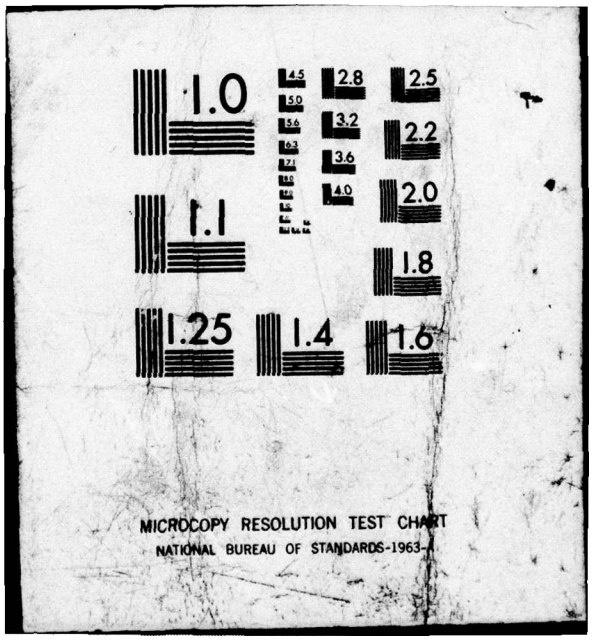
AD-A034 061 PURDUE UNIV LAFAYETTE IND SCHOOL OF ELECTRICAL ENGI--ETC F/G 12/2
STRUCTURE-PRESERVED ERROR-CORRECTING TREE AUTOMATA FOR SYNTACTI--ETC(U)
1976 S Y LU, K S FU AF-AFOSR-2661-74

UNCLASSIFIED AFOSR-TR-76-1311 NL

| OF |
AD
A034 061

END

DATE
FILMED
2-77


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

1976 IEEE Conference on Decision and Control, Dec. 1-3, 1976, Clearwater, Fla.

STRUCTURE-PRESERVED ERROR-CORRECTING TREE AUTOMATA FOR SYNTACTIC PATTERN RECOGNITION†

S. Y. Lu
Purdue University
School of Electrical Engineering
W. Lafayette, Indiana 47907

K. S. Fu
Purdue University
School of Electrical Engineering
W. Lafayette, Indiana 47907

Abstract

An error-correcting syntax analyzer for tree languages with substitution errors, called structure-preserved error-correcting tree automaton (ECTA), is studied. Substitution errors are defined in terms of transformations which can easily be accommodated to linguistic notion. Let L be a tree language, for a tree β not in L, the essence of ECTA is to search for a tree α in L such that the cost sequence of error transformations needed to transform α to β is the minimum among all the sentences in L. A LANDSAT data interpretation problem is used as an example to illustrate the operation of ECTA.

1. Introduction

In order to tackle with the uncertainty that usually exists in the process under study, error-correcting parsing techniques have recently been applied to the areas of compiling, computer communication, and syntactic pattern recognition [2, 13, 21, 22]. The most widely used error-correcting parsing scheme is formulated to include substitution, insertion, and deletion errors for string-to-string correction [26]. The basic approach is to define these three types of syntax errors in terms of error transformations. The original grammar is then expanded such that error transformations on each terminal symbol have their corresponding terminal error productions [1]. During the error-correcting process, a decision criterion is required when the parsing procedure faces multiple choices of the next move. Two decision criteria have been proposed: the minimum-distance criterion, where distance is measured in terms of the number of error transformations used in a deterministic model [1, 19, 21], and the maximum-likelihood criterion in a probabilistic model [13, 20, 22].

In applying syntactic methods to pattern recognition, one-dimensional (string) grammars are sometimes inefficient in describing two- or three-dimensional patterns. For the purpose of effectively describing high-dimensional patterns, high-dimensional grammars such as web grammars, graph grammars, and tree grammars have been proposed. In practical applications, tree grammars and tree automata have been used in the classification of fingerprint patterns [17], the analysis of bubble chamber events [3], and the interpretation of LANDSAT data [14].

Generalized finite automata, called tree automata, which accept finite trees of symbols as its input, have been studied by several authors [4, 6, 23 24]. Brainerd [4] proves that the class of systems which generate exactly the sets of trees accepted by the automata is a regular system. Fu and Bhargava [12] first introduced the application of tree systems into pattern recognition. Later, as studied by Brayer and Fu [5], tree languages are actually a subset of graph languages corresponding to the type 3 in the string languages case.

