CT216 Introduction to Communication Systems
Lecture 4: Source Coding

Yash M. Vasavada

Professor, DA-IICT, Gandhinagar

4th March 2024

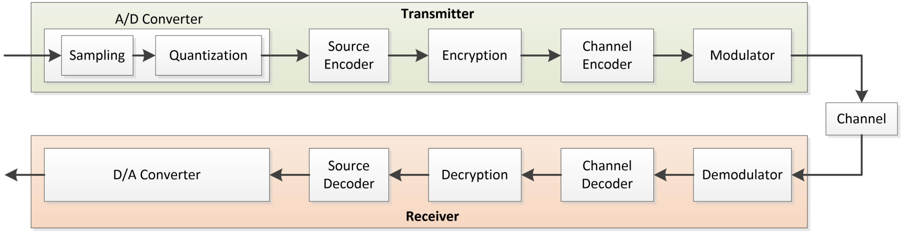
Overview of Today’s Talk

Overview of Today's Talk

Overview of Today's Talk

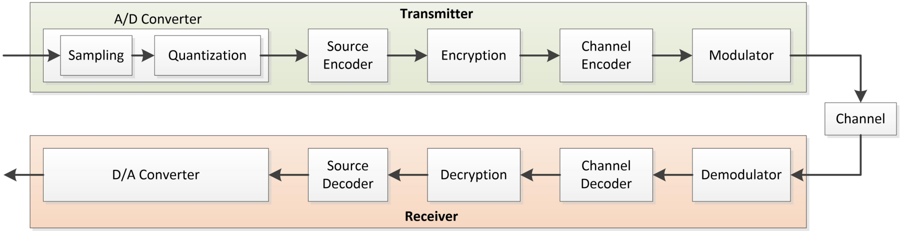
Overview of Today's Talk

The Model of a Communication System



- A preview of next several lectures: study (i) the source coding, (ii) the channel coding and (iii) the modulation blocks

The Model of a Communication System



- A preview of next several lectures: study (i) the source coding, (ii) the channel coding and (iii) the modulation blocks
  - ▷ The basic theoretical framework underlying all these blocks has already been developed!

How can the bits take fractional values?

- We have seen that the concepts of information is obtained in terms of fraction of bits. Both the entropy and the amount of information transfer take the values which are fractional bits. However, how can the bits be in fractional units?

How can the bits take fractional values?

- We have seen that the concepts of information is obtained in terms of fraction of bits. Both the entropy and the amount of information transfer take the values which are fractional bits. However, how can the bits be in fractional units?
- Shannon's answer: one bit at a time cannot be in fraction. However, the best way to think about the information generation and its transfer is to consider a large sequence of bits

Entropy $H(X)$ of the Source and Its Compressibility

Source Coding Problem

- For example, suppose that an informative source X has the entropy $H(X) = 0.5$ bits.
- Thus, only half of the bits generated by source actually contain the information. The remaining half are non-informative (i.e., redundant).
- Therefore, if the above source generates 10 million bits, this sequence can be shortened to 5 million bits without losing any information

Entropy $H(X)$ of the Source and Its Compressibility

Source Coding Problem

- For example, suppose that an informative source X has the entropy $H(X) = 0.5$ bits.
- Thus, only half of the bits generated by source actually contain the information. The remaining half are non-informative (i.e., redundant).
- Therefore, if the above source generates 10 million bits, this sequence can be shortened to 5 million bits without losing any information
    - ▷ This has vast implication in today's world where a lot of data is continuously generated and stored. If the data can be compressed before the storage or transmission over a communication channel, it is clear that that would provide a tremendous benefit

Entropy $H(X)$ of the Source and Its Compressibility

Source Coding Problem

- For example, suppose that an informative source X has the entropy $H(X) = 0.5$ bits.
- Thus, only half of the bits generated by source actually contain the information. The remaining half are non-informative (i.e., redundant).
- Therefore, if the above source generates 10 million bits, this sequence can be shortened to 5 million bits without losing any information
  - ▷ This has vast implication in today's world where a lot of data is continuously generated and stored. If the data can be compressed before the storage or transmission over a communication channel, it is clear that that would provide a tremendous benefit
  - ▷ We will study the problem of data compression, also called the source coding problem

Information Transfer $I(X;Y)$ of a Communication Channel

Channel Coding Problem

- Suppose the communication channel has an information transfer rate $I(X;Y) = 0.33$ bits. In this case, only one-third of the bits transmitted by the sender can be successfully received.

Information Transfer $I(X;Y)$ of a Communication Channel

Channel Coding Problem

- Suppose the communication channel has an information transfer rate $I(X;Y) = 0.33$ bits. In this case, only one-third of the bits transmitted by the sender can be successfully received.
- Thus, if the source has 10 million bits to be transferred over a communication channel, it has to actually transmit 30 million bits, where the extra 20 million bits ensure that the communication channel does not corrupt the main informative content of 10 million bits

Information Transfer $I(X;Y)$ of a Communication Channel

Channel Coding Problem

- Suppose the communication channel has an information transfer rate $I(X;Y) = 0.33$ bits. In this case, only one-third of the bits transmitted by the sender can be successfully received.
- Thus, if the source has 10 million bits to be transferred over a communication channel, it has to actually transmit 30 million bits, where the extra 20 million bits ensure that the communication channel does not corrupt the main informative content of 10 million bits
  - ▷ This is called the channel coding problem. We have already studied a basic scheme. . .

Information Transfer $I(X;Y)$ of a Communication Channel

Channel Coding Problem

- Suppose the communication channel has an information transfer rate $I(X;Y) = 0.33$ bits. In this case, only one-third of the bits transmitted by the sender can be successfully received.
- Thus, if the source has 10 million bits to be transferred over a communication channel, it has to actually transmit 30 million bits, where the extra 20 million bits ensure that the communication channel does not corrupt the main informative content of 10 million bits
  - ▷ This is called the channel coding problem. We have already studied a basic scheme. . .
  - ▷ . . ., i.e., the repetition coding. Would "Repeat-by-Three" scheme solve this problem?

