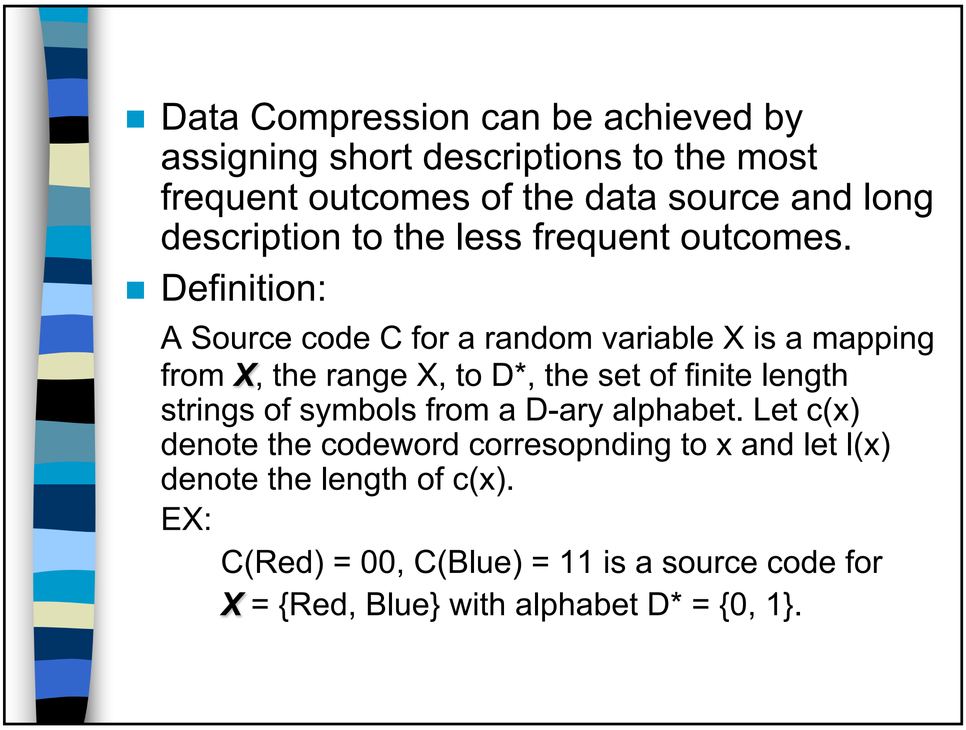


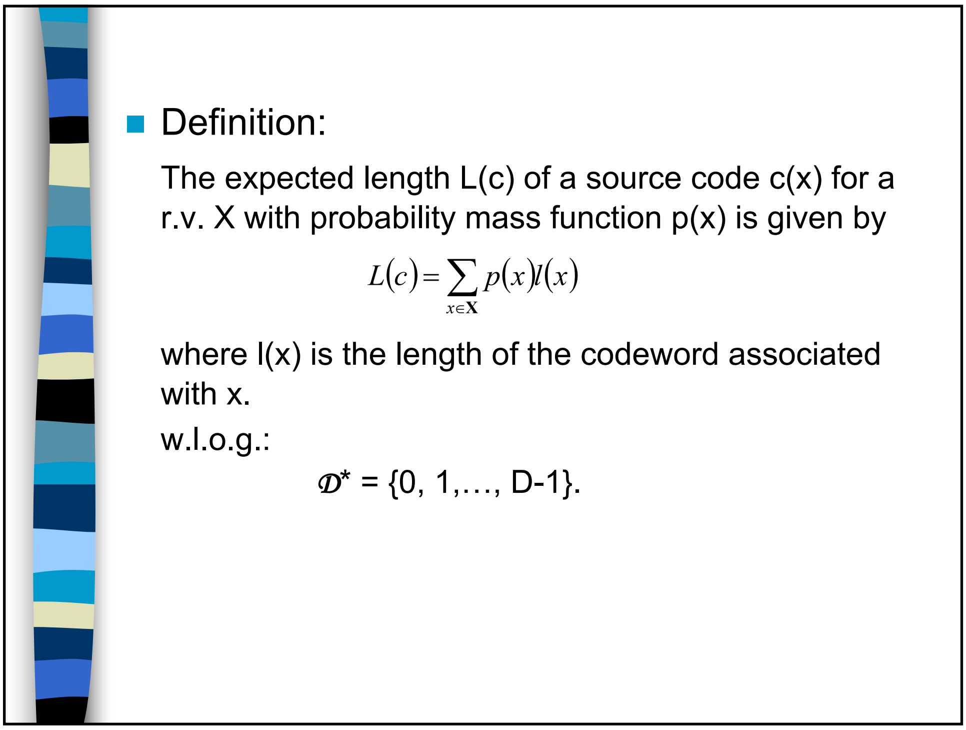
- Data Compression can be achieved by assigning short descriptions to the most frequent outcomes of the data source and long description to the less frequent outcomes.
- Definition:

A Source code C for a random variable X is a mapping from ***X***, the range X, to D*, the set of finite length strings of symbols from a D-ary alphabet. Let c(x) denote the codeword corresopnding to x and let l(x) denote the length of c(x).

EX:

C(Red) = 00, C(Blue) = 11 is a source code for ***X*** = {Red, Blue} with alphabet D* = {0, 1}.

1



- Definition:

The expected length L(c) of a source code c(x) for a r.v. X with probability mass function p(x) is given by

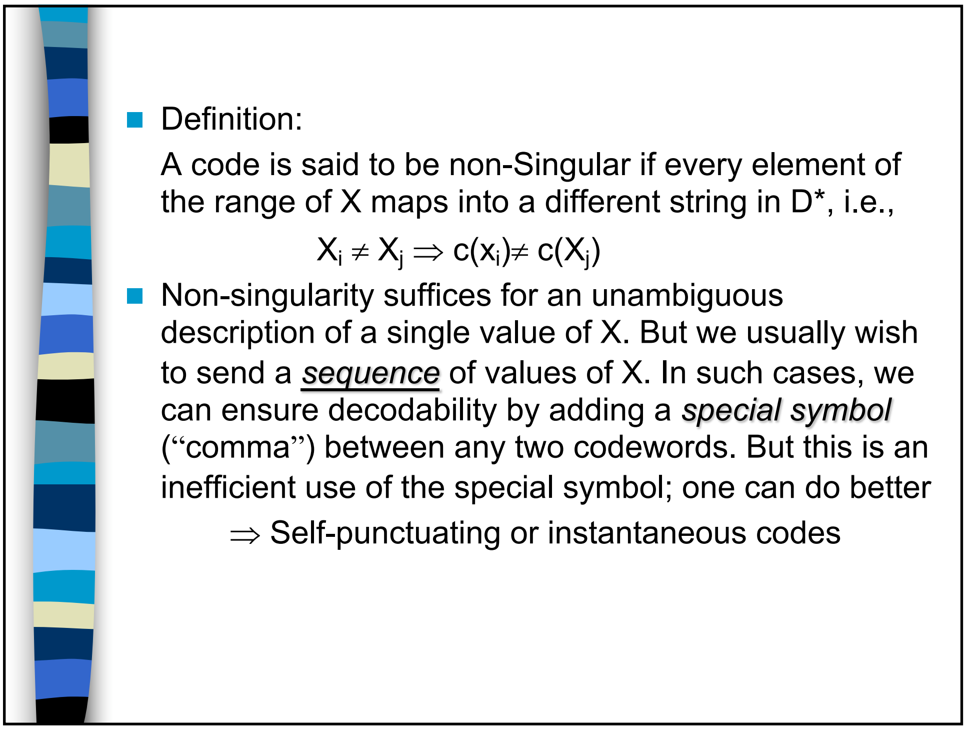
$$L(c)=\sum_{x\in \mathbf{X}} p(x)l(x)$$

where l(x) is the length of the codeword associated with x.

w.l.o.g.:

$\mathcal{D}^*$ = {0, 1,…, D-1}.

2



- Definition:

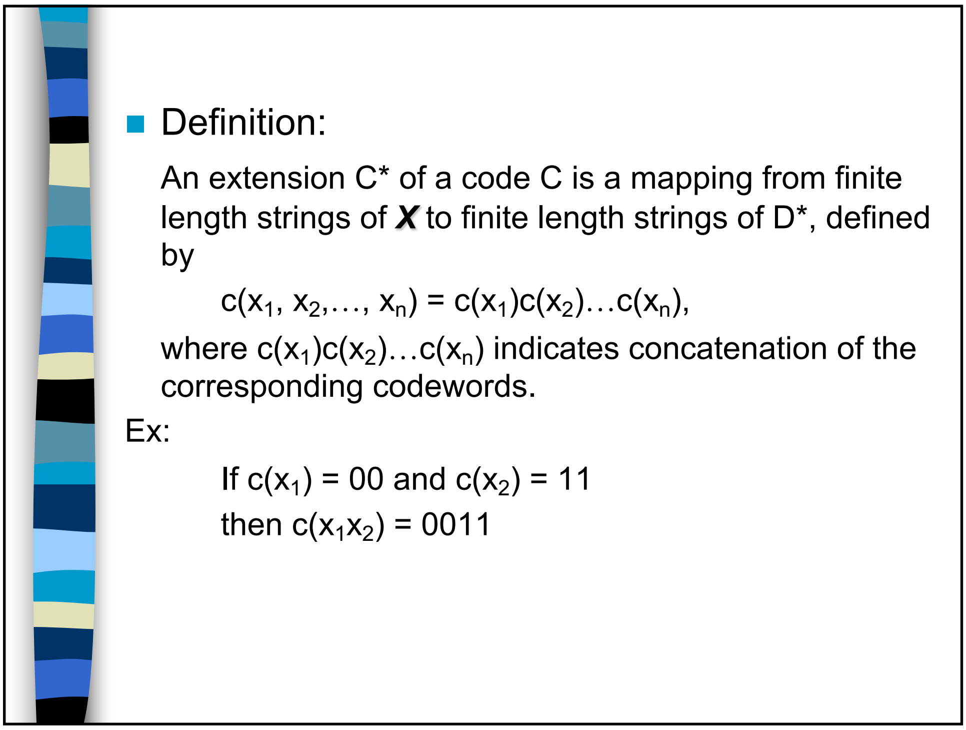
  A code is said to be non-Singular if every element of the range of X maps into a different string in D*, i.e.,

  $$X_i \neq X_j \Rightarrow c(x_i) \neq c(X_j)$$

- Non-singularity suffices for an unambiguous description of a single value of X. But we usually wish to send a *sequence* of values of X. In such cases, we can ensure decodability by adding a *special symbol* ("comma") between any two codewords. But this is an inefficient use of the special symbol; one can do better

  ⇒ Self-punctuating or instantaneous codes

3



- Definition:

  An extension C* of a code C is a mapping from finite length strings of ***X*** to finite length strings of D*, defined by

  $$c(x_1, x_2,\ldots, x_n) = c(x_1)c(x_2)\ldots c(x_n),$$

  where $c(x_1)c(x_2)\ldots c(x_n)$ indicates concatenation of the corresponding codewords.

