CS226 Practice Problem Set 2 (Spring 2016)

Date posted: Feb 8, 2017 **Expected Solving Time: 2 hours**

- *Be brief, complete and stick to what has been asked.*
- *Unless asked for explicitly, you may cite results/proofs covered in class without reproducing them.*
- *If you need to make any assumptions, state them clearly.*
- ***These are ungraded practice questions. You are strongly encouraged to solve these on your own to ensure you understand the material being taught in class.***
- ***Mutual discussion is allowed, but copying is not. Please read the guidelines on the course webpage if you don't understand the distinction between the two.***

1. (a) Write a boolean formula that corresponds to the ROBDD shown in Fig. 1. Note that it has few extra boxes fro the "0" terminal. This is to keep the diagram from getting cluttered by lots of edges to 0. To be fully reduced, there is only one "0" in the real graph.

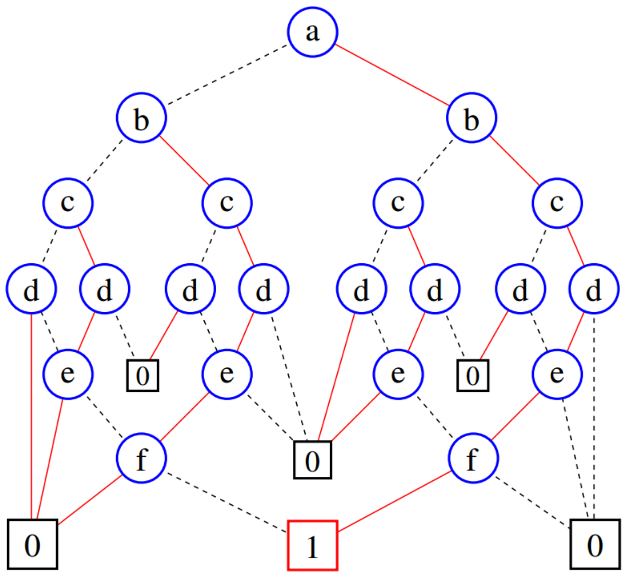

Figure 1: ROBDD example

   (b) State a variable ordering for which the function from questions 1.b can be represented with fewer nodes. Draw the ROBDD for the function using your proposed ordering.

2. Consider the function f = $x_1'x_2x_4' + x_1x_2'x_3x_4 + x_1x_2'x_3'x_4 + x_1x_2x_3'$

   (a) Construct the ROBDD for f using the variable ordering $x_1 < x_4 < x_3 < x_1$.

   (b) From the above ROBDD, use graph operations to get the ROBDDs for restrict(x_4, 0, f) and restrict(x_4, 1, f). You must indicate the graph transformations (which node/edges are deleted, re-directed etc.) that are used to construct ROBDDs for the restrict(...) operations. You must NOT construct ROBDDs for restrict(...) afresh after computing the function restrict(...) first.

3. Draw ROBDDs for $(x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2)$ with orderings $x_1 < x_2 < y_1 < y_2$ and $x_1 < y_1 < x_2 < y_2$. What variable ordering would you recommend for constructing the general case of : $(x_1 \leftrightarrow y_1) \wedge (x_2 \leftrightarrow y_2) \wedge (x_3 \leftrightarrow y_3) \wedge (x_4 \leftrightarrow y_4) \ldots\ldots \wedge (x_n \leftrightarrow y_n)$? Give justification for your answer.

4. *[Adapted from Spring 2016 endsem]* Usually, when we draw Karnaugh maps (henceforth called K-maps), we fill in the entries with 1, 0 or $-$ (for don't care). In this question, we want to consider symbolic Karnaugh maps, where we can fill in the entries with symbolic variables or even with other Boolean functions, in addition to using 1, 0 and $-$.

   Consider the symbolic K-map shown below, where F and G are Boolean variables, and F' and G' represent their respecctive Boolean complements.

$\mathbf{x_1,x_2\rightarrow}$ / $\mathbf{x_3,x_4\downarrow}$	**00**	**01**	**11**	**10**
00	F	G	F'	1
01	1	G	1	G'
11	F'	F	G	F'
10	G	F'	0	F

   Clearly, the above K-map gives rise to different Boolean functions of x_1, x_2, x_3 and x_4 for different combinations of Boolean values of F and G. In other words, the K-map represents a Boolean function of x_1, x_2, x_3, x_4, F and G. Let us call this function $\varphi(x_1, x_2, x_3, x_4, F, G)$.

   (a) Give a combination of Boolean values of F and G, such that φ with F and G set to these values can be implemented by the circuit shown below (Figure 2). Note that you are not allowed to change the interconnection between the AND and OR gates. You are not allowed to use any additional gates either. However, you are free to label the inputs of the AND gates with $x_1, x_1', x_2, x_2', x_3, x_3', x_4$ or x_4', as you consider appropriate. You must indicate the values of F and G in the space given below, and label the inputs of the AND gates directly in the circuit diagram (Figure 2) shown below.