Information Transfer $I(X;Y)$ of a Communication Channel

Channel Coding Problem

- Suppose the communication channel has an information transfer rate $I(X;Y) = 0.33$ bits. In this case, only one-third of the bits transmitted by the sender can be successfully received.
- Thus, if the source has 10 million bits to be transferred over a communication channel, it has to actually transmit 30 million bits, where the extra 20 million bits ensure that the communication channel does not corrupt the main informative content of 10 million bits
  - ▷ This is called the channel coding problem. We have already studied a basic scheme. . .
  - ▷ . . ., i.e., the repetition coding. Would "Repeat-by-Three" scheme solve this problem?
  - ▷ Shannon showed that the repetition coding is too simple and does not work well for the real-world channels. Schemes that are smarter and more complex are needed to solve the channel coding problem

Need of Modulation for Transmission over a Communication Channel

Mod/Demod Problem

- The bits which convey the information are not suitable for transmission over long distances — electromagnetic waves are needed for this purpose

Need of Modulation for Transmission over a Communication Channel

Mod/Demod Problem

- The bits which convey the information are not suitable for transmission over long distances — electromagnetic waves are needed for this purpose
- The communication transmitter converts the informative bits to the electromagnetic waves which are emitted by an antenna.

Need of Modulation for Transmission over a Communication Channel

Mod/Demod Problem

- The bits which convey the information are not suitable for transmission over long distances — electromagnetic waves are needed for this purpose
- The communication transmitter converts the informative bits to the electromagnetic waves which are emitted by an antenna.
  - ▷ This is called the modulation, since the bits are used to “modulate” one or multiple characteristics of the EM wave — e.g., the amplitude, the phase, or the frequency.

Need of Modulation for Transmission over a Communication Channel

Mod/Demod Problem

- The bits which convey the information are not suitable for transmission over long distances — electromagnetic waves are needed for this purpose
- The communication transmitter converts the informative bits to the electromagnetic waves which are emitted by an antenna.
  - This is called the modulation, since the bits are used to "modulate" one or multiple characteristics of the EM wave — e.g., the amplitude, the phase, or the frequency.
  - The received modulated wave has to be converted back to the bits. This is called the demodulation.

Need of Modulation for Transmission over a Communication Channel

Mod/Demod Problem

- The bits which convey the information are not suitable for transmission over long distances — electromagnetic waves are needed for this purpose
- The communication transmitter converts the informative bits to the electromagnetic waves which are emitted by an antenna.
  - ▷ This is called the modulation, since the bits are used to "modulate" one or multiple characteristics of the EM wave — e.g., the amplitude, the phase, or the frequency.
  - ▷ The received modulated wave has to be converted back to the bits. This is called the demodulation.
- We have studied a basic modulation and demodulation scheme called the BPSK (Binary Phase Shift Keying). We will expand upon this further

Source Coding

Also known as Data Compression

- Many digital data streams contain a lot of redundant information. For example, a digital image file may contain more zeros than ones: $0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1$
- We wish to squeeze out the redundant information to minimize the amount of data needed to be stored or transmitted
- Definition of Data Compression Problem:
  1. What are the good algorithms that achieve the maximal data compression?
  2. What is the maximum data compression that can be achieved if we want to recover the exact bit sequence after decompression?
- Importance of Data Compression Problem:
  - ▷ Cannot be overstated given so much data is getting uploaded/downloaded and stored in today's world of YouTube, Facebook, etc.

Source Coding

Discrete Memoryless Source (DMS)

- A Discrete Memoryless Source (DMS) generates discrete symbols (instead of continuous-valued output) which are independent of each other, i.e., the source is discrete and it does not have memory

Source Coding

Discrete Memoryless Source (DMS)

- A Discrete Memoryless Source (DMS) generates discrete symbols (instead of continuous-valued output) which are independent of each other, i.e., the source is discrete and it does not have memory
  - $\rightarrow$ Consider two discrete informative sources: (i) a playwriter composing a drama using her laptop keyboard, (ii) a monkey punching the keys of a typewriter placed in front of it. Which of the two sources is memoryless?

Source Coding

Discrete Memoryless Source (DMS)

- A Discrete Memoryless Source (DMS) generates discrete symbols (instead of continuous-valued output) which are independent of each other, i.e., the source is discrete and it does not have memory
  - → Consider two discrete informative sources: (i) a playwriter composing a drama using her laptop keyboard, (ii) a monkey punching the keys of a typewriter placed in front of it. Which of the two sources is memoryless?
  - → Source (ii)
- We will mostly assume that the source is DMS

Source Coding

Discrete Memoryless Source (DMS)

- A Discrete Memoryless Source (DMS) generates discrete symbols (instead of continuous-valued output) which are independent of each other, i.e., the source is discrete and it does not have memory
  - → Consider two discrete informative sources: (i) a playwriter composing a drama using her laptop keyboard, (ii) a monkey punching the keys of a typewriter placed in front of it. Which of the two sources is memoryless?
  - → Source (ii)
- We will mostly assume that the source is DMS
  - → With more complicated notations, the theory for DMS can be extended to the more realistic informative sources which exhibit a memory

Source Coding

for DMS

- A source code is a rule which maps symbols from the source alphabet to sequence of bits
- Fixed length source code maps source symbols into codewords of the same length:
  - → Example: $x_1 \Rightarrow 00, x_2 \Rightarrow 01, x_2 \Rightarrow 10, x_4 \Rightarrow 11$
- Variable length source code maps source symbols into codewords of different lengths:
  - → Example: $x_1 \Rightarrow 1, x_2 \Rightarrow 01, x_2 \Rightarrow 101, x_4 \Rightarrow 11$
- Variable length source code is uniquely decodeable if no codeword is a prefix of another codeword:
  - → Uniquely decodeable: $x_1 \Rightarrow 000, x_2 \Rightarrow 0010, x_3 \Rightarrow 01, x_4 \Rightarrow 10, x_5 \Rightarrow 1100, x_6 \Rightarrow 1101, x_7 \Rightarrow 111$
  - → Counter-example: $x_1 \Rightarrow 1, x_2 \Rightarrow 00, x_2 \Rightarrow 01, x_4 \Rightarrow 10$. If we get 001001, should we decode it as x_2, x_1, $x_2\,x_1$ or as x_2, x_4, x_3?

We will work only with the prefix-free codes.

Source Coding

for DMS: Variable Length Codes

- Why to consider variable-length codes, when the fixed length codes don't have the ambiguity problem outlined earlier?

Source Coding

for DMS: Variable Length Codes

- Why to consider variable-length codes, when the fixed length codes don't have the ambiguity problem outlined earlier?
  - $\rightarrow$ When different symbols have vastly varying probabilities (e.g., the letter "e" versus the letter "q" in English), using equal-length codewords is inefficient. Why?

