← • HOME • UP • →

Top/Bottom-up parsing/Shift-reduce

9.1. Shift-Reduce Parsing (Dragon Book pages 235–238)

Right-sentential forms and handles

Suppose that G is a grammar.

Recall that a *sentential form* of G is a sequence of tokens and nonterminals that can be derived from the start nonterminal.

Since a bottom-up parser does a rightmost derivation, it is to our advantage to focus attention on rightmost derivations.

Definition. A *right-sentential form* of G is a sequence of tokens and nonterminals that can be derived from the start nonterminal in a rightmost derivation.

Remember that the parser constructs a rightmost derivation *backwards*.

At any given point, it finds the right-hand side of a production, called a *handle*, and replaces the handle by the left-hand side of the production.

Handle pruning

For example, let's use our simple expression grammar.

Suppose that the parser is currently working on right-sentential form F * **n**.

The handle is F. That is the right-hand side of production $T \rightarrow F$. Replacing F by T yields T * **n**.

Here is a parse of **n** * **n** based on finding handles. The handle is shown in red.

Right-sent. form	Production
n * **n**	$F \rightarrow$ **n**
F * **n**	$T \rightarrow F$
T * **n**	$F \rightarrow$ **n**
$T * F$	$E \rightarrow T * F$
E	

If you read off the right-sentential forms from the end to the start, you get

$$
\begin{aligned}
E &\Rightarrow T * F \\
&\Rightarrow T * \mathbf{n} \\
&\Rightarrow F * \mathbf{n} \\
&\Rightarrow \mathbf{n} * \mathbf{n}
\end{aligned}
$$

Notice that the derivation is rightmost.

An obvious issue is how the parser can know what the handle is. That is the subject of later pages.

For now, let's see a convenient way of carrying out a bottom-up parse, assuming that some way has been found to identify handles.

Shift-reduce parsing

A *shift-reduce parser* keeps track of two things:

1. the remaining, unread, part of the input;
2. a stack that holds tokens and nonterminals.

The handle is always the top one or more symbols in the stack.

There are two main kinds of actions.

1. A *shift* action moves a token from the input to the top of the stack.
2. A *reduce* action finds a handle α on the stack and a production $N \rightarrow \alpha$, and replaces α by N.

There are also two minor actions.

1. An *accept* indicates that the parser has sucessfully found a derivation.
2. An *error* action indicates a syntax error.

Let's do an example parse of **n * n** using our expression grammar.

The stack is shown with its top at the right end.

Also, I will follow the Dragon Book's convention of showing a $ at the bottom of the stack and at the end of the input.

Initially, the stack contains only its bottom marker, $.

The handle is shown in red. There is no handle for a shift action.

Stack	Input	Action
$	**n * n $**	Shift
$ n	*** n $**	Reduce by $F \rightarrow$ **n**
$ F	*** n $**	Reduce by $T \rightarrow F$

$ *T*	***** **n** **$**	Shift
$ *T* *****	**n** **$**	Shift
$ *T* ***** **n**	**$**	Reduce by $F \rightarrow$ **n**
$ *T* ***** *F*	**$**	Reduce by $T \rightarrow T * F$
$ *T*	**$**	Reduce by $E \rightarrow T$
$ *E*	**$**	Accept

Now we need to turn to the issue of determining which action to perform.

← • HOME • UP • →

Top/Bottom-up parsing/Shift-reduce