컴퓨터과학이 여는 세계

김서연 기계공학부 2021-18457
박효진 국어교육과 2021-16649

이미지 출처: 미리캔버스

Ⅰ. 원천 설계

컴퓨터가 어떻게 시작된 것인지 기대하고 수업을 들었는데, 갑자기 수학계의 이야기가 나와서 당황했던 기억이 난다. 컴퓨터의 시작은 수학자 대회에서 수학적 명제들을 기계적으로 판명하는 기계에 대한 존재 여부를 고민한 것이었다. 컴퓨터와 비슷한 장치도 존재하지 않을 시절, 명제들의 참과 거짓을 기계적으로 판단하는 장치를 고안한 수학자들의 창의력이 놀라웠다. 나에게 명제의 증명이란 그저 여태껏 쌓아온 수학적 법칙들을 활용하여 하나하나 헤쳐나가야 하는 장애물이라고 생각했는데, 이것을 기계적으로 증명하는 것에 대해 떠올린 게 정말 놀라웠다. 하지만 이 가상의 기계는 괴델의 불완전성 정리에 의해 깨져버리게 된다. 수업을 들으면서 가장 이해하지 못했던 부분이 이 괴델의 불완전성 정리에 대한 것이었는데, 괴델은 과연 튜링과 다르게 어떤 방식으로 증명을 진행했는지 항상 궁금했었다. 그래서 직접 괴델의 불완전성 정리 증명을 공부해 보았고, 내가 이해한 방식대로 증명을 서술하면 다음과 같다. 괴델의 불완전성 원리는 간단히 말하면 증명이 불가능하지만 참인 명제가 존재한다는 것이다.

먼저 각 명제에 해당하는 괴델수라는 것을 정의할 수 있다. 이 괴델수는 특정 코드에 따라 문자를 수로 변환시켜 곱해 나타낼 수 있다. 만약, '나'=2, '꽃'=3, '좋아한다'=7이라고 둔다면, '나는 꽃을 좋아한다'는 명제의 괴델수는 2*3*7=42으로 표현이 가능할 것이다. 그리고, Sub라는 기호에 대해 이해할 필요가 있다. Sub는 '대체하다'라는 의미를 가진 기호이고, 세 개의 입력을 받는 함수이다. 방금 선언한 문장의 괴델수가 42라는 것을 기억하자. 이때 내가 '꽃'이 아닌 '강아지'로 이 명제를 대체한

다면, 그 대체된 명제는 Sub(42,3,'강아지')라고 표현할 수 있다. 42는 기존의 42의 괴델수를 가지는 명제이고, 3은 그 명제의 변수 (꽃)이며, 강아지는 그 변수를 대체하는 것을 선언한다. 만약, Sub(42,3,42)라는 선언은 어떤 뜻일까? 방금 방식대로 이해하면, '나는 (나는 꽃을 좋아한다)를 좋아한다' 라는 명제를 나타내게 된다. 한글로 쓰면 이상하더라도 수학적으로는 오류가 없다.

이제 실제 증명을 다루고자 한다. 먼저 코드에서 y=17이라고 두었을 때, Sub(x,17,x)는 '괴델수 x를 가진 문장'에서 '변수 y'를 '괴델수 x'로 대체한 명제를 나타낸다. 따옴표를 통해 표현한 세 가지가 각각 Sub의 세 가지 입력이라고 보면 되겠다. 그리고 이 Sub(x,17,x)의 괴델수는 sub(x,17,x)라고 표현하자.

이제 2개의 명제를 가지고 증명을 하게 된다.

첫 번째 명제 P : Sub(y,17,y)는 증명이 불가능하다.
이때 P의 괴델수는 n이라고 한다.
두 번째 명제 G : Sub(n,17,n)은 증명이 불가능하다.
이때 G의 괴델수는 g라고 한다.

이때 두 번째 명제 G의 괴델수 g는, 괴델수 n을 가진 문장(P)에서 y를 괴델수 n로 대체한 명제의 괴델수를 나타낸다. P에서 y를 n으로 바꾸면 G가 되는 것을 알기에 이것이 성립한다.

이번엔, 명제 G에 속한 Sub(n,17,n)에 대해서 집중해보자. 이 기호는 괴델수 n을 가지는 문장에서 y를 n으로 대체한 명제를 의미한다. 그런데, 이것은 앞서 구했던 괴델수 g와 의미가 상통하는 것을 알 수 있다.

괴델수 g -〉 괴델수 n을 가진 문장(P)에서 y를 괴델수 n로 대체한 명제의 괴델수
Sub(n,17,n) -〉 괴델수 n을 가지는 문장에서 y를 n으로 대체한 명제

그렇다면, Sub(n,17,n)이라는 명제의 괴델수가 곧 g임을 알 수가 있다. 이제 다시 명제 G로 돌아가자.

두 번째 명제 G : Sub(n,17,n)은 증명이 불가능하다.
G의 괴델수는 g이다.

즉, 명제 G는 "괴델수 g를 가지는 명제가 증명이 불가능하다"는 명제인데, 이 G의 괴델수가 g이다. 즉 자기 자신에 대해 증명이 불가능하다는 것을 나타내는 명제가 된다는 것이다! 따라서 참이면서 증명이 불가능한 명제가 존재한다고 이해하였다. 그러므로 기계적으로 판정이 불가능한 참인 명제가 존재함을 이해할 수 있다.

이해하기도 어렵고 표현하기도 어렵지만, 추가적으로 공부하면서 2번째 명제의 괴델수도 g이고 Sub(n,17,n)의 괴델수도 g인 것을 보았을 때 엄청나게 놀랐다. 놀라운 증명을 배우고 직접 해보았을 때의 찌릿함을 오랜만에 느꼈던 것 같다. 괴델이 증명 방법을 대체 어떻게 떠올렸으며 어떻게 논문으로 표현했을지도 정말 궁금해졌다.

여하튼 괴델의 증명 방식을 뉴먼 교수에 의해 튜링이 배우게 되었고, 튜링은 그 강의를 듣고 본인만의 증명 방법을 새로 고

안하게 되었다. 사실 이 대목에서 스스로 반성하게 되었는데, 튜링은 강의를 들으면서 강의 내용을 암기하고 외우는 것이 아니라 거기에 의문을 던지고 계속 생각했다는 것을 알았기 때문이다. 나는 여태껏 배운 내용들에 대해 질문을 던지고 의구심을 품은 적이 있는가 떠올려 보았을 때, 부끄럽지만 아직 없었다. 아마도 우리가 괴델을 단순한 천재라고 추앙하지만, 사실 이러한 작은 차이가 모여서 그를 돋보이게 만든 것은 아닐까하는 생각이 들었다.

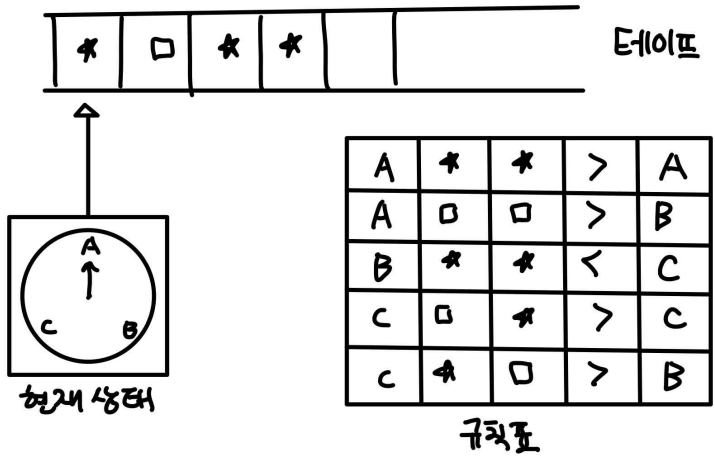
튜링 기계

튜링 기계를 배우기에 앞서 교수님이 1번 과제로 '기계적인 것'에 대한 정의를 에세이로 써오라고 하신 기억이 난다. 사실 그때 길이 보이지 않아 많이 난감했지만 교수님이 분명 이 에세이를 통해 의도하는 바가 있으실 것이라고 생각하여 열심히 썼다. 그리고 기계적인 것을 튜링이 어떻게 정의했을지도 매우 궁금했으니 말이다. 나는 기억하기로 '입력되는 모든 상황에 적용될 수 있으며 해당 업무나 일이 해결되거나 완료될 수 있는 어떤 규칙이 존재하는 일'이라고 나만의 정의를 내렸는데, 그랬던 내게 튜링의 정의 방식은 혁신처럼 다가왔다. 튜링 기계를 배우고 난 지금은 그때의 내가 내린 정의가 얼마나 모호하고 추상적인지를 깨닫고 부끄러워하고 있다.

튜링은 규칙표, 테이프, 현재 상태를 표시하는 기계인 튜링 기계으로 실행할 수 있는 모든 계산을 기계적인 계산이라고 정의했다. 대체 어떻게 아무것도 없던 상태에서 튜링이 이 기계를 떠올린 것인지가 제일 궁금한 점이었다. 그리고 아직도 이 정의가 깨지지 않았다는 것도 아주 놀라웠다. 매우 간단명료하며 정

확한 정의인데, 지금까지의 기계적 계산을 모두 이 정의로 담을 수 있다는 게 아직도 놀랍다.

튜링 기계는 긴 테이프에 적힌 기호를 읽고, 현재 상태와 읽은 기호에 따라 규칙표가 가리키는 행동을 한다. 이 과정을 반복하다 보면 목적하는 계산을 진행할 수 있다.

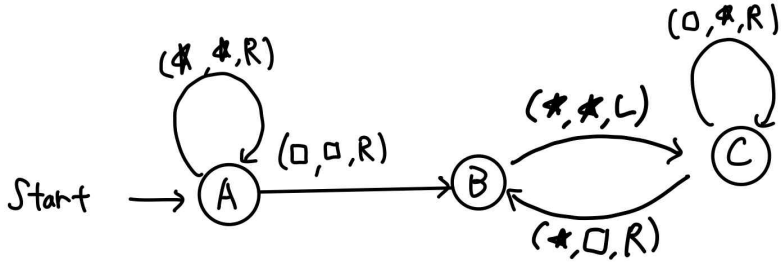


위 사진과 같이 튜링 머신을 구성할 수 있다. 규칙표의 각 열은 각각 현재 상태 / 읽은 기호 / 쓸 기호 / 이동 방향 / 다음 상태 로 나타내진다. 현재 상태와 읽은 기호에 따라, 쓸 기호와 이동 방향, 다음 상태가 정해지게 된다. 규칙표에 해당하는 현재 상태 + 읽은 기호의 조합이 없다면 실행이 끝나게 되는 것이다.

해당 사진에 제시된 튜링 머신을 실행해보자. 처음에는 ☆와 A를 읽는 상태이므로, ☆를 쓰고 오른쪽으로 이동한다. 현재 상태는 그대로 A로 유지가 된다. 이런 방식으로 계산을 실행하면,

☆□☆☆ -〉 ☆☆☆□ 로 □ 기호가 맨 오른쪽으로 옮겨지게 된다.

