BAB II

LANDASAN TEORI

2.1 Kompresi Data

Sistem komunikasi dirancang untuk mentransmisi informasi yang dibangkitkan oleh sumber ke beberapa tujuan. Sumber informasi mempunyai beberapa bentuk yang berbeda. Sebagai contoh, dalam radio *broadcasting*, sumber biasanya sumber audio (suara atau music). dalam TV *broadcasting*, sumber informasi biasanya sebuah sumber video yang keluarannya berupa *image* bergerak. Output dari sumber-sumber ini adalah sinyal analog dan sumbernya disebut sumber analog.

Kontras dengan komputer dan tempat penyimpanan data (*storage*) seperti disk magnetic atau optical, menghasilkaan ouput berupa sinyal diskrit (biasanya karakter binary atau ASCII). Sinyal yang menjadi sumbernya biasa disebut dengan sumber diskrit. baik sumber analog maupun diskrit, sebuah komunikasi digital dirancang untuk mentransmisi informasi dalam bentuk digital. Sehingga konsenkuensi keluaran dari sumber harus diubah dahulu menjadi bentuk keluaran sumber digital yang biasanya dilakukan pada *source encoder*. Keluaran dari sumber tersebut dapat diasumsikan menjadi sebuah digit biner sekuensial.

Pada akhir tahun 40-an dimana dimulainya tahun teori informasi, ide pengembangan metode coding yang efesien baru dimulai dan dikembangkan. Dimulainya penjelajahan ide dari *entropy*, *information content* dan *redudansi*. Salah satu ide yang popular adalah apabila probabilitas dari simbol dalam suatu pesan diketahui, maka terdapat cara untuk mengkodekan simbol, sehingga pesan memakan tempat yang lebih kecil. Ide inilah yang menjadi cikal bakal dalam terciptanya kompresi data.

Model pertama yang muncul untuk kompresi sinyal digital adalah Shannon-Fano coding. Shannon dan fano (1948) terus menerus mengembangkan algoritma ini yang menghasilkan codeord biner untuk setiap symbol (unik) yang terdapat pada data file.

Huffman coding (1952) memakai hampir semua karakteristik dari *Shannon-fano coding*. Huffman coding dapat menghasilkan kompresi data yang efektif dengan mengurangkan jumlah redudansi dalam mengkodingkan simbol. Telah dapat dibuktikan, bahwa Huffman coding merupakan metode *fixed-length* yang paling efesien.

Pada limabelas tahun terakhir, Huffman coding telah digantikan oleh *Arithmetic coding*. *Arithmetic coding* melewatkan ide untuk menggantikan sebuah simbol masukan dengan kode yang spesifik. Algoritma ini menggantikan sebuah aliran simbol masukan dengan sebuah angka keluaran *single floating-point*. Lebih banyak bit dibutuhkan dalam angka keluaran, maka semakin rumit pesan yang diterima.

Algoritma dictionary-based compression menggunakan metode yang sangat berbeda dalam mengkompres data. Algoritma ini menggantikan string variable-length dari simbol menjadi sebuah *token.* Token merupakan sebuah indek dalam susunan kata di kamus. Apabila token kecil dari susunan kata, maka token akan menggantikan prase tersebut dan kompresi pun terjadi.

Kompresi data (pemampatan data) merupakan suatu teknik untuk memperkecil jumlah ukuran data (hasil kompresi) dari data aslinya. Pemampatan data umumnya diterapkan pada mesin komputer, hal ini dilakukan karena setiap simbol yang muncul pada komputer memiliki nilai bit-bit yang berbeda. Misal pada ASCII setiap simbol yang dimunculkan memiliki panjang 8 bit, misal kode A pada ASCII mempunyai nilai decimal 65, jika dirubah dalam bilangan biner menjadi 010000001. Pemampatan data digunakan untuk mengurangkan jumlah bir-bit yang dihasilkan dari setiap simbol yang muncul. Dengan pemampatan ini diharapkan dapat mengurangi (memperkecil ukuran data) dalam ruang penyimpanan.

2.1.1 Faktor Penting Kompresi Data

Dalam kompresi data, terdapat 4 (empat) faktor penting yang perlu diperhatikan, yaitu :

1. *Time process*

   Yaitu waktu yang dibutuhkan untuk menjalankan proses kompresi.

2. *Completeness*

   Yaitu kelengkapan data setelah data tersebut dikompresi.

3. *Ratio Compress*

   Yaitu ukuran data setelah data tersebut dikompresi

4. *Optimaly*

   Yaitu perbandingan ukuran data sebelum dikompresi dengan data yang telah dikompresi.

2.1.2 Jenis Kompresi Data

Berdasarkan mode penerimaan data oleh manusia, kompresi data dapat dibagi menjadi dua, yaitu :

a. *Dialoque Mode*

   yaitu proses penerimaan data dimana pengirim dan penerima seakan berdialog (*real time*), dimana kompresi data harus berada dalam batas penglihatan dan pendengaran mausia. Contohnya pada *video conference*.

b. *Retrieval Mode*

   yaitu proses penerimaan data tidak dilakukan secara real time. Jenis kompresi ini dapat dilakukan *random access* terhadap data dan dapat bersifat interaktif.

2.1.3 Metode Kompresi Data

Berdasarkan tipe peta kode yang digunakan untuk mengubah pesan awal (isi file input) menjadi sekumpulan *codeword*, metode kompresi terbagi menjadi dua kelompok, yaitu :

a. Metode Statik

   Menggunakan peta kode yang selalu sama. Metode ini membutuhkan dua fase (*two-pass*) : fase pertama untuk menghitung probabilitas kemunculan tiap simbol/karakter dan menentukan peta kodenya dan fase kedua untuk mengubah pesan menjadi kumpulan kode yang akan ditransmisikan. Contohnya pada *Huffman static*, *arithmetic coding*

b. Metode Dinamik (adaptif)

   Menggunakan peta kode yang dapat diubah dari waktu ke waktu. Metode ini disebut adaptif karena peta kode mampu beradaptasi terhadap karakteristik isi file selama proses kompresi berlangsung. metode ini bersifat *onepass*, karena isi file selama dikompres hanya diperlakukan satu kali pembacaan terhadap isi file. Contohnya pada algoritma LZW dan DMC.

