BAB II
LANDASAN TEORI

II.1 Kecerdasan Buatan

Kecerdasan buatan adalah suatu ilmu yang mempelajari cara membuat komputer komputer melakukan sesuatu yang dilakukan oleh manusia (Minsky, 1989). Definisi lain diungkapkan oleh H. A. Simon [1987]. Kecerdasan buatan (*artificial intelegence*) merupakan kawasan penelitian, aplikasi dan instruksi yang terkait dengan pemrograman komputer untuk melakukan sesuatu hal yang dalam pandangan manusia adalah cerdas.

Rich and Knight [1991] mendifinisikan kecerdasan buatan (AI) sebagai sebuah studi tentang bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia. Sementara ensiklopedi Britannica mendefinisikan kecerdasan buatan (AI) sebagai cabang dari ilmu komputer yang dalam merepresentasi pengetahuan lebih banyak menggunakan bentuk simbol-simbol daripada bilangan dan memproses informasi berdasarkan metode *heuristic* atau dengan berdasarkan sejumlah aturan.

Ada tiga tujuan kecerdasan buatan, yaitu membuat komputer lebih cerdas, mengerti tentan kecerdasan dan membuat mesin lebih berguna. Yang dimaksud kecerdasan adalah kemampuan untuk belajar atau mengerti dari pengalaman , memahami pesan yang kontradiktif dan ambigu, menganggapi dengan cepat dan baik atas situasi yang baru, menggunakan penalaran dalam memecahkan masalah serta menyelesaikan dengan efektif (Winston dan Prendergast, 1994).

Kecerdasan buatan berbeda dengan program konvensional. Pemrograman konvensional berbasis pada algoritma yang mendifinisikan setiap langkah dalam penyelesaian masalah. Permrograman konvensional dapat menggunakan rumus matematika atau prosedur sekuensial untuk menghasilkan solusi. Lain halnya dengan pemrograman kecerdasan buatan yang berbasis pada representasi simbol dan manipulasi. Dalam kecerdasan buatan yang berbasis pada representasu simbol

dapat berupa kalimat, kata atau angka yang digunakan untuk merepresentasikan obyek, proses dan hubungannya.

Beberapa bidang penelitian dalam kecerdasan buatan diantaranya adalah:

1. *Formal tasks* (matematika, *games*).
2. *Mundane task* (*perception, robotics, natural language, common sense, reasoning*).
3. *Expert task* (*financial analysis, medical duagnotics, engineering, scientific analysis,* dan lain-lain).

Ada beberapa konsep yang harus dipahami dalam kecerdasan buatan, diantaranya:

1. *Turing Test* – Metode Pengujian Kecerdasan

   Turing Test meruakan sebuah metode pengujian kecerdasan yang dibuat oleh Alan Turing. Proses uji ini melibatkan penanya (manusia) dan dua obyek yang ditanyai. Yang satu adalah seorang manusia dan satunya adalah sebuah mesin yang akan diuji. Penanya tidak dapat melihat langsung kepada obyek yang ditanyai. Penanya diminta untuk membedakan mana jawaban komputer dan mana jawaban manusia berdasarkan jawaban kedua obyek tersebut. Jika penanya tidak dapat membedakan mana jawaban mesin dan mana jawaban manusia maka Turing bependapat bahwa mesin yang diuji tersebut dapat diasumsikan cerdas.

2. Pemrosesan Simbolik

   Komputer semula didesain untuk memproses bilangan atau angka-angka (pemrosesan numerik). Sementara manusia dalam berfikir dan menyelesaikan masalah lebih bersifat simbolik, tidak didasarkan pada sejumlah rumus atau melakukan komputasi matematis. Sifat penting dari AI adalah bahwa AI merupakan bagian dari ilmu komputer yang

melakukan proses secara simbolik dan non-algoritmik dalam penyelesaian masalah.

3. *Heuristic*

   Istilah *heuristic* diambil dari bahasa Yunani yang berarti menemukan. *Heruristic* merupajan suatu strategi untuk melakukan proses pencarian (*search*) ruang problem secara selektif, yang memandu proses pencarian yang kita lakukan sepanjang jalur yang memiliki kemungkinan sukses yang paling besar.

4. Penarikan Kesimpulan (*Inferencing*)

   AI mencoba membuat mesin memiliki kemampuan berpikir atau mempertimbangkan (*reasoning*). Kemampuan berpikir (*reasoning*) termasuk didalamnya proses penarikan kesimpulan (*inferencing*) berdasarkan fakta-fakta dan aturan dengan menggunakan metode heuristik atau metode pencarian lainnya.

