컴퓨터공학, 그리고 수학

2021-15738 양석훈

[요약]

컴퓨터는 만들겠다는 목적으로 인해 만들어진 것이 아닌, 수학의 한계를 확인하는 과정에서 우연히 만들어진 부산물이라는 사실을 알 수 있었다. 컴퓨터를 21세기로 안내해주는 프로그램 역시도, 수학의 역할이 중요하게 작용하였다. 이렇듯, 컴퓨터는 만들게 된 동기부터 발전과정까지, 수학의 영역과 매우 밀접하다는 사실을 알 수 있었다. 튜링은 이렇게 수학의 영역에서 튜링 머신이라는 기계를 고안함으로써 컴퓨터의 시초를 만들어 낸 것이다.

강연에서는 튜링은 천재가 아니라 충분히 비슷한 성과를 낼 수 있을 법한 일이라고 말하셨지만, 오히려 강연을 듣고 여러 자료를 보면서 튜링이 천재라는 사실을 더욱 강하게 느낄 수 있었다. 튜링이 이렇듯 단순히 증명방법을 받아들이지 않고 새로운 아이디어로 다른 증명에 도전했기에 현재의 컴퓨터가 나오게 된 것이라고 생각한다. 앞으로 컴퓨터공학 분야에서 발전할 주요 기술에 대한 논의 역시도 인상깊었다. 예시로, 요즘 가장 유명한 기술인 인공지능에 대한 설명이 있었다. 여러 읽을거리 들을 읽어보는 와중 나에게 잘 와닿지 않는 점들도 있었다

이번 읽을거리들을 읽으면서 가장 궁금했던 의문은 "그렇다면, 내가 앞으로 컴퓨터공학을 계속 공부해 나가기 위해서 필요한 수학의 범위는 어디일까?" 라는 것이다. 두 번째 질문은 "수학적 논리와 컴퓨팅의 연관성이 얼마나 클까?" 라는 것이다. 마지막 질문은 "튜링 머신으로부터 혹은 lambda calculus로부터 고안된 컴퓨터는 완벽한 컴퓨터일까?" 라는 것이다. 만약, 튜링 머신이 완벽한 컴퓨터가 아니라면 컴퓨터는 어디까지 더 발전할 수 있을까.

[알게 된 것]

컴퓨터는 만들겠다는 목적으로 인해 만들어진 것이 아닌, 수학의 한계를 확인하는 과정에서 우연히 만들어진 부산물이라는 사실을 알 수 있었다. 나는 프로그래밍 언어 수업을 듣기 이전에는 컴퓨터라는 것은 계산을 돕기 위해서 만들어진 것이고, 최초의 컴퓨터인 에니악에서 발전해온 것으로 알고 있었다. 그러나, 최초의 컴퓨터를 고안하고 디자인한 것은 튜링이며, 이런 점에서 튜링 머신을 최초의 컴퓨터라고 할 수 있는 것이다. 즉, 컴퓨터의 탄생에는 동기부터 디자인까지 수학이 큰 영향을 미쳤음을 알 수 있다. 그리고, 교수님이 추측하신 튜링의 생각 전개 과정을 통해서

튜링 머신의 탄생 배경과 이를 위한 증명과정 역시도 새롭게 깨달을 수 있었다.

컴퓨터를 21세기로 안내해주는 프로그램 역시도, 수학의 역할이 중요하게 작용하였다. 프로그램은 일종의 증명 과정이며, 그렇기에 증명 과정을 단순화하는 것은 프로그램을 실행시키는 것과 동일하다. 우리가 프로그래밍 언어 수업시간에 사용하는 OCaml을 포함한 여러 언어들의 실행과정도 결국 lambda calculus라는 표현법을 통해서 단순화시켜가는 과정과 동일하다는 것이다. 그러나, lambda calculus는 결국 수학의 영역, 그 중에서도 논리적인 추론을 해가는 과정이기에 프로그램도 수학과 밀접한 연관을 가지는 것이다.