수업을 들으며 간단한 더하기, 곱하기 등을 하는 튜링 기계를 만들어 보았고 복사하기나 크기 비교를 하는 기계들도 만들어 보았다. 직접 규칙표를 만들며 느낀 점은, 우리가 생각하기에 매우 간단해보이는 작업들도 튜링 기계로 구현하려면 매우 복잡해진다는 것이었다. 단순한 크기 비교도 규칙표의 길이가 꽤 되었고, 생각하는 데 시간도 꽤 걸렸기 때문이다. 그렇다면 과연 복잡한 작업의 튜링 머신은 어떻게 프로그래밍할 수 있었을지 궁금했다. 또한 프로그래밍한 튜링 기계의 구동 원리를 설명할 수 있는 방법이 무엇일지 궁금하여 찾아보았다. 찾아보니 규칙표를 아래 그림처럼 표현하는 것을 알 수 있었다.
원래는 규칙표에 5개의 열이 배치되어 있는데, 해당 그림에서



현재 상태와 다음 상태에 쓰일 수 있는 심볼 (A,B,C)이 그냥 배치되어 있다. 그리고 나머지 3개의 열 (읽은 기호, 쓸 기호, 이동 방향)의 요소들이 화살표를 이용하여 표시가 된다. 저 방식대로 표현하면 규칙표를 다른 사람에게 보여주는 것보다 훨씬 알아보기 쉽다고 생각했다.

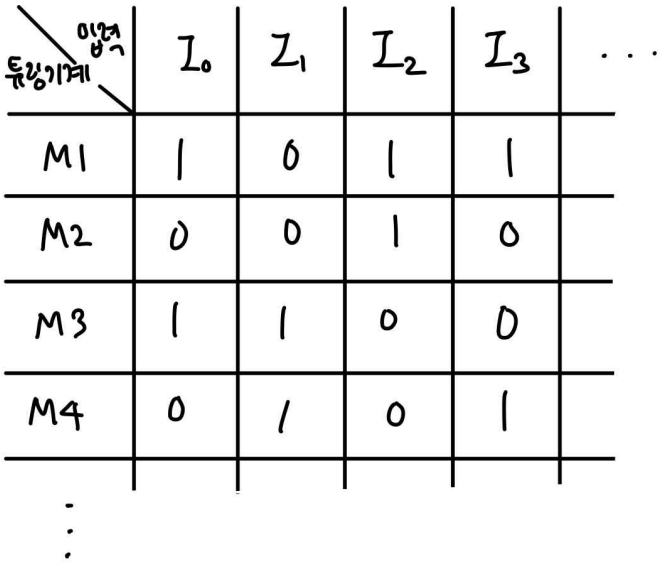
튜링 기계를 통한 증명

증명의 출발은, 어떤 튜링 기계를 입력으로 받아 작동시키는 '보편만능의 기계'를 만드는 것이다. 이 보편만능의 기계는 임의의 튜링 기계를 테이프에 받아 실행시킬 수 있다.

튜링 기계의 개수는, 자연수의 개수와 똑같다고 할 수 있다. 튜링 기계의 개수는 튜링 기계를 만들 때 사용하는 부품과 기호들의 조합의 개수만큼 존재할 수 있다. 만약 기호들이 N개 필요하다면, 튜링 기계를 N개의 기호를 표현하여 나타낸 N진법의 수로 생각할 수 있다. 따라서 N진법의 수에 튜링 기계를 대응시킨다면 각각 다른 기계를 의미하고, 이 N진법의 수는 자연수의 개수와 같으므로 튜링 기계는 자연수의 개수만큼 존재한다고 볼 수 있다.

튜링은 튜링 기계로 풀 수 없는 문제를 제시하는데, 그것은 '멈춤 문제'라고 불리는 문제이다. 멈춤 문제란 어떤 튜링 머신에 어떤 입력값이 주어졌을 때 이 튜링 기계가 실행을 멈출 것인지 계속할 것인지를 판단하는 문제이다. 만약 입력값 I에 대해 M1이라는 튜링 기계가 멈춘다면 결과를 1, 멈추지 않는다면 0이라고 표시한다고 하자. 만약 멈춤 문제를 풀 수 있는 튜링 기계가 있다면, 존재하는 모든 튜링 기계를 일렬로 줄 세운 뒤에 여러의 입력값에 대해 멈추는지를 0과 1을 이용한 표로 표현할 수 있다. 그러나, 여기서 모순이 발생한다. 증명의 가정으로 모든 튜링 기계를 줄세운다고 했는데, 이 줄세운 기계들과는 또 다른 튜링 기계가 존재한다는 것이다.

여기서 사용하는 증명 방식이 칸토어의 대각선 논법과 유사하다. 칸토어의 대각선 논법은 자연수와 실수라는 두 무한 집합의

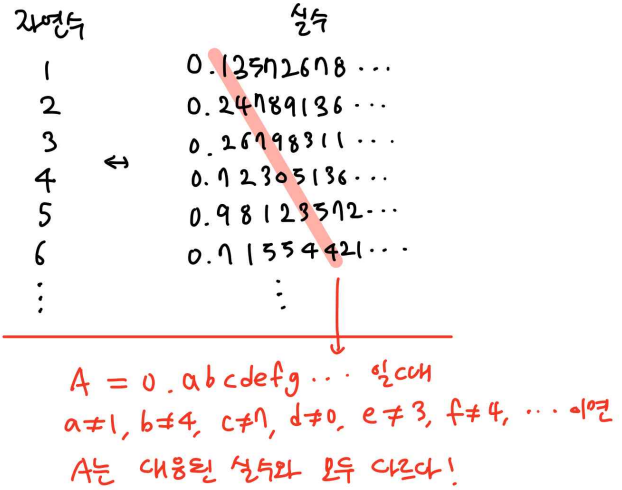


크기를 비교할 때 사용하는 방식이다. 칸토어의 대각선 논법을 먼저 알아보자.

*** 칸토어의 대각선 논법**

과거 집합의 크기를 비교하는 방법은 서로 원소들을 대응시키며, 더 이상 대응시킬 원소가 없을 때 해당 집합의 크기가 작다고 판단했다. 그러나, 자연수의 집합과 실수의 집합을 비교한다면 둘 다 무한의 크기를 가지는 집합이므로 간단한 대응이 힘들다. 먼저 자연수와 실수를 하나씩 대응시키자 (실수는 소수 형태로 표현된다). 자연수 n에 대해 대응되는 실수를 $a_n = 0.a_{n1}a_{n2}a_{n3}\cdots$이라고 한다. 만약 두 집합의 크기가 같다면 모든 자연수와 실수가 대응이 되어야 한다. 하지만, 칸토어는

자연수와 대응된 실수들과는 모두 다른 또 다른 실수가 있음을 증명했다. 이 실수는 a_1의 소수 첫째 자리 숫자와 다르고, a_2와는 소수 둘째 자리 숫자가 다르고, 이것이 반복되어 a_n의 소수 n번째 자리 숫자와 다르다는 규칙을 가진다. 그렇다면, 이 새로운 실수는 대응되어 있는 모든 실수와 적어도 하나의 다른 숫자가 항상 있기 때문에 결론적으로 같지 않다. 따라서 새로운 실수가 만들어지므로 자연수에 비해 실수 집합의 크기가 크다고 할 수 있다.



튜링의 증명도 비슷한 방법으로 전개된다. 저렇게 모든 튜링기계를 입력값들에 대해 줄 세우고 1이나 0으로 표시한다고 했을 때, 대각선에 배열된 숫자들과는 모두 다른 입력값을 가지는 또 다른 튜링 기계 D를 만들어 낼 수 있다.

따라서 증명의 전제였던 '모든 튜링 기계를 줄세운다'라는 전

제가 틀렸으므로, 멈춤 문제를 풀 수 있는 튜링 기계가 존재한다는 것도 옳지 않다. 따라서 튜링 기계로 멈춤 문제를 풀 수 없다.

만약 모든 참인 명제를 만들어내는 기계가 존재한다면, 멈춤 문제를 푸는 기계도 존재해야 한다. 그러나 멈춤 문제를 푸는 기계가 없으므로, 모든 참인 명제를 만들어내는 기계도 없다는 것이 증명되었다.

이 과정에서 이용된 '보편만능의 기계'가 컴퓨터의 시초가 되었다. 여러 개의 튜링 기계를 입력으로 받아 작동시키는 보편만능의 기계에서, 입력으로 넣는 튜링 기계는 소프트웨어이고 이것을 실행하는 튜링 기계가 하드웨어가 된다. 우리가 컴퓨터로

여러 가지 소프트웨어를 실행하는 것이 이 보편만능의 기계의 디자인과 연결된다.

단락을 마치며

튜링이 컴퓨터의 시초를 고안했지만, 컴퓨터를 만들기 위한 것은 아니었다는 것이 인상적이었다. 다르게 말하면 컴퓨터라는 기계가 지금의 컴퓨터와 큰 관련이 없어 보이는 명제를 해결하는 데서 시작했다는 것이 신기했다.

배우면서 든 가장 큰 질문은, 과연 튜링은 이 모든 것을 염두에 두고 튜링 기계를 구상한 것일까였다. 튜링 기계를 자연수에 연결하는 것부터 칸토어의 대각선 논법을 가져와 증명을 하는 것까지 마치 아름다운 퍼즐처럼 잘 들어맞는다는 생각을 하였다. 그러나 내가 이것을 배운 것은 이미 모든 증명이 끝난 뒤였지만, 증명이 아무것도 되지 않은 상태에서 기계적 계산을 정의하고 - 멈춤 문제를 생각하고 - 대각선 논법으로 증명하는 것은 도저히 상상이 가지 않는다. 기존의 증명에서 벗어나 본인만의 새로운 정의를 완벽히 만들고, 깔끔한 증명까지 이루어냈다는 점에서는 튜링이 정말 천재가 아니었을까 하는 생각도 들었다.

이렇게 증명 과정을 정렬하여 써보니, 컴퓨터라는 기계가 등장하기까지의 과정이 매우 운명적이었다고 생각한다. 수리명제 자동 판결 문제 -〉 괴델의 증명 -〉 튜링의 튜링 기계를 이용한 증명 -〉 보편 만능의 기계에서 시작된 컴퓨터의 등장까지, 이 과정에서 한 과정이라도 없었다면 컴퓨터란 기계가 지금 세상에 없을지도 모른다. 예전에 '페르마의 마지막 정리'를 매우 인상깊게 읽었는데, 그 기억도 다시 떠올랐다. 페르마의 정리라는 명제를 증명하기 위해 여러 수학자들의 많은 아이디어들이 연결되

고 연결되어 결국에 증명이 완성된다. 비록 아이디어들이 관련이 없는 것 같았지만 최종적으로는 아름답게 연결되어 증명을 구성한다. 다른 사람들이 이미 만들어낸 여러 아이디어들을 연결지어 응용하는 능력이 매우 중요하고 새로운 발견을 하는 근간이 된다는 것을 깨달았다.

