Arithmetic Coding

Expository Presentation

Aryan Nath

Problem Statement

- Consider the source alphabet $\{a_1, a_2, a_3\}$ with probabilities $P(a1) = 0.95, P(a2) = 0.02$, and $P(a3) = 0.03$.
- Source entropy $= 0.335$ bits/symbol
- Avg code length per symbol using Huffman coding $= 1.05$ bits.
- By coding in blocks of two symbols we achieve 0.611 bits/symbol.
- For block length $= 8$, possible combinations $= 3^8 = 6561$!

Arithmetic Coding

- Generate codewords for the entire letter sequence instead of for each block.
- Unique tag generation.
- Correct decoding.

Coding a Sequence

- Very large number of possible message sequences.
- Need very large number of values (infinite) to generate unique tags.
- One possible range: $[0, 1)$.

Generating a Tag

- Can we assign a unique sub-interval that respects the order of our sequence (and generate a tag from it)?
- Use the cumulative distribution function and partition the interval $[0, 1)$.
- From the message sequence, associate each symbol a_k with the sub-interval $[F_X(k-1), F_X(k))$.
- Partition the sub-interval in the same way as the unit interval.

Example

Consider the source:
$\mathcal{A} = \{a_1, a_2, a_4\}$ with $P(a_1) = 0.7, P(a_2) = 0.1, andP(a_3) = 0.2.$

Cumulative distribution function: $F_X(1) = 0.7, F_X(2) = 0.8$, and $F_X(3) = 1$.

Associated intervals: $\{a_1 : [0.0, 0.7), a_2 : [0.7, 0.8), a_3 : [0.8, 1)\}$.

Partition the unit interval.

Figure: Partition the unit interval

Example: Finding Sub-intervals

Suppose the first symbol in the sequence is a_1.

Figure: First sub-interval

Example: Finding Sub-intervals

Suppose the first symbol in the sequence is a_1.

We then divide this sub-interval in the same proportion as the original interval.

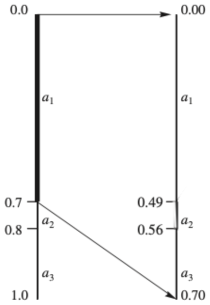

Figure: Partition the first sub-interval

Example: Finding Sub-intervals

Suppose the second symbol is a_2.

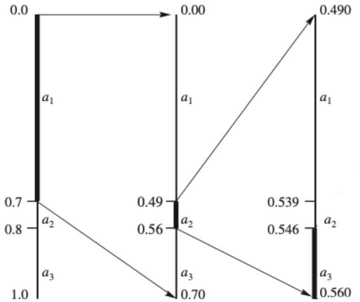

Figure: Partition the second sub-interval

Example: Finding Sub-intervals

The last symbol in the message is a_3.

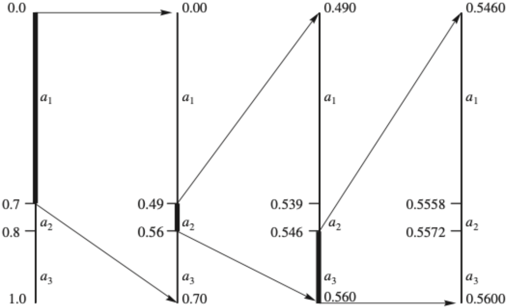

Figure: Partition the second sub-interval

A unique sub-interval is selected at each step, and it respects the order of the sequence.
Select a tag from the final interval. The binary representation of the tag gives a unique encoding for the sequence.

Example: Updating the Interval Bounds

After the n^{th} symbol in the sequence has been generated, the upper and lower limits of the updated sub-interval $[l^{(n)}, u^{(n)]})$ will be:

$$l^{(n)} = l^{(n-1)} + (u^{(n-1)} - l^{(n-1)})F_X(x_n - 1)$$

$$u^{(n)} = l^{(n-1)} + (u^{(n-1)} - l^{(n-1)})F_X(x_n)$$

Precision Problem

- As the sequence length increases, the subinterval becomes smaller and smaller.

