COMP2610/6261 - Information Theory

Lecture 15: Arithmetic Coding

Mark Reid and Aditya Menon

Research School of Computer Science
The Australian National University

Australian
National
University

September 23rd, 2014

Huffman Coding: Advantages and Disadvantages

Advantages:

- Huffman Codes are provably optimal
- Algorithm is simple and efficient

Huffman Coding: Advantages and Disadvantages

Advantages:

- Huffman Codes are provably optimal
- Algorithm is simple and efficient

Disadvantages:

- Assumes a fixed distribution of symbols
- The extra bit in the SCT
  - If $H(X)$ is large – not a problem
  - If $H(X)$ is small (e.g., ~ 1 bit for English) codes are $2\times$ optimal

Huffman codes are the best possible symbol code
but symbol coding is not always the best type of code

A Guessing Game

Let's play!

An Idealised Guessing Game

Encoding: Given message $\mathbf{x}$ and guesser G:
For i from 1 to $|\mathbf{x}|$:

1. Set count $n_i = 1$
2. While G guesses x_i incorrectly:
    1. $n_i \leftarrow n_i + 1$
3. Output n_i

An Idealised Guessing Game

Encoding: Given message $\mathbf{x}$ and guesser G:
For i from 1 to $|\mathbf{x}|$:

1. Set count $n_i = 1$
2. While G guesses x_i incorrectly:
    1. $n_i \leftarrow n_i + 1$
3. Output n_i

Decoding: Given counts $\mathbf{n}$ and guesser G:
For i from 1 to $|\mathbf{n}|$ and guesser G:

1. While G has made fewer than n_i guesses:
    1. $x_i \leftarrow$ next guess from G
2. Output x_i

An Idealised Guesser

If the guesser G is *deterministic* (i.e., its next output depends only on the history of values it has seen), the same guesser can be used to encode and decode.

An Idealised Guesser

If the guesser G is *deterministic* (i.e., its next output depends only on the history of values it has seen), the same guesser can be used to encode and decode.

Advantages

- Scheme works regardless of how the sequence $\mathbf{x}$ was generated
- Only need to compute codes for observed sequence
  (not all possible blocks of some size)

An Idealised Guesser

If the guesser G is *deterministic* (i.e., its next output depends only on the history of values it has seen), the same guesser can be used to encode and decode.

Advantages

- Scheme works regardless of how the sequence $\mathbf{x}$ was generated
- Only need to compute codes for observed sequence (not all possible blocks of some size)

How can we use this?

- Have the guesser guess distributions rather than symbols (repeatedly)
- Use bits instead of counts (with fewer bits for lower counts)
- Define a clever guesser (one that "learns")
- Prove that this scheme is close to optimal (at least, not much worse than Huffman on probabilistic sources)

Real Numbers in Binary

Real numbers are commonly expressed in decimal:

$$
\begin{aligned}
12_{10} &\rightarrow 1 \times 10^1 + 2 \times 10^0 \\
3.7_{10} &\rightarrow \qquad\qquad\qquad 3 \times 10^0 + 7 \times 10^{-1} \\
0.94_{10} &\rightarrow \qquad\qquad\qquad\qquad\qquad + 9 \times 10^{-1} + 4 \times 10^{-2}
\end{aligned}
$$

Real Numbers in Binary

Real numbers are commonly expressed in decimal:

$$
\begin{aligned}
12_{10} &\to 1 \times 10^1 + 2 \times 10^0 \\
3.7_{10} &\to \qquad\qquad 3 \times 10^0 + 7 \times 10^{-1} \\
0.94_{10} &\to \qquad\qquad\qquad\qquad + 9 \times 10^{-1} + 4 \times 10^{-2}
\end{aligned}
$$

Some real numbers have infinite, repeating decimal expansions:

$$\tfrac{1}{3} = 0.33333\ldots_{10} = 0.\overline{3}_{10} \quad \text{and} \quad \tfrac{22}{7} = 3.14285714\ldots_{10} = 3.\overline{142857}_{10}$$

Real Numbers in Binary

Real numbers are commonly expressed in decimal:

$$
\begin{aligned}
12_{10} &\to 1 \times 10^1 + 2 \times 10^0 \\
3.7_{10} &\to \qquad\qquad 3 \times 10^0 + 7 \times 10^{-1} \\
0.94_{10} &\to \qquad\qquad\qquad\qquad + 9 \times 10^{-1} + 4 \times 10^{-2}
\end{aligned}
$$

