INTRODUCTION TO COMPUTING
TOPIC V: SYMBOLIC LOGIC

PAUL L. BAILEY

1. Propositions

A *proposition* is a statement which is either true or false, although we may not know which. Propositions are denoted by lowercase letters such as p, q or r. The truth or falsity of the proposition is called its *truth value*, and the two possible truth values are labeled **T** for TRUE and **F** for FALSE. The truth value of the proposition p is denoted $\mathbf{V}(p)$.

For example, the statement "The sun rises in the east" is a proposition, and if we wish to label this statement p, we write

$$p = \text{"The sun rises in the east"}.$$

Similarly, we may write

$$q = \text{"The sun rises in the west"}.$$

In this case, $\mathbf{V}(p) = \mathbf{T}$ and $\mathbf{V}(q) = \mathbf{F}$.

2. Logical Operators

Propositions may be modified and combined by the use of *logical operators*, which take one or more propositions and create a new one which has its own truth value. The resultant truth value is uniquely determined by the truth value(s) of the proposition(s) operated upon and the operator(s) used. Operators which accept one input are called *unary* operators, and operators which accept two inputs are called *binary* operators.

The behavior of each logical operator is determined by a *truth table*. The truth table lists all possible combinations of the truth values of the inputs, and states the operator's output for each combination of inputs.

The simplest useful logical operator is the *negation* operator NOT ($\neg$), which operates on a single proposition and reverses its truth value. Thus

$$\neg(\text{"Pigs are mammals"}) = \text{"Pigs are not mammals"}.$$

Date: September 6, 2017.

The action that NOT has on the truth value of a proposition is defined by its truth table, which lists the possible truth values of a proposition p side by side with the truth value of $\neg p$:

p	$\neg p$
T	**F**
F	**T**

TABLE 1. NOT Truth Table

Assertion 1. *If p is any proposition, then*

$$\mathbf{V}(\neg(\neg p)) = \mathbf{V}(p)$$

.

Proof. If p is TRUE, then $\neg p$ is FALSE, and so $\neg(\neg p)$ is TRUE. If p is FALSE, then $\neg p$ is TRUE, and so $\neg(\neg p))$ is FALSE. □

The next logical operator we consider is the *conjunction* operator AND ($\wedge$). The proposition $p \wedge q$ is true only when both p and q are true propositions. For example, if $p =$ "Pigs are mammals" and $q =$ "Pigs fly", then $p \wedge q$ may be interpreted as "Pigs are flying mammals". The AND operator is defined by a truth table which lists all possible combinations of the truth values of p and q:

p	q	$p \wedge q$
T	**T**	**T**
T	**F**	**F**
F	**T**	**F**
F	**F**	**F**

TABLE 2. AND Truth Table

The *disjunction* operator OR ($\vee$) returns a value of TRUE whenever either proposition it operates upon is true, and therefore is defined by:

p	q	$p \vee q$
T	**T**	**T**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

TABLE 3. OR Truth Table

Thus if let p and q be as above and we assume that pigs are mammals who cannot fly, we have $\mathbf{V}(p) = \mathbf{T}$, $\mathbf{V}(q) = \mathbf{F}$, $\mathbf{V}(p \wedge q) = \mathbf{F}$ and $\mathbf{V}(p \vee q) = \mathbf{T}$.

At this point we adopt the convention that the NOT operator takes "binds tighter" than any other operator, that is, it takes precedence in the order of operations and applies only to the object on its immediate right. Thus $\neg p \wedge q$ means $(\neg p) \wedge q$ as opposed to $\neg(p \wedge q)$. We are now ready for our first theorem.

Theorem 1. *(DeMorgan's Laws) For any two propositions p and q we have*

(1) $\mathbf{V}(\neg(p \vee q)) = \mathbf{V}(\neg p \wedge \neg q)$;
(2) $\mathbf{V}(\neg(p \wedge q)) = \mathbf{V}(\neg p \vee \neg q)$.

