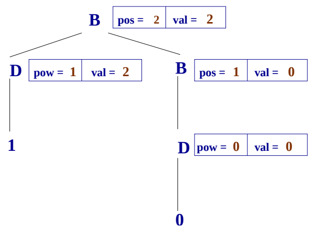

Figure 1: The decorated parse tree for $w = 10$.

Compiler Theory CS447b: Quiz 2. **UWO, March 20, 2003.**

Student ID number:

Guidelines. The quiz is closed book and all notes are forbidden. The duration of the quiz is 30 minutes. All answers should be written in the *answer boxes*. No justifications for the answers are needed. You are expected to do this quiz on your own without assistance from anyone else in the class. If possible, please avoid pencils and use pens with dark ink. Thank you.

Exercise 1. We consider the following syntax-directed definition with nonterminals B and D, terminals 0 and 1 and attributes $B.pos$, $B.val$, $D.pow$ and $D.val$.

Production	Semantic Rule
$B \longmapsto DB_1$	$B.pos := B_1.pos + 1$
	$B.val := B_1.val + D.val$
	$D.pow := B_1.pos$
$B \longmapsto D$	$B.pos := 1$
	$B.val := D.val$
	$D.pow := 0$
$D \longmapsto 0$	$D.val := 0$
$D \longmapsto 1$	$D.val := 2^{D.pow}$

Is the above syntax-directed definition L-attributed? Give a parse tree for $w = 10$. Then compute the attributes at each node.

Answer 1

No this is not an L-attributed syntax-directed definition since the inherited attribute $D.pow$ depends on the synthetized attribute $B_1.pos$ although B_1 appears on the right of D. See Figure 1 for the attributes at each node of the parse tree for $w = 10$.

Exercise 2. Consider the following syntax-directed definition for the point size and height of boxes containing mathematical formulas.

Production	Semantic Rule
$S \longmapsto B$	$B.ps := 10$
	$S.ht := B.ht$
$B \longmapsto B_1\ B_2$	$B_1.ps := B.ps$
	$B_2.ps := B.ps$
	$B.ht := \mathsf{max}(B_1.ht, B_2.ht)$
$B \longmapsto B_1\ \mathbf{sub}\ B_2$	$B_1.ps := B.ps$
	$B_2.ps := \mathsf{shrink}(B.ps)$
	$B.ht := \mathsf{disp}(B_1.ht, B_2.ht)$
$B \longmapsto \mathbf{text}$	$B.ht := \mathbf{text}.h \times B.ps$

Write a translation scheme for this syntax-directed definition.

Answer 2

S	$\longmapsto$		$\{B.ps := 10\}$
		B	$\{S.ht := B.ht\}$
B	$\longmapsto$		$\{B_1.ps := B.ps\}$
		B_1	$\{B_2.ps := B.ps\}$
		B_2	$\{B.ht := \mathsf{max}(B_1.ht, B_2.ht)\ \}$
B	$\longmapsto$		$\{B_1.ps := B.ps\}$
		B_1	
		sub	$\{\ B_2.ps := \mathsf{shrink}(B.ps)\ \}$
		B_2	$\{\ B.ht := \mathsf{disp}(B_1.ht, B_2.ht)\ \}$
B	$\longmapsto$	**text**	$\{\ B.ht := \mathbf{text}.h \times B.ps\ \}$

Exercise 3. We consider the following grammar. As in the course, the nonterminals P, D, S, T, E stand for *Program*, *Declaration*, *Statement*, *Type*, *Expression*. The terminals **id**, **boolean**, **integer**, **literal**, **num** stand for *identifier*, *boolean* (as a type), *integer* (as a type), *boolean literal* (as a value), *integer number* (as a value).

