CSEE E6861y **Homework 1** Handout 9
Prof. Steven Nowick January 28, 2016

This homework is due at the *beginning of class* on Thursday, February 11.

NOTE: A correct answer without adequate explanation or derivation will have points deducted. To get full credit, (a) write legibly/type, and (b) show all work (label relevant items, show deriviations, include explanations).

1. (30 points) **Introduction to Espresso and Espresso-Exact.** This problem is available online as *Handout #9a*. You should also refer to the web page *Getting Starting with SIS 1.3,* as well as a very useful clickable link on that page for the *Espresso man page*.

   In this problem, you will go over a brief tutorial on running an exact 2-level logic minimizer *espresso-exact* (also known as *mincov*) and a heuristic 2-level logic minimizer *espresso*, and then will perform and analyze some experiments. (Learning the tool and doing the experimental runs should take you 4-5 hours, with several hours of additional time for assembling results and answering some questions.)

2. (14 points) **Quine-McCluskey Method: Step #3.** Using the Quine-McCluskey Method, reduce and solve the prime implicant table (i.e. "constraint matrix") shown below. *Be sure* to follow the steps in the Quine-McCluskey Handout *in order*. Iterate through as many loops as necessary to produce an empty table. (*Note:* you should not need to apply Petrick's method in this example.) Show all work, and *clearly label* each step and iteration – i.e. for each operation, indicate which iteration you are on, and which step you are performing (refer to Handout #5 as a guide).

   NOTE: Finally, be sure to note the convention in Handout #5 (and common in the research literature): columns represent prime implicants, and rows represent ON-set minterms. Follow this convention throughout the course. (Some digital textbooks reverse the above usage!)

	p_1	p_2	p_3	p_4	p_5	p_6
1	0	0	0	1	1	0
2	0	1	1	0	0	1
3	1	0	0	1	0	1
4	1	0	0	1	0	0
5	0	1	0	0	1	0
6	1	0	1	0	0	0
7	0	1	1	0	1	0
8	0	0	1	0	0	1

3. (10 points) **Quine-McCluskey Method: Step #4.** For this problem, refer to the constraint matrix, which is already reduced, given below. This 4x4 matrix is cyclic. Apply Petrick's method to solve the matrix, as follows: (a) write the constraint equation for the matrix (i.e., product of sums equation); (b) multiply out and solve the equation, finding a minimum-cost cover, using Boolean simplification. (Refer to the Quine-McCluskey Handout, Example #2.) Show all work.

	p_1	p_2	p_3	p_4
m_1	1	0	1	0
m_2	1	1	0	1
m_3	0	1	1	0
m_4	0	0	1	1

4. (16 points) **Multi-Output Functions, Representations, and Minimization.** An incompletely specified 3-output function is given by the following minterm expansions:

$$f_1(x, y, z) = \Sigma m(0, 2)$$
$$f_2(x, y, z) = \Sigma m(2, 6, 7) + \Sigma d(0)$$
$$f_3(x, y, z) = \Sigma m(0, 4, 6, 7) + \Sigma d(3)$$

Given this multi-output function, do the following:

(a) Write the multi-output truth table (in a single table, listing all outputs).

(b) Write a *PLA input file* (also called *"cubical representation"* or *"cubical complex"*), using cubes to indicate the ON-set and DC-set (see Handouts #6 and #9a).

(c) Draw a *hypercube representation for the function*, indicating ON-set and DC-set minterms. Mark ON-set minterms with a black dot, DC-set minterms with an X, and OFF-set minterms with a white dot.

(d) On your figure in part(c), neatly mark all *multi-output prime implicants* (do by inspection; you do not need to use the Quine-McCluskey algorithm to derive these!).

Note: Refer to Handout #6 for discussion of multi-output primes. Remember that "multi-output primes" may contribute to 1, 2, or 3 of these outputs, but in all cases cannot be expanded further in the multi-output function without hitting the OFF-set (i.e. 0 minterms).

(e) List the multi-output primes of part(d) in *cubical representation*, as presented in Handouts #6 and #9a.

(f) Indicate all *essential multi-output primes*.

(g) Indicate a *minimum-cost cover* of multi-output prime implicants (do by inspection; you do not need to use the QM algoritm to derive these). Do not modify the primes!

(h) Write your solution to part (g) *cubical representation*, as presented in Handouts #6 and #9a.