Source Coding

for DMS: Variable Length Codes

- Why to consider variable-length codes, when the fixed length codes don't have the ambiguity problem outlined earlier?
  - $\rightarrow$ When different symbols have vastly varying probabilities (e.g., the letter "e" versus the letter "q" in English), using equal-length codewords is inefficient. Why?
  - $\rightarrow$ We have seen that the information generated in number of bits upon the occurrence of the k^{th} symbol, i.e., $I_k = -\log_2(p_k)$, is much smaller for "e" versus "q".

Source Coding

for DMS: Variable Length Codes

- Why to consider variable-length codes, when the fixed length codes don't have the ambiguity problem outlined earlier?
  - $\rightarrow$ When different symbols have vastly varying probabilities (e.g., the letter "e" versus the letter "q" in English), using equal-length codewords is inefficient. Why?
  - $\rightarrow$ We have seen that the information generated in number of bits upon the occurrence of the k^{th} symbol, i.e., $I_k = -\log_2(p_k)$, is much smaller for "e" versus "q".
  - $\rightarrow$ We will see that the highest amount of data compression can be achieved if the codeword length equals the information I_k in bits
- How to represent the variable/fixed length codewords?

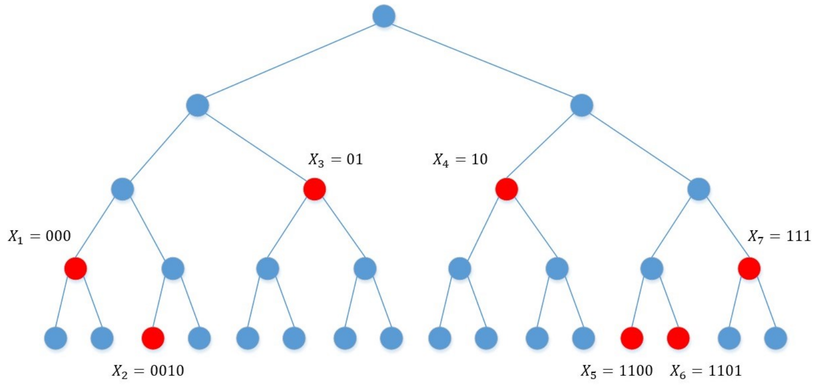
Source Coding

for DMS: Variable Length Codes

- Why to consider variable-length codes, when the fixed length codes don't have the ambiguity problem outlined earlier?
  - $\rightarrow$ When different symbols have vastly varying probabilities (e.g., the letter "e" versus the letter "q" in English), using equal-length codewords is inefficient. Why?
  - $\rightarrow$ We have seen that the information generated in number of bits upon the occurrence of the k^{th} symbol, i.e., $I_k = -\log_2(p_k)$, is much smaller for "e" versus "q".
  - $\rightarrow$ We will see that the highest amount of data compression can be achieved if the codeword length equals the information I_k in bits
- How to represent the variable/fixed length codewords?
  - $\rightarrow$ For binary codewords, a binary tree is a useful representation

Source Coding

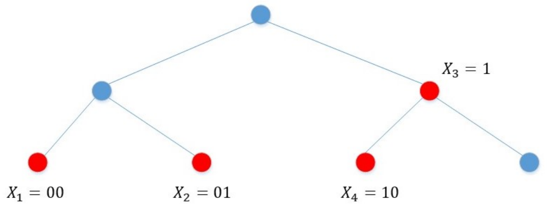
for DMS: An Example Prefix-Free Code on a Binary Tree



- An assigned codeword is never part of a subtree of another assigned codeword

Source Coding

for DMS: An Example of Non-Prefix-Free Code on a Binary Tree



- Codeword X_4 belongs to the subtree of X_3, hence violates the prefix-free requirement

Source Coding

Observations from the Binary Tree

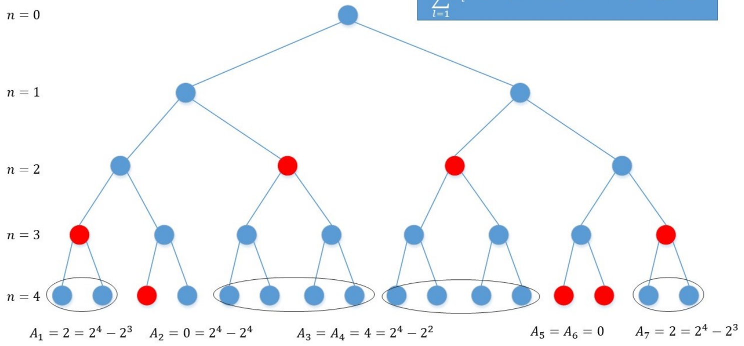
- Let n_ℓ be the number of bits in the codeword which represents a source symbol x_ℓ and let $n_{\max}$ be the length of the longest codeword.
- The subtree under x_ℓ bits has $A_\ell = 2^{n_{\max} - n_\ell}$ leaf nodes.

Source Coding

Observations from the Binary Tree

- Let n_ℓ be the number of bits in the codeword which represents a source symbol x_ℓ and let $n_{\max}$ be the length of the longest codeword.
- The subtree under x_ℓ bits has $A_\ell = 2^{n_{\max}-n_\ell}$ leaf nodes.

$$\sum_{l=1}^{7} A_l = 2+0+4+4+0+0+2 = 12 < 16$$



Kraft's Inequality

- For prefix free code, where individual codewords have lengths $1 \leq n_\ell \leq n_{\max}$,

$$\sum_{\ell=1}^{L} 2^{-n_\ell} \leq 1$$

Kraft's Inequality

- For prefix free code, where individual codewords have lengths $1 \leq n_\ell \leq n_{\max}$,

$$\sum_{\ell=1}^{L} 2^{-n_\ell} \leq 1$$

- Proof:

Kraft's Inequality

- For prefix free code, where individual codewords have lengths $1 \leq n_\ell \leq n_{\max}$,

$$\sum_{\ell=1}^{L} 2^{-n_\ell} \leq 1$$

- Proof:
  - The subtree under an assigned codeword of length n_ℓ bits has $A_\ell = 2^{n_{\max} - n_\ell}$ leaf nodes.
  - Since the code is prefix-free, the subtrees for different codewords are disjoint.
  - The total number of leaf nodes equals $2^{n_{\max}}$. Therefore,

Kraft's Inequality

- For prefix free code, where individual codewords have lengths $1 \leq n_\ell \leq n_{\max}$,

$$\sum_{\ell=1}^{L} 2^{-n_\ell} \leq 1$$

- Proof:
  - The subtree under an assigned codeword of length n_ℓ bits has $A_\ell = 2^{n_{\max} - n_\ell}$ leaf nodes.
  - Since the code is prefix-free, the subtrees for different codewords are disjoint.
  - The total number of leaf nodes equals $2^{n_{\max}}$. Therefore,

$$\sum_{\ell=1}^{L} A_\ell \leq 2^{n_{\max}} \Rightarrow \sum_{\ell=1}^{L} 2^{n_{\max} - n_\ell} \leq 2^{n_{\max}} \Rightarrow \sum_{\ell=1}^{L} 2^{-n_\ell} \leq 1$$

Kraft's Inequality

An Interesting Observation

- Suppose for some code, the Kraft's Inequality turns into an equality, i.e., $\sum_{\ell=1}^{L} 2^{-n_\ell} = 1$.

