BOUNDDED CONTEXT PARSABLE GRAMMARS

by

John H. Williams

Technical Report #58

March 1969
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>SECTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1. Definitions and Notation</td>
<td>3</td>
</tr>
<tr>
<td>2. Bounded Context Parsable Grammars</td>
<td>16</td>
</tr>
<tr>
<td>3. Relationship of Bounded Context Parsable Grammars with Other Subsets of Grammars</td>
<td>31</td>
</tr>
<tr>
<td>4. Testing for Bounded Context Parsability</td>
<td>41</td>
</tr>
<tr>
<td>Bibliography</td>
<td>46</td>
</tr>
</tbody>
</table>
INTRODUCTION

Context-free grammars have been found to be useful in the formal description of the syntax of programming languages [CHM], [FL3]. When designing a language and specifying its syntax, the designer would like to be able to know in advance that all the sentences in his language are unambiguous and that all the sentences in his language can be syntactically analysed, or parsed, efficiently. Consequently, much work has been done ([FL1], [KNU], [LYN], [ERL], [WW]) to discover subsets of the set of context-free grammars for which membership in the subset is recursively decidable and for which membership in the subset implies that sentences generated by the grammar can be parsed in time linearly proportional to the length of the sentence, i.e. in "real time" [HST]. The goal of these investigations has been to discover subsets that are sufficiently large and unrestricted so that the language designer may construct grammars that are in the subset without having to alter the desired constructs in his language or to introduce new syntactic types in his grammar in order to comply with the restrictions of the subset.

In this thesis we present a class of such subsets and call them the subsets of Bounded Context Parsable grammars. In section 1 we present the notation and previous work upon which the work of this thesis is built. In section 2 we give a formal definition of the Bounded Context Parsable
property and show that any sentence of a Bounded Context Parsable grammar can be parsed in real time. In section 3 we compare the sets of Bounded Context Parsable grammars to some other general sets of grammars, and we show that the set of Bounded Context Parsable languages properly contains the set of deterministic languages [GIN]. In section 4 we show that the problem to determine whether a grammar is in a particular member of the class of subsets of Bounded Context Parsable grammars is effectively decidable but that the problem to determine whether a grammar is in any one of the class is recursively undecidable.
SECTION 1. DEFINITIONS AND NOTATION

By a grammar $G$, we shall mean a context-free grammar; the class of context-free grammars has been extensively studied, [BPS], [GIN], [CHM], with varying descriptive notations. Given a finite set $X$ of characters, by $X^*$ we shall mean the set of all strings over $X$ including the empty string $\epsilon$, and by $X^+$ we shall mean the set $X^* - \{\epsilon\}$. The number of elements in $X$ will be denoted by $|X|$. We shall express a grammar as a 4-tuple, $G = (V, P, V_T, S)$ where:

i) $V$ is a finite set of symbols called the vocabulary of $G$.

ii) $V_T$ is a subset of $V$ called the terminal vocabulary of $G$.

(we call the complement of $V_T$ with respect to $V$ the non-terminal vocabulary of $G$ and denote it by $V_N$).

iii) $P$ is a finite set of pairs of strings over $V$ of the form $(A, x)$ where $A \in V_N$ and $x \in V^+$. $P$ is called the set of productions of $G$.

iv) $S \in V_N$ is called the sentence prototype of $G$.

We define the relation, $\rightarrow$, on $V^*$ by: $\varphi \rightarrow \psi$ iff

i) $\varphi = \chi_1 AX_2$

ii) $\psi = \chi_1 x \chi_2$

iii) $\chi_1, \chi_2 \in V^*$ and $(A, x) \in P$
We will denote the transitive completion of $\rightarrow$ by $\rightarrow^+$ and the reflexive and transitive completion of $\rightarrow$ by $\rightarrow^*$. For purposes of identification we will order the set $P$ and often write its $i$th member as $A_i \rightarrow x_i$. Such a string, $x_i$, will be called a phrase. A production of the form $A \rightarrow B$ where $B \in V_N$ will be called a 1-production. If $\varphi \rightarrow^+ \psi$, then $\varphi$ is said to derive $\psi$ and $\psi$ is said to be a $\varphi$-derivative. If $A \rightarrow x$ is a production, $x$ is said to be an immediate $A$-derivative.

If $\varphi \in V^*$, $\lambda(\varphi)$ will denote the length of $\varphi$, i.e., the number of characters in $\varphi$, and $\varphi^R$ will denote the reversal or mirror image of $\varphi$.

By the language of $G$ is meant the set $L(G) = \{ \varphi \mid S \rightarrow^+ \varphi \text{ and } \varphi \in V^*_T \}$. If $\varphi \in L(G)$, $\varphi$ is called a sentence of $G$.

By the sentential forms of $G$ is meant the set $SF(G) = \{ \varphi \mid S \rightarrow^+ \varphi \}$. Clearly $L(G) \subseteq SF(G)$.

A grammar is said to be reduced if for every $A \in V_N$,

i) $(\exists \varphi) (\exists \psi) S \rightarrow^* \varphi A \psi; \ \varphi, \psi \in V^*$

ii) $(\exists x) A \rightarrow^* x; \ x \in V^*_T$.

We will always assume a grammar to be reduced unless specifically stated otherwise.

A grammar is said to be linear if every phrase contains at most one nonterminal character; clearly if $G$ is linear, then every $\varphi \in SF(G)$ has at most one nonterminal character.
Given a grammar $G$ we define its description grammar by

$G' = (V', P', V'_T, S')$ where

i) $V'_T = V_T \cup \{ [ ] \cup \{ ] \} \mid 1 \leq i \leq o(P) \}$ where the $]$ and $[ \}$ are new symbols not in $V$.

ii) $V' = V \cup V'_T$

iii) $S' = S$

iv) $P' = \{ A_i \rightarrow [ x_i ] \mid A_i \rightarrow x_i \in P \}$.

The language $L(G')$ will be called the description language of $G$. We define the mapping, $m: V'^* \rightarrow V^*$ by

$m(A) = A$, if $A \in V_N$,

$m(a) = a$, if $a \in V_T$,

$m([ ) = \varepsilon$,

$m(\varepsilon) = \varepsilon$,

$m([ ] = \varepsilon$, $(1 \leq i \leq o(P))$,

and $m(\varphi) = m(x_1) \cdot m(x_2) \cdots m(x_n)$ if $\varphi = x_1 \cdot x_2 \cdots x_n$. Thus $m$ is a homomorphism with respect to concatenation. Let $\overline{m}$ be the restriction of $m$ to $SF(G')$. In general for $\varphi \in SF(G)$, $\overline{m}^{-1}(\varphi)$ will be a subset of $SF(G')$.

A grammar will be said to be unambiguous if $\overline{m}$ is $1:1$ and ambiguous otherwise.
If $A_1 \rightarrow x_1 \in P$, $\varphi \in SF(G)$, and $\varphi = \varphi_1 x_1 \varphi_2$, the phrase $x_1$ is said to be a handle of $\varphi$ if there is a string in $m^{-1}(\varphi)$ of the form $\psi_1 x_1 \psi_2$ such that $m(\psi_1) = \varphi_1$ and $m(\psi_2) = \varphi_2$.

To illustrate these concepts, consider the grammar

$$G_0 = (\{S, A, B, a, b\}, P_0, \{a, b\}, S)$$

with $P_0$:

- $S \rightarrow bAaB$
- $A \rightarrow aBA$
- $A \rightarrow ab$
- $B \rightarrow bA$
- $B \rightarrow b$

The phrases of the grammar are $bAaB$, $aBA$, $ab$, $bA$, and $b$.

The grammar is ambiguous because both

$$\varphi_1 = [b[a[b] [a b] ] a [ b ] ]$$

and

$$\varphi_2 = [b[a b] a [b [a b] ] ]$$

are mapped into bababab under $\bar{m}$.

Given a sentence $\varphi$, the process of computing $\bar{m}^{-1}(\varphi)$ is called parsing and a device for calculating $\bar{m}^{-1}(\varphi)$ is often called a parser for $G$. Parsers are usually constructed so that given an input, $\varphi$: 
i) if \( \varphi \in L(G) \), the parser outputs \( m^{-1}(\varphi) \)

ii) if \( \varphi \notin L(G) \), the parser outputs an error message.