Berdasarkan teknik pengkodean atau pengubahan simbol yang digunakan, metode kompresi dapat dibagi ke dalam tiga kategori, yaitu :

a. *Metode simbolwise*

   Menghitung peluang kemunculan dari tiap simbol dalam file input, lalu mengkodekan satu simbol dalam satu waktu, dimana simbol yang lebih sering muncul diberi kode lebih pendek dibandinglan simbol yang lebih jarang muncul. Contohnya pada Huffman coding, arithmetic coding.

b. *Metode dictionary*

   Menggantikan karakter/fragmen dalam file input dengan indeks lokasi dari karakter/fragmen tersebut dalam sebuah kamus (*dictionary*). Contohnya pada algoritma LZW

c. *Metode predictive*

   Menggunakan model *finite-context* atau *finite-state* untuk memprediksi distribusi probabilitas dari simbol-simbol selanjutnya. Contohnya pada algoritma DMC

2.1.4 Teknik Kompresi Data

Teknik kompresi data dapat digolongkan menjadi dua kelompok utama yaitu : lossy dan lossless. Teknik kompresi secara lossy dimaksudkan dengan teknik kompresi data dengan menghilangkan ketelitian data utama guna mendapatkan data sekecil mungkin (kompresi data sebesar mengkin). Teknik kompresi data secara lossless yaitu teknik kompresi data dengan mengurangkan jumlah data yang terjadi redudansi (memiliki symbol yang sama) sebelum terjadi kompresi.

a. *Lossy Compression*

   Merupakan teknik kompresi yang menghilangkan beberapa informasi data yang dianggap tidak penting. Sehingga hasil data yang telah terkompresi tidak sama dengan data yang sebelum dikompresi. Namun data yang telah terkompresi tersebut sudah cukup untuk digunakan, walaupun datanya telah berubah. Keunggulan dari teknik ini adalah file atau data yang telah terkompresi memiliki ukuran yang lebih kecil dari ukuran data aslinya. Biasanya teknik kompresi ini banyak diaplikasikan pada data gambar dan data audio.

b. *Lossless Compression*

   Merupakan teknik kompresi yang mempertahankan kebutuhan informasi yang dikandung oleh data, sehingga informasi yang terkandung pada file yang telah terkompresi tetap terjaga meskipun ukurannya telah berubah dari ukuran data aslinya. Keunggulan dari teknik ini adalah data yang telah terkompresi, apabila didekompresi kembali akan menghasilkan data yang

sama persis dengan data aslinya. Biasanya teknik kompresi ini banyak diaplikasikan pada data teks.

2.1.5 Klasifikasi Teknik Kompresi

Teknik kompresi dapat diklasifikasikan menjadi tiga kelompok, antara lain :

a. *Entropy Encoding*

Merupakan klasifikasi teknik kompresi yang bersifat *loseless compression.* Tekniknya tidak berdasarkan media dengan spesifikasi dan karakteristik tertentu, namun berdasarkan urutan data. Dan *statistical encoding* pada klasifikasi teknik ini tidak memperhatikan semantik data. Contoh kelompok teknik kompresi ini adalah pada *Run-Length* coding, *Huffman* coding, *Arithmetic* coding.

b. *Source Coding*

Merupakan klasifikasi teknik kompresi yang bersifat *lossy compression.* Tekniknya berkaitan dengan data semantik (arti data) dan media. Contoh kelompok teknik kompresi ini adalah pada *prediction (DPCM, DM), transformation (FFT, DCT), layered coding (bit position, subsampling, sub-band coding), vector quantization*

c. *Hybrid Coding*

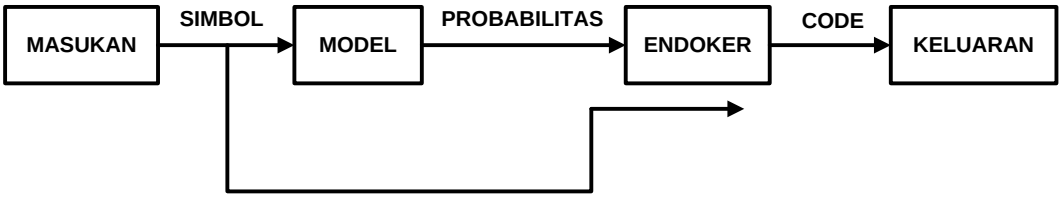
Merupakan klasifikasi teknik kompresi yang bersifat gabungan antara *lossy compression* dan *loseless compression.* Contoh kelompok teknik kompresi ini adalah pada JPEG, MPEG, H.261, DVI

2.1.6 Encoding dan Decoding

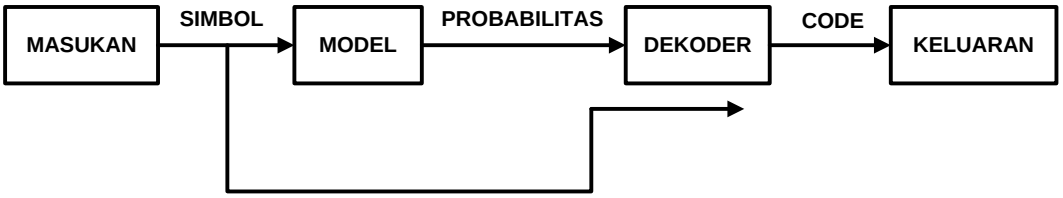
Encoding merupakan teknik untuk mendapatkan kode-kode tertentu (encoder). Dari kode-kode tersebut dapat diaplikasikan untuk pemampatan data dan keamanan data. Dari data-data yang telah dikodekan tersebut, format-format isi dari data tersebut berbentuk kode-kode yang tidak bisa dibaca oleh user. Agar kode-kode tersebut bisa dibaca oleh user, maka kita perlu mengkodekan ulang data tersebut. Hal ini biasa dikenal dengan nama *decoding* (decoder)

