Chapter 3

Lossless Data Compression

Po-Ning Chen, Professor

Institute of Communications Engineering

National Chiao Tung University

Hsin Chu, Taiwan 30010, R.O.C.

3.1 Principles of data compression

- Average codeword length

  E.g.

$$\begin{cases} P_X(x = \text{outcome}_A) &= 0.5; \\ P_X(x = \text{outcome}_B) &= 0.25; \\ P_X(x = \text{outcome}_C) &= 0.25. \end{cases} \quad \text{and} \quad \begin{cases} \text{code}(\text{outcome}_A) &= 0; \\ \text{code}(\text{outcome}_B) &= 10; \\ \text{code}(\text{outcome}_C) &= 11. \end{cases}$$

  Then the average codeword length is

$$\begin{aligned} & \text{len}(0) \cdot P_X(A) + \text{len}(10) \cdot P_X(B) + \text{len}(11) \cdot P_X(C) \\ = \; & 1 \cdot 0.5 + 2 \cdot 0.25 + 2 \cdot 0.25 \\ = \; & 1.5 \text{ bits.} \end{aligned}$$

- Categories of codes
  - Variable-length codes
  - Fixed-length codes (often treated as a subclass of variable-length codes)
    - ∗ Segmentation is normally considered an implicit part of the codewords.

3.1 Principles of data compression

Example of segmentation of fixed-length codes.

E.g. To encode the final grades of a class with 100 students.

Assume that there are three grade levels: A, B and C.

- Without segmentation

$$\lceil \log_2 3^{100} \rceil = 159 \text{ bits.}$$

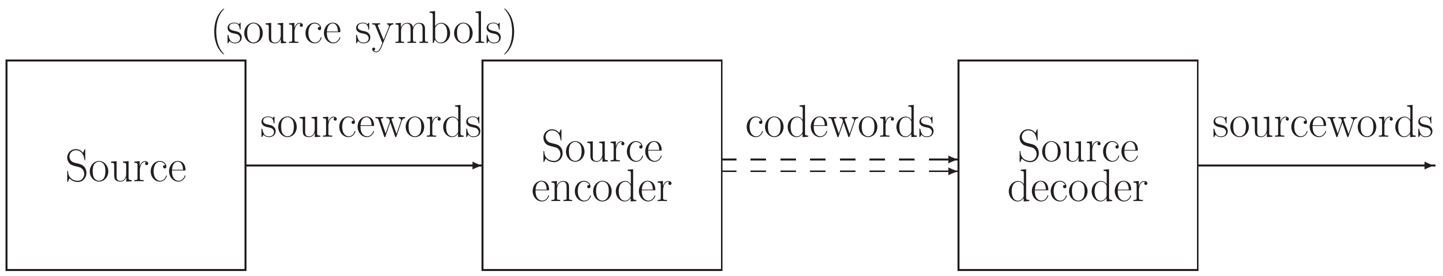
- With segmentation length of 10 students

$$10 \times \lceil \log_2 3^{10} \rceil = 160 \text{ bits.}$$

3.1 Principles of data compression

- Fixed-length codes
  - Block codes
    - ∗ The encoding (or decoding) of the next segment of source symbols is independent of the previous segments.
  - Fixed-length tree codes
    - ∗ The encoding/decoding of the next segment retains and uses some knowledge of earlier segments.

- Block diagram of a data compression system



Key difference in data compress schemes

- Block codes for *asymptotic* lossless data compression
  - Asymptotic in blocklength n
- Variable-length codes for *completely* lossless data compression

3.2.1 Block codes for DMS

Definition 3.1 (Discrete memoryless source) A discrete memoryless source (DMS) $\{X_n\}_{n=1}^{\infty}$ consists of a sequence of i.i.d. random variables, $X_1, X_2, X_3, \ldots$, all taking values in a common finite alphabet $\mathcal{X}$. In particular, if $P_X(\cdot)$ is the common distribution or probability mass function (pmf) of the X_i's, then

$$P_{X^n}(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P_X(x_i).$$

3.2.1 Block codes for DMS

Definition 3.2 An (n, M) block code with blocklength n and size M (which can be a function of n in general,[1] i.e., $M = M_n$) for a discrete source $\{X_n\}_{n=1}^{\infty}$ is a set $\mathscr{C}_n = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_M\} \subseteq \mathcal{X}^n$ consisting of M reproduction (or reconstruction) words, where each reproduction word is a sourceword (an n-tuple of source symbols).

To simplify the exposition, we make an abuse of notation by writing $\mathscr{C}_n = (n, M)$ to mean that $\mathscr{C}_n$ is a block code with blocklength n and size M.

- Note that $\boldsymbol{c}_i$ is not a "codeword" but a "reproduction word." It is an n-tuple of source symbols.
- One can binary-index (or enumerate) the reproduction words in $\mathscr{C}_n = \{\boldsymbol{c}_1$, $\boldsymbol{c}_2$, …, $\boldsymbol{c}_M\}$ using $k := \lceil \log_2 M \rceil$ bits.

[1]In the literature, both (n, M) and (M, n) have been used to denote a block code with blocklength n and size M. For example, R. W. Yeung [415, p. 149] adopts the former one, while T. M. Cover and J. A. Thomas [83, p. 193] use the latter. We use the (n, M) notation since $M = M_n$ is a function of n in general.

3.2.1 Block codes for DMS

- As such k-bit words in $\{0,1\}^k$ are usually stored for retrieval at a later date, the (n, M) block code can be represented by an encoder-decoder pair of functions (f, g):
  - the encoding function $f : \mathcal{X}^n \to \{0,1\}^k$ maps each sourceword x^n to a k-bit word $f(x^n)$, which we call a *codeword*.
  - the decoding function $g : \{0,1\}^k \to \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_M\}$ is a retrieving operation that produces the reproduction words.
- Since the codewords are binary-valued, such a block code is called a *binary code.*
- More generally, a *D-ary block code* (where $D > 1$ is an integer) would use an encoding function $f\colon \mathcal{X}^n \to \{0, 1, \ldots, D-1\}^k$ where each codeword $f(x^n)$ contains k D-ary code symbols.

3.2.1 Block codes for DMS

- Furthermore, since the behavior of block codes is investigated for sufficiently large n and M (tending to infinity), it is legitimate to replace $\lceil \log_2 M \rceil$ by $\log_2 M$ for the case of binary codes. With this convention, the *data compression rate* or *code rate* is

  $$\frac{k}{n} = \frac{1}{n}\log_2 M \text{ (in bits per source symbol).}$$

  Similarly, for D-ary codes, the rate is

  $$\frac{k}{n} = \frac{1}{n}\log_D M \text{ (in } D\text{-ary code symbols per source symbol).}$$

  For computational convenience, *nats* (under the natural logarithm) can be used instead of *bits* or *D-ary code symbols*; in this case, the code rate becomes:

  $$\frac{1}{n}\log M \text{ (in nats per source symbol).}$$

3.2.1 Block codes for DMS

- (Weakly) δ-typical set

$$\mathcal{F}_n(\delta) := \left\{ x^n \in \mathcal{X}^n : \left| -\frac{1}{n} \sum_{i=1}^{n} \log_2 P_X(x_i) - H(X) \right| < \delta \right\}.$$

E.g. $n = 2$ and $\delta = 0.4$ and $\mathcal{X} = \{A, B, C, D\}$.

$$\text{The source distribution is } \begin{cases} P_X(A) = 0.4 \\ P_X(B) = 0.3 \\ P_X(C) = 0.2 \\ P_X(D) = 0.1 \end{cases}$$

The entropy equals:

$$0.4 \log_2 \frac{1}{0.4} + 0.3 \log_2 \frac{1}{0.3} + 0.2 \log_2 \frac{1}{0.2} + 0.1 \log_2 \frac{1}{0.1} \quad = \quad 1.84644 \text{ bits}$$

Then for $x_1^2 = (A, A)$,

$$\begin{aligned} \left| -\frac{1}{2} \sum_{i=1}^{2} \log_2 P_X(x_i) - H(X) \right| &= \left| -\frac{1}{2} \left(\log_2 P_X(A) + \log_2 P_X(A) \right) - 1.84644 \right| \\ &= \left| -\frac{1}{2} \left(\log_2 0.4 + \log_2 0.4 \right) - 1.84644 \right| = 0.525 \end{aligned}$$

3.2.1 Block codes for DMS

Source	$\left\vert -\frac{1}{2}\sum_{i=1}^{2}\log_2 P_X(x_i) - H(X)\right\vert$
AA	$\underline{0.525}$ bits $\notin \mathcal{F}_2(0.4)$
AB	0.317 bits $\in \mathcal{F}_2(0.4)$
AC	0.025 bits $\in \mathcal{F}_2(0.4)$
AD	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$
BA	0.317 bits $\in \mathcal{F}_2(0.4)$
BB	0.109 bits $\in \mathcal{F}_2(0.4)$
BC	0.183 bits $\in \mathcal{F}_2(0.4)$
BD	$\underline{0.683}$ bits $\notin \mathcal{F}_2(0.4)$
CA	0.025 bits $\in \mathcal{F}_2(0.4)$
CB	0.183 bits $\in \mathcal{F}_2(0.4)$
CC	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$
CD	$\underline{0.975}$ bits $\notin \mathcal{F}_2(0.4)$
DA	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$
DB	$\underline{0.683}$ bits $\notin \mathcal{F}_2(0.4)$
DC	$\underline{0.975}$ bits $\notin \mathcal{F}_2(0.4)$
DD	$\underline{1.475}$ bits $\notin \mathcal{F}_2(0.4)$

$$\Rightarrow \mathcal{F}_2(0.4) = \{AB, AC, BA, BB, BC, CA, CB\}.$$

3.2.1 Block codes for DMS

Source	$\left\lvert -\frac{1}{2}\sum_{i=1}^{2}\log P_X(x_i) - H(X)\right\rvert$	codeword
AA	$\underline{0.525}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
AB	0.317 bits $\in \mathcal{F}_2(0.4)$	001
AC	0.025 bits $\in \mathcal{F}_2(0.4)$	010
AD	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
BA	0.317 bits $\in \mathcal{F}_2(0.4)$	011
BB	0.109 bits $\in \mathcal{F}_2(0.4)$	100
BC	0.183 bits $\in \mathcal{F}_2(0.4)$	101
BD	$\underline{0.683}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
CA	0.025 bits $\in \mathcal{F}_2(0.4)$	110
CB	0.183 bits $\in \mathcal{F}_2(0.4)$	111
CC	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
CD	$\underline{0.975}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
DA	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
DB	$\underline{0.683}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
DC	$\underline{0.975}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$
DD	$\underline{1.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$

We can therefore encode the **seven** outcomes in $\mathcal{F}_2(0.4)$ by **seven** distinct codewords, and encode all the remaining **nine** outcomes outside $\mathcal{F}_2(0.4)$ by a **single** codeword.

3.2.1 Block codes for DMS

Theorem 3.4 (Shannon1948-McMillan1953-Breiman1960) (Asymptotic equipartition property or AEP or Entropy stability property) If $\{X_n\}_{n=1}^{\infty}$ is a DMS with entropy $H(X)$, then

$$-\frac{1}{n}\log_2 P_{X^n}(X_1,\ldots,X_n) \to H(X) \quad \text{in probability.}$$

In other words, for any $\delta > 0$,

$$\lim_{n\to\infty} \Pr\left\{\left|-\frac{1}{n}\log_2 P_{X^n}(X_1,\ldots,X_n) - H(X)\right| > \delta\right\} = 0.$$

- Almost all the source sequences in $\mathcal{F}_n(\delta)$ are nearly *equiprobable* or *equally surprising* (cf. Property 1 of Theorem 3.5); Hence, Theorem 3.4 is named AEP.

  E.g. The probabilities of the elements in

  $$\mathcal{F}_2(0.3) = \{AB, AC, BA, BB, BC, CA, CB\}$$

  are respectively 0.12, 0.08, 0.12, 0.09, 0.06, 0.08 and 0.06.

  The sum of these seven probability masses are 0.61.

3.2.1 Block codes for DMS

Source	$\left\lvert -\frac{1}{2}\sum_{i=1}^{2}\log P_X(x_i) - H(X)\right\rvert$	codeword	reconstructed sequence
AA	$\underline{0.525}$ nats $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
AB	0.317 bits $\in \mathcal{F}_2(0.4)$	001	AB
AC	0.025 bits $\in \mathcal{F}_2(0.4)$	010	AC
AD	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
BA	0.317 bits $\in \mathcal{F}_2(0.4)$	011	BA
BB	0.109 bits $\in \mathcal{F}_2(0.4)$	100	BB
BC	0.183 bits $\in \mathcal{F}_2(0.4)$	101	BC
BD	$\underline{0.683}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
CA	0.025 bits $\in \mathcal{F}_2(0.4)$	110	CA
CB	0.183 bits $\in \mathcal{F}_2(0.4)$	111	CB
CC	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
CD	$\underline{0.975}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
DA	$\underline{0.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
DB	$\underline{0.683}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
DC	$\underline{0.975}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous
DD	$\underline{1.475}$ bits $\notin \mathcal{F}_2(0.4)$	$\underline{000}$	ambiguous

3.2.1 Block codes for DMS

Theorem 3.5 (Consequence of the AEP) Given a DMS $\{X_n\}_{n=1}^{\infty}$ with entropy $H(X)$ and any δ greater than zero, then the weakly δ-typical set $\mathcal{F}_n(\delta)$ satisfies the following.

