Context Free Languages: Problems for Week 3 with Model Solutions

Exercise 1 Andy has a grid of $2^n \times 2^n$ squares that are all white, except for one, which is red. A *triomino* is an L-shaped piece covering 3 squares. Show that Andy can cover the white part of the grid by triominoes.

Solution 1 *We proceed by induction on n. Base case: If $n = 0$, there is just the red square, so there's nothing to do. Inductive step: Suppose the result is true for n; we shall prove it for $n+1$. Given a grid of size $2^{n+1} \times 2^{n+1}$, divide it into four subgrids of size $2^n \times 2^n$. Each subgrid contains one of the grid's four central squares, and we paint each of them blue, except for the one that's in the same subgrid as the red square. Now each of the quarters contains a single non-white square, so, by the inductive hypothesis, it can be covered by triominos. Finally place a triomino over the three blue squares.*

Exercise 2 Show that every number $n > 1$ has a prime factor, by course-of-values induction.

Solution 2 *If n is prime, then it's a prime factor of itself. Otherwise n can be expressed as $a \times b$, where a and b are natural numbers and $1 < a < n$. By the inductive hypothesis, a has a prime factor, which is also a prime factor of n.*

Here's another proof. (You might take the view that it's just the same proof written a little differently.) Let $n_0 \overset{\text{def}}{=} n$. If n_0 isn't prime then let n_1 be a factor such that $1 < n_1 < n_0$. If n_1 isn't prime, then let n_2 be a factor such that $1 < n_2 < n_1$. And so on. If this process continues forever, we obtain an infinite sequence of natural numbers $n_0 > n_1 > \cdots$, which is absurd. So we must eventually reach a prime factor.

Exercise 3 *Consider the grammar over the alphabet $\{7, 3, +, \times, (,)\}$*

$$\Rightarrow M \quad ::= \quad 3 \mid 7 \mid M + M \mid M \times M \mid (M)$$

1. *Draw two derivation trees for the string $3 + (7 \times 3) \times 3$.*
2. *Write out the leftmost derivation for each of these derivation trees.*

Solution 3

1. *Two possible derivation trees are shown below:*

```
       M               |               M
     / \               |              / \
    /   \              |             /   \
   -------             |            -------
  M    +   M           |           M   x   M
 / \      / \          |          / \     / \
/   \   /   \          |         /   \   /   \
-----  -------         |        -------  -----
  3     M  x  M        |        M  +  M    3
       / \  / \        |       / \   / \
     /   \ /   \       |      /   \ /   \
     ----- -----       |      ----- -----
      (M)     3        |        3    (M)
       / \             |             / \
     /   \             |            /   \
    -------            |           -------
  M  x  M           |           M  x  M
 / \   / \          |          / \   / \
/   \ /   \         |         /   \ /   \
----- -----         |         ----- -----
  7     3           |           7     3
```

2. *First possible leftmost derivation is given below:*

$$\begin{aligned}
M &\Rightarrow M + M \\
&\Rightarrow 3 + M \\
&\Rightarrow 3 + M \times M \\
&\Rightarrow 3 + (M) \times M \\
&\Rightarrow 3 + (M \times M) \times M \\
&\Rightarrow 3 + (7 \times M) \times M \\
&\Rightarrow 3 + (7 \times 3) \times M \\
&\Rightarrow 3 + (7 \times 3) \times 3
\end{aligned}$$

The second possible leftmost derivation is:

$$\begin{aligned} M &\Rightarrow M \times M \\ &\Rightarrow M + M \times M \\ &\Rightarrow 3 + M \times M \\ &\Rightarrow 3 + (M) \times M \\ &\Rightarrow 3 + (M \times M) \times M \\ &\Rightarrow 3 + (7 \times M) \times M \\ &\Rightarrow 3 + (7 \times 3) \times M \\ &\Rightarrow 3 + (7 \times 3) \times 3 \end{aligned}$$