The essence of the parsing process is to be able to decide which phrases of a sentential form are handles, make the appropriate reductions in the sentential form, and then repeat the process on the new sentential form thus obtained. Younger [YNG] has shown that for any grammar, \( G \), there exists a parser for \( G \) that will parse any sentence \( \varphi \in L(G) \) in a time proportional to \( (\lambda(\varphi))^3 \). Earley [ERL] has shown that for any unambiguous grammar \( G \), there is a parser that will parse any \( \varphi \in L(G) \) in time proportional to \( (\lambda(\varphi))^2 \). In computing we are interested in grammars whose sentences can be parsed in time proportional to \( \lambda(\varphi) \) or in "real time" [HST]. In particular we wish to discover large subsets \( C \) of context-free grammars that possess the following properties:

Property 1) membership in \( C \) is recursively decidable,

Property 2) there is an effective procedure for producing a real time parser for any \( G \) in \( C \).

One such important class of subsets is the class of Bounded Context \((m, n)\), \((BC (m, n))\), grammars defined by R. W. Floyd [FL1]. A grammar \( G \) is said to be \( BC (m, n) \) if in any \( \varphi \in SF(G) \) containing an occurrence of a phrase \( x_i \) it can be decided whether or not that \( x_i \) must be an \( A_i \)-derivative by examining only the \( m \) characters to the left of \( x_i \) in \( \varphi \) and
the $n$ characters to the right of $x_i$ in $\varphi$.

In order to insure that every phrase of every sentential form will have $m$ characters occurring to the left and $n$ characters occurring to the right we modify the definition slightly and speak of $m,n$-sentential forms which are the set of strings,

$$\{ \vdash^m \varphi \vdash^n | \varphi \in SF(G) \}$$

where $\vdash$ and $\vdash$ are two new characters not in the vocabulary of $G$ and are called the left and right end markers. There are numerous ways of dealing with this formally; for example, given a grammar $G = (V, P, V_T', S)$ and specific values for $m$ and $n$, say 2 and 3 respectively, we select a new symbol, $S_0$, not originally in $V$, add it to $V_N'$, add $\vdash$ and $\vdash$ to $V_T'$, add a 0th production, $S_0 \rightarrow \vdash \vdash S \vdash \vdash$, to $P$, and make $S_0$ the new sentence prototype of $G$. Now all the sentential forms of $G$ are $m,n$-sentential forms (with the exception of the string, $\vdash \vdash S \vdash \vdash \vdash$) and any occurrence of a phrase $x_i$ ($1 \leq i \leq \omega(P)$) in a sentential form of $G$ always has at least two characters to the left and three characters to the right of it. We do not wish to introduce a cumbersome notation for this type of trivial modification to a grammar; therefore, a grammar that has been so modified will simply be said to include end markers.

Floyd shows that for any given values for $m$ and $n$, say $m_0$ and $n_0$, the set $BC(m_0, n_0)$ satisfies properties 1 and 2. Given
any grammar with end markers $G$, the following is a decision procedure to determine if $G \in BC(m_0, n_0)$.

For every production $A_i \rightarrow x_i$ of $P$ and for every pair of strings $(w, y)$ such that $w, y \in V^*$, $\lambda(w) = m_0$, and $\lambda(y) = n_0$, check to see if

\[ S \rightarrow^* \ldots wA_i y \ldots \]

That $\ast$ is decidable has been shown by Bar-Hillel, Perles, and Shamir [BPS]. If $\ast$ is false then the phrase $x_i$ occurring in the context, $\ldots wx_i y \ldots$, can never be a handle of a sentential form of $G$. If $\ast$ is true, then if $G$ is to be $BC(m_0, n_0)$, every occurrence of the phrase, $x_i$, in the context, $\ldots wx_i y \ldots$, must be a handle and in particular an $A_i$-derivative. We express this formally as,

\[ \ast \ast \]

if $\varphi$ is of the form $vwx_iyz$ and $\delta \in m^{-1}(\varphi)$, then

$\delta$ is of the form $\psi_1 \psi_2 [x_i] \omega_1 \omega_2$ where $m(\psi_1) = v$, $m(\psi_2) = w$, $m(\omega_1) = y$, and $m(\omega_2) = z$.

That $\ast \ast$ is decidable for given $w, y$, and $i$, Floyd shows by the following analysis by cases.

If $\ast \ast$ is false, there is a sentential form $\varphi'$ in which the phrase $x_i$ occurs in context $(w, y)$ but $x_i$ is not an immediate $A_i$-derivative; i.e., the characters, $x_i$, have a bracket structure in $\delta' \in m^{-1}(\varphi')$ other than $[x_i]$. Consider the innermost pair of brackets in $\delta'$ that contain at least one character of $x_i$; if there is more than
one such bracket pair, choose the leftmost. Now let $s$ be the substring of characters of $\varphi'$ that are bracketed in $\delta'$ by the pair thus chosen, and let $B$ denote the element of $V_N$ associated with the right bracket; i.e., if the right bracket chosen above is $\bigcup_k A_k$, $B$ is the character, $A_k$, in the $k$th production $A_k \rightarrow x_k$. The string $s$ will be of one of the following sixteen forms depending on the position of its left and right ends within the string $\varphi' = \ldots w x_{11} y \ldots$:

$s_{11}: \ldots w x_{11} s_{12}: \ldots w x_1 s_{13}: \ldots w x_1 y_1 s_{14}: \ldots w x_1 y \ldots$

$s_{21}: w_2 x_{11} s_{22}: w_2 x_1 s_{23}: w_2 x_1 y_1 s_{24}: w_2 x_1 y \ldots$

$s_{31}: x_{11} s_{32}: x_1 s_{33}: x_1 y_1 s_{34}: x_1 y \ldots$

$s_{41}: x_{14} s_{42}: x_{12} s_{43}: x_{12} y_1 s_{44}: x_{12} y \ldots$

where,

$w_1, w_2, y_1, y_2, x_{11}, x_{12}, x_{13}, x_{14}, x_{15} \in V^+$,

$w_1 w_2 = w$,

$y_1 y_2 = y$,

and $x_{11} x_{12} = x_1 = x_{13} x_{14} x_{15}$.

Thus if (***) is false, one of the following sets of relations must be true for appropriate values of the string variables $v_1$ and $v_2$. 
R_{11}: S \rightarrow * \ldots Bx_{12}y \ldots, \; B \rightarrow v_{1}x_{11}, \; v_{1} \rightarrow * \ldots w

R_{12}: S \rightarrow * \ldots B y \ldots, \; B \rightarrow v_{1}x_{1}, \; v_{1} \rightarrow * \ldots w

R_{13}: S \rightarrow * \ldots B y_{2} \ldots, \; B \rightarrow v_{1}x_{1}, \; v_{1} \rightarrow * \ldots w, \; v_{2} \rightarrow * y_{1}

R_{14}: S \rightarrow * \ldots B \ldots, \; B \rightarrow v_{1}x_{1}, \; v_{1} \rightarrow * \ldots w, \; v_{2} \rightarrow * y \ldots

R_{21}: S \rightarrow * \ldots w_{1}Bx_{12}y \ldots, \; B \rightarrow v_{1}x_{11}, \; v_{1} \rightarrow * w_{2}

R_{22}: S \rightarrow * \ldots w_{1}B y \ldots, \; B \rightarrow v_{1}x_{1}, \; v_{1} \rightarrow * w_{2}

R_{23}: S \rightarrow * \ldots w_{1}B y_{2} \ldots, \; B \rightarrow v_{1}x_{1}, \; v_{1} \rightarrow * w_{2}, \; v_{2} \rightarrow * y_{1}

R_{24}: S \rightarrow * \ldots w_{1}B \ldots, \; B \rightarrow v_{1}x_{1}, \; v_{1} \rightarrow * w_{2}, \; v_{2} \rightarrow * y \ldots

R_{31}: S \rightarrow * \ldots wBx_{12}y \ldots, \; B \rightarrow x_{1}

R_{32}: S \rightarrow * \ldots wB y \ldots, \; B \rightarrow x_{1}, \; \text{and } A_{1} \neq B

R_{33}: S \rightarrow * \ldots wB y_{2} \ldots, \; B \rightarrow x_{1}v_{1}, \; v_{1} \rightarrow * y_{1}

R_{34}: S \rightarrow * \ldots wB \ldots, \; B \rightarrow x_{1}v_{1}, \; v_{1} \rightarrow * y \ldots

R_{41}: S \rightarrow * \ldots wx_{13}Bx_{15}y, \; B \rightarrow x_{14}

R_{42}: S \rightarrow * \ldots wx_{11}B y \ldots, \; B \rightarrow x_{12}

R_{43}: S \rightarrow * \ldots wx_{11}B y_{2}, \; B \rightarrow x_{12}v_{1}, \; v_{1} \rightarrow * y_{1}

R_{44}: S \rightarrow * \ldots wx_{11}B \ldots, \; B \rightarrow x_{12}v_{1}, \; v_{1} \rightarrow * y \ldots
Observe that the method for choosing the string $s$ did not require that $s$ be an immediate B-derivative but only that the portion of $x_1$ contained in $s$ be immediately derived from $B$. It is this consideration that makes necessary the string variables such as $v_1$ in relation $R_{12}$ for example.