The descriptive power of tree languages and the efficient analytical capability of tree automata make the tree system approach to pattern recognition very attractive. This paper is concerned with the error-correcting version of tree automata. Unlike the strings case, where the only relation between symbols is left-right concatenation, a tree structure would be deformed under deletion or insertion errors. The structure-preserved error-correcting tree automaton (ECTA) proposed takes only substitution errors into consideration. By introducing a blank element, a deletion error can be treated as substitution of a non-blank element by a blank element, and an insertion error becomes a non-blank element in substitution for a blank element.

Both minimum-distance and maximum-likelihood error-correcting tree automata are formulated in this paper. An example of using ECTA in LANDSAT data interpretation is also presented.

2. Structure-Preserved Error-Correcting Tree Automaton (ECTA)

Let D be a tree domain, $D \subset U$, Σ be a set of terminal symbols, we define $T_\Sigma^D = \{\alpha | \alpha \in T_\Sigma, D_\alpha = D\}$ be the set of trees in the tree domain $D^{\dagger\dagger}$. In this section, substitution error is described in terms of the transformation η: $T_\Sigma^D \to T_\Sigma^D$.

†This work was supported by the AFOSR Grant 74-2661.

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFSC)
NOTICE OF TRANSMITTAL TO DDC
This technical report has been reviewed and is approved for public release IAW AFR 190-12 (7b).
Distribution is unlimited.
A. D. BLOSE
Technical Information Officer

For $a \in D$, $x \in \Sigma$ and $\alpha, \alpha' \in T_\Sigma^D$, we write $\alpha \mid\!\!\overset{\eta_{a/x}}{}\!\!\alpha'$ if α' is the result of replacing the label on node a of tree α by terminal symbol x. Furthermore, η^k denotes the composition of η with itself k times.

The distance on trees in T_Σ^D, $\mu(\alpha,\beta)$, is defined as the smallest integer k for which $\alpha \mid\!\!\overset{\eta^k}{}\!\!\beta$, if α and β are two trees in T_Σ^D for some $D \subset U$. The function μ is symmetric and satisfies triangle inequality.

Let L be a tree language, and tree $\beta \notin L$, the essence of ECTA is to search for a tree α, $\alpha \in L$ such that

$$\mu(\alpha,\beta) = \min_{\nu}\{\mu(\nu,\beta) \mid \nu \in L,\ D_\nu = D_\beta\} \qquad (1)$$

and reconstruct β as α. Eq. (1) is also defined as the distance of β from L, denoting $\mu_L(\beta)$. α is called the minimum-distance correction of β in L.

2.1. Minimum-Distance ECTA

By using the idea of adding terminal error production rules corresponding to substitution error transformations, as proposed by Aho and Peterson [1], the covering grammar $G_t' = (V', r', P', S)$ of a given tree grammar $G_t = (V, r, P, S)$ is constructed as follows:

Step 1. $V' = (V-\Sigma) \cup \Sigma'$ where $\Sigma' \supseteq \Sigma$ is a new set of terminal symbols.

Step 2. For each $y \in \Sigma'$ add to P'

$$X_0 \to \overset{y}{\underset{X_1 \cdots X_{r(x)}}{\diagup\diagdown}} \text{, if } X_0 \to \overset{x}{\underset{X_1 \cdots X_{r(x)}}{\diagup\diagdown}} \in P$$

or $X_0 \to y$ if $X_0 \to x$ is in P.

The language generated from G_t' consists of the language $L(G_t)$ and its corresponding erroneous trees. Hence, $L(G_t')$ can be written as

$$L(G_t') = \{\alpha' \mid \alpha' \in T_{\Sigma'},\ \text{and } \exists\, \alpha \in L(G_t) \text{ such that } D_{\alpha'} = D_\alpha\}$$

Following the concept of tree automaton, the ECTA is a backward procedure for constructing a tree-like transition table. Assume that α is the input tree. For each node $a \in D_\alpha$ there is a corresponding set of triplets, denoting t_a, in the transition table. Each triplet (X,ℓ,k) is added to t_a if X is a candidate state of node a, ℓ is the minimum number of errors in subtree α/a when node a is represented by state X, and k specifies the production rule used.

For a given tree $G_t = (V, r, P, S)$ the tree automaton that accepts $L(G_t)$ and emits a parse that consists of the minimum number of error production rules is given as follows:

ALGORITHM 1. Minimum-Distance ECTA.

Input: $G_t = (V, r, P, S)$ and tree α.

Output: Transition table of α and $\mu_{L(G_t)}(\alpha)$.

Method:

(1) If $r[\alpha(a)] = 0$, $\alpha(a) = x$, then add to t_a

(a) $(X_0,0,k)$ if $X_0 \to x$ is the k^{th} rule in P.