Secara umum pemampatan data merupakan merubah suatu simbol-simbol menjadi suatu kode-kode. Pemampatan dikatakan efektif jika ukuran perolehan kode-kode tersebut sangat kecil dibandingkan dengan ukuran kode simbol aslinya. Dari suatu kode-kode atau simbol-simbol dasar suatu model akan dinyatakan dalam kode khusus.

Proses decoding, yaitu proses pengembalian kode-kode yang telah dibuat menjadi simbol-simbol yang dikenal oleh user. Proses decoder ini membaca *header* dari kode-kode yang berisi informasi simbol dan jumlah simbol yang digunakan, setelah pembacaan header proses encoder dengan model ditunjukan pada gambar 2.1, sedangkan proses decoder dengan model ditunjukan pada gambar 2.2 dibawah ini



Gambar 2.1 Model Proses Encoder



Gambar 2.2 Model Proses Dekoder

2.2 Pengkodean Data

Data disimpan di dalam komputer pada *main memory* untuk diproses. Sebuah karakter data disimpan dalam *main memory* menempati posisi 1 byte. Komputer generasi pertama, 1 byte terdiri dari 4 bit, komputer generasi kedua, 1 byte terdiri dari 6 bit dan komputer generasi sekarang, 1 byte terdiri dari 8 bit. Suatu karakter data yang disimpan di *main memory* diwakili dengan kombinasi dari digit binary (binary digit atau bit). Suatu kode biner dapat digunakan untuk mewakili suatu karakter.

Suatu komputer yang berbeda menggunakan kode biner yang berbeda untuk mewakili suatu karakter. Komputer yang 1 byte terdiri dari 4 bit, menggunakan kode binary yang berbentuk kombinasi 4 bit, yaitu BCD (*Binary coded decimal*). Komputer yang menggunakan 6 bit untuk 1 bytenya, menggunakan kode biner yang terdiri dari kombinasi 6 bit, yaitu SBCDIC (*Standard Binary Coded Decimal Interchange Code*). Komputer yang terdiri dari 8 bit, menggunakan kode biner yang terdiri dari kombinasi 8 bit, yaitu EBCDIC (*extended Binary coded decimal interchange code*) atau ASCII (*American standard code of information interchange*).

2.2.1 BCD (Binary Coded Decimal)

BCD merupakan kode biner yang digunakan hanya untuk mewakili nilai desimal saja, yaitu angka 0 sampai dengan 9. BCD menggunakan kombinasi dari 4 bit, sehingga sebanyak 16 (2^4 = 16) kemungkinan kombinasi yang dapat diperoleh dan hanya 10 kombinasi yang dipergunakan.

Kode BCD yang orisinil sudah jarang dipergunakan untuk komputer generasi sekarang, karena tidak dapat mewakili huruf atau simbol-simbol karakter khusus.

2.2.2 SBCDIC (Standard Binary Coded Decimal Interchange Code)

SBCDIC merupakan kode biner perkembangan dari BCD. BCD dianggap tanggung, karena masih ada 6 kombinasi yang tidak dipergunakan, tetapi tidak dapat digunakan untuk mewakili karakter yang lainnya. SBCDIC menggunakan kombinasi 6 bit , sehingga lebih banyak kombinasi yang bisa dihasilkan, sebanyak 64 kombinasi kode, yaitu 10 kode untuk digit angka, 26 kode untuk huruf alphabet dan sisanya karakter-karakter khusus yang dipilih [1].

Posisi bit di SBCDIC dibagi menjadi 2 zone, yaitu 2 bit pertama disebut dengan *alphabet position* dan 4 bit berikutnya disebut dengan *numeric bit position*.

2.2.3 EBCDIC (Extended Binary Coded Decimal Interchange Code)

EBCDIC terdiri dari kombinasi 8 bit yang memungkinkan untuk mewakili karakter sebanyak 256 kombinasi karakter. Pada EBCDIC, *high-order* bits atau 4 bit pertama disebut dengan zone bits dan *low order* bits atau 4 bit kedua disebut dengan numeric bit.

2.2.4 ASCII (American standard code for information interchange) 7 bit

Kode ASCII yang standard menggunakan kombinasi 7 bit, dengan kombinasi sebanyak 127 dari 128 kemungkinan kombinasi.

Kode ASCII 7 bit ini terdiri dari dua bagian, yaitu *control characters* dan *information characters* merupakan karakter-karakter yang mewakili data [1].

2.2.5 ASCII 8 bit

Karakter-karakter graphic yang tidak dapat diwakili oleh ASCII 7 bit, dapat diwakili dengan ASCII 8 bit karena lebiih banyak memberikan kombinasi karakter.

