

AD-A077 391

MARYLAND UNIV COLLEGE PARK COMPUTER SCIENCE CENTER
PARALLEL SIGMA-ERASING ARRAY ACCEPTORS.(U)

F/G 9/2

UNCLASSIFIED

JUL 79 A NAKAMURA
TR-784

AFOSR-77-3271

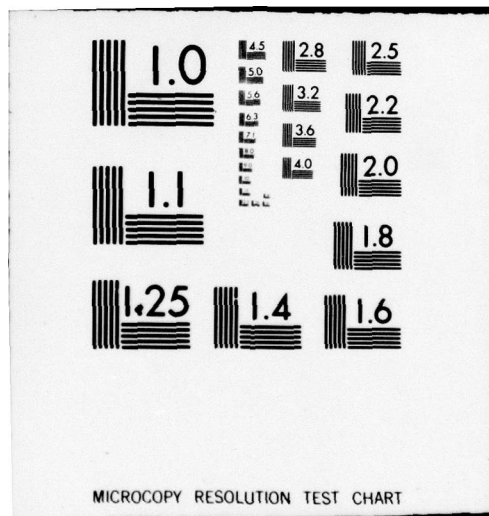
AFOSR-TR-79-1163

NL

| OF |
ADA
077391



END
DATE
FILMED
12-79
DDC

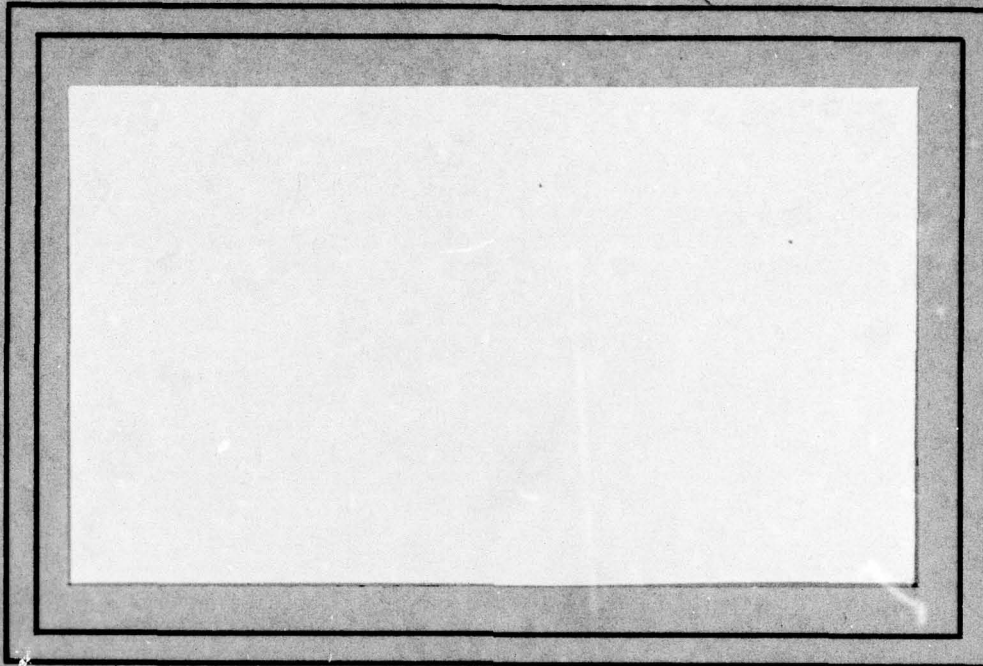


AEOSR-TR- 79 - 1163

10

LEVEL

AD A 077391



D D C
RECEIVED
NOV 29 1979
A

UNIVERSITY OF MARYLAND
COMPUTER SCIENCE CENTER

COLLEGE PARK, MARYLAND

20742

DDC FILE COPY

Approved for public release;
distribution unlimited.

70 11 05 001

TR-784
AFOSR-77-3271

July 1979

PARALLEL Σ -ERASING ARRAY ACCEPTORS

Akira Nakamura

Department of Applied Mathematics
Hiroshima University (Japan)
Hiroshima, Japan

and

Computer Science Center
University of Maryland
College Park, MD 20742

See 1473 in back

ABSTRACT

A parallel Σ -erasing array acceptor ($P\Sigma$ -EAA) is introduced. It is proved that the class accepted by $P\Sigma$ -EAA's is exactly the context-free array languages.