1. If $x^n \in \mathcal{F}_n(\delta)$, then
$$2^{-n(H(X)+\delta)} \leq P_{X^n}(x^n) \leq 2^{-n(H(X)-\delta)}.$$
2. $P_{X^n}\left(\mathcal{F}_n^c(\delta)\right) < \delta$ for sufficiently large n, where the superscript "c" denotes the complementary set operation.
3. $|\mathcal{F}_n(\delta)| > (1-\delta)2^{n(H(X)-\delta)}$ for sufficiently large n, and $|\mathcal{F}_n(\delta)| \leq 2^{n(H(X)+\delta)}$ for every n, where $|\mathcal{F}_n(\delta)|$ denotes the number of elements in $\mathcal{F}_n(\delta)$.

Proof:

- Property 1 is an immediate consequence of the definition of $\mathcal{F}_n(\delta)$. I.e.,
$$\mathcal{F}_n(\delta) := \left\{ x^n \in \mathcal{X}^n : \left| -\frac{1}{n}\sum_{i=1}^{n} \log_2 P_X(x_i) - H(X) \right| < \delta \right\}.$$

3.2.1 Block codes for DMS

In other words,

$$\left|-\frac{1}{n}\sum_{i=1}^{n}\log_2 P_X(x_i) - H(X)\right| < \delta \Leftrightarrow \left|-\frac{1}{n}\log_2 P_{X^n}(x^n) - H(X)\right| < \delta$$

$$\Leftrightarrow \ H(X) - \delta < -\frac{1}{n}\log_2 P_{X^n}(x^n) < H(X) + \delta.$$

- Property 2 is a direct consequence of the AEP. We nevertheless provide a direct proof of Property 2. Observe that by Chebyshev's inequality,

$$P_{X^n}(\mathcal{F}_n^c(\delta)) = P_{X^n}\left\{x^n \in \mathcal{X}^n : \left|-\frac{1}{n}\log_2 P_{X^n}(x^n) - H(X)\right| > \delta\right\} \le \frac{\sigma_X^2}{n\delta^2} < \delta,$$

for $n > \sigma_X^2/\delta^3$, where the variance

$$\sigma_X^2 := \mathrm{Var}[-\log_2 P_X(X)] \quad \left(= \sum_{x\in\mathcal{X}} P_X(x)\,[\log_2 P_X(x)]^2 - (H(X))^2 \right)$$

$$\le \ E[(\log_2 P_X(X))^2] = \sum_{x\in\mathcal{X}} P_X(x)(\log_2 P_X(x))^2 \le \sum_{x\in\mathcal{X}} \max_{0\le p\le 1} p(\log_2 p)^2 \quad (p^* = \frac{1}{e^2})$$

$$= \ \sum_{x\in\mathcal{X}} \frac{4}{e^2}[\log_2(e)]^2 = \frac{4}{e^2}[\log_2(e)]^2 \times |\mathcal{X}| < \infty \quad \text{for finite alphabet}$$

is a constant independent of n.

3.2.1 Block codes for DMS

- To prove Property 3, we have from Property 1 that

$$1 \geq \sum_{x^n \in \mathcal{F}_n(\delta)} P_{X^n}(x^n) \geq \sum_{x^n \in \mathcal{F}_n(\delta)} 2^{-n(H(X)+\delta)} = |\mathcal{F}_n(\delta)| 2^{-n(H(X)+\delta)},$$

and, using Properties 1 and 2, we have that

$$1-\delta < 1-\frac{\sigma_X^2}{n\delta^2} \leq \sum_{x^n \in \mathcal{F}_n(\delta)} P_{X^n}(x^n) \leq \sum_{x^n \in \mathcal{F}_n(\delta)} 2^{-n(H(X)-\delta)} = |\mathcal{F}_n(\delta)| 2^{-n(H(X)-\delta)},$$

for $n \geq \sigma_X^2/\delta^3$.

$\square$

Shannon's Source Coding Theorem

Theorem 3.6 (Shannon's source coding theorem) Given integer $D > 1$, consider a discrete memoryless source $\{X_n\}_{n=1}^{\infty}$ with entropy $H_D(X)$. Then the following hold.

- *Forward part (achievability):* For any $0 < \varepsilon < 1$, there exists $0 < \delta < \varepsilon$ and a sequence of D-ary block codes $\{\mathscr{C}_n = (n, M_n)\}_{n=1}^{\infty}$ with

$$\limsup_{n\to\infty} \frac{1}{n} \log_D M_n \leq H_D(X) + \delta \tag{3.2.1}$$

  satisfying

$$P_e(\mathscr{C}_n) < \varepsilon \tag{3.2.2}$$

  for all sufficiently large n, where $P_e(\mathscr{C}_n)$ denotes the probability of decoding error for block code $\mathscr{C}_n$.

- *Strong converse part:* For any $0 < \varepsilon < 1$, any sequence of D-ary block codes $\{\mathscr{C}_n = (n, M_n)\}_{n=1}^{\infty}$ with

$$\limsup_{n\to\infty} \frac{1}{n} \log_D M_n < H_D(X) \tag{3.2.3}$$

  satisfies

$$P_e(\mathscr{C}_n) > 1 - \varepsilon$$

  for all n sufficiently large.

Shannon's Source Coding Theorem

Note

- Since ε can be made arbitrarily small, (3.2.2) is equivalent to

$$\limsup_{n\to\infty} P_e(\mathcal{C}_n) = 0.$$

- In parallel, (3.2.3) is equivalent to

$$\limsup_{n\to\infty} P_e(\mathcal{C}_n) = 1.$$

Keys of the proof of the forward part

- Only need to prove the existence of such block code.
- The code chosen is indeed the weakly δ-typical set.

Shannon's Source Coding Theorem

Proof:

Forward Part:

- Without loss of generality, we will prove the result for the case of binary codes (i.e., $D = 2$). Also recall that subscript D in $H_D(X)$ will be dropped (i.e., omitted) specifically when $D = 2$.
- Given $0 < \varepsilon < 1$, fix δ such that $0 < \delta < \varepsilon$ and choose $n > 2/\delta$.
- Binary-index the sourcewords in $\mathcal{F}_n(\delta/2)$ with the following encoding map:

$$\begin{cases} x^n \to \text{binary index of } x^n, & \text{if } x^n \in \mathcal{F}_n(\delta/2); \\ x^n \to \text{all-zero codeword}, & \text{if } x^n \notin \mathcal{F}_n(\delta/2). \end{cases}$$

  Then by the Shannon-McMillan-Breiman AEP theorem (spec., the 3rd property in Theorem 3.5), we obtain that

$$M_n = |\mathcal{F}_n(\delta/2)|+1 \le 2^{n(H(X)+\delta/2)}+1 < 2\cdot 2^{n(H(X)+\delta/2)} < 2^{n\delta/2}\cdot 2^{n(H(X)+\delta/2)} = 2^{n(H(X)+\delta)},$$

  for $n > 2/\delta$. Hence, a sequence of $\mathscr{C}_n = (n, M_n)$ block code satisfying (3.2.1) is established.

- It remains to show that the error probability for this sequence of (n, M_n) block code can be made smaller than ε for all sufficiently large n.

Shannon's Source Coding Theorem

By the Shannon-McMillan-Breiman AEP theorem (spec., the 2nd property in Theorem 3.5),

$$P_{X^n}(\mathcal{F}_n^c(\delta/2)) < \frac{\delta}{2} \quad \text{for all sufficiently large } n.$$

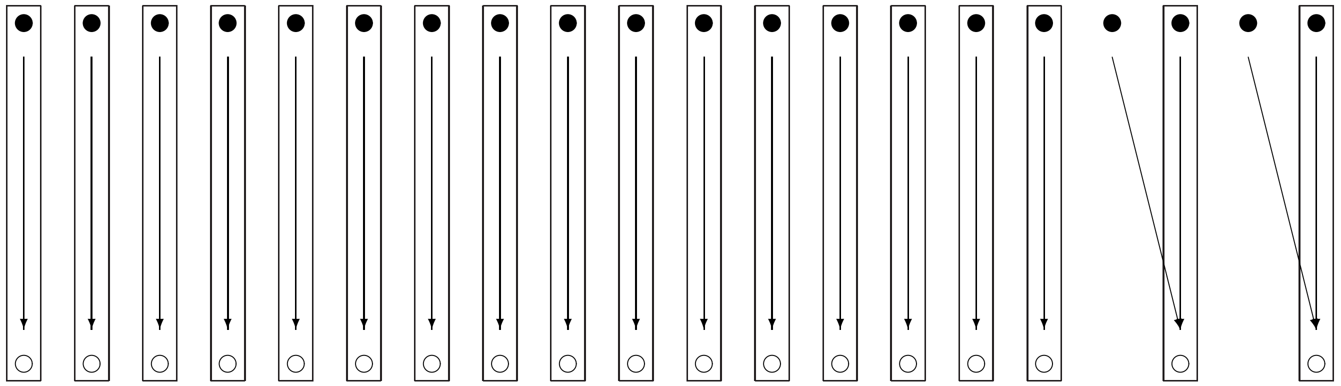
Consequently, for those n satisfying the above inequality, and being bigger than $2/\delta$,

$$P_e(\mathscr{C}_n) \leq P_{X^n}(\mathcal{F}_n^c(\delta/2)) < \delta \leq \varepsilon.$$

(See Slide I: 3-12 to confirm that only the "ambiguous" sequences outside the typical set contribute to the probability of error.)

Shannon's Source Coding Theorem

Set of Source Symbols $\mathcal{S}_n$



Codeword Set $\mathcal{C}_n$

Strong Converse Part:

- Fix any sequence of block codes $\{\mathcal{C}_n\}_{n=1}^{\infty}$ with

$$\limsup_{n\to\infty} \frac{1}{n}\log_2|\mathcal{C}_n| < H(X).$$

Let $\mathcal{S}_n$ be the set of source symbols that can be correctly decoded through $\mathcal{C}_n$-coding system. (A quick example is depicted above.) Then $|\mathcal{S}_n| = |\mathcal{C}_n|$.

Shannon's Source Coding Theorem

- By choosing δ small enough with $\varepsilon/2 > \delta > 0$, and also by definition of limsup operation, we have

$$(\exists\ N_0)(\forall\ n > N_0) \quad \frac{1}{n}\log_2|\mathcal{S}_n| = \frac{1}{n}\log_2|\mathcal{C}_n| < H(X) - 2\delta,$$

which implies

$$|\mathcal{S}_n| < 2^{n(H(X)-2\delta)}.$$

Shannon's Source Coding Theorem

- Furthermore, from Property 2 of the Consequence of the AEP, we obtain that

$$(\exists\ N_1)(\forall\ n > N_1) \quad P_{X^n}(\mathcal{F}_n^c(\delta)) < \delta.$$

Consequently, for $n > N := \max\{N_0, N_1, \log_2(2/\varepsilon)/\delta\}$, the probability of correctly block decoding satisfies

$$\begin{aligned}
1 - P_e(\mathscr{C}_n) &= \sum_{x^n \in \mathcal{S}_n} P_{X^n}(x^n) \\
&= \sum_{x^n \in \mathcal{S}_n \cap \mathcal{F}_n^c(\delta)} P_{X^n}(x^n) + \sum_{x^n \in \mathcal{S}_n \cap \mathcal{F}_n(\delta)} P_{X^n}(x^n) \\
&\le P_{X^n}(\mathcal{F}_n^c(\delta)) + |\mathcal{S}_n \cap \mathcal{F}_n(\delta)| \cdot \max_{x^n \in \mathcal{F}_n(\delta)} P_{X^n}(x^n) \\
&< \delta + |\mathcal{S}_n| \cdot \max_{x^n \in \mathcal{F}_n(\delta)} P_{X^n}(x^n) \\
&< \frac{\varepsilon}{2} + 2^{n(H(X)-2\delta)} \cdot 2^{-n(H(X)-\delta)} \\
&= \frac{\varepsilon}{2} + 2^{-n\delta} \\
&< \varepsilon,
\end{aligned}$$

which is equivalent to $P_e(\mathscr{C}_n) > 1 - \varepsilon$ for $n > N$.

□

Code Rates for Data Compression

Notes

- Ultimate data compression rate

$$R := \limsup_{n\to\infty} \frac{1}{n} \log_2 M_n \text{ nats per source symbol.}$$

- Shannon's source coding theorem
  - Arbitrary good performance can be achieved by **extending the block-length**.

$$\big(\forall\, \varepsilon > 0 \text{ and } 0 < \delta < \varepsilon\big)(\exists\, \mathscr{C}_n) \text{ such that } \frac{1}{n}\log_2 M_n < H(X)+\delta \text{ and } P_e(\mathscr{C}_n) < \varepsilon.$$

  So $R = \limsup_{n\to\infty} \frac{1}{n} \log_2 M_n$ can be made smaller than $H(X) + \delta$ for arbitrarily small δ.