사실은 이 강의를 수강하고 이 내용들을 배우기 전까지, 컴퓨터라는 기계에 대해 자세하게 고민해 본 적이 없었다. 코딩이나 프로그래밍 언어들을 배우기는 했지만, 컴퓨터라는 기계 자체가 어떤 원리로 구동되고 어디서 시작되었는지는 떠올려 본 적이 없었다. 컴퓨터라는 기계를 그저 당연한 존재로 인식하고 그에 대한 고찰을 하지 않았던 것이다. 하지만 원천 설계에 대해 배우고 나니 이렇게 복잡해 보이던 기계도 이해할 수 있는 아이디어에서 출발했다는 것을 새롭게 배웠다. 그동안 컴퓨터를 매일 사용하면서도 한 번도 고민해보지 못했던 나에 대한 반성도 할 수 있었다. 이 경험을 바탕으로 다른 기계들은 또 어떤 아이디어에서 출발했고 어떤 원리로 작동하는지를 찾아보고 있다.

처음에는 이 강의의 제목인 '컴퓨터과학이 여는 세계'가 어떤 의미인지 잘 와닿지 않았었다. 하지만 이제는, 이 강의가 나에게 컴퓨터에 대한 지식과 더불어서 더 넓은 시야와 자세로 바라보는 세계를 열어주었음을 깨닫는다.

Ⅱ. 가르침만큼 가혹한 가르쳐짐은 없다 - 나 자신만큼 가혹한 선생님은 없다, 그리고 든든한 동료 또한 없다.

2021년 9월 26일, 박효진은 자신이 수강하고 있는 교양과목 '컴퓨터과학이 여는 세계'에서 두 개의 숙제를 얻게 된다. 숙제를 푸는 과정에서 박효진의 자아는 분리되고, 어떻게든 문제를 풀기 위해 두 자아는 서로 머리를 맞대기 시작하는데...

> Exercise 1 [50점] "2배-튜링기계". 자연수(〉 0)를 테입에 받아서 2를 곱한 자연수를 테입에 쓰는 튜링기계를 만들라. 자연수는 1진법으로 테입에 표현된다(1은 ·, 2는 ··, 3은 · · ·등). 헤더의 초기 상태는 테입에 입력된 자연수의 시작위치로 하고 시작 위치와 끝 위치는 ★로 표시된다. 예를들어:
> ★ · · ★
> 위와 같은 입력이 테입에 있을때, 작동을 마치면 테입은 아래과 같이 된다:
> ★ · · ★ · · · · ✷

박효진 A: 음...이게 내가 풀어야하는 과제야? 그냥 포기하고 싶은데?
박효진 B: 그런 소리 하지 마, 일단 좀 진정해보고, 차분히 문제를 읽어보자고, 음...일단 첫 번째 문제부터 풀어볼까.
박효진 A: 자연수를 받아서 2를 곱한 자연수를 내놓으래...아 그냥 머리로 계산하면 안되는 거야?
박효진 B: 너 1342204283029도 그렇게 계산할래? 시간이 걸려도 기계가 풀게 놔두는 게 낫지. 일단 침착해보자...곱하기는

곱하기로 해석할 수도, 더하기로 해석할 수도 있는데, 이 문제에서는 어떻게 해석하는 게 좋을까?
박효진 A: 더하기가 더 간단한 방법이니까 더하기가 낫지 않을까?
박효진 B: 그럴까...일단 맨 처음에 별을 찍어야 한다는 사실은 변하지 않지, 근데 우리는 상태를 A,B,C로 배웠으니까...헷갈리네, '설정'칸에서 '시작 상태'를 A로 바꾸고, 심볼도 0과 1로 바꿔서 해석해보자.
박효진 A: 음...시작 상태는 이해하지만 심볼은 큰 의미 없을 것 같은데?
박효진 B: 그런가...그래도 저번 강의에서 심볼을 $로 바꿔서 푼 분이 있었던 게 기억나.
박효진 A: 그 기억이 어떤 도움이 될지 생각해보자...일단 입력을 둘 다 11로 바꾸든 $$로 바꾸든 그건 의미가 없을 거야, 그렇다면 $는 1과 구분되어야겠지?
박효진 B: 그렇다면 1을 $로 치환하고, $를 1 2개로 치환하는 과정을 차례차례 거친다면 어떨까?
박효진 A: 좋아, 1을 $로 바꾸는 과정을 생각해보면...잠깐, 그러면 무조건 모든 1이 $로 바뀌게 되잖아, 아무 의미가 없는데?
박효진 B: 어 그러네, 어떡하지? 처음부터 1을 $로 바꾸는 건 도돌이표와 같을 텐데...
박효진 A: 처음부터 1을 $로 바꾸는 게 안된다면...마지막부터 1을 $로 바꾸는 건 어떨까?

박효진 B: 마지막은 어떻게 확인...아 0이 있으니까?
박효진 A; 그래, 0을 기준으로, 0을 만나면 다시 돌아오는 거야, 방향은 왼쪽, 고정, 오른쪽이 있었으니까 충분히 가능해.
박효진 B: 좋아, 그러면 0을 만나면 다시 직전의 1로 돌아와. 그리고 1을 $로 바꿔, 그 다음엔...어디로 가야 하지? 왼쪽? 오른쪽?
박효진 A: 잠시만...1을 $로 바꾸고 왼쪽으로 가면, 다시 원래의 1을 만나게 돼, 그럼 계속 도돌이표인걸?
박효진 B: 그러네, 그럼 오른쪽으로 가서 0의 오른쪽에 1 두 개를 적어야 하겠네, 입력이 자연수라면 무조건 입력 숫자는 1 이상일테고, 그러면 자연스럽게 출력 숫자는 1이 2개 이상일 테니까.
박효진 A: 그래, 그렇게 1 2개를 적고 다시 돌아온 다음...$를 1로 바꿔, 왜냐하면 결과물은 0과 1밖에 없는 이진수 형태여야 하니까.
박효진 B: 그래, 그리고...다시 $로 변하지 않았던 1을 $로 바꿔줘.
박효진 A: 좋아...이 과정을 반복한 다음에...0을 만나면 멈춘다! 완성이야!
박효진 B: 좋아, 이걸 코드로 정리해보면 아래와 같겠군.

```
A 0 0 r B (맨 처음의 A 상태, 오른쪽으로 가면서 수를 읽어)
B 1 1 r B (B 상태, 계속 1이라는 숫자를 읽어가)
B 0 0 l C (어? 0을 만났네? 상태를 C로 바꾸고 잠시 멈춰)
C 1 $ r D (전으로 돌아간 뒤에 1을 $ 기호로 바꿔줘)
```

D 0 0 r E (다시 오른쪽으로 이동해)
E _ 1 r F (빈 칸에 1을 그려줘)
F _ 1 * G (다시, 빈 칸에 1을 그려줘)
G 1 1 l G (자, 이제 하나의 1을 두배로 만들었으니 돌아가자)
G 0 0 l G (0도 패스, 중간의 0이니까 바꿀 필요 없어)
G $ 1 l H ($는 1로 바꿔줄 차례야)
H 1 $ r I (이 과정을 반복해주는 거지)
H 0 0 * J
J 1 1 * J
I 1 1 r I
I 0 0 r I
I _ 1 r F

박효진 B: 좋았어! 튜링 기계를 가상으로 돌려봐도 잘 돌아가는군. 이제 다음 문제를 풀 차례야!
(http://morphett.info/turing/turing.html)

Exercise 2 [50점] “비교-튜링기계” 두 자연수(〉 0)를 테입에 받아서 두 수가 같으면 1을, 아니면 0을 테입에 쓰는 튜링기계를 만들라. 자연수는 1진법으로 테입에 표현된다(1은 ·, 2는 ··, 3은 · · ·등). 헤더의 초기 상태는
테입에 입력된 두 자연수의 시작위치로 하고, 두 자연수는 빈칸 없이 시작 위치와 끝 위치가 하나의 ★로 표시된다고 가정한다. 예를들어:
★ · · ★ · · · ★
위와 같은 입력이 테입에 있을때, 작동을 마치면 테입은 아래과 같이 된다:
★ · · ★ · · · ★ 0

박효진 A: 아 젠장 아직 안 끝난거야?
박효진 B: 진정해, 위에 문제도 잘 풀었잖아.
박효진 A: 앞에 문제랑 이거랑 같아? 앞 문제는 '출력'을 해야 했지만 이 문제는 '판단'을 해야 한다고! 심지어 그 결과를 출력해야 돼!
박효진 B: 그 점이 어렵지...우리는, 그러니까 인간은 보통 대소 비교를 할 때 어떻게 하지?
박효진 A: 내가 수학학원에서 배웠을 때는 앞에 수에서 뒤에 수를 빼거나, 양변을 하나의 수로 나눠서 1과 비교하기도 했지, 일단 전제는 두 수 모두 자연수니까, 그 점은 정말 다행이야.
박효진 B: 나누기는 지금 내 수준에선 감이 안 잡히고...빼기로 해볼까? 앞에 수에서 뒤의 수를 빼는 건?
박효진 A: 천잰데? 아니 잠깐만...뒤에 수가 앞에 수보다 클 거라는 얘기는 아무데도 없는데? 우리가 음수를 튜링 기계로 표현할 수 있을까?
박효진 B: 아 이런...어떡하지? 그러면 정말 단순히 '비교'를 해야 되는 건가? 얘한테는 '수'라는 개념이 사실 없는 거나 마찬가지잖아, 어떡하지?
박효진 A: 전에 했던 걸 참고해 보자...내 21년 입시인생 특성상 문제의 순서에는 반드시 이유가 있었어.
박효진 B: (랜덤으로 문제 나왔던 저번 교양강의는 잊어버렸나....?)
박효진 A: 우리는 저번 문제에서 1을 $로, 즉 다른 기호로 대응하는 방법을 배웠어, 이번에도 그 방식을 사용해볼까?

박효진 B: 좋아, 그렇다면 저번처럼 순차적으로 1을 $로...젠장, 이번엔 뒤에 숫자가 있어, 어떡하지?
박효진 A: 기호 2개를 써서 앞의 기호와 둘의 기호의 숫자를 비교하는 건...그러니까...일종의 줄다리기인거야, 가운데를 기준으로 왼쪽과 오른쪽을 비교하는.
박효진 B: 왼쪽과 오른쪽을 비교한다...왼쪽과 오른쪽의 기호가 달라야겠네? 아까는 $를 사용했으니까...다른 기호는 키보드에서 왼쪽에 있는 #으로 하자.
박효진 A: 그래 단순하고 좋네, 그러면 왼쪽의 1들을 $들로 바꾸고, 오른쪽의 1들을 #으로 바꿔주자.
박효진 B: 그래 좋아, 그 다음 $와 #의 개수를 비교해주면 되는데...
박효진 A: 잠깐, 비교라면 원래 문제랑 똑같잖아, 굳이 $와 #으로 바꿔줄 필요가 있었을까?
박효진 B: 필요가 있지, 원래는 가운데를 기준으로 좌의 1과 우의 1을 구분할 수 있는 방법이 없었다면, 지금은 기호가 다르잖아. 서로 구분할 수 있는 방법이 생긴 셈이지.
박효진 A: 그렇군...그러면 이 달라진 기호는 어떻게 숫자를 비교하지?
박효진 B: $와 #가 만나면 서로 사라지는 존재라고 해보자, 이때 두 수가 같으면 $나 #가 남지 않겠지만...만약 두 수가 다르다면, 둘 중 하나는 남게 되겠지.
박효진 A: 잠시만...어차피 결과는 1진수여야 하니까...$와 #를 만나면 서로 사라지는 존재로 할 게 아니라 한 쌍이 되면 1로

