자동화된 프로그래밍 과제 평가 시스템의 설계 및 구현

Design and Implementation of an Automatic Grading System for Programming Assignments

김 미 혜*
Kim, Mihye

요 약

프로그래밍 교육에 있어 학습자의 학업 성취도를 향상시킬 수 있는 중요한 요인 중의 하나는 다양한 형태의 과제를 학습자에게 부여하여 문제 해결 연습 기회를 많이 제공해 주는 것이다. 그러나 프로그래밍 과제 평가는 대부분 수작업으로 행해지고 있으면 정확한 평가 방법을 제공해 줄 수 있는 자동화된 도구 또한 결여되어 있는 게 현실이다. 이러한 제한된 환경 하에서 과제 평가는 교수자들에게 많은 시간과 노력을 요구하게 되어 다양한 형태의 과제 부여는 현실적으로 어려움이 있다. 이러한 문제를 극복하기 위해서는 교수자가 효율적이고 일괄적인 방법으로 과제를 쉽게 평가할 수 있고, 학습자들 상호간의 프로그램 소스코드의 표절 또한 용이하게 검사할 수 있는 자동화된 프로그래밍 평가 시스템이 필요하다. 따라서 본 논문에서는 교수자가 프로그램의 성능을 자동적인 방법으로 평가할 수 있을 뿐만 아니라 적절한 피드백과 함께 프로그램의 스타일과 표절에 대한 검사 또한 용이하게 수행할 수 있는 웹을 기반으로 한 프로그래밍 과제 평가 시스템을 설계하고 구현한다.

Abstract

One of important factors for improving the learning achievement of students in computer programming education is to provide plenty of opportunities of problem-solving experiences through variety forms of assignments. However, for the most cases, evaluation of programming assignments is performed manually by instructors and automated tools for the accurate evaluation are not equipped at the present time. Under this restricted environment, instructors need much work and time to grade assignments so that instructors could not deliver sufficient programming assignments to students. In order to overcome this problem, au automated programming assignment evaluation system is needed that would enable instructors to evaluate assignments easily in an effective and consistent way and also to detect any plagiarism activities among students in program source codes readily. Accordingly, in this paper we design and implement a Web-based programming assignment grading system that allows instructors to evaluate program performance automatically as well as to evaluate program styles and plagiarism easily with appropriate feedback.

☞ Keyword: Computer Education, Programming Assignment Evaluation System, Programming Assignment Grading System

1. 서 론

컴퓨터 프로그래밍은 컴퓨터 교육의 가장 기본이 되는 교과라 해도 과언이 아니다. 대부분의 컴퓨터 교과목에 있어 프로그래밍 실습 없이 컴퓨터의 개념을 이해하고 습득하기란 결코 쉬운 일이 아니기 때문이다. 또한 컴퓨터 교육은 컴퓨터 활용 능력뿐만 아니라, 궁극적으로는 보다 효과적이고 효율적인 방법으로 응용 시스템 개발 능력을 배양하는데 그 목적이 있기 때문이다. 따라서 컴퓨터 교육을 효과적으로 실시하고 학생들의 학습 효과를 높이기 위해서는 프로그래밍 교과목에서 뿐만 아니라 컴퓨터 관련 교과목에서도 다양한 형태의 프로그래밍 과제를 통한 문제 해결 연습 기회를 많이 제공해 주어야 한다.

그러나 프로그래밍 과제 평가는 대부분 수작업

* 종신회원 : 대구가톨릭대학교 컴퓨터교육과 교수
mihyekim@cu.ac.kr
[2007/07/19 투고 - 2007/07/30 심사 - 2007/08/13 심사완료]

으로 행해지고 있으며 이러한 제한된 여건과 환경 하에서 다양한 형태의 과제 부여는 교수자들에게 많은 시간과 노력을 요구할 수밖에 없고 주어진 시간 안에 공정하게 과제를 평가하는 것 또한 매우 어려워, 교수자로 하여금 간단한 몇 개의 과제만을 학생들에게 부여하거나 아예 프로그래밍 과제 부여를 기피하는 현상을 초래하게 한다 [1][2]. 몇몇 자동화된 프로그래밍 과제 평가 시스템에 관한 연구가 수행되기는 하였으나 [1][2][3], 아직도 편리하고 유연한 평가 방법을 제공해 줄 수 있는 자동화된 도구가 미흡하다. 수작업으로 프로그래밍 과제 평가가 이루어지는 제한된 환경 하에서는 컴퓨터 교과목의 학습 효과를 극대화하기란 매우 어려운 일이다. 따라서 이러한 어려움을 극복할 수 있는 교수자가 자동적이고 일괄적인 방법으로 적절한 피드백과 함께 학습자들의 과제를 쉽게 평가할 수 있고 학습자들 상호간의 프로그램 소스코드의 표절 또한 용이하게 검사할 수 있는 프로그래밍 과제 평가 시스템에 대한 연구가 필요하다. 본 논문에서는 기존 프로그래밍 과제 평가의 문제점을 극복하고 프로그램 성능을 자동적인 방법으로 평가할 수 있을 뿐만 아니라 교수자와 학습자간의 상호 작용을 극대화시키며 프로그래밍 과제를 용이하고 효율적인 방법으로 평가할 수 있는 프로그래밍 과제 평가 시스템인 PAGS (Programming Assignment Grading System)를 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 연구들에 대해 살펴보고 3장에서는 프로그래밍 과제평가 절차 및 이에 대한 구성 요소들에 대해 설펴본다. 4장에서는 제안한 프로그래밍 과제 평가 시스템의 설계 및 구현에 대해 기술하고 5장에서는 구현한 시스템의 유용성 실험 결과에 대해 분석한다. 마지막으로 6장에서는 결론과 함께 향후 연구 과제에 대해 논의한다.