For fixed $w$, $y$, and $i$, it can be decided if any one of these sixteen sets of relations is true by applying the decision procedures for the general questions, $\varphi \rightarrow^* \ldots \psi \ldots$, $\varphi \rightarrow^* \psi \ldots$, and $\varphi \rightarrow^* \ldots \psi$, presented in Bar-Hillel, Perles, and Shamir [BPS]. Therefore, given values $m_0$ and $n_0$ the set $BC(m_0, n_0)$ satisfies property 1. We will prove formally in the next section that the set $BC(m_0, n_0)$ satisfies property 2 as well.

In the same paper Floyd defines another class of subsets of grammars which he calls Bounded Right Context $(m, n)$ grammars. If in the above discussion we can assume that no handle lies to the left of the rightmost character of $x_1$ in the sentential form $\varphi' = \ldots wx_1 y \ldots$, then some of the above sixteen sets of relations can never be true and others can be simplified as follows:

Relations $R_{11}$, $R_{21}$, $R_{31}$, and $R_{41}$ cannot occur since they all assume the existence of a handle whose right end falls properly within the phrase $x_1$ and therefore violates the above condition that no handle
is to the left of the right end of \( x_i \). The following relations are simplified because now the characters of \( w \) that are contained in \( s \) must be immediately derived from \( B \) as well as the characters of \( x_i \) that are contained in \( s \), since there can be no handles to the left of \( x_i \) in \( \varphi' \).

\[
\begin{align*}
R'_{12}: & \quad S \rightarrow^* ...B... \rightarrow ...wx_i \\
R'_{13}: & \quad S \rightarrow^* ...B... \rightarrow ...wx_1v_1, \quad v_1 \rightarrow^* y_1 \\
R'_{14}: & \quad S \rightarrow^* ...B... \rightarrow ...wx_1v_1, \quad v_1 \rightarrow^* y... \\
R'_{22}: & \quad S \rightarrow^* ...w_1B... \rightarrow ...w_2x_1 \\
R'_{23}: & \quad S \rightarrow^* ...w_1B... \rightarrow ...w_2x_1v_1, \quad v_1 \rightarrow^* y_1 \\
R'_{24}: & \quad S \rightarrow^* ...w_1B... \rightarrow ...w_2x_1v_1, \quad v_1 \rightarrow^* y... \\
\end{align*}
\]

If for each \( i \) and for every pair \((w, y)\) such that \( \lambda(w) = m_0' \), \( \lambda(y) = n_0' \) and \( S \rightarrow^* ...wA_1y... \), it is the case that no one of the relations \( R'_{12}, R'_{13}, R'_{14}, R'_{22}, R'_{23}, R'_{24}, R'_{32}, R'_{33}, R'_{34}, R'_{42}, R'_{43} \) or \( R'_{44} \) is true, then the grammar is Bounded Right Context \((m_0', n_0')\). A parser that operates from left to right on its input string, thereby reducing all leftmost handles first, justifies the above assumption. Floyd defines the class of Bounded Left Context \((m, n)\) grammars in a similar fashion by removing relations \( R'_{41}, R'_{42}, R'_{43}, \) and \( R'_{44} \) simplifying the relations
\( R_{13}', R_{23}', R_{33}', R_{14}', R_{24}', \text{ and } R_{34}' \) and requiring the parser to operate from right to left. The above definitions do not exploit the full power afforded by a left to right parsing method as we shall exhibit in section 3.

One final very important class of subsets of grammars satisfying properties 1 and 2 is the class of grammars translatable from left to right with bound \( k \), (LR(\( k \))). As defined by D. E. Knuth [KNU]. Briefly, a grammar is said to be LR(\( k \)) if the leftmost handle of any sentential form of \( G \) is uniquely determined by the string to its left and the \( k \) characters to its right. The powerful feature of the definition of LR(\( k \)) grammars is that in parsing from left to right in a sentential form we are permitted to look arbitrarily far to the left to determine if a particular phrase is the leftmost handle. The remarkable thing about the LR(\( k \)) property is that Knuth is able to construct a parser that is a deterministic pushdown automaton [GIN]. Given a \( G \) and a \( k_0 \), Knuth has an effective procedure that will produce a real time parser for \( G \) if \( G \) is LR(\( k_0 \)) and will report failure otherwise. Therefore the class LR(\( k \)) satisfies properties 1 and 2. In the same paper Knuth shows that while the problem to determine of an arbitrary grammar \( G \) whether \( G \) is LR(\( k \)) is solvable for given \( k \), the problem to determine of an arbitrary grammar \( G \) whether there exists an integer \( k \) such that \( G \) is LR(\( k \)) is recursively unsolvable. Similarly he shows that the problem to determine of an arbitrary grammar \( G \) whether
there exist integers $m$ and $n$ such that $G$ is $BC(m,n)$ is recursively unsolvable. We will use this latter result in section 4.
SECTION 2. BOUNDED CONTEXT PARSABLE GRAMMARS

In this section we define the class of Bounded Context Parsable grammars and give an effective procedure for producing real time parsers for them.

Given a grammar $G = (V, P, V_T, S)$, a pair of strings $(w, y)$ will be called a derivation context for the production $A_1 \rightarrow x_1$ if:

i) $w, y \in V^*$, and

ii) $S \rightarrow^* \cdots w A_1 y \cdots$.

A pair of strings $(w, y)$ will be called a parsing context for the production $A_1 \rightarrow x_1$ if:

i) $(w, y)$ is a derivation context for $A_1 \rightarrow x_1$, and

ii) if $\varphi \in SF(G)$ is of the form $\varphi = vw x_1 y z$ for some $v, z \in V^*$ and $\delta \in m^{-1}(\varphi)$, then $\delta$ is of the form $\psi_1 \psi_2 [x_1] \omega_1 \omega_2$

where $m(\psi_1) = v$, $m(\psi_2) = w$, $m(\omega_1) = y$, and $m(\omega_2) = z$.

In other words, $(w, y)$ is a parsing context for the production $A_1 \rightarrow x_1$ if $x_1$ is always an immediate $A_1$-derivative whenever it appears in a sentential form $\varphi$ in the context $(w, y)$; i.e., $\varphi = \cdots wx_1 y \cdots$.

Observe that if $(w, y)$ is a parsing context for a production and $(vw, yz)$ is a derivation context for that production, then $(vw, yz)$
must also be a parsing context for that production.

A parsing context \((w, y)\) will be said to be of order \([m, n]\) if \(\lambda(w) = m\) and \(\lambda(y) = n\). A parsing context of order \([m_1, n_1]\) will be said to have order less than a parsing context of order \([m_2, n_2]\) if \(m_1 \leq m_2\) and \(n_1 \leq n_2\). The set of derivation contexts for the \(i\)th production of order \([m, n]\) or less will be denoted by \(DC_i[m, n]\). The set of parsing contexts for the \(i\)th production of order \([m, n]\) or less will be denoted by \(PC_i[m, n]\) and its \(j\)th member will be denoted by \((w_{ij}, y_{ij})\).

A particular occurrence of a phrase \(x_i\) in a sentential form \(\varphi\) will be said to occur in a parsing context of order \([m, n]\) in \(\varphi\) if the pair of strings \((\ell, r)\) is in \(PC_i[m, n]\) when \(\ell\) is the string consisting of the \(m\) characters immediately to the left of that particular occurrence of \(x_i\) in \(\varphi\) and \(r\) is the string consisting of the \(n\) characters immediately to the right of that particular occurrence of \(x_i\) in \(\varphi\).

A grammar \(G\) will be said to be Bounded Context Parsable with left bound \(m\) and right bound \(n\), \((BCP[m,n])\), if every sentential form of \(G\) contains at least one phrase occurring in a parsing context of order \([m, n]\) or less.

