Image Compression

- Kompresi ➔ untuk apa?
  - Volume data yang besar
  - Bit rate tinggi ➔ bandwidth yang tinggi
  - Bayangkan sebuah video dengan resolusi 640x480 dengan 30 fps, dimana menggunakan penyimpanan 24-bit. Bila video berdurasi 1 jam berapa ukuran file video tersebut?

Image Compression

- Alternatif Solusi
  - Penambahan storage dan bandwidth
  - Kompresi data (smart choice)

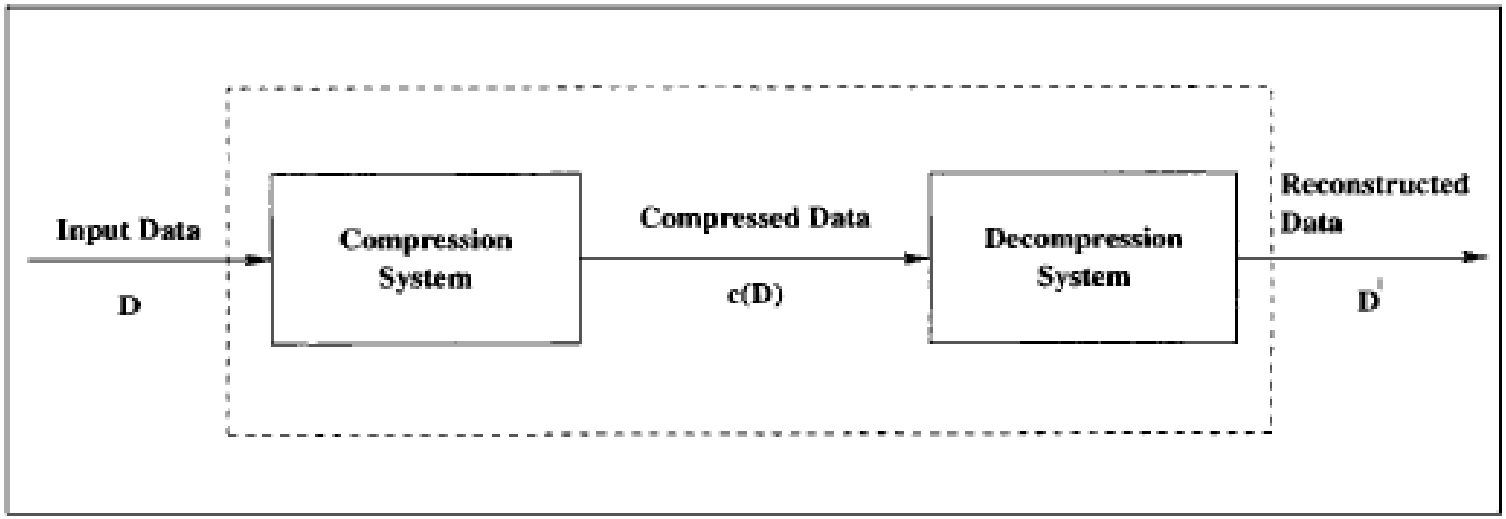


Image Compression

- Teknik kompresi yang diharapkan :
  - Proses kompresi/dekompresi yang cepat
  - Membutuhkan memory yang kecil
  - Kualitas citra kompresi yang bagus
  - Proses transfer dan penyimpanannya mudah

Image Compression

- Berdasarkan hasilnya, teknik kompresi ada 2 :
  - Lossless Compression
  - Lossy Compression
- Klasifikasi Teknik Kompresi :
  - Entropy Encoding (Lossless)
    - Run Length Encoding (RLE)
    - Pattern Substitution
    - Huffman
    - DPCM
  - Source Encoding (Lossy)
    - Quantizing Compression
    - Transfrom Encoding
  - Hybrid Encoding (Lossy)
    - JPEG

Quantizing Compression

- Teknik Quantizing Compression bersifat lossy
- Digunakan untuk mereduksi data dengan asumsi bahwa perubahan data tidak akan mempengaruhi content/informasi
- Melakukan pengkodean, rata-rata pada hitogram, menggunakan matrik kuantisasi

Quantizing Compression (kode)

2	9	6	4	8	2	6	3	8	5	9	3	7
3	8	5	4	7	6	3	8	2	8	4	7	3
3	8	4	7	4	9	2	3	8	2	7	4	9
3	9	4	7	2	7	6	2	1	6	5	3	0
2	0	4	3	8	9	5	4	7	1	2	8	3

