3

컴퓨터학과 학부 프로그램 재고

POSTECH ▌박 성 우

1. 서 론

전 세계적으로 CS 학과[1]의 학부 지망생 수는 2000년 이후로 꾸준히 줄어들고 있다[1,2]. 이른바 닷컴 버블의 전성기를 정점으로 CS 학부 지망생 수는 계속 줄어들었으며, 현재는 바닥에 도달했다고 보거나 미국의 월가를 중심으로 한 금융권의 붕괴와 함께 어느 정도 회복기에 들어섰다는 전망이 나오는 정도이다. 산업계에서 CS 관련 업종의 비중으로 볼 때 현재의 CS 학부 지망생 수는 비정상적으로 낮으며, 이는 조만간에 CS 관련 산업 전체의 위기로 이어질 수 있다고 예상되고 있다.

우리나라도 예외가 아니어서 CS 학과의 학부 지망생 수는 급격하게 떨어지고 있다. 이는 전자컴퓨터학부라는 통합 학부를 통해서 학생을 우선 선발한 뒤, 전기전자공학과와 CS 학과 중 선택할 수 있게 하는 학교나 학생들이 자유롭게 과를 정할 수 있게 하는 학교에서 두드러지게 나타난다(예: 서울대학교, KAIST, POSTECH).

더 심각한 문제는 이렇게 CS 학과를 지망한 학생들도 졸업 후 관련 산업으로 진출하는 비율이 다른 학과와 비교했을 때 상대적으로 낮다는 것이다. CS 학과 졸업생이 금융권에 취직하는 경우는 어렵지 않게 볼 수 있고 아예 전공을 바꾸어서 다시 학업을 시작하는 경우도 흔하다. 이런 추세가 지속되면 아직 국제적인 경쟁력을 갖추지 못한 우리나라의 소프트웨어 산업은 더 경쟁력을 잃게 되고, 최고의 제품만이 모든 시장을 독점하는 소프트웨어 산업의 특성상 외국 기업에 시장을 통째로 넘겨주게 되는 상황으로 치닫게 될 수도 있다.

CS 관련 산업은 그 자체로서 전체 산업계에서 차지하는 비중도 높지만, 군수, 항공, 자동차, 가전 등 CS와 직접 연관이 없는 분야에서도 핵심적인 부분을 차지하고 있다. 예를 들어 자동차 설계에서 소프트웨어 개발 비용이 전체 개발 비용의 30%가 넘는다는 것은 이미 잘 알려진 사실이다. 또한 컴퓨터를 제대로 이용하기 위한 계산적 사고는 컴퓨터 관련 연구에서만 필요한 게 아니라 거의 모든 자연과학 및 공학 분야의 연구에서 향후 핵심적인 역할을 할 것으로 전망되고 있다[3]. 이렇게 CS 관련 기술에 대한 수요는 꾸준히 있고 계산적 사고의 중요성은 계속 증가하는데도 CS 학과의 학부 지망생 수는 왜 계속 떨어지고 있을까?

2. CS 학과 쇠퇴의 이유

CS 학과의 학부 지망생 수가 줄어들고 있는 데는 결정적인 한 가지 이유가 있는 것이 아니다. 이는 아래에 나열한 여러 요소들이 복합적으로 작용하여 고등학생들이 대학교를 진학할 때 CS 학과에 등을 돌리거나, 대학에 들어온 신입생이 CS 학과를 선택하지 않거나, CS를 전공한 학생이 관련 분야로 진출하지 않고 다른 분야로 진로를 모색하는 결과로 이어지기 때문이다.