2.3 Komunikasi Data

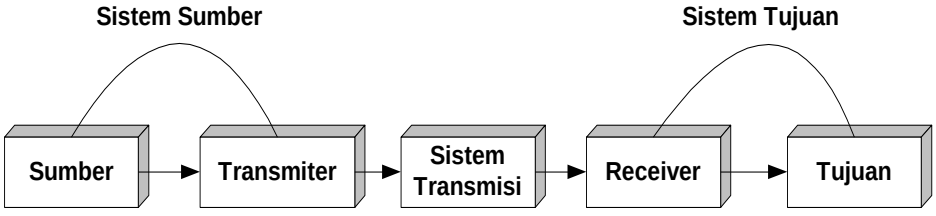
Komunikasi data adalah proses pengiriman informasi diantara dua titik menggunakan kode biner melewati saluran transmisi dan peralatan switching dapat terjadi antara komputer dengan komputer, komputer dengan terminal atau komputer dengan peralatan. Komunikasi data merupakan gabungan dari teknik telekomunikasi dengan teknik pengolahan data.

Adapun tujuan dari komunikasi data adalah sebagai berikut :

1. Memunkinkan pengiriman data dalam jumlah besar efisien, tanpa kesalahan dan ekomis dari suatu tempat ketempat yang lain.
2. Memungkinkan penggunaan sistem komputer dan perlatan pendukung dari jarak jauh (*remote computer use).*
3. Memungkinkan penggunaan komputer secara terpusat maupun secara tersebar sehingga mendukung manajemen dalam hal kontrol, baik desentralisasi ataupu sentralisasi.
4. Mempermudah kemungkinan pengelolaan dan pengaturan data yang ada dalam berbagai mcam sistem komputer.
5. Mengurangi waktu untuk pengelolaan data.
6. Mendapatkan data langsung dari sumbernya.
7. Mempercepat penyebarluasan informasi.

2.3.1 Model Komunikasi Data

Komunikasi data berkaitan dengan pertukaran data diantara dua perangkat yang terhubuang secara langsung yang memungkinkan adanya pertukaran data antar kedua pihak.gambar 2.3 menggambarkan proses komunikasi data.



Gambar 2.3 Komunikasi Data

Pada gambar diatas terdapat elemen-elemen dalam kunci model tersebut :

1. *Source* (sumber) : Alat ini membangkitkan data sehingga dapat ditransmisikan, contoh telepon, *Personal Computer (PC)*
2. *Transmitter* (pengirim): Biasanya data yang dibangkitkan dari sister sumber tidak ditransmisikansecara langsung dalam bentuk aslinya. Sebuah *transmitter* cukup memindah dan menandai informasi dengan cara yang sama seperti sinyal-sinyal elektromagnetik yang dapat ditransmisikan melewati beberapa sistem transmisi berurutan.
3. Sistem transmisi : Berupa jalur transmisi tunggal *(single transmission*)atau jaringan komplek(*complex networ*k)yang menghubungkan antara sumber dengan tujuan (*destination).*
4. Tujuan (*destination)* : menangkap data yang dihasilkan oleh *receiver*

2.3.2 Jaringan Komunikasi Data

Jaringan komunikasi dapat diartikan sebagai suatu sistem yang terbentuk dari interkoneksi fasilitas-fasilitas yang dirancang untuk membawa trafik dari beragam sumber telekomunikasi.

Suatu jaringan terdiri dari *link* dan *node*. Istilah *node* digunakan untuk merepresentasikan sentral, *junction* atau keduanya. Istilah *link* digunakan untuk merepresentasikan kabel, peralatan transmisi, dan sebagainya. Sedangkan trafik adalah informasi yang terdapat di dalam jaringan, yang mengalir melalui *link* dan *node.*

Suatu jaringan komunikasi merupakan sumber daya yang dapat dipakai secara bersamaan (*shared*) oleh sejumlah *end user* untuk berkomunikasi dengan *user* lain yang likasinya berjauhan. Tidak semua *user* menggunakan jaringan pada waktu yang bersamaan, oleh karena itu merupakan suatu hal yang logis apabila sumber daya jaringanyang sangat penting ini dipakai bersama-sama. Penggunaan sumber daya secara bersamaan ini melahirkan konsep sentral.

a. LAN (Local Area Network)

LAN digunakan untuk menghubungkan komputer yang berada di dalam suatu area yang kecil, misalnya di dalam suatu gedung perkantoran atau kampus. Jarak antar komputer yang dihubungkan bias mencapai 5 sampai 10 km. Suatu LAN biasnya bekerja pada kecepatan mulai 10 Mbps sampi 100 Mbps. LAN menjadi populer karena memungkinkan banyak pengguna untuk memakai sumber daya yang dapat digunakan itu misalnya suatu *mainframe*, *file server*, printer, dan sebagainya.

b. MAN (Metropolitan Area Network)

MAN merupakan suatu jaringan yang cakupannya meliputi suatu kota. MAN menghubungkan LAN-LAN yang lokasinya berjauhan. Jangkauan MAN mencapai 10 km sampai beberapa ratus km. Suatu MAN biasanya bekerja pada kecepatan 1,5 sampai 150 Mbps.

c. WAN (Wide Area Network)

WAN dirancang untuk menghubungkan komputer-komputer yang terletak pada suatu cakupan geografis yang luas,seperti hubungan dari suatu kota ke kota yang lain didalm suatu Negara. Cakupan WAN bias meliputi 100 km sampai 1.000 km, dan kecepatan antar kota bias bervariasi antara 1,5 Mbps sampai 2,4 Gbps. Dalam WAN, biaya untuk peralatan untuk transmisi sangat tinggi,dan biasanya jaringan WAN dimiliki dan dioperasikan sebagai suatu jaringan *public*.

d. GAN (Global Area Network)