Proof. The proofs of these assertions are truth tables in which each step is expanded, and the columns corresponding to either side of the equalities above are compared.

p	q	$p \vee q$	$\neg(p \vee q)$	$\neg p$	$\neg q$	$\neg p \wedge \neg q$
T	**T**	**T**	**F**	**F**	**F**	**F**
T	**F**	**T**	**F**	**F**	**T**	**F**
F	**T**	**T**	**F**	**T**	**F**	**F**
F	**F**	**F**	**T**	**T**	**T**	**T**

p	q	$p \wedge q$	$\neg(p \wedge q)$	$\neg p$	$\neg q$	$\neg p \vee \neg q$
T	**T**	**T**	**F**	**F**	**F**	**F**
T	**F**	**F**	**T**	**F**	**T**	**T**
F	**T**	**F**	**T**	**T**	**F**	**T**
F	**F**	**F**	**T**	**T**	**T**	**T**

□

If propositions are linked together to form new propositions via logical operators, the result may be called a *composite* proposition. Propositions which are not presented as composites are known as *atomic* propositions, or *atoms*. It is critical to realize that the propositional calculus we are developing cannot tell us anything about the truth or falsity of atoms. However, if we know the truth value of atoms prior to applying the propositional calculus to some composite of them, it will tell us the truth value of that composite.

The proof of DeMorgan's Laws points out that even complicated composites have corresponding truth tables which relate the possible truth values of potentially unknown propositions to the truth value of the composite. In particular, suppose we do not know the truth values of p and q, and we let $r = \neg(p \wedge q)$ and $s = \neg p \vee \neg q$. Then $\mathbf{V}(r) = \mathbf{V}(s)$ regardless of the meaning of p and q.

Corollary 1. *The disjunction operator OR may be defined in terms of the negation operator NOT and the conjunction operator AND as*

$$\mathbf{V}(a \vee b) = \mathbf{V}(\neg(\neg a \wedge \neg b)).$$

Proof. Apply Assertion 1 to DeMorgan's First Law (take the NOT of both sides). □

We may think of the NOT operator as distributing into the AND operator, but when it does so it changes AND to OR. An analogous statement applies to the OR operator. However, we do have a actual distributivity of AND over OR and of OR over AND.

Theorem 2. *(Distributive Laws) For any two propositions p and q we have*

(1) $\mathbf{V}((p \vee q) \wedge r) = \mathbf{V}((p \wedge r) \vee (q \wedge r))$;
(2) $\mathbf{V}((p \wedge q) \vee r) = \mathbf{V}((p \vee r) \wedge (q \vee r))$.

Proof. The tables tell the story.

p	q	r	$p \vee q$	$(p \vee q) \wedge r$	$p \wedge r$	$q \wedge r$	$(p \wedge r) \vee (q \wedge r)$
T	**T**	**T**	**T**	**T**	**T**	**T**	**T**
T	**T**	**F**	**T**	**F**	**F**	**F**	**F**
T	**F**	**T**	**T**	**T**	**T**	**F**	**T**
T	**F**	**F**	**T**	**F**	**F**	**F**	**F**
F	**T**	**T**	**T**	**T**	**F**	**T**	**T**
F	**T**	**F**	**T**	**F**	**F**	**F**	**F**
F	**F**	**T**	**F**	**F**	**F**	**F**	**F**
F	**F**	**F**	**F**	**F**	**F**	**F**	**F**

p	q	r	$p \wedge q$	$(p \wedge q) \vee r$	$p \vee r$	$q \vee r$	$(p \vee r) \wedge (q \vee r)$
T	**T**	**T**	**T**	**T**	**T**	**T**	**T**
T	**T**	**F**	**T**	**T**	**T**	**T**	**T**
T	**F**	**T**	**F**	**T**	**T**	**T**	**T**
T	**F**	**F**	**F**	**F**	**T**	**F**	**F**
F	**T**	**T**	**F**	**T**	**T**	**T**	**T**
F	**T**	**F**	**F**	**F**	**F**	**T**	**F**
F	**F**	**T**	**F**	**T**	**T**	**T**	**T**
F	**F**	**F**	**F**	**F**	**F**	**F**	**F**

□

Intuitively we realize that AND and OR are *commutative* operators, which is to say that $p \wedge q$ means the same thing as $q \wedge p$ and $p \vee q$ is just another way of saying $q \vee p$. Thus we are content when we notice that our truth tables agree. It is also easily verified that AND and OR are *associative* operators, and we leave it to the reader to verify this.

Assertion 2. *(Commutativity Laws) For any two propositions p and q we have*

(1) $\mathbf{V}(p \wedge q) = \mathbf{V}(q \wedge p)$;
(2) $\mathbf{V}(p \vee q) = \mathbf{V}(q \vee p)$.

Assertion 3. *(Associativity Laws) For any propositions p, q, and r we have*

(1) $\mathbf{V}((p \wedge q) \wedge r) = \mathbf{V}(p \wedge (q \wedge r))$;
(2) $\mathbf{V}((p \vee q) \vee r) = \mathbf{V}(p \vee (q \vee r))$.

