

Theory of Computation

Course note based on *Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science*, 2nd edition, authored by Martin Davis, Ron Sigal, and Elaine J. Weyuker.

course note prepared by

Tyng-Ruey Chuang

Institute of Information Science, Academia Sinica

Department of Information Management, National Taiwan University

Week 12, Spring 2010

About This Course Note

- ▶ It is prepared for the course *Theory of Computation* taught at the National Taiwan University in Spring 2010.
- ▶ It follows very closely the book *Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science*, 2nd edition, by Martin Davis, Ron Sigal, and Elaine J. Weyuker. Morgan Kaufmann Publishers. ISBN: 0-12-206382-1.
- ▶ It is available from Tyng-Ruey Chuang's web site:

<http://www.iis.sinica.edu.tw/~trc/>

and released under a Creative Commons
"Attribution-ShareAlike 3.0 Taiwan" license:

<http://creativecommons.org/licenses/by-sa/3.0/tw/>

Δ , A Context-free Grammar

Now let Δ be the grammar whose variables, start symbol, and terminals are those of Γ_s and whose productions are as follows:

1. all productions $V \rightarrow a$ from Γ with $a \in T$,
2. all productions $X_i \rightarrow (; Y_i, i = 1, 2, \dots, n$,
3. all productions $V \rightarrow a) ; Z_i, i = 1, 2, \dots, n$, for which $V \rightarrow a$ is a production of Γ with $a \in T$.

Lemma 2

Lemma 2. $L(\Delta)$ is regular.

Proof. Δ is right-linear. By Theorem 2.5, it is regular. □

Lemma 3

Lemma 3. $L(\Gamma_s) \subseteq L(\Delta)$.

Proof. We show that if $X \Rightarrow_{\Gamma_s}^* u \in (T \cup P)^*$ then $X \Rightarrow_{\Delta}^* u$. The proof is by an induction on the length of a derivation of u from X in Γ_s . Let

$$X = X_i \Rightarrow_{\Gamma_s} (i Y_i)_i Z_i \Rightarrow_{\Gamma_s}^* (i v)_i w = u,$$

where the induction hypothesis applies to $Y_i \Rightarrow_{\Gamma_s}^* v$ and $Z_i \Rightarrow_{\Gamma_s}^* w$. Thus $Y_i \Rightarrow_{\Delta}^* v$ and $Z_i \Rightarrow_{\Delta}^* w$. By Exercise 3. (p. 308 of the textbook), we can show that $v = z a, a \in T$. We conclude

$$Y_i \Rightarrow_{\Delta}^* z V \Rightarrow_{\Delta} z a = v,$$

where $V \rightarrow a$ is a production of Γ . But then we have

$$X_i \Rightarrow_{\Delta} (i Y_i \Rightarrow_{\Delta}^* (i z V \Rightarrow_{\Delta} (i z a)_i Z_i \Rightarrow_{\Delta}^* (i v)_i w = u.$$

Lemma 4

Lemma 4. $L(\Delta) \cap \text{PAR}_n(T) \subseteq L(\Gamma_s)$.

Proof. Let $X \Rightarrow_{\Delta}^* u$, where $u \in \text{PAR}_n(T)$. We shall prove that $X \Rightarrow_{\Gamma_s}^* u$. The proof is by an induction on the total number of pairs of the brackets $(i,)_i$ in u . If there is no such pair, then $u \in T$ and production $X \rightarrow u$ is in Δ hence in Γ_s . Thus $X \Rightarrow_{\Gamma_s}^* u$.

Suppose there are pairs of brackets in u . By observing all the available productions in Δ , we conclude that $u = (i z$ for some z and i . As $u \in \text{PAR}_n(T)$, we further conclude that $u = (i v)_i w$, where $v, w \in \text{PAR}_n(T)$.

