MAS.160 / MAS.510 / MAS.511 Signals, Systems and Information for Media Technology
Fall 2007

Balance

Suppose you have eight billiard balls. One of them is defective -- it weighs more than the others. How do you tell, using a balance, which ball is defective in two weighings?

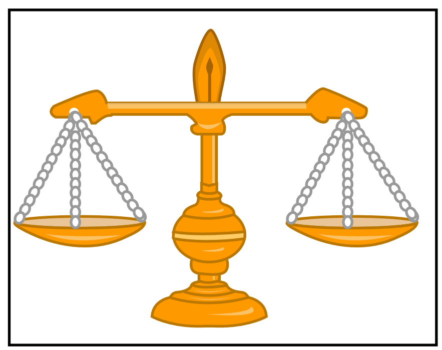
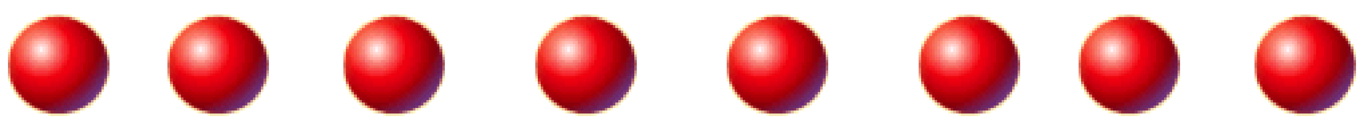


Figure by MIT OpenCourseWare.



Information Theory

How do you define the information a message carries?

How much information does a message carry? How much of a message is redundant?
How do we measure information and what are its units?

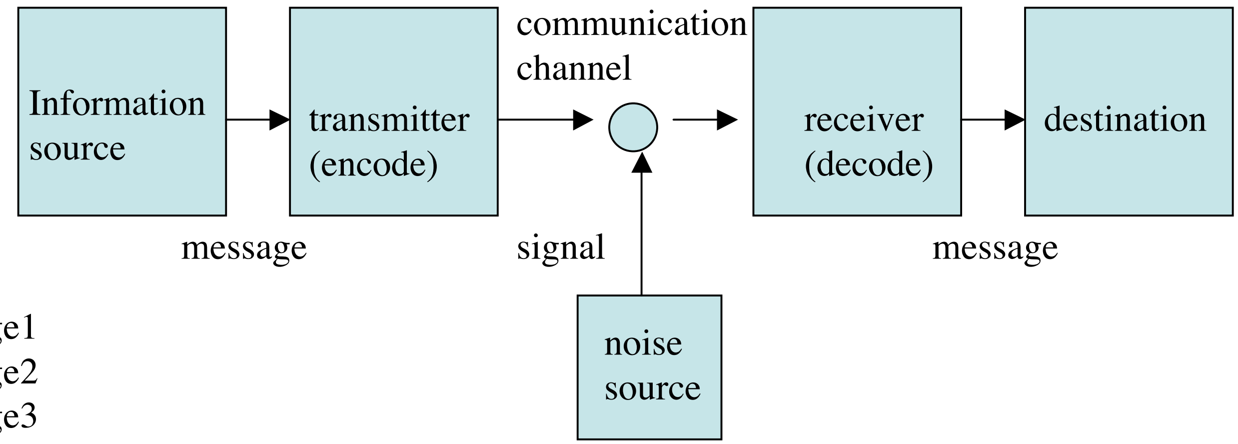
How do we model a transmitter of messages?
What is the average rate of information a transmitter generates?

How much capacity does a communication channel have (with a given data format and data frequency)?

Can we remove the redundancy from a message to fill the capacity of a channel? (lossless compression)
How much can we compress a message and still exactly recover message?

How does noise affect the capacity of a channel?
Can we use redundancy to accurately recover a signal sent over a noisy line? (error correction)

Information Theory



message1
message2
message3
…

OR

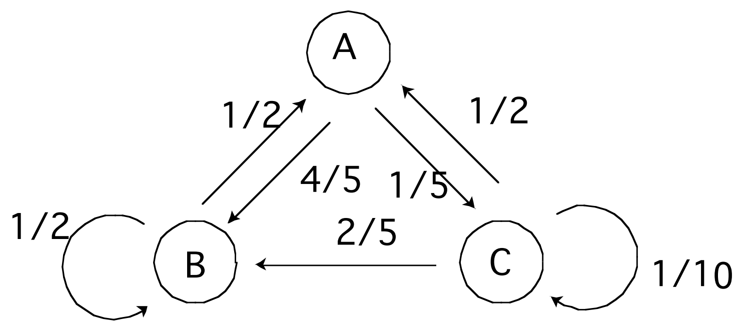
symbol1
symbol2
symbol3
…
AND
message1=symbol1, symbol2
message2=symbol3, symbol5

Information source selects a desired message from a set of possible messages
OR
selects a sequence of symbols from a set of symbols to represent a message.

Destination decides which message among set of (agreed) possible messages, the information source sent.

Why are we interested in Markov Models?

We can represent an information source as an engine creating symbols at some rate according to probabilistic rules. The Markov model represents those rules as transition probabilities between symbols.



ACBBA…

In the long term, each symbol has a certain steady state probability.

$$v_{ss} = \begin{bmatrix} \frac{1}{3} & \frac{16}{27} & \frac{2}{27} \end{bmatrix}$$

Based on these probabilities, we can define the amount of information, I, that a symbol carries and what the average rate of information or entropy, H, a system generates.

Discrete Markov Chain

Transition Matrix

A **Markov system** (or **Markov process** or **Markov chain**) is a system that can be in one of several (numbered) **states**, and can pass from one state to another each **time step** according to fixed probabilities.

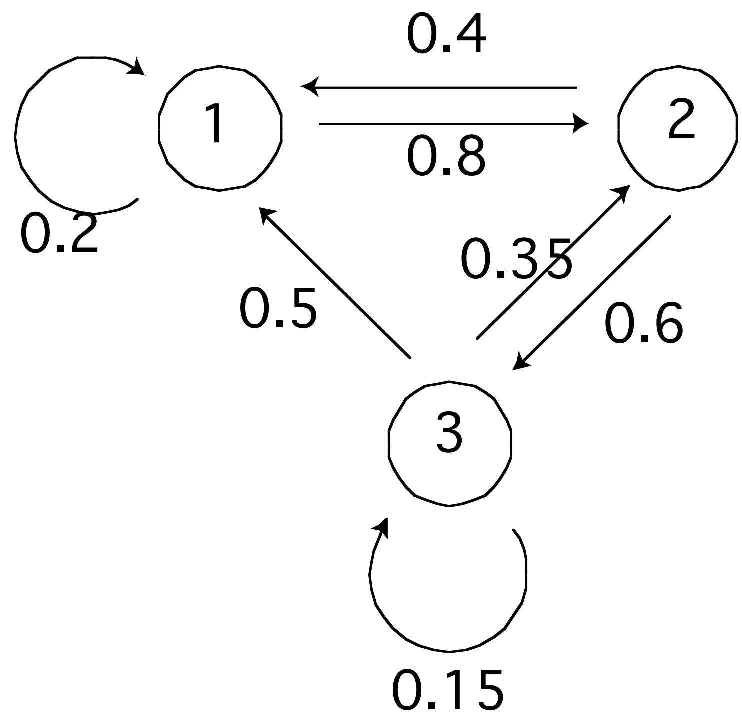
If a Markov system is in state i, there is a fixed probability, p_{ij}, of it going into state j the next time step, and p_{ij} is called a **transition probability**.

