게임의 역습

현대사회에서 게임은 많은 사람들이 즐기는 여가활동 중 한 개로 자리잡았다. 최근에는 이런 게임을 이용해 여러 현실의 문제를 해결하려는 시도가 늘어나고 있다. 가장 유명한 예로는 세바스찬 승이 개발한 아이와이어라는 게임이 있다. 이 게임은 집단 지성을 이용한 3D 두뇌 맵핑 게임이다. 플레이어는 세포를 구성하는 큐브로부터 표시되는 영역으로 연결하여 다 연결했을 때 정확도와 빠르기를 판단하여 매겨지는 점수를 목표로 한다.

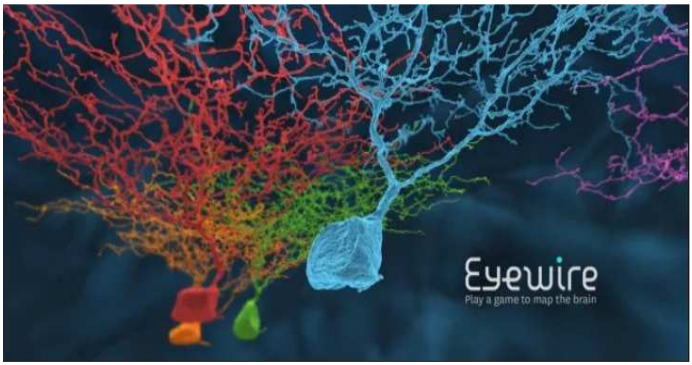

그림 1 아이와이어 게임의 뇌의 뉴런 맵핑 예시

다른 예시로는 기어박스 소프트웨어에서 제작된 보더랜드 3의 미니게임이 있다. 이 게임은 게임속에 존재하는 단순한 미니게임으로 기존에 위치한 블록을 옮겨 알맞게 고치는 것을 목표로 하는 게임이다. 이 게임은 게임 속 플레이어에게 보상을 주기위해 만든 것 같지만 사실은 미생물의 DNA 분석 연구에 도움이 되는 게임이라고 한다. 미생물의 DNA 데이터 중 컴퓨터에서 에러가 발생하거나 잘못해석하는 부분을 이런 블록 형태의 퍼즐로 만들어 플레이어로 하여금 해결하도록 만드는 것이다. 또한 이러한 해결과정 역시 DNA 해석 알고리즘에 참고자료로 저장되기 때문에 알고리즘의 향상에도 기여한다고 볼 수 있다.

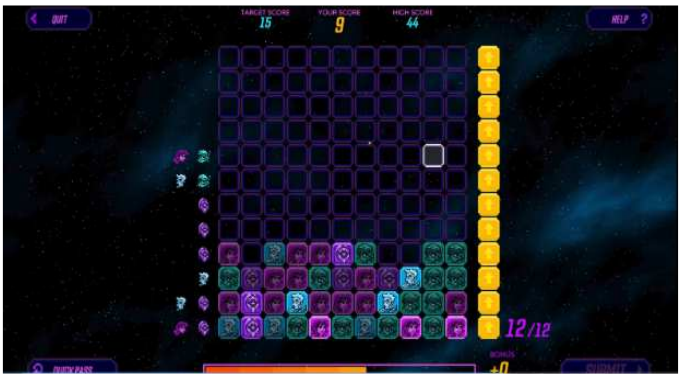

그림 2 미생물 연구에 도움이 되는 보더랜드 3 미니게임

게임은 인공지능의 발달에도 기여한다. 가장 유명한 예시로는 온라인 바둑게임 타이젬 바둑에서 데이터를 모으기 위해 Magister(P)라는 계정으로 활동한 알파고가 있다. 알파고는 머신 러닝을 이용한 바둑 프로그램이다. 이러한 머신 러닝 과정 중 하나로 온라인 게임으로 들어가 인터넷 상에서 바둑 기사들과 바둑을 두어 데이터를 확보한 것이다. 결과는 60전 60승으로 실제 인간과의 대국을 통한 데이터를 많이 얻을 수 있었다.

이러한 연구의 공통점은 게임을 통해 많은 양의 데이터를 모은다는 것이다. 게임은 다양한 사람들이 자유롭게 참여할 수 있다는 점에서 이러한 데이터를 모은 방법으로 적합한 것이다. 이러한 게임의 특징은 데이터가 많이 필요한 다른 분야에서도 활용될 수 있다. 최근 세계적인 이슈인 코로나바이러스감염증 -19의 범세계적인 유행의 연구를 위한 데이터 역시 게임을 통해 얻을 수 있을 것으로 보인다. 특

히 실제 사람들의 행동으로 인한 감염 모델의 모델링을 하기 용이할 것으로 생각된다.

