Picture Graphs, Grammars, and Parsing

by

Alan C. Shaw

(to be presented at the International Conference
on Frontiers of Pattern Recognition,
Honolulu, January 18-20, 1971)

Technical Report
No. 71-89
January, 1971

Department of Computer Science
Cornell University
Ithaca, New York 14850

Abstract

Picture Graphs, Grammars, and Parsing*

Alan C. Shaw
Cornell University**
Ithaca, N.Y. 14850

This paper is concerned with the syntactic description and analysis of pictures when graphs are employed as the primary description formalism. The present state of development, a number of significant open problems, and the advantages and limitations of this approach are discussed under the following three headings:

(a) representation of pictures by graphs,

(b) graph languages and grammars, and

(c) parsing of graphs and pictures.

In (a) we investigate transformations from pictures to graphs based on n-ary relations ($n \geq 1$) that exist among picture components, both at the primitive pattern level and among higher level subpictures; n-ary relations are reduced when $n>2$ or expanded when $n=1$ to binary relations. Several grammatical schemes for generating graph descriptions are then evaluated with respect to their descriptive adequacy, complexity, and practical and theoretical tractability. Syntax-directed analysis of graphs and pictures is treated from two points of view - how to parse efficiently and how to enlist the descriptive mechanism as an aid in the difficult lower level pattern recognition tasks. The latter point is particularly emphasized with the aim of promoting a more systematic approach to contextual recognition.

---

* This work was supported in part by The National Science Foundation grant GJ-108.

** Department of Computer Science

Picture Graphs, Grammars, and Parsing*

Alan C. Shaw

I. Introduction

This paper examines the use of graph representations for the description and analysis of "structured" pictures. Examples of structured pictures are particle trajectories produced in high energy particle physics experiments, circuit schematics, flow charts, organic chemistry molecules, architectural drawings, some biomedical pictures, and natural scenes. A linear list of the features, primitives, or objects contained in one of these pictures is usually not an adequate description; it is the relationships among the components that often provide the key to understanding and that require extended processing beyond standard pattern classification.

It is convenient to define pictures independently of any specific viewing or generation machine (e.g. flying spot scanner or plotter). At the lowest and most primitive level, we can use an extension of Rosenfeld's picture function (Rosenfeld, 1969;pp.1-2):

A picture α is a vector-valued function $\underline{f}$ of several real variables $\underline{x}$; i.e.

$\underline{f}(\underline{x}) = (f_1(\underline{x}),\ldots,f_k(\underline{x}))$, $\underline{x} = (x_1,\ldots,x_n)$,

where k is the number of attributes of α, f_i represents the

*This research was supported in part by the National Science Foundation, Grant GJ-108.

value associated with the i^{th} attribute, n is the dimension of α, and $\underline{x}$ is a point in n-space. Typical attributes are intensity or "gray level", colour, and opaqueness or transparency. n = 2, 3 is the normal situation but higher dimensions or n = 1 need not be excluded.

Descriptions at higher levels serve several purposes. They define encodings and classifications of pictures that allow humans and machines to comprehend and conveniently process them; from a slightly different point of view, they permit a useful articulation of the interesting parts of pictures while eliminating "noisy" or irrelevant components. Descriptions may also be employed to drive picture analyzers or generators. In the analysis case, this implies the existence of a grammar or set of rules that define the picture class of interest. Picture analysis then becomes a problem of <u>deriving</u> descriptions while generation involves the <u>evaluation</u> of descriptions.

The possible levels of description can be arranged in a hierarchy, where level i+1 is derived from level i by some set of picture processing operations:

Level 0: $\underline{f}_0(\underline{x})$, the lowest level mathematical description as discussed above.

Level 1: $\underline{f}_1(\underline{x})$, a machine's view of $\underline{f}_0$. This is normally a quantization of $\underline{f}_0$ over a regular array.

Level 2: $\underline{f}_2(\underline{x})$, the result of "preprocessing" $\underline{f}_1$. Noise removal and image enhancement are typical preprocessing goals.

Level 3: $\{F_i\}$, a set of the features F_i of $\underline{f}_2$. Examples are line segments and edges.

Level 4: $\{P_i\}$, a classification of all the primitive objects P_i in the picture represented by Level 3.

Level 5: $\{R_i\}$, the relations R_i existing among the primitives.

Levels 6-m: higher level groupings and their relationships.

Level m+1: C, the highest level. This generally consists of a simple classification C.

The hierarchy is illustrated in Fig. 1. We are mainly interested in describing pictures at levels 4 to m+1 and employing these descriptions as an aid in transforming pictures from level i to i+1 or i+1 to i for i=0,1,...,m.

Most description schemes that have been implemented or proposed use a generative grammar of some form which defines a picture language; this approach has been termed "linguistic" because of the analogy to natural and programming language specification and analysis. Picture grammar and language notations have been based on arrays, list structures, (relation, object list) pairs, predicate calculus, set theory, and graphs (Miller and Shaw, 1968; Fu, 1970).

Graph representations have been particularly attractive because of the natural interpretation of graphs in terms of objects, relations, and concatenations, and because of the large body of graph theory and algorithms that is available. In the next section, we investigate the general problem of representing pictures by graphs. Section III examines the virtues and limitations of several graph languages and grammars that have been developed. The following section discusses algorithms for parsing graphs and pictures emphasizing problems of efficiency and the use of contextual information. Throughout the paper, we list a number of problems and directions for future research.

II. Representation of Pictures by Graphs

The composition of a picture α can often be specified hierarchically as a set S of subpictures, the relationships among the elements of S, and a number of attribute-value pairs for α and each subpicture. (The notation of Evans(1969) is complete in this sense). An n-ary relation R ($n \geq 1$) satisfied by the objects $X_1,\ldots,X_n$ will be denoted $R(X_1,\ldots,X_n)$; an attribute-value pair will be designated by $\langle a,v \rangle$.

Consider the subscripted variable A_N of Fig. 2, interpreted as a picture α. One possible description of α is:

$\alpha = \{A,N\}$ <u>where</u> SUBSCRIPT(A,N)

$A = \{DP_1,DP_2,H,DM_1,DM_2\}$ <u>where</u>

$CAT(DP_1,DP_2) \wedge CAT(DP_1,H) \wedge CAT(DP_2,DM_2)$

$\wedge\ CAT(DM_2,DM_1) \wedge CAT(H,DM_1)$

$N = \{V_1,DM,V_2\}$ <u>where</u> $CAT(V_1,DM) \wedge CAT(DM,V_2)$

Let each object above have an ordered pair of points $\langle \underline{t},\underline{h} \rangle$ by which it may be connected to other objects; then CAT(X,Y) means that $\underline{h}_X$ is concatenated to $\underline{t}_Y$. $\underline{t}$ and $\underline{h}$ will be referred to as the tail and head points respectively.