Ex:

If $c(x_1) = 00$ and $c(x_2) = 11$

then $c(x_1x_2) = 0011$

4

- Definition:

  A code is called uniquely decodable if its extension is non-singular.

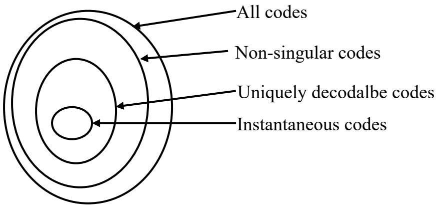
  In other words, any encoded string in a uniquely decodable code has only one possible source string producing it.

- Definition:

  A coded is called a prefix code or an instantaneous code if no codeword is a prefix of any other codeword.

  An instantaneous code can be decoded without reference to the future codewords since the end of a codeword is immediately recognizable. →

  For an instantaneous code, the symbol xi can be decoded as soon as we come to the end of the codeword corresponding to it. → self-punctuating



All codes

Non-singular codes

Uniquely decodalbe codes

Instantaneous codes

5

KRAFT INEQUALITY:

- Goal of source coding:

  constructing instantaneous codes of minimum expected length to describe a given source.

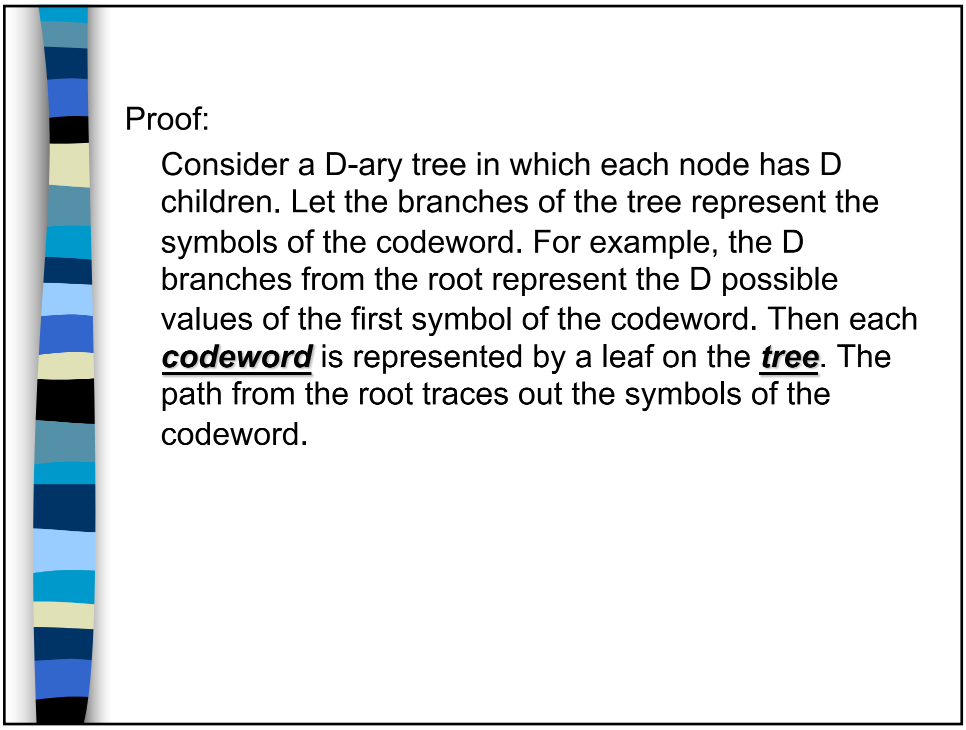
- **Theorem** (Kraft inequality)

  For any instantaneous code (prefix code) over an alphabet of size D, the codeword lengths $l_1, l_2, \dots, l_m$ must satisfy the inequality

$$\sum_i D^{-l_i} \leq 1$$

  Conversely, given a set of codeword lengths that satisfy this inequality, there exists an instantaneous code with these word lengths.

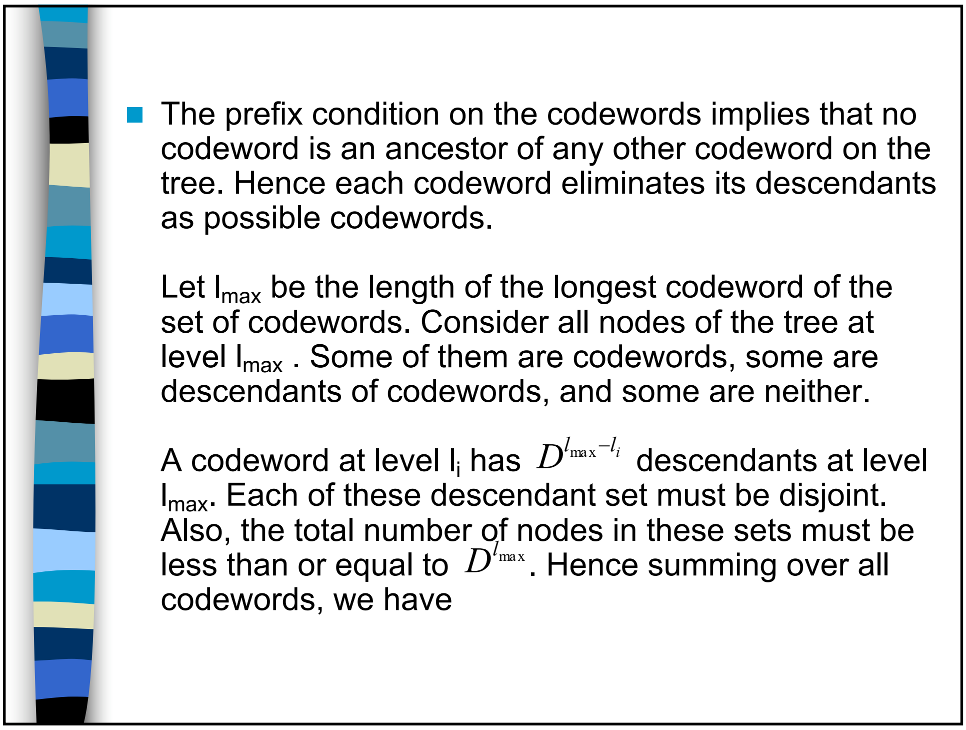
6



Proof:

Consider a D-ary tree in which each node has D children. Let the branches of the tree represent the symbols of the codeword. For example, the D branches from the root represent the D possible values of the first symbol of the codeword. Then each ***codeword*** is represented by a leaf on the ***tree***. The path from the root traces out the symbols of the codeword.

7



- The prefix condition on the codewords implies that no codeword is an ancestor of any other codeword on the tree. Hence each codeword eliminates its descendants as possible codewords.

  Let l_{max} be the length of the longest codeword of the set of codewords. Consider all nodes of the tree at level l_{max} . Some of them are codewords, some are descendants of codewords, and some are neither.

  A codeword at level l_i has $D^{l_{\max}-l_i}$ descendants at level l_{max}. Each of these descendant set must be disjoint. Also, the total number of nodes in these sets must be less than or equal to $D^{l_{\max}}$. Hence summing over all codewords, we have

8

$$\sum D^{l_{\max}-l_i} \le \sum D^{l_{\max}}$$

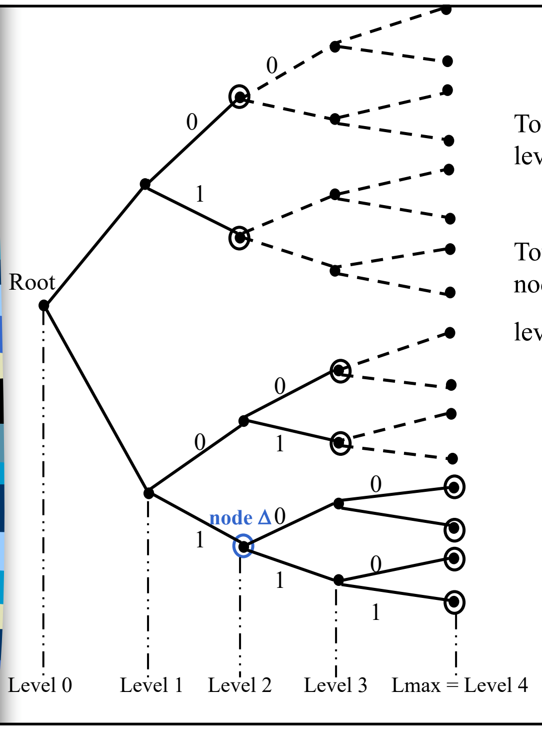
or

$$\sum D^{-l_i} \le 1$$

which is the Kraft inequality.

- Conversely, given any set of codewords l_1, l_2, …, l_m which satisfy the Kraft inquality, we can always construct a D-ary tree. Label the first node (lexicographically) of depth l_1, as codeword 1, and move its descendants from the tree. Then label the first remaining node of depth l_2 as codeword 2, etc. Processing this way, we construct a prefix code with the specified l_1, l_2, …, l_m

9

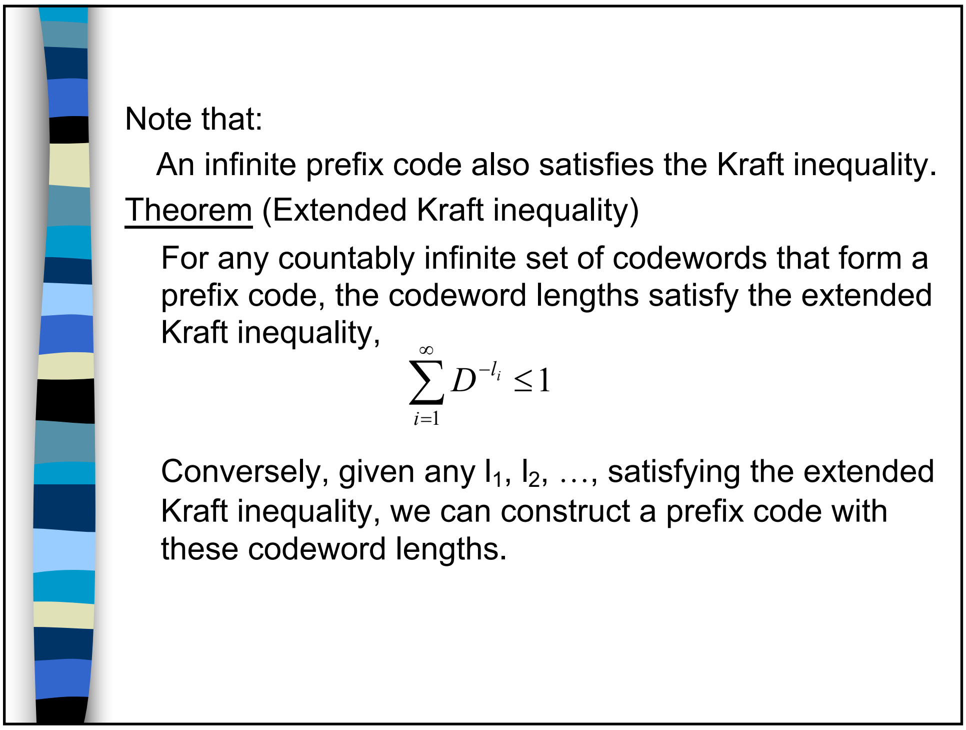


Total # of nodes at level $l_{max} = 2^4$

Total # of descendants of node node Δ (level 2) at

level $l_{max} = 4 = 2^{4}$-2

10



Note that:

An infinite prefix code also satisfies the Kraft inequality.