   Boolean value of F: ______________ , **Boolean value of G:** ______________

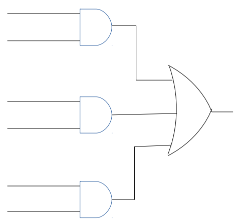

Figure 2: Circuit 1

   (b) Now suppose F and G are not Boolean variables, but Boolean functions of $\{x_1, x_2, x_3, x_4, x_5, x_6\}$ (or a subset thereof). Therefore, $\varphi(x_1, x_2, x_3, x_4, F, G)$ is now a Boolean function of $\{x_1, x_2, x_3, x_4, x_5, x_6\}$ (or a subset thereof). Give symbolic K-maps for F and G with as many don't cares entries as possible, in the spaces shown below, such that all of the following three conditions hold:

   - $\overline{(\varphi\,|_{x_5} \;\oplus\; \varphi\,|_{\overline{x_5}})}$ is $x_1.x_2 + x_2'.x_3'$
   - $\overline{(\varphi\,|_{x_6} \;\oplus\; \varphi\,|_{\overline{x_6}})}$ is $x_3'.x_4 + x_1.x_4'$.

- If the values of x_5 and x_6 are different, then the Boolean functions F and G evaluate to different values (for any combination of values of x_1, x_2, x_3, x_4). *Note that this is an if-then statement, and not an if-and-only-if statement.*

The entries in your K-maps should either be 0, 1, $-$ or functions of $\{x_5, x_6\}$ (or subsets thereof).

Symbolic K-map for F:

$\mathbf{x_1,x_2}\rightarrow$ / $\mathbf{x_3,x_4}\downarrow$	**00**	**01**	**11**	**10**
00				
01				
11				
10				

Symbolic K-map for G:

$\mathbf{x_1,x_2}\rightarrow$ / $\mathbf{x_3,x_4}\downarrow$	**00**	**01**	**11**	**10**
00				
01				
11				
10				

5. *[Adapted from Spring 2016 endsem]* Consider the Boolean function $F = (a \oplus b) \cdot (b \oplus c) \cdot (d \oplus e) \cdot (e \oplus f) \cdot (a + f)$, where $\oplus$ denotes EXOR.

   (a) *[10 marks]* Construct an *ROBDD with complement edges* for F using the variable ordering $a < b < c < d < e < f$. You must use only a single 1-leaf (no 0-leaf), and you must *not* have complementing bubbles on solid edges (or 1-labeled edges) in your final answer. If needed, you can use a root edge with a complemening bubble. You must present only your final ROBDD with complement edges in the space below.

   (b) *[10 marks]* You are required to implement the function F by interconnecting 2-*to*-1 *multiplexors and a* 2-*to*-4 *decoder*. You are given three working 2-to-1 multiplexors, four defective 2-to-1 multiplexors and a single defective 2-to-4 decoder, for this purpose. You are not allowed to use any other gate (not even an inverter) in your implementation. Assume that you have access to $a, a', b, b', c, c', d, d', e, e', f$ and f', so you can feed any of the literals directly to an input of the multiplexors or the decoder.

   Figure 3 shows a working 2-to-1 multiplexor, a defective 2-to-1 multiplexor and a defective 2-to-4 decoder, along with the functions they implement. Indicate in the space below how you would connect the (working and/or defective) multiplexors and decoder to implement F correctly. You must label each working multiplexor by "WM", and each defective multiplexor by "DM". You must also label the defective decoder by "DD".

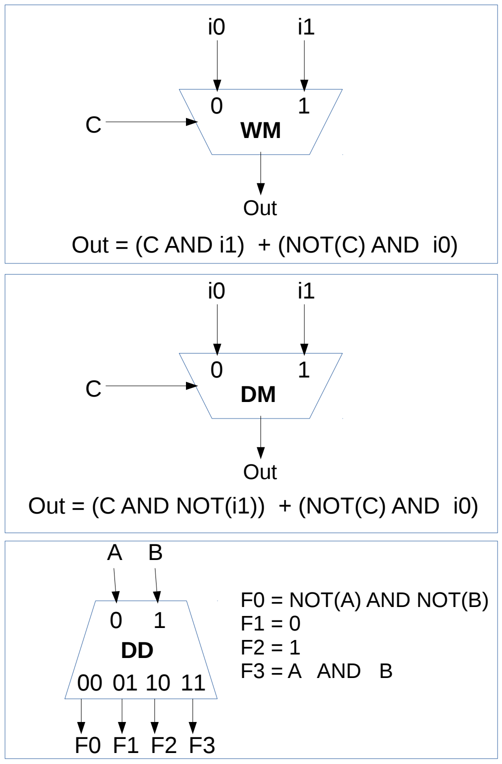

Figure 3: Circuit 1