BAB II

LANDASAN TEORI

2.1 *Artificial Intelligence* (AI)

Artificial intelligence (AI) merupakan cabang dari ilmu komputer yang *konsern* dengan pengautomatisasian tingkah laku cerdas. Pernyataan tersebut juga dapat dijadikan definisi dari AI. Definisi ini menunjukkan bahwa AI adalah bagian dari komputer sehingga harus didasarkan pada *sound theoretical* (teori suara) dan prinsip-prinsip aplikasi pada bidangnya. Prinsip-prinsip ini meliputi struktur data yang digunakan dalam representasi pengetahuan, algoritma untuk mengaplikasikan pengetahuan tersebut, serta teknik bahasa pemrograman yang digunakan dalam mengimplementasikannya[4].

2.1.1 Pengertian AI

Artificial intelligence (AI) disebut juga dengan kecerdasan buatan. AI merupakan salah satu bagian ilmu komputer yang mempelajari tentang bagaimana cara membuat agar komputer dapat melakukan pekerjaan seperti yang dilakukan oleh manusia[6]. Tujuan dari AI adalah untuk memecahkan persoalan dunia nyata (bersifat praktis) dan memahami intelijensia (bersifat memahami). AI merupakan salah satu bagian ilmu komputer yang mempelajari tentang bagaimana cara membuat agar komputer dapat melakukan pekerjaan seperti yang dilakukan oleh manusia[7].

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

Pada awal diciptakan, komputer hanya difungsikan sebagai alat hitung saja. Namun seiring dengan perkembangan zaman, maka peran komputer semakin mendominasi kehidupan umat manusia. Manusia bisa menjadi pandai dalam menyelesaikan segala permasalahan di dunia ini karena manusia mempunyai pengetahuan dan pengalaman. Semakin banyak bekal pengetahuan yang dimiliki oleh seseorang tentu saja diharapkan akan lebih mampu dalam menyelesaikan permasalahan. Namun bekal pengetahuan saja tidak cukup, manusia juga diberi akal untuk melakukan penalaran dan mengambil kesimpulan berdasarkan pengetahuan dan pengalaman yang mereka miliki. Tanpa memiliki kemampuan menalar yang baik, manusia dengan segudang pengalaman dan pengetahuan tidak akan dapat menyelesaikan masalah dengan baik. Demikian pula, dengan kemampuan menalar yang sangat baik, namun tanpa bekal pengetahuan dan pengalaman yang memadai, manusia juga tidak akan bisa menyelesaikan masalah dengan baik[6].

Komputer juga harus diberi bekal pengetahuan dan mempunyai kemampuan untuk menalar, agar komputer bisa bertindak seperti dan sebaik manusia. Untuk itu pada *artificial intelligence*, akan mencoba untuk memberikan beberapa metode untuk membekali komputer dengan kedua komponen tersebut agar komputer bisa menjadi mesin yang pintar. Untuk menciptakan aplikasi kecerdasan buatan ada dua bagian utama yang sangat dibutuhkan, yaitu:

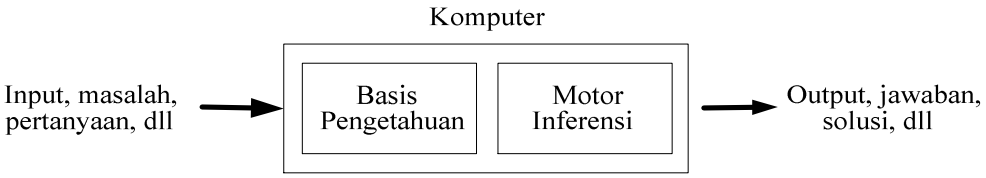
1. Basis Pengetahuan (*Knowledge Base*)

   Basis pengetahuan berisi fakta-fakta, teori, pemikiran dan hubungan antara satu dengan yang lainnya.

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

2. Motor Inferensi (*Inference Engine*)

   Motor inferensi merupakan kemampuan menarik kesimpulan berdasarkan pengalaman atau dapat juga disebut dengan penalaran. Dapat dilihat pada gambar 2.1.



Gambar 2.1 Sistem yang menggunakan AI

Definisi AI dapat dipandang dari berbagai sudut pandang, yaitu:

1. Sisi kecerdasan.

   AI akan membuat mesin menjadi ‘cerdas’ (mampu berbuat seperti apa yang dilakukan oleh manusia).

2. Sisi penelitian.

   AI adalah suatu studi bagaimana membuat agar komputer dapat melakukan sesuatu sebaik yang dikerjakan oleh manusia.