- 중학교나 고등학교에서 소개하는 컴퓨터 교육은 주로 프로그래밍 언어이고 결과적으로 CS는 프로그래밍과 같다는 선입관을 주입시킨다. 체계적인 프로그래밍 언어 교육이 현실적으로 불가능하므로 수학이나 다른 기초 과학 과목에 비해서 상대적으로 적은 학생만이 흥미를 가지게 된다.
- 20년 전과는 달리 컴퓨터는 사회의 모든 분야에서 이용되고 있고 생활의 일부분이 되었기 때문에 컴퓨터가 지적 호기심을 자극하는 기계로서 인식되지 않는다. CS 분야의 해결되지 않는 학문적 문제가 무수히 많다는 사실은 컴퓨터를 일상적으로 불편함 없이 사용하는 학생들에게 전달하기가 매우 어렵다.
- CS를 전공하면 할 수 있는 일의 범위는 제한되어

1) 우리나라의 컴퓨터관련 학과는 컴퓨터공학과, 컴퓨터과학과, 혹은 전산학과로 불린다. 본 기고문에서는 모든 과를 통칭하여 CS (Computer Science) 학과라고 부르기로 한다. 이 글에서 CS는 IT (Information Technology)로 해석해도 무방하다.

있고 업무량은 많지만 보수는 적다는 인식이 팽배해 있다. 또한 CS 관련 산업은 시간이 지날수록 인도나 중국을 통한 아웃소싱에 더 의존하게 될 것이라는 전망이 대중매체를 통해서 전해지고 있다.

- 대학 신입생들은 CS 전공 학생에 대해서 스테레오타입에 가까운 이미지를 가지는 경우가 많다. 항상 컴퓨터를 켠 채로 프로그래밍을 한다는 이미지 때문에 특히 여학생들의 전공 지망율은 매우 낮다. 또한 많은 학생들이 CS 전공을 위해서는 높은 수준의 수학 지식이 요구되고 학습해야 할 과목은 계속 증가하지만 응용 분야는 극히 제한적이라고 생각하고 있다.
- CS 관련 기술은 변화가 빠르기 때문에 산업계로 진출한 인력도 지속적으로 자기 계발과 충전이 필요하다. 특히 다른 산업 분야와 달리 경력에 비례하는 기술 축적이 어렵기 때문에 새로운 기술을 꾸준히 습득해야 한다. 그러나 국내 CS 관련 산업계는 이런 현실을 외면하고 소프트웨어 개발 인력은 일회용 인력으로 인식하여 인력을 자주 대체하는 방식을 주로 채택하고 있다. 결과적으로 이미 CS 관련 산업계에 진출한 인력마저 전문 경력을 유지하는 게 어렵게 된다. 이는 장기적으로 산업계의 경쟁력 약화로 이어지고 CS 학과 졸업생의 진로를 제한하게 된다.
- 미국의 Microsoft나 Google, 독일의 SAP처럼 CS 기술로 승부하여 세계적인 회사로 성장한 예가 국내에 전무하다. 전공분야 결정을 눈앞에 두고 있는 학생들에게 이러한 잠재적인 롤모델의 부재는 CS 분야로의 관심을 유도하는데 부정적인 영향을 미친다. 사실 이런 롤모델 회사의 부재는 CS 학과 쇠퇴의 원인이라기보다는 결과이고 이러한 결과는 다시 CS 학과 발전에 부정적인 영향을 미치는 악순환으로 해석해야 한다.
- 정부는 소프트웨어 산업을 육성하고자 하지만 그 제도의 시행에 있어서는 이상과는 거리가 멀다. 정부 자체에서 소프트웨어를 하드웨어와 동등하게 취급하여 그 가치를 인정해주기 시작한 것은 불과 몇 년 밖에 되지 않았다. 우주과학과 자연과학 분야에서 지속적으로 대국민 홍보를 하여 장기적인 경쟁력 강화를 준비하는 것과는 달리 CS 관련 분야는 이러한 노력의 흔적이 전무하다.