- Histogram :
  - Warna 0 = 2
  - Warna 1 = 2
  - Warna 2 = 9

  } ***Dikodekan menjadi 0 (Jumlahnya 13 pixel)***

  - Warna 3 = 11
  - Warna 4 = 9

  } ***Dikodekan menjadi 1 (Jumlahnya 20 pixel)***

  - Warna 5 = 4
  - Warna 6 = 5
  - Warna 7 = 8

  } ***Dikodekan menjadi 2 (Jumlahnya 17 pixel)***

  - Warna 8 = 9
  - Warna 9 = 6

  } ***Dikodekan menjadi 3 (Jumlahnya 15 pixel)***

Quantizing Compression (kode)

- Bisakah data ini dibalikkan?

0	3	2	1	3	0	2	1	3	2	3	1	2
1	3	2	1	2	2	1	3	0	3	1	2	1
1	3	1	2	1	3	0	1	3	0	2	1	3
1	3	1	2	0	2	2	0	0	2	2	1	0
0	0	1	1	3	3	2	1	2	0	0	3	0

Quantizing Compression (matrik)

- Menggunakan matrik dan pembulatan

3588	-86	-27	-105	-2	1	44	19
-100	-79	24	34	45	-22	-14	-43
18	-64	-35	-20	-15	-17	21	15
-184	3	72	-17	0	-53	8	-13
14	9	44	11	6	-15	-3	-23
39	99	122	-25	-13	-18	-6	26
7	-116	-46	-93	30	35	-4	28
-53	-110	137	-56	-19	-3	-11	3

÷

11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31
19	21	23	25	27	29	31	33
21	23	25	27	29	31	33	35
23	25	27	29	31	33	35	37
25	27	29	31	33	35	36	39

=

326	-6	-1	-6	0	0	1	0
-7	-5	1	1	2	0	0	-1
1	-3	-1	0	0	0	0	0
-10	0	3	0	0	-1	0	0
0	0	1	0	0	0	0	0
1	4	4	0	0	0	12	0
0	-4	-1	-3	0	1	0	0
-2	-4	4	-1	0	0	0	0

Run Length Encoding (RLE)

1	2	1	1	1	1
1	3	4	4	4	4
1	1	3	3	3	5
1	1	1	1	3	3

- Diubah dalam bentuk sekuensial

  ➔ 1 2 1 1 1 1 1 3 4 4 4 4 1 1 3 3 3 5 1 1 1 1 3 3 = 24 byte

- Dihitung jumlah kemunculan data

  ➔ (1,1) (2,1) (1,5) (3,1) (4,4) (1,2) (3,3) (5,1) (1,4) (3,2)

- Data Kompresi

  ➔ 1 1 2 1 1 5 3 1 4 4 1 2 3 3 5 1 1 4 3 2 = 20 byte

Shannon's *Source Coding Theory*

- Pada umumnya teknik lossless akan melakukan proses penggantian simbol dengan suatu simbol biner
- Masalahnya:
  - Menentukan simbol biner sehingga data yang dikompresi menjadi lebih kecil dan dapat "dibalikkan" kembali => harus unik

Shannon's *Source Coding Theory*

Simbol	MODEL _A	MODEL_B	MODEL_C
a	00	0	0
b	01	10	1
c	10	110	00
d	11	111	01

- Misalkan
  - S = *aacabad*
- Model manakah yang dapat digunakan untuk encode S ??

Shannon's *Source Coding Theory*

- Misalkan sebuah data:
  - A={a1,a2,…,an}
- Probabilitas data :
  - P={p1,p2,…,pn}
- Dengan kedua informasi tersebut maka kita dapat memperkirakan *information content* dari suatu simbol
- Semakin besar probabilitas maka akan dikodekan dengan biner yang kecil

Shannon's *Source Coding Theory*

- *Information Content I(a) entropy* dari suatu simbol a dimana kita telah mengetahui probabilitasnya dapat dirumuskan sebagai:

$$I(a_i) = \log_2 \frac{1}{p(a_i)} = -\log_2 p(a_i).$$

Basis log 2 menyatakan bahwa informasi dinyatakan dalam bentuk biner

Shannon's *Source Coding Theory*

- Setelah mengetahui nilai *Information Content* suatu simbol, kita dapat menyatakan:
  - Suatu simbol a dengan probabilitas p(a) sebaiknya direpresentasikan dalam biner dengan $-\log_2 p(a)$ simbol
- Sehingga *entropy* seluruh data adalah:

$$E = \sum_{i=1}^{N} p(a_i) I(a_i) = -\sum_{i=1}^{N} p(a_i) \log_2 p(a_i).$$

Huffman Code's

- Dari Teori Shannon :
  - Sebuah source data dapat dikodekan dengan rata-rata panjang kode mendekati entropy dari source data
- Tahun 1952 D.A.Huffman mengajukan teknik encoding untuk menghasilkan panjang kode terpendek dari suatu source data dengan memanfaatkan probabilitasnya.

