Checking a table which has conformed to Pollack's convention is child's play. Completeness and ambiguity are easily checked because of the bifurcation principle upon which the table has been developed.

By the way, Mr. King's limited entry table examples are all mixed entry since the action entries are shown in extended entry format!

Yours faithfully,

D. A. CARRICK

15 Rockfield Avenue
Ware
Hertfordshire
25 November 1969

Reference

KING, P. J. H. (1969). The interpretation of limited entry decision table format and relationships among conditions, *The Computor Journal*, Vol. 12, No. 4, pp. 320–326.

To the Editor
The Computer Journal

Chain processing in PL/1

Sir,

I would like to suggest some features that could be implemented in PL/1 to make list processing easier and less liable to error for programmers not well versed in the field. Although most of the generality of PL/1 list processing is retained by the suggestions below, pointer manipulation provides many facilities not covered here, and the following is not considered as a replacement for pointers.

A unified data type (like LISP lists) and a small number of operating concepts are essential for simplicity. The data type suggested by the PL/1 list processor is the *chain*, a series of *items* (structures, as at present, containing data fields) connected up and down by pointers. In a chain processor, the pointers will not be manipulated directly by the programmer, but only indirectly, through the operations he performs on the chain. Such operations are creating or deleting items in or from the middle or ends of a chain. The data fields of only one item in a chain can be accessed at a time, and this can be thought of as the *current* item. Thus the chain processor requires an operation of moving up and down the chain from the current item, making the item above or below current. Among the data fields is allowed the name of a *subchain* (or names of subchains). If the subchain is the same as the first (i.e. has the same name), a bifurcating tree results. In the case of plural subchains of the same name, we get a polyfurcating tree. If we add to the chain processor a facility for linking a chain to itself or to other chains, rings and other interconnected structures may be built up.

These concepts give very simply a wide range of list processing facilities. Only four basic operations are required, whose names suggest their functions: CREATE_ITEM, DELETE_ITEM, MOVALONG and LINK_CHAIN. In addition, the chain processor would require a declarator DECLARE_CHAIN and two logical functions, END_CHAIN and NULL_CHAIN (to detect the top or bottom item, or when a chain has no items).

Operands for these facilities would include: the name of the chain; ABOVE, BELOW and SUB for CREATE_ITEM (to place a new chain is to be a subchain); UP, DOWN, TO TOP, TO BOTTOM, TO (key) for MOVALONG (where the key is a logical condition on the data fields, to be satisfied by the item sought); and assignments to the data fields of new items.

I believe that the above suggestions are of general interest, and that in practice they would add to the appeal of PL/1 and to the ease of teaching it. Since they have the effect of restricting PL/1 to a more streamlined form (from the user's point of view), they can be implemented experimentally in PL/1, using the macro facilities of the language. Such an experiment is in progress. While this work cannot indicate the efficiency of a chain processor, implemented for example as a part of PL/1, it will give an idea of its usefulness.

Yours faithfully,

T. H. MERRET

Upper Flat
Craigmore
Shore Road
Skelmorlie
Ayrshire
28 November 1969

To the Editor
The Computer Journal

Further comments on a line-thinning scheme

Sir,

In a recent communication Mr. E. S. Deutsch has suggested some modifications to the set of rules given by Rutovitz, the realisation of which yields a line-thinning algorithm (this *Journal*, Vol. 12, p. 412). The complete set of rules now assumes a somewhat formidable appearance, and I would like to present a simpler alternative which I have been using for several months.

The line-thinning scheme which is suggested here examines only edge-points and so requires an edge-following routine which yields a table or set of tables giving points on the edge or edges of the pattern. These points are examined to see if they can be eliminated (set to binary zero in the pattern), a new set of tables defining edges is found, and the process is continued until no further elimination is possible. The edge-following routine gives both external and internal boundaries, the latter being found by edge-following after inverting the bits inside and on the boundary of the closed region defined by an external boundary.

The crossing-number χ is as defined by Rutovitz but in calculating χ it is to be noted that if $\gamma(1)$ and $\gamma(3)$ are both 1 the decision on whether a point can be eliminated or not does not depend on $\gamma(2)$. In such a case $\gamma(2)$ is considered to be 1. Similar statements apply in the other three quadrants.

With this modification in the calculation of χ, unless $\chi = 2$ the point cannot be eliminated. If $\chi = 2$, a further investigation is required before deciding to eliminate, since an end satisfies this condition and there must be a means of ensuring that further iteration does not simply shorten a skeleton line.

A line end satisfies the additional condition

$$\sum_{k=1}^{8} \gamma(k) = 1$$

but it is desirable to eliminate some points for which the conditions

$$\chi = 2 \qquad \sum_{k=1}^{8} \gamma(k) = 1$$

are satisfied. The size of the original picture area (512 × 512 points) makes it impossible to keep the original picture in store and to build a separate modified picture. The elimination process is used to modify the original picture, and unless some care is taken 'spikes' are generated due to minor irregularities in the edge of the initial pattern.

The scheme has been applied to samples of all the digits, and to certain other patterns to which line-thinning is appropriate. Uniformly good results have been obtained and are available as graph-plotter drawings. The plotter drawings show that the final irreducible line is centrally