

## Merging Ordered Binary Decision Diagrams

When building new ordered binary decision diagrams (obdd's) from old ones, we merge them together. Consider the obdd's in figure 1 named f and g.

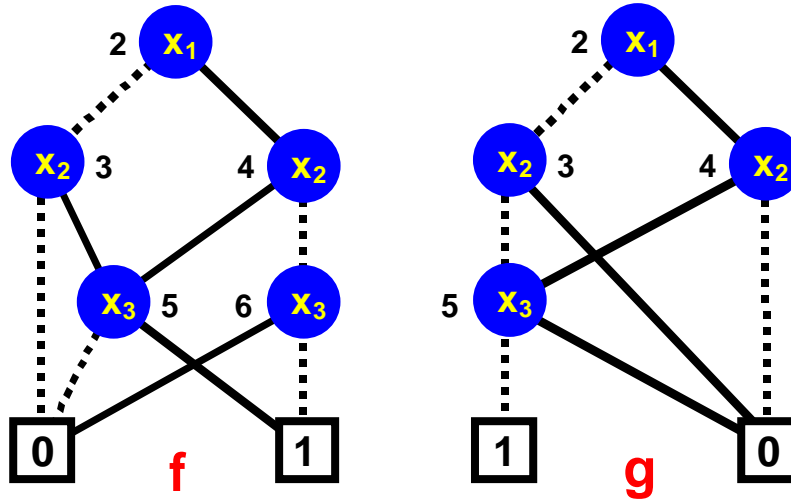


Figure 1 - Ordered Binary Decision Diagrams

When drawing decision diagrams, the 0-branches appear as dotted lines, while the 1-branches are drawn as solid lines. The nodes for the output values of false (0) and true (1) occur at the leaves.

We may also describe these diagrams as tables, such as those of figure 2 with f on the left and g on the right.

		Branch	
	var	0	1
0	0	0	0
1	1	1	1
2	$x_1$	3	4
3	$x_2$	0	5
4	$x_2$	6	5
5	$x_3$	0	1
6	$x_3$	1	0

		Branch	
	var	0	1
0	0	0	0
1	1	1	1
2	$x_1$	3	4
3	$x_2$	5	0
4	$x_2$	0	5
5	$x_3$	1	0

Figure 2 - OBDD Tables for f and g

Note that the node labels in figure 1 correspond to the table rows in figure 2 where the variables found at the nodes are stored. We shall always store the leaves (which have values of 0 and 1) in rows 0 and 1.

In order to construct an obdd for the function  $h = f + g$  we merge the  $f$  and  $g$  obdd's. To do this we begin with the standard rows for the 0 and 1 valued leaves as shown in figure 3. Next we add a row for the first variable,  $x_1$ , which appears at the root of  $h$ , and label it as node 2 of the new obdd as shown in the table of figure 3.

	var	Branch		Roots	
		0	1	f	g
0	0	0	0		
1	1	1	1		
2	$x_1$			2	2

Figure 3 - Setting up an OBDD Table

To the right of the branch columns we have added a pair of root columns. These indicate exactly where the variable for that row appears in the obdd's for  $f$  and  $g$ . In figure 3 the variable  $x_1$  in  $h$ 's obdd is the root of the obdd corresponding to the union of the two obdd's rooted at row 2 in both  $f$  and  $g$ , that is, the union of  $f$  and  $g$ .

We now wish to add the 0 and 1 branches coming from the first node of our new obdd  $h$ . Taking the  $x_1=0$  and  $x_1=1$  Shannon decompositions of both  $f$  and  $g$  allows us to do exactly this. (Recall that the Shannon decomposition of  $f$  with respect to  $x_1=0$  is found by replacing  $x_1$  by 0 in  $f$ ) These we combine and attach to  $x_1$  in the new obdd ( $h = f + g$ ) in the manner shown in figure 4. Note that these new nodes are labeled 3 and 4.