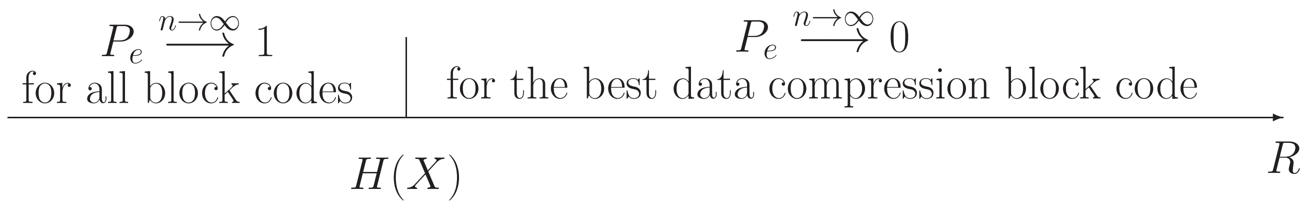
  In other words, at rate $R < H(X) + \delta$ for arbitrarily small $\delta > 0$, the error probability can be made *arbitrarily close to zero* $(< \varepsilon)$.

- How about further making $R < H(X)$? Answer:

$$\left(\forall\ \{\mathscr{C}_n\}_{n\ge 1} \text{ with } \limsup_{n\to\infty} \frac{1}{n}\log_2 |\mathscr{C}_n| < H(X)\right) \quad P_e(\mathscr{C}_n) \to 1.$$

Summary of Shannon's Source Coding Theorem

- Behavior of error probability as blocklength $n \to \infty$ for a DMS


$P_e \xrightarrow{n\to\infty} 1$
for all block codes
$P_e \xrightarrow{n\to\infty} 0$
for the best data compression block code
$H(X)$
R

- Key to the achievability proof

Existence of a typical-like set $\mathcal{A}_n = \{x_1^n, x_2^n, \ldots, x_M^n\}$ with $M \approx 2^{nH(X)}$ and $P_{X^n}(\mathcal{A}_n^c) \to 0$ (or $P_{X^n}(\mathcal{A}_n) \to 1$.)

In other words,

Existence of a typical-like set $\mathcal{A}_n$ whose size is prohibitively small, and whose probability mass is large.

  – This is the basic idea for the generalization of Shannon's source coding theorem to a more general (than i.i.d.) source.

Summary of Shannon's Source Coding Theorem

- Notes to the strong converse theorem
  - It is named the *strong converse theorem* because the result is very *strong*.
    - All code sequences with $R < H(X)$ have error probability approaching 1!
      - Of course, you can always design a lousy code with error probability approaching 1. Here, what the theorem truly claims is that all (sequences of) designs are (asymptotically) *lousy*.
  - The strong converse theorem applies to all stationary-ergodic sources.
- A weak converse statement (than the strong converse) is:
  - For general sources, such as non-stationary non-ergodic sources, we can find some code sequence with $R < H(\mathcal{X})$ whose error probability is only bounded away from zero, and does not approach 1 at all. Notably, for general sources, $H(\mathcal{X})$ is no longer in the form of a single-letter entropy $H(X)$ but the *entropy rate* $\frac{1}{n}H(X^n)$ or *sup-entropy rate* $\bar{H}(\boldsymbol{X})$.

3.2.2 Block Codes for Stationary Ergodic Sources

- Recall that the merit of the *stationary ergodic* assumption is on its validity of *law of large numbers*.
- In order to extend the Shannon's source coding theorem to stationary ergodic sources, we need to generalize the information measure for such sources.

Definition 3.8 (Entropy rate) The *entropy rate* for a source $\{X_n\}_{n=1}^{\infty}$ is denoted by $H(\mathcal{X})$ and defined by

$$H(\mathcal{X}) := \lim_{n\to\infty} \frac{1}{n} H(X^n)$$

provided the limit exists, where $X^n = (X_1, \cdots, X_n)$.

- **Comment:** The limit of $\lim_{n\to\infty} \frac{1}{n} H(X^n)$ exists for all stationary sources.

3.2.2 Block Codes for Stationary Ergodic Sources

Lemma 3.9 For a stationary source $\{X_n\}_{n=1}^{\infty}$, the conditional entropy

$$H(X_n|X_{n-1},\ldots,X_1)$$

is nonincreasing in n and also bounded from below by zero. Hence by Lemma A.20, the limit

$$\lim_{n\to\infty} H(X_n|X_{n-1},\ldots,X_1)$$

exists.

Proof: We have

$$\begin{aligned} H(X_n|X_{n-1},\ldots,X_1) &\le H(X_n|X_{n-1},\ldots,X_2) && (3.2.4)\\ &= H(X_n,\cdots,X_2) - H(X_{n-1},\cdots,X_2) \\ &= H(X_{n-1},\cdots,X_1) - H(X_{n-2},\cdots,X_1) && (3.2.5)\\ &= H(X_{n-1}|X_{n-2},\ldots,X_1) \end{aligned}$$

where (3.2.4) follows since conditioning never increases entropy, and (3.2.5) holds because of the stationarity assumption. Finally, recall that each conditional entropy $H(X_n|X_{n-1},\ldots,X_1)$ is nonnegative. $\square$

3.2.2 Block Codes for Stationary Ergodic Sources

Lemma 3.10 (Cesaro-mean theorem) If $a_n \to a$ as $n \to \infty$ and $b_n = (1/n)\sum_{i=1}^{n} a_i$, then $b_n \to a$ as $n \to \infty$.

Proof: $a_n \to a$ implies that for any $\varepsilon > 0$, there exists N such that for all $n > N$, $|a_n - a| < \varepsilon$. Then

$$\begin{aligned}
|b_n - a| &= \left|\frac{1}{n}\sum_{i=1}^{n}(a_i - a)\right| \\
&\le \frac{1}{n}\sum_{i=1}^{n}|a_i - a| \\
&= \frac{1}{n}\sum_{i=1}^{N}|a_i - a| + \frac{1}{n}\sum_{i=N+1}^{n}|a_i - a| \\
&\le \frac{1}{n}\sum_{i=1}^{N}|a_i - a| + \frac{n-N}{n}\varepsilon.
\end{aligned}$$

Hence, $\limsup_{n\to\infty}|b_n - a| \le \varepsilon$. Since ε can be made arbitrarily small, the lemma holds. □

3.2.2 Block Codes for Stationary Ergodic Sources

Theorem 3.11 The entropy rate of a stationary source $\{X_n\}_{n=1}^{\infty}$ always exist and is equal to

$$H(\mathcal{X}) = \lim_{n\to\infty} H(X_n|X_{n-1},\ldots,X_1).$$

Proof: The result directly follows by writing

$$\frac{1}{n}H(X^n) = \frac{1}{n}\sum_{i=1}^{n} H(X_i|X_{i-1},\ldots,X_1) \quad \text{(chain-rule for entropy)}$$

and applying the Cesaro-mean theorem. □

Observation 3.12 It can also be shown that for a stationary source, $(1/n)H(X^n)$ is nonincreasing in n and $(1/n)H(X^n) \geq H(X_n|X_{n-1},\ldots,X_1)$ for all $n \geq 1$. (The proof is left as an exercise. See Problem 3.)

Practices of Finding the Entropy Rate

- I.i.d. source

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X^n) = H(X)$$

since $H(X^n) = n \times H(X)$ for every n.

- First-order stationary Markov source

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X^n) = \lim_{n\to\infty} H(X_n|X_{n-1}, \ldots, X_1) = H(X_2|X_1),$$

where

$$H(X_2|X_1) := -\sum_{x_1\in\mathcal{X}} \sum_{x_2\in\mathcal{X}} \pi(x_1) P_{X_2|X_1}(x_2|x_1) \cdot \log P_{X_2|X_1}(x_2|x_1),$$

and $\pi(\cdot)$ is the stationary distribution for the Markov source.

  - In addition, if the Markov source is also *binary*,

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X^n) = \frac{\beta}{\alpha+\beta} H_b(\alpha) + \frac{\alpha}{\alpha+\beta} H_b(\beta),$$

where $H_b(\alpha) := -\alpha \log \alpha - (1-\alpha)\log(1-\alpha)$ is the binary entropy function, and $P_{X_2|X_1}(0|1) = \alpha$ and $P_{X_2|X_1}(1|0) = \beta$

Shannon's Source Coding Theorem Revisited

Theorem 3.14 (Generalized AEP or Shannon-McMillan-Breiman Theorem [12]) If $\{X_n\}_{n=1}^{\infty}$ is a stationary ergodic source, then

$$-\frac{1}{n}\log_2 P_{X^n}(X_1,\ldots,X_n) \xrightarrow{a.s.} H(\mathcal{X}).$$

Theorem 3.15 (Shannon's source coding theorem for stationary ergodic sources) Given integer $D > 1$, let $\{X_n\}_{n=1}^{\infty}$ be a stationary ergodic source with entropy rate (in base D)

$$H_D(\mathcal{X}) := \lim_{n\to\infty}\frac{1}{n}H_D(X^n).$$

Then the following hold.

- *Forward part (achievability):* For any $0 < \varepsilon < 1$, there exists δ with $0 < \delta < \varepsilon$ and a sequence of D-ary block codes $\{\mathscr{C}_n = (n, M_n)\}_{n=1}^{\infty}$ with

  $$\limsup_{n\to\infty}\frac{1}{n}\log_D M_n < H_D(\mathcal{X}) + \delta,$$

  and probability of decoding error satisfied

  $$P_e(\mathscr{C}_n) < \varepsilon$$

  for all sufficiently large n.

Shannon's Source Coding Theorem Revisited

- *Strong converse part:* For any $0 < \varepsilon < 1$, any sequence of D-ary block codes $\{\mathscr{C}_n = (n, M_n)\}_{n=1}^{\infty}$ with

$$\limsup_{n\to\infty} \frac{1}{n} \log_D M_n < H_D(\mathcal{X})$$

satisfies

$$P_e(\mathscr{C}_n) > 1 - \varepsilon$$

for all n sufficiently large.

Problems of Ergodicity Assumption

- A discrete memoryless (i.i.d.) source is stationary and ergodic.
- In general, it is hard to check whether a process is ergodic or not.
- If a stationary process is a "non-trivial" (see below where $0 < \theta < 1$) mixture of two or more "asymptotically mutually singular" stationary ergodic processes, i.e., its n-fold distribution can be written as the weighted sum of the n-fold distributions of "distinct" stationary ergodic processes, then it is not ergodic.
  - For example, let P and Q be two distinct distributions on a finite alphabet $\mathcal{X}$ such that the process $\{X_n\}_{n=1}^{\infty}$ is i.i.d. with distribution P and the process $\{Y_n\}_{n=1}^{\infty}$ is i.i.d. with distribution Q. Flip a biased coin (with Heads probability equal to θ, $0 < \theta < 1$) *once* and let

$$Z_i = \begin{cases} X_i & \text{if Heads} \\ Y_i & \text{if Tails} \end{cases}$$

for $i = 1, 2, \cdots$. Then the resulting process $\{Z_i\}_{i=1}^{\infty}$ has its n-fold distribution as a mixture of the n-fold distributions of $\{X_n\}_{n=1}^{\infty}$ and $\{Y_n\}_{n=1}^{\infty}$:

$$P_{Z^n}(a^n) = \theta P_{X^n}(a^n) + (1-\theta) P_{Y^n}(a^n)$$

for all $a^n \in \mathcal{X}^n$, $n = 1, 2, \cdots$. Then the process $\{Z_i\}_{i=1}^{\infty}$ is stationary but *not ergodic*.

Problems of Ergodicity Assumption

- A specific case that ergodicity can be easily verified is the case of stationary Markov sources.

  Observation

  1. An irreducible finite-state stationary Markov source is ergodic.
     - Note that irreducibility can be verified in terms of the transition probability matrix. For example, all the entries in transition probability matrix are non-zero.
  2. Hence, the generalized AEP theorem holds for irreducible finite-state stationary Markov sources. For example, if the Markov source is of the first-order, then

$$-\frac{1}{n}\log P_{X^n}(X^n) \xrightarrow{a.s.} \lim_{n\to\infty}\frac{1}{n}H(X^n) = H(X_2|X_1).$$

Redundancy for Lossless Data Compression

- A source can be compressed only when it has redundancy.
  - A very important concept is that **the output of a perfect lossless data compressor should be (asymptotic) i.i.d. with (asymptotic) uniform marginal distribution** (so that its entropy rate is $\log_2 |\mathcal{X}|$). Because if it were not so, there would be redundancy in the output and hence the compressor cannot be claimed **perfect**.
- This arises the need to define the redundancy of a (stationary ergodic) source.
- Categories of redundancy
  - intra-sourceword redundancy
    - ∗ due to non-uniform marginal distribution
  - inter-sourceword redundancy
    - ∗ due to the source memory

Redundancy for Lossless Data Compression

Definition (Redundancy)

1. Source redundancy due to the *non-uniformity of the source marginal distribution* ρ_d:
$$\rho_d := \log_2 |\mathcal{X}| - H(X_1).$$
2. Source redundancy due to the *source memory* ρ_m:
$$\rho_m := H(X_1) - H(\mathcal{X}).$$
3. Hence, the source total redundancy ρ_t is given by:
$$\rho_t := \rho_d + \rho_m = \log_2 |\mathcal{X}| - H(\mathcal{X}).$$

Redundancy for Lossless Data Compression

E.g.