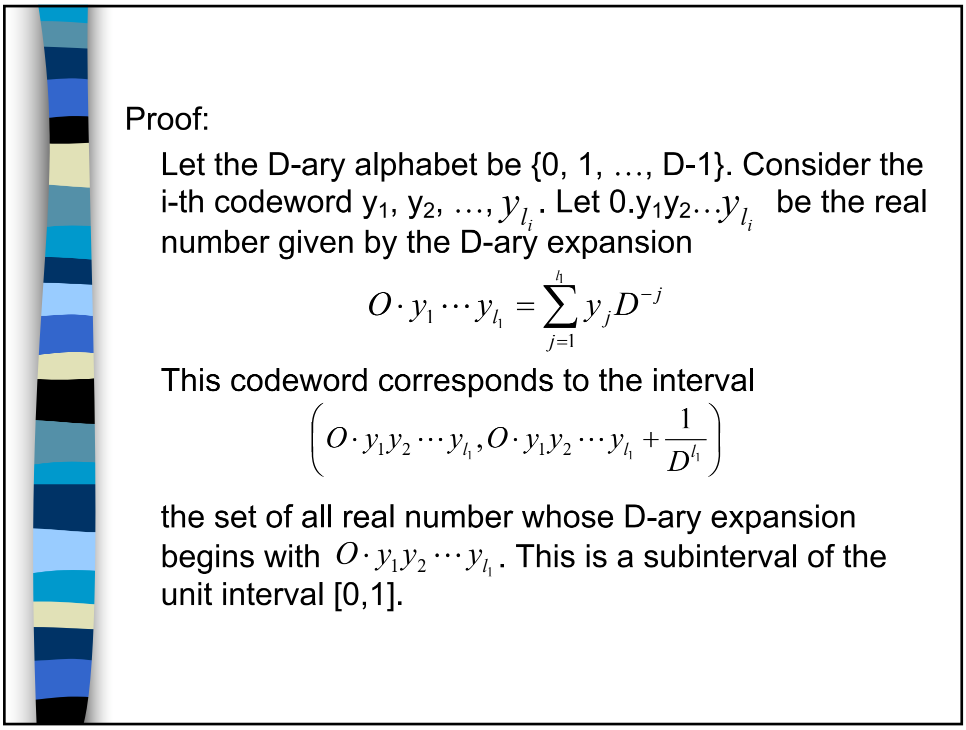
<u>Theorem</u> (Extended Kraft inequality)

For any countably infinite set of codewords that form a prefix code, the codeword lengths satisfy the extended Kraft inequality,

$$\sum_{i=1}^{\infty} D^{-l_i} \leq 1$$

Conversely, given any l_1, l_2, …, satisfying the extended Kraft inequality, we can construct a prefix code with these codeword lengths.

11



Proof:

Let the D-ary alphabet be {0, 1, …, D-1}. Consider the i-th codeword y_1, y_2, …, y_{l_i}. Let $0.y_1y_2\ldots y_{l_i}$ be the real number given by the D-ary expansion

$$O\cdot y_1\cdots y_{l_1} = \sum_{j=1}^{l_1} y_j D^{-j}$$

This codeword corresponds to the interval

$$\left(O\cdot y_1y_2\cdots y_{l_1}, O\cdot y_1y_2\cdots y_{l_1} + \frac{1}{D^{l_1}}\right)$$

the set of all real number whose D-ary expansion begins with $O\cdot y_1y_2\cdots y_{l_1}$. This is a subinterval of the unit interval [0,1].

12

by the prefix condition, those intervals are disjoint. Hence the sum of their lengths has to be less than or equal to 1. This prove that

$$\sum_{i=1}^{\infty} D^{-l_i} \leq 1$$

Just as in the finite case, we can reverse the proof to construct the code for a given $l_1, l_2, \ldots, l_m$ satisfies the Kraft inequality. First reordering the indexing so that $l_1 \geq l_2 \geq \ldots$. The simply assign the intervals in order from the low end of the unit interval.

13

- Optimal codes:

Question: How to find the prefix code with the minimum expected length?

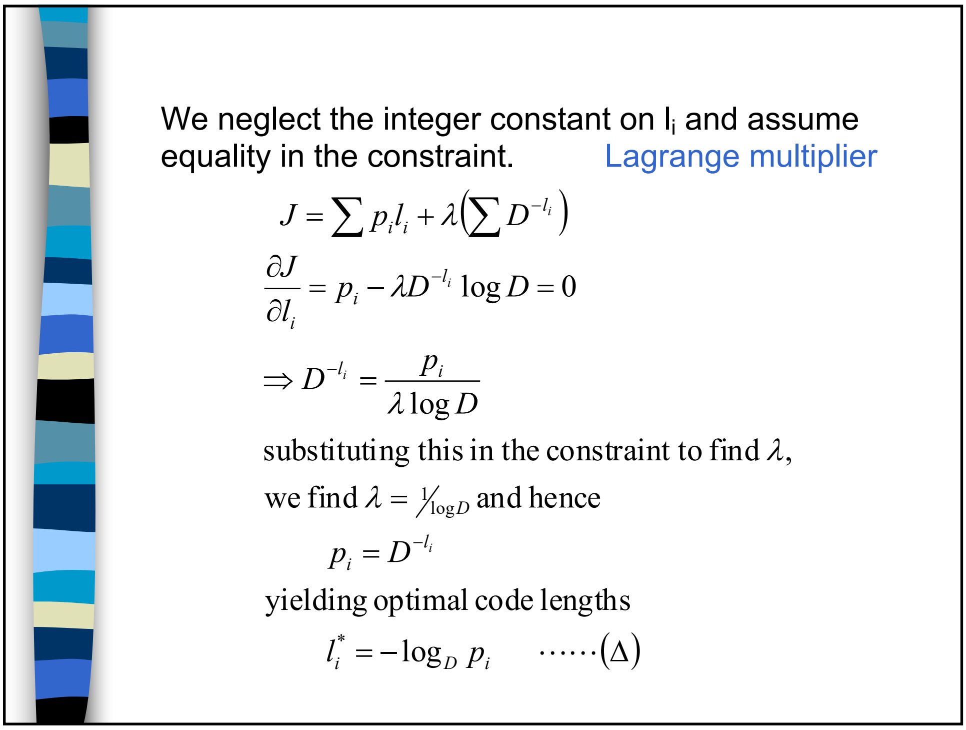
⇒ Find the set of lengths $l_1, l_2, \ldots, l_m$ satisfying the Kraft inequality and whose expected length $L=\Sigma p_i l_i$ is less than the expected length of any other prefix code.

Minimize $L=\Sigma p_i l_i$

Over the integers $l_1, l_2, \ldots, l_m$ satisfying

$$\sum_{i=1}^{\infty} D^{-l_i} \leq 1$$

14



We neglect the integer constant on l_i and assume equality in the constraint. Lagrange multiplier

$$J = \sum p_i l_i + \lambda\left(\sum D^{-l_i}\right)$$

$$\frac{\partial J}{\partial l_i} = p_i - \lambda D^{-l_i} \log D = 0$$

$$\Rightarrow D^{-l_i} = \frac{p_i}{\lambda \log D}$$

substituting this in the constraint to find λ,

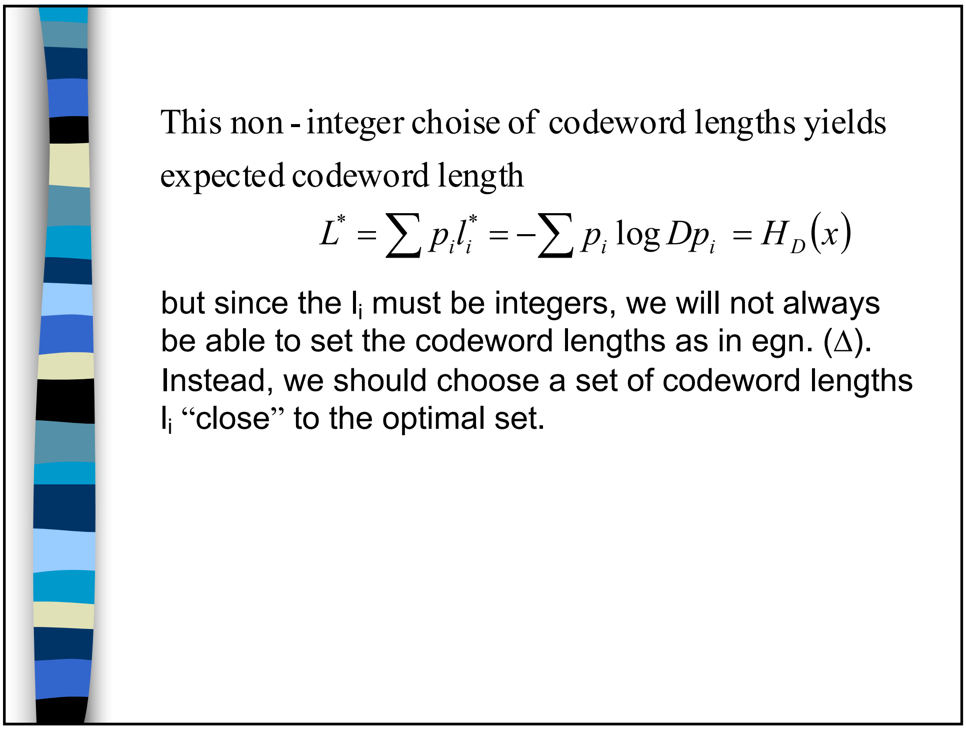
we find $\lambda = {}^{1}\!/_{\log D}$ and hence