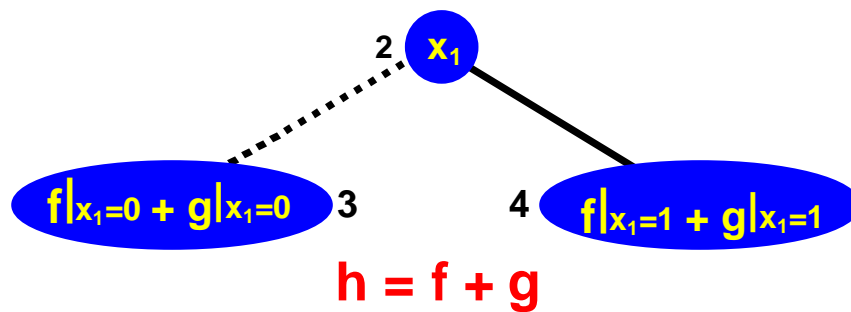


Figure 4 - Beginning a New OBDD

The obdd's corresponding to the Shannon decompositions for  $f$  mentioned above are shown in figure 5. These are found by retaining the sub-obdd's below the 0 and 1 branches of  $f$ 's  $x_1$  node.

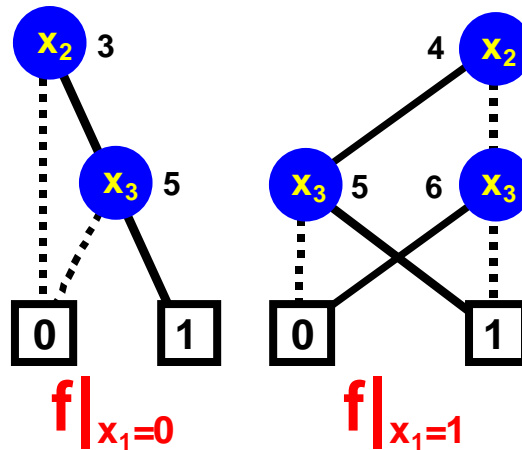


Figure 4 - Shannon Decompositions of f's OBDD

If we label the 0-branch of h's  $x_1$  node (as shown in figure 4) as 3 and its 1-branch as 4, we can add the corresponding rows to a new obdd table we shall build for  $h = f + g$ .

We now attach the obdd which is found by merging the  $f$  and  $g$  sub-obdd's rooted at the nodes labeled with a 3 in each (the zero branches from  $x_1$  in  $f$  and  $g$ ) to h's node 3. Likewise, the one-branch of h's  $x_1$  node goes to a merger of the sub-obdd's rooted at the nodes in  $f$  and  $g$  labeled with a 4.

A partial table for this new  $h = f + g$  obdd is found in figure 5.

	var	Branch		Roots	
		0	1	f	g
0	0	0	0		
1	1	1	1		
2	$x_1$	3	4	2	2
3	$x_2$			3	3
4	$x_2$			4	4

Figure 5 - Beginning the h OBDD Table

The last two columns provide the roots of the sub-obdd's that are to be merged and attached to the node for the variable. In this case, the obdd for  $f + g$  (rooted at node 2 in both  $f$  and  $g$ ) is attached to  $x_1$  at node 2 in the new obdd. Likewise, the sub-obdd's rooted at node 3 in both  $f$  and  $g$  are to be merged and attached to  $x_2$  at node 3 of the new obdd.

If we are to perform a depth-first construction, then we now build the sub-obdd rooted at  $x_2$  in node 3. Also at this point we jot down a reminder to build the sub-obdd which we shall attach to the  $x_2$  at node 4 and place this on a stack of tasks to be done.

To form the zero and one branches for  $x_2$  in node 3, we just look at the zero branches of  $f$  and  $g$  from their node 3's. These come out to be 0 (from  $f$ ) and 5 (from  $g$ ). This is placed in row 5 for the  $x_3$  attached to  $x_2$  in node 3. Likewise row 6 is formed and added to the table. This is illustrated in figure 6 along with the row 5 and 6 branch entries that shall be explained in a moment.

	var	Branch		Roots	
		0	1	f	g
0	0	0	0		
1	1	1	1		
2	$x_1$	3	4	2	2
3	$x_2$	5	6	3	3
4	$x_2$			4	4
5	$x_3$	1	0	0	5
6	$x_3$	0	1	5	0

Figure 6 - More of h's OBDD Table

Notes are also written to remind us to build rows 5 and 6.

