PENERBIT ANDI®

STMIK POTENSI UTAMA

sistem pakar

konsep dan teori

Rika Rosnelly

SISTEM PAKAR

Konsep dan Teori

Rika Rosnelly

Diterbitkan Atas Kerjasama

PENERBIT ANDI®

STMIK POTENSI UTAMA

Sistem Pakar Konsep dan Teori

Oleh: Rika Rosnelly

Editor : Inunk Nastiti
Setter : Arif Budi Permana
Desain Cover : R. Budi Wibowo
Korektor : Putri Christian

Penerbit: CV ANDI OFFSET (Penerbit ANDI)
Jl. Beo 38-40, Telp. (0274) 561881 (Hunting), Fax. (0274) 588282 Yogyakarta 55281

Percetakan: ANDI OFFSET
Jl. Beo 38-40, Telp. (0274) 561881 (Hunting), Fax. (0274) 588282 Yogyakarta 55281

Perpustakaan Nasional: Katalog dalam Terbitan (KDT)

Rosnelly, Rika
Sistem Pakar Konsep dan Teori /Rika Rosnelly
– **Ed. I.** – Yogyakarta: Andi,

20 - 19 - 18 - 17 - 16 - 15 - 14 - 13 - 12

viii + 124 hlm .; 16 x 23 Cm.

10 9 8 7 6 5 4 3 2 1

ISBN: 978 – 979 – 29 – 3416 - 8

I. Judul

I. Expert System - Computer

DDC'21: 006. 33

PRAKATA

Puji syukur kehadirat Allah SWT, alhamdulillah penulis mengucapkan rasa syukur yang sebesar-besarnya kepada Allah SWT karena hanya dengan karuniaNya-lah penulis bisa menyelesaikan buku ini.

Buku ini membahas konsep sistem pakar, struktur sistem pakar, representasi pengetahuan, metode inferensi, penalaran dengan ketidakpastian, peluang dengan teorema bayes, penalaran inexact, tahapan pengembangan sistem pakar, perancangan sistem pakar, penjelasan baru untuk penjelasan sistem pakar, project pembuatan sistem pakar.

Terima kasih yang sebesar-besarnya penulis sampaikan kepada semua pihak yang telah membantu hingga selesainya penulisan buku ini dan juga kepada Penerbit Andi Offset yang sudah bersedia menerbitkan buku ini.

Penulis menyadari bahwa buku ini masih sangat jauh dari sempurna. Semua saran, kritik dan tegur sapa yang sifatnya membangun akan penulis terima dengan kerendahan hati demi kesempurnaan di masa mendatang. Untuk itu, para pembaca dapat mengalamatkannya ke e-mail : rika@potensi-utama.ac.id

Harapan penulis semoga buku ini bermanfaat bagi pembaca sekalian. Amin.

Medan, Oktober 2011

Rika Rosnelly

Daftar Isi

BAB I

KONSEP SISTEM PAKAR

Pokok Bahasan

I.1. Apa itu sistem pakar
I.2. Kelebihan sistem pakar
I.3. Konsep umum sistem pakar
I.4. Elemen manusia pada sistem pakar

Tujuan mempelajari pada bab ini diharapkan dapat memahami apa itu sistem pakar, kelebihan sistem pakar, konsep umum sistem pakar dan elemen manusia pada sistem pakar. Langkah pertama dalam menyelesaikan setiap masalah adalah dengan mendefinisikan terlebih dahulu ruang lingkup permasalahan tersebut atau domain untuk permasalahan yang akan diselesaikan. Hal ini juga berlaku untuk pemrograman *Artificial Intelligence* (AI). Namun karena hal-hal yang berkaitan dengan mistis berpadu dengan AI maka masih ada sesuatu yang melekat untuk tetap mempercayai pepatah lama ”merupakan bagian dari masalah AI jika masalah tersebut belum diselesaikan. Definisi yang populer lainnya dari AI adalah bahwa ”AI menjadikan komputer berakting dan bergaya seperti halnya para artis berakting di bioskop”. Dan untuk saat ini banyak permasalahan dunia nyata yang diselesaikan menggunakan AI dan banyak juga aplikasinya yang dikomersialkan.

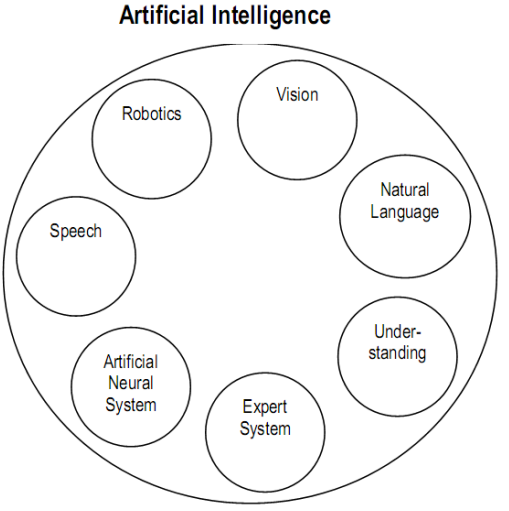
Walaupun penyelesaian umum untuk masalah AI klasik seperti translasi bahasa alami, pemahaman ucapan, dan visi belum ditemukan, tetapi pembatasan domain permasalahannya telah dapat menghasilkan suatu penyelesaikan yang bermanfaat. Sebagai contoh, tidaklah terlalu sukar untuk membangun suatu sistem bahasa alami yang sederhana jika masukan dibatasi untuk kalimat dengan bentuk kata benda, kata kerja dan objek.Untuk saat ini sistem dari tipe ini bekerja sangat baik dalam menyediakan antarmuka yang familiar dengan pengguna untuk banyak produk perangkat lunak seperti sistem database dan *spreadsheets.*

I.1. Apa itu sistem pakar ?

Artificial intelligence (AI) memiliki beberapa domain masalah / area seperti yang tergambar pada Gambar I.1. di bawah. Bidang sistem pakar merupakan penyelesaian pendekatan yang sangat berhasil/bagus untuk permasalahan AI klasik dari pemrograman *intelligent* (cerdas). Sistem pakar (expert system) merupakan solusi AI bagi masalah pemrograman pintar

(intelligent). Profesor Edward Feigenbaum dari Stanford University yang merupakan pionir dalam teknologi sistem pakar mendefinisikan sistem pakar sebagai : Sebuah program komputer yang pintar (intelligent computer program) yang memanfaatkan pengetahuan (knowledge) dan prosedur inferensi (inference procedure) untuk memecahkan masalah yang cukup sulithingga membutuhkan keahlian manusia yang khusus.

Dengan kata lain, sistem pakar adalah sistem komputer yang ditujukan untuk meniru semua aspek (emulates) kemampuan pengambilan keputusan (decision making) seorang pakar. Sistem pakar memanfaatkan secara maksimal pengetahuan khusus selayaknya seorang pakar untuk memecahkan masalah.



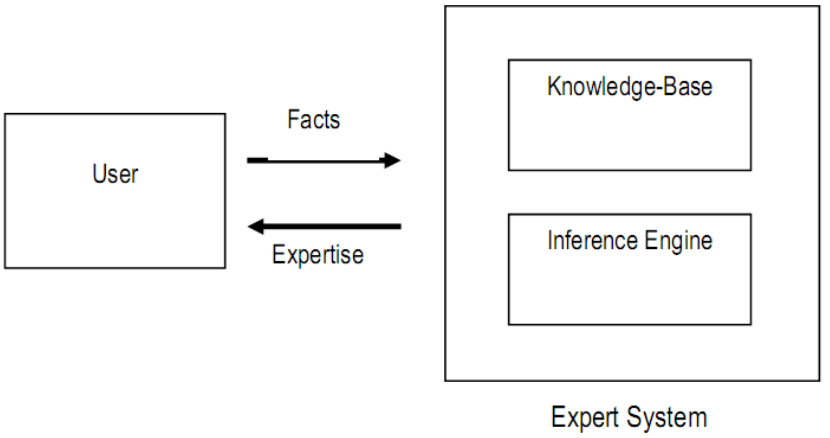
Gambar I.1. Area dari Artificial Intelligence (AI)

Pakar atau ahli (expert) disini didefinisikan sebagai seseorang yang memiliki pengetahuan atau keahlian khusus yang tidak dimiliki oleh orang kebanyakan. Seorang pakar dapat memecahkan masalah yang tidak mampu dipecahkan orang kebanyakan atau memecahkan suatu masalah dengan lebih efisien namun bukan berarti lebih murah.

Pengetahuan yang dimuat ke dalam sistem pakar dapat berasal dari seorang pakar, ataupun pengetahuan yang berasal dari buku, jurnal, majalah, dan dokumentasi yang dipublikasikan lainnya, serta orang yang memiliki pengetahuan meskipun bukan ahli. Istilah sistem pakar (expert system), sering disinonimkan dengan sistem berbasis pengetahuan (knowledge-based system) atau sistem pakar berbasis pengetahuan (knowledge-based expert

system).

Gambar I.2. dibawah ini mengilustrasikan konsep dasar sistem pakar berbasis pengetahuan (knowledge-based expert system). User memberikan informasi atau fakta kepada sistem dan menerima respon berupa saran ahli (advice/expertise). Secara internal, sistem terdiri dari dua komponen utama yaitu basis pengetahuan (knowledge-base) yang berisi pengetahuan yang akan digunakan oleh komponen lainnya yaitu mesin inferensi (inference engine) untuk menghasilkan kesimpulan sebagai respon terhadap kueri yang dilakukan user.

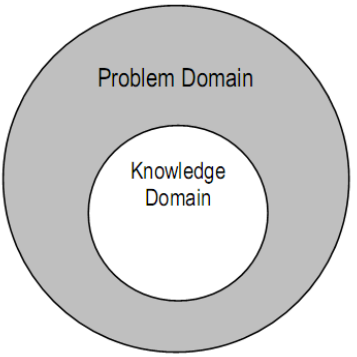


Gambar I.2. Konsep dasar fungsi Sistem Pakar Berbasis Pengetahuan

Pengetahuan yang dimiliki pakar bersifat spesifik dalam satu area masalah (problem domain). Area masalah merupakan satu wilayah masalah yang spesifik seperti kedokteran/pengobatan (medicine), keuangan (finance), rekayasa (engineering) dan lain sebagainya. Pengetahuan si pakar untuk memecahkan masalah yang spesifik tersebut dikenal sebagai area pengetahuan (knowledge domain). Gambar I.3. berikut ini menggambarkan hubungan antara area masalah (problem domain) dengan area pengetahuan (knowledge domain). Area pengetahuan seluruhnya berada dalam area masalah. Bagian yang berada di luar area pengetahuan menyatakan pengetahuan mengenai masalah yang tidak dimiliki oleh sistem. Sebuah sistem pakar umumnya tidak memiliki pengetahuan lain diluar Area pengetahuannya kecuali jika diprogram dan dimuat ke dalam sistem.

Misalkan sebuah sistem pakar yang memuat pengetahuan mengenai penyakit infeksi mungkin tidak memiliki pengetahuan lain dalam area masalah kedokteran. Dalam area pengetahuan yang dimiliki, sistem pakar melakukan inferensi / membuat kesimpulan dengan cara yang sama seperti seorang

pakar menarik kesimpulan.



Gambar I.3. Hubungan Problem dan Knowledge domain

I.2. Kelebihan Sistem Pakar

Sistem pakar memiliki beberapa fitur menarik yang merupakan kelebihannya, seperti :

- Meningkatkan ketersediaan *(increased availability)*. Kepakaran/keahlian menjadi tersedia dalam sistem komputer. Dapat dikatakan bahwa sistem pakar merupakan produksi kepakaran secara masal *(massproduction)*.
- Mengurangi biaya *(reduced cost)*. Biaya yang diperlukan untuk menyediakan keahlian per satu orang user menjadi berkurang.
- Mengurangi bahaya *(reduced danger)*. Sistem pakar dapat digunakan di lingkungan yang mungkin berbahaya bagi manusia.
- Permanen *(permanence)*. Sistem pakar dan pengetahuan yang terdapat didalamnya bersifat lebih permanen dibandingkan manusia yang dapat merasa lelah, bosan, dan pengetahuannya hilang saat sang pakar meninggal dunia.
- Keahlian multipel *(multiple expertise)*. Pengetahuan dari beberapa pakar dapat dimuat ke dalam sistem dan bekerja secara simultan dan kontinyu menyelesaikan suatu masalah setiap saat. Tingkat keahlian/pengetahuan yang digabungkan dari beberapa pakar dapat melebihi pengetahuan satu orang pakar.
- Meningkatkan kehandalan *(increased reliability)*. Sistem pakar meningkatkan kepercayaan dengan memberikan hasil yang benar sebagai alternatif pendapat dari seorang pakar atau sebagai penengah jika terjadi konflik antara beberapa pakar. Namun hal tersebut tidak berlaku, jika sistem dibuat oleh salah seorang pakar sehingga akan selalu sama dengan pendapat pakar tersebut kecuali

jika sang pakar melakukan kesalahan yang mungkin terjadi pada saat tertekan atau stres.

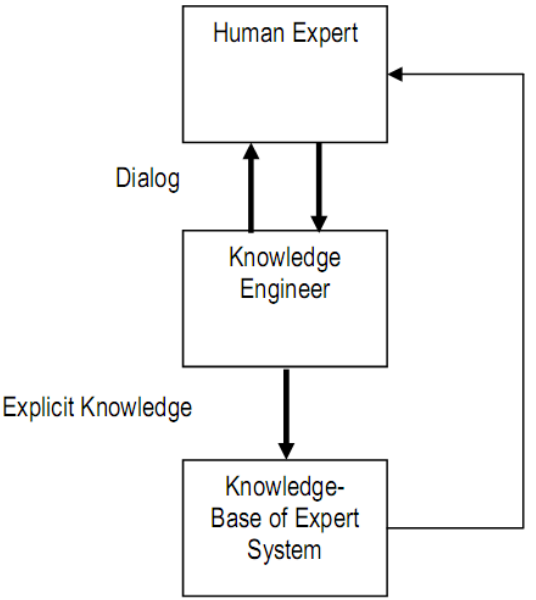
- Penjelasan (*explanation*). Sistem pakar dapat menjelaskan detail proses penalaran (*reasoning*) yang dilakukan hingga mencapai suatu kesimpulan. Seorang pakar mungkin saja terlalu lelah, tidakk bersedia atau tidak mampu melakukannya setiap waktu. Dan hal ini akan meningkatkan tingkat kepercayaan bahwa kesimpulan yang dihasilkan adalah benar.
- Respon yang cepat (*fast response*). Respon yang cepat atau *real-time* diperlukan pada beberapa aplikasi. Meskipun bergantung pada *hardware* dan *software* yang digunakan, namun sistem pakar relatif memberikan respon yang lebih cepat dibandingkan seorang pakar.
- Stabil, tidak emosional, dan memberikan respon yang lengkap setiap saat (*steady, unemotional, and complete response at all times*). Karakterisitik ini diperlukan pada situasi *real-time* dan keadaan darurat (*emergency*) ketika seorang pakar mungkin tidak berada pada kondisi puncak disebabkan oleh stres atau kelelahan.
- Pembimbing pintar (*intelligent tutor*). Sistem pakar dapat berperan sebagai *intelligent tutor* dengan memberikan kesempatan pada user untuk menjalankan contoh program dan menjelaskan proses *reasoning* yang dilakukan.
- Basis data cerdas (*intelligent database*). Sistem pakar dapat digunakan untuk mengakses basis data secara cerdas.

I.3. Konsep Umum Sistem Pakar

Pengetahuan yang dimiliki sistem pakar direpresentasikan dalam beberapa cara. Salah satu metode yang paling umum digunakan adalah tipe rules menggunakan format IF THEN. Banyak sistem pakar yang dibangun dengan mengekspresikan pengetahuan dalam bentuk rules. Bahkan pendekatan berbasis pengetahuan (knowledge-based approach) untuk membangun sistem pakar telah mematahkan pendekatan awal yang digunakan pada sekitar tahun 1950-an dan 1960-an yang menggunakan tehnik penalaran (reasoning) yang tidak mengandalkan pengetahuan.

Pengetahuan tidak tertulis yang dimiliki oleh seorang pakar harus diekstraksi melalui wawancara secara ekstensif oleh knowledge engineer. Proses pengembangan sistem pakar yang berhubungan dengan perolehan pengetahuan dari pakar maupun sumber lain dan codingnya disebut sebagai knowledge engineering yang dilaksanakan oleh knowledge engineer. Tahapan pengembangan sistem pakar secara umum tergambar pada gambar I.4 di bawah ini.

Tahap awal, knowledge engineer melakukan diskusi dengan pakar untuk mengumpulkan pengetahuan yang dimiliki pakar yang bersangkutan. Tahap ini serupa dengan proses diskusi persyaratan / kebutuhan yang dilakukan system engineer pada sistem konvensional dengan kliennya. Setelah itu *knowledge engineer* melakukan koding pengetahuan secara eksplisit ke dalam *knowledge base*. Pakar kemudian mengevaluasi sistem pakar dan memberikan kritik. Proses ini berlangsung secara iteratif hingga dinilai sesuai oleh pakar.



Gambar I.4. Pengembangan Sistem Pakar

Sistem pakar umumnya dirancang dengan cara yang berbeda dengan sistem konvensional lain, terutama karena masalah yang dihadapi umumnya tidak memiliki solusi algoritmik dan bergantung pada inferensi untuk mendapatkan solusi yang terbaik yang paling mungkin (reasonable). Oleh karena itu sistem pakar harus mampu menjelaskan inferensi yang dilakukannya sehingga hasil yang diperoleh dapat diperiksa. Explanation facility (fasilitas untuk menjelaskan) merupakan bagian terintegrasi dari sebuah sistem pakar. Sebuah explanation facility yang detail dirancang untuk memungkinkan user mengeksplorasi rules melalui tipe pertanyaan “what if” yang disebut hypothetical reasoning dan bahkan menerjemahkan natural language ke dalam rules. Beberapa sistem pakar bahkan mampu belajar membentuk rules *(learn rules by example)* dengan cara induksi *(rule*

induction) dari tabel data.

Memformalisasi pengetahuan pakar ke dalam rules tidaklah sederhana, terutama jika pengetahuan tersebut belum pernah disusun secara sistematik sebelumnya. Mungkin terjadi masalah-masalah seperti inkonsistensi, ambiguitas, duplikasi. Seorang pakar juga mengetahui batas pengetahuan yang mereka miliki dan membatasi saran yang mereka berikan.

Keterbatasan sistem pakar dalam prakteknya saat ini adalah kurangnya pengetahuan kausal *(causal knowledge)*, yaitu sistem tidak memiliki pemahaman yang melandasi sebab dan efek dalam sistem. Lebih mudah membangun sistem pakar dengan pengetahuan yang bersifat dangkal *(shallow knowledge)* yang didasarkan atas pengetahuan empirik atau heuristik dibandingkan dengan memprogram sistem dengan pengetahuan mendalam *(deep knowledge)* yang berdasarkan struktur, fungsi dan perilaku sebuah obyek.

Salah satu tipe *shallow knowledge* adalah pengetahuan heuristik atau empirik yang diperoleh dari pengalaman. Pengetahuan jenis ini dapat membantu pencapaian solusi namun tidak menjamin hasil yang sukses seperti halnya sebuah algoritma. Namun pada beberapa bidang seperti kedokteran atau perekayasaan (*engineering*), heuristik memainkan peran yang cukup penting dalam beberapa jenis pemecahan masalah.
Bahkan ketika solusi yang pasti diketahui namun dirasa tidak praktis karena biaya dan waktu yang lama, maka heuristik dapat menjadi jalan pintas bagi persoalan tersebut.

Masalah lain dalam sistem pakar adalah keterbatasan pengetahuan dalam area pengetahuan. Umumnya sebuah sistem pakar tidak dapat melakukan analogi terhadap sebuah masalah baru seperti yang dilakukan oleh manusia. Meskipun dapat diterapkan aturan induksi, namun hanya berlaku pada beberapa tipe pengetahuan saja. Cara yang umum dilakukan untuk mengembangkan sistem pakar adalah dengan membuat seorang *knowledge engineer* mengulangi siklus pengumpulan pengetahuan mulai dari mewawancarai pakar, membangun prototipe, menguji, dan seterusnya. Hal ini sangat memakan waktu dan menimbulkan masalah kemacetan pengumpulan pengetahuan (*knowledge aqcuisition bottleneck*). Namun diluar semua kekurangan yang dimilikinya, sistem pakar dinilai berhasil menghadapi masalah yang tidak mampu dipecahkan oleh metode pemrograman konvensional lain terutama yang berhubungan dengan informasi yang tidak pasti atau tidak lengkap.

I.4. Elemen manusia pada sistem pakar

Sistem pakar tidak lepas dari elemen manusia yang terkait di dalamnya. Personil yang terkait dengan sistem pakar ada 4, yaitu :

1. pakar *(expert)*
2. pembangun pengetahuan *(knowledge engineer)*
3. pembangun sistem *(system engineer)*
4. pemakai *(user)*

Paling tidak terdapat dua komponen orang atau lebih, berpartisipasi dalam pembangunan dan penggunaan sistem pakar, yakni sedikitnya seorang pembangun pengetahuan dan seorang pakar.

Pakar

Pakar adalah seorang individu yang memiliki pengetahuan khusus, pemahaman, pengalaman, dan metode-metode yang digunakan untuk memecahkan persoalan dalam bidang tertentu.

Seorang pakar memiliki kemampuan kepakaran, yaitu :

a. dapat mengenali dan merumuskan suatu masalah
b. menyelesaikan masalah dengan cepat dan tepat
c. menjelaskan solusi dari suatu masalah
d. restrukturisasi pengetahuan
e. belajar dari pengalaman
f. memahami batas kemampuan

Selain itu, pakar juga memiliki kemampuan untuk mengaplikasikan pengetahuannya dan memberikan saran serta pemecahan masalah pada domain tertentu. Ini merupakan pekerjaan pakar, memberikan pengetahuan tentang bagaimana seseorang melaksanakan tugas untuk menyelesaikan masalah. Penyelesaian masalah ini didukung atau bahkan secara ekstrim akan dilaksanakan oleh sistem berbasis pengetahuan/sistem pakar. Seorang pakar mengetahui fakta-fakta mana yang penting, sebab akibat, fenomena-fenomena yang terkait dengan fakta, memahami arti hubungan antar fakta, juga hubungan sebab akibat, dan hubungan dengan fenomena-fenomena yang terkait serta mampu menginterpretasikan akibat-akibat yang terjadi karena sesuatu sebab terjadi. Pada kasus diagnosa masalah sistem elektronik mobil, misalnya, pakar mekanik mengetahui bahwa tali kipas dapat rusak dan sebab kekosongan baterai. Seseorang akan langsung mengecek tali kipas dan menginterpretasikan arti dari lepas atau hilangnya tali dengan akibat-akibat yang ditimbulkannya. Ini adalah contoh keahlian. Jika digunakan lebih dari satu pakar, situasi akan menjadi sulit jika pakar-pakar saling tidak setuju. Perlu ada kesepakatan antar pakar dalam menyelesaikan permasalahan.

Pembangun/Pembuat Pengetahuan

Pembangun pengetahuan memiliki tugas utama menterjemahkan dan merepresentasikan pengetahuan yang diperoleh dari pakar, baik berupa pengalaman pakar dalam menyelesaikan masalah maupun sumber terdokumentasi lainnya ke dalam bentuk yang bisa diterima oleh sistem pakar. Dalam hal ini pembangun pengetahuan *(knowledge engineer)*

menginterpretasikan, dan merepresentasikan pengetahuan yang diperoleh dalam bentuk jawaban-jawaban atas pertanyaan-pertanyaan yang diajukan pada pakar, atau pemahaman, penggambaran analogis, sistematis, konseptual yang diperoleh dari membaca beberapa dokumen cetak seperti *text book*, jurnal, makalah, dan sebagainya,

Kurangnya pengalaman *knowledge engineer* merupakan kesulitan utama dalam mengkonstruksi sistem pakar. Untuk mengatasi hal tersebut, perancang sistem pakar menggunakan tools komersial.(seperti pada editor-editor khusus maupun logic debuggers), dan usahanya akan dipusatkan pada pembangunan mesin inferensi.

Pembangun/Pembuat Sistem

Pembangun sistem adalah orang yang bertugas untuk merancang antar muka pemakai sistem pakar, merancang pengetahuan yang sudah diterjemahkan oleh pembangun pengetahuan ke dalam bentuk yang sesuai dan dapat diterima oleh sistem pakar dan mengimplementasikannya ke dalam mesin inferensi. Selain hal tersebut pembangun sistem juga bertanggung jawab apabila sistem pakar akan diintegrasikan dengan sistem komputerisasi lain. Alat pembangun *(tool builder)* dapat dipakai untuk menyajikan atau membangun tool yang spesifik. Penjual *(vendor)* dapat memberikan tool dan saran, staf pendukung dapat memberikan saran dan bantuan secara teknis dalam proses pembangunan sistem pakar.

Pengguna

Banyak sistem berbasis komputer mempunyai susunan pengguna tunggal. Hal ini berbeda jauh dengan sistem pakar yang memungkinkan mempunyai beberapa kelas pengguna. Tabel I.1 menunjukkan beberapa contoh hubungan antara kelas pengguna, kepentingan pengguna, dan fungsi dari sistem pakar.

