View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

JOURNAL OF COMPUTER AND SYSTEM SCIENCES **22**, 178–197 (1981)

SR(s, k) Parsers: A Class of Shift–Reduce Bounded-Context Parsers

DAVID A. WORKMAN

Department of Computer Science, University of Central Florida, Orlando, Florida 32816

Received May 16, 1978; revised November 7, 1979

We introduce a class of bottom-up parsers called SR(s, k) parsers, where s denotes a stack bound and k the lookahead bound. The importance of this class of parsers is the fact that states are formed by the union of item-sets associated with a canonical LR(k) parser. One of our major results states that the class of SR(s, k) grammars properly includes the (s, k)-weak precedence grammars but is properly included within the class of (s, k)-bounded right-context grammars.

INTRODUCTION

Knuth's algorithm [7] for generating LR(k) parsers produced sets of "items" that served to characterize the states of a general bottom-up parser. Early [2] employed the "item-set" framework to describe an even more general, but generally less efficient algorithm for parsing any context-free grammar. More recently, Geller and Harrison [4] have developed the "item-set" framework and used it to describe strict-deterministic, SLR(k) and LALR(k) parsers. We present in this paper yet another application of Knuth's algorithm to describe precedence and bounded right-context parsers. In our adaptation of Knuth's algorithm we add a parameter "s" which represents a bound on the number of stack symbol employed by the resulting parser in determing the next "action" to perform. The algorithm produces a canonical collection of item-sets from which "action" and "goto" tables are constructed; these tables are defined in the same fashion as those for an LR(k) parser.

Section I contains definitions and notational conventions used throughout the paper. One method for generating the canonical collection of (s, k)-consistency sets is presented in section II. In section III we describe the procedure for constructing a shift–reduce parser based on the canonical collection of item-sets. We designate such parsers as "SR(s, k)" parsers. In Section IV we describe the properties of SR(s, k) grammars in relation to the classes of (s, k)-weak precedence and (s, k)-bounded right-context grammars. The paper is completed with a few comments regarding practical considerations for implementing SR(s, k) parsers followed by our conclusions.

0022-0000/81/020178–20$02.00/0

Copyright © 1981 by Academic Press, Inc.
All rights of reproduction in any form reserved.

I. Preliminaries

We assume the reader is familiar with the basic definitions, notation and terminology found in Aho and Ullman [1]. In particular we assume familiarity with Knuth's algorithm [7] for generating LR(k) parsers where "states" are sets of "LR(k)-items." In the paragraphs to follow, we give many of these definitions for the reader's convenience.

A grammar is denoted by a 4-tuple, $G = (N, T, P, S)$, where N, T and P are the sets of *nonterminals*, *terminals* and *productions*, respectively. $S \in N$ denotes the *start symbol* of G. The *vocabulary* of G will be denoted by $V_G (V_G = N \cup T)$. Throughout the paper we will be dealing exclusively with reduced context-free grammars. The null string will be denoted by "ε."

Definition 1. If α is some string in V^* and $s \geqslant 0$, then:

$$\begin{aligned} \mathrm{PREF}_s(\alpha) &= \alpha, && \text{if } |\alpha| \leqslant s; \\ &= \beta, && \text{if } |\alpha| > s \text{ and } \alpha = \beta\gamma, \text{ where } |\beta| = s. \\ \mathrm{SUFF}_s(\alpha) &= \alpha, && \text{if } |\alpha| \leqslant s; \\ &= \gamma, && \text{if } |\alpha| > s \text{ and } \alpha = \beta\gamma, \text{ where } |\gamma| = s. \end{aligned}$$

For $k \geqslant 0$,

$$\mathrm{FIRST}_k(\alpha) = \{\mathrm{PREF}_k(x) \mid \alpha \underset{G}{\overset{*}{\Longrightarrow}} x \in T^*\}$$

and

$$\begin{aligned} \mathrm{EFF}_k(\alpha) &= \mathrm{PREF}_k(\alpha), \qquad \text{if} \quad \alpha \in T^* \\ &= \{\mathrm{PREF}_k(x) \mid \alpha \underset{rm}{\overset{*}{\Longrightarrow}} \sigma y \underset{rm}{\Longrightarrow} x \in T^* \\ &\qquad \text{and either } \sigma \in T \text{ or } \sigma \in N \text{ and } y \neq x\}. \end{aligned}$$

Definition 2. A k-item $[X \to \alpha \cdot \beta \mid u]$ is said to be *valid for* $\theta\alpha$, a viable prefix of G, if there is a rightmost derivation, $S \Rightarrow_{rm}^* \theta X w \Rightarrow_{rm} \theta\alpha\beta w$, for which $u = \mathrm{PREF}_k(w)$.

In Knuth's algorithm for constructing LR(k) parsers, states of the parser are characterized by sets of k-items. For example, if the parse stack contains the string γ, then the state of the parser is defined by the set $\Lambda(\gamma)$ of all k-times valid for γ. Because the set $\Lambda(\gamma)$ is determined by γ, an LR(k) parser effectively uses the entire stack contents in deciding the action to perform at each step of the parse.

Our objective is to construct shift–reduce parsers in a manner similar to that of Knuth's but with one important difference; states will incorporate information determined only by the top s symbols of the stack. With this goal in mind we proceed with the next definition.

Definition 3. The k-item $I = [X \to \alpha \cdot \beta \mid u]$ is said to be *s-consistent with* γ, $s \geqslant 0$, iff I is valid for some viable prefix $\theta\alpha$, where $\gamma = \mathrm{SUFF}_s(\theta\alpha)$.

Notation. For given values of s and k, and $\gamma \in V^{*s}$ we let $\Lambda^G_{s,k}(\gamma)$ denote the *(s, k)-consistency set for* γ and contains all k-times s-consistent with γ. The subscripts and superscript will be dropped whenever s, k and G are understood. We use $\mathscr{C}_{s,k}(G)$ to denote the collection of all non-empty (s, k)-consistency sets, $\Lambda(\gamma)$, for G.

In the next section we describe how to form the canonical collection, $\mathscr{C}(G)$, of (s, k)-consistency sets for G from the canonical collection of LR(k) sets for G. Having computed $\mathscr{C}(G)$, a set of parsing tables can easily be obtained which form the basis for an SR(s, k) parser for G. Before formally introducing these concepts we present some notational conveniences via the following definitions.

Definition 4. Let $G = (N, T, P, S)$ and Γ be a set of k-items for G. Furthermore, let $I = [X \to \alpha \cdot \sigma\beta \mid u]$, where $\sigma \in V$. Then

$$\mathrm{MOVE}(I) = [X \to \alpha\sigma \cdot \beta \mid u],$$

$$\mathrm{START}(G) = \{[S \to \cdot\delta \mid \varepsilon] \mid S \to \delta \in P\},$$

$$\mathrm{CORE}(\Gamma, \sigma) = \{\mathrm{MOVE}(I) \mid I = [X \to \alpha \cdot \sigma\beta \mid u] \in \Gamma \text{ for some } x, \alpha, \beta \text{ and } u\}.$$

Definition 5. The *close relation*, $\supset$, is defined on the set of k-items for G as follows.

$$[Y \to \alpha \cdot A\beta \mid u] \supset [A \to \cdot\delta \mid v] \qquad \text{iff } A \to \delta \in P \text{ and } v \in \mathrm{FIRST}_k(\beta u).$$

Let "$\supset*$" and "$\supset+$" denote the reflexive–transitive and transitive closures of $\supset$, respectively.

If Γ is a set of k-items, then

$$\mathrm{CLOSE}(\Gamma) = \{I \mid J \supset* I \text{ and } J \in \Gamma\}.$$

II. Computing (s, k)-Consistency Sets

We have defined $\mathscr{C}_{s,k}(G)$ to be the collection of all non-empty (s, k)-consistency sets, $\Lambda(\gamma)$, where $\gamma \in V^{*s}$. In this section we show that each $\Lambda(\gamma) \in \mathscr{C}(G)$ is the union of some of the members of the canonical collection, $\mathscr{S}_k(G)$, of LR(k) sets for G.

From Definitions 2 and 3 it follows that an item I belongs to $\Lambda(\gamma)$ if and only if I is valid for some viable prefix, θ, for which $\gamma = \mathrm{SUFF}_s(\theta)$. Thus if Σ is the member of $\mathscr{S}_k(G)$ representing the set of all k-items valid for θ, then clearly $\Sigma \subseteq \Lambda(\gamma)$. Our next definition suggests a simple algorithm for computing exactly those Σ in $\mathscr{S}_k(G)$ which contribute to $\Lambda(\gamma)$ for a given $\gamma \in V^{*s}$ as well as those γ for which $\Lambda(\gamma) \in \mathscr{C}(G)$.

