CFL1 - Solution Sketches

February 16, 2019

We sketch here the solutions for the exercises of lecture CFL1 in a brief manner. Note that a proper solution would require more detailed descriptions, explanations, and in some cases examples. Some of the exercises may have more than one solution, and we just show one of them. When the exercise requests a context-free grammar, we provide one in a very compact way (as done in the slides) where we just provide the productions, with the implicit assumption that non-terminals start with capital letters, and that the initial symbol is the on corresponding to the first production.

Exercise 1.1.(a)

$$S \rightarrow \epsilon \mid 0S$$

To actually convince us that the above grammar G is a correct solution we need to prove that its language $L(G)$ coincides with $L = \{0^n \mid n \geq 0\}$. We can do this by proving that for every word w (1) $S \in L(G)$ implies $w \in L$ and (2) $w \in L$ implies $w \in L(G)$.

We show (1) by induction and using the bottom-up definition of a language (recursive inference). The base case is the production $S \rightarrow \epsilon$, from which we have that $\epsilon \in L(G)$. Trivially ϵ belongs also to L.

For the inductive step, we consider the production $S \rightarrow 0S$, from which we have that if a word $w' \in L(G)$ then $0w' \in L(G)$. Using the induction hypothesis $w' \in L(G)$ implies $w' \in L$. All we need to prove is that if $w' \in L$ then $0w' \in L$ which is trivial.

To show (2) we can proceed by induction on the length of $w = 0^n$. As base case take $n = 0$. In this case $w = \epsilon$. By using production $S \rightarrow \epsilon$ we know that $\epsilon \in L(G)$.

For the inductive step, we need to show that $w = 0^{n+1} \in L(G)$. By the inductive hypothesis we know that $0^n \in L(G)$. We can the use production $S \rightarrow 0S$ to conclude that $00^n = 0^{n+1}$ belongs to $L(G)$.

Exercise 1.1.(b)

$$S \rightarrow 0 \mid 0S$$

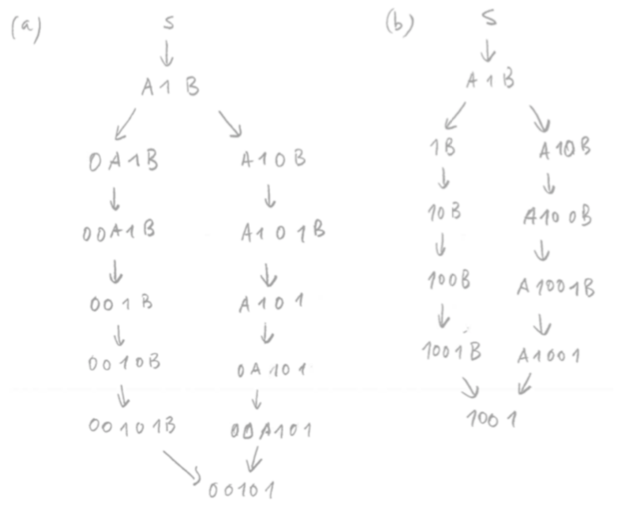
Exercise 1.1.(c)

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow 0 \mid 0A \\ B &\rightarrow 1 \mid 1B \end{aligned}$$

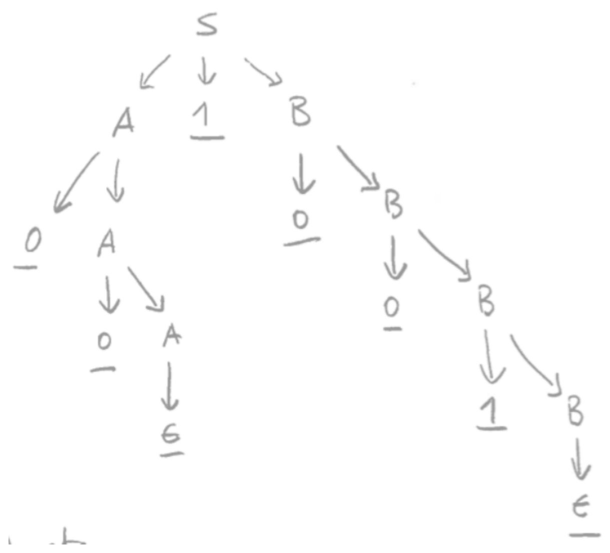
Exercise 1.1.(d)

$$S \rightarrow \epsilon \mid 0S1$$

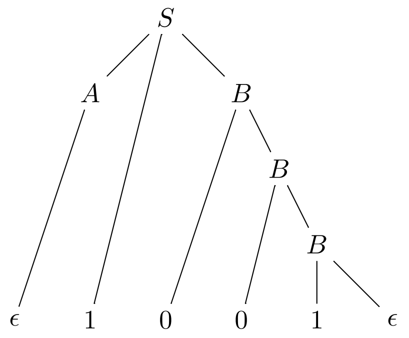
Exercise 1.2. For simplicity, the left- and right-most derivations are drawn in the same picture below.



