An Enquiry Concerning Logical Understanding

*Daniel Soutar*

*Table of Contents*

1. Preface (pg. 3)
2. Of Logical Methods (pg. x)
3. Of Atomic Formulae (pg. xx)
4. Of Negation and Double Negation (pg. xxx)
5. Of Conjunction (pg. xxxx)
6. Of Disjunction (pg. xxxx)
7. Of Conditionals and Biconditionals (pg. xxxx)
8. Of Explicit and Implicit Parentheses (pg. xxxx)

Preface

This paper aims to give a complete and formal account of our *method* in analysing logical statements. Whilst we are often given a sufficient explanation in how we analyse propositions and their logical relations with each other, this explanation is frequently vague and often expressed in a language that allows for ambiguities.

Given that this paper aims to ultimately be a manual on how to translate sentential or propositional logic (PL) into a computer program, the author hopes that a precise understanding of the process of analysing wffs will be of mutual benefit to all parties concerned with the field of Logic.

Of Logical Methods

When we are given an arbitrary input, how is it that we know that the input is a wff that is confined within the syntactic rules of PL?

Read and Wright give a convincing account of how we approach this problem:

* A formula consisting of an individual proposition variable is a wff;
* if A is a wff, then ~A is a wff.
* if A and B are both wffs, so is (A & B).
* if A and B are both wffs, so is (A v B).
* if A and B are both wffs, so is (A → B).
* if A and B are both wffs, so is (A ↔ B).
* no formula is a wff unless its being so can be shown by the above.

However, this paper now aims to make explicit how exactly we actually apply these rules. For instance, the following is a wff:

(P & ~Q) → R

But how do we show it? We have the ‘smallest-wff’ approach:

We take the input (P & ~Q) → R, examine the input from beginning to end, so (, P, &, Q… and determine, from the brackets, what the smallest wff is.

\*The smallest wff will always be the one enclosed in the largest number of brackets. If there are several that tie for the smallest wff, we will determine them from right to left. This is because our incrementing function would rise and fall and only record the rightmost of the smallest wffs.\*

Then we evaluate the wff by the following rules in this order (whereby if any is satisfied, we turn the mini-wff in question into a placeholder *n* to signify that it has been evaluated):

* If there are no connectives, then the wff can only be a single letter denoting a proposition *P*.
* If there is one and only one connective, the entire wff is the smallest-wff by definition.
* If there is more than one connective, we allocate a ranking whereby:
  + ~ is immediately assigned to whatever is on its right-hand side, with the lowest ranking (1). We should impose the restriction that there can only be, at most, one negation without having brackets involved. This is because there is no logical reason for requiring any more than this number (as the maximum number of one is sufficient in expressing different truth-values)
  + & and v are both assigned to whatever is on their left and right-hand sides (they cannot overlap, for they have equal ranking). Since we start from the smallest-wff first, we will always have a letter on either side. Anything else will be rejected. They both have a ranking (2).
  + → is immediately assigned to whatever is on its left and right-hand sides, with ranking (3).
  + ↔ is immediately assigned to whatever is on its left and right-hand sides, with the highest ranking (4).

*“Let us assign a placeholder variable n where n represents an evaluated wff. Thus:*

*n* → R

*represents a sequent whereby* (P & ~Q) *is represented by n.”*

60 test cases:

* P (Wff)
* Q (Not a wff – must include P)
* (P) (Wff)
* )P( (Not a wff – brackets the wrong way round)
* (((P))) (Wff)
* (()P()) (Not a wff – see above)
* (((P))) & (((Q))) & (((R))) … (Wff, but would be evaluated by the following: *n*, R, Q, P, N, N, N, N… N & N & N …)
* (((P))) & ) (((Q))) … (Not a wff, conjunction lacking a conjunct)
* ~P (Wff)
* ~~P (Technically not a wff – can be represented by P. This means that if a negation is succeeded by another negation, return false)
* ~P~ (Not a wff – negation does not contain a letter afterwards, needs another connective)
* ~P~Q (Not a wff – would evaluate to N, N, NN. This means that it is illegal for a formula to be of the form NN. How do we represent this? By stipulating that no two N can be directly next to each other – they must have a two-argument connective in between them.)
* P & Q (Wff)
* ~P & ~Q (Wff, but would be evaluated by the following: ~Q, ~P, N & N)
* ~P & Q (Wff, but would be evaluated by the following: ~P, N & N)
* P ~& Q (Not a wff – negation always has a letter afterwards)
* ~(P & Q) (Wff, but would be evaluated by the following: P & Q, ~N)
* ~P (& Q) (Not a wff – conjunction lacking a conjunct)
* (~P & Q) & R (Wff, but would be evaluated by the following: ~P, N & Q, N & R)
* P v Q (Wff)
* ~P v ~Q (Wff, but would be evaluated by the following: ~Q, ~P, N v N)
* ~P v Q (Wff, but would be evaluated by the following: ~P, N v N)
* P ~v Q (Not a wff – negation always has a letter afterwards)
* ~(P v Q) (Wff, but would be evaluated by the following: P v Q, ~N)
* ~P (v Q) (Not a wff – disjunction lacking a disjunct)
* (~P v Q) v R (Wff, but would be evaluated by the following: ~P, N v Q, N v R)
* P → Q (Wff)
* P → (Q) (Wff, but would be evaluated by the following: Q, P → N)
* P → ~Q (Wff, but would be evaluated by the following: ~Q, P → N)
* P → ~(Q) (Wff, but would be evaluated by the following: Q, ~N, P → N)
* ~(P & Q) → (~Q v ~P) (Wff, but would be evaluated by the following: ~Q, ~P, N v N, P & Q, ~N, N → N)
* Q → P (Wff)
* P → (P → Q) (Wff, but would be evaluated by the following: P → Q, P → N)
* P → ~(P → Q) (Wff, but would be evaluated by the following: P → Q, ~N, P → N)
* P → (~P → Q) (Wff, but would be evaluated by the following: ~P, N → Q, P → N
* (P → Q) & (Q → P) (Wff, but would be evaluated by the following: (Q → P, P → Q, N & N)
* P ↔ Q (Wff)
* (P ↔ Q) (Wff, but would be evaluated

37 ~(P & Q) → (~Q v ~P)

Of Atomic Formulae

Of Negation and Double Negation

Of Conjunction