3. Sisi bisnis.

   AI adalah kumpulan peralatan yang sangat *powerful* dan metodologis dalam menyelesaikan masalah-masalah bisnis.

4. Sisi pemrograman.

   AI meliputi studi tentang pemrograman simbolik, penyelesaian masalah (*problem solving*) dan pencarian (*searching*).

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

2.1.2 Perbedaan antara AI dan Kecerdasan Alami

Dibandingkan dengan kecerdasan kecerdasan yang dimiliki oleh manusia (kecerdasan alami), AI memiliki beberapa keuntungan secara komersial antara lain:[4]

1. AI lebih bersifat permanen.

   Kecerdasan alami akan cepat mengalami perubahan. Hal ini dimungkinkan karena sifat manusia yang pelupa. AI tidak akan berubah sepanjang sistem komputer dan program tidak diubah.

2. AI lebih mudah diduplikasi dan disebarkan.

   Mentransfer pengetahuan manusia dari satu orang ke orang lain membutuhkan proses yang sangat lama, dan keahlian itu juga tidak akan pernah dapat diduplikasi dengan lengkap. Oleh karena itu, jika pengetahuan terletak pada suatu sistem komputer, pengetahuan tersebut dapat disalin dari komputer tersebut dan dapat dipindahkan dengan mudah ke komputer yang lain.

3. AI lebih murah dibanding dengan kecerdasan alami.

   Menyediakan layanan komputer akan lebih mudah dan lebih murah dibandingkan dengan harus mendatangkan seseorang untuk mengerjakan sejumlah pekerjaan dalam jangka waktu yang sangat lama.

4. AI bersifat konsisten.

   Hal ini disebabkan karena AI adalah bagian dari teknologi komputer, sedangkan kecerdasan alami akan senantiasa berubah-ubah.

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

5. AI dapat didokumentasi.

   Keputusan yang dibuat oleh komputer dapat didokumentasi dengan mudah dengan cara melacak setiap aktivitas dari sistem tersebut. Kecerdasan alami sangat sulit untuk direproduksi.

6. AI dapat mengerjakan pekerjaan jauh lebih cepat dibanding dengan kecerdasan alami.

7. AI dapat mengerjakan pekerjaan lebih baik dibanding dengan kecerdasan alami.

Keuntungan dari kecerdasan alami adalah:

1. Kreatif, kemampuan untuk menambah ataupun memenuhi pengetahuan itu sangat melekat pada jiwa manusia. Pada kecerdasan buatan, untuk menambah pengetahuan harus dilakukan melalui sistem yang dibangun.
2. Kecerdasan alami memungkinkan orang untuk menggunakan pengalaman secara langsung, sedangkan pada kecerdasan buatan harus bekerja dengan *input-input* simbolik.
3. Pemikiran manusia dapat digunakan secara luas, sedangkan AI sangat terbatas.

2.1.3 Ruang Lingkup AI pada Aplikasi Komersial

Semakin pesatnya perkembangan teknologi menyebabkan adanya perkembangan dan perluasan lingkup yang membutuhkan kehadiran AI. Karakteristik ‘cerdas’ sudah mulai dibutuhkan di berbagai disiplin ilmu dan teknologi. AI tidak hanya dominan pada bidang ilmu komputer (informatika),

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

namun juga sudah merambah di berbagai disiplin ilmu yang lain. Irisan antara psikologi dan AI melahirkan sebuah area yang dikenal dengan nama *cognition* & *psycolinguistics*. Irisan antara teknik elektro dengan AI melahirkan berbagai ilmu, seperti: pengolahan citra, teori kendali, pengenalan pola dan robotika[6].

AI juga memberikan kontribusi yang cukup besar di bidang manajemen. Adanya sistem pendukung keputusan dan sistem informasi manajemen juga tidak terlepas dari andil AI. Adanya irisan penggunaan AI di berbagai disiplin ilmu tersebut menyebabkan cukup rumitnya untuk mengklasifikasikan AI menurut disiplin ilmu yang menggunakannya. Untuk memudahkan hal tersebut, maka pengklasifikasian lingkup AI didasarkan pada *output* yang diberikan, yaitu pada aplikasi komersial (meskipun sebenarnya AI itu sendiri bukan merupakan medan komersial)[7].