실제로 게임을 통한 전염병 연구를 시도하려는 시도가 존재하였다. 블리자드 엔터테인먼트의 게임 "World of warcraft", 통칭 WOW에서 버그에 의해 발생한 '오염된 피 사건'이 그 주인공이다. 2005년 북미서버에서 발생한 거대 게임 전염병 사건이다. 당시 레이드 보스 중 '학카르'라는 보스가 있었는데 이 보스는 플레이어들 사이 전염되는 디버프를 걸었다. 원래 던전을 나오게 되면 자동으로 디버프가 풀려 전염되는 일이 없겠지만 문제는 플레이어의 펫에 감염된 디버프는 해체되지 않았던 것이다. 이러한 디버프는 대도시의 NPC에게 전염되었고 죽어서 디버프를 없앨 수 있는 플레이어와 다르게 NPC는 죽지 않았기 때문에 슈퍼 감염자과 되었고 이 때문에 많은 플레이어들이 이 디버프에 걸려 죽게되는 것이었다. 실제 전염병과는 다른 형태의 전염병의 유행이었음에도 불구하고 미국질병통제예방센터에서 이 사

건의 통계자료를 요구했을 정도로 꽤나 가치 있는 전염병 모델이었다.

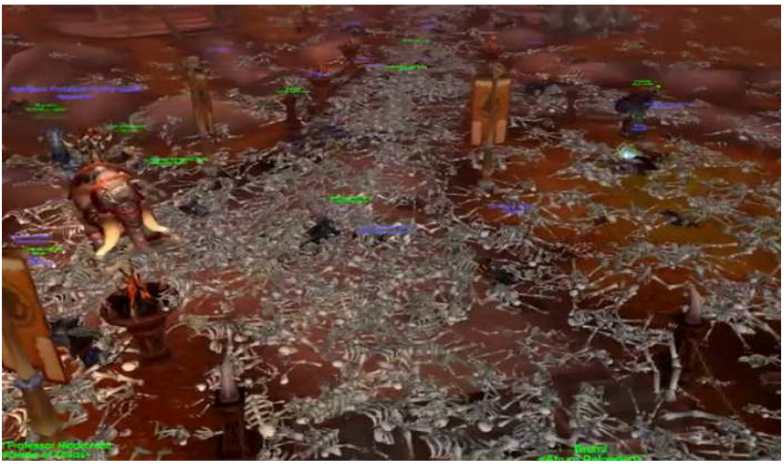

그림 3 오염된 피 사건 당시 wow 게임 내 풍경

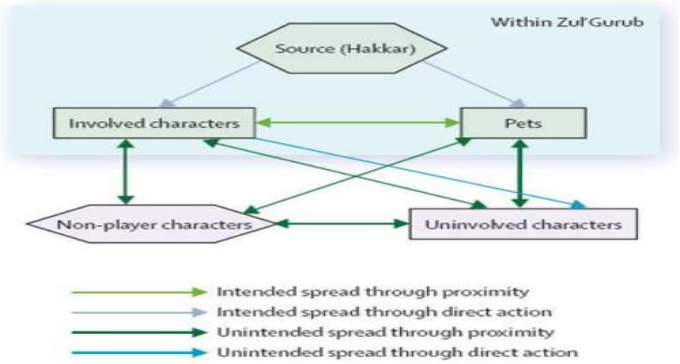

그림 4 오염된 피 사건의 감염 경로, 하나의 감염 모델로 볼 수 있다.

이미 전염병의 전파를 시뮬레이션하는 게임 역시 존재한다. Ndemic Creations에서 개발한 모바일 게임 Plague Inc, 한국어로 전염병 주식 회사가 그 주인공이다. 이 게임은 플레이어가 전염병을 이용해 전세계를 감염시키는 것을 목적으로 한다. 플레이어는 전염병의 특성을 결정하고 이를 통해 전세계에 전염병이 어떤 양상으로 전파되는지를 시뮬레이션하게 된다. 이 게임은 바이러스 이외의 박테리아, 곰팡이 등 여러 병원균 역시 시뮬레이션 가능하도록 했으며, 각 병원균의 특징을 잘 살렸다는 장점이 있다. 그러나 실제와 다르게 인간의 방역이 제대로 기능하지 않으며, 실제 질병에 비해 너무 감염성이 강하게 설정되어 있는 등 현실과 동떨어져 있는 요인도 다수 존재한다. 또한 플레이어가 1인

으로 고정되어 있어 다양한 사람의 데이터를 얻기 힘들고, 플레이어 간의 상호작용에 의한 다양성을 얻을 수 없다. 무엇보다 플레이어가 질병의 입장에서 플레이하는 것이 전염병의 방역에 도움이 되지 않는 데이터를 제공하며, 큰 단점으로 꼽히고 있다.

