Java Applet을 이용한 C 프로그램 함수 실행의 시각화

오세광, 유광호, 하상호
순천향대학교 정보기술공학부
osk97@sch.ac.kr, new78eca@sch.ac.kr, hsh@sch.ac.kr

Visualization of the function execution of C program using Java Applets

Sekwang Oh, Kwangho Yu, Sangho Ha
Division of Information Technology Engineering,
Soonchunhyang University

요 약

인터넷의 발달로, 사이버 환경을 위한 C 프로그래밍 교육 컨텐츠가 많이 개발되고 있으나, 적절한 실습 환경까지 제공되지 않고 있다. 그러나 프로그래밍 언어의 교육은 실습이 수반되지 않으면, 그효과가 상당히 떨어진다. 본 논문에서는 인터넷상에서 효과적으로 사용할수 있는 C프로그래밍의 실습 환경을 구축한다. 특히, 학생들이 이해하기 어려운 함수 프로그램의 실행과정을 Java Applet을 사용하여 시각화 보여준다. 함수 프로그램 실행 시각화기가 설계되고, Java 환경에서 구현된다.

1. 서론

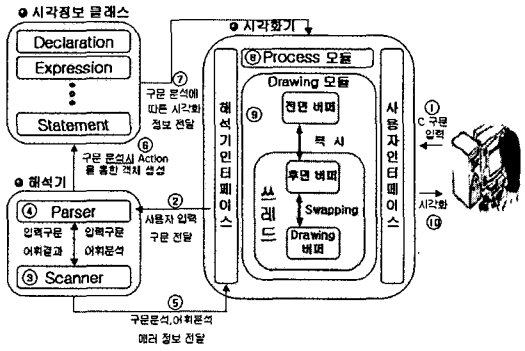
C언어는 컴퓨터를 교육하는 대부분의 대학에서 First Language로써 사용되고 최근 인터넷과 웹의 대중화로 많은 C언어 관련 교육용 웹 컨텐츠를 제공하고 있다. 이러한 교육용 컨텐츠는 이론과 예제 중심이며 실습 환경까지 고려되지 않고 있다. 특히 프로그래밍 언어 과목은 실습이 매우 중요하게 강조되고 있다. 따라서 학생들이 학습한 내용을 바로 그 자리에서 효과적으로 실습까지 이루어 질 수 있는 컨텐츠의 개발은 매우 절실하다.

본 논문에서는 실습환경을 갖춘 C프로그래밍 컨텐츠를 고려한다. 특히, 학생들이 이해하기 어려운 함수 실행에 대해서 다룬다. C언어 해석기를 개발하여 학생들이 함수를 학습하며 실행시켜볼 수 있는 실습환경을 구축하고, C 함수의 실행과정을 알기 쉽게 시각화하여 보여줄 수 있는 C 함수 실행 시각화기를 개발한다. C 언어 해석기는 JFlex[1]와 Cup[2]을 사용하여 Java[3]에 기반하여 개발하며, C 함수 실행 시각화기는 Java Applet을 사용하여 개발한다. 또한, C언어 함수 실행을 효과적으로 시각화할 수 있는 방법을 고려한다.

마지막으로 개발된 시스템에 대하여 C 함수 실행 예를 보인다.

2. 시스템 구조

그림1은 C 언어 해석기의 C 함수 실행 시각화기로 표현한 시스템의 전체 구조를 보여준다. 시스템의 해석기부에서는 입력된 문장을 구문분석기와 어휘분석기로 구문을 조사하여 시각화 객체를 생성하며 시각정보 클래스에서는 시각화 객체를 해석기에서 전달받아 객체 연산 및 시각화 정보를 만든다. 시각화기에서는 시각정보 클래스에서 시각화 정보를 얻어와 Process 모듈과 Drawing 모듈의 과정을 거치며 시각화기에 시각화 정보를 나타낸다.



[그림 1] 시스템의 전체구조

사용자가 시스템의 실행화면에서 C함수 입력 시 시스템이 입력된 함수를 해석하여 시각화하여 보여주는 과정을 단계별로 기술하면 아래와 같다.

