5. Parsing

parser

language recognizer

check if syntactically correct

text transformer

produce internal representation

parse tree, rule string(parse)

pushdown automaton

homomorphism: sentence → left / right parse

5.1 Pushdown Automaton

$M = (Q \cup \Sigma \cup \{\$, \mid\}, \Gamma)$ *is a rewriting system.*

$M=(Q, \Sigma, \Gamma, \gamma_s, F, \$, \mid)$ *is a* **pushdown automaton**

(1) ***stack alphabet*** Q,

(2) ***input alphabet*** Σ *where* $Q \cap \Sigma \neq \emptyset$,

(4) ***initial stack content*** $\gamma_s \in Q^*$,

(5) ***set of final stack contents*** $F \subseteq Q^*$,

(6) ***end marker*** $\$ \notin Q \cup \Sigma$,

(7) ***delimiter*** $\mid \notin Q \cup \Sigma$, *and*

(3) ***set of actions*** Γ,

action $\xi \in \Gamma$*: a pair of configuration strings*

$$\xi = \alpha \mid xy \rightarrow \beta \mid y \in \Gamma,$$

$$\alpha, \beta \in Q^*, x, y \in \Sigma^*.$$

configuration(instantaneous description) *of M*

$\$\gamma \mid w\$$,

$\gamma \in Q^*$ ***stack string***

$\$\gamma:1$ ***stack topmost symbol***

$w \in \Sigma^*$ ***remaining input string***

$1:w\$$ ***current input symbol***

Let $\$\delta\alpha \mid xyz\$$ *be a configuration. Then*

$\$\delta\underline{\alpha} \mid \underline{xy}z\$ \Rightarrow \$\delta\underline{\beta} \mid \underline{y}z\$$, *iff* $\alpha \mid xy \rightarrow \beta \mid y \in \Gamma$.

$\$\gamma_s \mid w\$$ ***initial configuration*** *for w*

$\$\gamma \mid \$,\ \gamma \in F$ ***final configuration***

no applicable action ***error configuration***

A ***computation****(or* ***process****) of M on w*

<u>any derivation</u> in M <u>from initial configuration</u> for w

accepting computation

a computation <u>ends</u> with <u>final configuration</u>.

nonaccepting computation

a computation <u>ends</u> with <u>error configuration</u>.

M ***accepts*** *w, if it has an <u>accepting computation</u> on w.*

M ***halts correctly*** *on w, if it accepts w.*

M ***halts incorrectly*** *on w,*

if <u>every</u> computation on w is <u>nonaccepting</u>.

M ***loops forever*** *on w, otherwise.*

i.e., if it has an <u>arbitrary long computation</u> on w but <u>no</u> accepting computation.

***language accepted(recognized**, or **described**) by M*

$L(M) = \{w \in \Sigma^* \mid \$\gamma_s \,|\, w\$ \Rightarrow^* \$\gamma \,|\, \$,\ \gamma \in F\}.$

*A pushdown automaton M is **ambiguous**,*
*if **two** accepting computations on some sentence*
*M is **unambiguous**, otherwise.*

*M is **nondeterministic**, if it has some computation*
*to which **two** actions are applicable.*
*M is **deterministic**, if it is not nondeterministic.*

***Fact 5.1** M is nondeterministic if and only if it has distinct actions*

$\alpha \,|\, x \to \alpha' \,|\, x',\ \beta \,|\, y \to \beta' \,|\, y' \in \Gamma,$

$\alpha{:}k = \beta$, *or* $\alpha = \beta{:}k$; *and* $l{:}x = y$ *or* $x = l{:}y$.

*one of α, β is **suffix** of other, and*
*one of x, y is **prefix** of other,*

*provided that M is **useful**(?).*

***Fact 5.2** Any **deterministic pushdown automaton** is **unambiguous** provided that <u>no action</u> is applicable to any of the <u>accepting configurations</u>.*

dpda ⇒ unambiguous

no action on accepting configuration

Fact* 5.3** *Any fa M can be transformed in time O(|M|) into an equivalent pda M' which is* **unambiguous(respectively* **deterministic***), iff M is. Also M' has* **bounded stack***(stack contents of every configuration in any computation of M' consists of single symbol only).*

Proof $\Gamma = \{q \mid x \to p \mid \mid qx \to p\}$.

pda: at least as descriptive and succinct as fa

Proposition 5.4 *There exist a constant c > 0 and an infinite sequence of regular languages L_1, L_2, … such that each L_n is accepted by some deterministic pda of size $O(n^3)$ but any fa accepting L_n must have size at least $O(2^n)$.*

dpda: exponentially more succinct than nfa

A pda is **normal-form** *if its action are of forms:*

$\alpha \mid x \rightarrow \beta \mid$ *where* $|\alpha| \leq 2$, $|x| \leq 1$.

Proposition 5.5 *Any pda M can be transformed into an equivalent* **normal-form** *pda M'. Moreover M' is* ***unambiguous****(respectively* **deterministic***), iff M is.*

Let $G = (N, \Sigma, P, S)$ *be a context-free grammar.*
The **predictive machine** *for G is a pda* $M_P{}^G(N \cup \Sigma, \Sigma, \Gamma_P, S, \{\varepsilon\}, \$, \mid)$*, where* Γ_P *are of form:*

(pa) $A \mid \rightarrow \omega^R \mid$ $A \rightarrow \omega \in P$, "**produce** *A to* ω",
(sa) $a \mid a \rightarrow \mid$ $a \in \Sigma$, "**shift** $a \in \Sigma$".

stack content is the **prediction** *of remaining input*

$\$\gamma \mid w\$$ $\gamma^R \Rightarrow^* w$

$A \mid \rightarrow \omega^R \mid$ ***guessing*** *A to* ω, $A \rightarrow \omega \in P$
$a \mid a \rightarrow \mid$ ***verifying*** a, $a \in \Sigma$

initial configuration $\$S \mid w\$$

initial prediction S: S derives w

final configuration $\$ \mid \$$

final prediction ε*: remaining input is empty*

Lemma 5.6 *For any G, the language accepted by the* ***predictive*** *machine M for G is the language generated by G. Moreover for any sentence, there is a* ***bijective*** *correspondence between* **leftmost** *derivation in G and accepting computation of predictive machine M on w.*

Theorem 5.7 *Any grammar G can be transformed in time O(|G|) into an equivalent* **normal-form** *pda M. Moreover M is* **unambiguous***, iff G is.*

Proposition 5.8 *Any normal-form pda M can be transformed in time* $O(|M| \cdot |V|^3)$ *into an equivalent* **context-free** *grammar G. Moreover G is* **unambiguous***, iff M is.*

Theorem 5.9 *Any language is* **context-free** *iff it is the language accepted by some* **pda***.*

$\exists$***inherently ambiguous context-free languages***
$\therefore \nexists$***algorithm:*** *pda* $\rightarrow$ **unambiguous** *pda*

A language is **deterministic***, if it is accepted by some* <u>*deterministic*</u> *machine.*

L_{match} *is deterministic.(p157)*

$L_{pal} = \{w \in \{0, 1\}^* | w^R = w\}$ ***palindromes***

$G_{pal} = (\{S\}, \{0, 1\}, \{S \rightarrow \varepsilon \mid 0 \mid 1 \mid 0S0 \mid 1S1\}, S).$

guessing center $S \rightarrow \varepsilon$ *even length*

$S \rightarrow 0 \mid 1$ *odd length*

no pda can guess center of palindrome deterministically

Proposition 5.10 L_{pal} *is* ***not deterministic****, but*

$L_{cpal} = \{wcw^R | w \in \{0, 1\}^*\}$ *is deterministic*

palindromes *with center marker c*

Theorem 5.11

(Characterization of Context-free Languages)

The following statements are logically equivalent.

(1) L is the language generated by some ***context-free*** *grammar.*

(2) L is the language generated by some ***canonical two-form*** *grammar.*

(3) L is the language generated by some ***Chomsky normal-form*** *grammar.*

(4) L is the language accepted by some ***pda****.*

(5) L is the language accepted by some ***normal-form pda****.*

Moreover, the descriptions of L can be transformed into an equivalent descriptions to the other classes.

Let $M = (Q, \Sigma, \Gamma, \gamma_s, F, \$, \vert)$. Then

$Time_M(w) = min\{Time_M(\$\gamma_s \vert w\$, \gamma \vert \$) \mid \gamma \in F\}$.

$Space_M(w) = min\{Space_M(\$\gamma_s \vert w\$, \gamma \vert \$) \mid \gamma \in F\}$.

where $Time_M(\phi_1, \phi_2)/Space_M(\phi_1, \phi_2)$ denote the time/space complexity for deriving ϕ_2 from ϕ_1 in M.

time/space complexity *of accepting w in M.*

$Time_M(w)/Space_M(w)$

M accepts w in time t, if $t \geq Time_M(w)$.

M accepts w in space s, if $s \geq Space_M(w)$.

M runs in time T(n)(respectively in space S(n), if M accepts every sentence of length n in time T(n) (respectively in space S(n)).

For example, for M_{match}

$Time_{M_{match}}(0^n 1^n) = 3n + 1$

$Space_{M_{match}}(0^n 1^n) = 2n + 4$

M_{match} runs in time 3n + 1 and in space 2n + 4.

5.2 Left and Right Parsers

left parse *of w in G:* $\pi_l \in P^*$.

$S \Rightarrow_{lm}^{\pi_l} w$ *in G*

right parse *of w in G:* $\pi_r^R \in P^*$.

$S \Rightarrow_{rm}^{\pi_r} w$ *in G* *Note that* ***right parse*** π_r^R *is the* ***reveral*** *of* π_r*(bottom-up parsing) by* ***definition****!*

left(right) parser *for a grammar G*

A (possibly nondeterministic) RAM program

recognizes the sentence in L(G)

produces <u>at least one</u> left(right) parse

for each sentence

left parse *leftmost derivation*

top-down *manner*

right parse ***reversal*** *of rightmost derivation*

bottom-up *manner*

M is a ***pushdown transducer*** *with* ***output alphabet*** Δ *and* ***output effect*** τ*, written* (M, τ)*, if M is a pda and* τ *is a homomorphism from* Γ^* *to* Δ^* *where* Γ *is a set of action of M.*

$M = (Q, \Sigma, \Gamma, \gamma_s, F)$	*fa*
$M = (Q, \Sigma, \Gamma, \Delta, \tau, \gamma_s, F)$	*pda*
(M, τ)	*pdt*

Let $\theta \in \Gamma^*$ be an action string of sentence w. Then pushdown transducer (M, τ) **produce output** *σ for w, if $\tau(\theta) = \sigma \in \Delta^*$.*

A pdt $M = (Q, \Sigma, \Gamma, \Delta, \tau, \gamma_s, F)$ is a **left/right parser** *for a grammar $G = (N, \Sigma', P, S)$, if*

- *(1) $\Sigma = \Sigma'$,*
- *(2) $L(M) = L(G)$,*
- *(3) $\Delta = P$,*
- *(4) $\forall \pi \in P^*$, π: left/right parse of G.*

Let $G = (N, \Sigma, P, S)$ be a context-free grammar.
$M = (Q, \Sigma, \Gamma, P, \tau, \gamma_s, F)$ is a left/right parser for G.
or (M, τ) is a left/right parser for G in short.

produce-shift parser *$(M(V, \Sigma, \Gamma, S, \{\varepsilon\}, \$, |), \tau)$*

- *(pa) $\tau(A \mid \rightarrow \omega^R \mid) = A \rightarrow \omega \in P$.*
- *(sa) $\tau(a \mid a \rightarrow \mid) = \varepsilon,\ a \in \Sigma$.*

Proposition

Produce-shift parser is indeed a left parser.