A Markov system can be illustrated by means of a **state transition diagram**, which is a diagram showing all the states and transition probabilities.

The matrix P whose ijth entry is p_{ij} is called the **transition matrix** associated with the system. The entries in each row add up to 1. Thus, for instance, a 2×2 transition matrix P would be set up as in the following figure.

http://people.hofstra.edu/faculty/Stefan_Waner/RealWorld/Summary8.html

Discrete Markov Chain



To

$$\begin{array}{cc} & \begin{array}{ccc}1 & 2 & 3\end{array} \\ \text{From}\quad \begin{array}{c}1\\2\\3\end{array} & \begin{bmatrix}0.2 & 0.8 & 0\\0.4 & 0 & 0.6\\0.5 & 0.35 & 0.15\end{bmatrix}\end{array}$$

Arrows originating in State 1

Arrows originating in State 2

Arrows originating in State 3

1-step Distribution

Distribution After 1 Step: vP

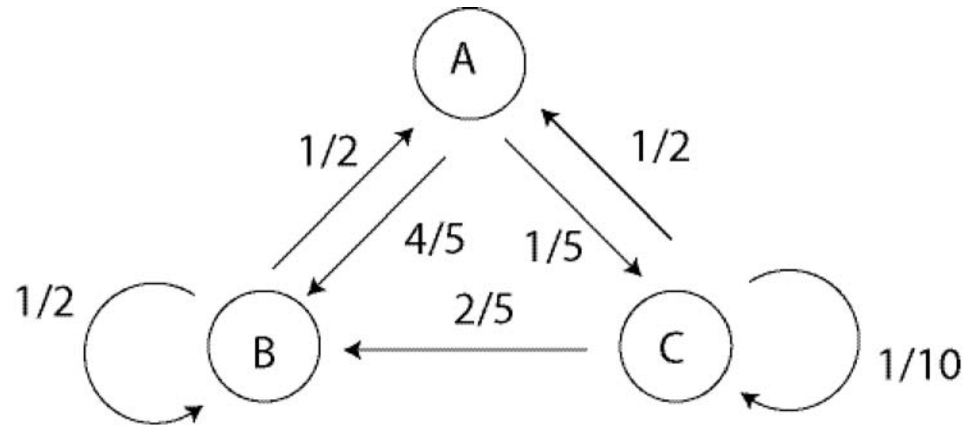
If v is an initial probability distribution vector and P is the transition matrix for a Markov system, then the probability vector after 1 step is the matrix product, vP.

initial probabilities

i	p(i)
A	$\frac{9}{27}$
B	$\frac{16}{27}$
C	$\frac{2}{27}$

transition probabilities

$p_i(j)$		j		
		A	B	C
i	A	0	$\frac{4}{5}$	$\frac{1}{5}$
	B	$\frac{1}{2}$	$\frac{1}{2}$	0
	C	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$



Initial probability vector

$$v = [\frac{9}{27}, \frac{16}{27}, \frac{2}{27}]$$

Transition matrix

$$P = \begin{bmatrix} 0 & \frac{4}{5} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{2}{5} & \frac{1}{10} \end{bmatrix}$$

Distribution after 1 step

$$vP = \begin{bmatrix} \frac{1}{3} & \frac{16}{27} & \frac{2}{27} \end{bmatrix}$$

n-steps Distribution

Distribution After 1 Step: vP

If v is an initial probability distribution vector and P is the transition matrix for a Markov system, then the distribution vector after 1 step is the matrix product, vP.

Distribution After 2 Steps: vP^2

The distribution one step later, obtained by again multiplying by P, is given by

$$(vP)P = vP^2.$$

Distribution After n Steps: vP^n

Similarly, the distribution after n steps can be obtained by multiplying v on the right by P n times, or multiplying v by P^n.

$$(vP)PP\dots P = vP^n$$

The ij^{th} entry in P^n is the probability that the system will pass from state i to state j in n steps.

Stationary

What happens as number of steps n goes to infinity?

$$V_{ss}P = V_{ss}$$

$v_{ss}=[v_x \; v_y \; v_z \; \ldots]$

$$v_x + v_y + v_z + \ldots = 1$$

n+1 equations

n unknowns

A steady state probability vector is then given by $v_{ss}=[v_x \; v_y \; v_z \; \ldots]$

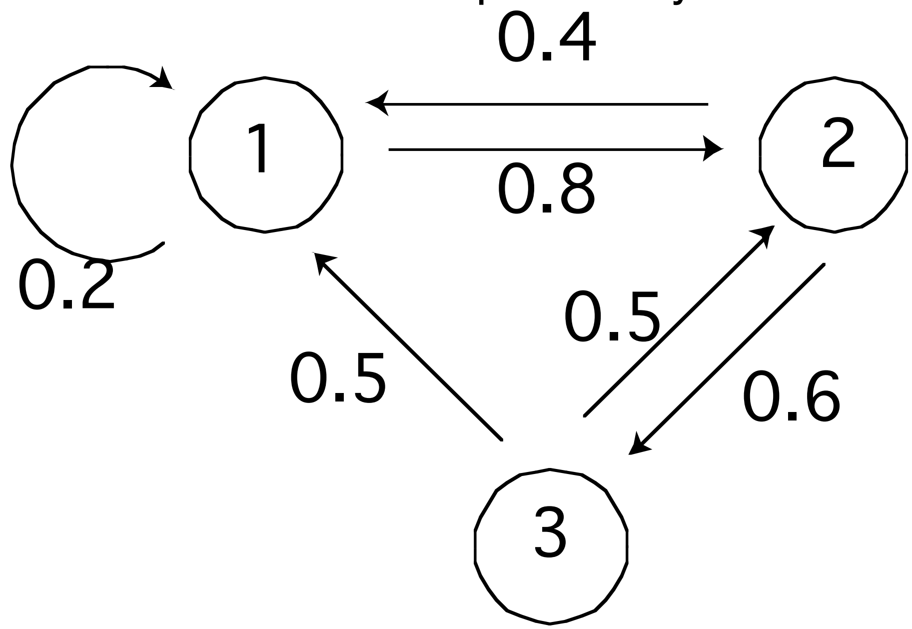
If the higher and higher powers of P approach a fixed matrix P_∞, we refer to P_∞ as the **steady state** or **long-term transition matrix**.

$$P_\infty = \begin{bmatrix} v_x & v_y & v_z \\ v_x & v_y & v_z \\ v_x & v_y & v_z \end{bmatrix}$$

$v_{ss}=[v_x \; v_y \; v_z \; \ldots]$

Examples

Let $P = \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.4 & 0 & 0.6 \\ 0.5 & 0.5 & 0 \end{bmatrix}$ and $v = \begin{bmatrix} 0.2 & 0.4 & 0.4 \end{bmatrix}$ be an initial probability distribution.