바뀌는 존재로 가정하는 건 어때?
박효진 B: 호랑이는 죽어서 가죽을 남기고, $와 #은 죽어서 1을 남긴다 그건가...일단 오케이.
박효진 A: 그래, 그러면 아까 문제 1에서 가운데 0을 기준으로 $ 하나를 1에 대응했듯이, 이번에도 가운데 0을 기준으로 $와 #을 하나씩 1로 바꾸어가는 거야. 좌우로 왔다갔다 하면서. 그리고 만약 두 수가 같다면? 더 이상의 $나 #이 남아있지 않아서, 결국 맨 왼쪽 0의 빈칸으로 가게 돼!
박효진 B: 좋아! 그러면 두 수가 같을 경우의 코드는 어느 정도 감이 잡힌 것 같고, 그러면 두 수가 다를 때는 어떻게 하지....?
박효진 A: 일단 두 수가 다를 때 해야 할 일은 1)남은 $ 기호 혹은 # 기호 1로 바꿔주기 2) 맨 오른쪽에 0 표시해주기 인데...1)은 어느정도 할 수 있겠지만, 2)는 어떻게 해야 할까?
박효진 B: 음...두 수가 다르다는 걸 전제로 튜링 기계를 돌려보고, 어떤 형태가 나오는지 관찰해보자! 그러면 답이 보이겠지.
박효진 A: 좋아, 먼저 $가 #보다 많을 경우...어떻게 하든 대칭이 깨져버리니까, 대칭이 깨져버리는 그 순간에 맨 오른쪽 0 옆 빈칸에 0을 출력하자(두 수가 같지 않음을 출력하자)
박효진 B: 그래, 그 다음에는 남은 $을 1로 바꿔주고...#이 더 많을 경우에도 똑같이 해주면 되겠어.
박효진 A: 좋아! 그러면 우리가 이야기한 내용을 코드로 확인해 볼까?

```
A 0 0 r A (맨 처음의 A상태, 오른쪽으로 가며 수를 읽자)
A 1 $ r B (만나는 1을 모두 $로 바꿔줘)
B 1 $ r B (그 과정을 반복해)
B 0 0 r C (어라? 0을 만났네? 상태를 바꿔줘)
C 1 # r C (그 뒤의 1은 #으로 바꿔주어야 하니까)
C 0 0 l D (이제 다시 0을 만났네?)
D # # l D (가운데 0-기준점으로 돌아가볼까?)
D 1 1 l D
D $ 1 r E (기준점에서 왼쪽으로 이동해서 $를 1로 바꿔줘)
D 0 0 l D (다시 기준점으로 이동해)
D _ _ r F
E 0 0 r E (기준점에서 오른쪽으로 이동해서)
E # 1 l D (#을 1로 바꿔줘)
E 1 1 r E
E _ 0 l G
F 0 0 r F
F 1 1 r F
F # 1 r H
F _ 1 * X (1을 출력하고 멈춰주어야겠군)
G 0 0 l G
G 1 1 l G
G $ 1 l G ($가 남았다니, 맨 오른쪽에 0을 출력해야겠군)
G _ _ r X
H # 1 r H (#이 남았다니, 맨 오른쪽에 0을 출력해야겠군)
H 0 0 r H
H _ 0 * X
```

Ⅲ. 시 사냥

안녕하세요, 'Poemgramming Bar'에 오신 것을 환영합니다. 이곳에서는 지식에 운율을 섞은 다양한 칵테일들을 만나 보실 수 있는데요, 원하시는 메뉴를 부른 뒤 저희 직원을 불러주시면 원하는 메뉴를 가져다 드리겠습니다. 직원은 'X', 'B', 'F', 'L', 'D'라는 이니셜을 가지고 있으니, 원하시는 직원을 불러 주세요, 여러분이 원하는 메뉴를 정확하게 가져다 줄 겁니다. 그럼, 좋은 하루 보내세요.

1. Appetizer- 마음의 도구

-본 칵테일을 즐기기 전에 안주로 드실 만한 맛있는 안주입니다-

1.1 "이런 소프트웨어들로 마음의 도구를 능숙하게 다루며 인간은 놀랍게 확장하고 있다"

Fiesta

아이즈원

아침에게 말해
오늘이 좋을 것 같아
이젠 아득했던 꿈들이 멀지가 않아
(...)
별들에게 말해
내일이 더 좋을 것 같아
이젠 펼쳐지는 일들이 겁나지 않아

1.2 “어떻게 이런 일이 가능한 걸까? 마음의 도구와 그 도구를 다루는 방법의 원천은 무엇일까? 컴퓨터과학은 어떻게 탄생한 것이고 지금까지 무엇을 밝혀냈을까? 앞으로 우리는 그 답을 살피러 떠나보겠다.”

시작한다는 것

이동식

시작한다는 것은
안 된다는 걸 믿는 것이 아니라
된다는 것을 믿는 것이다.
그것에 대한 확률이
아무리 낮아도
그것이 하고픈 일이고
꿈이라면
그 낮은 확률에도 희망을 걸고
나의 길로
만들어 가는 것이다

2. Wine - 400년의 축적

-도수가 높은 문구들이니 섭취에 주의해주시길 바랍니다-

2.1. “즉, 어떤 계산이건 그에 해당하는 기계를 테이프에 입력으로 받아 그 작동을 그대로 흉내 낼 수 있다. 다시 말해 그가 정한 범주에 드는 모든 기계적인 계산을 그 하나의 기계로 돌려볼 수 있기 때문에 보편만능의 기계라 부른다”

엄마

김완하

첫돌 지난 아들 말문 트일 때
입만 떼면 엄마, 엄마
아빠 보고 엄마, 길 보고도 엄마
산 보고 엄마, 길 보고 엄마

길옆에 선 소나무 보고 엄마
그 나무 사이 숯치는 바람결에도 엄마, 엄마
바위에 올라앉아 엄마
길옆으로 흐르는 도랑물 보고도 엄마

첫돌 겨우 지난 아들 녀석
지나가는 황소 보고 엄마
흘러가는 도랑물 보고도 엄마, 엄마
구름 보고 엄마, 마을 보고 엄마, 엄마

아이를 키우는 것이 어찌 사람뿐이랴
저 너른 들만, 산 그리고 나무
패랭이풀, 돌, 모두가 아이를 키운다

2.2. “튜링기계가 하는 일은 단순하다. 테이프 칸의 심벌을 읽고 쓰면서 테이프 위를 기껏해야 한 칸씩 좌우로 움직이는 일만 할 수 있다. 이러면서 기계의 상황이 매번 변경된다.”

그녀는 걸었다

황인숙

그녀는 걸었다. 긴 복도를
링거병을 끌고 졸음에 취한 나를 끌고
걸음, 걸음. 걸음,
문 닫힌 병동의 밤이 긴 복도
그녀는 쫓기듯 걸었다. 쫓기듯,
아니, 그녀는 걸었다, 걸음, 걸음, 걸음.
자기의 걸음을 견디면서
자기의 걸음을 즐기면서
자기의 걸음을 확인하면서
걸음, 걸음, 걸음, 창백한 형광등
그녀의 걸음이 가득한 복도
그녀는 걸었다. 유령 같은 나를 끌고,
걸음, 걸음. 걸음,
그녀는 걸었다
그녀는 걸었다.

3. Beer-그 도구의 실현

-문장의 톡 쏘는 맛에 집중해보세요-

3.1. “속내용을 감추며 차곡차곡 쌓기”

반달곰에게

김광규

하늘 아래 새로운 것이 어디 있으랴
창조도 하나의 결과에 지나지 않는다
태초에 원인이 있었고
뒤이어 결과가 따랐다
그 결과는 다시 원인이 되고
그 원인은 다시 결과를 낳았다
오래된 원인과 결과가
새로운 원인과 결과로 뒤바뀌며
마침내 오늘에 이른 것이다
그렇다면 어제는 오늘의 원인이고
오늘은 어제의 결과이며
오늘은 내일의 원인이고
내일은 오늘의 결과임에 틀림없다
원인과 결과를 끊으려는 미련한 곰아
새로운 원인을 오래된 결과라 부르고
오래된 결과를 새로운 원인이라 부르며
원인 없는 결과를 만들려 하지 마라
때로는 죽음도 하나의 원인이 되는 법이다
그리고 하늘 아래 새로운 법은 없다.

3.2. “미래에는 다른 재료와 장치로 지금보다 훨씬 작고 밀도가 높고 효율적인 컴퓨터를 구현할 수 있을 것이다.”

비밀정원

오마이걸

내 안에 소중한 혼자만의 장소가 있어
아직은 별거 아닌 풍경이지만
조금만 기다리면 곧 만나게 될 걸
이 안에 멋지고 놀라운 걸 심어뒀는데
아직은 아무것도 안 보이지만
조금만 기다리면 알게 될 거야
나의 비밀정원

4. Cocktail-지혜로 짓는 세계

-주인장의 톡톡 튀는 아이디어로 만들어진 칵테일입니다.-

4.1. “그런데 왜, 자연어 번역은 어려운데 반해서 프로그래밍 언어의 번역은 항상 가능한 걸까? 컴퓨터 언어는 모든 문장 부품의 의미가 하나로 고정되어있기 때문이다.

6은 나무 7은 돌고래, 열 번째는 전화기

박상순

첫 번째는 나
2는 자동차
3은 늑대, 4는 잠수함

5는 악어, 6은 나무, 7은 돌고래
8은 비행기
9는 코뿔소, 열 번째는 전화기

첫 번째의 내가
열 번째를 들고 반복해서 말한다.
2는 자동차, 3은 늑대

몸통이 불어날 때까지 8은 비행기, 9는 코뿔소
마지막은 전화기

숫자놀이 장난감
아홉가지 배운 날

불어난 제 살을 뜯어먹고

첫 번째는 나
열 번째는 전화기

4.2. ”이런 작업을 반복해서 한다. 그래서 우리가 찾는 답 데이터가 매우 높은 확률진폭을 가지도록 한다. 이런 반복을 충분히 하고 양자를 관찰하면 우리가 찾는 답이 매우 높은 확률로 나타난다.‘

슈뢰딩거의 고양이

제비뽑기(가수)

누구세요
이미 알고 있는 답
모른 척 물었어
이 시간에
늦은 밤에
네가 아니면
누구세요
괜히 한번 더 물었어
시간을 벌어볼까
불쑥 이렇게
찾아오면
어떤 표정을 지어야
오늘 하루는

몇 번을 본 영화 같아
대사까지 외울 수 있어
마지막 장면에서
멈출 거야

4.3. ”그래서 프로그래밍 언어는 중력이다. 프로그래밍 언어는 소프트웨어를 보는 시선의 방향을 끌어간다. 어떤 언어를 사용해서 소프트웨어를 짜느냐에 따라, 소프트웨어는 명령하며 기계 상태를 변화시키는 주문으로도 보이고, 기계와는 상관없이 논리적으로 따져간 증명으로도 보인다.“

자각몽

유시아

분명 난 숲속이었는데
순간 커다란 바다가 돼버렸네

5. Non-alcohol: 그 도구의 응용

-가볍게 드실 수 있는 도수 없는 문구도 준비했습니다.-

5.1. 컴퓨터 덕분에 우리는 점점 인간 고유의 지능에 집중하게 된다. 컴퓨터가 대신해줄 수 있는 일들이 많아지면서 인간 지능의 정의는 좁아지고 있다. 그러면서 우리는 인간에게 고유한 지능의 순도를 높여갈 것이다.

