Error recovery in predictive parsing

Error recovery in predictive parsing

- An error is detected during the predictive parsing when the terminal on top of the stack does not match the next input symbol, or when nonterminal A on top of the stack, a is the next input symbol, and parsing table entry M[A,a] is empty.
- Panic-mode error recovery is based on the idea of skipping symbols on the input until a token in a selected set of synchronizing tokens.

How to select synchronizing set?

1. Place all symbols in FOLLOW(A) into the synchronizing set for nonterminal A. If we skip tokens until an element of FOLLOW(A) is seen and pop A from the stack, it likely that parsing can continue.
2. We might add keywords that begins statements to the synchronizing sets for the nonterminals generating expressions.
3. Add FIRST(A) to the synchronizing set so as to resume parsing according to A.

How to select synchronizing set? (II)

4.If a nonterminal can generate the empty string, then the production deriving ε can be used as a default. This may postpone some error detection.This approach reduces the number of nonterminals that have to be considered during error recovery.

5. If a terminal on top of stack cannot be matched, a simple idea is to pop the terminal, issue a message saying that the terminal was inserted.

Example: Error recovery

"synch" indicating synchronizing tokens obtained from FOLLOW set of the nonterminal in question.

If the parser looks up entry M[A,a] and finds that it is blank, the input symbol a is skipped.

If the entry is synch, the the nonterminal on top of the stack is popped.

If a token on top of the stack does not match the input symbol, then we pop the token from the stack.

$$\text{FIRST}(E) = \text{FIRST}(T) = \text{FIRST}(F) = \{(, \textbf{id}\}.$$

$$\text{FIRST}(E') = \{+, \epsilon\}$$

$$\text{FIRST}(T') = \{*, \epsilon\}$$

$$\text{FOLLOW}(E) = \text{FOLLOW}(E') = \{), \$\}$$

$$\text{FOLLOW}(T) = \text{FOLLOW}(T') = \{+,), \$\}$$

$$\text{FOLLOW}(F) = \{+, *,), \$\}$$

NONTER-MINAL	INPUT SYMBOL					
	id	+	*	(	)	$
E	$E \to TE'$			$E \to TE'$	synch	synch
E'		$E' \to +TE'$			$E' \to \epsilon$	$E' \to \epsilon$
T	$T \to FT'$	synch		$T \to FT'$	synch	synch
T'		$T' \to \epsilon$	$T' \to *FT'$		$T' \to \epsilon$	$T' \to \epsilon$
F	$F \to \textbf{id}$	synch	synch	$F \to (E)$	synch	synch

Fig. 4.18. Synchronizing tokens added to parsing table of Fig. 4.15.

Example: error recovery (II)

STACK	INPUT	REMARK
$E	) id * + id $	error, skip)
$E	id * + id $	id is in FIRST(E)
$E'T	id * + id $	
$E'T'F	id * + id $	
$E'T'id	id * + id $	
$E'T'	* + id $	
$E'T'F*	* + id $	
$E'T'F	+ id $	error, M[F, +] = synch
$E'T'	+ id $	F has been popped
$E'	+ id $	
$E'T+	+ id $	
$E'T	id $	
$E'T'F	id $	
$E'T'id	id $	
$E'T'	$	
$E'	$	
$	$	

Fig. 4.19. Parsing and error recovery moves made by predictive parser.

Phrase level

- Filling in the blank entries in the parse table with pointers to error routines.