1. Excercise 5.7 (a), (b) and (c)

(a) $e(A \ldots B, k) \rightarrow$ 1st interval in $A \ldots B$ ending at or after k

```
f(A...B, K) ≡
    if K == ∞ then
        return [∞, ∞]
    if K == -∞ then
        return [-∞, -∞]
    [u,v] ← f(A, K) .... 1st interval in A ending at/after K
    if [u,v] == [∞,∞] then
        return [∞, ∞]
    [u', v'] ← f(B, v+1)... 1st interval in B ending at/after v+1
     if [u,v] == [∞, ∞] then
        return [∞, ∞]
    [u'', v''] ← f' [A, u'-1]
    return [u'', v']
```

Scanned with CamScanner

(b) $\tau(A \Delta B : K)$

(First interval in $A \Delta B$ starting at/after K

$\Rightarrow$ $\tau(A \Delta B, K) \equiv$

```
if K==∞ then
    return [∞,∞]
if K== -∞ then
    return [-∞,∞]
[u,v) ← τ(A,K) ... (get 1st interval in A
if [u,v] == [∞,∞] then  starting at/after K)
    return [∞,∞]
~~if v'>v~~
(u',v'] ← τ(B,K) ... get 1st interval in B
                    starting at/after K)
//Now we find then interval
 having both the intervals
if u'<u && v'<v then   // overlap case-1
   return [u',v]
if u>u' && v>v' then   // overlap case-2
   return [u,v']
if u>u' && v'>v then   // [u',v'] ∈ [u,v]
   return [u,v]
if v'>v && u'>v then   // disjoint intervals.
   return τ(A Δ B, v)
```

Scanned with CamScanner

(c) $\tau(A \nabla B, K)$

(First interval in $A \nabla B$ starting at/after K

$\tau(A \nabla B, K) \equiv$

```
if K==∞ then
    return [∞,∞)
if K== -∞ then
    return [-∞,-∞]
[u,v] ← τ(A,K)
if [u,v)==[∞,∞) then
    return [∞,∞]
[u',v'] ← τ(B,K)
// Now we find the first interval
having either A or B's interval
if u<u' && v'>v then  // [u',v'] ∈ [u,v]
    return [u,v)
if u<u' && v<v' then // overlap case 1
    return [u,v]
if u'<u && v'<v then // overlap case 2
    return [u',v']
if u'<u and v'<u' then
    return [u',v'].
```

Scanned with CamScanner

2. Pr["a"] = 0.55 & Pr["b"] = 0.45

When we group symbols in block of size 2, we get

Pr["aa"] = Pr["a"] x Pr["a"] = 0.55 x 0.55 = 0.3025

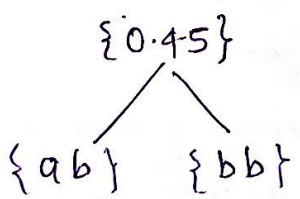
Pr["ab"] = Pr["a"] x Pr["b"] = 0.55 x 0.45 = 0.2475

Pr["ba"] = Pr["b"] x Pr["a"] = 0.45 x 0.55 = 0.2475

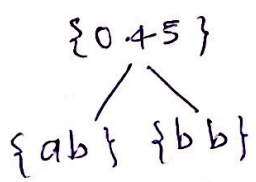
Pr["bb"] = Pr["b"] x Pr["b"] = 0.45 x 0.45 = 0.2025

to build huffman tree using above distribution we get,

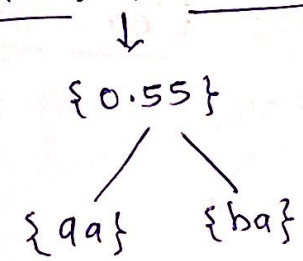
Step1: {"bb", 0.2025}, {"ab", 0.2475}, {"ba", 0.2475}, {"aa", 0.3025}



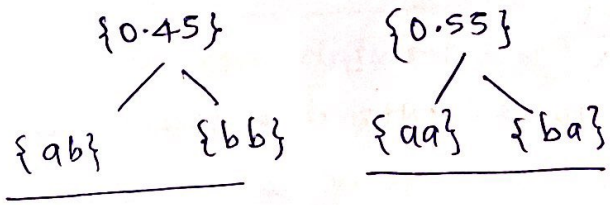
Step2:

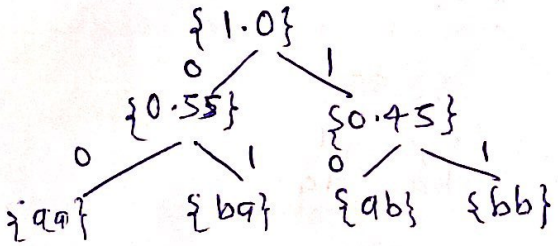


{"ba", 0.2475}, {"aa", 0.3025}



Step3:





Huffman code for aa = 00, ba = 01, ab = 10, bb = 11

Scanned with CamScanner

3. Exercise 6.2 - show that the γ code is prefix-free

Solution - γ code is a non parametric gap compression code used to compress the posting list under the assumption that Gaps between entries are relatively short.

A γ-code has two parts a selector and body.