$$p_i = D^{-l_i}$$

yielding optimal code lengths

$$l_i^* = -\log_D p_i \qquad \cdots\cdots(\Delta)$$

15

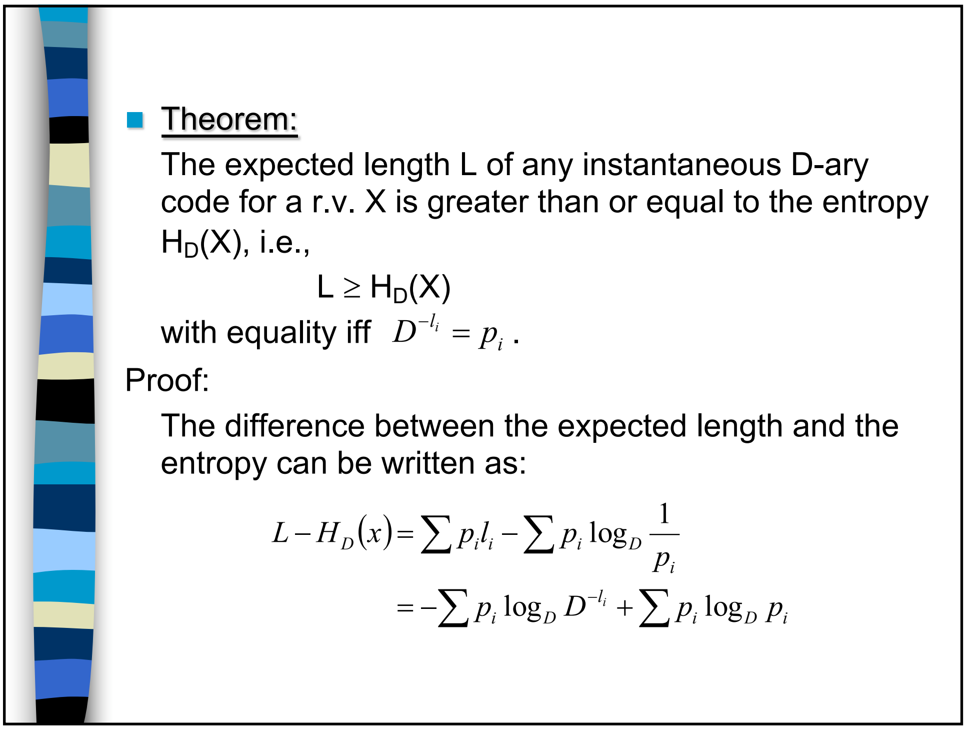


This non - integer choise of codeword lengths yields expected codeword length

$$L^* = \sum p_i l_i^* = -\sum p_i \log Dp_i = H_D(x)$$

but since the l_i must be integers, we will not always be able to set the codeword lengths as in egn. (Δ). Instead, we should choose a set of codeword lengths l_i "close" to the optimal set.

16



- **Theorem:**

  The expected length L of any instantaneous D-ary code for a r.v. X is greater than or equal to the entropy $H_D(X)$, i.e.,

  $$L \geq H_D(X)$$

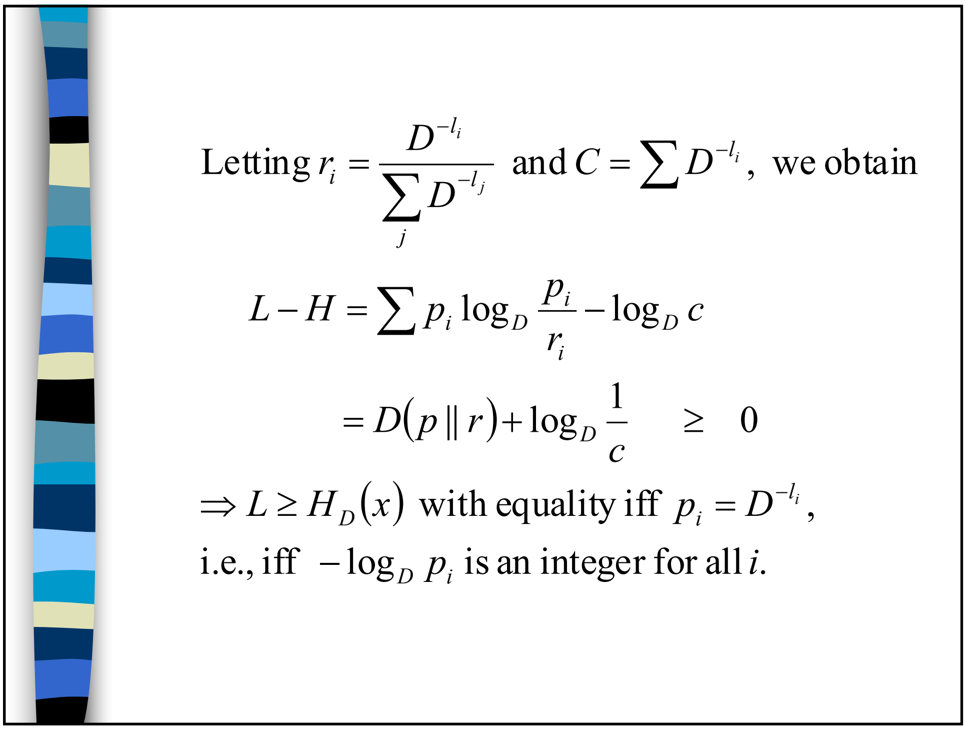
  with equality iff $D^{-l_i} = p_i$.

Proof:

The difference between the expected length and the entropy can be written as:

$$L - H_D(x) = \sum p_i l_i - \sum p_i \log_D \frac{1}{p_i}$$
$$= -\sum p_i \log_D D^{-l_i} + \sum p_i \log_D p_i$$

17



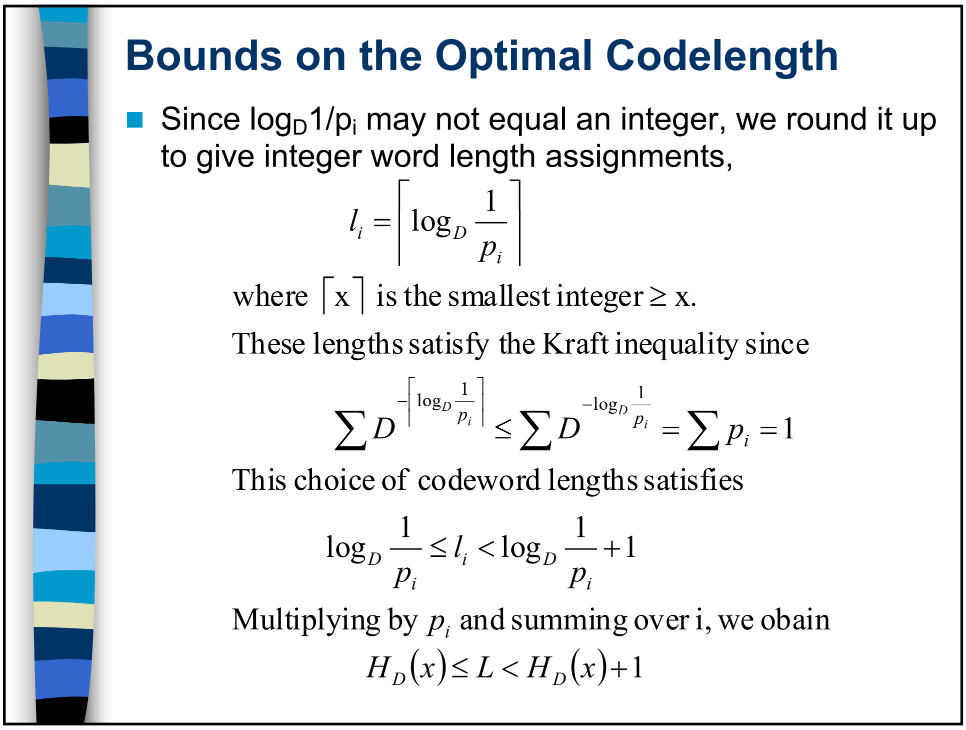
Letting $r_i = \dfrac{D^{-l_i}}{\sum_j D^{-l_j}}$ and $C = \sum D^{-l_i}$, we obtain

$$L - H = \sum p_i \log_D \frac{p_i}{r_i} - \log_D c$$
$$= D(p \| r) + \log_D \frac{1}{c} \quad \geq \quad 0$$

$\Rightarrow L \geq H_D(x)$ with equality iff $p_i = D^{-l_i}$,

i.e., iff $-\log_D p_i$ is an integer for all i.

18



Bounds on the Optimal Codelength

- Since $\log_D 1/p_i$ may not equal an integer, we round it up to give integer word length assignments,

$$l_i = \left\lceil \log_D \frac{1}{p_i} \right\rceil$$

where $\lceil x \rceil$ is the smallest integer $\geq$ x.

These lengths satisfy the Kraft inequality since

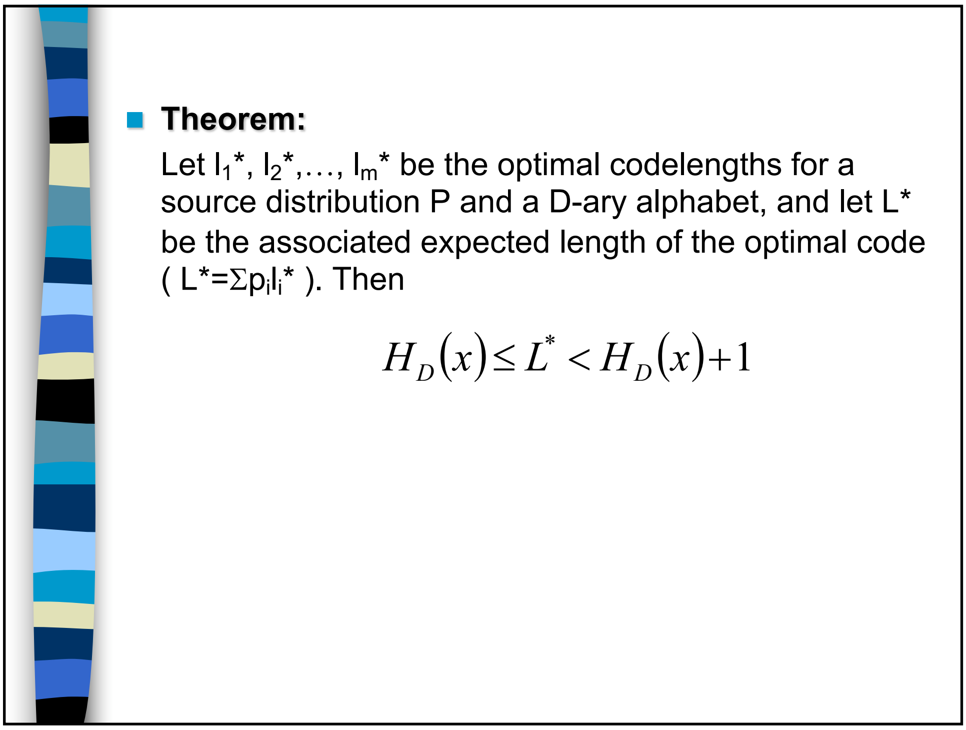
$$\sum D^{-\left\lceil \log_D \frac{1}{p_i} \right\rceil} \leq \sum D^{-\log_D \frac{1}{p_i}} = \sum p_i = 1$$