Definition 6. Let $G' = (N, T, P', S')$ be the augmented from of some grammar

G and let $\mathscr{S}_k(G)$ denote the canonical collection of LR(k) sets constructed from G'. Furthermore, let $\Sigma_0 = \text{CLOSE}(I)$, where $I = [S' \to \cdot S \mid \varepsilon]$ and let g denote the usual GOTO-function defined on $\mathscr{S}_k(G)$ extended to strings in V^*. Given $s \geqslant 0$ we define sets $\Gamma(\gamma)$, $\gamma \in V^{*s}$, as follows.

(i) For $|\gamma| < s$, $\Gamma(\gamma) = g(\Sigma_0, \gamma)$ whenever $g(\Sigma_0, \gamma)$ is defined;

(ii) For $|\gamma| = s$, $\Gamma(\gamma) = \bigcup \{g(\Sigma, \gamma) \mid \Sigma \in \mathscr{S}_k(G)$ and $g(\Sigma, \gamma)$ is defined$\}$.

If $\gamma \in V^{*s}$ such that $|\gamma| < s$, then I is s-consistent with γ if and only if γ is a viable prefix of G. Thus $\Lambda^G_{s,k}(\gamma) = g(\Sigma_0, \gamma)$ whenever $g(\Sigma_0, \gamma)$ is defined. For $|\gamma| = s$, γ must occur as a suffix of some viable prefix θ, where $|\theta| \geqslant s$. Letting $\theta = \theta'\gamma$ we have $g(\Sigma_0, \theta) = g(g(\Sigma_0, \theta'), \gamma) \subseteq \Gamma(\gamma)$. On the other hand, let Σ be any member of $\mathscr{S}_k(G)$ for which $g(\Sigma, \gamma)$ is defined, then there is some θ' such that $g(\Sigma_0, \theta') = \Sigma$ and it follows that $\theta'\gamma$ is a viable prefix of G. Thus $\Gamma(\gamma) = \Lambda^G_{s,k}(\gamma)$ for each γ a suffix of length s or less of some viable prefix of G. We have the following:

THEOREM 1. *The canonical collection of* (s, k)*-consistency sets,* $\mathscr{C}_{s,k}(G)$, *for a grammar,* G, *is specified according to Definition* 6, *where each member of* $\mathscr{C}(G)$ *is the union of valid* LR(k) *sets for* G.

III. The Class of SR(s, k) Parsers

In this section we show that the canonical collection of consistency sets, $\mathscr{C}_{s,k}(G)$, forms the basis for a nondeterministic right-parser for G. Each member of $\mathscr{C}(G)$ represents a state in a *S*hift-*R*educe parser defined in the same manner as the canonical LR(k) parser for G. For this reason we designate such a parser as the *canonical* SR(s, k) *parser for* G and denote it by $\mathscr{P}_{s,k}(G)$. Deterministic SR(s, k) parsers will be taken up in the next section.

To the end of demonstrating $\mathscr{P}_{s,k}(G)$ is in fact a right-parser, we establish the more general result that any "cover" for $\mathscr{S}_k(G)$ consistent with the GOTO-function, g, of an LR(k) parser defines a basis for a right-parser for G. The notion of a cover for G is made precise by the next definition.

DEFINITION 7. A pair (Ω, Ψ) is a k-cover for G if $\Omega = \{Q_0, Q_1, \ldots, Q_n\}$ is a collection of non-empty sets satisfying

(i) For each i, $0 \leqslant i \leqslant n$, there is a subset, $\bar{Q}_i$, of $\mathscr{S}_k(G)$ such that $Q_i = \bigcup \bar{Q}_i$;

(ii) $\bigcup \Omega = \bigcup \mathscr{S}_k(G)$;

and Ψ is a partial function, $\Psi \colon \Omega \times V_G \to \Omega$, satisfying

(iii) For each $Q \in \Omega$ and $Z \in V_G$ for which $\Psi(Q, Z)$ is defined, there exists $\Sigma \in \bar{Q}$ such that $g(\Sigma, Z)$ is defined and furthermore, for all $\Sigma \in \bar{Q}$ for which $g(\Sigma, Z)$ is defined, it holds that $g(\Sigma, Z) \in \overline{\Psi(Q, Z)}$; that is, $g(\Sigma, Z) \subseteq \Psi(Q, Z)$.

We can now formally introduce a class of right-parsers based on a k-cover for a grammar, G.

DEFINITION 8. For $k \geqslant 0$, a *k-parser for* $G = (N, T, P, S)$ is a 4-tuple $(\Omega, Q_0, \tau, \Psi)$, where (Ω, Ψ) is a k-cover for G, $Q_0 \in \Omega$ is the initial state and must satisfy $\Sigma_0 \in \bar{Q}_0$, where Σ_0 is the initial state of the canonical LR(k) parser and finally, τ is the action relation from $\Omega \times T^{*k}$ to subsets of $\{$*error*, *accept*, *shift*$\} \cup \{$*reduce* $p \mid p \in P\}$. Furthermore, for $Q \in \Omega$, τ satisfies

(i) If $[X \to \alpha . \beta \mid u] \in Q$, $\beta \neq \varepsilon$, then *shift* $\in \tau(Q, \varepsilon)$ if $k = 0$; for $k > 0$, *shift* $\in \tau(Q, v)$ for every $v \in \mathrm{EFF}_k(\beta u)$.

(ii) If $[X \to \alpha . \mid u] \in Q$ then *reduce* $p \in \tau(Q, u)$ where $p = X \to \alpha$.

(iii) If $[S' \to S. \mid \varepsilon] \in Q$, then *accept* $\in \tau(Q, \varepsilon)$.

(iv) $\tau(Q, v) = \{$*error*$\}$ whenever $\tau(Q, v)$ is not defined by one or more of (i)–(iii).

A k-parser operates similar to an LR(k) parser and begins with the stack initialized to Q_0. A *shift* causes the next input symbol, a, to be pushed on the stack along with the "next state," $\Psi(Q, a)$, where Q denotes the current state of the parser (the state symbol originally on top of the stack). If *reduce* p is selected as the parsing action, where $p = X \to w$, then the parser outputs "p" if and only if the rightpart, w, actually occurs on top of the stack (ignoring state symbols) and $\Psi(Q', X)$ is defined where Q' is the state symbol immediately below the handle w. A reduction by p causes the top $2\,|w|$ symbols of the stack to be replaced by $X\Psi(Q', X)$. When *accept* is executed, the parser halts and outputs "accept" provide the stack contains $Q_0 S \Psi(Q_0, S)$ and otherwise outputs "error."

The requirement that the handle, w, actually agree with the stack contents during a reduction is a necessary condition, for otherwise a parser could accept a string not in $L(G)$. This point will be demonstrated later by Example 2. We note further that this constraint is natural in the sense that it is a condition guaranteed with canonical LR(k) parsers.

We now formally establish the fact that every k-parser for a reduced *cfg*, G, defines a right-parser for G.

THEOREM 2. *Let $G = (N, T, P, S)$ be a reduced cfg. For $k \geqslant 0$, every k-parser for G produces* "πaccept" *as output if and only if its input, x, belongs to $L(G)$, where* $S \Rightarrow_{rm}^{\pi^R} x$.

Proof. The result is established in two steps. First, we show that if $x \in L(G)$, then any k-parser, A, for G can duplicate an accepting computation of the canonical LR(k) parser for G. Finally, we show that if A produces output "*πaccept*" for some input, $x \in T^*$, then $S \Rightarrow_{rm}^{\pi^R} x$ in G.

To begin we recall some notation frequently used to describe the action of a shift-reduce parser (see [1, 4]). Let (γ, x, ρ) denote a parser configuration, where γ denotes the stack contents with top on the right, x denotes the remainder of the input with lookahead, $\mathrm{PREF}_k(x)$, and ρ denotes the current output. Furthermore, $\vdash_A$, will denote the "move relation" on configurations of parser, A.