(b) $(X_0,1,k)$ if $X_0 \to y$ is the k^{th} rule in P and $y \neq x$.

(2) If $r[\alpha(a)] = n > 0$, $\alpha(a) = x$, then add to t_a

(a) (X_0,ℓ,k), if $X_0 \to \overset{x}{\underset{X_1 \cdots X_n}{\diagup\diagdown}}$ is the k^{th} rule in P and $(X_1,\ell_1,k_1) \in t_{a\cdot 1}, \ldots, (X_n,\ell_n,k_n) \in t_{a\cdot n}$ then $\ell = \ell_1 + \ldots + \ell_n$

(b) (X_0,ℓ,k), if $X_0 \to \overset{y}{\underset{X_1 \cdots X_n}{\diagup\diagdown}}$ is the k^{th} rule in P, $y \neq x$, and $(X_1,\ell_1,k_1) \in t_{a\cdot 1}, \ldots, (X_n,\ell_n,k_n) \in t_{a\cdot n}$ then $\ell = \ell_1 + \ldots + \ell_n + 1$.

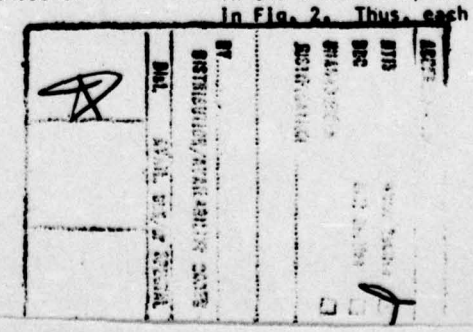
(3) Whenever more than one item in t_a has the same state, delete the item with larger number of errors.

(4) If $(S,\ell,k) \in t_0$, then $\mu_{L(G_t)}(\alpha) = \ell$. If no item in t_0 is of the form (S,ℓ,k), then no tree in $L(G_t)$ is in tree domain D_α, the input tree is rejected.

The parse of α can easily be traced out from the transition table.

The tree grammar in the following example is part of the highway grammar used in Section 3. In the meantime, we use it as an example here to illustrate the operation of ECTA and to demonstrate the highway patterns recognition procedure that will be discussed in Section 3.

EXAMPLE 1. Consider a set of vertical line patterns as given in Fig. 1. Assume that elements in the 4 x 4 array are connected as a tree shown in Fig. 2. Thus, each pattern has its



†† The notations and definitions of trees, tree grammars, and tree automaton follow those of Brainerd [4].

corresponding tree representation. For example, pattern (b) in Fig. 1 can be represented by the tree shown in Fig. 3, where nodes labeled by symbol "b" represent blank elements "□", and nodes labeled by "h" represent highway elements "▨". The tree grammar that generates these tree representations can be written as:

$G_H = (V, r, P, S)$ over $<\Sigma, r>$ where

$V = \{S, A_0, A_1, A_2, A_3, X_0, I_1, I_2, I_3, \$, b, h\}$

$\Sigma = \{ \cdot \quad \square \quad \blacksquare \}$

$\$ \quad b \quad h$

$r(\$) = 1, \quad r(b) = \{0, 1, 3\}, \quad r(h) = \{0, 1, 3\}$

P: $S \rightarrow$ (1) $\$(A_0)$ (2) $\$(A_1)$ (3) $\$(A_3)$ (4) $\$(A_4)$

(5) $A_0 \rightarrow h(A_0\ X_0\ X_0)$ (6) $h(A_0)$ (7) h

(8) $A_1 \rightarrow b(X_0\ A_0\ I_1)$ (9) $A_2 \rightarrow b(X_0\ A_1\ I_2)$

(10) $A_3 \rightarrow b(X_0\ A_2\ I_3)$ (11) $I_1 \rightarrow h(X_0)$

(12) $I_2 \rightarrow b(I_1)$ (13) $I_3 \rightarrow b(I_2)$

(14) $X_0 \rightarrow b(X_0\ X_0\ X_0)$ (15) $b(X_0)$ (16) b

(17) $I_1 \rightarrow h$

Given a noisy pattern and its tree representation β, which are shown in Fig. 4(a) and (b) respectively, the recognition of β by using ECTA is shown in Fig. 5. Since (S, 3, 2) is in t_0, β is accepted by the ECTA, and the number of errors is 3. The parse that generates the corresponding correct tree of β is shown in Fig. 6.

2.2. Maximum-Likelihood ECTA

When the probability distribution of patterns and/or deformation probabilities of each terminal are available, error-correcting parsing based on maximum-likelihood criterion become a plausible approach in handling noisy patterns. Definitions and properties on stochastic grammar, terminal deformation probabilities, and maximum-likelihood error-correcting parsers have been introduced in [10, 13, 16].