Source	ρ_d	ρ_m	ρ_t
i.i.d. uniform	0	0	0
i.i.d. non-uniform	$\log_2 \lvert\mathcal{X}\rvert - H(X_1)$	0	ρ_d
1st-order symmetric Markov	0	$H(X_1) - H(X_2 \vert X_1)$	ρ_m
1st-order non-symmetric Markov	$\log_2 \lvert\mathcal{X}\rvert - H(X_1)$	$H(X_1) - H(X_2 \vert X_1)$	$\rho_d + \rho_m$

- A first-order Markov process is symmetric if for any x_1 and $\hat{x}_1$,

$$\{a \ : \ a = P_{X_2|X_1}(y|x_1) \text{ for some } y\} = \{a \ : \ a = P_{X_2|X_1}(y|\hat{x}_1) \text{ for some } y\}.$$

3.3 Variable-Length Code for Lossless Data Comp.

3.3.1 Non-singular Codes and Uniquely Decodable Codes

- Non-singular codes
  - To encode all sourcewords with distinct variable-length codewords
- Uniquely decodable codes
  - Concatenation of codewords (without punctuation mechanism) can be uniquely decodable.

    E.g., a non-singular but non-uniquely decodable code

$$\begin{aligned}\text{code of } A &= 0,\\ \text{code of } B &= 1,\\ \text{code of } C &= 00,\\ \text{code of } D &= 01,\\ \text{code of } E &= 10,\\ \text{code of } F &= 11.\end{aligned}$$

The code is not uniquely decodable because the codeword sequence, 01, can be reconstructed as either AB or D.

3.3 Variable-Length Code for Lossless Data Comp.

Definition 3.19 Consider a discrete source $\{X_n\}_{n=1}^{\infty}$ with finite alphabet $\mathcal{X}$ along with a D-ary code alphabet $\mathcal{B} = \{0, 1, \cdots, D-1\}$, where $D > 1$ is an integer. Fix integer $n \geq 1$, then a *D-ary n-th order variable-length code* (VLC) is a map

$$f : \mathcal{X}^n \to \mathcal{B}^*$$

mapping (fixed-length) sourcewords of length n to D-ary codewords in $\mathcal{B}^*$ of variable lengths, where $\mathcal{B}^*$ denotes the set of all finite-length strings from $\mathcal{B}$ (i.e., $c \in \mathcal{B}^* \Leftrightarrow \exists$ integer $l \geq 1$ such that $c \in \mathcal{B}^l$).

The *codebook* $\mathcal{C}$ of a VLC is the set of all codewords:

$$\mathcal{C} = f(\mathcal{X}^n) = \{f(x^n) \in \mathcal{B}^* : x^n \in \mathcal{X}^n\}.$$

3.3 Variable-Length Code for Lossless Data Comp.

Definition 3.20 Let $\mathcal{C}$ be a D-ary n-th order VLC code

$$f : \mathcal{X}^n \to \{0, 1, \cdots, D-1\}^*$$

for a discrete source $\{X_n\}_{n=1}^{\infty}$ with alphabet $\mathcal{X}$ and distribution $P_{X^n}(x^n)$, $x^n \in \mathcal{X}^n$. Setting $\ell(\boldsymbol{c}_{x^n})$ as the length of the codeword $\boldsymbol{c}_{x^n} = f(x^n)$ associated with sourceword x^n, then the *average codeword length* for $\mathcal{C}$ is given by

$$\bar{\ell} := \sum_{x^n \in \mathcal{X}^n} P_{X^n}(x^n) \ell(\boldsymbol{c}_{x^n})$$

and its *average code rate* (in D-ary code symbols/source symbol) is given by

$$\overline{R}_n := \frac{\bar{\ell}}{n} = \frac{1}{n} \sum_{x^n \in \mathcal{X}^n} P_{X^n}(x^n) \ell(\boldsymbol{c}_{x^n}).$$

Theorem 3.21 (Kraft1949-McMillan1956 inequality for uniquely decodable codes) Let $\mathcal{C}$ be a uniquely decodable D-ary n-th order VLC for a discrete source $\{X_n\}_{n=1}^{\infty}$ with alphabet $\mathcal{X}$. Let the $M = |\mathcal{X}|^n$ codewords of $\mathcal{C}$ have lengths $\ell_1, \ell_2, \ldots, \ell_M$, respectively. Then the following inequality must hold

$$\sum_{m=1}^{M} D^{-\ell_m} \leq 1.$$

Proof: Suppose that we use the codebook $\mathcal{C}$ to encode N sourcewords ($x_i^n \in \mathcal{X}^n$, $i = 1, \cdots, N$) arriving in a sequence; this yields a concatenated codeword sequence

$$\boldsymbol{c}_1\boldsymbol{c}_2\boldsymbol{c}_3 \ldots \boldsymbol{c}_N.$$

Let the lengths of the codewords be respectively denoted by

$$\ell(\boldsymbol{c}_1), \ell(\boldsymbol{c}_2), \ldots, \ell(\boldsymbol{c}_N).$$

Consider

$$\left(\sum_{\boldsymbol{c}_1\in\mathcal{C}}\sum_{\boldsymbol{c}_2\in\mathcal{C}}\cdots\sum_{\boldsymbol{c}_N\in\mathcal{C}} D^{-[\ell(\boldsymbol{c}_1)+\ell(\boldsymbol{c}_2)+\cdots+\ell(\boldsymbol{c}_N)]}\right).$$

It is obvious that the above expression is equal to

$$\left(\sum_{\boldsymbol{c}\in\mathcal{C}} D^{-\ell(\boldsymbol{c})}\right)^N = \left(\sum_{m=1}^{M} D^{-\ell_m}\right)^N.$$

(Note that $|\mathcal{C}| = M$.) On the other hand, all the code sequences with length

$$i = \ell(\boldsymbol{c}_1) + \ell(\boldsymbol{c}_2) + \cdots + \ell(\boldsymbol{c}_N)$$

contribute equally to the sum of the identity, which is D^{-i}. Let A_i denote the number of N-codeword sequences that have length i. Then the above identity can be re-written as

$$\left(\sum_{m=1}^{M} D^{-\ell_m}\right)^N = \sum_{i=1}^{LN} A_i D^{-i}, \quad \text{where} \quad L := \max_{\boldsymbol{c}\in\mathcal{C}} \ell(\boldsymbol{c}).$$

Since $\mathcal{C}$ is by assumption a uniquely decodable code, the codeword sequence must be unambiguously decodable. Observe that a code sequence with length i has at most D^i unambiguous combinations. Therefore, $A_i \le D^i$, and

$$\left(\sum_{m=1}^{M} D^{-\ell_m}\right)^N = \sum_{i=1}^{LN} A_i D^{-i} \le \sum_{i=1}^{LN} D^i D^{-i} = LN,$$

which implies that

$$\sum_{m=1}^{M} D^{-\ell_m} \le (LN)^{1/N}.$$

The proof is completed by noting that the above inequality holds for every N, and the upper bound $(LN)^{1/N}$ goes to 1 as N goes to infinity. □

Source Coding Theorem for Variable-Length Code

Theorem 3.22 The average rate of every uniquely decodable D-ary n-th order VLC for a discrete memoryless source $\{X_n\}_{n=1}^{\infty}$ is lower-bounded by the source entropy $H_D(X)$ (measured in D-ary code symbols/source symbol).

Proof: Consider a uniquely decodable D-ary n-th order VLC code for the source $\{X_n\}_{n=1}^{\infty}$

$$f : \mathcal{X}^n \to \{0, 1, \cdots, D-1\}^*$$

and let $\ell(\boldsymbol{c}_{x^n})$ denote the length of the codeword $\boldsymbol{c}_{x^n} = f(x^n)$ for sourceword x^n.

Source Coding Theorem for Variable-Length Code

Hence,

$$\begin{aligned}
\overline{R}_n - H_D(X) &= \frac{1}{n}\sum_{x^n\in\mathcal{X}^n} P_{X^n}(x^n)\ell(\boldsymbol{c}_{x^n}) - \frac{1}{n}H_D(X^n)\\
&= \frac{1}{n}\left[\sum_{x^n\in\mathcal{X}^n} P_{X^n}(x^n)\ell(\boldsymbol{c}_{x^n}) - \sum_{x^n\in\mathcal{X}^n}(-P_{X^n}(x^n)\log_D P_{X^n}(x^n))\right]\\
&= \frac{1}{n}\sum_{x^n\in\mathcal{X}^n} P_{X^n}(x^n)\log_D\frac{P_{X^n}(x^n)}{D^{-\ell(\boldsymbol{c}_{x^n})}}\\
&\geq \frac{1}{n}\left[\sum_{x^n\in\mathcal{X}^n} P_{X^n}(x^n)\right]\log_D\frac{\left[\sum_{x^n\in\mathcal{X}^n} P_{X^n}(x^n)\right]}{\left[\sum_{x^n\in\mathcal{X}^n} D^{-\ell(\boldsymbol{c}_{x^n})}\right]}\\
&\quad \text{(log-sum inequality)}\\
&= -\frac{1}{n}\log\left[\sum_{x^n\in\mathcal{X}^n} D^{-\ell(\boldsymbol{c}_{x^n})}\right]\\
&\geq 0
\end{aligned}$$

where the last inequality follows from the Kraft inequality for uniquely decodable codes and the fact that the logarithm is a strictly increasing function. □

Summary for Unique Decodability

1. Uniquely decodability $\Rightarrow$ the Kraft inequality holds.
2. Uniquely decodability $\Rightarrow$ average code rate of VLCs for memoryless sources is lower bounded by the source entropy.

Exercise 3.23

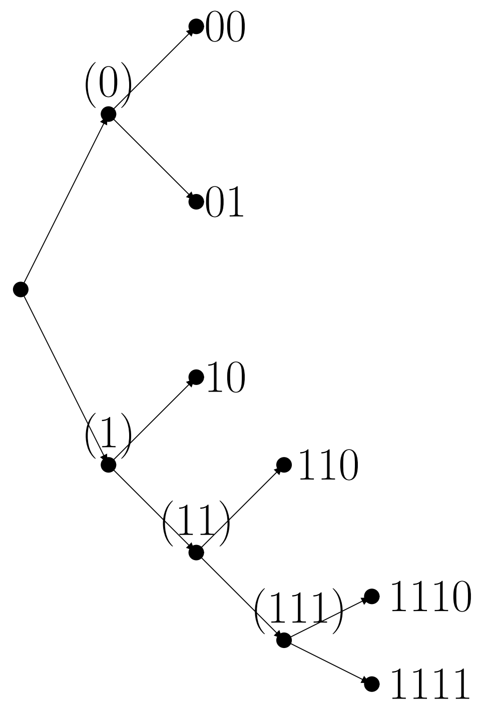
1. Find a non-singular and also non-uniquely decodable code that violates the Kraft inequality. (Hint: Slide I: 3-38.)
2. Find a non-singular and also non-uniquely decodable code that beats the entropy lower bound. (Hint: Same as the previous one.)

3.3.2 Prefix or Instantaneous Codes

- Prefix codes or instantaneous codes
  - A special case of uniquely decodable codes
  - Note that a uniquely decodable code may not necessarily be decoded instantaneously.

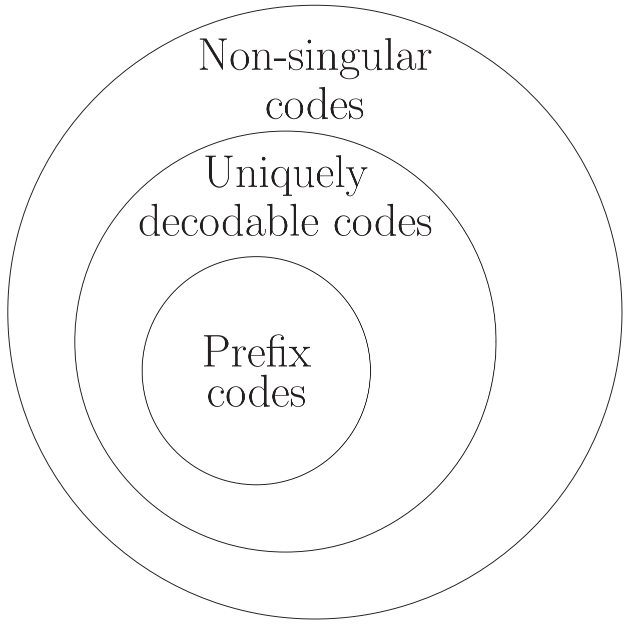
Definition 3.24 A code is called a *prefix(-free) code* or an *instantaneous code* if no codeword is a prefix of any other codeword.

Tree Representation of Prefix Codes



The codewords are those residing on the leaves, which in this case are 00, 01, 10, 110, 1110 and 1111.

Classification of Variable-Length Codes



Prefix Code to Kraft Inequality

Theorem 3.25 (Kraft inequality for prefix codes) There exists a D-ary nth-order prefix code for a discrete source $\{X_n\}_{n=1}^{\infty}$ with alphabet $\mathcal{X}$ if, and only if, the codewords of length ℓ_m, $m = 1, \ldots, M$, satisfy the Kraft inequality, where $M = |\mathcal{X}|^n$.

Proof: Without loss of generality, we provide the proof for the case of $D = 2$ (binary codes).

1. [**The forward part**] *Prefix codes satisfy the Kraft inequality.*

The codewords of a prefix code can always be put on a tree. Pick up a length

$$\ell_{\max} := \max_{1 \le m \le M} \ell_m.$$

- A tree has originally $2^{\ell_{\max}}$ nodes on level $\ell_{\max}$.
- Each codeword of length ℓ_m obstructs $2^{\ell_{\max}-\ell_m}$ nodes on level $\ell_{\max}$.
  - When any node is chosen as a codeword, all its children will be excluded from being codewords.
  - There are exactly $2^{\ell_{\max}-\ell_m}$ excluded nodes on level $\ell_{\max}$ of the tree. We therefore say that each codeword of length ℓ_m obstructs $2^{\ell_{\max}-\ell_m}$ nodes on level $\ell_{\max}$.