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

The support of the U.S. Air Force Office of Scientific Research under Grant AFOSR-77-3271 is gratefully acknowledged, as is the help of Kathryn Riley in preparing this paper.

1. Introduction

In [1], Rosenfeld introduced isotonic array grammars, and in [2] Cook and Wang presented a Chomsky hierarchy of isotonic array grammars and languages. Acceptors for type 0 and type 1 array languages were given in Milgram and Rosenfeld [3] and an acceptor for type 3 array languages was given in [2]. But an acceptor for type 2 or context-free languages remains as an open problem.

Cook and Wang's finite state array acceptors (FSAA's) for type 3 array languages are considered as erasing array acceptors in the following sense: At each step, an FSAA M is in some state q scanning an input symbol a . In a given move, M marks the scanned array symbol with ϕ (the input symbol is erased by changing it to a symbol with ϕ), changes state and moves one square left (L), right (R), up (U), or down (D). An array A is accepted iff M , beginning in its initial state scanning some array cell, scans and marks each non- $\#$ square of the A with a ϕ and ends in one of its final states. In this definition of acceptability, there is the problem that M itself never knows whether every non- $\#$ cell is marked with ϕ .

In this note, we introduce a kind of erasing array acceptor similar to the above mentioned one, which acts in parallel and erases input symbols according to given rules. Further, the acceptor is able to traverse the area of input symbols after

a cell falls into a special state. This ability is provided in order to avoid the above-mentioned problem. This automaton can be considered as a restricted cellular array acceptor; it is called a parallel Σ -erasing array acceptor ($P\Sigma$ -EAA).

The purpose of this note is to solve the open problem concerning the context-free array languages proposed in [2]. We prove that the class accepted by $P\Sigma$ -EAA's is exactly the context-free array languages. A key idea in the proof is based on a definition of a nondeterministic $P\Sigma$ -EAA such that it is able to retrace backward derivations in an isotonic context-free array grammar.

In this note, we assume that the reader is familiar with isotonic array grammars, languages, and their hierarchy given in [1], [2] and also with the fundamental definitions of cellular array acceptors as in [4].

Accession For	
NTIS GRA&I	☒
DDC TAB	☐
Unannounced	☐
Justification	☐
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

2. Definitions

We first give some notation and definitions about $P\Sigma$ -EAA's. In this note, we assume that input arrays are surrounded by the edge symbol #.

Definition 2.1 A parallel Σ -erasing array acceptor ($P\Sigma$ -EAA) is a two-dimensional cellular acceptor satisfying the following (i)-(vi):

- (i) For a set Σ of input states, a set Q of non-input states, and a set T of traversal states, $\Sigma \cap Q = \emptyset$, $\Sigma \cap T = \emptyset$, $Q \cap T = \emptyset$.
- (ii) For each $A \in Q$, A^L , A^R , A^U , and A^D are also in Q .
- (iii) $\emptyset \in Q$ is a special "dead" blank state.
- (iv) S^t , $S \in Q$ are the quasi-accepting state and the accepting state, respectively.
- (v) $\bar{S} \in Q$ is the "broken" state.
- (vi) δ is the transition function, i.e. $\delta: (\Sigma \cup Q \cup T)^5 \rightarrow 2^{\Sigma \cup Q \cup T}$

In this note, we consider exclusively nondeterministic acceptors. In Definition 2.1, $\delta(X, Y, Z, U, V) \ni W$ means that one of the values of δ over $\begin{array}{|c|c|c|} \hline & Y & \\ \hline X & V & Z \\ \hline & U & \end{array}$ is $\begin{array}{|c|c|c|} \hline & * & \\ \hline * & W & * \\ \hline & * & \end{array}$. That is, the first 4 arguments of δ are occupied by the 4 neighbors in the indicated positions of the last argument. For intuitive understanding, $\delta(X, Y, Z, U, V)$ is sometimes denoted by $\delta(X \begin{array}{c} Y \\ V \\ U \end{array} Z)$.