그림 5 전염병 주식 회사의 플레이 화면, 인류의 전염병에 대한 대응이 잘 나타나지 않는다.

위에 소개한 '오염된 피 사건'과 시뮬레이션 게임 'Plague Inc'의 단점을 보완하고 응용하여 전염병 감염 모델링 게임

의 설계는 다음과 같이 생각 해 볼 수 있었다. 먼저 게임의 장르는 오픈 월드 게임, 그 중에서도 MMORPG로 제작한다. 이후 서버는 여러 서버를 증설한다. 각 서버는 인구밀도가 서로 다르게 설정되도록 한다. 흔히 말하는 도시 서버와 시골서버를 구분하는 것이다. 대부분의 사람들은 도시 서버에 몰리겠지만 시골 서버는 시골 서버 나름대로의 어드벤티지를 주어 사람들이 서버에 참가하도록 한다. 게임에서는 실제 사회와 같이 자본주의 사회를 구축하고 사회활동을 하지 않으면 돈을 벌지 못해 살지 못하는 게임 구조를 설정한다. 또한 게임속에서도 병원, 여가활동 장소등을 등장시킨다. 요약하자면 위에서 언급한 World of warcaft나 스스로 우주를 개척해나가는 eve online과 같은 오픈 월드 RPG의 세계를 만드는 것이다. 대신 그 세계의 배경이 현대의 도시 또는 시골인 것이다. 게임내에서 플레이어는 자신이 원하는 직업을 선택하여 그 직업으로 인생을 살아가는 것을 목적으로 한다. 여기서 사람들이 원치 않지만 사회에 꼭 필요한 직업, 일명 3D직업들은 NPC나 운영하는 측에서 담당하도록 한다.

그림 6 대표적인 오픈 월드 게임인 eve 온라인의 자유도, 현재 표시된 영역 외에도 플레이어가 갈 수 있는 영역이 존재한다.

진정한 모델링은 여기서 시작된다. 사람들이 사회를 이루고 경제활동이 제대로 돌아가기 판단되면 전염병의 최초 발병자인 NPC를 게임에 등장시킨다. 여기서 이 NPC는 오염된 피 사건의 NPC와는 다르게 일정 시간이 지나면 완치, 혹은 사망하게 설정한다. 또한 게임내의 직업은 무작위로 설정되도록 한다. 그러면 이 NPC와 접촉한 플레이어들에게 전염병이 퍼져나가게 될 것이다. 이 가상 전염병의 모델은

현실성이 떨어졌던 'Plague inc' 의 전염병과 달리 코로나바이러스-19나 메르스, 사스 등 실제 전염병의 감염률, 사망률, 잠복기 등을 대입하여 설정한다. 전염병이 플레이어들 사이에 어느정도 퍼지게 되면 게임의 정부, 즉 GM으로부터 그 전염병의 정보와 감염예방 방법, 그리고 감염을 예방 수칙을 발표한다. 또한 마스크나 손소독제, 방호복과 같은 방역 물품을 게임 사회 내에서 판매를 시작한다. 보통 게임이라면 이 전염병이 없어질 때까지 플레이를 하지 않으면 되지만 이 게임은 실제 자본주의 사회처럼 돌아가기 때문에 플레이어들은 사회활동(플레이)을 계속해야 한다. 이에 플레이어는 방역 대책에 따라 행동하며 손해를 보는 것과 상관없이 평소와 같은 플레이를 하는 것, 둘 중 자유롭게 선택하여 플레이를 하면 된다. 각 역할에 따른 전염병 모델에서의 역할은 다음과 같이 볼 수 있다.