- ***Lemma** 5.12, 5.13*
  - *produce-shift parser $\Rightarrow$ linear left parser*
- ***Lemma** 5.14, 5.15*
  - *produce-shift parser $\Leftarrow$ linear left parser*
- ***Theorem** 5.16*
  - *produce-shift parser $\Leftrightarrow$ linear left parser*

Lemma 5.12 *Let $G = (N, \Sigma, P, S)$ be a grammar and $M = (V, \Sigma, \Gamma, P, \tau, S, \{\varepsilon\})$ be a* **produce-shift** *parser.*

If $\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$,\ \theta \in \Gamma^*$ *in M, then*

$$\gamma^R \underset{lm}{\overset{\tau(\theta)}{\Rightarrow}} x\delta^R \text{ in } G, \text{ and } |\theta| = |\tau(\theta)| + |x|.$$

Proof *Induction on the length of action string* θ.

i) $\theta = \varepsilon.\ \gamma = \delta,\ x = \varepsilon$, and $\tau(\varepsilon) = \varepsilon$.

ii) $\theta = r\theta'$.

ii.1) $r = A \mid \rightarrow \omega^R \mid \in \Gamma \qquad \theta = (A \mid \rightarrow \omega^R \mid)\cdot\theta'$

$$\$\gamma \mid xy\$ = \$\gamma' A \mid xy\$ \overset{r}{\Rightarrow} \$\gamma'\omega^R \mid xy\$ \overset{\theta'}{\Rightarrow} \$\delta \mid y\$$$

$$(\gamma'\omega^R)^R \underset{lm}{\overset{\tau(\theta')}{\Rightarrow}} x\delta^R \text{ in } G,\ |\theta'| = |\tau(\theta')| + |x| \text{ IH.}$$

$$\gamma^R = (\gamma' A)^R = A\gamma'^R \underset{lm}{\overset{A\rightarrow\omega}{\Rightarrow}} \omega\gamma'^R = (\gamma'\omega^R)^R.$$

$$\therefore \gamma^R \underset{lm}{\overset{\tau(r)\,\tau(\theta')}{\Rightarrow}} (= \underset{lm}{\overset{\tau(\theta)}{\Rightarrow}})\ x\delta^R \text{ in } G, \text{ and}$$

$$|\theta| = 1 + |\theta'| =_{IH} 1 + |\tau(\theta')| + |x| = |\tau(\theta)| + |x|.$$

ii.2) $r = a \mid a \rightarrow \ \mid \in \Gamma \qquad \theta = (a \mid a \rightarrow \ \mid)\cdot\theta'$

$$\$\gamma \mid xy\$ = \$\gamma' a \mid ax'y \overset{r}{\Rightarrow} \$\gamma' \mid x'y \overset{\theta'}{\Rightarrow} \$\delta \mid y\$$$

$$\gamma'^R \underset{lm}{\overset{\tau(\theta')}{\Rightarrow}} x'\delta^R \text{ in } G,\ |\theta'| = |\tau(\theta')| + |x'| \text{ IH.}$$

$$\gamma^R = (\gamma' a)^R = a\gamma'^R.$$

$$\therefore \gamma^R \underset{lm}{\overset{\tau(r)\cdot\tau(\theta')}{\Rightarrow}} (= \underset{lm}{\overset{\tau(\theta')}{\Rightarrow}})\ x\delta^R \text{ in } G, \text{ and}$$

$$|\theta| = 1 + |\theta'| =_{IH} |\tau(\theta')| + 1 + |x'| = |\tau(\theta)| + |x|$$

Lemma 5.13 *Let M be a* ***produce-shift*** *parser for G.*

(1) $L(M) \subseteq L(G)$,

(2) $^{\forall}\theta$: actions in M, $\tau(\theta)$ is a **left parse** of w,

(3) $Time_G(w) \le Time_M(w) - |w|$.

Lemma 5.14 *Let* $G = (N, \Sigma, P, S)$ *be a grammar and* $M = (V, \Sigma, \Gamma, P, \tau, S, \{\varepsilon\})$ *be a produce-shift parser.*

If $\gamma^R \underset{lm}{\overset{\pi}{\Rightarrow}} x\delta^R$ in G, $\delta^R = \varepsilon$ or $1{:}\delta^R \in N$, then

$$\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$, \ \theta \in \Gamma^* \text{ and}$$

$$\tau(\theta) = \pi, \ |\theta| = |\pi| + |x|.$$

Proof *Induction on the length of rule string* π.

i) $\pi = \varepsilon$. $\gamma^R = x\delta^R$.

$$\$\gamma \mid xy\$ = \$\delta x^R \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$, \text{ and}$$

$$|\theta| = |x|, \ \tau(\theta) = \pi = \varepsilon.$$

ii) $\pi = \pi' \cdot A \rightarrow \omega$, $\pi' \in P^*$, $^{\exists}\theta'$, $\tau(\theta') = \pi'$.

$$\gamma^R \underset{lm}{\overset{\pi'}{\Rightarrow}} x'\delta'^R = x'A\delta'' \underset{lm}{\overset{A\rightarrow\omega}{\Rightarrow}} x'\omega\delta'' = x'z\delta^R = x\delta^R,$$

where $|\theta'| = |\pi'| + |x'|$, $\delta'^R = A\delta''$, $\omega\delta'' = z\delta^R$, $x'z = x$

$$\therefore \ \$\gamma \mid xy\$ = \$\gamma \mid x'zy\$ \overset{\theta'}{\Rightarrow} \$\delta' \mid zy\$ = \$\delta''^R A \mid zy\$$$

$$\overset{r}{\Rightarrow} \$\delta''^R\omega^R \mid zy\$ = \$(\omega\delta'')^R \mid zy\$ = \$\delta z^R \mid zy\$$$

$$\overset{\theta''}{\Rightarrow} \$\delta \mid y\$, \text{ where } |\theta''| = |z|, \ \tau(\theta'') = \varepsilon.$$

$$\therefore \ |\theta| = |\theta'| + 1 + |\theta''| = |\pi'| + |x'| + 1 + |z| = |\pi| + |x|$$

$$\tau(\theta) = \pi' \cdot A \rightarrow \omega \cdot \varepsilon = \pi' \cdot A \rightarrow \omega = \pi.$$

Lemma 5.15 *Let M be a* **produce-shift** *parser for G.*

(1) L(G) $\subseteq$ L(M),

(2) $^\forall\pi$: ***left parse*** *of w in G, $\tau(\theta) = \pi$ in M,*

(3) $Time_M(w) \leq Time_G(w) + |w|$.

Theorem 5.16 *Let M be a* **produce-shift parser** *M for G. Then*

(1) M is a **left parser** *for G.*

(2) $^\forall w \in L(G)$, M produces ***all left parses*** *of w.*

(3) $Time_M(w) = Time_G(w) + |w|$.

produce-shift parser *runs in time* ***linear in the length of the sentence****.*

Let $G = (N, \Sigma, P, S)$ be a context-free grammar.
The **shift-reduce parser** *(M, τ) for G is with $M(V, \Sigma, \Gamma, \varepsilon, \{S\}, \$, \mid)$,*

(ra) $\omega \mid \rightarrow A \mid \in \Gamma$ "**reduce** by $A \rightarrow \omega \in P$"
(sa) $\mid a \rightarrow a \mid \in \Gamma$ "**shift** $a \in \Sigma$"

$\tau(\omega \mid \rightarrow A \mid) = A \rightarrow \omega$,
$\tau(\mid a \rightarrow a \mid) = \varepsilon$.

Theorem 5.21 *Let M be a* **shift-reduce parser** *M for G. Then*

(1) *M is a* **right parser** *for G.*
(2) $^{\forall} w \in L(G)$, *M produces* **all right parses** *of w.*
(3) $Time_M(w) = Time_G(w) + |w|$.

shift-reduce parser *runs in time linear in the length of the sentence.*

Lemma *5.17, 5.18*
shift-reduce parser $\Rightarrow$ linear right parser
Lemma *5.19, 5.20*
shift-reduce parser $\Leftarrow$ linear right parser
Theorem *5.21*
shift-reduce parser $\Leftrightarrow$ linear right parser

Lemma 5.17 *Let $G = (N, \Sigma, P, S)$ be a grammar and $M = (V, \Sigma, \Gamma, P, \tau, \{\varepsilon\}, S)$ be a* **shift-reduce** *parser. If $\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$$, $\theta \in \Gamma^*$ in M, then*

$$\delta \underset{rm}{\overset{\tau(\theta)^R}{\Rightarrow}} \gamma x \text{ in } G, \text{ and } |\theta| = |\tau(\theta)| + |x|.$$

Proof *Induction on the length of action string $\theta \in \Gamma^*$.*

i) $\theta = \varepsilon$. $x = \varepsilon$, $\gamma = \delta$, and $\tau(\varepsilon) = \varepsilon$.

ii) $\theta = r\theta'$.

ii.1) $r = \omega \mid \rightarrow A \mid \in \Gamma$

$$\$\gamma \mid xy\$ = \$\gamma''\omega \mid xy\$ \overset{r}{\Rightarrow} \$\gamma''A \mid xy\$ = \$\gamma' \mid xy\$$$

$$\overset{\theta'}{\Rightarrow} \$\delta \mid y\$, \text{ and } |\theta'| = |\tau(\theta')| + |x|.$$

$$\delta \underset{rm}{\overset{\tau(\theta')^R}{\Rightarrow}} \gamma' x = \gamma'' A x \underset{rm}{\overset{A \rightarrow \omega}{\Rightarrow}} \gamma''\omega x = \gamma x.$$

$$\therefore \delta \underset{rm}{\overset{(\tau(r\theta'))^R}{\Rightarrow}} \gamma x \text{ in } G, \text{ and}$$

$$|\theta| = 1 + |\theta'| = 1 + |\tau(\theta')| + |x| = |\tau(\theta)| + |x|.$$

ii.2) $r = \mid a \rightarrow a \mid \in \Gamma$

$$\$\gamma \mid xy\$ = \$\gamma \mid ax'y\$ \overset{r}{\Rightarrow} \$\gamma a \mid x'y\$ = \$\gamma' \mid x'y\$$$

$$\overset{\theta'}{\Rightarrow} \$\delta \mid y\$, \text{ and } |\theta'| = |\tau(\theta')| + |x'|.$$

$$\delta \underset{rm}{\overset{\tau(\theta')^R}{\Rightarrow}} \gamma' x' = \gamma a x' = \gamma x$$

$$\therefore \delta \underset{rm}{\overset{(\varepsilon \cdot \tau(\theta'))^R}{\Rightarrow}} \gamma x \text{ in } G, \text{ and}$$

$$|\theta| = 1 + |\theta'| = |\tau(\theta')| + 1 + |x'| = |\tau(\theta)| + |x|$$

Lemma 5.18 *Let M be a* **shift-reduce** *parser for G.*

(1) $L(M) \subseteq L(G)$,

(2) $^{\forall}\theta$: actions in M, $\tau(\theta)$ is a **right parse** *of w,*

(3) $Time_G(w) \le Time_M(w) - |w|$.