Prefix Code to Kraft Inequality

- Note that no two codewords obstruct the same nodes on level $\ell_{\max}$. Hence the number of totally obstructed codewords on level $\ell_{\max}$ should be no larger than $2^{\ell_{\max}}$, i.e.,

$$\sum_{m=1}^{M} 2^{\ell_{\max}-\ell_m} \le 2^{\ell_{\max}},$$

which immediately implies the Kraft inequality:

$$\sum_{m=1}^{M} 2^{-\ell_m} \le 1.$$

This part can also be proven by stating the fact that a prefix code is a uniquely decodable code. The objective of adding this proof is to illustrate the characteristics of a tree-like prefix code.

Prefix Code to Kraft Inequality

2. [**The converse part**] *Kraft inequality implies the existence of a prefix code.*

Suppose that $\ell_1, \ell_2, \ldots, \ell_M$ satisfy the Kraft inequality. We will show that there exists a binary tree with M selected nodes where the i^{th} node resides on level ℓ_i.

- Let n_i be the number of nodes (among the M nodes) residing on level i (namely, n_i is the number of codewords with length i or $n_i = |\{m : \ell_m = i\}|$), and let

$$\ell_{\max} := \max_{1 \le m \le M} \ell_m.$$

- Then from the Kraft inequality, we have

$$n_1 2^{-1} + n_2 2^{-2} + \cdots + n_{\ell_{\max}} 2^{-\ell_{\max}} \le 1.$$

- The above inequality can be re-written in a form that is more suitable for this proof as:

$$\begin{aligned}
&n_1 2^{-1} \le 1 \\
&n_1 2^{-1} + n_2 2^{-2} \le 1 \\
&\cdots \\
&n_1 2^{-1} + n_2 2^{-2} + \cdots + n_{\ell_{\max}} 2^{-\ell_{\max}} \le 1.
\end{aligned}$$

Prefix Code to Kraft Inequality

Hence,

$$\begin{aligned}
&n_1 \leq 2 \\
&n_2 \leq 2^2 - n_1 2^1 \\
&\cdots \\
&n_{\ell_{\max}} \leq 2^{\ell_{\max}} - n_1 2^{\ell_{\max}-1} - \cdots - n_{\ell_{\max}-1} 2^1,
\end{aligned}$$

which can be interpreted in terms of a tree model as:

- the 1st inequality says that the number of codewords of length 1 is less than the available number of nodes on the 1st level, which is 2.
- The 2nd inequality says that the number of codewords of length 2 is less than the total number of nodes on the 2nd level, which is 2^2, minus the number of nodes obstructed by the 1st level nodes already occupied by codewords.
- The succeeding inequalities demonstrate the availability of a sufficient number of nodes at each level after the nodes blocked by shorter length codewords have been removed.
- Because this is true at every codeword length up to the maximum codeword length, the assertion of the theorem is proved. $\square$

Source Coding Theorem for Variable-Length Codes

Corollary 3.26 A uniquely decodable D-ary n-th order code can always be replaced by a D-ary n-th order prefix code with the same average codeword length (and hence the same average code rate).

Proof: Uniquely decodability

$\Rightarrow$ the Kraft inequality holds.

$\Rightarrow$ [**The converse part**] *Kraft inequality implies the existence of a prefix code.*

Theorem 3.27 Consider a discrete memoryless source $\{X_n\}_{n=1}^{\infty}$.

1. For any D-ary n-th order prefix code for the source, the average code rate is no less than the source entropy $H_D(X)$.
2. There must exist a D-ary n-th order prefix code for the source whose average code rate is no greater than $H_D(X) + \frac{1}{n}$, namely,

$$\overline{R}_n := \frac{1}{n} \sum_{x^n \in \mathcal{X}^n} P_{X^n}(x^n) \ell(\boldsymbol{c}_{x^n}) \leq H_D(X) + \frac{1}{n}, \tag{3.3.1}$$

where $\boldsymbol{c}_{x^n}$ is the codeword for sourceword x^n, and $\ell(\boldsymbol{c}_{x^n})$ is the length of codeword $\boldsymbol{c}_{x^n}$.

Source Coding Theorem for Variable-Length Codes

Proof: A prefix code is uniquely decodable, and hence it directly follows from Theorem 3.22 that its average code rate is no less than the source entropy.

To prove the second part, we can design a prefix code satisfying both (3.3.1) and the Kraft inequality, which immediately implies the existence of the desired code by Theorem 3.25.

- Choose the codeword length for sourceword x^n as

$$\ell(\boldsymbol{c}_{x^n}) = \lfloor -\log_D P_{X^n}(x^n) \rfloor + 1. \tag{3.3.2}$$

  Then

$$D^{-\ell(\boldsymbol{c}_{x^n})} \le P_{X^n}(x^n).$$

- Summing both sides over all source symbols, we obtain

$$\sum_{x^n \in \mathcal{X}^n} D^{-\ell(\boldsymbol{c}_{x^n})} \le 1,$$

  which is exactly the Kraft inequality.

Source Coding Theorem for Variable-Length Codes

- On the other hand, (3.3.2) implies

$$\ell(\boldsymbol{c}_{x^n}) \le -\log_D P_{X^n}(x^n) + 1,$$

which in turn implies

$$\begin{aligned}\sum_{x^n\in\mathcal{X}^n} P_{X^n}(x^n)\ell(\boldsymbol{c}_{x^n}) &\le \sum_{x^n\in\mathcal{X}^n}\big[-P_{X^n}(x^n)\log_D P_{X^n}(x^n)\big] + \sum_{x^n\in\mathcal{X}^n} P_{X^n}(x^n)\\ &= H_D(X^n)+1 = nH_D(X)+1,\end{aligned}$$

where the last equality holds since the source is memoryless. $\square$

Prefix Codes for Block Sourcewords

E.g. A memoryless source with source alphabet

$$\{A, B, C\}$$

and probability distribution

$$P_X(A) = 0.8, \quad P_X(B) = P_X(C) = 0.1$$

has entropy being equal to

$$-0.8 \cdot \log_2 0.8 - 0.1 \cdot \log_2 0.1 - 0.1 \cdot \log_2 0.1 = 0.92 \text{ bits.}$$

- One of the best binary first-order or single-letter encoding (with $n = 1$) prefix codes for this source is given by

  $$\boldsymbol{c}(A) = 0, \boldsymbol{c}(B) = 10 \text{ and } \boldsymbol{c}(C) = 11,$$

  where $\boldsymbol{c}(\cdot)$ is the encoding function.
- Then the resultant average code rate for this code is

  $$0.8 \times 1 + 0.2 \times 2 = 1.2 \text{ bits } \geq \ 0.92 \text{ bits.}$$

The optimal variable-length code for a specific source X usually has average codeword length *strictly larger* than the source entropy.

Prefix Codes for Block Sourcewords

- Now if we consider a second-order (with $n = 2$) prefix code by encoding two consecutive source symbols at a time, the new source alphabet becomes

  $$\{AA, AB, AC, BA, BB, BC, CA, CB, CC\},$$

  and the resultant probability distribution is calculated by

  $$(\forall\ x_1, x_2 \in \{A, B, C\}) \quad P_{X^2}(x_1, x_2) = P_X(x_1)P_X(x_2)$$

  as the source is memoryless. Then one of the best binary prefix codes for the source is given by

  $$\begin{aligned}
  \boldsymbol{c}(AA) &= 0\\
  \boldsymbol{c}(AB) &= 100\\
  \boldsymbol{c}(AC) &= 101\\
  \boldsymbol{c}(BA) &= 110\\
  \boldsymbol{c}(BB) &= 111100\\
  \boldsymbol{c}(BC) &= 111101\\
  \boldsymbol{c}(CA) &= 1110\\
  \boldsymbol{c}(CB) &= 111110\\
  \boldsymbol{c}(CC) &= 111111.
  \end{aligned}$$

Prefix Codes for Block Sourcewords

- The average code rate of this code now becomes
$$\frac{0.64(1\times 1)+0.08(3\times 3+4\times 1)+0.01(6\times 4)}{2}=0.96 \text{ bits},$$
which is closer to the source entropy of 0.92 bits.
- As n increases, the average code rate will be brought closer to the source entropy.

Prefix Codes for Block Sourcewords

Theorem 3.28 (Lossless variable-length source coding theorem) Fix integer $D > 1$ and consider a discrete memoryless source $\{X_n\}_{n=1}^{\infty}$ with distribution P_X and entropy $H_D(X)$ (measured in D-ary units). Then the following hold.

- *Forward part (achievability):* For any $\varepsilon > 0$, there exists a D-ary n-th order prefix (hence uniquely decodable) code
$$f : \mathcal{X}^n \to \{0, 1, \cdots, D-1\}^*$$
for the source with an average code rate $\overline{R}_n$ satisfying
$$\overline{R}_n \le H_D(X) + \varepsilon$$
for n sufficiently large.

- *Converse part:* Every uniquely decodable code
$$f : \mathcal{X}^n \to \{0, 1, \cdots, D-1\}^*$$
for the source has an average code rate $\overline{R}_n \ge H_D(X)$.

Proof: The forward part follows directly from Theorem 3.27 by choosing n large enough such that $1/n < \varepsilon$, and the converse part is already given by Theorem 3.22 (cf. Slide I: 3-43). □

Final Note on Prefix Codes

Observation 3.29 Theorem 3.28 actually also holds for the class of *stationary sources* by replacing the source entropy $H_D(X)$ with the source entropy rate

$$H_D(\mathcal{X}) := \lim_{n\to\infty} \frac{1}{n} H_D(X^n),$$

measured in D-ary units. The proof is very similar to the proofs of Theorems 3.22 and 3.27 with slight modifications (such as using the fact that $\frac{1}{n}H_D(X^n)$ is nonincreasing with n for stationary sources).

Observation 3.30 (Rényi's entropy and lossless data compression)

- Implicit in Theorem 3.28, the use of average codeword length as a performance criterion is the assumption that the cost of compression varies *linearly* with codeword length.
- This is not always the case as in some applications, where the processing cost of decoding may be elevated and buffer overflows caused by long codewords can cause problems, an *exponential* cost/penalty function for codeword lengths can be more appropriate than a linear cost function [Campbell'65, Jelinek'68, Blumer & McEliece '88].

Final Note on Prefix Codes

- Given a D-ary n-th order VLC $\mathcal{C}$

$$f\colon \mathcal{X}^n \to \{0, 1, \ldots, D-1\}^*$$

for a discrete source $\{X_i\}_{n=1}^{\infty}$ with alphabet $\mathcal{X}$ and distribution P_{X^n}, Campbell considers the following exponential cost function, called the *average codeword length of order t*:

$$L_n(t) := \frac{1}{t} \log_D \left(\sum_{x^n \in \mathcal{X}^n} P_{X^n}(x^n) D^{t \cdot \ell(\boldsymbol{c}_{x^n})} \right),$$

where t is a chosen positive constant, $\boldsymbol{c}_{x^n} = f(x^n)$ is the codeword associated with sourceword x^n and $\ell(\cdot)$ is the length of $\boldsymbol{c}_{x^n}$.

- In the limiting case when $t \to 0$,

$$L_n(t) \to \sum_{x \in \mathcal{X}} P_X(x) \ell(\boldsymbol{c}_x) = \bar{\ell}$$

and we recover the average codeword length, as desired.

- Also, when $t \to \infty$, $L_n(t) \to \max_{x \in \mathcal{X}} \ell(\boldsymbol{c}_x)$, which is the maximum codeword length for all codewords in $\mathcal{C}$.

Final Note on Prefix Codes

Theorem 3.31 (Lossless source coding theorem under exponential costs) Consider a DMS $\{X_n\}$ with Rényi entropy in D-ary units and of order α given by

$$H_\alpha(X) = \frac{1}{1-\alpha} \log_D \left(\sum_{x \in \mathcal{X}} P_X^\alpha(x) \right).$$

Fixing $t > 0$ and setting $\alpha = \frac{1}{1+t}$, the following hold.

- For any $\varepsilon > 0$, there exists a D-ary n-th order uniquely decodable code $f\colon \mathcal{X}^n \to \{0, 1, \dots, D-1\}^*$ for the source with an average code rate of order t satisfying
$$\frac{1}{n} L_n(t) \le H_\alpha(X) + \varepsilon$$
for n sufficiently large.
- Conversely, it is not possible to find a uniquely decodable code whose average code rate of order t is less than $H_\alpha(X)$.

3.3.3 Examples of Binary Prefix Codes

(A) Huffman Codes: Optimal Variable-Length Codes

Lemma 3.32 Let $\mathcal{C}$ be an optimal binary prefix code with codeword lengths ℓ_i, $i = 1, \cdots, M$, for a source with alphabet $\mathcal{X} = \{a_1, \ldots, a_M\}$ and symbol probabilities $p_1, \ldots, p_M$. We assume, without loss of generality, that

$$p_1 \geq p_2 \geq p_3 \geq \cdots \geq p_M,$$

and that any group of source symbols with identical probability is listed in order of increasing codeword length (i.e., if $p_i = p_{i+1} = \cdots = p_{i+s}$, then $\ell_i \leq \ell_{i+1} \leq \cdots \leq \ell_{i+s}$). Then the following properties hold.