Exercise 4 *Consider the language generated by the grammar*

$$\Rightarrow S \ ::= \ \mathtt{b}SS \ | \ \mathtt{a}S \ | \ \mathtt{a}$$

and the string `aabbaaa`*.*

1. *Find a leftmost derivation for this string.*
2. *Draw the derivation tree.*

Solution 4

1. *Leftmost derivation is given below:*

$$\begin{aligned} S &\Rightarrow aS \\ &\Rightarrow aaS \\ &\Rightarrow aabSS \\ &\Rightarrow aabbSSS \\ &\Rightarrow aabbaSS \\ &\Rightarrow aabbaaS \\ &\Rightarrow aabbaaa \end{aligned}$$

2. *The derivation tree for the above string would be:*

```
  S
 / \
-----
a   S
   / \
  -----
  a   S
    /   \
   /     \
  /       \
-------------
b     S      S
    /   \   / \
   /     \ -----
  /       \  a
-------------
b     S     S
     / \   / \
    ----- -----
      a    a
```

Exercise 5 *Let's look at a "Natural Language" example. The alphabet is*

$\{$ *the, a, cat, dog, happy, tired, slept, died, ate, dinner, and, .* $\}$

The grammar is

Sentence	$\Rightarrow S$	$::=$	*C .*
Clause	*C*	$::=$	*NP VP* | *C and C*
Noun phrase	*NP*	$::=$	*Art N* | *dinner*
Noun	*N*	$::=$	*Adj N* | *cat* | *dog*
Adjective	*Adj*	$::=$	*happy* | *tired*
Verb phrase	*VP*	$::=$	*VI* | *VT NP*
Intransitive verb	*VI*	$::=$	*slept* | *died*
Transitive verb	*VT*	$::=$	*ate*
Article	*Art*	$::=$	*a* | *the*