Lemma 5.19 *Let M be a* **shift-reduce** *parser for G. If*

$\delta \overset{\pi^R}{\underset{rm}{\Rightarrow}} \gamma x$ *in G, $\gamma = \varepsilon$ or $\gamma{:}1 \in N$, then*

$\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$$, $\theta \in \Gamma^*$ *and*

$\tau(\theta) = \pi$, $|\theta| = |\pi| + |x|$.

Proof *Induction on the length of rule string $\pi \in P^*$.*

i) $\pi = \varepsilon$. $\delta = \gamma x$.

$\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\gamma x \mid y\$ = \$\delta \mid y\$$, *and*

$|\theta| = |x|$, $\tau(\theta) = \pi = \varepsilon$.

ii) $\pi = A\to\omega\cdot\pi'$, $\pi' \ne \varepsilon$, $^{\exists}\theta'$, $\tau(\theta') = \pi'$.

$\delta \overset{\pi'^R}{\underset{rm}{\Rightarrow}} \gamma' x' = \delta'' A x' \overset{A\to\omega}{\underset{rm}{\Rightarrow}} \delta''\omega x' = \gamma x$

where $|\theta'|=|\pi'|+|x'|$, $\gamma'=\delta''A$, $\delta''\omega=\gamma z$, $zx'=x$

$\therefore\ \$\gamma \mid xy\$ = \$\gamma \mid zx'y\$ \overset{\theta''}{\Rightarrow} \$\gamma z \mid x'y\$ = \$\delta''\omega \mid x'y\$$

$\overset{r}{\Rightarrow} \$\delta'' A \mid x'y\$ = \$\gamma' \mid x'y\$ \overset{\theta'}{\Rightarrow} \$\delta \mid y\$$.

where $|\theta''| = |z|$, $\tau(\theta'') = \varepsilon$.

$\therefore\ |\theta| = |\theta'|+1+|\theta''| = |\pi'|+|x'|+1+|z| = |\pi| + |x|$.

Lemma 5.20 *Let M be a* **shift-reduce** *parser for G.*

(1) $L(G) \subseteq L(M)$,

(2) $^{\forall}\pi$: **right parse** *of w in G, $\tau(\theta) = \pi$ in M,*

(3) $Time_M(w) \le Time_G(w) + |w|$.

produce-shift parser

$A \mid \to \omega_1^R \mid$ or $A \mid \to \omega_2^R \mid$ *produce-produce conf.*

$A \mid \to \omega^R \mid$ or $a \mid a \to \mid$ *no produce-shift conf.*

$a \mid a \to \mid$ or $b \mid b \to \mid$ *no shift-shift conf*

A grammar $G = (N, \Sigma, P, S)$ is in ***Greibach normal-form****, if its rules are of forms:*

(1) $A \to a\beta,\ a \in \Sigma,\ \beta \in (V \setminus \{S\})^*$ where $V = N \cup \Sigma$,

(2) $S \to \varepsilon$.

	one symbol lookahead
$A \mid \to \beta^R a \mid$	$A \mid a \to \beta^R a \mid a$
$S \mid \to \mid$	$S \mid \$ \to \mid \$$
$a \mid a \to \mid$	$a \mid a \to \mid$

A Greibach normal-form grammar has no pair of distinct rules of the form,

$A \to a\beta_1,\ A \to a\beta_2 \in P \Rightarrow \beta_1 = \beta_2.$

is called ***simple*** *grammar(or* ***s****-grammar)*

Predictive parser with one symbols lookahead for ***simple*** *grammar is* ***deterministic****!*

shift-reduce parser

$\mid a \to a \mid$ or $\omega \mid \to A \mid$ *shift-reduce conflict*

$\omega_1 \mid \to A_1 \mid$ or $\omega_2 \mid \to A_2$ *reduce-reduce conflict*

$\mid a \to a \mid$ or $\mid b \to b \mid$ *no shift-shift conf.*

Let G = (N, Σ, P, S) be a context-free grammar. Then **produce-shift***(or* **guess-verify***,* **predictive***;* **top-down) parser** *for G is a pdt $M_P(V, \Sigma, \Gamma_P, P, \tau, S, \{\varepsilon\}, \$, |\}$, where Γ_P and τ are of form:*

(pa) $A | \to \omega^R | \in \Gamma_P$ "**produce** *by* $A \to \omega \in P$",

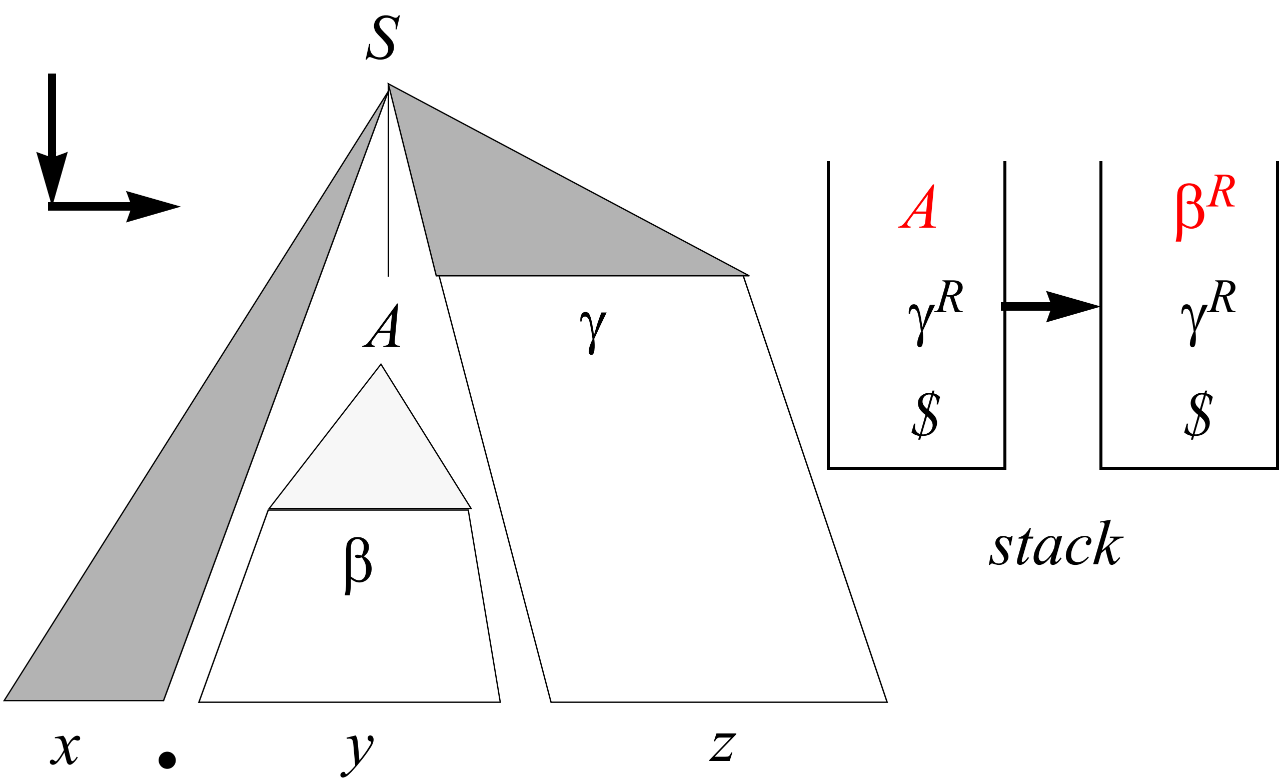
$\tau(A | \to \omega^R |) = A \to \omega$.

(sa) $a | a \to | \in \Gamma_P$ "**shift** $a \in \Sigma$",

$\tau(a | a \to |) = \varepsilon$.

$$S \Rightarrow_{lm}^{*} xA\gamma \Rightarrow_{lm}^{A\to\beta} x\beta\gamma \Rightarrow_{lm}^{*} xyz.$$

$$\$S | xyz\$ \Rightarrow^{*} \$\gamma^R A | yz\$ \Rightarrow_{p}^{A\to\beta} \$\gamma^R\beta^R | yz\$ \Rightarrow^{*} \$ | \$.$$



The **shift-reduce** *(or* **bottom-up***)* **parser** *for G is a pda* $M_S(V, \Sigma, \Gamma_S, P, \tau, \varepsilon, \{S\}, \$, |)$*, where* Γ_S *and* τ *are of form:*

(sa) $|a \to a| \in \Gamma_S$ "**shift** $a \in \Sigma$",
$\tau(|a \to a|) = \varepsilon$.

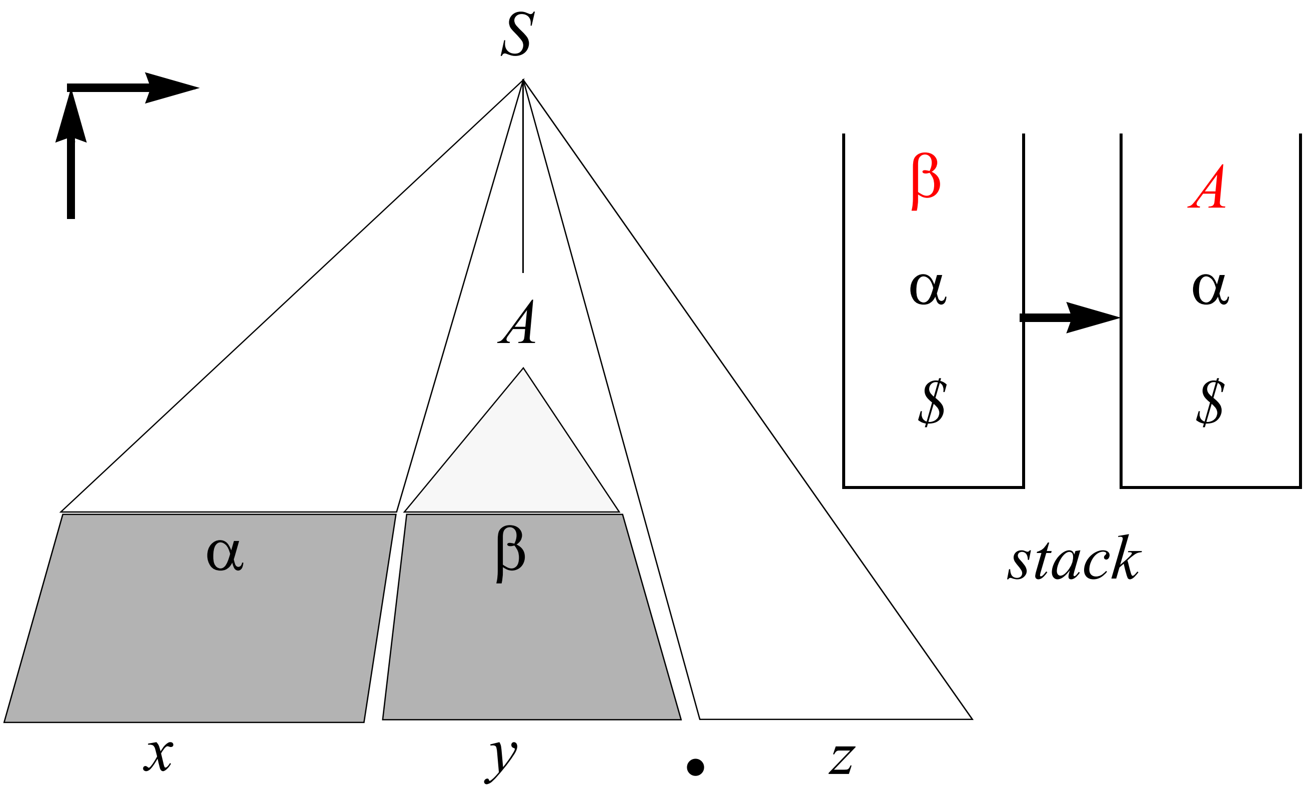
(ra) $\omega| \to A| \in \Gamma_S$ "**reduce** by $A \to \omega \in P$",
$\tau(\omega| \to A|) = A \to \omega$.

