzifikasi, variabel *input* (*crisp*) dari sistem *fuzzy* ditransfer ke dalam himpunan *fuzzy* untuk dapat digunakan dalam perhitungan nilai kebenaran dari premis pada setiap aturan dalam basis pengetahuan. Dengan demikian tahap ini mengambil nilai-nilai *crisp* dan menentukan derajat dimana nilai-nilai tersebut menjadi anggota dari setiap himpunan *fuzzy* yang sesuai. Setelah fungsi kenaggotaan dari nilai-nilai *crisp* ditentukan, selanjutnya nilai kebenaran dari premis dihitung.

Premis dari aturan dapat terdiri dari lebih dari satu proposisi yang dihubungkan dengan operasi seperti konjungsi (AND) dan disjungsi (OR). Untuk menghitung nilai kebenaran premis, operator *fuzzy* digunakan untuk memperoleh satu bilangan yang merepresentasikan hasil dari premis. Jika sebuah premis dari suatu aturan memiliki derajat kebenaran tidak nol maka aturan dikatakan terpicu (*fired*)

2.2.10.2 Inferensi

Inferensi diimplementasikan untuk masing-masing aturan dalam basis pengetahuan. Dalam inferensi, nilai kebenaran premis dari aturan-aturan yang terpicu digunakan untuk menentukan nilai kebenaran bagian kesimpulan dari aturan yang terpicu. Dengan demikian *input* untuk proses inferensi adalah nilai yang diberikan oleh premis, dan *output* adalah suatu himpunan *fuzzy*. Metode yang biasa digunakan dalam proses inferensi adalah *min* dan *product*.

Dalam metode inferensi *min*, fungsi kenaggotaan *output* dipotong pada ketinggian fungsi yang disesuaikan dengan nilai kebenaran dari premis. Dalam metode inferensi *product,* fungsi keanggotaan *output* diberi skala sesuai dengan nilai kebenaran dari premis (Havinga *et al,* 1999).

2.2.10.3 Komposisi

Komposisi adalah proses dimana himpunan *fuzzy* yang menyatakan *output* dari setiap aturan dikombinasikan bersama ke dalam sebuah himpunan *fuzzy*. Metode komposisi yang umum digunakan adalah *max* (*maximum*) dan *sum*.

Dalam komposisi *max,* himpunan *fuzzy* untuk *output* ditentukan dengan mengambil titik maksimum dari semua himpunan *fuzzy* yang dihasilkan oleh proses inferensi untuk masing-masing aturan.

Dalam komposisi *sum,* himpunan *fuzzy* untuk output ditentukan dengan mengambil penjumlahan titik dari semua himpunan *fuzzy* yang dihasilkan oleh proses inferensi untuk masing-masing aturan.

2.2.10.4 Defuzzifikasi

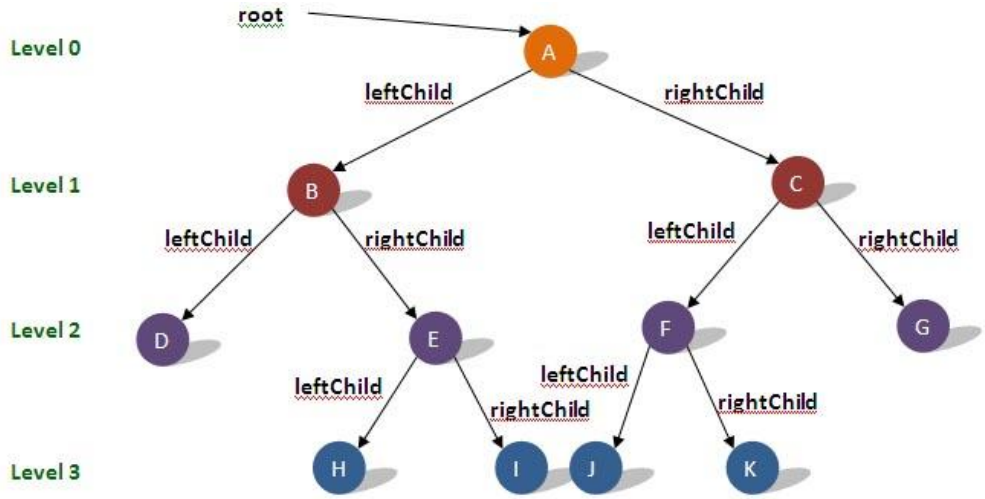
Input dari proses defuzzifikasi adalah himpunan *fuzzy* (yang dihasilkan dari proses komposisi) dan *output* adalah sebuah nilai (*crisp*). Untuk aturan *IF-THEN fuzzy* dalam persamaan (1), *defuzzifier* didefinisikan sebagai satu pemetaan dari himpunan *fuzzy* B^k dalam $V \in R$ (yang merupakan *output* dari inferensi *fuzzy*) ke titik *crisp* $y^* \in V$. terdapat tiga teknik yang paling umum digunakan, yaitu *center of gravity* (*centroid*) *deffuzifier, center average deffuzifier,* dan *maximum deffuzifier.*