There are several ways to map the description to a graph. The most common and straightforward representation is a <u>labelled</u> <u>node</u>-oriented <u>directed</u> graph, where nodes represent objects and edges denote (binary) relations* (Fig. 3 (a)). The unlabelled edges describe the "contains" relation C; for example, $C(\alpha,A)$ means that the object α contains A. (It is this relation that may define the levels in a description hierarchy). Note that all edges and nodes are implicity or explicity labelled ("coloured") and that the root and interior nodes decompose into graphs at lower levels. An edge-oriented graph can also be used in many cases (Fig. 3 (b)). Nodes represent the tail and head points of concatenation, each object is denoted by a labelled directed edge pointing from its tail node to its head node, and all (binary) relations other than concatenation are treated as "invisible" objects and defined as edges. Here it is not necessary to label the nodes. While the latter type of graph is less general than the node-oriented form and leads to ambiguous interpretations in some instances - for example, if we had CAT(X,Y), CAT(W,Y), C(X,A), and C(W,B) - , it has the virtues of simplicity and processing convenience for those pictures where component concatenations are the primary relations (Shaw, 1969a, 1970).

The above example was contrived to involve only binary relations. However, picture descriptions frequently also require the expression of unary relations (properties) and n-ary relations for $n>2$. For example, it might be more useful to describe an "A" as:

---

*If a relation is symmetric, it can be designated by an undirected edge.

(1) $A = \{L_1,L_2,L_3,L_4,L_5\}$ where $TRIANGLE(L_2,L_3,L_4)$
$\wedge HORIZONTAL(L_3) \wedge ABOVE(L_3,L_2) \wedge \ldots$

where the ternary relation TRIANGLE is satisfied by the lines L_2, L_3, and L_4, and L_3 has the property relation HORIZONTAL. A graph-representable description is obtained by mapping all relations to binary ones. A unary relation $R(X)$ can be changed to the binary one $\tilde{R}(X,\Lambda)$ where Λ denotes the "null" object. $R(X_1,\ldots,X_n)$ $(n>2)$ can be transformed to a composition of binary relations, for example, $R_1(X_1,R_2(X_2,\ldots,R_{n-1}(X_{n-1},X_n)\ldots))$, or to a conjunction of binary relations $R_1(X_{11},X_{12}) \wedge R_2(X_{21},X_{22}) \wedge \ldots \wedge R_k(X_{k1},X_{k2})$, or to a combination of these. $TRIANGLE(L_2,L_3,L_4)$ could be transformed into either the following equivalent relations:

$CAT(L_2,L_3) \wedge CAT(L_3,L_4) \wedge CAT(L_4,L_2)$ or

$\Delta(L_3,CAT(L_2,L_4))$

where $\Delta(X,Y)$ means that the line X is connected to form a triangle with the object Y consisting of two concatenated segments. Replacement of an n-ary relation with binary ones using composition requires the introduction of more levels (more $C(X,Y)$ relations) in the description; description (1) could be mapped to:

(2) $A = \{L_1,L_3,ANGLE,L_5\}$ where $\Delta(L_3,ANGLE) \wedge HORIZ(L_3,\Lambda)$
$\wedge ABOV(L_3,ANGLE) \wedge \ldots$

$ANGLE = \{L_2,L_4\}$ where $CAT(L_2,L_4) \wedge \ldots$

The node-oriented graph of (2) is given if Fig. 4. To complete the graph descriptions, <a,v> pairs may be associated with each object or higher level structure and included as part of the node or edge label. For example, "A" might have the attributes <HEIGHT,0.75">,<WIDTH,0.5">. In summary, we can completely describe any picture as a labelled directed graph by mapping all relations to binary ones and labelling the nodes and edges with relations, object names, and <a,v> pairs.

Two disadvantages of this approach are:

1. it is sometimes awkward to describe pictures by binary relations only, and
2. the graph representation can be unnecessarily complex when compared with other description methods.

The examples given by Minsky (1961) are a good illustration of the second point. There are 8 closed curves $X_0, \ldots, X_7$ which are related according to the following description:

(3) $\Theta(X_0, \rightarrow(\Theta(X_1, \downarrow(X_2, X_3)), \Theta(X_4, \nabla(X_5, X_6, X_7))))$

where $\Theta(X,Y)$ means Y is inside of X,

$\rightarrow(X,Y)$ means that Y is to the right of X,

$\downarrow(X,Y)$ means that Y is below X, and

$\nabla(X,Y,Z)$ means that Y is to the right of X and Z is underneath and between them.

The corresponding graph is awkward and messy compared with (3). We have also assumed that descriptions can always be arranged in a "tree" hierarchy. This is not always possible as the following example demonstrates:

Let a picture consist of 4 objects A,B,C,D with the relations $R_1(A,B)$, $R_2(C,D)$, $R_3(A,C)$, $R_4(B,D)$, $R_5(W,X)$, and $R_6(Y,Z)$, where $W = \{A,B\}$, $X = \{C,D\}$, $Y = \{A,C\}$, and $Z = \{B,D\}$. Fig. 5 portrays the relations.

The characterization of pictures by graphs seems natural and useful when the subpictures within a level in a description hierarchy are significantly related to one another and when there do not exist many interlevel relations. In this case, graph manipulation algorithms, and the model and results of graph theory can be (and have been) applied successfully in several theoretical studies and experiments in picture processing. Section III and IV mention some of these applications. It is, however, still not generally clear under what circumstances graph representations are useful, nor is it evident - despite the examples in this section - how to generally transform a picture to a graph.

III Graph Languages and Grammars

Both a set of pictures and a set of graphs may often be described by a grammar G generating a graph language L(G); each sentence $x \in L(G)$ specifies, up to isomorphism, a unique graph g_x, which in turn, is an abstract representation of a set P_{g_x} of one or more pictures. The purpose of such a grammar are:

1. to precisely define, by finite means, an infinite set of graphs (and pictures),
2. to impose a hierarchic structure on each $x \in L(G)$, g_x, and member of P_{g_x} (i.e. the C(X,Y) relation), and

3. to assist in the recognition, analysis, and interpretation of graphs and pictures through syntax-directed processing.