Let $A = (\Omega, Q_0, \tau, \Psi)$ be any k-parser for G. Let $C = (\mathscr{S}_k(G), \Sigma_0, f, g)$ denote the canonical LR(k) parser for G, where f and g are the action and goto functions, respectively, and $\Sigma_0 \in \mathscr{S}_k(G)$ is the initial state. Now suppose $x \in L(G)$, then there exists a sequence of moves of C accepting x; that is,

$$(\Sigma_0, x, \varepsilon) \overset{+}{\underset{C}{\vdash}} (\Sigma_0 Z_1 \Sigma_1 \cdots Z_m \Sigma_m, x', \rho) \overset{*}{\underset{C}{\vdash}} (\Sigma_0 S \Sigma_1, \varepsilon, \pi accept),$$

where $S \Rightarrow_{rm}^{\pi^R} x$ in G. To show x is also accepted by A, we establish the following

Claim 1. $(\Sigma_0, x, \varepsilon) \vdash_C^n (\Sigma_0, Z_1 \Sigma_1 \cdots Z_m \Sigma_m, x', \rho)$ implies $(Q_0, x, \varepsilon) \vdash_A^n (Q_0 Z_1 Q_1 \cdots Z_m Q_m, x', \rho)$, where $\Sigma_i \in \bar{Q}_i$, $0 \leqslant i \leqslant m$.

A formal proof of Claim 1 proceeds by induction on the number of moves, n. This is done by considering the conditions that must prevail for each type of move C could make in a given configuration and the showing the same move can be duplicated by A. The crucial observations are first, $\Sigma_0 \in Q_0$, and second, $\Sigma_{i+1} = g(\Sigma_i, Z) \subseteq \Psi(Q_i, Z) = Q_{i+1}$. From this the details are straightforward and have been omitted.

Now consider

Claim 2. If $(Q_0, x, \varepsilon) \vdash_A^n (Q_0 Z_1 Q_1 \cdots Z_m Q_m, x', \pi)$, with $\pi \in P^*$, then $Z_1 Z_2 \cdots Z_m x' \Rightarrow_{rm}^{\pi^R} x$ in G.

Claim 2 is also established by induction on the number of moves, n, where each move must be a shift or reduce action. The crucial observation is that a reduction by $p = X \to w$ is successful only if w actually appears on top of the stack and that $\Psi(Q_j, X)$ is defined, where Q_j is the state symbol immediately below w. Again the details are straightforward and not presented here.

To complete the proof suppose $(Q_0, x, \varepsilon) \vdash_A^+ (\gamma, \varepsilon, \pi accept)$. If $(Q_0 Z_1 Q_1 \cdots Z_m Q_m, \varepsilon, \pi)$ is the configuration just prior to the *accept* move, then $Q_0 Z_1 Q_1 \cdots Z_m Q_m = Q_0 S \Psi(Q_0, S)$. By Claim 2, $S \Rightarrow_{rm}^{\pi^R} x$ in G. Thus $x \in L(G)$. This concludes the proof.

As a corollary to Theorem 2 we formally establish that the canonical SR(s, k) parser for G is a k-parser and therefore a nondeterministic right-parser for G.

COROLLARY. *The canonical* SR(s, k) *parser*, $\mathscr{P}_{s,k}(G) = (\mathscr{C}_{s,k}(G), A_{s,k}^G(\varepsilon), \tau, \Psi)$ *is a k-parser for G where for all $A(\gamma) \in \mathscr{C}(G)$ and $Z \in V_G$ for which there exists an item $[X \to \alpha . Z\beta \mid u] \in A(\gamma)$ we define* $\Psi(A(\gamma), Z) = A(\mathrm{SUFF}_s(\gamma Z))$.

Proof. To demonstrate this result we need only show $(\mathscr{C}(G), \Psi)$ is a k-cover for G. In keeping with the notation of Definition 7, $\bar{A}(\gamma)$ will denote the subset of $\mathscr{S}_k(G)$ satisfying $A(\gamma) = \bigcup \bar{A}(\gamma)$ according to Definition 6. To begin we observe that conditions (i) and (ii) of Definition 7 follow immediately from Definition 6 and Theorem 1.

To establish (iii) of Definition 7 consider first the case $s = 0$. Since $\mathrm{SUFF}_0(\gamma) = \varepsilon$ for all $\gamma \in V_G^*$, then it follows that $\Sigma \in \bar{A}(\varepsilon)$ for each $\Sigma \in \mathscr{S}_k(G)$. Since $\mathscr{C}(G) = \{A(\varepsilon)\}$ it follows at once that $(\mathscr{C}_{0,k}(G), \Psi)$ is a k-cover for G.

Now suppose $s > 0$ and $\Psi(\Lambda(\gamma), Z) = \Lambda(\gamma')$ for some $\gamma \in V_G^{*s}$ and $Z \in V_G$, where $\gamma' = \mathrm{SUFF}_s(\gamma Z)$. By Definition 6 and the definition of Ψ there exists an item, $I = [X \to \alpha . Z\beta \mid u]$, and $\Sigma \in \mathscr{S}_k(G)$ such that $I \in \Sigma \in \bar{\Lambda}(\gamma)$. We must show $g(\Sigma, Z) \in \bar{\Lambda}(\gamma')$. From $\Sigma \in \mathscr{S}_k(G)$ and Definition 6 we know there exists $\theta \in V_G^*$ such that $\Sigma = g(\Sigma_0, \theta)$ and $\gamma = \mathrm{SUFF}_s(\theta)$. If $|\gamma| < s$, then $\theta = \gamma$ and Definition 6 imply $\Sigma = \Lambda(\gamma)$. But then we have $g(\Sigma, Z) = g(g(\Sigma_0, \theta), Z) = g(g(\Sigma_0, \gamma), Z) = g(\Sigma_0, \gamma Z)$. If $|\gamma Z| < s$, then $g(\Sigma_0, \gamma Z) = \Lambda(\gamma Z) = \Lambda(\mathrm{SUFF}_s(\gamma Z)) = \Lambda(\gamma')$. If $|\gamma Z| = s$, then since $g(\Sigma_0, \gamma Z)$ is defined, Definition 6 implies $\bar{\Lambda}(\gamma Z) = \bar{\Lambda}(\gamma')$ contains $g(\Sigma_0, \gamma Z)$. Thus $g(\Sigma, Z) \in \bar{\Lambda}(\gamma')$ for $|\gamma| < s$. If $|\gamma| = s$, then $g(\Sigma, Z) = g(g(\Sigma_0, \theta), Z) = g(\Sigma_0, \theta Z) = g(\Sigma_0, \theta' \gamma Z) = g(\Sigma_0, \theta' \gamma') = g(g(\Sigma_0, \theta'), \gamma') \in \bar{\Lambda}(\gamma')$. In any case, $\Psi(\Lambda(\gamma), Z) = \Lambda(\gamma')$ implies $g(\Sigma, Z) \in \bar{\Lambda}(\gamma')$ for every $\Sigma \in \bar{\Lambda}(\gamma)$ and $Z \in V_G$ for which $g(\Sigma, Z)$ is defined. We conclude $(\mathscr{C}_{s,k}(G), \Psi)$ is a k-cover for G and $\mathscr{P}_{s,k}(G)$ is a k-parser.

We illustrate the construction of a canonical SR(s, k) parser by the following example.

EXAMPLE 1. Consider $G = (N, T, P, S)$, where $N = \{S, T\}$, $T = \{+, (\,,), a\}$ and $P = \{1\colon S \to S + T,\ 2\colon S \to T,\ 3\colon T \to (S),\ 4\colon T \to a\}$. The canonical SR(1, 1) parser for G is constructed according to the following procedure.

Step 1. Construct $\mathscr{S}_1(G)$, the canonical collection of LR(1) sets for G and the associated GOTO-function, g. For the sake of brevity, items will generally be of the form $[X \to \alpha . \beta \mid z]$, where z denotes a string of symbols in $(T \cup \{\varepsilon\})^+$. This notation is used to represent a set of items of the form $[X \to \alpha . \beta \mid u]$, where $u \in T \cup \{\varepsilon\}$. $\mathscr{S}_1(G) = \{\Sigma_0, \Sigma_1 \cdots \Sigma_{15}\}$, where

$$\Sigma_0 = \{[S' \to .S \mid \varepsilon], [S \to .S + T \mid \varepsilon +], [S \to .T \mid \varepsilon +], [T \to .a \mid \varepsilon +], [T \to .(S) \mid \varepsilon +]\},$$

$$\Sigma_1 = g(\Sigma_0, S) = \{[S' \to S. \mid \varepsilon], [S \to S. + T \mid \varepsilon +]\},$$

$$\Sigma_2 = g(\Sigma_0, T) = \{[S \to T. \mid \varepsilon +]\},$$

$$\Sigma_3 = g(\Sigma_0, a) = \{[T \to a. \mid \varepsilon +]\},$$

$$\Sigma_4 = g(\Sigma_0, () = \{[T \to (.S) \mid \varepsilon +], [S \to .S + T \mid)+], [S \to .T \mid)+], [T \to .a \mid)+], [T \to .(S) \mid)+]\},$$

