Codes and Cryptography

Jorge L. Villar

MAMME, Fall 2015

PART III

Outline

Sources

Information source: sequence of random variables taking values on a finite source alphabet.

Sources

Information source: sequence of random variables taking values on a finite source alphabet.

Stateless and **stationary** source: the random variables are independent and identically distributed.

Sources

Information source: sequence of random variables taking values on a finite source alphabet.

Stateless and **stationary** source: the random variables are independent and identically distributed.

Mathematical model of a source $\mathcal{S}$:

- Source alphabet $\mathcal{A} = \{\alpha_1, \ldots, \alpha_n\}$
- Probability distribution $(p_1, \ldots, p_n)$ where p_i is the probability of symbol α_i

Sources

Information source: sequence of random variables taking values on a finite source alphabet.

Stateless and **stationary** source: the random variables are independent and identically distributed.

Mathematical model of a source $\mathcal{S}$:

- Source alphabet $\mathcal{A} = \{\alpha_1, \ldots, \alpha_n\}$
- Probability distribution $(p_1, \ldots, p_n)$ where p_i is the probability of symbol α_i

We simply say $\mathcal{S} = (p_1, \ldots, p_n)$

Optimal Codes (I)

Optimal code: an r-ary uniquely-decodable code $\mathcal{C}$ that minimizes the average codeword length

$$L(\mathcal{C}, \mathcal{S}) = \sum_{i=1}^{n} l(w_i) p_i$$

where w_i is the codeword associated to source symbol α_i

Optimal Codes (I)

Optimal code: an r-ary uniquely-decodable code $\mathcal{C}$ that minimizes the average codeword length

$$L(\mathcal{C}, \mathcal{S}) = \sum_{i=1}^{n} l(w_i) p_i$$

where w_i is the codeword associated to source symbol α_i

It is the meaningful efficiency measure when long sequences of input symbols are to be encoded

Optimal Codes (I)

Optimal code: an r-ary uniquely-decodable code $\mathcal{C}$ that minimizes the average codeword length

$$L(\mathcal{C}, \mathcal{S}) = \sum_{i=1}^{n} l(w_i) p_i$$

where w_i is the codeword associated to source symbol α_i

It is the meaningful efficiency measure when long sequences of input symbols are to be encoded

From McMillan's inequality we have the additional constraint

$$\sum_{i=1}^{n} r^{-l(w_i)} \leq 1$$

Optimal Codes (I)

Optimal code: an r-ary uniquely-decodable code $\mathcal{C}$ that minimizes the average codeword length

$$L(\mathcal{C}, \mathcal{S}) = \sum_{i=1}^{n} l(w_i) p_i$$

where w_i is the codeword associated to source symbol α_i

It is the meaningful efficiency measure when long sequences of input symbols are to be encoded

From McMillan's inequality we have the additional constraint

$$\sum_{i=1}^{n} r^{-l(w_i)} \leq 1$$

There always exists a prefix-free optimal code

Optimal Codes (II)

We solve the continuous optimization problem:

Minimize $L = \sum_{i=1}^{n} l_i p_i$ for $l_1, \ldots, l_n$ **subject to** $\sum_{i=1}^{n} r^{-l_i} \leq 1$

Optimal Codes (II)

We solve the continuous optimization problem:

Minimize $L = \sum_{i=1}^{n} l_i p_i$ for $l_1, \ldots, l_n$ **subject to** $\sum_{i=1}^{n} r^{-l_i} \leq 1$

As L is a linear function, its minima are on the boundary. Now, using Lagrange multipliers

$$\nabla L = (p_1, \ldots, p_n) = \lambda \nabla \left(\sum_{i=1}^{n} r^{-l_i} - 1 \right) = -\lambda \ln r (r^{-l_1}, \ldots, r^{-l_n})$$

Optimal Codes (II)

We solve the continuous optimization problem:

Minimize $L = \sum_{i=1}^{n} l_i p_i$ for $l_1, \ldots, l_n$ **subject to** $\sum_{i=1}^{n} r^{-l_i} \leq 1$

As L is a linear function, its minima are on the boundary. Now, using Lagrange multipliers

$$\nabla L = (p_1, \ldots, p_n) = \lambda \nabla \left(\sum_{i=1}^{n} r^{-l_i} - 1 \right) = -\lambda \ln r (r^{-l_1}, \ldots, r^{-l_n})$$

Thus, $-\lambda \ln r = 1$ since $\sum_{i=1}^{n} r^{-l_i} = \sum_{i=1}^{n} p_i = 1$, and $p_i = r^{-l_i}$.

Optimal Codes (II)

We solve the continuous optimization problem:

Minimize $L = \sum_{i=1}^{n} l_i p_i$ for $l_1, \ldots, l_n$ **subject to** $\sum_{i=1}^{n} r^{-l_i} \leq 1$

As L is a linear function, its minima are on the boundary. Now, using Lagrange multipliers

$$\nabla L = (p_1, \ldots, p_n) = \lambda \nabla \left(\sum_{i=1}^{n} r^{-l_i} - 1 \right) = -\lambda \ln r (r^{-l_1}, \ldots, r^{-l_n})$$

Thus, $-\lambda \ln r = 1$ since $\sum_{i=1}^{n} r^{-l_i} = \sum_{i=1}^{n} p_i = 1$, and $p_i = r^{-l_i}$.

Therefore $l_i = -\log_r p_i$ and $L_{opt} = -\sum_{i=1}^{n} p_i \log_r p_i$

Shannon's Lower Bound

We have find the following lower bound

Theorem (Shannon)

For any stateless stationary source $\mathcal{S} = (p_1, \ldots, p_n)$ *and any* r*-ary uniquely-decodable code* $\mathcal{C}$

$$L(\mathcal{C}, \mathcal{S}) \geq L_r(\mathcal{S}) = -\sum_{i=1}^{n} p_i \log_r p_i$$

Shannon's Lower Bound

We have find the following lower bound

Theorem (Shannon)

For any stateless stationary source $\mathcal{S} = (p_1, \ldots, p_n)$ *and any* r*-ary uniquely-decodable code* $\mathcal{C}$

$$L(\mathcal{C}, \mathcal{S}) \geq L_r(\mathcal{S}) = -\sum_{i=1}^{n} p_i \log_r p_i$$

However, this lower bound is achievable if and only if all $l_i = -\log_r p_i$ are integers (i.e., p_i are negative powers of r).

Example: The Uniform Source

Consider as an example the case $p_1 = \cdots = p_n = \frac{1}{n}$, where $n = r^d$ for some integer d

Example: The Uniform Source

Consider as an example the case $p_1 = \cdots = p_n = \frac{1}{n}$, where $n = r^d$ for some integer d

Now $L_r(\mathcal{S}) = -n\frac{1}{n}\log_r\left(\frac{1}{n}\right) = \log_r n = d$ is achievable by taking $l_i = -\log_r\left(\frac{1}{n}\right) = d$

The optimal code is the constant length code $\mathcal{C} = \{0, \ldots, r-1\}^d$

Shannon-Fano Codes

A good approximation for the optimal (continuous) case is taking $l_i = \lceil -\log_r p_i \rceil$.

There always exists a prefix-free code, the Shannon-Fano code, for this codeword length sequence, since

$$\sum_{i=1}^{n} r^{-l_i} = \sum_{i=1}^{n} r^{-\lceil -\log_r p_i \rceil} \leq \sum_{i=1}^{n} r^{\log_r p_i} = \sum_{i=1}^{n} p_i = 1$$

Shannon-Fano Codes

A good approximation for the optimal (continuous) case is taking $l_i = \lceil -\log_r p_i \rceil$.

There always exists a prefix-free code, the Shannon-Fano code, for this codeword length sequence, since

$$\sum_{i=1}^{n} r^{-l_i} = \sum_{i=1}^{n} r^{-\lceil -\log_r p_i \rceil} \leq \sum_{i=1}^{n} r^{\log_r p_i} = \sum_{i=1}^{n} p_i = 1$$

On the other hand, the code is near-optimal

$$L(\mathcal{C}, \mathcal{S}) = \sum_{i=1}^{n} p_i \lceil -\log_r p_i \rceil < \sum_{i=1}^{n} p_i(-\log_r p_i + 1) = L_r(\mathcal{S}) + 1$$

Source Coalescence

Consider two sources $\mathcal{S}_1 = (p_1, \ldots, p_n)$, $\mathcal{S}_2 = (q_1, \ldots, q_m)$. We can define a new source $\mathcal{S}_1 \times \mathcal{S}_2 = (p_1 q_1, p_1 q_2, \ldots, p_n q_m)$ for the cartesian product of the two source alphabets.