This choice of codeword lengths satisfies

$$\log_D \frac{1}{p_i} \leq l_i < \log_D \frac{1}{p_i} + 1$$

Multiplying by p_i and summing over i, we obain

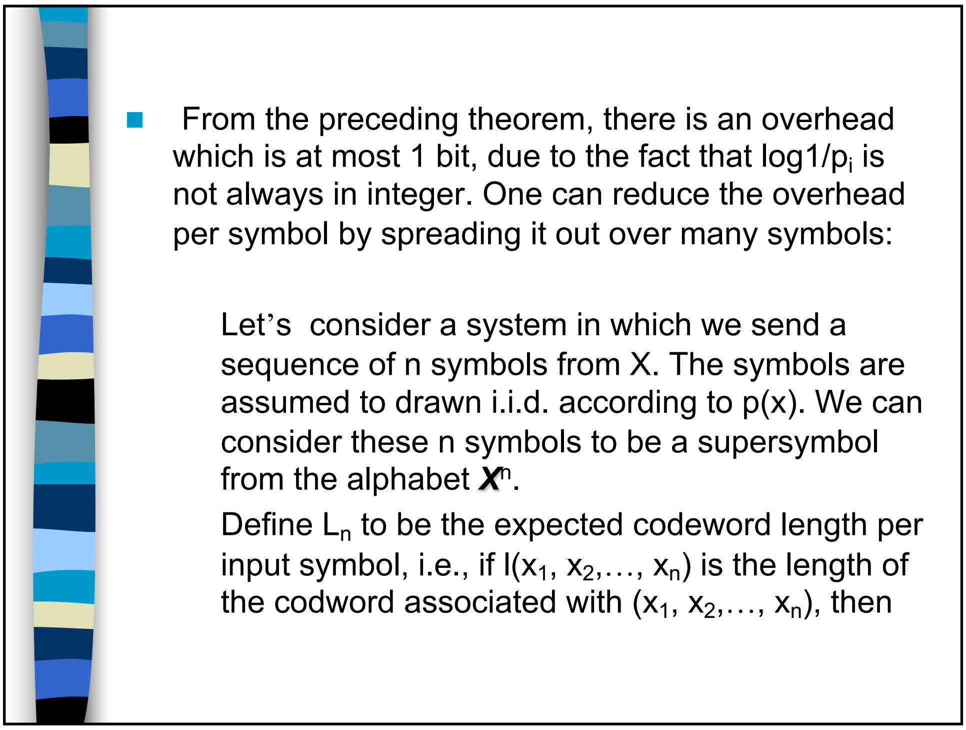
$$H_D(x) \leq L < H_D(x) + 1$$



- **Theorem:**

  Let l_1*, l_2*,…, l_m* be the optimal codelengths for a source distribution P and a D-ary alphabet, and let L* be the associated expected length of the optimal code ($L^*=\Sigma p_i l_i^*$). Then

$$H_D(x) \leq L^* < H_D(x) + 1$$

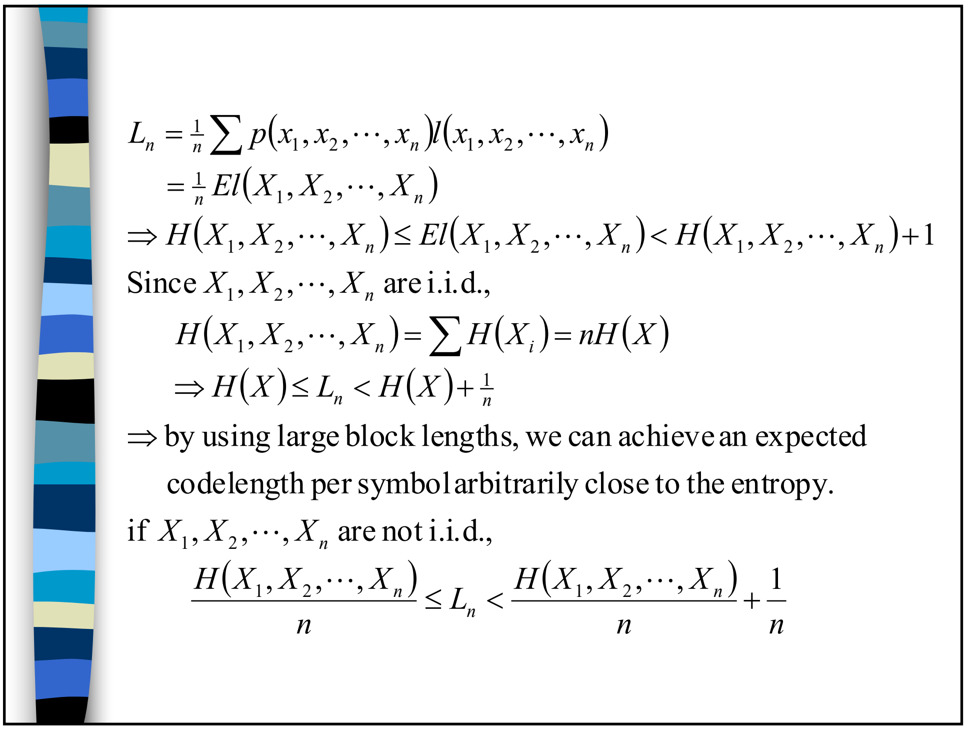


- From the preceding theorem, there is an overhead which is at most 1 bit, due to the fact that log1/p_i is not always in integer. One can reduce the overhead per symbol by spreading it out over many symbols:

  Let's consider a system in which we send a sequence of n symbols from X. The symbols are assumed to drawn i.i.d. according to p(x). We can consider these n symbols to be a supersymbol from the alphabet $\boldsymbol{X}^n$.

  Define L_n to be the expected codeword length per input symbol, i.e., if $l(x_1, x_2, \ldots, x_n)$ is the length of the codword associated with $(x_1, x_2, \ldots, x_n)$, then

21



$$L_n = \tfrac{1}{n}\sum p(x_1, x_2, \cdots, x_n) l(x_1, x_2, \cdots, x_n)$$
$$= \tfrac{1}{n} El(X_1, X_2, \cdots, X_n)$$
$$\Rightarrow H(X_1, X_2, \cdots, X_n) \le El(X_1, X_2, \cdots, X_n) < H(X_1, X_2, \cdots, X_n) + 1$$

Since $X_1, X_2, \cdots, X_n$ are i.i.d.,

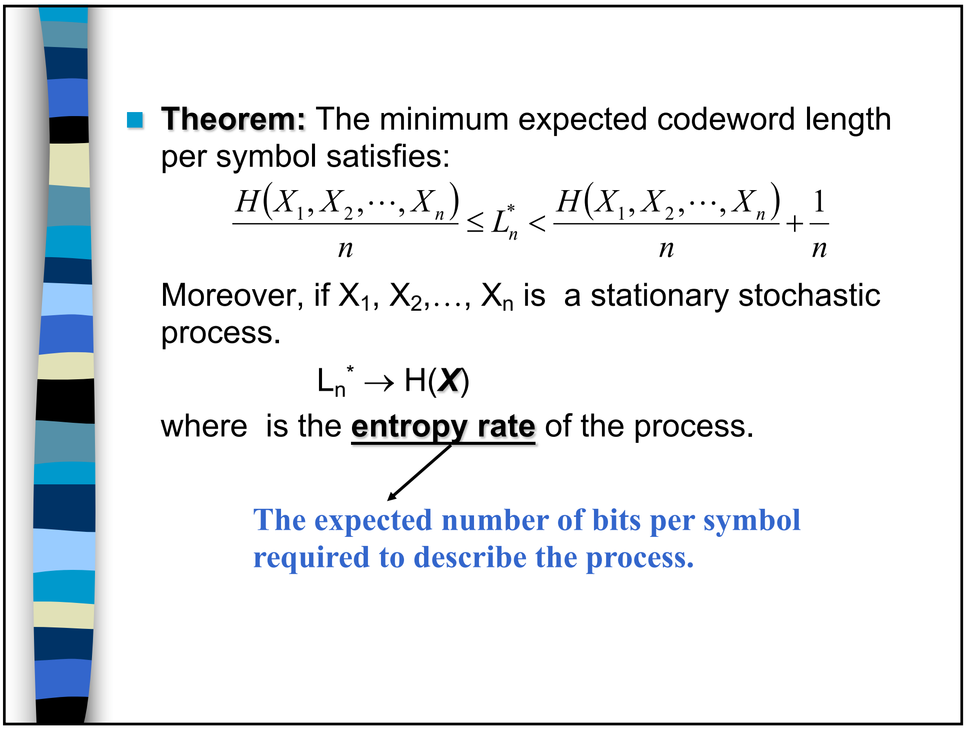
$$H(X_1, X_2, \cdots, X_n) = \sum H(X_i) = nH(X)$$
$$\Rightarrow H(X) \le L_n < H(X) + \tfrac{1}{n}$$

$\Rightarrow$ by using large block lengths, we can achieve an expected codelength per symbol arbitrarily close to the entropy.

if $X_1, X_2, \cdots, X_n$ are not i.i.d.,

$$\frac{H(X_1, X_2, \cdots, X_n)}{n} \le L_n < \frac{H(X_1, X_2, \cdots, X_n)}{n} + \frac{1}{n}$$

22



- **Theorem:** The minimum expected codeword length per symbol satisfies:

$$\frac{H(X_1,X_2,\cdots,X_n)}{n} \le L_n^* < \frac{H(X_1,X_2,\cdots,X_n)}{n} + \frac{1}{n}$$