Commutativity and associativity do not hold for all of the commonly used logical operators. This brings us to the *implication* operator IMP ($\Rightarrow$), where we read $p \Rightarrow q$ as "p implies q" or as "if p, then q". We have a name for the components of an implication: p is called the *hypothesis* and q is called the *conclusion*. One may be surprised by the truth table of this logical operator the first time it is encountered:

p	q	$p \Rightarrow q$
T	**T**	**T**
T	**F**	**F**
F	**T**	**T**
F	**F**	**T**

TABLE 4. IMP Truth Table

A false proposition implies anything one wishes it to imply. Thus the proposition "If pigs fly, then the earth if flat" is true whether or not the earth is indeed flat. Just to get our feet wet with the implication operator, we assert the following, which may be verified directly from the truth tables.

Assertion 4. *If p and q are propositions, then*

(1) $p \Rightarrow (p \vee q)$;
(2) $(p \wedge q) \Rightarrow p$.

Assertion 5. *If p and q are propositions, then*

$$\mathbf{V}(p \Rightarrow q) = \mathbf{V}(\neg q \Rightarrow \neg p).$$

Theorem 3. *The implication operator IMP may be built from the negation operator NOT and the conjunction operator AND operators since*

$$\mathbf{V}(p \Rightarrow q) = \mathbf{V}(\neg(p \wedge \neg q)).$$

At this point you may be asking why we chose for $p \Rightarrow q$ to be true even when p and q are both false. The others choices in the truth table for implication are easily justified by common sense, but why this one? The answer lies in the truth table for the equivalence operator and the theorem which follows it, a theorem which we very much want to be true and which depends on this choice.

The *equivalence* operator IFF ($\Leftrightarrow$) signifies logical equivalence, so that $p \Leftrightarrow q$ is read "p is logically equivalent to q" or "p if and only if q". This is the operator that answers the question "do p and q have the same truth value?"

p	q	$p \Leftrightarrow q$
T	**T**	**T**
T	**F**	**F**
F	**T**	**F**
F	**F**	**T**

TABLE 5. IFF Truth Table

The following theorem justifies our double sided arrow notation.

Theorem 4. *If p and q are propositions, then*

$$\mathbf{V}((p \Rightarrow q) \wedge (p \Rightarrow q)) = \mathbf{V}(p \Leftrightarrow q).$$

Proof. We have a proof by truth table.

p	q	$p \Rightarrow q$	$q \Rightarrow p$	$(p \Rightarrow q) \wedge (q \Rightarrow p)$	$p \Leftrightarrow q$
T	**T**	**T**	**T**	**T**	**T**
T	**F**	**F**	**T**	**F**	**F**
F	**T**	**T**	**F**	**F**	**F**
F	**F**	**T**	**T**	**T**	**T**

□

Theorem 5. *The equivalence operator IFF may be constructed from the negation operator NOT and the conjunction operator AND because*

$$\mathbf{V}(\neg(p \wedge \neg q) \wedge \neg(\neg p \wedge q)) = \mathbf{V}(p \Leftrightarrow q).$$

At this point we may abandon our $\mathbf{V}(p)$ notation in preference to usage of the IFF operator, for it is clear that for any two propositions p and q, then $\mathbf{V}(p) = \mathbf{V}(q)$ is the logical equivalent of $p \Leftrightarrow q$. For example, the above claim could be written

$$\mathbf{V}((\neg(p \wedge \neg q) \wedge \neg(\neg p \wedge q)) \Leftrightarrow (p \Leftrightarrow q)) = \mathbf{T},$$

or simply

$$(\neg(p \wedge \neg q) \wedge \neg(\neg p \wedge q)) \Leftrightarrow (p \Leftrightarrow q),$$

since asserting the above is taken to mean asserting that it is true.

3. Additional Operators

In this section we introduce primitive logical operators which do not arise in ordinary language but which, nonetheless, arise from definitional truth tables which differ from those we have already encountered. These are XOR, NOR, and NAND.

The *exclusion* operator XOR ($\Updownarrow$) stands for exclusive OR and means a or b, but not both.

a	b	$a \Updownarrow b$
T	**T**	**F**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

Table 6. XOR Truth Table

The XOR operator is often denoted in the context of computer science with $\oplus$; that is, $p \oplus q$ means the same thing as $p \Updownarrow q$.