Tabel I.1. Hubungan antara pengguna dan fungsi sistem pakar

Pengguna	Kepentingan	Fungsi Sistem Pakar
Klien bukan pakar	Mencari saran/nasehat	Konsultan atau penasehat
Mahasiswa	Belajar	Instruktur
Pembangun sistem	Memperbaiki/menambah Basis pengetahuan	Rekan (partner)
Pakar	Membantu analisis rutin Atau proses komputasi, Mencari (mengklasifikasi) Informasi, alat bantu Diagnosa	Rekan kerja atau asisten

Pengguna mungkin tidak terbiasa dengan komputer dan mungkin pada domain masalah. Bagaimanapun juga, banyak solusi permasalahan menjadi lebih baik dan kemungkinan lebih murah dan keputusan yang cepat bila menggunakan sistem pakar. Pakar dan pembangun sistem harus mengantisipasi kebutuhan-kebutuhan pengguna dan membuat batasan-batasan ketika mendesain sistem pakar.

Latihan

1. Pahami kembali konsep umum sistem pakar
2. Buat contoh sistem pakar
3. Coba defenisikan permasalahan sistem pakar tersebut

BAB II

STRUKTUR SISTEM PAKAR

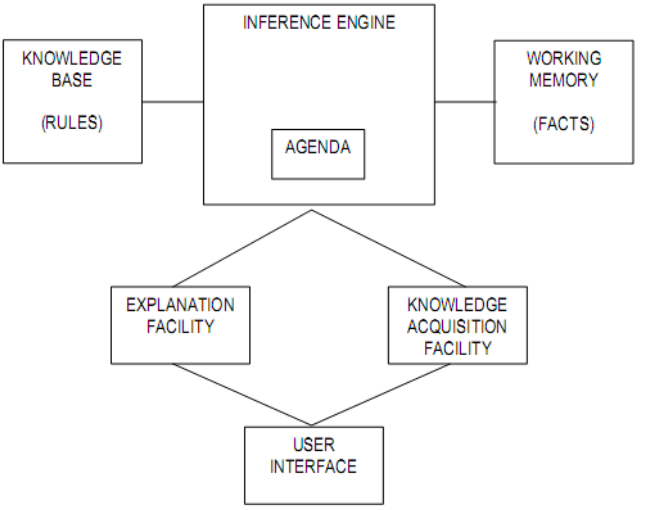
Pokok Bahasan

II.1. Struktur sistem pakar
II.2. Karakteristik sistem pakar

Tujuan mempelajari pada bab ini diharapkan dapat memahami struktur sistem pakar dan karakteristik sistem pakar tersebut.

II.1. Struktur Sistem Pakar

Adapun struktur sistem pakar dapat dilihat pada gambar II.1 :



Gambar II.1 Struktur Sistem Pakar

Komponen yang terdapat dalam struktur system pakar ini adalah kkowledge base (rules), inference engine, working memory, explanation facility, knowledge acquisition facility, user interface.

1. Knowledge Base (Basis Pengetahuan)

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

Pada struktur sistem pakar diatas, knowledge base disini untuk menyimpan pengetahuan dari pakar berupa rule / aturan (if <kondisi> then <aksi> atau dapat juga disebut condition-action rules).

2. Inference Engine (Mesin Inferensi)
   Mesin Inferensi merupakan otak dari sebuah sistem pakar dan dikenal juga dengan sebutan *control structure* (struktur control) atau *rule interpreter* (dalam sistem pakar berbasis kaidah). Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah.
   Mesin inferensi disini adalah processor pada sistem pakar yang mencocokkan bagian kondisi dari rule yang tersimpan di dalam knowledge base dengan fakta yang tersimpan di working memory.
3. Working Memory
   Berguna untuk menyimpan fakta yang dihasilkan oleh inference engine dengan penambahan parameter berupa derajat kepercayaan atau dapat juga dikatakan sebagai global database dari fakta yang digunakan oleh rule-rule yang ada.
4. Explanation facility
   yaitu menyediakan kebenaran dari solusi yang dihasilkan kepada user (reasoning chain)
5. Knowledge acquisition facility
   Meliputi proses pengumpulan, pemindahan dan perubahan dari kemampuan pemecahan masalah seorang pakar atau sumber pengetahuan terdokumentasi ke program computer, yang bertujuan untuk memperbaiki atau mengembangkan basis pengetahuan.
6. User Interface
   Mekanisme untuk memberi kesempatan kepada user dan sistem pakar untuk berkomunikasi. Antar muka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari sistem dan menyajikannya ke dalam bentuk yang dapat dimengerti oleh pemakai.
   Pada umumnya, antar muka pemakai juga berfungsi untuk menginputkan pengetahuan baru kedalam basis pengetahuan sistem pakar, menampilkan fasilitas penjelasan sistem dan memberikan tuntunan penggunaan sistem secara menyeluruh langkah demi langkah sehingga pemakai mengerti apa yang harus dilakukan terhadap sistem.

Syarat utama membangun antar muka pemakai adalah kemudahan dalam menjalankan sistem. Semua kesulitan dalam membangun suatu program harus disembunyikan, yang ditampilkan hanyalah tampilan yang interaktif, komunikatif dan kemudahan pakai.

Pada sistem pakar berbasis pengetahuan, ***knowledge base*** disebut sebagai *production memory*. *Production rules* dituliskan dalam bentuk berikut :

the light is red ⟶ *stop*
the light is green ⟶ *go*

Production rule juga dapat diekspresikan dalam bentuk IF THEN seperti :

Rule : Red_light
IF
    the light is red
THEN
    stop
Rule : Green_light
IF
    the light is green
THEN
    go

Setiap *rule* diidentifikasi dengan nama. Bagian antara IF dan THEN dikenal dengan berbagai istilah seperti *antecedent*, *conditional part*, *pattern part*, atau *left-hand-side* (LSH). Sedangkan bagian THEN dari sebuah rule yang berisi sekumpulan aksi disebut sebagai *consequent* atau *right-hand side* (RHS). Berikut ini dicontohkan *production rules* dari sistem pakar MYCIN :

MYCIN system for diagnosis of meningitis and bacteremia (bacterial infections)
IF
    The site of the culture is blood, and
    The identity of the organism is not known with certainty, and
    The stain of the organism is gramneg, and
    The morphology of the organism is rod, and
    The patient has been seriously burned
THEN
    There is weakly suggestive evidence (.4) thet the identity of the organism is
    pseudomonas

Pada sistem pakar berbasis pengetahuan, **inference engine** menentukan rule yang memenuhi atau sesuai dengan fakta yang diberikan. Terdapat dua metode umum dalam melakukan inferensi, yaitu ***forward chaining*** dan ***backward chaining***. Terdapat metode lain yang digunakan untuk kebutuhan yang lebih spesifik seperti means-end analysis, problem reduction, backtracking, plan-generate-test, hierarchical planning and the least commitment principle, dan constraint handling. ***Forward chaining*** adalah penalaran yang dilakukan dari fakta menuju kesimpulan yang dihasilkan dari fakta tersebut. Misalkan jika kita melihat hari sedang hujan sebelum meninggalkan rumah (fakta) maka kita seharusnya membawa payung (kesimpulan). ***Backward chaining*** melibatkan proses penalaran terbalik dari hipotesis, yang merupakan kesimpulan yang akan dibuktikan, ke fakta yang mendukung hipotesis. Sebagai contoh, jika anda melihat seseorang yang baru memasuki ruangan dalam keadaan basah maka anda memiliki hipotesis bahwa di luar sedang hujan. Untuk menguatkan hipotesis tersebut, anda akan mencari fakta yang mendukungnya, misalkan dengan bertanya kepada orang tersebut. Pemilihan metode inferensi bergantung pada masalah yang dihadapi. Masalah diagnosis akan lebih baik dipecahkan dengan backward chaining, sementara prognosis, monitoring, dan kontrol lebih baik diselesaikan dengan forward chaining.

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

a. *Depth-first search,* melakukan penelusuran kaidah secara mendalam dari simpul akar bergerak menurun ke tingkat dalam yang berurutan
b. *Breadth-first search,* bergerak dari simpul akar, simpul yang ada pada setiap tingkat diuji sebelum pindah ke tingkat selanjutnya
c. *Best-first search,* bekerja berdasarkan kombinasi kedua metode sebelumnya.

Working memory berisi fakta yang menggambarkan situasi terkini. Perhatikan perbedaan knowledge base dengan working memory. Pada working memory, fakta yang ada tidak saling berinteraksi. Sedangkan pada knowledge base terdapat suatu relasi berupa rule terhadap fakta untuk menghasilkan kesimpulan. Jika terdapat fakta pada working memory, **inference engine** akan melihat apakah fakta tersebut sesuai dengan aturan maka aturan tersebut akan diletakan di **agenda**. Jika aturan memiliki beberapa pola (*pattern*), maka aturan tersebut dapat dimuat ke dalam agenda bila seluruh polanya terpenuhi. Aturan yang terpenuhi polanya dikatakan sebagai aturan yang diaktifkan (*activated or instantiated*). Jika terdapat beberapa rule yang aktif pada saat yang bersamaan, maka **inference engine** harus memilih rule mana yang akan dieksekusi.

Sistem pakar yang berbasis aturan dibangun dengan menyertakan *refraction* untuk menghindari terjadinya loop yang tidak bermakna. Misalkan jika ada sebuah aturan yang terus menerus dieksekusi berdasarkan fakta yang sama berulang-ulang. Hal ini akan mengakibatkan tidak tercapainya tujuan. *Refraction* adalah sebuah istilah dan fenomena dalam bidang Neurophysiology yang mempelajari sistem syaraf. *Refraction* terjadi jika tidak ada stimulasi kepada sel syaraf yang menyebabkan neuron aktif. Beberapa metode dibangun untuk menyediakan *refraction* pada sistem pakar. Salah satu metode yang dipakai oleh expert system language, OPS5 adalah dengan memberikan sebuah identifier unik yang disebut *timetag* pada setiap fakta yang dimuat ke dalam working memory. Setelah aturan dijalankan berdasarkan fakta tertentu, maka inference engine tidak akan menggunakan fakta tersebut lagi karena *time-stamp*-nya telah digunakan.

Inference engine beroperasi dalam sebuah siklus (*cycle*). Beberapa istilah diberikan untuk
mendeskripsikan siklus tersebut, seperti *recognize-act cycle*, *select-execute cycle*, *situation-response cycle*, dan *situation-action cycle*. Dalam sebuah siklus, **inference engine** akan mengeksekusi sekumpulan tugas secara berulang hingga terdapat kriteria tertentu yang mengakibatkan eksekusi berhenti. Pada sebuah siklus mungkin terdapat lebih dari satu aturan yang aktif dan aturan yang masih aktif yang berasal dari siklus sebelumnya dalam **agenda**. Sehingga jumlah aturan yang aktif dalam agenda berubah-ubah seiring dengan berjalannya eksekusi. **Inference engine** akan mengeksekusi aksi dengan prioritas tertinggi, kemudian prioritas tinggi berikutnya dan seterusnya hingga tidak ada lagi yang perlu dieksekusi. Berbagai shell yang ada memiliki berbagai jenis skema prioritas. Namun umumnya, shell membebaskan knowledge engineer untuk mendefinisikan prioritas aturan.

Konflik di **agenda** terjadi jika aturan aktif yang berbeda memiliki prioritas yang sama. Dalam hal ini, **inference engine** harus memilih aturan yang akan dieksekusi. Beberapa shell memiliki pendekatan yang berbeda untuk menyelesaikan masalah ini. Paradigma Newell dan Simon mengasumsikan bahwa aturan yang pertama dimuat ke dalam sistem memiliki prioritas tertinggi. OPS5 menganggap aturan yang memiliki pola yang lebih kompleks memiliki prioritas yang lebih tinggi. Pada ART dan CLIPS, aturan memiliki default prioritas yang sama kecuali jika ditetapkan prioritas tertentu oleh knowledge engineer.

Setelah tahap ini, kendali sistem pakar beralih ke command interpreter tingkat tinggi (*top-level command interpreter*) yang digunakan user untuk memberikan instruksi lebih lanjut pada shell sistem pakar. *Top-level command interpreter* adalah **user interface** bagi shell selama sistem pakar dalam proses pengembangan. User interface yang lebih canggih umumnya dirancang untuk memfasilitasi operasi yang dilakukan sistem. Seperti

misalnya user interface untuk memonitor atau mengendalikan pabrik yang menampilkan diagram blok dengan display beresolusi tinggi dan bit-mapped color. Faktanya, lebih banyak usaha yang dilakukan untuk merancang dan mengimplementasikan user interface dibandingkan dengan knowledge base terutama dalam pembuatan prototipe.

Explanation facility memungkinkan user untuk bertanya bagaimana sistem dapat menghasilkan suatu kesimpulan dan mengapa suatu informasi diperlukan. Bahkan explanation facility yang lebih canggih membuat usernya dapat memberikan pertanyaan "bagaimana jika (*what if*)" untuk mengeksplorasi kemungkinan cara penalaran alternatif dengan hypothetical reasoning.

II.2. Karakteristik Sistem Pakar

Sistem pakar umumnya dirancang untuk memenuhi beberapa karakteristik umum berikut ini :

- Kinerja sangat baik (*high performance*). Sistem harus mampu memberikan respon berupa saran (*advice*) dengan tingkat kualitas yang sama dengan seorang pakar atau melebihinya.
- Waktu respon yang baik (*adequate respon time*). Sistem juga harus mampu bekerja dalam waktu yang sama baiknya (*reasonable*) atau lebih cepat dibandingkan dengan seorang pakar dalam menghasilkan keputusan. Hal ini sangat penting terutama pada sistem waktu nyata (*real-time*).
- Dapat diandalkan (*good reliability*). Sistem harus dapat diandalkan dan tidak mudah rusak / crash.
- Dapat dipahami (*understandable*). Sistem harus mampu menjelaskan langkah-langkah penalaran yang dilakukannya seperti seorang pakar. Hal ini penting untuk beberapa alasan, yaitu :
    1) Dimungkinkan bahwa sistem pakar berkaitan dengan nyawa manusia atau properti lainnya sehingga harus dapat menjelaskan mengapa dihasilkan suatu kesimpulan tertentu.
    2) Untuk mengkonfirmasi bahwa pengetahuan pakar telah dikumpulkan dengan benar dan digunakan oleh sistem dengan benar pula. Hal ini penting dalam proses debugging pengetahuan yang mungkin salah karena pengetikan atau pemahaman yang salah dari *knowledge engineer*.
- Fleksibel (*flexibility*). Sistem harus menyediakan mekanisme untuk menambah, mengubah, dan menghapus pengetahuan.

Explanation facility yang dimiliki sistem pakar dapat berbentuk sederhana atau detail. Bentuk yang sederhana, menjelaskan dengan mendaftar semua fakta yang membuat aturan dijalankan, sedangkan sistem yang lebih detail memberikan penjelasan dengan :

- *Mendaftar semua alasan (reason) yang mendukung dan menolak hipotesis tertentu*. Hipotesis adalah hal / tujuan yang akan dibuktikan atau sebagai sebuah fakta yang kebenarannya masih diragukan dan perlu dibuktikan, misalnya "Pasien terkena infeksi tetanus". Pada keadaan sebenarnya, dimungkinkan untuk memiliki lebih dari satu hipotesis.
- Mendaftar semua hipotesis yang mungkin menjelaskan bukti yang diperoleh.
- Menjelaskan semua konsekuensi hipotesis. Sebagai contoh, dengan mengasumsikan bahwa pasien terkena tetanus, maka harus ada bukti pasien mengalami demam karena infeksi tersebut. Jika terdapat suatu gejala yang dimaksudkan, maka akan memperkuat hipotesis dan sebaliknya.
- Memberikan prognosis atau prediksi yang akan terjadi jika hipotesis benar.
- Menjelaskan pertanyaan yang diberikan kepada user sebagai informasi lanjutan.
- Menjelaskan pengetahuan yang dimiliki sistem. Jika sistem memberikan hasil bahwa " Pasien terkena infeksi tetanus" maka user dapat meminta penjelasan mengenai hal tersebut. Sistem dapat menjelaskan/membenarkan kesimpulan tersebut berdasarkan *rules* yang mendukung kesimpulan.
  Dalam hal ini, sistem sesungguhnya merujuk pada *metarule* yang merupakan pengetahuan tentang *rule*. Hipotesis dinilai oleh pengetahuan dan pengetahuan dinilai/dibenarkan (*justified*) oleh sebuah *metaexplanation* yang menerangkan bagaimana sistem pakar menjelaskan proses *reasoning*-nya.

Dalam sebuah sistem berbasis aturan, pengetahuan dapat berkembang sedikit demi sedikit dengan bertambahnya *rules* sehingga kinerja bertambah dan kebenaran sistem dapat diperiksa secara kontinyu.

Latihan

Pahami kembali Struktur sistem pakar

BAB III

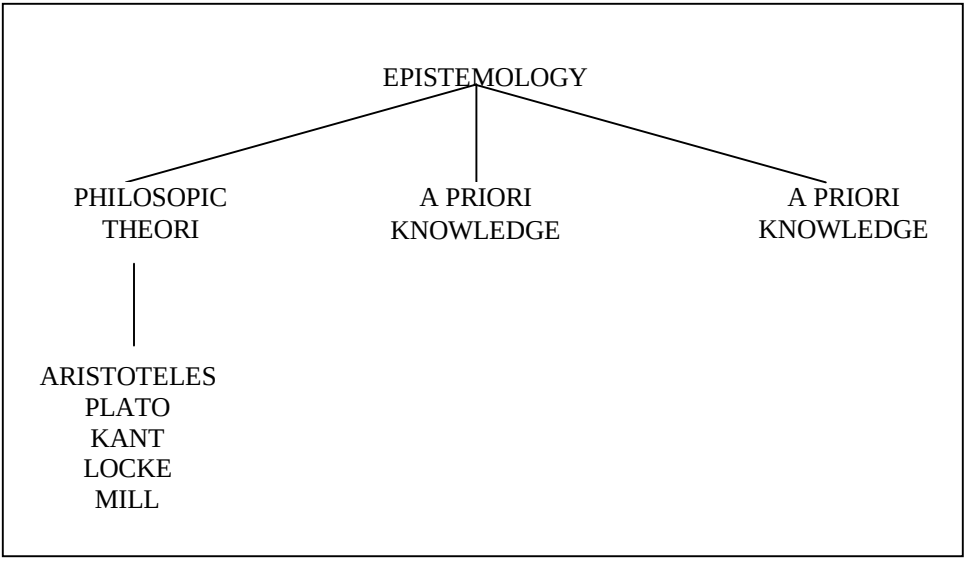
REPRESENTASI PENGETAHUAN

Pokok Bahasan
III.1. Defenisi Pengetahuan (Knowledge)
III.2. Jaringan Semantik
III.3. Frame
III.4. Script

Tujuan mempelajari bab ini diharapkan dapat memahami defenisi *Knowledge*, jaringan semantik, frame dan *script*. Kata *knowledge* (pengetahuan), seperti juga halnya kata *love* (cinta), adalah salah satu kata yang setiap orang tahu makna dan pengertiannya. Seperi halnya *love*, *knowledge* juga mempunyai pengertian yang bermacam-macam. *Knowledge* sering disinonimkan dengan data, fakta dan informasi.

III.1. Defenisi Pengetahuan *(Knowledge)*

Pelajaran dari *knowledge* merupakan suatu *epistemology* (Angels, 81) yang merupakan bagian dari ilmu filsafah yang membahas tentang asal. Hal ini berkenaan dengan sifat, struktur dan keaslian dari *knowledge*. Gambar III.1 menggambarkan beberapa kategori dari *epistemology*. Di samping jenis filosofi yang dikemukakan oleh Aristoteles, Plato, Descartes, Hume, Kant dan lainnya ada sua tipe khusus yang dinamakan *priori* dan *posteor*. Istilah priori berasal dari bahasa Latin yang berarti “yang mendahului”. Priori knowledge dianggap menjadi kebenaran yang universal dan tidak dapat disangkal tanpa kontradiksi. Pernyataan logika, hukum matematika dan *knowledge* yang dipengaruhi oleh anak belasan tahun merupakan contoh dari *priori knowledge*.



Gambar III.1. Beberapa kategori Epistemology

Kebalikan dari *priori knowledge* adalah *knowledge* yang diturunkan dari akal pikiran yang sehat, yaitu posteriori *knowledge.* Kebenaran atau kesalahan *posteriori knowlwdge* dapat dibuktikan dengan menggunakan pengalaman akal sehat, seperti pernyataan "lampu berwarna biru". Namun demikian karena berhubungan dengan pengalaman maka boleh jadi tidak selalu bisa dipercaya dan *posteriori knowledge* dapat disangkal berdasarkan *knowledge* baru tanpa memerlukan kontradiksi. Sebagai contoh, jika Anda melihat seorang yang bermata coklat, maka Anda akan percaya bahwa mata mereka adalah coklat, akan tetapi jika kemudian Anda melihat bahwa orang tersebut mengganti lensa contaknya menjadi biru, maka knowledge Anda harus diubah.

Knowledge dapat diklasifikasikan ke dalam 3 kategori, yaitu *procedural knowledge, declarative knowledge* dan *tacit knowledge.*
Procedural knowledge berkenaan untuk mengetahui bagaimana melakukan sesuatu. Sebagai contoh, pengetahuan tentang bagaimana mendidihkan air dalam mangkok. *Declarative knowledge* berkenaan untuk mengetahui sesuatu itu benar atau salah. Hal ini berkenaan dengan *knowledge* yang menunjukkan bentuk pernyataan deklarasi seperti "Jangan celupkan tangan Anda ke dalam mangkok air yang sedang mendidih."

Tacit knowledge kadang disebut juga dengan *unconcious knowledge*, karena tidak dapat diungkapkan dengan bahasa. Sebagai contoh adalah mengetahui bagaimana memindahkan tangan Anda dari dalam air panas. Pada suhu tinggi Anda harus mengatakan bahwa menarik tangan Anda cepat-cepat atau santai menggerakkannya. Tetapi bila pada suhu rendah bagaimana Anda tahu untuk menariknya cepat-cepat atau santai. Contoh lainnya adalah berjalan kaki atau bersepeda. Dalam sistem komputer, ANS (*Artificial Neural System)* dikaitkan dengan *tacit knowledge* karena secara normal jaringan saraf tidak secara langsung menjelaskan *knowledge*-nya akan tetapi mungkin mampu jika disediakan program yang sesuai.

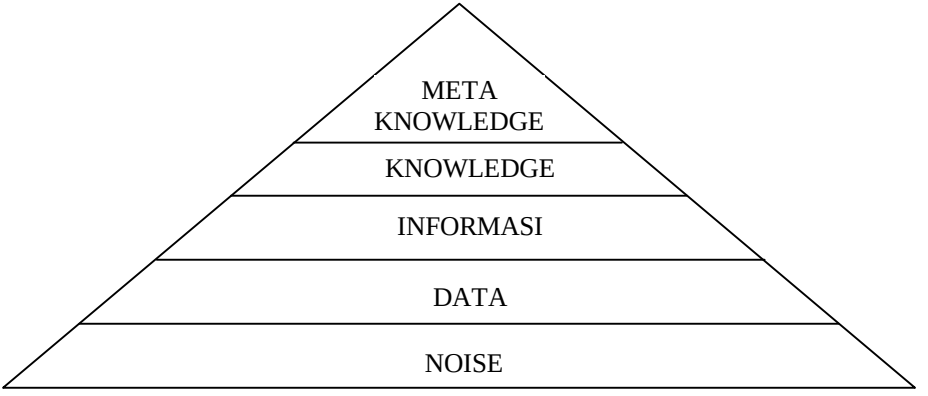
Knowledge merupakan kunci utama dari sistem pakar. Analoginya dengan ekspresi klasik dari Wirth adalah :

Algoritma + Struktur Data = Program

Dan untuk sistem pakar :

Knowledge + Inferensi = Sistem pakar

Knowledge adalah bagian dari suatu hierarki dan ini dapat ditunjukkan pada Gambar III.2. berikut ini.



Gambar III.2 *Hierarki Knowledge*

Yang ada pada level paling bawah adalah *noise* (gangguan). *Noise* merupakan data yang masih kabur. Level berikutnya adalah data yang merupakan hal potensial yang paling menarik. Data yang sudah diproses adalah informasi yang penting. Berikutnya adalah *knowledge* yang menggambarkan informasi sangat khusus. Level paling atas adalah *meta knowledge* yang merupakan *knowledge* dan keahlian. Suatu sistem pakar dapat dirancang dengan *knowledge* dari beberapa domain yang berbeda dan *meta knowledge* akan menentukan *knowledge base* yang dapat digunakan.