2. 관련연구

프로그래밍 교과목의 학습 효과를 증진시키고 보다 효율적인 학습 환경 구축을 위한 일환으로 학생들이 제출한 과제를 자동적이고 일괄적인 방법으로 평가할 수 있는 시스템에 관한 연구가 몇몇 대학을 중심으로 진행되어 왔다.

영국 리버플 대학의 컴퓨터과학과에서는 반자동화된(semi-automated) 프로그래밍 과제 평가 시스템을 Unix shell script로 개발하여 학생들의 프로그래밍 과제를 평가하였다 [3]. 이 평가 시스템은 이-메일을 통해 제출된 프로그래밍 과제를 자동 컴파일하고 미리 작성해 놓은 검사 데이터를 이용하여 프로그램의 정확성을 검증하여 평가보고서를 생성한다. 이 평가 시스템은 Unix shell script로 구현되었으며 개발된 시스템은 유닉스 환경의 텍스트모드 상에서 각 단계별로 처리하도록 구현되어 있으며 문서 파일에 대한 질적 평가, 소스코드에 대한 분석 및 돌발적으로 발생할 수 있는 불확실한 경우의 처리를 위하여 사람의 개입이 필요한 반자동화된(semi-automated) 프로그래밍 평가 시스템이다.

싱가포르 국립 대학교에서는 컴퓨터공학과에서 사용하기 위해 개발된 프로그래밍 과제 평가 시스템인 Online Judge을 소개하고 있다 [1, 4]. 이 연구에서는 자동화된 프로그래밍 과제 평가 시스템의 필요성과 수작업으로 이루어지는 평가 시스템의 문제점을 지적하고 이러한 문제점들을 해결할 수 있는 자동 평가 시스템을 개발하여 실제 프로그래밍 관련 교과목에 적용한 사례를 소개하고 있다. 이 과제 시스템은 학생들이 제출한 프로그램 중 악성 프로그램, 바이러스 프로그램 등을 점검할 수 있는 컴퓨터 보안 처리까지 상세히 기술하고 있지만 개발된 시스템은 리눅스 환경의 텍스트모드 상에서 각 단계별로 처리하도록 구현되어 있어 평가 절차를 총괄적으로 통제할 수 있는 웹을 기반으로 한 그래픽 환경의 통합시스템으로의 구축이 필요하다고 본다.

핀란드의 헬신키 산업 대학에서는 기본 프로그래밍 과정의 기능 프로그래밍 언어인 Scheme로

구현된 프로그래밍 과제를 자동으로 평가하기 위한 Scheme-robo를 개발하였다 [5]. 이 평가 시스템은 프로그램의 구조, 실행시간 및 프로그램 표절 등을 검사할 수 있는 도구들을 지원하고 있다. 그러나 Scheme-robo은 프로그램 전체에 대한 평가 시스템이기 보다는 Scheme 언어의 개별적인 프로시저 모듈만을 평가하기 위한 것이며 Scheme로 구현된 프로그래밍 과제 평가만을 위한 제한적인 시스템이다.

[6]의 연구에서는 원격 교육환경 하에서 다양한 과제 유형을 지원하고 등록 학생수가 대규모인 과목의 과제를 처리할 수 있는 멀티 에이전트 플랫폼을 지원하는 자동화된 과제 관리(Automatic Assignment Management) 도구를 제안하고 있다. 이 연구에서는 동일한 평가 패러다임을 이용하여 다양한 과제 유형을 평가할 수 있는 확장 가능한 개방된 구조의 과제 관리에 대한 플랫폼을 소개하고 있다. 이 연구는 자동화된 프로그래밍 평가 방법에 초점을 두기보다는 다양한 과제 유형을 동일한 플랫폼을 이용하여 제출하고 처리할 수 있는 통합적인 과제 관리 시스템 개발에 역점을 두고 있다.

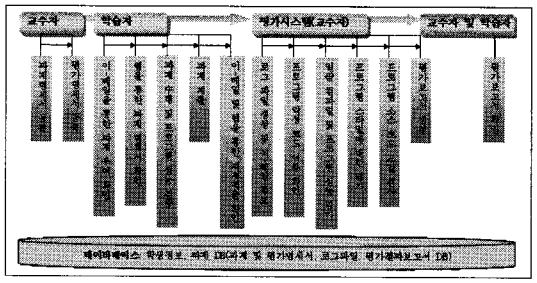
국내에서는 K대학교 컴퓨터교육과에서 웹을 기반으로 한 프로그래밍 교육 시스템(PES: Programming Education System)을 개발하여 "C 프로그래밍" 교과목에 적용한 사례를 소개하고 있다 [2]. 이 연구에서는 교수자와 학습자간의 향상된 상호작용을 통해 학습자가 시공간 제약 없이 프로그래밍을 학습할 수 있는 환경을 제공하며 프로그램 평가 및 피드백에 대한 효율적인 방법을 제시하였다. PES는 크게 학습자 관리 블록, 교수자 관리 블록, 평가 엔진 블록으로 구성되어 있으며 교수자 관리 블록과 평가 엔진 블록에서 프로그래밍 자동 평가 기능을 다루고 있다. 이 연구는 프로그래밍 자동 평가 시스템에 초점을 두기보다는 학생들의 프로그래밍 학습 효과를 증진시킬 수 있는 보다 광범위한 웹 기반 환경에서의 프로그래밍 교육 시스템 개발에 역점을 두고 있다.