32123…

Then the distribution after one step is given by

$$vP = \begin{bmatrix} 0.2 & 0.4 & 0.4 \end{bmatrix} \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.4 & 0 & 0.6 \\ 0.5 & 0.5 & 0 \end{bmatrix} = \begin{bmatrix} 0.4 & 0.36 & 0.24 \end{bmatrix}$$

$$0.2(0.2)+(0.4)(0.4)+(0.4)(0.5)=0.04+0.16+0.20=0.4$$

The distribution after one step is given by

$$vP = \begin{bmatrix} 0.2 & 0.4 & 0.4 \end{bmatrix} \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.4 & 0 & 0.6 \\ 0.5 & 0.5 & 0 \end{bmatrix} = \begin{bmatrix} 0.4 & 0.36 & 0.24 \end{bmatrix}$$

The two-step distribution one step later is given by

$$vP^2 = (vP)P = \begin{bmatrix} 0.4 & 0.36 & 0.24 \end{bmatrix} \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.4 & 0 & 0.6 \\ 0.5 & 0.5 & 0 \end{bmatrix} = \begin{bmatrix} 0.344 & 0.44 & 0.216 \end{bmatrix}$$

To obtain the two-step transition matrix, we calculate

$$P^2 = \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.4 & 0 & 0.6 \\ 0.5 & 0.5 & 0 \end{bmatrix} \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.4 & 0 & 0.6 \\ 0.5 & 0.5 & 0 \end{bmatrix} = \begin{bmatrix} 0.36 & 0.16 & 0.48 \\ 0.38 & 0.62 & 0 \\ 0.3 & 0.4 & 0.3 \end{bmatrix}$$

Thus, for example, the probability of going from State 3 to State 1 in two steps is given by the 3,1-entry in P^2, namely 0.3.

The steady state distribution is given by

$$v_{ss}P = v_{ss} \quad \longrightarrow \quad \begin{bmatrix} v_x & v_y & v_z \end{bmatrix} \begin{bmatrix} 0.2 & 0.8 & 0 \\ 0.4 & 0 & 0.6 \\ 0.5 & 0.5 & 0 \end{bmatrix} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}$$

$$v_x + v_y + v_z = 1$$

$$\downarrow$$

$$0.2v_x + 0.4v_y + 0.5v_z = v_x$$

$$0.8v_x + 0.5v_z = v_y$$

$$0.6v_y = v_z$$

$$v_x + v_y + v_z = 1$$

$$\downarrow$$

$$v_{ss} = \begin{bmatrix} 0.354 & 0.404 & 0.242 \end{bmatrix}$$

steady state distribution

$$P_\infty = \begin{bmatrix} 0.354 & 0.404 & 0.242 \\ 0.354 & 0.404 & 0.242 \\ 0.354 & 0.404 & 0.242 \end{bmatrix}$$

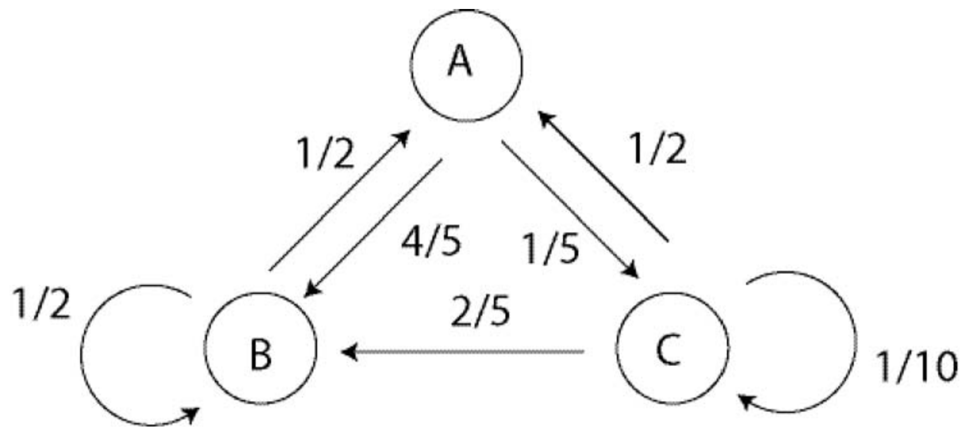
Digram probabilities

What are the relative frequencies of the combination of symbols ij=AA,AB,AC… (digram)? What is the joint probability p(i,j)?

$$p(i,j)=p(i)p_i(j)$$

i	p(i)
A	$\frac{9}{27}$
B	$\frac{16}{27}$
C	$\frac{2}{27}$

$p_i(j)$		j: A	j: B	j: C
i	A	0	$\frac{4}{5}$	$\frac{1}{5}$
i	B	$\frac{1}{2}$	$\frac{1}{2}$	0
i	C	$\frac{1}{2}$	$\frac{2}{5}$	$\frac{1}{10}$



$p(i,j)$		j: A	j: B	j: C
i	A	0	$\frac{4}{15}$	$\frac{1}{15}$
i	B	$\frac{8}{27}$	$\frac{8}{27}$	0
i	C	$\frac{1}{27}$	$\frac{4}{135}$	$\frac{1}{135}$

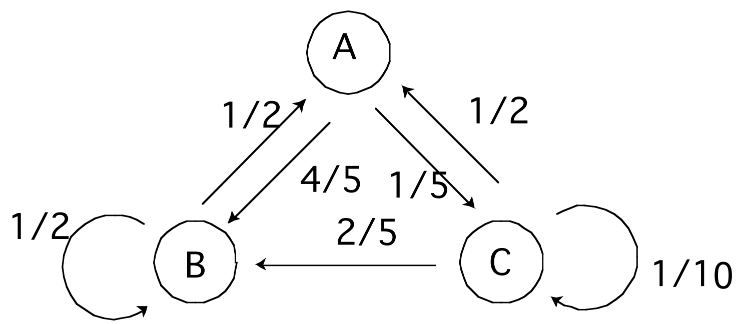
p(A,A)=p(B,C)=0; AA, BC never occurs

p(B,A) occurs most often; 4/15 times

Shannon & Weaver pg.41

Why are we interested in Markov Models?

