Bull. Korean Math. Soc.
Vol. 12, No. 2, 1975

컴파일러 구성시의 표현형식

申鉉千 · 鄭起東

1. 序 言.

Code generation phase는 compiler에서 실제로 internal source program을 machine language나 assembly language로 번역하는 phase이다.

이 phase에 의하여 원래의 source program에 대응하는 object program이 生成된다.

本 論文에서는 MELCOM-83 FORTRAN을 source language로하여 MELCOM-83 machine code를 生成하여 보았다.

원래의 source program을 internal source program 으로 바꾸기 위하여 postfix polish concept를 사용하였다.

2. Code generation.

code generation phase에서는 다음 3個의 element를 사용한다.

(1) AAC (Address Assignment Counter)

生成된 code에 address를 assign하기 위해서 사용된다. Code가 生成될 때 마다 그 code의 address가 주어지고 AAC의 값도 증가한다.

(2) ACC (Accumulator)

object program을 수행할 때 계산의 중간 結果를 저장하기 위해서 ACC가 사용된다. 한 operator에 對한 code를 生成할때는 그 operator와 2개의 top stack operand를 사용하며 2개의 operand대신 이 operation의 結果를 나타내는 ACC의 주소가 저장된다.

(3) STACK (stack)

Internal source program 이 postfix polish notation으로 되어있으므로 왼쪽으로부터 오른쪽으로 한번만 process하면 된다. 이때 operand를 저장하기 위해서 stack이 사용된다.

A. Code-generation for unlablled arithmatic statements.

Label이 없는 arithmetic statement에 대한 code 生成은 다음과 같다.

(1) 다음의 symbol이 operand이면 STACK에 저장하고 다음 symbol을 조사한다.

(2) 다음의 symbol이 operator이면 STACK으로부터 2개의 Operand (Unary operator이면 1개의 operand)를 사용해서 code를 生成하고, 사용한 operand들

대신에 ACC의 address를 STACK에 저장한다. Code를 1개 生成할 때마다 AAC의 값을 증가시켜서 address를 증가시킨다.

(3) 이과정을 "="이 나타날때까지 계속한다.

B. Code-generation for labelled statements and reserved word statements.

a. Labelled statements

Label이 있는 statement에 對한 code 生成은 다음과 같다.

(1) AAC의 값을 label table에 저장한다.

(2) 현재의 label이 DO loop의 range limit로 사용되었는지 조사한다.

Range limit로 사용되지 않았으며 unlabelled arithmetic statement와 꼭 같이 code를 生成한다. Range limit로 사용되는 경우는 DO statement에 對한 code generation에서 다루고 있다.

b. IF Statement

IF statement에 對한 code를 生成하기 위해서 IF statement속에 나오는 label과 variable의 table entry address를 사용한다. "IF(K) 20, 30, 40"의 statement에서 label 20, 30, 40에 對한 label table entry address가 801, 803, 805이고 K의 symbol table entry address가 747이고 AAC의 값이 25라 가정하면 위 statement에 대응하는 machine code는 다음과 같다.

(25) 407801997003
(26) 423027997003
(27) 026747000028
(28) 101000803997
(29) 423030997003
(30) 025747031000
(31) 101000805997
(32) 423033997000
(33) 028000000000

왼쪽 괄호속의 숫자는 오른쪽 machine code의 주소를 나타낸다. 27번지는 compare for sign instruction으로 026은 명령 code이고 747은 K의 address이다.

747번지의 값(즉 K의 값)이 음수이면 801 번지의 내용을 주소로 하는 곳으로 jump하고 747의 값이 음이 아니면 28번지로 jump한다. 30번지의 025는 compare for zero instruction의 명령 code이고 747번지의 값이 0이면 803번지의 내용을 주소로 하는 곳으로 jump하고 0이 아니면 31번지로 jump한다.

33번지는 jump instruction으로 805번지를 내용하는 곳으로 jump한다.

c. GOTO statement

"GOTO 40"과 같은 statement의 machine code는 다음과 같다. 이때 AAC의 값은 31이고 40의 label table entry address는 805이다.

