Network & Media Laboratory

Fundamentals of Multimedia

Authored by Ze-Nian Li Mark S. Drew

Lecturer： Lu Dongming

Network & Media Laboratory

6、Lossless Compression Algorithms

Content

- Introduction and Basics of Information Theory
- Lossless Coding Algorithms
  - Run-Length Coding
  - Variable-Length Coding (VLC)
  - Dictionary-Based Coding
  - Arithmetic Coding
- Lossless Image Compression

Network & Media Laboratory

1、Introduction and Basics of Information Theory

1.1 Background

- More and more data into digital form
  - Libraries, museums, governments
  - To be stored without any loss
- For example -- To encode 120 million call numbers
  - Each item need a 27-bit number，$2^{27} > 120M$
  - Compression to reduce the number of bits needed
- Different data appear at different frequencies
  - To assign fewer bits to present more frequently appeared data
  - VLC－Variable-Length Coding
- Lossless coding
  - Both the compression and decompression processes induce no information loss

1.2 Data Compression Scheme

- **Definition: Compression Ratio**
  - **Compression ratio = B0 / B1**
    - **The number of bits before compression is B0**
    - **The number of bits after compression is B1**
- **Compression ratio must be larger than 1.0;**
  - **The higher the compression ratio, the better the lossless compression scheme**

A general data compression scheme

1.3 Basics of Information Theory

- The entropy of an information source
  - With alphabet S＝ {s1, s2, …, sn}
  - η＝H(S)＝ $\sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} = -\sum_{i=1}^{n} p_i \log_2 p_i$
  - p_i is the probability that symbol si in S will occur

- $\log_2 \frac{1}{p_i}$ indicate the amount of information contained in characters (Self-information)

1.3 Basics of Information Theory

- **For example**: the probability of n in a manuscript is 1/32, so
  - The amount of information is 5 bits
  - A character sting nnn require 15 bits to code

- **What is entropy**?
  - A measure of the disorder of a system
  - The more entropy, the more disorder

1.3 Basics of Information Theory

- **Examples:**
  - **Suppose a system has 4 states outcome, each outcome has probability 1/4:**

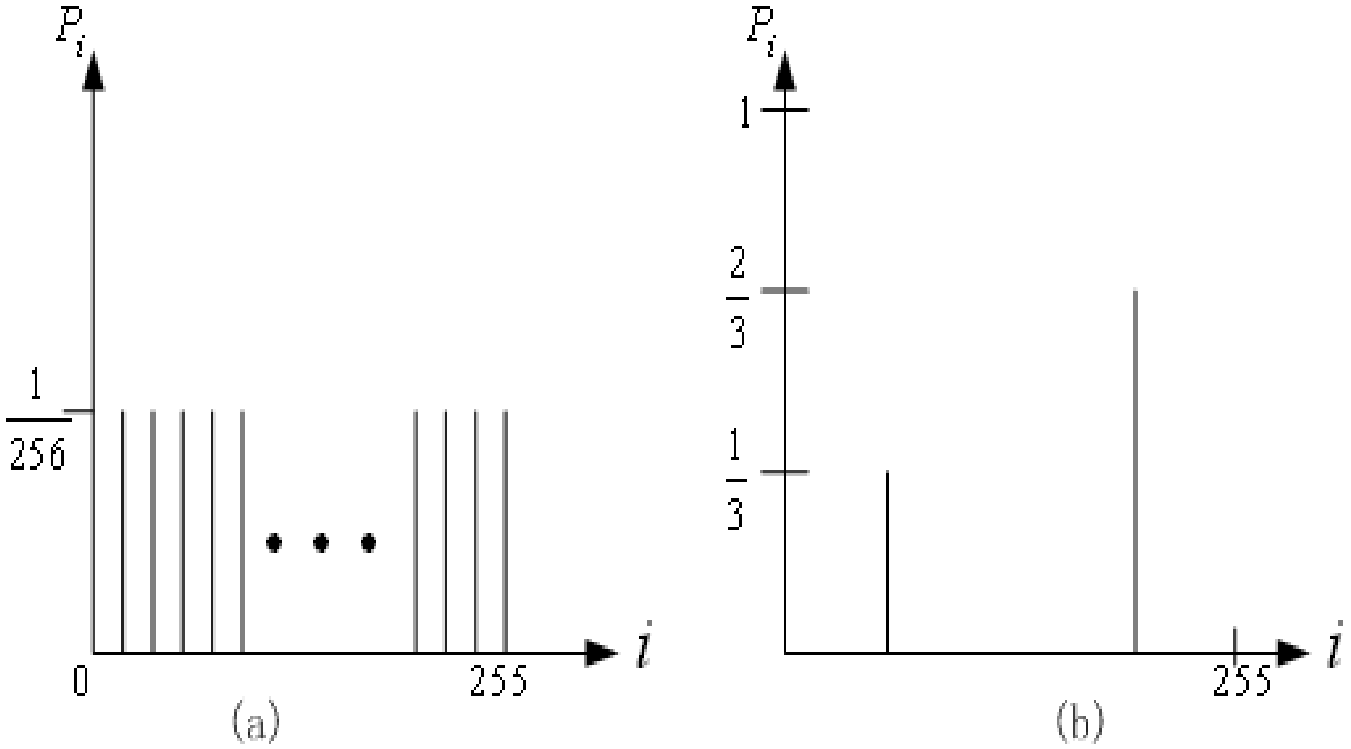
$$4 \times (1/4) \times \log_2 (1/(1/4)) = 2 bits$$