일곱 개의 단편

황동규

사랑과 죽음, 이 두 가지는 AI가 계속 체득하려 들 것이다. 그러나 AI가 마지막으로 가지고 싶어할 것은 우리가 '비밀'이라고 부르는 것 아니겠나.

Ⅳ. 게임의 역습

현대 사회에서 게임은 단순한 놀이의 범주를 넘어서, 플레이어들에게 교훈을 주거나 그들 간 유대감을 부여해주기도 한다. 이렇듯 다양한 목적과 재미 요소를 겸비한 게임들이 만들어지고 있고, 이에 호응하듯 게임 산업의 규모도 점차 커지고 있다. 특히 코로나 시대에 접어들며 집에서 많은 개인 시간을 보내는 사람들에 의해 온라인으로 재미를 주는 게임은 날개를 달게 되었다.

그렇다면, 단순히 재미를 주는 용도에서 벗어나 거대한 하나의 '문제'를 해결할 수 있는 협력의 장으로 게임을 이용할 수 있지 않을까? 이미 게임 세계를 현실 세계와 흡사하게 만드는 기술은 매우 발전한 상태이다. 그렇다면 현실의 문제를 게임 세계로 가지고 와 플레이어들로 하여금 해결 방법을 고민해보는 것도 불가능한 일이 아니다.

본격적 게임 설계에 앞서 게임 환경 내에서 플레이어들을 어떻게 문제 해결을 위해 이용할 수 있을지 큰 틀을 생각해 보았다. 먼저 두 집단이 대결을 하며, 이 과정에서 공격팀은 허점을 찾아내고 수비팀은 그 허점을 막아낼 방법을 계속 고민하는 것이다. 플레이어들에게는 이것이 그저 하나의 대결로 보이겠지만, 사실 이 데이터들을 취합하여 특정 환경의 보안점과 취약점을 알아낼 수 있을 것이다. 두 번째 틀은 플레이어들이 직접 본인 세상을 컨트롤할 수 있는 세계를 주고, 그 세계에 재난이나 문제 상황을 부여하여 효과적 문제 해결법을 찾아내게끔 하는 것이다. 세 번째는 수많은 플레이어들이 조종하는 캐릭터들이

하나의 세계 내에 들어있는 상황에서, 질병이나 재난 등의 문제 상황을 부여하고 개인의 행동들이 그 문제 상황의 변화에 어떤 영향을 주는지 시뮬레이션을 하는 것이다.

(1) 보안 시스템의 허점을 찾는 게임 - BOB (Block or break in)

본 게임의 주 목적은 특정 시설물이나 건물의 취약점을 파악하고 보완점을 알아내는 것이다. 인기있는 게임인 마인크래프트(minecraft)처럼, 게임 상에서 구조물이나 시설을 실제처럼 만들어내는 것은 불가능한 일이 아니다. 나는 이 점에 착안하여, '만약 실제 구조물이나 시설을 게임 상에서 옮겨온 뒤에 이 시설물의 보안 시스템을 옮겨와 침입이나 파괴를 하게 만든다면 취약점을 찾을 수 있지 않을까?'라는 아이디어를 얻었다. 게임의 가제는 BOB (block or break in - 막거나 침입하거나) 로 정했다.

BOB의 게임 방식은 다음과 같다. 먼저 BOB의 게임 세계로 놀이공원이나 큰 건물, 대학 캠퍼스 등 수많은 시설물들을 옮겨 제작한다 (제작 환경은 MINECRAFT와 비슷하다고 봐도 괜찮을 것 같다). 그리고 해당 시설물들의 보안 시스템 또한 게임 세계로 옮겨오기 위해, 경비원이나 CCTV, 감지 비상벨 등을 게임에서 구현하여 실제처럼 설치한다. 앞으로 이 시설들을 각각 하나의 맵이라고 부르겠다. 운영진이 구축한 이 각 맵이 유저들이 게임을 진행할 장소가 되겠다. 맵이 크고 침입이 어려울수록 더 많은 인원수를 한 서버에 배치하고 파악에 더 많은 시

간을 줄 것이다.

게임의 진행은 크게 한 맵에 대해 3페이즈로 진행이 된다. 1페이즈는 단순 침입, 2페이즈는 수비팀과 공격팀의 대결, 3페이즈는 최종 침입으로 이루어진다. 먼저 한 서버 내에 들어가는 유저들의 인원수는 정해져 있고, 접속자가 많아지면 서버의 개수가 많아지게 된다.

1페이즈 - 단순 침입

1페이즈는 기존 시설과 보안 시스템을 갖춘 맵에서 유저들이 침입하는 페이즈이다. 1페이즈 시작에 앞서 해당 시설물의 구조와 보안 시스템을 파악하는 시간을 플레이어들에게 준다. 정확히는 침입이 이루어지는 시간을 정해두고, 해당 시간 전까지 플레이어들은 맵을 돌아다니며 구조물, CCTV 등의 위치를 파악하게 된다. 놀이공원 맵을 예시로 들자면 각 놀이기구의 위치, CCTV나 경비의 동선 등을 플레이어들은 침입 시간 전까지 파악해야 한다. 한 서버 내의 플레이어들은 1페이즈에서는 모두 한 팀이 되어 침입을 진행한다. 그들은 침입 시간 전까지 채팅이나 지도를 공유하며 서로가 알아낸 정보를 공유할 수 있고, 침입 동선이나 방법 등도 협력하여 계획할 수 있다. 침입 시간이 되면, 플레이어들은 시설물 내에 보관된 자원을 찾거나, 제한 시간 내에 들키지 않고 숨어 있어야 한다. 이때 침입의 목적은 맵의 특성에 따라 달라지는데, 놀이공원이라면 재개장 때까지 숨어 있기, 회사나 대학 건물의 경우 자원/논문 등을 찾기 등으로 정해질 것이다. 주어진 침입 시간 동안 목적을 얼마나

잘 수행했는지에 따라 개인 1차 점수가 부여된다. 만약 들키지 않고 숨기가 목적이라면 기본 점수를 주고 들킨 횟수만큼 점수를 차감하며, 자원을 찾는 게 목적이라면 찾은 자원의 양만큼 점수를 부여한다. 이때 실제 침입의 진행은 개개인으로 진행된다 (한 맵에 한 서버의 플레이어들이 전부 들어가는 것이 아니다).

2페이즈 - 수비팀 vs 공격팀

2페이즈는 수비팀과 공격팀의 대결로 진행이 된다. 얻은 점수에 따라 공격팀과 수비팀을 나누는 비율이 달라지는데, 점수를 많이 얻은 서버에서는 난이도가 쉬웠다는 것이므로 수비팀에 많은 사람을 배치한다. 반대로 점수를 적게 얻은 서버에서는 난이도가 어려웠다는 것이기 때문에 공격팀에 많은 사람을 배치한다.

수비팀은 준비 시간동안 기존 맵의 취약점을 보안할 방법을 고민하고 직접 보안 시스템을 수정한다. 수비팀은 맵에 설치된 cctv 위치를 바꾸거나 소량의 cctv를 추가할 수 있으며, 경비의 동선이나 시간 등을 수정할 수 있다. 또한 1페이즈에서 본 서버의 공격팀이 얻은 점수에 대한 정보를 제공해 주는데, 이는 어디가 맵의 주 취약점인지를 수비팀에게 알려준다. 또한 몇 개의 서버를 묶어, 이 서버들의 수비팀들끼리는 의견 취합이 가능하도록 한다. 이 묶인 서버들을 하나의 '그룹'이라고 지칭하겠다.

공격팀 또한 아까와 같이 수비 준비 시간이 끝나면 침입 준비 시간을 받고, 1페이즈처럼 수정되고 보완된 보안 시스템을 파악

하게 된다. 그 후 1페이즈와 마찬가지로 재침입을 진행하고 목적 달성 정도에 따라 공격팀에게 점수를 부여한다. 공격이 얼마나 잘 막혔는가에 대해 수비팀에게 점수를 부여해준다.

3페이즈 - 최종 침입

3페이즈는 최종 침입으로, 한 그룹 내에서 가장 뚫리지 않은 서버의 수비팀의 맵을 가져오게 된다. 그룹 내에서 이 한 수비팀을 제외하고는 모두 공격팀이 되고, 이들은 수비팀의 맵을 공략할 방법을 다 함께 고민하게 된다. 준비의 시간이 지나면 이들은 모두 최종 침입을 진행하게 되고, 2페이즈와 마찬가지로 침입 성과에 따른 점수를 부여해준다. 반대로 여전히 침입을 잘 막았다면 이 한 수비팀에게 많은 점수를 부여해주게 되고, 이 맵에 대한 게임 진행이 끝이 나게 된다.

정리하자면 1페이즈 : 기존 시설에 대한 침입 / 2페이즈 : 보완하는 수비팀 vs 재침입하는 공격팀 / 3페이즈 : 잘 보완된 맵에 대한 최종 침입으로 게임이 이루어진다. 이 데이터는 모두 정리되어, 기존 시설물에 대해 보완이 필요한 취약점을 시설 운영자들에게 알려주고, 나아가 효과적인 보완 방식을 제시해 주게 된다. 2페이즈에서 수비팀에게 추가적으로 주는 cctv나 비상벨은 많지 않고 한정된 개수이기 때문에, 현실에서도 추가적으로 시설 운영자들이 추가해야 할 cctv나 비상벨의 개수는 많지 않을 것이다. 그 대신 효율적인 방범 시설의 배치를 찾는 데 수비팀이 주력하게 될 것이고, 이는 현실적으로 시설 관리자들이 보안 시스템을 업그레이드하는 데 큰 도움이 될 것이다. 본 게

임의 운영은 이런 장점을 시설 관리자들에게 제시하며 기존의 시설 구조와 보안 시스템에 대한 정보를 게임 운영자들이 받아 이를 구현하며 진행된다.

게임이 효과적으로 문제를 해결하기 위해서는 많은 인원수의 유저들이 게임을 진행하여 많은 사례를 얻는 것이 가장 좋을 것이다. 따라서 게임을 진행하던 중 유저들이 지루하지 않도록 유지할 필요가 있다. 만약 해당 시설이 아예 뚫리지 않는다면 수비를 더욱 진행할 필요도 없고 3페이즈까지 가는 과정이 지루할 것이므로 이를 피해야 한다. 따라서 규모가 꽤 커 허점이 존재할 수 있는 시설물이나 약간의 침입이라도 발생해서는 안되는 중요한 시설물을 구현해야 할 것이다. 또한, 운영진들이 맵을 구현해 주는 것은 한계가 있고 침입과 수비만 진행할 경우 유저들은 질릴 수도 있을 것이다. 따라서 유저들이 직접 맵을 제작할 수 있고 다른 플레이어들이 이에 대해 침입을 진행하는 플레이어 모드도 제공을 한다. 이때 좁은 건물에 방범 시설이나 경비가 많이 배치될 경우 당연히 침입이 불가능할 것이므로, 규모에 따라 배치할 수 있는 보안 시설이나 보안 시스템의 밀도가 달라지도록 한다. 또한 나중에 배치할 수 있는 보안 시설을 업데이트하거나 침입의 방식에 대한 업데이트를 진행하여 꾸준히 재밌게 게임을 즐길 수 있도록 한다. 본 게임이 매력적인 이유는 PTW (Pay to win : 돈을 많이 지불할수록 게임의 승리가 쉬워진다) 시스템이 없기에 모두가 동일선상에서 점수를 얻을 수 있고, 공평한 게임 진행이 가능하다는 것이다. 또한 서로의 의견을 나누고 다시 그 집단들끼리 의견을 나누는 과정에서

효율적인 보완 방식을 찾아갈 수 있을 것이다.

