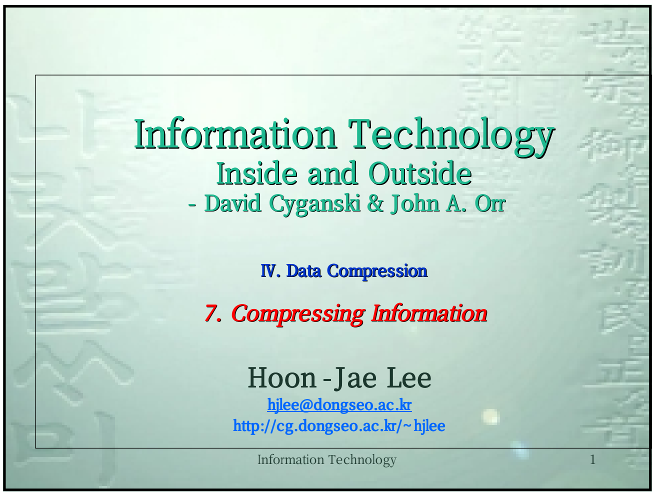


Information Technology

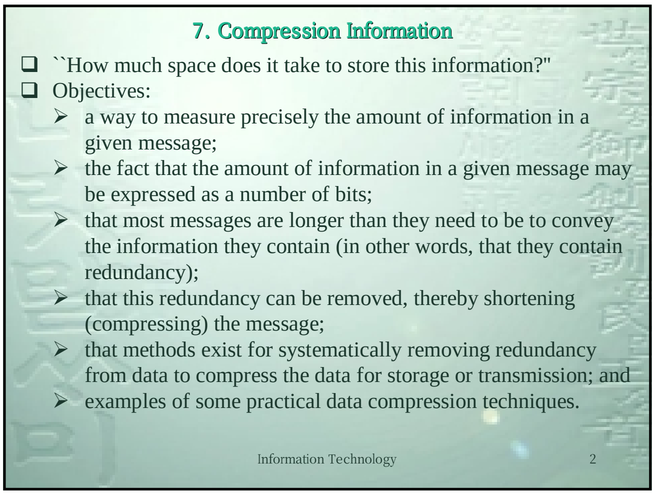
Inside and Outside

- David Cyganski & John A. Orr

IV. Data Compression

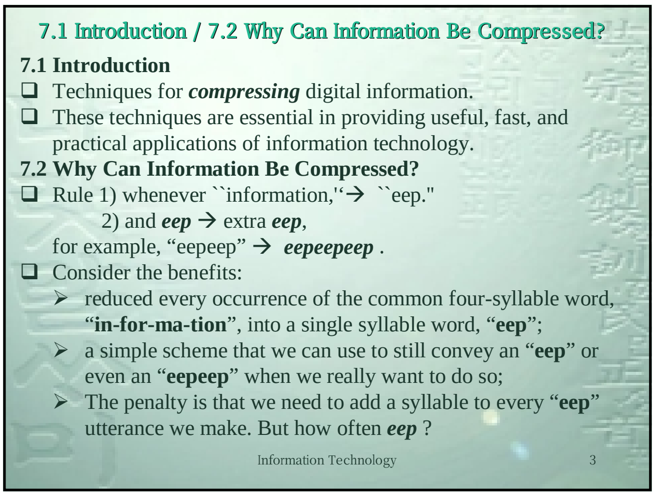
7. Compressing Information

Hoon-Jae Lee
hjlee@dongseo.ac.kr
http://cg.dongseo.ac.kr/~hjlee



7. Compression Information

- ``How much space does it take to store this information?"
- Objectives:
  - a way to measure precisely the amount of information in a given message;
  - the fact that the amount of information in a given message may be expressed as a number of bits;
  - that most messages are longer than they need to be to convey the information they contain (in other words, that they contain redundancy);
  - that this redundancy can be removed, thereby shortening (compressing) the message;
  - that methods exist for systematically removing redundancy from data to compress the data for storage or transmission; and
  - examples of some practical data compression techniques.



7.1 Introduction / 7.2 Why Can Information Be Compressed?

7.1 Introduction

- Techniques for ***compressing*** digital information.
- These techniques are essential in providing useful, fast, and practical applications of information technology.