Kraft's Inequality

An Interesting Observation

- Suppose for some code, the Kraft's Inequality turns into an equality, i.e., $\sum_{\ell=1}^{L} 2^{-n_\ell} = 1$.
- In this case, we can possibly consider the left side of the equality as forming. . .

Kraft's Inequality

An Interesting Observation

- Suppose for some code, the Kraft's Inequality turns into an equality, i.e., $\sum_{\ell=1}^{L} 2^{-n_\ell} = 1$.
- In this case, we can possibly consider the left side of the equality as forming. . .
- . . . a probability mass function (note that it is for the probability mass function that this equality is satisfied)

Kraft's Inequality

An Interesting Observation

- Suppose for some code, the Kraft's Inequality turns into an equality, i.e., $\sum_{\ell=1}^{L} 2^{-n_\ell} = 1$.
- In this case, we can possibly consider the left side of the equality as forming. . .
- . . . a probability mass function (note that it is for the probability mass function that this equality is satisfied)
- Thus, if 2^{-n_ℓ} is equated to (pseudo-) probability $\hat{p}_\ell$ of ℓ^{th} symbol, we obtain

$$\hat{p}_\ell = 2^{-n_\ell} \Rightarrow n_\ell = -\log_2(\hat{p}_\ell)$$

However, the right side is the information content of a symbol with the probability $\hat{p}_\ell$!

Kraft's Inequality

An Interesting Observation

- Suppose for some code, the Kraft's Inequality turns into an equality, i.e., $\sum_{\ell=1}^{L} 2^{-n_\ell} = 1$.
- In this case, we can possibly consider the left side of the equality as forming...
- ... a probability mass function (note that it is for the probability mass function that this equality is satisfied)
- Thus, if 2^{-n_ℓ} is equated to (pseudo-) probability $\hat{p}_\ell$ of ℓ^{th} symbol, we obtain

$$\hat{p}_\ell = 2^{-n_\ell} \Rightarrow n_\ell = -\log_2(\hat{p}_\ell)$$

  However, the right side is the information content of a symbol with the probability $\hat{p}_\ell$!
- Thus, when the Kraft's Inequality turns into the equality, the codeword lengths $\{n_\ell\}$ equal the Shannon's definition of the information generated by the symbols having the pseudo-probabilities $\{\hat{p}_\ell\}$

Source Coding
for DMS

- Rate R of a source code is the expected number of bits (mean value) required to represent a source symbol

Source Coding
for DMS

- Rate R of a source code is the expected number of bits (mean value) required to represent a source symbol
  - → Therefore, $R = \sum_{k=1}^{L} p_k n_k$ bits

Source Coding
for DMS

- Rate R of a source code is the expected number of bits (mean value) required to represent a source symbol
  - $\rightarrow$ Therefore, $R = \sum_{k=1}^{L} p_k n_k$ bits
  - $\rightarrow$ For fixed length codes, $R \geq \lceil \log_2 L \rceil$

Source Coding
for DMS

- Rate R of a source code is the expected number of bits (mean value) required to represent a source symbol
  - $\rightarrow$ Therefore, $R = \sum_{k=1}^{L} p_k n_k$ bits
  - $\rightarrow$ For fixed length codes, $R \geq \lceil \log_2 L \rceil$
- **Goal of Source Encoding:** find a *uniquely decodeable* code that minimizes R

Source Coding
for DMS

- Rate R of a source code is the expected number of bits (mean value) required to represent a source symbol
  - $\rightarrow$ Therefore, $R = \sum_{k=1}^{L} p_k n_k$ bits
  - $\rightarrow$ For fixed length codes, $R \geq \lceil \log_2 L \rceil$
- **Goal of Source Encoding:** find a *uniquely decodeable* code that minimizes R
- **Strategy:** assign short codewords (small n_k) to the most likely symbols (large p_k)

Source Coding Theorem

- *A fundamental theorem of the Information Theory [Shannon 1948 paper]:*
  - → For a uniquely decodeable source code that provides lossless compression (i.e., perfect reconstruction of the source symbols at the source decoder), $R \geq H(X)$.
    - ▷ Rate can be made arbitrarily close to, but not less than, $H(X)$

Source Coding Theorem

- *A fundamental theorem of the Information Theory [Shannon 1948 paper]:*
  - $\rightarrow$ For a uniquely decodeable source code that provides lossless compression (i.e., perfect reconstruction of the source symbols at the source decoder), $R \geq H(X)$.
    - ▷ Rate can be made arbitrarily close to, but not less than, $H(X)$
  - $\rightarrow$ This is the Source Coding (or Data Compression) Theorem

Source Coding Theorem

- *A fundamental theorem of the Information Theory [Shannon 1948 paper]:*
  - → For a uniquely decodeable source code that provides lossless compression (i.e., perfect reconstruction of the source symbols at the source decoder), $R \geq H(X)$.
    - ▷ Rate can be made arbitrarily close to, but not less than, $H(X)$
  - → This is the Source Coding (or Data Compression) Theorem
- How to *prove* this theorem?