Assertion 6. *The XOR operator is the negation of IFF, i.e.,*

$$(a \Updownarrow b) \Leftrightarrow \neg(a \Leftrightarrow b).$$

The *alternate denial* operator NOR ($\Downarrow$) means "neither a nor b".

a	b	$a \Downarrow b$
T	**T**	**F**
T	**F**	**F**
F	**T**	**F**
F	**F**	**T**

TABLE 7. NOR Truth Table

Assertion 7. *The NOR operator is the negation of OR, i.e.,*

$$(a \Downarrow b) \Leftrightarrow \neg(a \vee b).$$

The *joint denial* operator NAND ($\Uparrow$) means "possibly a and possibly b, but not both".

a	b	$a \Uparrow b$
T	**T**	**F**
T	**F**	**T**
F	**T**	**T**
F	**F**	**T**

TABLE 8. NAND Truth Table

Assertion 8. *The NAND operator is the negation of AND, i.e.,*

$$(a \Uparrow b) \Leftrightarrow \neg(a \wedge b).$$

A collection of operators *generates* another operator if the truth table of generated operator can be derived through a combination of the generators. For example, we have already seen that NOT and AND together generate OR, IMP, and IFF. Since XOR is NOT IFF, NOR is NOT OR, and NAND is NOT AND, we can see that NOT and AND generate XOR, NOR, and NAND.

Theorem 6. *The operators NOT, AND, OR, IMP, IFF, XOR, and NAND may be derived from NOR.*

Proof. It suffices to show that NOT and AND may be written in terms of NOR. The definition of NOR and DeMorgan's Law gives us that

(1) $\neg a \Leftrightarrow (a \Downarrow a)$;
(2) $(a \wedge b) \Leftrightarrow (\neg a \Downarrow \neg b)$.

□

Theorem 7. *The operators NOT, AND, OR, IMP, IFF, XOR, and NOR may be derived from NAND.*

Proof. It suffices to show that NOT and AND may be written in terms of NAND. The definition of NAND and a glance at the truth tables gives us that

(1) $\neg a \Leftrightarrow (a \Uparrow a)$
(2) $(a \wedge b) \Leftrightarrow \neg(a \Uparrow b)$

□

4. Problems

Problem 1. Determine the truth table of the following composite propositions.

(a) $(p \vee q) \Rightarrow (p \wedge q)$
(b) $(p \wedge q) \vee (p \Rightarrow q)$
(c) $(p \Rightarrow q) \Rightarrow p$
(d) $p \Rightarrow (q \Rightarrow p)$
(e) $(p \Rightarrow q) \Rightarrow q$
(f) $p \Rightarrow (q \Rightarrow p)$
(g) $(p \Rightarrow q) \Rightarrow r$
(h) $p \Rightarrow (q \Rightarrow r)$
(i) $((p \Rightarrow q) \wedge (q \Rightarrow r)) \Rightarrow (p \Rightarrow r)$
(j) $(p \wedge q) \Leftrightarrow (p \Updownarrow q)$
(k) $(p \Uparrow q) \Rightarrow (p \vee q)$

Problem 2. Complete the proof of Assertion 5.

Problem 3. Write a logically equivalent statement using NOT, AND, and OR.

(a) $\neg(p \Rightarrow q)$
(b) $(p \Rightarrow q) \Rightarrow r$

Problem 4. Use truth tables to prove the following assertions.

(a) $(a \Updownarrow b) \Leftrightarrow \neg(a \Leftrightarrow b)$
(b) $(a \Downarrow b) \Leftrightarrow \neg(a \vee b)$
(c) $(a \Uparrow b) \Leftrightarrow \neg(a \wedge b)$

Problem 5. Show that the logical operators NOT and OR are sufficient to generate AND, IMP, IFF, XOR, NOR, and NAND.

Problem 6. Develop the truth tables for logical operators of one proposition other than NOT. You should get three of these, and you will see that they may reasonably be called identity, constant truth, and constant falsehood.

Problem 7. Develop the truth tables for logical operators of two propositions other than AND, OR, IMP, IFF, XOR, NOR, and NAND. You should get nine of these. Give these new operators names. Relate them to the operators of one proposition identity, constant truth, constant falsehood, and negation. Relate them to the operators of two propositions AND, OR, IMP, IFF, XOR, NOR, and NAND.

Department of Mathematics, BASIS Scottsdale
E-mail address: paul.bailey@basised.com