Huffman Code's

- Teknik ini dikenal dengan Huffman Code's dengan pertimbangan:
  - The more frequently occurring symbols can be allocated with shorter codewords than the less frequently occurring symbols
  - The two least frequently occurring symbols will have codewords of the same length, and they differ only in the least significant bit.

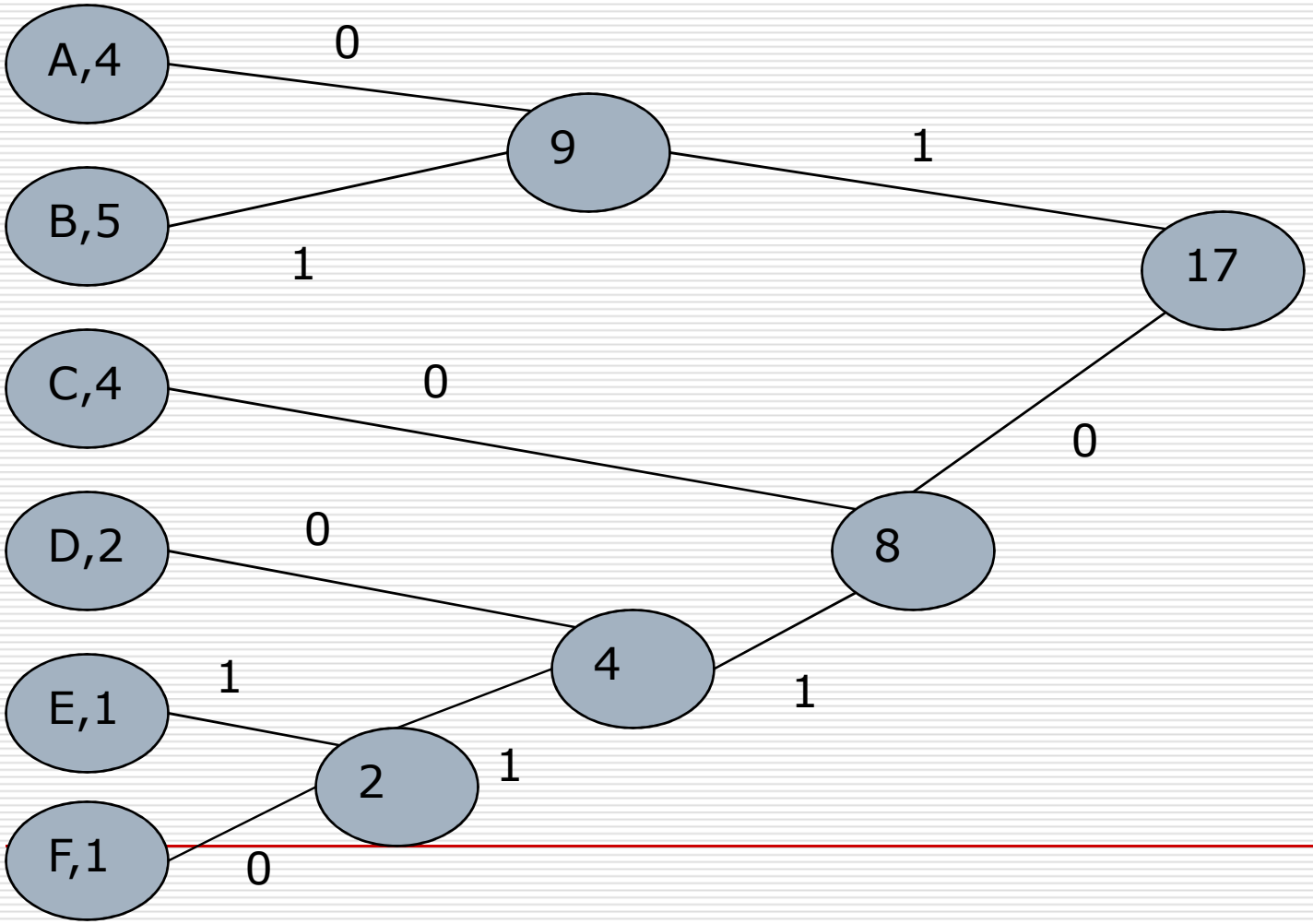
Huffman Code's

- Algoritma dalam pseudo code
  1. Hitunglah probabilitas dari setiap simbol yang ada
  2. Pasangkan setiap <simbol,probabilitas> dengan node
  3. Temukan 2 buah node dengan probabilitas terendah kemudian buatkah node parent dengan probabilitas gabungan dari 2 anaknya
  4. Berikan label untuk cabang dari anak ke parent dengan 0 dan 1 (sebaiknya konsisten)
  5. Update node (node anak diabaikan), lalu periksa jumlah node yang ada, bila jumlah node > 1 maka ulangi langkah 3
  6. Untuk menemukan kode setiap simbol, lakukan *traverse* dari root ke leaf, (label branch yang dilalui akan menjadi kode untuk leaf)

Huffman Code's

- Misalkan data:
  - S= AABAACCCCDDBBBBEF
- Hitung frekuensi data ≈ probalitas
  - a → 4, b → 5, c → 4, d → 2, e → 1, f → 1