  - **If one state had probability ½,the other three had probability 1/6:**

$$\frac{1}{2} \times \log_2 2 + 3 \times \frac{1}{6} \times \log_2 6 = 1.795 < 2 bits$$

  - **The most-occurring one means fewer bits to send**
- **The definition of entropy -- identifying often-occurring symbols as short *codewords***
  - **Variable-Length Coding**

1.3 Basics of Information Theory



Histograms for two gray-level images

1.3 Basics of Information Theory

- **The entropy of the two above images**
  - **The entropy of image a is:**

$$\eta = \sum_{i=0}^{255} \frac{1}{256} \cdot \log_2 256 = 8$$

  - **The entropy of image b is:**

$$\eta = \frac{1}{3} \cdot \log_2 3 + \frac{2}{3} \cdot \log_2 \frac{3}{2}$$

$$= 0.33 \times 1.59 + 0.67 \times 0.59 = 0.92$$

- **The entropy is greater when the probability is flat and smaller when it is more peaked.**

Network & Media Laboratory

2、 Lossless Coding Algorithms

2.1 Run-Length Coding

- RLC（RUN-LENGTH CODING）
  - One of the simplest forms of data compression
- Basic idea：
  - If symbols of information source tend to form continuous groups, Code one such symbol and the length of the group instead of coding each symbol individually
- Example： a bi-level image can be efficiently coded using RLC
- Two-Dimensional RLC is usually used to code bi-level image.

2.2 Variable-Length Coding

- **Basic idea**
  - **Entropy indicates the information content in an information source**
- **VLC is one of the best-know entropy coding methods**
  - **Shannon-Fano algorithm**
  - **Huffman coding**
  - **Adaptive Huffman coding**

