Quarterly Progress Report of the Research Laboratory of Electronics,
M.I. T., January, 1963

B. AN ORDINARY LANGUAGE INPUT FOR A COMIT PROOF-PROCEDURE PROGRAM

Work is in progress on a COMIT program for the translation of arguments from ordinary language into logical notation. This is conceived as an input device to a COMIT program, based on the Davis-Putnam proof-procedure algorithm, which tests quantified and nonqualified logical formulae for validity by the reductio ad absurdum method of attempting to deduce a contradiction from the negation of a formula.[1-3]

The translation program works essentially as follows.

(a) It divides the words of the input argument into two basic categories — the punctuative words, or "P-words," words like 'if', 'then', 'either', 'or', 'therefore', etc. which usually divide sentential clauses from one another, and the nonpunctuative words or "W-words," which are all the rest and constitute the content of the sentential clause. Accordingly, every word or punctuation mark in the input argument is subscripted, by means of a dictionary lookup, with either 'P' or 'W. In the sentence

If it rains then it pours.

the P-words are 'if', 'then', and '.', and the W-words are 'it', "rains', and 'pours'. A sentential clause is any finite sequence of W-words occurring between two P-words in the example, 'it rains' and 'it pours' are the two sentential clauses.

(b) It performs some simplifications and expansions on the input sentences. All verbal forms are reduced to the infinitive, so that 'If it rains then it pours', 'If it will rain then it will pour', 'If it is raining then it is pouring', 'If it would rain then it would pour', etc. (all of which state basically the same implication) are all reduced to 'If it rain then it pour'. Sentences in which several subjects are attached to the same predicate, or in which several predicates are attached to the same subject, are expanded in their implicit sentential clauses. Thus, 'Jack and Jill went up the hill' is expanded in 'Jack went up the hill and Jill went up the hill'; and 'Jack both fell down and broke his crown' (the program requires the 'both' as the cue to activate the expansion subroutine is expanded into 'Jack fell down and Jack broke his crown'. Sentences using 'either. . or. . . ' and 'neither. . .nor. . . ' constructions are likewise expanded whenever possible. A fallacy would result if this type of expansion were applied, e.g. , to 'Jack and Jill are cousins'; thus the input arguments must be stated without using relational sentences of this sort. Later, we hope to improve the program to handle them.

(c) It substitutes the letters 'A', 'B', 'C', etc. , for the sentential clauses in the argument, by using the same symbol for the same sentential clause whenever it occur

(d) It parenthesizes the argument, on the basis of a set of 20 rules that make explicit the groupings that are implicit in the use of the P-words. Thus, 'If A then B or if C then D' is parenthesized (by using the arrow for implication and the wedge for disjunction) as '(((A)→(B)) V ((C→(D)))'. In COMIT notation this formula appears as '*(+ *(+ *(+ A + *) + IMPLIES + *(+ B + *) + *) + OR + *(+ *(+ C + *) + IMPLIES + *(+ D + *) + *) + *) '. A logical argument is usually stated as a set of premises in the form of sentences separated by periods, which is followed by a conclusion introduced by a word like 'therefore' or 'hence'. Our program symbolizes an argument as one long sentence in the form of an implication, the 'therefore' (or 'hence', etc.) being replaced by 'implies', and the periods being replaced by 'and'.

The characteristics of the restricted English employed follow from the limitations on what the program can do. The restrictions fall into three general categories:
(i) Restrictions following from the fact that the program's only criterion of identity of two propositions or "ideas" is the identity of the sentences expressing them. Two propositions stated in different words, even though they may be synonymous, are symbolized with different letters, unless one or more of the subroutines mentioned in (b) above result in the sentences being expressed in the same wording and word order.
(ii) Restrictions on the use of the P-words and the C-words (which are a subset of the P-words and include the binary connectives 'and', 'or', etc.). The P-words that are at the same time C-words may be used only if a boundary is intended between two sentential clauses, so that a sentence like 'He implies that it will not rain' is ruled out, since the 'implies that' will fallaciously divide the sentence into two sentential clauses, 'He' and 'it will not rain', which are connected by an implication sign. Also, every intended division between two sentences must be made explicit, so that 'If it rains it pours' must have a 'then' inserted between 'rains' and 'it' before the program will correctly handle the sentence.

(iii) Restrictions on the vocabulary. The program employs no grammatical recoglition routine to speak of; thus a fixed vocabulary must be selected in advance, with all of the nouns, verbs, and adjectives specified. A given word may not be used in more than one of these categories; e.g., if 'praise' is used in the set of arguments submitted, it must be used either as a noun or as a verb, but not both.

The program has been successfully tested on a variety of examples taken, with the required changes in wording, from Copi.[4] Some of the improvements contemplated for he program in the future include coupling the program with a more powerful grammar, and enabling it to perform the more subtle intraclause analyses required for quantificational and relational logic. The proof-procedure program, based on the Davis-Putnam algorithm, is finished and runs reasonably well. It will do propositional, quantificational, and relational logic. Other improvements in this program will have to await improvements in the theory of proof procedures. The ordinary language input program discussed in this report will, however, do only some of the analyses required for propositional logic.

J. L. Darlington

References

1. M. Davis and H. Putnam, A computing procedure for quantification theory, J. Assoc. Computing Machinery 7, 201-215(1960).

2. J. Darlington, A COMIT program for the Davis-Putnam algorithm, Internal Memorandum, Mechanical Translation Group, Research Laboratory of Electronics, M. I.T. , May 1962.

3. J. Darlington, A restricted English for the formulation of arguments in propositional logic, Internal Memorandum, Mechanical Translation Group, Research Laboratory of Electronics, M.I.T., November 1962.

4. I. M. Copi, Introduction to Logic (Macmillan and Company, New York, 1955).