- Proses pembangunan code sebagai berikut

Huffman Code's



Huffman Code's

- Kode untuk setiap simbol:
  - A -> 10 B -> 11 C -> 00
  - D -> 010 E- > 0111 F -> 0110
- Jadi data S= aabaaacccdddbbbbef (17 byte) akan dikodekan menjadi :
  - 10101110100000000010010111111101110 110 = 40 bit ≈ 5 byte

Huffman Code's

- Dalam implementasinya teknik Huffaman Code's dapat dipercepat dengan menggunakan mekanisme sorting data (insertion sort)
- Bagaimana mekanisme Decodenya??
- Coba buatlah code-matlab untuk perbaikan huffman encoding ☺

Arithmetic Coding

- Pada teknik ini, simbol yang ada akan direpresentasikan dengan bilangan real mulai dari 0-1. Konsekuensi?
- Aritmethic Coding menawarkan eficiency yang lebih baik dibandingkan dengan Huffman Code's
- Sesuai digunakan untuk jumlah simbol sedikit (binary simbol) dan simbol dengan highly skewed probabilities

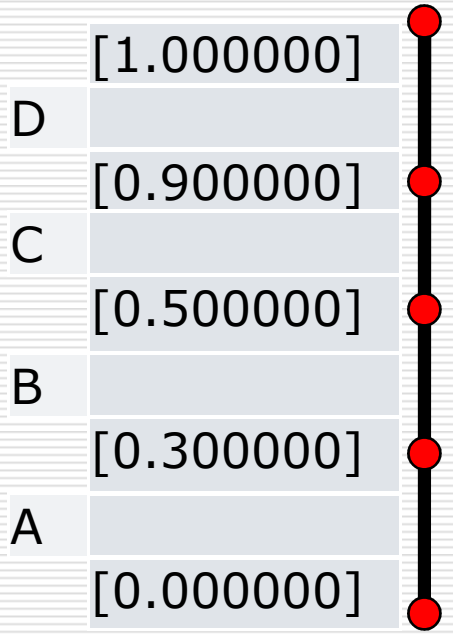
Arithmetic Coding [Encoding]

- Example penerapan arithmetic coding
- Informasi yang dimiliki

Sbl.	P(s)	Cumulative P(s)	Range Simbol
A	0.3	0.3	[0.000 , 0.300]
B	0.2	0.5	[0.300 , 0.500]
C	0.4	0.9	[0.500 , 0.900]
D	0.1	1	[0.900 , 1.000]

Arithmetic Coding [Encoding]