5. (10 points) **Espresso's EXPAND Step: Cube Ordering Heuristic.** You are given a single-output binary-valued function $f(a, b, c, d)$, specified as follows:

$$F^{ON} = abc'd + a'd' + a'bc'd + a'bc$$
$$F^{OFF} = a'b'd + abd' + abc + ab'$$
F^{DC} is empty

For this problem, assume that the initial "seed cover" F is simply F^{ON}. You will not do the complete "expand" step in this problem, just the first operation: determining cube order for expansion. (Note: you will not need to use F^{OFF} for this problem, it is only given for completeness.)

(a) *Initialization.* Do the following: (i) *draw the Karnaugh map,* and label each cube of the ON-set and OFF-set; and (ii) *write the PLA file* for seed cover F (i.e. F^{ON}).

(b) *Determine Final Cube Order for Expansion.* Following the method presented in lecture #2, and in the Lecture #2 part 2 slides, determine the order (from first to last) of the cubes for expansion. *Show all work:* (i) calculate the literal weight vector; (ii) calculate cube weights (for each of the 4 cubes in seed cover F), and show the resulting cube weight table; and (iii) sort the cubes by weight (low to high), and show the resulting sorted cube weight table.

6. (10 points) **New Algorithm: Weighted Unate Covering (Part I).** In this problem, you are to modify the Quine-McCluskey algorithm to handle a more sophisticated cost function, which takes into account the number of gate inputs.

   The Quine-McCluskey handout assumes *cost function #1:* each product has the same cost. Therefore, a minimum-cost cover is one with fewest products, also known as a *minimum-cardinality cover.* That is, each column in the prime implicant table has equal cost (or "weight").

   In particular, the goal is to modify the Quine-McCluskey algorithm to target *cost function #2:* A minimum-cost solution is one with **fewest literals** (i.e., fewest total inputs to all the selected prime implicants, or AND gates). Note that this cost function does *not* attempt to minimize the number of products: it can find solutions which have more products than cost function #1, but with fewer total literals. Therefore, cost function #2 is a more accurate model for minimizing the area of the circuit.

   For each substep in the Quine-McCluskey algorithm (see handout), (i) *indicate what change needs to be made (if any),* to handle the new cost function correctly; and (ii) give a brief butr justification for the change/no change. Do (i) and (ii) for each of the following substeps:

   (a) remove essential prime implicants;

   (b) row dominance;

   (c) column dominance;

   (d) Petrick's method.

   Hint: Consider the role of each of the above steps in producing an exact solution. Step (a) identifies when a cube must always be included in the solution; determine if the new cost function alters this step. Steps (b)-(c) are reduction techniques to simplify the problem being solved, i.e. by reducing the PI table; determine if these steps can be safely applied or if they can remove good solutions under the new cost function, and need to be modified. Finally, step (d) indicates how to solve a cyclic core and select an optimal solution; determine if the method remains unchanged or needs modification.

7. (10 points) **New Algorithm: Weighted Unate Covering (Part II).** You are now to repeat the previous problem but for another variant of the optimization problem. In this problem, focus again on *cost function #1:* each product has the same cost. Hence, a minimum-cost cover is one with fewest products, also known as a *minimum-cardinality cover.*

   The classic Quine-McCluskey algorithm is targeted to find *some* minimum-cost solution. However, its steps may delete other candidate minimum-cost solutions. For some applications, the designer will want the algorithm **to produce all minimum-cost solutions.**

   What to Do: repeat parts (a)-(d) in the previous problem, following the same guidelines, but now ensuring that the QM method can identify all possible minimum-cost solutions (with none eliminated during the algorithm).

8. (*EXTRA CREDIT:* 20 points) **Achilles' Heel Function.** The "Achilles' heel function" has 2n variables:

$$f = (a_1 + b_1) \cdot (a_2 + b_2) \cdot \ldots \cdot (a_n + b_n)$$

   It is an interesting case: it has 2^n prime implicants, and the minimum-cost cover includes all the 2^n primes, demonstrating that in the worst-case, prime implicant generation and covering can be exponential!

   In this problem, you are to prove this point. In particular, prove that each cube in the sum-of-products expansion of the above expression (i.e., after multiplying out) is an essential prime implicant of f. *Show all work;* give a clear and detailed argument to support your answer.

   Hint: to get intuition, first draw a 2-variable K-map for the case of n=1: $(a_1 + b_1)$. Then draw a 4-variable K-map for the case of n=2: $(a_1 + b_1) \cdot (a_2 + b_2)$. Identify the primes and essentials in each case. Then consider the generalization to arbitrary n.