$S \Rightarrow_{rm}^{*} \alpha A z \Rightarrow_{rm}^{A\to\beta} \alpha\beta z \Rightarrow_{rm}^{*} xyz.$

Reversed *order of the rightmost derivation*

$xyz \Leftarrow_{rm}^{*} \alpha\beta z \Leftarrow_{rm}^{\beta\to A} \alpha A z \Leftarrow_{rm}^{*} S.$

$\$|xyz\$ \Rightarrow^{*} \$\alpha\beta|z\$ \Rightarrow_{r}^{\beta\to A} \$\alpha A|z\$ \Rightarrow^{*} \$S|\$.$



Let $x = a_1 \ldots a_k$, $y = b_1 \ldots b_m$, *and* $z = c_1 \ldots c_n$. *Then*

$$\$S \mid a_1 \ldots a_k b_1 \ldots b_m c_1 \ldots c_n\$ \Rightarrow_p \$\sigma^R \mid a_1 \ldots a_k b_1 \ldots b_m c_1 \ldots c_n\$$$

$$\Rightarrow^* \$\gamma^R A \mid b_1 \ldots b_m c_1 \ldots c_n\$$$

$$\Rightarrow_p^{A \to \beta} \$\gamma^R \beta^R \mid b_1 \ldots b_m c_1 \ldots c_n\$$$

$$\Rightarrow^* \$\gamma^R \mid c_1 \ldots c_n\$ \Rightarrow^* \$ \mid \$.$$

$$\$ \mid a_1 \ldots a_k b_1 \ldots b_m c_1 \ldots c_n\$ \Rightarrow_s \$a_1 \mid a_2 \ldots a_k b_1 \ldots b_m c_1 \ldots c_n\$$$

$$\Rightarrow^* \$\alpha \mid b_1 \ldots b_m c_1 \ldots c_n\$ \Rightarrow^* \$\alpha\beta \mid c_1 \ldots c_n\$$$

$$\Rightarrow_r^{\beta \to A} \$\alpha A \mid c_1 \ldots c_n\$ \Rightarrow^* \$S \mid \$$$

Let $G = (N, \Sigma, S, P)$ *be a cfg,* w *be a sentence of G, and* π_l *and* π_r *in* P^* *be a left and right parses of* w*, respectively. And let* $M_L = (V, \Sigma, \Gamma_L, P, \tau_L, S, \{\varepsilon\})$ *and* $M_R = (V, \Sigma, \Gamma_R, P, \tau_R, \varepsilon, \{S\})$ *be a left(=produce-shift) and right(=shift-reduce) parsers for G, respectively; and* θ_L *in* Γ_L^* *and* θ_R *in* Γ_R^* *are sequences of action strings for* w *in* M_R *and* M_L*, respectively.*

$$S \Rightarrow_{lm}^{\pi_l} w \text{ in } G, \qquad w \Leftarrow_{rm}^{\pi_r^R} S (= S \Rightarrow_{rm}^{\pi_r} w) \text{ in } G$$

$$\$S \mid w\$ \Rightarrow^{\theta_L} \$ \mid \$ \text{ in } M_L, \ \$ \mid w\$ \Rightarrow^{\theta_R} \$S \mid \$ \text{ in } M_R$$

Then

(1) $|\theta_L| = |\theta_R| = |\pi_l| + |w| = |\pi_r| + |w|$, *and*

(2) $\tau_L(\theta_L) = \pi_l \qquad \tau_R(\theta_R) = \pi_r^R$.

Let $G = (N, \Sigma, S, P)$ be a cfg, w be a sentence of G, and Ξ be a parse tree of w. And let $M_L = (V, \Sigma, \Gamma_L, P, \tau_L, S, \{\varepsilon\})$ and $M_R = (V, \Sigma, \Gamma_R, P, \tau_R, \varepsilon, \{S\})$ be a left and right parsers for G, respectively. Then the following actions in M_L and M_R generates the parse tree Ξ of w.

(pa) $A \,|\, \rightarrow \omega^R \,|\, \in \Gamma_L$

make nodes $X_1, \ldots, X_n$ and A, where $\omega = X_1 \ldots X_n$

make a subtree(A, ω)

(sa) $a \,|\, a \rightarrow \,|\, \in \Gamma_L$

no action

(sa) $|\, a \rightarrow a \,|\, \in \Gamma_R$

make a leaf node a

(ra) $\omega \,|\, \rightarrow A \,|\, \in \Gamma_R$

make a interior node A;

make a subtree (A, ω)

5.3 Strong LL(k) Parsing

Fact 5.23 *Let* $M = (V, \Sigma, \Gamma, \Delta, \tau, \gamma_s, F)$ *and* $M' = (V, \Sigma, \Gamma', \Delta, \tau, \gamma_s, F)$ *be pdt's with for some* $\delta \in V^*$, $z \in \Sigma^*$,

$\underline{\delta}\alpha \,|\, xy\underline{z} \rightarrow \underline{\delta}\beta \,|\, y\underline{z} \in \Gamma'$, *if* $\alpha \,|\, xy \rightarrow \beta \,|\, y \in \Gamma$. *Then*

$L(M') \subseteq L(M)$.

adding ***lookback*** *and* ***lookaheads***

reduces language set

$First_{G,k}(\gamma) = k{:}L_G(\gamma)$.

$Follow_{G,k}(\gamma) = \{y \in \Sigma^* \mid S \overset{*}{\Rightarrow} \alpha\gamma\beta \text{ in } G,\ y \in First_{G,k}(\beta)\}$.

$First_{G,k} \equiv First_k$, $\qquad Follow_{G,k} \equiv Follow_k$.

$First_k(W) = \bigcup_{\gamma \in W} First_k(\gamma)$ *where* $W \subseteq V^*$.

Fact 5.24

(a) $First_k(\alpha\beta\gamma) = k{:}First_{k+n}(\alpha\beta\gamma) = First_k(\alpha First_{k+n}(\beta)\gamma)$

(b) $Follow_k(\gamma) = k{:}Follow_{k+n}(\gamma) = First_k(Follow_{k+n}(\gamma))$

$\$^k$-***augmented*** *grammar* G' *for* $G = (N, \Sigma, P, S)$.

$G' = \{N \cup \{S'\}, \Sigma \cup \{\$\}, P \cup \{S' \rightarrow \$^k S \$^k\}, S')$

where $S', \$ \notin V$.

$First'_k$, $Follow'_k$ *denotes* $First_{G',k}$ *and* $Follow_{G',k}$.

Fact 5.25 $Follow'_k(\gamma) = k{:}Follow_k(\gamma)\$^k$.

Let G = (N, Σ, P, S) be a grammar and k be a natural number. The **strong LL(k) parser***(or* **SLL(k) parser***) for G is a pdt M = (V, Σ, Γ, P, τ, S, {ε}),*

(pa) $A \mid y \to \omega^R \mid y \in \Gamma$

$y \in First_k(\omega Follow_k(A)\$^k)$

"produce by A→ω on lookahead y",

$\tau(A \mid y \to \omega^R \mid y) = A \to \omega.$

(sa) $a \mid a \to \mid \in \Gamma$ *"shift a"*

$a \in \Sigma,$

$\tau(a \mid a \to \mid) = \varepsilon.$

Fact 5.26 *For any reduced grammar, SLL(0) parser for G is just the produce-shift parser.*

Theorem 5.31 *The* **SLL(k) parser** *M for a grammar G is a* **left parser** *for G for all k ≥ 0. Moreover for any sentence w in L(G), M produces all left parses of w in G and* $Time_M(w) = Time_G(w) + |w|$.

Proof

SLL(k) parser ⇒ linear left parser

Lemma *5.27, 5.28*

It is trivial, since SLL(k) is a restriction of predictive parser

SLL(k) parser ⇐ linear left parser in **Lemma** *5.29 and 5.30.*

Lemma 5.29 *Let $G = (N, \Sigma, P, S)$ be a grammar and (M, τ) be an SLL(k) parser.*

If $\gamma^R \underset{lm}{\overset{\pi}{\Rightarrow}} x\delta^R$ *in G,* $k{:}y \in First_k(\delta^R Follow_k(\gamma^R))$,

and $\delta^R = \varepsilon$ *or* $1{:}\delta^R \in N$, *then*

$\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$$, $\theta \in \Gamma^*$ *and*

$\tau(\theta) = \pi$, $|\theta| = |\pi| + |x|$.

Proof *Induction on the length of rule string* π.

i) $\pi = \varepsilon$. $\gamma^R = x\delta^R$.

$\$\gamma \mid xy\$ = \$\delta x^R \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$$; $|\theta| = |x|$, $\tau(\theta) = \pi = \varepsilon$.

ii) $\pi = \pi' \cdot A \to \omega$, $\pi' \neq \varepsilon$, $\exists \theta'$, $\tau(\theta') = \pi'$.

$\gamma^R \underset{lm}{\overset{\pi'}{\Rightarrow}} x'\delta'^R = x'A\delta'' \underset{lm}{\overset{A\to\omega}{\Rightarrow}} x'\omega\delta'' = x'z\delta^R = x\delta^R$,

where $|\theta'| = |\pi'| + |x'|$, $\delta'^R = A\delta''$, $\omega\delta'' = z\delta^R$, $x'z = x$.

Let $y' = zy$.

$\$\gamma \mid xy\$ = \$\gamma \mid x'zy\$ = \$\gamma \mid x'y'\$ \overset{\theta'}{\Rightarrow} \$\delta' \mid y'\$ = \$\delta''^R A \mid y'\$$

$\overset{r}{\Rightarrow} \$\delta''^R \omega^R \mid y'\$ = \$(\omega\delta'')^R \mid zy\$ = \$\delta z^R \mid zy\$$

$\overset{\theta''}{\Rightarrow} \$\delta \mid y\$$, *where* $|\theta''| = |z|$, $\tau(\theta'') = \varepsilon$.