Several different notations and formalisms for graph languages and grammars have been developed and applied in experimental settings. Figures 6, 7, and 8 contain simple examples of grammars for the most prominent of these. The notation of Narasimhan (1966), formalized by Feder (1969) as "plex" grammars, describes the connectivity of picture components by using explicit lists of labelled concatenation points in each rule. Each of Feder's grammar rules, in the context-free case, is of the form:

$$A\ \Delta_A \rightarrow x\ \psi_x\ \Delta_{xA}$$

where x is an ordered list of symbols identifying primitive objects or higher level subpictures, ψ_x is a list of "joints" or connections among points of elements of x, Δ_{xA} establishes a correspondence between points of x and attachment point labels to be associated with A, and Δ_A specifies a list of attachment points for A. While the sentences generated by these grammars are not directly graphs, they can be so transformed by either assigning labelled nodes to both objects and concatenation points as suggested by Pfaltz and Rosenfeld (1969) or by mapping picture objects to nodes and concatenations to labelled edges (Fig. 6(b)). The cited references apply this notation to the description of English characters, flow charts, chemical diagrams, particle trajectories in bubble chambers, and electrical circuits.

The PDL notation (Shaw, 1969a; Miller and Shaw, 1968) (Fig. 7) is also based on concatenations but uses a set of binary operators, much like the CAT(X,Y) of the last section to specify connections; the original notation restricted each primitive object to two concatenation points but this restriction was eliminated in further work (Shaw, 1969b). In the figure, the network component α could be, for example, an electrical circuit element such as a resistor; the operator + indicates head to tail concatenation in a PDL description, * defines both a tail to tail and a head to head connection, and higher level structures described by ($S_1 \theta S_2$) have tails and heads defined recursively as tail(S_1) and head(S_2) respectively. A complete set of operators for local tail/head concatenations, tail/head reversal, and "rewriting" over the graph is specified in PDL; the notation allows the description of any directed edge-labelled graph. PDL has been applied to roughly the same sets of pictures as the plex grammars (but not to chemical diagrams yet) as well as pages of text and spark chambers photographs.

The web grammars of Pfaltz and Rosenfeld (1969) (Fig. 8) explicitly employ node labelled graphs ("webs") in the rewriting rules; each production describes the rewriting of a graph α into another graph β and also contains an "embedding" rule E which specifies the connections of β to its surrounding graph (host web) when α is rewritten. In the above reference, web grammars were presented for directed trees, directed two-terminal series-parallel networks (TTPSN's), directed triangles of graphs, and "Pascal's triangles"; later work has contained grammars for various classes of planar graphs and simulated neural nets (Montanari, 1969; Pfaltz, 1970). The formalism of web grammars has been mathematically analyzed to a greater extent than any of the others mentioned above and seems theoretically more pleasing. All of the notations appear to be formally equivalent but this fact has not been proven; it is not difficult, however, to map PDL grammars into plex grammars and plex grammars into web grammars.

The main limitations of Feder's description scheme are the awkwardness of the connection point lists and the difficulty of specifying relations other than concatenation. The virtue of the PDL scheme is its simplicity but it too suffers from several defects. The notation is edge-oriented and the discussion of this type of representation given in the last section is applicable. Nodes can also be labelled in PDL but not in a very clean manner. Relations are described through the use of "blank" primitives which is adequate, at least, for simple relations and graphs but can be ambiguous in more general situations. The PDL scheme, as defined, used a context-free grammar form which could generate non-graphs; this problem can probably be resolved by allowing a straightforward form of context-sensitive rule. Both of the above notations are linear strings and it may be argued that a multi-dimensional notation would be more natural. Finally, the authors of these schemes have concentrated primarily on picture description and analysis, as opposed to a deeper theoretical study of their formalisms; more work is required in this latter area.

Web grammars are the most recent of the notations. As specified, they are node-oriented with no labels on edges which means that only one relation can be specified; however, it appears as if the rules and embeddings could be extended to include labelled edges. The embedding part of each grammar rule has been given in a mixture of set theory and English; a formal language for describing these embeddings would be useful. At this point, it is not clear how convenient web grammars will be for describing pictures; some picture and graph classes seem to require grammars that are not as natural as, for example, PDL grammars. The last two points are illustrated by the examples in Figures 7 and 8.

None of the schemes provide facilities for expressing relations, other than the C(X,Y), among higher level subpictures. Attributes are not included in any general way but these could be obtained by associating an interpretation or "semantic" rule with each grammar rule (e.g. Knuth, 1968). The next section examines graph languages and grammars in the context of picture analysis.

IV Parsing of Graphs and Pictures

The analysis of a set of pictures or graphs described by a grammar G can be formulated as a parsing and graph-matching process. The parse of a graph g occurs by attempting to find a sentence $x \in L(G)$ such that g_x is isomorphic to g; a picture α is parsed by looking for an $x \in L(G)$ such that

$\alpha \in P_{g_x}$. On a successful parse, the resulting $x \in L(G)$ and the sequence of grammar rules that derive x comprise a syntactic description and classification of the input graph and/or picture. The rationale for this particular model is that it offers a systematic method for picture analysis that is amenable to mechanization.

The analysis of sentences in a linear string language can be inefficient (even in the context-free case) when it cannot be determined whether a substring u which matches the right part of some rule can be reduced until substrings immediately surrounding u are first examined. The same problem exists in the graph case and is further complicated by the multi-dimensional nature of the objects described; in addition, a combinatorial explosion can occur when determining graph isomorphisms (the entire $n!$ different matchings of two n node graphs must be tried in the worst case). Clearly, one is interested in classifying sets of graphs and pictures in terms of their underlying grammars and the complexity of the "machines" required to parse or "accept" them.

A second point concerning parsing techniques is the issue of sequential versus parallel processing. There is evidence of both types in animals. A reasonable conclusion from past experiments in picture processing in machines and animals is that recognition of all primitive objects in pictures and processing within individual primitives can be done in parallel but establishing relations among primitives and building higher level structures can best be accomplished in a primarily sequential manner. The main advantage of parallel operations is, of course, speed*; one of the chief disadvantages, even at the primitive level, is that it is difficult to use contextual information to assist in recognition. A parsing approach that combines some of the best points of both types of processing is to analyze serially but attempt several different parses (paths through the grammar) in parallel. Pfaltz (1970) argues for extending the parallel approach as far as possible through a parse and relying on sequential analysis only when the former "fails". Future experiences on parallel machines, hopefully, will resolve this controversy.