To get the zero and one branches of  $x_3$  in row 5 we examine the zero branches of row 0 in  $f$  and row 5 in  $g$ . This leads to 0+1 which of course is a 1. The one branch for row 5 of  $h$  is found to be 0+0 and thus a 0 is placed here. Row 6 is formed the exact same way. After adding row 6 to the table, we check to see that it is not the same as row 5. If it were then we would merge the two rows and adjust references to them that appear higher in the table. (This will be demonstrated below when row 7 is constructed.)

Now table 6 contains the entire left side of the obdd for  $h$  that we are building. The tree corresponding to this table appears as figure 7.

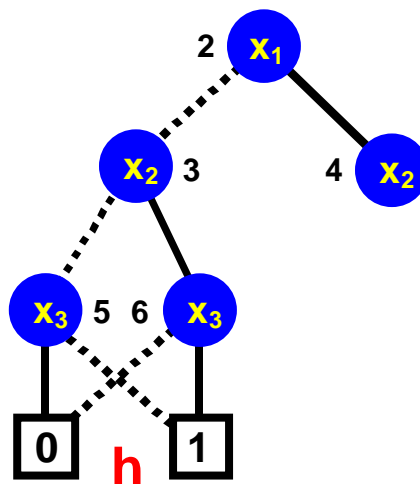


Figure 7 - Partial OBDD for  $h = f + g$

Now we must attach an obdd to the  $x_2$  node in row 4. It is attached to the nodes containing  $x_3$  which are placed in rows 7 and 8. Row 7 is then formed by looking at the zero and one branches of node 6 of  $f$  and the leaf 0 of  $g$ . All of this appears in the table of Figure 8.

	var	Branch		Roots	
		0	1	f	g
0	0	0	0		
1	1	1	1		
2	$x_1$	3	4	2	2
3	$x_2$	5	6	3	3
4	$x_2$	7	8	4	4
5	$x_3$	1	0	0	5
6	$x_3$	0	1	5	0
7	$x_3$	1	0	6	0
8	$x_3$			5	5

Figure 8 – More of the OBDD Table for  $h = f + g$

Now it is time to examine row 7 and compare it to all of the other rows in which  $x_3$  appears. We note that it is identical to row 5. Therefore we can combine row 7 with row 5 and direct all branches that went to row 7 to row 5. This changes row 4 whose zero branch is now row 5 (as in figure 9). Since row 4 changed we check to see if it is the same as any of the other rows that contain  $x_2$ . It is not, so we continue.

The next step is forming row 8. The zero-branches of node 5 in both  $f$  and  $g$  provide  $0+1$  and the one-branches provide  $1+0$ , both of which reduce to 1. The table for  $h$  at this point is shown in figure 9. Note that the roots for row 4 have been omitted because after the entry in the zero-branch was changed to 5, they no longer applied.

	var	Branch		Roots	
		0	1	f	g
0	0	0	0		
1	1	1	1		
2	$x_1$	3	4	2	2
3	$x_2$	5	6	3	3
4	$x_2$	5	8		
5	$x_3$	1	0	0	5
6	$x_3$	0	1	5	0
8	$x_3$	1	1	5	5

Figure 9 – A Smaller OBDD Table for  $h = f + g$

While examining row 8, we find that both entries in row 8 are the same. Thus row 8 may be deleted and all references to it changed to the value in row 8, namely 1. So, we now change row 4 so that its zero and one branches are 5 and 1. The final table for h appears as figure 10 below.

	var	Branch		Roots	
		0	1	f	g
0	0	0	0		
1	1	1	1		
2	$x_1$	3	4	2	2
3	$x_2$	5	6	3	3
4	$x_2$	5	1		
5	$x_3$	1	0	0	5
6	$x_3$	0	1	5	0

Figure 10 - The OBDD Table for  $h = f + g$

The final, reduced OBDD for h is shown in figure 11.

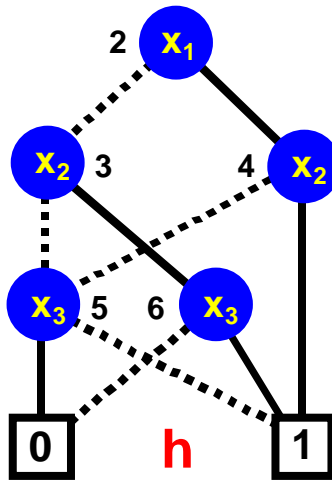


Figure 11 - The Reduced OBDD for  $h = f + g$