Untuk pembahasan selanjutnya istilah *knowledge* disebut sebagai

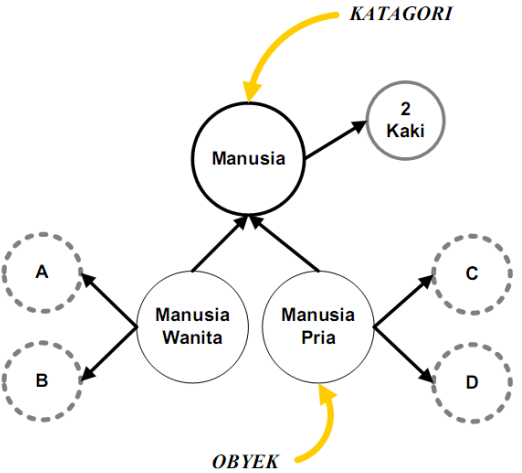
pengetahuan. Dalam pembangunan sistem berbasis pengetahuan, pengetahuan yang telah diekstrak direpresentasikan ke dalam bentuk yang dapat diproses oleh komputer. Menurut Firebaugh (1989), terdapat empat teknik yang telah dibuktikan efektif untuk representasi pengetahuan, yaitu jaringan semantik *(Semantic Network), frame dan script*, serta aturan produksi.

III.2. Jaringan Semantik

Dalam jaringan semantik, pengetahuan diorganisasikan dengan menggunakan jaringan yang disusun oleh dua komponen dasar, yaitu *node* dan *arc*. *Node* menyatakan objek, konsep atau situasi yang ditunjukkan oleh kotak atau lingkaran, sedangkan *arc* menyatakan hubungan antar-*node* yang ditunjukkan oleh tanda panah yang menghubungkan node-node dalam jaringan (Firebaugh, 1989).

Menurut Kusumadewi (2003) , jaringan semantik merupakan gambaran pengetahuan secara grafis, yang menunjukkan hubungan antar berbagai obyek. Jaringan semantik terdiri dari lingkaran-lingkaran yang menunjukkan obyek-obyek dan informasi tentang obyek-obyek tersebut. Obyek disini dapat berupa benda atau peristiwa. Antara 2 obyek dihubungkan dengan arc yang menunjukkan hubungan antar obyek.

a. Katagori
   Obyek-obyek yang mempunyai kemiripan karakteristik dapat digolongkan dalam kategori tertentu. Katagori merupakan pengorganisasian obyek yang merupakan representasi pengetahuan yang vital dalam *semantic net*. Sebagai contoh : jika ada dua objek ”elang” dan ”perkutut”, maka kedua objek tersebut dapat diturunkan dari kategori ”burung”.
b. Obyek
   Obyek merupakan individu tersendiri yang mempunyai sifat-sifat karakteristik yang spesifik. Dalam kaitan kategori, jika sebuah obyek diturunkan dari kategori, maka obyek akan mempunyai sifat dari katagori secara keseluruhan dan juga mempunyai sifat spesifik obyek itu sendiri. Berikut contoh jaringan semantik ditunjukkan pada Gambar III.3



Gambar III.3 Contoh Jaringan Semantik

Dari Gambar III.3. dapat dilihat bahwa obyek ”Manusia pria” merupakan turunan dari katagori ”Manusia”, obyek ”Manusia pria” selain mempunyai sifat-sifat individu (C dan D) juga mempunyai sifat dari ”Manusia” yaitu punya 2 kaki.

Object-Attribute-Value (OAV)

Object dapat berupa bentuk fisik atau konsep. Attribute adalah karakteristik atau sifat dari object tesebut. Values (Nilai) besaran/nilai/takaran spesifik dari attribute tersebut pada situasi tertentu, dapat berupa numerik, string atau boolean.

Sebuah *object* bisa memiliki beberapa attribute, biasa disebut OAV Multi-attribute.

Contoh representasi pengetahuan dengan OAV ditunjukkan pada Tabel III.1.

Tabel III.1. Representasi pengetahuan dengan OAV

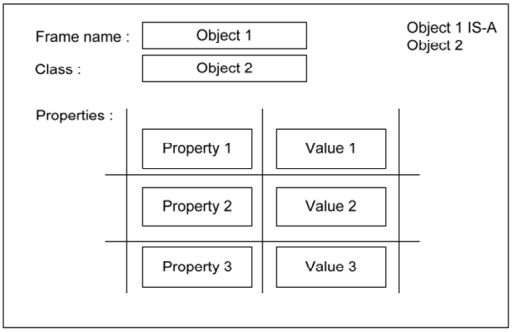
Object	Attribute	Value
Mangga	Warna	Hijau, Orange
Mangga	Berbiji	Tunggal
Mangga	Rasa	Asam, Manis
Mangga	Bentuk	Oval
Pisang	Warna	Hijau, Kuning
Pisang	Bentuk	Lonjong

III.3. Frame

Frame dan *script* digunakan untuk merepresentasikan pengetahuan dalam konteks dimana urutan kejadian dan objek muncul. Sebuah frame digambarkan dengan menggunakan jaringan dari node-node dan hubungan-hubungan. Level teratas dari frame menyatakan atribut-atribut sedangkan level terendah memiliki terminal dan slot yang harus diisi oleh data. *Script* menyerupai frame dengan informasi tambahan tentang urutan kejadian yang diharapkan serta tujuan dan rencana dari aktor yang terlibat (Firebaugh, 1989).

Menurut Minsky (1975), frame dapat dipandang sebagai struktur data statik yang digunakan untuk merepresentasikan situasi-situasi yang telah dipahami dan stereotipe. Struktur serupa frame seolah-olah mengatur pengetahuan kita tentang dunia sekitar. Kita berpindah ke situasi baru dengan cara memanggil informasi yang dibentuk oleh pengalaman masa lalu kita. Detail pengalaman masa lalu ini kemudian dibentuk atau diubah dengan merepresentasikan pengalaman masa lalu itu dengan merepresentasikan perbedaan-perbedaan individual untuk situasi yang baru ini.

Frame berupa kumpulan-kumpulan slot-slot yang merupakan atribut untuk mendeskripsikan pengetahuan. Pengetahuan yang termuat dalam slot dapat berupa kejadian, lokasi, situasi ataupun elemen-elemen lain. Frame digunakan untuk representasi pengetahuan deklaratif. Frame dapat disesain menyerupai bentuk *report card*. Adapun desain struktur Frame ditunjukkan pada Gambar III.4.



Gambar III.4 Contoh Frame

Frame tersebut juga memiliki field tambahan yang dinamakan dengan *Class* yang dapat diisi dengan sebuah nilai (misalnya object2) yang menyerupai nama dari frame lainyang berhubungan dengan object1. IS-A dari frame menerangkan bahwa object1 IS-A object2.

Dari Gambar III.4 dapat dilihat bahwa secara umum frame memiliki dua elemen dasar, yaitu slot dan facet yang merupakan subslot. Slot merupakan kumpulan atribut/properti yang menjelaskan objek yang direpresentasi oleh frame dan subslot menjelaskan pengetahuan atau prosedur dari atribut pada slot.

Subslot dapat berbentuk :

- Value : menjelaskan tentang nilai dari suatu atribut.
- Default : nilai yang digunakan jika suatu slot kosong atau tidak dideskripsikan pada frame instansiasi.
- Range : menandakan jenis dari informasi yang dapat muncul pada slot tersebut (misalnya dari 0 sampai 100)
- If Added : berisi informasi prosedural yang berupa suatu tindakan yang akan dikerjakan jika nilai dari slot diisi (atau berubah).
- If Needed : Facet ini digunakan pada kasus dimana tidak ada value pada slot. Suatu prosedur akan dikerjakan untuk memperoleh atau menghitung sebuah value.
- Other : Slot bisa berisi frame, rule, jaringan semantik ataupun tipe lain dari informasi.

Frame class merepresentasikan karakteristik umum dari suatu objek dimana dalam setiap frame tersebut dapat didefenisikan properti-properti umum yang biasanya dimiliki oleh semua objek dalam kelas tersebut.

Terdapat dua jenis properti, yaitu properti statis dan dinamis. Properti

statis merupakan fitur dari objek yang tidak dapat berubah, sedangkan propertis dinamis merupakan fitur yang dapat berubah selama sistem berjalan.

III.4. Script

Konsep script dikembangkan oleh Schank dan Abelson dari Universitas Yale. Tujuan dari penelitian ini adalah untuk mengembangkan teori yang tidak dimengerti oleh komputer. Script lebih menyerupai frame dengan penambahan informasi, termasuk tentang harapan rentetan kejadian dan tujuan serta perencanaan yang melibatkan para aktornya.

Script merupakan skema representasi pengetahuan yang sama dengan frame. Hanya saja frame menggambarkan objek sedangkan script menggambarkan urutan peristiwa. Penggambaran urutan peristiwa pada script menggunakan serangkaian slot yang berisi informasi tentang orang, objek dan tindakan-tindakan yang terjadi dalam suatu peristiwa. Contoh sederhana script dapat dilihat untuk script berangkat ke restauran seperti berikut ini :

SCRIPT Restoran
Jalur (Track) : Restoran swalayan (fast food)
Peran (Roles) : Tamu pelayan
Pendukung(Prop) : Counter, baki, makanan, uang, serbet, garam, merica, kecap sedotan, dan lain-lain
Kondisi masukan : tamu lapar-tamu punya uang
Adegan (Scene) 1 : Masuk

- Tamu parkir mobil
- Tamu masuk restoran
- Tamu Antri
- Tamu baca menu di list menu dan mengambil putusan tentang apa yang akan diminta.

Adegan 2 : Pesanan

- Tamu memberikan pesanan pada pelayan
- Pelayan mengambil pesanan dan meletakkan makanan di atas baki
- Tamu membayar

Adegan 3 : Makan

- Tamu mengambil serbet, sedotan, garam, dan lain-lain
- Tamu membawa baki makanan ke meja kosong
- Tamu makan dengan cepat

Adegan 4 : Pulang

- Tamu membersihkan meja
- Tamu membuang sampah
- Tamu meninggalkan restoran

- Tamu naik mobil dan pulang

Hasil

- Tamu merasa kenyang
- Uang tamu jadi kenyang
- Tamu senang
- Tamu kecewa
- Tamu sakit perut

Ada beberapa keistimewaan script yang perlu dicatat, yaitu :

1. Script menyediakan beberapa cara yang sangat alami untuk merepresentasikan ”suatu informasi yang lazim”, dengan masalah yang bersumber dari sistem AI dari permulaannya.
2. Script juga menyediakan struktur hierarki untuk merepresentasikan informasi melalui inklusi subscript dengan script.
3. Struktur representasi pengetahuan lainnya seperti aturan kaidah boleh jadi digunakan dalam cara alami dengan script formal.

Keistimewaan-keistimewaan ini memberikan script memiliki kemampuan seperti berikut ini :

1. Script dapat memprediksi kejadian dan menjawab pertanyaan tentang informasi yang tidak terinci di dalam baris cerita.
2. Script menyediakan suatu kerangka kerja untuk mengintegrasikan sekumpulan observasi ke dalam interpretasi yang jelas.
3. Script menyediakan skema untuk kejadian yang tidak biasanya.

Latihan

1. Pahami kembali representasi pengetahuan dengan menggunakan jaringan semantik, frame dan script
2. Buatlah contoh rancangan representasi pengetahuan dengan menggunakan jaringan semantik, frame dan script

BAB IV

REPRESENTASI PENGETAHUAN

Pokok Bahasan

IV.1. Logika dan Himpunan

IV.1.1 Logika Proposisi

IV.1.2 Logika Predikat

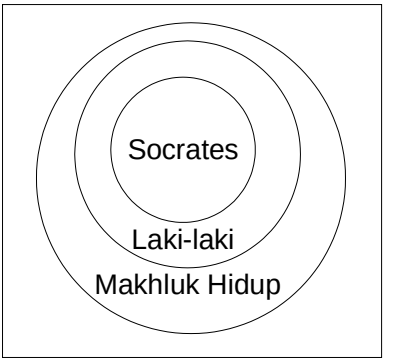
Tujuan mempelajari bab ini diharapkan dapat memahami logika dan himpunan dimana logika dan himpunan terdapat didalamnya logika proposisi dan logika predikat. Selain dengan aturan, frame dan jaringan semantik, pengetahuan dapat juga direpresentasikan dengan menggunakan simbol logika, yang studi aturannya merupakan bagian dari penalaran eksak. Bagian yang paling penting dari penalaran ini adalah untuk mengambil kesimpulan dari premis. Aplikasi komputer untuk melakukan penalaran yang telah dihasilkan dalam logika pemrograman dan pengembangan bahasa dasar logika seperti PROLOG. Logika juga sangat penting dalam sistem pakarsebagai mesin penarik kesimpulan dari fakta ke kesimpulan. Pada kenyataannya gambaran untuk istilah logika pemrograman dan sistem pakar merupakan sistem penalaran yang otomatis.

IV.1. Logika dan Himpunan

Pada mulanya logika dikembangkan oleh filosof Yunani Aristoteles pada abad ke 4 sebelum Masehi. Logika Aristoteles didasarkan pada silogisme. Dia menemukan 14 tipe dan 5 lagi ditemukan pada pertengahan waktu. Silogisme mempunyai dua premis dan satu konklusi yang disimpulkan dari premis. Berikut ini adalah beberapa contoh klasik dari silogisme :

Premise : Semua laki-laki adalah makhluk hidup
Premis : Socrates adalah laki-laki
Kesimpulan : Socrates adalah makhluk hidup

Dalam silogisme, premis merupakan fakta dari kesimpulan yang harus diikuti. Silogisme adalah salah satu cara merepresentasi pengetahuan. Cara lainnya untuk merepresentasikan pengetahuan adalah dengan diagram Venn, seperti yang ditunjukkan pada Gambar IV.1.



Gambar IV.1. Diagram Venn

Lingkaran paling luar merepresentasikan semua makhluk hidup. Lingkaran di dalamnya merepresetasikan laki-laki digambarkan secara keseluruhan dalam lingkaran makhluk hidup untuk menunjukkan bahwa semua alki-laki adalah makhluk hidup. Karena Socrates adalah seorang laki-laki, maka lingkaran yang merepresentasikannya digambarkan dalam manusia.

Dalam istilah matematika, lingkaran dari diagram Venn merepresentasikan sebuah himpunan yang merupakan kumpulan objek-objek. Objek dalam himpunan dinamakan dengan elemen/unsur. Beberapa contoh himpunan :

A={1,3,5,7}
B={1,2,3,4,5}
C={0,2,4…}
D={...,-4,-2,0,2,4,...}
E={pesawat,ballon,blimps,jet}
F={pesawat,ballon}

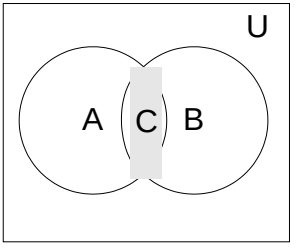
Dimana titik-titik,..., dinamakan *ellipses*, yang menunjukkan istilah seterusnya sampai tak berhingga.

Simbol Yunani epsilon ε menunjukkan bahwa suatu elemen merupakan anggota dari suatu himpunan. Sebagai contoh, 1 ε A, artinya bilangan 1 adalah elemen dari himpunan A yang telah didefinisikan sebelumnya. Jika suatu elemen bukan anggota dari suatu himpunan maka simbol yang digunakan adalah $\notin$. Misalnya 2 $\notin$ A.

Jika dua himpunan sembarang, misalkan X dan Y, didefinisikan bahwa setiap elemen X merupakan elemen Y, maka x adalah subset dari Y dan

dengan simbol matematika dapat dituliskan dalam bentuk $X \subset Y$ atau $Y \supset X$. Dari definisi tentang himpunan bagian, maka dapt dikatakan bahwa setiap himpunan merupakan himpunan bagian dari dirinya sendiri. Sedangkan himpunan bagian yang bukan merupakan himpunan dari dirinya sendiri dinamakan dengan *proper subset.* Sebagai contoh, himpunan yang telah didefinisikan sebelumnya adalah merupakan *proper subset* dari Y.

Gambar IV.2 menggambarkan himpunan bagian dalam bentuk irisan *(intersection)* dari dua himpunan.



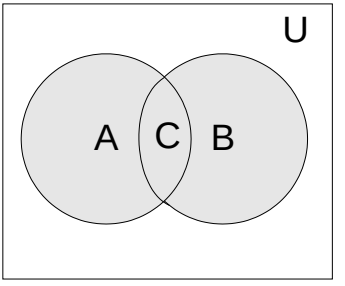
Gambar IV.2. Irisan Himpunan

$C = A \cap B$

$C = \{x \in U \mid (x \in A) \wedge (x \in B)\}$

Dimana : $\cap$ menyatakan irisan himpunan
| dibaca "sedemikian hingga"
$\wedge$ operator logika AND

Gambar IV.3 menggambarkan himpunan bagian dalam bentuk gabungan *(union)* dari dua himpunan.



Gambar IV.3. Himpunan Gabungan

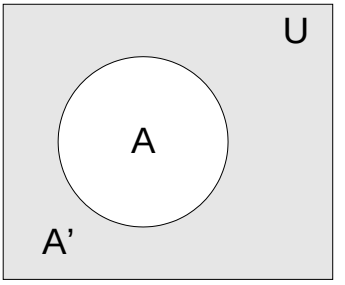
Operasi himpunan lainnya adalah gabungan *(union)*, yaitu semua elemen ada di A atau di B. Pernyataan ini juga dapat ditulis dalam bentuk :

$C = A \cup B$

$C = \{x \cup U \mid (x \in A) \vee (x \in B)\}$

Dimana : $\cup$ menyatakan gabungan himpunan
$\vee$ operator logika OR

Gambar IV.4 menggambarkan contoh dari komplemen yaitu himpunan A adalah himpunan yang bukan elemen dari A.



Gambar IV.4. Komplemen

Selanjutnya komplemen dari himpunan A adalah himpunan yang bukan elemen dari A, yang dapat dituliskan :

$A' = \{x \in U \mid \sim(x \in A)\}$

Dimana : ' menyatakan komplemen himpunan
~ operator logika NOT

IV.1.1 Logika Proposisi

Logika proposisi adalah suatu pernyataan yang dapat bernilai benar (B) atau salah (S). Simbol-simbol seperti P dan Q menunjukkan proposisi. Dua atau lebih proposisi daqpat digabungkan dengan menggunakan operator logika :

a. Konjungsi : ^ (AND)
b. Disjungsi : $\vee$ (OR)
c. Negasi : $\neg$ (NOT)
d. Implikasi : $\longrightarrow$ (IF Then)
e. Ekuivalensi : $\Leftrightarrow$

a. Operator NOT
Operator NOT digunakan untuk memberikan nilai negasi (lawan) dari pernyataan yang telah ada. Tabel IV.1 menunjukkan tabel kebenaran untuk operator NOT.

Tabel IV.1 Tabel Kebenaran Operator NOT

P	NOT(P)
B	S
S	B

b. Operator AND

Operator AND digunakan untuk mengkombinasikan 2 buah proposisi. Hasil yang diperoleh akan bernilai benar jika kedua proposisi bernilai benar, dan akan bernilai salah jika salah satu dari kedua proposisi bernilai salah. Tabel IV.2 menunjukkan tabel kebenaran untuk operator AND.

Tabel IV.2 Tabel Kebenaran Operator AND

P	Q	P AND Q
B	B	B
B	S	S
S	B	S
S	S	S

c. Operator OR

Operator OR digunakan untuk mengkombinasikan 2 buah proposisi. Hasil yang diperoleh akan bernilai benar jika salah satu dari kedua proposisi bernilai benar, dan akan bernilai salah jika kedua proposisi bernilai salah. Tabel IV.3 menunjukkan tabel kebenaran untuk operator OR.

Tabel IV.3 Tabel Kebenaran Operator OR

P	Q	P OR Q
B	B	B
B	S	B
S	B	B
S	S	S

d. Implikasi

Implikasi : Jika P maka Q akan menghasilkan nilai salah jika P benar dan Q salah, selain itu akan selalu bernilai benar. Tabel IV.4 menunjukkan tabel kebenaran untuk operator Implikasi.

Tabel IV.4 Tabel Kebenaran Operator OR

P	Q	$P \rightarrow Q$
B	B	B
B	S	S
S	B	B
S	S	B

e. Ekuivalensi

Ekuivalensi akan menghasilkan nilai benar jika P dan Q keduanya benar atau keduanya benar atau keduanya salah, Tabel IV.5 menunjukkan tabel kebenaran untuk operator ekuivalensi.

Tabel IV.5 Tabel Kebenaran Ekuivalensi

P	Q	$P \Leftrightarrow Q$
B	B	B
B	S	S
S	B	S
S	S	B

Untuk melakukan inferensi pada logika proposisi dapat dilakukan dengan menggunakan resolusi. Resolusi adalah suatu aturan untuk melakukan inferensi yang dapat berjalan secara efisien dalam suatu bentuk khusus. Bentuk khusus tersebut dikenal dengan nama *conjunctive normal form* (CNF). Bentuk CNF ini memiliki ciri-ciri sebagai berikut :

- Setiap kalimat merupakan disjungsi literal
- Semua kalimat terkonjungsi secara implisit

Kalimat yang ditulis dengan menggunakan logika proposisi dapat dikonversi ke bentuk CNF. Kita perlu menghapus operator-operator selain OR tanpa

harus mengubah arti dari kalimat tersebut.
Untuk mengubah suatu kalimat ke dalam bentuk CNF, dapat digunakan langkah-langkah sebagai berikut :

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

Pada logika proposisi, prosedur untuk membuktikan proposisi P dengan beberapa aksioma F yang telah diketahui, dengan menggunakan resolusi, dapat dilakukan melalui algoritma sebagai berikut :

1. Konversikan semua proposisi F ke bentuk CNF
2. Negasikan P, dan konversikan hasil negasi tersebut ke bentuk klausa. Tambahkan ke himpunan klausa yang telah ada pada langkah 1.
3. Kerjakan hingga terjadi kontradiksi atau proses tidak mengalami kemajuan :
   a. Seleksi 2 klausa sebagai klausa parent
   b. Bandingkan *(resolve)* secara bersama-sama. Klausa hasil *resolve* tersebut dinamakan *resolvent.* Jika ada pasangan literal L dan ¬L, eliminir dari *resolvent.*
   c. Jika *resolvent* berupa klausa kosong, maka ditemukan kontradiksi. Jika tidak, tambahkan ke himpunan klausa yang telah ada.

Contoh IV.1 :
Diketahui basis pengetahuan (fakta-fakta yang bernilai benar) sebagai berikut :

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

Buktikan kebenaran R !
Apabila kita ingin membuktikan kebenaran R dengan menggunakan resolusim maka pertama-tama kita harus ubah dulu keempat fakta diatas menjadi bentuk CNF. Konversi ke CNF dapat dilakukan sesuai pada Tabel IV.6.