As the symbol $)_i$ can only arise from the use of some production $V \rightarrow a)_i Z_i$ in Δ . So v must end in a terminal a , so we can write $v = \bar{v}a$, where

Lemma 4, Continued

Proof (Continued).

$$X = X_i \Rightarrow_{\Delta} (i Y_i \Rightarrow_{\Delta}^* (i \bar{v}V \Rightarrow_{\Delta} (i \bar{v}a)_i Z_i \Rightarrow_{\Delta}^* (i v)_i w$$

and

$$Z_i \Rightarrow_{\Delta}^* w.$$

Moreover, since $v \rightarrow a$ is a production of Γ , hence of Δ , we also have in Δ

$$Y_i \Rightarrow_{\Delta}^* \bar{v}V \Rightarrow_{\Delta} \bar{v}a = v.$$

Since v and w must each contain fewer pairs of brackets than u , we have by induction hypothesis

$$Y_i \Rightarrow_{\Gamma_s}^* v, \quad Z_i \Rightarrow_{\Gamma_s}^* w.$$

Hence,

$$X_i \Rightarrow_{\Gamma_s} (i Y_i)_i Z_i \Rightarrow_{\Gamma_s}^* (i v)_i w = u$$

A Main Theorem

Theorem 7.3. Let Γ be a grammar in Chomsky normal form with terminals T . Then there is a regular language R such that

$$L(\Gamma_s) = R \cap \text{PAR}_n(T).$$

Proof. Let Δ be defined as above and let $R = L(\Delta)$. The result follows from Lemmas 1-4. □

Chomsky-Schützenberger Representation Theorem

Theorem 7.4. A language $L \subseteq T^*$ is context-free if and only if there is a regular language R and a number n such that

$$L = \text{Er}_P(R \cap \text{PAR}_n(T))$$

where $P = \{(,)_i \mid i = 1, 2, \dots, n\}$.

Proof. By Theorem 7.1, 7.2, and 7.3. □

We will see that the Chomsky-Schützenberger Representation Theorem is instructional in the design of a class of machines — the Pushdown Automata — to recognize context-free languages.

Automata That Accept Context-free Languages?

What kind of automaton is needed for accepting context-free languages?

For a Chomsky normal form context-free grammar Γ with terminals T , and additional bracket symbols P ,

- ▶ Theorem 7.2 says $Er_P(L(\Gamma_s)) = L(\Gamma)$.
- ▶ Theorem 7.3 says $L(\Gamma_s) = R \cap \text{PAR}_n(T)$.
- ▶ We shall first try to construct an appropriate automaton for recognizing $L(\Gamma_s)$.
- ▶ R is accepted by a finite automaton; we need additional facilities to check if some given words belong to $\text{PAR}_n(T)$.
- ▶ A first-in-last-out “pushdown stack” is needed to recognize $\text{PAR}_n(T)$.

Pushdown Stack

At each step, one or both of the following operations can be performed:

1. The symbol at the “top” of the stack may be read and discarded. This operation is called “popping the stack”.
2. A new symbol may be “pushed” onto the stack.

A stack can be used to identify a string as belonging to $\text{PAR}_n(T)$ as follows:

- ▶ A special symbol J_i is introduced for each pair $(i,)_i, i = 1, 2, \dots, n$.
- ▶ As the automaton moves from left to right over a string, it pushes J_i onto the stack whenever it sees $(i$, and it pops the stack, eliminating a J_i , whenever it sees $)_i$.
- ▶ In case the string belongs to $\text{PAR}_n(T)$, the automaton will terminate with an empty stack after moving to the right end of the string.

Notations

Let T be a given alphabet and let $P = \{(i,)_i \mid i = 1, 2, \dots, n\}$.

Let $\Omega = \{J_1, J_2, \dots, J_n\}$, where we have introduced a single symbol J_i for each pair $(i,)_i, 1 \leq i \leq n$. Let $u \in (T \cup P)^*$, say, $u = c_1 c_2 \dots c_k$, where $c_1, c_2, \dots, c_k \in T \cup P$.