We can represent an information source as an engine creating symbols at some rate according to probabilistic rules. The Markov model represents those rules as transition probabilities between symbols.



ACBBA…

In the long term, each symbol has a certain steady state probability.

$$v_{ss} = \begin{bmatrix} \frac{1}{3} & \frac{16}{27} & \frac{2}{27} \end{bmatrix}$$

Based on these probabilities, we can define the amount of information, I, that a symbol carries and what the average rate of information or entropy, H, a system generates.

Information

We would like to develop a usable measure of the information we get from observing the occurrence of an event having probability p . Our first reduction will be to ignore any particular features of the event, and only observe whether or not it happened. In essence this means that we can think of the event as the observance of a symbol whose probability of occurring is p. We will thus be defining the information in terms of the probability p.

Information

We will want our information measure I(p) to have several properties:

1. Information is a non-negative quantity: $I(p) \leq 0$.

2. If an event has probability 1, we get no information from the occurrence of the event:
   $I(1) = 0$.

[information is surprise, freedom of choice, uncertainty…]

3. If two independent events occur (whose joint probability is the product of their individual probabilities), then the information we get from observing the events is the sum of the two informations:
   $I(p1 \bullet p2) = I(p1) + I(p2)$. (This is the critical property . . .)

4. We will want our information measure to be a continuous (and, in fact, monotonic) function of the probability (slight changes in probability should result in slight changes in information).

An introduction to information theory and entropy-- Tom Carter

Information

$I(p) = \log_b(1/p) = -\log_b (p)$,

$x=y^n \rightarrow \log_y(x)=n$

for some positive constant b. The base b determines the units we are using.
$\log_2$ units of I are bits

$\log_2(x)=\log_{10}(x)/\log_{10}(2)$

Ex. Flip a fair coin (p_H=0.5, p_T=0.5)

1 flip: H or T

$I= -\log_2(p)= -\log_2(0.5)= \log_2(2)=1$ bit

n flips: HTTH…n times

$I= -\log_2(p\ p\ p\ p\ \ldots)= -\log_2(p^n)$

$= -n\log_2(p) = n\log_2(1/p)= =n \log_2(2)$

= n bits

Additive property

$-\log_2(p_1p_2)= -\log_2(p_1) -\log_2(p_2)$

I_1 and I_2= I_1+ I_2

Also think of switches
1 switch = 1 bit (2^1=2 possibilities)
3 switches = 3bits (2^3 =8 possibilities)

Ex. Flip an unfair coin (p_H=0.3, p_T=0.7)

1 flip: H

$I= -\log_2(p_H)= -\log_2(0.3)= 1.737$ bit

less likely, more info

more likely, less info

1 flip: T

$I= -\log_2(p_T)= -\log_2(0.7)= 0.515$ bit

5 flips: HTTHT

$I= -\log_2(p_H\, p_T\, p_T\, p_H\, p_T)$

$= -\log_2(0.3 \bullet 0.7 \bullet 0.7 \bullet 0.3 \bullet 0.7) = -\log_2 (0.031)$

$= 5.018$ bits

1.004 bits/flip

5 flips: THTTT

$I= -\log_2(p_T\, p_H\, p_T\, p_T\, p_T)$

$= -\log_2(0.7 \bullet 0.3 \bullet 0.7 \bullet 0.7 \bullet 0.7) = -\log_2 (0.072)$

$= 3.795$ bits

0.759 bits/flip

Entropy

Ex. Flip an unfair coin (p_H=0.3, p_T=0.7)

1 flip:

I_H= 1.737 bits,

I_T= 0.515 bits

So what's the average bits/flip for n flips as $n \to \infty$?

Use a weighted average based on probability of information per flip.

Call this average information/flip, Entropy H

$$H = p_H\, I_H + p_T I_T$$
$$= p_H\,[-\log_2(p_H)] + p_T[-\log_2(p_T)]$$

$$= 0.3(1.737 \text{ bits}) + 0.7(0.515 \text{ bits})$$
$$= 0.822 \text{ bits}$$

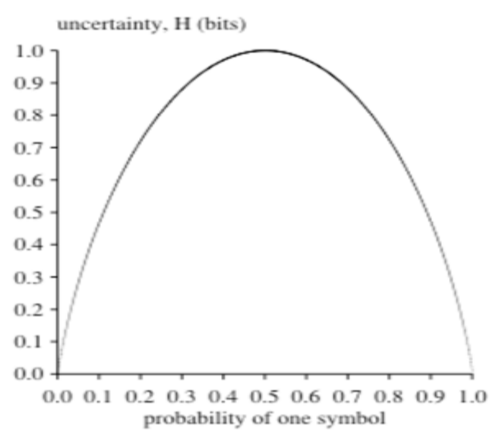
Entropy

Average information/symbol called Entropy H

$$H = -\sum_i p_i \log(p_i)$$

H(X,Y)=H(X)+H(Y)   H also obeys additive property if events are independent.

For unfair coin, p_H=p, p_T=(1-p)



The average information per symbol is greatest when the symbols equiprobable.

Balance

Suppose you have eight billiard balls. One of them is defective -- it weighs more than the others. How do you tell, using a balance, which ball is defective in two weighings?

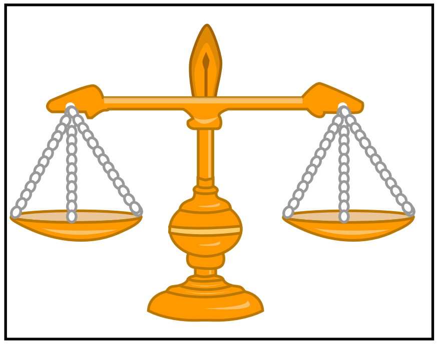
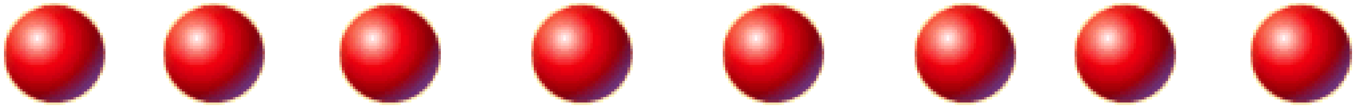


Figure by MIT OpenCourseWare.



Only allowed 2 weighings.