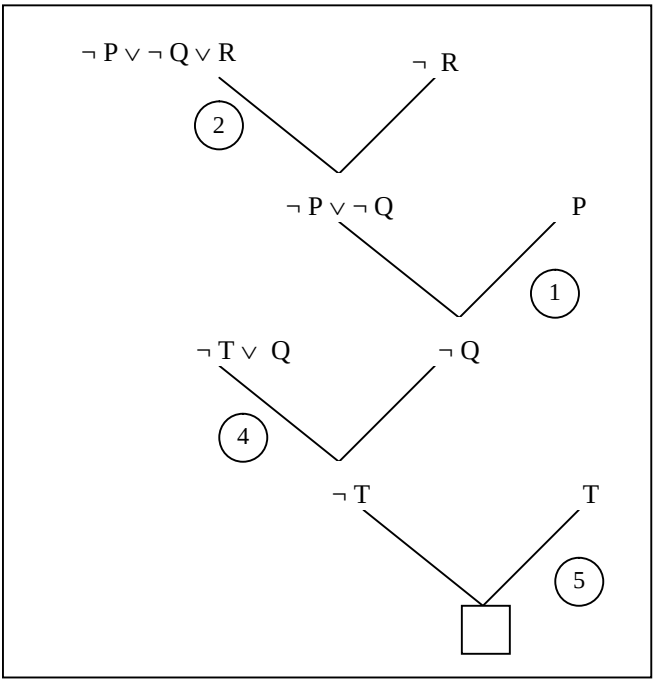
Tabel IV.6 Konversi ke CNF

Kalimat	Langkah-langkah	CNF
1. P	sudah merupakan bentuk CNF	P
2. $(P \wedge Q) \rightarrow R$	o Menghilangkan implikasi : $\neg (P \wedge Q) \vee R$ o Mengurangi lingkup negasi : $(\neg P \vee \neg Q) \vee R$ o Gunakan asosiatif : $\neg P \vee \neg Q \vee R$	$\neg P \vee \neg Q \vee R$
3. $(S \vee T) \rightarrow Q$	o Menghilangkan implikasi : $\neg (S \vee T) \vee Q$ o Mengurangi lingkup negasi : $(\neg S \wedge \neg T) \vee Q$ o Gunakan distributif : $(\neg S \vee Q) \wedge (\neg T \vee Q)$	$\neg S \vee Q$ $\neg T \vee Q$
4. T	Sudah merupakan bentuk CNF	T

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

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

Dengan demikian resolusi dapat dilakukan untuk membuktikan R sebagaimana terlihat pada Gambar IV.4



Gambar IV.4. Resolusi Pada Logika Proposisi

Contoh apabila diterapkan dalam kalimat :

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

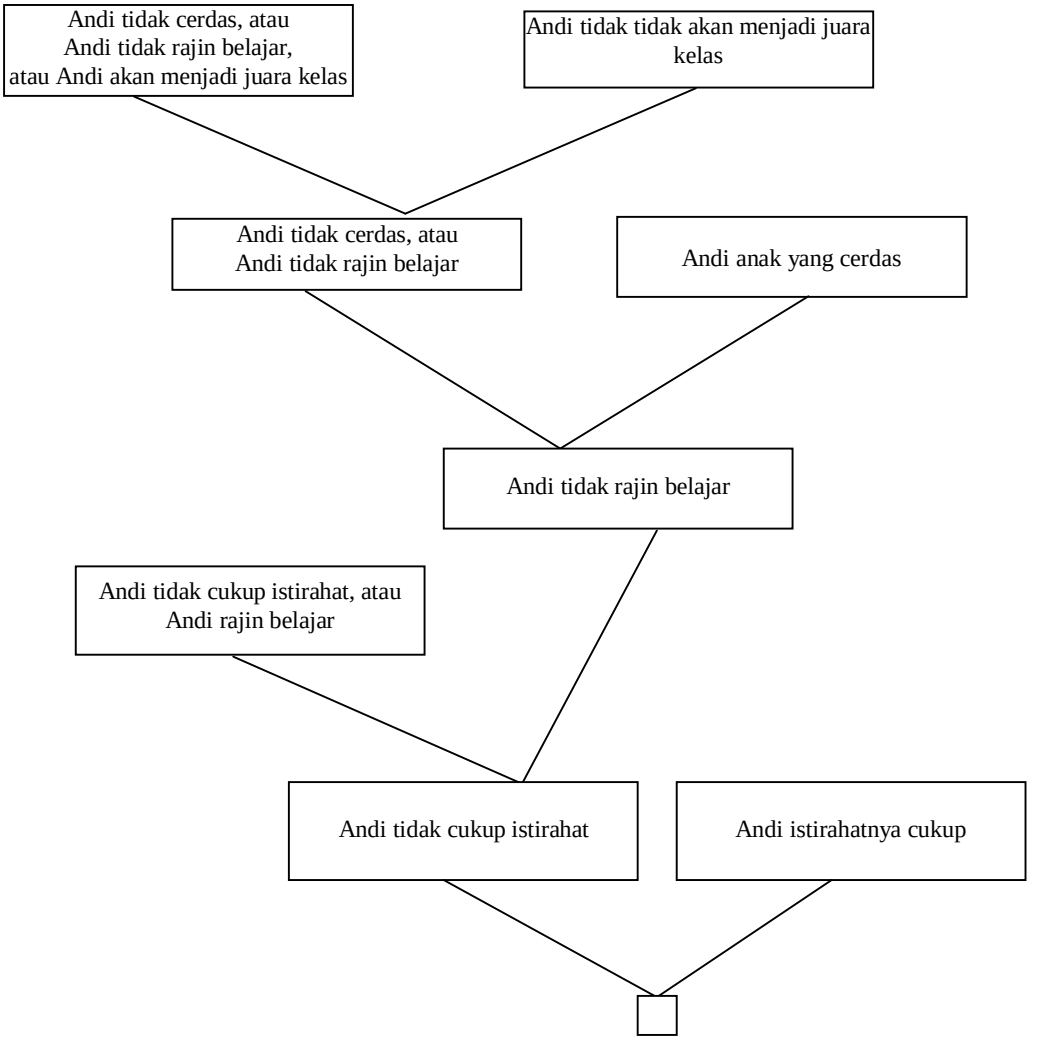
Kalimat yang terbentuk :

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

Setelah dilakukan ke bentuk CNF, didapat :

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

Gambar IV.5 menunjukkan pohon aplikasi resolusi untuk kejadian diatas.



Gambar IV.5. Resolusi Pada Logika Proposisi dengan pernyataan lengkap.

IV.1.2 Logika Predikat

A. Representasi Fakta Sederhana

Misalkan diketahui fakta-fakta sebagai berikut :

- Andi adalah seorang laki-laki : A
- Ali adalah seorang laki-laki : B
- Amir adalah seorang laki-laki : C

- Anto adalah seorang laki-laki : D
- Agus adalah seorang lakilaki : E

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

Logika predikat digunakan untuk merepresentasikan hal-hal yang tidak dapat direpresentasikan dengan menggunakan logika proposisi. Pada logika predikat kita dapat merepresentasikan fakta-fakta sebagai suatu pernyataan yang disebut dengan *wff (well-formed formula).*

Pada contoh diatas, dapat dituliskan :

Laki2(x)

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

Contoh IV.2 :

Misalkan terdapat pernyataan-pernyataan sebagai berikut :

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

Kedelapan pernyataan di atas dapat dibawa ke bentuk logika predikat, dengan mengunakan operator-operator : $\rightarrow$ (implikasi), $\neg$ (not), $\wedge$ (and), $\vee$ (or), $\forall$ (untuk setiap), $\exists$ (terdapat), sebagai berikut :

1. mahasiswa (Andi)
2. Elektro (Andi)
3. $\forall x$: Elektro (x) $\rightarrow$ Teknik (x)
4. sulit (Kalkulus)
5. $\forall x$: Teknik (x) $\rightarrow$ suka (x, Kalkulus) $\vee$ benci (x, Kalkulus)
6. $\forall x$: $\exists y$: suka (x,y)
7. $\forall x$: $\forall y$: mahasiswa (x) $\wedge$ sulit (y) $\wedge$ $\neg$hadir (x,y) $\rightarrow$ $\neg$suka(x,y)
8. $\neg$hadir (Andi, Kalkulus)

Andaikan kita akan menjawab pertanyaan “

”Apakah Andi suka mata kuliah kalkulus?”

Maka dari pernyataan ke-7 kita akan membuktikan bahwa Andi tidak suka dengan mata kuliah kalkulus. Dengan menggunakan penalaran backward bisa dibuktikan bahwa :

¬suka (Andi, Kalkulus)

Sebagai berikut :

¬suka(Andi,Kalkulus)
↑ (7, substitusi)
Mahasiswa (Andi) ∧
sulit (Kalkulus) ∧
¬hadir (Andi,Kalkulus)
↑ (1)
sulit (Kalkulus) ∧
¬hadir (Andi,Kalkulus)
↑ (4)
¬hadir (Andi,Kalkulus)
↑ (8)

□

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

Latihan

1. Pahami kembali logika proposisi dan logika predikat
2. Buatlah contoh rancangan representasi pengetahuan dengan menggunakan logika proposisi dan logika predikat

BAB V

REPRESENTASI PENGETAHUAN

Pokok Bahasan

V.1. Sistem Produksi

V.1.1. Defenisi Sistem Produksi

V.1.2. Kaidah Produksi, Pengetahuan dan Kaidah Inferensi

Setelah mempelajari uraian bab ini diharapkan dapat memahami defensi dari sistem produksi, kaidah produksi, pengetahuan dan kaidah inferensi.

V.1. Sistem Produksi

Konsep dari sistem produksi atau kaidah produksi telah diperkenalkan oleh Post pada tahun 1943. Konsep ini kemudian ditampilkan kembali dalam konteks proses bahasa alami dalam kaidah-kaidah penulisan dari Chomsky pada tahun 1957. Aturan produksi yang diusulkan adalah untuk memodelkan penyelesaian permasalahan tingkah laku manusia oleh Newell dan Simon pada tahun 1972 dan sejal itu Alan Newell menjadi salah satu pendukung utama dari teknik ini sebagai sebuah alat yang sangat berguna untuk sistem pakar. Kaidah produksi menjadi acuan yang sangat sering digunakan oleh sistem inferensi, sistem berbasis kaidah dan dalam kasus penyelesaian masalah tingkah laku manusia, ataupun dalam produksi sederhana.

V.1.1. Defenisi Sistem Produksi

Pengetahuan dalam sistem produksi boleh juga direpresentasikan oleh himpunan kaidah dalam bentuk :

IF [kondisi] THEN [aksi]

Dengan sebuah kontrol sistem dan basis data. Kontrol sistem memberikan aturan penerjemahan dan pengurutan. Basis data beraksi dengan konteks cadangan untuk record yang kondisinya dievaluasi oleh kaidah dan informasi dimana kaidah akan beraksi. Berikutnya untuk sintaks IF – THEN, kaidah produksi juga sering digambarkan sebagai pasangan-pasangan berikut kondisi-aksi, anteceden-konsequent, pola-aksi, situasi-responsi.

Contoh-contoh produksi dari pengalaman sehari-hari yaitu :

IF [mengendarai 15 mil per jam melewati batas AND melihat

melalui kaca spion ada lampu merah yang berkedip-kedip]

THEN [pinggirkan AND berhenti]

IF [Sakit kepala AND sakit tenggorokan AND hidung tersumbat]
THEN [ambil aspirin AND istirahat]

IF [barometer turun AND petir di barat laut]
THEN [arahkan ke pelabuhan yang aman]

Contoh-contoh lain yang lebih mendekati secara analitik untuk mengklasifikasikan suatu objek dimulai dengan kaidah-kaidah berikut ini :

IF [kategori adalah sebuah form OR sebuah pewarnaan OR sebuah teksture]
THEN [objek mempunyai sebuah permukaan]

IF [kategori adalah sebuah permukaan OR jelek OR kualitas tactile]
THEN [object mempunyai kualitas luar]

IF [kategori adalah sebuah ukuran OR kualitas luar OR massa OR suatu zat]
THEN [object mempunyai kualitas fisik]
……

Struktur intuitif secara alami untuk representasi pengetahuan yang telah dihasilkan oleh sistem produksi merupakan metode yang lebih disukai oleh banyak sistem pakar, termasuk perintis program DENDRAL dan MYCIN. Sebagai contoh, kaidah produksi dari sistem tersebut seperti yang ditunjukkan dibawah ini.

DENDRAL

IF [the spectrum of the molecule has two peaks at masses x_1 dan x_2 such that

a) $x_1 + x_2 = M + 28$ AND
b) $x_1 - 28$ is a high peak AND
c) $x_1 - 28$ is a high peak AND
d) at least one of x_1 or x_2 is high]

THEN [the molecule containts a ketone group]

MYCIN

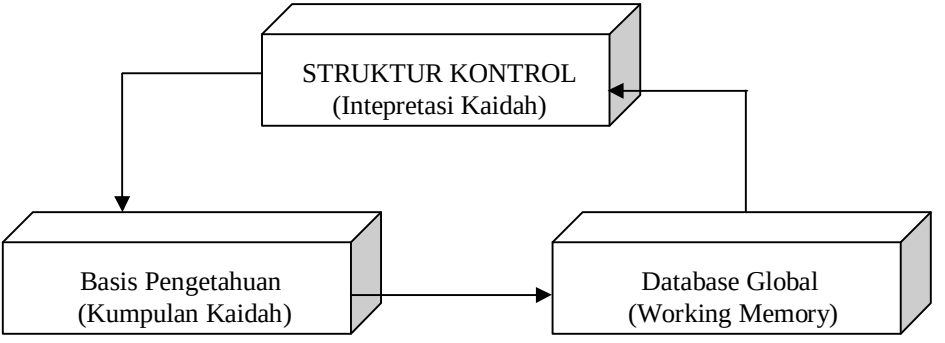
IF [a) the stain of the organism is gramneg AND
b) the morphology of the organism is rod AND
c) the patient is a compromised host]

THEN [there is suggestive evidence (0,6) that the identity of the organism is pseudomonas]

Dari kesamaan perumusan kedua kaidah produksi itu, kita catat suatu gambaran atraktif pertama dari sistem produksi secara formal : pemisahan secara rapi dari struktur program dari data. Ada tiga elemen utama dari semua sistem produksi, yaitu :

1. Database global
   Database global merupakan struktur data utama dari sistem produksi. Database mungkin mempunyai jangkauan dari sebuah daftar sederhana atau matriks kecil hingga kekompleksan, relasi dan struktur indeks. Hal ini merupakan struktur dasar di mana kaidah produksi dapat beroperasi. Hal ini juga merupakan struktur dinamis dan berubah-ubah secara kontinyu sebagai hasil dari operasi kaidah produksi. Database global juga menjadi acuan untuk konteks, memori cadangan jangka pendek, atau memori kerja.
2. Kaidah produksi
   Sebagaimana yang diindikasikan oleh contoh diatas, kaidah produksi mempunyai bagian kondisi (IF) yang disebut bagian kanan dan aksi (THEN) disebut bagian kiri. Jika sisi kiri kadang-kadang dinamakan kondisi atau premis yang dipenuhi oleh database, maka kaidah-kaidah dapat diterapkan dan subjek menjadi pemicu bagi sistem kontrol.
3. Sistem kontrol
   Sistem kontrol merupakan program penterjemah yang esensial untuk mengontrol urutan dimana kaidah-kaidah produksi dipicu dan menyelesaikan konflik jika lebih dari satu kaidah yang diaplikasikan. Sistem kontrol secara berulang-ulang mengaplikasikan kaidah-kaidah untuk database hingga sebuah gambaran dari tujuan yang dihasilkan. Kemudian mendeteksi kejadian seperti tujuan dan record kaidah yang telah diaplikasikan untuk mencapainya bagi referensi sebelumnya.

Hubungan antara ketiga elemen tersebut dan iterasi alami dari operasi sistem produksi diilustrasikan pada Gambar V.1



Gambar V.1 Komponen Sistem Produksi

Masalah yang dihasilkan oleh sistem nyata untuk tiga elemen ini dinamakan masalah representasi *(representation problem)*. Hal ini merupakan masalah utama dalam rancangan sistem produksi sembarang.

V.1.2. Kaidah Produksi, Pengetahuan dan Kaidah Inferensi

Kaidah produksi dituliskan dalam bentuk pernyataan IF-THEN (Jika-Maka). Pernyataan ini menghubungkan bagian premis (IF) dan bagian kesimpulan (THEN) yang dituliskan dalam bentuk :

IF [premis] THEN [konklusi].

Apabila bagian premis dipenuhi maka bagian kkonklusi akan bernilai benar. Sebuah kaidah terdiri dari klausa-klausa. Sebuah klausa mirip dengan sebuah kalimat dengan subyek, kata kerja dan obyek yang menyatakan suatu fakta. Suatu aturan juga dapat terdiri dari beberapa premis dan lebih dari satu konklusi. Untuk merumuskan beberapa domain pengetahuan secara akurat diperlukan banyak kaidah produksi. Aturan-aturan ini menyediakan rincian obyek, karakteristik, dan tindakan-tindakan yang harus diambil. Untuk dapat meliputi suatu obyek diperlukan banyak rincian aturan. Aturan-aturan ini biasanya saling berkaitan dan saling mengacu. Aturan ini membentuk pangkalan pengetahuan yang kemudian menjadi bagian sistem produksi.

Biasanya pengetahuan diturunkan dalam bentuk pernyataan linguistik dari pakar (Negoita,1985). Sebagai contoh :

bercak pada daun is sangat banyak
ujung daun kering is banyak

Dalam contoh pertama, variabel linguistik ”bercak pada daun” memiliki nilai linguistik ”sangat banyak”, sedangkan pada contoh kedua, variabel linguistik ”ujung daun kering” memiliki nilai linguistik ”banyak”.

Bagian premis dalam aturan produksi dapat memiliki lebih dari satu proposisi. Proposisi-proposisi tersebut dihubungkan dengan menggunakan operator logika AND atau OR. Sebagai contoh :

bercak pada daun is sangat banyak
AND ujung daun kering is banyak

Aturan produksi digunakan untuk menyatakan hubungan antara dua bentuk pernyataan linguistik yang masing-masing merupakan bagian premis dan konklusi/kesimpulan. Sebagai contoh :

IF bercak pada daun is sangat banyak
AND ujung daun kering is banyak
THEN penyakit bercak ungu is puso

Ada dua tipe kaidah yang umum dalam AI, yaitu pengetahuan dan inferensi. Kaidah pengetahuan atau kaidah deklaratif menyatakan semua fakta dan hubungan tentang suatu permasalahan. Kaidah inferensi atau kaidah prosedural pada sisi lain merupakan nasihat atau saran tentang bagaimana menyelesaikan suatu masalah yang diberikan dengan fakta tertentu yang diketahui.

Sebagai contoh asumsikan bahwa Anda seorang pebisnis dari penjualan dan pembelian emas. Kaidah pengetahuannya seperti yang terlihat berikut ini :

- Kaidah 1 IF konflik internasional dimulai THEN harga emas naik.
- Kaidah 2 IF tingkat inflasi mengalami kemunduran THEN harga emas turun
- Kaidah 3 IF konflik internasional berlangsung selama lebih dari 7 hari AND IF konflik ini terjadi di Timur Tengah THEN beli emas.

Kaidah Inferensi (prosedural) seperti yang diberikan berikut ini

- Kaidah 1 IF Data dibutuhkan tidak dalam sistem THEN permintaannya dari pengguna
- Kaidah 2 IF lebih dari satu kaidah digunakan THEN nonaktifkan kaidah sembarang yang tidak menambah data baru

Kaidah inferensi berisi kaidah tentang kaidah. Tipe kaidah ini dinamakan dengan *metarule*. Mereka menyinggung kaidah lain (atau bahkan sesama mereka).

Perekayasa pengetahuan memisahkan dua tipe ini di mana kaidah pengetahuan menjadi basis pengetahuan sedangkan kaidah inferensi menjadi bagian dari mesin inferensi.

Representasi kaidah terutama dapat diaplikasikan bila dibutuhkan untuk merekomendasikan suatu bagian aksi berdasarkan kejadian yang dapat diobservasi. Ada beberapa keuntungan penggunaan kaidah, yaitu :

- Kaidah mudah dimengerti. Mereka mudah disampaikan karena merupakan bentuk alami dari pengetahuan.
- Inferensi dan penjelasan mudah diperoleh atau diturunkan.
- Modifikasi dan perawatan relatif lebih mudah.
- Ketidakpastian lebih mudah dikombinasikan dengan kaidah.
- Setiap kaidah sering saling independent dari semua kaidah.

Selain itu terdapat pula keterbatasan dari representasi kaidah ini, yaitu :

- Pengetahuan yang kompleks membutuhkan beribu-ribu kaidah, yang mungkin agak sukar membuatnya, baik untuk menggunakan sistem maupun untuk perawatannya.
- Pembangun menyukai kaidah; oleh karenanya mereka mencoba kekuatan semua pengetahuan ke dalam kaidah dibandingkan

mencari representasi yang lebih sesuai.

- Sistem dengan banyak kaidah mungkin mempunyai batasan pencarian dalam kontrol program. Beberapa program mempunyai kesulitan dalam mengevaluasi sistem berbasis kaidah dan membuat inferensi.

Representasi kaidah mempunyai karakteristik-karakteristik seperti yang ditunjukkan dalam tabel berikut :

Tabel V.1 Karakteristik dari representasi kaidah

	Bagian pertama	Bagian Kedua
Nama	Premis Antecedent Situasi IF	Konklusi Konsekuen Aksi THEN
Alami	Kondisi, sama dengan Pengetahuan deklaratif	Resolusi, sama dengan Pengetahuan prosedural
Ukuran	Dapat mempunyai Banyak IF	Biasanya hanya mempunyai satu konklusi
Pernyataan	Pernyataan AND Pernyataan OR	Semua kondisi harus benar untuk konklusi benar Jika ada kondisi Pernyataan OR benar maka konklusinya benar

Latihan

1. Pahami kembali repsentasi pengetahuan sistem produksi
2. Buatlah contoh rancangan representasi pengetahuan dengan menggunakan sistem produksi

BAB VI

METODE INFERENSI

Pokok Bahasan
VI.1 Forward Chaining
VI.2 Backward Chaining

Setelah mempelajari bab ini diharapkan dapat memahami maksud dari *forward chaining* dan *backward chaining.*

VI.1 Forward Chaining

Forward Chaining merupakan pencocokan fakta atau pernyataan dimulai dari bagian sebelah kiri (IF dulu). Dengan kata lain, penalaran dimulai dari fakta terlebih dahulu untuk menguji kebenaran hipotesis. Contoh aturan-aturan ditunjukkan pada Tabel VI.1

Tabel VI.1 Contoh Aturan-aturan

No	Aturan
R-1	IF A & B THEN C
R-2	IF C THEN D
R-3	IF A & E THEN F
R-4	IF A THEN G
R-5	IF F & G THEN D
R-6	IF G & E THEN H
R-7	IF C & H THEN I
R-8	IF I & A THEN J
R-9	IF G THEN J
R-10	IF J THEN K

Contoh VI.1 :
Pada Tabel VI.1 terlihat ada 10 aturan yang tersimpan dalam basis pengetahuan. Fakta awal yang diberikan hanya : A & F (artinya : A dan F bernilai benar). Ingin dibuktikan apakah K bernilai benar (hipotesis : K)?
Langkah-langkah inferensi adalah sebagai berikut :

- Dimulai dari R-1. A merupakan fakta sehingga bernilai benar, sedangkan B belum bisa diketahui kebenarannya, sehingga C-pun juga belum bisa diketahui kebenarannya. Oleh karena itu kita tidak mendapatkan informasi apapun pada R-1 ini. Sehingga kita menuju ke

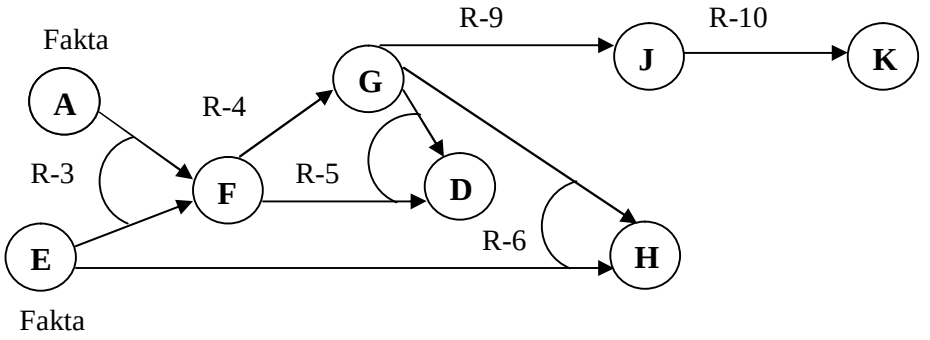
R-2.

- Pada R-2, kita tidak mengetahui informasi apapun tentang C, sehingga kita juga tidak bisa memastikan kebenaran D. Oleh karena itu kita tidak mendapatkan informasi apapun pada R-1 ini. Sehingga kita menuju ke R-3.
- Pada R-3, baik A maupun E adalah fakta sehingga jelas benar. Dengan demikian F sebagai konsekuen juga ikut benar. Sehingga sekarang kita mempunyai fakta baru yaitu F. Karena F bukan hipotesis yang hendak kita buktikan (=K), maka penelusuran kita lanjutkan ke R-4.
- Pada R-4, A adalah fakta sehingga jelas benar. Dengan demikian G sebagai konsekuen juga ikut benar. Sehingga sekarang kita mempunyai fakta baru yaitu G. Karena G bukan hipotesis yang hendak kita buktikan (=K), maka penelusuran kita lanjutkan ke R-5.
- Pada R-5, baik F maupun G bernilai benar berdasarkan aturan R-3, dan R-4. Dengan demikian D sebagai konsekuen juga ikut benar. Sehingga sekarang kita mempunyai fakta baru yaitu D. Karena D bukan hipotesis yang hendak kita buktikan (=K), maka penelusuran kita lanjutkan ke R-6.
- Pada R-6, baik A maupun G adalah benar berdasarkan fakta dan R-4. Dengan demikian H sebagai konsekuen juga ikut benar. Sehingga sekarang kita mempunyai fakta baru yaitu H. Karena H bukan hipotesis yang hendak kita buktikan (=K), maka penelusuran kita lanjutkan ke R-7.
- Pada R-7, meskipun H benar berdasarkan R-6, namun kita tidak tahu kebenaran C, sehingga I-pun juga belum bisa diketahui kebenarannya. Oleh karena itu kita tidak mendapatkan informasi apapun pada R-7 ini. Sehingga kita menuju ke R-8.
- Pada R-8, meskipun A benar karena fakta, namun kita tidak tahu kebenaran I, sehingga J-pun juga belum bisa diketahui kebenarannya. Oleh karena itu kita tidak mendapatkan informasi apapun pada R-8 ini. Sehingga kita menuju ke R-9.
- Pada R-9, J bernilai benar karena G benar berdasarkan R-4. Karena J bukan hipotesis yang hendak kita buktikan (=K), maka penelusuran kita lanjutkan ke R-10.
- Pada R-10, K bernilai benar karena J benar berdasarkan R-9. Karena H sudah merupakan hipotesis yang hendak kita buktikan (=K), maka terbukti bahwa K adalah benar.

Tabel munculnya fakta baru pada saat inferensi terlihat pada Tabel VI.2. Sedangkan alur inferensi terlihat pada Gambar VI.1

Tabel VI.2 Fakta baru

Aturan	Fakta Baru
R-3	F
R-4	G
R-5	D
R-6	H
R-9	J
R-10	K



Gambar VI.1 *Forward Chaining*

VI.2 *Backward Chaining*

Backward Chaining merupakan pencocokan fakta atau pernyataan dimulai dari bagian sebelah kanan (THEN dulu). Dengan kata lain, penalaran dimulai dari hipotesis terlebih dahulu, dan untuk menguji kebenaran hipotesis tersebut dicari harus dicari fakta-fakta yang ada dalam basis pengetahuan.