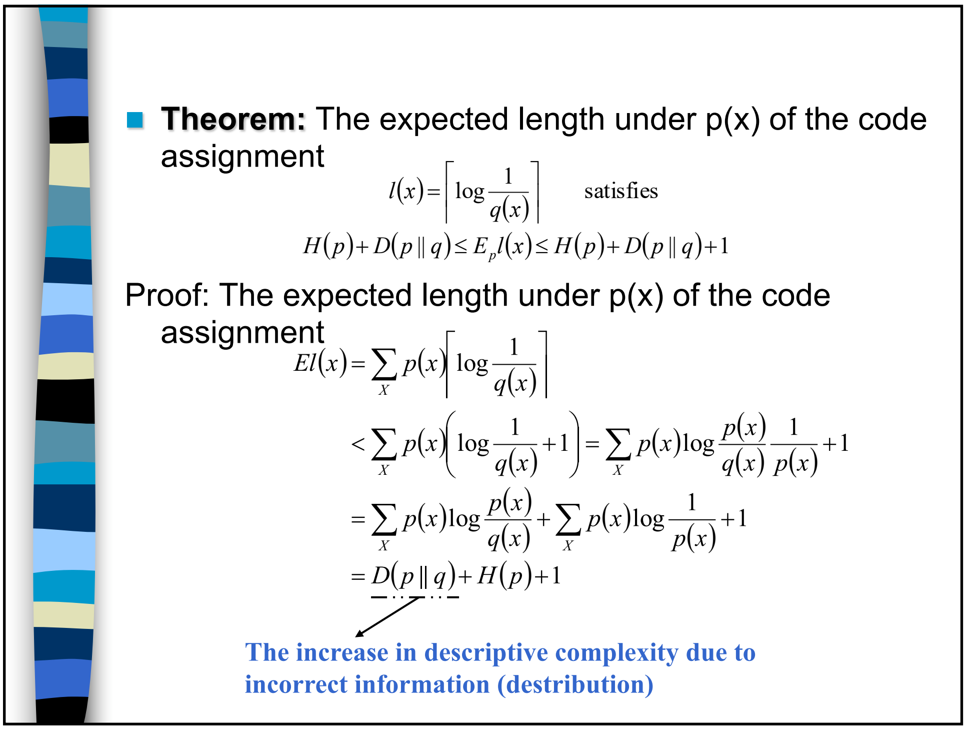
Moreover, if X_1, X_2,…, X_n is a stationary stochastic process.

$$L_n^* \rightarrow H(\boldsymbol{X})$$

where is the **entropy rate** of the process.

The expected number of bits per symbol required to describe the process.

23



- **Theorem:** The expected length under p(x) of the code assignment

$$l(x) = \left\lceil \log \frac{1}{q(x)} \right\rceil \quad \text{satisfies}$$

$$H(p) + D(p \| q) \le E_p l(x) \le H(p) + D(p \| q) + 1$$

Proof: The expected length under p(x) of the code assignment

$$El(x) = \sum_X p(x) \left\lceil \log \frac{1}{q(x)} \right\rceil$$

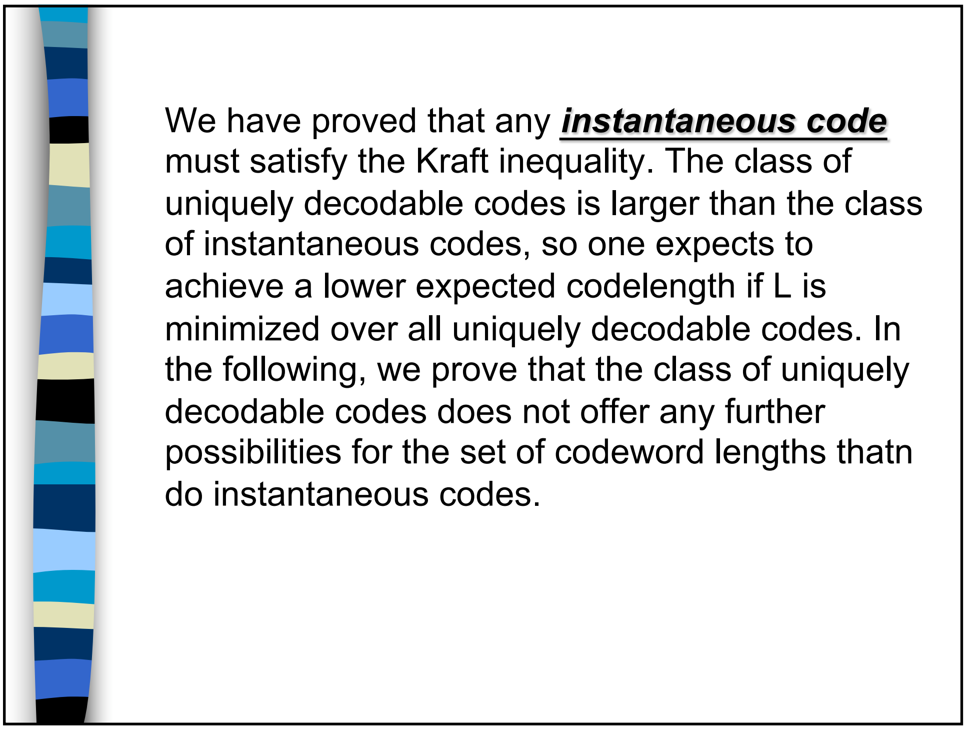
$$< \sum_X p(x) \left(\log \frac{1}{q(x)} + 1 \right) = \sum_X p(x) \log \frac{p(x)}{q(x)} \frac{1}{p(x)} + 1$$

$$= \sum_X p(x) \log \frac{p(x)}{q(x)} + \sum_X p(x) \log \frac{1}{p(x)} + 1$$

$$= \underline{D(p \| q)} + H(p) + 1$$

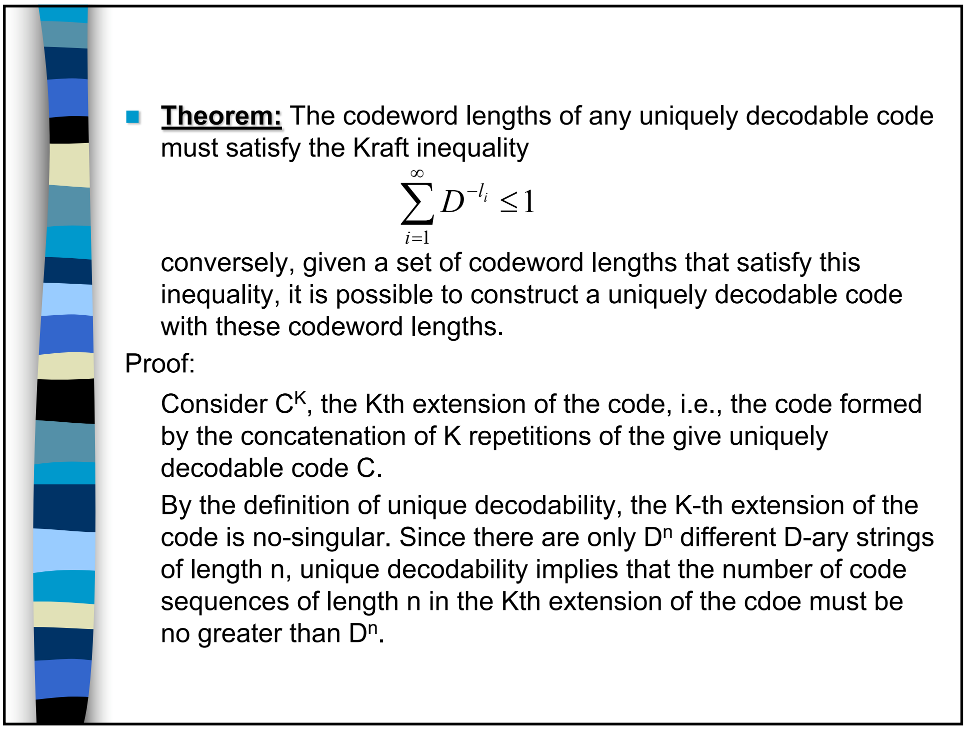
The increase in descriptive complexity due to incorrect information (destribution)

24



We have proved that any ***instantaneous code*** must satisfy the Kraft inequality. The class of uniquely decodable codes is larger than the class of instantaneous codes, so one expects to achieve a lower expected codelength if L is minimized over all uniquely decodable codes. In the following, we prove that the class of uniquely decodable codes does not offer any further possibilities for the set of codeword lengths thatn do instantaneous codes.

25



- **Theorem:** The codeword lengths of any uniquely decodable code must satisfy the Kraft inequality

$$\sum_{i=1}^{\infty} D^{-l_i} \leq 1$$

conversely, given a set of codeword lengths that satisfy this inequality, it is possible to construct a uniquely decodable code with these codeword lengths.

Proof:

Consider C^K, the Kth extension of the code, i.e., the code formed by the concatenation of K repetitions of the give uniquely decodable code C.

By the definition of unique decodability, the K-th extension of the code is no-singular. Since there are only D^n different D-ary strings of length n, unique decodability implies that the number of code sequences of length n in the Kth extension of the cdoe must be no greater than D^n.

26

- Let the codeword lengths of the symbols x∈***X*** be denoted by l(x). For the extension code, the length of the code-sequence is:

$$l(x_1, x_2, \cdots, x_k) = \sum_{i=1}^{k} l(x_1)$$

Let's consider the Kth power of the Kraft inequality:

$$\begin{aligned}\left(\sum_{x\in\mathbf{X}} D^{-l(x)}\right)^k &= \sum_{x_1\in\mathbf{X}}\sum_{x_2\in\mathbf{X}}\cdots\sum_{x_k\in\mathbf{X}} D^{-l(x_1)}D^{-l(x_2)}\ldots D^{-l(x_k)}\\ &= \sum_{x_1,x_2,\cdots,x_k\in\mathbf{X}^K} D^{-l(x_1)}D^{-l(x_2)}\ldots D^{-l(x_k)}\\ &= \sum_{x^K\in\mathbf{X}^K} D^{-l(x^k)}\\ &= \sum_{m=1}^{kl_{\max}} a(m)D^{-m}\end{aligned}$$

(or k)

27

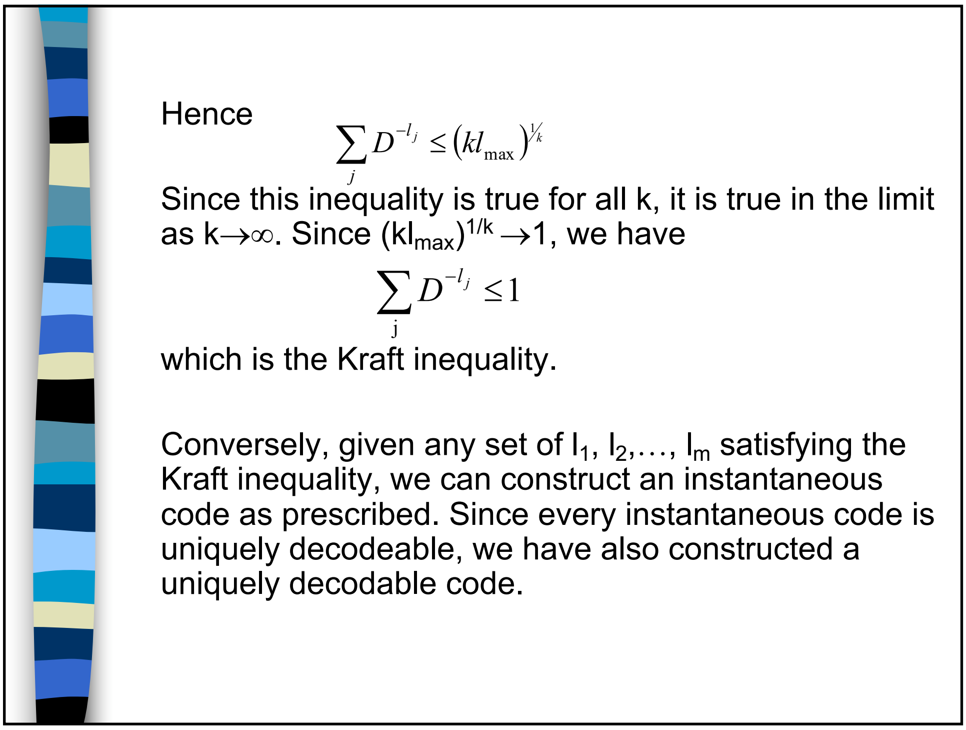
where $l_{\max}$ is the maximum codeword length and a(m) is the number of source sequences x^k mapping into codewords of length m.