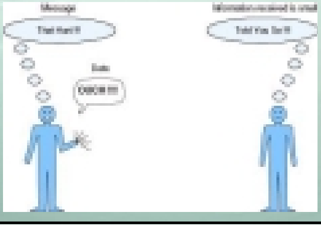
7.2 Why Can Information Be Compressed?

- Rule 1) whenever ``information,'‘→ ``eep."
  2) and ***eep*** → extra ***eep***,
  for example, “eepeep” → ***eepeepeep*** .
- Consider the benefits:
  - reduced every occurrence of the common four-syllable word, “**in-for-ma-tion**”, into a single syllable word, “**eep**”;
  - a simple scheme that we can use to still convey an “**eep**” or even an “**eepeep**” when we really want to do so;
  - The penalty is that we need to add a syllable to every “**eep**” utterance we make. But how often ***eep*** ?

7.3 Messages, Data, and Information

- Efficient storage and transmission of information in the form of digital data comes about by removing ***redundancy***.
- Redundancy
  - 1) some sequence of data bits that conveys the same message as another different sequence
    → briefer (less redundant) data representation
  - 2) a kind of redundancy that applies to the message itself

Figure 7.1: There is little information in a message that is expected.



7.3 Messages, Data, and Information(2)

- ***a priori knowledge*** includes the fact that we have a high likelihood of receiving a certain known message.
- How do we find schemes in other cases where the redundancy in the message is less obvious?
- What if we don't know anything about the message beforehand? Can we still compress it?
- These answers are contained within a field of study known as ***information theory***.
- Information theory is an area of mathematics that finds many applications in electrical and computer engineering.
- an amount of data that on average is ***equal to the information content measure***
- the smallest number of bits that can be transmitted to still convey the original message content.

7.4 Information Theory

- In July and October of 1948, a pair of papers were published by Claude E. Shannon of Bell Laboratories. This work created a new field at the intersection of mathematics and electrical communications theory, ***information theory***, and forever shaped the means and mathematics of information transmission, compression, and coding.
  - Claude E. Shannon, A mathematical theory of communication, ***Bell System Technical Journal***, Vol. 27, July, 1948, pages 379-423 and October, 1948, pages 623-656.
- The crux of information theory is the realization that the information content of a stream (that is, a sequence) of messages is connected directly with the probability of appearance of each possible message

7.4 Information Theory(2)

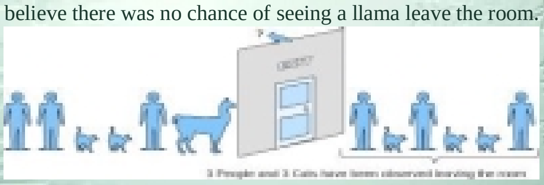
7.4.1 A Little Probability

- ❑ a ***probability of zero***
- ❑ a ***probability of one*** (or 100%)
- ❑ The probability that an event will occur takes on values anywhere **from zero to one**. The best way to understand the meaning of a statement such as **``this event has a probability of 1/4, or 0.25, or 25%,**'' is to understand how one would determine this value of probability by observations of events.
- ❑ The determination of probabilities by observation is an exercise of an area of mathematics called **statistics**.
- ❑ an estimate of the probability of each of these events by a statistical calculation of what is called the ***relative frequency*** of each event. Relative frequency is simply the fraction of times that a specified event occurred, relative to the total number of trials of the experiment (in this case, coin tosses).

7.4 Information Theory(3)

7.4.1 A Little Probability(2)

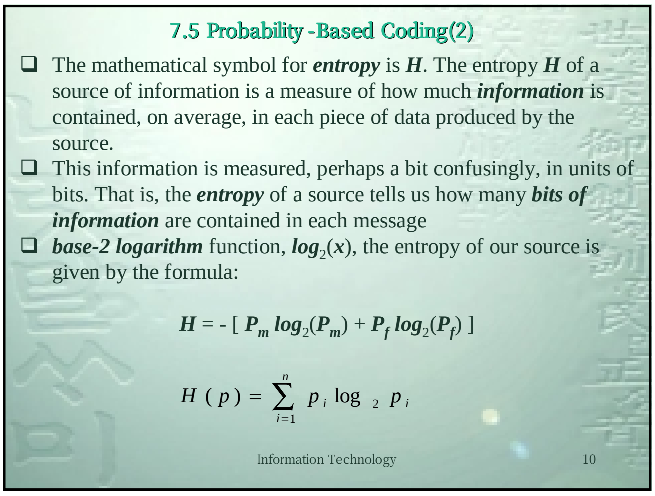
- ❑ ***fair*** coin.
- ❑ **Figure 7.2:**Statistics from past observations would have led us to believe there was no chance of seeing a llama leave the room.