$$\Sigma_5 = g(\Sigma_1, +) = \{[S \to S + .T \mid \varepsilon +], [T \to .a \mid \varepsilon +], [T \to .(S) \mid \varepsilon +]\},$$

$$\Sigma_6 = g(\Sigma_4, S) = \{[T \to (S.) \mid \varepsilon +], [S \to S. + T \mid)+]\},$$

$$\Sigma_7 = g(\Sigma_4, T) = \{[S \to T. \mid)+]\},$$

$$\Sigma_8 = g(\Sigma_4, a) = \{[T \to a. \mid)+]\},$$

$$\Sigma_9 = g(\Sigma_4, () = \{[T \to (.S) \mid)+], [S \to .S + T \mid)+], [S \to .T \mid)+], [T \to .a \mid)+], [T \to .(S) \mid)+]\},$$

$$\Sigma_{10} = g(\Sigma_s, T) = \{[S \to S + T. \mid \varepsilon +]\},$$

$$g(\Sigma_5, a) = \Sigma_3,$$

$$g(\Sigma_5, () = \Sigma_4,$$

$$\Sigma_{11} = g(\Sigma_6,)) = \{[T \rightarrow (S). \mid \varepsilon +]\},$$

$$\Sigma_{12} = g(\Sigma_6, +) = \{[S \rightarrow S + .T \mid) +], [T \rightarrow .a \mid) +], [T \rightarrow .(S) \mid) +]\},$$

$$\Sigma_{13} = g(\Sigma_9, S) = \{[T \rightarrow (S.) \mid) +], [S \rightarrow S. + T \mid) +]\},$$

$$g(\Sigma_9, T) = \Sigma_7,$$

$$g(\Sigma_9, a) = \Sigma_8,$$

$$g(\Sigma_9, () = \Sigma_9,$$

$$\Sigma_{14} = g(\Sigma_{12}, T) = \{[S \rightarrow S + T. \mid) +]\},$$

$$g(\Sigma_{12}, a) = \Sigma_8,$$

$$g(\Sigma_{12}, () = \Sigma_9,$$

$$\Sigma_{15} = g(\Sigma_{13},)) = [T \rightarrow (S). \mid) +],$$

$$g(\Sigma_{13}, +) = \Sigma_{12},$$

Step 2. Construct $\mathscr{C}_{1,1}(G)$ according to Definition 6.

$$\Lambda(\varepsilon) = \Sigma_0,$$

$$\Lambda(S) = \Sigma_1 \cup \Sigma_6 \cup \Sigma_{13},$$

$$\Lambda(T) = \Sigma_2 \cup \Sigma_7 \cup \Sigma_{10} \cup \Sigma_{14},$$

$$\Lambda(a) = \Sigma_3 \cup \Sigma_8,$$

$$\Lambda(+) = \Sigma_5 \cup \Sigma_{12},$$

$$\Lambda(() = \Sigma_4 \cup \Sigma_9,$$

$$\Lambda()) = \Sigma_{11} \cup \Sigma_{15}.$$

Step 3. Construct the action relation, τ, and the GOTO-function Ψ.

τ	a	$+$	$($	$)$	ε	Ψ	S	T	a	$+$	$($	$)$
ε	shift	error	shift	error	error		S	T	a		$($	
S	error	shift	error	error	accept					$+$		
T	error	reduce 2, 1	error	reduce 2, 1	reduce 2, 1							
a	error	reduce 4	error	reduce 4	reduce 4							
$+$	shift	error	shift	error	error			T	a		$($	
$($	shift	error	shift	error	error		S	T			$($	
$)$	error	reduce 3	error	reduce 3	reduce 3							

This completes the construction. Note that $\mathscr{P}_{1,1}(G)$ appears to be nondeterministic because of the multiple action entries for $\tau(T,+)$, $\tau(T,))$ and $\tau(T,\varepsilon)$. However, according to our definition of determinism presented in the next section, $\mathscr{P}_{1,1}(G)$ is actually a deterministic SR(1, 1) parser for G. We will prove this statement later in Section IV.

As we note following Definition 8, the requirement that the handle actually appear on the stack during a reduction is a necessary condition for k-parsers. This fact is demonstrated by our next example.

EXAMPLE 2. Let $G=(\{S,T,F\}, \{-, a, (,)\}, P, S)$, where $P=\{1{:}\, S\rightarrow S-T$, $2{:}\, S\rightarrow T$, $3{:}\, T\rightarrow -F$, $4{:}\, T\rightarrow F$, $5{:}\, F\rightarrow a$, $6{:}\, F\rightarrow (S)\}$. The action and goto tables are shown below for the canonical SR(1, 1) parser for G.

τ	$-$	a	(	)	ε
ε	shift	shift	shift	error	error
S	shift	error	error	shift	accept
T	reduce 2, 1	error	error	reduce 2, 1	reduce 2, 1
F	reduce 4, 3	error	error	reduce 4, 3	reduce 4, 3
$-$	shift	shift	shift	error	error
a	reduce 5	error	error	reduce 5	reduce 5
(	shift	shift	shift	error	error
)	reduce 6	error	error	reduce 6	reduce 6

Ψ	S	T	F	$-$	a	(	)
ε	S	T	F	$-$	a	(	
S							)
T				$-$			
F							
$-$		T	F	$-$	a	(	
a							
(	S	T	F	$-$	a	(	
)							

We now consider the behavior of $\mathscr{P}_{1,1}(G)$ when presented the input string "$---a$," which is not in $L(G)$. Since the grammar symbols and the state names correspond, we record only the grammar symbol in the stack. Whenever a stack reduction is called for, we remove the top n symbols of the stack without verifying they actually agree with the handle, w, where $n=|w|$. The stack top is the rightmost symbol of the string representing the stack.

Stack	Input	Action
ε	$---a$	shift
$\varepsilon-$	$--a$	shift
$\varepsilon--$	$-a$	shift
$\varepsilon---$	a	shift
$\varepsilon---a$	ε	reduce 5
$\varepsilon---F$	ε	reduce 3
$\varepsilon--T$	ε	reduce 1
εS	ε	accept

It is not difficult to verify that the canonical LR(1) parser for G would halt and report "error" after the first "shift." Using Definition 8, the SR(1, 1) parser would report "error" in attempting to perform "reduce 1" and finding "$S - T$" does not appear on top of the stack.

IV. Properties of SR(s, k) Grammars

The canonical SR(s, k) parser for a grammar, G, will generally be nondeterministic if no restrictions are imposed on G. In this section we shall be interested in defining a class of context-free grammars that give rise to deterministic, one-pass SR(s, k) parsers; this class we designate the class of SR(s, k) grammars. The section will be concluded with results that show the relationship of SR(s, k) grammars to the classes of (s, k)-weak precedence and (s, k) bounded right-context grammars.

The notion of determinism for SR(s, k) parser is not as clear cur as it is for LR(k) parsers; that is, simply requiring $\tau(\Lambda(\gamma), u)$ contain at most one entry may indeed yield a deterministic parser and, perhaps, an interesting class of grammars, but is a much too severe restriction. To arrive at a suitable definition of determinism we attempted to include the largest class of bounded-context grammars possible without imposing artificial or unnatural conditions.

Definition 9. $\mathscr{P}_{s,k}(G)$ is said to be deterministic if for every $\gamma \in V^{*s}$,

(i) $\mathrm{SHIFT}(\gamma) \cap \mathrm{REDUCE}(\gamma) = \Phi$.

(ii) If distinct items, $[A \to \alpha. \mid u]$ and $[B \to \beta\alpha. \mid u]$ belong to $\Lambda(\gamma)$, then for no $\gamma' \in V^{*s}$ is it true that both $[A \to .\alpha \mid u]$ and $[B \to \beta.\alpha \mid u]$ belong to $\Lambda(\gamma')$.

(iii) $\tau(\Lambda(\mathrm{SUFF}_s(S)), \varepsilon) = \{\mathit{accept}\}$, $\mathrm{SHIFT}(\gamma) = \{u \in T^{*k} \mid \mathit{shift} \in \tau(\Lambda(\gamma), u)\}$, $\mathrm{REDUCE}(\gamma) = \{u \in T^{*k} \mid \mathit{reduce}\ i \in \tau(\Lambda(\gamma), u)$ for some $i \in P\}$.

