

Plan for Today

Error recovery goals

Review panic mode error recovery for predictive parsers

Panic mode for LR parsers

Error recovery using error symbol in productions

- available in YACC

Lines and positions information

- in general how could it be improved?
- can we actually implement such improvements with SableCC?

Where should an overflow number error be handled?

CS453 Lecture

Error Recovery II

1

Error Handling Goals

Provide program with a list of as many errors as possible

Provide USEFUL error messages

- appropriate line and position information
- guidance for fixing the error

Avoid infinite loops or recursion

Find “all” errors in program before translation begins

CS453 Lecture

Error Recovery II

2

Predictive parser using panic mode error recovery

```
eat(tok) { if (tok==t) advance(); else throw ...; }
panic(tokset) { while (getToken() not in tokset); }
advance() { tok = getToken(); }
void S() { switch (tok) {
  case ID: case EOF:
    try { StmList(); } catch { panic(FOLLOW(StmList)); }
    eat(EOF); break;
  default: print "error"; panic(FOLLOW(S)); break;
}}
void StmList() { switch (tok) {
  case ID:
    try { Stm(); } catch { panic(FOLLOW(Stm)); }
    try { StmList(); } catch { panic(FOLLOW(StmList)); } break;
  case EOF:
    break;
  default: print "error"; panic(FOLLOW(StmList)); break;
}}
void Stm() { switch (tok) {
  case ID: eat(id); eat(assign); eat(float);
    break;
  default: print "error"; panic(FOLLOW(Stm)); break;
}}
```

CS453 Lecture

Error Recovery II

3

Grammar 3.1 from Tiger book

```
(0) S' -> S $
(1) S -> S ; S
(2) S -> id := E
(3) S -> print (L)
(4) E -> id
(5) E -> num
(6) E -> E + E
(7) E -> (S,E)
(8) L -> E
(9) L -> L, E
```

CS453 Lecture

Error Recovery II

4

LR parse table (Table 3.19)

	id	num	print	;	,	+	:=	()	\$	S	E	L
1	s4		s7								g2		
2				s3						a			
3	s4		s7								g5		
4							s6						
5				r1	r1					r1			
6	s20	s10					s8					g11	
7							s9						
8	s4		s7								g12		
9	s20	s10					s8					g15	g14
10				r5	r5	r5		r5	r5				
11				r2	r2	s16				r2			
12				s3	s18								
13				r3	r3					r3			
14					s19				s13				
15				r8					r8				

CS453 Lecture

Error Recovery II

5

LR parse table (Table 3.19) cont...

	id	num	print	;	,	+	:=	()	\$	S	E	L
16	s20	s10						s8					g17
17				r6	r6	s16			r6	r6			
18	s20	s10						s8					g21
19	s20	s10						s8					g23
20				r4	r4	r4			r4	r4			
21										s22			
22				r7	r7	r7			r7	r7			
23				r9	s16				r9				

CS453 Lecture

Error Recovery II

6

Error recovery using an error symbol

```
exp -> ( error )
exps -> error ; exp
```

Steps taken when error occurs

- (0) generate error indicating expected token(s)
- (1) pop off stack until have state with shift action for error token
- (2) shift the error token
- (3) throw away input tokens until hit token with non-error action
- (4) resume parsing

CS453 Lecture

Error Recovery II

7

Suggested Exercises

Write a predictive parser using panic mode error recovery for the grammar shown below.

```
(0) S -> E $
(1) E -> B E'
(2) E' -> 'or' B E'
(3) E' ->
(4) B -> t | f
```

Show the Stack, Input, and Action table (see Figure 3.18) using the parse table in Figure 3.19 where the parser is using panic mode recovery for the following inputs

```
:= b + c - ; $
( d := 5 + 6, 3 ) $
( ( ) ) $
```

CS453 Lecture

Error Recovery II

8