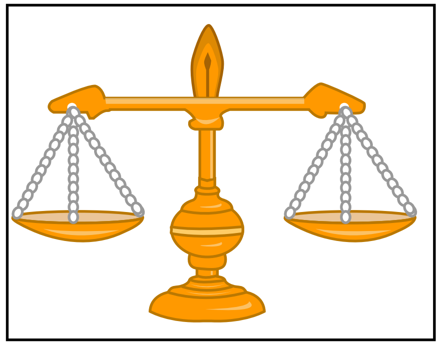


Figure by MIT OpenCourseWare.

Wrong way
50/50 split

Weighing #1

1 2 3 4 5 6 7 8

Heavy

Weighing #2

1 2 3 4

Heavy

Which is heavier, 1 or 2?

Wrong way
50/50 split

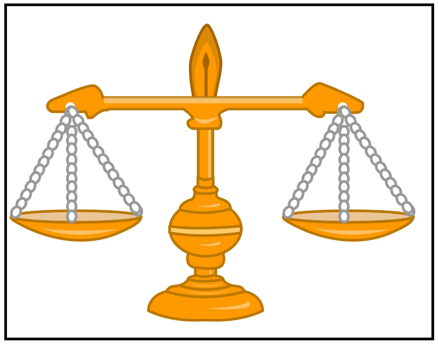


Figure by MIT OpenCourseWare.

Weighing #1

Only allowed 2 weighings.

Weighing #2

Which is heavier, 1 or 2?

Wrong way
50/50 split

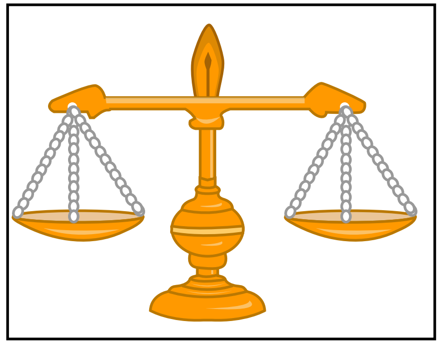


Figure by MIT OpenCourseWare.

Balance has 3 states
Heavy L, Heavy R, Equal

50/50 split doesn't let all
3 states be equally probable

Weighing #1

1 2 3 4 5 6 7 8

HeavyL

Weighing #2

1 2 3 4

HeavyL

Which is heavier, 1 or 2?

Optimal way
~1/3,~1/3,~1/3 split

Now, HL,HR,B
Almost equiprobable

Case #1

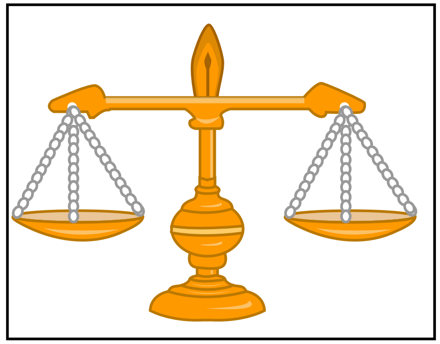


Figure by MIT OpenCourseWare.

Only allowed 2 weighings.

Weighing #1
(1,2,3) vs. (4,5,6)

1 2 3 4 5 6 7 8

HeavyL

Weighing #2
1 vs 2

1 2 3

HeavyL

2 is the odd ball

Optimal way
~1/3,~1/3,~1/3 split

Now, HL,HR,B
Almost equiprobable

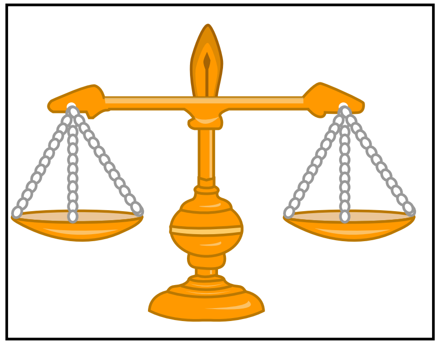


Figure by MIT OpenCourseWare.

Case #1b

Weighing #1
(1,2,3) vs. (4,5,6)

Only allowed 2 weighings.

1 2 3

4 5 6

7
8

HeavyL

Weighing #2
1 vs. 2

1 2 3

Balanced

3 is the odd ball

Optimal way
~1/3,~1/3,~1/3 split

Now, HL,HR,B
Almost equiprobable

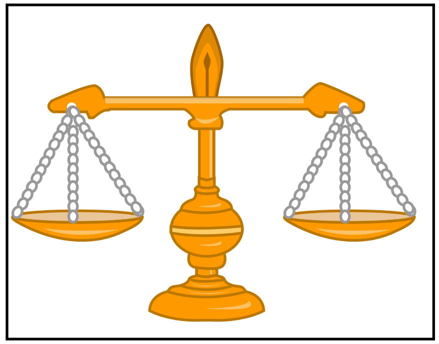
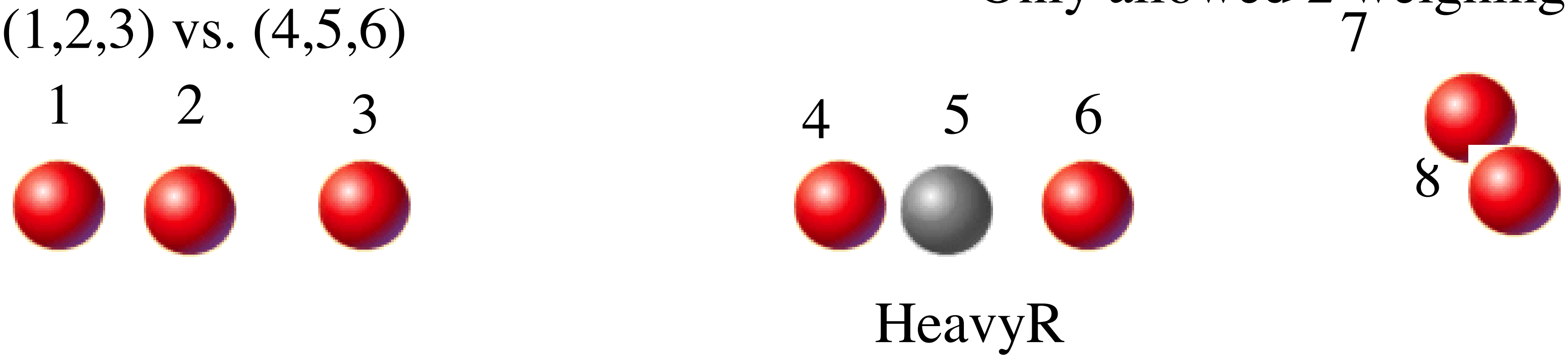


Figure by MIT OpenCourseWare.