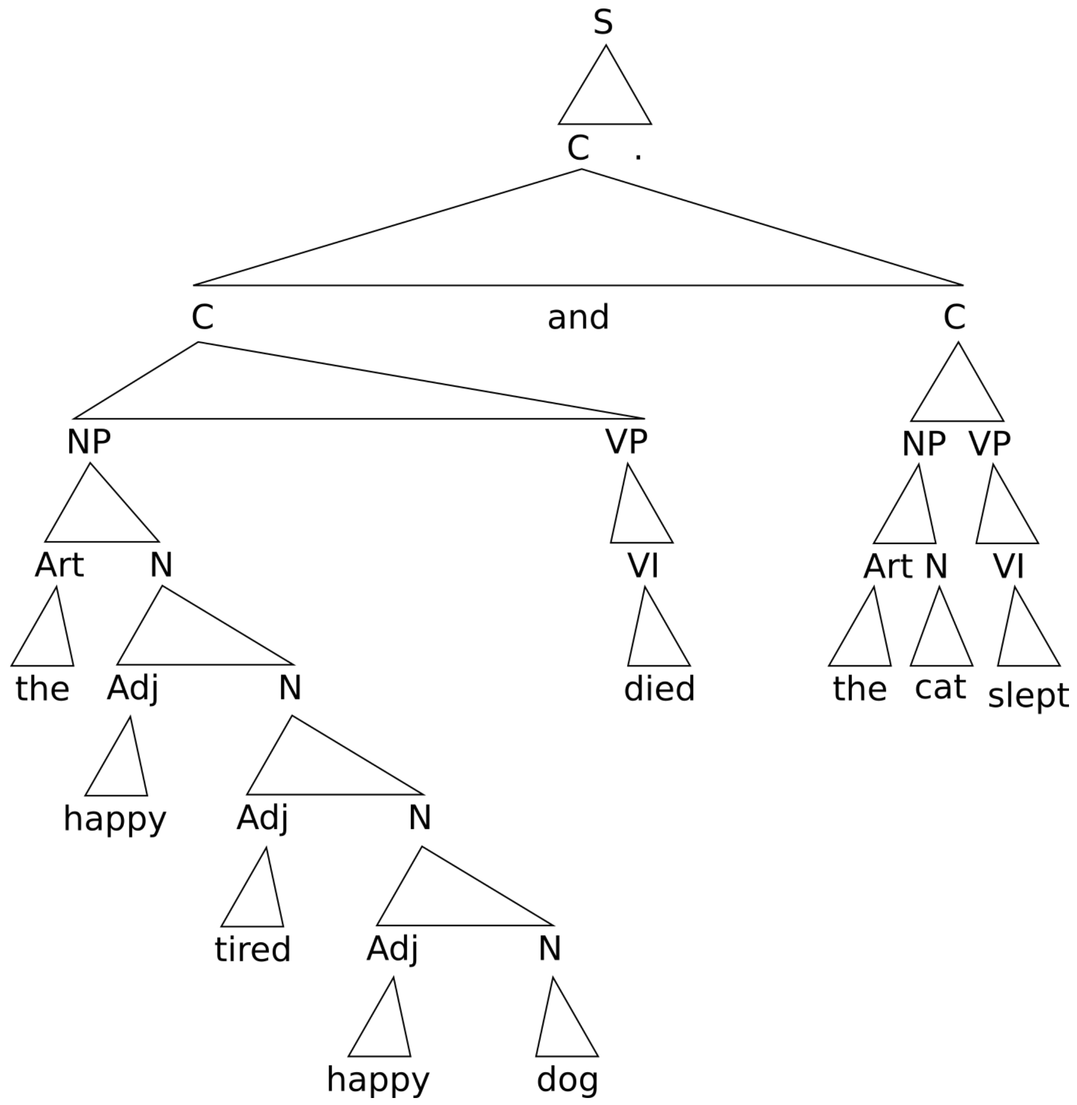
This grammar accepts "words" such as

the happy tired happy dog died and the cat slept.
the tired tired cat ate dinner.
dinner ate a happy dog.

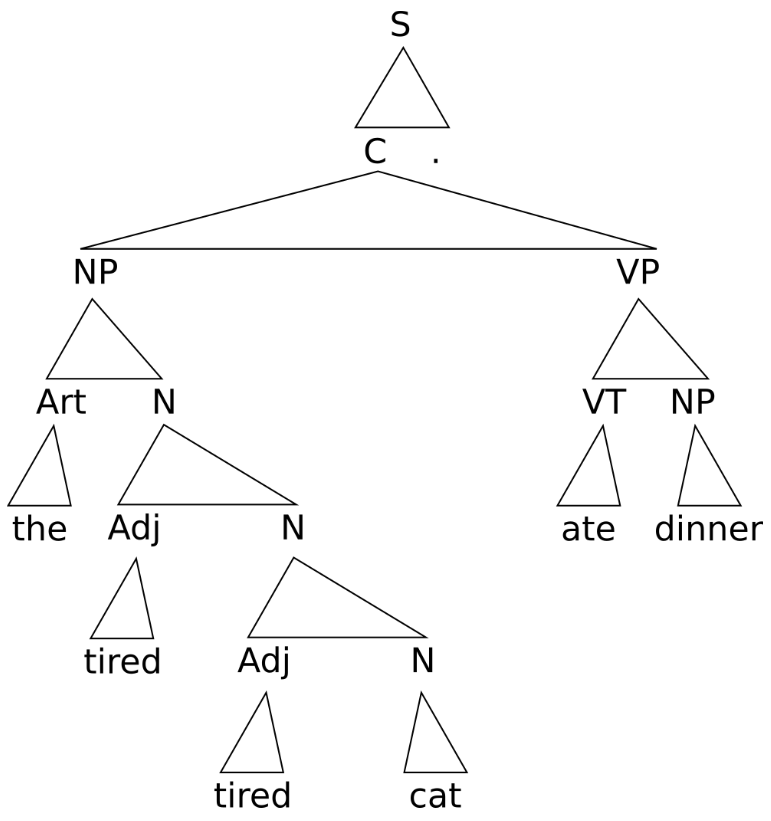
Try writing derivations and derivation trees for these sentences.