CS 학과의 지망생이 줄어드는 배경에는 이렇게 복합적인 이유가 악순환의 형태를 이루고 있다. 결정적인 원인이 없고 여러 원인이 함께 작용하여 나온 결과이므로 문제의 해결책도 결정적인 한 가지가 있을 수 없다. 정부의 소프트웨어 산업 육성 정책을 강화하는 것 하나만으로 CS 학과의 경쟁력이 올라갈 수 없으며, CS 학과 지망생이 늘어도 산업계에서 편리한 방법만을 추구한다면 결국에는 현재보다 더 나쁜 상황으로 빠져들 수밖에 없는 것이다. 따라서 CS 학과를 더 발전시키고 관련 분야의 국가적 경쟁력을 발전시키고자 하면 중,고등학교, 대학교, 산업계, 정부가 모두 현재의 문제를 인식하고 체계적인 변화를 시도해야 한다.

그렇다면 이러한 만성적인 CS 학과 쇠퇴라는 문제를 해결하기 위해서 가장 우선적으로 변화해야하는 부분은 어디일까? 그것은 CS 교육의 시작을 담당하는 CS 학부 교육이 아닐까 한다. CS 학부 교육이 재정비되면 졸업생의 경쟁력이 강화되고 우수한 졸업생은 대학원으로 진학을 하든 산업체로 진출하든 CS 분야의 핵심 홍보대사가 된다. 반대로 부실한 CS 학부 교육은 CS에 어렵게 진학한 학생들마저 다른 분야로 진로를 바꾸게 만드는데 이렇게 진로를 바꾼 학생은 지속적으로 CS 분야에 대한 부정적인 이미지를 전파하게 된다. CS 학부 교육은 장기적인 안목을 가지고 심혈을 기울여 운영해야 하는 가장 중요한 부분 중의 하나인 것이다.

3. 현재 CS 학부 교육의 문제점

CS 학부 지망생 수가 낮다는 문제는 일시적이거나 단순한 해결책이 효력을 발휘할 수 있는 문제가 아니다. 학과의 이름을 바꾸거나 학생들의 장학금 제도를 대폭 확대하거나 졸업요건을 완화시키거나 하는 것은 일시적인 효과를 낼 수는 있겠지만 근본적인 문제에 대한 손질 없이는 장기적으로는 오히려 역효과만 불러일으키게 된다. 정부의 이공계 산업에 대한 근본적인 인식 변화 없이 이공계 장학금만을 지원하는 것이 이공계 문제의 근본적인 해결책이 될 수 없는 것과 같은 이치이다. 이러한 뿌리 깊은 현상은 가장 기초적인 단계에서 발생하는 문제를 손질할 때만 해결할 수 있는 것이다. 즉 대학에서 제공하는 교육의 질 자체를 끌어올려야 하는 문제로 귀결된다.

현재 대부분의 국내 대학의 CS 교육 체계를 보면 특별히 문제를 삼을 만한 점은 없는 것으로 보일 수 있다. 기초 프로그래밍, 이산수학, 논리설계, 자료구조, 시스템프로그래밍, 운영체제, 컴퓨터 구조, 알고리즘, 프로그래밍언어, 오토마타 이론 등의 과목은 대부분의 대학에서 개설되며 학부 2년 또는 3년 과정에서 수강하도록 권장되고 있다. 심화 과목의 경우도 대이

터베이스, 인공지능, 그래픽스, 네트워크, 컴파일러 등 전통적 CS 과목이 주류를 이루고 있으며, 데이터마이닝이나 기계학습 등과 같이 최근에 중요성이 부각된 새로운 분야의 과목이 추가적으로 개설되고 있다.