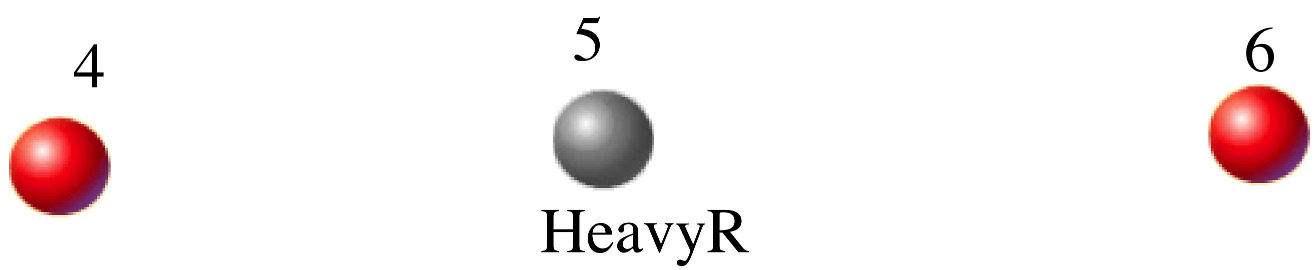
Case #2

Only allowed 2 weighings.

Weighing #1
(1,2,3) vs. (4,5,6)



Weighing #2
4 vs. 5



5 is the odd ball

Optimal way
~1/3,~1/3,~1/3 split

Case #3

Now, HL,HR,B
Almost equiprobable

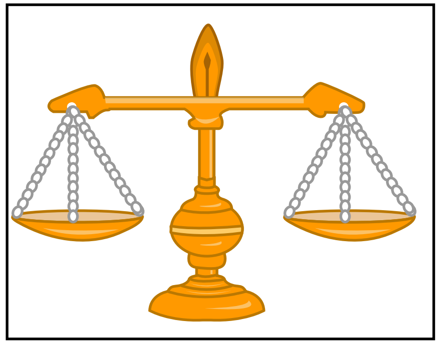


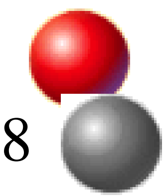
Figure by MIT OpenCourseWare.

Weighing #1
(1,2,3) vs. (4,5,6)

Only allowed 2 weighings.

1 2 3 4 5 6

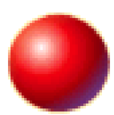
7
8



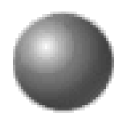
Balanced

Weighing #2
7 vs. 8

7



8



HeavyR

8 is the odd ball

Optimal way
~1/3,~1/3,~1/3 split

Case #3

Now, HL,HR,B
Almost equiprobable

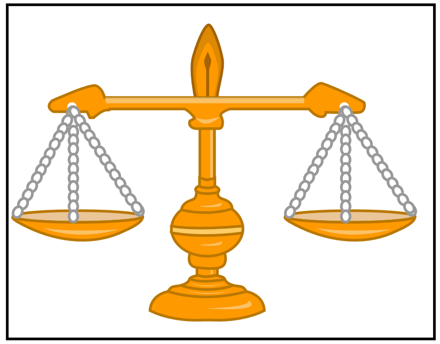


Figure by MIT OpenCourseWare.

Weighing #1
(1,2,3) vs (4,5,6)

Only allowed 2 weighings.

7

1 2 3 Balanced 4 5 6

8

Weighing #2
7 vs 8

7 8 Heavy

8 is the odd ball

Try to design your experiments to maximize the information extracted from each measurement by making possible outcomes equally probable.

Compression

Shannon Fano

ZIP implosion algorithm uses this

Split symbols so probabilities halved

Ex. "How much wood would a woodchuck chuck" 31 characters

ASCII 7bits/character, so 217 bits

Frequency chart

o	0.194
c	0.161
h	0.129
w	0.129
u	0.129
d	0.097
k	0.065
m	0.032
a	0.032
l	0.032

$$H = -\sum_i p_i \log(p_i)$$

$H=-(0.194 \log_2 0.194 + 0.161 \log_2 0.161 + \ldots)$

$H=2.706$ bits/symbol, so 83.7 bits for sentence

o has $-\log_2 0.194=2.37$ bits of information

l has $-\log_2 0.032= 4.97$ bits of information

The rare letters carry more information

Compression

Shannon Fano

Split symbols so probabilities halved (or as close as possible)

Ex. "How much wood would a woodchuck chuck" 31 characters

Frequency chart

o	0.194
c	0.161
h	0.129
w	0.129
u	0.129
d	0.097
k	0.065
m	0.032
a	0.032
l	0.032

0.484
0.516

0.194
0.29

0.258
0.258

0.161
0.129

0.129
0.129

0.097
0.161

0.065
0.096

0.032
0.064

0.032
0.032

Compression

Shannon Fano

Split symbols so probabilities halved

Ex. “How much wood would a woodchuck chuck”

31 characters

Frequency chart

o	0.194		11					
c	0.161			101				
h	0.129	1	10	100				
w	0.129			011				
u	0.129	0	01	010				
d	0.097		00	001				
k	0.065			000	0001			
m	0.032				0000	00001		
a	0.032					00000	000001	
l	0.032						000000	

Compression

Shannon Fano

Split symbols so probabilities halved

Ex. “How much wood would a woodchuck chuck” 31 characters

“Prefix free - one code is never the start of another code”

Frequency chart

11	o	0.194
101	c	0.161
100	h	0.129
011	w	0.129
010	u	0.129
001	d	0.097
0001	k	0.065
00001	m	0.032
000001	a	0.032
000000	l	0.032

1

0

11

10

01

00

101

100

011

010

001

000

0001

0000

00001

00000

000001

000000

Compression

Shannon Fano — Split symbols so probabilities halved

Ex. “How much wood would a woodchuck chuck” — 31 characters

Encoding chart		p	#
11	o	0.194	6
101	c	0.161	5
100	h	0.129	4
011	w	0.129	4
010	u	0.129	4
001	d	0.097	3
0001	k	0.065	2
00001	m	0.032	1
000001	a	0.032	1
000000	l	0.032	1

6(2)+5(3)+4(3)+4(3)+4(3)+
3(3)+2(4)+1(5)+1(6)+1(6)
=97 bits

97bits/31 characters
=3.129 bits/character

H=2.706 bits/symbol

Compression

Shannon Fano

Ex. "How much wood would a woodchuck chuck" 31 characters

Decoding chart

11	o
101	c
100	h
011	w
010	u
001	d
0001	k
00001	m
000001	a
000000	l

10011011000010101011000111111001 32
h o w m u c h w o o d
011110100000000000100000101111111001101 36
w o u l d a w o o d c
10001010100011011000101010001 29
h u c k c h u c k

