Assignment 15, Authomata Theory

Oleg Sivokon

<2016-01-01 Fri>

Contents

1 Problems

1.1 Problem 1

Given context-free grammar $V = \{S, M, N, W, X, Y, Z\}$ s.t. $T = \{1, 0\}$

$$\begin{aligned}
&S \to M \mid XN \mid W \mid 0N \mid 1Z1 \\
&M \to 0M0 \mid N \\
&N \to N0 \mid 0 \\
&W \to 0W \mid 00W0 \\
&X \to 0X1 \mid 0 \mid 0Y0 \\
&Z \to W \ .
\end{aligned}$$

1. Is V ambiguous?
2. Give a normalized grammar equivalent to V.

1.1.1 Answer 2

It is easier to normalize the grammar first and then to look for ambiguities, thus the answers are in reverse order.

1. Any derivation containing W cannot terminate, and so does Z.
2. Further, we can eliminate the rule $M \to N$.
3. Y has no derivation rules, thus we can also remove it.

Thus obtaining:

$$\begin{aligned}
&S \to M \mid XM \mid 0M \\
&M \to 0M0 \mid 0 \mid 0M \\
&X \to 0X1 \mid 0 \ .
\end{aligned}$$

1. It is easy to see that M derives number of zeros greater than one, thus $M \to 0M0$ is redundant. Subsequently, $S \to 0M$ is already covered by $S \to M$.

What remains is:

$$
\begin{aligned}
&S \to M \mid XM \\
&M \to 0 \mid 0M \\
&X \to 0 \mid 0X1 \ .
\end{aligned}
$$

1.1.2 Answer 1

Now it is easy to see that the string 00 can be derived in two different ways:

- $S \to M,\ M \to 0M,\ M \to 0.$
- $S \to XM,\ X \to 0,\ M \to 0.$

Hence V is ambiguous.

1.2 Problem 2

Given context-free grammar $V = \{S, M, N, W, X, Y, Z\}$ s.t. $T = \{1, 0\}$

$$
\begin{aligned}
&S \to 0W11 \mid 0X1 \mid 0Y \\
&W \to S \mid Z \\
&X \to S \mid W \\
&Y \to 1 \\
&Z \to X \ .
\end{aligned}
$$

1. Bring V to Chomsky's normal form.
2. What is the language of V?

1.2.1 Answer 3

1. We can easily eliminate Y variable, thus removing $Y \to 1$ rule, and adding $S \to 01$ rule.
2. We can eliminate Z variable by removing $Z \to X$ and $W \to Z$ rules and adding $W \to X$ rule.
3. We can eliminate X variable by removing $X \to S \mid W$ and $S \to 0X1$ rules, and adding: $S \to 0S1$ rule.
4. Finally, we can eliminate W variable by removing $W \to S$ and $S \to 0W11$ rules and adding $S \to 0S11$ rule.

The resulting grammar will be:

$$S \to 0S11 \mid 0S1 \mid 01 \; .$$

Since this is still not CNF, I introduce an extra variable: X and derivation rules $X \to 0$, $Y \to 1 \mid 11$ and $Z \to SY$ thus obtaining:

$$\begin{aligned} &S \to XZ \mid 01 \\ &X \to 0 \\ &Y \to 1 \mid 11 \\ &Z \to SY \; . \end{aligned}$$

Which is in CNF.

1.2.2 Answer 4

Using the results from the previous answer it is easy to see that the language $L(V) = \{0^n 1^k \mid n \leq k \wedge n, k > 0\}$.

1.3 Problem 3

Build a PDA accepting the language L by emptying the stack.

$$\begin{aligned} L = \{a^{i_1}b^{j_1}a^{i_2}b^{j_2}\dots a^{i_m}b^{j_m} &\mid m \geq 1 \\ &\mid \forall k : i_k \geq j_k \geq 1 \\ &\mid \exists k : i_k > j_k\} \end{aligned}$$