5. Pencocokan Pola (*Pattern Matching*)

   AI bekerja dengan metode pencocokan pola (*pattern matching*) yang berusaha untuk menjelaskan obyek, kejadian (*events*) atau proses, dalam hubungan logik atau komputasional **[7]**.

II.2 Algoritma

Algoritma adalah langkah-langkah untuk memecahkan suatu masalah. Terdapat beberapa definisi lain dari algoritma tetapi senada dengan definisi yang diungkapkan diatas yang dikutip dari beberapa literatur, antara lain:

1. Algoritma adalah deretan langkah-langkah komputasi yang mentransormasikan data masukan menjadi keluaran.
2. Algoritma adalah deretan instruksi yang jelas untuk memecahkan masalah, yaitu untuk memperoleh keluaran yang diinginkan dari suatu masukan dalam jumlah waktu yang terbatas.

3. Algoritma adalah prosedur komputasi yang terdefinisi dengan baik yang - menggunakan beberapa nilai sebagai masukan dan menghasilkan beberapa nilai yang disebut keluaran. Jadi, algoritma adalah langkah komputasi yang mentransformasikan masukan menjadi keluaran **[8]**.

II.3 Algoritma Pencarian

Algoritma pencarian adalah sebuah algoritma yang menerima masukan berupa sebuah masalah dan menghasilkan sebuah solusi untuk masalah tersebut, yang biasanya didapat dari evaluasi beberapa kemungkinan solusi. Pada algoritma pencarian, terdapat beberapa jenis, seperti pencarian *informed*, pencarian adversarial, pemenuhan kendala, pencarian interpolasi dan *uninformed.*

Uninformed search, sering juga disebut pencarian buta (*blind search*) atau pencarian naïf (*naïve search*) adalah algoritma yang tidak mempertimbangkan sifat alami dari permasalahan. Oleh karena itu algoritma tersebut dapat diimplementasikan secara umum, sehingga dengan implementasi yang sama dapat digunakan pada lingkup permasalahan yang luas. Pada *uninformed search,* terdapat pencarian list, pencarian graf dan pencarian pohon **[9]**.

II.4 Algoritma Pencarian Beam

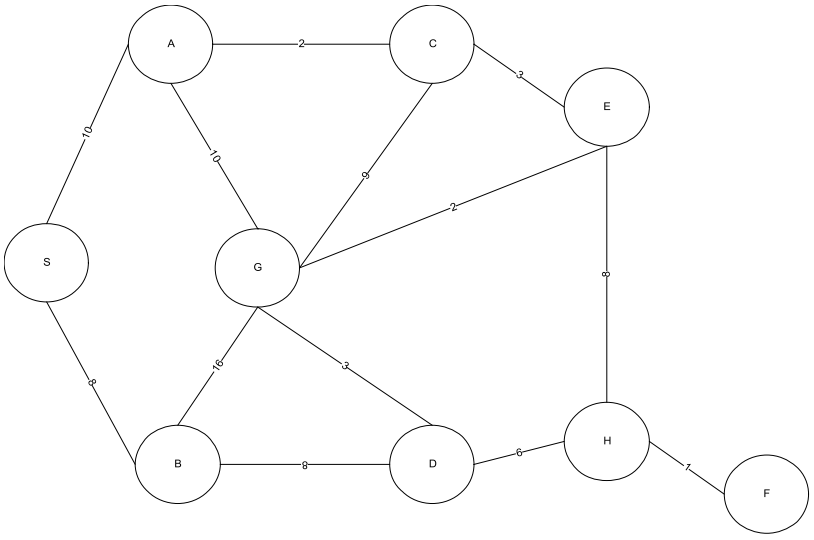
Algoritma pencarian beam adalah suatu metode heuristik untuk menyelesaikan masalah optimasi kombinatorial dan algoritma pencarian beam merupakan optimasi algoritma *best-first search* untuk mengurangi kebutuhan memori. Algoritma ini hanya menyimpan nilai terbaik saja, sedangkan nilai yang dianggap buruk langsung dihapus dan karena metode ini memang dirancang untuk pencarian yang cepat, meskipun sebenarnya simpul yang dihapus dapat menghasilkan solusi yang optimal.

Algoritma pencarian beam tidak mengizinkan pencarian kembali ke simpul sebelumnya karena pada algoritma ini, simpul yang dianggap buruk dan telah di

hapus, tidak dapat dibangkitkan kembali bila dibutuhkan untuk perhitungan selanjutnya. Algoritma pencarian beam membutuhkan nilai-nilai dalam melakukan pencariannya, yaitu nilai beam. Nilai beam digunakan untuk menentukan besar senarai yang akan digunakan untuk menyimpan simpul yang dikunjungi. Pada algoritma pencarian beam, simpul terbaik yang disimpan adalah sebanyak nilai beamnya. Fungsi evaluasi heuristik pada algoritma pencarian beam adalah $f(n) = h(n)$ dimana *h(n)* adalah nilai estimasi jarak menuju tujuan. Sebagai contoh, terdapat dua simpul yaitu simpul A dan B yang mana simpul tersebut memiliki busur sebesar 10. Berdasarkan fungsi evaluasi heuristiknya, maka nilai *f(n)* adalah 10.