본 논문에서는 과제 평가 방법에서는 연구 [1, 3]과 맥락을 같이하며 평가자가 일괄적인 방법으로 적절한 피드백과 함께 평가 절차를 총괄적으로 통제할 수 있는 웹을 기반으로 한 그래픽 환경의 통합 시스템 구축에 역점을 두고 있다. 특별히 평가자 측면에 초점을 맞추고 평가자가 과제를 쉽게 평가할 수 있고 학생들 상호간의 프로그램 소스코드의 표절 또한 용이하게 검사할 수 있는 프로그래밍 과제 평가 시스템(PAGS)을 설계하고 구현한다.

3. 프로그래밍 과제 평가의 구성요소

프로그래밍 과제 평가 절차를 자동화하기 위해서는 우선 먼저 교수자들은 명확하고 상세한 과제 명세서를 학생들에게 제공해 주어야 한다. 과제가 시스템에 등록되면 시스템은 이-메일을 통해 학생들에게 과제가 부여되었음을 알린다. 학생들은 온라인으로 과제를 제출할 수 있어야 하며 과제를 제출하기에 앞서 과제명세서에서 제공되는 검증 자료를 통해 자신의 프로그램의 성능을 시험해 볼 수 있어야 한다. 시스템은 제출한 과제 프로그램을 일정한 위치에 저장하고 학생들에게 과제 제출 성공 여부를 알린 후 과제 제출에 관련된 로그 파일을 생성하고 프로그램을 평가한다. 프로그램을 컴파일한 후 컴파일이 성공하면 일련의 검증자료를 사용하여 프로그램의 정확성 및 효율성에 대해 평가하고 프로그램 소스코드 표절 여부를 검사한 후 적절한 코멘트와 함께 모듈성, 명료성 등과 같은 프로그램 스타일에 대해 평가한다. 최종적으로 평가보고서를 생성하여 교수자와 학생들에게 알린다. 그림 1은 프로그래밍 과제 평가의 전체적인 구성 요소와 흐름도를 보여주고 있다. 과제 평가 과정에서 고려해야 할 주요 구성요소는 다음과 같다.

3.1 과제 명세서



(그림 1) 프로그래밍 과제 평가의 전체적인 구성 요소 및 흐름도

프로그래밍 과제 평가 절차를 자동화하기 위한 첫 번째 단계로서는 평가 결과의 공정성을 꾀하고 학생들이 평가 결과에 수긍할 수 있도록 반드시 교수자는 과제명세서를 명확히 기술하여 웹을 기반으로 한 과제를 학생들에게 제공해 주어야 한다. 과제명세서에 포함시켜야 할 항목은 다음과 같다.

- 과제 아이디와 과제명: 시스템이 식별할 수 있는 유일한 과제 아이디와 과제명을 제공해 주어야 한다.
- 요구사항: 프로그램 과제의 목적 및 구현해야 할 내용, 방법 및 주어진 가정들에 대해 상세히 기술한다. 특히 입력자료(input data) 및 출력자료(output data)가 필요한 경우 구체적인 입·출력 자료의 형식, 입·출력 방법 및 사용 예제를 명확하게 제시해 주어야 하며 프로그램에서 처리해야 할 오류 및 예외적인 상황에 대해서도 구체적인 처리 방법을 명시해 주어야 한다. 프로그램 성능 평가가 시스템에 의해 자동적인 방법으로 수행되기 때문에 명시한 오류 이외의 것은 다루지 않도록 반드시 주지시킨다. 평가시스템을 효율적으로 운용하기 위해서는 입력 자료 처리를 위한 모듈은 제공해 주는 것이 좋다.
- 총점: 과제의 총점을 제시해 준다.
- 평가항목, 평가기준 및 배점: 정확성, 효율성 및 프로그램 스타일 등과 같은 구체적인 평가 항목들과 이들 각각에 대한 평가기준 및 배점을 제시해 준다.
- 벌점: 마감 후 제출 및 표절 시 벌점을 명시해 준다.
- 과제 제출 마감일과 시간: 과제 제출 마감일과 시간을 초 단위까지 정확하게 제시해 준다.
- 제출해야 할 파일명: 제출해야 할 파일의 이름과 확장명을 포함한 형식을 명확히 명시하고 파일이름은 반드시 제시한 이름과 동일해야 함을 학생들에게 주지시킨다. 하나의 파일로 압축하여 제출하는 방법도 고려할 수 있다.
- 샘플 검증 자료: 견본 검증 자료(sample test data)와 이에 상응하는 출력 결과를 포함한 프로그램의 실행 예제를 제공해 준다.

3.2 과제 제출

학생들이 웹을 기반으로 하여 과제를 제출할 수 있도록 일정한 형식의 과제 제출 화면을 제공해 주어야 한다. 과제 제출 화면 구성 시 포함시켜야 할 항목은 다음과 같다.

- 학생번호 및 비밀번호: 학생번호와 비밀번호를 입력한 후 시스템에 로그인할 수 있도록 한다.
- 과목명 및 과제명: 수강신청관리시스템과 연계하여 학생의 등록된 교과목에 관한 정보를 화면에 표시해 주고 과제와 관련된 교과목을 선택하면 해당 교과목에 등록된 과제명을 표시하여 선택할 수 있도록 한다.
- 과제 제출 마감일과 시간: 과제 제출 마감일과 시간을 제공해 준다.
- 과제 명세서: 과제 명세서를 화면과 출력을 통해 확인할 수 있도록 한다.
- 과제 제출: 파일 전송(file upload) 기능을 제공하여 과제를 제출할 수 있도록 하며 과제 제출

마감일까지는 반복적으로 재 제출할 수 있도록 한다. 이 때 과제 제출 히스토리 정보에 관한 로그파일을 관리한다.

