A PROGRAM FOR GEOMETRICAL PATTERN RECOGNITION BASED ON THE LINGUISTIC METHOD OF THE DESCRIPTION AND ANALYSIS OF GEOMETRICAL STRUCTURES

Valentina Gall6
Computer and Automation Institute,
Hungarian Academy of Sciences
Budapest, Hungary

Abstract

The organization of a program for linguistic analysis of geometrical patterns is described. The program belongs to the system of pattern recognition for industrial drawings and objects [1,2,7]

We started from the formalism suggested by Evans [3,4]. In completing practical tasks, we have developed the methods of economical grammar construction, exact description of various geometrical relations between pattern elements and our program implementation. The program is written in FORTRAN with the use of the LIDI-T2 list processing system L5,63.

The grammar is represented by a stiitic lint structure. Geometrical relations and structural transformations of quantitative information, included in the sytitem, are implemented as a set of subroutines. The program algorithm to organize the analysis is grammar-independent, but the passing of the program control to the program segments to process the quantitative information is controlled by the grammar and vice-versa, i.e. the taking of information out of the grammar is controlled by the results. of the processing of the quantitative information about the actual object (and by the given atrategy of the use of the grammar as well).

Introduction

The program is aspart of the software of the industrial pattern recognition system presented by T.Vamos and Z.Vassy in their papers CI,71. The program variant running at present on the CDC-3300 computer is used for the recognition of two-dimensional patterns, like those .shown on Fig.*1* .

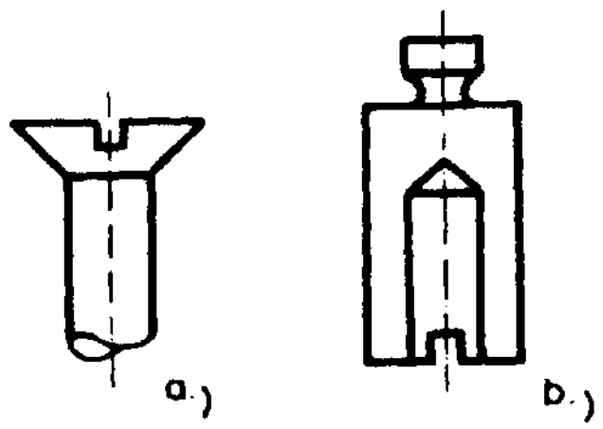


Fig.1

It is typical for the patterns that they consist of line segments, arcs and - because of the noise of the perceiving device - small indefinite elements. The geometrical relations between the pattern elements are complex and various: equivalence of end points of line segments and arcs, parallelness and perpendicularity of lines, equality and non-equality of lengths of segments and curvatures of arcs, a certain disposition of points with respect to a particular line, etc.

The input information for the program is a list of the primitives of the geometrical pattern, viz. line .segment (hereinafter: line), arc and small indefinite element (Fig.2) .

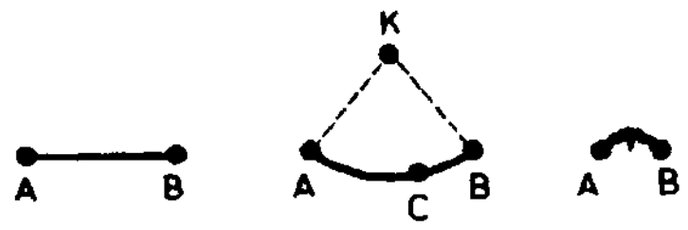


Fig.2

The description of the primitive consists of its identifier arid quantitative attributes:

co-ordinate of the ends $A=\begin{Bmatrix} x_A \\ y_A \end{Bmatrix}$ and $B=\begin{Bmatrix} x_B \\ y_B \end{Bmatrix}$

co-ordinate of the centre $K=\begin{Bmatrix} x_K \\ y_K \end{Bmatrix}$ (for arc only)

co-ordinate of an arbitrary point $C=\begin{Bmatrix} x_C \\ y_C \end{Bmatrix}$ (for arc only)

An input list in either obtained in the form of binary information from a small computer (C11 10010) as a result of the feature extraction procedure on a real object (*as* to the obtention and organization of this list of . [2]), or it may be a manually prepared list of primitive:; of an ideal geometrical object. 'J'he origin and organization of the input list are not relevant for the working of the program.