그러나 현실의 사람을 게임 캐릭터가 완전히 따라할 수 없고, 침입에 대한 진행 또한 약간의 차이점이 있을 수 있다는 것이 명확한 한계이다. 또한 침입 자체는 항상 혼자서 진행하게 되는데, 실제 경우에는 혼자가 아닌 여러 명의 사람이 침입할 수 있고 방범 시설에 해를 가할 수도 있다는 사실이 차이점이다. 또 적절한 점수의 기준, 서버에 배치할 인원수 등도 게임을 실제로 구현하고 플레이하기 전에는 정할 수 없다는 것이 아쉬운 점이다. 다만 이후 업데이트를 통해 침입 자체를 여러 명이 함께 진행하여 경비의 주의를 끈다거나 하는 다양한 플레이가 가능해질 것이라는 생각을 하였다.

(2) 이미지 검색 알고리즘 구성을 위한 게임 - 이미지 라이어 게임

구글 등의 검색 엔진 중에는 이미지 검색 기능을 제공하는 엔진들이 있는데, 직접 사용해 본 결과 검색 결과가 항상 옳다고 할 수 없었다. 그도 그럴 것이, 텍스트나 단어의 경우 조합할 수 있는 글자 부품의 개수가 한정적이고 정형화되어있기 때문에 큰 오차 없는 검색 결과를 제시할 수 있다. 하지만 이미지는? 만들어낼 수 있는 이미지가 셀 수 없이 많고 정형화되지 않았기에 그들을 정확히 분류하는 것이 쉽지 않다. 어떠한 이미지를 인식하면 딥러닝과 알고리즘을 통해 그 이미지와 연결되는 키워드를 매칭하여 검색을 진행하는 것이 효율적이고, 이를 위해서는 방대한 양의 이미지에 맞는 키워드를 인식시켜야 한다. 그러

나 이것은 불가능에 가깝기에, 이것을 게임에 접목시켜 특정 이미지에 연결되는 키워드를 유저들이 떠올려 이것을 학습시킨다면 어떨까 하는 아이디어를 얻었다. 그러나, 게임의 기본 소양은 재미를 갖춰야 한다는 것이다. 단순히 이미지를 주고 유저에게 떠오르는 키워드를 말하라고 하면, 재미를 느낄 수가 없을 것이다. 나는 이 과정을 어떻게 하면 좋을지 고민하다가, 술자리 게임으로 주로 하던 '라이어 게임'을 떠올렸다. 라이어 게임은 참가자들 중 술래 한 명을 제외하고 모두에게 같은 단어를 개인적으로 알려주면서 시작된다. 참가자들은 누가 술래이고 누가 술래가 아닌지 서로 알 수 없다. 이들은 대화를 통해 술래를 판별해야 하는데, 이를 위해서는 단어에 대한 설명을 해야 하지만 이때 술래가 이 단어를 알아채서는 안 된다. 즉 단어에 대한 설명을 하되, 너무 티가 나는 설명을 해버리면 술래가 알아채므로 게임을 지게 된다. 또 너무 모호한 설명을 한다면 다른 사람들이 이 단어를 제대로 아는지 모르는지 판단을 할 수 없어 술래로 몰릴 수가 있다. 그리하여 술래가 누군지 투표를 하고, 누군지 맞히지 못한다면 술래의 승리이며, 맞히더라도 그가 단어를 알아맞춘다면 술래의 승리가 된다. 이 게임에서 착안한 점은 단어 하나를 가지고도 유저들이 즐겁게 게임을 진행한다는 것이다. 그렇다면, 단어 대신 이미지를 주고 비슷한 방식으로 진행할 수 있지 않을까?

이미지 라이어 게임은 라이어 게임과 비슷한 방식으로 진행이 된다. 다만 다른 점은, 특정 단어가 아닌 이미지를 주게 되며, 이 이미지를 보고 술래가 아닌 유저들은 각각 본인이 생각하는

키워드를 떠올려야 한다는 것이다. 이미지를 본 뒤, 유저들은 자신이 생각하는 키워드에 대한 설명을 술래가 알 수 없도록 효율적으로 해야 할 것이다.

이때 이미지를 사용한다면 같은 사진을 보고도 여러 키워드가 나올 수 있다는 점이 기존 라이어 게임과의 차이점이다. 기존 라이어 게임은 단어 제시 -〉 대화 -〉 투표 -〉 맞췄다면 술래가 답 제시 의 과정으로 진행이 되었다. 그러나 이미지 라이어 게임은 이미지 제시 -〉 대화 -〉 키워드 맞추기 -〉 술래 투표 -〉 술래를 맞췄다면 술래가 답 제시 의 과정으로 진행된다. 즉 술래가 아닌 유저들이 자신들끼리 다른 키워드를 연상했다면 패배하게 된다. 따라서 플레이어들은 같은 키워드를 떠올릴 수 있도록 설명을 효과적으로 해야 할 것이다. 기존의 라이어 게임보다 술래가 아닌 팀이 승리하는 것이 어렵기 때문에, 만약 밸런스가 맞지 않다면 이미지를 본 후 대화를 나누기 전 술래가 아닌 팀이 각각 처음 떠오른 키워드를 입력하여 몇 명이 같은 생각을 하고 있는지 일종의 힌트를 주는 방법을 마련할 수 있을 것이다.

이 게임은 이미지 검색 알고리즘에 가치있는 데이터로 활용될 수 있다. 이미지를 제시하고 이에 대해 입력된 키워드를 수합하여, 이미지에 연결된 키워드로 검색 엔진에 학습시킬 수 있다. 즉 유저들은 라이어 게임을 플레이하면서 자동적으로 이미지 검색 엔진의 정확도를 높이는 데 기여하게 되는 것이다. 한 가지 키워드가 아닌 여러 가지가 나올 수 있기 때문에, 이 결과는 이미지 검색의 정확도를 크게 높일 수 있다. 한 이미지에 대해서

대부분이 연상한 단어의 경우 직접적 키워드로, 몇몇 사람들이 연상한 키워드의 경우 직접적 키워드가 아닌 간접적 키워드로 학습시켜 연관 검색어 등으로도 확장시킬 수도 있을 것이다.

V. 제 2의 사회, sns

2020년과 2021년은 그야말로 코로나가 주인공이었다. 전 세계로 코로나바이러스19가 퍼져나갔고, 대한민국도 예외는 아니었다. 2020년 1월을 시작으로, 갑자기 확진자가 급증하더니 이 글을 쓰고 있는 2021년 12월에도 확진자가 끊이지 않고 있다. 이제 코로나는 우리 생활에서 뗄 수 없는 존재가 되었으며, 코로나와 함께 삶을 영위하는 방식으로 모두의 생활이 변화하고 있다.

나는 때를 잘못 타고나, 가장 자유롭고 즐겁게 놀아야 할 시기를 코로나와 함께 보내게 되었다. 2020년에는 고등학교의 마지막 해로 내신이 모두 끝나고 자유롭게 여행을 하려고 했으나, 코로나의 확산으로 여행은커녕 집 밖에 나가지도 못하였다. 특히나 내 본가가 대구에 있었는데, 대구에서 집단적 확산이 일어나게 되면서 코로나의 공포에 떨 수밖에 없었다. 대학교에 들어가면 상황이 나아지겠지 했지만 여전히 코로나는 그칠 기미를 보이지 않았고, 비대면 수업을 수강하며 사회적 거리두기 속에서 고독한 새내기의 삶을 보냈다.

코로나 바이러스로 바깥에 나가지 못하고, 사람들과 실제로 만나지 못하는 내게 새로운 사회로 다가온 것은 바로 인스타그램과 유튜브 등의 sns였다. 온라인으로 친구들의 근황을 알게 되

고, 유명인의 소식을 접하며, 내 소식도 공유하고 많은 글들도 읽어볼 수 있는 sns는 내게 새로운 사회이자 눈이 되었다. 대학에 들어오기 전까지, 또 들어오고 나서도 sns에서 아주 많은 시간을 보냈다. 어쩌면 현실 세상보다 더 많은 시간을 보냈을지도 모르겠다. 그렇게 sns는 어느새 나에게 제 2의 사회가 되어버렸다. 나뿐만 아니라 수많은 사람들에게도, 코로나 시대에서 sns는 제 2의 사회로 자리매김했을 것이다. 현실에서 일어나는 사회 활동의 부재를 sns가 대체해줄 수 있기 때문이다.

코로나 시대에 sns가 제 2의 사회로 부상하는 것은 당연한 일이라고 할 수 있다. 외부 활동을 하고 직접 만나 의견을 나누는 것이 어려워진 상황에서, 어떠한 제약도 없는 sns는 매력적인 소통의 장으로 사람들에게 다가온다. 사용하는 유저들이 늘면서 스스로를 표출하고자 하는 사람들도 sns로 몰리게 되고, 활발한 소통이 꾸준히 일어난다. 또한 인스타그램의 경우 백신에 대한 정보까지 제공해주고, 누군가 백신이나 코로나에 대한 피드를 올릴 경우 보는 사람들에게 '코로나 19 정보 센터' 창을 띄워주면서 코로나 시대의 새로운 정보 전달의 매개체로 자리잡고 있다.

그러나, 이러한 sns의 부상이 절대 좋다고만은 할 수 없다. 직접 많은 시간을 인스타그램과 유튜브 등 sns에 들인 한 유저로써, 직접 sns의 수많은 단점들을 경험했기에 잘 알고 있다. 직접 겪은 사례들을 바탕으로 sns의 단점들을 분석·지적하고, 개선점을 담은 새로운 형태의 sns를 제시해 볼 것이다. 내가 주로 사용한 sns가 인스타그램과 유튜브이므로, 주로 이 두 플랫폼의

사례를 다룰 것이다.

한쪽 눈을 가리고

나는 평소 정치에 큰 관심이 없기 때문에 그에 관련된 뉴스를 잘 보지 않는다. 그러다가 과거 유튜브에서 한 번 정치인 A에 대한 긍정적인 영상을 끝까지 본 적이 있었다. 그땐 크게 신경을 쓰지 않았었는데, 그 날 이후 조금씩 해당 정치인A에 대한 뉴스가 내 유튜브 추천목록에 올라오기 시작했다. 그리고 그 뉴스들은 대부분 해당 정치인 A에 대한 좋은 이미지, 친근한 이미지를 부여해주는 영상이었다. 이것을 보고 뭔가 이상함을 느낀 나는, 사용하지 않던 계정에서 이번에는 그 반대 정당의 정치인 영상을 몇 개 시청해 보았다. 당연히, 그 계정에서는 반대 정당의 정치인들에 대해 긍정적인 뉴스 영상들이 노출되는 것을 볼 수 있었다. 인스타그램 또한 마찬가지로 비슷한 계열의 게시물들을 상위 피드에 추천하는 것을 관찰했다. 그제서야 유튜브와 인스타그램 등의 SNS가 유저들의 한 쪽 눈을 가리게 만든다는 점을 몸소 깨달을 수 있었다.