$\therefore$ $|\theta| = |\theta'| + 1 + |\theta''| = |\pi'| + |x'| + 1 + |z| = |\pi| + |x|$.

$k{:}y' \in First_k(\delta'^R Follow_k(\gamma^R)) = First_k(A\delta'' Follow_k(\gamma^R))$

$k{:}y' \in First_k(\omega\delta'' Follow_k(\gamma^R)) = First_k(z\delta^R Follow_k(\gamma^R))$

$\therefore k{:}y \in First_k(\delta^R Follow_k(\gamma^R))$. *Q.E.D.*

Lemma 5.30 *Let (M, τ) is a* ***SLL(k) parser*** *for G.*

$L(G) \subseteq L(M)$, $Time_M(w) \leq Time_G(w) + |w|$.

5.4 Strong LL(k) Grammars

G is ***strong LL(k)****(or* ***SLL(k)****),*

if its SLL(k) parser is **deterministic**.

L is ***strong LL(k)****(or* ***SLL(k)****),*

if L is generated by SLL(k) grammar.

Lemma 5.32 *Let* $G = (N, \Sigma, P, S)$ *be a grammar.*

If $Y \overset{n}{\Rightarrow} \alpha X\beta$, $\alpha \overset{*}{\Rightarrow} x$, *and* $\beta \overset{*}{\Rightarrow} y$ *in G. Then*

$Y \underset{lm}{\overset{*}{\Rightarrow}} xX\gamma$, $\gamma \overset{*}{\Rightarrow} y$ *in G.*

Proof *Induction on n*

i) $n=0$, $\alpha=\beta=\varepsilon$, $X=Y$, $x=y=\varepsilon$, *choose* $\gamma=\varepsilon$.

ii) $n>0$, *three cases*

(a) $Y \overset{n-1}{\Rightarrow} \alpha' A\beta' \Rightarrow \alpha'\omega\underline{\beta'} = \alpha'\omega\beta'' X\beta = \alpha X\beta$.

$\alpha' \overset{*}{\Rightarrow} x'$, $\beta' = \gamma \overset{*}{\Rightarrow} y$

$\therefore Y \underset{lm}{\overset{*}{\Rightarrow}} x'A\beta' \underset{lm}{\Rightarrow} x'\omega\beta'' X\beta' \underset{lm}{\overset{*}{\Rightarrow}} x'x'' X\beta' = xX\gamma$.

$\gamma = \beta' \overset{*}{\Rightarrow} y$

(b) $Y \overset{n-1}{\Rightarrow} \alpha' A\beta' \Rightarrow \alpha'\underline{\omega}\beta' = \alpha'\omega_1 X\omega_2\beta' = \alpha X\beta$.

$\alpha' \overset{*}{\Rightarrow} x'$, $\beta' \overset{*}{\Rightarrow} y'$, $\gamma' \overset{*}{\Rightarrow} y'$, *and*

$Y \underset{lm}{\overset{*}{\Rightarrow}} x'A\gamma' \underset{lm}{\Rightarrow} x'\omega\gamma' = x'\omega_1 X\omega_2\gamma' \underset{lm}{\overset{*}{\Rightarrow}} xX\omega_2\gamma' = xX\gamma$,

$\gamma = \beta = \omega_2\gamma' \overset{*}{\Rightarrow} y$.

(c) $Y \overset{n-1}{\Rightarrow} \alpha' A\beta' \Rightarrow \underline{\alpha'}\omega\beta' = \alpha X\alpha''\omega\beta' = \alpha X\beta$.

$\alpha \overset{*}{\Rightarrow} x$, $\alpha''\omega\beta' \overset{*}{\Rightarrow} y$

$Y \underset{lm}{\overset{*}{\Rightarrow}} xX\alpha'' A\beta' = xX\gamma$, $\gamma = \alpha''\omega\beta' \overset{*}{\Rightarrow} y$.

Lemma 5.33

$Follow_k(X) = \{y \in \Sigma^* \cup \Sigma^* \$^k \mid S \overset{*}{\underset{lm}{\Rightarrow}} xX\beta,\ y \in First_k(\beta \$^k)\}$

Two actions

$A_1 \mid y_1 \rightarrow \omega_1{}^R \mid y_1,\ A_2 \mid y_2 \rightarrow \omega_2{}^R \mid y_2,$

***exhibits a produce-produce conflict**, if*

$A_1 = A_2,\ y_1 = y_2,$ *and* $\omega_1 \neq \omega_2.$

*A nonterminal A has the **SLL(k) property**, if*

$First_k(\omega_1 Follow_k(A)) \cap First_k(\omega_2 Follow_k(A)) = \emptyset,$

$\forall A \rightarrow \omega_1 \mid \omega_2 \in P.$

Theorem 5.34

(Characterization of SLL(k) Grammars)

*(a) The SLL(k) parser for G is **deterministic**.*

*(b) **No** pair of produce actions in SLL(k) parser for G exhibits a **produce-produce conflicts**.*

*(c) All nonterminals of G has the **SLL(k) property**.*

(d) The conditions

$S \overset{*}{\underset{lm}{\Rightarrow}} x_1 A \beta_1 \underset{lm}{\Rightarrow} x_1 \omega_1 \beta_1 \overset{*}{\underset{lm}{\Rightarrow}} x_1 y_1,$

$S \overset{*}{\underset{lm}{\Rightarrow}} x_2 A \beta_2 \underset{lm}{\Rightarrow} x_2 \omega_2 \beta_2 \overset{*}{\underset{lm}{\Rightarrow}} x_2 y_2,$

$k{:}y_1 = k{:}y_2,$

***always imply** that* $\omega_1 = \omega_2.$

Proof.

M *is nondeterministic* (a')

M *has distinct produce actions* (b')

$$A \mid y' \rightarrow \omega_1^R \mid y',\ A \mid y'z \rightarrow \omega_2^R \mid y'z.$$

$A \rightarrow \omega_1 \mid \omega_2 \in P,$
$y' \in First_k'(\omega_1 Follow_k'(A)),$
$y'z \in First_k'(\omega_2 Follow_k'(A)),$
$\omega_1 \neq \omega_2$ *or* $z \neq \varepsilon.$

$y' = k{:}y\$^k,\ y \in First_k(\omega_1 Follow_k(A)),$
$y'z = k{:}v\$^k,\ v \in First_k(\omega_2 Follow_k(A)),$
$z = \varepsilon$ *and* $y = v.$

$A \rightarrow \omega_1 \mid \omega_2 \in P,\ \omega_1 \neq \omega_2,$ (c')
$y \in First_k(\omega_1 Follow_k(A)) \cap First_k(\omega_2 Follow_k(A)).$

$S \overset{*}{\underset{lm}{\Rightarrow}} x_1 A \beta_1,\ y \in First_k(\omega_1 \beta_1),$ (d')
$S \overset{*}{\underset{lm}{\Rightarrow}} x_2 A \beta_2,\ y \in First_k(\omega_2 \beta_2).$
$A \rightarrow \omega_1 \mid \omega_2 \in P,\ \omega_1 \neq \omega_2,$

Theorem 5.35 *For all natural number k, the class of SLL(k) grammars is properly contained in the class of SLL(k+1) grammars.*

Proof

$$
\begin{aligned}
& First_k(\omega Follow_k(A)) \\
= \; & First_k(\omega First_k(Follow_{k+1}(A))) \\
= \; & First_k(\omega Follow_{k+1}(A)) \\
= \; & k{:}First_{k+1}(\omega Follow_{k+1}(A))
\end{aligned}
$$

$G_k = (\{S\}, \{a\}, \{S \rightarrow a^k \mid a^{k+1}\}, S)$

G_k *is SLL(k+1) but not SLL(k).*

Proposition 5.36

$L_k = \{a^n w \mid n \geq 1, w \in \{b, c, b^k d\}^n\}$

L_k *is SLL(k) but not SLL(k-1) for* $k \geq 1$.

Theorem 5.37 *Any SLL(k) grammar is* ***unambiguous.***

A configuration ϕ *is* ***looping****, if*

$$\forall n \geq 0, \exists \phi_n, \phi \overset{n}{\Rightarrow} \phi_n.$$

Fact 5.38 *If* ϕ *is a* *looping configuration,*

$\exists y, \gamma_0, \gamma_1, \ldots :$

$$\phi \overset{*}{\Rightarrow} \$\gamma_0 \mid y\$, \forall i \geq 0: \$\gamma_i \mid y\$ \Rightarrow \$\gamma_{i+1} \mid y\$$$

Fact 5.39 *A pda M* ***loops forever*** *on input string w, iff the initial configuration for w is looping.*

A nonterminal A is ***left-recursive*** *if* $A \overset{+}{\Rightarrow} A\beta$.
A grammar is ***left-recursive****, if it has a left-recursive nonterminal.*

Theorem 5.40 *Let G be a reduced* ***left-recursive*** *grammar. Then the SLL(k) parser M for G* ***loops forever*** *on some sentence* $w \in L(G)$.

Proof *Since G is left-recursive and reduced.*

$$S \underset{lm}{\overset{*}{\Rightarrow}} xA\beta,\ A \underset{lm}{\overset{\pi}{\Rightarrow}} A\delta,\ A \overset{*}{\Rightarrow} u,\ \delta \overset{*}{\Rightarrow} v,\ \beta \overset{*}{\Rightarrow} z.$$

Let $\psi_n = (A\delta^n\beta)^R$, $y_n = uv^nz$. *Then*

$$\forall n \geq 0,\ S \underset{lm}{\overset{*}{\Rightarrow}} xA\beta \underset{lm}{\overset{\pi^n}{\Rightarrow}} xA\delta^n\beta = x\psi_n{}^R.$$

$$\$S \mid xy_n\$ \overset{*}{\Rightarrow} \$\beta A \mid y_n\$ \overset{\theta_n}{\Rightarrow} \$\beta(\delta^R)^n A \mid y_n = \psi_n \mid y_n\$,$$

$$\tau(\theta_n) = \pi^n.$$

Corollary 5.41 *A* ***reduced left-recursive*** *grammar is* ***not SLL(k)*** *for any* $k \geq 0$.

Lemma 5.42 *Let* $G = (N, \Sigma, P, S)$ *be a grammar and,* $r_i \in P$ *such that*

$$\forall i \geq 0: A_i\gamma_i \overset{r_i}{\underset{lm}{\Rightarrow}} A_{i+1}\gamma_{i+1}.$$

Then $\exists i \in \mathbb{N} \ni A_i$ *is left-recursive,.*

Proof *Let* $i_0; \forall i \geq 0: |\gamma_{i_0}| \leq |\gamma_i|$

Let $i_k; \forall i \geq i_k: i_k > i_{k-1}, |\gamma_{i_k}| \leq |\gamma_i|$. *Then*

$\forall k \geq 0: A_{i_k}\gamma_{i_k} \overset{\pi_k}{\underset{lm}{\Rightarrow}} A_{i_{k+1}}\gamma_{i_{k+1}}, \pi_k = r_{i_k}\ldots r_{i_{k+1}-1}.$

$|\pi_k| = |i_{k+1} - i_k| > 0, \ \pi_k = \pi'_k\pi''_k,$

$A_{i_k} \overset{\pi'_k}{\underset{lm}{\Rightarrow}} \alpha_k, \gamma_{i_k} \overset{\pi''_k}{\underset{lm}{\Rightarrow}} \beta_k, \alpha_k\beta_k = A_{i_{k+1}}\gamma_{i_{k+1}}, |\pi'_k| > 0.$

Whenever $|\pi''_k| > 0$, *then* $\alpha_k \in \Sigma^*$.