Bila dilihat dari sisi kecerdasan buatan, algoritma pencarian beam merupakan salah satu dari agen tunggal untuk memecahkan suatu masalah karena algoritma ini termasuk algoritma dengan tipe pemangkasan kedepan **[10]**.

Contoh sederhana dari algoritma pencarian beam misalkan simpul awal pada simpul S dan simpul tujuan adalah simpul F. Nilai beam yang digunakan dalam contoh kasus ini adalah dua.

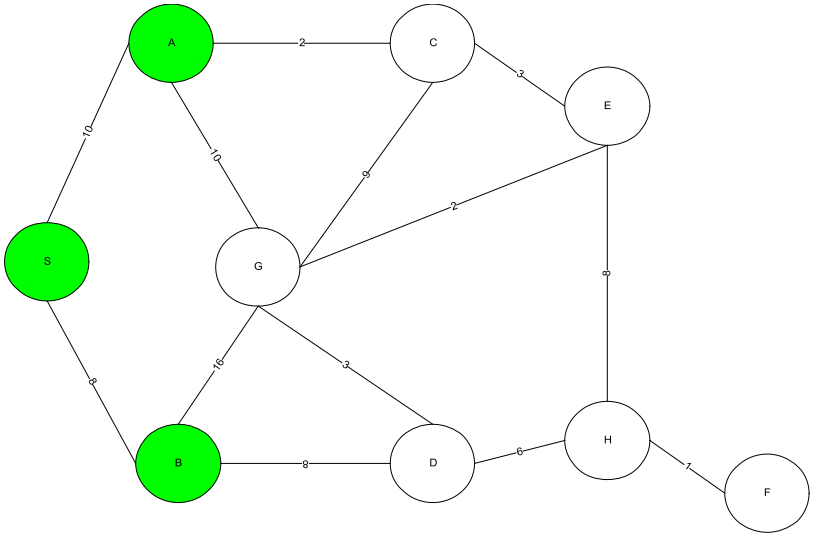


Gambar II-1 Graph Pencarian

Pada tahap pertama, kita tentukan besar nilai beam yang akan digunakan. Untuk contoh disini, besar nilai beam yang digunakan adalah dua. Setelah itu, periksa seluruh simpul tetangga dari simpul awal. Ambil simpul tetangga terbaik dari simpul awal sebesar nilai beam yang mana nilai beam disini adalah sebesar dua dan masukan simpul tersebut kedalam antrian dengan kondisi simpul memiliki nilai heuristik terkecil.

Tabel II-1 Kondisi Pada Penelusuran Pertama

Simpul Yang Diperiksa	Urutan Simpul
S	{S}
	{S,A},{S,B}

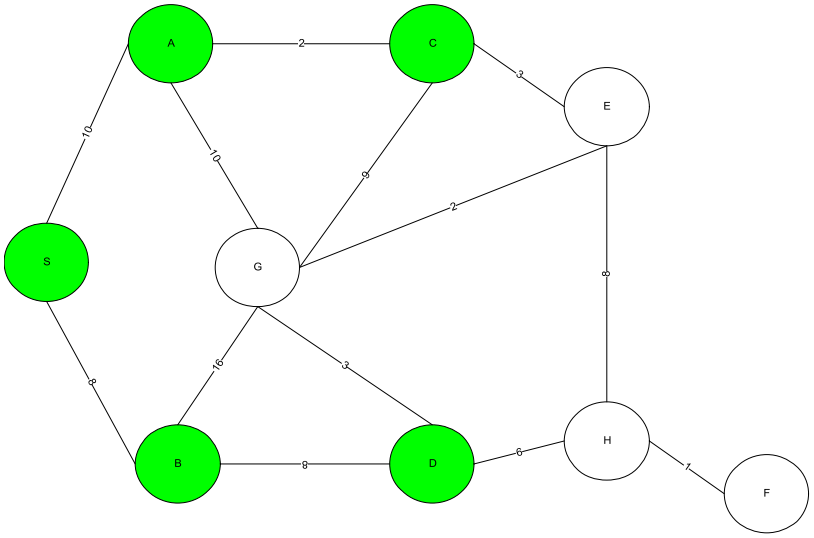


Gambar II-2 Penelusuran Pertama

Setelah penelusuran dari simpul awal didapatkan dua simpul tebaik, penelusuran simpul-simpul dilanjutkan dengan memeriksa simpul tetangga dari masing-masing simpul terbaik tersebut yaitu simpul A dan B. Sama seperti langkah pertama, ambil dua simpul terbaik dan masukan kedalam antrian.