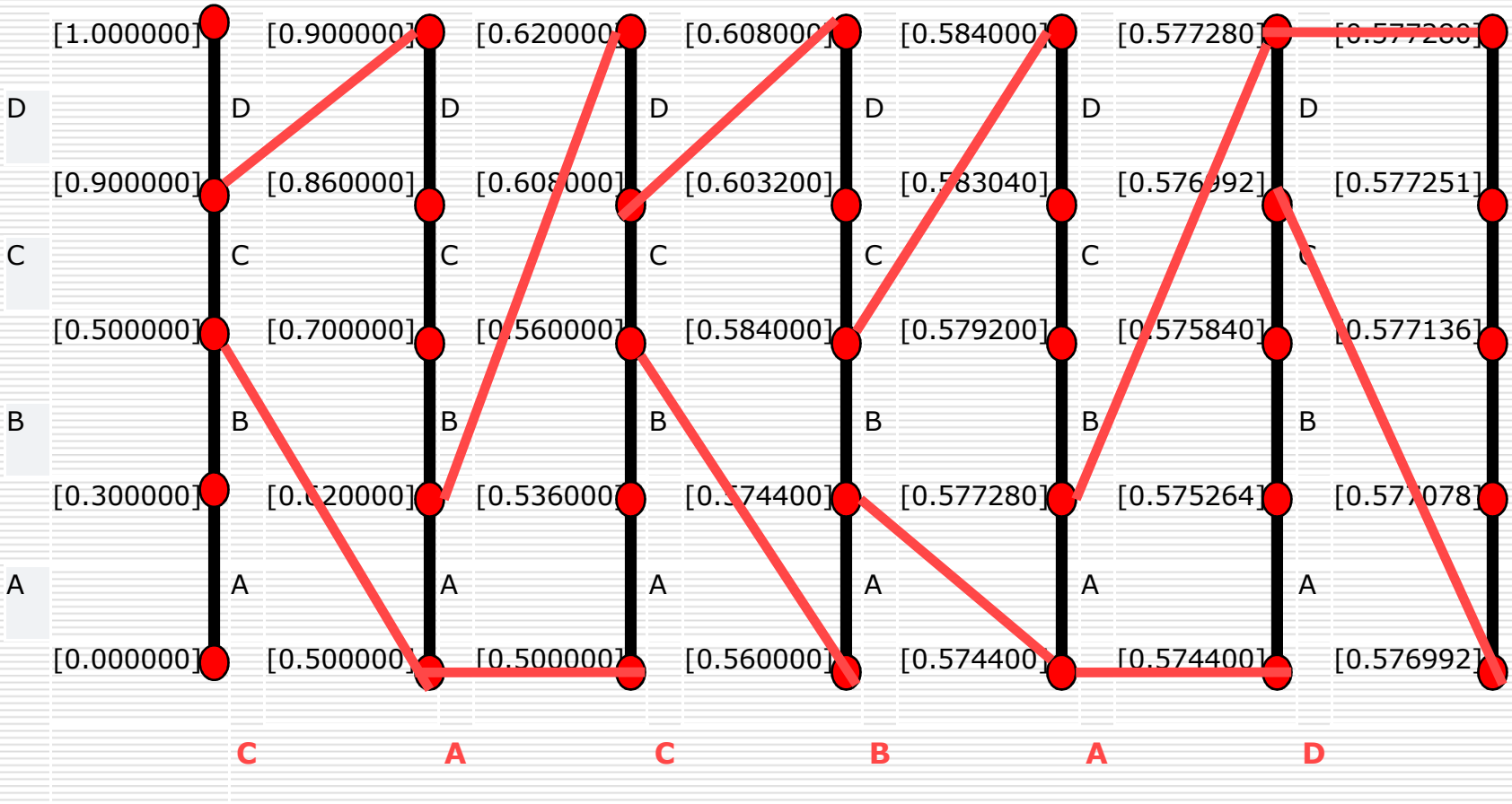
- Range Simbol Awal
  - Range simbol ini akan digunakan untuk menentukan bilangan yang mewakili data yang di kompresi
- Note: Pada umumnya hasil akhir proses encoding akan dikonversi dalam bentuk biner



Arithmetic Coding [Encoding]

- Pesan akan di encode menjadi sebuah bilangan beserta informasi range simbol
- Proses penghasilan dengan memasukkan pesan pada range simbol dan melakukan update range simbol
- Misalkan simbol : " C A C B A D"

Arithmetic Coding [Encoding]



Arithmetic Coding [Encoding]