$A_{i_k}\gamma_{i_k} \overset{\pi'_k}{\underset{lm}{\Rightarrow}} \alpha_k\gamma_{i_k} \overset{\pi''_k}{\underset{lm}{\Rightarrow}} \alpha_k\beta_k = A_{i_{k+1}}\gamma_{i_{k+1}}, \pi'_k\pi''_k = \pi_k$

$\alpha_k\gamma_{i_k} = A_j\gamma_j$, *where* $j = i_k + |\pi'_k|$.

$\exists\alpha'_k: \alpha_k = A_j\alpha'_k$, *since* $|\gamma_{i_k}| \leq |\gamma_j|$.

$\therefore \alpha_k \notin \Sigma^*, |\pi''_k| = 0$; *hence* $j = i_{k+1}$,

$\forall k \geq 0: A_{i_k} \overset{\pi'_k}{\underset{lm}{\Rightarrow}} \alpha_k = A_{i_{k+1}}\alpha'_k.$

Theorem 5.43 *Let M be the SLL(k) parser for a grammar G. If some configuration is looping in M, then G is left-recursive.*

Proof *If* ϕ *is a looping configuration,*

$$\exists y, \gamma_0, \gamma_1, \ldots, r_0, r_1, \ldots :$$

$$\phi \overset{*}{\Rightarrow} \$\gamma_0 \mid y\$,\ \forall i \geq 0: \$\gamma_i \mid y\$ \overset{r_i}{\Rightarrow} \$\gamma_{i+1} \mid y\$$$

$$\gamma_i = \alpha_i^R A_i,$$

$$\forall i \geq 0: A_i\alpha_i = (\alpha_i^R A_i)^R = \gamma_i^R \underset{lm}{\overset{\tau(r_i)}{\Rightarrow}} \gamma_{i+1}^R$$

$$= (\alpha_{i+1}^R A_{i+1})^R = A_{i+1}\alpha_{i+1}$$

Since $|\tau(r_i)| = 1$*, G is left-recursive.*

Theorem 5.44 *Let G be a reduced grammar and k a natural number. Then G is left-recursive iff the SLL(k) parser for G loops forever on some sentense in L(G).*

5.5 Construction of Strong LL(1) Parser

Fact 5.45 *The size of SLL(k) parser for* $G = (N, \Sigma, P, S)$ *is* $O(k \cdot |\Sigma|^k \cdot |G|)$.

Proof $A \mid y \to \omega^R \mid y \in \Gamma$, *size* $|A\omega| + 2k + 2$ *is* $O(k)$

$\forall A \mid y \to \omega^R \mid y \in \Gamma$, $|\Sigma|^k \cdot |P|$ *produce actions.*

X **begins** *A* — *if* $A \to \alpha X \beta \in P$, $\alpha \overset{*}{\Rightarrow} \varepsilon$

$A \; l \; X$.

X **ends** *A* — *if* $A \to \alpha X \beta \in P$, $\beta \overset{*}{\Rightarrow} \varepsilon$

$X \; r \; A$.

X **adjoins** *Y* — *if* $A \to \alpha X \gamma Y \beta \in P$, $\gamma \overset{*}{\Rightarrow} \varepsilon$

$X \; f \; Y$

X **terminals** *Y* — *if* $Y \in \Sigma$, $X = Y$.

$a = b$.

Lemma 5.47 *Let* $A \in N$, $X \in V$, $\alpha \in V^*$ *and* $m \le n$.

(1) $A \; l^m \; X$ *implies* $A \overset{n}{\underset{lm}{\Rightarrow}} X\alpha$.

(2) $A \overset{n}{\Rightarrow} X\alpha$ *implies* $A \; l^m \; X$.

(3) $X \; r^m \; A$ *implies* $A \overset{n}{\underset{rm}{\Rightarrow}} \alpha X$.

(4) $A \overset{n}{\underset{rm}{\Rightarrow}} \alpha X$ *implies* $X \; r^m \; A$.

$a \in First_1(A)$, *iff* $A \; l^* \; a$. .϶. $A \; l^* \; X$ *and* $X = a$.

$a \in Follow_1(A)$, *iff* $A \; r^* \; Y$, $Y \; f \; X$, $X \; l^* \; a$.

Proof *Induction on the length of the rule string* π.

i) $|\pi| = 1$, *choose* $X'=X$, $Y'=Y$, $r'=\pi=A\to\gamma X\psi Y\delta$, $\pi'=\varepsilon$, $\gamma'=\delta'=\alpha'=\beta'=\varepsilon$, $\psi'=\psi$.

ii) $\pi=\pi_1 r$, $r=A\to\omega_1$, $|\pi_1|\geq 1$.

$$A \overset{\pi_1}{\Rightarrow} \gamma_1 A_1 \delta_1 \overset{r}{\Rightarrow} \gamma_1\omega_1\delta_1 = \gamma X\psi Y\delta.$$

(a) $\gamma_1\omega_1\delta_1 = \gamma_1\omega_1\delta'_1\delta''_1 = \underline{\gamma}\delta''_1 = \gamma X\psi Y\delta.$

$g_1 A_1\delta_1 = \gamma_1 A_1\delta'_1\delta''_1 = \gamma_1 A_1\delta'_1 X\psi Y\delta.$

(b) $\gamma_1\omega_1\delta_1 = \gamma_1\underline{\alpha' X\psi'}\delta_1 = \gamma X\psi'\delta_1 = \gamma X\psi'\psi'' Y\delta$
$= \gamma X\psi Y\delta.$

$g_1 A_1\delta_1 = \gamma_1 A_1\psi'' Y\delta,$

$\psi'' \overset{*}{\Rightarrow} \varepsilon$, $A_1 \underset{rm}{\Rightarrow} \omega_1 = \alpha' X\psi' \underset{rm}{\overset{*}{\Rightarrow}} \alpha' X.$

(c) $\gamma_1\omega_1\delta_1 = \gamma X\psi'\omega_1\psi'' Y\delta = \gamma X\underline{\psi} Y\delta.$

$g_1 A_1\delta_1 = \gamma X\psi' A_1\psi'' Y\delta$, $\psi' A_1\psi'' \overset{*}{\Rightarrow} \varepsilon.$

(d) $\gamma_1\omega_1\delta_1 = \gamma_1\underline{\psi'' Y\beta'}\delta_1 = \gamma_1\psi'' Y\delta = \gamma X\psi'\psi'' Y\delta$
$= \gamma X\psi Y\delta.$

$g_1 A_1\delta_1 = \gamma X\psi' A_1\delta_1,$

$\psi' \overset{*}{\Rightarrow} \varepsilon$, $A_1 \underset{lm}{\Rightarrow} \omega_1 = \psi'' Y\beta' \underset{lm}{\overset{*}{\Rightarrow}} Y\beta'.$

(e) $\gamma_1\omega_1\delta_1 = \gamma'_1\gamma''_1\omega_1\delta_1 = \gamma'_1\underline{\delta} = \gamma X\psi Y\delta.$

$g_1 A_1\delta_1 = \gamma'_1\gamma''_1 A_1\delta_1 = \gamma X\psi Y\gamma''_1 A_1\delta_1.$

(f) $\gamma_1\omega_1\delta_1 = \gamma_1\underline{\alpha X\psi Y\beta}\delta_1 = \gamma X\psi Y\beta\delta_1 = \gamma X\psi Y\delta.$

Choose $X'=X$, $Y'=Y$, $r'=r$, $\gamma'=\gamma_1$, $\delta'=\delta_1$, $\alpha'=\beta'=\varepsilon$, $\pi'=\pi_1$.

Lemma 5.51 *Let $G = (N, \Sigma, P, S)$ be a grammar. Then $a \in Follow_1(X)$ implies $X\ r^*\ X'\ f\ Y'\ l^*\ a$. Moreover, if G is reduced then the converse also holds.*

Proof $a \in Follow_1(X)$ iff $S \overset{*}{\Rightarrow} \gamma X a y$.

$B \to \alpha X' \psi' Y' \beta \in P,\ X' \overset{*}{\Rightarrow} \alpha' X,\ \psi' \overset{*}{\Rightarrow} \varepsilon,\ Y' \overset{*}{\Rightarrow} a\beta'$.

If G is reduced, $S \overset{*}{\Rightarrow} \gamma' B \delta \Rightarrow \gamma' \alpha X' \psi' Y' \beta \delta$

$\overset{*}{\Rightarrow} \gamma' \alpha \alpha' X a \beta' \beta \delta \overset{*}{\Rightarrow} \gamma' \alpha \alpha' X a y$.

a **position***(or* **item core***) of G is of the form;*

$A \to \alpha \bullet \beta$, *where* $A \to \alpha\beta \in P,\ \bullet \notin (N \cup \Sigma)$

(a) $A \to \alpha \bullet X \beta$ **points**$(\uparrow)$ X.
(b) $A \to \alpha \bullet X \beta$ **passes-any**$(\downarrow)$ $A \to \alpha X \bullet \beta$.
(c) $A \to \alpha \bullet X \beta$ **passes-null**$(\Downarrow)$ $A \to \alpha X \bullet \beta$, *if* $X \overset{*}{\Rightarrow} \varepsilon$.

$^{\exists} a \in \Sigma,\ A \to \omega \in P$: $a \in First_1(\omega Follow_1(A))$ iff

$\omega = \alpha X \beta$, *where* $\alpha \overset{*}{\Rightarrow} \varepsilon$ *and* $a \in First_1(X)$, *or*

$\omega \overset{*}{\Rightarrow} \varepsilon$ *and* $a \in Follow_1(A)$.

$A \to \bullet\omega \Downarrow^* A \to \alpha \bullet X \beta \uparrow X\ l^*\ a$, *or*

$A \to \bullet\omega \Downarrow^* A \to \omega \bullet \wedge A\ r^* f\ l^*\ a$.

$First_1 : V^* \rightarrow 2^{\Sigma^{\leq 1}} = 2^{\Sigma \cup \{\varepsilon\}}$.

$Follow_1 : N \rightarrow 2^{\Sigma \cup \{\$\}^1}$.

$\therefore First_1(\alpha Follow_1(A)) : P \rightarrow 2^{\Sigma \cup \{\$\}^1}$.

$First_1(\alpha\beta) = First_1(\alpha)$, if $\alpha \not\Rightarrow^* \varepsilon$,
$First_1(\alpha) \cup First_1(\beta)$, if $\alpha \Rightarrow^* \varepsilon$, $\beta \Rightarrow^* \varepsilon$,
$First_1(\alpha) \cup First_1(\beta) \setminus \{\varepsilon\}$, if $\alpha \Rightarrow^* \varepsilon$, $\beta \not\Rightarrow^* \varepsilon$.

