Compiler Construction

Dr. Michael Petter, Raphaela Palenta SS 2019

Exercise Sheet 7

Assignment 7.1 Attribute Grammars

The following $LL(1)$ grammar implements the functionality of a pocket calculator. Every key of the calculator emits a token. The keys 0 to 9 emit the token `digit` and all other keys directly translate to the tokens `=`, `+`, `*`, `mw` (memory write), and `mr` (memory read).

rule	production	attribute system
1	*Start* ::= *Comp*	$v[0] := v[1] \quad m[1] := 0$
2	*Comp* ::= *Expr* `=`	$v[0] := v[1] \quad m[1] := m[0]$
3	| *Expr* `=` *Comp*	$v[0] := v[3] \quad m[1] := m[0] \quad m[3] := m[0]$
4	| *Expr* `mw` *Comp*	$v[0] := v[3] \quad m[1] := m[0] \quad m[3] := v[1]$
5	*Expr* ::= *Term* `+` *Expr*	$v[0] := v[1] + v[3] \quad m[1] := m[0] \quad m[3] := m[0]$
6	| *Term*	$v[0] := v[1] \quad m[1] := m[0]$
7	*Term* ::= *Atom* `*` *Term*	$v[0] := v[1] \cdot v[3] \quad m[1] := m[0] \quad m[3] := m[0]$
8	| *Atom*	$v[0] := v[1] \quad m[1] := m[0]$
9	*Atom* ::= `digit` *Number*	$v[0] := p[2] \cdot v[1] + v[2]$
10	| `mr`	$v[0] := m[0]$
11	*Number* ::= `digit` *Number*	$p[0] := p[2] \cdot 10 \quad v[0] := p[2] \cdot v[1] + v[2]$
12	| ε	$v[0] := 0 \quad p[0] := 1$

The result is computed in the v attribute of *Start*. The key `=` is used to evaluate the expression and clears the input. The value of the internal memory is stored in the attribute m which is initially zero. The key `mw` evaluates the expression and stores its value in the internal memory of the calculator, and clears the input. The key `mr` recalls this value. For each token `digit` the attribute v contains the digit as the natural number, i.e., $v \in [0, 9]$.

1. What is the result after parsing the following key strokes. For a parse error, write "err".
   - `3` `7` `+` `3` `=` — 40
   - `0` `1` `+` `3` `*` `3` `=` — 10
   - `0` `1` `+` `=` — err
   - `5` `+` `6` `mw` `3` `*` `mr` `=` — 33
   - `4` `mw` `2` `=` `mr` `=` — 4
2. Complete the definitions for the result value v and the content of the memory cell m so that the calculator has the described behavior. You may add other attributes as needed.

3. Which properties does the attributed grammar satisfy?
   - ☒ it is l-attributed
   - ☐ v is inherited
   - ☒ v is synthesized
   - ☒ m is inherited
   - ☐ m is synthesized
   - ☒ it is acyclic

Assignment 7.2 Strongly Acyclic Attribute Grammars

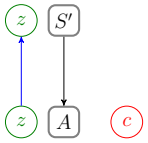
Consider Attribute Grammar G:

$$
\begin{array}{lcl|lll}
S' & \to & A^0 & z[0] := z[1] & c[1] := 0 & \\
A & \to & s\ B^0 & a[2] := y[2] & b[2] := c[0] & z[0] := x[2] \\
 & | & t\ B^1 & a[2] := c[0] & b[2] := x[2] & z[0] := y[2] \\
B & \to & u^0 & x[0] := a[0] & y[0] := b[0] & \\
 & | & v^1 & y[0] := x[0] & x[0] := 0 &
\end{array}
$$

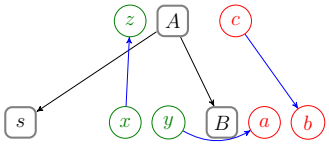
1. Draw the local dependency graphs for all production rules $p \in G$.
2. Enumerate all inputs and construct the dependency graphs.
3. Is G acyclic?

Suggested Solution 7.2

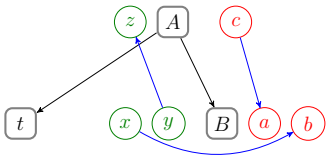
1. $D(S' \to A)$:



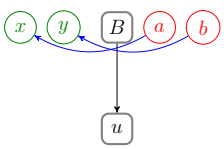
$D(A \to s\ B)$:



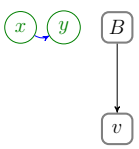
$D(A \to t\ B)$:



$D(B \to u)$:

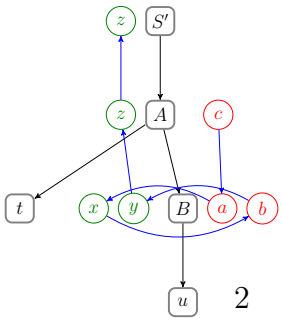


$D(B \to v)$:

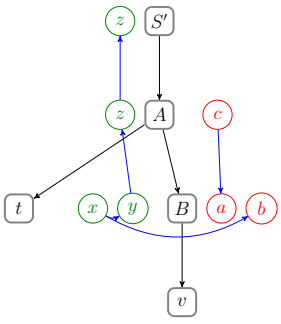


2. .

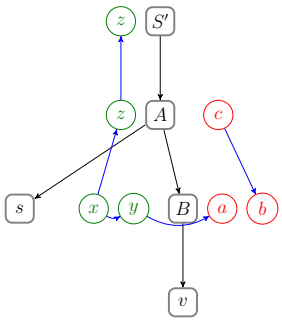
input tu:



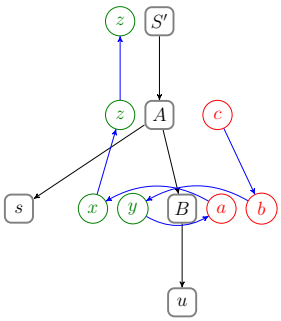
input tv:



input sv:



input su:



3. G is acyclic as there are no cyclic dependencies in the derivation trees.