Tabel II-2 Kondisi Pada Penelusuran Ke Dua

Simpul Yang Diperiksa	Urutan Simpul
Periksa Node Tetangga A dan B	{S,A},{S,B}
	{S,A,C},{S,B,D}

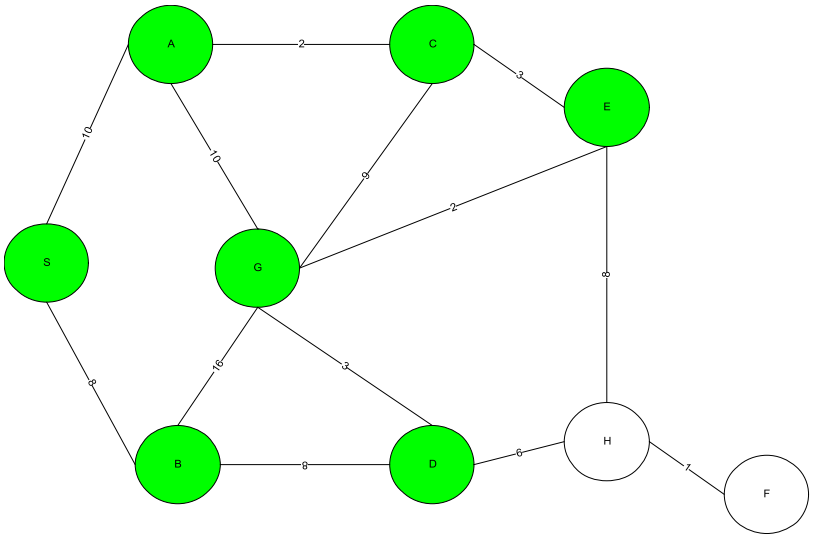


Gambar II-3 Penulusuran Ke Dua

Penelusuran dilanjutkan dengan menelusuri simpul-simpul tetangga dari simpul C dan D. Setelah itu, cari simpul tetangga terbaik dari simpul C dan D dan masukan simpul ke dalam antrian.

Tabel II-3 Kondisi Pada Penelusuran Ke Tiga

Simpul Yang Diperiksa	Urutan Simpul
C dan D	{S,A,C},{S,B,D}
	{S,A,C,E},{S,B,D,G}

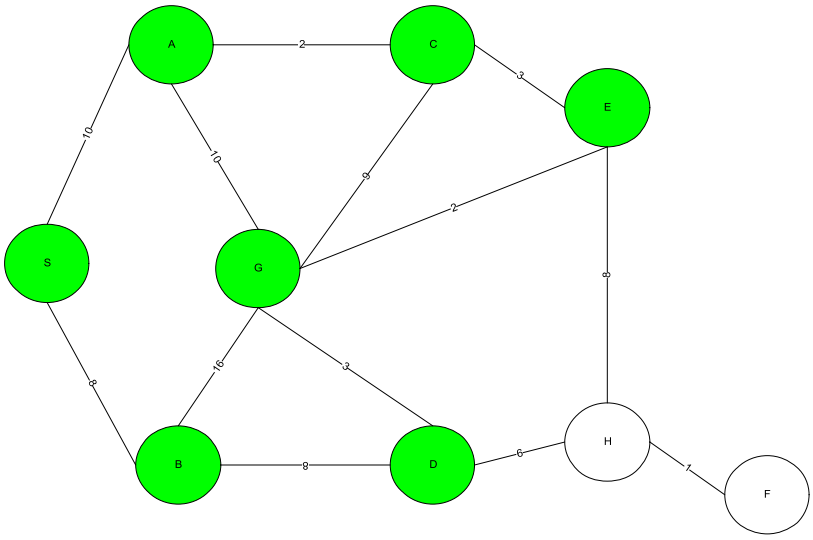


Gambar II-4 Penulusuran Ke Tiga

Penelusuran dilanjutkan dengan memeriksa simpul dari simpul terbaik yang didapatkan sebelumnya yang mana simpul tersebut adalah simpul E dan G. Langkah pencariannya sama seperti langkah sebelumnya dan masukan simpul terbaik dari hasil penelusuran simpul E dan G ke dalam antrian.