The objec t may be described by beginning with an arbitrary node and in an arbitrary direction. Every primitive is supposed to be described only once.

The analysis of the input list is performed on the basis of grammar - the structural description of patterns "familiar" to the computer. We have started from the formalism suggested by Evans [3,4] and we have developed it in the sense of the possibility of the exact description of the various geometrical

relations between subpatterns, the economical construction of grammars and the program implementation.

The terminal symbols of the grammar correspond to the primitives of the patterns, while the non-terminal symbols correspond to the subpatterns. The non-terminal symbols like the terminal ones huve quantitative characteristics, too. The rewriting rules have the following form:

Left-side non-terminal symboJ^i(Finite set of right-side terminal and non-terminalsymbols){Relationsbetween right-side symbols or between their quantitative attributes)(Definition of the quantitative attributes *of* the left-;.;ide non-terminal symbol by the quantitative attributes of the right-side symbols)(Indication of the symmetrical quantitative attributes)*

The geometrical relations permitted in the system and the operations to obtain the quantitative attributes of the non-terminal symboiti are described separately and interpreted in the system us sepainte program segments. Their set can be conveniently extended as much as the system develops.

Figure 3 describes a characteristic screw portion represented in Fig.1. Here the terminal symbol:; consist of lower case letters, and the non - terrnina ones of capital letters. The quant itativ attributes of the left-side symbol:; are denoted by unsubscripted letters, those of the right-side symbols by subscripted letters; the subscript indicates the ordinal number of the corresponding right-side symbol in the rule. The rules are numbered.

The output of the program in the course of its operation as an independent program (e.g. to test new grammars) is the printing of the names of the patterns recognized by the program on the input list as well as that of the description of the complete analysis tree. By using the program in system [1] only the end results are transmitted through the common data field for other programs.

Terminal symbols:
arc, its quantitative attributes:
A, B, C, K (Fig.3a)

Non-terminal symbols:
ARC (Fig.3a), ARC2(Fig.3b), LOOP (Fig.3c)
LA (Fig.3d)

Rewriting rules:

0/ $ARC \rightarrow |(arc)()(A=A_1,\ B=B_1,\ \varphi=ARCTG(A_1,B_1),\ K=K_1,\ C=C_1)(A,B)|$

1/ $ARC2 \rightarrow |(ARC,ARC)(A_1=A_2,\ B_1\neq B_2, K_1\neq K_2, |\varphi_1-\varphi_2|>90^\circ)\ (A=B_1,\ B=B_2,\ K=K_1,\ L=K_2,\ C=A_1)|$

2/ $LOOP \rightarrow |(ARC2,\ ARC)(A_1=A_2,\ B_1=B_2, K_1\neq K_2,\ L_1\neq K_2)\ (A=A_1,\ B=B_1,\ C=C_1)(A,B)|$

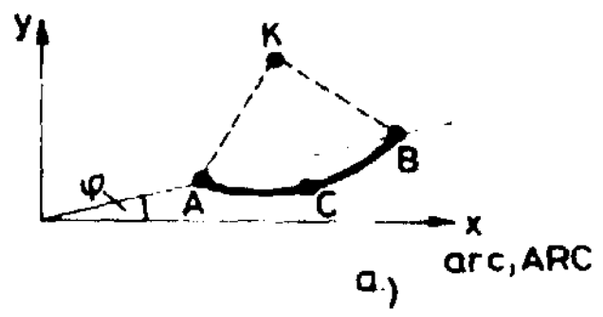
3/ $LA \rightarrow |(LOOP,\ ARC)(A_1=A_2, B_1\neq B_2,\ C_1\neq B_2)|$

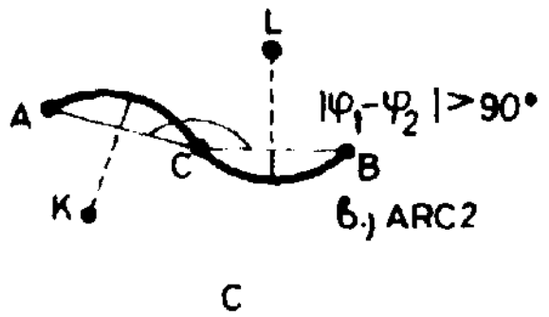
In the system there are program segments to check the geometrical relations denoted here:

$=,\ \neq,\ |\varphi_1-\varphi_2|>90^\circ$

and to realize the assignment operations, denoted here by

$=,$ ARCTG





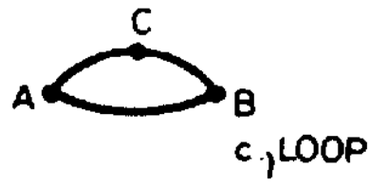


Fig.3

* Note: For all terminal symbols the quantitative attributes A and B (co-ordinates of the ends) are functionally identical - let us call them "symmetrical" for convenience. By analogy, there can be defined "symmetrical" quantitative attributes for non-terminal symbols, too. The definition of the symmetrical attributes enables us to reduce the number of the rewriting rules and allows for the arbitrariness proceeding in the direction of the contours.

The programming language

The program is written in FORTRAN with the use of the LIDI-72 list processing system developed in the Computer and Automation Institute, Hungarian Academy of Sciences [5,6] LIDI-72 is written in FORTRAN.

The Gystera LIDI-72 is applicable for the direct representation of arbitrary directed graphs in the computer. The node may have a name, with a node of the graph some information can be stored in the form of a linear sequence of any length, consisting of integer numbers, and the number of branches may be arbitrary. Fig. 4 represents the graph node by a LIDI-72 list element.

The LIDI-72 system has an advanced program system to operate on lists: the input of information vith a tree structure, described in parentheses and some other specific forms, the modification of any graphs, the conditional or unconditional carrying out of operations specified by the user on a certain sub-graph, etc.

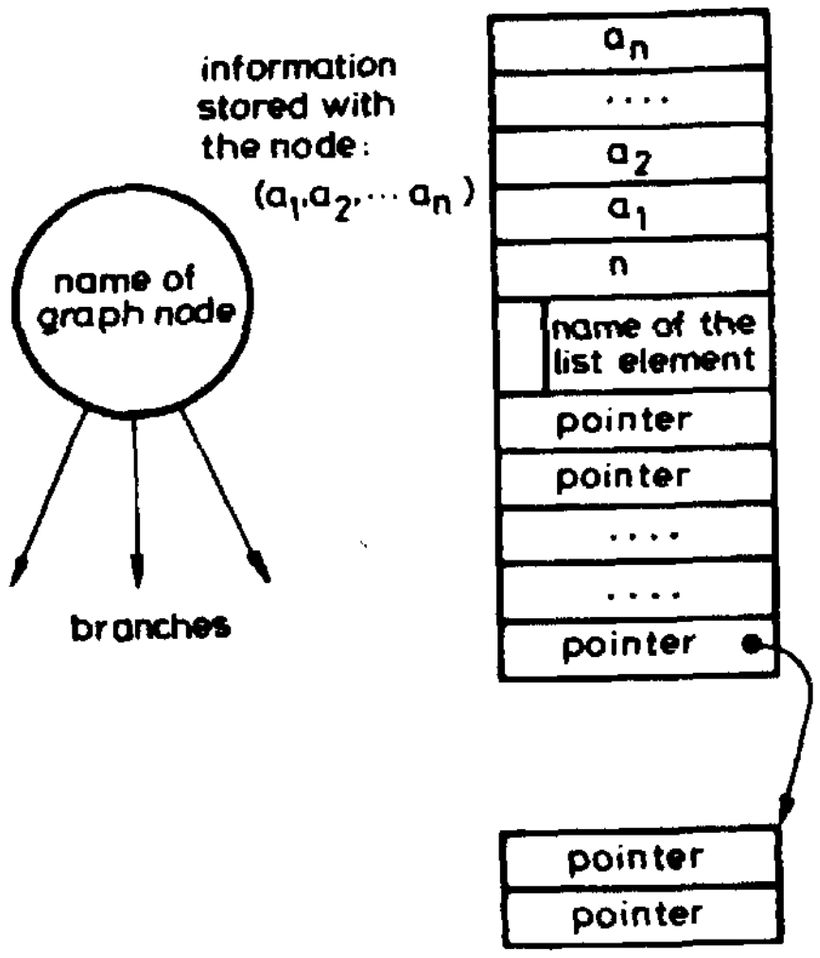
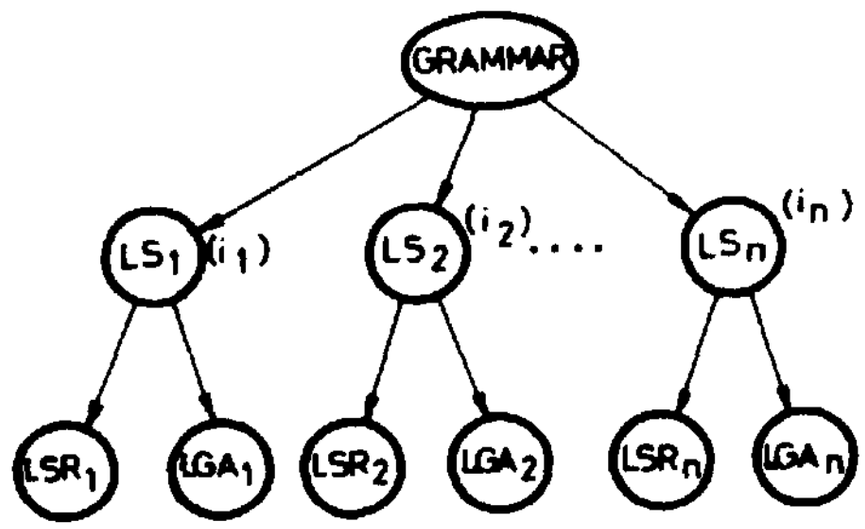


