BAB II

LANDASAN TEORITIS

Pada dasarnya, data apapun adalah rangkaian bit 0 dan 1. Yang membedakan antara suatu data dengan data yang lain adalah ukuran dari rangkaian bit dan bagaimana 0 dan 1 ditempatkan dalam rangkaian bit tersebut. Semakin kompleks suatu data maka rangkaian bit yang diperlukan pun menjadi semakin rumit dan panjang, dengan demikian ukuran keseluruhan data juga semakin besar. Rangkaian bit inilah yang nantinya akan diolah dalam proses kompresi.

Dalam tulisan ini, akan dibandingkan dua metode kompresi yaitu metode Huffman dan *Dynamic Markov Compression*. Untuk melakukan kedua metode tersebut diperlukan beberapa ilmu pendukung. Selanjutnya akan dibahas mengenai ilmu-ilmu pendukung tersebut.

2.1 Teori Peluang

2.1.1 Konsep Dasar Peluang

Dalam melakukan penelitian, sering kali dilakukan berbagai percobaan atau eksperimen. Eksperimen dalam hal ini merupakan eksperimen acak. Sebagaimana dikemukakan oleh Djauhari (1994:2) bahwa Eksperimen acak memiliki karakteristik:

i. Hasil eksperimen tak dapat diduga sebelumnya dengan tingkat keyakinan yang pasti.

ii. Semua hasil yang mungkin dapat diidentifikasi terkandung di dalm suatu himpunan.
iii. Dapat diasumsikan bisa dilakukan berulang-ulang dalam kondisi yang sama.

Setelah melakukan eksperimen, maka akan diperoleh hasil-hasil yang mungkin dari eksperimen itu, kumpulan semua hasil-hasil yang mungkin tersebut disebut dengan ruang sampel dan dinotasikan dengan *S*. Pengertian ruang sampel diperjelas pada definisi berikut:

Definisi 2.1 (Herrhyanto, 1994:19)

Apabila kita melakukan suatu eksperimen acak, maka semua hasil yang mungkin dari eksperimen itu dinamakan ruang sampel.

Sedangkan masing-masing hasil yang mungkin dari eksperimen atau setiap anggota dari ruang sampel dinamakan titik-titik sampel.

Definisi 2.2 (Herrhyanto, 1994:19)

Sebuah peristiwa adalah sebuah himpunan bagian dari ruang sampel. Setiap himpunan bagian adalah sebuah peristiwa.

Dalam suatu eksperimen acak selalu terjadi suatu ketidakpastian apakah suatu peristiwa akan terjadi atau tidak. Jika suatu peristiwa sudah pasti akan terjadi maka dapat dikatakan peristiwa tersebut memiliki peluang sama dengan 1. Sebaliknya, jika peristiwa tersebut sangat tidak mungkin terjadi, maka peluangnya sama dengan 0. Tetapi yang terjadi dalam kehidupan sehari-hari sangat jarang

ditemukan peristiwa yang memiliki peluang sama dengan 0 atau 1, kebanyakan peristiwa mempunyai peluang antara 0 dan 1. Berikut adalah definisi peluang.

Definisi 2.3

Misalkan untuk setiap peristiwa E dari ruang sampel S, $P(E)$ memenuhi tiga syarat:

(i) $0 \leq P(E) \leq 1$

(ii) $P(S) = 1$

(iii) $P\bigcup_{i=1}^{\infty} E_i = \sum_{i=1}^{\infty} P(E_i)$

Untuk setiap $E_1, E_2, \ldots$ di S dengan $E_i \cap E_j = \phi$, $i = j$

$P(E)$ dibaca peluang dari peristiwa E. Definisi di atas adalah definisi peluang, sedangkan cara menghitung peluang suatu peristiwa dapat dilihat pada definisi-definisi berikut:

Definisi 2.4 (Definisi Klasik)

Jika peristiwa E dapat terjadi sebanyak n kali di antara N peristiwa yang saling asing dan semuanya terjadi dengan "kesempatan yang sama", maka peluang terjadinya peristiwa E adalah

$$P(E) = \frac{n}{N}$$

Definisi 2.5 (Definisi Empiris)

Perhatikan frekuensi relatif sebuah peristiwa di antara sejumlah pengamatan. Peluang peristiwa tersebut adalah limit dari frekuensi relatifnya jika pengamatan di ulang sampai dengan tak hingga.