Dalam *center of grafity* (*centroid*) *deffuzifier,* nilai *crisp* dari variabel *output* dihitung dengan menemukan nilai variabel dari pusat gravitasi dari fungsi kenaggotaan dari himpunan *fuzzy*. Dalam *maximum defuzzifier,* salah satu dari nilai-nilai variabel dimana subset *fuzzy* memiliki nilai kebenaran maksimum dipilih sebagai nilai *crisp* untuk variabel *output*.

2.3 Pohon (*Tree*)

Suatu pohon (*tree*) adalah suatu hierarki struktur yang terdiri dari *node* (*simpul*) yang menyimpan informasi atau pengetahuan dan cabang yang menghubungkan *node*. Cabang disebut juga *link* atau *edge* dan *node* disebut juga *vertex.* Gambar 2,7 menunjukkan *binary tree* yang mempunyai 0, 1 atau 2 cabang per *node.*

Dengan berorientasi pada *tree* (pohon), akar *node* adalah *node* yang tertinggi dalam hierarki dan daun adalah paling bawah. *Tree* dapat dianggap sebagai suatu tipe khusus dari jaringan semantic yang setiap simpulnya, kecuali akar pasti mempunyai satu simpul orang tua dan mempunyai nol atau lebih simpul anak. Contoh *tree* dapat dilihat pada gambar 2.12.



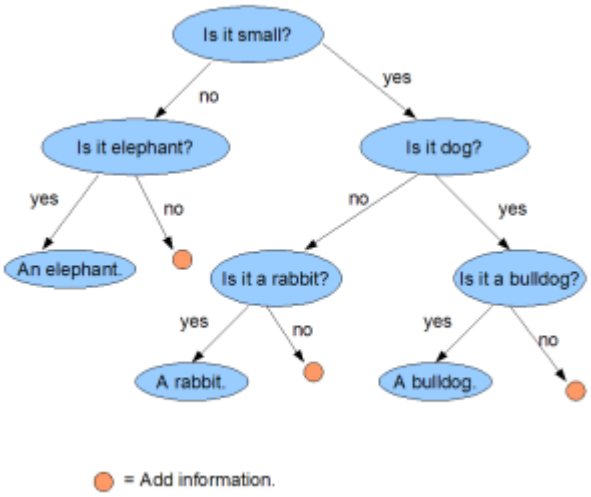
Gambar 2.12 Contoh *Tree*

Untuk tipe biasa dari *binary tree,* maksimum mempunyai dua anak untuk setiap simpul, dan sisi kiri dan kanan dari simpul anak dibedakan. Jika simpul mempunyai lebih dari datu orang tua maka disebut dengan jarringan. Gambar 2.7

menunjukkan hanya ada satu urutan dari *edge* atau *path* dari akar untuk tiap *node.* Oleh karena itu dalam hal ini tidak mungkin untuk memindahkan secara berlawanan dengan arah panah.