Tabel II-4 Kondisi Pada Penelusuran Ke Empat

Simpul Yang Diperiksa	Urutan Simpul
E dan G	{S,A,C,E},{S,B,D,G}
	{S,A,C,E,G},{S,B,D,G,E}

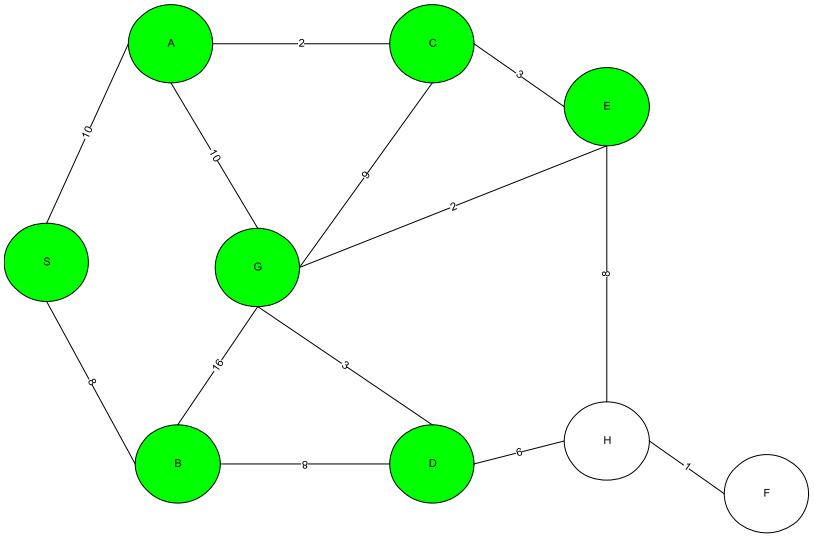


Gambar II-5 Penulusuran Ke Empat

Penelusuran dilanjutkan dengan langkah pencarian simpul sama seperti pencarian sebelumnya dan simpan simpul terbaik dari hasil penelusuran simpul G dan E.

Tabel II-5 Kondisi Pada Penelusuran Ke Lima

Simpul Yang Diperiksa	Urutan Simpul
G dan E	{S,A,C,E,G},{S,B,D,G,E}
	{S,A,C,E,G,D},{S,B,D,G,E,C}

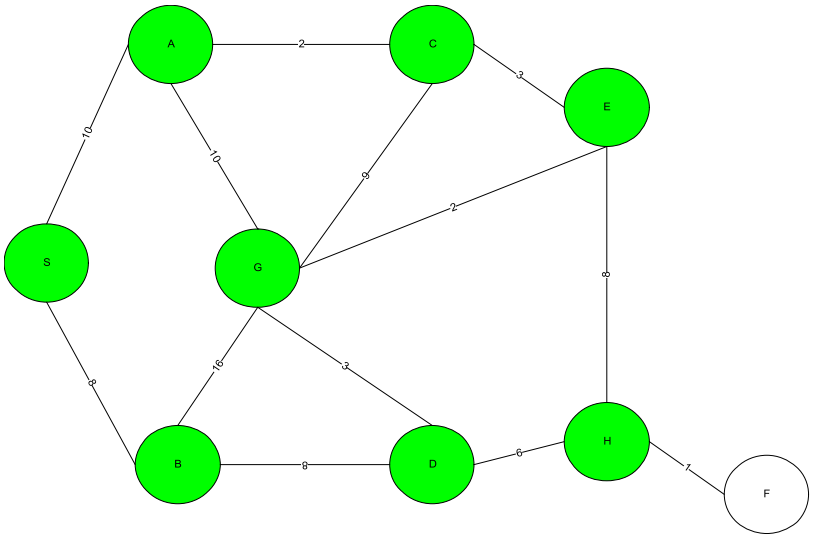


Gambar II-6 Penulusuran Ke Lima

Penelusuran dilanjutkan dengan langkah pencarian simpul sama seperti pencarian sebelumnya yang mana pada kondisi ini, simpul yang diperiksa adalah simpul simpul D dan C.

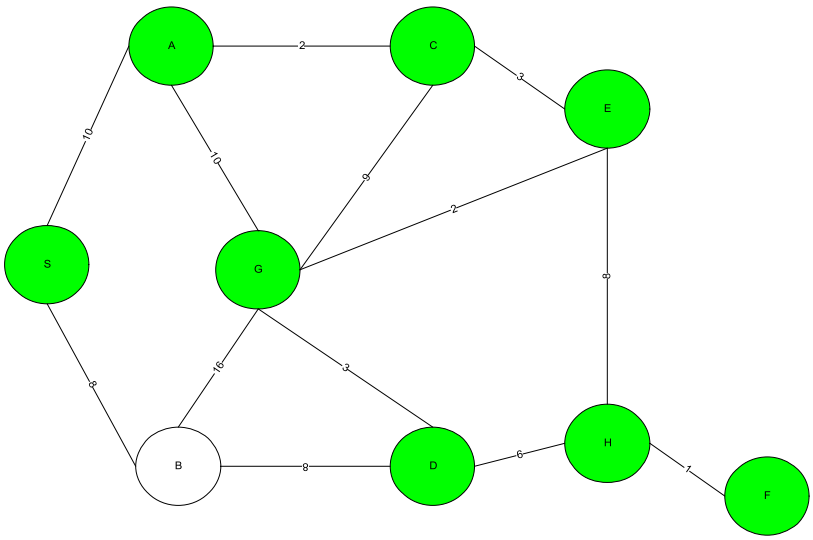
Tabel II-6 Kondisi Pada Penelusuran Ke Enam