To illustrate these definitions consider the grammar,

\[G_1 = \{ [S, A, B, E, a, b, e, \epsilon, \rightarrow], P_1, \{ a, b, e, \epsilon, \rightarrow \}, S \} \text{ with } P_1 : \]
$S \rightarrow \vdash AEa \vdash$

$S \rightarrow \vdash BEb \vdash$

$E \rightarrow eE$

$E \rightarrow e$

$A \rightarrow e$

$B \rightarrow e$

For each of the six productions we list the set of derivation contexts of order $[2, 2]$ or less and the set of parsing contexts of order $[2, 2]$ or less. As a notational convenience, $(\{s_1, s_2, s_3\}, \{s_4, s_5\})$ will be used to stand for the 6 contexts, $(s_1, s_4), (s_2, s_4), (s_3, s_4), (s_1, s_5), (s_2, s_5),$ and $(s_3, s_5)$.

1) $S \rightarrow \vdash AEa \vdash$

$DC_1[2, 2] : (\epsilon, \epsilon)$

$PC_1[2, 2] : (\epsilon, \epsilon)$

2) $S \rightarrow \vdash BEb \vdash$

$DC_2[2, 2] : (\epsilon, \epsilon)$

$PC_2[2, 2] : (\epsilon, \epsilon)$

3) $E \rightarrow eE$

$DC_3[2, 2] : (\{\epsilon, e, ee, \vdash e\}, \{\epsilon, a, b, a \vdash, b \vdash\})$

$(\{A, \vdash A, Ae\}, \{\epsilon, a, a \vdash\})$

$(\{B, \vdash B, Be\}, \{\epsilon, b, b \vdash\})$
\[ PC_3[2,2] : ( \{ e, ee, \vdash e \}, \{ e, a, b, a \vdash, b \vdash \} ) \]

\[ ( \{ A, \vdash A, Ae \}, \{ e, a \vdash \} ) \]

\[ ( \{ B, \vdash B, Be \}, \{ e, b \vdash \} ) \]

4) \[ E \rightarrow e \]

\[ DC_4[2,2] : \text{ (the same as } DC_3[2,2] \text{ by the definition of derivation context) } \]

\[ PC_4[2,2] : ( \{ e, e, \vdash e, ee \}, \{ a, b, a \vdash, b \vdash \} ) \]

\[ ( \{ A, \vdash A, Ae \}, \{ a, a \vdash \} ) \]

\[ ( \{ B, \vdash B, Be \}, \{ b, b \vdash \} ) \]

5) \[ A \rightarrow e \]

\[ DC_5[2,2] : ( \{ e, \vdash \}, \{ e, E, e, Ea, eE, ea, ee \} ) \]

\[ PC_5[2,2] : ( \vdash Ea), (\vdash ea) \]

6) \[ B \rightarrow e \]

\[ DC_6[2,2] : ( \{ e, \vdash \}, \{ e, E, e, Eb, eE, eb, ee \} ) \]

\[ PC_6[2,2] : (\vdash Eb), (\vdash eb) \]

Notice that:

i) \( (e, e) \) is a parsing context for production 1; that is, whenever \( \vdash AEAe \vdash \) occurs it can be replaced by \( S \) "regardless of its context."
ii) \((e,e)\) is a derivation context for production 3, \textit{i.e.} \(S \rightarrow^* \ldots E \ldots\), but \((e,e)\) is not a parsing context for 3; \textit{e.g.} in the sentential form, \(\vdash eEe\), the underlined occurrence of \(eE\) is not an \(E\)-derivative.

iii) \((e,e)\) is a parsing context for 3; \textit{i.e.} in any sentential form, \(\ldots eeE \ldots\), the underlined occurrence of \(eE\) is an immediate \(E\)-derivative.

iv) \((\vdash , eE)\) is a derivation context for both productions 5 and 6, but it is not a parsing context for either; \textit{i.e.} while \(\vdash B eE \ldots\) and \(\vdash A eE \ldots\) may occur in a sentential form, the underlined occurrence of \(e\) in \(\vdash eeeE \ldots\) may be either an immediate \(A\)-derivative or an immediate \(B\)-derivative and the context \((\vdash , eE)\) is not sufficient to dictate which. Note that \(e\) is also the phrase of production 4, but \((\vdash , eE)\) is not even a derivation context for production 4 since \(\vdash E eE\) can never occur in a sentential form.

Now any sentential form of \(G_1\) must be of one of the following twelve forms:
1. \( \vdash A\varepsilon a \downarrow \)
2. \( \vdash A\varepsilon^n a \downarrow \), \( n \geq 1 \)
3. \( \vdash A\varepsilon^n a \downarrow \), \( n \geq 1 \)
4. \( \vdash e\varepsilon a \downarrow \)
5. \( \vdash e\varepsilon^n a \downarrow \), \( n \geq 2 \)
6. \( \vdash e\varepsilon^n a \downarrow \), \( n \geq 2 \)
7. \( \vdash B\varepsilon b \downarrow \)
8. \( \vdash B\varepsilon^n b \downarrow \), \( n \geq 1 \)
9. \( \vdash B\varepsilon^n b \downarrow \), \( n \geq 1 \)
10. \( \vdash e\varepsilon b \downarrow \)
11. \( \vdash e\varepsilon^n b \downarrow \), \( n \geq 2 \)
12. \( \vdash e\varepsilon^n b \downarrow \), \( n \geq 2 \)

That each of these forms contains at least one phrase occurring in a parsing context of order \([2,2]\) or less may be seen by analyzing the possible cases:

1.) Form 1 contains the phrase, \( \vdash A\varepsilon a \downarrow \) occurring in context \((\varepsilon, \varepsilon)\) which is a parsing context for production 1, \( S \rightarrow \vdash A\varepsilon a \downarrow \).

2.) Form 2 contains an occurrence of \( eE \) in context \((\vdash A, a)\) if \( n = 1 \) or \((\varepsilon, a)\) if \( n > 1 \) and both of these are parsing contexts for production 3, \( E \rightarrow eE \).
3.) Forms 3 and 6 contain an occurrence of $e$ in context $(e,a)$ which is a parsing context for production $4$, $E \rightarrow e$.

4.) Form 4 contains an occurrence of $e$ in the context $(\vdash, Ea)$ which is a parsing context for production $5$, $A \rightarrow e$.

5.) Form 5 contains an occurrence of $eE$ in the context $(e,e)$ which is a parsing context for production $3$, $E \rightarrow eE$.

6.) Similarly for forms 7-12.

Therefore the grammar $G_1$ is $BCP[2,2]$ since all of the parsing contexts used in the above analysis are of order $[2,2]$ or less. If at case 2) in the analysis we use the parsing context $(A,a)$ instead of $(\vdash A,a)$ for $n = 1$, then it is seen that $G_1$ is $BCP[1,2]$ since all contexts used in the analysis would then be of order $[1,2]$ or less.

It is natural to ask if there is some analysis that will show $G_1$ to be $BCP[1,1]$. Since there is no parsing context for the production $A \rightarrow e$ of order less than $[1,2]$, if there is any sentential form of $G_1$ in which the only handle is the phrase $e$ occurring as an immediate $A$-derivative, $G_1$ cannot be $BCP[1,1]$; $\vdash eEa \vdash$ is such a sentential form.

If a grammar is $BCP[m,n]$, then since every sentential form has a handle occurring in parsing context, the grammar is clearly unambiguous by induction on the number of steps in the derivation of a sentential form.
In the remainder of this section we will show that the class of Bounded Context Parsable grammars satisfies Property 2. Let 

\[ G = (V, P, V_T, S) \] be a BCP\([m, n]\) grammar, \( p = o(P) \), \( PC_i[m, n] \) be the set of \( k_i \) parsing contexts for the \( i^{th} \) production of \( P \) \((1 \leq i \leq p)\), and \( \phi \in L(G) \). We can construct the parse of \( \phi \) by the following procedure:

i) set \( \psi_1 = \varnothing \phi \varnothing \) and \( j = 1 \)

ii) search \( \psi_j \) for a phrase \( x_i \) occurring in parsing context and consider the string formed by replacing that occurrence of \( x_i \) by \( A_i \); set \( \psi_{j+1} \) equal to the string thus formed.

iii) Set \( j = j + 1 \)

iv) If \( \psi_j = S \), stop; otherwise, go back to step ii).

Step ii) will always be possible since every sentential form, and therefore every \( \psi_j \), will contain a phrase occurring in a parsing context of order \([m, n]\) or less and for each of the productions in \( P \) there are only finitely many such parsing contexts for which we have to look.

