**컴파일러의 기초 Project 2 보고서**

전기정보공학부 2016-10769 이권형

**Conflict 발생과 그 해결**

grammar를 그대로 사용하면 몇 가지 conflict가 발생한다. 그 종류는 shift/reduce conflict 또는 reduce/reduce로 두 가지인데, LR parsing의 경우 precedence를 설정하여 conflict가 발생하는 두 action중 하나를 선택하면 이를 해결할 수 있다.

1) Unary operation conflict

Operation &, -, \* 는 binary operation으로 사용될 수도 있고, 이로 인해 이 operation들에 대해 shift/reduce conflict가 발생한다. 즉, binary의 context와 unary의 context를 구분하여야 한다.

State 79

79 unary: unary . PLUS\_PLUS

80 | unary . MINUS\_MINUS

81 | '&' unary .

83 | unary . '[' expr ']'

84 | unary . STRUCTOP ID

85 | unary . '(' args ')'

86 | unary . '(' ')'

PLUS\_PLUS shift, and go to state 105

MINUS\_MINUS shift, and go to state 106

STRUCTOP shift, and go to state 107

'[' shift, and go to state 108

'(' shift, and go to state 109

‘[‘ [reduce using rule 81 (unary)]

‘(‘ [reduce using rule 81 (unary)]

$default reduce using rule 81 (unary)

위는 bison의 output을 나타낸 것이다. &에 대해서, lookahead가 ‘[‘이나 ‘(‘일 때 shift를 통해 &unary를 unary operation으로 쓰일 때로 해석하여 unary로 reduce할 지, binary로 쓰이는 것으로 해석하여 binary에 해당하는 state로 shift할 지 충돌한다. 다른 operation도 이와 같은 conflict가 발생하는데, 이를 해결하기 위해, &, -, \*에 %prec을 이용해 ‘!’와 같은 precedence를 주어 우선순위를 unary로 해석하는 것에 두었다. 따라서 reduce를 택할 것이고, 이로 인해 unary로 해석될 것이다.

2) if stmt else stmt conflict

if else 구문에서, 다음과 같이 conflict가 발생한다.

State 142

43 stmt: IF '(' test ')' stmt .

44 | IF '(' test ')' stmt . ELSE stmt

ELSE shift, and go to state 146

ELSE [reduce using rule 43 (stmt)]

$default reduce using rule 43 (stmt)

위와 같이, ELSE에 대해서 shift를 택하면 if stmt if stmt else stmt에 대해 else에서 자신과 가까운 if를 택하게 되고, reduce하면 if stmt if stmt else stmt -> if stmt (if stmt) else stmt로 해석되어 자신과 가장 먼, 첫 번째로 나오는 if를 택하게 된다.

따라서 C grammar와 같이 하려면 자신과 가까운 if를 택하도록 shift를 선택해야 한다. 따라서 43보다 44에 높은 precedence를 주기 위해, ELSE보다 precedence가 낮은 THEN이라는 nonassoc token을 새로 만들고, ELSE 또한 nonassoc으로 지정한 후에 43에 %prec THEN을 통해 낮은 precedence를 지정한다.

위 방법들을 이용하여 성공적으로 conflict를 모두 해결할 수 있다.