1. Higher probability source symbols have shorter codewords, i.e., $p_i > p_j$ implies $\ell_i \leq \ell_j$, for $i, j = 1, \cdots, M$.

2. The two least probable source symbols have codewords of equal length: $\ell_{M-1} = \ell_M$.

3. Among the codewords of length ℓ_M, two of the codewords are identical except in the last digit.

3.3.3 Examples of Binary Prefix Codes

Proof:

1) If $p_i > p_j$ and $\ell_i > \ell_j$, then it is possible to construct a better code $\mathcal{C}'$ by interchanging ("swapping") codewords i and j of $\mathcal{C}$, since

$$\begin{aligned} \overline{\ell}(\mathcal{C}') - \overline{\ell}(\mathcal{C}) &= p_i\ell_j + p_j\ell_i - (p_i\ell_i + p_j\ell_j) \\ &= (p_i - p_j)(\ell_j - \ell_i) \\ &< 0. \end{aligned}$$

Hence code $\mathcal{C}'$ is better than code $\mathcal{C}$, contradicting the fact that $\mathcal{C}$ is optimal.

2) We first know that $\ell_{M-1} \le \ell_M$, since:

- If $p_{M-1} > p_M$, then $\ell_{M-1} \le \ell_M$ by result 1) above.
- If $p_{M-1} = p_M$, then $\ell_{M-1} \le \ell_M$ by our assumption about the ordering of codewords for source symbols with identical probability.

Now, if $\ell_{M-1} < \ell_M$, we may delete the last digit of codeword M, and the deletion cannot result in another codeword since $\mathcal{C}$ is a prefix code. Thus the deletion forms a new prefix code with a better average codeword length than $\mathcal{C}$, contradicting the fact that $\mathcal{C}$ is optimal. Hence, we must have that $\ell_{M-1} = \ell_M$.

3.3.3 Examples of Binary Prefix Codes

3) Among the codewords of length ℓ_M, if no two codewords agree in all digits except the last, then we may delete the last digit in all such codewords to obtain a better codeword. $\square$

3.3.3 Examples of Binary Prefix Codes

Lemma 3.33 (Huffman) Consider a source with alphabet $\mathcal{X} = \{a_1, \ldots, a_M\}$ and symbol probabilities $p_1, \ldots, p_M$ such that

$$p_1 \geq p_2 \geq \cdots \geq p_M.$$

Consider the *reduced source* alphabet $\mathcal{Y}$ obtained from $\mathcal{X}$ by combining the two least likely source symbols a_{M-1} and a_M into an equivalent symbol $a_{M-1,M}$ with probability $p_{M-1} + p_M$. Suppose that $\mathcal{C}'$, given by $f' : \mathcal{Y} \to \{0,1\}^*$, is an optimal code for the reduced source $\mathcal{Y}$. We now construct a code $\mathcal{C}$, $f : \mathcal{X} \to \{0,1\}^*$, for the original source $\mathcal{X}$ as follows:

- The codewords for symbols $a_1, a_2, \cdots, a_{M-2}$ are exactly the same as the corresponding codewords in $\mathcal{C}'$:

$$f(a_1) = f'(a_1), f(a_2) = f'(a_2), \cdots, f(a_{M-2}) = f'(a_{M-2}).$$

- The codewords associated with symbols a_{M-1} and a_M are formed by appending a "0" and a "1", respectively, to the codeword $f'(a_{M-1,M})$ associated with the letter $a_{M-1,M}$ in $\mathcal{C}'$:

$$f(a_{M-1}) = [f'(a_{M-1,M})0] \quad \text{and} \quad f(a_M) = [f'(a_{M-1,M})1].$$

Then code $\mathcal{C}$ is optimal for the original source $\mathcal{X}$.

3.3.3 Examples of Binary Prefix Codes

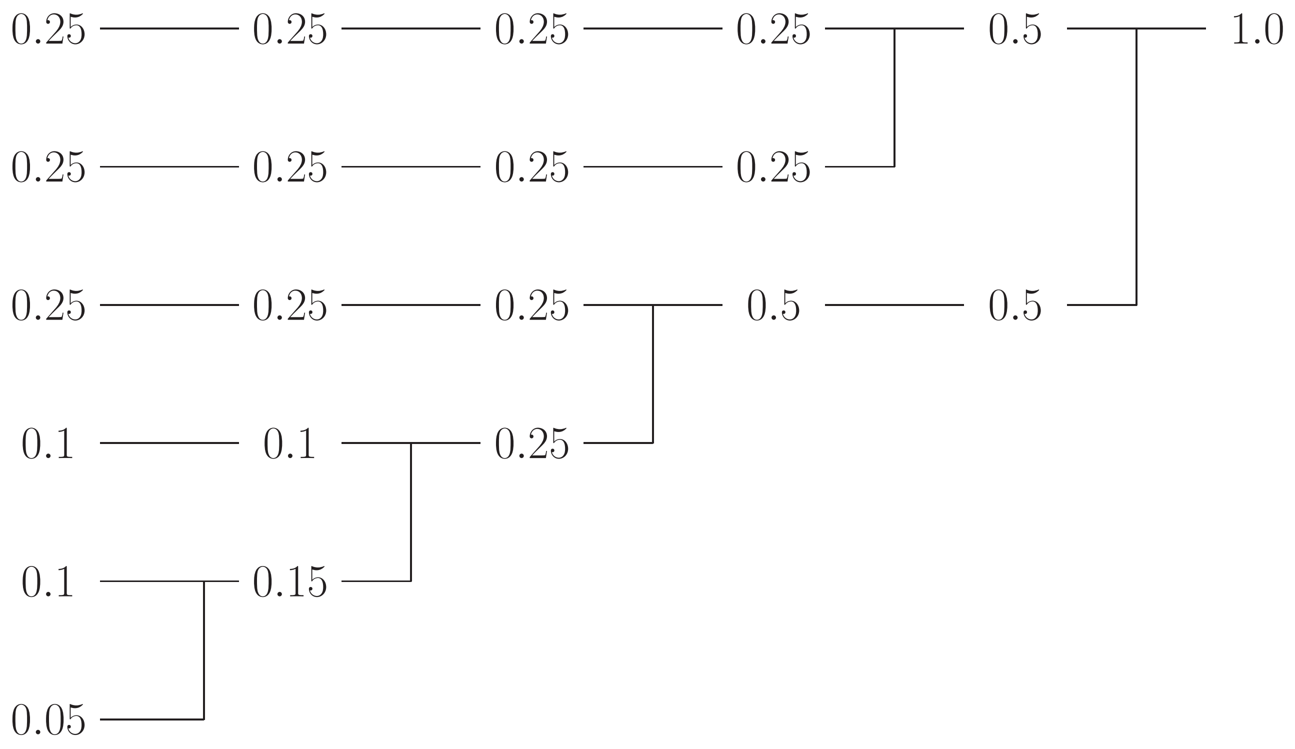
Huffman encoding algorithm:

1. Repeatedly apply the above lemma until one is left with a reduced source with two symbols. An optimal binary prefix code for this source consists of the codewords 0 and 1.
2. Then proceed backwards, constructing (as outlined in the above lemma) optimal codes for each reduced source until one arrives at the original source.

3.3.3 Examples of Binary Prefix Codes

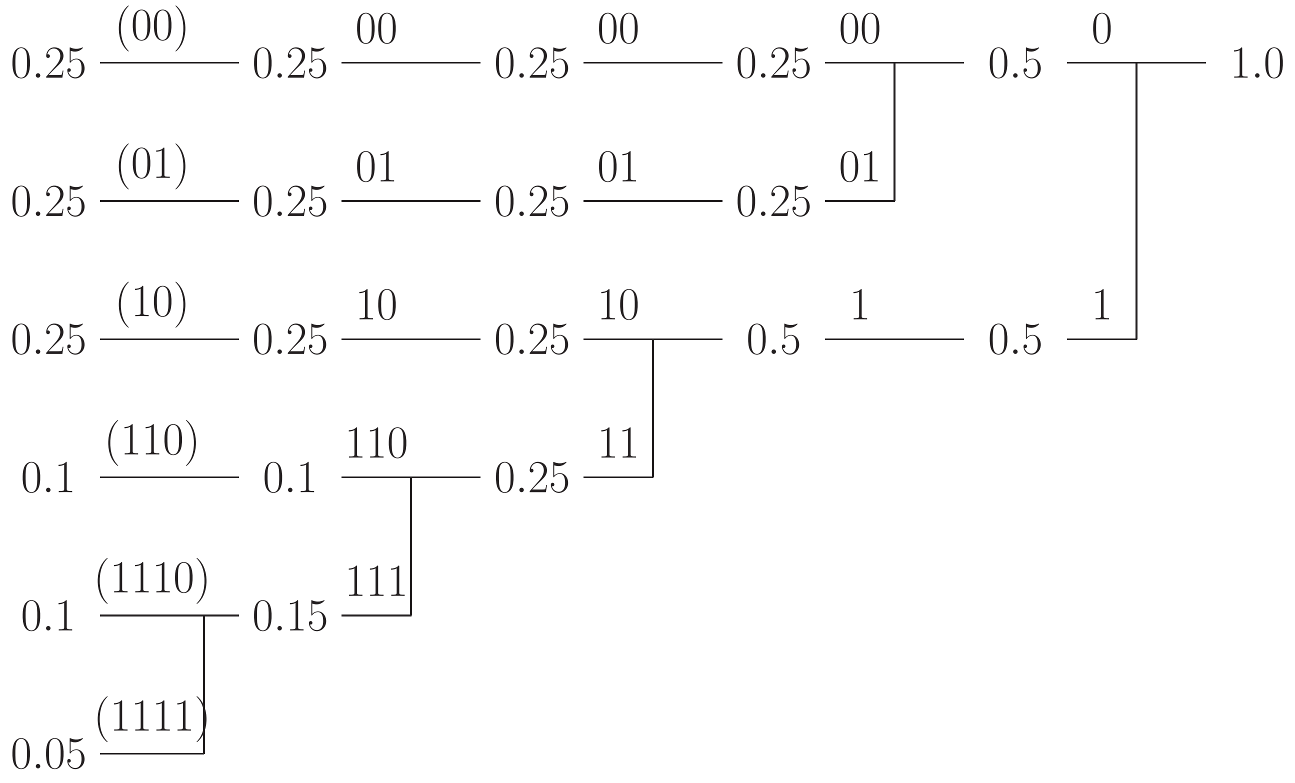
Example 3.34 Consider a source with alphabet $\{1, 2, 3, 4, 5, 6\}$ and symbol probabilities $0.25, 0.25, 0.25, 0.1, 0.1$ and 0.05, respectively.

- Step 1:



3.3.3 Examples of Binary Prefix Codes

- Step 2:



By following the Huffman encoding procedure as shown in above figures, we obtain the Huffman code as

$$00, 01, 10, 110, 1110, 1111.$$

3.3.3 Examples of Binary Prefix Codes

Observation 3.35

- Huffman codes are not unique for a given source distribution;
  - e.g., by inverting all the code bits of a Huffman code, one gets another Huffman code,
  - or by resolving ties in different ways in the Huffman algorithm, one also obtains different Huffman codes.
- One can obtain optimal codes that are not Huffman codes;
  - e.g., by interchanging two codewords of the same length of a Huffman code, one may get another non-Huffman (but optimal) code.
  - Furthermore, one can construct an optimal *suffix(-free)* code (i.e., a code in which no codeword can be a suffix of another codeword) from a Huffman code by reversing the Huffman codewords.
  - Binary Huffman codes always satisfy the Kraft inequality with equality (their code tree is "saturated").

3.3.3 Examples of Binary Prefix Codes

- Any n-th order binary Huffman code $f : \mathcal{X}^n \to \{0,1\}^*$ for a stationary source $\{X_n\}_{n=1}^{\infty}$ with finite alphabet $\mathcal{X}$ satisfies:
$$H(\mathcal{X}) \le \frac{1}{n} H(X^n) \le \overline{R}_n < \frac{1}{n} H(X^n) + \frac{1}{n}.$$
Thus, as n increases to infinity, $\overline{R}_n \to H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X^n)$ but the complexity as well as encoding-decoding delay grows exponentially with n.

- Finally, note that *non-binary* (i.e., for $D > 2$) Huffman codes can also be constructed in a mostly similar way as for the case of binary Huffman codes by designing a D-ary tree and iteratively applying Lemma 3.33, where now the D least likely source symbols are combined at each stage.
  - The only difference from the case of binary Huffman codes is that we have to ensure that we are ultimately left with D symbols at the last stage of the algorithm to guarantee the code's optimality.
  - This is remedied by expanding the original source alphabet $\mathcal{X}$ by adding "dummy" symbols (each with zero probability) so that the alphabet size of the expanded source $|\mathcal{X}'|$ is the smallest positive integer greater than or equal to $|\mathcal{X}|$ with
$$|\mathcal{X}'| = 1 \quad (\text{modulo } D - 1) \quad (\text{For } D > 2).$$

3.3.3 Examples of Binary Prefix Codes

– In fact, we wish

$$|\mathcal{X}'| = D + k(D-1) \quad \text{for some integer } k,$$

which trivially holds for $D = 2$.

– For example, if $|\mathcal{X}| = 6$ and $D = 3$ (ternary codes), we obtain that

$$|\mathcal{X}'| = 7,$$

meaning that we need to enlarge the original source $\mathcal{X}$ by adding one dummy (zero-probability) source symbol.