The expansive stochastic tree grammars and languages defined in [3] are briefly reviewed.

DEFINITION 1. A stochastic tree grammar $G_S = (V, r, P, S)$ over V_T is expansive if and only if each rule in P is of the form

$X_0 \xrightarrow{p} x(X_1 \cdots X_{r(x)})$ or $X_0 \xrightarrow{p} x$ where $x \in \Sigma$

and $X_0, X_1 \cdots, X_{r(x)} \in V - \Sigma$ are nonterminals.

DEFINITION 2.

$$L(G_S) = \{(\alpha, p(\alpha)) \mid \alpha \in T_\Sigma,\ S \xrightarrow{p_i} \alpha,\ i = 1 \cdots k,\ p(\alpha) = \sum_{i=1}^{k} p_i\}$$

where k is the number of all distinctly different derivation of α from S, and p_i is the probability associated with the i^{th} distinct derivation of α from S.

Assume that the occurrence of substitution error on a terminal is independent from its neighboring terminals. Fund and Fu [13] define substitution error as a stochastic mapping: $\delta: \Sigma \rightarrow \Sigma$ such that $\delta(a) = b$, $a, b \in \Sigma$, with probability $q(b|a)$, and furthermore,

$$\sum_{b \in \Sigma} q(b|a) = 1 \qquad (2)$$

Assume that $t = t_1 \cdots t_n\ x$ is a term over $<\Sigma, r>$. We have

$$\delta(t_1 \cdots t_n\ x) = \delta(t_1) \cdots \delta(t_n)\delta(x) \qquad (3)$$

A tree $t = t_1 \cdots t_n\ x$ is said to be locally corrupted under δ mapping. If two trees $\alpha = t_1 \cdots t_n\ x$, $\alpha' = t_1' \cdots t_n'\ x'$ are in T_Σ^D, the probability of α' being the noisy deformed tree of α is

$$q(\alpha'|\alpha) = q(t_1'|t_1) \cdots q(t_n'|t_n)\, q(x'|x) \qquad (4)$$

We further have

$$\sum_{\alpha' \in T_\Sigma^D} q(\alpha'|\alpha) = 1$$

for ang $\alpha \in T_\Sigma^D$, $D \supseteq U$ (5)

For a given stochastic grammar $G_S = (V, r, P, S)$ over $<\Sigma, r>$, when the deformation probabilities $q(y|x)$ are known for all $x \in \Sigma$, $y \in \Sigma$, the

stochastic covering grammar becomes $G_S' = (V', r', P', S)$ over $\langle \Sigma', r' \rangle$ where $V' = (V-\Sigma) \cup \Sigma'$ and $\Sigma' \subseteq \Sigma$ is the set of terminal symbols, and for all $y \in \Sigma'$ $X_0 \xrightarrow{p'} y$ is in P' if $X_0 \xrightarrow{p} x$ is in P, or $X_0 \xrightarrow{p'}$  is in

P' if $X_0 \xrightarrow{p}$  is in P and $p' = p\,q(y|x)$.

Apparently, $L(G_S')$ can be written as $L(G_S') = \{(\alpha', p'(\alpha')) | \alpha' \in T_{\Sigma'}$,

$$p'(\alpha') = \sum_{\substack{\alpha \in L(G_S) \\ D_{\alpha'} = D_\alpha}} q(\alpha'|\alpha)\, p(\alpha)\}$$

Suppose that the given noisy input tree α' is in tree domain D, i.e., $\alpha' \in T_\Sigma^D$, the maximum-likelihood error-correcting decision rule in this case is to choose a tree α in $L(G_S)$ of domain D, i.e., $\alpha \in L(G_S)$ and $\alpha \in T_\Sigma^D$, such that

$$q(\alpha'|\alpha)p(\alpha) = \max_{\beta \in L(G_S) \cap T_\Sigma^D} q(\alpha'|\beta)p(\beta) \tag{6}$$

We call this value, $q(\alpha'|\alpha)p(\alpha)$, the probability of α' being a noise deformed tree of $L(G_S)$ and denote it as $q(\alpha'|L)$.

The structure-preserved maximum-likelihood ECTA is given as follows:

ALGORITHM 2. Maximum-Likelihood ECTA

Input: (1) Stochastic grammar $G_S = (V, r, P, S)$.

(2) Deformation probabilities $q(y|x)$ for all $x \in \Sigma$, $y \in \Sigma'$.

(3) Input tree α.