Some real numbers have infinite, repeating decimal expansions:

$$\frac{1}{3} = 0.33333\ldots_{10} = 0.\overline{3}_{10} \quad \text{and} \quad \frac{22}{7} = 3.14285714\ldots_{10} = 3.\overline{142857}_{10}$$

Real numbers can also be similarly expressed in binary:

$$
\begin{aligned}
11_{2} &\to 1 \times 2^1 + 1 \times 2^0 \\
1.1_{2} &\to \qquad\qquad 1 \times 2^0 + 1 \times 2^{-1} \\
0.01_{2} &\to \qquad\qquad\qquad\qquad + 0 \times 2^{-1} + 1 \times 2^{-2}
\end{aligned}
$$

$$\frac{1}{3} = 0.010101\ldots_{2} = 1.\overline{01}_{2} \quad \text{and} \quad \frac{22}{7} = 11.001001\ldots_{2} = 11.\overline{001}_{2}$$

Real Numbers in Binary

Real numbers are commonly expressed in decimal:

$$\begin{aligned} 12_{10} &\to 1 \times 10^1 + 2 \times 10^0 \\ 3.7_{10} &\to \qquad\qquad 3 \times 10^0 + 7 \times 10^{-1} \\ 0.94_{10} &\to \qquad\qquad\qquad\qquad + 9 \times 10^{-1} + 4 \times 10^{-2} \end{aligned}$$

Some real numbers have infinite, repeating decimal expansions:

$$\frac{1}{3} = 0.33333\ldots_{10} = 0.\overline{3}_{10} \quad \text{and} \quad \frac{22}{7} = 3.14285714\ldots_{10} = 3.\overline{142857}_{10}$$

Real numbers can also be similarly expressed in binary:

$$\begin{aligned} 11_2 &\to 1 \times 2^1 + 1 \times 2^0 \\ 1.1_2 &\to \qquad\qquad 1 \times 2^0 + 1 \times 2^{-1} \\ 0.01_2 &\to \qquad\qquad\qquad\qquad + 0 \times 2^{-1} + 1 \times 2^{-2} \end{aligned}$$

$$\frac{1}{3} = 0.010101\ldots_2 = 1.\overline{01}_2 \quad \text{and} \quad \frac{22}{7} = 11.001001\ldots_2 = 11.\overline{001}_2$$

Question: 1) What is 0.1011_2 in decimal? 2) What is $\frac{7}{5}$ in binary?

Intervals in Binary

An interval $[a, b)$ is the set of all the numbers at least as big as a but smaller than b. That is,

$$[a, b) = \{x : a \leq x < b\}.$$

Examples: $[0, 1)$, $[0.3, 0.6)$, $[0.2, 0.4)$.

Intervals in Binary

An interval $[a, b)$ is the set of all the numbers at least as big as a but smaller than b. That is,

$$[a, b) = \{x : a \leq x < b\}.$$

Examples: $[0, 1)$, $[0.3, 0.6)$, $[0.2, 0.4)$.

The set of numbers in $[0, 1)$ that start with a given sequence of bits $\mathbf{b} = b_1 \dots b_n$ form the interval $[0.b_1 \dots b_n, 0.(b_1 \dots b_n + 1))$.

- $1 \rightarrow [0.1, 1.0)$ $\quad [0.5, 1]_{10}$
- $01 \rightarrow [0.01, 0.10)$ $\quad [0.25, 0.5)_{10}$
- $1101 \rightarrow [0.1101, 0.1110)$ $\quad [0.8125, 0.875)_{10}$

An interval $[a, b)$ is the set of all the numbers at least as big as a but smaller than b. That is,

$$[a, b) = \{x : a \leq x < b\}.$$

Examples: $[0, 1)$, $[0.3, 0.6)$, $[0.2, 0.4)$.

The set of numbers in $[0, 1)$ that start with a given sequence of bits $\mathrm{b} = b_1 \ldots b_n$ form the interval $[0.b_1 \ldots b_n, 0.(b_1 \ldots b_n + 1))$.