그러나 이러한 정형화된 CS 교육 체계는 20년 전에 이미 완성된 것으로서 현재도 여전히 유효한지는 재고해 볼 필요가 있다. 예를 들어 20년 전 기초 프로그래밍 과목에서 C 언어를 주로 이용했지만 지금도 C 언어가 최상의 선택인지는 논란의 여지가 깊은 문제일 수 있다. 또한 과목 자체의 내용도 다시 고려해 보아야 한다. 예를 들어 20년 전 프로그래밍언어 과목에서 다루던 내용이 지금도 학생들의 지적 호기심을 자극하고 CS의 기초 과목에서 다루는 내용으로 충분한지는 다시 고려해 봐야 한다. 결국 CS의 변화에 맞추어서 CS의 학부 교육 내용도 바뀌어야 했지만 현재의 전형적인 국내 CS 교육 체계를 고려해 볼 때 이는 제대로 시행되지 않았던 것으로 보인다. 산업체는 생존 경쟁의 특성상 시대의 변화에 맞추어 지속적으로 변화할 수밖에 없었지만 CS 교육은 그렇지 않았으며, 결과적으로 CS 졸업생의 실제 능력과 산업체에서 기대하는 CS 졸업생의 능력 사이의 격차가 커지게 된 것이다.

시대의 변화에 맞춰 바뀌지 않은 CS 교육 체계의 문제점을 암시하는 예는 심화과목의 진입 장벽이 다른 학과에 비해서 낮다는 것이다. 적지 않은 CS 심화과목은 데이터구조나 이산수학과 같은 기초과목을 제외하고는 선수과목을 따로 요구하지는 않는다. 이런 과목은 20년 전에는 새로운 학문을 소개하는 CS의 심화과목으로 충분했겠지만 지금은 CS의 기초적 지식이 있는 사람이라면 누구나 수강할 수 있는 과목으로 인식되고 있다. 물리학과 학부 4년생이 CS 학부 심화과목을 큰 어려움 없이 수강할 수는 있겠지만 그 반대는 매우 어려울 것이다.

CS 학부 교육 체계를 어떤 방향으로 개선해야 하는가 하는 문제에 대해서 모든 사람이 공감하는 최선의 답을 찾기는 어려울 것이다. 그러나 지난 20년간 그다지 변하지 않았던 CS 교육 체계를 시대의 요구에 맞게 혁신시키는 것이 더 이상 늦출 수 없는 시급한 과제가 되었다는 데는 이견이 없을 것이다. CS 학부 교육은 CS 학과만이 제공할 수 있는 차별화된 내용으로 재정비되어야 하고 CS 학과의 쇠퇴가 피부로 느껴지는 지금이 이를 위한 최적의 시기이다.

4. CS 학부 교육의 개선 방향

시대의 요구에 맞게 재정비된 CS 학부 교육의 총체적 틀을 제시하는 대신에 몇 CS 교과목을 개별적으로 분석하여 문제 해결을 위한 한 가지 방향을 제안해 보고자 한다. 이 제안의 핵심은 1) 각 교과목은 20년 전이 아닌 현재의 CS 분야 상황을 고려해서 설계되어야 하고 2) 각 교과목은 CS 학과에서만 제공할 수 있는 차별화된 내용을 다루어야 한다는 것이다. 여기서 제안하는 방향은 CS 학부 교육을 재정비하는데 이용할 수 있는 하나의 의견으로 해석하면 된다. 그리고 분석하는 CS 교과목은 CS 학부 교육의 대표적인 과목이라고 볼 수는 없으며 CS 학부 교육의 개선을 위해서 변화가 필요한 예로서 참고하면 된다.

(1) 기초 프로그래밍

기초 프로그래밍 과목은 프로그래밍을 처음 가르치는 과목으로서 예전에는 C, Pascal, Fortran 등을 다루었지만 현재는 C와 Java가 대세를 이루고 있다. 이는 C와 Java가 산업계에서 가장 널리 쓰이는 언어 중의 하나이기 때문일 것이다. 그러나 언어를 선별하기 전에 먼저 고려해야 하는 것은 기초 프로그래밍 과목의 목표이다.

기초 프로그래밍 과목의 목표가 이후 수강할 CS 교과목에서 필요로 하는 프로그래밍 능력을 배양하는 것이 목표라면 C, C++, Java 등과 같이 이후 교과목에서 이용하는 언어를 선택하는 것이 필수적이다.