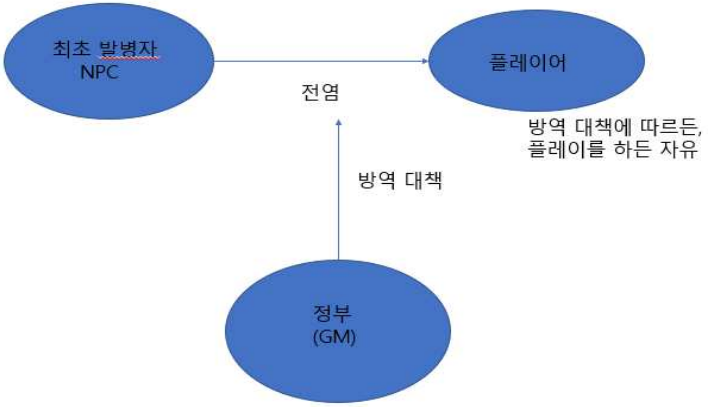

그림 7 게임 속 역할군의 전염병에 대한 역할

그 후 플레이어들의 행동 양상을 보는 것이다. 감염되는 플레이어 수에 따라 GM은 다양한 방역 대책을 사용한다. 또한 게임 속에서 따로 플레이어들의 검사를 실시해 게임내 사회에서 확진자를 구별해 내어 통계를 계산한다. 물론 실제 감염자 수 역시 게임 데이터를 통해 측정한다. (게임 사회에서 이 숫자는 현실처럼 모르는 것으로 가정한다.) 게임 속 질병의 확진자 수에 따라 GM은 사회적 거리두기의 선포, 외출 시 벌금 부여, 긴급 재난 구호 자금 지원, 등교 연

기 등 현실에서 사용될 수 있는 다양한 방역 대책을 시행하는 것이다. 방역 대책은 각 서버에 따라 다르게 설정한다. 이를 통해 어떤 방역 대책이 감염의 예방에 어떠한 효과를 보이는 지 확인한다. 게임이기 때문에 현실보다 전염병의 통계를 내기 쉬우며, 이를 통해 모델링을 하는 것도 간단할 것이다. 이러한 과정을 통해 감염 모델의 확보가 가능하다.

언뜻 보면 컴퓨터를 이용한 시뮬레이션이 더 돈이 들지 않고 더욱 많은 데이터를 확보가능 할 것이라고 생각할 수도 있다. 이 방법은 시뮬레이션에서 완벽히 흉내 낼 수 없는 사람의 심리를 모델에 적용이 가능하다는 장점이 있다. 실제로 현재 문제가 되고 있는 이태원 클럽의 집단 감염 사건은 시뮬레이션에 쉽게 포함될 수 없는 무한이기주의 심리에 의해 발생한 것으로 이러한 상황에 대한 감염 모델을 짤 수 있는 것이다. 단, 단점도 존재하는데 실제 현실과 다르게 사람들이 게임이라며 현실세계와는 다른 심리를 가지고 행

할 수도 있는 것이다. 특히 이러한 MMORPG 장르에 방역 대책 같은 여러 제약을 두게 된다면 게임으로서의 재미가 떨어져 모델링에 참여하는 인원수가 줄어들 것이다. 또한 이러한 방식의 게임은 지역 감염에 대한 모델링 밖에 할 수 없다. 지역간의 교류가 없기 때문이다. 따라서 지역 외 감염에 대한 방역 대책의 평가나 모델링은 힘들 것으로 보인다. 이러한 문제점에 대한 해결책이 있긴하다. 참여해주는 사람들에게 게임에서의 어드벤티지를 부여하거나, 한 서버에 다양한 도시를 만드는 것이 그 예시이다. 그러나 이러한 해결법은 자본이 많이 들기 때문에 당장은 힘들 것으로 보인다. 이런 식으로 많은 사람이 참여할 수 있는 게임을 활용해 여러 문제를 해결해내는 사례가 더욱 많이 생겼으면 한다.

다중지능이론과 메타 학습의 새로운 방향성 제시

인공지능은 현재 많은 발전을 이루었지만 제한된 범위에서 문제를 해결하는 데 전문화된 약인공지능에 머물러 있다. 이 이유는 인류의 기술적 문제도 있겠지만 사람에 지능에 대한 정의이 완전히 이루어지지 않은 상태에서 유용한 부분에 집중하여 개발이 이루어지는 것이 근본적인 문제일 것이다. 이를 해결하기 위해 우리는 먼저 인간의 지능에 대해 이해해야 할 것이다. 그렇다면 인간의 지능을 어떻게 보아야 인공지능에 적용시킬 수 있을까? 여기서는 아직 인공지능에 적용되지 않은 것으로 보여지는 다중지능이론을 이용한 인공지능의 발전 가능성과 방향에 대해 논하고자 한다.

다중지능이론에서는 인간의 지능을 단순한 하나의 능력이 아닌, 다수의 능력에 구성에 의해 나타나는 형태라고 주장한다. 다중지능 중 하나로 인정받기 위해

서는 지능으로 생각되는 것이 두뇌의 어떤 부위를 차지하고 있다는 것의 증명, 독립된 형태의 관찰, 일련의 작동 체제의 보유, 특유의 발달 과정 등과 같은 총 8개의 요소를 만족시켜야 한다. 현재의 지능 요소 여겨지는 것으로는 언어지능, 논리수학지능, 자기성찰지능 등 8개가 있다.

