[컴퓨터과학이 여는 세계_Homework 2]

# 최고가 아니어도 좋아!

자유전공학부 심지원

## II. 내가 느낀 것, 그리고 상상하게 된 것

컴퓨터가 전기 스위치를 통해 실현되는 과정을 보며 융합의 중요성을 느낄 수 있었다. 부울 논리와 스위치에 대한 기본 지식을 모두 알고 있다면 부울의 세 가지 접속사가 스위치와 대응된다는 사실은 누구나 발견할 수 있는 것이었다. 이러한 발견을 통해 주먹구구식으로 활용되던 스위치 분야에 이해를 통한 체계가 불어넣어 졌으며, 발전을 통해 복잡해 보이는 컴퓨터를 구현하는 것까지 성공했다. 전혀 다른 세계라 생각되는 논리학과 스위치 분야에 대한 지식이 융합되면서 컴퓨터라는 전혀 새로운 기계가 탄생한 것이다. 클로드 섀넌이 부울 논리와 스위치를 접목시킨 것처럼 학문들의 융합은 그 동안 생각하지 못했던 획기적인 아이디어의 탄생을 가능하게 할 것이다. 그 융합의 시작이 거창할 필요는 전혀 없다. 직렬로 연결된 스위치를 보며 '그리고' 조합을 떠올릴 수 있는 찰나의 스파크만 필요한 것이다. 이를 위해서는 다양한 분야에서 배경지식을 갖추고 있어야 한다. 논리학의 전문가여도 스위치에 대한 지식이 전혀 없다면 위 발견은 불가능했을 것이기 때문이다. 한 분야만을 파는 specialist도 물론 중요하다. 하지만 다방면에 능숙한 generalist에게 융합의 스파크가 튈 가능성이 더 높다고 생각된다. 우리 사회에서 generalist의 가치가 인정받기 시작한 지 몇 년이 흘렀다. 앞으로 전혀 관계없어 보이는 분야들의 융합으로부터 어떠한 혁신적인 결과들이 탄생할지 벌써부터 기대가 된다.

하지만 이렇게 구현된 컴퓨터의 효율이 점점 높아지는 모습을 보며 두렵기도 하다. 1970년대 수천 개의 스위치가 약 0.5와트로 구동되었던 것과 비교해 지금은 약 $10^6$배 많은 스위치를 약 80와트로 구동할 수 있다고 한다.[1] 그리고 그 효율을 높이려는 움직임은 지금도 매우 활발하다. 양자 컴퓨터가 그 예다. 앞으로 컴퓨터의 효율이 더욱 높아질 것임을 예상가능하다. 하지만 '효율을 높이는 것이 꼭 좋은가'라는 질문을 던지게 된다. 양자 컴퓨터를 통해 속도가 빨라지고 메모리 소모가 줄어 몇몇 NP문제들이 다항 시간 안에 가능하게 된 것을 보면, 앞으로 효율이 계속해서 높아진다면 언젠가는 P=NP의 시대가 도래 할 가능성을 배제하지 못한다.[2] 책에서는 이를 "명작을 현실적인 비용으로 컴퓨터가 자동 생산"[3]할 수 있는 세상으로 비유한다. 명작이 기계적으로 만들어지는 세상을 상상해보자. 화가, 음악가, 작가, 시인이 사라지고 기계가 예술의 영역을 독점할 것이다. 나에게 P=NP의 세상은 재미없는 세상이 될 것만 같다. 인간의 고유기능이 점차 컴퓨터에게 위협받으며, '어차피 컴퓨터가 해줄거야'라며 점점 더 컴퓨터에게만 의지하는, 창작과 문제해결을 위한 인간의 의욕과 노력이 점차 사라지는 그런 재미없는 세상이 말이다. 이러한 두려움에 컴퓨터 효율이 눈에 띄게 향상되는 것을 마냥 두 팔 벌려 기다릴 수 없다.