Tree merupakan kasus yang khusus dari matematika yang disebut dengan graph. Istilah jaringan atau jaringan sederhana sering sigunakan sebagai istilah sinonim dari graph bila menggambarkan suatu contoh kasus dari graph seperti jaringan telepon, graph dapat mempunyai nol atau lebih *link* di antara *node* dan tidak ada perbedaan antara orangtua dan anak.

Aplikasi dari *tree* adalah pembuatan keputusan dan biasa disebut dengan istilah *decision tree* (pohon keputusan). Contoh *decision tree* untuk klasifikasi hewan seperti yang ditunjukkan pada gambar 2.13. Simpul berisikan pertanyaan, cabanganya berisikan perkiraan hewan apa yang dijadikan kesimpulan.



Gambar 2.13 Contoh *Decision Tree*

Struktur keputusan dapat diterjemahkann secara mekanis kedalam kaidah produksi (*production rule*) dengan menggunakan aturan *IF-THEN Rules.*

berhubungan membentuk satu kesatuan sehingga tujuan atau sasaran sistem tersebut dapat tercapai.

Suatu sistem mempunyai maksud tertentu. Ada yang menyebutkan maksud dari suatu sistem adalah untuk mencapai suatu tujuan (*goal*) dan ada yang menyebutkan untuk mencapai sasaran (*objective*). Suatu sistem pada dasarnya merupakan suatu susunan yang teratur dari kegiatan yang berhubungan satu sama lainnya dan prosedur-prosedur yang berkaitan yang melaksanakan dan memudahkan pelaksanaan kegiatan utama dari suatu organisasi. [1]

2.4 Pengertian Basis Data

Basis data terdiri dari dua kata yaitu basis dan data. Basis dapat diartikan sebagai gudang tempat berkumpul. Sedangkan data adalah representasi fakta dunia nyata yang mewakili objek seperti manusia, barang, hewan, peristiwa konsep, yang dinyatakan dalam bentuk angka, huruf, simbol teks, gambar, bunyi atau kombinasinya.

Basis data dapat didefinisikan dari beberapa sudut pandang diantaranya sebagai berikut :

1. Sekumpulan data *persistence* (data disimpan *defile sekunder* atau data yang tahan lama) yang saling terkait, menggambarkan suatu organisasi (*enterprise*).
2. Himpunan kelompok data (arsip) nyang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.

3. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
4. Kumpulan *file* atau arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

2.4.1 Tujuan Basis Data

Adapun tujuan dari penggunaan basis data adalah sebagai berikut :

1. Memberikan kemudahan dan kecepatan dalam pengambilan kembali data
2. Efisiensi ruang dan waktu
3. Keakuratan data
4. Ketersediaan untuk proses pengambilan data yang diperlukan setiap saat
5. Kelengkapan data-data yang diperlukan atau yang tersimpan
6. Keamanan data
7. Kebersamaan.

2.4.2 Keuntungan Basis Data

Sedangkan keuntungan yang dapat diperoleh dari penggunaan basis data pada suatu aplikasi adalah sebagai berikut :

1. Mereduksi redudansi yang akibatnya mengurangi inkonsistensi
2. Data dapat di-*share* antar aplikasi
3. Standardisasi data dapat dilakukan
4. Batasan security dapat diterapkan

5. Mengelola integritas (keterjaminan akurasi) data
6. Menyeimbangkan kebutuhan yang saling konflik
7. Independensi data (objektif DBS) : kekebalan aplikasi terhadap perubahan struktur penyimpanan dan teknik pengaksesan data (basis data harus dapat berkembang tanpa mempengaruhi aplikasi yang telah ada).

2.5 *Tools* Analisis

Metode analisis data akan lebih mudah dilakukan jika menggunakan bantuan alat atau *tools* analisis. Adapun beberapa *tools* yang digunakan pada proses analisis ini adalah ERD (*Entity Relationship Diagram*), DFD (*Data Flow Diagram*), *Flowchart* dan Kamus Data.