We define a sequence $\gamma_j(u)$ of elements of Ω^* to characterize the content of the pushdown stack as follows:

$$\begin{aligned} \gamma_1(u) &= 0 \\ \gamma_{j+1}(u) &= \begin{cases} \gamma_j(u) & \text{if } c_j \in T \\ J_i \gamma_j(u) & \text{if } c_j = (i \\ \alpha & \text{if } c_j =)_i \text{ and } \gamma_j(u) = J_i \alpha \end{cases} \end{aligned}$$

for $j = 1, 2, \dots, k$. Note that if $c_j =)_i$, $\gamma_{j+1}(u)$ will be undefined unless γ_j begins with the symbol J_i for the very same value of i . Furthermore, if a particular $\gamma_r(u)$ is undefined, all $\gamma_j(u)$ with $j > r$ will also be undefined.

Words in $\text{PAR}_n(T)$ Are Balanced

Definition. We say that the words $u \in (T \cup P)^*$ is *balanced* if $\gamma_j(u)$ is defined for $1 \leq j \leq |u| + 1$ and $\gamma_{|u|+1}(u) = 0$. □

Theorem 8.1. Let T be an alphabet and let

$$P = \{(i,)_i \mid i = 1, 2, \dots, n\}, \quad T \cap P = \emptyset.$$

Let $u \in (T \cup P)^*$. Then $u \in \text{PAR}_n(T)$ if and only if u is balanced.
□

The proof of Theorem 8.1 is via a series of easy lemmas.

Lemmas

Lemma 1. If $u \in T^*$, then U is balanced. □

Lemma 2. If u and v are balanced, so is uv . □

Lemma 3. Let $v = ({}_i u)_i$. Then u is balanced if and only if v is balanced. □

Lemma 4. If u is balanced and uv is balanced, then v is balanced. □

Lemma 5. If $u \in \text{PAR}_n(T)$, then u is balanced. □

Lemma 6. If u is balanced, then $u \in \text{PAR}_n(T)$. □

Pushdown Automata

A pushdown automaton \mathcal{M} consists of

- ▶ a finite set of states $Q = \{q_1, \dots, q_m\}$, where q_1 is the initial state, and $F \subseteq Q$ is the set of final, or accepting, states,
- ▶ a tape alphabet A ,
- ▶ a pushdown alphabet Ω ,
- ▶ a symbol $\mathbf{0}$ not in A nor in Ω , and
- ▶ a finite set of transitions which each is a quintuple of the form

$$q_i a U : V q_j$$

where $a \in \bar{A} = A \cup \{\mathbf{0}\}$, $U, V \in \bar{\Omega} = \Omega \cup \{\mathbf{0}\}$.

Intuitively, if $a \in A$ and $U, V \in \Omega$, the quintuple reads: “In state q_i scanning a , with U on top of the stack, move one square to the right, ‘pop’ the stack removing U , ‘push’ V onto the stack, and enter state q_j .”

Pushdown Automata, Continued

For the quintuple

$$q_i a U : V q_j$$

where either a, U, V is $\mathbf{0}$, the transition is defined as the following.

- ▶ If $a = \mathbf{0}$, motion to the right does not take place and the stack action can occur regardless of what the symbol is actually being scanned.
- ▶ If $U = \mathbf{0}$, then nothing is to be popped.
- ▶ If $V = \mathbf{0}$, then nothing is to be pushed.

Furthermore, the *distinct* transitions $q_i a U : V q_j, q_i b W : X q_k$ are called *incompatible* if one of the following is the case:

1. $a = b$, and $U = W$;
2. $a = b$, and U or W is $\mathbf{0}$;
3. $U = W$, and a or b is $\mathbf{0}$;
4. a or b is $\mathbf{0}$, and U or W is $\mathbf{0}$.

A pushdown automaton is *deterministic* if it has no pair of incompatible transitions.

Configurations of Pushdown Automata

Let $u \in A^*$ and let \mathcal{M} be a pushdown automaton. Then a u -configuration for \mathcal{M} is a triple $\Delta = (k, q_i, \alpha)$, where $1 \leq k \leq |u| + 1$, q_i is a state of \mathcal{M} , and $\alpha \in \Omega^*$.

Intuitively, the u -configuration (k, q_i, α) stands for the situation in which u is written on \mathcal{M} 's tape, \mathcal{M} is scanning the k th symbol of U — or, if $k = |u| + 1$, has completed scanning u — and α is the string of symbols on the pushdown stack.