$First : V^* \rightarrow 2^\Sigma$
$First(\alpha) = \{a \in \Sigma \mid \alpha \Rightarrow^* ax, x \in \Sigma^*\}$
$First(\alpha\beta) = First(\alpha)$, if $\alpha \not\Rightarrow^* \varepsilon$,
$First(\alpha) \cup First(\beta)$, if $\alpha \Rightarrow^* \varepsilon$.
But $\alpha\beta \Rightarrow^* \varepsilon$, iff $\alpha \Rightarrow^* \varepsilon$, $\beta \Rightarrow^* \varepsilon$.

$First_1(\alpha Follow_1(A)) = First_1(\alpha Follow_1(A\$))$
$= First(\alpha)$, if $\alpha \not\Rightarrow^* \varepsilon$,
$First(\alpha) \cup Follow(A)$, if $\alpha \Rightarrow^* \varepsilon$.

$First(\alpha) = \{a \mid a \in First(X), \alpha = \beta X\gamma, \beta \Rightarrow^* \varepsilon\}$
$= \{a \mid \alpha = \beta a\gamma, \beta \Rightarrow^* \varepsilon\}$
$\cup \{a \mid a \in First(A), \alpha = \beta A\gamma, \beta \Rightarrow^* \varepsilon\}$

$First: N \rightarrow 2^{\Sigma}.$

$First(A) = \{a \mid a \in First(\alpha), A \rightarrow \alpha \in P\}$

$= \{a \mid A \rightarrow \alpha a \beta \in P, \alpha \Rightarrow^* \varepsilon\}$

$\cup \{a \mid a \in First(B), A \rightarrow \alpha B \beta, \alpha \Rightarrow^* \varepsilon\}$

$= \{a \mid A\ l\ a\} \cup \{a \mid a \in First(B), A\ l\ B\}$

$= \{a \mid a \in First(X), A\ l\ X\}$

$= \{a \mid a \in Init\_First(A)\} \cup \{a \mid a \in First(B), A\ l\ B\}$

where $Init\_First(A) = \{a \mid A\ l\ a\}$

$= Init\_First(A) \cup \cup_{A\ l\ B} First(B)$

$F(x) = G(x) \cup \cup_{x\ R\ y} F(y)$ *recursion*

$= \cup_{x\ R^*\ y}\ G(y)$ *iteration*

$First(A) = \cup_{A\ l^*\ B}\ Init\_First(B)$

for $A \rightarrow X_1 \ldots X_n \in P$ **do**

$i := 1$; $more :=$ **true**;

while $i \leq n \wedge more$ **do**

if $i < n \wedge X_i \Rightarrow^* \varepsilon \wedge X_{i+1} \in \Sigma$

$\rightarrow Init\_First(A) := Init\_First(A) \cup \{X_{i+1}\}$

$\mid i < n \wedge X_i \Rightarrow^* \varepsilon \wedge X_{i+1} \in N$

$\rightarrow make\_l\_rel(A, X_{i+1})$

$\mid X_i \not\Rightarrow^* \varepsilon \rightarrow more :=$ **false**

od od fi

5.7 Simple Precedence Parsing

one lookahead in SLL(1) parser

*one **lookback** into "stack"*

* **precedence** relation*

simple precedence parser

* ε-free*

simple precedence parser

(1) $X\omega \,|\, a \rightarrow XA \,|\, a$ "reduce by $A \rightarrow \omega$"

on lookback X and lookahead a

$X \lessdot 1{:}\omega,\ \omega{:}1 \gtrdot a,$

$X \in V \cup \{\$\},\ a \in \Sigma \cup \{\$\}.$

$\tau(X\omega \,|\, a \rightarrow XA \,|\, a) = A \rightarrow \omega.$

(2) $Y \,|\, b \rightarrow Yb \,|$ "shift b"

on lookback Y and lookahead b

$Y \lesseqdot b,$

$Y \in V \cup \{\$\},\ b \in \Sigma.$

$\tau(Y \,|\, b \rightarrow Yb \,|\,) = \varepsilon.$

Precedence relations

Let $G = (N, \Sigma, P, S)$.

$\doteq$ = ***adjoins*** *(f)*,

$X \doteq Y$, *if* $A \to \alpha X \gamma Y \beta \in P$, $\gamma \Rightarrow^* \varepsilon$.

$\lessdot$ = ***adjoins (begins*** $^+)^{-1}$ *(*$f(l^+)^{-1}$*)*,

$X \lessdot Y$, *if* $A \to \alpha X \gamma B \beta \in P$, $\gamma \Rightarrow^* \varepsilon$, $B \underset{lm}{\overset{+}{\Rightarrow}} Y\delta$.

$\gtrdot$ = ***ends***$^+$ ***adjoins (begins***$^*)^{-1}$ ***terminal***

$(r^+ f(l^*)^{-1})$,

$X \gtrdot a$, *if* $B \underset{rm}{\overset{+}{\Rightarrow}} \gamma X$, $A \to \alpha B \xi Y' \beta \in P$, $\xi \Rightarrow^* \varepsilon$,

$Y' \underset{lm}{\overset{*}{\Rightarrow}} a\delta$.

where $\doteq \cup \lessdot = \leqdot$

Lemma 5.58 *Let G be a* ε*-free grammar. Then*

(1) $X \doteq Y$, *iff* $A \to \alpha XY\beta \in P$.

(2) $X \lessdot Y$, *iff* $A \to \alpha XB\beta \in P$, $B \underset{lm}{\overset{+}{\Rightarrow}} Y\delta$

or $X \doteq Y$, $B\ l^*\ Y$.

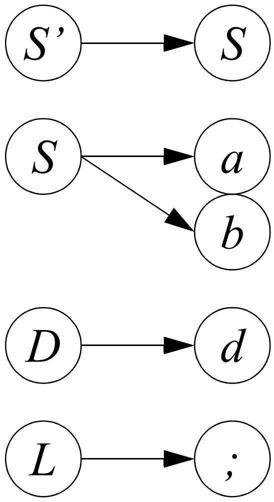
(3) $X \leqdot Y$, *iff* $A \to \alpha XY'\beta \in P$, $Y' \underset{lm}{\overset{*}{\Rightarrow}} Y\delta$.

(4) $X \gtrdot a$, *iff* $A \to \alpha BY'\beta \in P$, $B \underset{rm}{\overset{+}{\Rightarrow}} \gamma X$, $Y' \underset{lm}{\overset{*}{\Rightarrow}} a\delta$.

or $B \doteq Y'$, $B\ r^+\ X$, $Y'\ l^*\ a$.

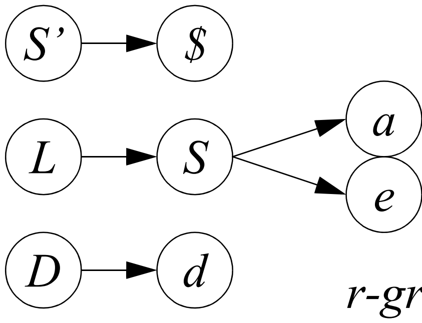
Example

$S' \rightarrow \$ \; S \; \$$
$S \rightarrow a \mid b \; D \; L \; e$
$D \rightarrow d$
$L \rightarrow ; \; S \mid L \; ; \; S$



l-graph

$X \doteq A$	$X \lessdot A \; l^{+} \; Y$
$\$ \doteq S$	$\$ \lessdot \{a, b\}$
$b \doteq D$	$b \lessdot \{d\}$
$D \doteq L$	$D \lessdot \{L, ;\}$
$; \doteq S$	$; \lessdot \{a, b\}$



r-graph

$A \doteq X$	$A \; r^{+} Y \gtrdot X \; l^{*} \; a$
$S \doteq \$$	$\{a, e\} \gtrdot \{\$\}$
$D \doteq L$	$\{d\} \gtrdot \{;\}$
$L \doteq e$	$\{S, a, e\} \gtrdot \{e\}$
$L \doteq ;$	$\{S, a, e\} \gtrdot \{;\}$

	S	*D*	*L*	*a*	*d*	*b*	*;*	*e*	*$*
S							$\gtrdot$	$\gtrdot$	$\doteq$
D			$\underline{\lessdot}$				$\lessdot$		
L							$\doteq$	$\doteq$	
a							$\gtrdot$	$\gtrdot$	$\gtrdot$
d							$\gtrdot$		
b		$\doteq$			$\lessdot$				
;	$\doteq$			$\lessdot$		$\lessdot$			
e							$\gtrdot$	$\gtrdot$	$\gtrdot$
$	$\doteq$			$\lessdot$		$\lessdot$			

reduce actions

	***lookback** X*	***lookahead** a*
$S \rightarrow a$	$\{\$, ;\} \lessdot a = 1{:}\omega,$	$a = \omega{:}1 \gtrdot \{e, ;, \$\}$
$S \rightarrow bDLe$	$\{\$, ;\} \lessdot b = 1{:}\omega,$	$e = \omega{:}1 \gtrdot \{e, ;, \$\}$
$D \rightarrow d\ ,$	$b \lessdot d = 1{:}\omega,$	$d = \omega{:}1 \gtrdot\ ;$
$L \rightarrow\ ;\ S$	$D \lessdot\ ; = 1{:}\omega,$	$S = \omega{:}1 \gtrdot \{e, ;\}$
$L \rightarrow L\ ;\ S$	$D \lessdot L = 1{:}\omega,$	$S = \omega{:}1 \gtrdot \{e, ;\}$

shift actions

$\{;, \$\} \underline{\lessdot} a$

$b \underline{\lessdot} d$

$\{;, \$\} \underline{\lessdot} b$

$\{D, L\} \underline{\lessdot}\ ;$

$L \underline{\lessdot} e,\ S \underline{\lessdot} \$.$

Theorem 5.65 *The simple precedence parser M for any ε-free grammar G is a right parser for G. Moreover, for each sentence w∈L(G), M produces all right parses of w in G, and* $Time_M(w) = Time_G(w) + |w|$.

Proof

simple precedence parser ⇒ linear right parser
in **Lemma** *5.60, 5.61.*
Since simple precedence parser is
a restriction of shift-reduce parser,
it is trivial.
simple precedence parser ⇐ linear right parser
in **Lemma** *5.62, 5.63, 5.64*

Lemma 5.60 *If* $\$\gamma \,|\, xy\$ \overset{\pi}{\Rightarrow} \$\delta \,|\, y\$$ *in M. Then*

$$\delta \overset{\tau(\pi)^R}{\underset{rm}{\Rightarrow}} \gamma x \text{ in } G, \text{ and } |\pi| = |\tau(\pi)| + |x|.$$

Lemma 5.61 *Let (M, τ) is a* ***simple precedence parser*** *for G.*

$L(G) \supseteq L(M),\ Time_G(w) \le Time_M(w) - |w|.$

Lemma 5.62 *Let $G = (N, \Sigma, P, S)$ be an ε-free grammar, $X, Y \in V$, $A \in N$, $a \in \Sigma$, and $\alpha, \beta \in V^*$.*

(1) If $X \lessdot A$ and G has $A \to Y\beta$, then $X \lessdot Y$.

