Homework 4! (probably Monday evening)

Due to Chomsky, Evey, and Schützenberger (1962/3).

Theorem

Context-free grammars and Pushdown automata are equivalent.

$\hookrightarrow$(1) PDA acceptance by empty stack

$\hookrightarrow$(2) triplet construction, CFG nonterminals $[p, A, q]$ for PDA computations

References

N. Chomsky. Context-free grammars and push-down storage. Quarterly Progress Report No. 65, Research Laboratory of Electronics, M.I.T., Princeton, New Jersey (1962)

R.J. Evey. The theory and application of pushdown store machines. In Mathematical Linguistics and Automatic Translation, NSF-IO, pages 217–255. Harvard University, May 1963,
and

M. P. Schützenberger. On context-free languages and pushdown automata. Inform. and Control, 6:217–255, 1963. doi:10.1016/S0019-9958(63)90306-1

From lecture 11:
$M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$

configuration (q, x, α) $q \in Q$, $x \in \Sigma^*$, $\alpha \in \Gamma^*$

state, remaining input, stack with top left

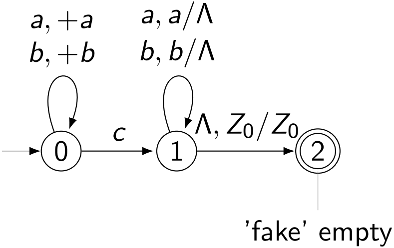
step $(p, ax, B\alpha) \vdash_M (q, x, \beta\alpha)$ when $(q, \beta) \in \delta(p, a, B)$
$\vdash_M^n$ $\vdash_M^*$ $\vdash$ $\vdash^n$ $\vdash^*$

Definition

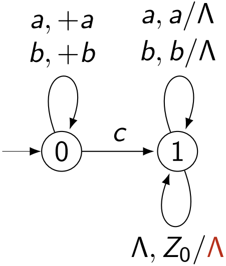
String x accepted by M (by *final state*), if
$(q_0, x, Z_0) \vdash^* (q, \Lambda, \alpha)$ for some $q \in A$, and some $\alpha \in \Gamma^*$
Language accepted by M (by *final state*)
$L(M) = \{\ x \in \Sigma^* \mid x \text{ accepted by } M\ \}$

read complete input, end in accepting state, some path

[M] D 5.2



check state



check stack

ABOVE

On many cases the PDA moves to the accepting state after checking that the stack is empty, when the topmost symbol is a special Z_0 that always has been at the bottom of the stack.

It may be more natural to accept directly by looking at the stack rather than by looking at the state. This leads to the notion of the *empty stack* language of a PDA.

$M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$

Definition

Language accepted by M by *empty stack*

$L_e(M) = \{ \ x \in \Sigma^* \mid (q_0, x, Z_0) \vdash^* (q, \Lambda, \Lambda)$ for some state $q \in Q \ \}$

[M] D 5.27

Theorem

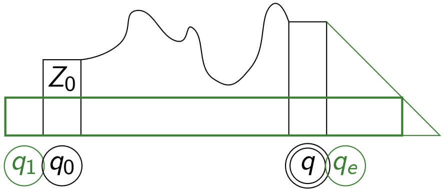
If M is a PDA then there is a PDA M_1 such that $L_e(M_1) = L(M)$.

Sketch of proof. . .

[M] Th 5.28

Simulate $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$

Simulate $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$
– empty stack ‘at’ final state
– prohibit early empty stack



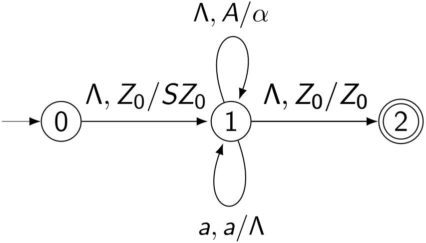
Construction PDA $M_1 = (Q_1, \Sigma, \Gamma_1, q_1, Z_1, A_1, \delta_1)$ such that $L_e(M_1) = L(M)$