For uniquely decodable code, there is at most one sequence mapping into each code m-sequence and there are at most D^m code m-sequences.

$$\Rightarrow \qquad a(m) \le D^m$$

$$\begin{aligned}\left(\sum_{x\in\mathbf{X}} D^{-l(x)}\right)^k &= \sum_{m=1}^{kl_{\max}} a(m)D^{-m}\\ &\le \sum_{m=1}^{kl_{\max}} D^m D^{-m}\\ &= kl_{\max}\end{aligned}$$

28



Hence

$$\sum_j D^{-l_j} \le (kl_{\max})^{1/k}$$

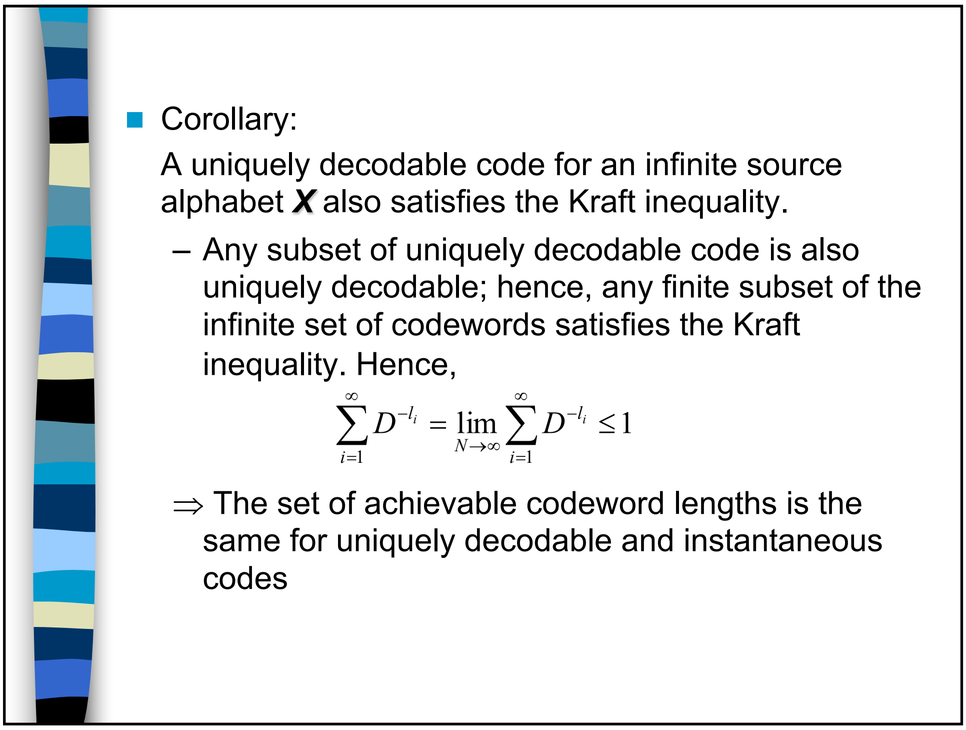
Since this inequality is true for all k, it is true in the limit as k→∞. Since $(kl_{\max})^{1/k} \to 1$, we have

$$\sum_{\mathrm{j}} D^{-l_j} \le 1$$

which is the Kraft inequality.

Conversely, given any set of $l_1, l_2, \ldots, l_m$ satisfying the Kraft inequality, we can construct an instantaneous code as prescribed. Since every instantaneous code is uniquely decodeable, we have also constructed a uniquely decodable code.

29



- Corollary:

  A uniquely decodable code for an infinite source alphabet ***X*** also satisfies the Kraft inequality.

  – Any subset of uniquely decodable code is also uniquely decodable; hence, any finite subset of the infinite set of codewords satisfies the Kraft inequality. Hence,

$$\sum_{i=1}^{\infty} D^{-l_i} = \lim_{N\to\infty} \sum_{i=1}^{\infty} D^{-l_i} \le 1$$

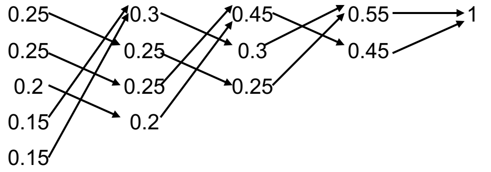
  ⇒ The set of achievable codeword lengths is the same for uniquely decodable and instantaneous codes

30

Huffman Codes

- D. A. Huffman, “A method for the construction of minimum redundancy codes,” Proc. IRE, vol. 40; pp. 1098–1101, 1952.
- Example:

Code word length	codeword	X	Probability				
2	01	1	0.25	0.3	0.45	0.55	1
2	10	2	0.25	0.25	0.3	0.45	
2	11	3	0.2	0.25	0.25		
3	000	4	0.15	0.2			
3	001	5	0.15				



31

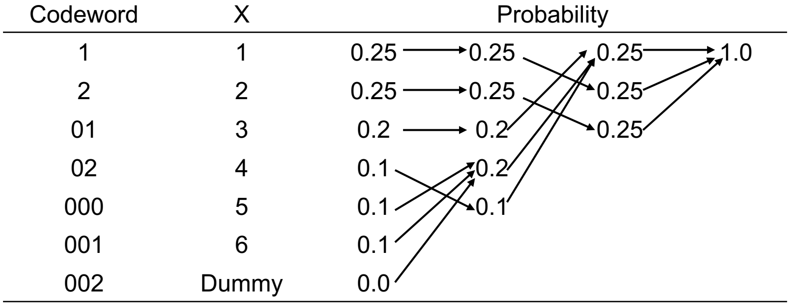
We expect the optimal binary code for X to have the longest codewords assigned to the symbols 4 and 5. Both these lengths must be equal, since otherwise we can delete a bit from the longer codeword and still have a prefix code, but with a shorter expected length. In general, we can construct a code in which the two longest codewords differ only in the last bit.

32

- Example:

  If D≥3, we may not have a sufficient number of symbols so that we can combine them D at a them. In such a case, we add dummy symbols to the end of the set of symbols. The dummy symbols have **probability 0** and are inserted to fill the tree.

  Since at each stage of the reduction, the number of symbols is reduced by D-1, we want the total number of the symbols to be 1+k(D-1), where k is the number of levels in the tree.



Codeword	X	Probability			
1	1	0.25	0.25	0.25	1.0
2	2	0.25	0.25	0.25	
01	3	0.2	0.2	0.25	
02	4	0.1	0.2		
000	5	0.1	0.1		
001	6	0.1			
002	Dummy	0.0			

Optimality of Huffman codes

- There are many optimal codes:

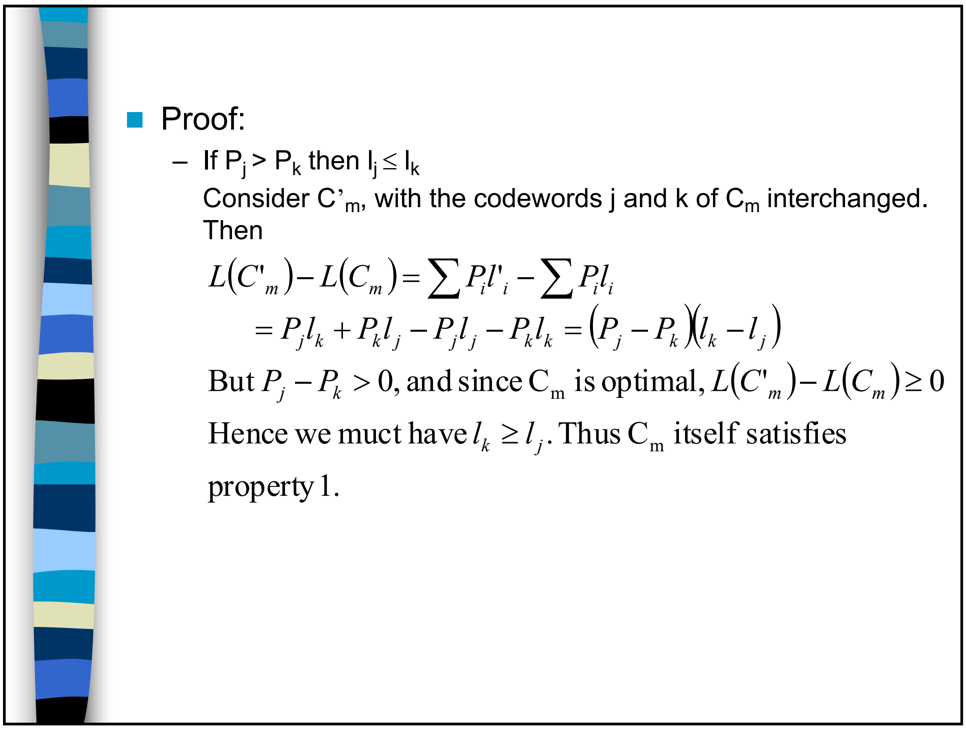
  inverting all the bits or exchanging two codewords of the same length will give another optimal code.

w.l.o.g., we assume $P_1 \geq P_2 \geq \ldots \geq P_m$

- Lemma:

  For any distribution, there exists an optimal instantaneous code (with minimum expected length) that satisfies the following properties:

  1. If $P_j > P_k$ then $l_j \leq l_k$
  2. The two longest codewords have the same length.
  3. The two longest codewords differ only in the last bit and correspond to the two least likely symbols.