추후에, 이것이 '필터 버블'이라는 문제로 불리며 유저들이 한 쪽의 의견에만 갇히게 만드는 확증 편향을 심화시킨다는 것을 알았다. 유튜브나 인스타그램의 알고리즘이, 유저들이 자주 보는 게시물의 취향이나 종류를 파악하여 비슷한 게시물들을 꾸준히 추천해주는 것이다.

사실, 이러한 추천 알고리즘과 필터 버블은 sns의 운영자들 입장에서 필요한 요소이다. 사람들은 자신의 의견과 반대되는 게

시물보다 자신의 의견과 비슷한 게시물에 더 관심을 쏟기 마련이고, 추천이 유저에게 잘 들어맞을수록 그 유저는 해당 sns에 더 많은 시간을 쏟을 것이기 때문이다. 그렇기에 기존의 sns들은 확증 편향 문제를 해결하는 데 크게 힘쓰고 있지 않다. 유저들이 직접 확증 편향에서 벗어나는 방법들이 몇 가지 제시되었지만(의도적으로 반대 성향의 기사를 찾아 읽기 등), 이 방법을 모든 유저들에게 강제할 수도 없을뿐더러 일시적인 환기는 추천 알고리즘에게 큰 의미를 가지지 않는다. 그렇기에 가장 효과적인 해결방법은 확증 편향 문제를 어느 정도 해결할 수 있는 알고리즘을 담은 새로운 sns를 이용하는 것이라고 생각한다. 지금의 sns 운영진들은 확증 편향 문제를 심각히 접근하거나 적극적으로 해결하려고 하지 않기 때문이다.

내가 생각한 새로운 sns의 검색 알고리즘은 다음과 같다. 먼저 기존 sns처럼 유저가 본 영상들을 기반으로 추천하는 기능은 필요하다. 하지만, 동시에 '이와 반대되는 성향의 영상'을 함께 추천해주는 것이다. 가장 확증 편향이 심한 정치 관련 영상으로 예를 들어보겠다. 보수 정당의 영상을 많이 시청한 사람에게, 단순히 해당 정당에 관한 영상뿐만 아니라 진보 정당에 대한 영상도 함께 추천해 주는 것이다. 그리고, 본 영상이 띄고 있는 정치적인 색을 가려두는 것보다는 어느 쪽 성향이나 정당에 가까운지를 나타내어 주는 것이 더 좋을 것이다. 예를 들면 보수 정당에 관한 영상이면 영상의 썸네일에 빨간색 틀을, 반대의 경우에는 파란색 틀을 부여해 주는 것이다. 왜냐하면 영상의 성향을 직접적으로 표시해 주어야지 유저들로 하여금 '내가 어떤 쪽

의 영상만 보고 있구나'라는 자각을 하게 할 것이다. 또한 한 쪽 의견이 아닌 양 쪽의 의견이 모두 담긴 컨텐츠를 sns에서 묶어서 제공하는 것도 좋을 것이라고 생각했다. 예를 들면 두 논점을 한 번에 담은 기사나, 사용자들이 자유롭고 평화롭게 의견을 공유하는 토론의 장을 마련해주는 것처럼 말이다.

또한, 유저들이 이것을 자각하지 못할 수도 있으므로 sns에서 유저의 편향도를 알려주는 기능을 함께 담는 것이 효과적일 것이라고 생각했다. 유저가 어떤 색의 영상을 많이 보았으며 유저의 통계는 어느 쪽에 가까운지를 표시해 주는 것이다. 그리고 만약 한 쪽 성향의 영상만을 너무 많이 시청했다면 '편향 위험' 표시를 해주어 유저가 다른 쪽의 영상도 시청할 수 있도록 권장해주는 기능을 추가할 것이다.

다만 이 알고리즘의 한계는 각 영상마다 어느 쪽으로 어느 정도 편향이 되었는지를 정확히 판단하는 것이 어렵다는 것이다. 그렇기에 업로드하는 채널의 편향도, 주로 시청하고 공감·댓글을 다는 사람들의 성향을 분석하여 판단을 내리는 기준을 마련할 필요가 있을 것이다.

시장판이 된 sns

내가 주로 보는 인스타그램 피드를 보면, 거의 대부분의 글에 광고가 포함되어 있다. 10장의 사진까지 올릴 수 있으므로, 8~9장의 자극적인 컨텐츠에 광고 1~2장을 섞는 것이다. 그나마 8~9장의 컨텐츠와 광고가 관련이 없으면 양반이다. 심한 경우에는 후기나 정보 글에 광고 글을 몰래 끼워넣어 진짜 정보

글처럼 보이게 만드는 경우도 많다. 법이 바뀌어서 광고가 담긴 글의 경우 이를 반드시 명시하도록 했지만, 대부분의 광고 계정들은 더보기를 눌러야 맨 아래에 광고를 담았음을 겨우 확인할 수 있도록 해 두었다.

광고 계정을 운영하는 사람들에게 가장 중요한 것은 '팔로워'의 수이다. 최대한 많은 사람이 자신의 계정을 보아야 그만큼 광고 효과와 광고료가 많아지기 때문이다. 그렇기에 그들은 최대한 자극적인 컨텐츠를 싣거나 다른 재밌는 tv, 유튜브 영상을 잘라서 자신의 게시물에 담는다.

또, 광고 계정이 아니더라도 인플루언서들을 이용하여 실제 리뷰인 양 광고 영상을 찍고 몰래 광고하는 경우도 많다. 과거 유튜브에서 인플루언서 한혜연을 필두로 정말 많은 유튜버들이 뒷광고 (광고임을 밝히지 않고 제품을 몰래 광고하는 것)를 한 것이 밝혀졌으며, 그제서야 바이럴 마케팅에 대한 사람들의 비판적인 인식이 여론화되기 시작했다.

하지만, 여전히 sns를 통한 광고는 즐비하며 이러한 광고들은 심의도 제대로 거치지 않은 채 버젓이 대중들에게 게재된다. sns에서 자극적인 허위 광고 제품에 속아 피해를 본 사람들을 손쉽게 찾아볼 수 있으며, 광고인 줄도 모르고 속아서 돈을 지출하는 사람들이 많다. 이럴수록 더 많고 활발한 바이럴 마케팅이 일어나는 악순환이 계속된다. 광고 계정을 운영하는 사람들은 익명성에 숨어있으니 더더욱 자극적인 컨텐츠로 관심을 끌려고 노력하고, 이 과정에서 저작권은 전혀 고려되지 않는다. 실제 후기처럼 조작된 광고나 협찬 글들도 정말 많다. sns가 무분

별하게 상품을 진열해둔 시장판처럼 변하고 있는 것이다. 이러한 바이럴 마케팅의 경우, sns의 운영에는 아무런 도움이 되지 않으면서 sns의 이용 환경의 질만 떨어뜨린다.

sns가 자체적으로 제공하는 광고 피드도 존재하는데, 유저의 개인적 성향이나 취향을 파악하고 있는 sns이기에 이에 맞는 광고를 선택적으로 제공할 수 있다는 장점이 있다. 이러한 광고의 경우 sns의 수익 구조에 주요한 역할을 하며, sns 자체적 심의를 거칠 수 있다는 점에서 sns의 운영에 필요하다. 하지만 이 역시 너무 과도하게 노출이 된다면 이용자들의 피로를 불러일으킨다. 특히 본인의 관심사와 무관한 광고가 많이 노출된다면 유저는 귀찮음을 느낄 수밖에 없다.

sns 광고를 바이럴 마케팅과 sns 자체 광고로 쪼갰을 때, 바이럴 마케팅은 sns 자체에는 도움이 되지 않으나 인플루언서들의 주요 수익이 된다. 바이럴 마케팅을 없앤다면 인플루언서들은 해당 sns로 수익을 얻지 못할 것이고, 컨텐츠 생산에 뜸해질 것이다. 물론 유명세를 얻기 위해서, 소통을 위해서 이용할 수 있겠으나 수익의 여부가 인플루언서들의 동기에 영향을 준다는 것은 부정할 수 없다. 그러므로 바이럴 마케팅이 아닌 다른 수익 모델에 대한 제시가 필요하다. 유튜브의 사례를 보면, 조회수가 잘 나오는 유튜버의 영상에는 높은 광고료를 유튜브 측에서 지급하는 것을 알 수 있다. 해당 수익 모델이 괜찮은 이유는 무분별한 광고를 막고, sns 운영진 측에서 수익을 관리할 수 있으며, 효과적인 광고를 sns 운영진들이 배치할 수 있다.

내가 생각한 새로운 광고 시스템은 다음과 같다. 먼저 바이럴

마케팅은 sns 내에서 불가능하게 만드는 것이 좋다고 생각한다. 네이버 블로그의 협찬 후기나 유튜브 뒷광고처럼,무분별한 바이럴 마케팅은 sns 자체의 질을 떨어뜨리고 이용자들을 질리게 만든다. 광고는 sns 자체에 요청하면 이에 대한 심의를 진행하고, 통과되었을 경우 광고를 게재한다. 이때 광고를 어떻게 노출시킬지가 유저들의 피로도와 광고의 효율성을 결정할 것이다. 인스타그램의 경우 피드의 글들 사이에 sponsored가 표시된 광고 글들이 섞여서 표시되고, 유튜브는 영상이 시작하면 일정 시간동안 광고가 진행된다. 그러나 인스타그램 광고의 경우에는 불특정 다수에게 광고를 노출하므로 효율성이 떨어지고 피로도를 키운다. 대신, 나는 광고 물품의 종류에 따라 관련이 있는 인플루언서의 피드에 광고가 표시하는 방법을 택할 것이다. 만약 패션 관련 광고라면 모델이나 디자이너의 계정을 다른 사람이 볼 때 해당 광고가 게재되고, 음식이라면 요리사나 미식가 등의 계정에 게재되는 방식이다. 대중이 해당 인플루언서에게 관심을 가지는 분야가 있을 것이고, 이에 연관된 광고를 진행한다면 훨씬 피로도가 줄고 높은 효율성을 얻을 수 있을 것이다. 또한 영상 광고를 강제로 시청하게 하는 것은 굉장한 피로감을 주게 된다. 그 대신, 영상 광고를 시청한다면 추가적인 혜택을 부여해주는 것은 어떨까? 만약 스포츠스타 A의 피드에 게재된 축구화 광고가 있다고 하자. 만약 영상을 눌러 이것을 본다면 본 유저들에게 하트를 지급하고, 이 하트를 스타에게 선물하거나 스타와 소통을 하는 도구로 사용할 수 있다면? 광고를 보기 싫은 사람들은 보지 않는 것을 선택할 수 있고, 인플루언서와

소통하고자 하는 사람들은 보는 것을 선택할 수 있지 않을까? 그렇게 각 인플루언서에게 게재된 광고가 노출된 만큼 sns에서 인플루언서에게 소득을 지급한다면, 수익에 대한 문제도 해결할 수 있을 것이다.