그러나 목표가 프로그래밍의 핵심 개념을 전달하고 CS에 대한 관심을 유발하면서 제한적으로나마 CS라는 학문을 소개하는 것이라면 C나 Java 같은 언어는 좋은 언어라고 볼 수 없다. 예를 들어 프로그래밍을 처음 배우는 학생들에게 컴파일 과정 자체가 이미 어려운 단계로 느껴질 수 있고 이후 프로그래밍의 본질과는 거리가 먼 언어 요소를 습득해야하는 것은 교육 목표에 정반대되는 과정이기 때문이다. 이 경우는 인터프리터를 지원하여 접근하기 쉽고 문법도 간단한 스크립트 언어나 함수형 언어가 대안이 될 수 있다.

기초 프로그래밍 과목에서 스크립트 언어나 함수형 언어를 채택하는데 어려운 점의 하나는 CS 학과에 이런 언어로 학부 교육을 받은 교수가 별로 없다는 것이다. 이는 비단 우리나라만의 문제가 아니며 전 세계적인 문제이다. 그러나 상당수의 미국과 영국의 대학에서는 Scheme이나 Haskell 같은 함수형 언어를 기초 프로그래밍 과목에서 이용하고 있다.

(2) 논리설계 및 시스템프로그래밍

논리회로를 설계하고 직접 만들어보는 것이 주된 흐름인 논리설계 과목과 어셈블리 프로그래밍과 C

프로그래밍을 통해서 하드웨어 레벨의 프로그래밍을 익히는 시스템프로그래밍 과목은 CS 교육 목표에 따라서 급진적인 변화가 필요한 과목이다. CS 교육의 목표가 소프트웨어 설계와 구현에 전문화된 엔지니어를 양성하는 것이라면 저수준의 하드웨어가 본질적인 문제가 아닌 현재는 논리회로 과목은 CS 교육 과정에서 제외하고 시스템프로그래밍 과목은 전면 개편하는 것이 바람직하다. 시스템 프로그래밍 과목에서는 소프트웨어를 구현하는 프로그래머의 관점에서 컴퓨터 시스템이 어떻게 작동하는지를 이해시키면 되므로 어셈블리 프로그래밍 언어 교육은 불필요하며 C 언어와 같은 저수준 언어로 실습을 하는 것이 충분하다(미국의 유수 CS 프로그램에서는 논리회로 과목은 CS 교과과정에서 제외했으며 시스템프로그래밍 과목에서는 어셈블리 언어를 다루지 않는다).

이런 CS 교과목을 급진적으로 바꾸기 위해서는 현재 CS 학부 교육에서 이들 과목에서 전통적으로 다루었던 내용이 앞으로도 계속 중요한 위치를 고수할 것인지에 대한 전반적인 의견 일치가 필요할 것이다.

(3) 프로그래밍 언어

현재 대부분의 프로그래밍 언어 과목에서 다루는 내용은 20년 전 학부 교과목에서 다루었던 내용과 그다지 차이가 나지 않는다. 이렇게 오래된 내용은 현재 프로그래밍 언어 분야에서의 연구와는 거의 연관이 없으며 그 깊이에 있어서도 CS 전공자가 아닌 사람도 혼자서 어렵지 않게 공부할 수 있는 내용이다(예: 프로그래밍 언어의 역사). 이런 내용으로 교육 받은 학생은 프로그래밍 언어 이론 분야에서 어떤 연구를 수행하는지에 대해서 그릇된 인식을 갖게 되고 또한 이런 그릇된 인식을 전파하게 된다.