97bits

Compression

Shannon Fano

0110000010000000001000001000010000010010010101010001

Decoding chart

52bits

11	o
101	c
100	h
011	w
010	u
001	d
0001	k
00001	m
000001	a
000000	l

Compression

Shannon Fano

011	000001	000000	0001	000001	00001	000001	001	001	010	101	0001
w	a	l	k	a	m	a	d	d	u	c	k

Decoding chart

11	o
101	c
100	h
011	w
010	u
001	d
0001	k
00001	m
000001	a
000000	l

52bits/ 12 characters = 4.333 bits/character

greater than before
because character
frequencies are different

Huffman Coding

JPEG, MP3

Add two lowest probabilities
group symbols
Resort
Repeat

Ex. “How much wood would a woodchuck chuck”

Frequency chart

o	0.149	o	0.149	o	0.149	o	0.149
c	0.161	c	0.161	c	0.161	c	0.161
h	0.129	h	0.129	h	0.129	almk	0.161
w	0.129	w	0.129	w	0.129	h	0.129
u	0.129	u	0.129	u	0.129	w	0.129
d	0.097	d	0.097	d	0.097	u	0.129
k	0.065	k	0.065	alm	0.096	d	0.097
m	0.032	al	0.064	k	0.065		
a	0.032	m	0.032				
l	0.032						

Huffman Coding

Ex. "How much wood would a woodchuck chuck"

o	0.194
c	0.161
h	0.129
w	0.129
u	0.129
d	0.097
k	0.065
m	0.032
a	0.032
l	0.032

o	0.194
c	0.161
h	0.129
w	0.129
u	0.129
d	0.097
k	0.065
al	0.064
m	0.032

o	0.194
c	0.161
h	0.129
w	0.129
u	0.129
d	0.097
alm	0.096
k	0.065

o	0.194
c	0.161
almk	0.161
h	0.129
w	0.129
u	0.129
d	0.097

ud	0.226
o	0.194
c	0.161
almk	0.161
h	0.129
w	0.129

hw	0.258
ud	0.226
o	0.194
c	0.161
almk	0.161

almkc	0.322
hw	0.258
ud	0.226
o	0.194

udo	0.420
almkc	0.322
hw	0.258

almkchw	0.580
udo	0.420

almkchwudo 1

Huffman Coding

Ex. "How much wood would a woodchuck chuck"

o
c
h
w
u
d
k
m
a 110111
l 110110

o
c
h
w
u
d
k
al 11011
m 11010

o
c
h
w
u
d
alm 1101
k 1100

o
c 110
almk
h
w
u 011
d 010

ud 01
o
c
almk
h 101
w 100

hw 10
ud
o
c 111
almk 110

almkc 11
hw
ud 01
o 00

udo 0
almkc 11
hw 10

almkchw 1
udo 0

almkchwudo 1

backward pass
assign codes

Huffman Coding

Ex. “How much wood would a woodchuck chuck”

o
c
h
w
u
d
k
m
a 110111
l 110110

o
c
h
w
u
d
k
al
m 11010

o
c
h
w
u
d
alm
k 1100

o
c
almk
h
w
u 011
d 010

ud
o
c
almk
h 101
w 100

hw
ud
o
c 111
almk

almkc
hw
ud
o 00

udo
almkc
hw

almkchw
udo

almkchwudo 1

find codes for single letters

Huffman Coding

Ex. “How much wood would a woodchuck chuck”

Huffman

o	00	6
c	111	5
h	100	4
w	101	4
u	011	4
d	010	3
k	1100	2
m	11010	1
a	110111	1
l	110110	1