We speak of q_i as the *state of configuration* Δ , and of α as the *stack contents at configuration* Δ . If $\alpha = 0$, we say the *stack is empty at* Δ .

Configurations of Pushdown Automata, Continued

For a pair of u -configurations, we write

$$u : (k, q_i, \alpha) \vdash_{\mathcal{M}} (l, q_j, \beta)$$

if \mathcal{M} contains a transition $q_i a U : V q_j$, where $\alpha = U\gamma$, $\beta = V\gamma$ for some $\gamma \in \Omega^*$, and either

1. $l = k$ and $a = \mathbf{0}$, or
2. $l = k + 1$ and the k th symbol of u is a .

Note that the equation $\alpha = U\gamma$ is to be read simply $\alpha = \gamma$ in case $U = \mathbf{0}$; likewise for $\beta = V\gamma$.

Computation by Pushdown Automata

A sequence $\Delta_1, \Delta_2, \dots, \Delta_m$ of u -configurations is called a u -computation by \mathcal{M} if

1. $\Delta_1 = (1, q, 0)$ for some $q \in Q$,
2. $\Delta_m = (|u| + 1, p, \gamma)$ for some $p \in Q$ and $\gamma \in \Omega^*$, and
3. $u : \Delta_i \vdash_{\mathcal{M}} \Delta_{i+1}$, for $1 \leq i < m$.

This u -computation is called *accepting* if the state at Δ_1 is the initial state q_1 , the state p at Δ_m is in F , and the stack at Δ_m is empty.

We say that \mathcal{M} accepts the string $u \in A^*$ if there is an accepting u -computation by \mathcal{M} . We write $L(\mathcal{M})$ for the set of strings accepted by \mathcal{M} , and we call $L(\mathcal{M})$ the *language accepted by \mathcal{M}* .

Pushdown Automata, Examples

See Examples \mathcal{M}_1 , \mathcal{M}_2 , and \mathcal{M}_3 at page 312 in the textbook.

Separators and Deterministic Pushdown Automata

Theorem 8.2. Let Γ be a Chomsky normal form grammar with separator Γ_s . Then there is a deterministic pushdown automaton \mathcal{M} such that $L(\mathcal{M}) = L(\Gamma_s)$.

Proof Outline. By Theorem 7.3, for suitable n ,

$$L(\Gamma_s) = R \cap \text{PAR}_n(T),$$

where R is a regular language, and T is the set of terminals of Γ . Let $P = \{(i,)_i \mid i = 1, 2, \dots, n\}$, and \mathcal{M}_0 be a dfa with alphabet $T \cup P$ that accepts R . Let $Q = \{q_1, q_2, \dots, q_m\}$ be the states of \mathcal{M}_0 , q_1 the initial states, $F \subseteq Q$ the accepting states, and δ the transition function.

We construct a pushdown automaton \mathcal{M} with tape alphabet $T \cup P$ and the same states, initial state, and accepting states as \mathcal{M}_0 . \mathcal{M} is to have the pushdown alphabet $\Omega = \{J_1, \dots, J_n\}$.

Separators and Deterministic Pushdown Automata, Continued

Proof Outline (Continued). The transitions of \mathcal{M} are as follows for all $a \in Q$:

1. for each $a \in T$, $qa\mathbf{0} : \mathbf{0}p$, where $p = \delta(q, a)$;
2. for $i = 1, 2, \dots, n$, $q(i\mathbf{0} : J_i p_i$, where $p_i = \delta(q, (i)$;
3. for $i = 1, 2, \dots, n$, $q)_i J_i : \mathbf{0}\bar{p}_i$, where $\bar{p}_i = \delta(q,)_i$

Note that, by definition, \mathcal{M} is deterministic.

It remains to be proved that, for any $u \in L(\Gamma_s)$, there is an accepting u -computation by \mathcal{M} (\Rightarrow). Conversely, we need to prove that, if \mathcal{M} accepts $u \in (T \cup P)^*$, then there is a derivation of u in Γ_s (\Leftarrow). □

Separators and Deterministic Pushdown Automata, Continued

Proof Outline (Continued). (\Rightarrow) Let $u = c_1 c_2 \dots c_K \in L(\Gamma_s)$, where $c_1, c_2, \dots, c_K \in (T \cup P)$. Then there is a sequence of states $p_1, p_2, \dots, p_{K+1} \in Q$ such that $p_1 = q_1$, $p_{K+1} \in F$, and $\delta(p_i, c_i) = p_{i+1}$, $i = 1, 2, \dots, K$.