Definisi 2.6

Misalkan X diskrit dengan nilai $X = x_1, x_2, x_3, \ldots, x_n$, fungsi yang memenuhi syarat:

(a) $P(x_i) \geq 0$, untuk setiap x_i, dan

(b) $\sum_{i=1}^{n} P(x_i) = 1$

Disebut fungsi peluang

Contoh:

Misalkan dilakukan eksperimen pengundian 2 mata uang, jika x menyatakan banyaknya kejadian munculnya angka, ruang sampel untuk pengundian ini adalah $S = \{0, 1, 2\}$. Berikut adalah tabel distibusi peluang dari kejadian munculnya angka dalam dua kali pengundian

Tabel 2.1
Distibusi Peluang Pengundian Koin

x	P(x)
0	1/4
1	1/2
2	1/4
$\sum_{i=1}^{n} P(x_i)$	1

2.1.2 Rantai Markov

Proses statokastik $X = \{X(t), t \in T\}$ adalah kumpulan dari variabel-variabel acak. T dalam hal ini adalah himpunan index atau parameter waktu dan $X(t)$ adalah keadaan dari suatu proses pada waktu t. Jika himpunan T mempunyai jumlah anggota terbatas maka X dinamakan proses stokastik dengan parameter waktu diskrit. Sedangkan jika himpunan T mempunyai jumlah anggota terbatas maka X adalah proses stokastik dengan parameter waktu kontinu. Himpunan nilai $X(t)$ yang mungkin yaitu range $X(t)$ dinamakan ruang keadaan dari proses stokastik

Salah satu proses stokastik yang cukup dikenal adalah model rantai Markov. Model rantai Markov pertama kali diperkenalkan oleh Andrei A. Markov pada tahun 1902. “Model ini berhubungan dengan suatu rangkaian proses di mana kejadian akibat suatu eksperimen hanya bergantung pada kejadian yang langsung mendahuluinya dan tidak tergantung pada rangkaian kejadian sebelum-sebelumnya yang lain” (Abdurrachman, 1999:499). Model ini dapat digunakan untuk memperkirakan perubahan-perubahan di waktu yang akan datang dalam variabel-variabel dinamis atas dasar perubahan-perubahan dari variabel-variabel dinamis tersebut di waktu yang lalu. Tetapi model ini tidak memberikan keputusan rekomendasi, melainkan hanya informasi probabilitas mengenai situasi keputusan yang dapat membantu pengambil keputusan mengambil keputusan.

Aplikasi rantai Markov yang terkenal diantaranya adalah analisa perpindahan merek dari pelanggan, analisa gempa, analisa probabilitas kerusakan mesin, analisa hutang tak tertagih, dan masih banyak lagi yang lainnya.

Definisi 2.7

Proses Markov adalah proses stokastik masa lalu tidak mempunyai pengaruh pada masa yang akan datang bila masa sekarang diketahui.

Bila $t_{n-1} < t_n$ maka :

$$P\{X(t_n) \leq X_n \mid X(t), t \leq t_{n-1}\} = P\{X(t_n) \leq X_n \mid X(t_{n-1})\}$$

Bila $t_1 < t_2 < \ldots\ldots < t_n$ maka :

$$P\{X(t_n) \leq X_n \mid X(t_{n-1}), \ldots\ldots X(t_1)\} = P\{X(t_n) \leq X_n \mid X(t_{n-1})\}$$

Definisi 2.8

Diberikan sebuah himpunan N dengan keadaaan $E = \{E_1, E_2, \ldots, E_N\}$ dan rantai keadaan itu :

$$E_{j1}, E_{j2}, E_{j3}, \ldots\ldots E_{jN}$$

Rantai tersebut adalah rantai Markov bila :

$$P(E_k | E_{j1} E_{j2} \ldots\ldots E_{ji}) = P(E_k | E_{ji})$$

Definisi 2.9

Probabilitas transisi adalah probabilitas pergerakan dari keadaan E_i ke E_j, dinotasikan dengan p_{ij}.

$$P(E_j | Ek_1 . Ek_2, \ldots Ek_v, E_i) = P(E_j | E_i) = p_{ij}$$

Untuk semua i dan j, $p_{ij} \geq 0$ dan untuk setiap i $\sum_{j=1}^{N} p_{ij}$

Definisi 2.10

Matriks transisi sebuah sistem dengan N keadaan, E_1, E_2,…., E_N dan probabilitas transisi p_{ij} = 1,2,….N adalah:

$$T = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1N} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2N} \\ p_{31} & p_{342} & p_{33} & \cdots & p_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{N1} & p_{N2} & p_{N3} & \cdots & p_{NN} \end{bmatrix}$$

Contoh:

Misalkan di suatu kota terdapat tiga perusahaan pengiriman barang yaitu perusahaan A, B dan C. Jumlah penduduk kota tersebut adalah 2000 orang, Setiap bulannya penduduk kota tersebut menggunakan layanan dari salah satu dari ketiga perusahaan tersebut. Setelah diadakan survei, ternyata pelanggan tidak setia sepenuhnya pada perusahaan pengiriman manapun. Hasil surveinya adalah sebagai berikut :

a. Jika pelanggan melakukan transaksi dengan perusahaan A bulan ini, ada probabilitas sebesar 50% bahwa pelanggan akan melakukan transaksi dengan A kembali dibulan berikutnya. Sedangkan bahwa pelanggan akan berpindah ke B dan C, terdapat probabilitas sebesar 30% dan 20%
b. Untuk pelanggan yang bulan ini mengadakan transaksi dengan perusahaan B, terdapat probabilitas sebesar 55% bahwa pelanggan tersebut akan kembali pada mereka dibulan berikutnya. Sedangkan bahwa pelanggan akan berpindah ke A dan C, terdapat probabilitas sebesar 20% dan 25%

c. Untuk pelanggan yang bulan ini mengadakan transaksi dengan perusahaan C, probabilitas bahwa pelanggan akan kembali pada mereka dibulan berikutnya adalah 60%. Sedangkan probabilitas pelanggan akan beralih ke A dan B adalah 20% dan 20%.

Probabilitas pergerakan pelanggan perbulan dapat dijabarkan sebagai matriks transisi:

$$T = \begin{bmatrix} 0,5 & 0,3 & 0,2 \\ 0,2 & 0,55 & 0,25 \\ 0,2 & 0,2 & 0,6 \end{bmatrix}$$

dan diperjelas melalui tabel berikut:

Tabel 2.2
Probabilitas Transisi

Bulan ini	Bulan Berikutnya		
	A	B	C
A	0,5	0,3	0,2
B	0,2	0,55	0,25
C	0,2	0,2	0,6

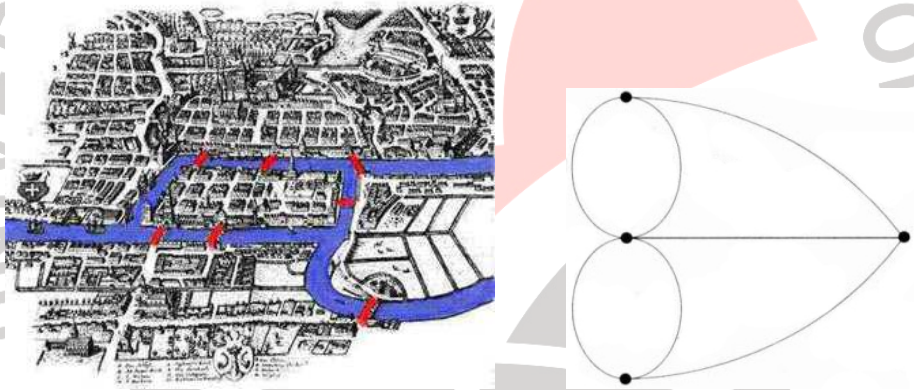
Sumber: Yashinta.net

Dari tabel probabilitas transisi di atas, perusahaan dapat menentukan langkah selanjutnya untuk mendapatkan kembali pelanggan, misalnya dengan meningkatkan pelayanan, memberi diskon, atau memasang iklan lebih banyak.

2.2 Pohon dalam Graf

2.2.1 Graf

Graf digunakan untuk merepresentasikan objek-objek dan hubungan antar objek-objek tersebut. Representasi visual dari graf adalah dengan menyatakan objek sebagai titik dan hubungan antar objek sebagai garis. Sebagai contoh lihat gambar (2.1) , gambar 2.1(a) adalah peta kota Konigsberg dan tujuh jembatannya yang terkenal, sedangkan gambar 2.1(b) adalah graf yang merepresentasikan jembatan kota Konigsberg dan 7 jembatannya. Daratan dinyatakan oleh titik-titik dan jembatan dinyatakan oleh garis-garis yang menghubungkan titik-titik tersebut.



(a) (b)

Gambar 2.1 (a) Peta Kota Konigsberg; (b) Representasi Jembatan Konigsberg dalam Graf

Definisi 2.11

Graf *G* didefinisikan sebagai pasangan himpunan (*V*,*E*), ditulis dengan notasi *G* = (*V*,*E*) yang dalam hal ini *V* adalah himpunan tidak kosong dari simpul-simpul dan *E* adalah himpunan sisi yang menghubungkan sepasang simpul.

Dari definisi di atas dapat dilihat bahwa *V* tidak boleh kosong sedangkan *E* boleh kosong. Jadi, suatu graf boleh tidak mempunyai sisi satu pun, tetapi minimal harus ada satu simpul.

Sisi pada graf dapat menmpunyai orientasi arah. Berdasarkan orientasi arah pada sisi, maka graf dapat dibedakan menjadi 2 jenis yaitu:

1. Graf Tak Berarah (*Undirected Graph*)

   Graf yang sisi-sisinya tidak mempunyai orientasi arah disebut graf tak berarah. Urutan pasangan simpul yang dihubungkan oleh sisi tidak diperhatikan, dengan kata lain $(u,v) = (v,u)$.

