all_comb(tval, begin+1, n) will be called, begin = −1, and we know that our first call to all_comb has begin = 0. This is a contradiction.

∴ In both cases, the function all_comb(tval, begin, n), where begin = 0 is called 1 time, where $1 = 2^0 = 2^{begin}$. This is correct (Base case holds)

$\boxed{\text{I.H:}}$ For a particular begin = K, where begin < n, the function all_comb on the parameters (tval, begin, n) is called 2^{begin} times.

$\boxed{\text{Induction Step:}}$ Now, for the value begin = K+1. The if block from lines 4–9 doesn't call any instance of the function all_comb. Only the else block does that on lines 15 and 17. If on the current call, begin = K+1, then it's easy to see that in the previous call, begin = K.

For each case, when begin = K, the function all_comb on the parameters (tval, K+1, n) is called twice, i.e. on lines 15, 17.

By I.H, the function all_comb() on the parameters (tval, K, n) is called 2^K times.

∴ The total no. of times the function all_comb on the parameters (tval, K+1, n) is called $= 2 \cdot 2^K = 2^{K+1} = 2^{begin}$. □

∴ We have shown for a fixed n, and ∀ begin < n, begin ≥ 0, the function all_comb(tval, begin, n) is called 2^{begin} times.

Now, we can do a similar proof when the no. of variables is n+1, and from here, our original claim follows. □

∴ Total no. of function calls: $2^0 + 2^1 + 2^2 + \ldots\ldots + 2^n = (2^{n+1} - 1)$

We leave aside the original function call. ∴ Now it's $(2^{n+1} - 2)$
$= 2(2^n - 1)$