A final consideration relates to the question of "who does the dirty work"; i.e. is the classification of picture primitives, the "lexical" analysis, to be done within the parsing system or as a preprocessing phrase? The main argument for performing this recognition within the parsing is that the grammar generally contains a wealth of contextual information that can potentially be employed to simplify this task; the past and current rationale for doing this externally is that parallel operations or different grammars or <u>ad hoc</u> procedures can often be used to perform the recognition more efficiently.

The author (1970) has specified a general parsing algorithm for pictures and graphs based on PDL descriptions, built an experimental system for a subset of PDL, and applied the latter to the analysis of digitized spark chamber photographs. The algorithm is analogous to a goal-oriented top-

---

*This assumes that a parallel machine is available.

down linear string analyzer where a tabular or list representation of the grammar directs the parse; instead of a single one-dimensional string pointer, we employ two graph or picture pointers-one for the tail node and one for the head node. One of the aims of this work is to simplify the primitive pattern classification tasks. During a parse, the partially completed analysis in conjunction with the concatenation operators in the grammar rules tell the system where to look and what to look for; the primitive pattern recognition part of the system is directed to look for a particular primitive satisfying given tail/head constraints. Analysis was successful and efficient (despite the backtracking) in the spark chamber application but no systematic study of other non-trivial classes of pictures or graphs has been made.

Feder (1969) has applied plex grammars in an experimental system and analyzed pictures from classes of "houses," leaf vein patterns, and bubble chamber trajectories. In his system, a plex language representation of a picture α is first obtained by analyzing a chain-encoding (Freeman, 1961) of α in terms of a given set of chain pattern languages. The next stage then parses the resulting plex according to a given plex grammar. (The plex grammar notation was put in tabular form and extended to include both parameters associated with terminal symbols and relationship subroutines). A top-down sequential scheme was used here also. Feder's experiments with the bubble chamber photographs are impressive and further demonstrate the potential benefits of syntax-directed picture processing; it would be interesting to see whether some of the recognition errors could be reduced if the primitive classification were done within the plex grammar parse instead of during a preceding analysis stage.

The web grammar group has recently published the results of two experiments in picture and graph parsing (Pfaltz, 1970). One dealt with graphs of two terminal series parallel networks (TTSPN's) and the other analyzed simulated pictures of neural networks. In each case, a specific parsing algorithm was devised based on the web grammar of the picture class. The parsing scheme for TTSPN's is bottom-up and takes advantage of the properties of that particular class of graphs; to determine whether a node n is part of a reducible subweb, it is only necessary to examine nodes in the immediate vicinity of n. The neural network analysis was also bottom-up. Perhaps the most interesting part of this experiment was the use of parallel recognition for both primitives and (whenever possible) the higher level syntactic components. While recognition of each component of a right part of a grammar rule can not be done generally in parallel - Pfaltz illustrates this point even in the string case (see also Rosenfeld, 1971) - , there nevertheless still exists much parallelism that can be exploited.

The above works suggest several important problems that are unsolved. Of most theoretical and practical significance is the classification problem mentioned at the end of the second paragraph of this section. Such a systematic character-

ization of graphs and pictures should permit one to answer related questions such as:
What is the context "surrounding" a particular subgraph or subpicture α that must be examined before it is known that α can be reduced as a unit during a parse?,
What parts of the parse for a given grammar and/or language can be done in parallel?, and
What are the machine time and space requirements to analyze a given set of pictures or graphs?
Some generalization of the string language notions of "bounded-context" or "LR(k)" (e.g. Feldman and Gries, 1968), would be useful. It is perhaps significant to note that none of this work employs any probability theory or statistics, probably because of the difficulties in formulating and using "stochastic" grammars (Fu, 1970).

V Conclusions

The development of systematic methods for describing and analyzing structured pictures is still at an early stage. The work on picture graphs, grammars, and parsing has demonstrated that many interesting and non-trivial classes of pictures can be naturally described in terms of graph languages, and that syntax-directed analysis of pictures and graphs is feasible and promising. The purpose of this paper has been to examine past and current work in the field and to suggest problems for future work.

References

1. Evans, T.G. (1969). A grammar-controlled pattern analyzer. In Information Processing 68, Proc. IFIP Congr. 1968, Morell, A.J.H.(Ed.), Vol. 2, pp. 1592-1598 (North-Holland, Amsterdam, 1969)

2. Feder, J. (1969). Linguistic specification and analysis of classes of line patterns. Technical report no. 403-2, Dept. of Electrical Engineering, New York University (April).

3. Feldman, J. and Gries, D. (1968). Translator writing systems. Comm. ACM 11, 2 (Feb.), 77-113.

4. Freeman, H. (1961). On the encoding of arbitrary geometric configurations. IRE Trans. Electron. Comp., EC-10, 2 (June), 260-268.

5. Fu, K.S. and Swain, P.H. (1970). On syntactic pattern recognition. in Software Engineering, Academic Press, N.Y. (to be published).

6. Knuth, D.E. (1968). Semantics of context-free languages. Math. Systems Theory 2, 2 (June), 127-145.

7. Miller, W.F. and Shaw, A.C. (1968). A picture calculus Proc. Conf. on Emerging Concepts in Computer Graphics, University of Illinois. W.A. Benjamin Press, New York.

8. ________________ (1968). Linguistic methods in picture processing-a survey. Proc. AFIPS 1968 Fall Joint Computer Conference, Vol. 33, pp. 279-290 (Thompson Books, Washington, D.C.).

9. Minsky, M. (1961). Steps Forward artifical intelligence. Proceedings of the IRE 49, 1 (Jan.) 8-30.

10. Montanari, U. (1969). Separable graphs, planar graphs, and web grammars. Inform. Contr. (May), 243-268.

11. Narasimhan, R. (1966). Syntax-directed interpretation of classes of pictures. Comm. ACM 9, 3 (March), 166-173.

12. Pfaltz, J.L. and Rosenfeld, A. (1969). Web grammars. Proc. Joint International Conference on Artificial Intelligence, Washington, D.C. Tech. Report 69-84, Computer Science Center, University of Maryland (Jan.).

