Mathematical Logic- Converting a formula from truth table to CNF / DNF

August 10, 2022

We have studied syntax and semantics. We now show hot to convert a formula given in semantic form (truth table) into a formula in syntactic form. We will provide two methods for this construction, resulting in one case in a formula in Conjunctive Normal Form (CNF) and in the other case a formula in Disjunctive Normal Form (DNF).

We first look at the concept of **sensitivity of boolean functions** before giving the algorithms for generating a syntactic formula from a semantic one.

We say a boolean formula is very sensitive under a particular assignment of truth variables to its variables if changing the truth value of any one variable alters the truth value of the formula. The following are examples:

1. The generalised AND (AND applied to several literals) is very sensitive under the assignment in which it evalutes to true (that is all the literals are set to true).

2. The generalised OR (OR applied to several literals) is very sensitive under the assignment in which it evalutes to false (that is all the literals are set to false).

3. The generalised XOR (XOR applied to several literals) is very sensitive under **every** assignment.

4. The implication ($\Rightarrow$) is very sensitive under the assignment that makes it false.

5. The generalised implication ($\Rightarrow$) is ambiguous as the $\Rightarrow$ logical connective is neither commutative nor associative.

We took a detour from the main content of the lecture and counter the number of ways of bracketing an iterated application of a non-associative

operator. We assumed a sequence of k variables and applying the operator $k-1$ times. Using the formula tree concept introdced in an earlier lecture, we saw that the number of bracketings is the same as the number of binary trees on $k-1$ nodes, when looking at a sequence of k variables and $k-1$ applications of the operation.

Since we mapped the number of bracktings to the number of binary trees on a given number of nodes, we defined rooted binary trees inductively and also wrote a recurrence relation for the number of binary trees on n nodes. We then gave the solution of the recurrence relation, without the details of how it was obtained. This **might be** covered later on in the course.

Structural Inductive definition of an n node binary tree.

Base Case: There are two binary trees that are directly defined. These are

- The empty binary tree on zero nodes. There is exactly one such tree.
- The one node binary tree where the single node is both a the root and a leaf. This treeis also unique.

Inductive step: For any $n \geq 2$, the number of binary trees on n nodes is given by the recurrence relation

$$B_n = \Sigma_{t=0}^{n-1} B_t \times B_{n-1-t}$$

Here B together with an integer variable subscript indicates the number of binary trees on that many nodes. We expressed the value for n in terms of partitioning the remaining $n-1$ nodes (other than the root) in terms of how many go to the left and how many go toi the right. The total of these numbers must be $n-1$ and every break of this integer into two non-negative integers works. Thus we apply the **counting technique** called **rule of sum** in writing down the index variables of the summation. The choice of structures of the subtreeson te left and the right of the root are independent of each other. Thus we multiply the number of possibilities for the left by the number of possibilities for the right, for each partitioning of the integer $n-1$. This latter counting technique is called the **rule of product**.

Solving this recurrence relation gives the answer:

$$B_n = \frac{1}{n+1}\binom{2n}{n}$$

This part of the lecture, although not in the original flow, constitutes an important part of the course material.

Coming back to the main goal, the sensitivity rules 1 and 2 given above will be used to constructthe formula s DNF and CNF respectively.

DNF Construction Rule

1. Consider only the rows of the truth table in which the formula evaluates to true.

2. Create a clause for each such row where the clause is the conjunction of all variables that are set to true and the negarions of all variables set to false. The assignment corresponding to this row is the only one that will make this clause true.

3. Take the disjunction of all the clauses constructed in step 2 as the formula in DNF.

Explanation: Since the top level operator of the formula is OR, the formula evaluates to true only if at least one of the clauses evaluates to true. Each clause evaluates to true **only** in the assignment corresponding to the row from which it was created. Since the clauses were formed using only the rows where the formula evaluated to true, the constructed formula is true in exactly those assignments. Thus the syntactic formula corresponds to the truth table.

CNF Construction Rule

1. Consider only the rows of the truth table in which the formula evaluates to false.

2. Create a clause for each such row where the clause is the disjunction of all variables that are set to false and the negarions of all variables set to true. The assignment corresponding to this row is the only one that will make this clause false.

3. Take the conjunction of all the clauses constructed in step 2 as the formula in CNF.

Explanation: Since the top level operator of the formula is AND, the formula evaluates to true if and only if **all** the clauses evaluate to true. Each clause evaluates to true **only** in all assignments except the one corresponding to the row from which it was created. For all the clauses to be true, and thus the formula to evaluate to true, we must have any assignment other than the rows from which the clauses were constructed. This exactly corresponds to the assignments in which the truth table shows true, since the clauses were constructed from the rows where it was false.

In order to generate an example formula randomly, we used the roll number of one student converted to a binary number of 8 bits. Each student is advised to practice this using their own roll number. This has the advantage of learning one of the skills of this course (converting a number from decimal to binary) as well as each student practising the conversion from truth table to syntactic form using a different formula than all other students.

Students may observe that for a formula that evaluates to true in smaller number of assignments, the DNF formula will be smaller and simpler. Similarly for a formula that evaluates to false in smaller number of assignments, the CNF formula will be smaller and simpler. For formulae where the number of assignments in which it evaluates to true is the same as the number in which it evaluates to false, both formulae will be big and complex.