- $1 \rightarrow [0.1, 1.0)$ $\quad [0.5, 1]_{10}$
- $01 \rightarrow [0.01, 0.10)$ $\quad [0.25, 0.5)_{10}$
- $1101 \rightarrow [0.1101, 0.1110)$ $\quad [0.8125, 0.875)_{10}$

If $\mathbf{b}'$ is a prefix of $\mathbf{b}$ the interval for $\mathbf{b}$ is contained in the interval for $\mathbf{b}'$.

$$\mathbf{b}' = 01 \text{ is prefix of } \mathbf{b} = 0101 \text{ so } \underbrace{[0.0101, 0.0110)}_{[0.3125, 0.375)_{10}} \subset \underbrace{[0.01, 0.10)}_{[0.25, 0.5)_{10}}$$

Cumulative Distribution

Suppose $\mathbf{p} = (p_1, \ldots, p_K)$ is a distribution over $\mathcal{A} = \{a_1, \ldots, a_K\}$.

Assume the alphabet $\mathcal{A}$ is ordered and define $a_i \leq a_j$ to mean $i \leq j$. Consider F the *cumulative distribution* for $\mathbf{p}$:

$$F(a) = P(x \leq a) = \sum_{a_i \leq a} p_i$$

Each symbol $a_i \in \mathcal{A}$ is associated with the interval $[F(a_{i-1}), F(a_i))$. (The value of $F(a_0) = 0$).

Cumulative Distribution

Suppose $\mathbf{p} = (p_1, \ldots, p_K)$ is a distribution over $\mathcal{A} = \{a_1, \ldots, a_K\}$.

Assume the alphabet $\mathcal{A}$ is ordered and define $a_i \leq a_j$ to mean $i \leq j$.
Consider F the *cumulative distribution* for $\mathbf{p}$:

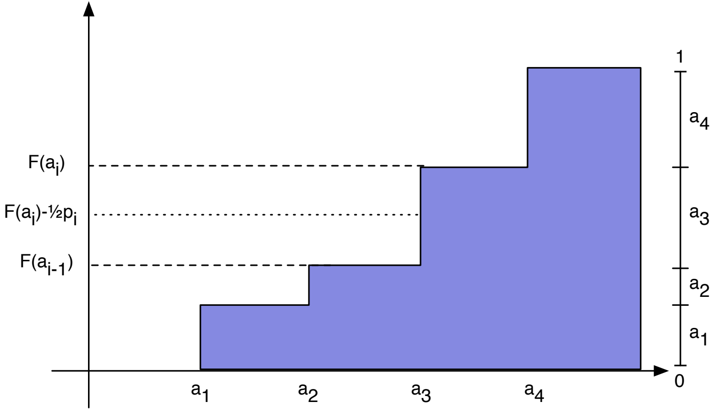
$$F(a) = P(x \leq a) = \sum_{a_i \leq a} p_i$$

Each symbol $a_i \in \mathcal{A}$ is associated with the interval $[F(a_{i-1}), F(a_i))$.
(The value of $F(a_0) = 0$).

Example:
$\mathcal{A} = \{r, g, b\} \implies a_1 = r,\ a_2 = g,\ a_3 = b \implies r \leq b$ since $1 \leq 3$.
If $\mathbf{p} = (\frac{1}{2}, \frac{1}{4}, \frac{1}{4})$ then $F(r) = \frac{1}{2}$, $F(g) = \frac{3}{4}$, $F(b) = 1$.

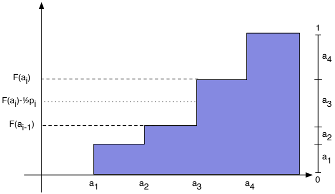
Cumulative Distribution

Example



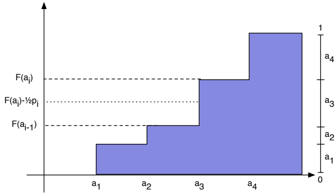
Cumulative distribution for $\mathbf{p} = (\frac{2}{9}, \frac{1}{9}, \frac{1}{3}, \frac{1}{3})$

Shannon-Fano-Elias Coding



$$[0, \tfrac{2}{9})_{10} \qquad [0, 0.\overline{001110})_2$$
$$[\tfrac{2}{9}, \tfrac{1}{3})_{10} \qquad [0.\overline{001110}, 0.\overline{01})_2$$
$$[\tfrac{1}{3}, \tfrac{2}{3})_{10} \qquad [0.\overline{01}, 0.\overline{10})_2$$
$$[\tfrac{2}{3}, 1)_{10} \qquad [0.\overline{10}, 1)_2$$