Simpul Yang Diperiksa	Urutan Simpul
D dan C	{S,A,C,E,G,D},{S,B,D,G,E,C}
	{S,B,D,G,E,C,A}, {S,A,C,E,G,D,H}



Gambar II-7 Penulusuran Ke Enam

Penelusuran dilanjutkan dengan memeriksa simpul A dan H. Pada saat penelusuran simpul H, solusi simpul tujuan telah ditemukan oleh karena itu, pencarian selesai dengan menghasilkan solusi tujuan dengan solusi S-A-C-E-G-D-H-F.



Gambar II-8 Tujuan Ditemukan

II.5 Struktur Data

Struktur data adalah suatu kumpulan objek-objek data yang memiliki aturan (terorganisir) yang kemudian digolongkan berdasarkan operasi-operasi untuk memanipulasinya. Kompponen yang dimuat daam struktur data ini dapat berupa objek data elementer atau objek terstruktur lainnya. Contoh objek data terstruktur adalah *array*, *record*, *list*, *stack*/tumpukan, *quwuw*/antrian, *tree*/pohon, set/himpunan, file/berkas dan lain-lain.

Atribut-atribut penting untuk suatu tipe data terstruktur meliputi:

1. Jumlah komponen

   Berdasarkan perubahan umlah komponen selama eksekusi program, maka dapat dikelompokan menjadi struktur data statis yang jumlah

komponennya tidak berubah dan struktur data dinamis yang jumlah komponennya dapat berubah.

2. Tipe untuk setiap komponen

   Apabila tipe data untuk seluruh komponennya harus sama, maka disebut struktur data homogen, dan bila dimungkinkan komponennya mempunyai tipe yang berbeda-beda, maka disebut struktur data heterogen.

3. Nama-nama untuk memilih komponen

   Hampir semua struktur data menyediakan operasi untuk mengakses komponen secara individu. Pada suatu array, hal ini dilakukan dengan sebuag subscript/indeks berupa angka.

4. Jumlah maksimum komponen

   Untuk sebuah struktur data dinamis mungkin perlu ditentukan dengan jelas.

5. Pengorganisasian semua komponennya.

   Susunan yang paling umum adalah berupa barisan linier seperti pada *array* berdimensi 1, *record*, *list*, *stack* dan *file*. Selanjutnya ada yang dapat dikembangkan menjadi struktur non linear seperti *array* multi dimensi dan juga pohon/*tree.*

Operasi-operasi pada struktur data meliputi:

1. Operasi seleksi komponen

   Operasi seleksi komponen adalah operasi untuk mengakses komponen struktur data dan membuatnya tersedia untuk pemrosesan dengan operasi lain. Operasi ini dapat dibedakan menjadi operasi akses secara random dimana struktur data yang diakses dapat berganti-ganti dan operasi akses secara berurutan/sekuensial dimana komponen-komponen diakses satu demi satu dalam urutan atau rangkaian pemrosesan.

2. Operasi struktur data secara keseluruhan
   Operasi ini memungkinkan struktur data secara keseluruhan sebagai argumenya dan menghasilkan struktur data yang baru.

3. Penyisipan dan penghapusan komponen
   Operasi ini akan mengubah jumlah komponen dan berpengaruh pada representasi dan pengelolaan penyimpanan.

4. Pembuatan dan penghapusan struktur data **[11]**.

II.6 Rute Terpendek

Masalah rute terpendek dapat didefinisikan sebagai graf, baik tidak berarah, berarah atau campuran. Masalah ini juga terkadang disebut *single-pair shortest path problem*, dan dibedakan sebagai berikut:

1. *Single-source shortest path problem*, di mana kita harus menemukan jalan terpendek dari sumber titik v untuk semua simpul lain dalam grafik.
2. *Single-destination shortest path problem*, di mana kita harus menemukan jalur terpendek dari semua simpul dalam grafik diarahkan ke tujuan simpul tunggal v. Masalah ini dapat direduksi menjadi *single-source shortest path problem* dengan membalik busur dalam grafik terarah.
3. *All-pairs shortest path problem*, di mana kita harus menemukan jalur terpendek antara setiap pasangan dari simpul v, v' dalam grafik **[12]**.