The two information sources are merged into a single one

Source Coalescence

Consider two sources $\mathcal{S}_1 = (p_1, \ldots, p_n)$, $\mathcal{S}_2 = (q_1, \ldots, q_m)$. We can define a new source $\mathcal{S}_1 \times \mathcal{S}_2 = (p_1 q_1, p_1 q_2, \ldots, p_n q_m)$ for the cartesian product of the two source alphabets.

The two information sources are merged into a single one

A code $\mathcal{C}$ for $\mathcal{S}_1 \times \mathcal{S}_2$ is trivially obtained by concatenating $\mathcal{C}_1$ for $\mathcal{S}_1$ and $\mathcal{C}_2$ for $\mathcal{S}_2$. (But better codes can exist!)

Source Coalescence

Consider two sources $\mathcal{S}_1 = (p_1, \ldots, p_n)$, $\mathcal{S}_2 = (q_1, \ldots, q_m)$. We can define a new source $\mathcal{S}_1 \times \mathcal{S}_2 = (p_1 q_1, p_1 q_2, \ldots, p_n q_m)$ for the cartesian product of the two source alphabets.

The two information sources are merged into a single one

A code $\mathcal{C}$ for $\mathcal{S}_1 \times \mathcal{S}_2$ is trivially obtained by concatenating $\mathcal{C}_1$ for $\mathcal{S}_1$ and $\mathcal{C}_2$ for $\mathcal{S}_2$. (But better codes can exist!)

However, $L_r(\mathcal{S}_1 \times \mathcal{S}_2) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p_i q_j \log_r(p_i q_j) =$

$$-\sum_{i=1}^{n} p_i \log_r p_i - \sum_{j=1}^{m} q_j \log_r q_j = L_r(\mathcal{S}_1) + L_r(\mathcal{S}_2)$$

Source Extensions (I)

Given a source $\mathcal{S} = (p_1, \ldots, p_n)$ we can define its k-th extension $\mathcal{S}^k = \underbrace{\mathcal{S} \times \cdots \times \mathcal{S}}_{k \text{ times}}$.

Idea: encode k-tuples instead of separate symbols

Source Extensions (I)

Given a source $\mathcal{S} = (p_1, \ldots, p_n)$ we can define its k-th extension $\mathcal{S}^k = \underbrace{\mathcal{S} \times \cdots \times \mathcal{S}}_{k \text{ times}}$.

Idea: encode k-tuples instead of separate symbols

Now, $L_r(\mathcal{S}^k) = kL_r(\mathcal{S})$ and the Shannon-Fano code $\mathcal{C}_k$ for $\mathcal{S}^k$ satisfies $L_r(\mathcal{S}^k) \leq L(\mathcal{C}_k, \mathcal{S}^k) < L_r(\mathcal{S}^k) + 1$

Source Extensions (I)

Given a source $\mathcal{S} = (p_1, \ldots, p_n)$ we can define its k-th extension $\mathcal{S}^k = \underbrace{\mathcal{S} \times \cdots \times \mathcal{S}}_{k \text{ times}}$.

Idea: encode k-tuples instead of separate symbols

Now, $L_r(\mathcal{S}^k) = kL_r(\mathcal{S})$ and the Shannon-Fano code $\mathcal{C}_k$ for $\mathcal{S}^k$ satisfies $L_r(\mathcal{S}^k) \leq L(\mathcal{C}_k, \mathcal{S}^k) < L_r(\mathcal{S}^k) + 1$

But the correct performance measure for k-tuples is

$$L_r(\mathcal{S}) \leq \frac{L(\mathcal{C}_k, \mathcal{S}^k)}{k} < L_r(\mathcal{S}) + \frac{1}{k}$$

In the context of source extensions, **Shannon's lower bound is asymptotically achieved** by Shannon-Fano codes.

Source Extensions (II)

Theorem (Noiseless Coding Theorem [Shannon, 1948])

For any stateless stationary source $\mathcal{S}$ and any family $\{\mathcal{C}_k\}_{k\in\mathbb{Z}^+}$ of r-ary optimal uniquely-decodable codes for $\{\mathcal{S}^k\}_{k\in\mathbb{Z}^+}$

$$\lim_{k\to\infty} \frac{L(\mathcal{C}_k, \mathcal{S}^k)}{k} = L_r(\mathcal{S})$$

Source Extensions (II)

Theorem (Noiseless Coding Theorem [Shannon, 1948])

For any stateless stationary source $\mathcal{S}$ and any family $\{\mathcal{C}_k\}_{k\in\mathbb{Z}^+}$ of r-ary optimal uniquely-decodable codes for $\{\mathcal{S}^k\}_{k\in\mathbb{Z}^+}$

$$\lim_{k\to\infty} \frac{L(\mathcal{C}_k, \mathcal{S}^k)}{k} = L_r(\mathcal{S})$$

The proof simply considers the fact that any optimal code performs better than any other code, in particular, better that Shannon-Fano codes

Outline

Construction of Optimal Codes (I)

. . . Coming back to the discrete optimization problem. . .

Shannon-Fano codes are guaranteed to be optimal only when all p_i are negative powers of r, but we still need a construction for the general case.

Construction of Optimal Codes (I)

. . . Coming back to the discrete optimization problem. . .

Shannon-Fano codes are guaranteed to be optimal only when all p_i are negative powers of r, but we still need a construction for the general case.

Given $\mathcal{S} = (p_1, \ldots, p_n)$ we look for an optimal r-ary prefix-free code $\mathcal{C}$ with some codeword lengths $l_1, \ldots, l_n$. Let's assume that $p_1 \geq \cdots \geq p_n$).

Construction of Optimal Codes (I)

. . . Coming back to the discrete optimization problem. . .

Shannon-Fano codes are guaranteed to be optimal only when all p_i are negative powers of r, but we still need a construction for the general case.

Given $\mathcal{S} = (p_1, \ldots, p_n)$ we look for an optimal r-ary prefix-free code $\mathcal{C}$ with some codeword lengths $l_1, \ldots, l_n$. Let's assume that $p_1 \geq \cdots \geq p_n$).

Lemma (1)

In an optimal code, $p_i > p_j \Rightarrow l_i \leq l_j$.

Otherwise, swapping i-th and j-th codewords in $\mathcal{C}$ would result in a better code.

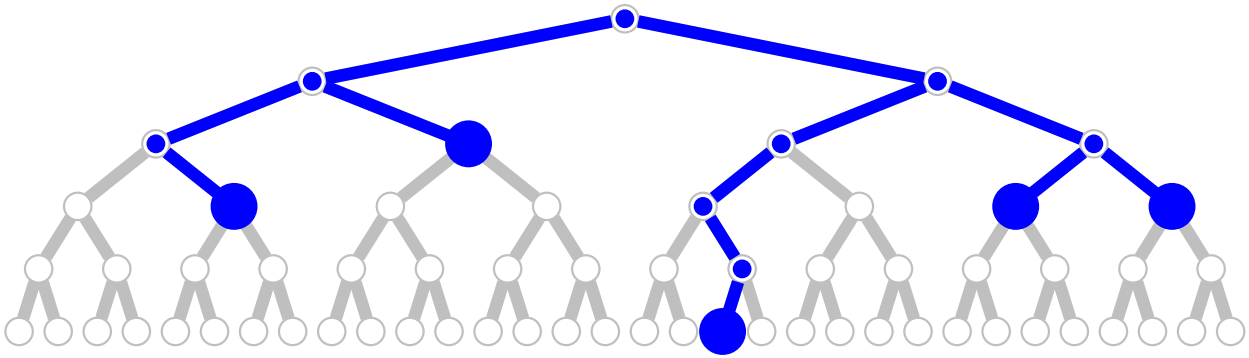
Construction of Optimal Codes (II)

Let $T(\mathcal{C})$ be the subtree of a complete r-ary tree formed by the paths from the root node to the nodes of the codewords in $\mathcal{C}$.

