INFORMATION AND CONTROL 3, 372–375 (1960)

Note on the Boolean Properties of Context Free Languages*

STEPHEN SCHEINBERG

Massachusetts Institute of Technology, Department of Mathematics, Cambridge, Massachusetts†

A language is considered to be any set of sentences (strings) made up from a finite vocabulary (alphabet). A grammar is a device which enumerates a language, e.g., a Turing machine with any set of input signals, a finite automaton, or a Post system. One often considers the family of languages which have grammars meeting certain specifications or restrictions, e.g., the finite state languages (the languages generatable by finite automata). A major question of interest is whether such a family is a closed system with respect to the Boolean operations: set-theoretic union, intersection, and difference. The family of recursively enumerable languages is well known to be closed under union and intersection, but not under difference. It is also well known that the family of finite state languages is closed under all three operations. In this note we answer this question for another family of languages, called "type 2" or "context free" by Chomsky (1959).

For the sake of having a convenient reference we shall present in modified form Chomsky's definitions. It will be seen that a context free grammar is essentially a special case of a semi-Thue system, cf. Davis (1958).

We consider a *context free* (CF) *grammar* to be a finite set G of "rewriting rules" $\alpha \rightarrow \psi$, where α is a single symbol and ψ is a finite string of symbols from a finite alphabet (vocabulary) V. V contains precisely the symbols appearing in these rules plus the "boundary" symbol $\#$, which does not appear in these rules. Rules of the form $\alpha \rightarrow \alpha$ (which

* I am indebted to Prof. Noam Chomsky of the Massachusetts Institute of Technology for helpful discussion and encouragement.

† Present address: % Department of Mathematics, Princeton University, Princeton, New Jersey.

have no effect) are not allowed. We define V_N (nonterminal vocabulary) and V_T (terminal vocabulary) by requiring that $V_N \cup V_T = V$, $V_N \cap V_T$ be empty, and V_N = all α in V which appear on the left in a rule of G; i.e., all α such that there is a string φ with $\alpha \to \varphi$ a rule of G. Furthermore, we distinguish an "initial" symbol S, which belongs to V_N.

Notational convention: If $\rho = \alpha_1 \cdots \alpha_n$ and $\sigma = \beta_1 \cdots \beta_m$, then $\rho\sigma$ is the string $\alpha_1 \cdots \alpha_n\beta_1 \cdots \beta_m$. The *length* of ρ is n. ρ^k stands for $\rho \cdots \rho$ (k times).

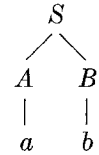
G generates a binary relation $\to$ on the set of all strings of symbols of V: $\chi \to \omega$ if and only if there are strings τ_1, τ_2, α, σ such that $\chi = \tau_1\alpha\tau_2$, $\omega = \tau_1\sigma\tau_2$, and $\alpha \to \sigma$ is a rule of G. The (improper) ancestral of $\to$ is denoted $\Rightarrow$: $\varphi \Rightarrow \psi$ (φ *dominates* ψ) if and only if there are strings $\chi_0, \cdots, \chi_n$ ($n \geqq 0$) such that $\chi_0 = \varphi$, $\chi_n = \psi$, and $\chi_i \to \chi_{i+1}$ ($i = 0, \cdots, n-1$). We say that ψ is derived from φ in n steps.

A string is called *terminal* (and is denoted $a,b, \cdots, x,y,z$) if it consists entirely of elements of V_T. The *terminal language* L_G generated by G is the set of all terminal strings x such that $\$S\$ \Rightarrow x$ (equivalently, all $\$y\$$ such that $S \Rightarrow y$). A language L is called *context free* (CF) if there is a CF grammar G such that $L = L_G$; then G is called a *CF grammar for* L.

Comment: Every finite state language is CF, and every CF language is recursive, cf. Chomsky (1959).

We shall henceforth assume that every symbol α of V_N is part of some string dominated by S and that every α dominates a terminal string. There is no loss of generality in doing this. In fact, given a CF grammar G we can effectively find a CF grammar G' meeting this restriction and generating the same terminal language as G.

Observe that derivations for a CF grammar G can be represented (in a many-one manner) by trees in which we label a node by a single symbol and we connect lines from the node labeled by α to each of the symbols of the string φ when α is rewritten φ. For example, we would have the tree



for either of the derivations $S \to AB \to Ab \to ab$ or $S \to AB \to aB \to ab$.

LEMMA 1. *If* L_1, L_2 *are CF languages, then so is* $L_1 \cup L_2$.

Proof: Let G_1, G_2 be CF grammars for L_1, L_2. For $i = 1, 2$, replace at every occurrence each nonterminal symbol α of G_i by α_i, where the α_i are new and distinct, and add the rule $S \to S_i$ to the resulting set of rules of G_i. We thus obtain CF grammars G_1', G_2' for L_1, L_2. S (the "initial symbol") is the only nonterminal symbol common to both, and appears only in the first step of a derivation. For the CF grammar

$$G_1' \cup G_2'$$

the use of S is precisely to determine whether the resulting terminal string is to be one of L_1 or one of L_2. Thus $G_1' \cup G_2'$ is a CF grammar for $L_1 \cup L_2$.

