4. CALCULATIONS AND ALGORITHMS IN PROPOSITIONAL LOGIC

Remarkably, (despite his internet connections) **Al Gore** didn't invent any of the **algor**ithms in this chapter; the word "algorithm" comes rather from the name of an Islamic mathematician from the 9th century. His name is sometimes written **al-Khuwarizmi**; a more complete version, with slightly different Roman alphabet spelling is **Abu Ja'far Mohammed ibn Mûsâ al-Khowârizm**. An influential book of his dating from about 825 AD is entitled Kitab al-jabr wa'l-muqabala. The word *algebra* is derived from the obvious one of those Arabic words.

Here is a brief preview of a main topic from the initial five sections below. Suppose we came upon what looked like a truth table, but for some reason we couldn't read the formula whose truth values the table was supposed to have calculated. From the values in the table, it's not possible to *exactly* determine the formula. But one can at least determine a few formulae which are *truth equivalent* to it.

For example, here is such a table.

P	Q	R	$F = ?$
Tr	Tr	Tr	Fs
Tr	Tr	Fs	Fs
Tr	Fs	Tr	Tr
Tr	Fs	Fs	Fs
Fs	Tr	Tr	Fs
Fs	Tr	Fs	Fs
Fs	Fs	Tr	Tr
Fs	Fs	Fs	Tr

For this table, one such formula is

$$F = (P \wedge \neg Q \wedge R) \vee (\neg P \wedge \neg Q \wedge R) \vee (\neg P \wedge \neg Q \wedge \neg R)$$

Another is

$$F = (\neg P \vee \neg Q \vee \neg R) \wedge (\neg P \vee \neg Q \vee R) \wedge (\neg P \vee Q \vee R) \wedge (P \vee \neg Q \vee \neg R) \wedge (P \vee \neg Q \vee R)$$

Having written these down, you can directly check that both do have that truth table above. But how would you find them in the first place? I'll just say that the first came from looking only at the lines above giving Tr for F. And the second came from looking only at the lines above giving Fs for F. You might enjoy trying to puzzle out the method, before reading all about it in the following sections. The first is a DNF; the second, a CNF.

4.1 Disjunctive Normal Form

Definitions. A *DNF-constituent from* $P_1, \cdots, P_n$ is a string of the form $L_1 \wedge L_2 \wedge \cdots \wedge L_n$, where each L_i is either the literal P_i or the literal $\neg P_i$. A *literal*, to give a cute definition, is just a formula of length less than 3. But let's not be cute—the literals are simply the following :

$$P_1 \;,\; \neg P_1 \;,\; P_2 \;,\; \neg P_2 \;,\; P_3 \;,\; \neg P_3 \;,\; P_4 \;,\; \cdots\cdots \quad .$$

So, for example, $\neg P_1 \wedge P_2 \wedge \neg P_3 \wedge \neg P_4$ is one of the "$2^4 = 16$" DNF-constituents from P_1, P_2, P_3, P_4. The complete list of DNF-constituents from P_1, P_2 is

$$P_1 \wedge P_2 \quad , \quad \neg P_1 \wedge P_2 \quad , \quad P_1 \wedge \neg P_2 \quad , \quad \neg P_1 \wedge \neg P_2 \quad .$$

Note. For $n \geq 3$, such a constituent is not quite a formula, or even an abbreviated formula, since some pair(s) of 'associativity' brackets would need to be added to make it into a formula. But all the different formulae you'd get by adding brackets in different ways are equivalent, so we speak (loosely) of the 'formula' $L_1 \wedge L_2 \wedge \cdots \wedge L_n$.

Definition. A *disjunctive normal form* (or just *DNF*) from $P_1, \cdots, P_n$ is a string $(C_1) \vee (C_2) \vee \cdots \vee (C_r)$, where each C_j is some DNF-constituent from $P_1, \cdots, P_n$, and all the C_j are different from each other.

For example, $(P_1 \wedge \neg P_2 \wedge P_3) \vee (\neg P_1 \wedge \neg P_2 \wedge \neg P_3)$ is a DNF from P_1, P_2, P_3 in which the number, "r", of constituents is 2. There are "$2^{2^3} - 1 = 255$" other DNF's from P_1, P_2, P_3, if we ignore the order in which the constituents appear. (See the next paragraph.)

Note. Again, to get (an abbreviation for) a formula, one needs to insert more brackets. But all formulae you get by inserting them, in different possible ways, are equivalent. Also, changing the order of appearance of the constituents, C_j, will not alter the truth equivalence class of that formula. We refer (loosely) to a DNF as a "*formula in DNF-form*", sometimes adding the phrase "*from* $P_1, \cdots, P_n$". Since there are "8=2^3" DNF constituents from P_1, P_2, P_3, there are (ignoring bracketing and order of constituents) a total of "$2^8 = 256$" different DNF's from P_1, P_2, P_3 ; in general , "2^{2^n}" from $P_1, \cdots, P_n$. Note also that the convention we have given is, within each constituent, to keep the literals in their natural order with increasing subscripts.

Notice that *a DNF is true at some particular truth assignment* e *if and only if it contains a constituent all of whose literals are true at* e , partly since it's a *disjunction* of its constituents. But such a constituent clearly determines all the values of e at the relevant propositional variables (the first "n"), since that constituent is a *conjunction* of literals, and there's one literal corresponding to every relevant variable. And every other constituent (whether appearing in the given DNF or not) is false at that particular e.