Construction of Optimal Codes (II)

Let $T(\mathcal{C})$ be the subtree of a complete r-ary tree formed by the paths from the root node to the nodes of the codewords in $\mathcal{C}$.

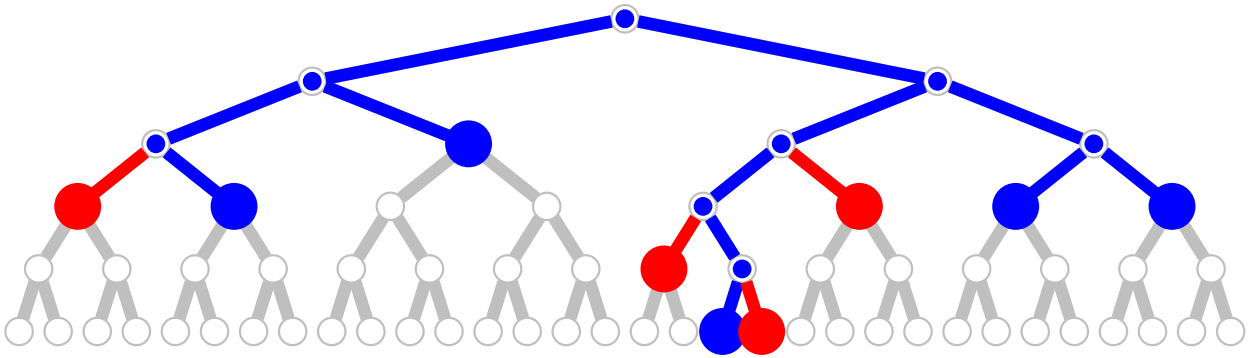
For example, $\mathcal{C} = \{001, 01, 10010, 110, 111\}$



Construction of Optimal Codes (II)

Let $T(\mathcal{C})$ be the subtree of a complete r-ary tree formed by the paths from the root node to the nodes of the codewords in $\mathcal{C}$.

For example, $\mathcal{C} = \{001, 01, 10010, 110, 111\}$

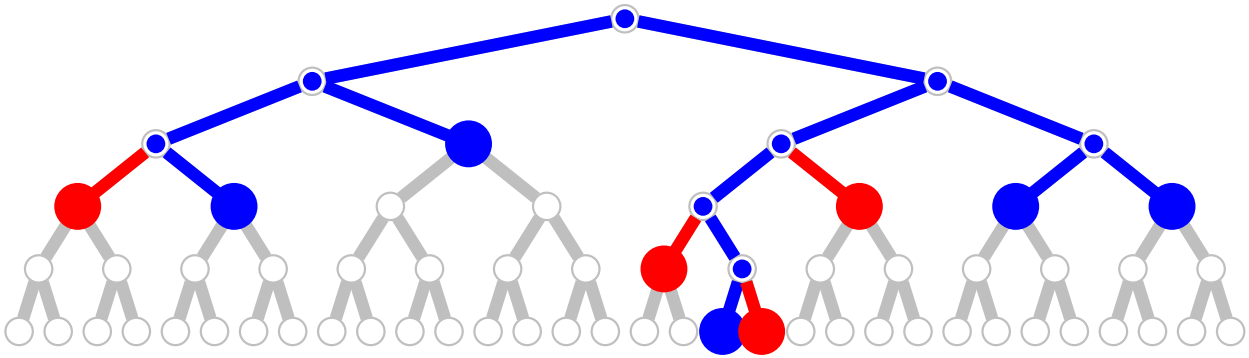


We call **available** the nodes in the complement of $T(\mathcal{C})$ with siblings in $T(\mathcal{C})$.

Construction of Optimal Codes (II)

Let $T(\mathcal{C})$ be the subtree of a complete r-ary tree formed by the paths from the root node to the nodes of the codewords in $\mathcal{C}$.

For example, $\mathcal{C} = \{001, 01, 10010, 110, 111\}$



We call **available** the nodes in the complement of $T(\mathcal{C})$ with siblings in $T(\mathcal{C})$. After adding a new codeword corresponding to an available node, $\mathcal{C}$ remains prefix-free!

Construction of Optimal Codes (III)

Lemma (2)

If $\mathcal{C}$ is optimal, then available nodes of $T(\mathcal{C})$ can only occur in the lowest level.

Otherwise, we can change one codeword in the lowest level by the codeword associated to the available node, then improving the code.

Construction of Optimal Codes (III)

Lemma (2)

If $\mathcal{C}$ is optimal, then available nodes of $T(\mathcal{C})$ can only occur in the lowest level.

Otherwise, we can change one codeword in the lowest level by the codeword associated to the available node, then improving the code.

Lemma (3)

There exists an optimal code $\mathcal{C}$ with $(-n+1) \bmod (r-1)$ available nodes, and all of them are siblings and they are in the lowest level.

In the binary case, $r = 2$, there are no available nodes.

Construction of Optimal Codes (IV)

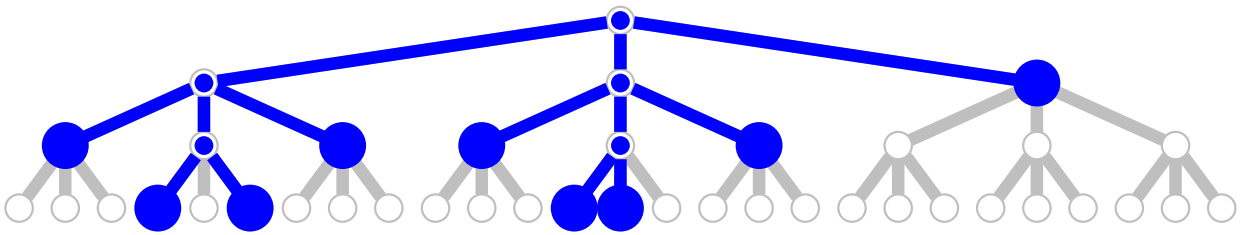
Example: a ternary code

$\mathcal{C} = \{2, 00, 02, 10, 12, 010, 012, 110, 111\}$

Construction of Optimal Codes (IV)

Example: a ternary code

$\mathcal{C} = \{2, 00, 02, 10, 12, 010, 012, 110, 111\}$

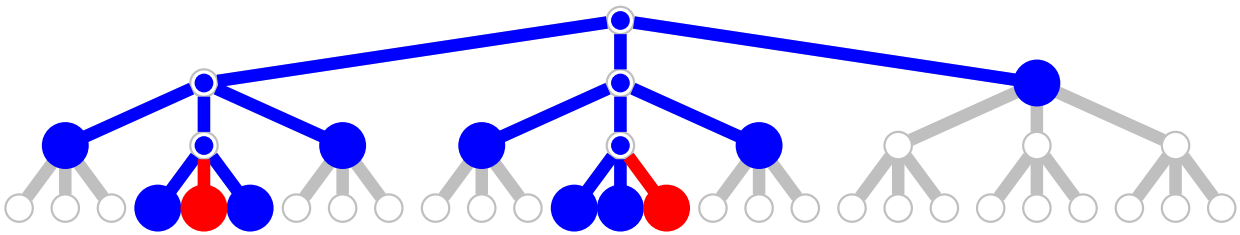


All available nodes are at the lowest level

Construction of Optimal Codes (IV)

Example: a ternary code

$\mathcal{C} = \{2, 00, 02, 10, 12, 010, 012, 110, 111\}$

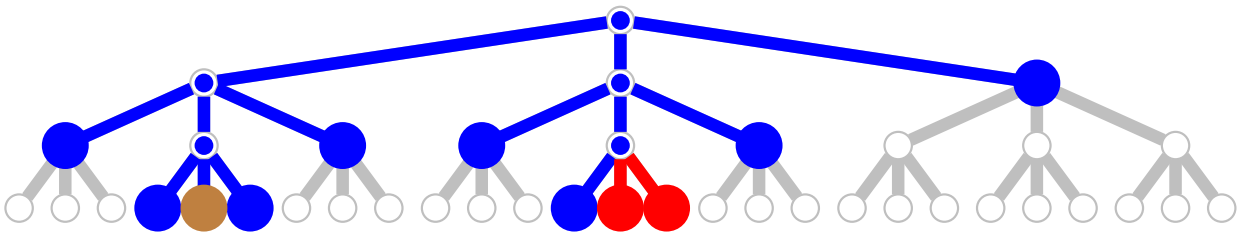


Rearrange lowest level nodes to make all available nodes siblings

Construction of Optimal Codes (IV)