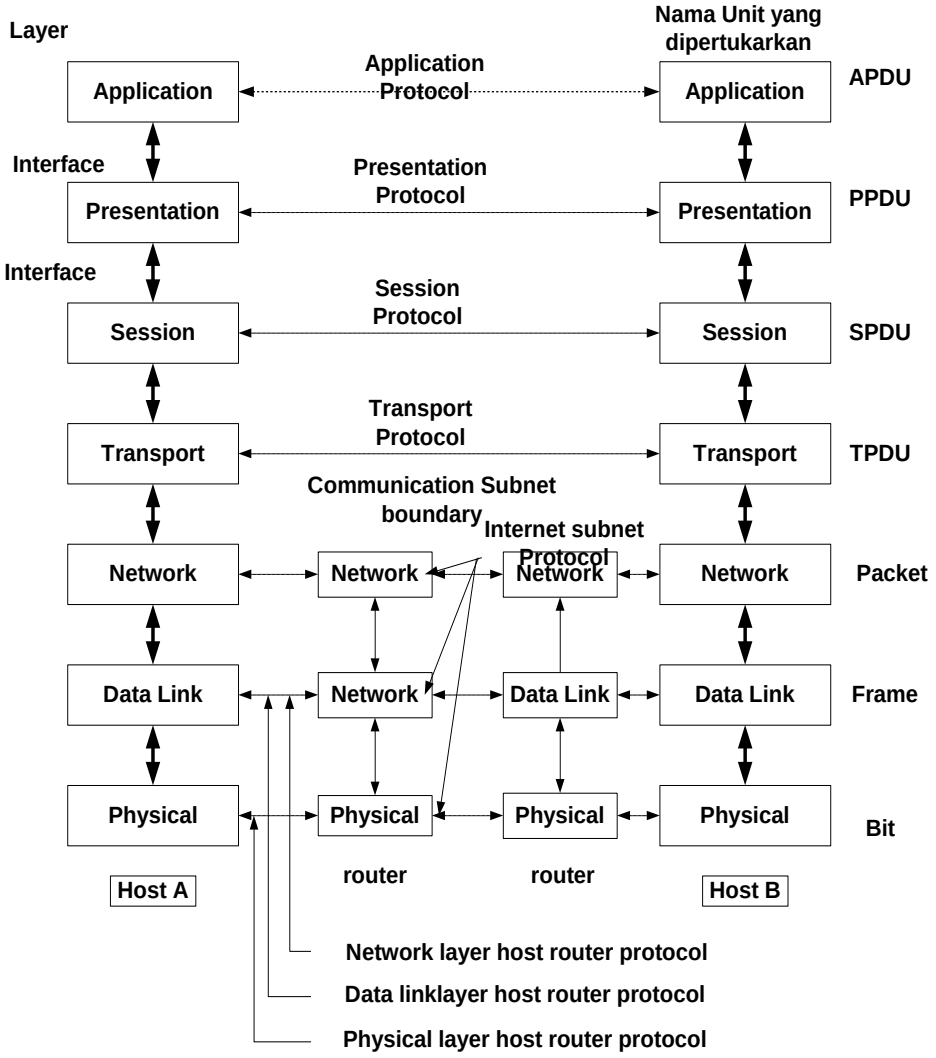
GAN merupakan suatau jarinagn yang menghubungkan Negara-negara diseluruh dunia. Kecepatan GAN bervariasi mulai dari 1,5 Mbps sampai dengan 100 Gbps dan cakupannya mencakupi ribuan kilometer.

2.3.3 Arsitektur Protokol

Open system interconnection (OSI), istilah ini mengacu pada standar untuk pertukaran informasi diantara sistem-sistem yang”terbuka” yaitu sistem yang menerapkan standar OSI. Fakta bahwa suatu sistem merupakan sistem yang terbuka tidak berpengaruh terhadap implementasi, teknologi, atau interkoneksi sistem, melainkan berhubungan dengan kesesuaian suatu sistem terhadap standar yang diterapkan. Berdasarkan hal ini, ISO (*organization for Standarization)*telah menspesifikasikan suatu model referensi OSI yang membagi-bagi fungsi komunikasi ke dalam tujuh lapis *(layer)*. Model OSI menyediakan secara konseptual kerangka kerja untuk komunikasi. Sebenarnya komunikasi dapat terjadi karena menggunakan protokol komunikasi. Dalam konteks jaringan (komunikasi data),sebuah protokol adalah suatu aturan formal dan kesepakatan

yang menentukan bagaimana komputer bertukar informasi melalui sebuah media jaringan. Sebuah protokol mengimplementsikan salah satu atau lebih *layer-layer* OSI. Setiap layer ditugaskan untuk melaksankan suatu subset dari fungsi komunikasi tertentu yang pada gilirannya diterapkan di dalam suatu *data terminal equipment (DTE)* yang berkomunikasi dengan DTE yang lain. Setiap layer mengandalkan layer yang ada dibawahnya untuk melaksanakan fungsi-fungsi yang lebih primitive serta sekaligus menyediakan layanan untuk mendukung layer yang berada diatasnya. Layer-layer ini dirancang agar setiap perubahan yang terjadi di dalam suatu layer tidak mempengaruhi layer yang lainnya.

Model Referensi OSI diperlihatkan pada gambar 2.4



Gambar 2.4 Model Referensi OSI

a. *Physical Layer*

Lapisan ini bertanggung jawab untuk mengaktifkan dan mengatur Physical interface jaringan komputer. Pada lapisan ini, hubungan antar interface-interface dari perangkat keras diatur seperti hubungan antara DTE dan DCE. Interface yang didefinisikan pada lapisan ini antara lain 10BaseT, 100BaseTX,V35, X.21 dan *High Speed Serial Interface (HSSI).*

b. *Data Link Layer*

Lapisan ini mengatur topologi jaringan, *error notification* dan *flow control*. Tugas utama *data link layer* adalah sebagai fasilitas transmisi raw data dan mentransformasikan data tersebut ke saluran yang bebas dari kesalahan transmisi.

c. *Network Layer*

Network Layer berfungsi untuk mengendalikan operasi subnet dengan meneruskan paket-paket dari saru *node* ke *node* yang lain dalam jaringan. Masalah desain yang penting adalah bagaimana cara menetukan route pengiriman paket dari sumber ke tujuannya.