Lingkup utama dalam kecerdasan buatan adalah:

1. Sistem Pakar (*Expert System*). Disini, komputer digunakan sebagai sarana untuk menyimpan pengetahuan para pakar. Dengan demikian, komputer akan memiliki keahlian untuk menyelesaikan permasalahan dengan meniru keahlian yang dimiliki oleh pakar.
2. Pengolahan Bahasa Alami (*Natural Language Processing*). Dengan pengolahan bahasa alami ini diharapkan *user* dapat berkomunikasi dengan komputer dengan menggunakan bahasa sehari-hari.
3. Pengenalan Ucapan (*Speech Recognition*). Melalui pengenalan ucapan diharapkan manusia dapat berkomunikasi dengan komputer menggunakan suara.

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

4. Robotika & Sistem Sensor (*Robotics & Sensory Systems*).
5. *Computer Vision*, mencoba untuk dapat menginterpretasikan gambar atau objek-objek tampak melalui komputer.
6. *Intelligent Computer-Aided Instruction*. Komputer dapat digunakan sebagai tutor yang dapat melatih dan mengajar.
7. *Game Playing*.

Beberapa karakteristik yang ada pada sistem yang menggunakan AI adalah pemrogramannya yang cenderung bersifat simbolik dari pada algoritmik, bisa mengakomodasi *input* yang tidak lengkap, bisa melakukan inferensi dan adanya pemisahan antara kontrol dengan pengetahuan. Namun, seiring dengan perkembangan teknologi, muncul beberapa teknologi yang juga bertujuan untuk membuat agar komputer menjadi cerdas sehingga dapat menirukan kerja manusia sehari-hari. Teknologi ini juga mampu mengakomodasi adanya ketidakpastian dan ketidaktepatan data *input*. Dengan didasari pada teori himpunan, maka pada tahun 1965 muncul Logika Fuzzy. Kemudian pada tahun 1975, John Holland mengatakan bahwa setiap masalah berbentuk adaptasi (alami maupun buatan) secara umum dapat diformulasikan dalam terminologi genetika. Algoritma genetika ini merupakan simulasi proses evolusi Darwin dan operasi genetika atas kromosom[6].

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

2.2 Representasi Pengetahuan

Langkah pertama dalam membuat program AI adalah membangun pangkalan pengetahuan. Agar komputer bisa bertingkah laku seperti seorang inteligensia, maka ia harus mempunyai pengetahuan terlebih dahulu tentang sesuatu domain tertentu. Sebenarnya pengetahuan itu sudah ada. Apakah ia masih berada dalam otak seseorang atau beberapa orang pakar, atau mungkin masih dalam bentuk buku, artikel, memo dan prosedur, yang penting semuanya harus sudah terkodifikasi. Proses pengumpulan dan pengorganisasian pengetahuan disebut rekayasa pengetahuan. Hal inilah yang paling sulit dan memerlukan waktu yang banyak dalam proses pembuatan program AI[4].

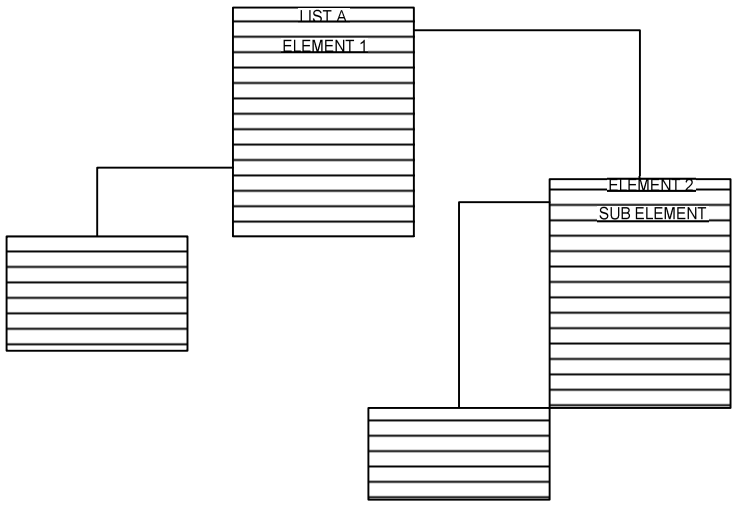
2.2.1 List

Algoritma ini bekerja dengan cara mencari secara berurutan. Kita bisa membayangkannya seperti saat kita ingin mencari seseorang dalam sebuah antrian. Maka kita mencarinya dengan cara memeriksa satu persatu, dari awal antrian hingga kita menemukan orang yang ingin kita cari. Cara atau algoritma seperti ini biasanya digunakan saat kita ingin mencari dengan menggunakan satu faktor atau satu kunci saja sebagai penentu. Untuk antrian yang pendek, cara ini mungkin cukup efektif dan efisien. Tapi untuk mencari sebuah kata dari milyaran web page yang ada di internet, maka akan membutuhkan waktu yang sangat lama.