- 제출 결과 확인: 과제 제출이 정상적으로 이루어졌는지 확인할 수 있도록 한다. 과제를 제출한 날짜와 시간, 제출한 파일명 등을 표시해 준다.
- 점수 및 평가보고서: 학생들은 평가보고서를 화면과 출력을 통해 확인할 수 있어야 한다.

3.3 과제 평가명세서

제출한 과제 프로그램을 일정한 위치에 저장하고 과제 제출에 관한 로그 파일을 생성한 후 프로그램을 평가한다. 일괄처리 방법을 통해 프로그램을 컴파일하고 검증자료를 사용하여 프로그램의 정확성 및 효율성에 대해 평가한다. 성능에 대한 평가가 완료되면 프로그램 스타일 평가와 프로그램 소스코드 표절 여부를 평가한다. 일련의 평가 과정이 완료되면 시스템은 평가보고서를 생성하여 교수자와 학생들에게 알린다. 프로그래밍 과제 평가를 자동화하기 위해서는 어떠한 방법으로 과제를 평가할 것인지에 대한 구체적인 평가 방법과 기준을 시스템에 등록시켜야 한다. 평가명세서에 포함시켜야 할 항목은 다음과 같다.

- 과목명 및 과제명: 과제와 관련된 과목명과 과제명을 지정한다.
- 총점: 과제의 총점을 명시한다.
- 과제 제출 마감일과 시간: 과제 제출 마감일과 시간을 지정한다. 예) 23:59:59
- 벌점: 마감 후 제출 시 일당 벌점을 지정한다.
- 제출해야 할 파일명: 과제명세서에서 명시한 제출해야 할 파일명을 등록한다. 등록된 파일 이외의 것은 과제 제출 시 허용되지 않도록 한다.
- 컴파일러명: 과제 평가에 사용할 컴파일러명을 지정한다. 예) cc, javac
- 소스코드 파일명: 컴파일 시에 사용할 소스코드 파일명을 등록한다.
- 컴파일 옵션: 컴파일 시에 사용할 옵션이 있는 경우 해당 옵션을 등록한다.
- 실행 파일명: 실행 파일명을 지정하여 준다.
- 최대실행시간: 프로그램이 무한 루프 등에 빠질 경우를 대비하여 프로그램의 최대실행시간을 지정하고 지정한 시간이 경과하면 자동으로 프로그램의 실행을 중지한다.
- 검증자료(**test data sets**) 및 출력결과: 과제의 정확성을 검증할 수 있는 다양한 형태의 검증자료와 이에 상응하는 출력결과를 등록한다. 다양한 형태의 검증자료 설계는 프로그램 성능 평가의 아주 중요한 항목이므로 프로그램의 정확성, 논리적인 오류 및 예외적인 경우 등의 존재 가능한 모든 경우가 시험될 수 있도록 다양한 검증 자료를 설계해야 한다. 검증자료별로 점수를 지정하여 준다.
- 평가항목 및 배점: 프로그램 성능과 프로그램 스타일 (즉, 코멘트, 가독성, 명료성, 모듈성 등) 등에 대한 평가 항목과 이들 각각에 대한 배점을 지정한다. 각 항목에 대한 배점의 합은 과제의 총점과 같도록 한다. 프로그램 성능의 총점은 검증자료별 점수 합계이다.

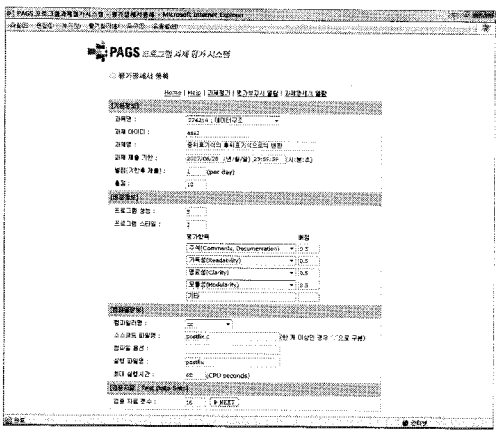
4. 과제 평가 시스템의 설계 및 구현

제안된 접근방법을 평가하기 위해 리눅스 환경 하에서 웹을 기반으로 한 프로토타입용 프로그램 과제 평가 시스템 PAGS(Programming Assignment Grading System)를 구현하였다. 본 논문에서는 지면 관계상 평가명세서의 등록과 등록된 평가명세서에 의해 과제 평가가 어떠한 절차에 따라 수행되는지에 대한 교수자 측면에서의 프로그래밍 과제 평가 시스템의 설계와 구현만을 다루도록 한다.

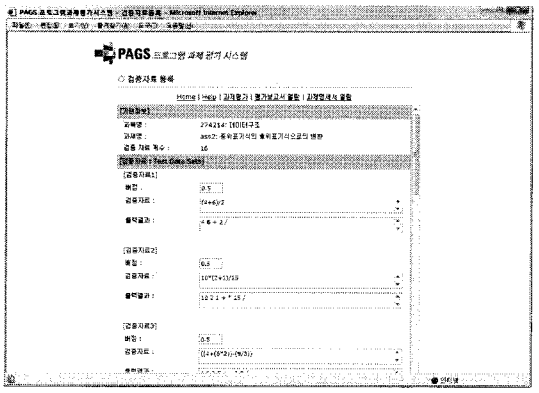
4.1 평가명세서 등록

앞 절 "3.3 과제 평가"에서 언급한 바와 같이 프로그래밍 과제 평가를 자동화하기 위해서는 교수자는 어떠한 방법으로 과제를 평가할 것인지에 대한 평가명세서를 시스템에 등록해 주어야 한다. 교수자가 시스템의 로긴(login) 과정을 거쳐 과제 평가 시스템 초기 화면에서 "평가명세서 등록" 항목을 선택하면 그림 2와 같은 과제 평가 등록 화면이 나타난다.