이렇듯, 컴퓨터는 만들게 된 동기부터 발전과정까지, 수학의 영역과 매우 밀접하다는 사실을 알 수 있었다. 이를 통해, 프로그래밍 언어 역시도 수학과 매우 밀접한 연관이 있다고 주장해볼 수 있을 것이다. 교수님이 수업시간에 말씀해주신 것처럼, 프로그래밍 언어의 주된 두 축인 튜링 머신과 lambda calculus 모두가 수학을 중심으로 디자인된 것들이다. 읽기자료들을 읽고, 궁금해져서 프로그래밍 언어와 관련된 몇 가지 점들을 찾아보았다. 최초의 고급 언어라고 불리는 Fortran 역시도 수학, 과학의 응용을 위해서 만들어진 언어로 계산을 대신해주는 역할을 한다. 또한, 수학적으로 최초의 프로그래밍 언어를 찾아본다면, 괴델이 불완정성 정리를 증명하면서 알고리즘을 추상화시켜서 사용한 원시 재귀 함수를 고르기도 한다. 결국, 수학과 프로그래밍 언어, 컴퓨터는 뗄 수 없는 관계라는 것이다.

튜링은 이렇게 수학의 영역에서 튜링 머신이라는 기계를 고안함으로써 컴퓨터의 시초를 만들어낸 것이다. 튜링 머신의 디테일은 다음과 같다. 무한히 많은 칸을 가지는 테이프와 테이프에 기록된 여러 상태들을 읽고 쓰는 장치, 그리고 테이프에 기록되는 심볼과 장치의 상태 등으로 구성된다. 이제, 이 장치가 테이프에 기록된 심볼을 읽으며, 장치의 상태를 변경시켜가며 우리에게 필요한 연산들을 진행하게 되는 것이다. 이제, 튜링 머신을 컴퓨터에 대응시켜 보자면, 테이프가 컴퓨터의 메모리에 해당하는 역할을 한다. 읽고 쓰는 장치는 컴퓨터의 입출력을 담당하는 파트에 해당하고, 각 심볼에 따라 어떻게 상태를 변경할지 정해주는 규칙, 작동규칙표가 컴퓨터의 핵심인 중앙처리장치에 대응되는 것이다. 이러한 튜링 머신은 컴퓨터의 다른 한 축이었던 lambda calculus에서 연산이 수행되는 과정과도 잘 대응된다. 이렇듯, 읽을거리들을 읽으며 튜링 머신과 lambda calculus가 무엇인지, 또 이들이 어떤 연관이 있고, 수학적 논리가 들어가며, 컴퓨터의 탄생비화 등을 알 수 있었다.

[느낀 것]

강연에서는 튜링은 천재가 아니라 충분히 비슷한 성과를 낼 수 있을 법한 일이라고 말하셨지만, 오히려 강연을 듣고 여러 자료를 보면서 튜링이 천재라는 사실을 더욱 강하게 느낄 수 있었다.

만약, 내가 현장에서 괴델의 증명방법을 들었다면, 새롭게 다른 방법으로 증명할 수 있는 방법을 찾는 것이 아니라 단순히 이를 받아들이고 넘어갔을 것 같다. 또한, 스스로 '기계적인 방식'을 직접 정의하며, 이를 이용해서 본인의 증명 과정을 새롭게 구성해가는 행위 역시도 튜링이 천재이기에 가능한 것이 아닐까 생각된다. 결론만 본다면, 다른 사람들도 도전해볼 수 있었겠지만, 실제 이를 생각하고 실행에 옮긴 것은 튜링이다. 이런 점을 보면, 튜링이 천재로 불리기에 부족한 점은 없다고 생각된다. 그러나, 우리는 단지 튜링은 천재구나 하며 넘어가는 것이 아니라, 그를 본받으며 창의적인 사고를 하는 것이 적절한 태도일 것임을 생각해볼 수 있었다.

튜링이 이렇듯 단순히 증명방법을 받아들이지 않고 새로운 아이디어로 다른 증명에 도전했기에 현재의 컴퓨터가 나오게 된 것이라고 생각한다. 그리고, 튜링이 이렇게 새로운 증명을 생각해낼 수 있었던 것은 튜링이 가지고 있는 수학적 베이스가 탄탄했기 때문이라고 추측된다. 물론, 튜링의 튜링 머신 설계도만 본다면 그리 어려워 보이지는 않는다. 그러나, 이를 위해서는 기계적인 계산 자체를 새롭게 정의해야 하며, 멈춤 문제를 풀 수 없음을 보이는 과정에서 칸토르의 대각화 논법과 같은 개념을 알아야 한다. 단순히 튜링 머신에 제한하지 않고, lambda calculus까지 보더라도 수학적 베이스가 매우 필요하다는 생각이 든다. 특히, 수리논리, 이산수학의 영역이 많이 필요할 것이다. 이처럼 컴퓨터공학을 계속해 나가기 위해서는 수학지식이 필수적임을 새삼스럽게 다시금 느낄 수 있었다. 두 번째 자료에서 앞으로 더욱 수리 논리와 창의성이 중요해질 것이라고 강조하는 것 역시도 같은 의미를 가질 것이다.