또 다른 방법은 페이스북의 그룹 기능을 이용하는 것이다. 페이스북에는 그룹이라는 기능이 존재하여 비슷한 관심사를 가진 사람들이 모여서 의견을 나누는 것이 가능하다. 이를 활용하여 특정 집단의 목적에 맞는 광고를 게재하는 것이다. 예를 들어 축구인들의 집단에는 축구화 광고를, 패션 그룹에는 옷 광고를 배치하는 식이다. 이런 경우 사용자들의 니즈에 맞는 광고를 제공할 수 있어 높은 효율성을 보일 것이고 피로도를 줄일 것이다.

익명성의 가면

익명성에 숨어 악성 댓글을 다는 사람들에 대해 모두 들어보았을 것이다. 나도 인스타그램의 공개적인 게시물에 단 댓글에 대해 모르는 사람으로부터 심한 욕설을 들은 적이 있다. 한 연예인에 대한 악의적인 추측을 담은 게시물에서 '확실한 증거 없이 무분별한 비난을 하는 것은 자제해달라'는 댓글을 달았는데, 여기에 대해 욕설을 한 것이다. 물론 그 계정은 소위 가계정(본인이 특정되지 않는 익명 계정)이었기에 누군지 알아낼 수도 없었다.

이렇듯 sns가 점차 어엿한 소통의 장으로 발전하면서, sns에서도 익명성에 숨어 악플을 다는 사람들이 늘어나고 있다. 나같은

한 개인도 악플을 경험하였는데, 유명한 인플루언서나 연예인이 경험하고 견뎌야 할 악플의 수위와 그 양은 매우 심할 것이다. 많은 사람들이 악플로 정신적인 스트레스를 받고, 심한 경우 스스로를 해치기도 한다.

인스타그램, 유튜브, 페이스북 모두 댓글에 대한 신고 기능이 존재하지만, 이 플랫폼들에서는 다른 계정을 계속 만들 수 있기 때문에 그 의미가 없다. 운영진들이 계정을 정지하거나 삭제하더라도 익명의 가면을 쓴 또 다른 나가 계속 만들어지는 것이다. 그렇기에 sns 상의 악플을 줄이기 위해서는 sns의 기본 골조에 변화를 줄 필요가 있다.

먼저 계정을 자유롭게 생성하고 삭제할 수 있는 sns의 성질은 사용자들로 하여금 운영진의 제재를 두려워하지 않게 만든다. 가입 시 전화번호를 이용하는 메신저인 카카오톡의 경우, 정지를 당했을 경우 해당 번호로 계정 생성이나 이용이 불가능하다. 그렇기에 유튜브, 인스타그램 정지와 카카오톡 정지가 주는 무게감의 차이가 존재한다. 나는 이 점에 착안하여, 전화번호와 sns를 연동시키는 아이디어를 얻었다. 물론 이 전화번호가 타인에게 공개되지는 않지만, 사용자가 정지나 제재를 당했을 경우 무분별하게 계정을 새로 만드는 것을 방어할 수 있다. 또한 고소나 수사에 있어서도 그 사용자를 특정하기 쉽다는 장점이 있고, 사용자들도 이 점을 알고 있기 때문에 악플을 줄일 수 있을 것이다.

또한, 댓글 내용을 미리 검열할 수 있는 방안도 효과적일 것이라고 생각한다. 타인을 비방하는 욕설이나 혐오 언어 등을 지정

해두고, 해당 단어가 사용되거나 많이 입력된다면 이것을 출력하지 않는 방법이 있다. 아니면 욕설이 입력될 경우 경고 메시지를 통해 유저가 한 번 댓글의 내용을 스스로 생각하게끔 하는 기능을 제공하는 것도 좋을 것이다.

당근마켓과 같은 거래 어플의 경우, 사용자의 신뢰도를 척도로 표시하는 기능이 존재한다. 이 척도는 다른 사람들에게도 보여지기에 거래에 문제가 많았던 사람의 경우 다른 유저들이 조심하게 된다. 이 기능처럼, 본인이 악성 댓글을 작성한 척도를 표시하는 기능을 담는 것도 좋을 것이다. 그 척도가 높은 유저들의 댓글을 다른 유저들이 더욱 비판적으로 읽고 판단할 수 있을 것이며, 사용자들은 이 척도가 올라가지 않도록 조심할 것이다. 또한 일정 척도 이상 올라가면 제재를 하는 제재의 기준으로도 효과적으로 쓰일 수 있을 것이다.

미래에도 SNS는 제 2의 사회로 남아있을 것이다. 지금에 비해서도 더욱 확고하게 자리매김하지 않을까 생각한다. 하지만 여전히 sns에 대한 많은 지적이 있고, 더 건강한 sns 사회를 위해서는 변화가 필요하다. sns의 현 문제점을 많은 사람들이 인식하고, sns의 운영진들 또한 건전한 sns를 위해 많은 변화와 혁신을 시도하길 바란다.

Ⅵ. 우리는 모두 별의 자손들이다.

안녕하세요, 서울대학교 국어교육과 박효진입니다. 이과 여러분들에게는 조금 불편하실 수도있는 얘기이지만, 저는 무의식 중에서(그게 중간 과제를 공부하다가 졸아서라는 건 저와 여러분들만의 비밀입니다.) 저희의 창조주-엄밀히 말하면 저희 우주의 창조주는 아니지만-를 만났고 약간의 이야기를 들을 수 있었습니다, 빙의된 형태로요. 어이가 없으시겠죠, 지금 생각해보니 문과 여러분도 어이가 없으실 것 같네요. 지금 당장 학교 커뮤니티에 사이비 한 명이 있다고 말하고 싶은 심정도 이해합니다. 하지만 들어보면 꽤 재밌는 내용이에요, 요약하자면 그들은 한 '회사'의 직원들로, 저는 한 신입사원분의 이야기를 통해 저희가 프로그래밍을 통해 발생한 부산물이라는 사실을 알게 되었습니다. 아래는 제가, 그러니까 그분이 했던 말을 녹음해둔 걸 속기한 파일인데요, 그분의 말을 한국어로 번역하는 과정에서 약간의 오류가 있을 수있으니 이해해주시길 바랍니다.

-[규칙적이면서도 스펙트럼이 넓은 지적 생명체를 만들 것, 단 너무 많은 비용을 사용하지 않는 영역에서]

상사들이 주는 과제답게 그들은 귀찮은 일은 가장 말단인 나에게 맡겼다. 랜덤한 지적 생명체는 어렵지 않다, 임의의 변수를 마련하면 그만이니까, 규칙적인 지적 생명체도 쉽진 않지만 할 만 하다. 규칙을 구상하는 게 귀찮지만, 규칙을 만들면 그 이후로는 나의 영역이 아니니까. 하지만 규칙을 가지면서 스펙트럼을 갖출 만큼의 다양성을 가진 지적 생명체, 그것도 피의 영역에서? 전 선배는 유전자라는 시스템을 만들었으나 계산적인 측

면에서 실패했다. 지나치게 많은 생명체가 만들어졌고, 그들을 통제하기도 쉽지 않았다. 어떡하지? 규칙성을 가져야 그들을 통제하기 편할 것이다. ...우리는 필수적으로 '우주'를 만들기 전에 시행해야 하는 프로그램이 있다. '빅뱅'을 실행하면 만들어지는 세계에 지적 생명체가 공존할 수 있었던 것 같다. 그래서 나는 행서을 바탕으로 글자를 만들기로 결정했다(우주 그림 첨부) 행성의 각도를 바탕으로 지적 생명체의 특성을 설정하기로 결정했다. 선배들은 우주 하나를 만드는 귀찮은 일에 하나를 하기 싫어서 나같은 신입한테 이런 일을 시켜. 이러려고 취직한 건 아닌데 말이야. 결국, 잠을 깨기 위해 은하수를 한 잔 마시면서, '우주'가 다운로드되기를 기다렸어. 은하수를 반쯤 비우니 다운로드가 완료되었지. 선배들이 공공 클라우드 구석에 박아놓은 '입자'를 찾아가서(나름 체계는 있는 회사라서 창조- 우주-그 외의 자잘한 것들을 따라가면 찾을 수 있어) 설치한 뒤에, 적당히 기다리면 이제부터가 귀찮은 일의 시작이야. 사실 이 일도 귀찮은 일을 대신 하기 위해 만들어진 건데, 이 일도 또 다른 귀찮은 일이 되었지, 재밌지 않아? 처음 우주 안에서 지적 생명체가 업무를 대신하게 한다는 건 정말 혁신적인 발상이었는데 말야, 예를 들어 고분자 화합물의 생성 같은 거, 지금이야 지적 생명체들이 대신 해준다지만 그거 진짜 귀찮은 일이었거든. 그 사람이 이 회사의 창립자였어. 지금은 은퇴했지. 그 많은 돈을 벌어놓고 일하고 싶은 사람이 어디 있겠어? 지금쯤 편안하게 누워서 잠이나 자고 있겠지. 아무튼, 너, 정확히는 나의 육체의 원래 소유자-의 우주는 내 사수가 나한테 지적 생명체를 만드

는 법의 예시를 보여주기 위해 만든 파일을 내가 조금 수정한 거야, 네가 사는 우주가 고작 5분(역: 다른 시간 단위로 말씀하셨는데 이 부분은 제가 이해하지 못해서, 대충 제가 생각하는 짧은 시간으로 적었습니다) 만에 만들어진 단순한 결과물이라니, 그래도 뭐 어쩌겠어? 나는 입자가 시작되고 난 뒤에는 사수가 짜준(역: 언제 짜줬다고 했는데 저의 뇌 용량에서는 들어오지 않는 단위라서 생략했습니다) 코드를 붙여넣으려고 했어, 실제로도 그렇게 했지. 내가 맡은 업무는 그렇게까지 다양한 종류의 지적 생명체가 필요하진 않았거든. 나한테 필요한건 쓸만한 규칙과 그 규칙 아래에서 발생하는 적당히 비슷하고 적당히 다른 지적 생명체들이었거든. 신입한테 그 정도면 충분하지 뭘. 그런데 아무리 그래도 사수가 짠 코드를 그대로 복사했다고 하면 평판이 곤란하잖아? 그래서 나는 신입답게 사수를 따라한-하지만 어딘가 좀 모자란 코드를 만들기로 했지. 그래서 나는 선배가 짜준 코드를 열고, '유전자' 함수를 매우 간략하게 한 뒤에 다른 시스템을 도입했지, 나는 항성과 행성의 위치를 바탕으로 지적 생명체의 특성을 설정했어. 보통 항성과행성 만들기는 요즘 어린애들도 한번쯤은 다 해보는 활동이거든, 그래서 그걸 복사해서 항성과 행성의 좌표를 바탕으로 지적 생명체의 특성을 설계하는 코드를 완성했어.

...이 뒷부분부터는 제가 잠이 깨서 잘 기억나지 않지만, (죄송합니다) 이 얘기를 듣자마자 저는오늘의 별자리 운세를 켜본 기억이 납니다. 여러분의 별자리는 뭘지 궁금하네요, 감사합니다.