- ❑ ***Independent events:*** Events are said to be ***independent*** if the occurrence of one has no influence on the occurrence of the other, and vice versa. For example, the probability that it will rain today is ***not*** independent of the probability of rain yesterday or tomorrow, because rainstorms often last two or three days and hence the events are linked.

7.5 Probability -Based Coding

- ❑ Suppose we have a source of binary data--for example, a transaction sat a point of sale POS terminal (electronic cash register) at a drugstore that notes and transmits the gender of each patron to a customer research database.
  - ➢ Numbers of male customers, $\boldsymbol{N_m}$, and
  - ➢ Numbers of female customers, $\boldsymbol{N_f}$.
  - ➢ the probabilities of a customer being male, $\boldsymbol{P_m}$, and female, $\boldsymbol{P_f}$, are both equal to 1/2.
  - ➢ using a 0 for each male patron and a 1 for each female patron.
  - ➢ How much information do we expect this stream of data to convey in the future? Shannon's theory showed that the average information content of a message stream, which is known as the ***entropy*** of the source of information, can be calculated.

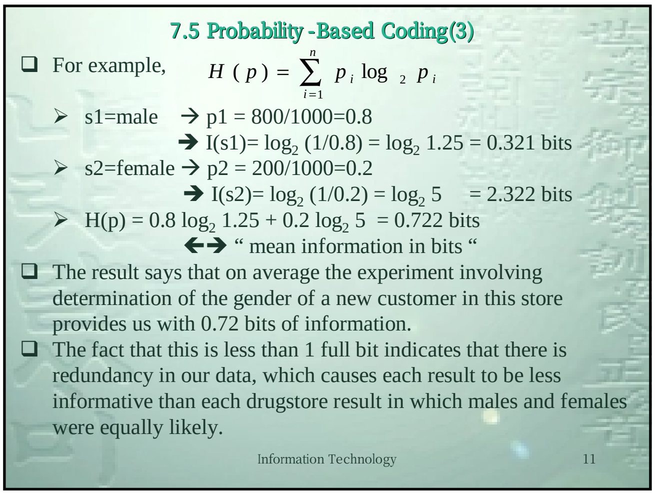


7.5 Probability -Based Coding(2)

- ❑ The mathematical symbol for ***entropy*** is ***H***. The entropy ***H*** of a source of information is a measure of how much ***information*** is contained, on average, in each piece of data produced by the source.
- ❑ This information is measured, perhaps a bit confusingly, in units of bits. That is, the ***entropy*** of a source tells us how many ***bits of information*** are contained in each message
- ❑ ***base-2 logarithm*** function, $\boldsymbol{log_2(x)}$, the entropy of our source is given by the formula:

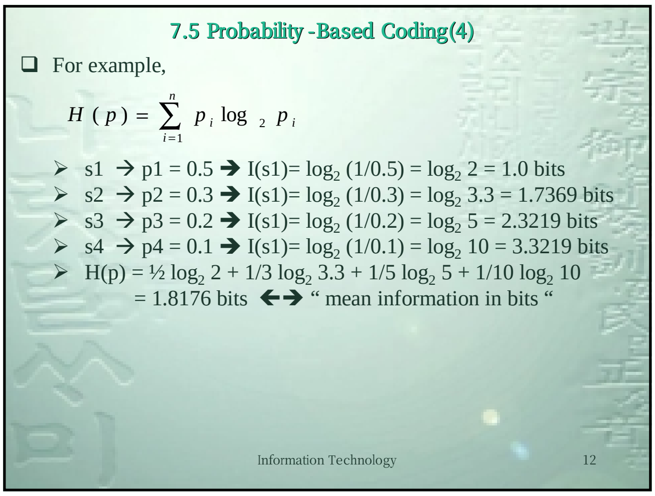
$$H = -\,[\, P_m \log_2(P_m) + P_f \log_2(P_f)\,]$$