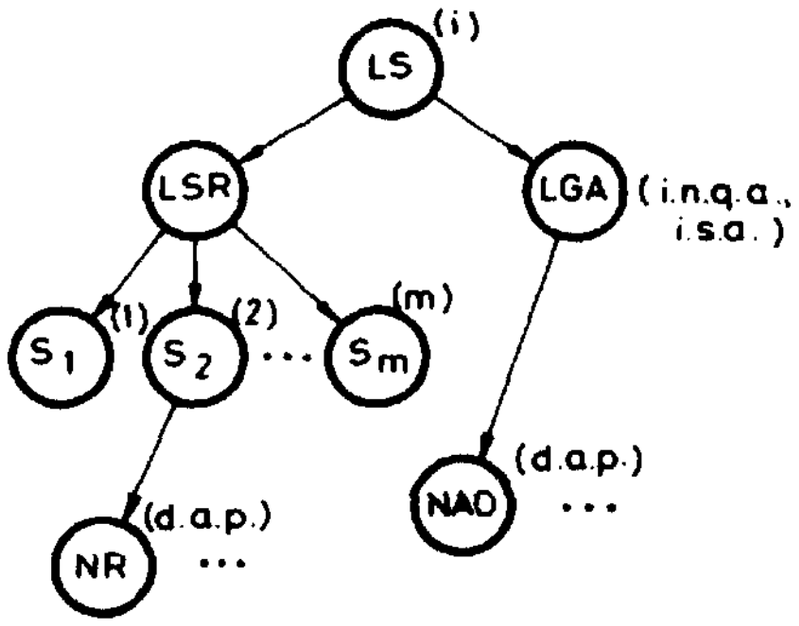
Fig.4

The organization of the grammatical analysis

In working with certain limited class of patterns, the grammar describing these patterns is an always used and unchanging information and is organized as a part of the permanent data structure. Fig.^a shows the structure of the grammar and Fig.5b details the structure of a rule. Appendix contains the detailed description of the construction of the grammar.



a/



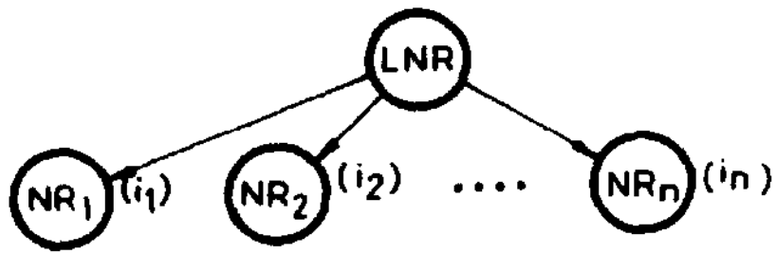
b/

LS - Left-side Symbol
i - number of the rule
LSR - List of right-side Symbols and Relations
LGA - List for Generation of the quantitative Attributes of the left-side symbol
S - Symbol
NR - Name of geometrical Relation
NAO - Name of Assignment Operation
d.a.p. - description of actual parameters
i.n.q.a. - indication of number of quantitative attributes
i.s.a. - indication of symmetrical quantitative attributes

Fig.5.