The intent of condition (i) is to avoid any confusion between shift and reduce actions; this is consistent with all other deterministic shift–reduce methods with no backup. Condition (ii) is similar to the condition on weak-precedence grammars that permits the resolution of multiple reductions by examining symbols below the handle on the parse stack. Condition (iii) simply guarantees the parser can determine when to halt in an accepting configuration.

As we noted in the previous Section, the grammar of Example 1 is an SR(1, 1) grammar. Conditions (i) and (iii) of Definition 9 are clearly met. The reduction conflicts result from the fact that items $[S \to T. \mid) + \varepsilon]$ and $[S \to S + T. \mid) + \varepsilon]$ belong to $\mathcal{A}(T)$. However, the items $[S \to .T \mid) + \varepsilon]$ belong to $\mathcal{A}(\varepsilon) \cup \mathcal{A}(()$ whereas the items $[S \to S + .T \mid) + \varepsilon]$ belong to $\mathcal{A}(+)$. Thus (ii) of Definition 9 is also met implying G is an SR(1, 1) grammar. In resolving these conflicts during a parse, an SR(1, 1) parser would examine the grammar symbol below "T" in the stack; a reduction by $S \to S + T$ would be attempted only if "+" is found; otherwise, $S \to T$ would be attempted.

Our first result in this section relates SR(s, k) grammars to (s, k)-bounded right-context grammars. Before presenting this result, we formally define the class of SR(s, k) grammars. The definition we shall use for bounded right-context grammars is found in [1] and due primarily to Floyd [3].

Definition 10. A grammar, G, is said to be SR(s, k) if its canonical SR(s, k) parser is deterministic.

Theorem 3. *If G is* SR(s, k), *then G is* (s, k)*-bounded right-context.*

Proof. We proceed by assuming G is not (s, k)-bounded right-context and then show $\mathscr{P}_{s,k}(G)$ is not deterministic. Suppose in the augmented grammar, G', there are rightmost derivations,

$$S' \underset{rm}{\overset{*}{\Longrightarrow}} \alpha A w \underset{rm}{\Longrightarrow} \alpha\beta w$$

and

$$S' \underset{rm}{\overset{*}{\Longrightarrow}} \gamma B x \underset{rm}{\Longrightarrow} \gamma\delta x \qquad \text{such that} \qquad \gamma\delta x = \alpha'\beta y,$$

where $y \in T^*$, $|x| \leqslant |y|$, $\text{SUFF}_s(\alpha') = \text{SUFF}_s(\alpha)$, $\text{PREF}_k(w) = \text{PREF}_k(y)$ and $B \to \delta \neq A \to \beta$. Now if $x = y$, then $\gamma\delta = \alpha'\beta$ and β is a suffix of δ or vice versa. In either case $[B \to \delta. \mid \text{PREF}_k(y)]$ and $[A \to \beta. \mid \text{PREF}_k(y)]$ belong to $\mathcal{A}(\text{SUFF}(\gamma\delta))$; this follows because $\text{SUFF}_s(\alpha\beta) = \text{SUFF}_s(\text{SUFF}_s(\alpha)\beta) = \text{SUFF}_s(\text{SUFF}_s(\alpha')\beta) = \text{SUFF}_s(\alpha'\beta)$. Suppose $\delta = \theta\delta$. Then $[B \to \theta.\beta \mid \text{PREF}_k(y)] \in \mathcal{A}(\text{SUFF}_s(\gamma\theta)) = \mathcal{A}(\text{SUFF}_s(\alpha'))$ and $[A \to .\beta \mid \text{PREF}_k(w)] = [A \to .\beta \mid \text{PREF}_k(y)] \in \mathcal{A}(\text{SUFF}_s(\alpha))$. Since $\text{SUFF}_s(\alpha) = \text{SUFF}_s(\alpha')$, then condition (ii) of Definition 9 is violated and G is not SR(s, k). The argument assuming $\beta = \theta\delta$ is similar and for the case $x = y$ we have shown G not to be SR(s, k).

Consider the case, $|x| < |y|$. There are two subcases, $|x| < |y| \leqslant |\delta x|$ and $|x| < |y| > |\delta x|$. In the first of these subcases we assume $\delta = \delta_1\delta_2$ for some δ_1, δ_2 such that $\delta_2 \neq \varepsilon$ and $\gamma\delta_1 = \alpha'\beta$. It follows from the second derivation above that

$[B \to \delta_1 . \delta_2 \mid \text{PREF}_k(x)] \in \Lambda(\text{SUFF}_s(\gamma\delta_1))$. From the first derivation we also have, $[A \to \beta . \mid \text{PREF}_k(w)] \in \Lambda(\text{SUFF}_s(\alpha\beta))$. By relationships and assumption established earlier, we have that $[A \to \beta . \mid \text{PREF}_k(y)]$ belongs to $\Lambda(\text{SUFF}_s(\alpha'\beta)) = \Lambda(\text{SUFF}_s(\gamma\delta_1))$. Since $y = \delta_2 x$, $\text{EFF}_k(\delta_2 \text{PREF}_k(x)) = \text{PREF}_k(y)$. Thus $[B \to \delta_1 . \delta_2 \mid \text{PREF}_k(x)]$, $\delta_2 \neq \varepsilon$, and $[A \to \beta . \mid \text{PREF}_k(y)]$ in $\Lambda(\text{SUFF}_s(\gamma\delta_1))$ imply $\text{SHIFT}(\text{SUFF}_s(\gamma\delta_1)) \cap \text{REDUCE}(\text{SUFF}_s(\gamma\delta_1)) \neq \Phi$. This violates condition (i) of Definition 9.

If $|y| > |\delta x|$, then let $y = v\delta x$ for some $v \in T^+$. The second derivation can therefore be written

$$S \underset{rm}{\overset{*}{\Longrightarrow}} \gamma' Z x \underset{rm}{\Longrightarrow} \gamma' \theta_1 \theta_2 x' = \alpha'\beta\theta_2 x' \underset{rm}{\overset{*}{\Longrightarrow}} \alpha'\beta v B x \underset{rm}{\Longrightarrow} \alpha'\beta v\, \delta x = \alpha'\beta y.$$

From this we have, $[Z \to \theta_1 . \theta_2 \mid \text{PREF}_k(x')] \in \Lambda(\text{SUFF}_s(\alpha'\beta))$, where $\theta_2 \neq \varepsilon$ and $\text{PREF}_k(y) \in \text{EFF}_k(\theta_2 \text{PREFF}_k(x'))$. Since we have already established $[A \to \beta . \mid \text{PREF}_k(y)] \in \Lambda(\text{SUFF}_s(\alpha'\beta))$ it now follows that $\text{SHIFT}(\text{SUFF}_s(\alpha'\beta)) \cap \text{REDUCE}(\text{SUFF}_s(\alpha'\beta)) \neq \Phi$. Once again (i) of Definition 9 is violated and G is not SR(s, k). This concludes the proof.

COROLLARY. *If G is* SR(s, k), *then $\mathscr{P}_{s,k}(G)$ always halts.*

Proof. If G is SR(s, k), then by Theorem 3 G is (s, k)-bounded right-context and therefore LR(k). By an argument similar to the proof of Theorem 5.2 in [4], it follows that $\mathscr{P}_{s,k}(G)$ must halt on every input.

The converse of the previous theorem is not true as we shall shortly demonstrate. The reason is because shift-reduce conflicts arising in bounded right-context parsers can be resolved by examining s symbols below the handle whenever G is an (s, k)-bounded right-context grammar. Our next example illustrates this situation.

EXAMPLE 3. Let $G = (\{S, A, B\}, \{a, b, c\}, P, S)$, where $P = \{1\colon S \to aAc, 2\colon S \to bbB, 3\colon A \to bbc, 4\colon B \to bcc\}$. The states and action table for $\mathscr{P}_{1,1}(G)$ are shown below.

$\Lambda(\varepsilon)$	$\Lambda(b)$
$[S \to .aAc \mid \varepsilon]$	$[S \to b.bB \mid \varepsilon]$
$[S \to .bbB \mid \varepsilon]$	$[S \to bb.B \mid \varepsilon]$
	$[B \to .bcc \mid \varepsilon]$
$\Lambda(a)$	$[B \to b.cc \mid \varepsilon]$
$[S \to a.Ac \mid \varepsilon]$	$[A \to b.bc \mid c]$
$[A \to .bbc \mid c]$	$[A \to bb.c \mid c]$
$\Lambda(B)$	$\Lambda(c)$
$[S \to bbB. \mid \varepsilon]$	$[B \to bc.c \mid \varepsilon]$
	$[B \to bcc. \mid \varepsilon]$
$\Lambda(A)$	$[S \to aAc. \mid \varepsilon]$
$[S \to aA.c \mid \varepsilon]$	$[A \to bbc. \mid c]$

τ	a	b	c	ε
ε	shift	shift	error	error
a	error	shift	error	error
b	error	shift	shift	error
c	error	error	shift reduce 3	reduce 4 reduce 1
A	error	error	shift	error
B	error	error	error	reduce 2
S	error	error	error	accept

$$\text{SHIFT}(c) \cap \text{REDUCE}(c) \neq \Phi$$

G is not SR(1, 1), but is (1, 1) bounded right-context; the shift-reduce conflict when "c" appears on top of the stack can be resolved by looking one symbol below the handle "bbc."