Practical Source Coding Algorithms

- There are two widely used source coding methods which make R approach the lower bound of $H(X)$ promised by data compression theorem:
  1. Huffman Coding: requires the knowledge of probabilities of DMS
  2. Lempel-Ziv Coding: operates in a *blind* manner (i.e., without requiring the knowledge of the DMS' PMF)

Practical Source Coding Algorithms

- There are two widely used source coding methods which make R approach the lower bound of $H(X)$ promised by data compression theorem:
  1. Huffman Coding: requires the knowledge of probabilities of DMS
  2. Lempel-Ziv Coding: operates in a *blind* manner (i.e., without requiring the knowledge of the DMS' PMF)
- Several other algorithms: Shannon-Fano coding, Shannon-Fano-Elias coding, Arithmetic coding, etc. — we will not study these schemes

Source Coding Theorem

Huffman Coding Algorithm

Huffman coding is based upon the idea of allocating short codewords to highly probable symbols. The algorithm for Huffman coding belongs to a class of algorithms based upon the Greedy Approach.

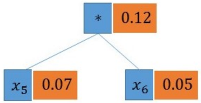
1. List all symbols in descending order of probability
2. Group the two lowest probability symbols into a new combined symbol
3. Repeat steps 1 and 2 until only a single combined symbol remains
4. Assign a 1 to each upper branch and 0 to each lower branch of the resulting binary tree
5. Codeword for each symbol is the sequence of bits from the root of the tree to that symbol

Source Coding Theorem

Huffman Coding Algorithm: An Example

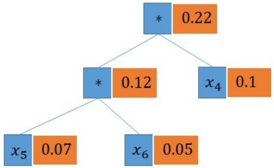
1

Symbol	Probability
x_1	0.45
x_2	0.2
x_3	0.13
x_4	0.1
x_5	0.07
x_6	0.05



2

Symbol	Probability
x_1	0.45
x_2	0.2
x_3	0.13
$*$	0.12
x_4	0.1

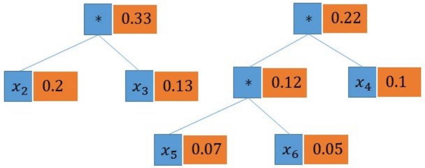


Source Coding Theorem

Huffman Coding Algorithm: An Example

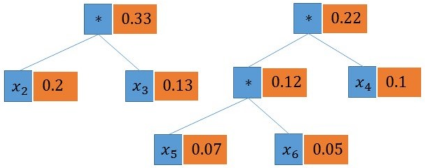
3

Symbol	Probability
x_1	0.45
$*$	0.22
x_2	0.2
x_3	0.13



4

Symbol	Probability
x_1	0.45
$*$	0.33
$*$	0.22

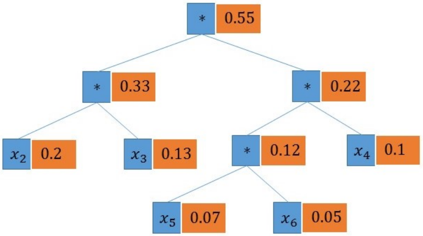


Source Coding Theorem

Huffman Coding Algorithm: An Example

5

Symbol	Probability
$*$	0.55
x_1	0.45

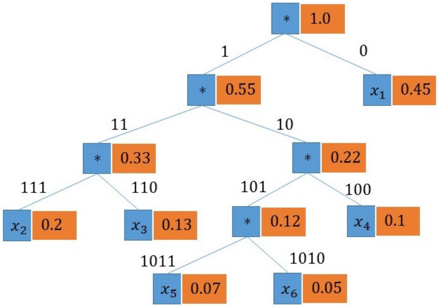


Source Coding Theorem

Huffman Coding Algorithm: An Example

6

Symbol	Probability	Codeword
x_1	0.45	0
x_2	0.2	111
x_3	0.13	110
x_4	0.1	100
x_5	0.07	1011
x_6	0.05	1010



Source Coding Theorem

Huffman Coding Algorithm: Numerical Analysis

Symbol	Probability p_k	Codeword	Length L_k	$p_k L_k$	$-log_2 p_k$	$-p_k log_2 p_k$
x_1	0.45	0	1	0.45	1.15	0.518
x_2	0.2	111	3	0.6	2.32	0.464
x_3	0.13	110	3	0.39	2.94	0.382
x_4	0.1	100	3	0.3	3.32	0.332
x_5	0.07	1011	4	0.28	3.83	0.268
x_6	0.05	1010	4	0.2	4.32	0.216
				2.2		2.18

Source Coding Theorem

Huffman Coding Algorithm: Summary

- How do we get even closer to Entropy?
  - $\rightarrow$ Use Huffman Algorithm on pairs of symbols, or
  - $\rightarrow$ Even better, on triplets of symbols, or quadruplets or,
  - $\rightarrow$ In general, *m*-tuple of symbols.
- A limitation of Huffman Algorithm is that it requires the source symbol probabilities to be known. These are not known exactly but they can be estimated from a sufficiently large sample of source output. Mismatch between the estimated and exact symbol probabilities contributes to a degraded performance of Huffman's Algorithm.
- Another limitation of Huffman Code is that the symbol probabilities are assumed to be fixed. If these probabilities change with time, Huffman Code may provide suboptimal data compression.

Lempel-Ziv Coding

Encoding Algorithm

- In Lempel-Ziv coding, the source sequence is sequentially parsed into strings that have not appeared so far.
  - → Continually parse the symbol sequence and insert a comma on detecting the shortest sequence that has not been seen earlier
- Example sequence: 1011010100010. . .
- Sequential parsing method in Lempel-Ziv encoding:
  - → 1,011010100010. . .
  - → 1,0,11010100010. . .
  - → 1,0,11,010100010. . .
  - → 1,0,11,01,0100010. . .
  - → 1,0,11,01,010,0010. . .
  - → 1,0,11,01,010,00,10. . .
  - → 1,0,11,01,010,00,10,. . .

Lempel-Ziv Coding
Encoding Algorithm

- Prefix of each "phrase" (phrase is the symbol string between two commas), i.e., the string consisting of all but the last symbol of each phrase, must have occurred earlier.
- Encode this phrase by giving the location of the prefix and the value of the last bit.
- Let $C(n)$ be the number of phrases in the parsing of the input sequence with n symbols.
- Thus, $\log C(n)$ bits are needed to describe the location of the prefix to the phrase and one bit to describe the last bit.
- In the above example, the code sequence is: $(000, 1), (000, 0), (001, 1), (010, 1), (100, 0), (010, 0), (001, 0)$
  - $\rightarrow$ First number of each pair gives the index of the prefix and the second number gives the last bit of the phrase.