We can describe this procedure as a kind of Floyd reduction system \([FL2]\) containing two classes of reduction rules, I and II, where the reduction process always attempts to reduce by a rule of type I before attempting to reduce by a rule of type II. Class I consists of the rules:
\[
\begin{align*}
& w_{11}x_iy_{1l}\Delta \rightarrow w_{11}A_{11}y_{1l}\Delta \\
& w_{12}x_iy_{12}\Delta \rightarrow w_{12}A_{12}y_{12}\Delta \\
& \vdots \\
& w_{1k_1}x_iy_{1k_1}\Delta \rightarrow w_{1k_1}A_{11}y_{1k_1}\Delta \\
& w_{21}x_2y_{21}\Delta \rightarrow w_{21}A_{21}y_{21}\Delta \\
& w_{22}x_2y_{22}\Delta \rightarrow w_{22}A_{22}y_{22}\Delta \\
& \vdots \\
& w_{2k_2}x_2y_{2k_2}\Delta \rightarrow w_{2k_2}A_{21}y_{2k_2}\Delta \\
& w_{p1}x_{p1}\Delta \rightarrow w_{p1}A_{p1}y_{p1}\Delta \\
& \vdots \\
& w_{pk_p}x_{p1}\Delta \rightarrow w_{pk_p}A_{p1}y_{pk_p}\Delta 
\end{align*}
\]

I.e., class I consists of the \( \sum_{i=1}^{p} k_i \) rules,

\[
\begin{align*}
& w_{ij}x_iy_{ij}\Delta \rightarrow w_{ij}A_{i}y_{ij}\Delta, \quad l \leq j \leq k_i, \quad l \leq i \leq p.
\end{align*}
\]

Class II consists of the \( o(V) \) rules,

\[
\Delta v_i \rightarrow v_i\Delta, \quad \text{for all } v_i \in V.
\]

The reduction process operates as described by Floyd for the rules of class II. However, after application of a class I rule, the scanning marker \( \Delta \) is moved back to the beginning of the string before the next
scan for a reduction rule. This is because step ii) in the procedure searches each string from the beginning and not from the point of the last reduction. Using this reduction system a large amount of time may be wasted in scanning each sentential form for a phrase occurring in parsing context. For example, with the grammar \( \{ \{ S, a \}, \{ S \rightarrow aS, S \rightarrow a \} \}, \{ a \}, S \) since the only handle in any sentential form is at the right hand end, the entire string must be scanned each time and the number of reduction rules applied will be about \( \frac{\lambda(\psi)^2}{2} \).

The parsing method can be improved by observing that it is not necessary to start scanning at the beginning of \( \psi_j \) in step ii) for \( j > 1 \). If the \( A_i \) introduced in the application of step ii) to \( \psi_{j-1} \) is the \( q \)th character in \( \psi_j \), then no phrase in \( \psi_j \) whose right-most character is to the left of the \((q-n)\)th character of \( \psi_j \) can occur in a parsing context since it would have been discovered and reduced by an earlier application of step ii). Therefore we may resume scanning \( n \) characters to the left of the most recently introduced symbol in \( \psi_j \) rather than going back to the beginning. We can implement this modification in the reduction system by removing the special treatment of type I reductions, i.e., by not moving the marker \( \Delta \) back to the beginning of the string, and by rewriting each type I rule in the form:

\[ (I') \quad w_{ij} x y_{ij} \Delta \rightarrow w_{ij} A_{i} y_{ij}, \quad 1 \leq j \leq k_i, \quad 1 \leq i \leq p. \]
With this improvement the reduction system will parse in a time linearly proportional to the length of the sentence $\varphi$ since the number of reductions applied with the modified reduction system is bounded above by a linear function of $\varphi$ as is demonstrated in the following:

**Theorem:** If $G = (V, P, V_T, S)$ is BCP[$m, n$], there exists an integer $K$ such that the number of reductions used by the modified reduction system described above in parsing any $\varphi \in L(G)$ is $K \cdot \lambda(\varphi)$ or less.

**Proof:**

1) Let $p = o(P)$.

2) Since $G$ is BCP, it is unambiguous and no infinite cycling of 1-productions can occur in the derivation of a sentence of length $\ell$. Therefore the number of applications of productions in $P$ used in deriving a sentence of length $\ell$ can be at most $2p \cdot \ell$ since at least one terminal character or one additional non-terminal character must be produced after the application of every $2p$ rewriting rules.

3) Therefore from 2), the number of applications of reduction rules of type $I'$ can be at most $2p\ell$ for a sentence of length $\ell$. 

4) The number of applications of type II rules can be at most \( \ell \) (to scan the entire sentence) plus the number of characters that must be rescanned due to moving the pointer back in applications of type I' rules. Since the pointer backs up at most \( n \) characters at each application of a type I' rule and the number of such applications is at most \( 2p\ell \) by 2) above, the number of applications of reductions of type II can be at most \( \ell + 2p\ell n \).

5) Therefore the total number of rules applied in reducing a sentence of length \( \ell \) will be at most \( 2p\ell + \ell + 2p\ell n \).

6) Thus \( K = 2p(n+1) + 1 \), and since \( p \) and \( m \) are constants of the grammar independent of \( \ell \), the theorem is proved.

Q.E.D.

To illustrate the parsing method we again consider the grammar \( G_1 \) defined earlier in this section. We have shown that \( G_1 \) is BCP\([2,2]\). We will construct a reduction system for \( G_1 \) and use it to parse the sentence, \( \cdot e e e e e e a \cdot \). Whereas each of the productions may have a large number of parsing contexts of order \([2,2]\) or less, it often will not be necessary to include them all in the reduction system constructed
for parsing; indeed, if \((w, y)\) and \((vw, yz)\) are both parsing contexts for production \(i\), it will not be necessary to include the reduction rule \(vw_{1}yz\Delta \rightarrow vwA_{1}\Delta yz\) since the phrases reduced by the rule \(wx_{1}y\Delta \rightarrow wA_{1}\Delta y\) will include all those of the former. For example in \(G_{1}\) the production \(E \rightarrow eE\) has thirty-three parsing contexts of order \([2, 2]\) or less but only three, \((e, e)\), \((A, e)\), and \((B, e)\), need to be included in the reduction system. While the number of rules in the system \(R_{1}\) has been greatly reduced by eliminating unnecessary parsing contexts, there are still some redundant rules in the system. We see that rules \(17, 18,\) and \(20\) are useless since \(A, B,\) and \(S\) are nonterminals and are never introduced to the right of \(\Delta\) in any reduction. Therefore these rules can never be applicable to any sentential form. A more subtle redundancy occurs in rules \(9\) and \(11\). Rule \(9\) is in the system because \((\_, ea)\) is a parsing context for the production \(A \rightarrow e\). However, any sentential form containing the phrase \(e\) in that context must also contain an occurrence of \(e\) in the context \((e, a)\) which is a parsing context for \(E \rightarrow e\). If the latter reduction is made first, i.e., rule \(6\) is applied first, then rule \(9\) will never be applicable, since the phrase \(e\) in question will now occur in the context \((\_, Ea)\) and be reduced by rule \(8\). Similarly rules \(7\) and \(10\) make rule \(11\) redundant.
The following twenty rule reduction system, \( R_1 \), will parse sentences in \( L(G_1) \):