Shannon-Fano-Elias Coding



$[0, \frac{2}{9})_{10}$ $[0, 0.\overline{001110})_2$

$[\frac{2}{9}, \frac{1}{3})_{10}$ $[0.\overline{001110}, 0.\overline{01})_2$

$[\frac{1}{3}, \frac{2}{3})_{10}$ $[0.\overline{01}, 0.\overline{10})_2$

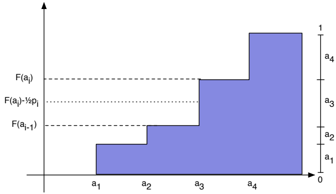
$[\frac{2}{3}, 1)_{10}$ $[0.\overline{10}, 1)_2$

Define the midpoint $\bar{F}(a_i) = F(a_i) - \frac{1}{2}p_i$ and length $\ell(a_i) = \left\lceil \log_2 \frac{1}{p_i} \right\rceil + 1$.

Code $x \in \mathcal{A}$ using first $\ell(x)$ bits of $\bar{F}(x)$.

x	$p(x)$	$F(x)$	$\bar{F}(x)$	$\bar{F}(x)_2$	$\ell(x)$	Code
a_1	2/9	2/9	1/9	$0.\overline{000111}_2$	4	0001
a_2	1/9	1/3	5/18	$0.01\overline{000111}_2$	5	01000
a_3	1/3	2/3	1/2	0.1_2	3	100
a_4	1/3	1	5/6	$0.1\overline{10}_2$	3	110

Shannon-Fano-Elias Coding



$[0, \frac{2}{9})_{10}$	$[0, 0.\overline{001110})_2$
$[\frac{2}{9}, \frac{1}{3})_{10}$	$[0.\overline{001110}, 0.\overline{01})_2$
$[\frac{1}{3}, \frac{2}{3})_{10}$	$[0.\overline{01}, 0.\overline{10})_2$
$[\frac{2}{3}, 1)_{10}$	$[0.\overline{10}, 1)_2$

Define the midpoint $\bar{F}(a_i) = F(a_i) - \frac{1}{2}p_i$ and length $\ell(a_i) = \left\lceil \log_2 \frac{1}{p_i} \right\rceil + 1$.

Code $x \in \mathcal{A}$ using first $\ell(x)$ bits of $\bar{F}(x)$.

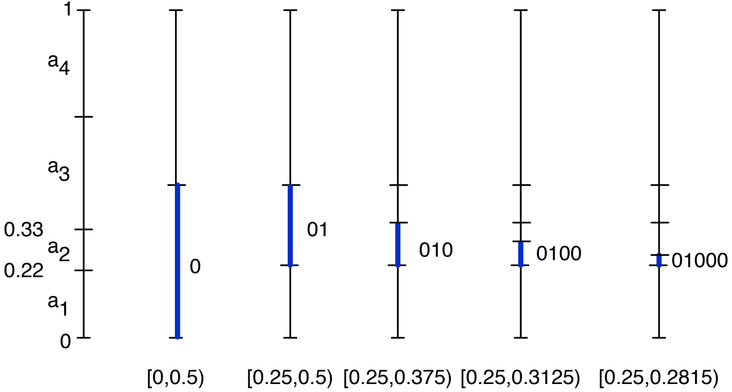
x	$p(x)$	$F(x)$	$\bar{F}(x)$	$\bar{F}(x)_2$	$\ell(x)$	Code
a_1	2/9	2/9	1/9	$0.\overline{000111}_2$	4	0001
a_2	1/9	1/3	5/18	$0.01\overline{000111}_2$	5	01000
a_3	1/3	2/3	1/2	0.1_2	3	100
a_4	1/3	1	5/6	$0.1\overline{10}_2$	3	110

Example: Sequence $\mathbf{x} = a_3 a_3 a_1$ coded as 100 100 0001.

Shannon-Fano-Elias Decoding

Let $\mathbf{p} = \{\frac{2}{9}, \frac{1}{9}, \frac{1}{3}, \frac{1}{3}\}$. Suppose we want to *decode* 01000:

Find symbol whose interval contains interval for 01000



Note: We did not need to know all the codewords in C.