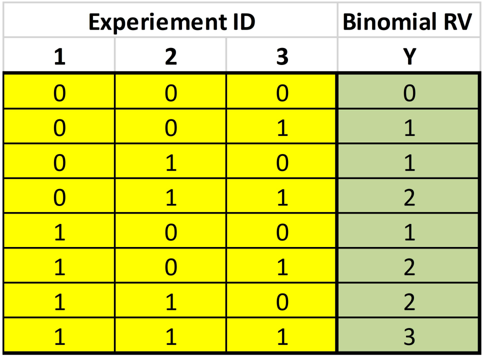
Understanding Binomial Distribution

- Suppose I perform an experiment N times, where each outcome is a Binary RV (Bernouilli RV), with probability of 1 equal to p and 0 equal to $1 - p$
- Questions:
  - What are all the possible outcomes?
  - What is the *sum* of N binary outcomes?
- Let us look at the cases $N = 2, 3$ and 4.

Understanding Binomial Distribution

Experiement ID		Binomial RV
1	2	Y
0	0	0
0	1	1
1	0	1
1	1	2

Understanding Binomial Distribution



Experiment ID			Binomial RV
1	2	3	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	2
1	0	0	1
1	0	1	2
1	1	0	2
1	1	1	3

Understanding Binomial Distribution

Experiement ID				Binomial RV
1	2	3	4	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	2
0	1	0	0	1
0	1	0	1	2
0	1	1	0	2
0	1	1	1	3
1	0	0	0	1
1	0	0	1	2
1	0	1	0	2
1	0	1	1	3
1	1	0	0	2
1	1	0	1	3
1	1	1	0	3
1	1	1	1	4

Understanding Binomial Distribution

1. Outcomes of N experiments form a binary "string" of length N bits
2. When these N bits are *added up*, we get a variable Y, which is the Binomial RV
3. Value of Y can be any integer between 0 (all-zeros binary string) to N (all-ones string)
4. There is a total of 2^N possible strings (these form a set of all possible outcomes)
5. A total of $\binom{N}{y}$ strings have y ones and $N - y$ zeros. When any of these strings occur, the variable Y will take a value y.
6. Due to independence between N experiments, probability of occurrence of any one of $\binom{N}{y}$ string is the same, and it is given as $p^y(1-p)^{N-y}$.
7. Therefore, the total probability that $Y = y$ is given as $\binom{N}{y}p^y(1-p)^{N-y}$.

Understanding Binomial Distribution

Another Problem

- *You are told:*

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?
- Answer:
  - ▷ $N = 10$, $p = 0.1$, $y = 3$.

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?
- Answer:
  - ▷ $N = 10$, $p = 0.1$, $y = 3$.
  - ▷ $\binom{N=10}{y=3} = 120$

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?
- Answer:
  - ▷ $N = 10$, $p = 0.1$, $y = 3$.
  - ▷ $\binom{N=10}{y=3} = 120$
  - ▷ $p^y(1-p)^{N-y} = 0.3^3 0.7^7 = 0.0022235661$

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?
- Answer:
  - ▷ $N = 10,\ p = 0.1,\ y = 3.$
  - ▷ $\binom{N=10}{y=3} = 120$
  - ▷ $p^y(1-p)^{N-y} = 0.3^3 0.7^7 = 0.0022235661$
  - ▷ $\binom{N=10}{y=3} p^y(1-p)^{N-y} = 120 \times 0.0022235661 = 0.2668$

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?
- Answer:
  - ▷ $N = 10$, $p = 0.1$, $y = 3$.
  - ▷ $\binom{N=10}{y=3} = 120$
  - ▷ $p^y(1-p)^{N-y} = 0.3^3 0.7^7 = 0.0022235661$
  - ▷ $\binom{N=10}{y=3} p^y(1-p)^{N-y} = 120 \times 0.0022235661 = 0.2668$

  Therefore, the chances that 30% of $N = 10$ cycles you're inspecting will fail are $\approx 27\%$.

Understanding Binomial Distribution

Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?
- Answer:
  - ▷ $N = 10$, $p = 0.1$, $y = 3$.
  - ▷ $\binom{N=10}{y=3} = 120$
  - ▷ $p^y(1-p)^{N-y} = 0.3^3 0.7^7 = 0.0022235661$
  - ▷ $\binom{N=10}{y=3} p^y(1-p)^{N-y} = 120 \times 0.0022235661 = 0.2668$

  Therefore, the chances that 30% of $N = 10$ cycles you're inspecting will fail are $\approx 27\%$.
- What are the chances that 2 cycles will fail, or 5 will fail, or all 10 will fail?

Understanding Binomial Distribution

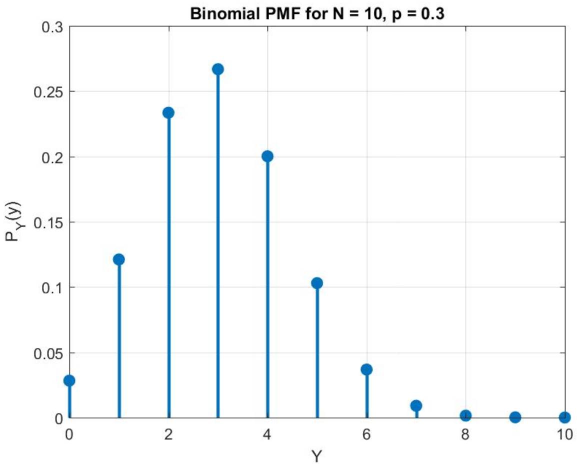
Another Problem

- *You are told:*
  - ▷ At a bicycle company, 70% of the newly made cycles pass the quality inspection and 30% fail.
- *Question:*
  - ▷ What are the chances that 3 of 10 cycles that you're inspecting will fail?
- Answer:
  - ▷ $N = 10$, $p = 0.1$, $y = 3$.
  - ▷ $\binom{N=10}{y=3} = 120$
  - ▷ $p^y(1-p)^{N-y} = 0.3^3 0.7^7 = 0.0022235661$
  - ▷ $\binom{N=10}{y=3} p^y(1-p)^{N-y} = 120 \times 0.0022235661 = 0.2668$

  Therefore, the chances that 30% of $N = 10$ cycles you're inspecting will fail are $\approx 27\%$.
- What are the chances that 2 cycles will fail, or 5 will fail, or all 10 will fail?
  - ▷ Binomial PMF gives the answers to all such questions.