Geometrical relations and operations for assignment of value to quantitative attributes are interpreted as FORTRAN logical funtions and subroutines respectively. The grammatical rules are giving only the names of relations and assignment operations, further the description of actual parameters. The names of relations

and assignment operations are collected in two lists analogous with each other, belonging to the permanent data structure Fig.6 shows the organization of one of them. The so-called switching index is used for the activation of the respective program segment at the time of the running of the program.



LNR - List of Names of geometrical Relations
NR - Name of geometrical Relation
i - switching index

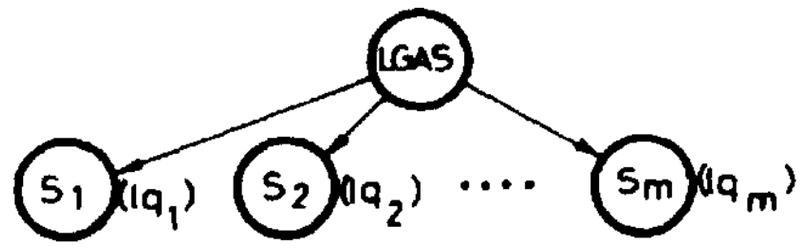
Fig.6

The analyzer - the program of linguistic analysis - consists of the following subroutines: subroutine of the analysis according to the given rule, subroutine of the synthesis according to the given rule, subroutines for the organization of the work of program segments checking the geometrical relations and realizing of assignment operations, strategy subroutine for the organization of the order of the analysis according to grammatical rules and a set of auxiliary subroutines.

The algorithm of analysis can be characterized as analysis by synthesis. The grammatical rules are numbered in an increasing hierarchical order, "bottom-up" (in the increasing complexity of pattern elements to be described). The algorithm consists of information processing concerning the actual object on basis of grammatical rules considered in an increasing order of their numbers,of trials to synthesize the left-side symbol, considering every possible combination of the already synthesized symbols (the syntactic type of which is identical with the types given on the right-side) and checking the fulfilment of the geometrical relations required by the given rule. In the course of cross-recursion between the rules, such a "linear" way of the algorithm is violated, "loops" appear, its general trend, however, does not change.

It must be noted that the greater part of the algorithm is made of the subroutines of the analysis according to a given grammatical rule on basis of which it is easy to organize any other strategy of the order of the uses of grammatical rules.

The quantitative attributes of the symbols synthesized during the working of the algorithm (i.e. of the pattern elements recognized by the program? ere written in the List of the Generated quantitative Attribute Sets (LGAS) branching into sublists, corresponding to syntactic types (Fig.7). These sublists are created and existing only for the running time of the linguistic analysis program.



LGAS - List of Generated quantitative Attribute Sets
S - Symbol (Syntactical type of picture element)
lq - List of quantitative attributes of elements with syntactical type S for actual input picture