Contoh VI.2 :

Seperti halnya pada Contoh VI.1, pada Tabel VI.1 terlihat ada 10 aturan yang tersimpan dalam basis pengetahuan. Fakta awal yang diberikan hanya : A & F (artinya : A dan F bernilai benar).

Ingin dibuktikan apakah K bernilai benar (hipotesis : K)?

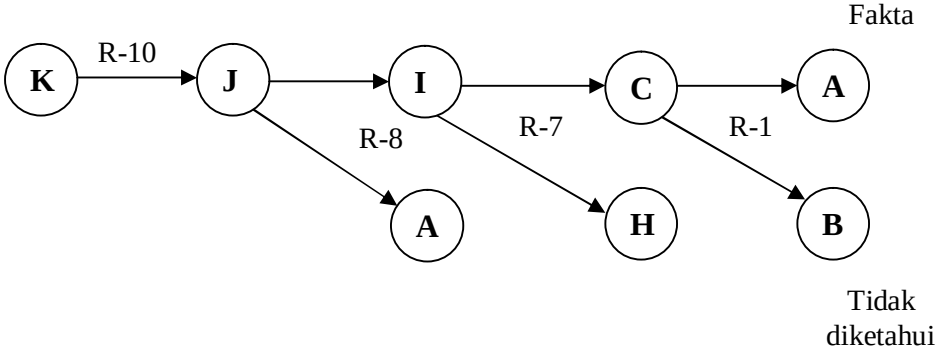
Langkah-langkah inferensi adalah sebagai berikut :

- Pertama-tama kita cari terlebih dahulu mulai dari R-1, aturan mana yang memiliki konsekuen K. Ternyata setelah ditelusur, aturan dengan

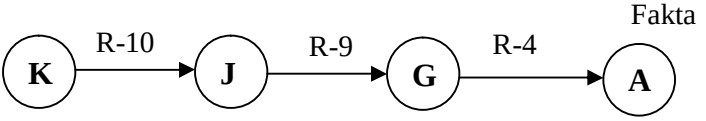
konsekuen K baru ditemukan pada R-10. Untuk membuktikan bahwa K benar, maka perlu dibuktikan bahwa J benar.

- Kita cari aturan yang memiliki konsekuen J. Kita mulai dari R-1, dan ternyata kita baru akan menemukan aturan dengan konsekuen J pada R-8. Untuk membuktikan bahwa J benar, maka perlu dibuktikan bahwa I dan A benar. Untuk membuktikan kebenaran I, kita perlu cari aturan dengan konsekuen I, ternyata ada di R-7.
- Untuk membuktikan I benar di R-7, kita perlu buktikan bahwa C dan H benar. Untuk itu kitapun perlu mencari aturan dengan konsekuen C, dan ada di R-1.
- Untuk membuktikan C benar di R-1, kita perlu buktikan bahwa A dan B benar. A jelas benar karena A merupakan fakta. Sedangkan B kita tidak bisa membuktikan kebenarannya, karena selain bukan fakta, di dalam basis pengetahuan juga tidak ada aturan dengan konsekuen B. Dengan demikian maka dari penalaran ini kita tidak bisa buktikan kebenaran dari hipotesis K. Namun demikian, kita masih punya alternatif lain untuk melakukan penalaran.
- Kita lakukan backtracking. Kita ulangi lagi dengan pembuktian kebenaran C dengan mencari aturan lain dengan konsekuen C. Ternyata tidak ditemukan.
- Kita lakukan backtracking lagi dengan mencari aturan dengan konsekuen I, ternyata juga tidak ada.
- Kita lakukan backtracking lagi dengan mencari aturan dengan konsekuen J, ternyata kita temukan pada R-9. Sehingga kita perlu buktikan kebenaran G.
- Kita mendapatkan R-4 dengan konsekuen G. Kita perlu buktikan kebenaran A. Karena A adalah fakta, maka terbukti bahwa G benar. Dengan demikian berdasarkan penalaran ini bisa dibuktikan bahwa K bernilai benar.

Alur inferensi terlihat pada Gambar VI.2



(a) Pertama : Gagal



(b) Kedua : Sukses

Gambar VI.2 *Backward Chaining*

Latihan

1. Pahami kembali metode inferensi *forward chaining* dan *backward chaining*
2. Buatlah contoh merancang menggunakan metode inferensi *forward chaining* dan *backward chaining*

BAB VII

METODE INFERENSI *FUZZY*

Pokok Bahasan
VII.1 Pendahuluan
VII.2 Alasan Digunakannya Logika *Fuzzy*
VII.3 Aplikasi
VII.4 Himpunan *Fuzzy*
VII.5 Fungsi Keanggotaan

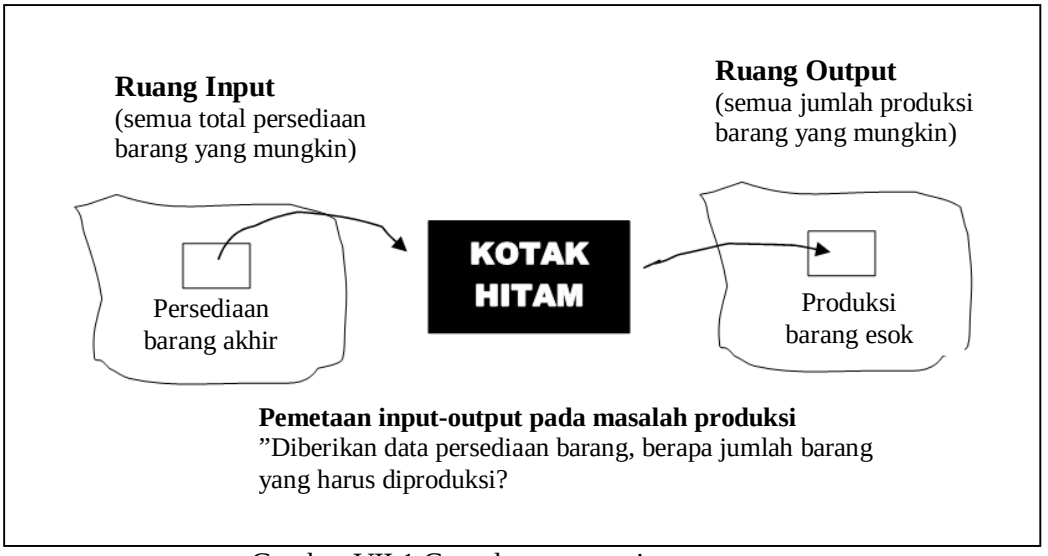
Setelah mempelajari pada bab ini diharapkan dapat memahami logika *fuzzy*, alasan digunakannya logika *fuzzy*, aplikasi menggunakan logika *fuzzy*,himpunan *fuzzy*, fungsi keanggotaan.

VII.1 Pendahuluan

Orang yang belum pernah mengenal logika fuzzy pasti akan mengira bahwa logika fuzzy adalah sesuatu yang amat rumit dan tidak menyenangkan. Namun, sekali seseorang mulai mengenalnya, ia pasti akan sangat tertarik dan akan menjadi pendatang baru untuk ikut serta mempelajari logika fuzzy. Logika fuzzy dikatakan sebagai logika baru yang lama, sebab ilmu tentang logika fuzzy modern dan metodis baru ditemukan beberapa tahun yang lalu, padahal sebenarnya konsep tentang logika fuzzy itu sendiri sudah ada pada diri kita sejak lama.
Logika fuzzy adalah suatu cara yang tepat untuk memetakan suatu ruang input ke dalam suatu ruang output. Sebagai contoh :

1. Manajer pergudangan mengatakan pada manajer produksi seberapa banyak persediaan barang pada akhir minggu ini, kemudian manajer produksi akan menetapkan jumlah barang yang harus diproduksi esok hari.
2. Pelayan restoran memberikan pelayanan terhadap tamu, kemudian tamu akan memberikan tip yang sesuai atas baik tindakan pelayan yang diberikan
3. Anda mengatakan pada saya seberapa sejuk ruangan yang anda inginkan, saya akan mengatur putaran kipas yang ada pada ruangan ini.
4. Penumpang taksi berkata pada sopir taksi seberapa cepat laju kendaraan yang diinginkan, sopir taksi akan mengatur pijakan gas taksinya.

Salah satu contoh pemetaan suatu input-output dalam bentuk grafis seperti terlihat pada Gambar VII.1



Gambar VII.1 Contoh pemetaan input-output

Antara input dan output terdapat satu kotak hitam yang harus memetakan input ke output yang sesuai.

VII.2 Alasan Digunakannya Logika *Fuzzy*

Ada beberapa alasan mengapa orang menggunakan logika fuzzy, antara lain :

1. Konsep logika fuzzy mudah dimengerti. Konsep matematis yang mendasari penalaran fuzzy sangat sederhana dan mudah dimengerti.
2. Logika fuzzy sangat fleksibel.
3. Logika fuzzy memiliki toleransi terhadap data-data yang tidak tepat.
4. Logika fuzzy mampu memodelkan fungsi-fungsi nonlinear yang sangat kompleks.
5. Logika fuzzy dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.
6. Logika fuzzy dapat bekerjasama dengan teknik-teknik kendali secara konvensional
7. Logika fuzzy didasarkan pada bahasa alami

VII.3 Aplikasi

Beberapa aplikasi logika fuzzy, antara lain :

1. Pada tahun 1990 pertama kali dibuat mesin cuci dengan logika fuzzy di Jepang (Matsushita Electric Industrial Company). Sistem fuzzy digunakan untuk menentukan putaran yang tepat secara otomatis berdasarkan jenis dan banyaknya kotoran serta jumlah yang akan dicuci. Input yang digunakan adalah : seberapa kotor, jenis kotoran, dan banyaknya yang dicuci. Mesin ini menggunakan sensor optik, mengeluarkan cahaya ke air dan mengukur bagaimana cahaya tersebut sampai ke ujung lainnya. Makin kotor, maka sinar yang sampai makin redup. Disamping itu, sistem juga dapat menentukan jenis kotoran (daki atau minyak).
2. Transmisi otomatis pada mobil. Mobil Nissan telah menggunakan sistem fuzzy pada transmisi otomatis, dan mampu menghemat bensin 12-17%.
3. Kereta bawah tanah Sendai mengontrol pemberhentian otomatis pada area tertentu.
4. Ilmu kedokteran dan biologi, seperti sistem diagnosis yang didasarkan pada logika fuzzy, penelitian kanker, manipulasi peralatan prostetik yang didasarkan logika fuzzy, dll.
5. Manajemen dan pengambilan keputusan, seperti manajemen basisdata yang didasarkan pada logika fuzzy, tata letak pabrik yang didasarkan pada logika fuzzy, sistem pembuat keputusan di militer yang didasarkan pada logika fuzzy, pembuatan games yang didasarkan pada logika fuzzy, dll
6. Ekonomi, seperti pemodelan fuzzy pada sistem pemasaran yang kompleks, dll.
7. Klasifikasi dan pencocokan pola
8. Psikologi, seperti logika fuzzy untuk menganalisis kelakuan masyarakat, pencegahan dan investigasi kriminal, dll
9. Ilmu-ilmu sosial, terutama untuk pemodelan informasi yang tidak pasti.
10. Ilmu lingkungan, seperti kendali kualitas air, prediksi cuaca, dll.
11. Teknik, seperti perancangan jaringan komputer, prediksi adanya gempa bumi, dll.
12. Riset operasi, seperti penjadwalan dan pemodelan, pengalokasian, dll.
13. Peningkatan kepercayaan, seperti kegagalan diagnosis, inspeksi dan monitoring produksi.

VII.4 Himpunan *Fuzzy*

Pada himpunan tegas *(crisp)*, nilai keanggotaan suatu item x dalam suatu himpunan A, yang sering ditulis dengan $\mu_A[x]$, memiliki 2 kemungkinan,

yaitu :

- satu (1), yang berarti bahwa suatu item menjadi anggota dalam suatu himpunan, atau
- nol (0), yang berarti bahwa suatu item tidak menjadi anggota dalam suatu himpunan.
-

Contoh VII.1 :
Jika diketahui :

S = {1,2,3,4,5,6} adalah semesta pembicaraan.
A = {1,2,3}
B = {3,4,5}

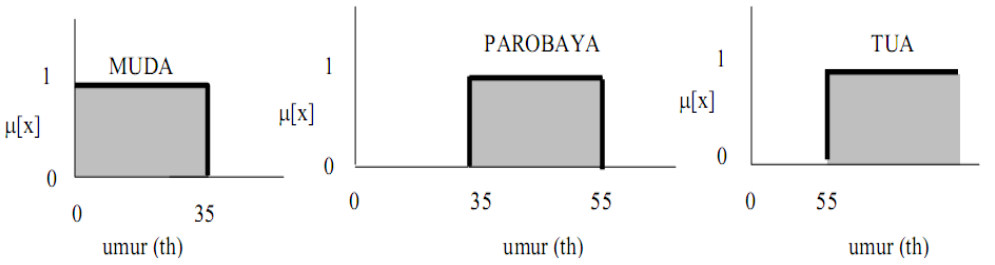
bisa dikatakan bahwa :

– Nilai keanggotaan 2 pada himpunan A, $\mu_A[2]=1$, karena $2 \in A$.
– Nilai keanggotaan 3 pada himpunan A, $\mu_A[3]=1$, karena $3 \in A$
– Nilai keanggotaan 4 pada himpunan A, $\mu_A[4]=1$, karena $4 \notin A$.
– Nilai keanggotaan 2 pada himpunan B, $\mu_B[2]=1$, karena $2 \notin B$
– Nilai keanggotaan 3 pada himpunan B, $\mu_B[3]=1$, karena $3 \in B$.

Contoh VII.2 :
Misalkan variabel umur dibagi menjadi 3 kategori, yaitu :

MUDA umur < 35 tahun
PAROBAYA 35 ≤ umur ≤ 55 tahun
TUA umur > 55 tahun

Nilai keanggotaan secara grafis, himpunan MUDA, PAROBAYA dan TUA ini dapat dilihat pada Gambar VII.2



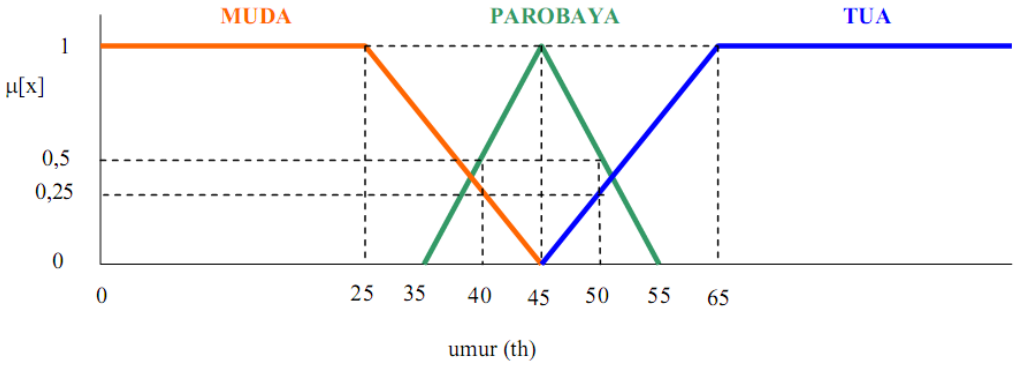
Gambar VII.2 Himpunan : MUDA, PAROBAYA, dan TUA

Gambar VII.2, dapat dilihat bahwa :

– apabila seseorang berusia 34 tahun, maka ia dikatakan MUDA ($\mu_{MUDA}[34]=1$);
– apabila seseorang berusia 35 tahun, maka ia dikatakan TIDAK MUDA ($\mu_{MUDA}[35]=0$);

- apabila seseorang berusia 35 tahun kurang 1 hari, maka ia dikatakan TIDAK MUDA (μ_{MUDA}[35 th – 1 hr]=0);
- apabila seseorang berusia 35 tahun, maka ia dikatakan PAROBAYA ($\mu_{PAROBAYA}$[35 th]=1);
- apabila seseorang berusia 34 tahun, maka ia dikatakan TIDAK PAROBAYA ($\mu_{PAROBAYA}$[34 th]=0);
- apabila seseorang berusia 35 tahun, maka ia dikatakan PAROBAYA ($\mu_{PAROBAYA}$[35]=1);
- apabila seseorang berusia 35 tahun kurang 1 hari, maka ia dikatakan TIDAK PAROBAYA ($\mu_{PAROBAYA}$[35 th – 1 hr]=0);

Dari sini bisa dikatakan bahwa pemakaian himpunan *crisp* untuk menyatakan umur sangat tidak adil, adanya perubahan kecil saja pada suatu nilai mengakibatkan perbedaan kategori yang cukup signifikan.
Himpunan fuzzy digunakan untuk mengantisipasi hal tersebut. Seseorang dapat masuk dalam 2 himpunan yang berbeda, MUDA dan PAROBAYA, PAROBAYA dan TUA, dsb. Seberapa besar eksistensinya dalam himpunan tersebut dapat dilihat pada nilai keanggotaannya. Gambar VII.3 menunjukkan himpunan fuzzy untuk variabel umur.



Gambar VII.3 Himpunan *fuzzy* untuk variabel Umur.

Pada Gambar VII.3, dapat dilihat bahwa :

- Seseorang yang berumur 40 tahun, termasuk dalam himpunan MUDA dengan (μ_{MUDA}[40]=0,25; namun dia juga termasuk dalam himpunan PAROBAYA dengan ($\mu_{PAROBAYA}$[40]=0,5.
- Seseorang yang berumur 50 tahun, termasuk dalam himpunan TUA dengan (μ_{TUA}[50]=0,25; namun dia juga termasuk dalam himpunan PAROBAYA dengan ($\mu_{PAROBAYA}$[50]=0,5.

Kalau pada himpunan *crisp*, nilai keanggotaan hanya ada 2 kemungkinan, yaitu 0 atau 1, pada himpunan fuzzy nilai keanggotaan terletak pada rentang 0 sampai 1. Apabila x memiliki nilai keanggotaan fuzzy μ_A[x]=0 berarti x

tidak menjadi anggota himpunan A, demikian pula apabila x memiliki nilai keanggotaan fuzzy $\mu_A[x]=1$ berarti x menjadi anggota penuh pada himpunan A.

Terkadang kemiripan antara keanggotaan fuzzy dengan probabilitas menimbulkan kerancuan. Keduanya memiliki nilai pada interval [0,1], namun interpretasi nilainya sangat berbeda antara kedua kasus tersebut. Keanggotaan fuzzy memberikan suatu ukuran terhadap pendapat atau keputusan, sedangkan probabilitas mengindikasikan proporsi terhadap keseringan suatu hasil bernilai benar dalam jangka panjang. Misalnya, jika nilai keanggotaan suatu himpunan fuzzy MUDA adalah 0,9;maka tidak perlu dipermasalahkan berapa seringnya nilai itu diulang secara individual untuk mengharapkan suatu hasil yang hampir pasti muda. Di lain pihak, nilai probabilitas 0,9 muda berarti 10 % dari himpunan tersebut diharapkan tidak muda.

VII.5 Fungsi Keanggotaan

Fungsi keanggotaan *(membership function)* adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya (sering juga disebut dengan derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Salah satu cara dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi. Dari gambar VII.3 adalah suatu fungsi keanggotaan untuk variabel UMUR yang dibagi menjadi 3 kategori atau 3 himpunan fuzzy yaitu MUDA, PAROBAYA, TUA, dimana dapat direpresentasikan sebagai berikut :

$$\mu MUDA[x] = \begin{cases} 1, & x \le 25 \\ \dfrac{45-x}{45-25} & 25 < x < 45 \\ 0 & x \ge 45 \end{cases} \qquad \mu TUA[x] = \begin{cases} 0, & x \le 45 \\ \dfrac{x-45}{65-45} & 45 < x < 65 \\ 1 & x \ge 65 \end{cases}$$

$$\mu PAROBAYA[x] = \begin{cases} 0, & x \le 35 \text{ atau } x \ge 55 \\ \dfrac{x-35}{45-35} & 35 < x < 45 \\ \dfrac{55-x}{55-45} & 45 \le x \le 55 \end{cases}$$

a. Aturan Rule If Then Fuzzy

- Aturan IF-THEN fuzzy adalah pernyataan IF-THEN dimana beberapa kata-kata dalam pernyataan tersebut ditentukan oleh fungsi keanggotaan
- Aturan produksi fuzzy adalah relasi fuzzy antara dua proposisi. Aturan tersebut dinyatakan dalam bentuk :
  IF (proposisi fuzzy 1) THEN (proposisi fuzzy 2)

  ↓ ↓

  Disebut anteceden/premis Disebut consequent/kesimpulan
- Proposisi fuzzy adalah memiliki derajat kebenaran yang dinyatakan dalam suatu bilangan dalam bentuk interval [0,1], dimana benar dinyatakan oleh nilai 1 dan salah dinyatakan oleh nilai 0.
- Premis dari aturan fuzzy dapat memiliki lebih dari satu bagian (premis1, premis2,...dst), semua bagian dari premis dihitung secara simultan dan diselesaikan untuk sebuah nilai tunggal dengan menggunakan operator fuzzy dalam himpunan fuzzy.

  IF premis 1 AND premis 2 THEN kesimpulan 1 AND kesimpulan 2

  Dimana : AND adalah operator fuzzy

  Premis 1 dan premis 2 berupa variabel masukan

  Kesimpulan 1 dan kesimpulan 2 berupa variabel-keluaran

Contoh :

IF permintaan turun AND persediaan banyak THEN produksi barang berkurang

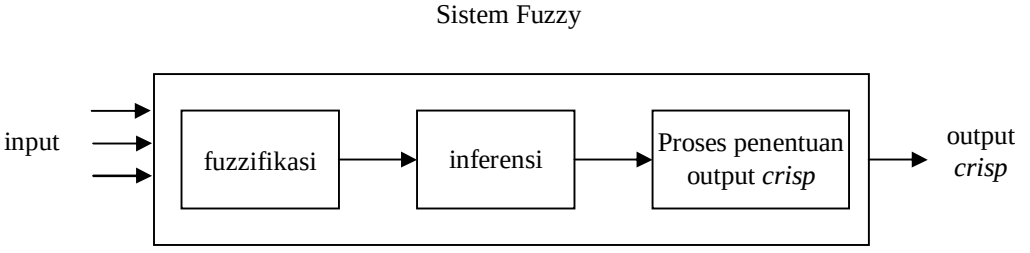
IF permintaan naik AND persediaan sedikit THEN produksi barang bertambah

Dimana :

Permintaan, persediaan	:	variabel masukan
Produksi barang	:	variabel keluaran
Turun, naik	:	kategori himpunan fuzzy dari permintaan
Banyak, sedikit	:	kategori himpunan fuzzy dari persediaan
Berkurang, bertambah	:	kategori himpunan fuzzy dari produksi barang

b. Tahapan Membangun Sistem Fuzzy

Tahapan membangun sistem fuzzy tergantung metode yang digunakan, karena banyak teori /metode untuk membangun sistem fuzzy. Namun secara garis besar dapat disimpulkan sebagai berikut :



Fuzzifikasi :

Mengambil masukan nilai *crisp* dan menentukan derajat dimana nilai-nilai tersebut menjadi anggota dari setiap himpunan fuzzy yang sesuai membuat fungsi keanggotaan. →

Contoh : masukan crisp 75 derajat ditransformasikan sebagai panas dalam bentuk fuzzy dengan derajat keanggotaan 0,80.

Inferensi :

- Mengaplikasikan aturan pada masukan fuzzy yang dihasilkan pada proses fuzzyfikasi
- Mengevaluasi tiap aturan dengan masukan yang dihasilkan dari proses fuzzyfikasi dengan mengevaluasi hubungan atau derajat keanggotaan anteceden/premis setiap aturan.
- Derajat keanggotaan /nilai kebenaran dari premis digunakan untuk menentukan nilai kebenaran bagi consequent/kesimpulan.

Proses penentuan Output *Crisp :*
Tergantung teori/metode yang digunakan

Latihan
1. Pahami kembali metode inferensi menggunakan *fuzzy*
2. Buatlah contoh merancang menggunakan metode inferensi menggunakan *fuzzy*

BAB VIII

PENALARAN DENGAN KETIDAKPASTIAN
(UNCERTAINTY)

Pokok Bahasan
VIII.1 Ketidakpastian (Uncertainty)
VIII.2 Kesalahan dan Induksi
VIII.3 Peluang

Setelah mempelajari uraian pada bab ini diharapkan dapat memahami ketidakpastian *(uncertainty)*, kesalahan dan induksi dan peluang.

VIII.1 Ketidakpastian (Uncertainty)

Ketidakpastian dapat dianggap sebagai suatu kekurangan informasi yang memadai untuk membuat suatu keputusan. Ketidakpastian merupakan suatu permasalahan karena mungkin menghalangi kita dalam membuat suatu keputusan yang terbaik bahkan mungkin dapat menghasilkan suatu keputusan yang buruk. Dalam dunia medis, ketidakpastian mungkin menghalangi pemeriksaan yang terbaik untuk para pasien dan berperan untuk suatu terapi yang keliru. Dalam bisnis, ketidakpastian dapat berarti kerugian keuangan.