Solution 5

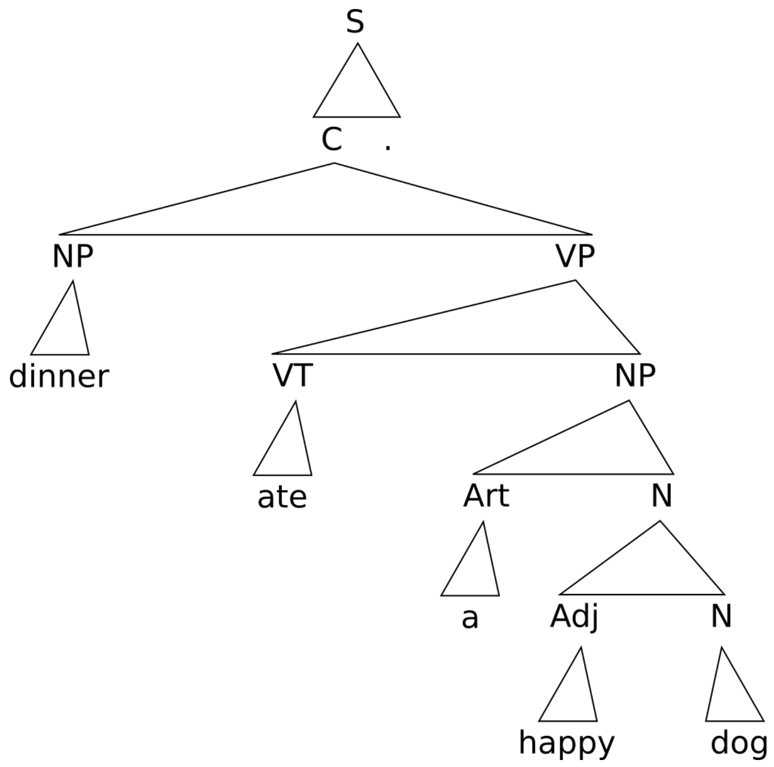
1. *The derivation tree for "the happy tired happy dog died and the cat slept."*



2. *The derivation tree for "the tired tired cat ate dinner."*

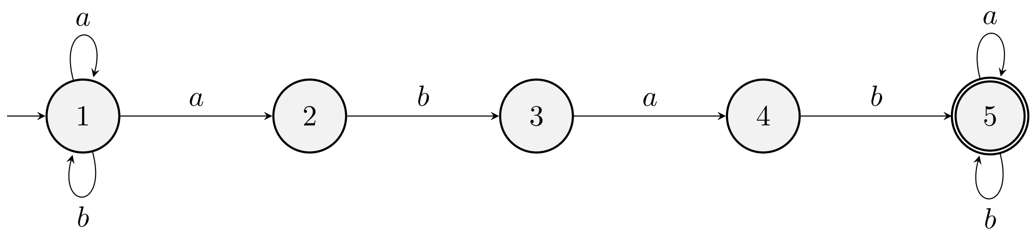


3. *The derivation tree for "dinner ate a happy dog."*



We will leave the derivations to you!

