1 Boolean Functions \((10+10+15 = 35 \text{ points})\)

1.1 Unate Functions

A Boolean function \(f(x_1, \ldots, x_n)\) is said to be positive unate in input variable \(x_i\) if changing \(x_i\) from 0 to 1, while keeping the other inputs constant at any of their possible values, can never make \(f\) go from 1 to 0. In other words,

\[
f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) \geq f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \\
\forall \ x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n
\]

Similarly, a Boolean function is negative unate in input variable \(x_i\) if changing \(x_i\) from 0 to 1 can never make \(f\) go from 0 to 1. A function that is neither positive unate nor negative unate in an input variable is said to be binate in that variable. A function that is unate in all its input variables is called a unate function.

Determine whether the following functions are positive unate, negative unate, or binate in each of their input variables. Justify your answers.

10 points. The threshold function: \(\text{Thresh}_k(x_1, x_2, \ldots, x_n) = 1\) if and only if at least \(k\) of the inputs, \(i.e., x_1, x_2, \ldots, x_n\) are 1. Note that \(k\) is a constant, \(i.e.,\) it is not an input to the function.

10 points. The parity function: \(\text{parity}(x_1; x_2, \ldots, x_n) = 1\) if and only if an odd number of the inputs are 1.

1.2 Symmetric Functions

A symmetric Boolean function is a Boolean function whose value does not change under any permutation of the input values, \(i.e.,\) it only depends on the number of inputs that assume values of 0 and 1. The threshold and parity functions given in the previous question are examples of symmetric functions.

**Question (15 points):** It is well known that there are \(2^n\) distinct Boolean functions with \(n\) inputs and 1 output. How many distinct symmetric Boolean functions exist with \(n\) inputs and 1 output? Justify your answer.
2 Logic minimization (10+15=25 points)

Boolean functions may be represented as sum-of-products expressions, and implemented using two-level logic. In two-level logic minimization, the objective is to derive a minimum sum-of-products expression, or one that contains the minimum number of product terms, for a given Boolean function. For the special case of unate functions (defined in the previous section), the process of deriving a minimum sum-of-products expression is greatly simplified. Any sum-of-products expression that satisfies the following two properties is a minimum expression:

1. All variables appear in the expression only in the polarity of unateness (uncomplemented, if the function is positive unate in the variable; complemented, if the function is negative unate in the variable).
2. The expression does not have any redundant product terms, i.e., deleting any of the product terms changes the function that is computed by the expression.

Answer the following questions:

10 points. Show that the following function is unate in all its input variables:
\[ f(a, b, c, d, e) = ab + ace + bcd + abce + cde + b'e + a'bc 

15 points. Derive a sum-of-products expression for this function that satisfies the two properties given above, and is therefore minimum.

3 Timing (20+20=40 points)

A logic circuit’s clock period is determined by the longest time or delay that the combinational logic can take to produce its outputs once its inputs are applied. It is common to estimate the delay of a combinational circuit as the length of its longest path(s). However, in general, this estimate may be pessimistic due to the presence of false paths, or paths that can never affect the circuit’s delay.

Consider the circuit given in Figure 1. Assume that (i) all gates have a delay of 1 unit, and (ii) a path’s delay is the sum of the delays of the gates along the path. Answer the following questions:
20 points. What is the length of the longest path in this circuit? Identify all longest paths.

20 points. For each of the longest paths, state whether or not it is a false path. Justify your answer.