Understanding Binomial Distribution

Another Problem



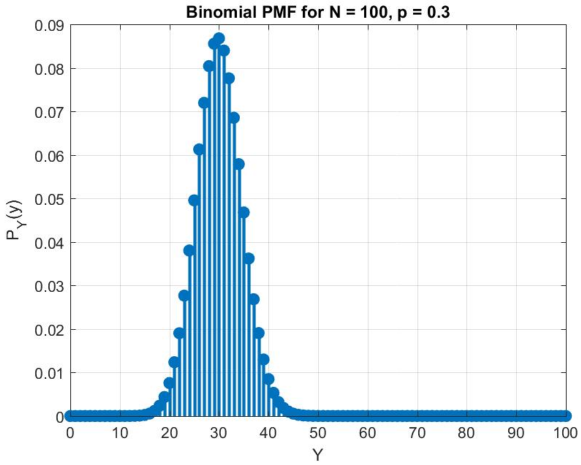
Understanding Binomial Distribution

Another Problem

- Although the failure rate is 30%, there is a good amount of chance that only one out of 10 cycles fails (10% failure rate) or even that none fail (0% failure rate)
- This happens because $N = 10$ is not large enough as the number of independent experiments
  - ▷ Let us see what happens as we increase N

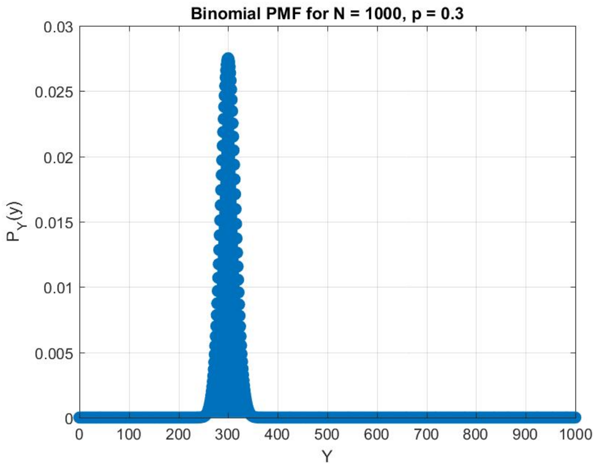
Understanding Binomial Distribution

Another Problem



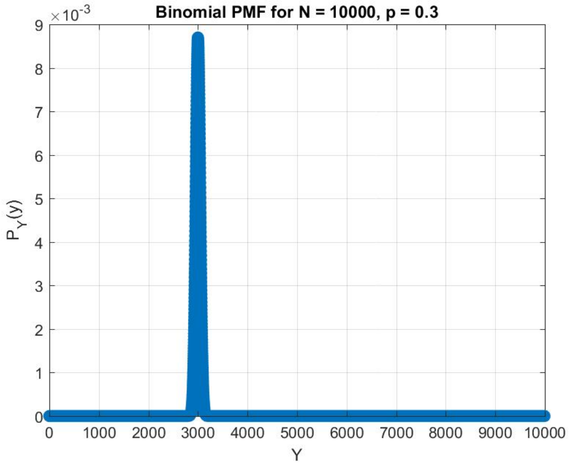
Understanding Binomial Distribution

Another Problem



Understanding Binomial Distribution

Another Problem



Understanding Binomial Distribution

An Observation

- As $N \to \infty$, we will indeed (as expected) see with a very high probability that $0.3 \times N$ cycles failing
- Alternatively, the chances that the number of cycles failing is different from $0.3 \times N$ will diminish and become near zero

Understanding Binomial Distribution

An Observation

- As $N \to \infty$, we will indeed (as expected) see with a very high probability that $0.3 \times N$ cycles failing
- Alternatively, the chances that the number of cycles failing is different from $0.3 \times N$ will diminish and become near zero
  - ▷ In general, when the individual Bernouilli variable has a probability of one of q, there will be $N\,q$ failed cycles (ones) and $N\,(1-q)$ good cycles (zeros) as N becomes sufficiently large

Understanding Binomial Distribution

An Observation

- As $N \to \infty$, we will indeed (as expected) see with a very high probability that $0.3 \times N$ cycles failing
- Alternatively, the chances that the number of cycles failing is different from $0.3 \times N$ will diminish and become near zero
  - ▷ In general, when the individual Bernouilli variable has a probability of one of q, there will be $N\,q$ failed cycles (ones) and $N\,(1-q)$ good cycles (zeros) as N becomes sufficiently large
  - ▷ Another justification: the relative-frequency definition of the probability $q = \lim_{N\to\infty} N_1/N$. However, this implies that the number of ones $N_1 \to q\,N$ as $N \to \infty$. This is observed in the asymptotic form of the Binomial PDF in the prior slide

When is the Fixed-Length Coding Optimal?

- If the source DMS has 4 symbols, each with the probability $1/4$, a fixed-length coding which assigns two bits to each source symbol is optimal
  - ▷ $X_1 = 00, X_2 = 01, X_3 = 10, X_4 = 11$
- If the source DMS has 8 symbols, each occurring with the probability $1/8$, a fixed-length coding which assigns three bits to each source symbol is optimal
  - ▷ $X_1 = 000, X_2 = 001, X_3 = 010, X_4 = 011, X_5 = 100, X_6 = 101, X_7 = 110, X_8 = 111$

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- A Bernouilli(q) source has two symbols, 1 and 0, that occur with unequal probabilities, q and $1 - q$, respectively
- However, if we generate a sequence of N symbols out of this source, what can we say about its composition as $N \to \infty$?

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- A Bernouilli(q) source has two symbols, 1 and 0, that occur with unequal probabilities, q and $1 - q$, respectively
- However, if we generate a sequence of N symbols out of this source, what can we say about its composition as $N \to \infty$?
  - ▷ Any sequence with sufficiently large N will have $N\,q$ ones and $N\,(1 - q)$ zeros

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- A Bernouilli(q) source has two symbols, 1 and 0, that occur with unequal probabilities, q and $1 - q$, respectively
- However, if we generate a sequence of N symbols out of this source, what can we say about its composition as $N \to \infty$?
  - ▷ Any sequence with sufficiently large N will have $N\,q$ ones and $N\,(1 - q)$ zeros
  - ▷ This is what we have concluded earlier based on the asymptotic form of the Binomial PDF.