Method: (1) If $r[\alpha(a)] = 0$, $\alpha(a) = y$ then add to t_a, (X_0, p', k), if $X_0 \xrightarrow{p} x$ is the k^{th} rule in P and $p' = p\,q(y|x)$.

(2) If $r[\alpha(a)] = n$, $n > 0$, $\alpha(a) = y$ then add to t_a (X_0, p', k), if $X_0 \xrightarrow{p}$  is the k^{th} rule in P and $(X_1, p_1', k_1) \in t_{a\cdot 1}$ ... $(X_n, p_n', k_n) \in t_{a\cdot n}$ then

$p' = p_1' * \ldots * p_n' * p * q(y|x)$.

(3) Whenever more than one item in t_a have the same state, delete the item associated with smaller probability.

(4) If $(S, p', k) \in t_0$, then $q(\alpha|L) = p'$.

If no item in t_0 is associated with the start state S, then no tree in $L(G_S)$ is in tree domain D_α. Input tree is rejected.

3. Application of ECTA to Recognition of Highway Patterns from LANDSAT Data

Recently, syntactic methods have been used to analyze and interpret data obtained from the earth resource technology satellite (LANDSAT) [5, 14]. The input data used by Brayer and Fu or Li and Fu are the results of pointwise classification [25]. Each pixel collected by LANDSAT represents a ground area of approximately 60 x 70 m^2. According to spectral and/or temporal measurements of the object, a pixel is then classified into classes of water, cloud, downtown, concrete, or grass, etc. Due to the resolution size, spectral signals of smaller objects are usually composed of reflectance of several different kinds of ground cover. For instance, the spectral signal of a segment of a highway actually results from a combined reflectance of concrete surface, grass, and transportation vehicles. Consequently, the variation of size of smaller objects and their surroundings changes their reflectances, and thus, their spectral properties from point to point. This uncertainty causes some difficulty in setting threshold for classification based on spectral information of individual points only. One example of the results of pointwise classified patterns is given in Fig. 7 which covers the area of the northern part of Grand Rapids, Michigan. Each symbol "H" represents a pixel that is classified as a segment of highway. Fig. 8, which is obtained from the official highway map, indicates the major divided highways of the same area.

As illustrated in Fig. 7, the inadequate resolution of highways and the mass of scattered concrete and grass-mixed objects other than highways result in discontinuity of highways and spurious points from pointwise classification. Therefore, the need for using syntactic methods as a refinement becomes evident. Syntactic pattern recognition has the advantage of using contextual and structural information contained in patterns for recognition purposes.

Brayer and Fu [5] use web grammars modeling classes of clouds, downtown, highways, etc. Due to the lack of an efficient parsing procedure for web grammar, a matching process is used in the actual recognition of patterns. In [14], Li and Fu use a tree grammar and a tree automaton to analyze line patterns such as highways and rivers in the Grand Rapids area. It demonstrates fairly good results in recognizing rivers among ponds

and some modern buildings with glass walls having similar reflecting surfaces as water, but poor in analyzing highway patterns. Evidently, highway patterns are usually too noisy to be effectively analyzed by a conventional grammatical inference procedure and parsing methods.

In [5] and [14], pictures are scanned window by window. A window is an array of pixels. Each pixel is either represented by symbol "H" or is a blank. Some subpatterns consisting of several pixels are selected as primitives. A syntactic method is then used to decide whether the pattern in a window is a desired pattern. In our application, we choose the size of window to be an 8 x 8 array of pixels, and the labels on single pixels to be primitives. Thus, we have two kinds of primitives: "h" represent a segment of highway and "b" represent a nonhighway area.

Similar to the procedure illustrated in Example 1, an array of primitives in a window are connected as a tree. Each primitive becomes a labeled node in the tree representation. We fix the tree domain to be D_H, and allow node label to be either "h" or "b." Hence, there are 2^{64} tree representations in T_Σ^D, where $\Sigma = \{b,h,\$\}$, \$ is a start terminal. Apparently, the set of all possible patterns in an 8 x 8 window and the set of all labeled trees in the tree domain D_H are one-to-one correspondence.

Assume that only a single segment of highway or at most two intersected highways that appeared within a window is considered. Hence, positive sample patterns are patterns of vertical, horizontal, diagonal lines or two intersected lines. A tree grammar is then inferred to generate the set of positive sample patterns. This highway grammar is given in Appendix A of [15]. The idea of using error-correcting tree automaton is to measu re the distance between the input pattern and patterns in the set of positive samples. If the input pattern is not one of the positive sample patterns, it will further be reconstructed to its best matching pattern according to the minimum-distance criterion. Details of the recognition procedure are discussed in [15].

