Weak Precedence Parsing

Weak Precedence Parsing Algorithm

Input: A weak precedence matrix (simple precedence matrix including any <= conflicts)
A weak precedence grammar
Sentence with ⊥ appended to the end

Assumptions: ⊥ < Vi > ⊥ where Vi ∈ V, ⊥ ∉ V

Algorithm:

1. Push ⊥ onto stack
2. Read first input symbol and push it onto stack
3. Do
   3.1 Obtain WP relation between the symbol on top of the stack and the next input symbol
   3.2 If WP relation is <, =, or <=
       Then
       3.2.1 Stack input symbol.
   Else { relation is > }
       3.2.2 Match the longest production in the grammar with the top of the stack
       3.2.3 Replace the matched RHS with its LHS on the stack
4. Until ⊥ is the input, and ⊥ and the start symbol are on the stack

Note: The algorithm above does not detect any syntax errors.

Modify the Weak Precedence Parsing Algorithm above to include the detection of weak precedence parsing errors.

Def: A weak precedence parse configuration triple is (S, α, π) where S is the stack, α is the input string, and π is the action sequence.

Initially, we have (⊥, α, λ) where ⊥ represents the empty stack and α is the input string. We terminate when we have (S, λ, π) where S is the start symbol and π is the action sequence. Using the grammar and weak precedence matrix below, we construct the weak precedence parse configuration sequence for the input string V * ( V + V )
(⊥, \quad V^*(V+V)⊥, \quad \lambda)

1. \quad E \to E + T
2. \quad \mid T
3. \quad T \to T * P
4. \quad \mid P
5. \quad P \to (E)
6. \quad \mid V

\[\begin{align*}
E & \quad T & \quad P & + & * & ( & ) & V \\
E & \quad . & \quad . & \quad = & \quad . & \quad . & \quad = & \quad . \\
T & \quad . & \quad . & \quad G & \quad = & \quad . & \quad G & . \\
P & \quad . & \quad . & \quad G & \quad G & \quad . & \quad G & . \\
+ & \quad . & \quad 4 & \quad L & \quad . & \quad L & \quad . & \quad L \\
* & \quad . & \quad . & \quad = & \quad . & \quad L & \quad . & \quad L \\
( & \quad . & \quad 4 & \quad L & \quad L & \quad . & \quad L & \quad L \\
) & \quad . & \quad . & \quad G & \quad G & \quad . & \quad G & . \\
V & \quad . & \quad . & \quad G & \quad G & \quad . & \quad G & . 
\end{align*}\]

**Note:** The reverse RMDS is 6, 4, 6, 4, 2, 6, 4, 1, 5, 3, 2
ANSWERS:

(⊥ , V*(V+V)⊥ , λ )

(⊥ V, *( V + V )⊥, S )

(⊥ P, *( V + V )⊥, R6 )

(⊥ T, *( V + V )⊥, R4 )

(⊥ T *, ( V + V )⊥, S )

(⊥ T * ( V, + V )⊥, S )

(⊥ T * ( P, + V )⊥, R6 )

(⊥ T * ( T, + V )⊥, R4 )

(⊥ T * ( E, + V )⊥, R2 )

(⊥ T * ( E +, V )⊥, S )

(⊥ T * ( E + V, )⊥, S )

(⊥ T * ( E + P, )⊥, R6 )

(⊥ T * ( E + T, )⊥, R4 )

(⊥ T * ( E, )⊥, R1 ) (Longest Match)

(⊥ T * ( E ), ⊥, S )

(⊥ T * P, ⊥, R5 )

(⊥ T, ⊥, R3 ) (Longest Match)

(⊥ E, ⊥, R2 )