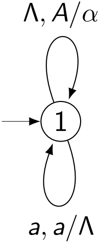
– $Q_1 = Q \cup \{q_1, q_e\}$
– $\Gamma_1 = \Gamma \cup \{Z_1\}$
– new instructions:

$\delta_1(q_1, \Lambda, Z_1) = \{(q_0, Z_0 Z_1)\}$
$\delta_1(q, \Lambda, X) \ni (q_e, X)$ for $q \in A$, and $X \in \Gamma_1$
$\delta_1(q_e, \Lambda, X) = \{(q_e, \Lambda)\}$ for $X \in \Gamma_1$

Expand-match with empty stack

$A \to \alpha \in P$, $a \in \Sigma$





Theorem

For every CFL L there exists a single state PDA M such that $L_e(M) = L$.

ABOVE

Now that we have empty stack acceptance we can reconsider the expand-match technique. In fact we do not need two extra states to introduce a bottom of stack symbol, and can make a single state PDA.

BELOW

The expand-match method can be used for any CFG. If we slightly restrict the grammars, we can combine each match with the expand step just before, that introduced the terminal. This gives a very direct translation between grammar and its leftmost derivation, and a single state PDA and its computation.

On this normal form each production is of the form $A \to a\alpha$, where $a \in \Sigma \cup \{\Lambda\}$ can be the only terminal at the right. That means that any terminal pushed on the stack will be on top, and immediately will be matched.

cfg G	$\Longleftrightarrow$	1-pda M	
$A \to \alpha$		$\delta(-, \Lambda, A) \ni (-, \alpha)$	expand
		$\delta(-, a, a) = \{(-, \Lambda)\}$	match

normal form $\alpha \in (\Sigma \cup \{\Lambda\}) \cdot V^*$

$A \to a\alpha$ $\delta(-, a, A) \ni (-, \alpha)$ combined

SimplePal: $S \to aSA \mid bSB \mid c$ $A \to a$ $B \to b$

leftmost derivation $\Longleftrightarrow$ computation

	S		$(-, abbcbba, S)$
$\Rightarrow$	$a\,SA$	$\vdash$	$(-, bbcbba, SA)$
$\Rightarrow$	$ab\,SBA$	$\vdash$	$(-, bcbba, SBA)$
$\Rightarrow$	$abb\,SBBA$	$\vdash$	$(-, cbba, SBBA)$
$\Rightarrow$	$abbc\,BBA$	$\vdash$	$(-, bba, BBA)$
$\Rightarrow$	$abbcb\,BA$	$\vdash$	$(-, ba, BA)$
$\Rightarrow$	$abbcbb\,A$	$\vdash$	$(-, a, A)$
$\Rightarrow$	$abbcbba$	$\vdash$	$(-, \Lambda, \Lambda)$

In this case: deterministic PDA

Exercise 5.21.
Prove the converse of Theorem 5.28:
If there is a PDA $M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$ accepting L by empty stack (that is, $x \in L$ if and only if $(q_0, x, Z_0) \vdash_M^* (q, \Lambda, \Lambda)$ for some state q), then there is a PDA M_1 accepting L by final state (i.e., the ordinary way).

Theorem

If $L = L_e(M)$ is the empty stack language of PDA M, then there exists a CFG G such that $L = L(G)$.

[M] Th 5.29

$$M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$$

Theorem

If $L = L_e(M)$ is the empty stack language of PDA M, then there exists a CFG G such that $L = L(G)$.