2. Graf Berarah (*Directed Graph* atau *Digraph*)

   Graf yang sisi-sisinya mempunyai orientasi arah disebut graf berarah. Pada graf berarah sisi (u,v) tidak sama dengan sisi (v,u), dengan kata lain $(v,u) \neq (v,u)$. Sisi (u,v) artinya sisi dengan tanda panah mengarah dari simpul *u* ke simpul *v*.

Berikut ini adalah beberapa terminologi dalam graf yang sering digunakan.

1. Bertetangga (*Adjasent*)

Definisi 2.12

Dua buah simpul pada graf tak berarah *G* dikatakan bertetangga bila keduanya terhubung langsung dengan sebuah sisi. Dengan kata lain, *u* bertetangga dengan v jika (u,v) adalah sisi pada graf *G*.

2. Bersisian (*Incident*)

Definisi 2.13

Untuk sembarang sisi $e = (u,v)$, sisi e dikatakan bersisian dengan simpul u dan simpul v.

3. Derajat (*Degree*)

Definisi 2.14

Derajat suatu simpul v, ditulis $d(v)$ menyatakan banyaknya sisi yang *incident* dengan v.

Untuk graf berarah, $d(v) = d_{in}(v) + d_{out}(v)$ dengan

$d_{in}(v)$: banyaknya sisi yg masuk ke v

$d_{out}(v)$: banyaknya sisi yg keluar dari v

4. Lintasan (*Path*)

Definisi 2.15

Suatu lintasan u-v adalah perjalanan u-v yang tidak mengulangi sebarang simpul dan sebarang sisi.

Lintasan u-v dengan panjang n dinyatakan dengan urutan berselang-seling simpul-simpul dan sisi-sisi $v_0, e_1, v_1, e_2, v_2, \ldots, e_n, v_n$. Sedemikian sehingga $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2)$, ... , $e_n = (v_{n-1}, v_n)$ adalah sisi-sisi pada graf G.

5. Siklus (*Cycle*) atau Sirkuit (*Circuit*)

Definisi 2.16

Lintasan yang berawal dan berakhir pada simpul yang sama disebut sirkuit atau siklus.

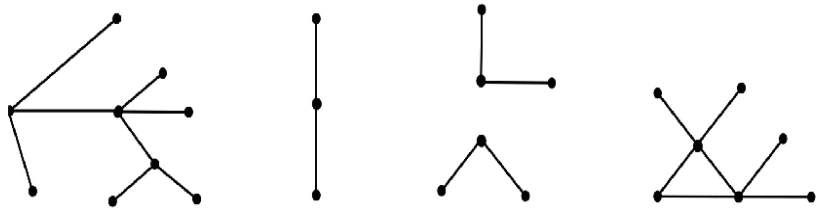
2.2.2 Pohon

Pohon adalah salah satu bentuk khusus dari graf. Berikut adalah definisi pohon:

Definisi 2.17

Pohon (*tree*) adalah graf tak berarah terhubung yang tidak memuat sirkuit.

Perhatikan gambar (2.2), gambar 2.2(a), 2.2(b) dan 2.2(c) adalah contoh pohon, 2.1(d) bukan merupakan sebuah pohon, karena memuat sirkuit.



(a) **(b)** **(c)** **(d)**

Gambar 2.2 Graf

Teorema 2.1

Jika *T* pohon, maka untuk setiap dua titik *u* dan *v* yang berbeda di *T* terdapat tepat satu lintasan yang menghubungkan kedua titik tersebut.

Bukti

Misalkan pohon *T* dan ada dua lintasan yang berbeda yang menghubungkan titik *u* dan *v*, maka dua litasan tersebut akan menghubungkan titik *u* dan *v* sehingga kedua lintasan ini akan menghubungkan kedua titik tersebut, kedua lintasan ini akan membentuk sirkuit. Berdasarkan definisi, *T* tidak memiliki sirkuit, dengan demikian haruslah lintasan pertama sama dengan lintasan kedua. Hal ini bertentangan dengan pemisalan, jadi, terbukti bahwa setiap dua titik yang berbeda di *T* memiliki tepat satu lintasan yang menghubungkan kedua titik tersebut.