Sejumlah teori yang berhubungan dengan ketidakpastian telah ditemukan, diantaranya probabilitas klasik, probabilitas Bayes, teori Hartley yang berdasarkan pada himpunan klasik, teori Shanon yang didasarkan pada peluang, teori Dempster-Shafer dan teori fuzzy Zadeh.

Semua makhluk hidup adalah ahli pada urusan yang berhubungan dengan ketidakpastian atau mereka tidak akan dapat bertahan hidup dalam semesta ini. Secara khusus manusia telah menggunakan ketidakpastian tentang lalu lintas, cuaca, pekerjaan, sekolah, dan lain sebagainya. Setelahnya kita semua akan menjadi ahli mengemudi dalam berbagai macam kondisi lalulintas, harus melakukan apa jika dalam cuaca dingin, mudah mengambil matakuliah dan masih banyak hal lainnya. Berhubungan dengan ketidakpastian maka dibutuhkan penalaran atas ketidakpastian itu.

Ada banyak aplikasi sistem pakar yang dapat dikerjakan dengan penalaran pasti. Banyak juga aplikasi lain yang membutuhkan penalaran tidak pasti, yang melibatkan fakta yang tidak pasti, kaidah atau kedu-duanya. Contoh-contoh klasik sistem pakar yang sukses yang berhubungan dengan ketidakpastian adalah MYCIN yang berguna untuk diagnosa medis dan PROSPECTOR untuk eksplorasi mineral.

Dalam sistem MYCIN dan PROSPECTOR, konklusi dicapai bila semua fakta untuk meyakinkan membuktikan kesimpulan tidak diketahui. Walaupun hal ini kemungkinan untuk mencapainya pada konklusi yang lebih dapat dipercaya dengan melakukan banyak pengujian. Ada masalah dengan penambahan waktu dan biaya pelaksanaan pengujian Batasan waktu dan uang penting sekali dalam kasus pengobatan medis. Penundaan pengobatan untuk pengujian mempertimbangkan penambahan biaya. Dalam pada itu ada kemungkinan pasien akan meninggal. Dalam kasus eksplorasi mineral, biaya dari penambahan pengujian juga merupakan faktor yang sangat signifikan.

Banyak kemungkinan dan ketidakpastian menyertai dalam masalah dan solusinya. Ada beberapa sumber dari ketidakpastian, beberapa diantaranya adalah :

a. Masalah

   Beberapa masalah meliputi faktor-faktor yang oleh sifat mereka, tidak pasti atau acak. Sebagai contoh, dalam pengobatan, penyakit yang sama dapat memberi gejala yang berbeda untuk pasien yang lain. Mungkin saja untuk mendeteksi pola-pola dalam masalah-masalah ini tetapi tidak semuanya dapat diperkirakan seorang pakar. Dengan melihat pola-pola ini dapat dipakai aturan-aturan yang masing-masing biasanya sesuai.

b. Data

   Beberapa masalah mungkin memiliki batasan yang kurang jelas bagi seseorang. Orang yang menghadirkan masalah mungkin mengetahui beberapa fakta untuk kepastian, menuduh lainnya dan tidak mengetahui lainnya. Angka-angka dan nilai-nilai dapat tidak tepat, ditebak atau tidak diketahui. Mungkin akan menjadi sangat mahal dan berbahaya atau hampir tidak mungkin untuk menyediakan semua informasi secara lengkap. Pakar harus mampu membuat keputusan yang memadai diatas basis pengetahuan seperti itu.

c. Pakar

   Manusia sering dapat memakai pengetahuan mereka tanpa mengetahui secara eksplisit apa pengetahuan itu sendiri. Mereka mungkin harus meningkatkan secara detail apa yang mereka lakukan dan bagaimana dan tampak tak jelas atau bahkan bertentangan dengan dirinya sendiri. Teknik-teknik perolehan pengetahuan dirancang untuk mengatasi ketidakpastian yang disebabkan oleh hal ini.

d. Solusi

   Ada beberapa area tertentu dimana tidak terdapat pakar yang diakui. Pakar sendiri mungkin tidak setuju satu sama lain dan tak seorangpun dapat memutuskan solusi yang baik. Domain seperti itu dapat berupa strategi militer.

   Ketidakpastian berhubungan dengan sesuatu yang tidak diyakini dan ketakpersisan berhubungan dengan sesuatu yang nilainya tidak diketahui

secara akurat. Contoh berikut mengungkapkan bahwa ketakpastian dan ketakpersisan dapat muncul secara bebas satu sama lain.

VIII.2 Kesalahan dan Induksi

Proses induksi merupakan lawan dari deduksi. Deduksi merupakan hasil dari hal yang umum ke khusus, seperti :

Semua laki-laki adalah makhluk hidup.

Socrates adalah laki-laki

Dapat ditarik kesimpulan :

Socrates adalah makhluk hidup.

Induksi mencoba untuk menggeneralisasikan dari hal khusus ke umum, seperti :

Disk saya belum pernah rusak

∵ Disk saya tidak akan pernah rusak

Dimana simbol ∵ mewakili ”oleh karena” untuk induksi dan ∴ mewakili ”oleh karena” untuk deduksi.

Kecuali untuk induksi matematika, argumen induksi tidak pernah dapat dibuktikan dengan benar. Bahkan argumen induksi hanya dapat menyediakan beberapa tingkat kepercayaan bahwa konklusi tersebut adalah benar. Kita belum mempunyai kepercayaan yang lebih pada argumen induktif terdahulu. Berikut ini adalah contoh argumen yang lebih kuat :

Alarm kebakaran berbunyi

∵ ada kebakaran

Argumen yang lebih kuat lainnya adalah :

alarm kebakaran berbunyi

saya mencium bau asap

∵ ada kebakaran

Walaupun kalimat ini adalah argumen kuat, hal ini tidaklah membuktikan ada kebakaran. Asap dapat berasal dari masakan humberger diatas panggangan dan alarm kebakaran mungkin telah dipasang secara kebetulan.Bukti suatu kebakaran adalah sebagai berikut :

alarm kebakaran berbunyi

saya mencium bau asap

pakaian saya terbakar

∴ ada kebakaran.

Argumen diatas adalah argumen deduktif karena dari argumen tersebut jelas adanya pernyataan yang menyatakan adanya api.

Sistem pakar boleh jadi terdiri dari kaidah deduktif dan induktif, dimana kaidah induktif merupakan suatu heuristik alami. Induksi juga dapat digunakan untuk pembangkitan kaidah secara otomatis.

VIII.3 Peluang

Suatu sudah lama sekali tetapi masih tetap sangat penting sebagai alat dalam penyelesaian masalah AI adalah probabilitas (Farley: 1983). Probabilitas merupakan suatu cara kuantitatif yang berhubungan dengan ketidakpastian yang telah ada sejak abad 17, ketika penjudi-penjudi Perancis meminta bantuan dari para ahli matematika yang terkemuka seperti Pascal, Fermat dan lainnya. Perjudian telah menjadi sangat populer karena melibatkan uang yang sangat besar di dalamnya. Penjudi menginginkan metode untuk membantu mereka dalam menghitung semua rintangan untuk menuju kemenangan.

Teori probabilitas klasik pertama kali diperkenalkan oleh Pascal dan Fermat pada tahun 1654 (Parrat:1961). Kemudian banyak kerja yang telah dilakukan untuk mengerjakan probabilitas dan ada beberapa cabang baru dari probabilitas yang dikembangkan. Banyak aplikasi probabilitas telah ditunjukkan dalam ilmu pengetahuan teknik, bisnis, ekonomi, dan bidang-bidang lainnya.

Teori Probabilitas

Ada tiga aksioma dari teori probabilitas ini, yaitu :

Jika probabilitas suatu kejadian dinyatakan oleh P(A) dan semesta pembicaraannya adalah S maka :

1. $0 \leq P(A) \leq 1$

   Aksioma ini menjelaskan bahwa jangkauan dari probabilitas itu berada anatar 0 dan 1. Tidak ada probabilitas yang bernilai negatif. Jika suatu kejadian itu pasti terjadi maka nilai probabilitasnya adalah 1 dan jika kejadiannya tidak mungkin terjadi maka probabilitasnya adalah 0.
2. $\sum_{i} P(A_i)=1$

   Aksioma ini menyatakan bahwa jumlah semua kejaadian dalam suatu semesta pembicaraan S adalah 1 atau P(S)=1. Dan sebagai corollary dari aksioma ini adalah :

   $P(A)+P(A')=1$
3. $P(A_1 \cup A_{2)=} P(A_1)+ P(A_2)$

   Dimana A_1, A_2 adalah kejadian yang *mutually exclusive.* Aksioma ini mempunyai makna bahwa jika A_1, A_2 keduanya tidak dapat terjadi secara simultan maka probabilitas dari satu atau kejadian lainnya adalah jumlah dari masing-masing probabilitasnya.

Latihan

1. Pahami kembali penalaran dengan ketidakpastian *(Uncertainty)*
2. Pahami teori peluang (Probabilitas)

BAB IX

PELUANG DAN TEOREMA BAYES

Pokok Bahasan
Probabilitas Teorema Bayes

Tujuan mempelajari pada bab ini diharapkan dapat memahami probabilitas teorema bayes, penerapan rumus teorema bayes.

Probabilitas bayes merupakan salah satu cara untuk mengatasi ketidakpastian data dengan menggunakan formula Bayes yang dinyatakan :

$$P(H|E) = \frac{P(E|H).P(H)}{P(E)}$$

Dimana :

$P(H|E)$: probabilitas hipotesis H jika diberi Evidence E

$P(E|H)$: probabilitas munculnya evidence E jika diketahui hipotesis H

$P(H)$: probabilitas hipotesis H tanpa memandang evidence apapun

$P(E)$: probabilitas evidence E

Dalam bidang kedokteran teorema Bayes sudah dikenal tetapi teorema ini lebih banyak diterapkan dalam logika kedokteran modern (Cutler:1991). Teorema ini lebih banyak diterapkan pada hal-hal yang berkenaan dengan diagnosis secara statistik yang berhubungan dengan probabilitas serta kemungkinan dari penyakit dan gejala-gejala yang berkaitan.

Secara umum teorema Bayes dengan E kejadian dan hipotesis H dapat dituliskan dalam bentuk :

$$P(H_i|E) = \frac{P(E \cap H_i)}{\sum_j P(E \cap H_i)}$$

$$= \frac{P(E|H_i)\,P(H_i)}{\sum_j P(E|H_j)\,P(H_j)}$$

$$= \frac{P(E|H_i)\,P(H_i)}{P(E)}$$

Berikut ini adalah contoh penghitungan probabilitas menggunakan probabilitas Bayes yang diambil dari Iswati (2004) dan Kusumadewi (2002) :

1. Seorang dokter mengetahui bahwa penyakit maningtis menyebabkan ”stiff neck” adalah 50 %. Probabilitas pasien menderita maningtis adalah 1/50000 dan probabilitas pasien menderita stiff neck adalah 1/20 dari nilai-nilai tersebut. Didapatkan :
   - P (stiff neck |maningitis) = 50 % = 0.5
     P (maningitis) = 1/50000
     P (stiff neck) = 1/20

     Maka :

$$P(\text{mangitis} \mid \textit{stiffneck}) = \frac{P(\text{stiff neck} \mid \text{maningitis})\, P(\text{maningitis})}{P(\text{stiff neck})}$$

$$= \frac{5/10 \times 1/50000}{1/10} = 0{,}0002$$

$$= \frac{1}{5000}$$

Hasil diatas menunjukkan bahwa hanya 1 diantara 5000 pasien yang mengalami *stiff neck.*

Teorema Bayes dapat dikembangkan jika setelah dilakukan pengujian terhadap hipotesis kemudian muncul lebih dari sebuah evidence. Dalam hal ini maka persamaannya akan menjadi :

$$P(H \mid E,e) = P(H \mid E)\, \frac{P(e \mid E,H)}{P(e \mid E)}$$

Dimana :

e	: evidence lama
E	: evidence baru
P(H |E,e)	: probabilitas hipotesis H benar jika muncul evidence baru E dari evidence lama e.
P(H |E)	: probabilitas hipotesis H benar jika diberikan evidence E.
P(e | E,H)	: kaitan antar e dan E jika hipotesis H benar
P(e | E)	: kaitan antara e dan E tanpa memandang hipotesis apapun.

2. Si Ani mengalami gejala ada bintik-bintik diwajahnya. Dokter menduga bahwa si Ani terkena cacar dengan :

   - Probabilitas munculnya bintik-bintik diwajah, jika si Ani terkena cacar; P(bintik-bintik |cacar)= 0,8
     - Probabilitas si Ani terkena cacar tanpa memandang gejala apapun; P(cacar)=0,4
     - Probabilitas munculnya bintik-bintik di wajah jika si Ani alergi; P(bintik-bintik |alergi)= 0,3
     - Probabilitas si Ani terkena alergi tanpa memandang gejala apapun; P(alergi)= 0,7
     - Probabilitas munculnya bintik-bintik di wajah, jika si Ani jerawatan;P(bintik-bintik | jerawatan)= 0,5

   Maka :

   - Probabilitas si Ani terkena cacar karena ada bintik-bintik diwajahnya adalah

     P(cacar | bintik-bintik)=

$$\frac{P(\text{bintik-bintik} \mid \text{cacar}) * P(\text{cacar})}{P(\text{bintik-bintik} \mid \text{cacar}) * P(\text{cacar}) + P(\text{bintik-bintik} \mid \text{alergi}) * P(\text{alergi}) + P(\text{bintik-bintik} \mid \text{jerawatan}) * P(\text{jerawatan})}$$

$$P(\text{cacar} \mid \text{bintik-bintik}) = \frac{(0,8)(0,4)}{(0,8)(0,4)+ (0,3)(0,7)+ (0,9)(0,5)} = \frac{0,32}{0,98} = 0,327$$

   - Probabilitas si Ani terkena alergi karena ada bintik-bintik di wajahnya adalah :

     P(alergi | bintik-bintik)=

$$\frac{P(\text{bintik-bintik} \mid \text{alergi}) * P(\text{alergi})}{P(\text{bintik-bintik} \mid \text{cacar}) * P(\text{cacar}) + P(\text{bintik-bintik} \mid \text{alergi}) * P(\text{alergi}) + P(\text{bintik-bintik} \mid \text{jerawatan}) * P(\text{jerawatan})}$$

$$P(\text{alergi} \mid \text{bintik-bintik}) = \frac{(0,3)(0,7)}{(0,8)(0,4)+ (0,3)(0,7)+ (0,9)(0,5)} = \frac{0,21}{0,98} = 0,214$$

- Probabilitas si Ani terkena jerawatan karena ada bintik-bintik di wajahnya adalah :
  P(jerawatan | bintik-bintik)=

$$\frac{P(\text{bintik-bintik} \mid \text{jerawatan})*P(\text{jerawatan})}{P(\text{bintik-bintik} \mid \text{cacar})*P(\text{cacar}) + P(\text{bintik-bintik} \mid \text{alergi})*P(\text{alergi}) + P(\text{bintik-bintik} \mid \text{jerawatan})*P(\text{jerawatan})}$$

$$P(\text{jerawatan} \mid \text{bintik-bintik}) = \frac{(0,9)(0,5)}{(0,8)(0,4)+ (0,3)(0,7)+ (0,9)(0,5)} = \frac{0,45}{0,98} = 0,459$$

3. Si Ani mengalami gejala bintik-bintik diwajahnya. Dokter menduga bahwa si Ani terkena cacar dengan probabilitas terkena cacar apabila ada bintik-bintik diwajah, P(cacar |bintik-bintik)=0,8. Ada observasi bahwa orang terkena cacar pasti mengalami panas badan. Jika diketahui probabilitas orang terkena cacar apabila ada bintik-bintik diwajah, P(cacar |panas)=0,5; keterkaitan antara adanya bintik-bintik di wajah dan panas badan apabila seseorang terkena cacar, P(bintik-bintik |panas,cacar)=0,4; sedangkan keterkaitan antara adanya bintik-bintik di wajah dan panas badan, P(bintik-bintik |panas)=0,6., maka :

$$P(\text{cacar} \mid \text{panas,bintik-bintik}) = P(\text{cacar, panas}) \times \frac{P(\text{bintik-bintik} \mid \text{panas,cacar})}{P(\text{bintik} \mid \text{panas})}$$

$$P(\text{cacar} \mid \text{panas,bintik-bintik}) = 0,5 \times \frac{0,4}{0,6} = 0,33$$

Latihan

1. Pahami kembali peluang dan teorema bayes
2. Buatlah contoh rancangan dengan menggunakan teorema bayes

BAB X

PENALARAN INEXACT

Pokok Bahasan
X.1 Ketidakpastian dan Kaidah
X.2 Faktor Kepastian (Certainty Factor)

Setelah mempelajari pada bab ini diharapkan dapat memahami ketidakpastian dan kaidah, faktor kepastian. dan dapat menerapkan metode *certainty factor* pada kasus.

X.1 Ketidakpastian dan Kaidah

Salah satu karakteristik umum dari suatu informasi yang tersedia untuk seorang ahli ketidak sempurnaan atau kecacatan. Informasi bisa jadi tidak lengkap, tidak konsisten, tidak tentu dan sebagainya. Dengan kata lain informasi sering tidak sesuai untuk meyelesaikan suatu permasalahan, akan tetapi seorang pakar dapat mengatasi kerusakan dan biasanya dapat membuat suatu pertimbangan benar dan keputusan yang benar. Sistem pakar juga harus mampu untuk mengatasi ketidakpastian dan menggambarkan suatu konklusi yang valid.

Ketidakpastian dalam sistem berbasis kaidah dapat berasal dari 3 hal berikut, yaitu :

1. Kaidah Tunggal *(individual rule)*
   Kaidah tunggal dipengaruhi oleh 3 hal, yaitu kesalahan (error), probabilitas dan kombinasi premis.
   Kesalahan disebabkan antara lain oleh :
   a. Ambiguitas, yaitu sesuatu yang didefinisikan berlebihan
   b. Ketidaklengkapan data
   c. Kesalahan informasi
   d. Kesalahan pengukuran

   Probabilitas disebabkan oleh ketidakmampuan seorang pakar untuk merumuskan kaidah secara pasti. Dalam bidang kedokteran dapat diambil contoh pada saat dokter menentukan diagnosis, maka dia harus mempertimbangkan bahwa jika seorang penderita datang dengan gejala-gejala yang ada, yaitu demam, sakit kepala dan bersin-bersin, maka ada kemungkinan si pasien terserang influensa. Akan tetapi tidak selalu pasien yang datang dengan gejala tersebut pasti menderita influensa.

   Kombinasi premis di dalam anteseden jika premis lebih dari sebuah perlu diperhatikan. Beberapa kombinasi yang dapat dibentuk, yaitu :

E_1 AND E_2 AND E_3

atau E_1 AND E_2 OR E_3

atau E_1 AND NOT E_2 OR E_3

atau beberapa kemungkinan kombinasi lainnya yang menggunakan penghubung AND, OR dan NOT.

2. Ketidaksesuaian antarkaidah *(incompatibility of rules)*

   Ketidak sesuaian antar kaidah dapat disebabkan oleh beberapa hal yaitu :

   a. Kontradiksi kaidah *(contrdiction rules),*
   b. Subsumsi kaidah *(subsumption),*
   c. Redudansi kaidah *(redudancy rules),*
   d. Kehilangan kaidah *(missing rules),*
   e. Penggabungan data *(data fussion).*

   Contoh kontradiksi aturan adalah :

   Asumsikan dua buah kaidah dalam basis pengetahuan :

   Kaidah 1 IF terdapat api THEN siramlah dengan air

   Kaidah 2 IF terdapat api THEN jangan siram dengan air

   Kaidah 1 adalah jika benar-benar terdapat api seperti terbakarnya kayu, maka akan dilakukan pemadaman dengan menyiramkan dengan air, sedangkan untuk kaidah 2 bahwa terdapat api memang disengaja untuk melakukan proses pembakaran yang tidak boleh disiram dengan air. Dari kedua kaidah tersebut terlihat ketidaksesuaian konsekuen (kontradiksi) yang disebabkan oleh anteseden yang kurang spesifik.

   Subsumsi kaidah yang dimaksud adalah jika anteseden merupakan bagian dari kaidah yang lain. Contoh subsumsi tersebut adalah :

   Kaidah 1 IF E_1 THEN H

   Kaidah 2 IF E_1 and E_2 THEN H

   Jika E_1 muncul maka tidak terdapat masalah karena kaidah 1 yang akan dijalankan, tetapi jika E_1 dan E_2 kedua-duanya akan sama-sama dijalankan sehingga konflik resolusi dibutuhkan.

   Redudansi aturan adalah kaidah-kaidah yang mempunyai konsekuen dan evidence yang sama, misalnya :

   Kaidah 1 IF E_1 and E_2 THEN H

   Kaidah 2 IF E_2 and E_1 THEN H

   Kedua kaidah nampak berbeda tetapi sebenarnya adalah sama.

   Kehilangan aturan merupakan penyebab ketidaksesuaian antarkaidah yang terjadi jika seorang ahli lupa atau tidak sadar akan membuat kaidah seperti :

   IF E_4 THEN H

   Jika E_4 diabaikan maka H tidak akan pernah dapat disimpulkan selayaknya.

3. Resolusi konflik (Conflict Resolution)

   Sumber ketidakpastian lainnya adalah resolusi konflik. Resolusi konflik

merupakan proses menyeleksi atau memilih kaidah yang ada jika terdapat lebih dari satu kaidah yang diaktivasi dan resolusi konflik disebabkan oleh interaksi antarkaidah. Ada beberapa metode untuk resolusi konflik, antara lain :

a. Memicu kaidah berdasarkan prioritas. Kaidah dirangking menurut urutan tertentu dan memicu kaidah-kaidah tersebut sesuai dengan urutannya. Kaidah dengan urutan teratas mempunyai prioritas tertinggi. Bila suatu konklusi telah ditemukan saat memicu kaidah tertentu maka pemicu dihentkan.
b. Mempunyai kaidah yang mempunyai banyak premis yang harus dipenuhi. Metode ini dikenal dengan the longest matching strategies.
c. Memilih kaidah yang paling banyak digunakan.
d. Memilih kaidah yang paling akhir ditambahkan pada sekumpulan kaidah.
e. Memilih yang waktu eksekusinya paling singkat
f. Memilih semua kaidah dari sekumpulan kaidah yang ada

Dari metode-metode untuk resolusi konflik diatas dapat disimpulkan bahwa proses pemilihan kaidah ada yang berdasar.

X.2 Faktor Kepastian *(Certainty Factor)*

Certaainty Factor (CF) menunjukkan ukuran kepastian terhadap suatu fakta atau aturan. Notasi Faktor Kepastian :

$$CF[h,e] = MB[h,e]-MD[h,e]$$

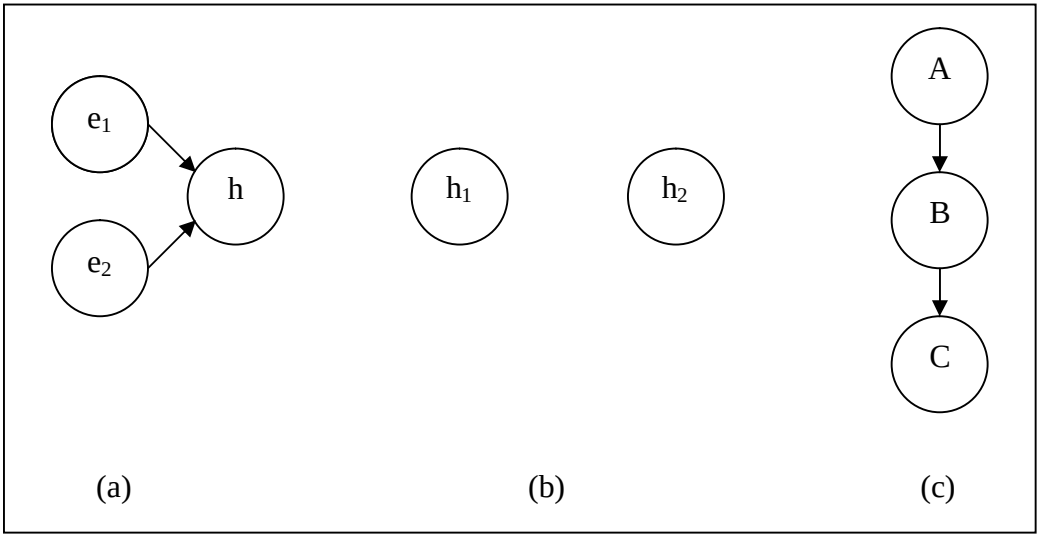
dengan :

CF[h,e] = faktor kepastian

MB[h,e] = ukuran kepercayaan terhadap hipotesis h, jika diberikan evidence e (antara 0 dan 1).

MD[h,e] = ukuran ketidakpercayaan terhadap hipotesis h, jika diberikan evidence e (antara 0 dan 1).