d. *Transport Layer*

Fungsi utama *Transport layer* adalah menerima data dari *session layer*, memecah data menjadi bagian-bagian yang lebih kecil bila perlu, meneruskan data ke *network layer,* dan manjamin bahwa semua potongan data tersebut bisa tiba disis lainnya dengan benar. Selain itu, semua hal tersebut harus dilaksnakan secara efisien, dan bertujuan dapat melindungi *layer-layer* bagian atas dari perubahan teknologi *hardware* yang tidak dapat dihindari.

e. *Session Layer*

Session Layer mengijinkan para pengguna untuk menetapkan *session* dengan pengguna lainnya. Layer ini membuka, mengatur dan menutup suatu *session* antara aplikasi-aplikasi.

f. *Presentation Layer*

Presentation Layer melakukan fungsi-fungsi tertentu yang diminta untuk menjamin penemuan sebuah penyelesaian umum bagi masalah tertentu. Selain memberikan sarana-sarana pelayanan untuk konversi, format dan enkripsi data, *Presentation Layer* juga beberja dengan file berformat ASCII, EBCDIC, JPEG, MPEG, TIFF, PICT, MIDI, dan Quick Time.

g. *Aplication Layer*

Lapisan ini memberikan saran pelayanan langsung ke *user*, yang berupa aplikasi-aplikasi dan mengadakan komunikasi dari program ke program. Jika kita mencari suatu *file* dari *file server* unruk digunakan sebagai aplikasi pengolah kata, maka proses ini bekerja melaui *layer* ini. Demikian pula ketika melakukan pengiriman *email, browse,* ke internet, *chatting,* membuka *telnet session*, atau menjalankan FTP, maka semua proses tersebut dilaksanakan di *layer* ini.

2.4 Algoritma kompresi PPM

Algoritma *Prediction by Partial Matching* (PPM) adalah teknik kompresi data statis berdasarkan konteks pemodelan dan prediksi.. Algoritma ini termasuk dalam kelas *losless compression* yaitu tidak menghilangkan informasi sedikitpun, hanya mewakilkan beberapa informasi yang sama. Metode ini bersifat *two-pass* yaitu menggunakan peta kode yang selalu sama, fase pertama untuk menghitung

kemungkinan tiap simbol dan menentukan peta kodenya dan fase kedua untuk mengubah pesan menjadi kumpulan kode yang akan di transmisikan.

Pada proses, PPM menjalankan rangkaian model deret konteks, dari 0 hingga nilai maksimum *k* yang ditetapkan sebelumnya, untuk memprediksi karakter-karakter berikutnya. Untuk tiap statistik model konteks order *i* yang memuat semua simbol yang telah tersusun, setiap panjang *i* berikutnya dalam input dan berapa kali kejadiannya. Peluang kemungkinan dihitung dari statistik-statistik dengan cara: setiap model, deret 0 hingga deret *k*, diperoleh dari distribusi peluang kemungkinan yang terpisah dan secara efektif disatukan ke dalam satu bagian. Model deret terbesar adalah salah satu model yang secara *default*, digunakan untuk memprediksi simbol awal. Namun, jika suatu simbol yang tidak diketahui terhubung dengan suatu simbol "lepas" maka model lebih kecil akan digunakan untuk prediksi. Proses akan berlanjut hingga salah satu model deret tertentu memprediksikan simbol berikutnya.