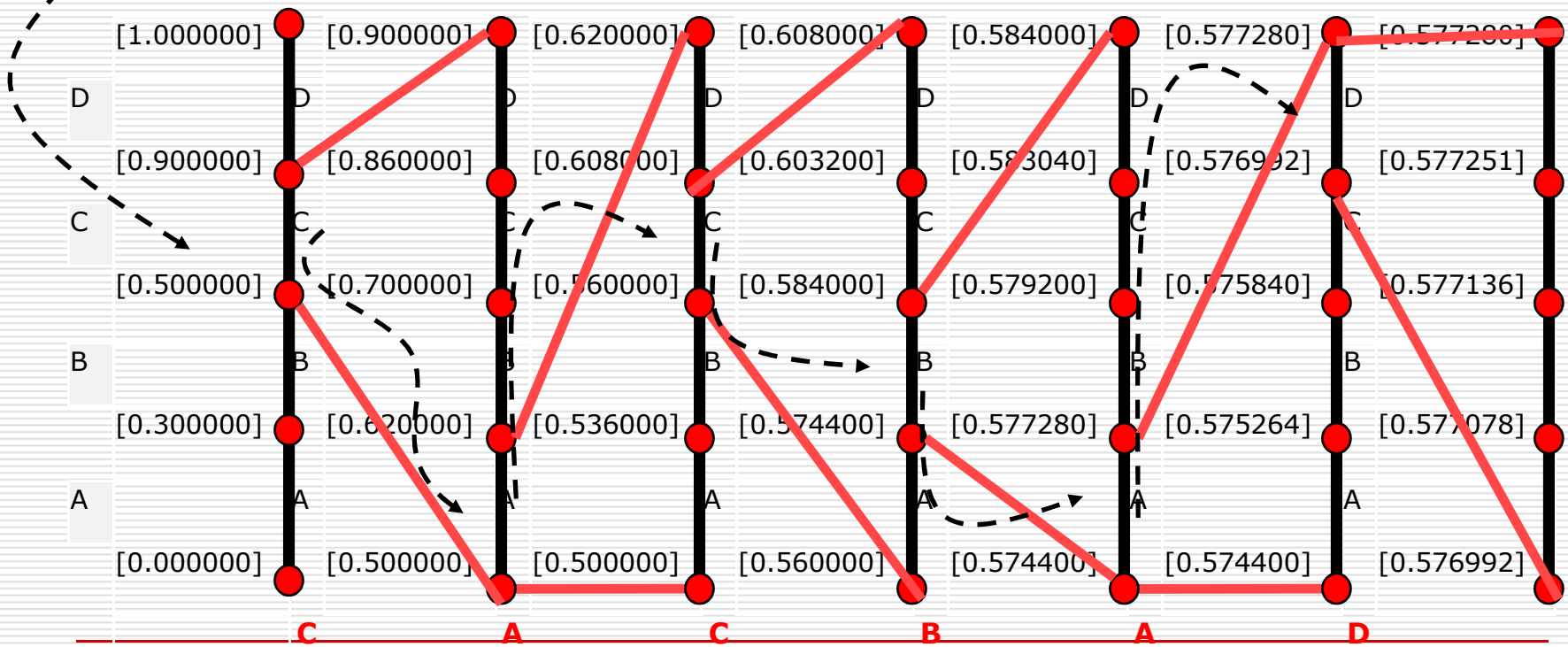
- Pada akhir proses kita akan memperoleh sebuah range simbol akhir, dalam kasus ini adalah [0.576992,0.57728]
- Pesan " C A C B A D" dapat kita kodekan menjadi suatu bilangan pada range terakhir
- Misalkan dengan midpoint interval maka pesan dikodekan menjadi 0.577136

Arithmetic Coding [Decoding]