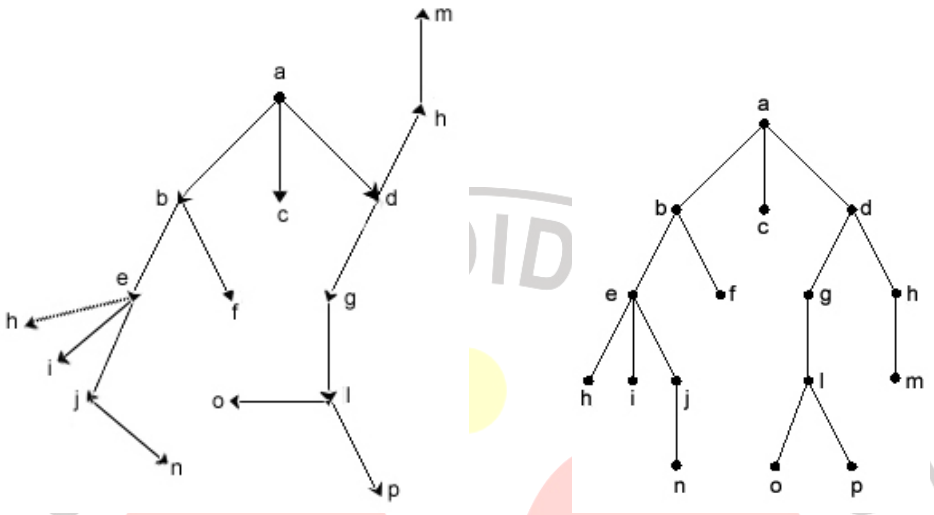
Definisi 2.18 (Priatna, 2008:4.9)

Pohon berakar adalah graph berarah (*digraph*) *T* yang mempunyai dua syarat:

1. Bila arah sisi-sisi pada *T* diabaikan, hasil graph tidak berarahnya merupakan sebuah pohon, dan
2. Ada titik tunggal *R* sedemikian hingga derajat masuk *R* adalah 0 dan derajat masuk sembarang titik lainnya adalah 1. Titik *R* disebut akar dari pohon berakar itu.

Contoh pohon berakar dapat dilihat pada Gambar (2.3). Gambar 2.3(a) adalah gambar graf berakar dengan *a* sebagai akar karena derajat masuk *a* adalah sama dengan 0 dan derajat masuk semua titik lainnya adalah 1. Sedangkan Gambar 2.3(b) adalah cara menggambar graf berakar dengan mengabaikan arahnya.

Simpul *a* sebagai akar diletakkan paling atas, simpul yang bertetangga dengan *a* yaitu *b*, *c*, *d* diletakkan di bawah simpul *a* dan seterusnya.



(a) (b)

Gambar 2.3 (a)Pohon Berarah; (b)Pohon Berakar

Berikut adalah beberapa terminologi pada pohon berakar, perhatikan gambar (2.3):

1. Anak (*Child* atau *Successor*) dan Orangtua (*Parent*)

   Simpul anak adalah simpul yang mempunyai derajat masuk sama dengan 1. dan orang tua dari simpul tersebut adalah simpul yang ajasen dengan simpul tersebut. Contoh anak pada Gambar (2.3) adalah simpul *h*, *i* dan *j* dan *e* adalah orang tua dari simpul *h*, *i* dan *j*.

2. Lintasan (*Path*)

   Suatu lintasan *u*-*v* adalah perjalanan *u*-*v* yang tidak mengulangi sebarang simpul dan sebarang sisi. Contoh lintasan pada Gambar (2.3) adalah lintasan dari *a* ke *m* ayaitu *a*, *d*, *h*, *m*, dan panjang lintasannya adalah 3.

3. Daun (*Leaf*)

Daun adalah simpul yang berderajat nol (atau tidak mempunyai anak). Contoh daun pada Gambar (2.3) simpul *h, i, n, o, p* dan *m*.

4. Simpul Dalam (*Internal Nodes*)

   Simpul dalam adalah simpul yang memiliki derajat keluar yang tidak nol. Contoh simpul dalam pada Gambar (2.3) adalah *b, d, e, g, h, j* dan *l*.

5. Aras (*Level*) atau Tingkat

   Aras adalah tingkat suatu simpul pada pohon berarah, aras 0 di mulai dari akar pohon anak dari simpul a memiliki aras 1, dan seterusnya. Pada Gambar (2.3) Simpul *a* berada pada aras nol, simpul *b, c* dan *d* berada pada aras satu dan seterusnya.

6. Tinggi (*Height*) atau Kedalaman (*Depth*)

   Tinggi atau kedalaman pohon adalah aras maksimum dari suatu pohon disebut Pohon berakar pada gambar (2.3) di atas mempunyai tinggi 4.

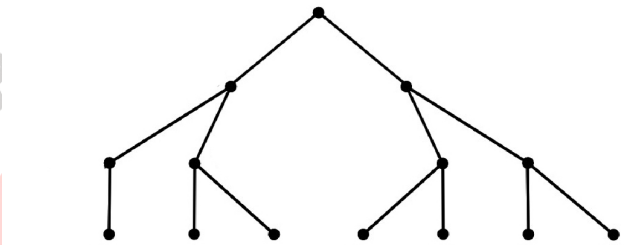
Dalam struktur data, pohon berakar memegang peranan yang cukup penting. Struktur ini biasanya digunakan terutama untuk menyajikan data yang mengandung hubungan hierarki antara elemen-elemennya. Bentuk pohon khusus yang lebih mudah dikelola dalam komputer adalah pohon biner.

