November 1969

Brief 69-10656

AEC-NASA TECH BRIEF

Cogent Programming Manual

The problem:

To achieve both the programming conciseness of a syntax-directed compiler system and the full generality of a recursive list-processing program.

The solution:

The COGENT (COmpiler and GENeralized Translator) programming system is a compiler whose input language is designed for the description of symbolic and linguistic manipulation algorithms. Although the system is intended primarily for use as a compiler-compiler (a compiler that compiles other compilers), it is also applicable to such problem areas as algebraic manipulation, mechanical theorem-proving, and heuristic programming.

An initial version of the COGENT system has been written for the Control Data-3600 computer. The manual (ref. 1) is intended primarily to describe the input language for this system; operating procedures and details of the specific machine implementation are not discussed.

How it's done:

The COGENT system is a compiler that runs on the CD-3600 and translates COGENT language into 3600-assembly language. When the system is used as a compiler-compiler, its output is a compiler that must run on the 3600 but which may translate an arbitrary input language into an arbitrary output language; e.g., numeric or symbolic code for any machine.

Fundamentally a program compiled by COGENT is a list-processing program in which the list structures represent phrases of object language. The correspondence between the strings of object language, which appear externally on input-output media, and the list of structures, which represent these strings within the computer, is determined by the syntax of the object languages. Thus the COGENT language itself contains two types of structure: *productions*, which specify the syntax of the object languages; and *generator definitions*, which specify list-processing procedures called generators. The format of the generator definitions is designed to let the programmer think directly in terms of the phrases of language that are being manipulated, rather than of the list structures that actually represent these phrases.

The main routine of a COGENT program is always a syntax-analyzer, which is compiled from the productions describing the input object language. This analyzer reads character strings from the input medium and converts them into list structures. At certain points in this process, the analyzer calls a generator to operate on the currently recognized sublist; the result of this generator then replaces the current sublist in the list being constructed. In addition to being called by the syntax-analyzer, generators may call each other and may call themselves recursively. Ultimately, primitive (built-in) output-generators are called to decompose the list structures back into character strings and write these strings on the output medium.

Reference:

1. Reynolds, J. C.: COGENT Programming Manual. ANL-7022, Argonne National Laboratory, March 1965.

Notes:

1. This innovation may interest designers or manufacturers of computers or computer software.

(continued overleaf)

2. Inquiries concerning this innovation may be directed to:
   Office of Industrial Cooperation
   Argonne National Laboratory
   9700 South Cass Avenue
   Argonne, Illinois 60439
   Reference: B69-10656

Source: J. C. Reynolds
Mathematics Division
(ARG-10463)

Patent status:

Inquiries concerning rights for commercial use of this innovation may be made to:

Mr. George H. Lee, Chief
Chicago Patent Group
U.S. Atomic Energy Commission
Chicago Operations Office
9800 South Cass Avenue
Argonne, Illinois 60439