  $\perp \rightarrow a_1 \rightarrow a_2 \rightarrow a_3$.

  $[0, 1] \rightarrow [0.00, 0.70) \rightarrow [0.490, 0.560) \rightarrow [0.5460, 0.5600)$.
- Fixed bit representation (64 bits) for floating points, having more bits for the tag will lead to information loss, and hence incorrect decoding.
- To avoid this, we need to rescale the intervals while preserving the uniqueness of the tag.

Solution Overview

While computing the sub-intervals, we will have 3 cases:

1. The interval is entirely contained in the lower half of the unit interval $[0, 0.5)$ (E_1 rescaling).
2. The interval is entirely contained in the upper half of the unit interval $[0.5, 1.0)$ (E_2 rescaling).
3. The interval straddles the midpoints of the unit interval (E_3 rescaling).

Solution Overview: Incremental Encoding

- The most significant bit of tag in the interval $[0, 0.5)$ has to be $\mathbf{0}$.
- The most significant bit of tag in the interval $[0.5, 1)$ has to be $\mathbf{1}$.
- Either of these intervals uniquely determines the most significant bit of the tag. We send this bit to the decoder without waiting for subsequent sequence members.
- Now, we take the same proportion as the unit interval inside the sub-interval, and then we also rescale the sub-interval.
- We can safely remove the other half of the unit interval, through the following scaling:

$$E_1 : [0, 0.5) \rightarrow [0, 1); E_1(x) = 2x$$

$$E_2 : [0.5, 1) \rightarrow [0, 1); E_2(x) = 2(x - 0.5)$$

- Since we are sending the most significant bit of the tag whenever the most significant bit of the upper and lower limit of its sub-interval are equal, we are generating the binary representation of the tag itself.
- The scaling operations are just left shifts on the tag.
- This is the same tag as our original encoding, which we have already argued to be unique.

Solution Overview: Decoding

- Initialize $u^{(0)}$ to 1 and $;^{(0)}$ to 0.
- Given the tag value, find the symbol a_k such that the tag is within the interval $[l^{(1)}, u^{(1)})$, where:

$$l^{(1)} = 0 + (1 - 0)F_X(x_n - 1)$$

$$u^{(1)} = 0 + (1 - 0)F_X(x_n)$$

- This is element a_k will be the first element in the decoding.
- Repeat this with $[l^{(2)}, u^{(2)})$ and onwards.
- Every time we our interval $[l^{(i)}, u^{(i)})$ satisfies the condition for E_1 scaling or E_2 scaling, we perform the scaling and remove the MSB of the tag.

Example

```
Source:
Symbols: [1, 2, 3, 4, 5]
Probabilities: {1: 0.5, 2: 0.25, 3: 0.125, 4: 0.0625, 5: 0.0625}

Cumulative distribution function: {1: 0.5, 2: 0.75, 3: 0.875, 4: 0.9375, 5: 1.0}


Message generated by the source: [3, 1, 4, 3]
Message length: 4
```

```
Encoding message...
Original Interval: [0.0, 1.0)

Encoding first symbol
Current symbol: 3
L_(1) = 0.0 + (1.0 - 0.0)*0.75
U_(1) = 0.0 + (1.0 - 0.0)*0.875
Updated Interval: [0.75, 0.875)
```

Example

```
E2 Scaling
L_(1) = 2*(0.75 - 0.5)
U_(1) = 2*(0.875 - 0.5)
Adding 1 to the encoding
Current encoding: 1
Scaled Interval: [0.5, 0.75)


E2 Scaling
L_(1) = 2*(0.5 - 0.5)
U_(1) = 2*(0.75 - 0.5)
Adding 1 to the encoding
Current encoding: 11
Scaled Interval: [0.0, 0.5)


E1 Scaling
L_(1) = 2*0.0
U_(1) = 2*0.5
Adding 0 to the encoding
Current encoding: 110
Scaled Interval: [0.0, 1.0)
```