The error-correcting scheme of the highway recognition problem is programmed in Fortran IV on a CDC 6500 computer and tested by using the data shown in Fig. 7. The result is shown in Fig. 9. There are 80 x 160 pixels in the input data. The cpu time for processing is 150 sec.

We also use the highway grammar which is inferred from the Grand Rapids data to analyze some other noisy data, such as data obtained from Lafayette, Indiana. The pointwise classified data of Lafayette is shown in Fig. 10 which contains 125 x 125 pixels. The result of the error-correcting analysis is shown in Fig. 11. The cpu time used is 101 sec. For comparison, we use the highway map shown in Fig. 12 as ground truth.

4. Conclusion

The formulations of structure-preserved error-correcting tree automaton both for minimum-distance criterion and maximum-likelihood criterion are presented. Its application to LANDSAT data analysis is illustrated in Section 3. An alternative solution, if not using the error-correcting scheme, of course, is to obtain a sufficient set of positive samples and a set of negative samples from training data, and to infer a grammar so that all the negative samples are excluded from the language and all the positive samples are included [11]. It is difficult to infer such a grammar when patterns are irregular. Besides, due to the large variation of input data, a slight difference in the training set will cause some patterns to be rejected during parsing.

Due to the restriction on using fixed tree domains, the application of structure-preserved error-correcting tree automaton lacks flexibility. To remove this restriction, a generalized error-correcting tree automaton is under study [15] where special types of insertions and deletions are considered.

References

1. Aho, A. V. and T. G. Peterson, "A Minimum Distance Error-Correcting Parser for Context-Free Languages," SIAM J. Comput., Vol. 4, December, 1972.
2. Bahl, L. R. and F. Jelinek, "Decoding for Channels with Insertions, Deletions, and Substitutions with Applications to Speech Recognition," IEEE Trans. on Information Theory, Vol. IT-21, No. 4, July, 1975.
3. Bhargava, B. K., "Tree Systems for Syntactic Pattern Recognition," Ph.D. Thesis, Purdue University, May, 1974.
4. Brainerd, W. F., "Tree Generating Regular Systems," Inf. and Control, 14, 217-231, 1969.
5. Brayer, J. M. and K. S. Fu, "Web Grammars and Their Application to Pattern Recognition," Purdue University, TR-EE 75-1, December, 1975.
6. Doner, J., "Tree Acceptors and Some of Their Applications," J. of Comut. and Sys. Sci. 4, pp. 406-451, 1970.
7. Ellis, C. A., "Probabilistic Tree Automata," Information and Control, Vol. 19, pp. 401-416, 1971.
8. Fu, K. S., "On Syntactic Pattern Recognition and Stochastic Languages," Frontiers of Pattern Recognition, ed. S. Watanabe, Academic Press, 1972.
9. Fu, K. S., "Stochastic Language for Picture Analysis," Computer Graphics and Image Processing, Vol. 2, pp. 433-453, 1973.
10. Fu, K. S., Syntactic Methods in Pattern Recognition, Academic Press, 1974.
11. Fu, K. S. and T. L. Booth, "Grammatical Inference: Introduction and Survey - Part I and Part II," IEEE Trans. on Systems, Man, & Cybernetics, Vol. SMC-5, 1975.
12. Fu, K. S. and B. K. Bhargava, "Tree Systems for Syntactic Pattern Recognition," IEEE Trans. on Computers, Vol. C-22, No. 12, 1087-1099, December, 1973.

13. Fung, L. W. and K. S. Fu, "Stochastic Syntactic Decoding for Pattern Classification," IEEE Trans. on Computers, Vol. C-24, No. 6, July, 1975.
14. Li, R. Y. and K. S. Fu, "Tree System Approach for LANDSAT Data Interpretation," Purdue-LARS Symposium on Machine Processing of Remotely Sensed Data, W. Lafayette, Indiana, June, 1976.
15. Lu, S. Y. and K. S. Fu, "Error-Correcting Syntax Analysis for Tree Languages," Purdue University, TR-EE 76-24, July, 1976.
16. Lu, S. Y. and K. S. Fu, "Efficient Error-Correcting Syntax Analysis for Recognition of Noisy Patterns," Purdue University, TR-EE 76-9, March, 1976.
17. Moayer, B. and K. S. Fu, "Syntactic Pattern Recognition of Fingerprints," Purdue University, TR-EE 74-36, December, 1974.
18. Rounds, W. C., "Mappings and Grammars on Trees," Math. System Theory, Vol. 4, No. 3, 257-286, 1970.
19. Tanaka, E. and K. S. Fu, "Error-Correcting Parser for Formal Language," Purdue University, TR-EE 76-7, March, 1976.
20. Thomason, M. G., "Errors on Regular Language," IEEE Trans. on Computers, Vol. C-23, No. 6, January, 1974.
21. Thomason, M. G. and R. C. Gonzalez, "Syntactic Recognition of Imperfectly Specified Patterns," IEEE Trans. on Computers, January, 1975.
22. Thompson, R. A., "Language Correction Using Probabilistic Grammars," IEEE Trans. on Computers, Vol. C-25, No. 3, March, 1976.
23. Thatcher, J. W., "Characterizing Derivation Trees of Context-Free Grammars Through a Generation of Finite Automata Theory," J. of Comput. and Sys. Sci., 1, 1967, 317-322.
24. Thatcher, J. W., "Tree Automata: An Informal Survey," Currents in the Theory of Computing, ed. by A. V. Aho, Prentice Hall, 1973.
25. Todd, W. J. and M. F. Baumgardner, "Land Use Classification of Marion County, Indiana, Data," LARS Information Note 101673, LARS, Purdue University, 1973.
26. Wagner, R. A. and M. J. Fisher, "The String to String Correction Problem," J. ACM, Vol. 21, No. 1, January, 1974.