13. Pfaltz, J.L. (1970). Web grammars and picture description. Tech. Report 70-138, Computer Science Center, University of Maryland (Sept.).

14. Rosenfeld,A. (1969). Picture Processing by Computer. Academic Press, N.Y.

15. Rosenfeld,A. (1971). Isotonic grammars, parallel grammars, and picture grammars. in Machine Intelligence VI, Elsevier Press.

16. Shaw, A.C. (1969a.) A formal picture description scheme as a basis for picture processing systems. Inform. Contr. 14, 1 (Jan.), 9-52.

17. __________ (1969b.) On the interactive generation and interpretation of artificial pictures. SLAC - PUB - 664, Stanford Linear Accelerator Center, Stanford (Sept.). Presented at the 1969 ACM/SIAM/IEEE Conf. on Math. and Computer Aids to Design, Anaheim.

18. __________(1970). Parsing of graph-representable pictures. J. ACM 17, 3 (July), 453-481.

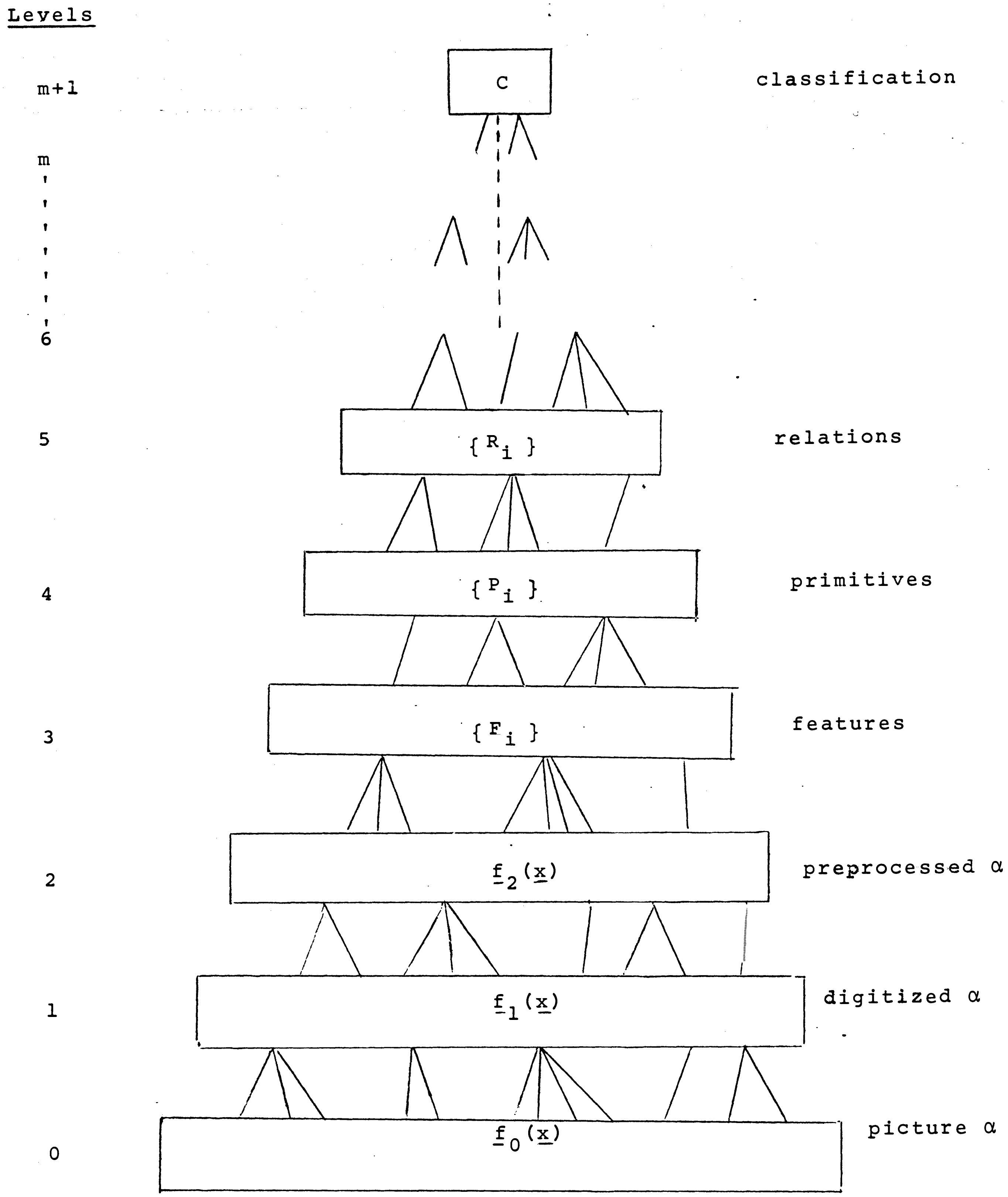
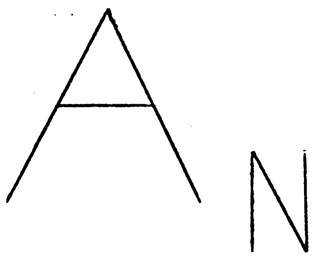
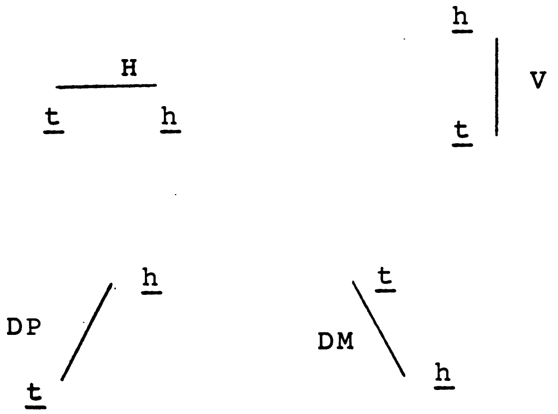