2.2 Variable-Length Coding

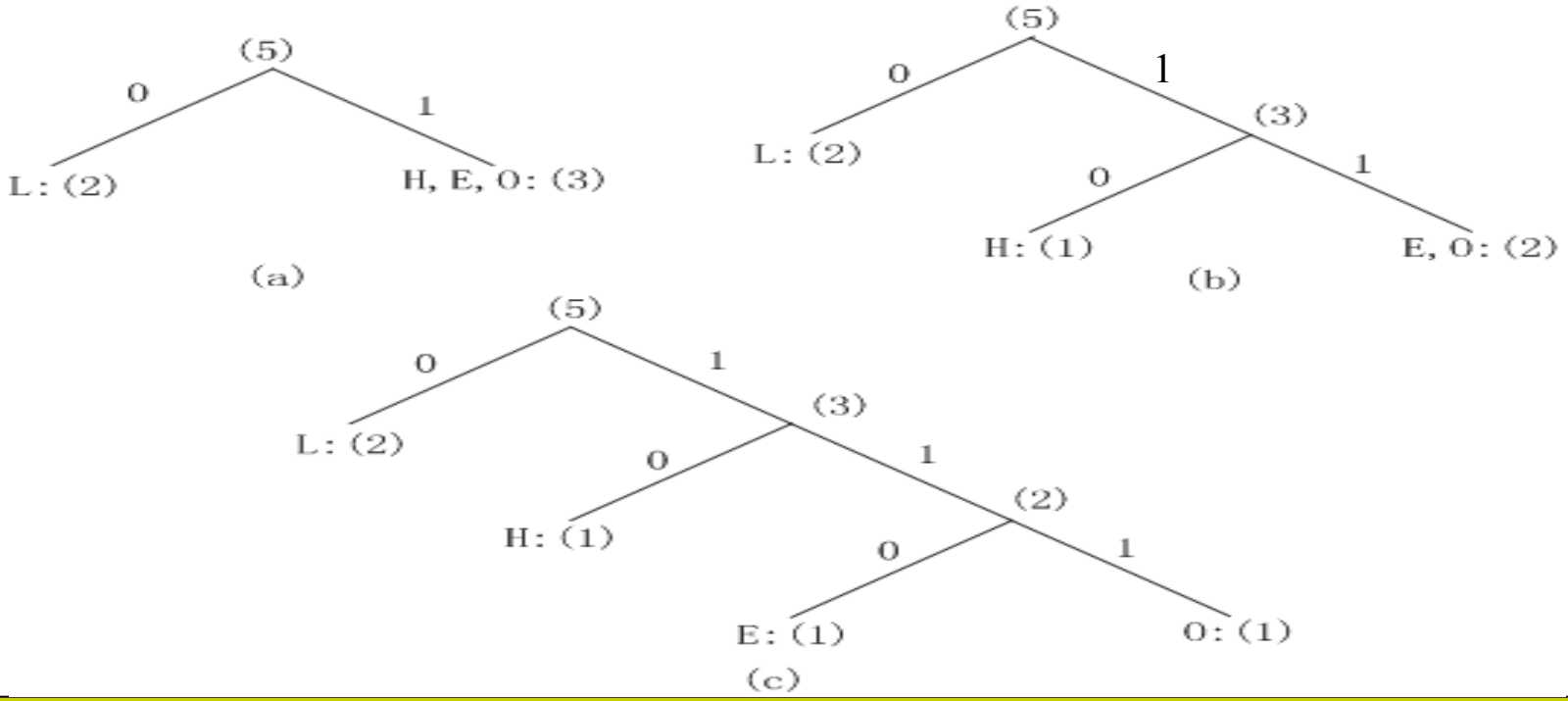
Shannon-Fano Algorithm

- Developed by Shannon at Bell Labs and Robert Fano at MIT
- The Top-Down manner
  - Sort the symbols according to the frequency count of their occurrences
  - Recursively divide the symbols into two parts, each with approximately the same number of counts, until all parts contain only one symbol
- A way of implementing the above procedure is to build a binary tree

2.2 Variable-Length Coding

Example: Hello

Symbol	H E L O
Count	1 1 2 1



2.2 Variable-Length Coding

- The entropy of the example：

  $0.4\times1.32+0.2\times2.32+0.2\times2.32+0.2\times2.32$

  $=1.92$

- One result of performing the S-F algorithm on "Hello"： average bits 10/5=2

symbol	Count	$\text{Log}_2 P_i^{-1}$	Code	Number of bits used
L	2	1.32	0	2
H	1	2.32	10	2
E	1	2.32	110	3
O	1	2.32	111	3
Total number of bits： 10				

