Huffman Coding

Also known as Minimum Redundancy code or Optimum code. This is the procedure for obtaining a compact code with least redundancy. Steps are:

Step 1: Source symbols are listed in non-increasing order of probabilities.

Step 2: Consider the equation $q = r + (r-1)\alpha$ where

q = number of source symbols and

r = number of different symbols used in the code alphabet.

from the equation calculate the quantity 'x' and it should be an integer. If it is not, then "dummy symbols" with zero probabilities are added to 'q', to make x an integer.

For binary codes, x will always be an integer. So this step is not need for Huffman binary codes.

Step 3: The last 'r' symbols of source "S" are combined into a "single composite symbol" by adding their probabilities to get a reduced source "S_a". Then the symbols of "S_a" are arranged in the non increasing order.

Step 4: The last "r" symbols of "S_a" are combined to form another composite symbol by adding their probabilities to get a reduced source "S_b". Then the symbols of "S_b" are arranged in the non increasing order.

Step 5: The process is continued till we arrive a last source having 'r' symbols (if 'x' is an integer, then this condition will automatically met).

Step 6: The last source with 'r' symbols are now encoded with 'r' different code symbols ie 0, 1, 2 .... (r-1).

Step 7: As we "comming backwards" we will decompose the codeword, depending on the level which have been combined to get the last reduced source symbol.

In binary coding, the last two symbols are encoded with '0' and '1'. As commeing backwards either '0' may be decomposed as '00' and '01' or '1' may be as '10' and '11' depending on the level we have combined.