Logic Synthesis & Optimization

Spring 2019, Solutions to Homework # 3
Ashenhurst-Curtis Decomp

Solution to Q6, Simple disjunctive decomp:

The support set of f is w, x, y, z. We are asked to partition the variables in such a way that the bound set size is two. The possibilities are: $(w, x; y, z), (w, y; x, z), (w, z; x, y), (x, y; w, z), (x, z; w, y), (y, z; w, x)$. When you draw the corresponding Karnaugh-maps and look for column multiplicity ≤ 2 (for feasible simple decompositions), only three partitions lead to a possible decomposition. These are: $(w, x; y, z), (w, y; x, z)$ and $(x, y; w, z)$.

Let us consider the one with w, x as the bound set variables.

yz \ wx	00	01	10	11
00	0	0	1	0
01	1	1	0	1
10	0	0	0	0
11	1	1	1	1

identical entries

00 +01 +11	10
0	1
1	0
0	0
1	1

Fig. 1. Functional Decomposition Chart.

Notice that the function can be decomposed simple-disjunctively, and encoded with $\phi(w, x) = w'x' + w'x + wx$ and $\overline{\phi} = wx'$ or *vice-versa*. The entries in the ϕ column that have ones correspond to the free-set terms: $\phi \cdot y'z + \phi \cdot yz$. Similarly, free-set terms corresponding to $\overline{\phi}$ are $y'z' + yz$. Hence, F is decomposed as: $\phi \cdot y'z + \phi \cdot yz + \overline{\phi}y'z' + \overline{\phi}yz$.

If you transpose the above matrix - you'll have w, x as the free set and y, z as the bound set. But this won't work as column multiplicity is 4 (with one column of all 0s). This would require Ashenhurst-Curtis Decomp..... but thats another story.

You can also see that x, y as bound set (and w, z as free set) does also give column multiplicity = 2; which means that the # of cut-set nodes in the BDD should be equal to 2. And you can confirm that by looking at the following BDD.

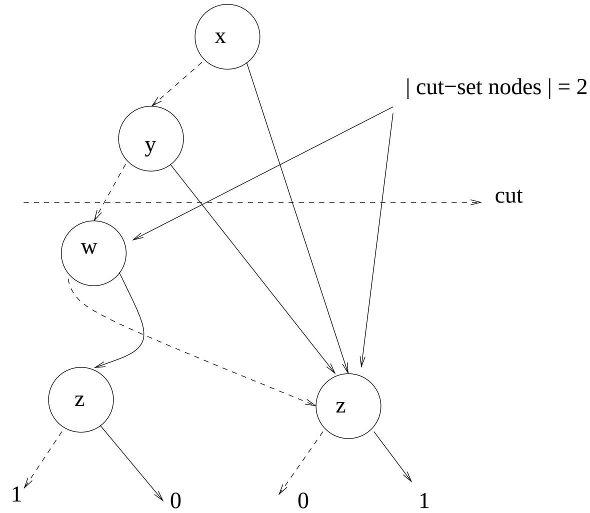

Fig. 2. Simple Decomposition on BDD

Q 7 For Ashenhurst-Curtis Decomp: you can draw the BDD with the given order and see that a simple disjunctive decomp is not possible. In order to perform the decomposition, we will have to encode the bound-set cubes corresponding to each cut-node. The BDD for f is shown below:

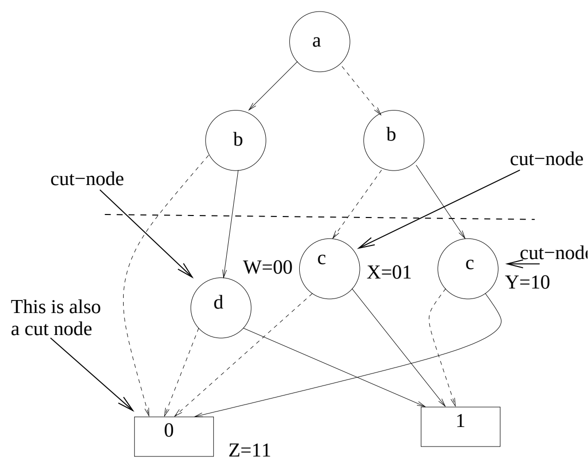

Fig. 3. Ashenhurst-Curtis Decomposition on BDD

Now comes the tricky part. You are supposed to encode ALL THE CUBES of the bound set. Please follow this basic procedure:

- Starting from the root, traverse ALL paths and "pierce the cut". These paths correspond to the cubes in the bound set.
- The **first node that you hit across the cut** represents the subfunction that will be implemented in the free-set part.
- In the above BDD, you will find that the path corresponding to cube $a \cdot b'$ hits terminal node **0** -

this is the first node after the cut on this particular path (ab'). Which means that corresponding to bound-set cubes ab', the subfunction implemented in the free-set part will be $a \cdot b' \cdot 0$.

- This does not mean that you can ignore to encode the terminal node. If you ignore this terminal node, you will get an **incorrect** answer: *i.e.*, f will not compute zero when ab' is applied.

Let us encode the 4 cut-set nodes W, X, Y, Z with two bits g_0, g_1. Let $Z = 11$ correspond to the encoding of the terminal **0** node. $g_0 = 1$ for node Z and node Y.

Again, note the following: For node Z, only the incoming path ab' is to be considered for g_0. Why not the other ones? Because, the bound-set cubes corresponding to the other paths will be encoded by the remaining cut-set nodes. This is because the remaining paths hit cut-nodes W, X, Y.

$$\begin{aligned} g_0 &= a \cdot b'(nodeZ) + a' \cdot b(nodeY) && (1) \\ g_1 &= a \cdot b'(nodeZ) + a' \cdot b'(nodeX) && (2) \\ H &= g_0'g_1'd + g_0'g_1c + g_0g_1'c' + g_0g_10 && (3) \\ &= g_0'g_1'd + g_0'g_1c + g_0g_1'c' && (4) \end{aligned}$$

You may try any other encoding, but traverse the bound-set (paths) cubes properly! In the above, apply ab' as the input to the bound set function, and notice that the output of the overall function F also results in 0.

Don't cares: Notice, in this case there are NO DON'T CARES! All four value combinations of g_0g_1 will be produced.