2.2 Variable-Length Coding

Huffman Coding

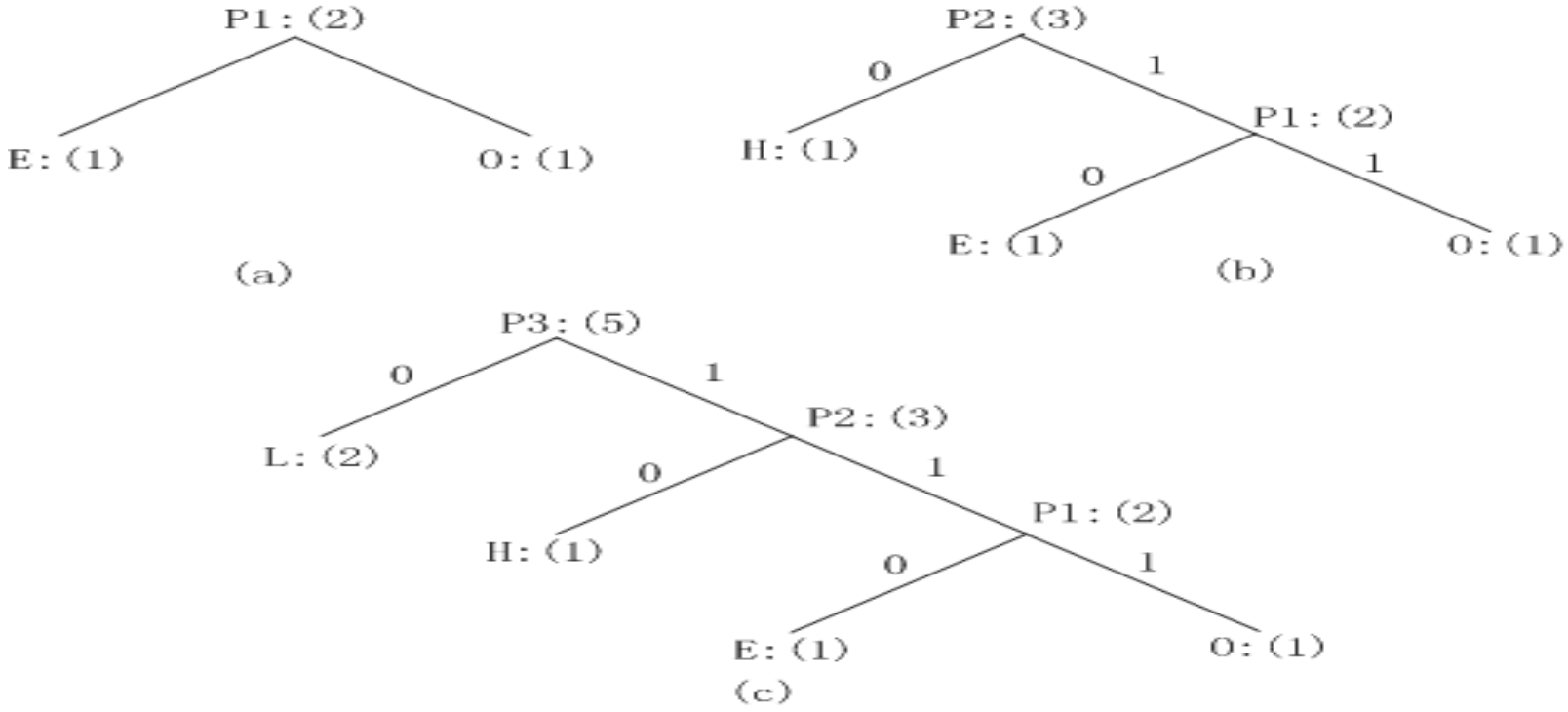
- First presented by David A. Huffman in 1952
- Adopted in applications, Such as fax、JPEG、MPEG