① 학습자는 C프로그램을 시각화기를 통해 입력한다.
② C프로그램은 구문분석을 위해 해석기로 전달된다.
③ Scanner의 Lexer는 최소의미를 가진 토큰으로 나눈다.
④ 분리된 토큰은 Parser에 의해서 구문분석 과정을 거치게 된다.
⑤ 구문 분석 시 에러가 발생하면 이를 보고한다.
⑥ 구문 분석 시 Parser는 Action을 통하여 전체 함수 구문에 해당하는 클래스를 생성하게 되는데 이 클래스는 함수구문에 대한 시각화 정보를 담고 있다.
⑦ 시각정보를 담은 클래스는 시각화기로 전달되어

Process 모듈로 입력되어 가공 처리된다.

⑧ Process 모듈은 시각정보를 담은 클래스를 통하여 Drawing 모듈에서 처리 가능한 형태로 정보를 추출 가공하여 Drawing 모듈로 전달한다.

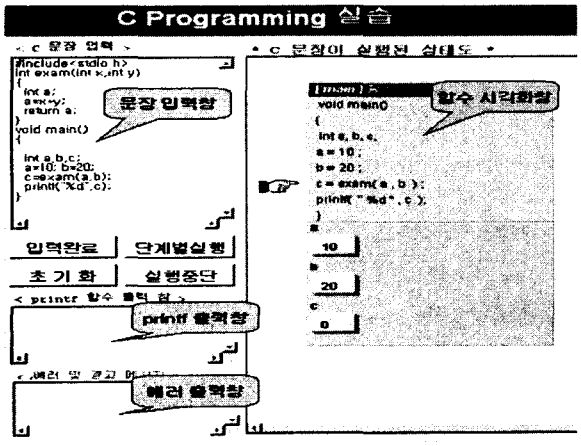
⑨ Process 모듈을 거친 정보를 Thread와 Drawing 모듈의 Triple버퍼링을 통해서 이를 학습자의 실습창에 보여주게 된다.

3. C 언어 함수 실행의 시각화

여기서는 함수선언/정의, 함수호출, 매개변수의 전달 등 C언어 함수의 실행에 대한 시각화 방법에 대하여 논의한다.

3.1. 함수 실행 시각화

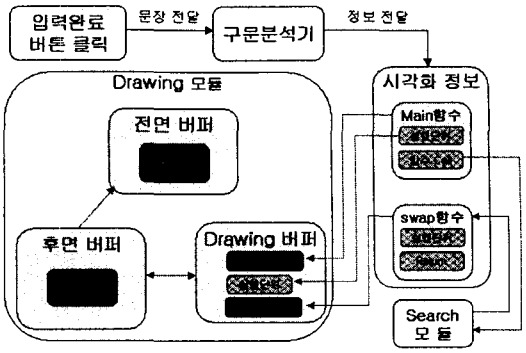
학습자는 그림2의 문장 입력창에 전체구문을 입력하고 입력완료 버튼을 클릭하면 그림 3의 과정이 실행된다.



[그림 2] 함수 실행 시각화 창

- 문장입력창 : 학습자가 전체 구문을 입력하는 창
- 함수시각화창 : 구문에 대한 시각화를 보여주는 창
- printf 출력창 : printf함수 결과를 보여주는 창
- 에러 출력창 : 에러 발생시 메시지 출력 창
- 입력완료버튼 : 입력된 구문을 분석 실행하는 버튼
- 단계별 실행버튼 : 구문단위로 시각화하는 버튼
- 초기화 버튼 : 처음상태로 Applet을 초기화하는 버튼
- 실행 중단 : 실행을 잠시 중단하는 버튼

그림3에서 입력문장이 구문분석기에 전달되고 구문분석 후 시각화정보에 함수단위의 정보를 저장한다. Main 함수는 항상 처음 실행되므로 Drawing모듈이 Main함수의 정보를 가져와서 DrawThread가 Drawing버퍼에 Main함수를 시각화하고 후면버퍼와 스왑이 이루어진다. 그리고 메인Thread가 후면버퍼를 전면버퍼에 복사하는 Triple Buffering기법으로 시각화기에 나타난다. 단계별 실행 버튼을 클릭하면 Drawing모듈이 Main함수의 실행단위의 시각화 정보를 가져와서 Triple Buffering기법으로 시각화한다.