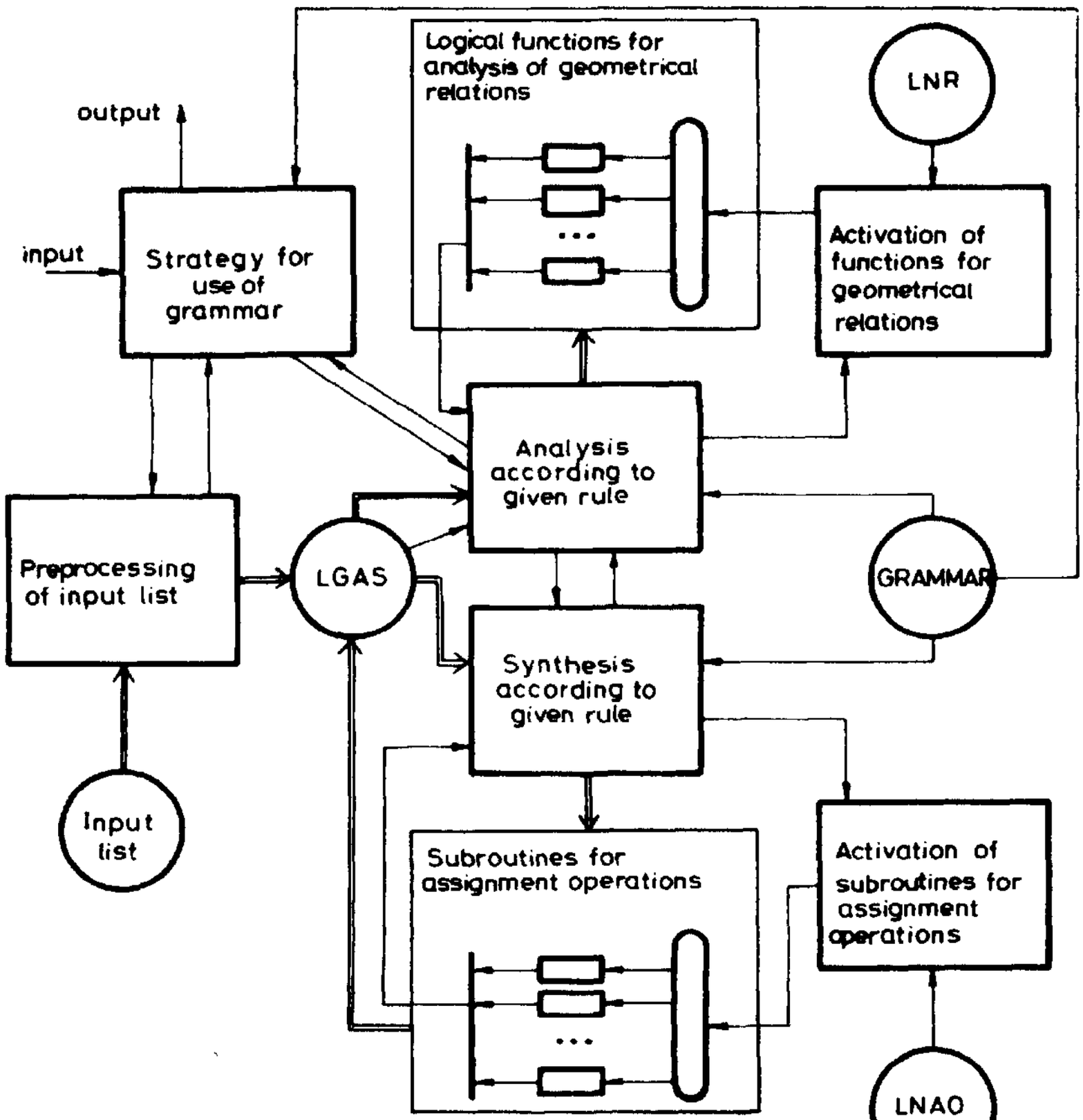
Fig.7

The run of the program begins vith the permanent data structure putting into the memory followed by the reading of the input list and its preprocessing, involving the location of the quantitative attributes of the primitives in the three basic sublists (line, arc, undefined) of the List of Generated quantitative Attribute Sets intended for information storage about the actual pattern.

Fig.8 shows how the organization of the running of the program is imagined. Here the lists are drawn up by circular blocks and the program parts by rectangular ones. The transmission of information about the actual object is shown by double directed lines and the transmission of information controlling the analysis is shown by simple directed lines. By running the program-analyzer no modification taXes place on the permanent data structure (Grammar, Mst of Names of geometrical Relations /LNR/ , L_ist of Names of Assignment Operations /LNAO/), therefore we call it permanent; the information stored in it is used to control the analysis of information about the actual object. During the analysis a lot of consecutive structural transformations are carried out on the List of Generated Attribute Sets.

There are two stages in the use of the program:

1/ Creation of a permanent data structure (Grammar, LNR, LNAO) and programming of geometrical relations and assignment operations. This part of the system can be changed and extended without limit during the definition of the set of tasks to be completed with the help of the program. Meanwhile the analyzer program is left unchanged, ine pei-



LGAS - List of Generated Attribute Sets
LNR - List of Names of geometrical Relations
LNAO - List of Names of Assignment Operations

Fig.8

manent data structure must be described in I.IDI-72 input language (in parenthetical form) on cards. A very short program organize:; the input of the permanent data structure in the memory of the computer with the help of the 1J0I-72 input program. During the tost ing of the data structure this shurt program calls at the end of its run the program-analyzer which perform:, the analysis; of an input list. After mirvi'y of the result:¯, of the carrying out of the analysis corrections can be made in tin- description of the data structure. The correct, data structure is transferred on a disc in the form of binary information.