Example: a ternary code

$\mathcal{C} = \{2, 00, 02, 10, 12, 010, 012, 110, 011\}$

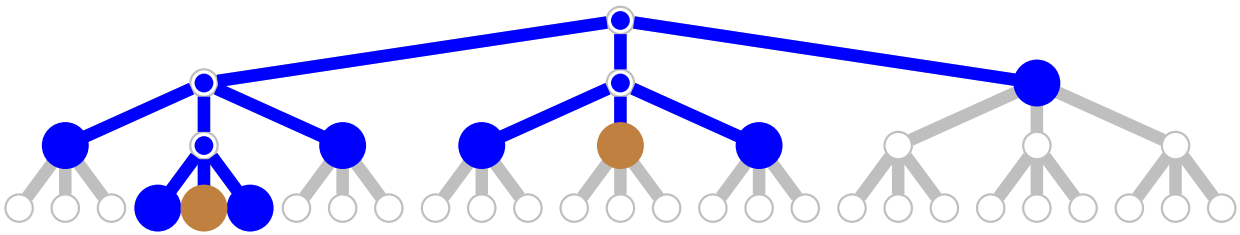


A single child can always be replaced by its parent

Construction of Optimal Codes (IV)

Example: a ternary code

$\mathcal{C} = \{2, 00, 02, 10, 12, 11, 010, 012, 011\}$

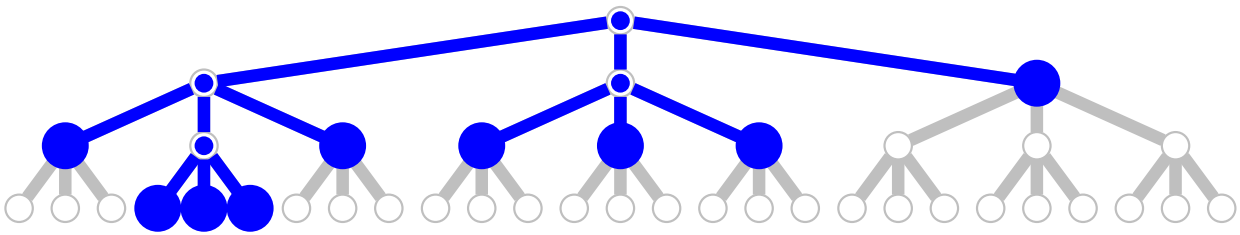


Now the code can be optimal!

Construction of Optimal Codes (IV)

Example: a ternary code

$\mathcal{C} = \{2, 00, 02, 10, 11, 12, \boxed{010, 011, 012}\}$



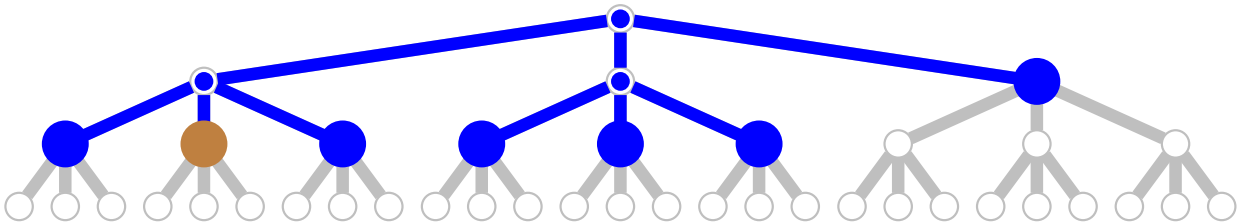
Now the code can be optimal!

Still the lowest level nodes can be rearranged to make the set of $r - (-n+1) \bmod (r-1)$ siblings have the lowest probabilities.

Construction of Optimal Codes (IV)

Example: a ternary code

$\mathcal{C} = \{2, 00, 02, 10, 11, 12, \boxed{01}\}$



Still the lowest level nodes can be rearranged to make the set of $r - (-n + 1) \bmod (r - 1)$ siblings have the lowest probabilities.

If they are replaced by their parent, the code should remain optimal!

Huffman Codes: Algorithm

A recursive construction algorithm:

Input: A sequence of probabilities $p_1 \geq \cdots \geq p_n$
Output: An optimal r-ary code $\{w_1, \ldots, w_n\}$.

- If $n \leq r$, then output $\mathcal{C} = \{0, \ldots, n-1\}$. Otherwise, ...
- Compute $m = r - ((-n+1) \bmod (r-1))$.
- Replace $(p_{n-m+1}, \ldots, p_n)$ by $p'_{n-m+1} = p_{n-m+1} + \ldots + p_n$.
- Sort the new sequence and compute an optimal code $\mathcal{C}'$ for it.
- Undo the above sorting procedure and replace the last codeword w by the m codewords $w0, \ldots, w(m-1)$.

Huffman Codes: Algorithm

A recursive construction algorithm:

Input: A sequence of probabilities $p_1 \geq \cdots \geq p_n$
Output: An optimal r-ary code $\{w_1, \ldots, w_n\}$.

- If $n \leq r$, then output $\mathcal{C} = \{0, \ldots, n-1\}$. Otherwise, ...
- Compute $m = r - ((-n+1) \bmod (r-1))$.
- Replace $(p_{n-m+1}, \ldots, p_n)$ by $p'_{n-m+1} = p_{n-m+1} + \ldots + p_n$.
- Sort the new sequence and compute an optimal code $\mathcal{C}'$ for it.
- Undo the above sorting procedure and replace the last codeword w by the m codewords $w0, \ldots, w(m-1)$.

Huffman Codes: Algorithm

A recursive construction algorithm:

Input: A sequence of probabilities $p_1 \geq \cdots \geq p_n$
Output: An optimal r-ary code $\{w_1, \ldots, w_n\}$.

- If $n \leq r$, then output $\mathcal{C} = \{0, \ldots, n-1\}$. Otherwise, ...
- Compute $m = r - ((-n+1) \bmod (r-1))$.
- Replace $(p_{n-m+1}, \ldots, p_n)$ by $p'_{n-m+1} = p_{n-m+1} + \ldots + p_n$.
- Sort the new sequence and compute an optimal code $\mathcal{C}'$ for it.
- Undo the above sorting procedure and replace the last codeword w by the m codewords $w0, \ldots, w(m-1)$.

Huffman Codes: Algorithm

A recursive construction algorithm:

Input: A sequence of probabilities $p_1 \geq \cdots \geq p_n$
Output: An optimal r-ary code $\{w_1, \ldots, w_n\}$.

- If $n \leq r$, then output $\mathcal{C} = \{0, \ldots, n-1\}$. Otherwise, ...
- Compute $m = r - ((-n+1) \bmod (r-1))$.
- Replace $(p_{n-m+1}, \ldots, p_n)$ by $p'_{n-m+1} = p_{n-m+1} + \ldots + p_n$.
- Sort the new sequence and compute an optimal code $\mathcal{C}'$ for it.
- Undo the above sorting procedure and replace the last codeword w by the m codewords $w0, \ldots, w(m-1)$.

Huffman Codes: Algorithm

A recursive construction algorithm:

Input: A sequence of probabilities $p_1 \geq \cdots \geq p_n$
Output: An optimal r-ary code $\{w_1, \ldots, w_n\}$.

- If $n \leq r$, then output $\mathcal{C} = \{0, \ldots, n-1\}$. Otherwise, ...
- Compute $m = r - ((-n+1) \bmod (r-1))$.
- Replace $(p_{n-m+1}, \ldots, p_n)$ by $p'_{n-m+1} = p_{n-m+1} + \ldots + p_n$.
- Sort the new sequence and compute an optimal code $\mathcal{C}'$ for it.
- Undo the above sorting procedure and replace the last codeword w by the m codewords $w0, \ldots, w(m-1)$.

Huffman Codes: Algorithm

A recursive construction algorithm:

Input: A sequence of probabilities $p_1 \geq \cdots \geq p_n$
Output: An optimal r-ary code $\{w_1, \ldots, w_n\}$.