- A bottom-up manner:
  - Initialization: put all symbols on the list sorted according to their frequency counts
  - Repeat until the list has only one symbol left:
    - From the list, pick two symbols with the lowest frequency counts, form a Huffman sub-tree that has these two symbols as child nodes and create a parent node for them
    - Assign the sum of the children's frequency counts to the parent and insert it into the list, such that the order is maintained.
    - Delete the children from the list
  - Assign a codeword for each leaf based on the path from the root

2.2 Variable-Length Coding

Example: Huffman Coding "Hello"

Symbol	H	E	L	O
Count	1	1	2	1



2.2 Variable-Length Coding

- For above example, Huffman coding generate the same coding result as Shannon-Fano algorithm

- Another example
  - A:(15), B:(7), C:(6), D:(6) and E:(5)
  - Shannon-Fano needs 89bits；
  - Huffman needs 87 bits
- Conclusions
  - If correct probabilities are available，Huffman coding produces good compression results.
  - Important properties:
    - Unique prefix property
    - Optimality
- Extended Huffman coding

2.2 Variable-Length Coding

Adaptive Huffman Coding

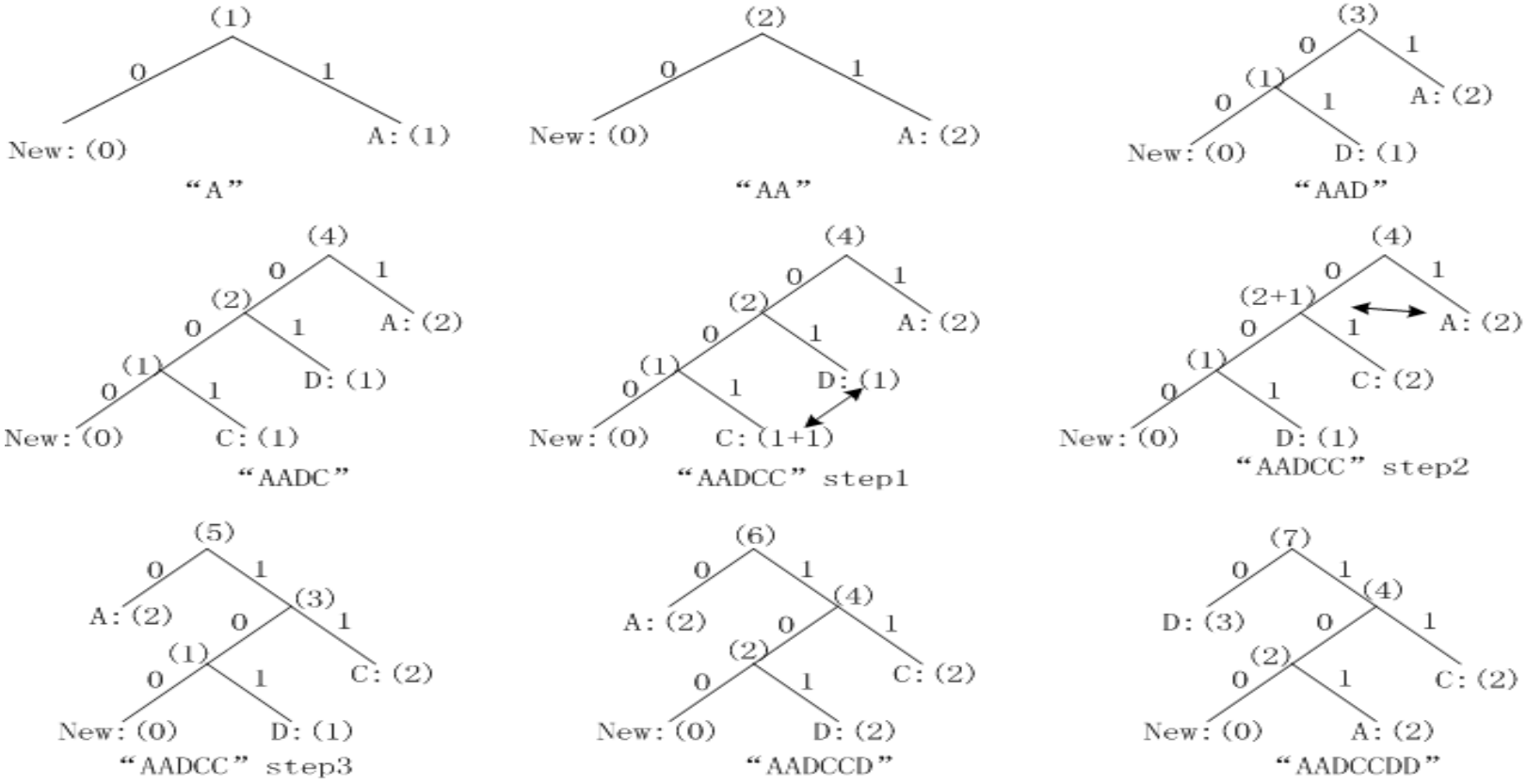
- Huffman Coding requires prior statistical knowledge about the information source, which is often not available
- Even when the statistics are available, the transmission of the symbol table could represent heavy overhead
- Adaptive algorithm, statistics are gathered and updated dynamically as the data-stream arrives
- The probabilities are no longer based on prior knowledge but on actual data received so far