Since $u \in \text{PAR}_n(T)$, by Theorem 8.1, u is balanced, so that $\gamma_j(u)$ is defined for $j = 1, 2, \dots, K + 1$ and $\gamma_{K+1}(u) = 0$. We let

$$\Delta_i = (j, p_j, \gamma_j(u)), \quad j = 1, 2, \dots, K + 1.$$

It follows that

$$u : \Delta_j \vdash_{\mathcal{M}} \Delta_{j+1}, \quad j = 1, 2, \dots, K.$$

Thus $\Delta_1, \Delta_2, \dots, \Delta_{K+1}$ is an accepting u -computation by \mathcal{M} .

Separators and Deterministic Pushdown Automata, Continued

Proof Outline (Continued). (\Leftarrow) Conversely, let \mathcal{M} accept $u = c_1 c_2 \dots c_K$. Thus $\Delta_1, \Delta_2, \dots, \Delta_{K+1}$ is an accepting u -computation by \mathcal{M} . Let $\Delta_j = (j, p_j, \gamma_j), j = 1, 2, \dots, K + 1$. Since

$$u : \Delta_j \vdash_{\mathcal{M}} \Delta_{j+1}, \quad j = 1, 2, \dots, K$$

and $\gamma_1 = 0$, we see that γ_j satisfies the defining recursion for $\gamma_j(u)$ and hence, $\gamma_j = \gamma_j(u)$ for $j = 1, 2, \dots, K + 1$. Since $\gamma_{K+1} = 0$, u is balanced and hence $u \in \text{PAR}_n(T)$. Finally, we have $p_1 = q_1$, $p_{K+1} \in F$, and $\delta(p_j, c_j) = p_{j+1}$. Therefore the dfa \mathcal{M}_0 accepts u , and $u \in R$. We conclude that $u \in L(\Gamma_s)$. \square

Atomic Pushdown Automata

A pushdown automaton is called *atomic* (whether or not it is deterministic) if all of its transitions are one of the following forms:

1. $pa0 : 0q$,
2. $p0U : 0q$,
3. $p00 : Vq$.

Thus, at each step in a computation an atomic pushdown automaton can read the tape and move right, or pop a symbol off the stack or push a symbol on the stack. But, unlike pushdown automata in general, it cannot perform more than one of these actions in a single step.

We will later show that for any pushdown automaton \mathcal{M} , there is an atomic pushdown automaton $\bar{\mathcal{M}}$ such that $L(\mathcal{M}) = L(\bar{\mathcal{M}})$.

Computation Records of Atomic Pushdown Automata

Let \mathcal{M} be a given atomic pushdown automata with tape alphabet T and pushdown alphabet $\Omega = \{J_1, J_2, \dots, J_n\}$. We set

$$P = \{(i,)_i \mid i = 1, 2, \dots, n\}$$

and show how to use the “brackets” to define a kind of “records” of a computation by \mathcal{M} .

Let $\Delta_1, \Delta_2, \dots, \Delta_m$ be a v -computation by \mathcal{M} , where $v = c_1 c_2 \dots c_K$ and $c_k \in T, k = 1, 2, \dots, K$, and where $\Delta_i = (l_i, p_i, \gamma_i), i = 1, 2, \dots, m$. We set

$$w_1 = 0$$

$$w_{i+1} = \left\{ \begin{array}{ll} w_i c_i & \text{if } \gamma_{i+1} = \gamma_i \\ w_i(j & \text{if } \gamma_{i+1} = J_j \gamma_i \\ w_i)_j & \text{if } \gamma_i = J_j \gamma_{i+1} \end{array} \right\} \quad 1 \leq i < m$$

Computation Records of Atomic Pushdown Automata, Continued

Now let $w = w_m$, so that $\text{Er}_P(w) = v$ and $m = |w| + 1$. This word w is called the record of the given v -computation $\Delta_1, \dots, \Delta_m$ by \mathcal{M} .