$$H(p) = \sum_{i=1}^{n} p_i \log_2 p_i$$



7.5 Probability -Based Coding(3)

- ❑ For example, $H(p) = \sum_{i=1}^{n} p_i \log_2 p_i$
  - ➢ s1=male → p1 = 800/1000=0.8
    → $I(s1)= \log_2 (1/0.8) = \log_2 1.25 = 0.321$ bits
  - ➢ s2=female → p2 = 200/1000=0.2
    → $I(s2)= \log_2 (1/0.2) = \log_2 5 = 2.322$ bits
  - ➢ $H(p) = 0.8 \log_2 1.25 + 0.2 \log_2 5 = 0.722$ bits
    ←→ “ mean information in bits “
- ❑ The result says that on average the experiment involving determination of the gender of a new customer in this store provides us with 0.72 bits of information.
- ❑ The fact that this is less than 1 full bit indicates that there is redundancy in our data, which causes each result to be less informative than each drugstore result in which males and females were equally likely.



7.5 Probability -Based Coding(4)

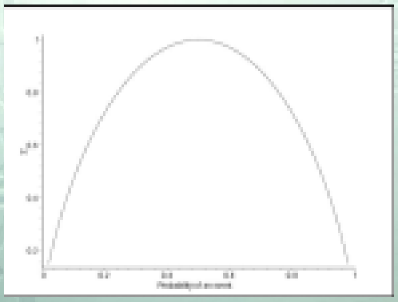
- ❑ For example,

$$H(p) = \sum_{i=1}^{n} p_i \log_2 p_i$$

  - ➢ s1 → p1 = 0.5 → $I(s1)= \log_2 (1/0.5) = \log_2 2 = 1.0$ bits
  - ➢ s2 → p2 = 0.3 → $I(s1)= \log_2 (1/0.3) = \log_2 3.3 = 1.7369$ bits
  - ➢ s3 → p3 = 0.2 → $I(s1)= \log_2 (1/0.2) = \log_2 5 = 2.3219$ bits
  - ➢ s4 → p4 = 0.1 → $I(s1)= \log_2 (1/0.1) = \log_2 10 = 3.3219$ bits
  - ➢ $H(p) = ½ \log_2 2 + 1/3 \log_2 3.3 + 1/5 \log_2 5 + 1/10 \log_2 10$
    $= 1.8176$ bits ←→ “ mean information in bits “

7.5 Probability -Based Coding(5)

Figure 7.3: The entropy of a binary event (an event with two possible outcomes) as a function of the probability of one of the outcomes.



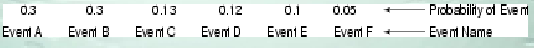
7.5 Probability -Based Coding(6)

- The probability of the next customer being male is still 80%, and the probability of a female customer is still 20%.
  - **Event A, Male-Male:** Because the probability of a male is 0.8, and the probability of independent events is the product of the pair of probabilities, we have that the probability of this event is **0.64**. We will **assign** the very short code of **a single 0 bit** to send the message in this case.
  - **Event B, Male-Female:** By similar reasoning to that in the previous case, the probability of this event is **0.16**, and we will **assign** it the **2-bit code 10**.
  - **Event C, Female-Male:** Again, the probability is **0.16**, and we will **assign** it the **3-bit code 110**. While it doesn't seem fair to make this a 3-bit code in light of the similar previous pair's encoding, we have no choice but to achieve a property known as ***unique decodability***. More will be said on this point below.
  - **Event D: Female-Female:** This event has a probability of **0.04**. This rather infrequent event will have a **3-bit code also, 111**.
- In a complete derivation of the coding method at which we are hinting here, we would calculate entropies for each event and choose codes appropriately. Hence, the resulting coding method is called ***entropy coding***.

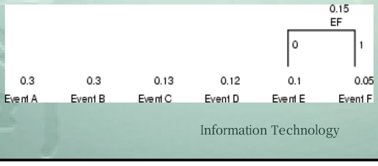
7.5 Probability -Based Coding(7)

- ***Entropy coding*** : The codes were assigned such that **the longest codes** were associated with **the most infrequent events** to the greatest extent possible, while maintaining unique decodability.