Even though Example 3 denies the converse to Theorem 3, we can obtain a positive result by choosing a larger value for s. This is the substance of our next theorem.

THEOREM 4. *If G is an (m, n)-bounded right-context grammar and l is the length of the longest right-part of any production in the augmented grammar, G', then $\mathscr{S}_{s,n}(G)$ is deterministic, where $s = m + l$.*

Proof. We establish this result by contradiction; that is, we show that if any condition of Definition 9 is not met, then G cannot be (m, n)-bounded right-context.

Case 1. By definition of l it follows that $s \geqslant 1$. Suppose $l = 1$, then assuming G is BRC it follows that S does not appear in the right-part of any production of G. We can conclude $\Lambda(S) = \{[S' \to S. \mid \varepsilon]$ and $\tau(\Lambda(S), \varepsilon) = \{accept\}$. Assume $l > 1$ and $\tau(\Lambda(S), \varepsilon) \neq \{accept\}$. If $shift \in \tau(\Lambda(S), \varepsilon)$, then $n = 0$ and $[X \to \alpha S.\beta \mid u] \in \Lambda(S)$ for some $\beta \neq \varepsilon$. Since $s = m + l > 1$, we can conclude $\alpha = \varepsilon$ and $[X \to .S\beta \mid u] \in \Lambda(\varepsilon)$. This implies there exist derivations

$$S' \underset{rm}{\Longrightarrow} S$$

and

$$S' \overset{+}{\underset{rm}{\Longrightarrow}} Xz \underset{rm}{\Longrightarrow} S\beta z \overset{*}{\underset{rm}{\Longrightarrow}} Sy,$$

where $y \in T^*$. The existence of these derivations contradict the assumption that G is $(m, 0)$ BRC. Thus $shift \notin \tau(\Lambda(S), \varepsilon)$ if G is (m, n) BRC. Now suppose $reduce\ p \in \tau(\Lambda(S), \varepsilon)$ for some $p = X \to w$ in G. Since $s > 1$, it follows that $w = S$ or $w = \varepsilon$; that is, $[X \to S. \mid \varepsilon] \in \Lambda(S)$ or $[X \to . \mid \varepsilon] \in \Lambda(S)$. In the former case it follows that $S \Rightarrow^+ S$ in G implying G is ambiguous and therefore contradicting the assumption G

is (m, n) BRC. If $[X \to . \mid \varepsilon] \in \Lambda(S)$, then there exists an item, $I = [Y \to S.Z\delta \mid u] \in \Lambda(S)$, where Z is a nonterminal, $[X \to . \mid \varepsilon] \in \text{CLOSE}(I)$ and $\varepsilon \in \text{FIRST}_n(\delta u)$. This implies there exists derivations

$$S' \underset{rm}{\Longrightarrow} S$$

and

$$S' \overset{+}{\underset{rm}{\Longrightarrow}} Y \underset{rm}{\Longrightarrow} SZ\delta \overset{+}{\underset{rm}{\Longrightarrow}} SX \underset{rm}{\Longrightarrow} S$$

in G'. These derivations imply G is ambiguous and therefore not (m, n) BRC, a contradiction. Thus if G is (m, n) BRC, then *reduce* $p \notin \tau(\Lambda(S), \varepsilon)$. We conclude $\tau(\Lambda(S), \varepsilon) = \{accept\}$.

Case 2. Suppose (i) of Definition 9 is violated. Then for some $\gamma \in V^{*s}$, $[B \to \theta_1 . \theta_2 \mid v]$, $[A \to \beta. \mid u] \in \Lambda(\gamma)$, where $\theta_2 \neq \varepsilon$ and $u \in \text{EFF}_n(\theta_2 v)$. From this we have

$$S \overset{*}{\underset{rm}{\Longrightarrow}} \delta A w \underset{rm}{\Longrightarrow} \alpha\beta w$$

and

$$S \overset{*}{\underset{rm}{\Longrightarrow}} \lambda B x' \underset{rm}{\Longrightarrow} \lambda\theta_1\theta_2 x',$$

where $u = \text{PREF}_n(w)$, $v = \text{PREF}_n(x')$ and $\text{SUFF}_s(\lambda\theta_1) = \text{SUFF}_s(\alpha\beta) = \gamma$. Now if $\theta_2 \in T^+$, then $\lambda\theta_1\theta_2 x' = \alpha'\beta\theta_2 x'$ for some α' and $u = \text{EFF}_n(\theta_2 v) = \text{EFF}_n(\theta_2 x')$. In addition, since $s = m + l$, we have $\text{SUFF}_m(\alpha') = \text{SUFF}_m(\alpha)$. Thus we obtain a violation of the definition of (m, n)-bounded right-context. In the case θ_2 contains some nonterminal, then we can write,

$$S \overset{*}{\underset{rm}{\Longrightarrow}} \lambda B x' \underset{rm}{\Longrightarrow} \lambda\theta_1\theta_2 x' \overset{*}{\underset{rm}{\Longrightarrow}} \lambda\theta_1 v C x \underset{rm}{\Longrightarrow} \lambda\theta_1 v\delta x,$$

where $v\delta x \in T^*$.

Since $u \in \text{EFF}_n(\theta_2 v) = \text{EFF}_n(\theta_2\, \text{PREF}_n(x'))$ we assume $v\delta \neq \varepsilon$ and $u = \text{PREF}_n(v\delta x)$. Again, $\lambda\theta_1 = \alpha'\beta$ and $\text{SUFF}_{m+l}(\lambda\theta_1) = \text{SUFF}_{m+l}(\alpha'\beta) = \text{SUFF}_{m+l}(\alpha\beta)$ implies $\text{SUFF}_m(\alpha) = \text{SUFF}_m(\alpha')$. Letting $y = v\delta x$ we have $\lambda\theta_1 v\delta x = \alpha'\beta y$, where $|x| < |y|$, $\text{PREF}_n(y) = \text{PREF}_n(w)$ and $\text{SUFF}_m(\alpha) = \text{SUFF}_m(\alpha')$. Since $x \neq y$ we have a violation of the requirements for G to be (m, n)-bounded right-context.

Case 3. If (ii) of Definition 9 does not hold, then for some $\gamma \in V^{*s}$, $[A \to \beta. \mid u]$ and $[B \to \alpha\beta. \mid u] \in \Lambda(\gamma)$ where $(A \to \beta) \neq (B \to \alpha\beta)$ and $[A \to .\beta \mid u]$ and $[B \to \alpha.\beta \mid u]$ belong to $\Lambda(\gamma')$ for some γ'. Thus we can write

$$S \overset{*}{\underset{rm}{\Longrightarrow}} \theta A w \underset{rm}{\Longrightarrow} \theta\beta w,$$

$$S \overset{*}{\underset{rm}{\Longrightarrow}} \lambda B x \underset{rm}{\Longrightarrow} \lambda\alpha\beta x,$$

where $u = \mathrm{PREF}_n(w) = \mathrm{PREF}_n(x)$ and $\mathrm{SUFF}_s(\lambda a) = \mathrm{SUFF}_s(\theta)$. Since $\lambda Bx \neq \lambda aAx$ we have a violation of the condition for (m, n)-bounded right-context.

Thus if $\mathscr{P}_{s,n}(G)$ is not deterministic for $s = m + l$, then G is not (m, n)-bounded right-context. This concludes the proof.

We turn now to the relationship between extended weak-precedence grammars and SR grammars. In view of our results for bounded right-context grammars, the primary issue is not the direction of the inclusion relation between the classes, but rather the values of s and k required to establish the inclusion. Since $\mathrm{SR}(s, k)$ grammars need not be uniquely invertible nor restricted to nonerasing productions we would expect SR grammars to properly include the extended weak-precedence grammars. Indeed, this is the case. The definition we shall use for the extended precedence relations is that given in [1] and due to Wirth and Weber [8] and Gray [5].