2/ Use of the program for pattern recognition (Fig.8). For this purpose the data structure in directly transferred from the disc t.o the memory of the computer before the beginning of the run of the program-analyzer.

Appendix

Grammar

The rules of the grammar can be subdivided into two groups:

1/ HuJes where there are only terminal symbols (e.g. rule n .in Fig. 3) on the right ride;

I'/ Hu1es where there arc only non-terminal symbols on the right side[1] (e.g. rules 1-;_i in Fig.3).

The rules of the first group are programire[1]d as preprocessing °* input Jiat because this, is a common basis to all grammars interestittf. us. The rules of ttie

second group are contained in the Grammar which is a part of the permanent data structure for the actual set of tasks. Such a subdivision simplifies the algorithm to a great extent as it is dealing with non-terminal symbols only.

Numbering of grammatical rules: the rul<T. are numbered sueh a way that every Mymho] appears first. a:; a left. -side one. Kxcept.ions arc only recursion:; across the rules (which are also permitted in the Grammar). Rules with identical left-side symbols get different numbers and enter the grammar as separate rules.

The structure of a rule is shown on Fig.56.

Each symbol in the list of right-side symbols gets a number. The number!ng is completely arbitrary, but necessary for the unambipuous description of geometric relations between r.ymboJs, because the same rule may contain several symbols of identical syntactic type. For every symbol, there j;, a list, of names of relations, which the symbol is bound to satisfy with respect to the preceeding riphf-side symbols, of the rule. For every relation in the rule is pivon the description of actual parameters of the relation. The following order is accepted for the description of parameters: *every* purameter is described by a pair of integer!; the first of which is the order t.umbe r of the ri. tf hi. - s i do symbol which must, satisfy this relation, while the second is thie number of that quantitative attribute of this symbol, the value of which is the actual parameter of the relation.

The quantitative attributes of the symbols are identified with numbers {i.e. integers), at first because the values of the Quantitative attributes of the picture elements are stored in linear lists (at nodes whose names correspond to the syntactic type of these elements), (Fig.7) in a fix order: e.g. first A, second B, etc., thereafter because for the description of actual parameters in the Grammar there are also used linear lists in which only integer-type variables can be stored.

A simple example explains the details of the construction of the rule- Let us consider one of the rules of -grammar, presented in Fig.3. In the meta-i anguage of the description of grammars explained in the introduction, it looks as follows:

1/ (ARC2•|(ARC, ARC)($A_1=A_2$, $B_1\neq B_2$, $C_1=K_2$ $|\Psi_1-\Psi_2|>90^\circ$)($A=B_1$, $B=B_2$, $K=K_2$, $L=K_2$, $C=A_1$)|

In the LIDI-72 input language the names of the nodes of graph are given by an alphabetic string, and their attributes, i.e. elements of linear list by integers. The attributes are enclosed between the signs (= =) and are delimited by commas and comments.

Let be defined in the system the following geometrical relations:

notations in the meta-language: = , ≠
names: EQ, NEQ

and the following assignment operation:

notation in the meta-language: =
name: ASSIGN

All List of right-side Symbols and Relations have the name LSR, and all Lists for Generation of quantitative attributes have the name LGA.

Fig. 9 shows the structure of the rule for ARC2. The description of the rule in the input language LIDI will be as follows:

```
C ARC2(=1=)(LSR(ARC(=1=), ARC(=2=)
C (EQ(=1,1,2,1=),
C NEQ(=1,2,2,2=), NEQ(=1,4,2,4=),
C OBTANG(=1,3,2,3=))),
C LGA(=5,0=)
C (ASSIGN(=1,1,2,2,2,1,4,2,4,1,1=)))

C is a sign for continuation.
```