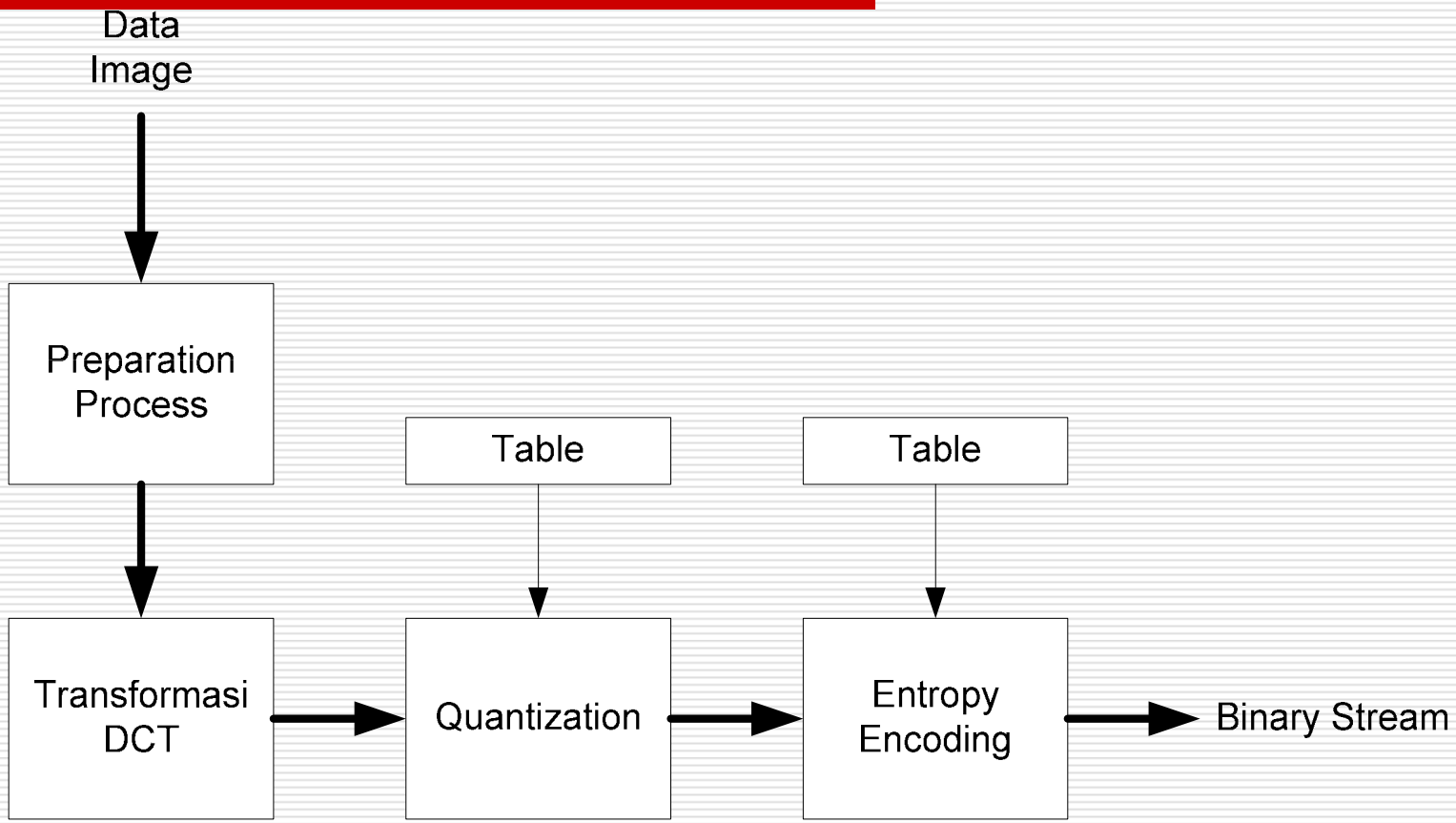
- Untuk melakukan Decoding kita membutuhkan range simbol awal
- Langkahnya adalah:
  - Cocokkan nilai data kode dengan range yang ada pada range simbol, lalu ekstrak kode yang sesuai dengan range simbol
  - Pecah range simbol sesuai dengan hasil pencocokan (Mengubah range simbol)

Arithmetic Coding [Decoding]

0.577136 (data yang akan di-decoding, di cocokkan dengan interval kemudian dilakukan pembagian interval seperti pada tahap encoding



JPEG **Joint Photographic Experts Group**



JPEG

1. Tahap Persiapan (*Preparation Process*)

Pada tahap ini dilakukan proses membagi citra menjadi blok 8x8

10	12	11	10	11	12	13	20
11	11	12	16	17	18	12	11
10	10	15	16	14	13	19	20
20	18	16	16	15	14	12	11
12	13	11	11	10	13	14	17
10	11	17	16	15	12	12	13
17	18	18	10	11	12	14	15
11	12	20	19	18	17	17	15

JPEG

2. Tranformasi DCT

- **Transformasi DCT bertujuan mengubah menghitung frekuensi-frekuensi pembentuk dari citra blok 8x8 dan memisahkan frekuensi rendah dan frekuensi tinggi dari hasil tranformasi DCT.**
- **Transformasi DCT terhadap blok 8x8 dapat dilakukan dengan rumus :**