앞으로 컴퓨터공학 분야에서 발전할 주요 기술에 대한 논의 역시도 인상깊었다. 처음 컴퓨터가 등장한 이후로 컴파일러, 프로그래밍 언어, 타입 시스템, 자동화 등과 같은 많은 단계를 약 60년 동안 거쳐왔기에 우리는 지금 컴퓨터로 안전하고, 보안되는 소프트웨어를 생산할 수 있는 능력이 많이 향상되었다고 한다. 그러나, 한편으로는 사양대로 정확하게 작동하는 이상적인 소프트웨어의 완전성과는 점점 멀어지고 있다고 한다.

예시로, 요즘 가장 유명한 기술인 인공지능에 대한 설명이 있었다. AI는 기존의 기계가 할 수 있는 작업에서 더 발전해서 스스로 학습하며 인식 작업, 생각하는 활동을 할 수 있는 기술인 만큼 혁신이라고 할 수 있다. 그러나, AI는 블랙박스 이론, 우리는 입력 값에 대한 결과물은 알 수 있어도, 내부의 각 과정에서는 어떤 처리를 진행해서 어떤 상태인지 확인할 수 없다. 이로 인해서 AI는 노이즈에 취약하고, 누군가가 악의적으로 사용할 경우 막기 힘들다는 단점이 있다. 이로 인해서, 신뢰성에 관한 문제는 오히려 20년 전으로 퇴보하여 새로운 검증이 필요하다고 전한다. 결국, 우리는 기술이 발전되고 있다고 여기며 더더욱 기술의 발전을 원하지만, 현재의 발전은 완벽한 발전이 아닌, 일부는 퇴보하는 형태인 것으로 여겨진다. 즉, 일부를 포기하는 형태인데 포기하는 부분이 보안과 관련된 현재의 인공지능의 경우, 매우 위험하다고도 느껴진다. 따라서, 이렇듯 무리하게 기술의 발전을 고려하는 것이 꼭 좋은 것일지에 대해서도 생각해볼 필요가 있을 것 같다. 이 사실도 인공지능에만 한정된 것이 아닌, 양자컴퓨팅과 같이 새롭게 떠오르고 있는 기술들에 대해서도 마찬가지 얘기를 할 수 있을 것이다.

여러 읽을거리 들을 읽어보는 와중 나에게 잘 와 닿지 않는 점들도 있었다. 특히, 내가 지금 사

용하는 컴퓨터가 튜링 머신과 근본적으로 동일하다는 것이 그랬던 것 같다. 또, 질문하고 싶은 점에도 서술하겠지만, 컴퓨팅의 전 분야가 수학적 논리를 기초로 구성되어 왔다는 점이다. 그래도, 새롭게 알게 된 점들이 많고, 대학교 생활을 하는 동안 한 가지의 목표가 생긴 것 같아서 좋다. 전체적으로 기존에 프로그래밍 언어 분야에 가지고 있던 관심이 확대된 느낌인 것 같다. 앞으로도 이에 대한 공부를 계속 이어나갈 수 있는 방법을 찾아보고 싶다는 생각도 들었다. 전체적으로 이번 읽을거리를 읽으며 튜링과 컴퓨터공학, 그리고 수학의 관계에 대한 여러가지를 느낄 수 있었던 것 같다.