List merupakan bentuk yang paling sederhana, tapi bila ada dua *list* atau lebih yang berkaitan, maka harus digabungkan dan perlu dibuat suatu hirarki. Untuk mengidentifikasikannya, list dapat diberi nama misalnya *List* A dengan dua

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

atau tiga elemen, maka dapat dilihat satu elemen dalam satu *list*, dan nama *list* lain mengandung sub-elemen. Contoh sederhana dari sebuah *list* dapat dilihat pada gambar 2.2.



Gambar 2.2 Contoh List

2.2.2 Skema

Skema dapat juga disebut dengan *frame*. Banyak pengetahuan yang digunakan dalam proses penalaran sehari-hari yang sudah pasti dan dikenal dengan baik. Hal ini didasarkan kepada berbagai penampilan situasi dan objek-objek khusus, seperti pengalaman biasa yang konvensional dan proses yang tak bervariasi. Pengetahuan semacam ini disebut sebagai pengetahuan stereotype dan direpresentasikan ke dalam sistem skema. Skema adalah satu metode pengorganisasian, presentasi dan penggunaan pengetahuan stereotype agar komputer bisa menalar.

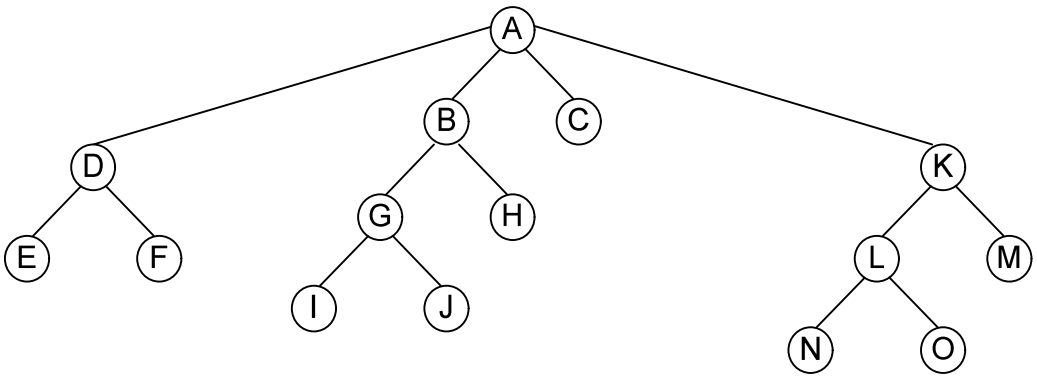
This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

Sebenarnya berbagai representasi skema pengetahuan itu sudah dikembangkan sejak bertahun-tahun yang lalu. Representasi skema ini mempunyai dua karakteristik umum, yaitu :

1. Bisa diprogram dengan bahasa komputer yang ada dan disimpan dalam memori.
2. Dirancang agar fakta-fakta dan pengetahuan lainnya yang terkandung di dalamnya bisa digunakan untuk penalaran.

2.2.3 Pohon

Struktur pohon adalah struktur grafik hirarki. Struktur ini merupakan cara yang sederhana untuk menggambarkan list hirarki pengetahuan lainnya. Contoh struktur pohon dapat dilihat pada gambar 2.3.



Gambar 2.3 Contoh Pohon

Struktur ini terdiri dari *node-node* yang mencakup nama *list* dan *arc* yang menunjukkan hubungan antar *node*. Struktur atau grafik ini disebut pohon karena mempunyai cabang-cabang, tetapi cabang dari pohon yang terbalik berbeda dengan pohon yang sebenarnya. Pohon sangat umum dipakai untuk

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

menggambarkan pengetahuan yang akan digunakan dalam AI. Contoh aplikasi pohon yang dapat kita lihat sehari-hari adalah pengelolaan *file* dalam direktori penyimpanan. Pohon merupakan struktur data yang memiliki suatu struktur hirarki pada sekumpulan elemen dan memiliki hubungan satu ke banyak (*one to many relationship*) seperti yang kita lihat dalam struktur organisasi sebuah perusahaan atau daftar isi sebuah buku[1].