If we want to encode 'k' in γ code we need $2\lfloor \log_2(K) \rfloor + 1$ bits i.e, $\log_2(K)+1$ bits for selector and $\log_2(K)$ bits for the body.

To prove that γ code is prefix free we consider following 2 cases.

(a) when length of selectors is same

when we have 2 messages (Gap numbers) which need to be converted into γ code and having same length for selector field, we can assure that they are prefix free because the body field of those 2 numbers will be different.

Eg: K1 = 6 → 001 10
K2 = 7 → 001 11
} Here even when the selector is of same size γ code is prefix free because body is different for every number.

(b) when length of selectors is not same

When selectors have different lengths, bodies of those codes will also differ in their respective lengths. selectors for rice codes are themself prefix free making the γ-code prefix free as a whole

Eg: K1 = 17 → 00001 0001
K2 = 11 → 0001 011
} Here, selector field for both K1 and K2 is different and prefix free.

Eg: K1 = 4 → 001 00
K2 = 12 → 0001 100
} As the selector increases in size, it follows prefix free property making γ-code prefix free.

Scanned with CamScanner

4. What is the expected number of bits per codeword when using a rice code with parameter $M=2^6$ to compress a geometrically distributed posting list for term T with $N_T/N=0.05$?

Solution:

$$|E| = \sum_{K=1}^{\infty} Pr[\Delta=K] \cdot L(\Delta=K) \quad \text{------ (1)}$$

where, $|E|$ - expected no. of bits/codeword

$Pr[\Delta=K]$ - geometric distribution with $\Delta=K$

$L(\Delta=K)$ - Length of codeword with gap K ($\Delta=K$)

we know that,

$$Pr[\Delta=K] = (1-p)^{K-1} \cdot p$$

since we know the distribution is $N_T/N = p = 0.05$

$$Pr[\Delta=K] = (1-N_T/N)^{K-1} \cdot N_T/N$$

$$= (1-0.05)^{K-1} \cdot 0.05$$

$$Pr[\Delta=K] = (0.95)^{K-1} \cdot 0.05 \quad \text{------ (2)}$$

and,

$$L(\Delta=K) = \left\lfloor \frac{K-1}{M} \right\rfloor + 1 + \lfloor \log_2(M) \rfloor$$

$$= \left\lfloor \frac{K-1}{2^6} \right\rfloor + 1 + \lfloor \log_2(2^6) \rfloor$$

$$L(\Delta=K) = 7 + \left\lfloor \frac{K-1}{2^6} \right\rfloor \quad \text{------ (3)}$$

Substituting (2) and (3) in original equation (1) we get,

$$|E| = \sum_{K=1}^{\infty} 0.05(0.95)^{K-1} \cdot \left(7 + \left\lfloor \frac{K-1}{2^6} \right\rfloor\right)$$

$$= 0.35 \sum_{K=1}^{\infty} (0.95)^{K-1} + 0.00078125 \cdot \sum_{K=1}^{\infty} (K-1)(0.95)^{K-1}$$

Scanned with CamScanner

$$|E| = 0.35 \sum_{k=1}^{\infty} (0.95)^{k-1} + 7.8125 \times 10^{-4} \sum_{k=1}^{\infty} (0.95)^{k-1} (k-1)$$

$$= 0.35 \cdot \left(\frac{1}{1-x}\right) + 7.8125 \times 10^{-4} \cdot \frac{d}{dx}\left(\frac{1}{1-x}\right)$$

where $x = 0.95$ &

$$\sum_{k=1}^{\infty} x^k = \frac{1}{1-x}$$

$$= \frac{0.35}{1-0.95} + \frac{7.8125 \times 10^{-4}}{(1-0.95)^2}$$

$$|E| = 7.3125$$

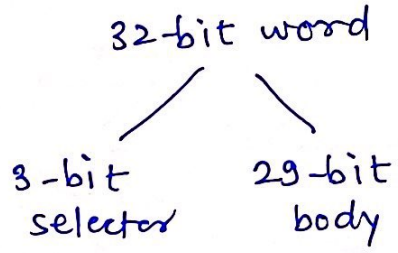
Thus, expected no. of bits per codeword is 7.3125 bits.

(Reference - Homework 4 (2012))

Scanned with CamScanner

5. Come up with a variation on simple-9 that uses only a 3-bit selector rather than 4 bit selector.

Solution - If we use simple-9 with a 3 bit selector on Δ-values with 32-bit integer format, we get 29 bits for the body and 3 bits for selector



Following table shows all possible ways to divide the 29 bit body into chunks of equal size

selector	0	1	2	3	4	5	6	7
Number of Δ's	1	2	3	4	7	9	14	29
Bits per Δ	29	14	9	7	4	3	2	1
Unused bits/word	0	1	2	1	1	2	1	0

Case when above distribution is better -

As, shown in the table simple-9 with 3 bit selector can encode 29 1-bit numbers in a single word (32-bit) as compared to simple-9 with 4 bit selector which can only accomodate 28 1-bit numbers.

Case when simple-9 is better:

If a Gap distribution has varying integers, (i.e, more number of selectors are useful), then simple-9 with 4-bit selector would be better than simple-9 with 3-bit selector

Scanned with CamScanner