[그림 3] 실행 시각화 과정

3.2. 함수호출

실행단위에서 함수호출이 발생하면 그림3에서 Search 모듈이 시각화 정보에서 호출된 함수를 찾아서 Drawing 모듈에 전달한다. Drawing모듈은 Swap함수를 위의 과정으로 시각화한다.

3.3. 함수의 리턴 (Return)

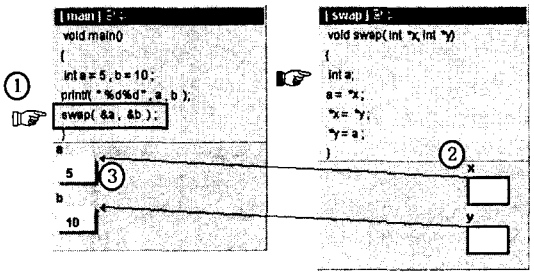
함수 내의 실행단위에서 리턴이 되면 그림3의 Drawing 버퍼에서 함수가 사라지며 리턴 값은 함수를 호출했던 곳의 배정변수로 이동되며 배정변수가 리턴 값으로 갱신되는 시각화가 나타난다.

3.4. 재귀함수 (Recursion)

재귀 호출이 되면 그림3에서 Search모듈이 호출된 함수의 정보를 찾아서 Drawing모듈에 전달하고 종료조건을 만날 때까지 계속 시각화한다. 시각화할 때마다 메모리의 소비와 화면이 협소한 관계로 재귀함수의 시각화는 제한사항이 존재한다.

3.5. 함수의 포인터 매개변수(Call by Reference)

그림4는 포인터를 함수와 관계하여 보다 쉽게 시각화하여 보여준다. ①에서 함수호출이 일어났을 때 swap함수에 정의된 변수와 함수호출시 대입된 변수와 비교후 일치하면 위의 그림처럼 인자로 a, b의 주소값이 전달될 때 ②처럼 함수내의 포인터 변수가 인자를 가르키는 모습을 시각화하여 나타낸다. ③에서 인수의 값을 변경할 경우 포인터 변수가 가리키는 변수의 값이 변화되는 모습이 시각화된다.



[그림 4] 함수의 포인터 매개변수

4. 구현

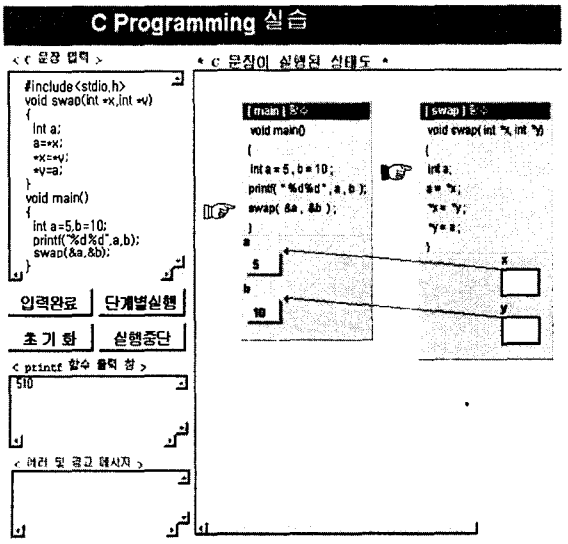
구문을 분석하는 CUP[2]은 C언어의 전반적인 구문을 EBNF구문으로 정의하였으며 어휘분석을 위해 JFlex[1]을 이용하여 C언어에서 사용되는 어휘를 정의하였다.

시각화 클래스는 구문 분석을 통해 얻어진 정보들을 저장 및 객체들의 연산을 거치면서 시각화 정보를 포함한 객체로 만들어 그 객체들을 시각화기에 전달한다.

시각화기의 Drawing모듈이 객체정보를 분석 후 Triple Buffering 기법을 통해서 실습 창에 보여주게 된다.