Definition 11. Let $G = (N, T, P, S)$ be a reduced grammar with no ε-productions. For values of $m, n \geqslant 1$ the extended precedence relations, $\lessdot$, $\doteq$ and $\gtrdot$ are subsets of $(V \cup \{\$\})^m \times (V \cup \{\$\})^n$ and are defined as follows. Let

$$\$^m S \$^n \underset{rm}{\overset{*}{\Longrightarrow}} \theta A w \underset{rm}{\Longrightarrow} \theta \delta w$$

be any rightmost derivation in G. Then

(i) $\mathrm{SUFF}_m(\theta\delta) \gtrdot \mathrm{PREF}_n(w)$,

(ii) $\mathrm{SUFF}_m(\theta) \lessdot y$, where $y = \mathrm{PREF}_n(\delta w)$ or $y \in \mathrm{FIRST}_n(\delta w)$ if $\delta \in TV^*$,

(iii) for every δ_1, $\delta_2 \in V^+$ such that $\delta = \delta_1\, \delta_2$, $\mathrm{SUFF}_m(\theta\delta_1) \doteq y$, where $y = \mathrm{PREF}_n(\delta_2 w)$ or $y \in \mathrm{FIRST}_n(\delta_2 w)$ if $\delta_2 \in TV^*$.

Note that since G is ε-free, EFF_n could have been used in place of FIRST_n in the above definition. From Definition 11 it is easy to see that $[A \to \delta.\,|\, u]$ belongs to $A^G_{m,n}(\gamma)$ if and only if $\gamma' \gtrdot u''$, where $\gamma' = \mathrm{SUFF}_m(\$^m\gamma)$ and $u'' = \mathrm{PREF}_n(u\$^n)$. Relating items and (m, n)-consistency sets to the extended precedence relations is the subject of our next porposition. No formal proof is given as the result follows directly from Definitions 11, 2 and 3.

Proposition 2. *Let* $G = (N, T, P, S)$ *and* $\lessdot, \doteq, \gtrdot$ *be as described in Definition* 11. *For* $z \in V^{*m}$ *with* $|z| = i$ *we shall let* z' *denote* $\$^{m-i}z$; *similarily,* z'' *shall denotes* $z\$^{n-i}$ *whenever* $z \in V^{*n}$. *Then*

(i) *there is* $[A \to \delta.\,|\, u] \in A^G_{m,n}(\gamma)$ *if and only if* $\gamma' \gtrdot u''$,

(ii) $\gamma' \doteq y''$ *if and only if there is* $[A \to \alpha.\beta \,|\, v]$ *in* $A^G_{m,n}(\gamma)$ *with* $\alpha, \beta \neq \varepsilon$ *such that* $y = \mathrm{PREF}_n(\beta v)$ *when* $y \in NV^*$ *or* $y \in \mathrm{EFF}_n(\beta v)$ *when* $\beta \in TV^*$.

(iii) $\gamma' \lessdot y''$ *if and only if there is* $[A \to .\beta \,|\, v]$ *in* $A^G_{m,n}(\gamma)$ *such that* $y = \mathrm{PREF}_n(\beta v)$ *when* $\beta \in NV^*$ *or* $y \in \mathrm{EFF}_n(\beta v)$ *when* $\beta \in TV^*$.

Parsing algorithms employing the extended precedence relations use the $\gtrdot$ relation to locate the right-end of the handle in a bottom-up parse. The relation $\lessdot$ is used to isolate the left-end of handle in the parse stack. The handle can always be uniquely determined if the extended precedence relations are pairwise disjoint and the underlying grammar is uniquely invertible. The notion of "weak" precedence parsing was introduced by Ichibiah and Morse [6] and applies to a more general class of grammars by virtue of relaxing the requirement of disjointness between $\lessdot$ and $\doteq$. To uniquely determine the handle in this situation, the rule normally applied is to reduce by the production with the longest right part in instances where the unique invertibility property cannot be applied. The notion of weak precedence is generalized in our next definition.

DEFINITION 12. Let $G = (N, T, P, S)$ be a uniquely invertible proper[1] grammar. G is said to be *(m, n)-weak precedence* provided,

(i) $\gtrdot \cap (\doteq \cup \lessdot) = \Phi$, where $\lessdot, \doteq$ and $\gtrdot$ are the extended precedence relations and

(ii) if $p = A \to \alpha\beta$ and $B \to \beta$ are two productions with $\alpha \neq \varepsilon$ and $\$^m S \$^n \Rightarrow_{rm}^* \theta A w \Rightarrow_{rm} \theta\alpha\beta w$ is any rightmost derivation involving p, then $(\mathrm{SUFF}_m(\theta\alpha), z)$ in $(\doteq \cup \lessdot)$ implies $z \neq Bz'$ for any z'.

Condition (i) of the previous definition guarantees there are no conflicts between shift and reduce actions of extended weak-precedence parsers. In view of Proposition 2, this is consistent with condition (i) of Definition 9. Condition (ii) states that if two productions could be used to reduce the stack, then the production with the longest right-part should be used; this together with the unique invertibility property resolve all potential conflicts when a reduction is called for. Theorem 5 establishes that, indeed the weak-precedence conditions are sufficient to guarantee the canonical SR-parser is deterministic.

THEOREM 5. *Let $G = (N, T, P, S)$ be an (m, n)-weak precedence grammar. Then G is an* SR(m, n) *grammar; that is, $\mathscr{S}_{m,n}(G)$ is deterministic.*

Proof. We proceed by showing that if $\mathscr{S}_{m,n}(G)$ is not deterministic for some ε-free grammar, G, and some $m, n \geqslant 1$, then G is not (m, n)-weak precedence.

If condition (iii) of Definition 9 is not met, then by an argument similar to that given in the proof of Theorem 4, G can be shown to possess the cycle, $S \Rightarrow^+ S$, and thus G cannot be an (m, n)-weak precedence grammar. If condition (i) of Definition 9 is violated, then for some $\gamma \in V^{*m}$, $[A \to \delta . \mid u]$ and $[B \to \alpha.\beta \mid v]$ belong to $\Lambda_{m,n}^G(\gamma)$ such that $\beta \neq \varepsilon$ and $u \in \mathrm{EFF}_n(\beta v)$. By Proposition 2 we have $\gamma' \gtrdot u''$ and if $\beta \in TV^*$, then $\gamma' \doteq u''$ or $\gamma' \lessdot u''$. If $\beta \in NV^*$, then because G is ε-free and by definition of CLOSE there is $[C \to .\theta \mid x] \in \Lambda_{m,n}^G(\gamma)$ such that $\theta \in TV^*$ and $u \in \mathrm{EFF}_n(\theta x)$. Again, by Proposition 2 we have $\gamma' \lessdot u''$. Thus for either form of β we find $\gtrdot$ is not disjoint from $\lessdot \cup \doteq$ and hence G cannot be (m, n)-weak precedence. Finally, suppose (ii) of

[1] A grammar is proper if it is ε-free, reduced and cycle-free (see [1]).

Definition 9 is violated. Then there exist items $[A \to \alpha.\beta \mid u]$ and $[B \to .\beta \mid u]$ in $A^G_{m,n}(\gamma)$ for some γ in V^{*m}. $[B \to .\beta \mid u]$ in $A_{m,n}(\gamma)$ implies there exists an item $[Z \to \delta_1.B\delta_2 \mid v]$ in $A^G_{m,n}(\gamma)$ such that $u \in \mathrm{EFF}_n(\delta_2 v)$. From Definition 11 we conclude $\gamma' \doteq \mathrm{PREF}_n(B\delta_2 v)$. Thus (ii) of Definition 12 is violated and G is not (m, n)-weak precedence. This concludes the proof.

Our next example illustrates a uniquely invertible proper grammar that is SR(1, 1) but is not (1, 1)-weak precedence.

EXAMPLE 4. Let $G = (\{S, A, B\}, \{0, 1\}, P, S)$, where $P = \{1{:}\ S \to A1,\ 2{:}\ S \to 0B0,\ 3{:}\ A \to 01,\ 4{:}\ B \to 1\}$. The matrix of precedence relations for G is given below.

	A	B	0	1	\$
A				$\doteq$	
B			$\doteq$		
0		$\doteq$		$\doteq\ \lessdot$	$\gtrdot$
1			$\gtrdot$	$\gtrdot$	$\gtrdot$
\$	$\lessdot$		$\lessdot$		

G fails to be (1, 1)-weak precedence because it violates (ii) of Definition 12. This hold because of the productions $A \to 01$ and $B \to 1$ and the relation $0 \doteq B$.