(31) 101000805997

(32) 423033997000

(33) 028000000000

d. DO statement

DO statement에 대한 code를 生成하기 위해서 아래 그림과 같은 5개의 column으로된 하나의 stack을 사용한다.

label	address	variable	do list element2	do list element3

DO statement가 나타나면 그것의 parameter들이 이 stack에 저장된다. 위 stack에서 address column은 DO statement 바로밑에 있는 statement에 대한 machine code의 address를 말한다. Label의 column에는 DO statement에 나오는 label이 저장되어 있으므로 label이 붙은 statement가 나올 때마다 이 label column과 비교해서 DO statement의 range limit로 사용되었는지 조사해야한다.

Range limit로 사용되었 을때는 하나의 block이 되도록 code를 生成해야 한다. DO statement와 machine code는 다음과 같다.

```
**  Given statement  **
        :
    DO  20  I=K, L, M
        :
        :
20  CONT
        :
        :
**  Machine code  **
(25)        101000745747
                :
                :
(40)        100747741747
(41)        120743747997
(42)        026997043026
                :
                :
```

여기서 747, 745, 743, 741은 I, K, L, M의 symbol table entry address이고 AAC

의 처음값은 25이다.

e. Read statement

Read statement에는 paper tape reader의 unit를 決定하는 parameter 1과 data의 format를 決定하는 parameter 2가 있으므로 code를 生成하기 위해서는 이들 parameter를 조사해야 한다. parameter와 대응하는 machine code는 다음과 같다.

	parameter 2	
parameter 1	1	2
1	270	370
2	231	331

"READ (1, 1) A, B, C, D"의 machine code는 다음과 같다.

270747747000
270745745000
270743743000
270741741000

f. PRINT statement

print statement에는 data의 format를 決定하는 parameter가 있으므로 code를 構成하기 위해서는 이 parameter를 조사해야 한다. parameter와 대응하는 machine code는 다음과 같다.

parameter	machine code
1	251
2	351

"PRINT (1) A, B, C, M"의 machine code는 다음과 같다.

251747747000
251745745000
251743743000
251737737000

g. END statement

이 statement는 source statement의 끝을 나타내므로 machine code "HALT"에 해당하는 "000000000000"가 生成되며 이것은 object program의 끝을 나타낸다.

이 statement에 의하여 compiling이 끝이난다.

FORTRAN program과 object program은 다음과 같다.

```
FORTRAN statements
                    ↓
R F A D ( 1
, 1 ) A , B
, C , D .
≥
                      ↓
D O 2 0
K = A , B ,
C .
≥
                        ↓
M = A + B ✻
C .
≥
                        ↓
P R I N T (
1 ) A , B ,
C , M .
≥
                          ↓
                    2 0
C O N T .
≥
                          ↓
E N D .
≥
```

address	codes
20……	270747747000
21……	270745745000
22……	270747747000
23……	270741741000
24……	101000747739

25……	140745743997
26……	100747997997
27……	101000997737
28……	251747747000
29……	251745745000
30……	251743743000
31……	251737737000
32……	100739743739
33……	120745739997
34……	026997035025
35……	000000000000

3. 結 論.

미니 컴퓨터는 memory가 작기 때문에 memory allocation이 매우 중요하다. code를 生成하는 동안 disk에 해당하는 외부 memory를 사용하여 memory를 allocate하였고 生成된 code에 address를 주기 위하여 하나의 counter를 사용하였다.

code 生成은 compiler개발에 필수적 요소이므로 이방면의 硏究가 계속될것을 기대한다.

참 고 문 헌

[1] J. J. Donovan, "*Systems programming*", New York, McGraw Hill, 1972.

[2] D. Gries, "*Compiler construction for digital computers*", New York, Willey international edition, 1971.

[3] IBM, "*FORTRAN compiler*", IBM 1130, Systems reference library, Disk monitor system version 2.

[4] IBM, "IBM 1130/1800 *basic FORTRAN* IV *language*", IBM systems reference library.

[5] O. S. Madrigal, "*The structure of a generalized programming language compiler*"

[6] MELCOM -83 manual.

[7] A. C. Shaw, "*Lecture notes on a course in systems programming*", Stanford technical report No. 52, 1966.

[8] University of Utah, "*Algol* 60, *revised report on the algorithmic language*", Computer science department.

서울대학교