From w we can read off not only the word v but also the sequence of “pushes” and “pops” as they occur. In particular, $w_i, 1 < i \leq m$, indicates how \mathcal{M} goes from Δ_{i-1} to Δ_i .

An Atomic Automaton for $L(\Gamma)$

We now modify the pushdown automaton \mathcal{M} of Theorem 8.2 so that it will accept $L(\Gamma)$ instead of $L(\Gamma_s)$. The idea is to use nondeterminism to “guess” the location of the “brackets” $(i,)_i$.

Continuing to use the notation of the proof of Theorem 8.2, We define a pushdown automaton $\bar{\mathcal{M}}$ with the same states, initial state, accepting states, the pushdown alphabet as \mathcal{M} . However, the tape alphabet of $\bar{\mathcal{M}}$ will be T (rather than $T \cup P$). The transitions of $\bar{\mathcal{M}}$ are, for all $q \in Q$:

1. for each $a \in T$, $qa\mathbf{0} : \mathbf{0}p$, where $p = \delta(q, a)$;
2. for $i = 1, 2, \dots, n$, $q\mathbf{0}\mathbf{0} : J_i p_i$, where $p_i = \delta(q, (i)$;
3. for $i = 1, 2, \dots, n$, $q\mathbf{0}J_i : \mathbf{0}p_i$, where $p_i = \delta(q,)_i$.

Depending on the transition function δ , $\bar{\mathcal{M}}$ can certainly be non-deterministic. Note that $\bar{\mathcal{M}}$ is atomic (though \mathcal{M} is not). It remains to be proved that $L(\bar{\mathcal{M}}) = L(\Gamma)$.

$$v \in L(\Gamma) \Rightarrow v \in L(\bar{\mathcal{M}})$$

Let $v \in L(\Gamma)$. Then, since $\text{Er}_P(L(\Gamma_S)) = L(\Gamma)$, there is a word $w \in L(\Gamma_S)$ such that $\text{Er}_P(w) = v$. By Theorem 8.2, $w \in L(\mathcal{M})$.
Let

$$\Delta_i = (i, p_i, \gamma_i), \quad i = 1, 2, \dots, m$$

be an accepting w -computation by \mathcal{M} (with $m = |w| + 1$).

Let $n_i = 1$ if $w : \Delta_i \vdash_{\mathcal{M}} \Delta_{i+1}$ is via transition $qa0 : 0p$ (with $p = \delta(q, a)$); otherwise $n_i = 0$, $1 \leq i < m$. Let

$$\begin{aligned} l_1 &= 1, \\ l_{i+1} &= l_i + n_i, \quad 1 \leq i < m. \end{aligned}$$

Finally let

$$\bar{\Delta}_i = (l_i, p_i, \gamma_i), \quad 1 \leq i < m.$$

Now, it can be checked that

$$v : \bar{\Delta}_i \vdash_{\bar{\mathcal{M}}} \bar{\Delta}_{i+1}, \quad 1 \leq i < m.$$

Since $\bar{\Delta}_m = (|v| + 1, q, 0)$ with $q \in F$, we conclude $v \in L(\bar{\mathcal{M}})$.

$$v \in L(\bar{\mathcal{M}}) \Rightarrow v \in L(\Gamma)$$

Let $v \in L(\bar{\mathcal{M}})$. Let

$$\bar{\Delta}_i = (l_i, p_i, \gamma_i), \quad i = 1, 2, \dots, m$$

be an accepting v -computation by $\bar{\mathcal{M}}$. Using the fact that $\bar{\mathcal{M}}$ is atomic, we can let w be the record of this computation as defined earlier so that $\text{Er}_P(w) = v$ and $m = |w| + 1$. Let $\Delta_i = (i, p_i, \gamma_i)$, $i = 1, 2, \dots, m$, and we observe that

$$w : \Delta_i \vdash_{\mathcal{M}} \Delta_{i+1}, \quad i = 1, 2, \dots, m.$$

Since $p_m \in F$ and $\gamma_m = 0$, $\Delta_1, \Delta_2, \dots, \Delta_m$ is an accepting w -computation by \mathcal{M} . Thus by Theorem 8.2, $w \in L(\Gamma_s)$. Hence, $v \in L(\Gamma)$.