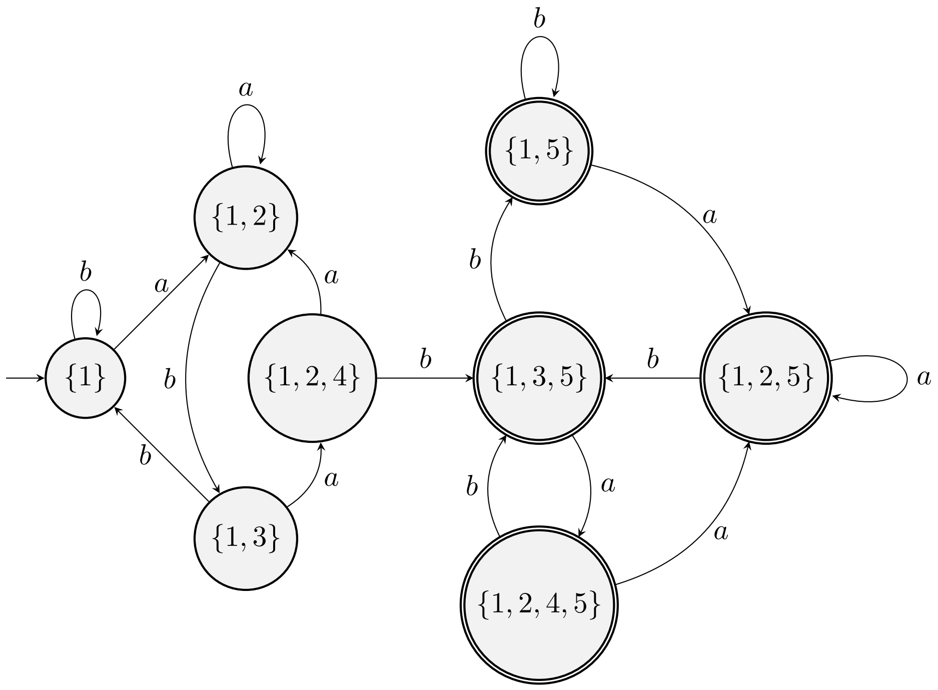
Exercise 6 *Lets consider an NFA that accepts any string that contains the substring "abab".*



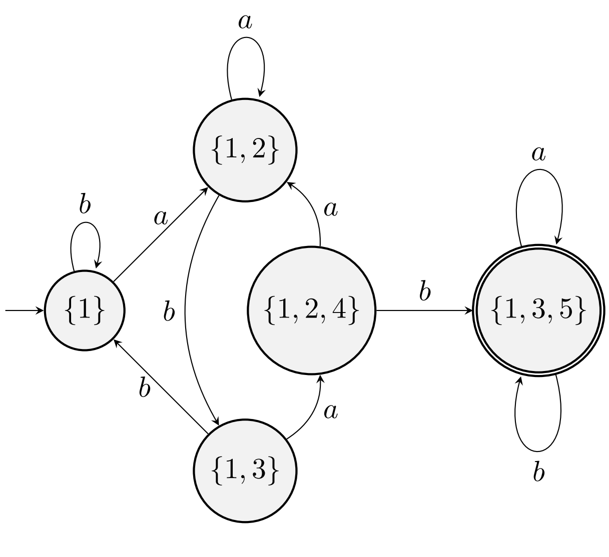
1. *Convert the above NFA into its equivalent total DFA.*
2. *Convert the resultant DFA in an equivalent CFG. It is suggested to minimize the DFA before writing CFG.*

Solution 6

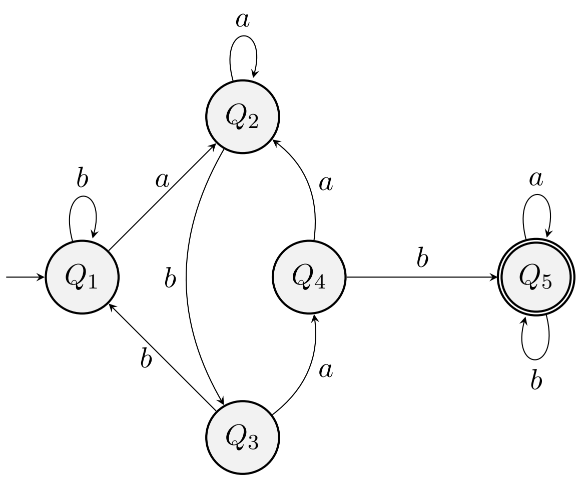
1. *On converting the NFA to its equivalent total DFA, we get the following:*



2. *We can see that the states* $\{1, 3, 5\}$, $\{1, 5\}$, $\{1, 2, 4, 5\}$, *and* $\{1, 2, 5\}$ *are equivalent and could be removed. On minimizing, we get the following DFA:*



We can rename the states, to make it easier for us to write the equivalent CFG.