- If $n \leq r$, then output $\mathcal{C} = \{0, \ldots, n-1\}$. Otherwise, ...
- Compute $m = r - ((-n+1) \bmod (r-1))$.
- Replace $(p_{n-m+1}, \ldots, p_n)$ by $p'_{n-m+1} = p_{n-m+1} + \ldots + p_n$.
- Sort the new sequence and compute an optimal code $\mathcal{C}'$ for it.
- Undo the above sorting procedure and replace the last codeword w by the m codewords $w0, \ldots, w(m-1)$.

Huffman Codes: Algorithm

A recursive construction algorithm:

Input: A sequence of probabilities $p_1 \geq \cdots \geq p_n$
Output: An optimal r-ary code $\{w_1, \ldots, w_n\}$.

- If $n \leq r$, then output $\mathcal{C} = \{0, \ldots, n-1\}$. Otherwise, ...
- Compute $m = r - ((-n+1) \bmod (r-1))$.
- Replace $(p_{n-m+1}, \ldots, p_n)$ by $p'_{n-m+1} = p_{n-m+1} + \ldots + p_n$.
- Sort the new sequence and compute an optimal code $\mathcal{C}'$ for it.
- Undo the above sorting procedure and replace the last codeword w by the m codewords $w0, \ldots, w(m-1)$.

Remark: $m = r$ except perhaps in the first step for $r \geq 3$

Huffman Codes: Example (I)

Example: $r = 3,\ n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

$m = 2$, Sort:

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

$m = 3$, Sort:

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228}, 0.0130)$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228, 0.0130})$

$m = 3$, Sort:

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769, 0.0520, 0.0510})$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228}, 0.0130)$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769}, 0.0520, 0.0510)$

$m = 3$, Sort:

[4] $(0.4355, 0.1821, \underbrace{0.1799}, 0.1161, 0.0864)$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228, 0.0130})$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769, 0.0520, 0.0510})$

[4] $(0.4355, 0.1821, \underbrace{0.1799, 0.1161, 0.0864})$

$n = 3 \leq r$, Sort: (not strictly necessary here)

[5] $(0.4355, 0.3824, 0.1821)$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228, 0.0130})$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769, 0.0520, 0.0510})$

[4] $(0.4355, 0.1821, \underbrace{0.1799, 0.1161, 0.0864})$

[5] $(0.4355, 0.3824, 0.1821)$

Assign codewords:

[5] $(0, 1, 2)$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228}, 0.0130)$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769}, 0.0520, 0.0510)$

[4] $(0.4355, 0.1821, \underbrace{0.1799}, 0.1161, 0.0864)$

[5] $(0.4355, 0.3824, 0.1821)$

[5] $(0, 1, 2)$

Unsort and expand codewords:

[4] $(0, 2, \overbrace{10}, 11, 12)$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228, 0.0130})$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769, 0.0520, 0.0510})$

[4] $(0.4355, 0.1821, \underbrace{0.1799, 0.1161, 0.0864})$

[5] $(0.4355, 0.3824, 0.1821)$

[5] $(0, 1, 2)$

[4] $(0, 2, \overbrace{10, 11, 12})$

Unsort and expand codewords:

[3] $(0, 2, 11, 12, \overbrace{100, 101, 102})$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228}, 0.0130)$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769}, 0.0520, 0.0510)$

[4] $(0.4355, 0.1821, \underbrace{0.1799}, 0.1161, 0.0864)$

[5] $(0.4355, 0.3824, 0.1821)$

[5] $(0, 1, 2)$

[4] $(0, 2, \overbrace{10, 11, 12})$

[3] $(0, 2, 11, 12, \overbrace{100, 101, 102})$

Unsort and expand codewords:

[2] $(0, 2, 11, 12, 101, 102, \overbrace{1000, 1001, 1002})$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228, 0.0130})$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769, 0.0520, 0.0510})$

[4] $(0.4355, 0.1821, \underbrace{0.1799, 0.1161, 0.0864})$

[5] $(0.4355, 0.3824, 0.1821)$

[5] $(0, 1, 2)$

[4] $(0, 2, \overbrace{10, 11, 12})$

[3] $(0, 2, 11, 12, \overbrace{100, 101, 102})$

[2] $(0, 2, 11, 12, 101, 102, \overbrace{1000, 1001, 1002})$

Unsort and expand codewords:

[1] $(0, 2, 11, 12, 101, 102, 1000, 1002, \overbrace{10010, 10011})$

Huffman Codes: Example (I)

Example: $r = 3$, $n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228}, 0.0130)$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769}, 0.0520, 0.0510)$

[4] $(0.4355, 0.1821, \underbrace{0.1799}, 0.1161, 0.0864)$

[5] $(0.4355, 0.3824, 0.1821)$

[5] $(0, 1, 2)$

[4] $(0, 2, \overbrace{10, 11, 12})$

[3] $(0, 2, 11, 12, \overbrace{100, 101, 102})$

[2] $(0, 2, 11, 12, 101, 102, \overbrace{1000, 1001, 1002})$

[1] $(0, 2, 11, 12, 101, 102, 1000, 1002, \overbrace{10010, 10011})$

Finally, unsort:

$\mathcal{C} = (2, 11, 102, 1002, 10010, 0, 12, 1000, 10011, 101)$

Huffman Codes: Example (I)

Example: $r = 3,\ n = 10$

$\mathcal{S} = (0.1821, 0.1161, 0.0510, 0.0130, 0.0124, 0.4355, 0.0864, 0.0411, 0.0104, 0.0520)$

[1] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, 0.0411, 0.0130, \underbrace{0.0124, 0.0104})$

[2] $(0.4355, 0.1821, 0.1161, 0.0864, 0.0520, 0.0510, \underbrace{0.0411, 0.0228}, 0.0130)$

[3] $(0.4355, 0.1821, 0.1161, 0.0864, \underbrace{0.0769}, 0.0520, 0.0510)$

[4] $(0.4355, 0.1821, \underbrace{0.1799, 0.1161, 0.0864})$

[5] $(0.4355, 0.3824, 0.1821)$

[5] $(0, 1, 2)$

[4] $(0, 2, \overbrace{10, 11, 12})$

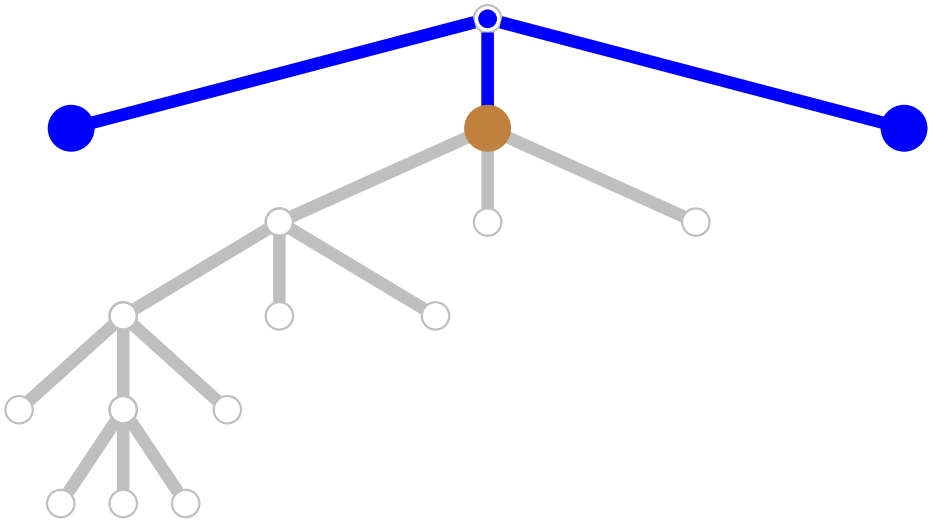
[3] $(0, 2, 11, 12, \overbrace{100, 101, 102})$