Context-free Languages and Pushdown Automata

Theorem 8.3. Let Γ be a Chomsky normal form context-free grammar. Then there is a pushdown automaton $\bar{\mathcal{M}}$ such that $L(\bar{\mathcal{M}}) = L(\Gamma)$. □

Theorem 8.4. For every context-free grammar L , there is a pushdown automaton \mathcal{M} such that $L = L(\mathcal{M})$. □

Note that to prove Theorem 8.4, we need to take care of the case where $0 \in L$, hence $L = L(\Gamma) \cup \{0\}$ for a Chomsky normal form context-free grammar Γ . For such a case, we need to modify the pushdown automaton $\bar{\mathcal{M}}$ that accepts $L(\Gamma)$. Actually we modify the dfa component \mathcal{M}_0 of $\bar{\mathcal{M}}$ to build an equivalent nonrestarting dfa. After that, we add the initial state of this new dfa to the set of accepting states so that 0 will be recognized.

Atomic Pushdown Automata, Revisited

Theorem 8.5. Let \mathcal{M} be a pushdown automaton. Then there is an atomic pushdown automaton $\bar{\mathcal{M}}$ such that $L(\mathcal{M}) = L(\bar{\mathcal{M}})$.

Proof. For each transition $paU : Vq$ of \mathcal{M} for which $a, U, v \neq \mathbf{0}$, we introduce two new states r, s and let $\bar{\mathcal{M}}$ have the transitions

$$pa\mathbf{0} : \mathbf{0}r$$

$$r\mathbf{0}U : \mathbf{0}s$$

$$s\mathbf{0}\mathbf{0} : Vq$$

If exactly one of a, U, V is $\mathbf{0}$, the only two transitions are needed for $\bar{\mathcal{M}}$. For each transition $p\mathbf{0}\mathbf{0} : \mathbf{0}q$, we introduce a new state t and replace $p\mathbf{0}\mathbf{0} : \mathbf{0}q$ with the transitions

$$p\mathbf{0}\mathbf{0} : Jt$$

$$t\mathbf{0}J : \mathbf{0}q$$

where J is an arbitrary symbol of the pushdown alphabet.

Otherwise, $\bar{\mathcal{M}}$ is exactly like \mathcal{M} . Clearly, $L(\bar{\mathcal{M}}) = L(\mathcal{M})$.

Context-free Languages and Pushdown Automata

Theorem 8.6. For every pushdown automaton \mathcal{M} , $L(\mathcal{M})$ is a context-free language.

Proof Outline. Without loss of generality, we assume \mathcal{M} is atomic. The plan is to prove that for the language L consisting exactly of the records of all accepting u -computation by \mathcal{M} , where $u \in L(\mathcal{M})$, we will have $L = R \cap \text{PAR}_n(T)$. R will be a regular language accepted by a ndfa \mathcal{M}_0 devised from \mathcal{M} , and T is tape alphabet of \mathcal{M} . As $L(\mathcal{M}) = \text{Er}_P(L)$, it follows that $L(\mathcal{M})$ is a context-free language.

To prove $L = R \cap \text{PAR}_n(T)$, we need to show both $L \subseteq R \cap \text{PAR}_n(T)$ and $R \cap \text{PAR}_n(T) \subseteq L$.

Context-free Languages and Pushdown Automata

Proof Outline of Theorem 8.6, Continued. Let \mathcal{M} have states $Q = \{q_1, q_2, \dots, q_m\}$, initial state q_1 , final states F , tape alphabet T , and pushdown alphabet $\Omega = \{J_1, \dots, J_m\}$.

Context-free Languages and Pushdown Automata

Proof Outline of Theorem 8.6, Continued. Let \mathcal{M} have states $Q = \{q_1, q_2, \dots, q_m\}$, initial state q_1 , final states F , tape alphabet T , and pushdown alphabet $\Omega = \{J_1, \dots, J_m\}$.