Now, we can very easily write the CFG for the above DFA:

$$
\begin{array}{rcl}
R_1 & ::= & aR_2 \mid bR_1 \\
R_2 & ::= & aR_2 \mid bR_3 \\
R_3 & ::= & aR_4 \mid bR_1 \\
R_4 & ::= & aR_2 \mid bR_5 \\
R_5 & ::= & aR_5 \mid bR_5 \mid \varepsilon \\
\textit{Start:} & & R_1
\end{array}
$$

Exercise 7 *Give a context free grammar for the set of palindromes over the alphabet* $\{a, b\}$.

Solution 7

The CFG for set of palindromes over the alphabet $\{a, b\}$ *is given below:*

$$\Rightarrow S \quad ::= \quad \varepsilon \mid \texttt{a} \mid \texttt{b} \mid \texttt{aSa} \mid \texttt{bSb}$$

Exercise 8 *Try deriving the string* $3 + 5 \times 3$ *in two different ways using leftmost derivation only, using the grammar given below:*

$$
\begin{array}{rcl}
\Rightarrow A & ::= & A + B \mid B \\
B & ::= & B \times C \mid C \\
C & ::= & (A) \mid 3 \mid 5
\end{array}
$$

Solution 8

There is only one possible leftmost derivation, as shown below (because the grammar is unambiguous):

$$
\begin{array}{rcl}
A & \Rightarrow & A + B \\
& \Rightarrow & B + B \\
& \Rightarrow & C + B \\
& \Rightarrow & 3 + B \\
& \Rightarrow & 3 + B \times C \\
& \Rightarrow & 3 + C \times C \\
& \Rightarrow & 3 + 5 \times C \\
& \Rightarrow & 3 + 5 \times 3
\end{array}
$$

Exercise 9 *Show that the following grammar is ambiguous. The alphabet is* $\{\mathtt{a}, \mathtt{b}\}$.

$$\begin{array}{rcl} \Rightarrow P & ::= & \varepsilon \mid Q\mathtt{a} \mid \mathtt{aQ} \\ Q & ::= & \mathtt{aaP} \mid \mathtt{bR} \\ R & ::= & Q\mathtt{a} \end{array}$$

Solution 9

Here is the first leftmost derivation for the string "aaa":

$$\begin{array}{rcl} P & \Rightarrow & Qa \\ & \Rightarrow & aaPa \\ & \Rightarrow & aaa \end{array}$$

We can get the same "aaa" string using the following leftmost derivation:

$$\begin{array}{rcl} P & \Rightarrow & aQ \\ & \Rightarrow & aaaP \\ & \Rightarrow & aaa \end{array}$$

Therefore, the given grammar is ambiguous!

Exercise 10 *Convert the following CFG into an equivalent CFG in Chomsky normal form*

$$\begin{array}{rcl} \Rightarrow A & ::= & BAB \mid B \mid \varepsilon \\ B & ::= & 00 \mid \varepsilon \end{array}$$

Solution 10

1. *Add a new start variable* S_0

$$\begin{array}{rcl} \Rightarrow \boldsymbol{S_0} & \boldsymbol{::=} & \boldsymbol{A} \\ A & ::= & BAB \mid B \mid \varepsilon \\ B & ::= & 00 \mid \varepsilon \end{array}$$

2a. *Remove* ε*-rule* $B ::= \varepsilon$

$$\begin{array}{rcl} \Rightarrow S_0 & ::= & A \\ A & ::= & BAB \mid B \mid \boldsymbol{AB} \mid \boldsymbol{BA} \mid \boldsymbol{A} \mid \varepsilon \\ B & ::= & 00 \end{array}$$