수강신청관리 시스템과 연계하여 교수자가 시스템에 로긴하면 교수자의 담당 교과목을 제공하여 교수자의 과제와 관련된 과목명을 선택할 수 있도록 구성되어 있다. 과목명을 선택한 후 해당 과제 아이디와 과제명을 입력한다. 이때 교수자는 "과제명세서 열람" 링크를 통해 과제명세서를 확인하여 볼 수 있다. 과제 제출 기한일과 시간, 마감 후 제출 시 일당 벌점과 과제의 총점을 지정한다. 그 다음 프로그램 성능과 스타일에 대한 배점을 등록한다. 간단한 과제인 경우에는 프로그램 스타일에 대한 평가는 제외시킬 수 도 있을 것이다. 이때는 프로그램 성능의 합이 총점과 동일하도록 지정해야 한다. 프로그램 스타일에 대한 배점을 지정한 경우에는 프로그램 평가 시 지정한 프로그램 스타일 평가 항목에 대한 평가를 해야 한다. 이에 대한 자세한 설명은 다음절에서 다룬다.



(그림 2) 프로그래밍 과제 평가명세서 등록 화면



(그림 3) 프로그래밍 과제 평가 검증 자료 등록 화면

그 다음에는 컴파일에 대한 정보를 등록한다. 과제 평가에 사용될 컴파일러명, 소스코드 파일명, 컴파일 옵션, 실행 파일명과 프로그램 최대실행시간을 등록하여 준다. 프로그램 최대실행시간은 CPU의 초에 해당하는 시간이다. 컴파일 정보의 등록 및 검증자료 개수의 입력을 완료하며 과제의 기능 및 정확성, 즉 프로그램 성능에 사용할 검증자료를 등록할 수 있는 그림 3과 같은 화면이 나타난다. 검증자료와 이에 상응하는 출력결과와 배점을 지정하여 준다. 검증자료와 출력결과 입력 시 자료간의 경계기호(delimiter)가 필요한 경우에는 반드시 사용할 구분문자(공백, 콤마, 탭 등)와 사용 개수를 명확히 제시하고 반드시 그와 동일하게 입력되도록 주지시킨다. 동일하지 않은 경우에는 평가가 자동으로 이루어짐으로 출력 결과 불일치로 인해 오류로 처리될 수도 있기 때문이다.

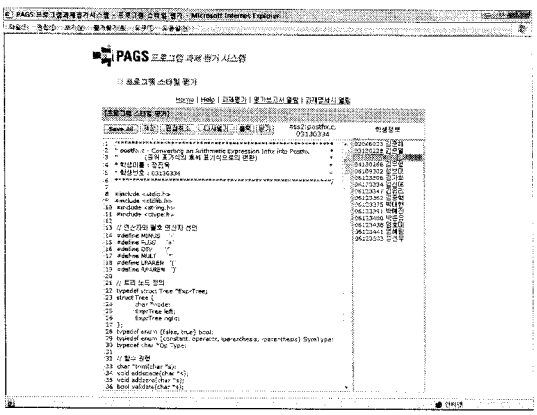
검증자료를 입력한 후 평가명세서의 내용을 저장하면 시스템은 평가명세서의 내용이 구문 오류 없이 입력되었는지 점검한 후 오류가 존재하지 않으면 자동 평가 수행에 필요한 환경을 구축한다. 과제명세서 등록이 완료되면 과목 홈 디렉터리, 작업 디렉터리, 제출 과제 저장 디렉터리 및 샘플 검증자료 입·출력 결과 파일 등이 생성된다. 평가명세서 등록이 완료되면 추가로 각각의 검증자료에 대한 데이터 파일과 이에 상응하는

출력 결과 파일 등이 생성된다.

4.2 과제 평가

과제 평가명세서의 등록이 완료되면 등록된 평가명세서에 의하여 과제 평가를 수행할 수 있다. 프로그램 과제 평가 시스템 PAGS 화면에서 "과제평가" 항목을 선택하면 과제 평가 화면이 나타난다. 과제 평가 화면은 프로그램 성능 평가, 프로그램 스타일 평가, 프로그램 소스코드 표절 검사 및 과제 평가 완료를 수행할 수 있는 평가 항목으로 구성되어 있다. 우선 먼저 "프로그램 성능 평가" 항목을 선택하여 프로그램 정확성에 대한 성능을 평가한다. 프로그램 성능 평가는 일괄 평가와 개별 평가 방법으로 수행할 수 있으며 일괄 평가는 과제 평가명세서에 의거하여 제출된 모든 과제를 일괄 컴파일하고 등록된 검증자료를 사용하여 한꺼번에 모든 과제의 평가가 실행되면 개별 평가는 원하는 학생의 과제만을 평가할 수 있도록 구성되어 있다. 프로그램 성능 평가 과정은 화면을 통해 출력되면 동시에 텍스트 파일로도 저장된다.

프로그램 성능 평가가 완료되면 성능 평가 결과를 바탕으로 프로그램 스타일 평가를 수행한다. 프로그램 스타일 평가는 그림 4와 같은 인터페이스를 통해 반자동화된 방법으로 수행되며 제공된 화면을 통해 소스코드를 검토할 수 있으면 소스코드 검토 결과에 대한 의견, 자동 성능 평가 결과에 대한 피드백 등을 소스코드에 추가할 수 있다. 평가명세서에 명시된 프로그램 스타일 평가 항목이 소스코드에 추가되어 표시되면 이에 대한 점수를 부여할 수 있도록 구성되어 있다.