To devise ndfa \mathcal{M}_0 , we need $P = \{(\cdot)_i \mid i = 1, \dots, m\}$. \mathcal{M}_0 has the same states, initial state, and accepting states as \mathcal{M} , and transition function δ defined as follows. For each $q \in Q$,

$$\begin{aligned} \delta(q, a) &= \{p \in Q \mid \mathcal{M} \text{ has the transition } qa\mathbf{0} : \mathbf{0}p\} \text{ for } a \in T \\ \delta(q, (\cdot)_i) &= \{p \in Q \mid \mathcal{M} \text{ has the transition } q\mathbf{0}J_i : \mathbf{0}p\}, \quad i = 1, \dots, m, \\ \delta(q,)_i) &= \{p \in Q \mid \mathcal{M} \text{ has the transition } q\mathbf{0}\mathbf{0} : J_i p\}, \quad i = 1, \dots, m. \end{aligned}$$

Context-free Languages and Pushdown Automata

Proof Outline of Theorem 8.6, Continued. Let \mathcal{M} have states $Q = \{q_1, q_2, \dots, q_m\}$, initial state q_1 , final states F , tape alphabet T , and pushdown alphabet $\Omega = \{J_1, \dots, J_m\}$.

To devise ndfa \mathcal{M}_0 , we need $P = \{(\cdot)_i \mid i = 1, \dots, m\}$. \mathcal{M}_0 has the same states, initial state, and accepting states as \mathcal{M} , and transition function δ defined as follows. For each $q \in Q$,

$$\begin{aligned} \delta(q, a) &= \{p \in Q \mid \mathcal{M} \text{ has the transition } qa\mathbf{0} : \mathbf{0}p\} \text{ for } a \in T \\ \delta(q, (\cdot)_i) &= \{p \in Q \mid \mathcal{M} \text{ has the transition } q\mathbf{0}J_i : \mathbf{0}p\}, \quad i = 1, \dots, m, \\ \delta(q,)_i) &= \{p \in Q \mid \mathcal{M} \text{ has the transition } q\mathbf{0}\mathbf{0} : J_i p\}, \quad i = 1, \dots, m. \end{aligned}$$

Let $w \in L$ be the record of an accepting u -computation $\Delta_1, \dots, \Delta_m$, where $\Delta_i = (l_i, p_i, \gamma_i), i = 1, \dots, m$. By an induction, we can show that $p_m \in \delta^*(q_1, w)$. As $p_m \in F$, we have $w \in R$. By another induction, we can show that $\gamma_i(w) = \gamma_i, i = 1, \dots, m$. As $\gamma_{|w|+1}(w) = \gamma_{|w|+1} = \mathbf{0}$, we know w is balanced. We conclude that $w \in R \cap \text{PAR}_n(T)$.

Context-free Languages and Pushdown Automata

Proof Outline of Theorem 8.6, Continued. Conversely, let $w = c_1 \dots c_r \in R \cap \text{PAR}_n(T)$, and let $u = \text{Er}_P(w) = d_1, \dots, d_s$. Let p_1, \dots, p_{r+1} be some sequence of states such that $p_1 = q_1$, $p_{r+1} \in \delta(p_i, c_i)$ for $i = 1, \dots, r$. We claim that

$$(l_1, p_1, \gamma_1(w)), (l_2, p_2, \gamma_2(w)), \dots, (l_{r+1}, p_{r+1}, \gamma_{r+1}(w))$$

where

$$l_1 = 1$$

$$l_{i+1} = \begin{cases} l_i + 1 & \text{if } c_i \in T \\ l_i & \text{otherwise} \end{cases}$$

is an accepting u -computation by \mathcal{M} and w is its record. That is, we need to show that

$$u : (l_r, p_r, \gamma_r(w)) \vdash_{\mathcal{M}} (l_{r+1}, p_{r+1}, \gamma_{r+1}(w))$$

for $i = 1, \dots, r$. This is done by an induction i and based on the transitions that are used. We then conclude $w \in L$, the language of the records of all accepting u -computation by \mathcal{M} .