FIGURE 1

Levels of Picture Description



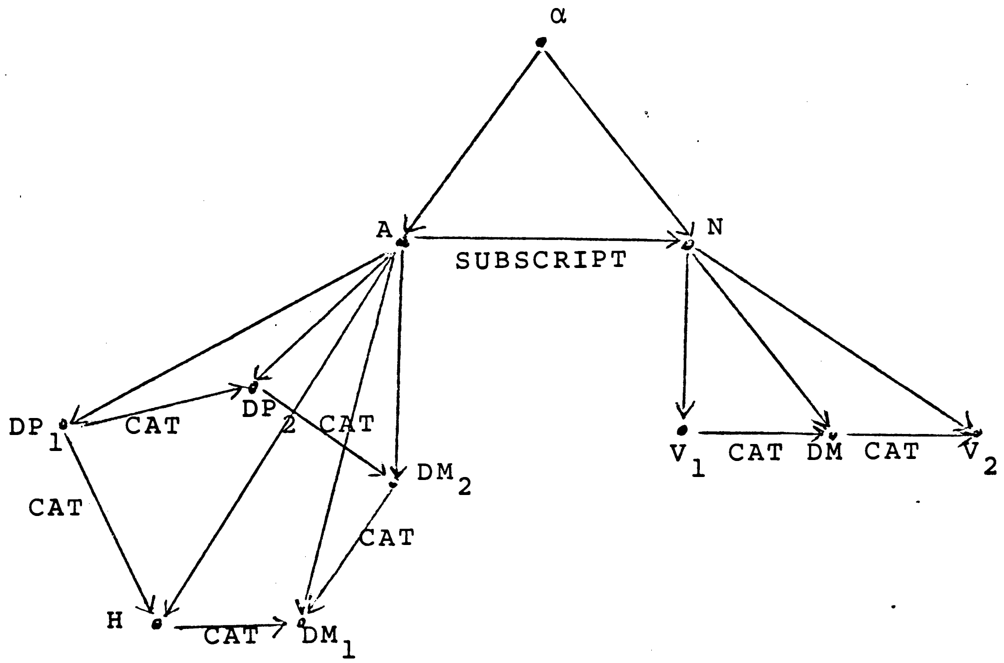
(a) Picture of a Subscripted Variable



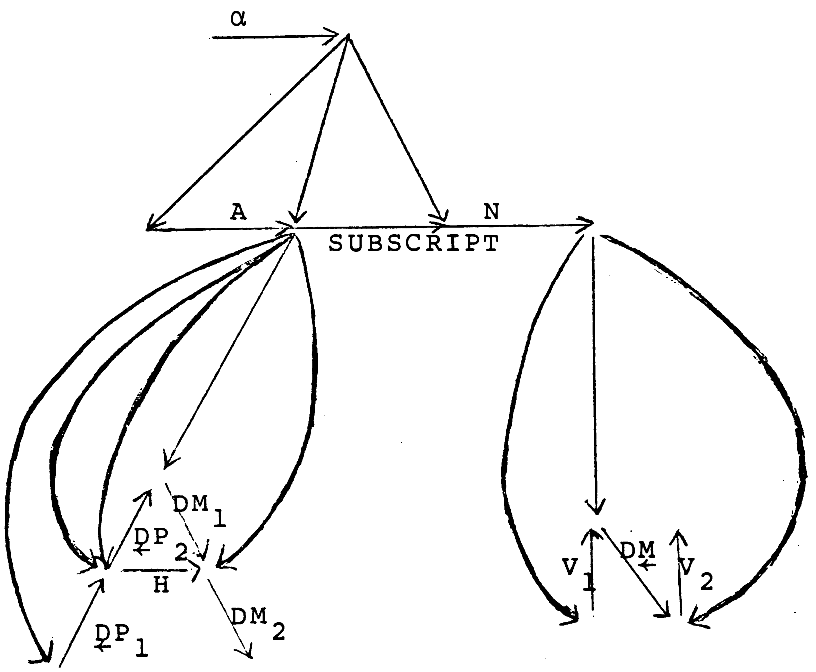
(b) Primitive Objects

<u>FIGURE 2</u>

Simple Picture Example



(a) Node-Oriented Graph



(b) Edge-Oriented Graph

FIGURE 3

Graph Representations of Figure 2(a)