2.2 Variable-Length Coding

- **Basic idea of Adaptive Huffman Coding**
  - **Intial_code: assigns symbols with some initially agreed-upon codes;**
  - **Update_Tree: constructing an adaptive Huffman tree**
    - **Increment the frequency counts for the symbols;**
    - **Update the configuration of the tree.**
  - **The encoder and decoder must use exactly the same intial_code and Update_Tree routines.**

2.2 Variable-Length Coding

- **An example: "AADCCDD"**



2.3 Dictionary-Based Coding

- First proposed by Ziv and Lempel in 1977 and 1978 respectively
- Terry Welch improved the technique in 1984
- Lempel-Ziv-Welch algorithm (called **LZW compression**)
- It is used in e.g., UNIX *compress*, GIF, V.42 bis for modems

2.3 Dictionary-Based Coding

- **LZW Compression Algorithm**

```
w = NIL;
while ( read a character k ) {
 if wk exists in the dictionary
      w = wk;
 else
      add wk to the dictionary;
      output the code for w;
      w = k;
 }
```

Suppose a dictionary contains 4,096 entries, with the first 256(0~255) entries being ASCII codes.

2.3 Dictionary-Based Coding

Example:

Input string:

"^WED^WE

^WEE^WEB

^WET".

w	k	Output	Index	Symbol
NIL	^			
^	W	^	256	^W
W	E	W	257	WE
E	D	E	258	ED
D	^	D	259	D^
^	W			
^W	E	256	260	^WE
E	^	E	261	E^
^	W			
^W	E			
^WE	E	260	262	^WEE
E	^			
E^	W	261	263	E^W
W	E			
WE	B	257	264	WEB
B	^	B	265	B^
^	W			
^W	E			
^WE	T	260	266	^WET
T	EOF	T		

2.3 Dictionary-Based Coding

- LZW Decompression

```
w = NIL;
 while ( read a character k ) { /* k could be a character or a code. */
       entry = dictionary entry for k;
       output entry;
     if( s!=NIL)
       add w + entry[0] to dictionary with a new code;
       w = entry;
}
```

2.3 Dictionary-Based Coding

Example (continued):

Input string:

"^WED<256>E<260><261><257>B<260>T".