<table>
<thead>
<tr>
<th></th>
<th>( \Delta A E a \rightarrow \Delta )</th>
<th>( \rightarrow )</th>
<th>( S \Delta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>( \Delta B E b \rightarrow \Delta )</td>
<td>( \rightarrow )</td>
<td>( S \Delta )</td>
</tr>
<tr>
<td>3</td>
<td>( e e E \Delta )</td>
<td>( \rightarrow )</td>
<td>( e E \Delta )</td>
</tr>
<tr>
<td>4</td>
<td>( A e E \Delta )</td>
<td>( \rightarrow )</td>
<td>( A E \Delta )</td>
</tr>
<tr>
<td>5</td>
<td>( B e E \Delta )</td>
<td>( \rightarrow )</td>
<td>( B E \Delta )</td>
</tr>
<tr>
<td>6</td>
<td>( e a \Delta )</td>
<td>( \rightarrow )</td>
<td>( E \Delta a )</td>
</tr>
<tr>
<td>7</td>
<td>( e b \Delta )</td>
<td>( \rightarrow )</td>
<td>( E \Delta b )</td>
</tr>
<tr>
<td>8</td>
<td>( \Delta A E a )</td>
<td>( \rightarrow )</td>
<td>( \Delta A \Delta E a )</td>
</tr>
<tr>
<td>9</td>
<td>( \Delta e e a )</td>
<td>( \rightarrow )</td>
<td>( \Delta A \Delta e a )</td>
</tr>
<tr>
<td>10</td>
<td>( \Delta e E b )</td>
<td>( \rightarrow )</td>
<td>( \Delta B \Delta E b )</td>
</tr>
<tr>
<td>11</td>
<td>( \Delta e e b )</td>
<td>( \rightarrow )</td>
<td>( \Delta B \Delta e b )</td>
</tr>
<tr>
<td>12</td>
<td>( \Delta a )</td>
<td>( \rightarrow )</td>
<td>( a \Delta )</td>
</tr>
<tr>
<td>13</td>
<td>( \Delta b )</td>
<td>( \rightarrow )</td>
<td>( b \Delta )</td>
</tr>
<tr>
<td>14</td>
<td>( \Delta e )</td>
<td>( \rightarrow )</td>
<td>( e \Delta )</td>
</tr>
<tr>
<td>15</td>
<td>( \Delta \rightarrow )</td>
<td>( \rightarrow )</td>
<td>( \Delta \rightarrow )</td>
</tr>
<tr>
<td>16</td>
<td>( \Delta \rightarrow )</td>
<td>( \rightarrow )</td>
<td>( \Delta \rightarrow )</td>
</tr>
<tr>
<td>17</td>
<td>( \Delta A )</td>
<td>( \rightarrow )</td>
<td>( A \Delta )</td>
</tr>
<tr>
<td>18</td>
<td>( \Delta B )</td>
<td>( \rightarrow )</td>
<td>( B \Delta )</td>
</tr>
<tr>
<td>19</td>
<td>( \Delta E )</td>
<td>( \rightarrow )</td>
<td>( E \Delta )</td>
</tr>
<tr>
<td>20</td>
<td>( \Delta S )</td>
<td>( \rightarrow )</td>
<td>( S \Delta )</td>
</tr>
</tbody>
</table>
Using the reduction system $R_1$, the sentence, $\vdash \text{eeeeea} \downarrow$, would be parsed as follows:

<table>
<thead>
<tr>
<th>Sentential form</th>
<th>First rule applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta \vdash \text{eeeeea} \downarrow$</td>
<td>15</td>
</tr>
<tr>
<td>$\vdash \Delta \text{eeeeea} \downarrow$</td>
<td>14</td>
</tr>
<tr>
<td>$\vdash e \Delta \text{eeeeea} \downarrow$</td>
<td>14</td>
</tr>
<tr>
<td>$\vdash e e \Delta \text{eeeeea} \downarrow$</td>
<td>14</td>
</tr>
<tr>
<td>$\vdash e e e \Delta \text{eeeeea} \downarrow$</td>
<td>14</td>
</tr>
<tr>
<td>$\vdash e e e e \Delta \text{eeea} \downarrow$</td>
<td>14</td>
</tr>
<tr>
<td>$\vdash e e e e e \text{eeea} \downarrow$</td>
<td>14</td>
</tr>
<tr>
<td>$\vdash e e e e e e \text{eeea} \downarrow$</td>
<td>14</td>
</tr>
<tr>
<td>$\vdash e e e e e e e \Delta \text{a} \downarrow$</td>
<td>12</td>
</tr>
<tr>
<td>$\vdash e e e e e e e e \text{a} \Delta \downarrow$</td>
<td>6</td>
</tr>
<tr>
<td>$\vdash e e e e e E \Delta \text{a} \downarrow$</td>
<td>3</td>
</tr>
<tr>
<td>$\vdash e e e e E \Delta a \downarrow$</td>
<td>3</td>
</tr>
<tr>
<td>$\vdash e e E \Delta a \downarrow$</td>
<td>8</td>
</tr>
<tr>
<td>$\vdash e E a \Delta \downarrow$</td>
<td>8</td>
</tr>
<tr>
<td>$\vdash A \Delta \text{a} \downarrow$</td>
<td>19</td>
</tr>
<tr>
<td>$\vdash A \Delta \text{Ea} \downarrow$</td>
<td>12</td>
</tr>
<tr>
<td>$\vdash A \Delta \text{Ea} \downarrow$</td>
<td>12</td>
</tr>
<tr>
<td>$\vdash A \text{Ea} \Delta \downarrow$</td>
<td>16</td>
</tr>
<tr>
<td>$\vdash A \text{Ea} \downarrow \Delta$</td>
<td>1</td>
</tr>
<tr>
<td>$S \Delta$</td>
<td>done</td>
</tr>
</tbody>
</table>
SECTION 3. RELATIONSHIP OF BOUNDED CONTEXT PARSABLE GRAMMARS WITH OTHER SUBSETS OF GRAMMARS

In this section we will compare the set of $BCP[m, n]$ grammars with the sets of grammars that are $BC(m, n)$, $BRC(m, n)$, $BLC(m, n)$, $LR(n)$, or $RL(m)$. We will also show that the set of languages that are $BCP[m, n]$ properly contains the set of deterministic languages defined by Ginsburg and Greibach [GIN].

We will let $S_{BCP[m, n]}$ denote the set of $BCP[m, n]$ grammars. The definition of Bounded Context Parsable induces the partial ordering on the sets $S_{BCP[n, m]}$ defined by:

$$S_{BCP[m_1, n_1]} \supseteq S_{BCP[m_2, n_2]} \quad \text{if} \quad m_1 \geq m_2 \quad \text{and} \quad n_1 \geq n_2 .$$

We will let $S_{BCP}$ denote the set, $\bigcup_{m, n \geq 0} S_{BCP[m, n]}$: i.e., a grammar $G$ is in $S_{BCP}$ if $\exists m \exists n$ such that $G$ is $BCP[m, n]$. It is interesting to compare the sets of Bounded Context Parsable grammars with other sets of grammars that satisfy properties 1 and 2 as given in Section 1 above. We present the results of these comparisons as a Venn diagram in figure 1 on page 37.

$$S_{BCP} \not\supset S_{BC} .$$

If a grammar is Bounded Context $(m, n)$, then every derivation context of order $[m, n]$ must be a parsing context by the definition of Bounded
Context grammars. Since every sentential form has some phrase occurring as a handle and therefore occurring in a derivation context of order \([m, n]\), every sentential form has a phrase occurring in a parsing context of order \([m, n]\) or less; therefore the grammar must be \(BCP[m, n]\). That the inclusion is proper is demonstrated by the existence of \(G_1\) which we have shown is \(BCP[1, 2]\). \(G_1\) is not \(BC(1, 2)\) nor is it \(BC(m, n)\) for any \(m\) and \(n\). This is seen in that \((\vdash^m, e^n)\) is a derivation context for the production \(A \rightarrow e\), i.e. \(S \rightarrow^* \cdots \vdash^m A e^n \cdots\) (recall our implicit assumption about having the requisite number of end markers present to allow all phrases to occur in a context of order \([m, n]\).) But the context \((\vdash^m, e^n)\) is not a parsing context for \(A \rightarrow e\) since there are sentential forms in which the occurrence of \(e\) in the context \((\vdash^m, e^n)\) is not an \(A\)-derivative, e.g. \(\vdash^m e e^n b \downarrow\). Therefore \(G_1\) is not in \(S_{BC(m, n)}\) for any \(m\) or \(n\); i.e. \(G_1 \notin S_{BC}\).

Whereas we have used \(S_{BC(m, n)}\) to denote the set of Bounded Context \((m, n)\) grammars exactly as defined by Floyd, we will modify his definitions of bounded right context and bounded left context grammars somewhat. As we mentioned in Section 1, his definitions do not fully exploit the power of left to right or right to left syntactic analysis. To demonstrate this, consider \(G_1^R\) the reversal of the grammar \(G_1\). \(G_1^R\) has productions:
Now by Floyd's definition, $G^R_1$ is not Bounded Right Context $(2, 1)$; this is because $S \rightarrow^* \cdots e e A \downarrow$, but the set of relations $R_{32}$ is true, i.e. $S \rightarrow^* \cdots e e B \downarrow$, $B \rightarrow e$, and $B \neq A$. That is, one of the derivation contexts for $A \rightarrow e$, $(e e, \downarrow)$, is not a parsing context even under the modified tests for parsing context. But a parser operating from left to right would never need to consider the context $(e e, \downarrow)$ for the phrase $e$. By the time it reached an $e$ occurring next to the right end marker, it would be working on either the sentential form $\vdash a E e \downarrow$ or the sentential form $\vdash b E e \downarrow$, and both of these derivation contexts for the phrase $e$, $(a E, \downarrow)$ and $(b E, \downarrow)$, are parsing contexts, the former for the production $A \rightarrow e$ and the latter for the production $B \rightarrow e$. Therefore, for a parser operating from left to right, the only derivation contexts that need to be parsing contexts are those that occur leftmost in a sentential form. That is, only those derivation contexts for a production whose phrase occurs as the leftmost handle of a sentential form need to be parsing contexts for that production.
We define the set of left-restricted derivation contexts of a production as follows:

The pair \((w, y)\) will be said to be a left-restricted derivation context of order \([m, n]\) for the \(i\)th production \(A_i \rightarrow x_i\), \((\text{LDC}^*_i[m, n])\), iff

1) \(\lambda(w) = m\)

2) \(\lambda(y) = n\)

3) \(\exists v \exists z\) such that
   i) \(v, z \in V^*\)
   ii) \(S \rightarrow ^* v w A_i y z\)
   iii) in the string \(\delta = m^{-1}(v w A_i y z) = \varphi_1 A_i \varphi_2\) where

\(m(\varphi_1) = v w\) and \(m(\varphi_2) = y z\), there are no right
brackets in \(\varphi_1\).