Example

```
Current Tag: 0.75

Current encoding: 110
Current symbol: 1
L_(2) = 0.0 + (1.0 - 0.0)*0.0
U_(2) = 0.0 + (1.0 - 0.0)*0.5
Updated Interval: [0.0, 0.5)


E1 Scaling
L_(2) = 2*0.0
U_(2) = 2*0.5
Adding 0 to the encoding
Current encoding: 1100
Scaled Interval: [0.0, 1.0)
```

Example

```
Current Tag: 0.75

Current encoding: 1100
Current symbol: 4
L_(3) = 0.0 + (1.0 - 0.0)*0.875
U_(3) = 0.0 + (1.0 - 0.0)*0.9375
Updated Interval: [0.875, 0.9375)


E2 Scaling
L_(3) = 2*(0.875 - 0.5)
U_(3) = 2*(0.9375 - 0.5)
Adding 1 to the encoding
Current encoding: 11001
Scaled Interval: [0.75, 0.875)


E2 Scaling
L_(3) = 2*(0.75 - 0.5)
U_(3) = 2*(0.875 - 0.5)
Adding 1 to the encoding
Current encoding: 110011
Scaled Interval: [0.5, 0.75)


E2 Scaling
L_(3) = 2*(0.5 - 0.5)
U_(3) = 2*(0.75 - 0.5)
Adding 1 to the encoding
Current encoding: 1100111
Scaled Interval: [0.0, 0.5)
```

Example

```
E1 Scaling
L_(3) = 2*0.0
U_(3) = 2*0.5
Adding 0 to the encoding
Current encoding: 11001110
Scaled Interval: [0.0, 1.0)

Current Tag: 0.8046875

Current encoding: 11001110
Current symbol: 3
L_(4) = 0.0 + (1.0 - 0.0)*0.75
U_(4) = 0.0 + (1.0 - 0.0)*0.875
Updated Interval: [0.75, 0.875)


E2 Scaling
L_(4) = 2*(0.75 - 0.5)
U_(4) = 2*(0.875 - 0.5)
Adding 1 to the encoding
Current encoding: 110011101
Scaled Interval: [0.5, 0.75)


E2 Scaling
L_(4) = 2*(0.5 - 0.5)
U_(4) = 2*(0.75 - 0.5)
Adding 1 to the encoding
Current encoding: 1100111011
Scaled Interval: [0.0, 0.5)
```

Example

```
E2 Scaling
L_(4) = 2*(0.5 - 0.5)
U_(4) = 2*(0.75 - 0.5)
Adding 1 to the encoding
Current encoding: 1100111011
Scaled Interval: [0.0, 0.5)


E1 Scaling
L_(4) = 2*0.0
U_(4) = 2*0.5
Adding 0 to the encoding
Current encoding: 11001110110
Scaled Interval: [0.0, 1.0)

Current Tag: 0.8076171875

Generated tag: 0.8076171875



Encoded message: 11001110110
```

Example

```
Decoding message...
Original Interval: [0.0, 1)
Word length: 40
Padding tag with 29 0s on the right
Orignal tag encoding: 11001110110
Padded: 1100111011000000000000000000000000000000

Current tag: 0.8076171875
L_(1) = 0.0 + (1 - 0.0)*0.75
U_(1) = 0.0 + (1 - 0.0)*0.875
Decoded first symbol: 3
Updated Interval: [0.75,0.875)
Current decoded message [3]

E2 Scaling
L_(1) = 2*(0.75 - 0.5)
U_(1) = 2*(0.875 - 0.5)
Removing MSB from the tag and retaining the word length
Current tag encoding: 1001110110000000000000000000000000000000
Scaled Interval: [0.5, 0.75)


E2 Scaling
L_(1) = 2*(0.5 - 0.5)
U_(1) = 2*(0.75 - 0.5)
Removing MSB from the tag and retaining the word length
Current tag encoding: 0011101100000000000000000000000000000000
Scaled Interval: [0.0, 0.5)
```