Exercise 1.3.(a)



Exercise 1.3.(b)



Exercise 1.4.(b)

$$S \;\rightarrow\; \epsilon \;\mid\; SS \;\mid\; (S)$$

Exercise 1.4.(c)

$$S \;\rightarrow\; 010 \;\mid\; 0S0$$

Exercise 1.4.(d)

$$\begin{array}{rcl} S & \rightarrow & 0S2 \;\mid A \\ A & \rightarrow & \epsilon \;\mid\; 1A \end{array}$$

Exercise 1.4.(e)

$$\begin{array}{lcl} S & \rightarrow & AB \\ A & \rightarrow & \epsilon \mid 0A1 \\ B & \rightarrow & 1 \mid 1B \end{array}$$

Exercise 1.4.(f)

$$S \rightarrow 011 \mid 0S11$$

Exercise 1.5. For list without separators the grammar is simple:

$$\begin{array}{lcl} S & \rightarrow & \epsilon \mid AS \\ A & \rightarrow & 0 \mid 1 \end{array}$$

If, instead, we want to admit separators (for example, ",") we need a different grammar:

$$\begin{array}{lcl} S & \rightarrow & \epsilon \mid N \\ N & \rightarrow & A \mid A, N \\ A & \rightarrow & 0 \mid 1 \end{array}$$

intuitively, N represents non-empty lists.

Exercise 1.6.

$$\begin{array}{lcll} T & \rightarrow & \epsilon & \text{(an empty tree)} \\ & \mid & A & \text{(a leaf)} \\ & \mid & [T \bullet T] & \text{(a root } \bullet \text{ with two subtrees)} \\ A & \rightarrow & 0 \mid 1 & \end{array}$$

Exercise 1.7.a Yes.

Exercise 1.7.b It depends on the semantic interpretation of the words. For example, consider the following word:

$$0 \mapsto a, 0 \mapsto b$$

There are several ways of interpreting it as a map: as an ill-formed one (because the value of key 0 is not well-defined), as the map $0 \mapsto b$ (if we interpret the list as updates on a map, overwriting previous maps), or as the map $0 \mapsto a$ (if we discard overwriting already defined pairs of key/value).

Exercise 1.7.c It depends on the semantic interpretation of the words, but for a reasonable interpretation the answer is yes. Here are some examples

$$0 \mapsto a, 1 \mapsto b$$

and

$$1 \mapsto b, 0 \mapsto a$$

denote the same map (the order of single maps is irrelevant).

Also $0 \mapsto a$, and $0 \mapsto a, \mathsf{nil}$, and $0 \mapsto a, \mathsf{nil}, \mathsf{nil}$, etc. denote the same map (since nil is irrelevant).

Exercise 1.8.

$$E \quad \rightarrow \quad e \mid \emptyset \mid 0 \mid 1 \mid E+E \mid EE \mid E* \mid (E)$$

Exercise 1.9. We define a function cfg that, given a regular expression E (i.e. a word recognised by the grammar of Exercise 1.8), and a non-terminal symbol S provides a context-free grammar $\mathsf{cfg}(E, S)$ that recognises the same language, for S being its initial symbol. The function is defined by induction on the structure of E:

$$\begin{aligned}
\mathsf{cfg}(e) &= S_e \rightarrow \epsilon \\
\mathsf{cfg}(\emptyset) &= S_\emptyset \\
\mathsf{cfg}(0) &= S_0 \rightarrow 0 \\
\mathsf{cfg}(1) &= S_1 \rightarrow 1 \\
\mathsf{cfg}(E+E') &= S_{E+E'} \rightarrow S_E \mid S_{E'} \\
&\quad \cup \mathsf{cfg}(E) \cup \mathsf{cfg}(E') \\
\mathsf{cfg}(EE') &= S_{EE'} \rightarrow S_E S_{E'} \\
&\quad \cup \mathsf{cfg}(E) \cup \mathsf{cfg}(E') \\
\mathsf{cfg}(E*) &= S_{E*} \rightarrow \epsilon \mid S_E S \\
&\quad \cup \mathsf{cfg}(E, S_E) \\
\mathsf{cfg}((E)) &= S_{(E)} \rightarrow S_E \\
&\quad \cup \mathsf{cfg}(E)
\end{aligned}$$