[질문하고 싶은 것]

이번 읽을거리들을 읽으면서 가장 궁금했던 의문은 "그렇다면, 내가 앞으로 컴퓨터공학을 계속 공부해 나가기 위해서 필요한 수학의 범위는 어디일까?" 라는 것이다. 현재 나는 컴퓨터공학 전공자로써 여러 분야들 중에도 보안과 인공지능이 결합된 보안인공지능에 관심을 가지고 있다. 보안 분야를 위해서는 정수론 분야를 시작으로 암호론, 선형대수학 분야에 대한 지식을 가지고 있어야 한다고 한다. 또, 인공지능 분야는 더더욱 수학과 밀접해서 선형대수학, 벡터 미적분학은 기본이고, 통계학 지식과 일정수준을 넘어간다면 대학원 수준의 수학과 통계학 지식 역시도 필요하다고 한다. 그러나, 이들 지식을 갖추어도 새로운 사실을 발견하고 만들어 내기 위해서는 추가적인 정의가 필요하고, 더 높은 차원의 지식을 가지고 있어야 할 것이다. 그리고, 내가 컴퓨터공학의 다른 분야에 관심을 가지게 된다면 또 더 많은 양의 수학 지식이 필요할 것이다. 그렇다면, 내가 앞으로 컴퓨터공학 분야에서 활동하기 위해 알아 두어야 할 수학지식은 어떤 것이며, 어디까지 일지 질문하고 싶어졌다.

두 번째 질문은 "수학적 논리와 컴퓨팅의 연관성이 얼마나 클까?" 라는 것이다. 교수님께서 저술하신 책과 강연에서는 컴퓨터의 발전이 수학적 논리로 이루어진다고 한다. 또, lambda calculus에 대해서 소개해주고 있는 논문 역시도 수학적 논리를 근거로 한 것이며, 마지막으로 프로그래밍 언어 분야와 컴퓨터공학에서의 발전을 다루고 있는 논문 역시도 수학 논리가 매우 중요함을 계속해서 언급하고 있다. 그러나, 이들이 잘 와닿지 않는 분야도 있다. 그래픽스, 비전, 인공지능, 보안과 같은 분야들에서는 각각 미적분학, 선형대수학, 통계학, 정수론과 같은 수학적 지식들이 필요하다. 그러나, 이 분야 들에서도 수학 논리가 중요하다고 얘기할 수 있을까? 물론, 프로그래밍 언어 분야에서는 수업시간에 배운 것처럼 수학 논리 그 자체로 구성되기도 하기에 그 중요성이 매우 크다고 할 수 있을 것이다. 그러나, 읽을거리에서는 이를 너무 성급하게 컴퓨팅의 전 분야로 확장시키는 것이 아닌가 라는 생각이 들었다.

마지막 질문은 "튜링 머신으로부터 혹은 lambda calculus로부터 고안된 컴퓨터는 완벽한 컴퓨터

일까?" 라는 것이다. 과연, 모든 기계적인 계산들을 믿을 수 있게 하는지, 또, 빠트리지 않고 할 수 있는지에 대한 의문이 든 것이다. 여기서 말하는 기계적인 계산들은 튜링의 정의와는 약간 다르게, 말 그대로 우리가 상상할 수 있는 모든 종류의 기계적인 계산들을 포괄하고 있다고 말하고 싶은 것이다. 물론, 정지문제와 같은 문제는 튜링 머신을 통해서는 해결할 수 없다는 것이 이미 증명되었다. 그러나, 과연 튜링 머신이 아닌 더욱 완벽한 컴퓨터라면 이 문제 역시도 해결할 수 있는 것이 아닐까? 라는 의문이 든다. 지금 다시 생각해보면, 튜링 머신과는 다른 방식으로, 현재 수학의 논리적 추론에서 출발한 lambda calculus 역시도 저 문제를 해결하지 못한다는 결론이 나온 것을 보면 매우 힘들거나 불가능한 부분일 수도 있을 것 같다. 그러나, 이는 아무도 답하지 못하는 것이기에 대학교, 더 나아가서 대학원에 가서도 이에 대해서 고민해보며 튜링처럼 졸업 전에 이에 대한 해답을 낼 수 있으면 좋을 것 같다.

만약, 튜링 머신이 완벽한 컴퓨터가 아니라면 컴퓨터는 어디까지 더 발전할 수 있을까. 이 질문을 마지막으로 이번 레포트를 마무리하려고 한다. 이 역시도 매우 궁금하다. 한가지 예시를 들어보자면, 컴퓨터를 사람의 뇌가 하는 모든 기능을 할 수 있도록 만들 수 있을까? 앞으로 기술이 점차 발전해가면서 이에 대한 해답이 나오겠지만, 현재로서 이를 상상해보는 것도 매우 재미있는 주제라고 생각된다. 이번 읽을거리들을 읽다 보니 이런 창의력들이 컴퓨터를 발전시켜 나가는 주된 요인이라고 느껴졌다.