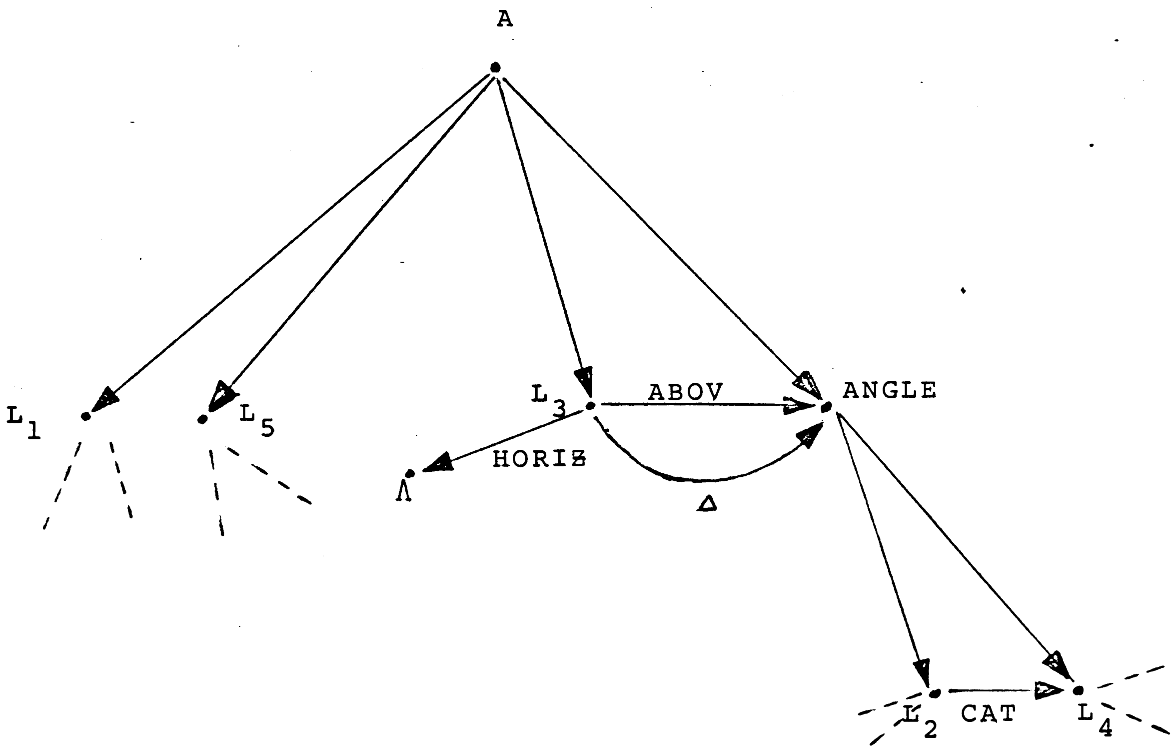


FIGURE 4

n-ary Relations as Binary Relations

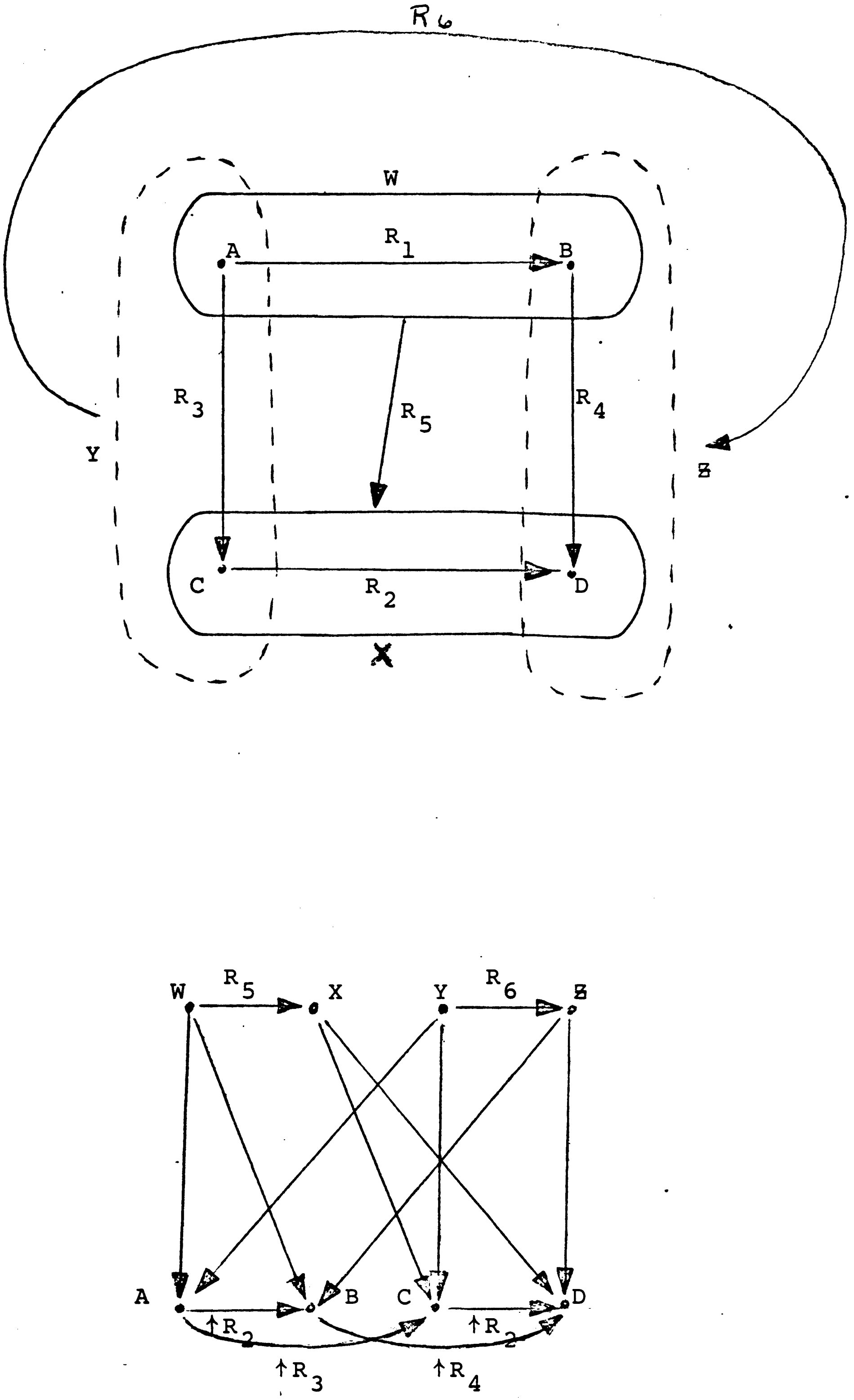
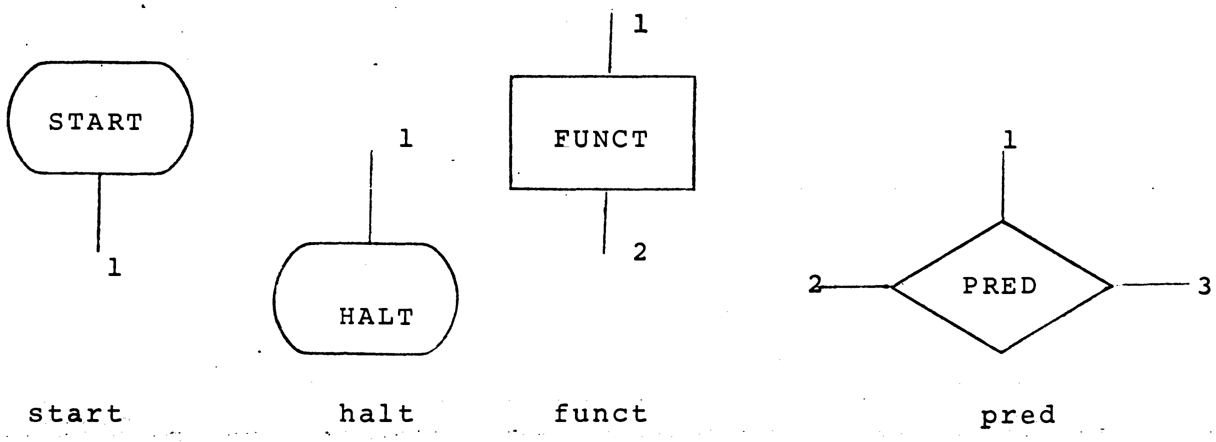


FIGURE 5

Example Where a Tree Hierarchy is Not Possible.



