Dr. Matthias Heizmann
Dominik Klumpp

Hand in until 10:00 on May 17, 2021
Discussion: May 17, 2021

Tutorial for Program Verification
Exercise Sheet 7

In this exercise sheet we work with the syntax and semantics definitions for the Boostan language.

Submit your solution by uploading it as PDF in ILIAS.

Exercise 1: Boostan Grammar 4 Points

In this exercise you should propose a syntax for the Boostan programming language. State a context-free grammar $\mathcal{G}_{\mathsf{Boostan}} = (\Sigma_{\mathsf{Boostan}}, N_{\mathsf{Boostan}}, P_{\mathsf{Boostan}}, S_{\mathsf{Boostan}})$ such that a word of the generated language is a program of (your version of) the Boostan language.

In the lecture slides we propose the grammar $\mathcal{G}_{\mathsf{I}} = (\Sigma_{\mathsf{I}}, N_{\mathsf{I}}, P_{\mathsf{I}}, S_{\mathsf{I}})$ for integer expressions, where $\Sigma_{\mathsf{I}} = \{-, +, *, /, \%, (,), 0, \dots, 9, a, \dots, z, A, \dots Z\}$,
$N_{\mathsf{I}} = \{X_{iexpr}, X_{num}, X_{num'}, X_{var}, X_{var'}\}$, $S_{\mathsf{I}} = X_{iexpr}$ and the following derivation rules.

$$
\begin{aligned}
P_{\mathsf{I}} = \{X_{iexpr} &\rightarrow (X_{iexpr}) \\
X_{iexpr} &\rightarrow \texttt{-}X_{iexpr} \\
X_{iexpr} &\rightarrow X_{iexpr}\texttt{+}X_{iexpr} | X_{iexpr}\texttt{-}X_{iexpr} | X_{iexpr}\texttt{*}X_{iexpr} | X_{iexpr}\texttt{/}X_{iexpr} | X_{iexpr}\texttt{\%}X_{iexpr} \\
X_{iexpr} &\rightarrow X_{var} \\
X_{iexpr} &\rightarrow X_{num} \\
X_{num} &\rightarrow \texttt{0}X_{num'} | \dots | \texttt{9}X_{num'} \\
X_{num'} &\rightarrow \texttt{0}X_{num'} | \dots | \texttt{9}X_{num'} | \varepsilon \\
X_{var} &\rightarrow \texttt{a}X_{var'} | \dots | \texttt{z}X_{var'} | \texttt{A}X_{var'} | \dots | \texttt{Z}X_{var'} \\
X_{var'} &\rightarrow \texttt{a}X_{var'} | \dots | \texttt{z}X_{var'} | \texttt{A}X_{var'} | \dots | \texttt{Z}X_{var'} | \texttt{0}X_{var'} | \dots | \texttt{9}X_{var'} | \varepsilon\}
\end{aligned}
$$

Next, we proposed the grammar $\mathcal{G}_{\mathsf{B}} = (\Sigma_{\mathsf{B}}, N_{\mathsf{B}}, P_{\mathsf{B}}, S_{\mathsf{B}})$ for Boolean expressions, where $\Sigma_{\mathsf{B}} = \Sigma_{\mathsf{I}} \cup \{!, \&\&, \|, ==>, ==, <, >, <=, >=\}$, $N_{\mathsf{B}} = N_{\mathsf{I}} \cup \{X_{bexpr}\}$, $S_{\mathsf{B}} = X_{bexpr}$ and the following derivation rules.