1) **Figure 7.4:** Preparing for **Huffman code** construction. List all events in descending order of probability.



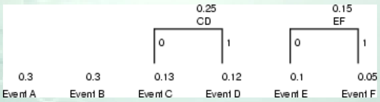
2) **Figure 7.5:** Step one in Huffman code construction: pair the two events with lowest probabilities.



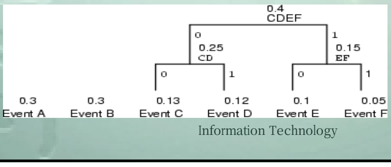
7.5 Probability -Based Coding(8)

- ❑ ***Entropy coding : Huffman coding procedure*** (2)

3) Figure 7.6: Repeat for the pair with the next lowest probabilities.



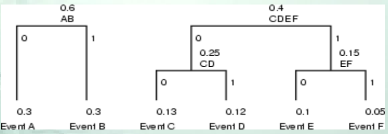
4) Figure 7.7: Repeat again. Note that for this example, previous pairs a repaired.



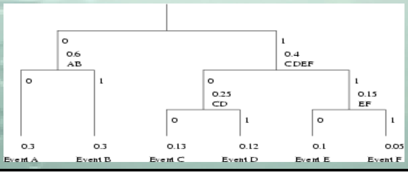
7.5 Probability -Based Coding(9)

- ❑ ***Entropy coding : Huffman coding procedure*** (3)

5) Figure 7.8: Continue pairing the lowest-probability events.

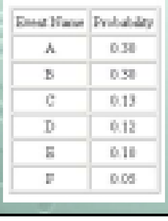


6) Figure 7.9: The complete Huffman code tree.

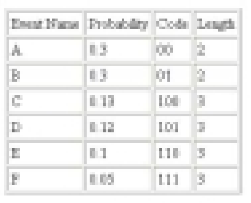


7.5 Probability -Based Coding(10)

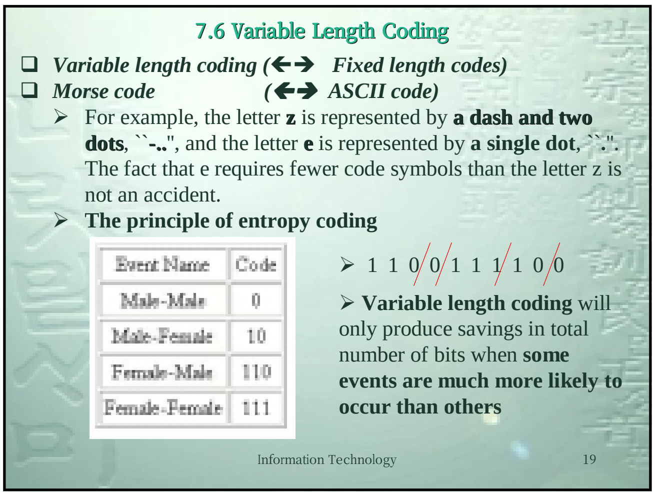
- ❑ ***Entropy coding : Huffman coding procedure*** (4)
- ❑ The Huffman coding method is based on the construction of what is known as a ***binary tree***. The path from the top or ***root*** of this tree to a particular event will determine the code group we associate with that event.
- ❑ If we sum the products of the event probabilities and the code lengths for this case, we find that the average bit rate needed to represent these events is $2(0.3)+2(0.3)+3(0.13)+3(0.12)+3(0.1)+3(0.05)=2.4\,bits/event$



Event Name	Probability
A	0.30
B	0.30
C	0.13
D	0.12
E	0.10
F	0.05



Event Name	Probability	Code	Length
A	0.3	00	2
B	0.3	01	2
C	0.13	100	3
D	0.12	101	3
E	0.1	110	3
F	0.05	111	3