For example, 'the' truth assignment e which makes $\neg P \wedge Q \wedge R \wedge \neg S$ true is given by

$$\mathsf{e}(P) = \mathsf{Fs} \quad , \quad \mathsf{e}(Q) = \mathsf{Tr} \quad , \quad \mathsf{e}(R) = \mathsf{Tr} \quad , \quad \mathsf{e}(S) = \mathsf{Fs} \quad .$$

The Fs's correspond to where a "$\neg$" occurs, of course. And every other DNF-constituent in P, Q, R and S is necessarily false at the e just above. And finally, e is only non-unique to the extent that it can do what it wants on variables other than P, Q, R and S.

Another way of saying this important general fact is as follows: In the truth table of a DNF-constituent from $P_1, \cdots, P_n$, that constituent has Tr in only one line, and Fs in all the "$2^n - 1$" other lines. The 'true line' is different for different constituents. And so, the number of lines where a DNF formula is true is the same as the number of constituents in that formula.

Just so that the next theorem can be stated elegantly, we shall allow the **empty DNF** to *be* a DNF. This is the one where $r = 0$ in the previous definition; that is, there are no constituents at all. Looking at the criterion for a DNF being true just above, we must say that *the empty DNF is false at* **every** e (because we cannot find a constituent all of whose literals are true at e, since there are no constituents at all!) So a formula corresponding to the empty DNF is any contradiction; that is, any unsatisfiable formula. (The empty DNF was included above when we did the counting.)

At the opposite end of the spectrum is the DNF which contains **every** constituent from the given relevant set of the first "n" propositional variables. It is true at every truth assignment, so corresponds to (and belongs to) the equivalence class consisting of tautologies.

Theorem 4.1 *If F is a formula not involving any P_i for any $i > n$, then F is truth equivalent to a DNF-formula from $P_1, \cdots, P_n$. Furthermore, that DNF is unique, up to rearranging the order of the constituents*
(and up to choice of bracketing, if you insist on writing it as an actual abbreviation for a formula)

We'll give a second proof and algorithm for this further down, one which doesn't use truth tables.

Proof and algorithm. By the remarks in the four paragraphs prior to the theorem, a pair of DNF's give a pair of truth equivalent formulae if and only if those DNF's have the same constituents. This establishes the uniqueness claimed in the second sentence of the theorem.

But those paragraphs also tell us how to find a DNF for a formula F. Write out a basic truth table for F, that is, one which has just "n" columns for the relevant propositional variables, plus a last column for F. Look down that last column for lines with "Tr" under F. For each such line (if any), write down one constituent, determined by looking at the other entries in that line. If "Tr" occurs under P_i, just use P_i as the literal L_i for this constituent. If "Fs" occurs under P_i, use $\neg P_i$ as the literal. This DNF (obtained by '$\vee$-ing' all those constituents, one for each line with a Tr under F) has exactly the same truth table as the given formula F, and is therefore truth equivalent to it by definition, as required.

Here is an example of the algorithm described in the above proof. I'll use the example $F = R \wedge \neg(\neg(P \wedge \neg Q) \wedge P)$ from the introduction (page 7), whose truth table was determined in the second chapter:

P	Q	R	F
Tr	Tr	Tr	Fs
Tr	Tr	Fs	Fs
Tr	Fs	Tr	Tr
Tr	Fs	Fs	Fs
Fs	Tr	Tr	Tr
Fs	Tr	Fs	Fs
Fs	Fs	Tr	Tr
Fs	Fs	Fs	Fs

Only the third, fifth and seventh lines in the truth table above have "Tr" under F, so there will be three constituents. The DNF is given below, with the constituents, from left to right obtained from those three lines in that order:

$$F \quad {}^{\text{treq}} \quad (P \wedge \neg Q \wedge R) \vee (\neg P \wedge Q \wedge R) \vee (\neg P \wedge \neg Q \wedge R) \quad .$$

This DNF formula clearly has a much greater degree of symmetry than the formula, $F = R \wedge \neg(\neg(P \wedge \neg Q) \wedge P)$, with which we started. Every process of real interest in the semantic part of logic essentially doesn't change if we replace a formula by an equivalent formula. (Of course, you can't just replace a formula on one line in a derivation by another one to which it is truth equivalent!) For many purposes, the DNF form of a formula, and/or the CNF form four sections below, are very convenient. For example, after calculating a few of these, you'll see how you more or less have the truth table built right into the DNF—you can just read off which are the e's where it's true (and the rest of the e's are where it's false). That comes from the example and the paragraph preceding it, before the statement of **4.1**.

We have treated P, Q and R as 'abbreviations' for the relevant variables P_1, P_2 and P_3, respectively, which occurred in the general treatment. In anything up to 5 or 6 variables, it's cleaner to get away from all those subscripts. One needs to use at least all the variables occurring in the formula. The only time you want to include some other variables, each of which will double the number of constituents, is for the purpose of comparing the DNF's of several formulae, which may involve slightly different variables. Then you write a DNF for each of them using all the variables occurring in all of them. By renaming things, these can always be taken to be the *first* "n" variables for some n. Alternatively, in the general treatment, we clearly could have been dealing with any sequence of "n" variables, not just the first "n" in their natural order. If it's necessary to add one or more extra variables which don't occur in the formula being dealt with, the algebraic manipulation procedure in the third section of this chapter (rather than use of truth tables) is the best way to 'pump up' a DNF, throwing in an extra variable which actually didn't occur in the formula itself. Effectively you just add both the variable and its negation, producing two constituents out of each constituent which occurred before adding the 'irrelevant' variable.

This chapter's introductory paragraphs give a second example of a DNF calculated by the truth table method.

Corollary *The number of truth equivalence classes of formulae in* $P_1, \cdots, P_n$ *is* 2^{2^n} . This includes formulae which don't actually involve **all** those propositional variables.

This is clear, since that's the number of DNF's. (The set of constituents has 2^n elements, and each DNF is essentially a *subset* of that set, of which there are 2^{2^n} .)