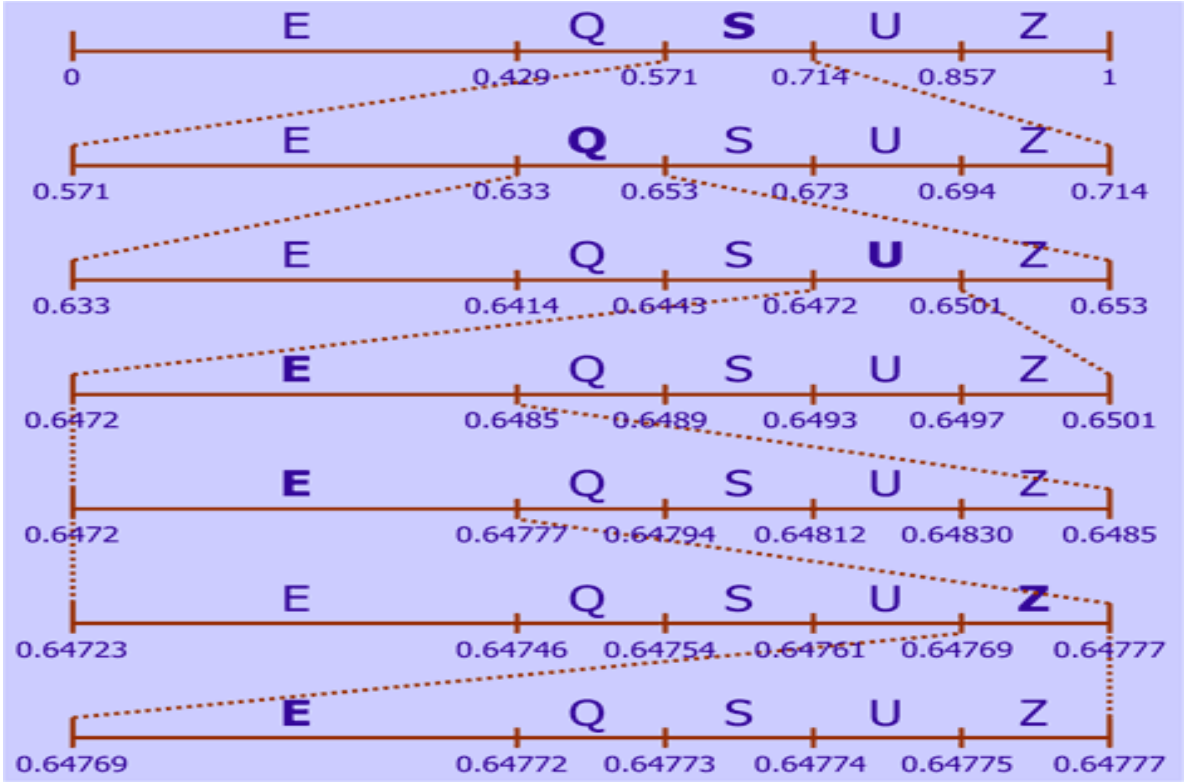
w	k	Output	Index	Symbol
	^	^		
^	W	W	256	^W
W	E	E	257	WE
E	D	D	258	ED
D	<256>	^W	259	D^
<256>	E	E	260	^WE
E	<260>	^WE	261	E^
<260>	<261>	E^	262	^WEE
<261>	<257>	WE	263	E^W
<257>	B	B	264	WEB
B	<260>	^WE	265	B^
<260>	T	T	266	^WET

2.4 Arithmetic Coding

- **Basic idea**
  - **Instead of present each character as a codeword, Arithmetic Coding represent the whole message by a half-open interval [a,b) contained in [0,1].**
  - **The length of the interval [a,b) equals the probability of the message. Choose a decimal in [a,b) and transform it into binary form as coding output.**
  - **Each character can shorten the interval, so the more characters ,the more shorter the interval will be.**
  - **As the interval become shorter, more bits are needed to present the interval**
- **Average bits used for each character can be decimal**

2.4 Arithmetic Coding

- **Example: "SQUEEZE"**



Network & Media Laboratory

3、 Lossless Image Compression

3.1 Lossless Image Compression

- **Differential coding**
  - One of the most commonly used compression techniques in multimedia data compression

- The basic of data reduction in differential coding
  - Existing redundancy in consecutive symbols in a data-stream

3.2 Differential Coding of Images

Given an original image I (x, y), defining a difference image d (x, y)