Definisi 2.19

sebuah pohon biner (*binary tree*) adalah sebuah pohon struktur data dimana setiap simpul memiliki paling banyak dua anak.

Setiap simpul didalam pohon biner hanya dapat mempunyai 0, 1 atau 2 suksesor. Untuk menyajikan pohon biner, simpul akar adalah simpul yang digambar pada bagian paling atas. Sedangkan suksesor kiri (*left successor*) digambarkan sebagai garis ke kiri bawah dan suksesor kanan (*right successor*) sebagai garis ke kanan bawah.

Contoh pohon biner dapat dilihat pada Gambar (2.4)



Gambar 2.4 Pohon Biner

Terdapat beberapa jenis pohon biner, di antaranya:

1. Pohon Biner Berakar (rooted binary tree)

   Pohon biner berakar adalah sebuah pohon berakar dimana setiap simpul paling banyak mempunyai dua anak

2. Pohon Biner Penuh (*Full Binary Tree*)

   Pohon biner penuh adalah Semua simpul kecuali daun memiliki dua anak dan tiap cabang memiliki panjang ruas yang sama. Dengan kata lain panjang lintasan dari akar ke setiap daun adalah sama. Simpul-simpul pada pohon biner ini hanya mempuanyai 0 atau 2 anak.

3. Pohon Biner Lengkap (*Complete Binary Tree*)

Semua simpul kecuali daun memiliki dua anak tetapi tiap cabang tidak memiliki panjang ruas yang sama.

2.3 Sistem Bilangan Biner

Sebelum mempelajari sistem bilangan biner lebih lanjut, sebaiknya terlebih dahulu mengetahui mengenai apa itu sistem bilangan. “Sistem bilangan adalah suatu cara untuk mewakili besaran dari suatu item fisik. Sistem bilangan menggunakan basis tertentu yang tergantung dari jumlah bilangan yang digunakan“(Cepi, 2008:26).

Berikut adalah beberapa sistem bilangan yang dikenal dalam ilmu komputer:

1. Sistem bilangan desimal dengan basis 10.
2. Sistem bilangan biner dengan basis 2.
3. Sistem bilangan hexadesimal dengan basis 16.

Sistem bilangan yang paling umum dipakai adalah sistem bilangan desimal.

Definisi 2.20

Sistem bilangan biner atau sistem bilangan basis dua adalah sebuah sistem penulisan angka dengan menggunakan dua simbol yaitu 0 dan 1.

Pengelompokan bilangan biner dalam komputer selalu berjumlah 8, dengan istilah 1 *Byte*/bita. Dalam istilah komputer, 1 *Byte* = 8 bit. Berikut adalah beberapa contoh bilangan desimal dan bilangan binernya dalam *Byte*:

Tabel 2.3
Bilangan Desimal dan Representasi Biner dalam *Byte*

Desimal	Biner (8 bit)
0	0000 0000
1	0000 0001
2	0000 0010
3	0000 0011
4	0000 0100
5	0000 0101
6	0000 0110
7	0000 0111
8	0000 1000

Desimal	Biner (8 bit)
9	0000 1001
10	0000 1010
11	0000 1011
12	0000 1100
13	0000 1101
14	0000 1110
15	0000 1111
16	0001 0000
17	0001 0001

Sumber: Wikipedia.org

2.3.1 Konversi Bilangan Desimal ke Biner

Untuk konversi bilangan desimal ke bilangan biner, perhatikan contoh berikut. Misalkan akan dikonversi suatu bilangan desimal yaitu 1335 ke dalam bentuk bilangan biner.

Pertama-tama bilangan 1335 dikurangi dengan bilangan 2^x yang paling mendekatinya yaitu $1024 = 2^{10}$, selanjutnya hasil pengurangan $1335 - 1024 = 311$ di kurangi dengan bilangan 2^x yang paling mendekatinya yaitu $256 = 2^8$, dan seterusnya sehingga dapat dijabarkan seperti berikut:

$1335 - 1024\ (2^{10}) = 311$

$311 - 256\ (2^8) = 55$

$55 - 32\ (2^5) = 23$

$23 - 16\ (2^4) = 7$

$7 - 4\ (2^2) = 3$

$3 - 2\ (2^1) = 1$

$1 - 1\ (2^0) = 0$

atau

$1335 = (\mathbf{1}x2^{10}) + (\mathbf{0}x2^{9}) + (\mathbf{1}x\ 2^{8}) + (\mathbf{0}x\ 2^{7}) + (\mathbf{0}x\ 2^{6}) + (\mathbf{1}x\ 2^{5}) + (\mathbf{1}x\ 2^{4}) +$

$(\mathbf{0}x\ 2^{2}) + (\mathbf{1}x\ 2^{2}) + (\mathbf{1}x\ 2^{1}) + (\mathbf{1}x\ 2^{0})$

Dari perhitungan di atas dapat disimpulkan bahwa $1335_{(10)} = 10100110111_{(2)}$. Atau bilangan desimal dari 1335 sama dengan bilangan biner 10100110111.