LEMMA 2. *If $\varphi_1, \varphi_2, \psi$ are strings such that $\varphi_1\varphi_2 \Rightarrow \psi$, then there exists ψ_1, ψ_2 such that $\psi = \psi_1\psi_2$ and $\varphi_i \Rightarrow \psi_i$ $(i = 1, 2)$.*

Proof: This is clear from the nature of the rules of a CF grammar.

LEMMA 3. *If G is a CF grammar, then L_G is infinite if and only if there is some α for which $\alpha \Rightarrow \varphi\alpha\psi$ with at least one of φ, ψ not null.*

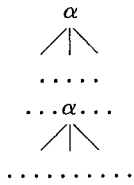
Note: We say α self-repeats in the case $\alpha \Rightarrow \varphi\alpha\psi$ with φ or ψ not null.

Proof: If $\alpha \Rightarrow \varphi\alpha\psi$, then $\alpha \Rightarrow \varphi\alpha\psi \Rightarrow \varphi\varphi\alpha\psi\psi = \varphi^2\alpha\psi^2$. Similarly

$$\alpha \Rightarrow \varphi^n\alpha\psi^n, \qquad n = 1, 2, 3, \cdots.$$

So there are an infinite number of derivations and an infinite number of sentences in L_G.

Conversely, if L_G is infinite, there must be arbitrarily long strings, and hence arbitrarily long derivations, trees, and branches of trees. Since there are only a finite number of symbols, there is a branch with two nodes similarly labeled, say by α. Consider this tree and branch and, in particular, the part of the tree dominated by the first α-node. This is



If we erase everything connected to the second α from below, we then have the tree of a derivation $\alpha \to \cdots \to \varphi\alpha\psi$.

Example of a language which is not CF:

$$L = \{ \divideontimes a^n b^n a^n \divideontimes \mid n = 1, 2, 3, \cdots \}.$$

L is not CF. If it is, let G be a CF grammar for L. By Lemma 3 we can

choose n so that some derivation from S to $a^nb^na^n$ contains a self-repetition, say $S \to \cdots \to \chi\alpha\omega \to \cdots \to \chi\varphi\alpha\psi\omega \to \cdots \to a^nb^na^n$, in which $\alpha \Rightarrow \varphi\alpha\psi$. By Lemma 2 there are (terminal) strings z, x, t, y, w such that $\chi \Rightarrow z$, $\varphi \Rightarrow x$, $\alpha \Rightarrow t$, $\psi \Rightarrow y$, $\omega \Rightarrow w$ and $zxtyw = a^nb^na^n$. But there is a derivation $S \to \cdots \to \chi\alpha\omega \to \cdots \to \chi\varphi^m\alpha\psi^m\omega \to \cdots \to zx^mty^mw$ for $m = 1, 2, 3, \cdots$. Therefore, $\divideontimes zx^mty^mw \divideontimes \in L$ for $m = 1, 2, 3, \cdots$. So all the strings zx^mty^mw are of the form $a^kb^ka^k$.

As $m \to \infty$ these strings become arbitrarily long, hence contain arbitrarily many b's. Since z, t, w are fixed, there must be a b in either x or y. Suppose b is in x. So $x = x'bx''$. Choosing m large enough, we have $zx^mty^mw = a^jb^ja^j$ for $j >$ length of zx'. But $zx^mty^mw = zx'bx'' \cdots$ and zx' cannot contain ja's, a contradiction. By symmetry the argument is essentially the same if we assume originally that there is a b in y. Therefore L is not a CF language.

Now we can show the main result:

THEOREM. *The set of context free languages is closed under the operation of set union, but not under difference nor intersection.*

Proof: Union—Lemma 1.

Intersection—Let $L_1 = \{ \divideontimes a^kb^ka^r \divideontimes \mid k, r = 1, 2, 3, \cdots\}$.
$L_2 = \{ \divideontimes a^mb^ja^j \divideontimes \mid m, j = 1, 2, 3, \cdots\}$.

Clearly $L_1 \cap L_2 = L$, the language of the above example.

Let G_1 be the CF grammar: $S \to CD$
$C \to aCb$, $C \to ab$ } this generates a^kb^k, $k = 1, 2, 3, \cdots$
$D \to aD$, $D \to a$ } this generates a^r, $r = 1, 2, 3, \cdots$

Clearly $L_1 = L_{G_1}$. By symmetry we can obtain a CF grammar G_2 for L_2.

Difference: This now follows by DeMorgan's law.
Let $U = \{ \divideontimes x \divideontimes \mid x$ any string of a's and b's$\}$. U is the finite state and hence CF. If the CF languages were closed under difference, then $U - L_1$ and $U - L_2$ would be CF, hence also $(U - L_1) \cup (U - L_2)$, and finally $L = L_1 \cap L_2 = U - [(U - L_1) \cup (U - L_2)]$ would be CF.

REFERENCES

CHOMSKY, N. (1959). On certain formal properties of grammars. *Information and Control* **2**, 137–167.

DAVIS, M. (1958). "Computability and Unsolvability," pp. 81–84. McGraw-Hill, New York.