Dalam struktur organisasi, kita dapat melihat bahwa ada level atas biasanya hanya ada satu pimpinan tertinggi. Pada level berikutnya diisi oleh beberapa orang dengan jabatan yang berbeda tetapi dalam tingkatan yang sama. Selanjutnya, dapat dipecah lagi ke level berikutnya sampai struktur dapat memenuhi fungsi dan tujuan organisasi. Biasanya satu atasan memiliki beberapa bawahan yang berada dalam ruang lingkup wewenang dan tugas atasan. Begitu juga dalam daftar isi buku, dimana satu buku terdiri dari beberapa bab dan setiap bab terdiri dari beberapa sub bab, satu sub bab terdiri dari beberapa sub-sub bab dan seterusnya. Dengan demikian hirarki dapat kita anggap sebagai “terdiri dari” atau “bawahan” atau “diawasi” dari atas ke bawah. Salah satu keuntungan pohon dibandingkan dengan struktur data linier adalah waktu cari sebuah node maksimum (dapat) lebih kecil dari n jika jumlah data = n[3].

Sebuah *tree* dapat mempunyai satu buah simpul tanpa sebuah sisi pun. Dengan kata lain, jika $G = (V, E)$ adalah tree, maka V tidak boleh berupa himpunan kosong, namun E boleh kosong. *Tree* juga seringkali didefinisikan sebagai graf tak-berarah dengan sifat bahwa hanya terdapat sebuah lintasan unik

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

antara setiap pasang simpul. Selain itu, di dalam *tree* jumlah sisinya adalah jumlah simpul dikurangi satu[5].

Secara sederhana, sebuah *tree* bisa didefenisikan sebagai kumpulan dari elemen–elemen yang disebut dengan *node / vertex* (simpul) dimana salah satu *node* disebut dengan *root* (akar), dan sisa *node* lain terpecah menjadi himpunan yang saling tidak berhubungan satu sama lain dan disebut dengan *subtree* (pohon bagian). Jika dilihat pada setiap *subtree* maka *subtree* juga mempunyai *root* dari *subtree*-nya masing – masing[4].

Dengan melihat istilah dasar di atas, maka sebuah *tree* secara rekursif dapat didefinisikan sebagai berikut :

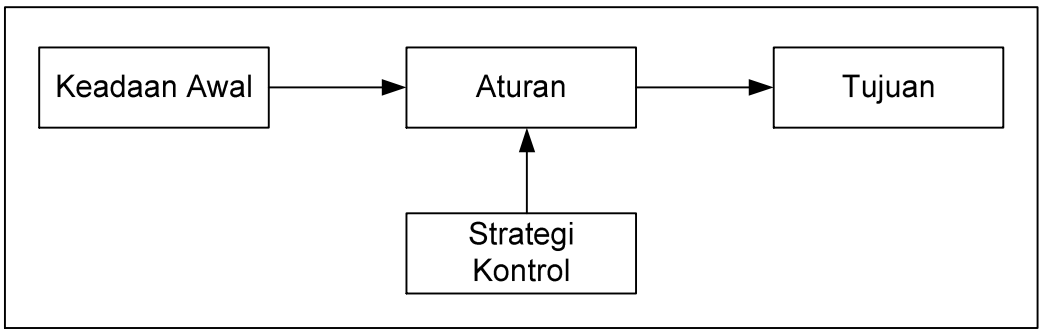
1. Sebuah *node* tunggal adalah sebuah *tree*.
2. Jika terdapat sebuah *node* N dan beberapa *subtree* N_1, N_2, N_3, …, N_k maka dari *node* N dan *subtree* yang ada dapat dibentuk sebuah *tree* yang mempunyai *root* pada *node* N.

2.2.4 Sistem Produksi (*Production System*)

Sistem produksi secara umum terdiri dari komponen-komponen sebagai berikut :

1. Ruang keadaan, yang berisi keadaan awal, tujuan dan kumpulan aturan yang digunakan untuk mencapai tujuan.
2. Strategi kontrol, yang berguna untuk mengarahkan bagaimana proses pencarian akan berlangsung dan mengendalikan arah eksplorasi.

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com



Gambar 2.4 Bagan Sistem Produksi

Sistem produksi ini merupakan salah satu bentuk representasi pengetahuan yang sangat populer dan banyak digunakan[7]. Representasi pengetahuan dengan sistem produksi pada dasarnya berupa aplikasi aturan (*rule*) yang berupa :

1. *Antecedent*, yaitu bagian yang mengekspresikan situasi atau premis.
2. Konsekuen, yaitu bagian yang menyatakan suatu tindakan tertentu atau konklusi yang diterapkan jika suatu situasi atau premis bernilai benar.