δ is defined as a mapping satisfying the following (1)-(10):

- (1) $\delta(X, Y, Z, U, \emptyset)$ is always $\emptyset$, except in the case of traversal mentioned in (9) below. This means that $\emptyset$ is the special dead blank state.
- (2) $\delta(X, Y, Z, U, A) \ni A^H$, where $H \in \{L, R, U, D\}$, in cases not contradicting (5) and (6) below.
- (3) For each $H \in \{L, R, U, D\}$ and $A \in Q$, $\delta(X, Y, Z, U, A^H)$ is always $\emptyset$.
Conditions (1) and (3) correspond to "erasing".
- (4) For each input state a , the first 4 variables X, Y, Z, U of $\delta(X, Y, Z, U, a)$ are dummy.
- (5) $\delta(A \begin{smallmatrix} B \\ E \\ D^U \end{smallmatrix} C)$ depends on E and D only. In the other cases, the situation is the same, i.e., for example $\delta(A \begin{smallmatrix} B \\ E \\ D \end{smallmatrix} C^L)$ depends on E and C only, $\delta(A^R \begin{smallmatrix} B \\ E \\ D \end{smallmatrix} C)$ depends on E and A only, and so on.
- (6) In case of a collision in (5), the acceptor goes into the broken state $\bar{S}$, i.e., for example, $\delta(A \begin{smallmatrix} B \\ E \\ D^U \end{smallmatrix} C^L) = \bar{S}$, $\delta(A^R \begin{smallmatrix} B \\ E \\ D \end{smallmatrix} C^L) = \bar{S}$, and so on.
- (7) For the broken state $\bar{S}$, $\delta(X, Y, Z, U, \bar{S}) = \bar{S}$.
- (8) $\delta(X \begin{smallmatrix} Y \\ E \\ U \end{smallmatrix} Z)$ takes a value different from E only when $X = A^R$, $Y = B^D$, $Z = C^L$, or $U = D^U$ for some A, B, C, D . This corresponds to the "context-free" property.
- (9) When a cell falls into the quasi-accepting state S^t , δ is defined to begin traversal of the input. The states of the set T are used for this traversal.
- (10) In the process of traversal, a cell changes to the broken state when two states of $\Sigma \cup Q$ are in the input area;

otherwise it falls into the accepting state at the starting cell of S^t after finishing the traversal. The detailed definition of δ to satisfy (10) is complicated and will not be given here.

3. Main theorem

We prove that the class accepted by $P\Sigma$ -EAA's is exactly the isotonic context-free array languages. Let G be an isotonic context-free array grammar. $L(G)$ represents the language generated by this grammar. Also, let M be a $P\Sigma$ -EAA. $T(M)$ means the set accepted by this acceptor M .

Lemma 3.1 Any context-free array language is accepted by a $P\Sigma$ -EAA.

Proof:

Let G be a type 2 grammar $G=(N,T,\#,P,S)$ and let M be a $P\Sigma$ -EAA. We prove that $\forall G \exists M (L(G)=T(M))$.

Any isotonic context-free array language (ICFAL) is generated by the following normal form of rewriting ([2]):

$$\#A \rightarrow BC, \quad A\# \rightarrow BC, \quad \begin{matrix} \# \\ A \end{matrix} \rightarrow \begin{matrix} B \\ C \end{matrix}, \quad \begin{matrix} A \\ \# \end{matrix} \rightarrow \begin{matrix} B \\ C \end{matrix}, \quad A \rightarrow a.$$

Therefore, an array generated by the normal form is accepted by a $P\Sigma$ -EAA which retraces backward rewriting rules. That is,

δ of this $P\Sigma$ -EAA is defined as follows:

$$\begin{aligned} \delta(\dots\dots\dots a) &\ni A && \text{if } A \rightarrow a \text{ is in } P, \\ \delta(\cdot \begin{matrix} \cdot \\ B \\ \cdot \end{matrix} C^L) &\ni A && \text{if } A\# \rightarrow BC \text{ is in } P, \\ \delta(B^R \begin{matrix} \cdot \\ C \\ \cdot \end{matrix} \cdot) &\ni A && \text{if } \#A \rightarrow BC \text{ is in } P, \\ \delta(\cdot \begin{matrix} B \\ CU \\ BD \end{matrix} \cdot) &\ni A && \text{if } \begin{matrix} A \\ \# \end{matrix} \rightarrow \begin{matrix} B \\ C \end{matrix} \text{ is in } P, \\ \delta(\cdot \begin{matrix} C \\ \cdot \end{matrix} \cdot) &\ni A && \text{if } \begin{matrix} \# \\ A \end{matrix} \rightarrow \begin{matrix} B \\ C \end{matrix} \text{ is in } P. \end{aligned}$$

In this definition, if a symbol A in the production is S , then the corresponding A for δ is S^t . It is easily seen that an input array ends in the accepting state S at some square and each non- $\#$ cell of the array goes into $\emptyset$ iff the input array is generated by G .//

Lemma 3.2 The set accepted by a $P\Sigma$ -EAA is a context-free array language.