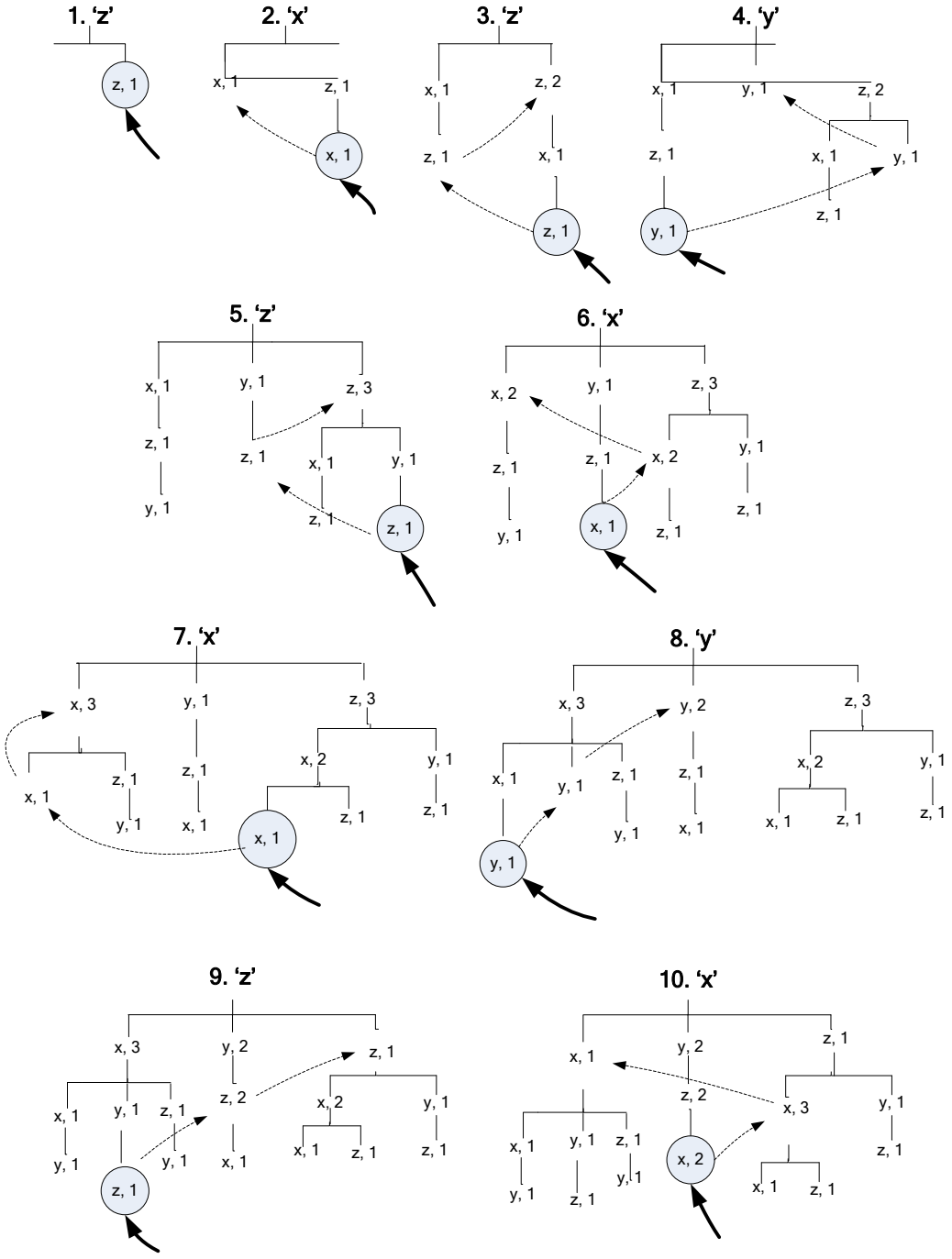
2.4.1 Pembentukan pohon string

Masalah utama dalam pembentukan string adalah membangun struktur data dimana semua konteks saling berhubungan dari setiap simbol ke simbol lainnya. Struktur yang digambarkan disini adalah suatu jenis pohon tertentu. Disebut juga sebagai *tree*. *Tree* adalah pohon, dimana struktur bercabang dari setiap tingkat ditentukan oleh sebagian item data, tidak oleh keseluruhan item. Dalam proses ini, deretan konteks *N* adalah suatu string yang mencakup semua konteks dengan deret-deret $N - 1$ hingga 0, sehingga setiap konteks secara efektif menambahkan satu simbol pada pohon.

Berikut langkah-langkah dalam sebuah pembentukan pohon string zxzyzxxyzx dalam algoritma PPM adalah sebagai berikut :

1. Masukkan sebuah karakter pada kalimat yang akan di input ke dalam pohon PPM sebagai *node* baru pada level 1.
2. Masukkan karakter selanjutnya pada kalimat yang akan di input. Periksa pohon PPM tersebut, apakah ada *node* jembatan atau tidak pada pohon PPM, jika ya, tambahkan karakter sebagai anak *node* dari jembatan, dan jika tidak, tambahkan karakter sebagai *node* baru pada level 1.
3. Periksa *node* baru yang ditambahkan pada pohon PPM, apakah node tersebut terletak pada level 1, jika ya, maka proses telah selesai, dan jika tidak cari *node* terakhir pada pohon PPM yang mengalami perubahan pada level 1, kemudian tambahkan node yang di input sebagai anak dari node tersebut.
4. Periksa *node-node* pada level 1, apakah ada karakter yang sama dengan karakter node yang di input, jika ya, tambahkan *node*,jika tidak, buat *node* baru pada level 1.
5. Ulangi langkah no 2, 3 dan 4 sampai karakter-karakter yang ada pada kalimat yang akan di input telah digunakan pada pohon PPM.

Pada gambar 2.5 diperlihatkan pembentukan pohon string zxzyzxxyzx di bawah ini :



Gambar 2.5 Pembentukan pohon string 'zxzyzxxyzx'

2.4.2 Proses Encoding

Pada umumnya, algoritma kompresi data melakukan penggantian satu atau lebih simbol input dengan kode tertentu. Berbeda dengan cara tersebut, algoritma kompresi PPM menggantikan satu deretan simbol input dengan sebuah bilangan *floating point.* Semakin panjang dan semakin kompleks pesan yang dikodekan, semakin banyak bit yang diperlukan untuk keperluan tersebut.

Output dari algoritma kompresi PPM ini adalah satu angka yang lebih kecil dari 1 dan lebih besar atau sama dengan 0. Angka ini secara unik dapat di-*decode* sehingga menghasilkan deretan simbol yang dipakai untuk menghasilkan angka tersebut.

Untuk menghasilkan angka output tersebut, tiap simbol yang akan di-*encode* diberi satu set nilai probabilitas. Contoh, andaikan kata **SWISS_MISS** akan di- *encode.* Akan didapatkan tabel probabilitas berikut :

Tabel 2.1 Contoh Tabel Probabilitas untuk Kata

"SWISS_MISS"

Karakter	**Probabilitas**
S	5/10
W	1/10
I	2/10
M	1/10
_	1/10

Setelah probabilitas tiap karakter diketahui, karakter akan diberikan range tertentu yang nilainya berkisar di antara 0 dan 1, sesuai dengan probabilitas yang ada. Dalam hal ini tidak ada ketentuan urut-urutan penentuan segmen, asalkan antara *encoder* dan *decoder* melakukan hal yang sama.

Dari tabel 2.1 di atas dibentuk tabel 2.2 berikut:

Tabel 2.2 Range Simbol untuk Kata

"SWISS_MISS"

Karakter	**Probabilitas**	**Range**
S	5/10	0.50 – 1.00
W	1/10	0.40 – 0.50
I	2/10	0.20 – 0.40
M	1/10	0.10 – 0.20
_	1/10	0.00 – 0.10

Dari tabel ini, satu hal yang perlu dicatat adalah tiap karakter melingkupi range yang disebutkan kecuali bilangan yang tinggi.