Ada 3 hal yang mungkin terjadi :



Gambar X.1 Kombinasi aturan Ketidakpastian

1. Beberapa evidence dikombinasikan untuk menentukan CF dari suatu hipotesis (Gambar X.1.a). Jika e_1 dan e_2 adalah observasi, maka :

$$MB[h, e_1 \wedge e_2] = \begin{cases} 0 & MD[h, e_1 \wedge e_2]=1 \\ MB[h, e_1]+ MB[h, e_2].(1- MB[h, e_1]) & \text{lainnya} \end{cases}$$

$$MD[h, e_1 \wedge e_2] = \begin{cases} 0 & MB[h, e_1 \wedge e_2]=1 \\ MD[h, e_1]+ MD[h, e_2].(1- MD[h, e_1]) & \text{ainnya} \end{cases}$$

Contoh X.1 :

Andaikan suatu observasi memberikan kepercayaan terhadap h dengan MB[h, e_1]=0,3 dan MD[h, e_1] =0.

sehingga CF[h,e_1] =0,3-0 = 0,3

Jika ada observasi baru dengan MB[h, e_2] =0,2 dan MD[h, e_2]=0, maka :

MB[h, $e_1 \wedge e_2$] = 0,3+0,2*(1-0,3) = 0,44

MD[h, $e_1 \wedge e_2$] = 0

CF[h, $e_1 \wedge e_2$] = 0,44-0 = 0,44

Contoh X.2 :

Si Ani menderita bintik-bintik di wajahnya. Dokter memperkirakan Si Ani terkena cacar dengan kepercayaan, MB[cacar, bintik-bintik] = 0,8 dan MD[cacar, bintik-bintik]= 0,01. Maka :

CF[cacar, bintik-bintik] = 0,8 – 0,01 = 0,79

Jika observasi tersebut juga memberikan kepercayaan bahwa Si Ani mungkin juga terkena alergi dengan kepercayaan, MB[alergi, bintik-bintik] = 0,4 dan MD[alergi, bintik-bintik] = 0,3; maka :

CF[alergi, bintik-bintik] = 0,4-0,3 = 0,1.

Untuk mencari CF[cacar $\wedge$ alergi,bintik-bintik] dapat diperoleh dari :

MB[cacar$\wedge$alergi,bintik-bintik] = min(0,8;0,4) = 0,4

MD[cacar$\wedge$alergi,bintik-bintik] = min(0,01;0,3) = 0,01

CF[cacar$\wedge$alergi,bintik-bintik] = 0,4 – 0,01 = 0,39

Untuk mencari CF[cacar$\vee$alergi,bintik-bintik] dapat diperoleh dari :

MB[cacar$\vee$alergi,bintik-bintik] = max(0,8;0,4) = 0,8

MD[cacar$\vee$alergi,bintik-bintik] = max(0,01;0,3) = 0,3

CF[cacar$\vee$alergi,bintik-bintik] = 0,8 – 0,3 = 0,5

Dari contoh X.2 ini dapat dilihat bahwa, semula faktor kepercayaan bahwa si Ani terkena cacar dari gejala munculnya bintik-bintik di wajah adalah 0,79. Demikian pula faktor kepercayaan bahwa si Ani terkena alergi dari gejala munculnya bintik-bintik di wajah adalah 0,1. Dengan adanya gejala yang sama mempengaruhi 2 hipotesis yang berbeda ini, memberikan faktor kepercayaan bahwa :

- Si Ani menderita cacar dan alergi = 0,39.
- Si Ani menderita cacar atau alergi =0,5

Contoh X.3 :

a. Pada pertengahan tahun 2002, ada indikasi bahwa turunnya devisa Indonesia disebabkan oleh permasalahan TKI di Malaysia. Apabila diketahui : MB[DevisaTurun,TKI] = 0,8 dan MD[DevisaTurun,TKI]=0,3; maka carilah berapa CF[DevisaTurun,TKI]?
   Jawab :
   CF[DevisaTurun,TKI] = MB[DevisaTurun,TKI] – MD[DevisaTurun,TKI]
   = 0,8 - 0,3 = 0,5

b. Ternyata pada akhir September 2002, kemarau yang berkepanjangan mengakibatkan gagal panen yang cukup serius, hal ini ternyata juga berdampak pada turunnya ekspor Indonesia. Apabila diketahui : MB[DevisaTurun,EksporTurun]=0,75 dan MD[DevisaTurun,EksporTurun]=0,1; maka carilah berapa CF[DevisaTurun,EksporTurun] dan berapa CF[DevisaTurun,TKI∧EksporTurun]?
   Jawab :
   – CF[DevisaTurun,EksporTurun] =
     = MB[DevisaTurun,EksporTurun] MD[DevisaTurun,EksporTurun]
     = 0,75 – 0,1
     = 0,65

   – MB[DevisaTurun,TKI∧EksporTurun]
     = MB[DevisaTurun,TKI] + MB[DevisaTurunEksporTurun]* (1-MB[DevisaTurun,TKI])
     = 0,8+0,75*(1-0,8)
     = 0,95

     MD[DevisaTurun, TKI∧EksporTurun]

     = MD[DevisaTurun,TKI] + MD[DevisaTurunEksporTurun]* (1-MD[DevisaTurun,TKI])
     = 0,3 + 0,1 * (1-0,3)
     = 0,37

CF[DevisaTurun,TKI∧EksporTurun]

= MB[DevisaTurun,TKI∧EksporTurun]-
MD[DevisaTurun,TKI∧EksporTurun]
= 0,95-0,37
= 0,58

c. Isu terorisme di Indonesia pasca peristiwa Bom Bali pada tanggal 12 Oktober 2002 ternyata juga ikut mempengaruhi turunnya devisa Indonesia sebagai akibat berkurangnya wisatawan asing. Apabila diketahui : MB[DevisaTurun,BomBali] = 0,5 dan MD[DevisaTurun,BomBali]=0,3; maka carilah berapa CF[DevisaTurun,BomBali] dan berapa CF[DevisaTurun,TKI∧BomBali]?

Jawab :

- CF[DevisaTurun,BomBali] =
  = MB[DevisaTurun,BomBali] –
  MD[DevisaTurun,BomBali]
  = 0,5-0,3
  = 0,2
- MB[DevisaTurun,TKI∧EksporTurun∧BomBali]
  = MB[DevisaTurun,TKI∧EksporTurun]+
  MB[DevisaTurunBomBali]*
  (1-MB[DevisaTurun,TKI∧EksporTurun])
  = 0,95 + 0,5 * (1-0,95)
  = 0,975
  MD[DevisaTurun,TKI∧EksporTurun∧BomBali]
  = MD[DevisaTurun,TKI∧EksporTurun]+
  MD[DevisaTurunBomBali]*
  (1-MD[DevisaTurun,TKI∧EksporTurun])
  = 0,37 + 0,3 * (1-0,37)
  = 0,559
  CF[DevisaTurun,TKI∧EksporTurun∧BomBali]
  = MB[DevisaTurun,TKI∧EksporTurun∧BomBali]-
  MD[DevisaTurun,TKI∧EksporTurun∧BomBali]
  = 0,975 – 0,559
  = 0,416

Latihan

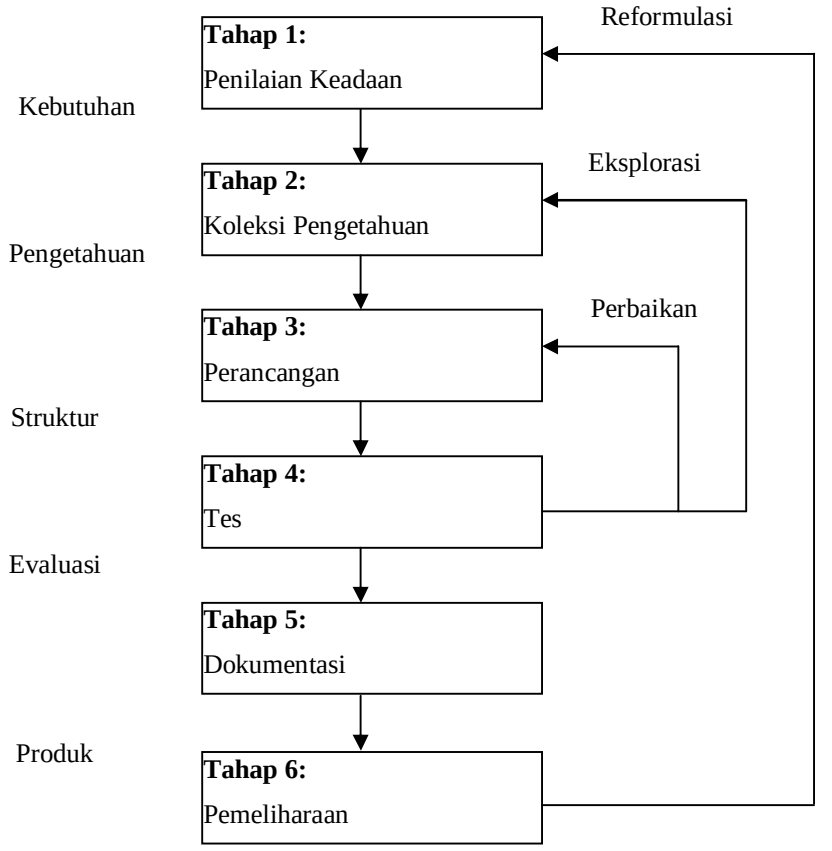
1. Pahami kembali penalaran Inexact
2. Buatlah contoh rancangan dengan menggunakan *certainty factor*

BAB XI

TAHAPAN PENGEMBANGAN SISTEM PAKAR

Pokok Bahasan
Mengembangkan Sistem Pakar

Setelah mempelajari uraian pada bab ini diharapkan dapat memahami bagaimana mengembangkan sistem pakar.
Seperti layaknya pengembangan perangkat lunak, pada pengembangan sistem pakar inipun diperlukan beberapa tahapan seperti terihat pada Gambar XI.1



Gambar XI.1 Tahap-tahap pengembangan sistem pakar

Secara garis besar pengembangan sistem pakar pada Gambar XI.1 adalah sebagai berikut :

1. Mengidentifikasi masalah dan kebutuhan. Mengkaji situasi dan memutuskan dengan pasti tentang masalah yang akan dikomputerisasi dan apakah dengan sistem pakar bisa lebih membantu atai tidak. Misalnya :
   - Penjual komputer mendapat kesulitasn dalam mencocokkan hardware dengan software untuk aplikasi tertentu dan merakit sistem yang lengkap karena mereka tidak punya pengalaman yang luas atau pengetahuan yang cukup tentang produk dan inventory.
   - Perusahaan bagian pemeliharaan penerbangan komersial memerlukan bantuan yang terus-menerus dari beberapa spesialis yang ada. Hal ini memperlambat operasi perbaikan dan pelayanan.
   - Operator proyek tenaga nuklir harus mengawasi belasan indikator dan mencatat status proyek, kondisi, dan keamanan, dan harus segera menerapkan pengetahuannya dengan cepat untuk mencegah terjadinya kerusakan atau segera memperbaikinya. Dalam contoh tersebut ada 2 kondisi, yaitu : (1) user tidak tahu informasi atau tidak pernah berhubungan dengan seseorang yang ahli mengerjakannya; (2) user perlu pengetahuan untuk melakukan pekerjaan itu dengan cepat.
2. Menentukan masalah yang cocok. Ada beberapa syarat yang harus dipenuhi agar sistem pakar dapat bekerja dengan baik, yaitu :
   - Domain masalah tidak terlalu luas;
   - Kompleksitasnya menengah, artinya jika masalah terlalu mudah (dapat diselesaikan dalam beberapa detik saja) atau masalah yang sangat kompleks seperti peramalan inflasi tidak perlu menggunakan sistem pakar;
   - Tersedianya ahli;
   - Menghasilkan solusi mental bukan fisik, artinya sistem pakar hanya memberikan anjuran tidak bisa melakukan aktivitas fisik seperti membau atau merasakan.
   - Tidak melibatkan hal-hal yang bersifat *common sense*, yaitu penalaran yang diperoleh dari pengalaman, seperti : adanya gravitasi membuat benda jatuh, atau jika lampu lalu lintas merah maka kendaraan harus berhenti.
3. Mempertimbangkan alternatif. Dalam hal ini ada 2 alternatif yaitu menggunakan sistem pakar atau komputer tradisional.
4. Menghitung pengembalian investasi. Termasuk diantaranya : biaya pembuatan sistem pakar, biaya pemeliharaan, dan biaya training.
5. Memilih alat pengembangan. Bisa digunakan *software* pembuat sistem pakar (seperti: SHELL) atau dirancang dengan bahasa pemrograman

sendiri (misalnya : dengan menggunakan PROLOG).

6. Rekayasa Pengetahuan. Perlu dilakukan penyempurnaan terhadap aturan-aturan yang sesuai
7. Merancang sistem. Bagian ini termasuk pembuatan *prototype*, serta menterjemahkan pengetahuan menjadi aturan-aturan.
8. Melengkapi pengembangan. Termasuk pengembangan *prototype* apabila sistem yang telah ada sudah sesuai dengan keinginan.
9. Menguji dan mencari kesalahan sistem.
10. Memelihara sistem. Dalam hal ini harus dilakukan: memperbaharui pengetahuan, mengganti pengetahuan yang sudah ketinggalan, dan meluweskan sistem agar bisa lebih baik lagi dalam menyelesaikan masalah.

BAB XII

PERANCANGAN SISTEM PAKAR

Pokok Bahasan
XII.1 Pendahuluan
XII.2 Pemilihan Masalah yang Tepat
XII.3 Jenis Alat Pengembangan

Setelah mempelajari uraian pada bab ini diharapkan dapat memahami dalam pemilihan masalah yang tepat dan jenis alat pengembangan.

XII.1 Pendahuluan

Proses pembangunan suatu sistem pakar dikenal juga sebagai rekayasa pengetahuan *(knowledge engineering.* Pembangunan sistem pakar melibatkan pembinaan pangkalan pengetahuan dengan melibatkan pakar atau sumber yang didokumentasikan. Pengetahuan dalam pembangunan sistem ini, biasanya dibagi atas deklaratif (fakta) dan prosedural. Selain itu, pembangunan suatu sistem pakar melibatkan komponen-komponen dari sistem pakar seperti yang telah disebutkan diatas, yaitu *user interface* (antar muka pengguna), basis pengetahuan, akuisisi pengetahuan, mesin inferensi, workplace, fasilitas penjelasan, dan perbaikan pengetahuan. Orang-orang yang terlibat dalam pembangunan ini adalah pakar, perekayasa pengetahuan, sistem analis dan programmer.

Pada bab ini akan membahas tentang penuntun atau pedoman umum untuk membangun suatu sistem pakar yang dirancang untuk aplikasi di dunia nyata, bukan prototipe penelitian. Metodologi rekayasa perangkat lunak *(software engineering)* digambarkan sedemikian hingga suatu sistem pakar dapat menghasilkan suatu pengembangan produk yang berkualitas dengan biaya murah dan dengan waktu yang baik.

Dalam hal ini kualitas memberi arti bahwa produk harus memenuhi karakter-karakter berikut ini :

1. Ketepatan, yaitu program harus memenuhi spesifikasinya sehingga dapat sungguh-sungguh melaksanakan tugas-tugas yang wajar. Wajar disini bermakna apa yang telah dapat diharapkan untuk diberikan kepadanya dalam pengertian spesifikasi (tak wajar) umumnya sistem menggambarkan tugas di mana sistem tidak mungkin dirancang.
2. Ketegaran, yaitu harus tidak terlalu sensitif terhadap error, dan kesalahan dalam tugas-tugas atau presentasi wajar untuk tugas-tugas

yang tak wajar. Dengan kata lain, ia harus dapat mengatasi error dengan mengubahnya tanpa membuat kesalahan berat dan penurunannya lebih lembut daripada terjal.

3. Readibilitas, yaitu penyandian harus ditulis sedemikian sehingga ia mampu dimenegerti oleh pemrograman lain.
4. Maintainabilitas, yaitu sistem harus dirancang dan diimplementasikan sedemikian rupa sehingga dengan melakukan relatif sedikit perubahan telah dapat memberi efek tanpa harus menulis ulang secara lengkap. Ini melibatkan perancangan dengan pengubahan dalam pikiran, walaupun berdasarkan pengamatan tidak semua tipe perubahan dapat diantisipasi kemudian.

XII.2 Pemilihan Masalah yang Tepat

Sebelum Anda membangun suatu sistem pakar, Anda harus memilih suatu masalah yang tepat. Ada beberapa jenis masalah yang dianggap cocok untuk sistem pakar, yaitu :

Kelas	Area umum
Konfigurasi	Merakit komponen sistem dengan cara yang benar
Diagnosa	Menarik kesimpulan dari masalah yang didasarkan pada fakta-fakta yang diobservasi
Instruksi	Pengajaran yang cerdas sehingga siswa dapat bertanya mengapa, bagaimana dan *what if*, seperti pengajaran yang dilakukan manusia
Monitoring	Membandingkan data yang diobservasi dengan data yang diharapkan untuk menilai performanya
Perencanaan	Merencanakan tindakan untuk mendapatkan hasil yang diinginkan
Prognosis	Memprediksi hasil dari situasi yang sudah ada
Perbaikan	Menentukan perlakuan untuk suatu masalah
Kontrol	Mengatur proses, yang mungkin membutuhkan interpretasi, diagnosis, monitoring, perencanaan, prognosis dan perbaikan

Kemungkinan daerah aplikasinya adalah industri manufaktur, computer aided design, pertahanan, finansial, adminitrasi, penjualan, pemasaran, pendidikan dan latihan, data prosesing dan lain sebagainya.

1. Pemilihan Paradigma (Model Pola) yang Tepat

Pertanyaan yang timbul dari pemilihan ini adalah mengapa kita membangun suatu sistem pakar? Pertanyaan ini boleh sangat penting untuk dijawab bagi setiap proyek yang akan dilakukan. Seperti yang telah diuraikan pada bab sebelumnya tentang keuntungan umum dari pembangunan sistem pakar, maka hanya managemen yang mengautorisasi sistem dan personil teknik mengimplementasikan yang dibutuhkan. Secara khusus dapat dikatakan bahwa jawaban pertanyaan ini pada akhirnya harus diberikan kepada pemilik atau pemegang saham yang membiayai pengembangan sistem. Sebelum memulai pembangunan sistem inisebaiknya ditentukan dulu dengan jelas identifikasi masalahnya, pakarnya, dan penggunanya.

2. Payoff

Pertanyaan yang muncul untuk ini adalah apa *payoff*-nya? Pertanyaan ini berhubungan dengan pertanyaan sebelumnya. Walaupun pertanyaan ini lebih pragmatis karena lebih tertuju kepada investasi dari orangnya, sumber, waktu dan uang.

3. Tools

Pertanyaan dari bagian ini adalah tool-tool apa saja yang tersedia untuk membangun sistem?

Banyak tool sistem pakar yang ada saat ini dengan keuntungan dan kelebihannya. Secara umum Anda dapat menghitung suatu substansi dari perbaikan setiap tahunnya bagi tiap tool dan revisi utama dalam jangka dua sampai tiga tahun.

Perbaikan-perbaikan ini tidak terbatas untuk tool-tool perangkat lunak. Panduan terbaik adalah melakukan pemeriksaan literatur terkini dan membicarakannya kepada orang yang telah membangun sistem pakar.

4. Biaya

Pertanyaannya adalah berapa biaya yang akan dihabiskan?

Biaya untuk pembangunan suatu sistem pakar tergantung pada manusia, sumber daya, dan waktu yang tersedia untuk mengkonstruksinya. Disamping itu perangkat keras dan lunak juga dibutuhkan untuk menjalankan tool sistem pakar. Selain itu jugaperlu dipertimbangkan biaya untuk pelatihan. Jika ada personil Anda yang memiliki sedikit pengalaman tentang tool, bahkan tidak sama sekali, maka akan diperlukan biaya untuk melatih mereka.

XII.3 Jenis Alat Pengembangan

Suparman (1991) menuliskan bahwa ada dua jenis alat untuk mengembangkan sistem pakar. Dasar yang bisa dipergunakan untuk dipertimbangkan penggunaannya adalah :

1. Bahasa Pemrograman

   Bahasa tentunya mengacu pada bahasa pemrograman yang digunakan oleh komputer untuk membuat suatu perangkat lunak baru. Sistem pakar dewasa ini telah dibuat atau diciptakan dengan berbagai jenis bahasa pemrograman dan bahkan hampir semua bahasa pemrograman telah digunakan untuk pembuatan sistem pakar ini. Beberapa jenis lebih mudah digunakan untuk maksud ini bila dibandingkan dengan yang lainnya. Sistem pakar telah dibuat dengan menggunakan bahasa pemrograman BASIC, Fortran, C, Pascal, Fort, FoxPro, Delphi, dan bahasa Assembly. Tentu saja bahasa yang lebih tinggi seperti Pascal dan Fortran lebih mudah digunakan, tetapi kedua bahasa ini membutuhkan jumlah kode yang sangat besar. Selama hasil programnya besar, bahasa C dan Fort merupakan dua bahasa yang lebih efisien. Untuk bisa mencapai efisiensi yang lebih tinggi, tidak ada yang mengalahkan bahasa Assembly karena ia lebih kompak dan kecepatannya diandalkan.

2. Shell

   Shell sistem pakar atau disebut juga generator merupakan paket perangkat lunak yang khusus dibuat untuk membantu pembuatan sistem pakar. Dalam beberapa hal sama dengan DBMS atau spreedsheet, shell menyediakan kerangka kerja dasar di dalam mana data atau pengetahuan dapat dimasukkan atau dimanipulasi dengan cara yang sudah ditentukan terlebih dahulu.

   Shell sistem pakar tidak mengandung pangkalan pengetahuan. Perbedaan antara sistem pakar dan lainnya terletak pada isi pangkalan pengetahuannya. Motor inferensi pangkalan data, dan user interface akan bisa bekerja dengan semua pangkalan pengetahuan.Singkatnya Anda hanya perlu memasukkan kode dalam format yang sudah dirancang ke dalam pangkalan pengetahuan.

   Generator sistem pakar adalah sama dengan shell, bahkan ia bisa menyediakan lingkungan pengembangan yang lebih canggih. Shell merupakan utilitas terpasang yang bisa mempermudah dan mempercepat proses pengembangan. Salah satu utilitasnya adalah editor. Editor ini memberikan kemungkina pada ada untuk memasukkan pengetahuan ke dalam format kaidah yang sudah ditemukan terlebih dahulu dan kemudian mengeditnya sesuai kebutuhan. Beberapa generator sistem pakar sebenarnya adalah sejenis compiler. Sesudah kaidah dimasukkan ke dalam editor, generator mengkompilasinya ke

dalam kode terakhir, kemudian motor inferensi dan user interface menggunakannya.

Hampir semua shell dan generator sistem pakar menggunakan skema representasi format kaidah IF...THEN. Kaidah ini sederhana, luwes dan sangat populer. Bila Anda sudah selesai merekayasa pengetahuan dan merancang program, hasilnya akan dipasang kedalam kaidah IF...THEN yang Anda masukkan ke dalam generator sistem pakar.

Latihan

1. Buatlah contoh perancangan sistem pakar

BAB XIII

PENJELASAN BARU UNTUK PENJELASAN SISTEM PAKAR

Pokok Bahasan
XIII.1 Penjelasan Sistem Pakar
XIII.2 Tipe Pengetahuan dalam Penjelasan Sistem Pakar
XIII.3 Asisten Keamanan
XIII.4 Metodologi

Setelah mempelajari uraian pada bab ini diharapkan dapat memahami penjelasan sistem pakar, tipe pengetahuan dalam penjelasan sistem pakar, asisten keamanan, dan metodologi.

XIII.1 Penjelasan Sistem Pakar

Sistem pakar merupakan salah satu aplikasi pertama yang muncul dari riset awal dalam bidang kecerdasan buatan, dan penjelasan dari penalaran sistem pakar merupakan salah satu aplikasi pertama dari generasi bahasa alami. Hal ini disebabkan oleh kebutuhan untuk penjelasan adalah nyata, dan generasi dari aplikasi basis pengetahuan seperti penalaran harus secara relatif dan secara langsung. Namun demikian manakala penjelasan yang bersifat universal telah diakui sebagai suatu keinginan fungsionalitas dalam sistem pakar. Generasi bahasa alami belum mengambil suatu posisi penting dalam sistem pakar. Generasi bahasa alami belum mengambil suatu posisi penting dalam sistem pakar. Sebagai contoh, dalam buku teks karangan Giarratano dan Riley yang populer tentang sistem pakar. Dalam buku ini ada sebanyak dua kali penekanan tentang pentingnya fasilitas penjelasan tersebut.

Dalam bab ini akan diuraikan suatu pendekatan baru untuk melengkapi sistem pakar dengan suatu fasilitas penjelasan. Pendekatan ini meliputi komponen perangkat lunak dan suatu metodologi untuk kumpulan komponennya.

Hasil awal yang sangat penting adalah (berdasarkan pada pengalaman dengan penjelasan dalam sistem pakar seperti MYCIN (Shortlife, 1976) yang telah menemukan bahwa ”Penalaran strategi yang dikerjakan oleh program tidak membentuk suatu basis yang baik bagi penjelasan yang dapat dipahami.” (Moore,1994). Secara khusus hanya menafsirkan urutan atau rantai penalaran dari sistem pakar tidak membiarkan seorang pemakai

dengan mudah memahami penalaran tersebut.