II.7 Peta

Peta adalah bentuk muka bumi yang digambarkan dalam bidang datar dan diperkecil melalui sistem proyeksi peta. Penggambaran peta didasarkan pada aturan-aturan dan teknik tertentu. Aturan dan teknik tertentu itu disebut kartografi, yaitu seni, ilmu, dan teknik pembuatan peta yang melibatkan ilmu geodesi,

fotogrametri (pembuatan peta dengan teknik fotografi), kompilasi peta dan reproduksi peta.

Pada dasarnya peta berfungsi sebagai alat komunikasi. Peta menginformasikan kondisi suatu wilayah secara sederhana sehingga menjadi mudah dipahami. Peta juga merupakan alat bantu untuk mempelajari suatu wilayah tanpa perlu mendatangi tempat tersebut.

Lebih lanjut, fungsi peta adalah sebagai berikut.

1. Untuk menggambarkan atau memperlihatkan bentuk muka bumi.
2. Untuk menunjukan posisi dan letak
3. Untuk menunjukan jarak dan luas
4. Sebagai media informasi untuk berbagai keperluan

Peta tebagi berdasarkan jenis data, skala peta dan bentuk peta. Berdasarkan jenis data, peta terbagi atas peta tematik yaitu peta yang menggambarkan unsur/kenampakan, baik yang bersifat kualitatif dan kuantitatif, dan berhubungan dengan tema tertentu saja, peta topografi yaitu peta yang menggambarkan unsur-unsur topografi di suatu wilayah dan memuat informasi relief (tinggu rendahnya permukaan buni) yang dihubungkan dengan garis kontur.

Berdasarkan sumber data, peta terdiri dari peta dasar, yaitu peta yang menjadi acuan dalam pembuatan peta-peta tematik, peta topografi, peta turunan. Peta turunan yaitu peta yang diturunkan dari peta induk dengan skala lebih kecil dan umumnya telah mengalami proses pemilahan dan penyederhanaan informasi.

Berdasarkan skala peta, peta terdiri dari peta skala besar (1:10.000, 1:2.000, 1:500), peta skala sedang (1:25.000, 1:50.000, 1:100.000) dan peta skala kecil (1:500.000, 1:1.000.000).

Dan berdasarkan bentuknya, peta terdiri dari peta analog, peta streometri dan peta digital yaitu peta yang diolah dengan bantuan komputer dan disimpan dalam data digital **[13]**.

II.8 Tools Pemodelan Perangkat Lunak

II.8.1 UML (Unified Modeling Language)

UML singkatan dari *Unified Modeling Language* yang berarti bahasa pemodelan standar. Ketika kita membuat model menggunakan konsep *UML* ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita bua berhubungan satu dengan lainnya harus mengikuti standar yang ada. *UML* bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk:

1. Merancang perangkat lunak.
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara terinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

Beberapa literatur menyebutkan bahwa *UML* menyediakan sembilan jenis diagram, yang lain menyebutkan delapan. Namun demikian model-model itu dapat dikelompokan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain:

1. Diagram kelas. Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi serta relasi-relasi.
2. Diagram paket. Bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas, merupakan bagian dari diagram komponen.

3. Diagram *Use-Case.* Bersifat statis. Diagram ini memperlihatkan himpunan *use-case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini sangat penting terutama untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.
4. Diagram interaksi dan *Sequence* (urutan). Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.
5. Diagram komunikasi. Bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi UML 1.4 yang menekankan organisasi struktural daru objek-objek yang menerima serta mengirimkan pesan.
6. Diagram *suksesorchart.* Bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status, transisi, kejadian serta aktifitas.
7. Diagram aktivitas. Bersifat dinamis. Diagram aktivitas ada;ah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem.
8. Diagram komponen. Bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen – komponen yang telah ada sebelumnya.
9. Diagram *deployment*. Bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run time*) **[14]**.

II.8.2 DFD

Data Flow Diagram (DFD) menggambarkan aliran data atau informasi dimana di dalamnya terlihat keterkaitan di antara data-data yang ada. Terdapat banyak simbol-simbol yang digunakan dalam pembuatan DFD. Hal tersebut tergantung konvensi yang disepakati. DFD merupakan salah satu alat analisis dan teknik pemodelan terbaik untuk menggambarkan proses dan kebutuhan fungsional dari suatu sistem.