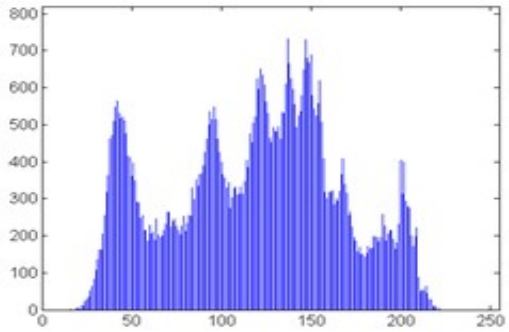
- **Using a simple difference operator**
  - **d(x,y)=I(x,y)-I(x-1,y)**
- **Discrete 2D Laplacian operator**
  - **d(x,y)＝4I(x,y)-I(x,y-1)-I(x,y+1)-I(x+1,y)-I(x-1,y)**

- **Image I has larger entropy than image d**
  - **VLC -- shorter bit-length for the difference image**
  - **Compression works better on a difference image**

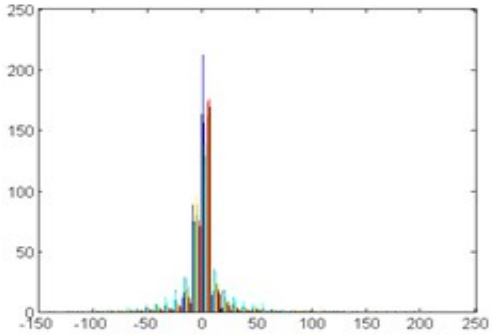
3.2 Differential Coding of Images

(a)

(b)



(c)



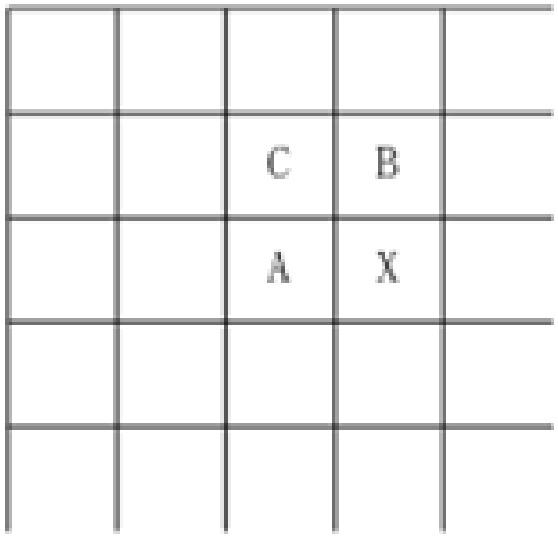
(d)

3.3 Lossless JPEG

- **Lossless JPEG**
  - **Special case of JPEG image compression which has no lossy steps**
- **Involves two steps: forming a differential prediction and encoding**
  - **Predictor：combines the values of up to three neighboring pixels as the predicted value for the current pixel**
  - **Encoder：Compares the prediction with the actual pixel and encodes the difference using lossless compression algorithm**
    - **Lossless JPEG usually yields a relatively low compression ratio, which renders it impractical for most multimedia applications**

3.3 Lossless JPEG

- **Lossless JPEG Predictors**



Predictor	Prediction
P1	A
P2	B
P3	C
P4	A+B-C
P5	A+(B-C)/2
P6	B+(A-C)/2
P7	(A+B)/2

3.3 Lossless JPEG

- Comparison of Lossless JPEG with other lossless compression programs

Compression program	Compression ratio			
	Lena	Football	F-18	Flower
Lossless JPEG	1.45	1.54	2.29	1.26
Optimal lossless JPEG	1.49	1.67	2.71	1.33
Compress(LZW)	0.86	1.24	2.21	0.87
gzip(LZ77)	1.08	1.36	3.10	1.05
gzip-9(optimal LZ77)	1.08	1.36	3.13	1.05
pack (Huffman coding)	1.02	1.12	1.19	1.00

The End!

Thanks !