Proof:

The proof is similar to Lemma 3.1. We prove that $\forall M \exists G (L(G) = T(M))$. We define an isotonic context-free array grammar $G = (N, T, \#, P, S)$ from a given $P\Sigma$ -EAA M as follows:

$$\begin{array}{ll}
 A \rightarrow a & \text{if } \delta(\dots, a) \ni A \\
 A\# \rightarrow BC & \text{if } \delta(\cdot \begin{smallmatrix} \cdot \\ B \\ \cdot \end{smallmatrix} C^L) \ni A \\
 \#A \rightarrow BC & \text{if } \delta(B^R \begin{smallmatrix} \cdot \\ C \\ \cdot \end{smallmatrix}) \ni A \\
 \begin{smallmatrix} A \\ \cdot \end{smallmatrix} \rightarrow \begin{smallmatrix} B \\ C \end{smallmatrix} & \text{if } \delta(\cdot \begin{smallmatrix} B \\ C^u \end{smallmatrix} \cdot) \ni A \\
 \begin{smallmatrix} \cdot \\ \cdot \end{smallmatrix} \rightarrow \begin{smallmatrix} B \\ C \end{smallmatrix} & \text{if } \delta(\cdot \begin{smallmatrix} B^D \\ C \end{smallmatrix} \cdot) \ni A \\
 S \rightarrow S^t &
 \end{array}$$

For the other cases of δ , the corresponding rewriting rules are not given. It can be seen without difficulty that an

input array is generated by this isotonic context-free array grammar G iff it is accepted by M .//

Theorem 3.3 The class accepted by PE -EAA's is exactly the isotonic context-free array languages.

Proof:

This is obtained from Lemmas 3.1 and 3.2.//

4. Remark

In this note, we have considered parallel Σ -erasing array acceptors for the isotonic context-free array languages. We could also define sequential acceptors which simulate $P\Sigma$ -EAA's; but it seems that these acceptors are somewhat more complicated.

Acknowledgement

The author would like to thank Professor Azriel Rosenfeld for arranging a very enjoyable four week stay at the Computer Science Center, July 1979, and for valuable comments. He also thanks Dr. T. Ae of Hiroshima University for his useful suggestions.

References

- [1] A. Rosenfeld: Isotonic grammars, parallel grammars and picture grammars, in Machine Intelligence 6 (D. Michie and B. Meltzer, Eds.), 281-294, Univ. of Edinburgh Press, Edinburgh, Scotland, 1971.
- [2] C. R. Cook and P. S. Wang: A Chomsky hierarchy of isotonic array grammars and languages, Computer Graphics and Image Processing 8, 144-152, 1978.
- [3] D. L. Milgram and A. Rosenfeld: Array automata and array grammars, Proc. IFIP Congress 71, booklet TA-2, 166-173, North-Holland, Amsterdam, 1971.
- [4] A. Rosenfeld: Picture Languages, Academic Press, New York, 1979.

Unclassified

19 TR-79-1163

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle)	5. TYPE OF REPORT & PERIOD COVERED	
6. AUTHOR(s)	7. PERFORMING ORG. REPORT NUMBER	
8. AUTHOR(s)	9. CONTRACT OR GRANT NUMBER(s)	
10. PERFORMING ORGANIZATION NAME AND ADDRESS	11. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS	
12. CONTROLLING OFFICE NAME AND ADDRESS	13. REPORT DATE	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report)	
16. DISTRIBUTION STATEMENT (of this Report)		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
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)		

18 AFOSR

6 PARALLEL Σ -ERASING ARRAY ACCEPTORS.
sigma

10 Akira Nakamura

9 Technical Repts.

14 TR-784

15 AFOSR-77-3271

Computer Science Center
University of Maryland
College Park, MD 20742

17 A2
6702F 16 2344/A2

Math. & Info. Sciences, AFOSR/NM
Bolling AFB
Wash., DC 20332

11 Jul 79

12 13

Unclassified

Approved for public release; distribution unlimited.

Automata
Formal languages

Sigma

A parallel Σ -erasing array acceptor (PE-EAA) is introduced. It is proved that the class accepted by PE-EAA's is exactly the context-free array languages.

443 140