$$
\begin{aligned}
P_{\mathsf{B}} = \{X_{bexpr} &\rightarrow (X_{bexpr}) \\
X_{bexpr} &\rightarrow \texttt{!}X_{bexpr} \\
X_{bexpr} &\rightarrow X_{bexpr}\texttt{\&\&}X_{bexpr} | X_{bexpr}\|X_{bexpr} | X_{bexpr}\texttt{==}X_{bexpr} \\
X_{bexpr} &\rightarrow X_{iexpr}\texttt{==}X_{iexpr} | X_{iexpr}\texttt{<}X_{iexpr} | X_{iexpr}\texttt{>}X_{iexpr} | X_{iexpr}\texttt{<=}X_{iexpr} | X_{iexpr}\texttt{=>}X_{iexpr} \\
X_{bexpr} &\rightarrow X_{var} \\
X_{bexpr} &\rightarrow \textbf{true} | \textbf{false}\} \cup P_{\mathsf{I}}
\end{aligned}
$$

We propose that you use $\Sigma_{\mathsf{Boostan}} = \Sigma_{\mathsf{B}} \cup \{\texttt{while}, \texttt{if}, \texttt{else}, \{, \}, \texttt{;}, \texttt{:=}\}$ and your language should have the following properties.

- There should be a while statement, an if-then-else statement and an assignment statement.
- The concatenation of statements should be a statement.
- A program should be a statement and we do not need statements for declaring variables.

Exercise 2: Derivation Tree 1 Point

Give a derivation tree for the grammar $\mathcal{G}_{\mathsf{I}}$ and the word $15 + a + 4$.

Exercise 3: Semantics of Assignment 3 Points

In the lecture we defined the semantics for the assignment statement. In this exercise we use this definition and compute for statements the corresponding relations.

(a) Let $V = \{x, y\}$, $\mu(x) = \mathbb{Z}$ and $\mu(y) = \mathbb{Z}$.

Write down the relation $[\![\texttt{x := x-y;}]\!]$. Give a pair of states $(s_1, s_2) \in [\![\texttt{x := x-y;}]\!]$, and give a pair of states $(s_3, s_4) \notin [\![\texttt{x := x-y;}]\!]$.

(b) Let $V = \{x, y\}$, $\mu(x) = \{\textbf{true}, \textbf{false}\}$ and $\mu(y) = \{\textbf{true}, \textbf{false}\}$.

Write down the relation $[\![\texttt{x := !y;}]\!]$ by listing all elements of the relation explicitly.

In the next part of the lecture, we will continue to define the semantics of Boostan formally. For this purpose, we will use the *reflexive transitive closure* of a relation. The exercises below should make you familiar with that term.

Given a set X, a *binary relation* over X is a subset of the Cartesian product $X \times X$. We call a binary relation R *reflexive* if for all $x \in X$ the pair (x, x) is an element of R. We call a binary relation R *transitive* if for all $x, y, z \in X$ the following property holds:

$$\text{If } (x, y) \in R \text{ and } (y, z) \in R \text{ then } (x, z) \in R.$$

Exercise 4: Reflexivity and Transitivity 2 Points

State for each of the following relations if the relation is reflexive and if the relation is transitive.

(a) The "strictly smaller" relation over integers, $R_a = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} \mid x < y\}$

(b) The relation $\mathsf{win_{RPS}}$ over the set $\{\mathsf{Rock}, \mathsf{Paper}, \mathsf{Scissors}\}$.

$R_b = \{(\mathsf{Rock}, \mathsf{Scissors}), (\mathsf{Scissors}, \mathsf{Paper}), (\mathsf{Paper}, \mathsf{Rock})\}$

(c) $R_c = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} \mid (x - y) \bmod 42 = 0\}$

(d) $R_d = \{(x, y) \in \mathbb{Z} \times \mathbb{Z} \mid (x - y) = 2 \text{ or } (y - x) = 2\}$

Given a binary relation R over the set X, the *reflexive transitive closure*, denoted R^*, is the smallest relation such that $R \subseteq R^*$, R^* is reflexive and R^* is transitive.
We note that in this context "smallest" means that there is no strict subset that has the same properties and that this minimum is indeed unique since reflexive and transitive relations are closed under intersection.

Exercise 5: Reflexive Transitive Closure 2 Points

Compute for each of the four relations above the reflexive transitive closure.