3.3.3 Examples of Binary Prefix Codes

Generalized Huffman codes under exponential costs:

- When the lossless compression problem allows for exponential costs, a straightforward generalization of Huffman's algorithm, which minimizes the average code rate of order t, $\frac{1}{n}L_n(t)$, can be obtained.
- More specifically, while in Huffman's algorithm, each new node (for a combined or equivalent symbol) is assigned weight $p_i + p_j$, where p_i and p_j are the lowest weights (probabilities) among the available nodes, in the generalized algorithm, each new node is assigned weight $2^t(p_i + p_j)$.
- With this simple modification, one can directly construct such generalized Huffman codes.

Historical Note

David A. Huffman had finished his B.S. and M.S. in electrical engineering and also served in the U.S. Navy before he became a Ph.D. student at the Massachusetts Institute of Technology (MIT). There, in 1951, he attended an information theory class taught by Prof. Robert M. Fano who was working at that time together with Claude E. Shannon on finding the most efficient code, but could not solve the problem. So Fano assigned the question to his students in the information theory class as a term paper. Huffman tried a long time and was about to give up when he had the sudden inspiration to start building the tree backwards from leaves to root instead from root to leaves. Once he had understood this, he was quickly able to prove that his code was the most efficient one. Naturally, Huffman's term paper was later on published.

Huffman became a faculty member of MIT in 1953, and later in 1967 he changed to the University of California, Santa Cruz, where he stayed until his retirement in 1994. He won many awards for his accomplishments like, e.g., in 1999 the Richard Hamming Medal from the Institute of Electrical and Electronics Engineers (IEEE). Huffman died in 1999.

(B) Shannon-Fano-Elias Codes

Assume $\mathcal{X} = \{1, \ldots, M\}$ and $P_X(x) > 0$ for all $x \in \mathcal{X}$.

Note that it is not guaranteed that

$$P_X(1) \geq P_X(2) \geq \cdots \geq P_X(M).$$

Define

$$F(x) := \sum_{a \leq x} P_X(a),$$

and

$$\bar{F}(x) := \sum_{a < x} P_X(a) + \frac{1}{2} P_X(x).$$

Encoder: For any $x \in \mathcal{X}$, express $\bar{F}(x)$ in decimal binary form, say

$$\bar{F}(x) = .c_1 c_2 \ldots c_k \ldots,$$

and take the first k (fractional) bits as the codeword of source symbol x, i.e.,

$$(c_1, c_2, \ldots, c_k),$$

where

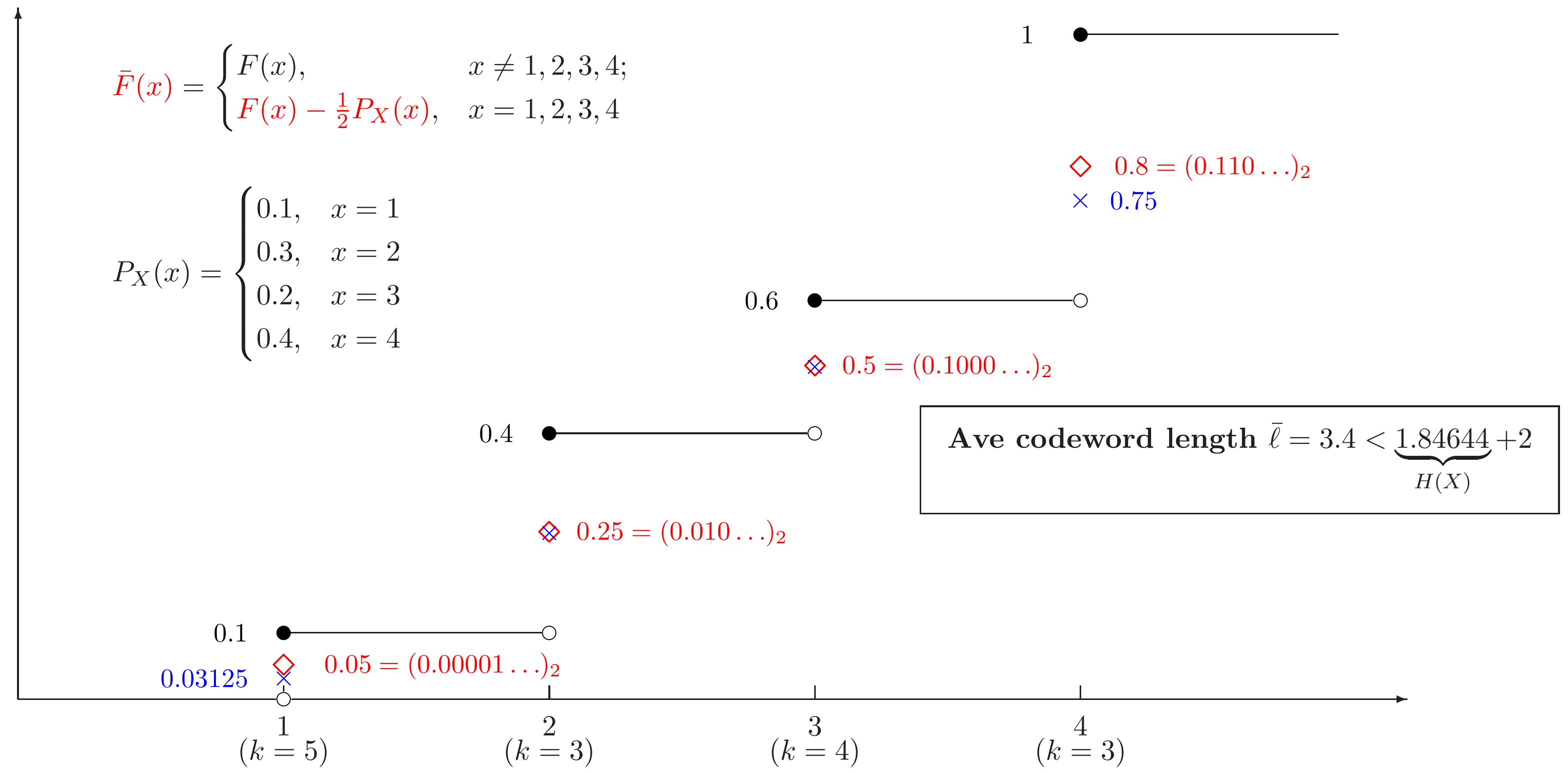
$$k := \lceil \log_2(1/P_X(x)) \rceil + 1.$$

(B) Shannon-Fano-Elias Codes

Decoder: Given codeword $(c_1, \ldots, c_k)$, compute the cumulative sum of $F(\cdot)$ starting from the smallest element in $\{1, 2, \ldots, M\}$ until the first x satisfying

$$F(x) \geq .c_1 \ldots c_k.$$

Then x should be the original source symbol.


$\bar{F}(x) = \begin{cases} F(x), & x \neq 1,2,3,4; \\ F(x) - \frac{1}{2}P_X(x), & x = 1,2,3,4 \end{cases}$
$P_X(x) = \begin{cases} 0.1, & x = 1 \\ 0.3, & x = 2 \\ 0.2, & x = 3 \\ 0.4, & x = 4 \end{cases}$
1
$0.8 = (0.110\ldots)_2$
0.75
0.6
$0.5 = (0.1000\ldots)_2$
0.4
Ave codeword length $\bar{\ell} = 3.4 < \underbrace{1.84644}_{H(X)} + 2$
$0.25 = (0.010\ldots)_2$
0.1
$0.05 = (0.00001\ldots)_2$
0.03125
1 $(k = 5)$
2 $(k = 3)$
3 $(k = 4)$
4 $(k = 3)$

(B) Shannon-Fano-Elias Codes

Proof of decodability: For any number $a \in [0,1]$, let $[a]_k$ denote the operation that chops the binary representation of a after k bits (i.e., removing the $(k+1)^{\text{th}}$ bit, the $(k+2)^{\text{th}}$ bit, etc). Then

$$\bar{F}(x) - \left[\bar{F}(x)\right]_k < \frac{1}{2^k}.$$

Since $k = \lceil \log_2(1/P_X(x)) \rceil + 1 \geq \log_2(1/P_X(x)) + 1$, we have for $x \in \{1, 2, \ldots, M\}$,

$$\begin{aligned} \frac{1}{2^k} &\leq \frac{1}{2} P_X(x) \\ &= \left[\sum_{a<x} P_X(a) + \frac{P_X(x)}{2}\right] - \sum_{a \leq x-1} P_X(a) \\ &= \bar{F}(x) - F(x-1). \end{aligned}$$

Hence,

$$F(x-1) = \left[F(x-1) + \frac{1}{2^k}\right] - \frac{1}{2^k} \leq \bar{F}(x) - \frac{1}{2^k} < \left[\bar{F}(x)\right]_k.$$

In addition,

$$F(x) > \bar{F}(x) \geq \left[\bar{F}(x)\right]_k.$$

Consequently, x is the first element satisfying

$$F(x) \geq \left[\bar{F}(x)\right]_k = .c_1 c_2 \ldots c_k.$$

(B) Shannon-Fano-Elias Codes

Average codeword length:

$$\begin{aligned}\bar{\ell} &= \sum_{x\in\mathcal{X}} P_X(x)\left\lceil \log_2 \frac{1}{P_X(x)}\right\rceil + 1\\ &< \sum_{x\in\mathcal{X}} P_X(x)\log_2 \frac{1}{P_X(x)} + 2\\ &= (H(X)+2)\ \text{bits}.\end{aligned}$$

Observation 3.36 The Shannon-Fano-Elias code is a prefix code.

(B) Shannon-Fano-Elias Codes

Historical Note about Fano code

- *Fano code*: The *Fano code* is generated according to the following algorithm:
  1. Arrange the symbols in order of nonincreasing probability.
  2. Divide the list of ordered symbols into two parts, with the total probability of the left part being as close to the total probability of the right part as possible.
  3. Assign the binary digit 0 to the left part of the list, and the digit 1 to the right part.
  4. Recursively apply Step 2 and Step 3 to each of the two parts, subdividing into further parts and adding bits to the codewords until each symbol is the single member of a part.

  Note that effectively this algorithm constructs a tree. Hence, the Fano code is a prefix code.

(B) Shannon-Fano-Elias Codes

Exemplified construction of Fano code for alphabet of size 5 and probability distributions 0.35, 0.25, 0.15, 0.15 and 0.1, respectively.

p_1	p_2	p_3	p_4	p_5
0.35	0.25	0.15	0.15	0.1
0.6		0.4		
0		**1**		
0.35	0.25	0.15	0.15	0.1
		0.15	0.25	
0	**1**	**0**	**1**	
			0.15	0.1
			0	**1**
00	**01**	**10**	**110**	**111**

Remarks

– In the literature, the Fano code is usually known as the *Shannon-Fano code*, even though it is an invention of Professor Robert Fano from MIT and not of Shannon. (Another example for such mis-naming is Stein's Lemma, which is not the invention of Prof. Charles M. Stein.)

(B) Shannon-Fano-Elias Codes

– To make things even worse, there exists another code that is also known as *Shannon-Fano code*, but actually should be called *Shannon code* because it was proposed by Shannon.

  * Under the premise that the symbols must be in order of nonincreasing probability, this *Shannon code* is exactly the Shannon-Fano-Elias code just introduced with
$$k{:=}\lceil \log_2(1/P_X(x))\rceil.$$
Note that without symbols being arranged in order of nonincreasing probability, a larger k should be used:
$$k{:=}\lceil \log_2(1/P_X(x))\rceil + 1.$$

– The Kraft Inequality is still satisfied for one less k:
$$\sum_{x\in\mathcal{X}} 2^{-\left\lceil \log_2 \frac{1}{P_X(x)}\right\rceil} \le \sum_{x\in\mathcal{X}} 2^{-\log_2 \frac{1}{P_X(x)}} = \sum_{i=1}^{M} P_X(x) = 1.$$
So the existence of such prefix code (with k being one less) is priori known. Shannon code substantiates this prognosis.

(B) Shannon-Fano-Elias Codes

- Shannon code performs similarly to the Fano code, but Fano code is in general slightly better.
- The idea of Shannon-Fano-Elias code has been additionally credited to the late Professor Peter Elias from MIT (hence the name *Shannon-Fano-Elias coding*), but actually Elias denied this. The concept has probably come from Shannon himself during a talk that he gave at MIT.

3.3.4 Universal Lossless Variable-Length Codes

- The Huffman codes and Shannon-Fano-Elias codes can be constructed when the source statistics is known.
- If the source statistics is unknown, is it possible to find a code whose average codeword length is arbitrarily close to entropy? Yes, if "asymptotic achievability" is allowed.

(A) Adaptive Huffman Codes (Gallager 1978)

- Let the source alphabet be $\mathcal{X}:=\{a_1,\ldots,a_M\}$.
- Define
$$N(a_i|x^n):=\text{number of } a_i \text{ appearances in } x_1,x_2,\ldots,x_n.$$
- Then the (current) relative frequency of a_i is
$$\frac{N(a_i|x^n)}{n}.$$
- Let $\boldsymbol{c}_n(a_i)$ denote the Huffman codeword of source symbol a_i with respect to distribution
$$\left[\frac{N(a_1|x^n)}{n},\frac{N(a_2|x^n)}{n},\cdots,\frac{N(a_J|x^n)}{n}\right].$$

(A) Adaptive Huffman Codes (Gallager 1978)

Now suppose that $x_{n+1} = a_j$.