Selanjutnya untuk melakukan proses *encoding* dipakai algoritma berikut:

Set low = 0.0

Set high = 1.0

While (simbol input masih ada) do

Ambil simbol input

CR = high – low

High = low + CR*high_range (simbol)

Low = low + CR*low_range (simbol)

End While

Cetak Low

Di sini ‘Low’ adalah output dari proses algoritma PPM.

Untuk kata ‘SWISS_MISS’ di atas, pertama kita ambil karakter ‘S’. Nilai CR adalah 1-0 = 1. High_range (S) = 1.0, Low_range(S) = 0.5.

Kemudian didapatkan nilai

high = 0.00 + CR*1.0 =1.0

low = 0.00 + CR*0.5 = 0.5

Kemudian diambil karakter ‘W’. Nilai CR adalah 1.0 – 0.5 = 0.5. High_range(W)= 0.5, Low_range (W) = 0.4. Kemudian didapatkan nilai

high = 0.5 + CR*0.5 = 0.75

low = 0.5 + CR*0.4 = 0.70

Dan seterusnya yang diringkaskan dalam tabel 2.3 berikut.

Tabel 2.3 Proses Encoding untuk Kata

"SWISS_MISS"

Karakter	LOW	HIGH	CR
	0.0	1.0	1.0
S	0.5	1.0	0.5
W	0.70	0.75	0.05
I	0.71	0.72	0.001
S	0.715	0.72	0.0005
S	0.7175	0.72	0.0025
_	0.7175	0.71775	0.00025
M	0.717525	0.717550	0.000025
I	0.717530	0.717535	0.0000025
S	0.7175325	0.717535	0.00000025
S	0.71753375	0.717535	0.00000125

Dari proses ini didapatkan nilai

Low = 0.71753375

Nilai inilah yang ditransmisikan untuk membawa pesan 'SWISS_MISS'.

Untuk lebih jelasnya, alur algoritma dari proses encoding ini dapat dilihat pada gambar 3.5.1

2.4.3 Proses Decoding

Untuk melakukan *decoding* dipakai algoritma berikut:

Ambil *encoded-symbol (ES)*

Do

Cari range dari simbol yang melingkupi *ES*

Cetak simbol

CR = high_range – low_range

ES = ES – low_range

ES = ES / CR

Until simbol habis

Dalam hal ini simbol habis bisa ditandai dengan simbol khusus (End of Message misalnya) atau dengan menyertakan panjang pesan waktu dilakukan transmisi.

Untuk pesan yang tadi di- *encode* (ES = 0.71753375) dilakukan proses *decoding* sebagai berikut. Didapatkan range simbol yang melingkupi ES adalah simbol/karakter ‘T’.

Low_range= 0.5

High_range=1.0

CR = 1.0 - 0.5 = 0.5

ES = 0.71753375 – low_range

ES = 0.21753375

ES = 0.21753375 / CR

ES = 0.4350675

Untuk ringkasnya, proses *decoding* bisa dilihat dalam tabel 2.4 berikut:

Tabel 2.4. Proses Decoding untuk Kata “SWISS_MISS”

ES	Karakter	L	H	CR
0.4350675	S	0.5	1.0	0.5
0.350675	W	0.4	0.5	0.1
0.753375	I	0.2	0.4	0.2
0.50675	S	0.5	1.0	0.5
0.0135	S	0.5	1.0	0.5
0.135	_	0	0.1	0.1
0.35	M	0.1	0.2	0.1
0.75	I	0.2	0.4	0.2
0.5	S	0.5	1.0	0.5
0	S	0.5	1.0	0.5

Untuk lebih jelasnya, alur algoritma dari proses decoding ini dapat dilihat pada gambar 3.5.2

2.5 File Transfer Protokol

File Transfer Protokol (FTP) adalah suatu protokol yang berfungsi untuk tukar-menukar file dalam suatu network yang mensupport TCP/IP protokol. Dua hal penting yang ada dalam FTP adalah FTP server dan FTP Client. FTP server menjalankan software yang digunakan untuk tukar menukar file, yang selalu siap memberian layanan FTP apabila mendapat request dari FTP client. FTP client adalah komputer yang merequest koneksi ke FTP server untuk tujuan tukar menukar file (mengupload atau mendownload file). Contoh FTP server yang bisa digunakan adalah *Filezilla server.*

Tujuan FTP server adalah sebagai berikut :

1. Untuk men-*sharing* data.

2. Untuk menyediakan *indirect* atau implicit *remote computer*.

3. Untuk menyediakan tempat penyimpanan bagi user.

4. Untuk menyediakan transfer data yang *reliable* dan efisien.

FTP sebenarnya cara yang tidak aman untuk mentransfer file karena file tersebut ditransfer tanpa melalui enkripsi terlebih dahulu tetapi melalui clear text. Mode text yang dipakai untuk transfer data adalah format ASCII atau format Binary. Secara default, ftp menggunakan mode ASCII untuk transfer data. Karena pengirimannya tanpa enkripsi, maka username, password, data yang ditransfer, maupun perintah yang dikirim dapat di *sniffing* oleh orang dengan menggunakan protocol analyzer (Sniffer). Solusi yang digunakan adalah dengan menggunakan SFTP (SSH FTP) yaitu FTP yang berbasis pada SSH atau menggunakan FTPS (FTP over SSL) sehingga data yang dikirim terlebih dahulu dienkripsi (dikodekan).

FTP biasanya menggunakan dua buah port yaitu port 20 dan 21 dan berjalan exclusively melalui TCP. FTP server Listen pada port 21 untuk incoming connection dari FTP client. Biasanya port 21 untuk command port dan port 20 untuk data port. Pada FTP server, terdapat 2 mode koneksi yaitu aktif (active) mode dan pasif (passive) mode.