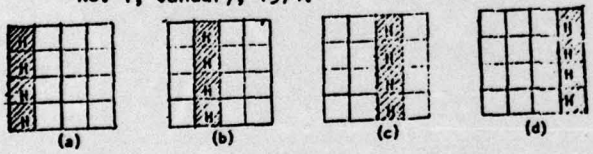


Fig. 1. Vertical Line Patterns

Fig. 2. A Tree ← Structure

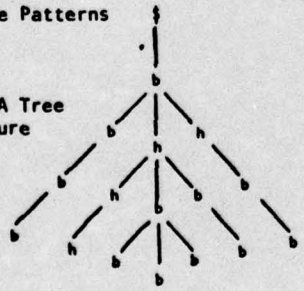


Fig. 3. Tree Representation of Pattern (b) →

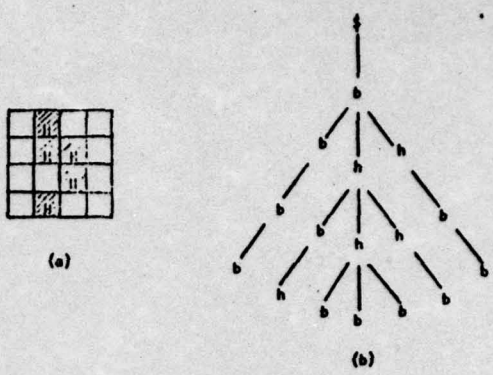


Fig. 4. A Noisy Pattern and Its Tree Representation β

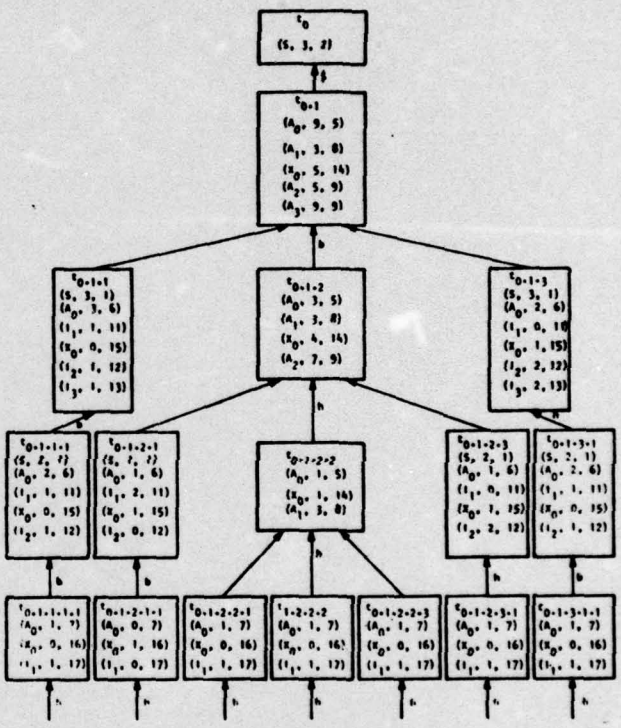


Fig. 5. Transition Table of Tree β

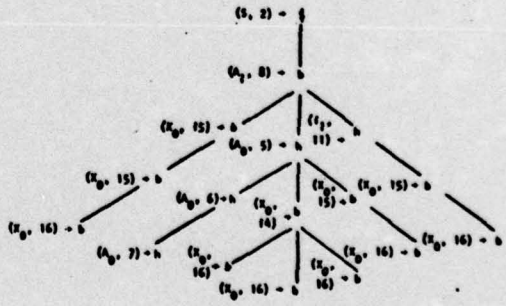


Fig. 6. The Parse of β

Fig. 7. Pointwise Classified Highway Data Obtained from Grand Rapids, Mich.

