Design and Development of Compiler for C- Language

3. Design and Implementation of Semantic Analyzer

과목명 : [CSE4120] 기초 컴파일러 구성

담당교수 : 서강대학교 컴퓨터공학과 정성원

개발자 : 17 조 엄태경 (20120085)

개발기간 : 2019. 5. 9. ~ 2019. 5. 28.

현 개발 단계 결과 보고서

프로젝트 제목: Design and Development of Compiler for C-Language:
Phase 2: Design and Implementation of LALR Parser
제출일: 2019. 5. 28.
개발자: 17 조 엄태경(20120085)

I. 개발 목표

본 프로젝트는 C- 프로그래밍 언어를 처리할 수 있는 컴파일러 구현의 세번째 단계로, 의미 분석 기능을 구현한다. 이 단계에서 코드 생성을 위한 심볼 테이블이 구성되고, 형 검사와 범위 검사가 이루어진다.

II. 개발 범위 및 내용목표

가. 개발 범위

본 프로젝트에서는 [1]의 Appendix B 에 제시된 TINY 언어의 컴파일러 소스코드 중 의미 분석에 필요한 부분만을 발췌하여 C- 언어에 맞게 참고 및 수정한다. 해당하는 파일은 C 언어로 작성된 "main.c", "analyze.c/h", "symtab.c/h" 이다. 여기에 더해 소스코드를 컴파일하고 실행파일을 생성하는 명령어를 처리하는 "Makefile"도 관련 소스코드의 수정에 따라 고쳐써야 한다.

나. 개발 내용

이전 프로젝트에서 구현한 문법 분석기가 생성하는 추상문법트리를 이용한다.

1. 추상문법트리를 순회하여 심볼 테이블을 구성한다. 이 과정에서 범위 검사를 수행한다.
2. 추상문법트리를 다시 순회하여 형 검사를 수행하고 그밖의 의미 오류를 검사한다.

검사해야 할 오류는 다음과 같다.

[오류 1] 선언되지 않은 변수나 함수의 참조
[오류 2] 같은 범위에서 변수나 함수, 함수 파라미터의 이름 중복
[오류 3] void 형 변수 및 파라미터
[오류 4] 대입문에서 형 불일치
[오류 5] int 형이 아닌 배열 인덱스
[오류 6] 배열이 아닌 심볼에 배열 참조 연산자 [,] 이용
[오류 7] 함수가 아닌 심볼에 함수 호출 연산자 (,) 이용
[오류 8] 함수 선언 시 반환형과 실제 반환문의 형 불일치
[오류 9] void 형 함수 내의 반환문
[오류 10] main 함수 이후의 전역 변수 및 함수 선언
[오류 11] void 형이 아닌 main 함수 선언
[오류 12] main 함수에 파라미터 선언
[오류 13] int 형이 아닌 if/while 문 조건식
[오류 14] 함수 호출 시 파라미터 형 및 개수 불일치

III. 추진 일정 및 개발 방법

가. 추진 일정

5 월 9 일: 과제 명세서 수령 및 요구사항 분석
5 월 10-25 일: C- 프로그래밍 언어 의미분석기 작성
5 월 26 일: 테스트용 C- 프로그램 작성 및 의미분석기 테스트, 디버깅
5 월 27-28 일: 보고서 작성

나. 개발 방법

1. 개발 환경

로컬 환경에서 개발과 테스트를 마친 뒤 학교에서 제공되는 cspro 서버에 파일을 업로드해 정상 작동을 다시 확인한다. 소프트웨어 버전에 다소 차이가 있으나, 각 프로그램의 업데이트 로그 등을 통해 버전에 따른 기능 차이가 본 프로젝트 수행에 영향을 주지 않음을 확인하고 진행한다.

로컬 환경: Ubuntu 18.0.4, gcc 7.4.0, flex 2.6.4, bison 3.0.4
서버 환경: Ubuntu 16.0.4, gcc 5.4.0, flex 2.6.0, bison 3.0.4

2. 개발 절차

(1) [1]에 주어진 TINY 컴파일러의 소스코드와 Yacc 파일을 분석하여 작동 구조를 이해하고, 의미분석기 작동에 필요한 부분과 TINY 프로그래밍 언어에 특화된 부분을 각각 파악한다.
(2) 과제 명세에 따라 필요한 심볼테이블의 구조를 고안하고 구현한다.
(3) 과제 명세에 따라 체크해야 할 오류를 구현한다.
(4) C- 프로그래밍 언어 문법에 따라 테스트 프로그램을 작성하고 완성된 의미분석기에 입력하여 결과를 확인한다. 심볼테이블이 올바르게 구성되어 적절하게 출력되는 것을 확인하고, 체크해야 할 오류를 올바르게 걸러내는 것을 확인한다.

IV. 연구 결과

*** 문법 수정**

본 프로젝트 명세 상, void 형 함수에서는 return 문을 쓸 수 없다는 제약이 있다. 이는 의미 분석보다는 문법 분석 단계에서 처리하는 것이 옳다고 판단하여 cm.y 파일에서 "return;"을 허용하던 문법을 삭제했다. ([오류 9])
심볼테이블에 파라미터가 역순으로 삽입될 수 있도록, cm.y 파일에서 "param_list"와 "args_list"에 해당하는 부분에서 파라미터 및 호출인자가 오른쪽에서 왼쪽으로 연결되도록 수정했다.