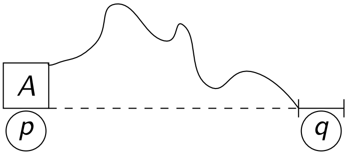
[M] Th 5.29

$M = (Q, \Sigma, \Gamma, q_0, Z_0, A, \delta)$

triplet construction

nonterminals $[p, A, q]$ $\quad p, q \in Q,\ A \in \Gamma$

$[p, A, q] \Rightarrow_G^* w$ iff $(p, w, A) \vdash_M^* (q, \Lambda, \Lambda)$

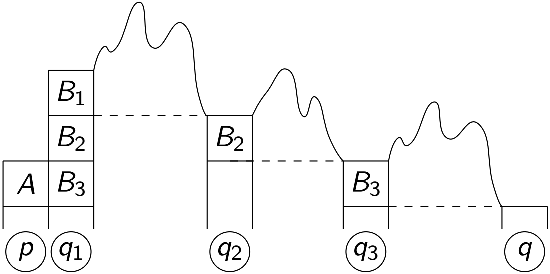


– productions

$$p \xrightarrow{\sigma,\, A/\Lambda} q$$

$[p, A, q] \to \sigma$ for $(q, \Lambda) \in \delta(p, \sigma, A)$

$$p \xrightarrow{\sigma,\, A/B_1 \ldots B_n} q_1$$

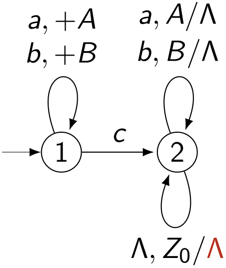


$[p, A, q] \to \sigma\ [q_1, B_1, q_2][q_2, B_2, q_3] \cdots [q_n, B_n, q]$
for $(q_1, B_1 \cdots B_n) \in \delta(p, \sigma, A)$, and $q, q_2, \ldots, q_n \in Q$

$S \to [q_0, Z_0, q]$ for all $q \in Q$

N.B.: σ may also be Λ

Construction from PDA to CFG, and the intuition behind it, must be known for the exam.
The details of the proof that $L(G) = L_e(M)$ do not have to be known for the exam.



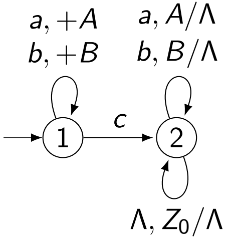
check stack

Example

$$L_e(M) = SimplePal = \{\ wcw^r \mid w \in \{a,b\}^*\ \}$$

12 transitions $\Rightarrow$ 33 $(+2)$ productions (!)

$$X \in \{A, B, Z_0\}$$



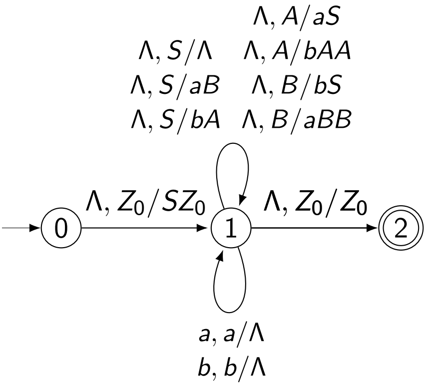
$$\begin{array}{ll}
 & S \to [1, Z_0, 1] \mid [1, Z_0, 2] \\
\delta(1, a, X) = \{(1, AX)\} & [1, X, 1] \to a\, [1, A, 1][1, X, 1] \\
 & [1, X, 1] \to a\, [1, A, 2][2, X, 1] \\
 & [1, X, 2] \to a\, [1, A, 1][1, X, 2] \\
 & [1, X, 2] \to a\, [1, A, 2][2, X, 2] \\
\delta(1, b, X) = \{(1, BX)\} & \ldots \\
\delta(1, c, X) = \{(2, X)\} & [1, X, 1] \to c\, [2, X, 1] \\
 & [1, X, 2] \to c\, [2, X, 2] \\
\delta(2, a, A) = \{(2, \Lambda)\} & [2, A, 2] \to a \\
\delta(2, b, B) = \{(2, \Lambda)\} & [2, B, 2] \to b \\
\delta(2, \Lambda, Z_0) = \{(2, \Lambda)\} & [2, Z_0, 2] \to \Lambda
\end{array}$$

not ‘live’

From lecture 12:

$AeqB = \{ x \in \{a, b\}^* \mid n_a(x) = n_b(x) \}$

$$S \to \Lambda \mid aB \mid bA$$
$$A \to aS \mid bAA$$
$$B \to bS \mid aBB$$



5.5. Parsing: make PDA (more) deterministic by looking ahead one symbol in input.
See Compiler Construction

Section 6

Context-Free and Non-Context-Free Languages

5 Context-Free and Non-Context-Free Languages
- Pumping Lemma
- Decision problems

From lecture 2:
Regular language is language accepted by an FA.