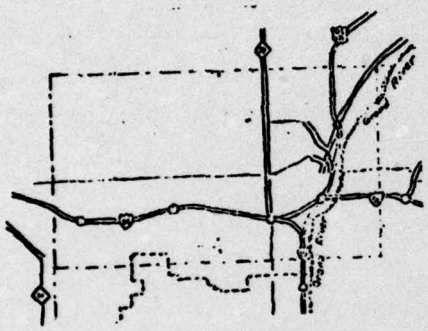


Fig. 8. Divided Highway Map of Northern Part of Grand Rapids, Mich.

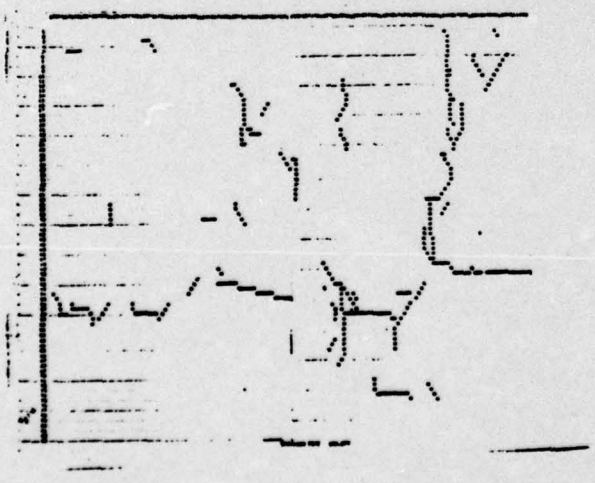


Fig. 9. ECTA Processed Result of the Grand Rapids Area

Fig. 10. Pointwisely Classified Data of the Lafayette, Ind. Area

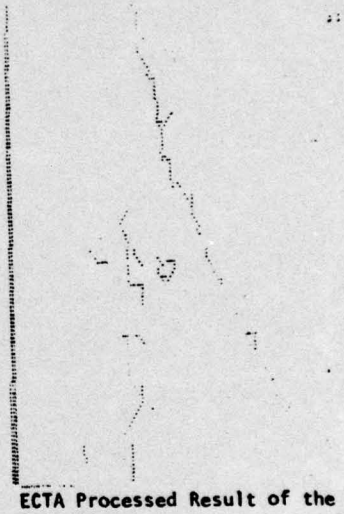


Fig. 11. ECTA Processed Result of the Lafayette, Ind. Area

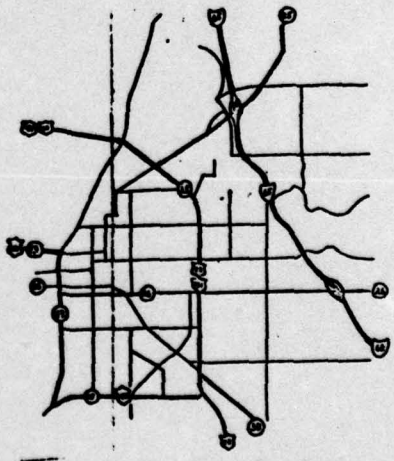


Fig. 12. Highway Map of Lafayette, Ind.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: AFOSR - TR-76-1311	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): STRUCTURE-PRESERVED ERROR-CORRECTING TREE AUTOMATA FOR SYNTACTIC PATTERN RECOGNITION	5. TYPE OF REPORT & PERIOD COVERED: Interim
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): S. Y. Lu, K. S. Fu	8. CONTRACT OR GRANT NUMBER(s): AF-AFOSR-2661-74
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Purdue University, School of Electrical Engineering, West Lafayette, IN 47907	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F 2304/A2
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Office of Scientific Research/NM, Bolling AFB, Washington, DC 20332	12. REPORT DATE: 1976
	13. NUMBER OF PAGES: 7
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): 8 p.	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An error-correcting syntax analyzer for tree languages with substitution errors, called structure-preserved error-correcting tree automaton (ECTA), is studied. Substitution errors are defined in terms of transformation which can easily be accommodated to linguistic notion. Let L be a tree language, for a tree β not in L, the essence of ECTA is to search for a tree α in L such that the cost sequence of error transformations needed to transform α to β is the minimum among all the sentences in L. A LANDSAT data interpretation problem is used as an example to illustrate the operation of ECTA.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED 292 000

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)