Example

```
E1 Scaling
L_(1) = 2*0.0
U_(1) = 2*0.5
Removing MSB from the tag and retaining the word length
Current tag encoding: 0111011000000000000000000000000000000000
Scaled Interval: [0.0, 1.0)



Current tag: 0.4609375
L_(2) = 0.0 + (1.0 - 0.0)*0.0
U_(2) = 0.0 + (1.0 - 0.0)*0.5
Decoded symbol: 1
Updated Interval: [0.0,0.5)
Current decoded message [3, 1]

E1 Scaling
L_(2) = 2*0.0
U_(2) = 2*0.5
Removing MSB from the tag and retaining the word length
Current tag encoding: 1110110000000000000000000000000000000000
Scaled Interval: [0.0, 1.0)



Current tag: 0.921875
L_(3) = 0.0 + (1.0 - 0.0)*0.875
U_(3) = 0.0 + (1.0 - 0.0)*0.9375
Decoded symbol: 4
Updated Interval: [0.875,0.9375)
Current decoded message [3, 1, 4]
```

Example

```
E2 Scaling
L_(3) = 2*(0.875 - 0.5)
U_(3) = 2*(0.9375 - 0.5)
Removing MSB from the tag and retaining the word length
Current tag encoding: 1101100000000000000000000000000000000000
Scaled Interval: [0.75, 0.875)


E2 Scaling
L_(3) = 2*(0.75 - 0.5)
U_(3) = 2*(0.875 - 0.5)
Removing MSB from the tag and retaining the word length
Current tag encoding: 1011000000000000000000000000000000000000
Scaled Interval: [0.5, 0.75)


E2 Scaling
L_(3) = 2*(0.5 - 0.5)
U_(3) = 2*(0.75 - 0.5)
Removing MSB from the tag and retaining the word length
Current tag encoding: 0110000000000000000000000000000000000000
Scaled Interval: [0.0, 0.5)


E1 Scaling
L_(3) = 2*0.0
U_(3) = 2*0.5
Removing MSB from the tag and retaining the word length
Current tag encoding: 1100000000000000000000000000000000000000
Scaled Interval: [0.0, 1.0)
```

Example

```
Current tag: 0.75
L_(4) = 0.0 + (1.0 - 0.0)*0.75
U_(4) = 0.0 + (1.0 - 0.0)*0.875
Decoded symbol: 3
Updated Interval: [0.75,0.875)
Current decoded message [3, 1, 4, 3]

E2 Scaling
L_(4) = 2*(0.75 - 0.5)
U_(4) = 2*(0.875 - 0.5)
Removing MSB from the tag and retaining the word length
Current tag encoding: 1000000000000000000000000000000000000000
Scaled Interval: [0.5, 0.75)


E2 Scaling
L_(4) = 2*(0.5 - 0.5)
U_(4) = 2*(0.75 - 0.5)
Removing MSB from the tag and retaining the word length
Current tag encoding: 0000000000000000000000000000000000000000
Scaled Interval: [0.0, 0.5)


E1 Scaling
L_(4) = 2*0.0
U_(4) = 2*0.5
Removing MSB from the tag and retaining the word length
Current tag encoding: 0000000000000000000000000000000000000000
Scaled Interval: [0.0, 1.0)
```

Example

```
Decoded message: [3, 1, 4, 3]
Original message: [3, 1, 4, 3]

Encoding and decoding successful!

Avg bits per symbol from arithmetic coding: 2.75
Source Entropy: 1.875
Percentage difference: 46.666666666666664
```

Thank you