EE219B: Homework 4

Professor Robert K. Brayton

Due February 20 (Tuesday)*

1. Let $f : P_1 \times P_2 \times \ldots \times P_n \rightarrow \{0, 1\}$. Let $\mid P_i \mid = p_i$. Compute the following:

   (a) The maximum number of minterms of f.

   (b) The maximum number of non-null cubes of f.

   (c) The number of cubes in positional notation which represent the null cube.

   (d) The number of possible such functions f.

   Notice that there is more than one cube in the positional notation which represents the null cube. Does that mean that positional notation is redundant? Can one represent multivalued functions adequately by another representation (or encoding) which uses $\log(p_i)$ bits instead of p_i bits per variable? Explain why or why not.

2. Prove that the complement of a strongly-unate function is strongly-unate.

3. For the following functions:

$$\begin{aligned} F &= ab + c \\ G &= bc' + d \\ H &= c + d' \end{aligned}$$

   (a) Draw BDD's F, G, and H with complement edges using the variable ordering

$$a < b < c < d.$$

   (b) Compute *ite*(F, G, H) and show the contents of the unique table and computed table (assuming that the computed table was originally empty).

   (c) Draw the corresponding BDD with complement edges.

   (d) Compute *ite_constant*(F, G, H).

   Note that you have to figure out the details of how to deal with complement edges in the *ite* algorithm.

*TA: Sriram Krishnan (krishnan@eecs.berkeley.edu), Office hours: M5-6.

4. Construct linear time (in the number of BDD nodes) algorithms to count the following:

   (a) The number of disjoint paths to 1 in the BDD for F.

   (b) The number of minterms in the BDD for F (i.e. the size of the onset). Your algorithms should account for complement edges.

5. Variable swapping is the action of interchanging the positions of two variables in a BDD (changing their order). Given a BDD for a function f, prove that swapping any two variables x and y, where x is before y in the original ordering, does not change the structure of the BDD for f above the original position of x and below the original position of y.

6. Totally symmetric functions are characterized by the fact that the value of each such function is determined by the number of variables which are 1; it does not matter which particular variables they are. For example, AND, OR and XOR are totally symmetric functions. A totally symmetric function on n variables can be described by a set $S \subseteq \{0, 1, \ldots, n\}$, where, for minterm $x \in B^n$, $f(x) = 1$ iff the number of ones in x is an element of S. For example, for a four-input XOR gate, $S = \{1, 3\}$.

   (a) The odd parity on n variables is a totally symmetric function such that, for any $x \in B^n$, $f(x) = 1$ iff the number of variables set to 1 in x is odd. Show a general BDD construction for the odd parity function with n variables, under any ordering; how many BDD nodes are required ? Make use of complement pointers and count the terminal node.

   (b) Prove that, for any variable ordering, the BDD for a totally symmetric function on n variables has $O(n^2)$ nodes.

7. Let $|f|$ denote the size of the BDD for $f : B^n \rightarrow B$, under a fixed ordering.

   (a) Let $m \in B^n$ be a minterm. Prove that $|f + m| \leq |f| + n$. In words, adding a minterm to the onset of a function does not increase the size of the BDD for that function by more than n nodes.

   (b) Prove that if the onset of f has size k, then an upper bound on $|f|$ is $k \cdot n + 1$.

8. Consider the family of logic functions $\mathcal{F} = \{f_n\}$, defined as follows. f_n has $2n + \lceil \log n \rceil$ inputs, say $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, m_1, \ldots, m_{\lceil \log n \rceil}$. $f_n = g_{n_i}$ if $value(m_1, \ldots, m_{\lceil \log n \rceil}) = i$, where $value$ is a function that evaluates to integer i on the boolean input $(m_1, \ldots, m_{\lceil \log n \rceil})$ that represents i in binary notation. The function g_{n_i} is defined as:

$$g_{n_i} = \sum_{j=0}^{n-1} (a_j \cdot b_{(j+i) \bmod n}), \quad 0 \leq i < n$$

.

(a) What is the best upper bound that you can find on the size (number of product-terms) for a two-level representation of f_n ?

(b) What is best upper bound that you can find on the size (number of ROBDD nodes) for a ROBDD representation of f_n ?

(c) How do compare the sizes of two-level and ROBDD representations of logic functions ? What does this example suggest ?