2.5.1 ERD (Entity Relationship Diagram)

Model E-R didasarkan pada persepsi bahwa dunia nyata merupakan sekumpulan dari sejumlah objek dasar (entitas) dan relasi antar objek-objek dara tersebut. Diagram yang menggunakan struktur *logic* keseluruhan basis data, simbol yang digunakan adalah Persegi empat merepresentasikan himpunan entitas, Elips merepresentasikan atribut, Belah ketupat merepresentasikan himpunan keterhubungan dan Garis yang berfungsi untuk menghubungkan symbol-simbol pada diagram.

Label dari persegi empat, elips dan wajik menunjukkan nama, Kardinalitas pemetaan dinyatakan dengan dua cara, yaitu : [Korth] garis berarah (1) dan garis

tidak berarah (Banyak), [*Date*] menuliskan kardinalitasnya pada garis dan Peran dapat dituliskan sebagai label dari garis.

Pemakaian elemen-elemen dalam ERD ada tiga diantaranya adalah sebagai berikut :

1. Entitas (*Entity*) adalah sebuah objek yang dapat dibedakan dari objek-objek lainnya, yang memiliki sejumlah property atau atribut, dimana setiap atribut memiliki sekumpulan nilai yang diizinkan yang disebut domain, himpunan entitas yaitu kumpulan jumlah entitas yang memiliki tipe yang sama dan sebuah basis data mengandung sekumpulan himpunan entitas yang masing-masing memiliki sejumlah entitas dari tipe yang sama.
2. Relasi (*Relationship*) merupakan hubungan antar entitas yaitu sebuah relasi menggambarkan suatu asosiasi antar sejumlah entitas, himpunan relasi (*Relationship set*) adalah kumpulan sejumlah relasi yang memiliki tipe yang sama yang merupakan relasi matematis terhadap dua atau lebih himpunan entitas : $\{(e_{1,} e_{2,\ldots,}e_n)|\ (e_1 \in E_{1,} e_2 \in E_{2,\ldots,}e_n \in E_n)\}$, jumlah entitas terlihat dalam dua relasi disebut derajat. Kebanyakan relasi yang muncul adalah relasi binary, ada beberapa yang ternary, lebih dari itu sangat jarang. Fungsi sebuah entitas di dalam relasi peran (*role*) dan sebuah relasi dapat memiliki atribut.

2.5.2 DFD (Data Flow Diagram)

DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut akan disimpan. DFD merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur, selain itu merupakan alat yang cukup popular dikarenakan dapat menggambarkan arus data dalam didalam secara jelas dan terstruktur.

Dalam mengembangkan suatu aliran data atau proses yang terjadi didalam sistem data flow diagram menggunakan symbol-simbol yang memiliki arti ntersendiri dalam menerangkan :

a. *Eksternal Entity*

   Eksternal entity dapat merupakan kesatuan (*entity*) di lingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lainnya, yang memberikan *input-output* dari sistem.

b. Data Flow

   Arus data ini mengatur diantaranya proses, simpan data dan kesatuan luar. Arus data ini menunjukkan arus data berupa masukan sistem atau hasil proses sistem.

c. Proses

   Untuk *physical* data flow diagram (PDFD), data dilakukan oleh orang, mesin atau komputer. Sedangkan untuk *logical* data flow diagram (LDFD), suatu proses hanya menunjukkan proses dari komputer.

d. Penyimpanan Data

Simpanan data (*data store*) merupakan tempat penyimpanan data. Simpanan data dari DFD disimbolkan dengan sepasang garis horizontal parallel.

2.5.3 Flowchart

Flowchart merupakan gambaran dalam bentuk diagram alir dari algoritma-algoritma dalam suatu program, yang meyetakan arah alur program tersebut. Hal tersebut memungkinkan untuk memecah proses menjadi kejadian-kejadian individual atau ektifitas untuk menunjukkan secara singkat hubungan diantaranya.

Konstruksi *flowchart* memungkinkan pengertian lebih baik kepada proses dan pengertian yang lebih baik terhadap proses akan membawa kepada perbaikan pengembangan suatu sistem.

2.5.4 Kamus Data

Kamus data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi.