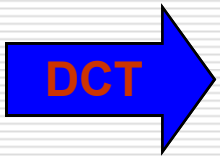
$$DCT(u,v) = \frac{1}{4}.C(u).C(v)\sum_{x=0}^{7}\sum_{y=0}^{7}.f(x,y).\cos\left[\frac{(2.x+1).u.\pi}{16}\right].\cos\left[\frac{(2.y+1).v.\pi}{16}\right]$$

Dimana : $C(z) = \begin{cases} \frac{1}{\sqrt{2}}, & z = 0 \\ 1, & z > 0 \end{cases}$

JPEG

2. Tranformasi DCT (cont.)

10	12	11	10	11	12	13	20
11	11	12	16	17	18	12	11
10	10	15	16	14	13	19	20
20	18	16	16	15	14	12	11
12	13	11	11	10	13	14	17
10	11	17	16	15	12	12	13
17	18	18	10	11	12	14	15
11	12	20	19	18	17	17	15



Frekuensi >>, Penting <<

3588	-86	-27	-105	-2	1	44	19
-100	-79	24	34	45	-22	-14	-43
18	-64	-35	-20	-15	-17	21	15
-184	3	72	-17	0	-53	8	-13
14	9	44	11	6	-15	-3	-23
39	99	122	-25	-13	-18	-6	26
7	-116	-46	-93	30	35	-4	28
-53	-110	137	-56	-19	-3	-11	3

Frekuensi >>, Penting <<

JPEG

3. Quantisasi

- **Proses Quantisasi bertujuan untuk menghilangkan nilai-nilai yang tidak penting (dalam hal ini nilai-nilai yang berada pada daerah frekuensi tinggi) pada matrix hasil dari Transformasi DCT.**

$$Quantized\_Value(i,j) = \frac{DCT\_Matrix(i,j)}{Quantum\_Matrix(i,j)}$$

JPEG

3. Quantisasi (Cont.)

3588	-86	-27	-105	-2	1	44	19
-100	-79	24	34	45	-22	-14	-43
18	-64	-35	-20	-15	-17	21	15
-184	3	72	-17	0	-53	8	-13
14	9	44	11	6	-15	-3	-23
39	99	122	-25	-13	-18	-6	26
7	-116	-46	-93	30	35	-4	28
-53	-110	137	-56	-19	-3	-11	3

÷

11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31
19	21	23	25	27	29	31	33
21	23	25	27	29	31	33	35
23	25	27	29	31	33	35	37
25	27	29	31	33	35	36	39

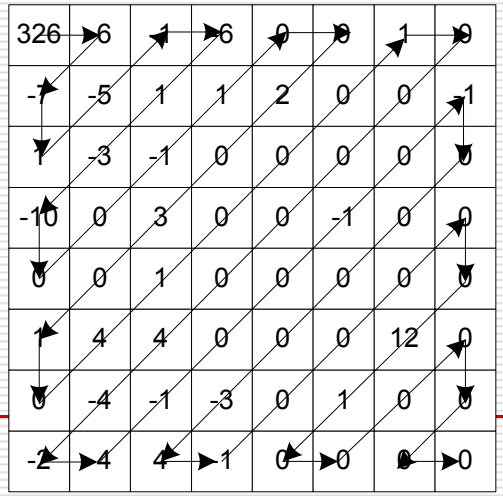
=

326	-6	-1	-6	0	0	1	0
-7	-5	1	1	2	0	0	-1
1	-3	-1	0	0	0	0	0
-10	0	3	0	0	-1	0	0
0	0	1	0	0	0	0	0
1	4	4	0	0	0	12	0
0	-4	-1	-3	0	1	0	0
-2	-4	4	-1	0	0	0	0

JPEG

4. Entropy Encoding

- **Entropy Encoding adalah teknik kompresi yang bersifat lossless. Tahap ini bertujuan untuk mengkompresi matrix hasil quantisasi, bisa menggunakn metode huffman atau RLE**
- **Proses Entropy Encoding terhadap hasil quantisasi di atas dengan pembacaan zig-zag :**



Hasil encoding jika menggunakan RLE :
326,-6,-7,1,-5,1,6,1,-3, [***0,3***] , -1,1,[***0,2***],2,[***0,1***],3, [***0,1***], 1, [***0,1***],4,1,[***0,3***],1,[***0,5***],4,-4,-2,4,-1,[***0,2***],-1,[***0,1***], -1, [***0,4***],-3,4,1,[***0,5***],12,1,[***0,7***] = 49 byte