5. 함수 실행의 예

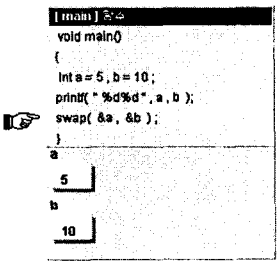
위에서 논의한 함수 시각화의 구현내용에 대한 실제 실행 예로써 Call by Reference의 방식으로 Main함수의 변수 a, b값을 서로 바꾸는 과정을 살펴보자.



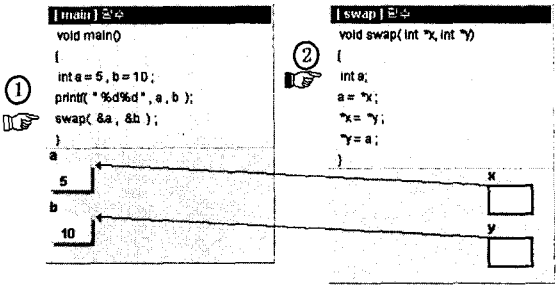
[그림 5] C언어 함수 실행 시각화 예

그림5에서 함수호출이 일어나기 전에 변수 a, b의 값은 그림6과 같다. 단계별 실행을 클릭하면 변수 a의 주소값과 b의 주소값이 swap함수의 인수로 전달되면서 그림7처럼 보여진다. ①이미지가 함수 호출한 곳의 위치를 가리키고 ②이미지가 현재 제어의 위치를 가리킨다. swap함수의 **a=*x;** 문장을 보면 변수 a의 값이 포인터 변수 x가 가리키는 변수(Main의 a)의 값으로 배정이 되는 것을 알 수 있다.

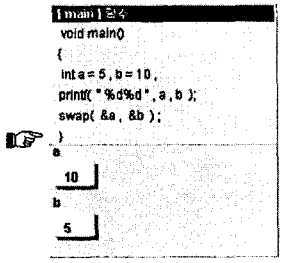
단계별 실행을 클릭하여 swap함수의 ***x=*y;** 문장을 실행하면 Main함수의 b의 주소를 가리키고 있는 포인터 변수 y가 Main함수의 a의 주소를 가리키고 있는 포인터 변수 x로 배정됨에 따라 Main함수의 변수 a의 값이 b의 값으로 바뀌는 것을 볼 수 있다. 단계별 실행을 클릭하여 ***y=a;** 문장을 실행하면 Main함수의 b의 주소를 가리키고 있는 포인터변수 y가 swap함수의 변수 a로 배정됨에 따라 Main함수의 변수 b의 값이 바뀌는 것을 볼수 있다. swap함수가 종료되면 그림8에서 보는 바와 같이 변수 a,b의 값이 서로 바뀐 것을 확인 할 수 있다. 이렇게 call by reference에 의한 함수 호출 과정을 상세히 알아 볼 수 있다.



[그림 6] swap함수 호출 전



[그림 7] swap함수 호출 후



[그림 8] swap함수종료 후

5.결론

본 논문이 개발한 실행환경이 탑재된 컨텐츠에서는 학습자가 학습한 내용을 즉시 그 자리에서 효과적으로 시각화 할 수 있다. 또한, C함수 실행의 시각화를 통해서 함수에 대해서 쉽게 이해를 도와줄 수 있다. 현재 개발된 C해석기는 모든 C구문을 지원하지는 않는다. 앞으로 더 많은 구문을 지원할 수 있도록 C해석기를 보완하고 다른 구문의 C 문장 실행에 대한 시각화를 추가할 계획이다.

참고문헌

[1] http://www.jflex.de/
[2] http://www.cs.princeton.edu/~appel/modern/java/CUP/
[3] 이현우 & 천영환, "Java Programming Bible for JDK 1.3", (주) 영진닷컴, 2001
[4] Brian W.Kernighan,Dennis M.Ritchie, "The C pro gramming Language", 2rd Ed, Prentic-Hall, 1983
[5] Robert W.Sebesta, "Concepts of Programming La nguages", Addison-Wesley, 1999