Dengan adanya kamus data, analisis dapat mendefinisikan data yang mengalir didalam sistem dengan lengkap dan dibuat berdasarkan arus data yang ada. Pada tahap perancangan sistem, kamus data dapat digunakan untuk merancang *input*, *output* dan perancangan *database* program.

2.6 *Database* MySQL

MySQL merupakan susunan salah satu konsep utama dalam *database,* yaitu SQL (*Structure Query Language*). Kendala dari suatu sistem *database* (DBMS) dapat diketahui dari cara kerja *optimizer*nya dalam melakukan perintah-perintah SQL yang dibuat *user* ataupun program-program aplikasinya.

Sebagai *database server,* MySQL dapat dikatakan lebih unggul disbanding *databse server* lainnya dalam query data. MySQL adalah salah satu dari sekian banyak sistem *database* yang merupakan terobosan solusi yang tepat dalam aplikasi *database.*

MySQL adalah multi *user database* yang menggunakan bahasa *Strucktured Query Language* (SQL). MySQL mampu menangani data yang cukup besar. Perusahaan yang mengembangkan MySQL yaitu TcX, mengaku menyimpan data lebih dari 40 *database,* 10.000 tabel dan sekitar 7 juta baris, totalnya kurang lebih 100 *Gigabyte* data. Bahasa ini pada awalnya dikembangkan oleh IBM, namun telah diadopsi 25 dan digunakan sebagai standar industri.

Dengan menggunakan SQL, proses akses *database* menjadi lebih *user friendly* dibandingkan dengan menggunakan *dBASE* atau *Clipper* yang masih menggunakan perintah-perintah pemrograman. MySQL merupakan *software database* yang paling poluler di lingkungan Linux, kepopuleran ini karena ditunjang performa *query* dari dari *database*nya yang saat ini bisa dikatakan paling cepat dan jarang bermasalah. MySQL ini juga sudah dapat berjalan pada lingkungan *Windows.*

Perintah untuk mengelola *database* dibagi menjadi 3 (tiga) kelompok, diantaranya :

1. Perintah untuk mendefinisikan data/DDL (*Data Definition Language*).
2. Perintah untuk memanipulasi data/DML (*Data Manipulation Language*).
3. Perintah untuk mengendalikan data/DCL (*Data Control Language*).

2.7 Sekilas Tentang PHP

PHP (*PHP Hypertext Preprocessor*) merupakan bahasa pemrograman web yang bersifat server side (bekerja di sisi server). PHP merupakan script yang menyatu dengan HTML dan berada pada server (*server side HTML embedded scripting*).

PHP paling banyak dipakai saat ini untuk memrogram situs web dinamis, meskipun tidak menutup kemungkinan untuk membuat aplikasi-aplikasi ain yang berbasis web. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru/up to date. Semua script PHP dieksekusi pada server dimana script tersebut dijalankan.

2.7.1 Sejarah PHP

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs Personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama FI (*Form Interpreted*), yang wujudnya berupa sekumpulan script yang digunakan untuk mengolah data form dari web. Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI, kependekan dari *Hypertext Preprocessing*/*Form Interpreter*.

Dengan perilisan kode sumber ini menjadi open source, maka banyak programmer yang tertarik untuk ikut mengembangkan PHP. Pada November 1997, dirilis PHP/FI 2.0. Pada rilis ini interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan. Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik, dan lebih cepat.

Kemudian pada Juni 1998, perusahaan tersebut merilis interpreter baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0. Pada pertengahan tahun 1999, Zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi.

Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model

pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek.

2.7.2 Kelebihan PHP

Sebagai bahasa pemrograman, PHP memiliki beberapa kelebihan bila dibandingkan dengan bahasa pemrograman lain. Kelebihan-kelebihan tersebut diantaranya adalah sebagai berikut :

1. Bahasa pemrograman PHP adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaanya
2. Web Server yang mendukung PHP dapat ditemukan dimana – mana dari mulai IIS (*Internet Information Server)* sampai dengan apache, dengan configurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis – milis dan developer yang siap membantu dalam pengembangan.
4. Dalam sisi pemahamanan, PHP adalah bahasa scripting yang paling mudah karena referensi yang banyak.
5. PHP adalah bahasa open source yang dapat digunakan di berbagai mesin (linux, unix, windows) dan dapat dijalankan secara runtime melalui console serta juga dapat menjalankan perintah-perintah sistem.