Konsekuensi atau konklusi baru dinyatakan benar, jika bagian *antecedent* pada sistem tersebut juga benar atau sesuai dengan aturan tertentu. Apabila pengetahuan direpresentasikan dengan aturan, maka ada dua metode penalaran yang dapat digunakan yaitu :

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

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

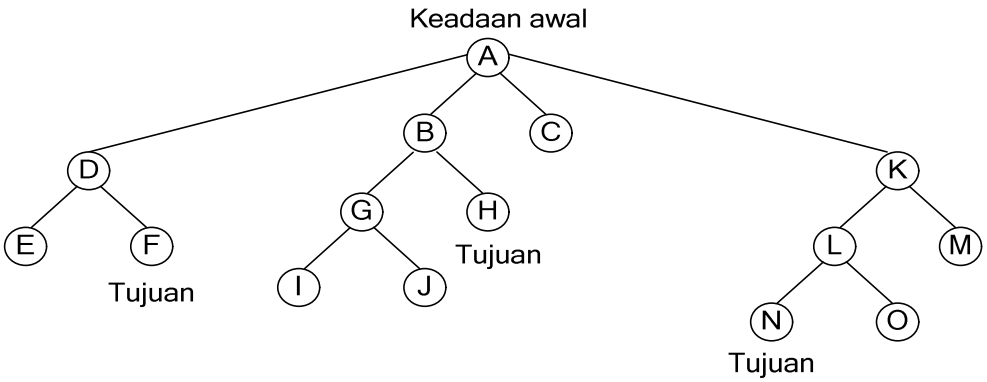
Ada beberapa faktor yang mempengaruhi pemilihan *backward* atau *forward* dalam memilih metode penalaran, antara lain :

a. Banyaknya keadaan awal dan tujuan. Jika jumlah keadaan awal lebih kecil daripada tujuan, maka digunakan penalaran *forward*. Sebaliknya, jika jumlah tujuan lebih banyak daripada keadaan awal, maka dipilih penalaran *backward*.
b. Rata-rata jumlah *node* yang dapat diraih secara langsung dari suatu *node*. Lebih baik dipilih yang jumlah *node* tiap cabangnya lebih sedikit.
c. Apakah program perlu untuk menanyakan *user* untuk melakukan justifikasi terhadap proses penalaran? Jika ya, maka lebih baik jika dipilih arah yang lebih memudahkan *user*.
d. Bentuk kejadian yang akan memicu penyelesaian masalah. Jika kejadian berupa fakta baru, maka lebih baik dipilih penalaran *forward*. Namun, jika kejadian berupa *query*, maka lebih baik digunakan penalaran *backward*.

2.2.4.1 Penalaran Maju

Hal ini dimulai dengan membuat pohon (*tree*) yang memuat urutan gerakan yang mungkin merupakan solusi dengan cara mengawalinya dari keadaan awal di akar pohon. Kemudian, kembangkan *level* pohon dengan cara menggeser keadaan ke sebuah keadaan baru dengan mematuhi atau tanpa melanggar aturan-aturan yang telah ditetapkan. Akar merupakan keadaan awal, sedangkan daun-daun pohon merupakan tujuan yang akan dicapai[2]. Contoh ilustrasi dari penalaran *forward* dapat dilihat pada gambar 2.5.

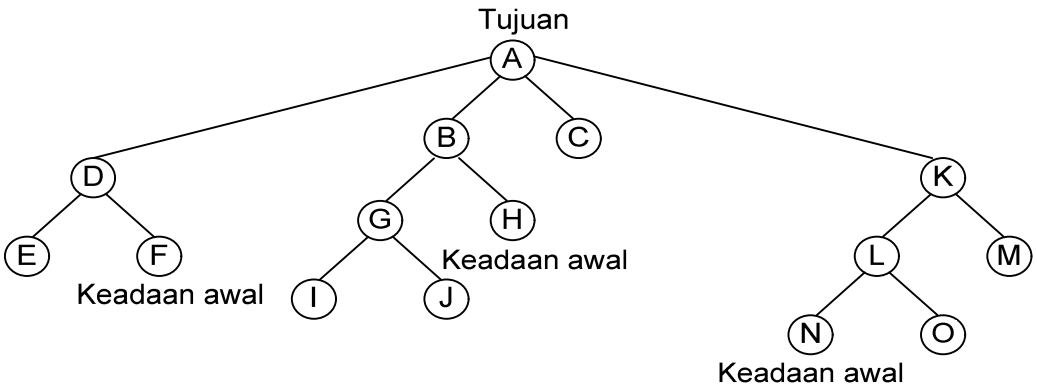
This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com