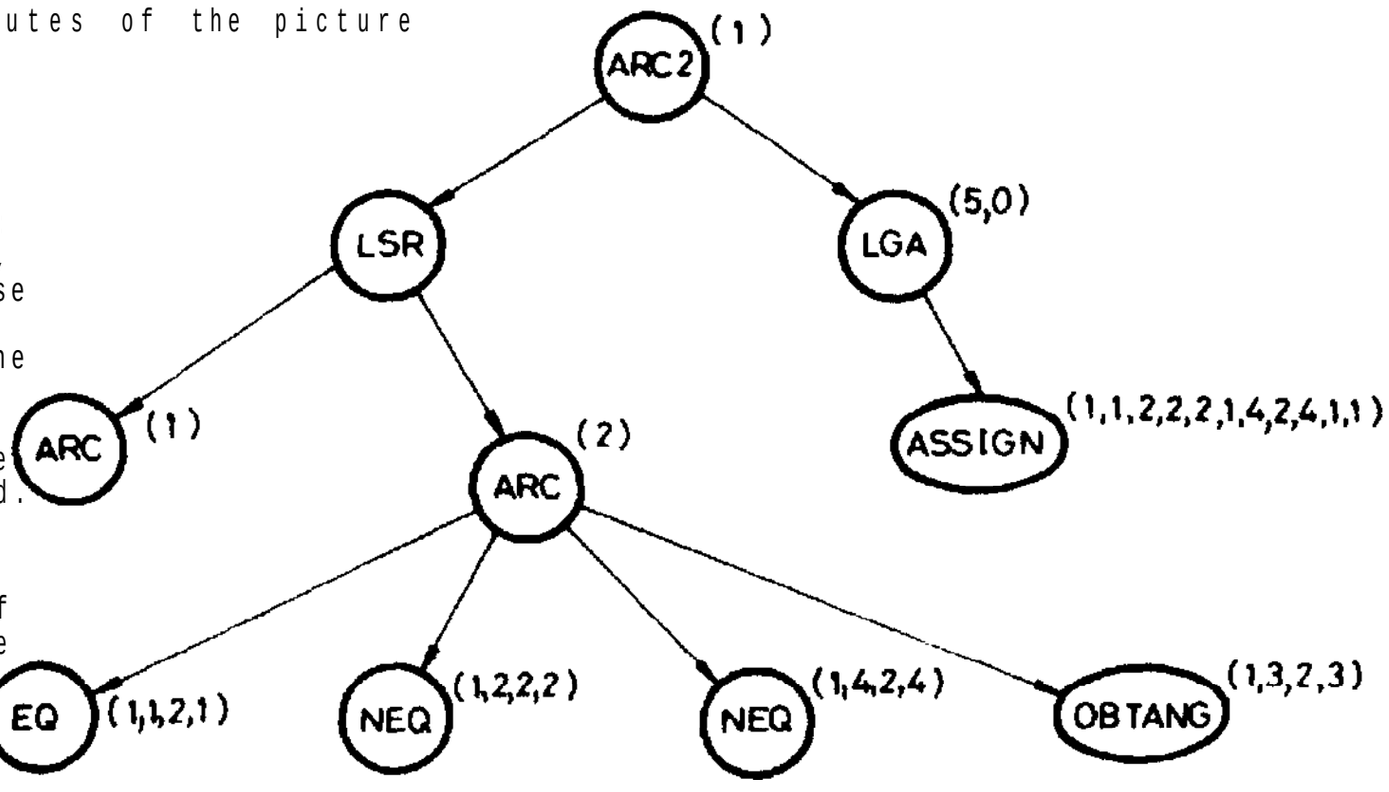


Fig.9

Beferences

[1] Vamos T., Vassy Z.: Industrial Pattern Recognition Experiment - A Sintax Aided Approach. First International Joint Conference on Pattern Recognition. Washington, Oct. 30 - Nov. 1. 1973. IEEE Publ. 73-CHO 821-9C pp.*hk*^~*k*52.

[2] Vamos T., Vassy 2.: Restricted Goal Sintax Aided Pattern Recognition Experiment. Acta Techn.Acad.Sci.H.Tom,/3-4/,1972.

[3] Evans, T.G.: A Grammar-controlled Pattern Analyzer. IFIP Congress 68, Appl.3.

[4] Evans, T.G.: Grammatical Inference Techniques in Pattern Analysis. Software Engineering, 1971.

[5] Fidrich I.: LIDI-72 MTA SzTAKI Tanulmanyok (Reports), 21/1974.

[6] Fidrich I., Uzsoky M.: LIDI-72, /A List-operating system on the DIgTTal Department - variation of the year 1972./ MTA SxTAKI Tanulmanyok (Reports), 16/1974.

[7] Vamos T., Vassy Z.: The Budapest Robot - Progmatic Intelligence IFAC World Congress, Boston-Cambridge August 2*-30, 1975.