두려움 속에서 나는 소프트웨어를 궁리하고 만드는 사람들이 조금 더 느림의 미학을 추구했으면 한다. 소프트웨어는 컴퓨터를 만능이게 하지만 그 소프트웨어를 짜는 것은 결국

---

1) Ibid.p.89(각주7).
2) 물론 아직까지 P≠NP임이 밝혀지지는 않았지만, 현재는 P≠NP임을 전제한다.
3) Ibid.p.114.

인간이다. 따라서 사람들은 컴퓨터의 효율을 높이는 것도 중요하지만 소프트웨어가 인간에게 유용할 뿐, 위협이 되지 않도록 주의하는 것이 더욱 중요하다. "컴퓨터는 소프트웨어 그대로를 무심히 실행에 옮길 뿐, 빈틈이 있다면 고스란히 드러낸다"[4]는 말에서 나는 '무심히 실행에 옮길 뿐'이라는 구절을 그냥 지나칠 수가 없다. 컴퓨터는 소프트에어를 실행함에 있어 그 결과에 대한 고려를 전혀 하지 않는다. 그냥 주어진 일을 '생각 없이' 실행할 뿐이다. 어떤 윤리적인 고려도, 실행결과로 파급될 수 있는 문제들에 대한 우려도 없다. 따라서 복잡한 소프트웨어를 짜는 과정에서 발생한 사소한 작은 실수가 원하는 결과 대신 전혀 예상하지 못한 피해를 가져올 수 있다. 그렇기 때문에 나는 소프트웨어를 만드는 사람들은 느림의 미학을 추구하며 최대한 실수를 줄이기 위한 노력을 해야 한다고 생각한다. 소프트웨어 관련 분야에서 최적화 프로그램을 만들려는 일련의 노력들이 계속되고 있다. 물론, 완벽한 최적화 프로그램이 가능하다면 더 빠르고 쉽게 결과를 얻을 수 있어 긍정적으로 평가된다. 예컨대 수백 개의 부품들로 이루어진 정수식 중 0을 곱하는 부품이 하나라도 있다면 수백 번의 번역과정 없이 최적화 컴파일러를 통해 '0'이라는 결과를 바로 얻을 수 있기 때문이다. 하지만 우리 주변의 많은 소프트웨어들은 그 글자 수가 웬만한 대하소설보다 많고, 웬만한 포유류 뇌 속 뉴런들 연결 관계만큼 복잡한 논리 흐름의 복잡도를 가지고 있다고 한다.[5] 이렇게 복잡한 소프트웨어를 짜는 데 실수는 당연히 존재할 수밖에 없는데, 소프트웨어를 최적화하는 것은 실수의 위험을 더욱 높이는 길이라 생각된다. 빠르게 처리하지 않아도 된다. 가끔은 천천히 정석대로 가는 게 좋을 수 있다. 위험이 클 때 특히 그렇다. 무서운 소프트웨어를 짤 때 결코 성급해서는 안 된다고 생각한다.

점점 더 '최고의 컴퓨터'에 근접해가는 듯 한 컴퓨터에게 '최고가 아니어도 좋아!'라는 말을 하고 싶다. 효율이 높아짐에 따라 점점 다항 시간 안에 답을 찾을 수 있는 문제들의 영역이 확대되고, 프로그램이 잘 돌아가는 지 '어느 정도' 자동으로 검산할 수 있는 기술도 만들어졌다. 어쩌면 튜링의 '기계적인 계산'을 벗어난 일을 할 수 있는 새로운 개념의 컴퓨터가 만들어질 수도 있다고 생각한다. 컴퓨터 분야의 발전의 끝이 어디인지는 아무도 알 수가 없는 것이다. 하지만 '최고의 컴퓨터'를 향해 점점 발전해가는 것을 보며 컴퓨터가 인간의 능력을 뛰어넘는 날이 올 수 있다는 생각이 엄습한다. 최고의 컴퓨터 대신 안전한 컴퓨터를 원한다. 소프트웨어로 인간의 기능까지 위협받는 세상이 만들어지지 않기를 바란다. 인간이 컴퓨터에게 지배받는 세상이 온다면 '최고'를 향한 지금의 거침없는 질주를 후회하게 될 것만 같다.

---

4) Ibid. p.93.
5) Ibid. p.135.