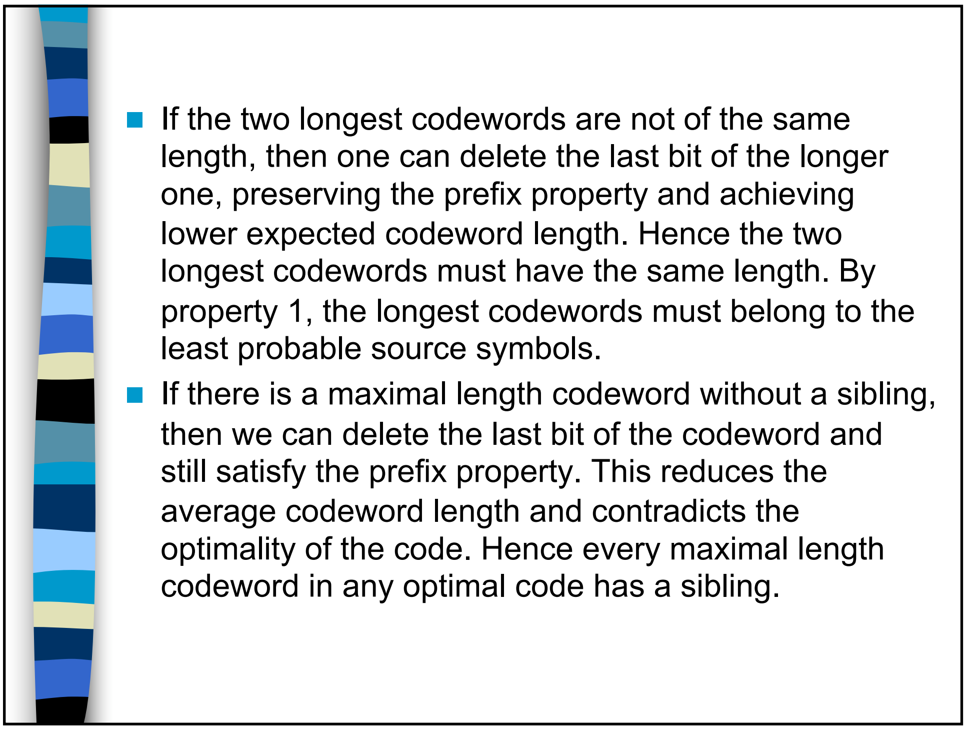
- Proof:
  - If $P_j > P_k$ then $l_j \leq l_k$
    Consider C'_m, with the codewords j and k of C_m interchanged. Then

    $$L(C'_m) - L(C_m) = \sum P_i l'_i - \sum P_i l_i$$
    $$= P_j l_k + P_k l_j - P_j l_j - P_k l_k = (P_j - P_k)(l_k - l_j)$$

    But $P_j - P_k > 0$, and since $\mathrm{C_m}$ is optimal, $L(C'_m) - L(C_m) \geq 0$

    Hence we muct have $l_k \geq l_j$. Thus $\mathrm{C_m}$ itself satisfies property 1.

35



- If the two longest codewords are not of the same length, then one can delete the last bit of the longer one, preserving the prefix property and achieving lower expected codeword length. Hence the two longest codewords must have the same length. By property 1, the longest codewords must belong to the least probable source symbols.
- If there is a maximal length codeword without a sibling, then we can delete the last bit of the codeword and still satisfy the prefix property. This reduces the average codeword length and contradicts the optimality of the code. Hence every maximal length codeword in any optimal code has a sibling.

36

Now we can exchange the longest length codewords so that the two lowest probability source symbols are associated with two sibling on the tree. This does not change the expected length Σ $P_i l_i$. Thus the codewords for the two lowest probability source symbols have maximal length and agree in all but the last bit.

⇒If $P_1 \geq P_2 \geq \ldots \geq P_m$, then there exists an optimal code with $l_1 \leq l_2 \leq \ldots \leq l_{m-1} = l_m$, and codewords $c(X_{m-1})$ and $C(X_m)$ differ only in the last bit.

37

For a code C_m satisfying the preperties of the above lemma, we now define a "**merged**" code C_{m-1} for m-1 symbols as follows: take the common prefix of the two largest codewords (corresponding to the two least likely symbols), and allot it to a symbol with probability P_{m-1} and P_m. All the codewords remain the same.

	C_{m-1}		C_m	
P_1	w'_1	l'_1	$w_1 = w'_1$	$l_1 = l'_1$
P_2	w'_2	l'_2	$w_2 = w'_2$	$l_2 = l'_2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
P_{m-2}	w'_{m-2}	l'_{m-2}	$w_{m-2} = w'_{m-2}$	$l_{m-2} = l'_{m-2}$
$P_{m-1} + P_m$	w'_{m-1}	l'_{m-1}	$w_{m-1} = w'_{m-1}0$	$l_{m-1} = l'_{m-1}+1$
			$w_m = w'_{m-1}1$	$l_m = l'_{m-1}+1$

38

where w denotes a binary codeword and l denotes its length. The expected length of the code C_m is

$$
\begin{aligned}
L(C_m) &= \sum_{i=1}^{m} p_i l_i \\
&= \sum_{i=1}^{m-2} p_i l'_i + p_{m-1}(l'_{m-1}+1) + p_m(l'_{m-1}+1) \\
&= \sum_{i=1}^{m-2} p_i l'_i + (p_{m-1} + p_m) l'_{m-1} + p_{m-1} + p_m \\
&= L(C_{m-1}) + p_{m-1} + p_m
\end{aligned}
$$

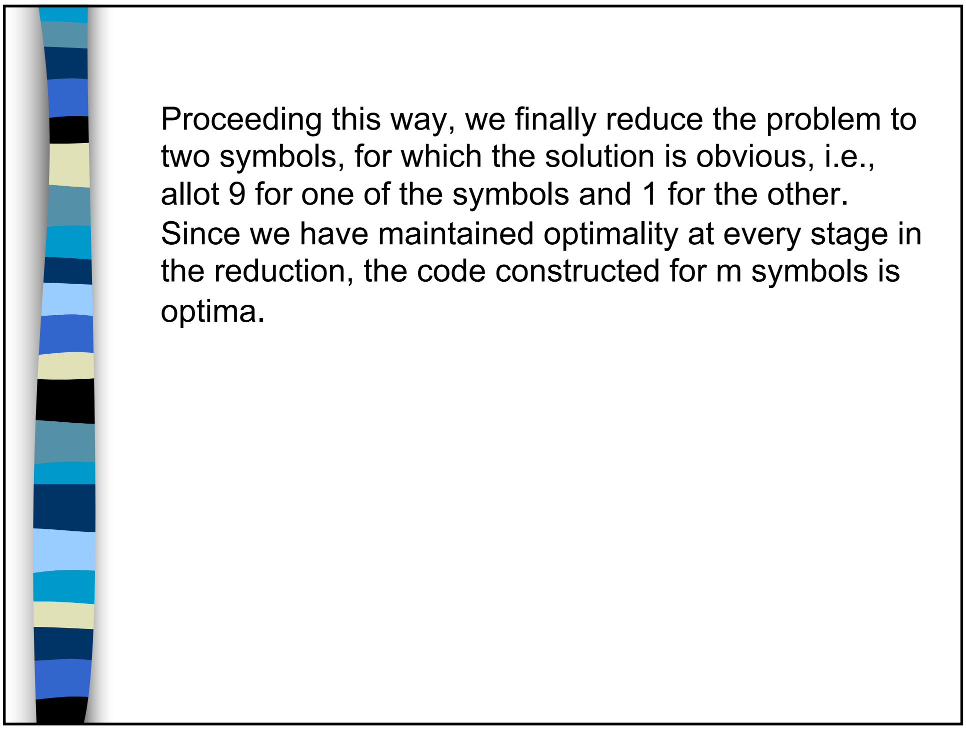
39

- Thus the expected length of the code C_m differs from the expected length of C_{m-1} by a fixed amount **independent** of C_{m-1}. Thus minimizing the expected length $L(C_m)$ is equivalent to minimizing $L(C_{m-1})$. Thus we have reduced the problem to one with m-1 symbols and probability masses (P_1, P_2,…, P_{m-1}+ P_m).

  ⇒

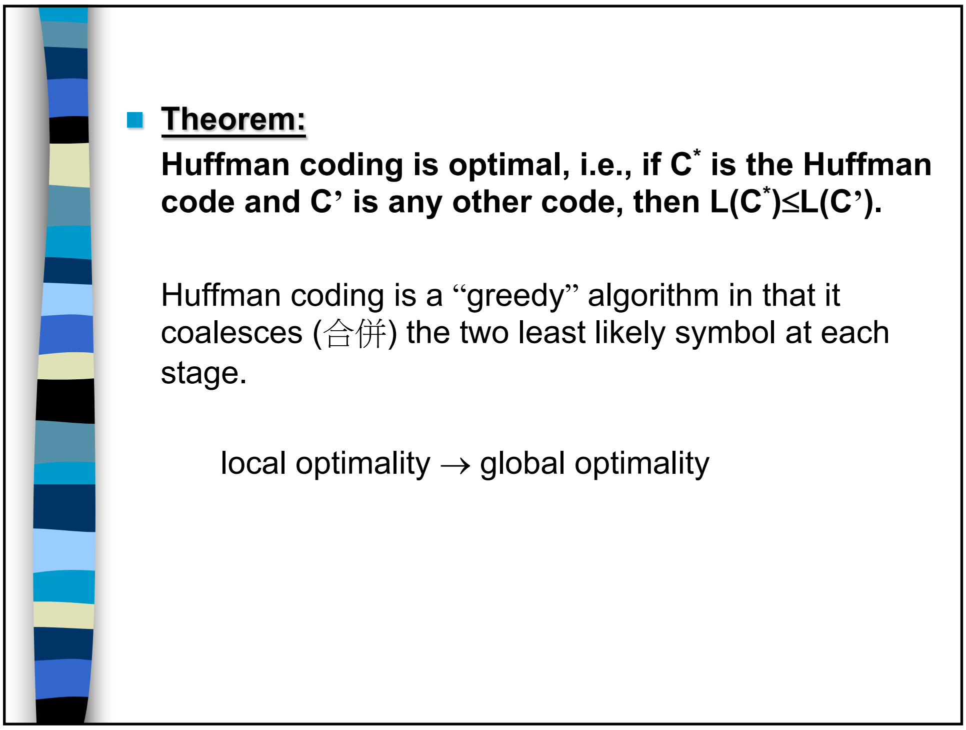
  we again look for a code which satisfies the properties of the lemma for there m-1 symbols and then reduce the problem to find the optimal code for m-2 symbols with the appropriate probability masses obtained by merging the two lowest probabilities on the previous marged list.

40



Proceeding this way, we finally reduce the problem to two symbols, for which the solution is obvious, i.e., allot 9 for one of the symbols and 1 for the other. Since we have maintained optimality at every stage in the reduction, the code constructed for m symbols is optima.

41



- **Theorem:**

  Huffman coding is optimal, i.e., if C^* is the Huffman code and C’ is any other code, then $L(C^*)\leq L(C')$.

  Huffman coding is a “greedy” algorithm in that it coalesces (合併) the two least likely symbol at each stage.

  local optimality → global optimality

42