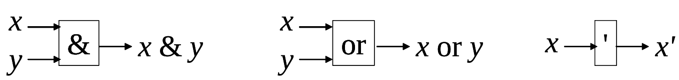
Unit C2: Boolean Algebra, 10/28/03

Exercise 1: Pseudo-Electrical Engineer

In a certain country at the fall of communism, a mathematician called Borisky needed to work as an electrical engineer. He knew nothing about digital circuits, but he knew that digital circuits, set theory, and propositional logic are all isomorphic and instances of Boolean algebra. So, he thought that he could handle his new job, relying on his expertise, set theory and propositional logic.

Due to the shortage of components in his country, Borisky needed to use ‘&’ and ‘'’ components to simulate the unavailable ‘or’ components (show below).



x	*y*	*x* & *y*
1	1	1
1	0	0
0	1	0
0	0	0

x	*y*	*x* or *y*
1	1	1
1	0	1
0	1	1
0	0	0

x	*x*'
1	0
0	1

Borisky guessed that there is an expression that involves only ‘&’ and ‘'’ but not ‘or’ that is still equivalent to expression *x* or *y*. If this is true, he can use ‘&’ and ‘'’ to simulate all the cases involving ‘or’.

A. What would be the equivalent expression Borisky guessed?

   Hint: Since digital circuits are an instance of Boolean algebra, all the properties that apply to set theory and propositional logic also apply to digital circuits. Review the reference slides in Unit B5 (Propositional Logic).

B. **Convert** the two expressions (i.e., *x* or *y* and your answer to A.) to the corresponding expressions in *set theory* (i.e., using ‘∪’, ‘∩’, ‘'’, etc.) and **justify** the equivalence of the converted set expressions (within set theory).

   Hint: You may use schematic representations to show the equivalence.

In order to minimize the cost of mass-producing a digital circuit, Borisky needed to reduce the number of components used in a circuit represented as (*x*' & *y*')' & (*z*' & *x*')'. By this time, the production of ‘or’ components are catching up. He guessed that the circuit is equivalent to *x* or (*y* & *z*). However, before mass-producing the circuit, he is required to verify the equivalence.

C. First, convert the expressions (*x*' & *y*')' & (*z*' & *x*')' and *x* or (*y* & *z*) to the corresponding wff’s in propositional logic. Then, prove [**FIX THIS!!!**] *x* or (*y* & *z*) by considering (*x*' & *y*')' & (*z*' & *x*')' as the hypothesis.

Exercise 2: Exotic Logic and Circuits

Note: In this exercise, you will explore some exotic forms of propositional logic and digital circuits, and their connection to set theory. This one may be slightly challenging. Try as much as you can. Your experience will be useful as a preview of Unit C3, where this exercise will be discussed.

Porpositional Logic involves two values, true (**T**) and false (**F**). However, in reality, we often encounter situations where we have to say “neither,” or “undefined” (**U**) (recall your response to my questions in class). Note that there are logicians who seriously study variants of logic of this sort. Then, we can define a structure such as the following:

$\mathbf{Prop_3} = (V, \wedge, \vee, \neg)$
where $V = \{\mathbf{T}, \mathbf{F}, \mathbf{U}\}$

A. Can $\mathbf{Prop}_{23}$ be a Boolean Algebra? You can define the operations in favor of your answer. Concisely explain.

   Hint: Unit C2 slides on the definition of Boolean Algebra.

Next, let us turn to digital circuits. Suppose that with advancement in electronics engineering, basic electronic devices use 4 states, instead of 2. Note that related phenomena may be happening in the industry; can you point out? Then, we can define a structure such as the following:

$\mathbf{Bool_4} = (D, \&, \text{or}, ')$
where $D = \{0, ♥, ♠, 1\}$

B. Can $\mathbf{Bool}_{24}$ be a Boolean Algebra? You can define the operations in favor of your answer. Concisely explain.

Let us now turn to the other cousin of Boolean Algebra, set theory. Here are two possible extensions of the structure **Set** discussed in Unit C2.

$\mathbf{Set_3} = (S_3, \cap, \cup, ')$
where $S_3 = \{\varnothing, \{1\}, \{2\}\}$

$\mathbf{Set_4} = (S_4, \cap, \cup, ')$
where $S_4 = \{\varnothing, \{1\}, \{2\}, \{1, 2\}\}$

C. Concisely explain why $\mathbf{Set_3}$ is ***not*** a Boolean Algebra, regardless of how the operations are defined. You could also try $S_{3.1} = \{\varnothing, \{1\}, \{1, 2\}\}$, if you wish.

D. Concisely explain why $\mathbf{Set_4}$ is a Boolean Algebra, with the usual definitions of the operations.

E. Draw some conclusion from your observation in Questions A. through D. above.

Let us shift the discussion a bit. Earlier in class, we noted that the following equivalence (verifiable using Venn diagrams):

Equivalence:

1. $x \subseteq y$
2. $x \cap y = x$
3. $x \cup y = y$

Now, let us consider the following two structures involving S_4, along with the usual definitions of the operations/relation.

$\mathbf{Set_4} = (S_4, \cap, \cup, ')$
$\mathbf{Set_5} = (S_4, \subseteq)$
where $S_4 = \{\varnothing, \{1\}, \{2\}, \{1, 2\}\}$

Note that $\mathbf{Set_4}$ and $\mathbf{Set_5}$ are ***not*** isomorphic (why?). So, $\mathbf{Set_5}$ is not a Boolean Algebra. However, if we compare the information conveyed by the two structure, we notice that they are equivalent. Neither $\mathbf{Set_4}$ nor $\mathbf{Set_5}$ says more than the other. Naturally, this is due to the fact shown in **Equivalence** above.

F. Can you connect the above observation with your answer to Question E?

<End>