(그림 4) 프로그래밍 스타일 평가 화면

평가의 마지막 단계로 프로그램 소스코드 표절 여부를 확인한다. 프로그램 과제 평가에 있어 주요 요인 중의 하나는 프로그램 소스코드의 표절을 검사하는 것이다. 프로그램 표절의 형태에는 여러 가지가 있을 수 있으나 대표적인 방법으로는 인터넷상에 존재하는 이미 구현된 소스코드나 다른 학생들의 과제를 복사해서 변수이름, 코멘트 등의 간단한 편집 과정만을 통해 제출하는 것이다 [1]. 이를 탐지하거나 방지하기 위한 대책 마련으로 국내외 일부 대학에서 학생들의 표절 행위를 자동으로 검사하는 시스템을 연구 개발하여 제공하고 있다 [8, 9 10, 11]. 본 과제 평가 시스템에서는 1994년 버클리(Berkeley) 대학에서 개발되어 2006년부터 스탠포드(Stanford) 대학에서 운영중인 MOSS (Measure of Software Similarity)을 사용하였다 [9]. MOSS는 C, C++, Java, Visual Basic, Javascript, Perl, Pascal, 혹은 Lisp 등의 언어로 구현된 프로그램 소스코드의 유사성을 자동적으로 판정하는 시스템이며 주로 프로그래밍 교과목의 프로그램 과제 표절을 검사하기 위한 도구로 이용되어 왔다. 과제 평가 화면에서 "프로그램 소스코드 표절 검사" 항목을 선택하면 시스템은 MOSS 서버에게 표절 검사를 의뢰하게 되고 MOSS 서버는 표절 검사를 수행한 후 유사성이 있는 소스코드 파일명 쌍들에 대한 리스트, 상호 중복 백분율, 일치한 토큰 및 라인의 수 등이 포함된 HTML 페이지를 생성하여 이 페이지가 포함된 URL을 통보하여 준다. 유사한 두 프로그램을 좌우 두 프레임 안에 나란히 두고 유사성의 레벨에 따라 소스코드를 다른 배경색으로 표시하여 표절 여부를 쉽게 확인할 수 있는 HTML 페이지도 포함되어 있다. 실험을 통해 50% 이상의

상호 중복이 있는 경우에는 표절이 거의 확실했지만 기계적인 평가 결과에만 의존할 경우 학생들이 평가 결과에 불만을 제기하는 문제들이 야기되어 자동적으로 검사 결과를 시스템에 등록하지 않고 해당 학생들에게 이-메일을 통해 표절 여부를 알린 후 서면이나 면담을 통해 최종적으로 표절 여부를 결정하였다. 즉 평가자가 과제 평가 화면에서 "프로그램 소스코드 표절 검사 결과 등록"을 통해 최종 결과를 등록하도록 구성하였다.

과제 평가 화면에서 "과제 평가 완료"를 선택하면 컴파일 결과, 검증자료별 실행결과와 점수, 성능평가의 총점, 소스코드에 추가한 피드백과 스타일 평가 항목별 점수, 소스코드 표절검사 결과 및 평가 총점이 포함된 평가보고서가 자동 생성된다. 이와 함께 시스템은 과제평가가 완료되었음을 교수자 (혹은 평가자)와 학생들에게 이-메일을 통해 알린다. 학생들은 과제 평가보고서를 출력하여 평가 결과를 확인할 수 있다.

5. 평가 시스템 운영 실험 및 결과

프로그래밍 과제 평가 시스템 PAGS의 유용성을 실험하기 위하여 2007년도 1학기 D대학교 컴퓨터교육과에 개설된 3개의 프로그래밍 관련 교과목(컴퓨터프로그래밍:C, 자바프로그래밍 및 데이터 구조)에 적용하였다. 해당 교과목에 수강한 학생의 총 수는 48명이었으며 총 6개의 과제에 적용하였다.

5.1 과제 평가 수행시간

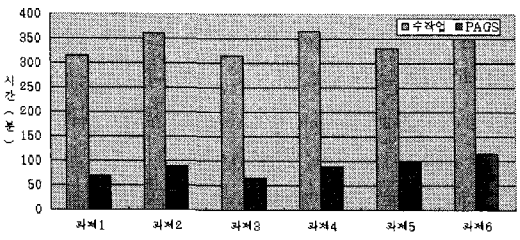
프로그램 과제 평가를 자동화하는 궁극적인 목적 중의 하나는 최소의 시간과 노력으로 주어진 시간 안에 공정하게 과제를 평가하기 위함이다. 따라서 제안된 시스템의 과제 평가 수행 시간의 효율성을 평가하기 위하여 본 논문에서는 6개의 과제에 대하여 수작업에 의한 평가 소요시간과 PAGS에 의한 평가 소요시간을 비교 분석하였다. 2명의 평가자가 참여하여 동일 과제에 대하여 한 평가자는 수작업으로, 다른 한 평가자는 PAGS를 이용하여 과제를 평가하도록 하였다. 프로그래밍 스타일 평가는 PAGS를 이용한 경우에도 반 수작업으로 수행됨으로 이에 대한 소요시간은 비교 분석에서 제외시켰다.