2b. *Remove* ε*-rule* $A ::= \varepsilon$

$$\begin{array}{rcl} \Rightarrow S_0 & ::= & A \mid \boldsymbol{\varepsilon} \\ A & ::= & BAB \mid B \mid AB \mid BA \mid A \mid \boldsymbol{BB} \\ B & ::= & 00 \end{array}$$

Note: $S_0 ::= \varepsilon$ *is allowed in CNF.*

3a. *Remove the unit rule* $A ::= A$

$$\begin{array}{rcl} \Rightarrow S_0 & ::= & A \mid \varepsilon \\ A & ::= & BAB \mid B \mid AB \mid BA \mid BB \\ B & ::= & 00 \end{array}$$

3b. Remove the unit rule $A ::= \quad B$

$$\begin{aligned} \Rightarrow S_0 &::= \quad A \mid \varepsilon \\ A &::= \quad BAB \mid \mathbf{00} \mid AB \mid BA \mid BB \\ B &::= \quad 00 \end{aligned}$$

3c. Remove the unit rule $S_0 ::= \quad A$

$$\begin{aligned} \Rightarrow S_0 &::= \quad \boldsymbol{BAB} \mid \mathbf{00} \mid \boldsymbol{AB} \mid \boldsymbol{BA} \mid \boldsymbol{BB} \mid \varepsilon \\ A &::= \quad BAB \mid 00 \mid AB \mid BA \mid BB \\ B &::= \quad 00 \end{aligned}$$

4a. Convert the rules into proper form; introduce rule $D ::= \quad 0$

$$\begin{aligned} \Rightarrow S_0 &::= \quad BAB \mid \boldsymbol{DD} \mid AB \mid BA \mid BB \mid \varepsilon \\ A &::= \quad BAB \mid \boldsymbol{DD} \mid AB \mid BA \mid BB \\ B &::= \quad \boldsymbol{DD} \\ \boldsymbol{D} &\boldsymbol{::=} \quad \mathbf{0} \end{aligned}$$

4b. Convert the rules into proper form; introduce rule $C ::= \quad AB$

$$\begin{aligned} \Rightarrow S_0 &::= \quad BC \mid DD \mid AB \mid BA \mid BB \mid \varepsilon \\ A &::= \quad BC \mid DD \mid AB \mid BA \mid BB \\ B &::= \quad DD \\ \boldsymbol{C} &\boldsymbol{::=} \quad \boldsymbol{AB} \\ D &::= \quad 0 \end{aligned}$$

The above CFG is now in Chomsky Normal Form.

Exercise 11 *Give grammars for the following two languages:*

1. *All binary strings with both an even number of zeroes and an even number of ones.*
2. *All strings of the form $0^a 1^b 0^c$ where $a + c = b$.*

Solution 11

1. *All binary strings with both an even number of zeroes and an even number of ones.*

$$\begin{aligned} \Rightarrow S &::= \quad 0X \mid 1Y \mid \varepsilon \\ X &::= \quad 0S \mid 1Z \\ Y &::= \quad 1S \mid 0Z \\ Z &::= \quad 0Y \mid 1X \end{aligned}$$

The idea here is that:

(a) *Production S corresponds to having an even number of zeroes and an even number of ones.*

(b) *Production X corresponds to having an odd number of zeroes and an even number of ones.*

(c) *Production Y corresponds to having an even number of zeroes and an odd number of ones.*

(d) *Production Z corresponds to having an odd number of zeroes and an odd number of ones.*

You can check this yourself by producing some binary strings using the grammar and keeping track of the parity of the number of zeroes and ones as you move through the productions.

2. *All strings of the form* $0^a 1^b 0^c$ *where* $a + c = b$.

$$\begin{aligned} \Rightarrow S &::= \quad TU \\ T &::= \quad 0T1 \mid \varepsilon \\ U &::= \quad 1U0 \mid \varepsilon \end{aligned}$$

This works because for every 0 *generated on the right, we generate a* 1 *in the middle (see* U*) and for every* 0 *generated on the left, we generate another* 1 *in the middle (see* T*).*