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- Since each sequence i with sufficiently long length N has the same number of ones and zeros, i.e., $N\,q$ and $N\,(1-q)$, they all have the same probability given as

Fixed-Length Coding for Long Symbol Sequences
generated by Bernouilli(q) DMS

- Since each sequence i with sufficiently long length N has the same number of ones and zeros, i.e., $N\,q$ and $N\,(1-q)$, they all have the same probability given as

$$p_i = q^{N\,q}\,(1-q)^{N\,(1-q)}$$

Fixed-Length Coding for Long Symbol Sequences
generated by Bernouilli(q) DMS

- Since each sequence i with sufficiently long length N has the same number of ones and zeros, i.e., $N\,q$ and $N\,(1-q)$, they all have the same probability given as

$$p_i = q^{N\,q}\,(1-q)^{N\,(1-q)}$$

- A mathematical trick:

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- Since each sequence i with sufficiently long length N has the same number of ones and zeros, i.e., $N\,q$ and $N\,(1-q)$, they all have the same probability given as

$$p_i = q^{N\,q}\,(1-q)^{N\,(1-q)}$$

- A mathematical trick:

$$\begin{aligned} p_i &= q^{N\,q}\,(1-q)^{N\,(1-q)} \\ &= 2^{log_2(q^{N\,q}\,(1-q)^{N\,(1-q)})} \\ &= 2^{N(q\log_2 q + (1-q)\log_2(1-q))} \\ &= 2^{-NH_b(q)}, \end{aligned}$$

where $H_b(q) = -\,(q\log_2 q + (1-q)\log_2(1-q))$ is the Binary Entropy function of variable q.

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- With large N, the Bernouilli(q) DMS becomes a source which generates all the sequences with the same probability p_i
- The total number of such sequences K has to be such that $Kp_i = 1$. Why?

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- With large N, the Bernouilli(q) DMS becomes a source which generates all the sequences with the same probability p_i
- The total number of such sequences K has to be such that $Kp_i = 1$. Why?
  - ▷ Generation of these binary sequences are mutually exclusive (atomic) events

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- With large N, the Bernouilli(q) DMS becomes a source which generates all the sequences with the same probability p_i
- The total number of such sequences K has to be such that $Kp_i = 1$. Why?
  - ▷ Generation of these binary sequences are mutually exclusive (atomic) events
  - ▷ Union of these atomic events is the universal set whose probability is 1 — since the DMS generates only one of these sequences as $N \to \infty$

Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- With large N, the Bernouilli(q) DMS becomes a source which generates all the sequences with the same probability p_i
- The total number of such sequences K has to be such that $Kp_i = 1$. Why?
  - ▷ Generation of these binary sequences are mutually exclusive (atomic) events
  - ▷ Union of these atomic events is the universal set whose probability is 1 — since the DMS generates only one of these sequences as $N \to \infty$
  - ▷ Therefore, from Axiom 3 of the Probability Theory, $Kp_i = 1$

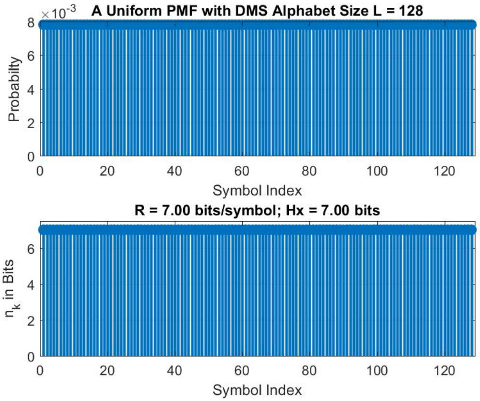
Fixed-Length Coding for Long Symbol Sequences

generated by Bernouilli(q) DMS

- With large N, the Bernouilli(q) DMS becomes a source which generates all the sequences with the same probability p_i
- The total number of such sequences K has to be such that $Kp_i = 1$. Why?
  - ▷ Generation of these binary sequences are mutually exclusive (atomic) events
  - ▷ Union of these atomic events is the universal set whose probability is 1 — since the DMS generates only one of these sequences as $N \to \infty$
  - ▷ Therefore, from Axiom 3 of the Probability Theory, $Kp_i = 1$
- Therefore $K = 1/p_i = 2^{NH_b(q)}$. These sequences — members of what is called the typical set, therefore, need $NH_b(q)$ bits for their representation, i.e., the maximum average rate per source symbol is $H_b(q)$ bits
- This completes the proof of the source coding theorem. The average number of bits per source symbol cannot be less than $H_b(q)$.

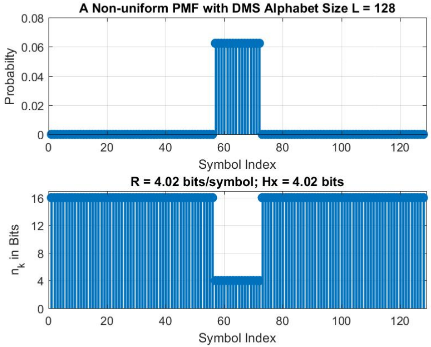
A Visual Summary

- When the DMS alphabet size is $L = 2^n$, and its PMF is uniform, the probability of each letter is $p = 1/L = 2^{-n}$, and each letter is encoded using a fixed-length code of length $-\log_2(p) = n$ bits. Shown here when $n = 7$ bits.



A Visual Summary

- Suppose $L = 128$, however, only $M = 16$ symbols actually occur realistically (these form the typical set). Remaining 112 symbols have vanishing probability of occurrence. In this case of non-uniform PMF, one needs $n = 4$ bits instead of 7 bits (using less than 4 bits will not be enough, more than 4 is wasteful)



- An arbitrary DMS with non-uniform probabilities can be converted to a case in which most of its output sequences have vanishing probability

Summary

- The importance of source coding in digital communication systems cannot be overemphasized
- We have studied the basic mathematical framework and several algorithms of source coding — a detailed study of the source coding requires at least an entire semester-long course
- Although the important concepts are covered, a lot of details are not examined (for example, the information source is rarely memoryless; a significant data compression is possible if the algorithm can model the dependence between successive source symbols)

What Next?

- The source coding is often combined with the quantization of the analog source. Advanced algorithms to represent the analog source output in an efficient manner are currently being developed/proposed
- These algorithms leverage the machine learning techniques. An efficient source encoder is an intelligent machine that can model and predict the output of the information source, such as audio, video, etc.
- These algorithms often are based on linear prediction/regression, dynamic programming and hidden Markov models, deep neural networks and related techniques. These algorithms often also utilize the Fourier transformation, discrete cosine transformation, wavelet transformation, etc.
- The research and development in this field is currently on-going; Apple SIRI, Amazon ALEXA, Google Assistant, etc. are intelligent machines that are capable of data compression since they closely model the information source