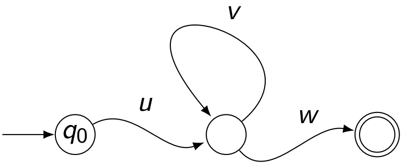
Theorem

Suppose L is a language over the alphabet Σ. If L is accepted by a finite automaton M, and if n is the number of states of M, then

$\forall$ *for every $x \in L$*
satisfying $|x| \geq n$
$\exists$ *there are three string u, v, and w,*
such that $x = uvw$ and the following three conditions are true:
(1) $|uv| \leq n$,
(2) $|v| \geq 1$
$\forall$ *and (3) for all $i \geq 0$, uv^iw belongs to L*

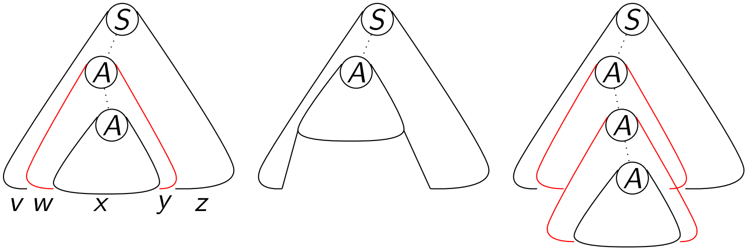
[M] Thm. 2.29

From lecture 2:



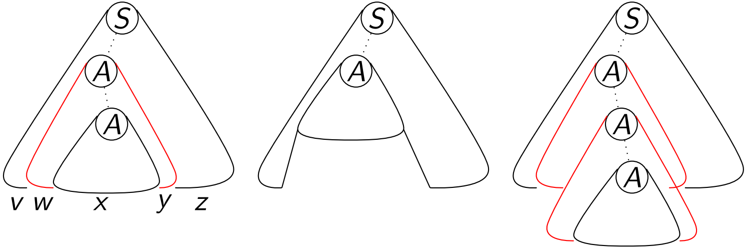
[M] Fig. 2.28

Pumping CF derivations



$$S \Rightarrow^* vAz \Rightarrow^* vwAyz \Rightarrow^* vwxyz,\ v, w, x, y, z \in \Sigma^*$$

$$S \underset{(1)}{\Rightarrow^*} vAz,\ A \underset{(2)}{\Rightarrow^*} wAy,\ A \underset{(3)}{\Rightarrow^*} x$$



$$S \underset{(1)}{\Rightarrow^*} vAz \underset{(3)}{\Rightarrow^*} vxz$$

$$S \underset{(1)}{\Rightarrow^*} vAz \underset{(2)}{\Rightarrow^*} vwAyz \underset{(2)}{\Rightarrow^*} vwwAyyz \underset{(3)}{\Rightarrow^*} vwwxyyz$$

Theorem (Pumping Lemma for context-free languages)

$\forall$ *for every context-free language* L

$\exists$ *there exists a constant* $n \geq 2$

such that

$\forall$ *for every* $u \in L$

with $|u| \geq n$

$\exists$ *there exists a decomposition* $u = vwxyz$

such that

(1) $|wy| \geq 1$

(2) $|wxy| \leq n$,

$\forall$ *(3) for all* $m \geq 0$, $vw^m xy^m z \in L$

[M] Thm. 6.1

Example

AnBnCn is not context-free.

[M] E 6.3