Expected Code Length of SFE Code

The extra bit for the code lengths is because we code $\frac{p_i}{2}$ and

$$\log_2 \frac{2}{p_i} = \log_2 \frac{1}{p_i} + \log_2 2 = \log_2 \frac{1}{p_i} + 1$$

What is the expected length of a SFE code C for ensemble X with probabilities $\mathbf{p}$?

$$\begin{aligned} L(C, X) = \sum_{i=1}^{K} p_i \ell(a_i) &= \sum_{i=1}^{K} p_i \left(\left\lceil \log_2 \frac{1}{p_i} \right\rceil + 1 \right) \\ &\leq \sum_{i=1}^{K} p_i \left(\log_2 \frac{1}{p_i} + 2 \right) \\ &= H(X) + 2 \end{aligned}$$

Similarly, $H(X) + 1 \leq L(C, X)$ for the SFE codes.

What does the extra bit buy us?

Let X be an ensemble, C_{SFE} be a Shannon-Fano-Elias code for X and C_H be a Huffman code for X.

$$\underbrace{H(X) \leq L(C_H, X) \leq H(X) + 1}_{\text{Source Coding Theorem}} \leq L(C_{SFE}, X) \leq H(X) + 2$$

What does the extra bit buy us?

Let X be an ensemble, C_{SFE} be a Shannon-Fano-Elias code for X and C_H be a Huffman code for X.

$$\underbrace{H(X) \leq L(C_H, X) \leq H(X) + 1}_{\text{Source Coding Theorem}} \leq L(C_{SFE}, X) \leq H(X) + 2$$

The extra bit *guarantees* that the interval for each code word lies entirely within the interval for each symbol. [Why?]

What does the extra bit buy us?

Let X be an ensemble, C_{SFE} be a Shannon-Fano-Elias code for X and C_H be a Huffman code for X.

$$\underbrace{H(X) \le L(C_H, X) \le H(X) + 1}_{\text{Source Coding Theorem}} \le L(C_{SFE}, X) \le H(X) + 2$$

The extra bit *guarantees* that the interval for each code word lies entirely within the interval for each symbol. [Why?]

Example: SFE Code for $\mathbf{p} = \{\frac{2}{9}, \frac{1}{9}, \frac{1}{3}, \frac{1}{3}\}$ is $C = \{0001, 01000, 100, 110\}$. Intervals for $\mathbf{p}$ are

$$[0, 0.22), [0.22, 0.33), [0.33, 0.66), [0.66, 1)$$

Intervals for $\{000, 0100, 10, 11\}$ are

$$[0, 0.125), [0.25, 0.3125), [0.5, 0.75), [0.75, 1)$$

Summary and Reading

Main points:

- Problems with Huffman coding
- Guessing game for coding without assuming fixed symbol distribution
- Binary strings to/from intervals in $[0, 1]$
- Shannon-Fano-Elias Coding:
  - Code C via cumulative distribution function for p
  - $H(X) + 1 \leq L(C, X) \leq H(X) + 2$
- Extra bit guarantees interval containment

Summary and Reading

Main points:

- Problems with Huffman coding
- Guessing game for coding without assuming fixed symbol distribution
- Binary strings to/from intervals in $[0, 1]$
- Shannon-Fano-Elias Coding:
  - Code C via cumulative distribution function for p
  - $H(X) + 1 \leq L(C, X) \leq H(X) + 2$
- Extra bit guarantees interval containment

Reading:

- Guessing game and interval coding: MacKay §6.1 and §6.2
- Shannon-Fano-Elias Coding: Cover & Thomas §5.9

Summary and Reading

Main points:

- Problems with Huffman coding
- Guessing game for coding without assuming fixed symbol distribution
- Binary strings to/from intervals in $[0, 1]$
- Shannon-Fano-Elias Coding:
  - Code C via cumulative distribution function for p
  - $H(X) + 1 \leq L(C, X) \leq H(X) + 2$
- Extra bit guarantees interval containment

Reading:

- Guessing game and interval coding: MacKay §6.1 and §6.2
- Shannon-Fano-Elias Coding: Cover & Thomas §5.9

Next time:

Extending SFE Coding to sequences of symbols