*** 심볼테이블 구성**

심볼테이블은 각 스코프마다 하나씩 구성하는 것을 원칙으로 하고, 스코프는 함수 선언문, 또는 함수의 첫 줄이 아닌 복합문에서 새로 만들어진다. 심볼테이블은 양방향 연결리스트로 구현하여 스코프 증감이 용이하도록 하고, 전역 스코프에 해당하는 심볼 테이블을 가리키는 포인터 변수를 따로 관리한다.
연결리스트의 각 심볼테이블 노드에는 스코프의 깊이, 해당 스코프의 베이스 메모리 주소, 그리고 해시테이블이 포함된다. 해시테이블은 해시 버킷 연결리스트의 배열로 구현되며, 각 해시 버킷 노드에는 심볼 이름, 선언된 줄 번호, 등장하는 줄 번호 연결리스트, 메모리 주소 오프셋, 심볼 종류(변수, 파라미터, 함수), 배열 여부, 배열일 경우 크기, 그리고 변수형이 포함된다.
추상문법트리 노드에 심볼리스트를 가리키는 포인터 변수를 추가하여, 함수선언문 및 함수의 첫 줄이 아닌 복합문에서 스코프와 그에 따른 심볼테이블이 새로 구성될 때마다 추상문법트리 노드에 해당 심볼테이블을 연결한다. 이렇게 하면 심볼테이블 구성을 마친 뒤에 추상문법트리를 순회하면서 각 심볼테이블에 다시 접근할 수 있다.
메모리 주소 오프셋을 처리할 때에는, 메모리 주소와 정수 자료형 등 관련된 모든 정보의 크기가 1 word = 4 byte 라고 가정했다.
선언문에서 심볼테이블에 심볼을 추가할 때 같은 스코프에 같은 이름이 있으면 에러가 발생한다. ([오류 2])
심볼 참조 시에는 심볼테이블에서 이름을 찾을 수 없거나 ([오류 1]), 이름을 찾았지만 변수/배열/함수 등의 사용이 잘못되면 에러가 발생한다. 심볼이 함수가 아닌데 (,)를 사용하는 경우나 배열이 아닌데 [,]를 사용하는 경우, 그리고 변수가 아닌데 (,),[,] 없이 사용하는 경우가 해당된다. ([오류 6], [오류 7])

*** 형 검사 및 의미 오류 처리**

그밖의 오류는 심볼테이블이 모두 구성된 후 다시 심볼테이블을 순회하면서 처리한다.
변수 선언, 배열 선언, 변수 파라미터 선언, 배열 파라미터 선언 시 타입이 void 면 오류이다. ([오류 3])
대입문에서 등호 좌우의 타입이 일치하지 않으면 오류이다. ([오류 4])
배열 인덱스가 정수가 아니면 오류이다. ([오류 5])
함수 선언문이 나올 때마다 해당 함수의 타입을 전역변수로 저장한다. 반환문이 나올 때 이 타입이 정수형이 아니면 오류이다. ([오류 8])
main 함수가 선언될 때 플래그를 체크해두었다가, 다음 선언문이 발생할 때마다 스코프가 전역인지를 검사한다. main 함수가 이미 선언된 상황에서 전역 스코프에 무언가 새로 선언되면 오류이다. ([오류 10])

main 함수 선언문에서 함수 타입을 체크해서 void 가 아니면 오류이다. ([오류 11]) 함수 파라미터가 void 가 아니면 오류다. ([오류 12])
if 및 while 문의 조건식이 정수형이 아니면 오류이다. ([오류 13])
함수 호출 시 인자의 수 및 형이 함수 선언부의 파라미터 수 및 형과 일치하지 않으면 오류이다. ([오류 14])

*** 테스트**

다양한 토큰을 이용하는 테스트 프로그램을 작성하여 의미분석이 정상적으로 수행됨을 확인했다. 테스트 프로그램인 test.tny 는 화면에 구구단을 2 단부터 9 단까지 출력하는 프로그램이다. (단, 숫자를 입력받아 그대로 출력하는 output(int), 공백을 출력하는 space(), 줄바꿈문자를 출력하는 newline() 함수의 본문은 구현하지 않았다.) main 함수 내에 while 문이 최대 3 중으로 겹친 스코프를 구성했다. 각 스코프마다 변수를 선언하여 심볼테이블이 정상적으로 구성됨을 확인하였다.

5. 평가

본 프로젝트는 [1]에 명세된 C- 프로그래밍 언어의 컴파일러를 제작하는 세번째 단계로 의미분석기를 구현하였다. 필요한 정보를 적절히 저장하는 심볼테이블을 구현하여 화면에 출력했으며, 그밖의 다양한 의미 오류를 검사하였다. 다양한 테스트 프로그램을 통해 심볼테이블 구성 및 의미분석을 성공적으로 수행함을 확인하였다.

V. 참고 문헌

[1] Kenneth C. Louden. 1997. *Compiler Construction: Principles and Practice*. PWS Publishing Co. Boston, MA, USA. pp. 491-544.