2.3.2 Konversi Bilangan Biner ke Desimal

Setelah mengetahui bagaimana cara mengonversi bilangan desimal ke bilangan biner, sekarang akan dibahas cara mengonversi bilangan biner ke bilangan desimal.

Untuk mengkonversi bilangan biner ke dalam bentuk desimal, kalikan bit dari yang paling kanan ke kiri secara berurutan dengan 2^0, 2^1, 2^2, 2^3, 2^4, dan seterusnya. Misalkan akan dikonversi suatu bilangan biner yaitu 1010011 ke bilangan desimal. Caranya adalah sebagai berikut:

$1010011_{(2)} = (\mathbf{1}x2^{0}) + (\mathbf{1}x2^{1}) + (\mathbf{0}x\ 2^{2}) + (\mathbf{0}x\ 2^{3}) + (\mathbf{1}x\ 2^{4}) + (\mathbf{0}x\ 2^{5}) + (\mathbf{1}x\ 2^{6})$

$= 1 + 2 + 0 + 0 + 16 + 0 + 64$

$= 83_{(10)}$

Jadi, $1010011_{(2)} = 83_{(10)}$. Dari contoh di atas dapat disimpulkan bahwa bilangan biner 1010011 sama dengan bilangan desimal dari 83.

2.4 *Finite State Automata*

Definisi 2.21

FSA dinyatakan oleh 5 tupel yaitu : M = (Q, $\sum$, δ , S, F) dengan:

Q = himpunan *state*

$\sum$ = himpunan simbol input/masukan

δ = fungsi transisi

S = state awal/initial state, S .Q

F = himpunan state akhir, F . Q

Himpunan state akhir (F) hanya menyatakan state yang diterima.

Contoh:

Misalkan terdapat petani (P), kambing (K), serigala (S), rumput (R). Terdapat sebuah perahu yang hanya bisa ditumpangi oleh dua orang. Bagaimana cara menyeberang dengan selamat, dengan syarat kambing tidak boleh satu perahu dengan runput dan serigala.?

Penyelesaian :

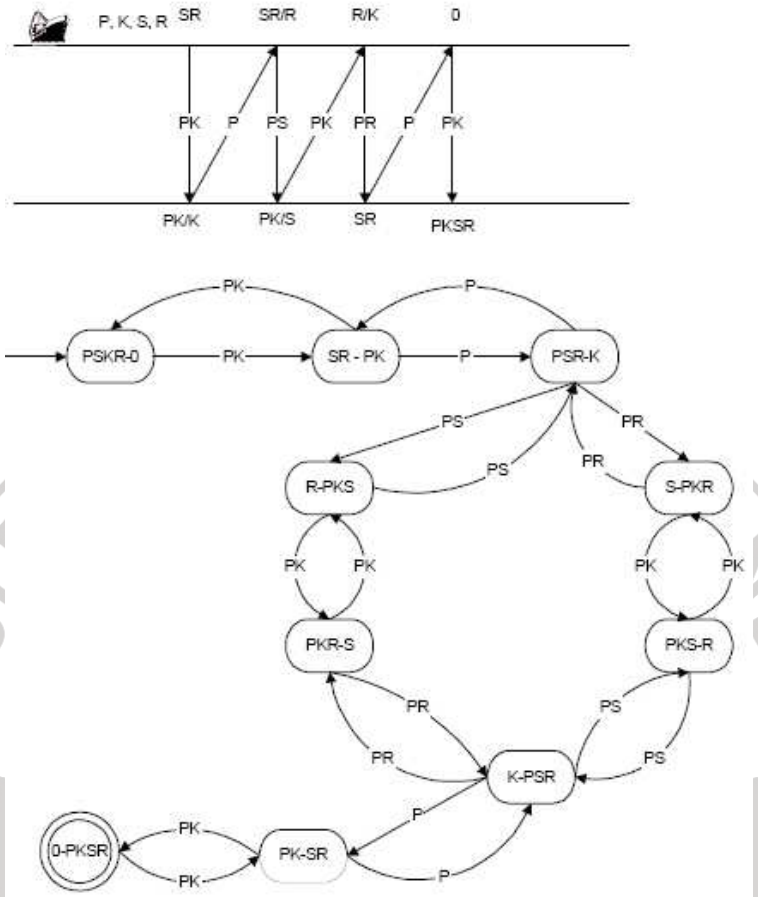
Q = {PKSR-Ø, SR-PK, PSR-K, R-PSK, S-PKR, PKR-S, PSK-R, K-PSR, PK-SR, Ø-PKSR}