Gambar 2.5 Penalaran Maju

2.2.4.2 Penalaran Mundur

Hal ini dimulai dengan membuat pohon (*tree*) yang memuat urutan gerakan yang mungkin merupakan solusi dengan cara mengawalinya dari keadaan tujuan di akar pohon. Kemudian, kembangkan level pohon dengan cara menggeser keadaan ke sebuah keadaan baru dengan mematuhi atau tanpa melanggar aturan-aturan yang telah ditetapkan. Akar merupakan tujuan yang akan dicapai, sedangkan daun-daun pohon merupakan keadaan awal dari tujuan tersebut[2]. Contoh ilustrasi dari penalaran *backward* dapat dilihat pada gambar 2.6.



Gambar 2.6 Penalaran Mundur

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

2.3 Masalah Ruang Kadaan dan Pencarian

State adalah representasi suatu keadaan pada suatu saat ataupun deskripsi konfigurasi sistem. *State space* adalah semua *state* (keadaan) yang mungkin, dan biasanya digambarkan sebagai jaringan dengan *verteks* merupakan *state* dan *edge* merupakan perubahan yang mungkin.

Deskripsi formal suatu masalah adalah:

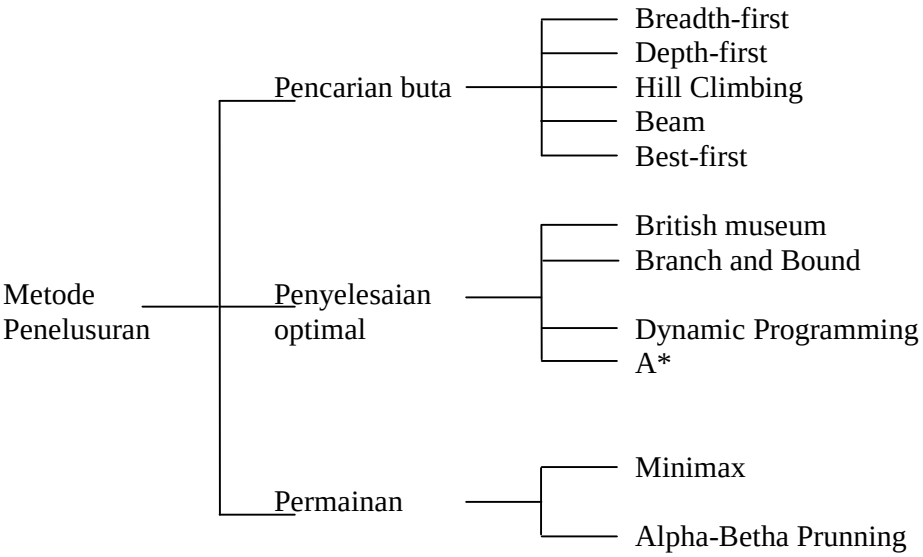
1. Definisikan *state space*.
2. Tentukan *initial state* dalam *goal state*.
3. Tentukan aturan-aturan.

Representasi *state space* memungkinkan definisi formal suatu masalah sebagai persoalan mengubah status dengan menggunakan sekumpulan aturan (*rule*) dan juga mendefinisikan masalah sebagai *search* yaitu mencari lintasan di dalam *state space* dari *initial state* ke *goal state*. Dalam hipotesis sistem simbol Newell dan Simon, yang menyatakan bahwa masalah-masalah dapat dipecahkan dengan melakukan pencarian (*search*) diantara pilihan-pilihan yang ada, terdukung oleh pandangan cara penyelesaian manusia yang bisa dilakukan. Manusia umumnya mempertimbangkan sejumlah alternatif strategi dalam menyelesaikan suatu masalah, dalam permainan catur misalnya, seorang pemain mempertimbangkan sejumlah kemungkinan tentang langkah-langkah berikutnya, memilih yang terbaik menurut kriteria tertentu seperti kemungkinan respon lawannya atau memilih sejumlah langkah menurut suatu strategi global yang dirancangnya. Seorang pemain catur juga memperhitungkan pencapaian jangka pendek (seperti strategi untuk memperoleh keuntungan materi atau kualitas)[6].

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

2.4 Metode Pencarian Melebar Pertama

Hal terpenting dalam menentukan keberhasilan sistem yang berdasarkan AI adalah kesuksesan dalam pencarian dan pencocokan. Permasalahan pencarian merupakan hal yang sering dijumpai oleh peneliti di bidang kecerdasan buatan. Permasalahan ini merupakan hal penting dalam menentukan keberhasilan sistem kecerdasan buatan. Dalam gambar 2.7 terdapat bagan untuk metode *searching*.