DFD merupakan serangkaian diagram yang menggambarkan kegiatan-kegiatan yang ada dalam suatu sistem. Teknik pembuatan DFD dimulai dengan menggambarkan sistem secara global dan dilanjutkan dengan analisis masing-masing bagian. Pada awalnya, digambarkan konteks diagram yang menggambarkan sebuah sistem secara menyeluruh yang akan diinvestigasi. Konteks diagram tersebut dapat dikatakan sebagai DFD level 0. Analisis sistem yang lebih detail selanjutnya dapat dilakukan dengan menggambarkan DFD level 1, 2 dan seterusnya. Kebanyakan investigasi akan dihentikan pada DFD level 2. *Data Flow Diagram* (DFD) secara umum harus dapat dimengerti oleh programer karena akan menjadi panduan untuk pembuatan program.

Untuk memperoleh gambaran logika dari sistem yang diinginkan dapat digambarkan dalam *logical model*. *Logical model* dari sistem informasi lebih menjelaskan kepada pengguna bagaimana nantinya fungsi-fungsi pada sistem informasi secara logika akan bekerja. *Logical model* secara keseluruhan dan lebih rinci tentang proses, *input*, *output* dan *storage* **[15]**.

II.9 Tools Implementasi

II.9.1 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencangkup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat piranti lunak untuk ponsel/*smartphone.* Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsiun dari 34 perusahaan piranti keras, piranti lunak dan telekomunikasi termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile dan Nvidia.

Sistem operasi yang mendasari Android dilisensikan di bawah GNU, Genereal Public Lisensi Versi 2 (GPLv2), yang sering dikenal dengan istilah “copyleft” lisensi dimana setiap perbaikan pihak ketiga harus terus jatuh dibawah

terms. Android didistribusikan dibawah lisensi Apache *Software* (ASL/Apache2), yang memungkinkan untuk distribusi kedua dan seterusnya. Pengembang aplikasi Android diperbolehkan untuk mendistribusikan aplikasi mereka dibawah skema lisensi apapun yang mereka inginkan **[16]**.

II.9.2 Eclipse

Eclipse adalah IDE yang paling populer untuk pengembangan Android karena memiliki *plug-in* yang tersedia untuk memfasilitasi pengembangan Android. Selain itu, *Eclipse* juga mendapat dukungan langsung dari Google untuk menjadi IDE pengembangan aplikasi Android, ini terbukti dengan adanya penambahan *plugins* untuk *Eclipse* untuk membuat projek Android dimana *source software* langsung dari situs resmi Google **[16]**.

II.9.3 Web Service

Web service adalah suatu sistem perangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antar sistem pada suatu jaringan. Web service digunakan sebagai suatu fasilitas yang disediakan oleh suatu web site untuk menyediakan layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan-layanan (service) yang disediakan oleh suatu sistem yang menyediakan web service. Web service menyimpan data informasi dalam format XML, sehingga data ini dapat diakses oleh sistem lain walaupun berbeda platform, sistem operasi, maupun bahasa *compiler*.

Web service memiliki tiga entitas dalam arsitekturnya, yaitu:

1. *Service Requester* (peminta layanan). Peminta layanan yang mencari dan menemukan layanan yang dibutuhkan serta menggunakan layanan tersebut.

2. *Service Provider* (penyedia layanan). Berfungsi untuk menyediakan layanan/service dan mengolah sebuah registry agar layanan-layanan tersebut dapat tersedia.
3. *Service Registry* (daftar layanan). Berfungsi sebagai lokasi central yang mendeskripsikan semua layanan/service yang telah di-register **[17]**.

II.9.4 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau yang dikenal dengan DBMS (*database management system*), database ini *multithread*, *multi-user*. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus yang bersifat khusus.

MySQL didistribusikan secara gratis dibawah lisensi GPL (General Public License). Di mana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat closed source atau komesial. MySQL sebenarnya merupakan salah satu konsep utama dalam database sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian database, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan penoperasian data dikerjakan dengan mudah secara otomatis.

Sebagai databse server, MySQL dapat dikatakan lebih unggul dibandingkan database server lainnya, terutama dalam kecepatan. Berikut ini beberapa keistimewaan MySQL, antara lain:

1. Portability

   MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, Free BSD, Mac Os X server, Solaris, Amiga dan masih banyak lagi.

2. Multiuser

   MySQL dapat digunakan oleh beberapa user dalam waktu yang bersamaan tanpa menglami masalah atau konflik.

3. Security

   MySQL memiliki beberapa lapisan sekuritas seperti level subnetmask, nama host, dan izin akses user dengan sistem perizinan yang mendetail serta password terenkripsi.

4. Scalability dan limits

   MySQL mampu menangani database dalam skala besar, jumlah *record* lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu, batas indeks yang dapat ditampung mencapai 32 indeks pada tiap tabelnya **[18]**.