2.8 Masalah Kerusakan *Hardware*

Meski komputer saat ini telah mengalami perkembangan yang sedemikian pesat, bukan berarti masalah menjadi hilang. Merki masalah tidak sebanyak

komputer generasi sebelumnya, komputer modern saat ini juga tak luput dari masalah kerusakan *hardware*. Berdasarkan fungsinya, perangkat keras (*hardware*) dibagi menjadi tiga jenis, yaitu :

1. Piranti masukan (*input device*), perangkat masukan berfungsi untuk memasukkan data, baik berupa teks, foto, maupun gambar ke dalam komputer. Contoh perangkat input misalnya keyboard, mouse, light-pen, scanner, dan sebagainya.
2. Piranti pemroses (*processo*), perangkat pengolah data dipergunakan untuk mengolah data.Pengolah data meliputi unit pengolah pusat CPU (*Central Processing Unit*) dan juga mikroprosesor.
3. Piranti keluaran (*output device*), perangkat keluaran dipergunakan untuk menampung dan menghasilkan data yang dikeluarkan, misalnya monitor dan printer.

2.8.1 Kategori Pemeriksaan *Hardware*

Pemeriksaan terhadap sebuah *hardware* komputer yang merupakan spesifikasi bentuk-bentuk penelusuran terhadap kesalahan atau kerusakan yang terjadi, dapat dikategorikan menjadi beberapa kategori. Kategori tersebut yakni :

1. Kategori pemeriksaan visual

   Pemeriksaan visual merupakan tahap awal pemeriksaan, yaitu pemeriksaan beberapa masalah yang dapat ditangkap dengan mudah melalui pandangan mata atau panca indera. Misalnya pemeriksaan sambungan kabel, pemeriksaan jumper, dan sebagainya.

2. Kategori pemeriksaan bunyi

   Pemeriksaan bunyi merupakan pemeriksaan komponen dengan mendengar bunyi-bunyi atau *beep code* yang dapat memberika arti tersendiri akan kondisi *hardware* komputer. Misalnya pendeteksian bunyi *beep* yang muncul ketika POST.

3. Kategori pemeriksaan kondisi

   Pemeriksaan ini memerlukan interaksi yang lebih khusus seperti misalnya menyentuh chip untuk merasakan suhunya. CPU yang mempunyai panas berlebihan tentunya akan menimbulkan masalah.

4. Kategori pemeriksaan isyarat

   Ini adalah cara terakhir jika semua langkah yang dilakukan tidak mampu mendeteksi dan menyelesaikan masalah. Pemeriksaan ini lebih *complicated,* memerlukan peralatan khusus, dokumen dan gambar sistem serta pengetahuan yang mendalam mengenai *hardware.*

2.8.2 Prinsip Troubleshooting

Troubleshooting adalah sebuah bentuk pemecahan masalah pada suatu sistem yang sedang mengalami gangguan yaitu dengan mencari sumber yang menjadi penyebab masalah tersebut dan kemudian mencari pemecahan atas permasalahan yang terjadi sehingga sistem tersebut bisa beroperasi kembali secara normal.

Melakukan *troubleshooting* suatu perangkat keras (*hardware*) mempunyai aturan dan langkah-langkah untuk berhasil memecahkan masalah, aturan tersebut yakni [6] :

1. Melakukan analisis kerusakan awal
2. Menyiapkan perangkat yang dibutuhkan
3. Mencabut kabel *power* sebelum memegang komponen komputer
4. Menggunakan pengaman listrik
5. Membuat hanya satu langkah perubahan saat melakukan pengecekkan
6. Mencatat semua langkah yang diambil
7. Melakukan pengecekkan awal sebelum komputer dinyalakan
8. Minta pendapat ahli.