표 1은 과제 평가에 대한 실험 결과를 보여주고 있으며 소요시간은 반올림하여 분으로 표시하였다. 그림 5는 표 1의 데이터의 총소요시간을 그래프로 표현하고 있다. 표 1의 PAGS에 의한 성능 평가 소요시간에서 괄호 밖에 있는 숫자는 시스템에 의한 자동 성능 평가 소요시간을 나타내며 괄호 안의 있는 숫자는 자동 성능 평가가 실패한 경우에 실패한 과제에 대하여 수작업으로 프로그램 소스 코드 및 출력 결과 등을 점검하여 재평가하는데 소요된 시간을 나타낸다.

PAGS를 이용한 경우의 평가 총소요시간이 크게 감소되었음을 알 수 있다. 자동 성능 평가에 소요된 시간은 아주 짧았으며 소스코드의 라인 수 및 사용된 검증자료의 개수에도 별 영향이 없

(표 1) 과제 평가 수행 결과

과목명	학생수	과제	언어	소스코드 라인 수	수작업에 의한 평가 소요시간(분)			PAGS에 의한 평가 소요시간(분)		
					성능	표절검사	총소요시간	성능	표절검사	총소요시간
기본 프로그래밍	21명	과제1	C	80~100	120	195	**315**	2 (35)	32	**69**
		과제2	C	120~140	125	235	**360**	2 (40)	48	**90**
자바 프로그래밍	12명	과제3	Java	260~300	75	240	**315**	3 (30)	33	**66**
		과제4	Java	410~460	86	280	**366**	3 (35)	51	**89**
데이터구조	15명	과제5	C	250~300	85	245	**330**	3 (45)	49	**97**
		과제6	C	300~350	90	260	**350**	3 (50)	62	**115**



(그림 5) 과제 평가 소요시간 비교 그래프

음이 관찰되었다. 위에서 언급한 바와 같이 등록된 검증자료를 사용하여 일괄적으로 프로그램 성능 평가를 수행한 후 성능 평가가 실패한 과제에 대해서는 수작업으로 프로그램 소스 코드 및 출력 결과 등을 점검하여 재평가하였으며 이에 다소 시간이 소요되었다. 다시 말해, 자동 성능 평가에 실패한 과제에 대해서는 시스템의 출력 결과와 학생의 출력 결과를 수작업으로 재검토하였다. 실험 결과에 의하면 과제명세서에 명시한 구분문자(공백, 콤마, 탭 등의 경계기호 및 사용개수)의 사용 불일치와 불필요한 문자 출력 및 오류처리 등으로 인해 자동 성능 평가에 실패한 사례들이 관찰되었다. 이러한 경우에는 출력 부분의 소스코드만을 수정한 후 수정한 부분에 코멘트를 하고 벌점을 부여하여 개별 평가 방법으로 과제를 재평가하여 점수를 부여하였다. 또한 학생들이 사소한 혹은 아주 간단한 프로그램 오류로 인해 성능 평가가 실패하였음을 제시한 경우에는 학생들이 요구한 부분의 소스코드만을 수정하여 프로그램을 재평가하였다. 표 1의 PAGS에 의한 평가의 성능 부분에서 괄호 안의 있는 시간은 바로 이러한 경우를 처리하는데 소요된 평가시간을 의미한다. 그러나 전체적인 성능 평가 소요시간은 수작업에 의한 평가 소요시간에 비하면 크게 단축되었음을 알 수 있다. 또한 학생들이 여러 과제 수행을 통해 이러한 종류의 오류에 점차 주의를 기울이게 될 것이라 사료되며 따라서 이러한 경우를 평가하는데 소요되는 시간 또한 감소될 수 있으리라 판단된다.

표절검사에 소요된 시간은 학생수와 소스코드 라인 수에 비례하였으며 수작업에 의한 평가 시 정확성을 보증할 수 없음이 관찰되었다. 실제로 수작업 표절 검사 결과가 PAGS의 의한 검사 결과와 상이한 결과를 보인 경우가 있었으며 (28%), 특히 학생의 수가 많은 경우, 변수 및 함수이름 변경, 반복문 및 조건문 유형변경, 구조변경 등의 경우 수작업으로 표절 여부를 정확히 판별한다는 것은 어려운 일임을 확인할 수 있었다. PAGS에 의한 표절 검사 시간은 MOSS에서 제공된 표절 검사 결과를 평가자가 다시 한번 재확인하는 시간으로 인해 다소 길게 소요되었으나 이 또한 수작업에 의한 평가 소요시간과는 큰 차이가 있음을 알 수 있다.

5.2 학습자의 선호도

학습자의 관점에서 시스템의 효율성을 평가하기 위하여 실험에 적용된 3개 교과목(기본프로그래밍, 자바프로그래밍, 데이터구조)의 각 교과목의 하나의 과제에는 기존의 수작업 평가 방법을 적용한 후 PAGS를 이용한 경우와 비교하여 학습자의 선호도를 설문조사를 통하여 실시하였다. 설문조사에 참여한 총 학생의 수는 40명이었다. 3개 과목에 등록된 학생의 총수는 48이었으나 다중 등록된 학생들은 한번만 설문조사에 참여하도록 하였으며 이로 인해 학생수에 차이점을 보인다. 이 조사에서 평균 76%의 학생은 기존 방식에 비해 PAGS 방법을 선호하는 것으로 나타난다. 약 24%의 학생은 기존 방식을 선호하였으며 그 이유를 분석한 결과, 기존 방식으로는 컴파일이 실패한 경우나 프로그램이 완전히 수행되지 않은 경우에도 프로그램 성능 평가에서 부분 점수를 받았지만 PAGS에 의한 평가에서는 그러한 요소가 배제되었기 때문이라고 응답하였다. 또한 기존 평가 방법에서는 부분적인 표절의 경우 문제 제기가 되지 않았지만 PAGS 방법으로는 모든 표절 부분이 측정되어 문제가 제기됨으로 이에 대한 거부 반응이 있었던 것으로 조사되었다. 대부분의

학생들은 새로운 평가 방법에서 과제를 제출하기에 앞서 제공되는 검증자료를 통해 자신의 프로그램 성능을 시험해 볼 수 있다는 점과 모든 평가 항목 및 피드백이 명확히 기술되어 있는 정확한 평가 결과보고서 제공 측면에서 PAGS에 의한 평가 방법을 특별히 선호하는 것으로 나타난다.