1. <PROG> → start <P> halt (110,021)

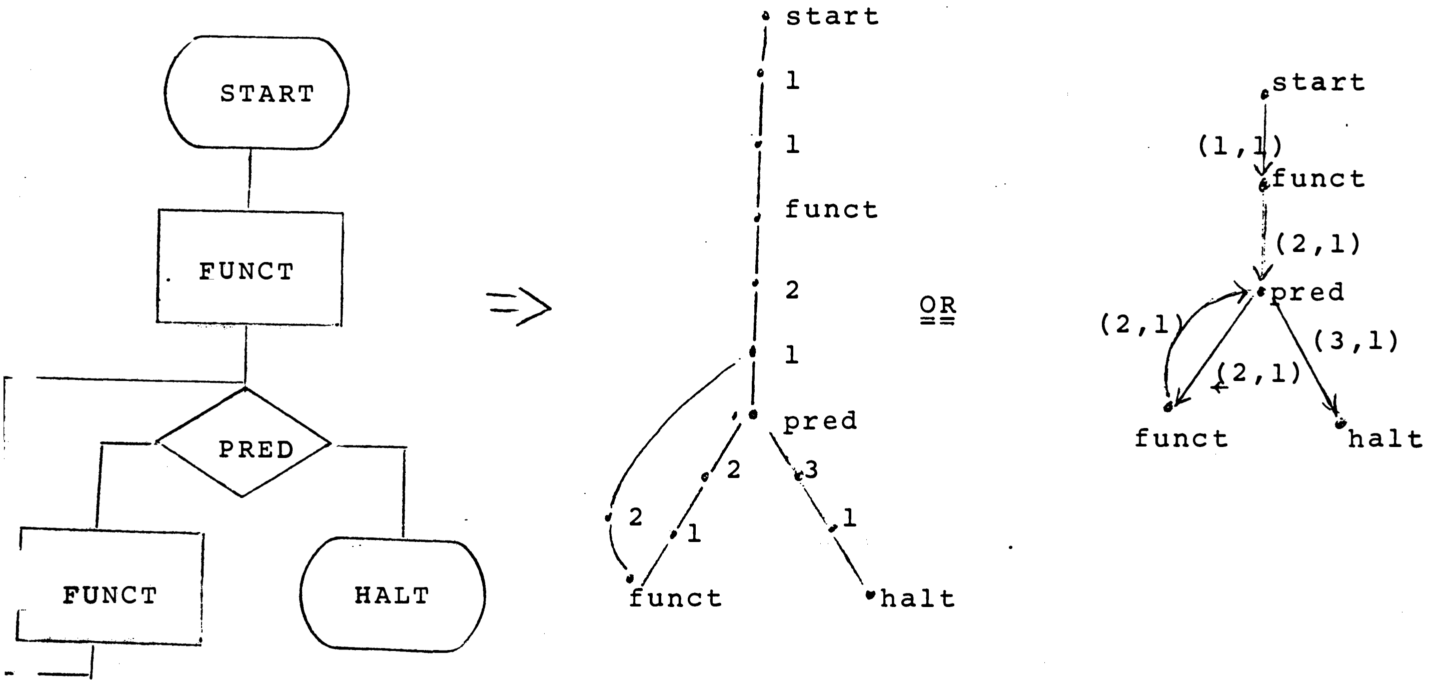
2. <P> (1,2) → funct () (1,2) | funct <P> (21) (10,02) |

pred <P> (21,12) (12,30) |

pred <P> <P> (210,301,022) (100,022)

Grammar rules

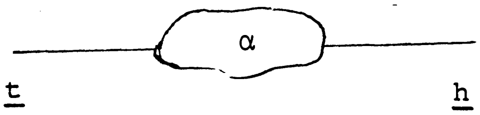
(a) Flow Charts with Simple Nested Loops



(b) Mapping Plexes to Graphs

FIGURE 6

Plex Grammars (Feder, 1969) and Graphs

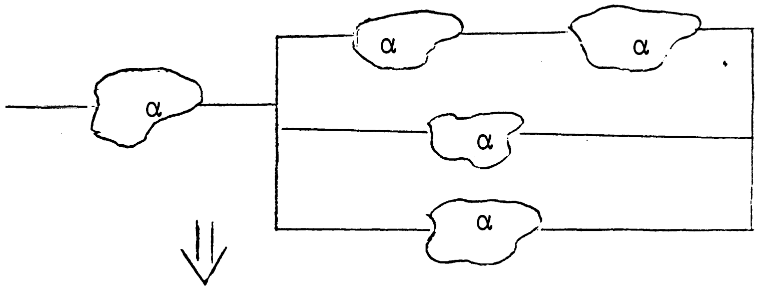


Primitive Object a

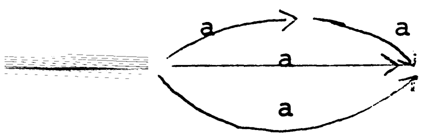
S → a | (S * S) | (S + S)

Grammar

(a) Grammar for Series-Parallel Networks



(a + (((a + a) * a) * a)) PDL description



graph

(b) Mapping Pictures to PDL to graphs

FIGURE 7

PDL Grammars (Shaw, 1969a)

Initial Webs $\left\{ \overset{t_1}{\cdot} \longrightarrow \overset{t_2}{\cdot} \;,\; \overset{t_1}{\cdot} \longrightarrow \overset{A}{\cdot} \longrightarrow \overset{t_2}{\cdot} \right\}$

Grammar Rules:

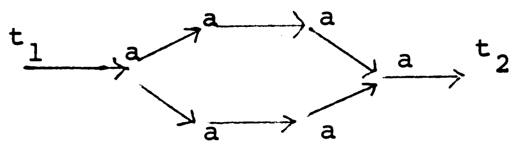
(1) $\underset{A}{\cdot} \;:=\; \underset{A_1}{\cdot} \longrightarrow \underset{A_2}{\cdot}$; $E = \{(p,A_1) \mid (p,A) \text{ an edge in the host web}\} \cup \{(A_2,q) \mid (A,q) \text{ an edge in the host web}\}$

(2) $\underset{A}{\cdot} \;:=\; \begin{matrix} .A \\ .A \end{matrix}$ provided there exists unique p and q such that (p,A) and (A,q) in the host web;

$E = \{(p,A) \mid (p,A) \text{ an edge in the host web}\} \cup \{(A,q) \mid (A,q) \text{ an edge in the host web}\}$

(3) $\underset{A}{\cdot} \;:=\; \underset{a}{\cdot}$; E same as in (2).

(a) Web Grammar for TTPSN's (context-sensitive)



(b) Example of a TTPSN

FIGURE 8

Web Grammars (Pfaltz and Rosenfeld, 1969)