[2] $(0, 2, 11, 12, 101, 102, \overbrace{1000, 1001, 1002})$

[1] $(0, 2, 11, 12, 101, 102, 1000, 1002, \overbrace{10010, 10011})$

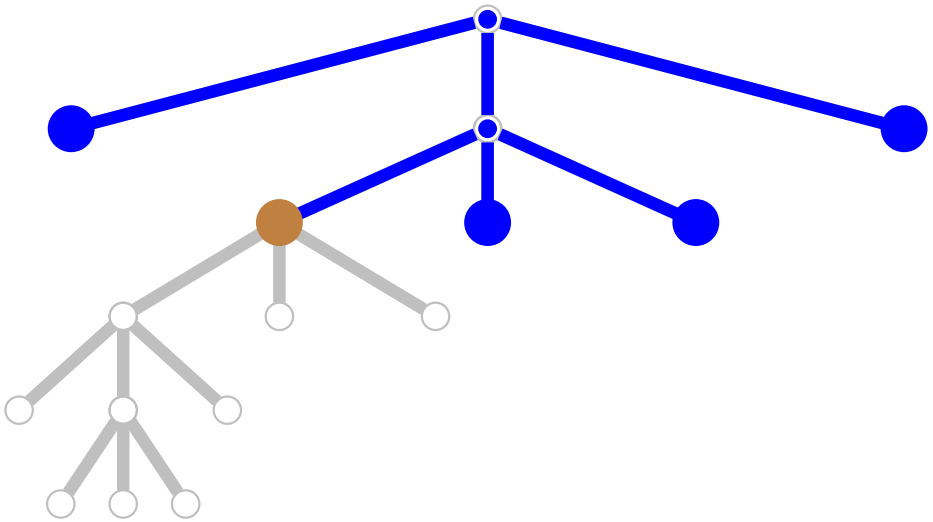
$\mathcal{C} = (2, 11, 102, 1002, 10010, 0, 12, 1000, 10011, 101)$

Huffman Codes: Example (II)



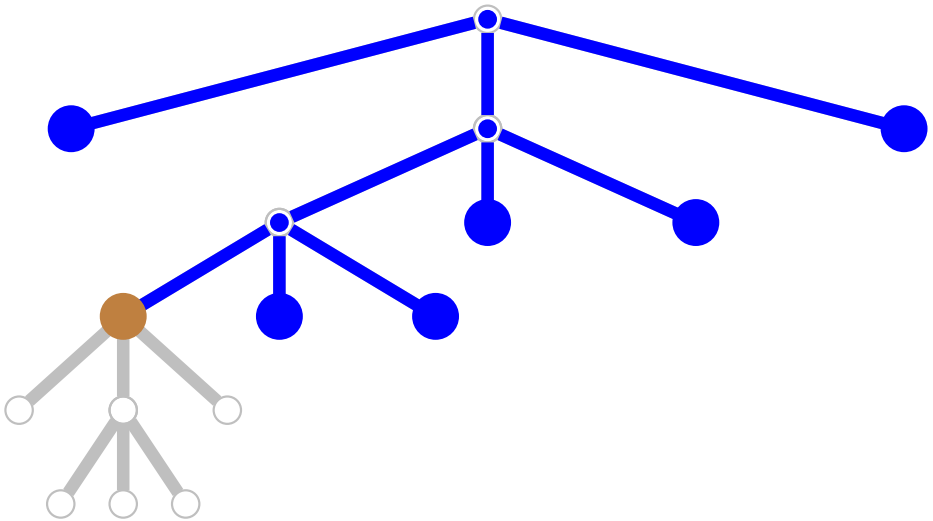
$\mathcal{C}_5 = \{0, 1, 2\}$

Huffman Codes: Example (II)



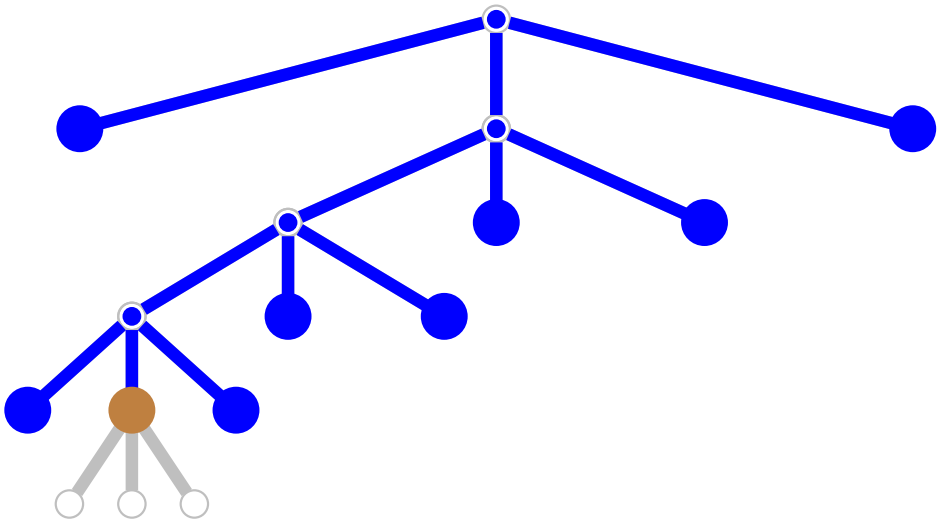
$$\mathcal{C}_4 = \{0, 2, \overbrace{10, 11, 12}\}$$

Huffman Codes: Example (II)



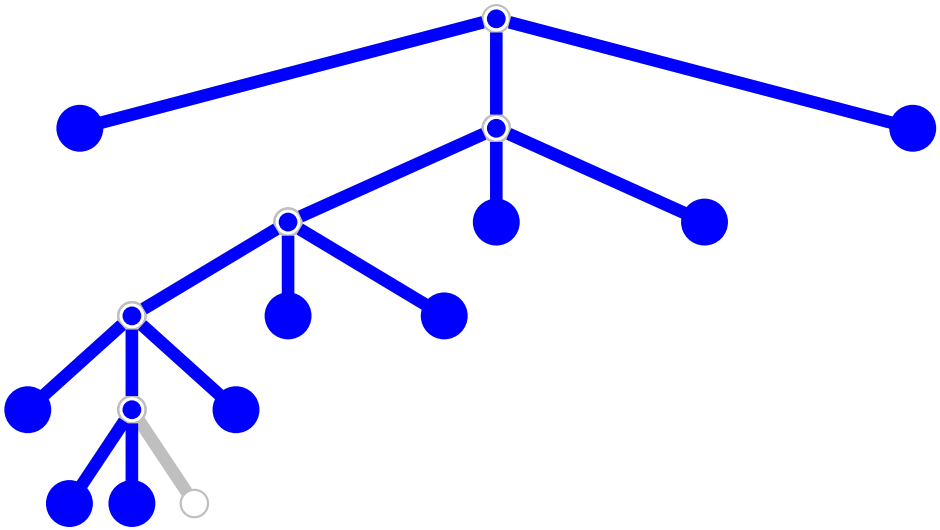
$$\mathcal{C}_3 = \{0, 2, 11, 12, \overbrace{100, 101, 102}\}$$

Huffman Codes: Example (II)



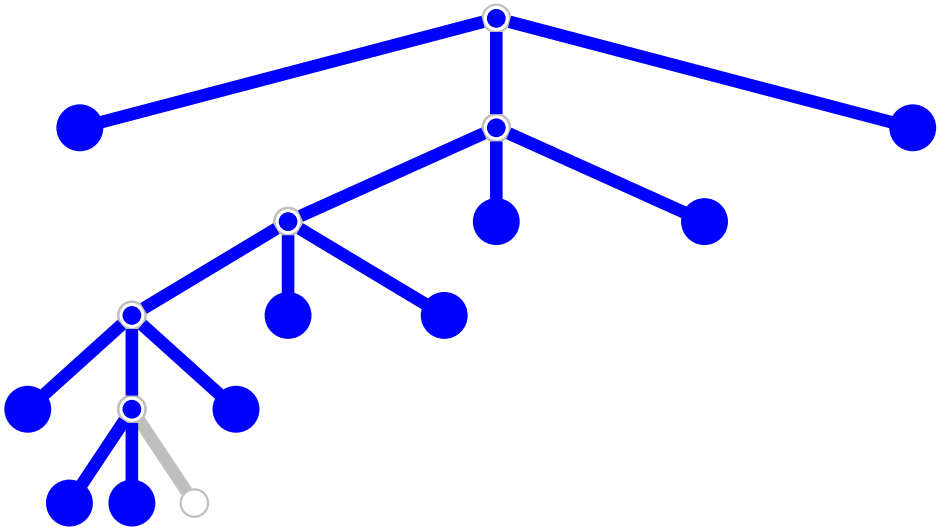
$$\mathcal{C}_2 = \{0, 2, 11, 12, 101, 102, \overbrace{1000, 1001, 1002}\}$$