$\sum$ = {P, K, S, R}

S = {PKSR-Ø}

F = {Ø-PKSR}

Ilustrasi Finite state aotumata ini dapat dilihat pada gambar (2.8)



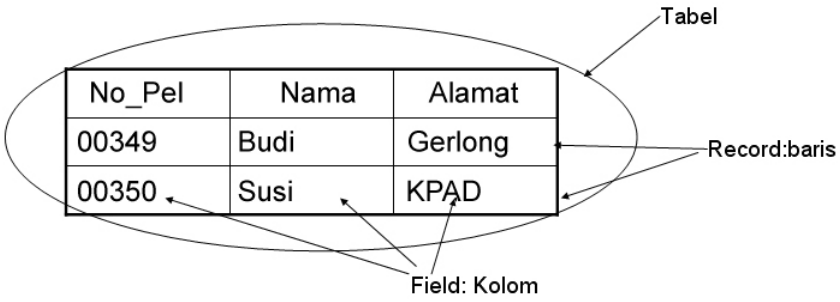
Gambar 2.5 Ilustrasi Finite State Automata

2.5 Basis Data

Basis data (*database*) adalah wadah sekelompok data yang disusun secara sistematis. Basis data pada prinsipnya digunakan untuk mengelola informasi terstruktur. Yang dimaksud informasi terstruktur adalah jenis informasi terdefinisi. Perangkat lunak yang digunakan untuk mengelola dan memanggil query basis data disebut managemen basis data atau dalam bahasa Inggris *Database Management Sistem*. Perangkat lunak yang sering digunakan dalam

basis data di antaranya *MS SQL Server*, *Microsoft Access*, *Oracle*, *Paradox*, *MySQL*, *Firebird* dan lain-lain.

Pada basis data seperti *MS SQL Server*, *Access* dan termasuk *Oracle*, data tersimpan dalam bentuk tabel-tabel. Setiap tabel terdiri dari kolom (*field*) dan baris (*record*). Gambar dibawah memperlihatkan bagaimana data pelanggan direpresentasikan dalam sebuah tabel.



Gambar 2.6 Ilustrasi Tabel pada Basis Data

Tabel data pelanggan di atas tersusun dari tiga *field* (kolom) yaitu No_Pel, Nama, Alamat dan dua *record* (baris) yaitu data pelanggan Budi dan Susi. Data yang berada pada kolom yang sama, harus memiliki jenis yang sama.

Query adalah fasilitas untuk melihat, mengganti dan menganalisis data dari berbagai sudut pandang. Hasil dari *query* dapat digunakan untuk *form* dan *report*. *SQL* (*Structure Query Language*) mempunyai peranan yang sangat penting dalam *RDBMS* (*Relational Database Management System*). *SQL* sesuai dengan namanya adalah sebuah bahasa. Bahasa ini telah menjadi standard semua *RDBMS* sebagai alat komunikasi dengan basisdata. Bahasa ini mendefinisikan cara untuk mengambil,menyisipkan, mengupdate dan menghapus data pada sebuah basisdata.

Ada beberapa jenis *query* yang sering digunakan, diantaranya:

1. *SQL* untuk mengambil data

   Query jenis ini adalah *query* yang paling sering digunakan.

   Sintaks *SQL* untuk mengambil data adalah sebagai berikut:

   SELECT [Fields yang diiginkan] FROM [nama tabel] WHERE [Kondisi]

2. *SQL* untuk Menghapus Data

   Sintaks *SQL* untuk menghapus data adalah sebagai berikut:

   SELECT [Fields yang diiginkan] FROM [nama tabel] WHERE [Kondisi]

3. *SQL* untuk Fungsi Agregasi

   Fungsi agregasi adalah fungsi yang Digunakan dalam sebuah kelompok record.

   Terdapat lima fungsi agregasi:

   1. AVG ( ) : Menghitung rata-rata
   2. MIN ( ): Menghitung nilai minimum
   3. MAX ( ): Menghitung nilai maksimum
   4. SUM ( ) : Menghitung jumlah
   5. COUNT (*): Menghitung jumlah record

   Beberapa contoh Query:

1. SELECT golongan_tarif, daya, COUNT(*) as jumlah, daya as jumlah_daya
   FROM rincian_rekening
   WHERE kode_satuan_daya = 'V'
   group by golongan_tarif,daya
2. SELECT golongan_tarif, daya, count(*) as jumlah, daya as jumlah_daya
   FROM rincian_rekening

```
WHERE kode_satuan_daya = 'V'

group by golongan_tarif,daya

having count(*) <100
```