6(2)+5(3)+4(3)+4(3)+4(3)+
3(3)+2(4)+1(5)+1(6)+1(6)
=97 bits

97bits/31 characters
=3.129 bits/character

Shanon Fano

11	o
101	c
100	h
011	w
010	u
001	d
0001	k
00001	m
000001	a
000000	l

97bits/31 characters
=3.129 bits/character

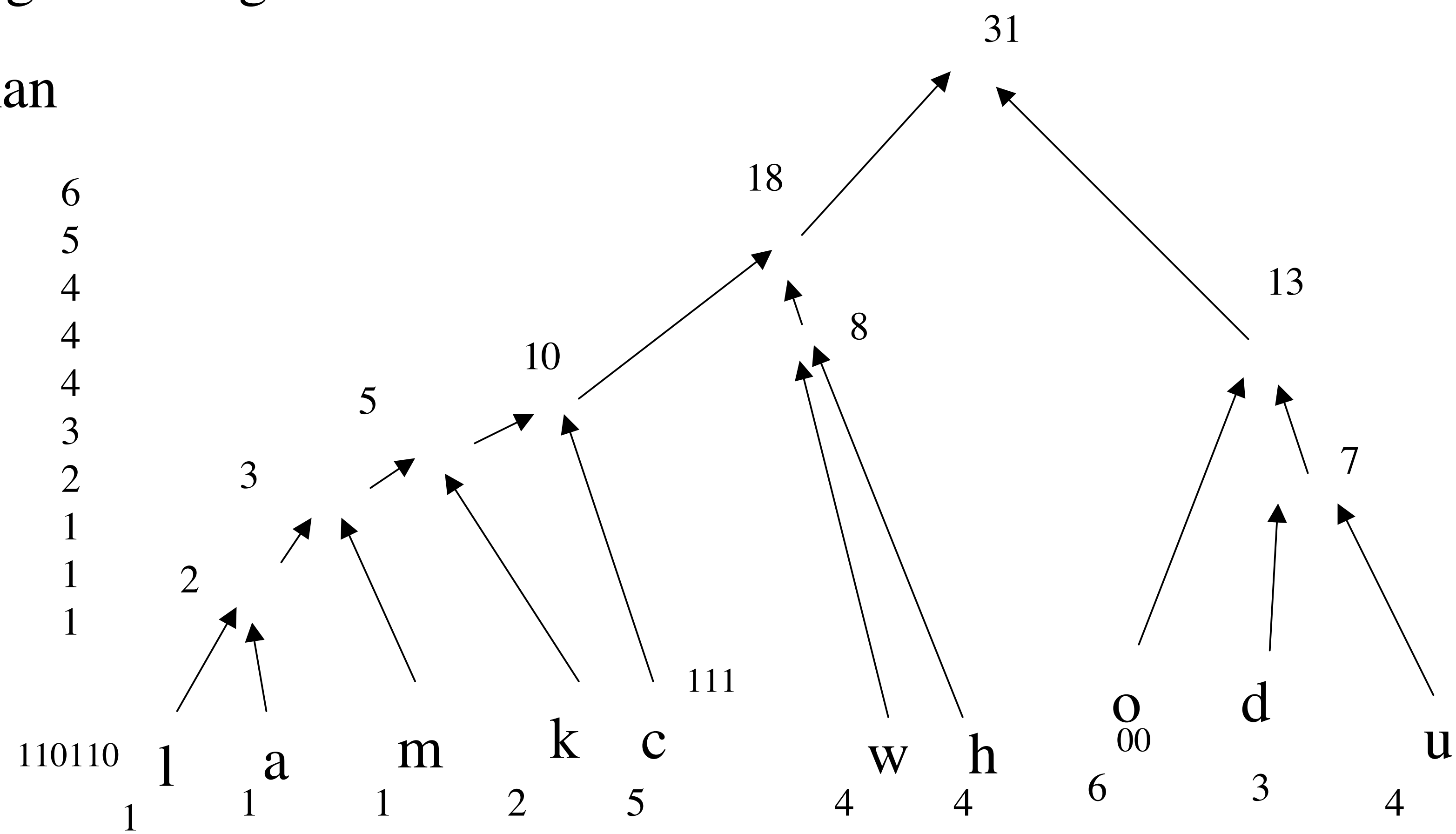
Notice o has 2 bits;
a,l have 6 bits

H=2.706 bits/symbol

Huffman's algorithm is a method for building an extended binary tree of with a minimum weighted path length from a set of given weights.

Huffman

o	00	6
c	111	5
h	100	4
w	101	4
u	011	4
d	010	3
k	1100	2
m	11010	1
a	110111	1
l	110110	1

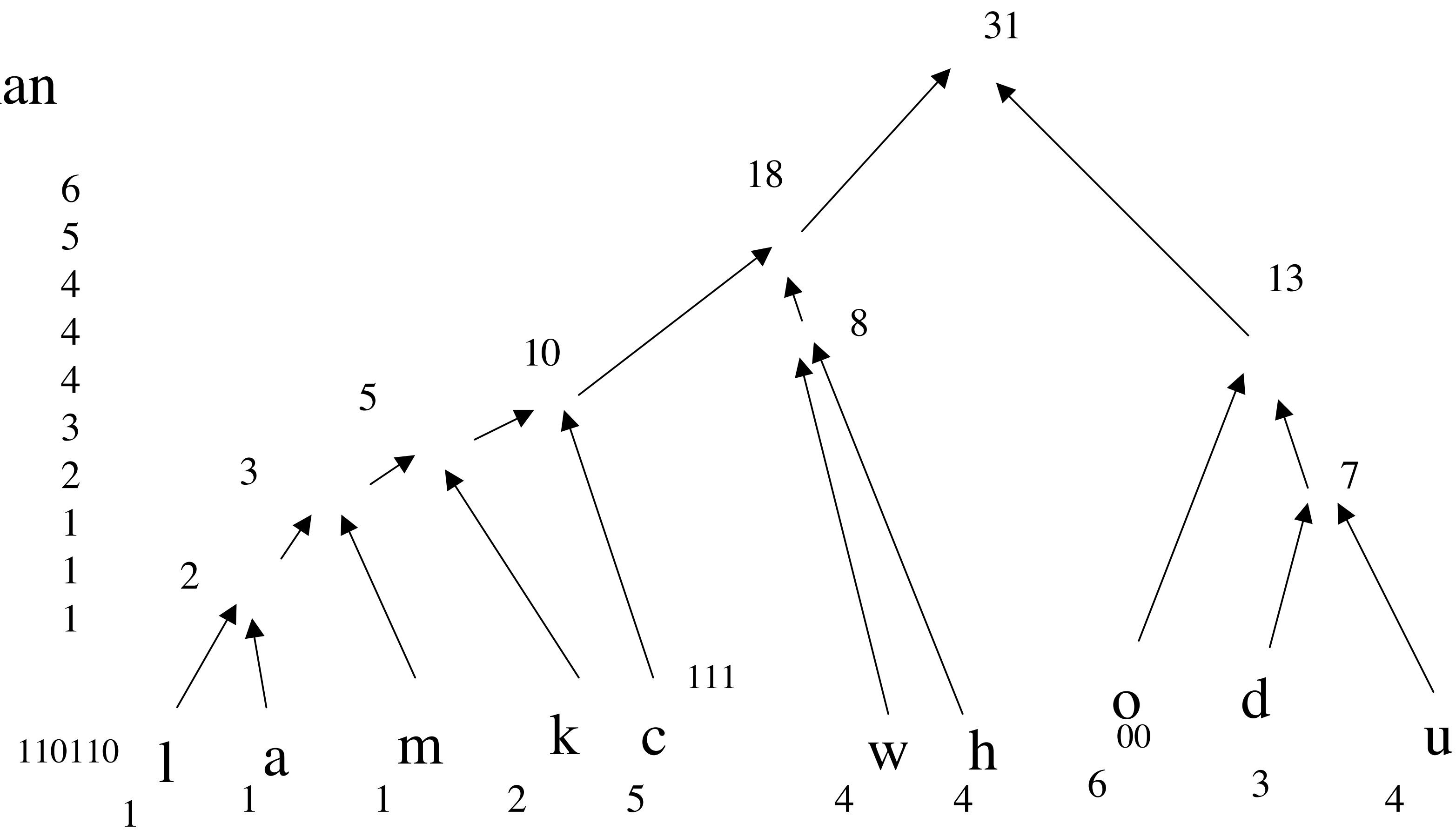


Frequencies*(edges to root)= weighted path length

Huffman's algorithm is a method for building an extended binary tree of with a minimum weighted path length from a set of given weights.

Huffman

o	00	6
c	111	5
h	100	4
w	101	4
u	011	4
d	010	3
k	1100	2
m	11010	1
a	110111	1
l	110110	1



Each branch adds a bit. Minimize (#branches * frequency)
Least frequent symbol further away. More frequent, closer.

MP-3

Huffman coding is used in the final step of creating an MP3 file. The MP3 format uses frames of 1152 sample values. If the sample rate is 44.1kHz, the time that each frame represents is ~26ms. The spectrum of this 1152-sample frame is spectrally analyzed and the frequencies are grouped in 32 channels (critical bands). The masking effects within a band are analyzed based on a psycho-acoustical model. This model determines the tone-like or noise-like nature of the masking in each channel and then decides the effect of each channel on its neighboring bands. The masking information for all of the channels in the frame is recombined into a time varying signal. This signal is numerically different from the original signal but the difference is hardly noticeable aurally. The signal for the frame is then Huffman coded. A sequence of these frames makes up an MP3 file.

http://webphysics.davidson.edu/faculty/dmb/py115/huffman_coding.htm