(2) If $A (\lessdot \cup \gtrdot) a$ and G has $A \to \alpha X$, then $X \gtrdot a$.

Proof

(1) $B \to \alpha' X Y' \beta'$ *and* $Y' \overset{*}{\underset{lm}{\Rightarrow}} A\delta$.

$Y' \overset{+}{\underset{lm}{\Rightarrow}} Y\beta\delta$, $\therefore X \lessdot Y$.

(2) $B \to \alpha' X' Y' \beta'$, *where* $X' \overset{*}{\underset{rm}{\Rightarrow}} \gamma A$ *and* $Y' \overset{*}{\underset{lm}{\Rightarrow}} a\delta$.

$X' \overset{+}{\underset{rm}{\Rightarrow}} \gamma\alpha X$, $\therefore X \gtrdot a$.

A string $\gamma \in V^*$ is a **valid stack string**, *if $\gamma = X_1 \ldots X_n$ is ε or* $\$ \lessdot X_1 \lessdot X_2 \ldots \lessdot X_n$.

Any two succesive symbols in γ must be $\lessdot$ related.

Lemma 5.63 *If* $\delta \overset{\pi^R}{\underset{rm}{\Rightarrow}} \gamma x$ *in G,* δ *is valid stack string,*

$\$\delta{:}1\ (\lessdot \cup \gtrdot)\ 1{:}y\$$, *and either* $\gamma=\varepsilon$ *or* $\gamma{:}1 \in N$.

Then $\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\delta \mid y\$$ *in M,* γ *is valid stack string,*

$\$\gamma{:}1\ (\lessdot \cup \gtrdot)\ 1{:}xy\$$, $\tau(\theta)=\pi$, $|\theta| = |\pi| + |x|$.

Proof

Induction on $|\pi|$.

i) $\pi = \varepsilon$, $\delta = \gamma x$, *and the symbols in x are* $\lessdot$ *related.*

$\$\gamma \mid xy\$ \overset{\theta}{\Rightarrow} \$\gamma x \mid y\$ = \$\delta \mid y\$$, $\tau(\theta)=\varepsilon$, $|\theta| = |x|$.

Since $\$\gamma x{:}1\ (\lessdot \cup \gtrdot)\ 1{:}y\$$, *and* γx *is valid stack string,* $\$\gamma{:}1\ (\lessdot \cup \gtrdot)\ 1{:}xy\$$ $(\gamma{:}1 \lessdot 1{:}x$, *if* $x \neq \varepsilon)$.

ii) Assume $\pi = r\pi'$, $r = A \rightarrow \omega$,

$\delta \overset{\pi'^R}{\underset{rm}{\Rightarrow}} \gamma' x' = \gamma'' A x' \Rightarrow_{rm} \gamma'' \omega x' = \gamma x$ *in G, and*

$\$\delta{:}1\ (\lessdot \cup \gtrdot)\ 1{:}y\$$. *Then*

$\$\gamma' \mid x'y\$ \overset{\theta'}{\Rightarrow} \$\delta \mid y\$$ *in M,*

<u>$\gamma''A$</u> *is a valid stack string,* $\$\gamma''A{:}1(\lessdot \cup \gtrdot)\ 1{:}x'y\$$,

$\tau(\theta') = \pi'$, *and* $|\theta'| = |\pi'| + |x'|$.

$\$\gamma' \mid x'y\$ = \$\gamma''A \mid x'y\$$

Since $\gamma''{:}1 \lessdot 1{:}\omega(\gamma''{:}1 \lessdot A)$, $\omega{:}1 \gtrdot 1{:}x'y$. **(L5.62a,b)**,

$\$\gamma''\omega \mid x'y\$ \Rightarrow \$\gamma''A \mid x'y\$ = \$\gamma' \mid x'y\$$.

$\therefore\ \$\gamma \mid xy\$ = \$\gamma''\omega' \mid \underline{x''}x'y\$ \Rightarrow^{|\theta''|} x\$\gamma''\omega' x'' \mid x'y\$ =$

$\$\gamma''\omega \mid x'\$ \Rightarrow \$\gamma''A \mid x'y\$ = \$\gamma' \mid x'y\$ \overset{\theta'}{\Rightarrow} \$\delta \mid y\$$

$\gamma = \gamma''\omega$ *is v.s.t,* $\$\gamma''\omega{:}1\ (\lessdot \cup \gtrdot)\ 1{:}x''x'y\$$

$\pi = \theta'' \cdot red(A \rightarrow \omega) \cdot \theta'$ *where* θ'' *is* x''*-shift actions.*

Lemma 5.64 *Let (M, τ) is a* ***simple precedence parser*** *for G.*

$$L(G) \subseteq L(M),\ Time_M(w) \le Time_G(w) + |w|.$$

Proof

$\delta = S,\ x = w,\ \gamma = y = \varepsilon.$

If $S \overset{\pi^R}{\underset{rm}{\Rightarrow}} w$ *in G and* $\$S{:}1\ (\lessdot \cup \gtrdot)\ 1{:}\$$*, Then*

$\$ \mid w\$ \overset{\theta}{\Rightarrow} \$S \mid \$$ *in M,* $\$\ (\lessdot \cup \gtrdot)\ 1{:}w\$,$

$\tau(\theta) = \pi,\ |\theta| = |\pi| + |w|$

Since, $\$ \doteq S \doteq \$,$

$\$S \mid \$ \Rightarrow \$S\$ \mid.$

An ε-free grammar is a ***simple precedence grammar*** *if its simple precedence parser is* ***deterministic****, and* $S \Rightarrow^+ S$ *is* ***impossible*** *in G.*

Ambiguity $\$S \mid \$ \Rightarrow^+ \$S \mid \$$

$\$S \mid \$ \Rightarrow \$S\$ \mid$ *or*

$\Rightarrow \$S \mid \$.$

Example

$S \to S \mid a$

Theorem 5.66 *Any* ***simple precedence grammar*** *is* ***unambiguous****.*

Lemma 5.67 *Let G=(N, Σ, P, S) be a reduced ε-free grammar. Then*

$\forall X \in V, \exists Y \in V \cup \{\$\}, \exists a \in \Sigma \cup \{\$\}: Y \lessdot X\ (\lessdot \cup \gtrdot)\ a$ *holds in the \$-augmented grammar G' for G.*

Proof $S' \stackrel{*}{\Rightarrow} \gamma X a \delta$ *in G',* $\gamma \in \$V^*, \delta \in (V \cup \{\$\})^*$.

$A \rightarrow \alpha Y X' \beta \in P'$, *where* $X' \underset{lm}{\stackrel{*}{\Rightarrow}} X\beta'$. ***(L5.50)***

$\therefore\ Y \lessdot X.$

$A \rightarrow \alpha X' Y' \beta \in P'$,

where $X' \underset{rm}{\stackrel{*}{\Rightarrow}} \alpha' X,\ Y' \underset{lm}{\stackrel{*}{\Rightarrow}} Y\beta' = a\beta'$. ***(L5.50)***

$\therefore\ X' = \alpha' X$ *or* $X' \underset{rm}{\stackrel{+}{\Rightarrow}} \alpha' X.$

$\therefore\ X \lessdot a$ *or* $X \gtrdot a$. ***(L5.50 (3)(4))***

Theorem 5.68 *Let G=(N, Σ, P, S) be a ε-free grammar. The simple precedence parser M for G is deterministic whenever the following conditions are satisfied:*

(a) $(\lessdot \cap \gtrdot) = \emptyset$.

(b) G is invertible, i.e., no two rules in G have identical right-hand sides.

(c) $\forall X, Y, A \rightarrow \alpha X Y \beta, B \rightarrow Y\beta: X \lessdot Y$ *is impossible.*

If M is deterministic then all of the above conditions hold provided that G is reduced.

Proof *M is nondeterministic iff*
M has one of pairs of distinct actions:
(1) $X\mid a \to Xa\mid,\quad Y\alpha X\mid a \to YA\mid a.$ *s-r*
(2) $X\omega\mid a \to XA\mid a,\quad X\omega\mid a \to XB\mid a.$ *r-r*
(3) $Z\alpha XY\beta\mid a \to ZA\mid a,\quad XY\beta\mid a \to XB\mid a.$ *r-r*

By definition, $X\ (\lessdot \cap \gtrdot)\ a$ *in (1),* $X \lessdot Y$ *in (3)*
Conversely,

(1) If $\exists X,a: X\ (\lessdot \cap \gtrdot)\ a;\ \exists X\mid a \to Xa\mid$ *in M.*
$A' \to \alpha' X' Y' \beta'$ *in G', where* $\exists \gamma,\delta: X' \overset{+}{\underset{rm}{\Rightarrow}} \gamma X,\ Y' \overset{*}{\underset{lm}{\Rightarrow}} a\delta$
$\therefore \exists A \to \alpha X,\ \exists Y: Y \lessdot A$ *(**L5.67**),* $\therefore Y \lessdot 1{:}\alpha X,$ *(**L5.62**)*
Then $\exists Y\alpha X\mid a \to YA\mid a$ *in M.(1)*

(2) If G is not invertible; $\exists A,B: A \to \omega,\ B \to \omega,\ A \neq B.$
$\exists X \in V \cup \{\$\},\ a \in \Sigma \cup \{\$\}: X \lessdot A\ (\lessdot \cup \gtrdot)\ a,$
$\therefore X \lessdot 1{:}\omega,\ \omega{:}1 \gtrdot a$ *(**L5.62**). Then both of (2) in M.*

(3) Let $\exists A \to \alpha XY\beta,\ B \to Y\beta$ *in G.*
$\exists Z \in V \cup \{\$\},\ a \in \Sigma \cup \{\$\}: Z \lessdot A\ (\lessdot \cup \gtrdot)\ a,$ *(**L5.67**)*
$Z \lessdot 1{:}\alpha XY\beta,\ \alpha XY\beta{:}1 \gtrdot a$ *(**L5.62**)*
$\therefore \exists Z\alpha XY\beta\mid a \to ZA\mid a$ *in M.*
Since $Y\beta{:}1 = \alpha XY\beta{:}1,$
$\exists XY\beta\mid a \to XB\mid a$ *in M, whenever* $X \lessdot Y.$

Corollary 5.69 *A reduced ε-free grammar G is simple precedence iff the following conditions are satisfied:*

(a) $(\lessdot\!\!\!\!\!\underline{\ \ } \cap \gtrdot) = \emptyset$.

(b) G is invertible.

(c) $\forall X, Y,\ A \to \alpha XY\beta,\ B \to Y\beta$: $X \lessdot Y$ *is impossible.*

(d) $S \overset{+}{\Rightarrow} S$ *is impossible.*

Corollary 5.70 *An ε-free grammar G is simple precedence whenever the following conditions are satisfied:*

(a) $(\lessdot\!\!\!\!\!\underline{\ \ } \cap \gtrdot) = (\lessdot \cap \doteq) = \emptyset$.

(b) G is invertible.

(c) $S \overset{+}{\Rightarrow} S$ *is impossible.*

Theorem 5.71 *Given any reduced ε-free grammar G = (N, Σ, P, S), it is decidable in deterministic time* $O(|V| \cdot |G|)$ *whether or not G is simple precedence.*