프로그래밍 언어 과목에서는 이렇게 현실과 동떨어진 내용을 다룰 게 아니라 현재 활발하게 진행되고 있는 연구를 수행하고 있는 기본적 지식을 연마할 수 있는 기회로 제공되어야 한다. 멀티코어 아키텍쳐의 등장과 함께 활발하게 연구되고 있는 병렬 프로그래밍, 객체지향 프로그래밍의 한계가 인식된 뒤 Java, C# 등에 도입되고 있는 함수형 프로그래밍, 소프트웨어의 크기가 증가하면서 그 중요성이 커지고 있는 소프트웨어 분석 및 검증, 프로그램의 사양을 직접 표현할 수 있게 해 주는 타입이론과 같은 주제의 기초적인 지식을 이 과목에서 다룰 수 있어야 한다.

(4) 컴파일러

컴파일러 과목은 CS 학부 교육에 반드시 포함되고 보통 컴파일러를 한 학기에 구현하는 것을 목표로 하는 대표적인 CS 과목이다. 그러나 컴파일러를 강의한다는 사실만으로 CS 교육에서 컴파일러가 커버된다고 결론 내리는 것은 위험하다. 왜냐하면 컴파일러 구현은 그 복잡도가 매우 높아 다루는 내용에 있어서 극단적인 차이가 학교마다 있을 수 있기 때문이다.

예를 들어 20년 전에 컴파일러 과목에서 강의했던 내용은 당시로서는 컴파일러 분야의 최신 이론과 동향을 접해 볼 수 있게 했겠지만 지금은 현실과는 동떨어진 내용일 수 있다. 여전히 많은 학교에서 컴파일러 과목은 구문분석 이론을 다룬 뒤 구문분석기를 구현하고 중간언어를 구현해 보는 단계에서 끝이 난다. 그러나 구문분석은 현재 거의 완성된 기술이므로 학생들은 구문분석기를 사용하는 방법만 익히고 코드 생성 및 최적화 같은 실질적인 문제에 더 많은 노력을 쏟을 수 있도록 설계하는 것이 현 시대의 추세를 반영한 교육이라고 할 수 있다.

5. 결 론

CS 학과의 학부 지망생 수 감소라는 문제는 대학의 CS 학부 교육 뿐 아니라 중, 고등학교 교육, 정부의 정책, 산업계의 문화에 걸쳐 모두 변화가 있어서 해결될 수 있는 뿌리 깊은 문제이다. 본 기고문에서는 이러한 다방면의 노력의 한 부분으로서 CS 학부 교육을 시대에 맞게 수정하고 보완해야 하는 중요성을 피력하고 몇 가지 교과목을 예로 들어 제한적이나마 변화 방향을 제시했다.

CS 학부 프로그램의 개편이 성공하기 위해서는 교육의 중요성을 대학과 정부가 인식하고 CS 학부 교육을 책임질 교수의 능력 평가를 현재 SCI 연구 논문 수로써 결정짓는 제도도 함께 바뀌어야 할 것이다. 그러나 가장 중요한 점은 각 과목을 책임지는 교수 개개인의 교육에 대한 노력이 모여 CS 학부 프로그램의 경쟁력이 결정된다는 점이다.

참고문헌

[1] Jacob Slonim, Sam Scully, Michael McAllister. Cross-roads for Canadian CS Enrollment. Commnications of the ACM, 2008년 51권 10호, 66-70. ACM.

[2] Vesgo, J. Low Interest in CS and CE Among Incoming Freshmen. Computing Research Association, USA, 2007. http://www.cra.org/wp/index/.php?=p104.

[3] Towards 2020 Science, Microsoft Research. http://research.microsoft.com/en-us/um/cambridge/projects/towards2020science/.

박 성 우

1996 KAIST 전산학과 학부졸업
1998 KAIST 전산학과 석사졸업
2005 카네기멜런대학 전산학과 박사졸업
현재 POSTECH 컴퓨터공학과 조교수
E-mail : gla@postech.ac.kr

The 1st Workshop on Korean Database Society

- ❑ 일　　자 : 2009년 2월 26일~27일
- ❑ 장　　소 : 제주 라마다 호텔
- ❑ 주　　최 : 학회 데이터베이스소사이어티
- ❑ 상세안내 : http://db.kookmin.ac.kr/dbsw2009/