$$
\boxed{\begin{array}{l}
P \longmapsto D;\ S \\
D \longmapsto D;\ D \\
D \longmapsto \mathbf{id} : T \\
T \longmapsto \mathbf{boolean} \\
T \longmapsto \mathbf{integer} \\
E \longmapsto \mathbf{literal} \\
E \longmapsto \mathbf{num} \\
E \longmapsto \mathbf{id} \\
E \longmapsto E + E \\
E \longmapsto E = E \\
S \longmapsto S_1;\ S_2 \\
S \longmapsto E \\
S \longmapsto \mathbf{id}{:=}\, S_1 \\
S \longmapsto \mathbf{if}\ E\ \mathbf{then}\ \{S_1\} \\
S \longmapsto \mathbf{while}\ E\ \mathbf{repeat}\ \{S_1\}
\end{array}}
$$

There's one sligth difference w.r.t. the grammar of the course in the *Type checking* chapter: each valid statement has a value and thus a type (rather than no value and type `void`). The following rules compute the value of a valid statement.

- if S_1 and S_2 are valid statements then the value of $S_1;\ S_2$ is that of S_2.
- if $\boxed{\mathbf{id}{:=}\, S_1}$ is a valid statement then its value is that of S_1.
- if $\boxed{\mathbf{if}\ E\ \mathbf{then}\ \{S_1\}}$ is a valid statement then its value is that of S_1.
- if $\boxed{\mathbf{while}\ E\ \mathbf{repeat}\ \{S_1\}}$ is a valid statement then its value is that of S_1.

We associate attributes $T.type$, $E.type$, $S.type$, to the grammar symbols T, E, S. Each of these attributes may be **boolean**, **integer** or **type_error**. Then, for instance, the statement $\boxed{\mathbf{while}\ E\ \mathbf{repeat}\ \{S_1\}}$ is valid as soon as $E.type = \mathbf{boolean}$ and $S_1.type \neq \mathbf{type\_error}$. The attribute $\mathbf{id}.entry$ refers to the entry of **id** in the symbol table.

Question. Complete the following type checker for the above grammar.

Answer 3

$D \longmapsto \mathbf{id} : T$		{ addtype($\mathbf{id}.entry, T.type$) }
$T \longmapsto \mathbf{boolean}$	{ $T.type$:=	$\mathbf{boolean}$ }
$T \longmapsto \mathbf{integer}$	{ $T.type$:=	$\mathbf{integer}$ }
$E \longmapsto \mathbf{literal}$	{ $E.type$:=	$\mathbf{boolean}$ }
$E \longmapsto \mathbf{num}$	{ $E.type$:=	$\mathbf{integer}$ }
$E \longmapsto \mathbf{id}$	{ $E.type$:=	lookup($\mathbf{id}.entry$) }
$E \longmapsto E_1 + E_2$	{ $E.type$:=	if $E_1.type = \mathbf{integer}$ and $E_2.type = \mathbf{integer}$ then $\mathbf{integer}$ else $\mathbf{type\_error}$ }
$E \longmapsto E_1 = E_2$	{ $E.type$:=	if $E_1.type = E_2.type$ and $E_1.type \neq \mathbf{type\_error}$ then $\mathbf{boolean}$ else $\mathbf{type\_error}$ }
$S \longmapsto S_1;\ S_2$	{ $S.type$:=	if $S_1.type \neq \mathbf{type\_error}$ and $S_2.type \neq \mathbf{type\_error}$ then $S_2.type$ else $\mathbf{type\_error}$ }
$S \longmapsto E$	{ $S.type$:=	$E.type$ }
$S \longmapsto \mathbf{id}{:=}S_1$	{ $S.type$:=	if $S_1.type =$lookup($\mathbf{id}.entry$) then $S_1.type$ else $\mathbf{type\_error}$ }
$S \longmapsto \mathbf{if}\ E\ \mathbf{then}\ \{S_1\}$	{ $S.type$:=	if $E.type = \mathbf{boolean}$ and $S_1.type \neq \mathbf{type\_error}$ then $S_1.type$ else $\mathbf{type\_error}$ }
$S \longmapsto \mathbf{while}\ E\ \mathbf{repeat}\ \{S_1\}$	{ $S.type$:=	if $E.type = \mathbf{boolean}$ and $S_1.type \neq \mathbf{type\_error}$ then $S_1.type$ else $\mathbf{type\_error}$ }