1.3.1 Answer 5

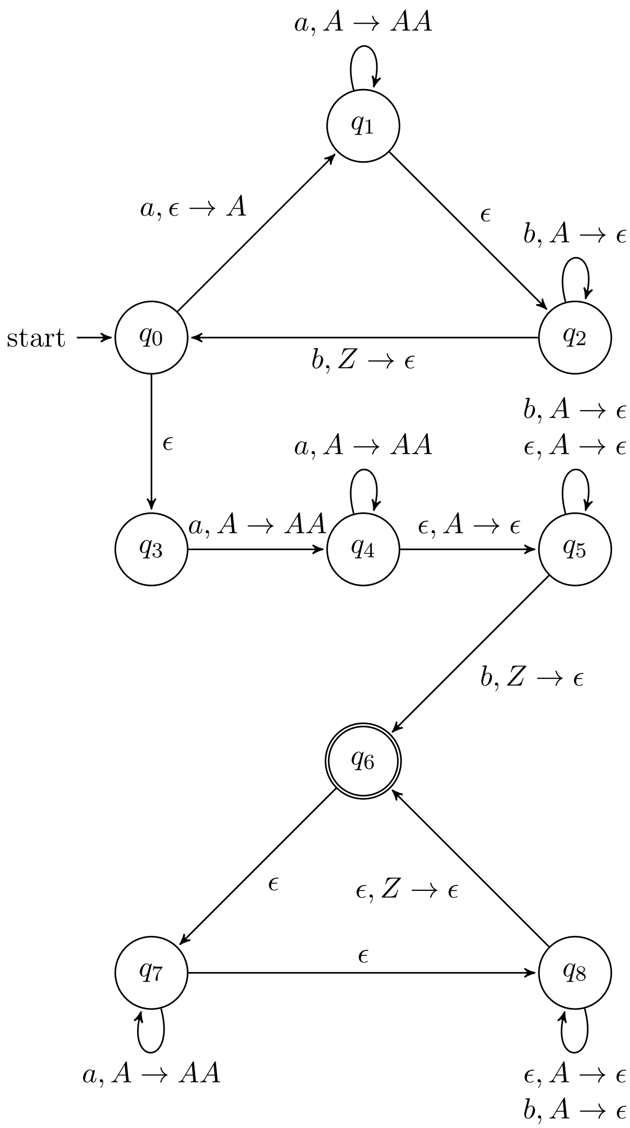
First, I'll write the grammar for L (because it's easier to do):

$$\begin{aligned} S &\to aSb \mid K \mid SS \\ K &\to aKb \mid aaKb \\ X &\to aXb \mid aaXb \mid \epsilon \ . \end{aligned}$$

1. X generates strings of the form $\{a^n b^m \mid n \geq m\}$.
2. Similarly, K generates strings of the form $\{a^n b^m \mid n > m\}$.
3. The derivatin of S can only terminate when it eventually derives either K. It can repeat as many times as needed to accept the entire string. Where the repeated element is, again, of the form of either $\{a^n b^m \mid n \geq m\}$, or $\{a^n b^m \mid n > m\}$.

Thus, at least informally, we are convinced the grammar generates L.

Now, the automaton:



The idea behind this diagram is as follows:

1. Loop as many times as needed (possibly zero) over strings $a^n b^n$, where $n \geq 1$.
2. Nondeterministically parse a string $a^n b^m$ where $n > m$.

3. Loop as many times as needed (possibly zero) over strings $a^n b^m$ where $n \geq m$.
4. Before accepting state, keep discarding A until none remain for as long as you don't see an a.
5. Accept the string.

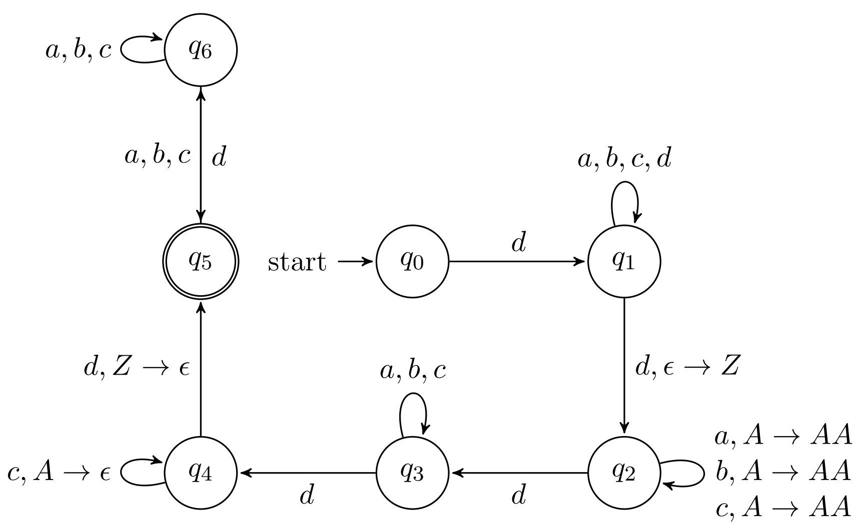
1.4 Problem 4

Construct a PDA for L defined as follows:

$$\begin{aligned} L = \{dw_1dw_2d\dots w_nd \mid\ & n \geq 3 \\ \mid\ & \forall k : w_k \in \{a, b, c\}^* \\ \mid\ & \exists k : 1 \leq k \leq n-2 \wedge |w_k| = \#_c(w_{k+2})\} \end{aligned}$$

1.4.1 Answer 6

The automaton for L may look like this:



(Whenever stack symbols neither read, nor added nor removed, they are omitted from the diagram to make it easier to read)

The rationale for this diagram is as follows:

1. Read the first d.
2. Keep reading as many of as, bs, cs or ds as necessary.
3. Non-deterministically, upon reading d assume reading the "special" sequence of as, bs and cs, which must be equal in length to the sequence of cs to follow after. "Notice" this even by pushing Z on stack.
4. Read as, bs and cs pushing As on stack.
5. Read d.
6. Read any number of as, bs or cs.
7. Upon reading d start reading cs while discarding As.
8. Once you pop Z you also must read d, this will ensure the number of cs is the same as the number of as, bs and cs, read in step (4).
9. After this had happened, we've already found the "special" substring, now we just need to make sure to terminate when we read d.