Ada dua pendekatan terpisah yang telah diusulkan untuk menunjukkan masalah ini :

1. Pendekatan *Explainable Expert System* (EES), (Swartout, dkk.: 1991; Swartout dan Moore: 1993), representasi pengetahuan yang digunakan oleh sistem pakar diperkaya untuk memasukkan pengetahuan strategi eksplisit. Sebagai contoh, pengetahuan tentang bagaimana menalarkan dan domain pengetahuan spesifik. Dari pengetahuan ini, aturan yang digunakan oleh sistem pakar dikompail atau disusun, dan pengetahuan ini juga digunakan untuk memberikan penjelasan yang lebih abstrak dari suatu penalaran sistem.
2. Pendekatan *Reconstructive Explainer* (Rex), (Wick: 1993). Sistem pakar tidak diubah, tetapi setelah itu sistem pakar telah melakukan penalarannya, suatu rangkaian sebab-akibat untuk penjelasandibangun dari masukan data sampai kesimpulan yang dicapai sebelumnya oleh sistem pakar sebagai suatu proses yang terpisah. Pekerjaan (Tanner dkk., 1993) dapat juga dilihat sebagai suatu kemerosotan dalam paradigma ini, karena pemisahan (representasi fungsional) digunakan hanya untuk penjelasan dan penjelasan harus secara khusus diperoleh dari ini.

Kedua pendekatan ini secara umum mempunyai keasyikan dengan penggolongan pengetahuan menggunakan sistem ke dalam tipe yang berbeda-beda. *Explainable Expert System* berkonsentrasi pada suatu penyajian abstrak dari pengetahuan strategis (bagaimana cara melakukan tindakan tertentu dari sistem sehubungan dengan keseluruhan goal (tujuan?) dan pada representasi dari perancangan rasional (mengapa tidakan layak yang nampak dari domain tujuan?). Sebagai tambahan, hal ini menurut domain pengetahuan (definisi istilah). *Reconstructive Explainer* dan pendekatan terkait mempunyai suatu representasi domain pengetahuan, sepanjang domain kaidah pengetahuan (sebagian besar, hubungan sebab akibat), yang secara lebih lengkap terpisah dari yang digunakan oleh sistem pakar itu sendiri. Pengetahuan ini digunakan untuk memperoleh suatu "alur penjelasan" melalui domain representasi pengetahuan.

Ada masalah dengan kedua pendekatan tersebut, yaitu EES belum terbukti sebagai suatu solusi yang memuaskan untuk masalah penjelasan sistem pakar. Masalahnya adalah bahwa penulis tentang sistem pakar belum begitu cepat atau bagus untuk mengadopsi *frameworks* (kerangka) seperti EES. Persyaratan untuk representasi abstrak pengetahuan (dari kaidah sistem pakar yang sebenarnya dikompile) menunjukkan EES yang dipaksakan mungkin lebih dipertimbangkan oleh pengembang sistem pakar, muncul tanpa motivasi dari segi pandangan inti sistem, yang dinamakan dengan

penalaran (merupakan lawan dari penjelasan). Agaknya, hal ini sukar bagi satu atau orang yang sama pakar dalam suatu domain pakar pada komunikasi dalam domain.

Dalam bab ini akan diuraikan pendekatan baru, yaitu sistem dan metodologi untuk penjelasan sistem pakar dimana tidak membutuhkan penulis sistem pakar untuk mempertimbangkan kebutuhan penjelasan pada saat penulisan kaidah. Pada saat yang sama juga mencegah keperluan kepemilikan pemisahan domain komponen penalaran untuk generasi penjelasan. Sebagai gantinya, sistem pakar sebagian besar dipertimbangkan sebagai suatu aplikasi yang berdiri sendiri, daripada sebagai penjelasan yang ditambahkan. Akan tetapi ini dilakukan oleh kepemilikan suatu rancangan komunikasi pakar representasi pengetahuan kedua (memisahkan dari domain sistem pakar representasi pengetahuan) yang secara spesifik ditujukan untuk kepentingan komunikasi penjelasan. Representasi ini didiami oleh sistem pakar, seperti penalarannya, bukan *post-hoc*. Jadi bukanlah memisahkan fasilitas penalaran yang dibutuhkan.

XIII.2 Tipe Pengetahuan dalam Penjelasan Sistem Pakar

Berdasarkan pembahasan sebelumnya dalam pembedaan jenis pengetahuan yang berbeda, maka dalam tulisan ini pengetahuan diklasifikasikan menjadi dua, yaitu untuk apa digunakan dan siapa yang bertanggungjawab untuk rekayasanya, bukan oleh struktur atau isinya. Secara khusus dalam bab ini ada tiga tipe pengetahuan yang akan dibahas, yaitu :

1. *Resoning Domain Knowledge* (RDK), merupakan domain pengetahuan yang dikodekan oleh domain pakar dalam sistem pakar yang sesuai.
2. *Communication Domain Knowledge* (CDK), merupakan pengetahuan tentang domain yang diperlukan untuk komunikasi tentang domain itu.
3. *Domain Communication Knowledge (DCK)* merupakan pengetahuan tentang bagaimana cara mengkomunikasikan domain itu.

Perbedaan mungkin pada mulanya nampak tidak begitu jelas, namun demikian setiap tipe pengetahuan tersebut berbeda satu dengan lainnya. CDK adalah domain pengetahuan, tetapi hanya domain pengetahuan yang dibutuhkan untuk komunikasi, tidak untuk penalaran. RDK dan CDK saling tumpang tindih, tetapi keduanya tidak identik.

CDK berbeda dengan DCK dimana CDK adalah pengetahuan tentang domain yang dibutuhkan untuk komunikasi sedangkan DCK adalah pengetahuan tentang bagaimana cara mengkomunikasikan domain tersebut. DCK bukanlah pengetahuan tentang domain, tetapi tentang teks. Sebagai

contoh, hal ini mungkin saja telah diekspresikan dalam komunikasi rencana operator dimana tujuan yang dicapai berhubungan dengan keadaan kognitif pendengar. Sementara itu domain pengetahuan tidak akan pernah meliputi rencana operator yang berhubungan dengan keadaan kognitif pendengar karena pendengar bukanlah bagian dari domain.

CDK bukanlah konsep baru. Banyak peneliti telah mengidentifikasikan kebutuhan untuk pengemasan domain pengetahuan secara berbeda untuk komunikasi. Sebagai contoh, pandangan dari Lester dan Poster (1997) dapat dilihat sebagai bentuk dari CDK, meskipun mereka tidak mendeklarasikan representasinya. Apa yang baru dari bahasan ini, tetapi apakah usulan bahwa CDK seharusnya direpresentasikan secara eksplisit dalam representasi yang berbeda dari domain pengetahuan.

XIII.3 Asisten Keamanan

Security assistant atau disingkat dengan SA (Webber, dkk. :1998) adalah bagian dari tool perancangan pakar (Ehrhat, dkk.: 1998) yang membantu perekayasa perangkat lunak menganalisis sistem cerdas (atau bukan fungsional) yang membutuhkan keamanan, toleransi kesalahan dan interaksi antara manusia dan komputer. SA membantu perekayasa perangkat lunak dalam memilih besarnya pengamanan untuk memproteksi aset sistem yang bernilai (seperti data yang penting) melawan adanya kemungkinan ancaman (seperti penyingkapan atau korupsi). Berikut ini akan dibahas tentang bagaimana ketiga tipe pengetahuan sebelumnya, yaitu RDK, CDK, dan DCK direpresentasikan dan digunakan dalam SA.

XIII.4 Metodologi

Dalam tulisan ini ada beberapa metodologi yang diajukan untuk pengembangan suatu penjelasan sistem pakar. Diasumsikan ada tiga *role* (peran), yaitu pakar domain (dimana domain mengacu pada suatu sistem pakar, seperti keamanan komputer, atau infeksi penyakit), perekayasa pengetahuan (spesialis dalam mengemukakan dan merepresentasikan model domain, khususnya dalam bentuk sistem pakar), dan perekayasa komunikasi (spesialis dalam analisis dan representasi pengetahuan yang dibutuhkan untuk efisiensi komunikasi). Metodologi tersebut dapat dijelaskan sebagai berikut :

1. Perekayasa pengetahuan *(knowledge engineer)* menciptakan sistem pakar yang dikonsultasikan dengan pakar domain dengan menggunakan sembarang tool atau shell dan sembarang jenis metodologi yang dipercaya.
2. Pakar domain *(domain expert)* menulis beberapa contoh penjelasan

tekstual dari tipe yang dibutuhkan untuk aplikasi yang ditanyakan, berdasarkan skenario yang dapat ditangani oleh sistem pakar.

3. Perekayasa komunikasi *(communication engineer)* menganalisis jumlah dari penjelasan tulisan tangan sepanjang dua baris :
    - Konsep domain yang dilaporkan dalam bentuk teks dianalisis dan direkam/disimpan menggunakan suatu teknik pemodelan berorientasi objek, barangkali dengan perluasan dengan konstruksi yang lebih ekspresif seperti meta-relasi (relasi antarrelasi). Struktur ini dinamakan dengan *content representation graph,* merepresentasikan komunikasi domain pengetahuan.
    - Struktur teks disimpan menggunakan beberapa notasi standar untuk struktur tulisan.
4. Menggunakan representasi komunikasi domain pengetahuan, perekayasa komunikasi mengkonsultasikan dengan pakar domain dan perekayasa pengetahuan untuk mendefenisikan pemetaan dari representasi domain yang digunakan oleh sistem pakar untuk representasi komunikasi domain pengetahuan yang dipikirkan oleh perekayasa komunikasi. Representasi komunikasi domain pengetahuan boleh jadi dimodifikasikan sebagai suatu hasil.
5. Perekayasa pengetahuan menambah aturan untuk sistem pakar bahwa instansiasi representasi komunikasi domain pengetahuan dengan contoh pembangkitan hingga proses penalaran.
6. Perekayasa komunikasi merancang perencana teks yang mendukung pengetahuan dalam representasi CDK dan menghasilkan teks. *Task* ini termasuk kreasi representasi eksplisit dari DCK untuk domain dan *task.*

Sistem yang dihasilkan bersifat modular dan berkaitan dengan modul perangkat lunak. Sistem pakar dipelihara sebagai suatu model yang berdiri sendiri seperti perencana teks. Sebagai tambahan, metodologi merupakan modul yang berkaitan dengan teks dan kepakaran, maka pakar domain dan perekayasa pengetahuan tidak perlu belajar tentang komunikasi, dan perekayasa pengetahuan tidak perlu memahami kerja dari sistem pakar.

BAB XIV
PROJECT PEMBUATAN SISTEM PAKAR

Setelah mempelajari uraian pada bab ini diharapkan dapat memahami dalam pembuatan sistem pakar. Dalam bab ini membahas contoh penerapan teorema bayes dalam sistem pakar untuk diagnosis penyakit tropis. Domain yang akan diambil dibatasi pada penyakit malaria, demam berdarah dan demam tifoid (tifus).

1. Jenis Penyakit

a. Malaria

Malaria adalah suatu penyakit yang disebabkan oleh sporozoa dari genus Plasmodium, yang secara klinis ditandai dengan serangan paroksismal dan periodik, disertai anemia, pembesaran limpa dan kadang-kadang dengan komplikasi pernisiosa seperti ikterik, diare, black water fever, acute tubular necrosis dan malaria cerebral

Penduduk yang terancam malaria pada umumnya adalah penduduk yang bertempat tinggal di daerah endemis malaria tinggi dan daerah endemis malaria sedang diperkirakan ada sekitar 15 juta. Proses terjadinya penularan malaria di suatu daerah meliputi tiga faktor utama :
(a) adanya penderita baik dengan adanya gejala klinis ataupun tanpa gejala klinis. (b) adanya nyamuk atau vector. (c) adanya manusia yang sehat.

Malaria sebagai penyakit infeksi yang disebabkan oleh plasmodium mempunyai gejala utama demam. Diduga terjadinya demam berhubungan dengan proses skizogoni (pecahnya erozoit/skizon). Gambaran karakteristik malaria ialah demam periodik, anemia dan splenomegali. Berat-ringan manifestasi malaria bergantung pada jenis plasmodium yang menyebabkan infeksi.

b. Demam Berdarah Dengue (DBD)

Penyakit Demam Berdarah Dengue (DBD) adalah suatu penyakit menular yang disebab-kan oleh virus dengue terutama menyerang anak-anak dengan ciri-ciri demam tinggi mendadak disertai manifestasi perdarahan dan bertendensi menimbulkan syok dan kematian. Infeksi virus dengue dapat memperlihatkan spektrum klinis bervariasi dari derajat paling ringan sampai berat. Infeksi dengue yang paling ringan adalah demam tanpa penyebab yang jelas (*undifferentiated febrile ill-ness*), diikuti dengan demam dengue (DD), demam berdarah dengue (DBD) dan sindrom syok de-ngue (SSD).

c. Demam Typoid *(Typhoid Fever)*

Demam typoid masih merupakan penyakit infeksi tropik sistemik, bersifat endemis, dan ma-sih merupakan problema kesehatan masyarakat pada negara-negara sedang berkembang di dunia, ter-masuk Indonesia. Data secara epidemiologi setiap tahun diperoleh dari beberapa negara yang mencat-at hasil laporannya dari diagnosis klinik atau isolat laboratorium, karena data yang benar-benar dapat menggambarkan insiden penyakit ini di masyarakat sukar didapatkan. Hal ini disebabkan karena gam-baran klinik penyakit demam typoid menyerupai penyakit infeksi lainnya dan juga konfirmasi laboratorik tidak selalu dapat dikerjakan pada semua daerah.

Dengan melihat data tersebut di atas, baik insiden penyakit demam typoid yang makin meni-ngkat maupun angka kematian yang disebabkan penyakit tersebut, maka diagnosis dini demam typoid perlu segera ditegakkan, oleh karena itu pemeriksaan baku atau rutin secara serologi yang sampai saat ini masih dikerjakan hampir pada semua pasien yang dirawat dengan demam di Rumah Sakit, yaitu uji Widal.

2. Pembahasan

a. Penyusunan Basis Pengetahuan

Sistem pakar untuk mendiagnosa penyakit malaria, demam berdarah dan demam typoid pada manusia ini membutuhkan pengetahuan dan mesin inferensi untuk mendiagnosa penyakit yang dialami pengguna. Basis pengetahuan ini berisikan faktor-faktor yang dibutuhkan oleh sistem. Sedangkan mesin inferensi digunakan untuk menganalisa faktor-faktor yang dimasukkan pengguna sehingga dapat ditemukan suatu kesimpulan basis pengetahuan yang diperlukan sistem terdiri dari gejala penyakit, jenis penyakit. Data yang menjadi input sistem data gejala yang dapat dari pemeriksaan yang dila-kukan oleh para medis. Data tersebut digunakan oleh sistem untuk menentukanjenis penyakit yang di-derita pasien. Pembentukan aturan gejala penyakit dari ini ditunjukkan pada tabel XIV.1.

Tabel XIV.1. Aturan Gejala Penyakit

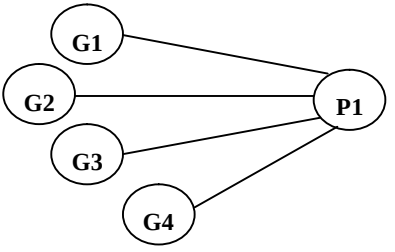
No	Aturan
1	If Demam1 and endemis and muntah or pucat then sus. Malaria
2	If Demam1 and endemis and muntah or splenomegali then sus. Malaria
3	If Demam1 and endemis and muntah or hepatomegali then sus. Malaria
4	If sus. Malaria and parasit malaria then Malaria
5	If Demam2 and bercak merah and muntah or mimisan then sus. Demam berdarah
6	If Demam2 and bercak merah and muntah or gusi

	berdarah then sus. Demam berdarah
7	If Demam2 and bercak merah and muntah or splenomegali then sus. Demam berdarah
8	If Demam2 and bercak merah and muntah or splenomegali then sus. Demam berdarah
9	If Demam2 and bercak merah and muntah or hepatomegali then sus. Demam berdarah
10	If sus. Demam berdarah and Anti DHF-IgG positive and Anti DHF-IgM positive then Demam berdarah
11	If Demam3 and muntah and hepatomegali and mencret or susah buang air besar then sus Demam Typoid
12	If Demam3 and muntah and hepatomegali and mencret or lidah kotor then sus Demam Typoid
13	If sus. Demam Typoid and tes widal positive then Demam Typoid
14	If Menggigil and demam > 38^oC secara periodik and berkeringat then Demam1
15	Dan aturan lainnya

Dalam sistem ini metode inferensi yang digunakan adalah *forward chaining* karena proses yang dialami dengan menampilkan gejala penyakit. *Forward chaining* digunakan untuk menguji fa-ktor-faktor yang dimasukkan pengguna dengan aturan yang disimpan dalam sistem satu demi satu hingga dapat diambil satu kesimpulan *forward chaining.* Berikut ini diberikan contoh Graf penelusuran penyakit untuk 1 penyakit :

- Malaria

Graf penelusuran penyakit malaria ditunjukkan pada gambar XIV.1 mempunyai empat gejala yang digunakan sebagai berikut :



Gambar XIV.1 Graf penyakit malaria

Keterangan :

Gejala :

G1	:	Menggigil, demam > 38^{o}C secara periodik, berkeringat
G2	:	tinggal di daerah endemis
G3	:	muntah
G4	:	pucat

b. Pengujian kebenaran

Pengujian kebenaran sistem ini dilakukan dengan perhitungan untuk mengetahui hasil akhir. Untuk hasil akhir harus mengetahui beberapa gejala yang dirasakan oleh pasien kemudian setelah se-lesai maka akan menampilkan hasil jenis penyakit yang dialami oleh pasien. Perhitungan nilai kemu-ngkinan dilakukan contoh dengan beberapa gejala beberapa jenis penyakit .

Pada perhitungan beberapa gejala dengan beberapa penyakit, percobaan 1 akan menggunakan gejala hepatomegali dengan nilai = 0.7, dan dengan gejala mual & muntah dengan nilai = 0.5 dengan kemungkinan mengalami penyakit malaria dengan nilai = 0.2

Berdasarkan data diatas, maka hasil perhitungan sebagai berikut :

$$P(H_1|E_1E_2) = \frac{(0.7)(0.5)(0.2)}{(0.7)(0.5)(0.2)+(0.6)(0.5)(0.1)+(0.5)(0.6)(0.3)}$$

$$= \frac{0.07}{0.19}$$

= 0.36 (Penyakit Malaria)

Percobaan 2 akan menggunakan gejala hepatomegali dengan nilai = 0.6, dan dengan gejala mual & muntah dengan nilai = 0.5 dengan kemungkinan mengalami penyakit demam berdarah dengan nilai = 0.1 Berdasarkan data diatas, maka hasil perhitungan sebagai berikut :

$$P(H_2|E_1E_2) = \frac{(0.6)(0.5)(0.1)}{(0.7)(0.5)(0.2)+(0.6)(0.5)(0.1)+(0.5)(0.6)(0.3)}$$

$$= \frac{0.03}{0.19}$$

= 0.15 (Penyakit Demam Berdarah)

Percobaan 3 akan menggunakan gejala hepatomegali dengan nilai = 0.5, dan dengan gejala mual & muntah dengan nilai = 0.6 dengan kemungkinan mengalami penyakit demam typoid dengan nilai = 0.3 Berdasarkan data diatas, maka hasil perhitungan sebagai berikut :

$$P(H_3|E_1E_2) = \frac{(0.5)(0.6)(0.3)}{(0.7)(0.5)(0.2)+(0.6)(0.5)(0.1)+(0.5)(0.6)(0.3)}$$

$$= \frac{0.09}{0.19}$$

= 0.47 (Penyakit Demam Typoid)

$= \max(\ H_1, H_2, H_3 | E_1, E_2\ )$

$= \max(\ 0.36, 0.15, 0.47\ )$

$= 0,47$ (Penyakit Demam Typoid)

3. Kesimpulan

Dari uraian sebelumnya, dapat diambil kesimpulan bahwa sistem pakar diagnosa penyakit malaria, demam berdarah dan demam typoid dengan menggunakan metode probabilistik teorema bayes ini mampu memberikan nilai kemungkinan untuk beberapa jenis penyakit berdasarkan gejala-gejala yang dialami pasien.

DAFTAR PUSTAKA

Arhami, Muhammad., 2005, **Konsep Dasar Sistem Pakar**, Penerbit Andi, Yogyakarta.

Desiani, Anita dan Arhami, Muhammad., 2006, **Konsep Kecerdasan Buatan**, Penerbit Andi, Yogyakarta.

Giarratano J.,G. Riley.,1994**, Expert System Principles and Programming**, Carlson, second edition, PWS Publishing Company, Boston

Hartati, Sri., Iswanti, Sari., 2008, **Sistem Pakar dan Pengembangannya**, Graha Ilmu, Yogyakarta

Kusumadewi S., 2003, **Artificial Intellegence (Teknik dan Aplikasinya)**, Penerbit Graha Ilmu, Yogyakarta, Indonesia.

Kusrini, 2006, **Sistem Pakar Teori dan Aplikasi**, Penerbit Andi, Yogyakarta

Rosnelly Rika., Hartati Sri.,2010, **Penggunaan Teorema Bayes Dalam Sistem Pakar Untuk Mendiagnosa Penyakit Pada Manusia**, Infosys Journal,Volume I No.1

Rosnelly Rika., Wardoyo Retantyo., 2010, **Penerapan Teorema Bayes untuk Mendiagnosa Penyakit Pada Manusia**, Snikom, Volume I.

Rosnelly Rika., Hardjoko Agus., 2011, **Pengembangan Sistem Informasi Diagnosis Penyakit Tropis Menggunakan Algoritma Naive Bayesian**, KNS&I, Stikom Bali, ISSN:1979-9845

Tim Penerbit Andi, 2003, **Pengembangan Sistem Pakar Menggunakan Visual Basic**, Andi Yogyakarta.

PUBLIKASI PENULIS :

Rosnelly, Rika., Hartati, Sri.,2010, **Penggunaan Teorema Bayes Dalam Sistem Pakar Untuk Mendiagnosa Penyakit Pada Manusia**, Infosys Journal,Volume I No.1.

Rosnelly, Rika., Hartati, Sri.,2010, **Penalaran Berbasis Kasus *(Case Based Reasoning)* Untuk Diagnosis Penyakit Menggunakan Algoritma C4.5**, Infosys Journal,Volume I No.1.

Rosnelly, Rika., Wardoyo, Retantyo., 2010, **Penerapan Teorema Bayes untuk Mendiagnosa Penyakit Pada Manusia**, Snikom Universitas Sumatera Utara, Volume I.

Rosnelly, Rika., Hardjoko, Agus., 2011, **Pengembangan Sistem Informasi Diagnosis Penyakit Tropis Menggunakan Algoritma Naive Bayesian**, KNS&I, Stikom Bali, ISSN:1979-9845.

Rosnelly, Rika., Wardoyo, Retantyo., 2011, **Penerapan Fuzzy Multi Criteria Decision Making (FMCDM) Untuk Diagnosis Penyakit Tropis**, Seminar Nasional Informatika UPN Veteran Yogyakarta, ISSN: 1979-2328.

Rosnelly, Rika., Wardoyo, Retantyo., 2011, **Java Virtual Machine Pada Sistem Operasi Windows XP Untuk Mengeksekusi Sistem Penjualan**, Seminar Nasional Informatika UPN Veteran Yogyakarta, ISSN: 1979-2328.

Rosnelly, Rika., Pulungan, Reza., 2011, **Membandingkan Analisa Trafik Data Pada Jaringan Komputer Antara Wireshark dan NMap**, Konferensi Nasional Sistem Informasi, ISBN: 978-602-98768-0-2

Rosnelly, Rika., 2011, **Group Decision Support System (GDSS) Untuk Diagnosis Penyakit Tropis**, Seminar Nasional Hasil Penelitian MIPA, Universitas Gadjah Mada, Yogyakarta.

sistem pakar

konsep dan teori

Sistem pakar merupakan penyelesaian pendekatan yang tepat dan bagus untuk permasalahan AI (*Artificial Intelligence*) klasik dari pemrograman intelligent (cerdas). Sistem pakar (*expert system*) merupakan solusi AI bagi masalah pemrograman cerdas (*intelligent*).

Buku ini membahas konsep sistem pakar, struktur sistem pakar, representasi pengetahuan, metode inferensi, penalaran dengan ketidakpastian, peluang dengan teorema bayes, penalaran inexact, tahapan pengembangan sistem pakar, perancangan sistem pakar, penjelasan baru untuk penjelasan sistem pakar, dan project pembuatan sistem pakar.

Penerbit ANDI
Jl. Beo 38-40 Yogyakarta
Telp. (0274) 561881 Fax. (0274) 588282
e-mail: penerbitan@andipublisher.com
website: www.andipublisher.com

TEKS - KOMPUTER
ISBN: 978-979-29-3416-8
9 789792 934168 12301