Huffman Codes: Example (II)



$$\mathcal{C}_1 = \{0, 2, 11, 12, 101, 102, 1000, 1002, \overbrace{10010, 10011}\}$$

Huffman Codes: Example (II)

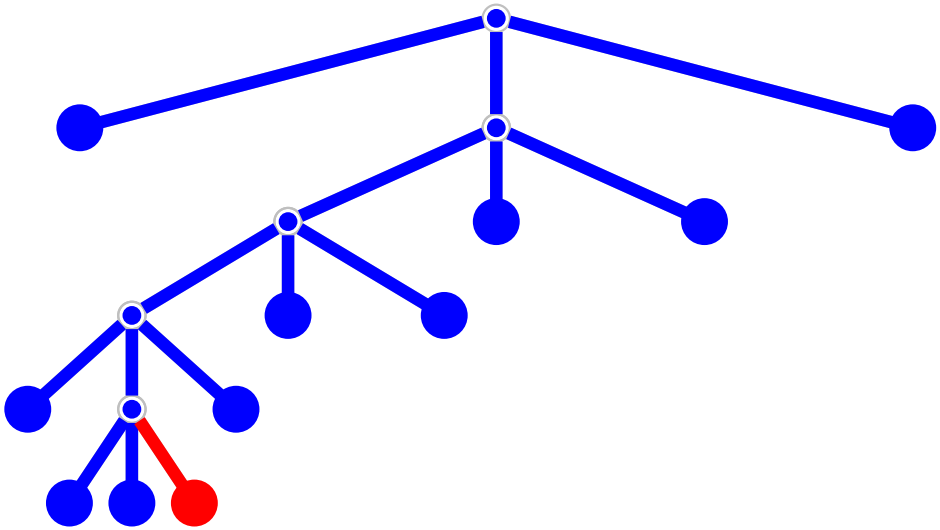


$\mathcal{C} = \{2, 11, 102, 1002, 10010, 0, 12, 1000, 10011, 101\}$

$\mathcal{C}_{\text{Sh-F}} = \{10, 11, 120, 2010, 2011, 0, 121, 122, 20120, 200\}$

$L(\mathcal{C}, \mathcal{S}) = 1.6620 \qquad L_r(\mathcal{S}) = 1.5736 \qquad L(\mathcal{C}_{\text{Sh-F}}, \mathcal{S}) = 1.8770$

Huffman Codes: Example (II)



$\mathcal{C} = \{2, 11, 102, 1002, 10010, 0, 12, 1000, 10011, 101\}$

$\mathcal{C}_{\text{Sh-F}} = \{10, 11, 120, 2010, 2011, 0, 121, 122, 20120, 200\}$

$L(\mathcal{C}, \mathcal{S}) = 1.6620 \qquad L_r(\mathcal{S}) = 1.5736 \qquad L(\mathcal{C}_{\text{Sh-F}}, \mathcal{S}) = 1.8770$

There is still one available node

Outline

Shannon's Entropy of a Discrete Random Variable (I)

It measures the average amount of information in a random variable.

Definition (Shannon's Entropy)

Let X be a random variable with range a finite set $\mathcal{X}$.

$$H(X) = -\sum_{x \in \mathcal{X}} \Pr[X = x] \log_2 \Pr[X = x]$$

Shannon's Entropy of a Discrete Random Variable (I)

It measures the average amount of information in a random variable.

Definition (Shannon's Entropy)

Let X be a random variable with range a finite set $\mathcal{X}$.

$$H(X) = -\sum_{x \in \mathcal{X}} \Pr[X = x] \log_2 \Pr[X = x]$$

Relationship with optimal codes:

$$L_r(\mathcal{S}) = H(\mathcal{S}) / \log_2 r$$

"The minimum possible average length for a lossless code for a source $\mathcal{S}$ is actually the average amount of (r-ary) information per source symbol."

Shannon's Entropy of a Discrete Random Variable (II)

Properties of Shannon's entropy:

- **Main inequality:** $H(X,Y,Z) + H(Z) \leq H(X,Z) + H(Y,Z)$ (equality holds only if $\forall z \in \mathcal{Z}$, conditioned to $Z = z$, X and Y are independent) ▸ details...
- $0 \leq H(X) \leq \log_2 |\mathcal{X}|$ (r.h.s. equality holds only for uniform X, and l.h.s. holds for a trivial X)
- $H(X) \leq H(X,Y) \leq H(X) + H(Y)$ (r.h.s. equality holds only for independent X, Y, and l.h.s. holds only if $Y = g(X)$)
- $H(g(X)) \leq H(X)$ (equality holds only for injective g)
- $H(X, g(X,Y)) \leq H(X,Y)$ (equality holds only if $y \mapsto g(x,y)$ is injective $\forall x \in \mathcal{X}$)
- $H(X, g(X)) = H(X)$

Shannon's Entropy of a Discrete Random Variable (II)

Properties of Shannon's entropy:

- **Main inequality:** $H(X, Y, Z) + H(Z) \leq H(X, Z) + H(Y, Z)$ (equality holds only if $\forall z \in \mathcal{Z}$, conditioned to $Z = z$, X and Y are independent) ▸ details...
- $0 \leq H(X) \leq \log_2 |\mathcal{X}|$ (r.h.s. equality holds only for uniform X, and l.h.s. holds for a trivial X)
- $H(X) \leq H(X, Y) \leq H(X) + H(Y)$ (r.h.s. equality holds only for independent X, Y, and l.h.s. holds only if $Y = g(X)$)
- $H(g(X)) \leq H(X)$ (equality holds only for injective g)
- $H(X, g(X, Y)) \leq H(X, Y)$ (equality holds only if $y \mapsto g(x, y)$ is injective $\forall x \in \mathcal{X}$)
- $H(X, g(X)) = H(X)$

Shannon's Entropy of a Discrete Random Variable (II)

Properties of Shannon's entropy:

- **Main inequality:** $H(X,Y,Z)+H(Z) \leq H(X,Z)+H(Y,Z)$ (equality holds only if $\forall z \in \mathcal{Z}$, conditioned to $Z=z$, X and Y are independent) ▸ details...
- $0 \leq H(X) \leq \log_2 |\mathcal{X}|$ (r.h.s. equality holds only for uniform X, and l.h.s. holds for a trivial X)
- $H(X) \leq H(X,Y) \leq H(X)+H(Y)$ (r.h.s. equality holds only for independent X, Y, and l.h.s. holds only if $Y=g(X)$)
- $H(g(X)) \leq H(X)$ (equality holds only for injective g)
- $H(X,g(X,Y)) \leq H(X,Y)$ (equality holds only if $y \mapsto g(x,y)$ is injective $\forall x \in \mathcal{X}$)
- $H(X,g(X)) = H(X)$

Shannon's Entropy of a Discrete Random Variable (II)

Properties of Shannon's entropy:

- **Main inequality:** $H(X, Y, Z) + H(Z) \leq H(X, Z) + H(Y, Z)$ (equality holds only if $\forall z \in \mathcal{Z}$, conditioned to $Z = z$, X and Y are independent) ▸ details...
- $0 \leq H(X) \leq \log_2 |\mathcal{X}|$ (r.h.s. equality holds only for uniform X, and l.h.s. holds for a trivial X)
- $H(X) \leq H(X, Y) \leq H(X) + H(Y)$ (r.h.s. equality holds only for independent X, Y, and l.h.s. holds only if $Y = g(X)$)
- $H(g(X)) \leq H(X)$ (equality holds only for injective g)
- $H(X, g(X, Y)) \leq H(X, Y)$ (equality holds only if $y \mapsto g(x, y)$ is injective $\forall x \in \mathcal{X}$)
- $H(X, g(X)) = H(X)$

Shannon's Entropy of a Discrete Random Variable (II)

Properties of Shannon's entropy:

- **Main inequality:** $H(X, Y, Z) + H(Z) \leq H(X, Z) + H(Y, Z)$ (equality holds only if $\forall z \in \mathcal{Z}$, conditioned to $Z = z$, X and Y are independent) ▸ details...
- $0 \leq H(X) \leq \log_2 |\mathcal{X}|$ (r.h.s. equality holds only for uniform X, and l.h.s. holds for a trivial X)
- $H(X) \leq H(X, Y) \leq H(X) + H(Y)$ (r.h.s. equality holds only for independent X, Y, and l.h.s. holds only if $Y = g(X)$)
- $H(g(X)) \leq H(X)$ (equality holds only for injective g)
- $H(X, g(X, Y)) \leq H(X, Y)$ (equality holds only if $y \mapsto g(x, y)$ is injective $\forall x \in \mathcal{X}$)
- $H(X, g(X)) = H(X)$

Shannon's Entropy of a Discrete Random Variable (II)

Properties of Shannon's entropy:

- **Main inequality:** $H(X,Y,Z)+H(Z) \leq H(X,Z)+H(Y,Z)$ (equality holds only if $\forall z \in \mathcal{Z}$, conditioned to $Z=z$, X and Y are independent) ▸ details...
- $0 \leq H(X) \leq \log_2 |\mathcal{X}|$ (r.h.s. equality holds only for uniform X, and l.h.s. holds for a trivial X)
- $H(X) \leq H(X,Y) \leq H(X)+H(Y)$ (r.h.s. equality holds only for independent X, Y, and l.h.s. holds only if $Y=g(X)$)
- $H(g(X)) \leq H(X)$ (equality holds only for injective g)
- $H(X,g(X,Y)) \leq H(X,Y)$ (equality holds only if $y \mapsto g(x,y)$ is injective $\forall x \in \mathcal{X}$)
- $H(X,g(X)) = H(X)$

Conditional Entropy

Definition (Conditional Shannon's Entropy)

Let X, Y be random variables with respective ranges $\mathcal{X}, \mathcal{Y}$.
$H(Y \mid X) = H(X, Y) - H(X)$

It measures the average amount of information still in Y after revealing X.

Conditional Entropy

Definition (Conditional Shannon's Entropy)

Let X, Y be random variables with respective ranges $\mathcal{X}, \mathcal{Y}$.
$H(Y \mid X) = H(X, Y) - H(X)$

It measures the average amount of information still in Y after revealing X.

Equivalently, $H(Y \mid X) = \sum_{x \in \mathcal{X}} \Pr[X = x] H(Y \mid X = x)$ or

$$H(Y \mid X) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \Pr[X = x, Y = y] \log_2 \Pr[Y = y \mid X = x]$$

Conditional Entropy: Properties

Similarly to (non-conditional) Shannon's entropy:

- **Main inequality:**
  $H(Y, Z, T \mid X) + H(T \mid X) \leq H(Y, T \mid X) + H(Z, T \mid X)$
- $0 \leq H(Y \mid X) \leq H(Y)$ (r.h.s. equality holds only for independent X, Y, and l.h.s. holds only if $Y = g(X)$)
- $H(Z \mid X, Y) = H(Y, Z \mid X) - H(Y \mid X)$
- $H(Y \mid X) \leq H(Y, Z \mid X) \leq H(Y \mid X) + H(Z \mid X)$
- $H(g(X, Y) \mid X) \leq H(Y \mid X)$
- $H(Y, g(X, Y, Z) \mid X) \leq H(Y, Z \mid X)$
- $H(Y, g(X, Y) \mid X) = H(Y \mid X)$

Mutual Information

Definition (Mutual Information)

Let X, Y be random variables with respective ranges $\mathcal{X}, \mathcal{Y}$.
$I(Y;X) = H(X) + H(Y) - H(X,Y) = H(Y) - H(Y \mid X)$

It measures the amount of information about Y conveyed by X.

Mutual Information

Definition (Mutual Information)

Let X, Y be random variables with respective ranges $\mathcal{X}, \mathcal{Y}$.
$I(Y;X) = H(X) + H(Y) - H(X,Y) = H(Y) - H(Y \mid X)$

It measures the amount of information about Y conveyed by X.

Some properties

- $I(Y,Z,T;X) + I(T;X) \leq I(Y,T;X) + I(Z,T;X)$
- $I(X;Y) = I(Y;X)$
- $0 \leq I(Y;X) \leq \min(H(X), H(Y))$

Mutual Information

Definition (Mutual Information)

Let X, Y be random variables with respective ranges $\mathcal{X}, \mathcal{Y}$.
$I(Y;X) = H(X) + H(Y) - H(X,Y) = H(Y) - H(Y \mid X)$

It measures the amount of information about Y conveyed by X.

Some properties

- $I(Y,Z,T;X) + I(T;X) \leq I(Y,T;X) + I(Z,T;X)$
- $I(X;Y) = I(Y;X)$
- $0 \leq I(Y;X) \leq \min(H(X), H(Y))$

Definition (Conditional Mutual Information)

Let X, Y, Z be random variables with respective ranges $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$.
$I(Z;Y \mid X) = H(Z \mid X) - H(Z \mid X,Y)$

Codes and Cryptography

Jorge L. Villar

MAMME, Fall 2015

END OF PART III

Shannon's Entropy Main Inequality (Submodularity)

The map $f(t) = t \log_2 t$ is convex in $[0, +\infty)$ (because $f'' > 0$)

By discrete Jensen's inequality, for all $a_i > 0$, $t_i > 0$, $i \in \mathcal{I}$

$$\sum_{i \in \mathcal{I}} a_i t_i = 1 \quad \Rightarrow \quad \sum_{i \in \mathcal{I}} a_i t_i \log_2 t_i \geq -\log_2 \sum_{i \in \mathcal{I}} a_i$$

and the equality implies that all t_i are equal.

The main inequality is proven by letting

$$t_{x,y,z} = \frac{\Pr[X = x, Y = y, Z = z] \Pr[Z = z]}{\Pr[X = x, Z = z] \Pr[Y = y, Z = z]}; \qquad \sum_{x,y,z} a_{x,y,z} t_{x,y,z} = 1$$

$$a_{x,y,z} = \frac{\Pr[X = x, Z = z] \Pr[Y = y, Z = z]}{\Pr[Z = z]}; \qquad \sum_{x,y,z} a_{x,y,z} = 1$$

go back...

Shannon's Entropy Main Inequality (Submodularity)

The map $f(t) = t \log_2 t$ is convex in $[0, +\infty)$ (because $f'' > 0$)

By discrete Jensen's inequality, for all $a_i > 0$, $t_i > 0$, $i \in \mathcal{I}$

$$\sum_{i \in \mathcal{I}} a_i t_i = 1 \quad \Rightarrow \quad \sum_{i \in \mathcal{I}} a_i t_i \log_2 t_i \geq -\log_2 \sum_{i \in \mathcal{I}} a_i$$

and the equality implies that all t_i are equal.

The main inequality is proven by letting

$$t_{x,y,z} = \frac{\Pr[X = x, Y = y, Z = z] \Pr[Z = z]}{\Pr[X = x, Z = z] \Pr[Y = y, Z = z]}; \qquad \sum_{x,y,z} a_{x,y,z} t_{x,y,z} = 1$$

$$a_{x,y,z} = \frac{\Pr[X = x, Z = z] \Pr[Y = y, Z = z]}{\Pr[Z = z]}; \qquad \sum_{x,y,z} a_{x,y,z} = 1$$

go back...

Shannon's Entropy Main Inequality (Submodularity)

The map $f(t) = t \log_2 t$ is convex in $[0, +\infty)$ (because $f'' > 0$)

By discrete Jensen's inequality, for all $a_i > 0$, $t_i > 0$, $i \in \mathcal{I}$

$$\sum_{i \in \mathcal{I}} a_i t_i = 1 \quad \Rightarrow \quad \sum_{i \in \mathcal{I}} a_i t_i \log_2 t_i \geq -\log_2 \sum_{i \in \mathcal{I}} a_i$$

and the equality implies that all t_i are equal.

The main inequality is proven by letting

$$t_{x,y,z} = \frac{\Pr[X = x, Y = y, Z = z] \Pr[Z = z]}{\Pr[X = x, Z = z] \Pr[Y = y, Z = z]};$$

$$a_{x,y,z} = \frac{\Pr[X = x, Z = z] \Pr[Y = y, Z = z]}{\Pr[Z = z]};$$

$$\sum_{x,y,z} a_{x,y,z} t_{x,y,z} = 1$$

$$\sum_{x,y,z} a_{x,y,z} = 1$$

go back...