Gambar 2.7 Bagan metode pencarian (*searching*)

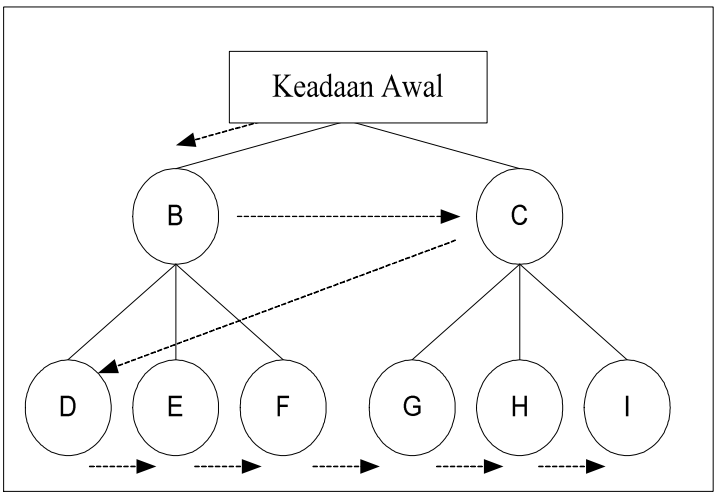
Pada dasarnya, terdapat dua teknik pencarian yang digunakan, yaitu:

1. Pencarian buta (*blind search*), dapat dibagi menjadi:
   a. Pencarian Melebar Pertama (*Breadth-First Search*)
   b. Pencarian Mendalam Pertama (*Depth-First Search*)
2. Pencarian Terbimbing (*Heuristic Search*)

Pada metode BFS, semua *node* pada level *n* akan dikunjungi terlebih dahulu sebelum mengunjungi *node-node* pada level *n*+1 (level berikutnya). Pencarian dimulai dari *node* akar terus ke level ke-1 dari kiri ke kanan, kemudian

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

berpindah ke level-2 dan seterusnya. Pohon dikembangkan hingga ditemukannya solusi.



Gambar 2.8 Prosedur Pencarian Melebar Pertama

Dalam algoritma BFS, *node* anak yang telah dikunjungi disimpan dalam suatu antrian. Antrian ini digunakan untuk mengacu *node-node* yang bertetangga dengannya yang akan dikunjungi kemudian sesuai urutan pengantrian. Untuk memperjelas cara kerja algoritma BFS beserta antrian yang digunakannya, berikut langkah-langkah algoritma BFS:

1. Masukkan *node* ujung (akar) ke dalam antrian.
2. Ambil *node* dari awal antrian, lalu cek apakah *node* merupakan solusi.
3. Jika *node* merupakan solusi, pencarian selesai dan hasil dikembalikan.
4. Jika *node* bukan solusi, masukkan seluruh *node* yang bertetangga dengan *node* tersebut (*node* anak) ke dalam antrian.
5. Jika antrian kosong dan setiap *node* sudah dicek, pencarian selesai dan mengembalikan hasil solusi tidak ditemukan.
6. Ulangi pencarian dari langkah kedua.

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com

Karena seluruh *node* yang telah dikunjungi harus disimpan, maka kompleksitas ruang algoritma BFS adalah O (jumlah *node* +jumlah sisi). Hal ini menunjukkan bahwa algoritma BFS memerlukan ruang besar dan tidak cocok untuk pemecahan masalah besar. Angka yang sama menunjukkan kompleksitas waktu terburuk, di mana setiap simpul harus diperiksa. Kompleksitas waktu terbaik, O(1), didapat jika solusi ditemukan pada pencarian pertama. Walaupun memiliki kompleksitas ruang dan waktu yang kurang baik, algoritma BFS dapat mencari solusi secara tuntas, berarti algoritma ini akan menemukan solusi terbaik dan terdekat bagaimanapun bentuk graf tersebut.

Keuntungan dari metode ini adalah tidak akan menemui jalan buntu dan Jika terdapat lebih dari satu solusi, maka BFS akan menemukannya.Kelemahan dari metode ini adalah membutuhkan memori yang cukup banyak karena menyimpan semua *node* dalam satu pohon dan membutuhkan waktu yang cukup lama, karena akan menguji semua node pada level-*n* untuk mendapatkan solusi pada level ke- (*n* + 1).

This document was created using SOLID CONVERTER PDF To remove this message, purchase the product at www.SolidDocuments.com