1. The codeword $\boldsymbol{c}_n(a_j)$ is outputted.
2. Update the relative frequency for each source outcome according to:

$$\frac{N(a_j|x^{n+1})}{n+1} = \frac{n \times [N(a_j|x^n)/n] + 1}{n+1}$$

and

$$\frac{N(a_i|x^{n+1})}{n+1} = \frac{n \times [N(a_i|x^n)/n]}{n+1} \quad \text{for } i \neq j.$$

(A) Adaptive Huffman Codes (Gallager 1978)

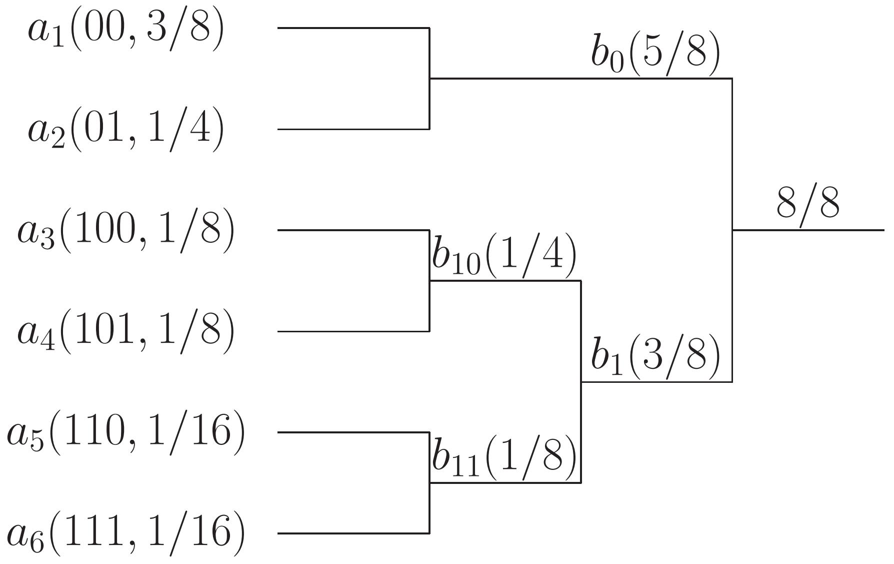
Definition 3.37 (Sibling property) A prefix code is said to have the *sibling property* if its codetree satisfies:

1. every node in the code-tree (except for the root node) has a sibling (i.e., the code tree is saturated), and
2. the node can be listed in nondecreasing order of probabilities with each node being adjacent to its sibling.

Observation 3.38 A prefix code is a Huffman code if, and only if, it satisfies the sibling property.

(A) Adaptive Huffman Codes (Gallager 1978)

E.g.



$$\underbrace{b_0\left(\frac{5}{8}\right) \geq b_1\left(\frac{3}{8}\right)}_{\text{sibling pair}} \geq \underbrace{a_1\left(\frac{3}{8}\right) \geq a_2\left(\frac{1}{4}\right)}_{\text{sibling pair}}$$

$$\geq \underbrace{b_{10}\left(\frac{1}{4}\right) \geq b_{11}\left(\frac{1}{8}\right)}_{\text{sibling pair}} \geq \underbrace{a_3\left(\frac{1}{8}\right) \geq a_4\left(\frac{1}{8}\right)}_{\text{sibling pair}} \geq \underbrace{a_5\left(\frac{1}{16}\right) \geq a_6\left(\frac{1}{16}\right)}_{\text{sibling pair}}$$

(A) Adaptive Huffman Codes (Gallager 1978)

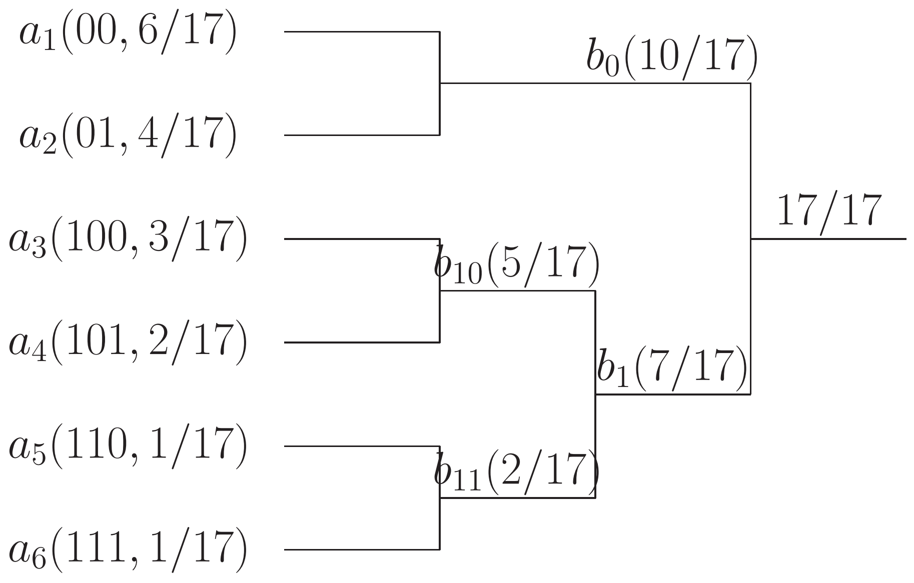
E.g. (cont.)

- If the next observation ($n = 17$) is a_3, then its codeword 100 is outputted.
- The estimated distribution is updated as

$$P_{\hat{X}}^{(17)}(a_1) = \frac{16 \times (3/8)}{17} = \frac{6}{17}, \quad P_{\hat{X}}^{(17)}(a_2) = \frac{16 \times (1/4)}{17} = \frac{4}{17}$$
$$P_{\hat{X}}^{(17)}(a_3) = \frac{16 \times (1/8)+1}{17} = \frac{3}{17}, \quad P_{\hat{X}}^{(17)}(a_4) = \frac{16 \times (1/8)}{17} = \frac{2}{17}$$
$$P_{\hat{X}}^{(17)}(a_5) = \frac{16 \times [1/(16)]}{17} = \frac{1}{17}, \quad P_{\hat{X}}^{(17)}(a_6) = \frac{16 \times [1/(16)]}{17} = \frac{1}{17}.$$

The sibling property is no longer satisfied; hence, the Huffman codetree needs to be updated.

(A) Adaptive Huffman Codes (Gallager 1978)



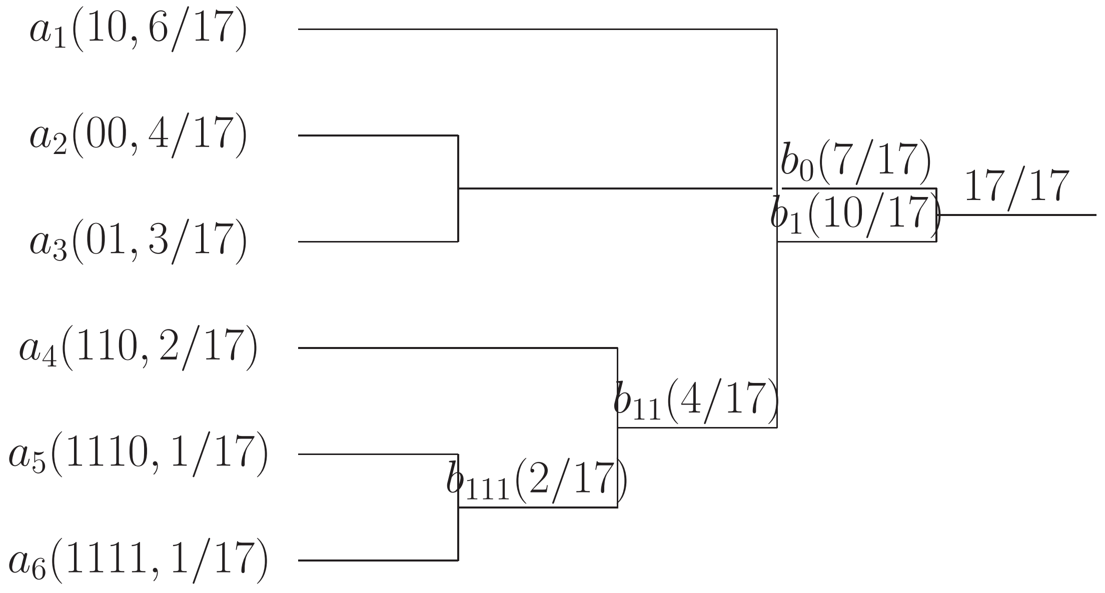
$$\underbrace{b_0\left(\frac{10}{17}\right) \geq b_1\left(\frac{7}{17}\right)}_{\text{sibling pair}} \geq \underline{a_1\left(\frac{6}{17}\right)} \geq b_{10}\left(\frac{5}{17}\right)$$

$$\geq \underline{a_2\left(\frac{4}{17}\right)} \geq \underbrace{a_3\left(\frac{3}{17}\right) \geq a_4\left(\frac{2}{17}\right)}_{\text{sibling pair}} \geq b_{11}\left(\frac{2}{17}\right) \geq \underbrace{a_5\left(\frac{1}{17}\right) \geq a_6\left(\frac{1}{17}\right)}_{\text{sibling pair}}$$

a_1 is not adjacent to its sibling a_2.

(A) Adaptive Huffman Codes (Gallager 1978)

E.g. (cont.) The updated Huffman codetree.



$$\underbrace{b_1\left(\frac{10}{17}\right) \geq b_0\left(\frac{7}{17}\right)}_{\text{sibling pair}} \geq \underbrace{a_1\left(\frac{6}{17}\right) \geq b_{11}\left(\frac{4}{17}\right)}_{\text{sibling pair}}$$

$$\geq \underbrace{a_2\left(\frac{4}{17}\right) \geq a_3\left(\frac{3}{17}\right)}_{\text{sibling pair}} \geq \underbrace{a_4\left(\frac{2}{17}\right) \geq b_{111}\left(\frac{2}{17}\right)}_{\text{sibling pair}} \geq \underbrace{a_5\left(\frac{1}{17}\right) \geq a_6\left(\frac{1}{17}\right)}_{\text{sibling pair}}$$

(B) Lempel-Ziv Codes (1977-1978)

Encoder:

1. Parse the input sequence into strings that have never appeared before.
2. Let L be the number of distinct strings of the parsed source. Then we need $\log_2 L$ bits to index these strings (starting from one). The codeword of each string is the index of its prefix concatenated with the last bit in its source string.

E.g.

- The input sequence is 1011010100010;
- Step 1:
  - The algorithm first grabs the first letter 1 and finds that it has never appeared before. So 1 is the *first string*.
  - Then, the algorithm scoops the second letter 0 and also determines that it has not appeared before, and hence, put it to be the *next string*.
  - The algorithm moves on to the next letter 1 and finds that this string has appeared. Hence, it hits another letter 1 and yields a new string 11, and so on.
  - Under this procedure, the source sequence is parsed into the strings

$$1, 0, 11, 01, 010, 00, 10.$$

(B) Lempel-Ziv Codes (1977-1978)

- Step 2:
  - We need $\lfloor \log_2(L) \rfloor + 1$ bits to index these strings (starting from one). In the above example, $\lfloor \log_2(L) \rfloor + 1 = 3$. So the indices will be:

$$\begin{array}{rccccccc} \textit{parsed source}: & 1 & 0 & 11 & 01 & 010 & 00 & 10 \\ \textit{index}: & 001 & 010 & 011 & 100 & 101 & 110 & 111 \end{array}$$

  - The codeword of each string is then the index of its prefix concatenated with the last bit in its source string. For example, the codeword of source string 010 will be the index of 01, i.e., 100, concatenated with the last bit of the source string, i.e., 0.

- The resultant codeword string is:

$$(000,1)(000,0)(001,1)(010,1)(100,0)(010,0)(001,0)$$

or equivalently,

$$0001000000110101100001000010.$$

Theorem 3.39 The Lempel-Ziv algorithm asymptotically achieves the entropy rate of any stationary ergodic source (with unknown statistics).

Notes on Lempel-Ziv Codes

- It is also named as the Lempel-Ziv-Welch compression algorithm after Welch developed an efficient version of the original Lempel-Ziv technique in 1984.
- Note that the conventional Lempel-Ziv encoder requires two passes: the first pass to decide L, and the second pass to generate the codewords.
- The algorithm, however, can be modified so that it requires only one pass over the entire source string.
- Also, note that the above algorithm uses an equal number of bits to all the location indices, which can also be relaxed by proper modification.

Key Notes

- Average "per-source-symbol" codeword length versus "per-source-symbol" entropy
- Categories of codes
  - Fixed-length codes (and their relation with segmentation or blocking)
    - Block codes
    - Fixed-length tree codes
  - Variable-length codes
    - Non-singular codes
    - Uniquely decodable codes
    - Prefix codes
- AEP theorem
- Weakly δ-typical set and Shannon-McMillan-Breiman theorem
- Shannon's source coding theorem and its converse theorem for DMS

Key Notes

- Entropy rate and the proof of its existence for stationary sources
- Shannon's source coding theorem and its converse theorem for stationary-ergodic sources
- Redundancy of sources
- Kraft inequality and its relation to uniquely decodable codes, as well as prefix codes
- Source coding theorem for variable-length codes
- Huffman codes and adaptive Huffman codes
- Shannon-Fano-Elias codes
- Lempel-Ziv codes