The state sets, action and goto tables for the canonical SR(1, 1) parser are shown next.

$$A(\varepsilon) = \{[S' \to .S \mid \varepsilon], [S \to .A1 \mid \varepsilon], [S \to .0B0 \mid \varepsilon], [A \to .01 \mid 1]\},$$
$$A(S) = \{[S' \to S. \mid \varepsilon]\},$$
$$A(A) = \{[S \to A.1 \mid \varepsilon]\},$$
$$A(B) = \{[S \to 0B.0 \mid \varepsilon]\},$$
$$A(0) = \{[S \to 0.B0 \mid 0], [A \to 0.1 \mid 1], [B \to .1 \mid 0], [S \to 0B0. \mid \varepsilon]\},$$
$$A(1) = \{[S \to A1. \mid \varepsilon], [A \to 01. \mid 1], [B \to 1. \mid 0]\}.$$

τ	0	1	ε
ε	shift	error	error
S	error	error	accept
A	error	shift	error
B	shift	error	error
0	error	shift	reduce 2
1	reduce 4	reduce 3	reduce 1

Ψ	S	A	B	0	1
	S	A	*	0	*
	*	*	*	*	*
	*	*	*	*	1
	*	*	*	0	*
	*	*	B	*	1
	*	*	*	*	*

Clearly $\mathscr{P}_{1,1}(G)$ is deterministic. Observe that G fails to be (1, 1)-weak precedence parsable because the conflict between productions $3: A \to 01$ and $4: B \to 1$ can only be resolved by taking into account the lookahead symbol when "01" is on top of the stack. $\mathscr{P}_{1,1}(G)$ correctly reduces the stack in this situation.

As our final example, we illustrate an SR(1, 1) grammar that is not uniquely invertible and also contains an ε-production.

EXAMPLE 5. Consider $G = (\{S, A, B, C\}, \{0, 1\}, P, S)$, where $P = 1: S \to 0A$, $2: S \to 1BC$, $3: A \to 01$, $4: B \to 1$, $5: C \to 0$, $6: C \to \varepsilon$, $7: A \to 0\}$.

$\mathscr{A}(\varepsilon)$
$[S' \to .S \mid \varepsilon]$
$[S \to .0A \mid \varepsilon]$
$[S \to .1BC \mid \varepsilon]$

$\mathscr{A}(A)$
$[S \to 0A. \mid \varepsilon]$

$\mathscr{A}(B)$
$[S \to 1B.C \mid \varepsilon]$
$[C \to .01 \mid \varepsilon]$
$[C \to . \mid \varepsilon]$

$\mathscr{A}(C)$
$[S \to 1BC. \mid \varepsilon]$

$\mathscr{A}(0)$
$[S \to 0.A \mid \varepsilon]$
$[A \to .01 \mid \varepsilon]$
$[A \to .0 \mid \varepsilon]$
$[A \to 0.1 \mid \varepsilon]$
$[A \to 0. \mid \varepsilon]$
$[C \to 0. \mid \varepsilon]$

$\mathscr{A}(1)$
$[S \to 1.BC \mid \varepsilon]$
$[B \to .1 \mid 0\varepsilon]$
$[A \to 01. \mid \varepsilon]$
$[B \to 1. \mid 0\varepsilon]$

$\mathscr{A}(S)$
$[S' \to S. \mid \varepsilon]$

τ	0	1	ε
ε	shift	shift	error
S	error	error	accept
A	error	error	reduce 1
B	shift	error	reduce 6
C	error	error	reduce 2
0	shift	shift	reduce 5 reduce 7
1	reduce 4	shift	reduce 3 reduce 4

There are clearly no shift–reduce conflicts or conflicts with *accept* in $\tau(S, \varepsilon)$. The reduce conflicts in $\tau(0, \varepsilon)$ are resolved by observed that $[C \to .0 \mid \varepsilon]$ belongs to $\mathscr{A}(B)$

while $[A \to .0 \mid \varepsilon]$ belongs to $\Lambda(0)$. Similarly, the conflicts in $\tau(1, \varepsilon)$ resolve by observing $[B \to .1 \mid \varepsilon]$ belongs to $\Lambda(1)$ while $[A \to 0.1 \mid \varepsilon]$ belongs to $\Lambda(0)$. Thus $\mathscr{P}_{1,1}(G)$ satisfies Definition 12 and G is an SR(1, 1) grammar.

We conclude this section with our final result.

COROLLARY. *The class of* SR(s, k) *grammars properly includes the* (s, k)*-weak precedence grammars.*

V. PRACTICAL CONSIDERATIONS

One of the advantages of the conceptual framework surrounding SR(s, k) parsing is that it brings into sharper focus the trade-offs between precedence and bounded-context parsing. To clarify this point consider an SR(s, k) grammar. If the underlying grammar is also an (s, k)-weak precedence grammar, then stack reductions can be made by examining at most r stack symbols, where r is the length of the handle; this is normally done by comparing right-parts with the top of the stack in descending order by length. On the other hand suppose the grammar is SR(s, k) but not (s, k)-weak precedence. In this case the reduction procedure described above is not sufficient; at least one symbol below the handle must be examined to resolve conflicts like those found in $\tau(0, \varepsilon)$ or $\tau(1, \varepsilon)$ of Example 5. Thus the canonical SR(s, k) parser must be augmented with additional tables and/or logic to permit the resolution of all reduction conflicts. Enlarging the class grammars that can be parsed deterministically from (s, k)-weak precedence to SR(s, k) carries with it a penalty of space and time in almost direct proportion to the number of conflicts that must be resolved by examining left-context of the handle.

Concerning the implementation of SR(s, k) parsers there are one or two points we wish to emphasize. First of all, the action and goto tables frequently can be made significantly smaller by merging rows that are "similar" in their action structure; that is, merging rows that result only in additional reduction conflicts. Although row merger often produces a savings in space it may not always be desirable because of the additional overhead involved in resolving new conflicts and/or because of induced delays in error detection due to the destruction of some *error* entries in the goto-table. Finally, we observe that a more practical definition of determinism for SR(s, k) parsers obtains if (ii) of Definition 9 replaced by

(ii′) If distinct items, $[A \to \alpha. \mid u]$ and $[B \to \beta\alpha. \mid u]$ belong to $\Lambda_{s,k}(\gamma)$, then for some $m \geqslant s$ it holds that not both items, $[A \to .\alpha \mid u]$ and $[B \to \beta.\alpha \mid u]$ belong to $\Lambda_{m,k}(\gamma')$, $\gamma' \in V^{*m}$.

This change suggests parsing tables should be constructed using the smallest value of s for which (i) and (iii) are satisfied provided all reduction conflicts can be resolved by examining bounded left-context of the handle. In effect, this approach yields a mixed strategy type of parser.

V. Conclusion

We have presented here an algorithm for constructing efficient bounded-context and precedence parsers using the same conceptual framework employed by Knuth [7], Early [2] and Geller and Harrison [4]. Our work has added one more part to an emerging unified theory of bottom-up parsing.

Acknowledgment

I would like to express my gratitude to the referees for their many helpful suggestions and comments. In particular, the notion of "k-cover" greatly simplified the development of SR(s, k) parsers used in the original manuscripts. I appreciate their contribution.

References

1. A. V. Aho and J. D. Ullman, "The Theory of Parsing, Translation and Compiling, Vol. 1," Prentice–Hall, Englewood Cliffs, N.J., 1972.
2. J. Earley, An efficient context-free parsing algorithm, *Comm. ACM* **13**, 2 (1970), 94–102.
3. R. W. Floyd, Bounded context syntactic analysis, *Comm. ACM* **7**, 2 (1964), 62–67.
4. M. Geller and M. A. Harrison, Characteristic parsing: A framework for producing compact deterministic parsers, I and II, *J. Comput. System Sci.* **14** (1977), 265–317 and 318–343.
5. J. N. Gray, "Precedence Parsers for Programming Languages," Phd. Thesis, University of California, Berkeley, 1909.
6. J. D. Ichbiah and S. P. Morse, A technique for generating almost optimal Floyd–Evans productions for precedence grammars, *Comm. ACM* **13**, 8 (1970), 501–508.
7. D. E. Knuth, On the translation of languages from left to right, *Inform. Contr.* **8**, 6 (1965), 607–639.
8. N. Wirth and H. Weber, Euler-a generalization of ALGOL and its formal definition, Parts 1 and 2, *Comm. ACM* **9** (1966), 13–23 and 9, 2 (1966), 89–99.