The set of right-restricted derivation contexts is defined similarly.

Clearly, \(\text{LDC}^*_i[m, n] \subseteq D_i[m, n]\) and \(\text{RDC}^*_i[m, n] \subseteq D_i[m, n]\).

We now define our modified versions of bounded right and left context grammars. A grammar \(G\) will be said to be \(\text{BRC}(m, n)\) if for all productions \(A_i \rightarrow x_i\), \((w, y) \in \text{LDC}^*_i[m, n]\) implies that \((w, y) \in \text{PC}^*_i[m, n]\); i.e., \(G\) is \(\text{BRC}(m, n)\) if every left-restricted derivation context for the \(i\)th production is a parsing context for the \(i\)th production. Similarly, a grammar will be said to be \(\text{BLC}(m, n)\) if for all productions \(A_i \rightarrow x_i\), \((w, y) \in \text{RDC}^*_i[m, n]\) implies \((w, y) \in \text{PC}^*_i[m, n]\).
With this definition, \( G^R_1 \) is seen to be \( \text{BRC}(2, 1) \).

Now if a grammar \( G \) is \( \text{BRC}(m, n) \), then since every sentential form has a phrase occurring in a left-restricted derivation context of order \([m, n]\), in particular the leftmost handle of the sentential form, every sentential form has a phrase occurring in a parsing context of order \([m, n]\) and thus \( G \) is \( \text{BCP}(m, n) \). Thus we have the following inclusion relationship:

\[
S_{\text{BCP}} \supsetneq S_{\text{BRC}}
\]

To see that this inclusion is proper, recall the grammar \( G_1 \). \( G_1 \) is \( \text{BCP}(1, 2) \), but it is not \( \text{BRC}(m, n) \) for any \( m \) and \( n \); \((I^m, e^n)\) is a left-restricted derivation context for \( A \rightarrow e \), but it is not a parsing context for \( A \rightarrow e \) as was shown above. Similarly we have:

\[
S_{\text{BCP}} \supsetneq S_{\text{BLC}}
\]

This inclusion is proper since \( G^R_1 \) is \( \text{BCP}(2, 1) \) but not \( \text{BLC}(m, n) \) for any \( m \) and \( n \). \( G^R_1 \) is \( \text{BCP}(2, 1) \) since \( G_1 \) is \( \text{BCP}(1, 2) \). \( G^R_1 \) is not \( \text{BLC}(n, m) \) for any \( m \) and \( n \) since \( G_1 \) is not \( \text{BRC}(m, n) \) for any \( m \) and \( n \).

For an example of a grammar that is \( \text{BCP}(2, 2) \) but is neither \( \text{BRC}(m, n) \) nor \( \text{BLC}(m, n) \) for any values of \( m \) and \( n \), consider the grammar \( G_2 \) with productions:
\[
S \rightarrow \epsilon A E a E A \epsilon \\
S \rightarrow \epsilon B E b E B \epsilon \\
E \rightarrow e E \\
E \rightarrow e \\
A \rightarrow e \\
B \rightarrow e
\]

When we compare \( S_{BCP} \) with the set of grammars that are LR(\( n \)) for some \( n \), \( (S_{LR}) \), it is seen that neither set contains the other. \( G_1 \) is not LR(\( n \)) for any value of \( n \) since the leftmost handle of the sentential form, \( \epsilon \epsilon \epsilon ^{n} a \epsilon \epsilon \), cannot be parsed by looking only at the characters to the left, \( \epsilon \epsilon \), and the \( n \) characters to the right, \( \epsilon ^{n} \). The unbounded left context available to the LR analysis does no good in this particular situation; the information needed to determine how to parse the first \( \epsilon \) lies arbitrarily far to the right. As above, \( G_1 \) is RL(1), but \( G_2 \) is neither LR(\( n \)) nor RL(\( n \)) for any value of \( n \). Knuth has shown [KNU] that the grammar \( G_3 \) with productions:

\[
\begin{align*}
S & \rightarrow \epsilon T \epsilon \\
T & \rightarrow a U c \\
T & \rightarrow b \\
U & \rightarrow a T c \\
U & \rightarrow b
\end{align*}
\]
is LR(0). \( L(G_3) = \{ a^k b c^k \mid k \geq 0 \} \), and the \( b \) must be reduced to
T or U according as \( k \) is even or odd. For any values of \( m \) and \( n \),
\( (a^m, c^n) \) is a derivation context for both of the productions \( T \to b \) and
\( U \to b \), but it is a parsing context for neither. Since there is only one
handle in any sentential form of \( G_3 \), the sentential form \( \vdash a^k b c^k \vdash \),
where \( k = \max(m, n) \), contains no phrase occurring in a parsing context
of order \([m, n]\) or less. Therefore \( G_3 \) is not \( BCP[m, n] \) for any values
of \( m \) and \( n \). We summarize the results of these comparisons as a
Venn diagram in the following figure:

![Venn diagram of subsets of context-free grammars.](image)

As is obvious from the definition and was noted above, the Bounded
Context Parsable property is symmetric in the following sense. If \( G \)
is \( BCP[m, n] \), then \( G^R \) is \( BCP[n, m] \) where \( G^R \), the reversal of \( G \),
is obtained by reflecting the right hand sides of all the productions of \( G \).
This observation leads to an interesting comparison of subsets of languages.

A language $L$ will be said to be Bounded Context Parsable (BCP) if there exists a context-free grammar $G$ such that $G$ is $\text{BCP}[m, n]$ for some values of $m$ and $n$ and $L = L(G)$. Similarly $L$ will be said to be BC, BRC, BLC, LR or RL if there exists a $G$ such that $L = L(G)$ and $G$ is $\text{BC}(m, n)$, $\text{BRC}(m, n)$, $\text{BLC}(m, n)$, $\text{LR}(n)$ or $\text{RL}(m)$ for some values of $m$ and $n$. Knuth shows [KNU] that the set of languages that are LR, which we will denote $L_{\text{LR}}$, is exactly the set $D$ of deterministic languages defined by Ginsburg and Greibach [GIN]. Knuth also shows that $L_{\text{BRC}} = L_{\text{LR}}$ and that the language $\{\text{a}^n\text{b}^n\text{c}^n | n \geq 1\}$ is not deterministic and therefore not LR or BRC. We will construct a grammar $G_4$ for this language and show that it is BCP.

Let $G_4$ be the grammar with productions:

1) $S \rightarrow \text{a} \text{U} \text{a}$
2) $S \rightarrow \text{b} \text{V} \text{c}$
3) $U \rightarrow \text{a} \text{U} \text{B}$
4) $U \rightarrow \text{a} \text{B}$
5) $V \rightarrow \text{a} \text{V} \text{D} \text{D}$
6) $V \rightarrow \text{a} \text{D} \text{D}$
7) $B \rightarrow \text{b}$
8) $D \rightarrow \text{b}$
For each of productions 1) - 6), \((\epsilon, \epsilon)\) is a parsing context. \((\epsilon, \dagger)\) and \((\epsilon, B)\) are parsing contexts for \(B \rightarrow b\); \((\epsilon, c)\) and \((\epsilon, D)\) are parsing contexts for \(D \rightarrow b\). Any sentential form of \(G_4\) has either a phrase of one of productions 1) - 6) occurring in it or an occurrence of \(b\) with one of the characters, \(B, D, c,\) or \(\dagger\) on its right. Therefore, every sentential form of \(G_4\) has a phrase occurring in a parsing context of order \([0, 1]\) or less and the grammar is seen to be BCP[0, 1]. Thus we have that:

\[
L_{BCP} \supseteq D.
\]

If \(L\) is deterministic, the construction of Knuth produces a BRC grammar for \(L\), and that grammar is also BCP [KNU]. That the inclusion is proper is demonstrated by the existence of \(G_4\).