6. 결론 및 향후과제

본 논문에서는 교수자가 웹을 기반으로 프로그래밍 과제 평가 절차를 총괄적으로 관리할 수 있는 프로그래밍 과제 평가 시스템(PAGS: Programming Assignment Grading System)을 개발하였다. 특별히 평가자 측면에 초점을 맞추고 평가자가 적절할 피드백과 함께 학생들의 과제를 일괄적으로 쉽게 평가할 수 있을 뿐만 아니라 프로그램 소스코드의 표절 또한 용이하게 검사할 수 있는 평가 시스템을 제시하였다.

본 논문에서 제시한 과제 평가 시스템을 3개의 프로그래밍 관련 교과목 6개 과제에 적용한 결과 기존의 수작업 평가 방법에 비해 과제 평가 소요 시간이 크게 감소되었음을 보여주었다. 따라서 제안된 평가 시스템은 과제를 정확하게 평가할 수 있고 교수자의 평가에 대한 부담감을 해결할 수 있기 때문에 학습자에게 문제 해결 연습 기회를 많이 제공해 줄 수 있는 기반이 되리라 믿는다. 또한 제안된 평가 방법은 인터넷으로부터 혹은 동료 학생들의 과제를 복제해 제출하는 문제점을 해결하고 학생들 스스로 문제 해결 능력을 키워 나갈 수 있는 동기 부여가 되어 궁극적으로는 학습자의 학업 성취도를 향상시킬 수 있는 발판이 되리라 판단된다. 대부분의 학습자들 또한 기존 평가 방식에 비해 새로운 평가 방법을 선호하는 것으로 나타났으며 특히 피드백과 함께 모든 평가 항목이 정확히 기술되어 있는 평가 결과보고서에 만족함을 보였다.

본 연구의 향후 과제로는 편리한 학습 환경을 제공하는 웹 기반 강의 지원 시스템인 CMS(Course Management System) 혹은 LMS (Learning Management System)와 연계하여 교과목에 관련된 모든 과제 및 성적(중간, 수시, 기말고사)을 통합하여 관리할 수 있는 시스템에 관한 연구가 필요하다고 본다. 더불어 과제 평가자가 한명 이상인 경우의 평가 절차 및 좀 더 다양한 프로그램 스타일 평가 항목 등을 제공하여 좀 더 유연한 평가 시스템으로의 확장이 필요하다고 본다.

참 고 문 헌

[1] Cheang, B., Kurnia, A., Lim, A. and Oon, W.C., "An automated grading of programming assignments in an academic institution", Computer & Education, Elsevier Science, Vol.41, No.2, pp. 121-131, 2003.

[2] 김영지, 김현철. "분산교육환경에서의 프로그래밍 교육 시스템의 설계 및 구현", 2005년도 한국컴퓨터교육학회 동계 학술발표논문집, 제9권, 제1호, pp. 67-69, 2005.

[3] Jackson, D., "A semi-Automated Approach to Online Assessment", In Proceedings of Conference on innovation and Technology in Computer Science Education(ITiCSE'2000), ACM Press, pp. 164-167, 2000.

[4] Kurnia, A., Lim, A., Cheang. B., "Online Judge", Computers & Education, Elsevier Science, Vol.36, No.4, pp. 299-315, 2001.

[5] Saikkonen, R., Malmi, L., Korhonen, A., "Fully Automatic Assessment of Programming Exercise", ITiCSE'01, ACM Press, pp. 133-136, 2001.

[6] Pardo, A., "A Multi-Agent Platform for Automatic Assignment Management", ITiCSE'02, ACM Press, pp. 60-64, 2002.

[7] Edlun, J.R., "What is "plagiarism" and why do people do it?", Writing Centre Director,

California State University, LA, USA, 1998.
[8] 강은미, 황미녕, 조환규. "유전체 서열의 정렬 기법을 이용한 소스 코드 표절 검사", 정보과학회논문지:컴퓨팅의 실제, 제9권, 제3호, pp. 352-367, 2003.
[9] Aiken, A., "MOSS(Measure Of Software Similarity): A System for Detecting Software Plagiarism", http://theory.stanford,edu/~aiken/moss, University of Stanford, 2006.
[10] Malpohl, G., "JPlag: Detecting Software Plagiarism", available http://www.ipd.uka.de:2222, Karlsruhe, Germany, 2006.
[11] 김영철, 황석천, 최재영. "프로그램 유사도 평가 알고리즘", 한국인터넷정보학회논문지, 제6권, 제1호, pp. 51-64, 2004.

◐ 저 자 소 개 ◑

김 미 혜(Kim, Mihye)
1984년 전북대학교 전산통계학과 졸업(학사)
1999년 University of New South Wales 컴퓨터공학과 졸업(석사)
2003년 University of New South Wales 컴퓨터공학과 졸업(박사)
2004년 9월~현재 대구가톨릭대학교 컴퓨터교육과 교수
관심분야 : 지식 관리 및 검색, 온토로지, 컴퓨터교육, 컴퓨터보안 etc.
E-mail : mihyekim@cu.ac.kr