Encoding Sets of Symbols in Binary Logic

Abyan Majid

July 6, 2023

1 Encoding sets of symbols

To encode a set of symbols you must:

1. Derive n, which is the minimum amount of bits to encode all symbols in the set.

   $$n \text{ bits} \rightarrow 2^n \text{ combinations}$$

   Recall that given a set S, $|S|$ is the cardinality of the set and is a measure of how much elements are in the set. So, you should get a value of n that allows a number of combinations that is greater than or equal to $|S|$, the number of elements.

   $$2^n \geq |S|$$

   Now, to get n, you make use of the logarithmic identity: $\log_a b = c \longleftrightarrow a^c = b$. In our context, we can specifically do (and this is what you should remember):

   $$\boxed{n = \lceil \log_2 |S| \rceil}$$

   Here, $\log_2 |S|$ gives the "exact" number of bits needed for encoding S. But, since it's invalid to have a floating-point number of bits, we use the ceiling function "$\lceil x \rceil$" to always round up to the nearest integer.

   FOR EXAMPLE: Suppose we have
   $CLASS = \{Storm, Fire, Ice, Death, Life, Myth, Balance\}$. Given that $|CLASS| = 7$, we have:

   $$n = \lceil log_2(7) \rceil = \boxed{3}$$

   We know that "3" is indeed the minimum amount of bits needed to encode the symbols of $CLASS$ because:

   $$2^3 = 8 \geq |S| = 7$$

2. Assign a unique binary of n bits for each symbol where necessary.

This step should be pretty easy. Following from our previous example, we can assign a unique 3-bit binary for each symbol in $CLASS$. Of course there is no right order, but here's one way to do it:

Symbol	3-Bit Binary
Storm	000
Fire	001
Ice	010
Death	011
Life	100
Myth	101
Balance	110

Since a 3-bit binary system has 8 combinations (2^3), and $CLASS$ has only 7 elements, we do not have any symbol to assign the leftover binary (i.e. 111) - and that's perfectly okay!