We will let \(D^R\) denote the set of languages whose reversals are deterministic. \(L(G_4) \in D^R\) since \(G_4^R\) is LR(0). Using the same approach as above we can construct a grammar \(G_5\) such that \(G_5\) is BCP but \(L(G_5) \not\in D \cup D^R\). Let \(G_5\) be the grammar with productions:

1) \(S \rightarrow \dagger s_1 d s_2^\dagger\)

2) \(s_1 \rightarrow u_1\)

3) \(s_2 \rightarrow u_2\)

4) \(s_1 \rightarrow v_1 c\)

5) \(s_2 \rightarrow c v_2\)

6) \(u_1 \rightarrow a u_1 b_1\)

7) \(u_2 \rightarrow b_2 u_2 a\)
8) \( U_1 \rightarrow a B_1 \)
9) \( U_2 \rightarrow B_2 a \)
10) \( V_1 \rightarrow a V_1 D_1 D_1 \)
11) \( V_2 \rightarrow D_2 D_2 V_2 a \)
12) \( V_1 \rightarrow a D_1 D_1 \)
13) \( V_2 \rightarrow D_2 D_2 a \)
14) \( B_1 \rightarrow b \)
15) \( B_2 \rightarrow b \)
16) \( D_1 \rightarrow b \)
17) \( D_2 \rightarrow b \)

Clearly \( G_5 \) is \( BCP[1,1] \), but \( L(G_5) \) cannot be the language accepted by a deterministic push-down automaton. We illustrate these comparisons of sets of languages in the following figure:

![Diagram](image)

Figure 2. Venn diagram of subsets of context-free languages.
SECTION 4. TESTING FOR BOUNDED CONTEXT PARSABILITY

In this section we will show that the set $S_{BCP}[m,n]$ satisfies our property 1, but that $S_{BCP}$ does not.

So far we have given no general method for determining of an arbitrary grammar $G$ and integers $m$ and $n$, whether $G$ is $BCP[m,n]$. $G_1$ was shown to be $BCP[1,2]$ but not $BCP[1,1]$ by a special analysis by cases of the different sentential forms possible. It was shown that $G_3$ fails to be $BCP[m,n]$ for any values of $m$ and $n$ by showing that one particular sentential form of $G_3$ had no phrase occurring in a parsing context of order $[m,n]$ or less. We demonstrate the existence of a general decision procedure in the following:

Theorem: There is an algorithm to determine of an arbitrary reduced context-free grammar $G$ and arbitrary integers $m$ and $n$, whether $G$ is $BCP[m,n]$.

Proof: 1) Let $G = (V, P, V_T, S)$ be a reduced context-free grammar, $P = o(P)$, and $m$, $n \geq 0$.

2) For each production $A_i \rightarrow x_i$ in $P$, compute $DC_i[m,n]$, the set of derivation contexts of order $[m,n]$ or less for $A_i \rightarrow x_i$. This can be effectively computed by deciding for each of the pairs $(w,y)$ such that $\lambda(w) \leq m$ and $\lambda(y) \leq n$, whether $S \rightarrow^* \ldots wA_i y \ldots$ [BPS].
3) Compute the subsets $PC_1[m, n] \subseteq DC_1[m, n]$ for each of the $p$ productions in $P$. That is, for each $(w, y) \in DC_1[m, n]$, determine if $(w, y)$ is a parsing context for the $i$th production. Floyd's method of analysing the sixteen sets of relations which we described in section 1 is an effective method for determining whether an occurrence of $v_i$ in the context $(w, y)$ is necessarily an $A_i$-derivative.

4) Let $C_1[m, n] = \{ w_{ij} x_{ij}, y_{ij} | (w_{ij}, y_{ij}) \in PC_1[m, n] \}$ for all $1 \leq i \leq p$. Let $C[m, n] = \bigcup_{i=1}^{p} C_1[m, n]$.

Since $C[m, n]$ is a finite set, it is regular, and therefore the set $R = V^* \cdot C[m, n] \cdot V^*$ is regular where we use $\cdot$ to indicate the complex product as usual. Notice that $R$ is the set of all strings in $V^*$ that contain at least one phrase occurring in a parsing context of order $[m, n]$ or less.

5) Construct $G'$ such that $L(G') = SF(G)$. Let $G' = (V', P', V'_T, S')$ where:

- $V' = V \cup \{ A' | A \in V_N \} \cup S''$
- $V'_T = V$
- $P'$ is the set of productions obtained as follows:
i) If \( A \rightarrow x \) is a production of \( P \), then
\[ A' \rightarrow x' \] is a production of \( P' \) where \( x' \)
is the string over \( V' \) obtained from \( x \) by
priming all the non-terminal characters in \( x \).

ii) \( A' \rightarrow A \) is in \( P' \) for all \( A \in V_N \).

iii) If \( S \rightarrow x \) is in \( P \), then \( S'' \rightarrow x' \) is in
\( P' \) where again \( x' \) is obtained by priming the
non-terminals in \( x \). The reason for treating
\( S \) differently from the other non-terminals of
\( G \) is that we do not wish to consider \( \vdash S \vdash \)
to be a sentential form of \( G \) (unless, of
course, \( S \vdash S \) in which case \( G \) is
ambiguous). That is, we want every sentential
form of \( G \) to have a handle.

6) Since \( R \) is regular, \( \overline{R} = V^* - R \) is regular \([RS]\).

7) Since \( \overline{R} \) is regular and \( G' \) is context-free, there
exists a context-free grammar \( \overline{G} \) such that \( L(\overline{G}) = L(G') \cap \overline{R} \). \([BPS]\).

8) Now, \( G \) is \( BCP[m,n] \)
iff every sentential form of \( G \) has a phrase occurring
in a parsing context of order \([m,n]\) or less
iff \( L(G') \subset R \)
iff \( L(G') \cap \overline{R} \) is empty.

iff \( L(\overline{G}) \) is empty.

9) There is an effective procedure for determining whether the language of an arbitrary context-free grammar is empty [BPS].

Therefore, we can effectively determine whether \( G \) is \( BCP[m, n] \).

Q.E.D.

Notice that when the above decision procedure responds affirmatively, we can immediately construct a reduction system for the sets \( PC_i[m, n] \) to parse sentences of \( G \) as was shown in section 2. Thus for each pair of values for \( m \) and \( n \), \( S_{BCP[m, n]} \) satisfies properties 1 and 2.

The above decision procedure will tell us whether a grammar is \( BCP[m, n] \) only for given \( m \) and \( n \). Therefore given a grammar \( G \), we can first determine whether \( G \) is \( BCP[1, 1] \), and if not, we can then determine whether \( G \) is \( BCP[2, 2] \), and so forth. Before beginning this sequence of tests, we would like to be assured that at some point the decision procedure will respond affirmatively. That is, we would like to be able to decide the more general question, do there exist integers \( m \) and \( n \) such that \( G \) is \( BCP[m, n] \). We show that this question is recursively undecidable for context-free grammars with the help of the following:
Lemma: If $G$ is a linear context-free grammar, then $G$ is $BC(m, n)$ iff $G$ is $BCP[m, n]$.

Proof: 1) If $G$ is $BC(m, n)$, it is $BCP[m, n]$ since $S_{BC(m, n)} \subseteq S_{BCP[m, n]}$ as was shown in section 3.

2) If $G$ is $BCP[m, n]$, i.e., if every sentential form of $G$ has a phrase occurring in a parsing context of order $[m, n]$ or less, then every derivation context of order $[m, n]$ is a parsing context since every sentential form has only one handle. Therefore $G$ is $BC(m, n)$.

Knuth has shown that the problem to determine of an arbitrary linear context-free grammar $G$ whether there exist integers $m$ and $n$ such that $G$ is $BC(m, n)$ is recursively unsolvable [KNU]. Since $G$ is $BC(m, n)$ iff $G$ is $BCP[m, n]$ for linear $G$, we have immediately the following:

Theorem: The problem to determine of an arbitrary context-free grammar $G$ whether there exist integers $m$ and $n$ such that $G$ is $BCP[m, n]$ is recursively unsolvable.
BIBLIOGRAPHY