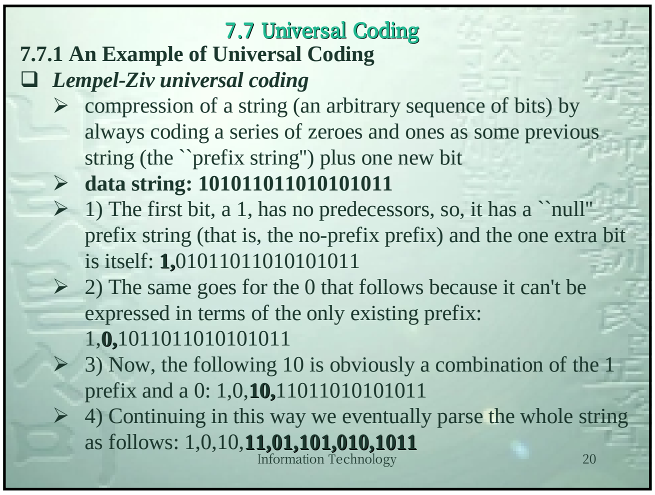
7.6 Variable Length Coding

- ❑ *Variable length coding (🡐🡒 Fixed length codes)*
- ❑ *Morse code (🡐🡒 ASCII code)*
  - ➢ For example, the letter **z** is represented by **a dash and two dots**, ``**-..**'', and the letter **e** is represented by **a single dot**, ``**.**''. The fact that e requires fewer code symbols than the letter z is not an accident.
  - ➢ **The principle of entropy coding**

Event Name	Code
Male-Male	0
Male-Female	10
Female-Male	110
Female-Female	111

➢ 1 1 0/0/1 1 1/1 0/0

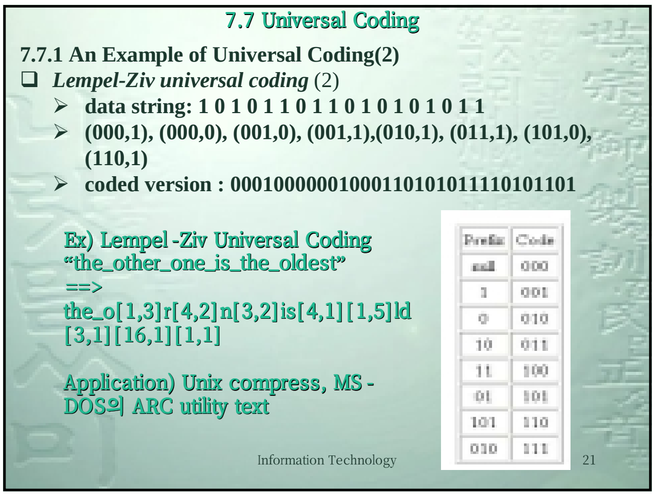
➢ **Variable length coding** will only produce savings in total number of bits when **some events are much more likely to occur than others**



7.7 Universal Coding

7.7.1 An Example of Universal Coding

- ❑ *Lempel-Ziv universal coding*
  - ➢ compression of a string (an arbitrary sequence of bits) by always coding a series of zeroes and ones as some previous string (the ``prefix string'') plus one new bit
  - ➢ **data string: 101011011010101011**
  - ➢ 1) The first bit, a 1, has no predecessors, so, it has a ``null'' prefix string (that is, the no-prefix prefix) and the one extra bit is itself: **1,**01011011010101011
  - ➢ 2) The same goes for the 0 that follows because it can't be expressed in terms of the only existing prefix: 1,**0,**1011011010101011
  - ➢ 3) Now, the following 10 is obviously a combination of the 1 prefix and a 0: 1,0,**10,**11011010101011
  - ➢ 4) Continuing in this way we eventually parse the whole string as follows: 1,0,10,**11,01,101,010,1011**



7.7 Universal Coding

7.7.1 An Example of Universal Coding(2)

- ❑ *Lempel-Ziv universal coding* (2)
  - ➢ **data string: 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 0 1 1**
  - ➢ **(000,1), (000,0), (001,0), (001,1),(010,1), (011,1), (101,0), (110,1)**
  - ➢ **coded version : 0001000000010001101010111110101101**

Ex) Lempel -Ziv Universal Coding
“the_other_one_is_the_oldest”
==>
the_o[1,3]r[4,2]n[3,2]is[4,1][1,5]ld
[3,1][16,1][1,1]

Application) Unix compress, MS - DOS의 ARC utility text

Prefix	Code
null	000
1	001
0	010
10	011
11	100
01	101
101	110
010	111