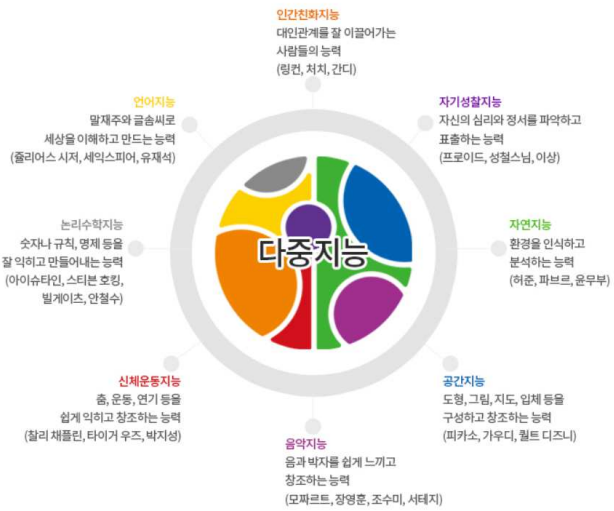

그림 1 다중지능이론에서 지능의 종류

현재까지의 인공지능들의 발전 상태인 약인공지능은

다중지능이론 중 논리수학지능을 컴퓨터 프로그램을 이용하여 구현해 실생활에 활용하고자 한 예라고 할 수 있을 것이다. 하지만, 강인공지능은 자기 자신에 대한 파악과 이를 통한 자체적인 학습이 가능해야 하므로 자기성찰지능을 가져야 한다고 볼 수 있다. 최근에 실제로 이러한 부분과 관련된 인공지능 연구가 진행되고 있다. 메타인지능력 구현을 목표로 하는 이 연구는 인간의 능력 중 자신이 모르는 것이 무엇인지에 대한 인지인 메타인지를 프로그램이 가질 수 있도록 하는 것을 목표로 한다. 학습하는 방법을 학습하는 인공지능을 만들고자 하는 것이다.

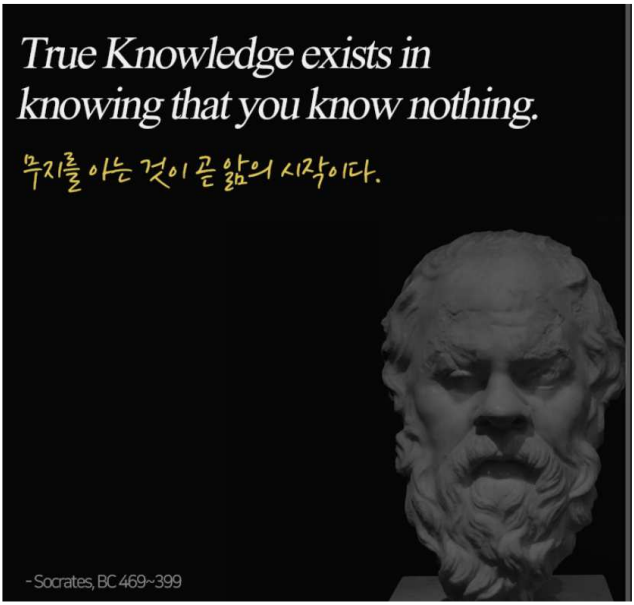

그림 2 옛 소크라테스의 명언과 같이 인공지능 프로그램도 무지를 알아야 앎을 시작할 수 있을 것이다.

이러한 메타인지학습법은 인공지능의 문제점 중 하나인 데이터 효율성 문제와 이로 인한 새로운 형태의 학습이 적은 양의 데이터로 불가능한 것을 해결해 주기도 한다. 이 메타인지 학습법 중 하나인 원 샷 학습법은 현재 인공지능의 학습법으로서 연구되고 있는 대

표적 예이다. 원 샷 학습법은 한 번 만 학습시켜도 된다는 의미의 원 샷이며, 이 학습법은 인간이 많은 양의 데이터를 보지 않고도 각 물체의 특징점을 활용해 다른 형태의 같은 물건을 보더라도 동일한 물건임을 이해할 수 있는 능력을 인공지능에게 학습시키고자 한다. 즉, 다양한 강아지 종의 사진을 인공지능이 보았을 때, 같은 종의 다른 사진을 보고도 이것이 해당 종임을 알아챌 수 있도록 하는 것이다.

그림 3 원 샷 학습법이 대표적으로 사용되는 얼굴 인

식, 현재 많은 기기에 사용되고 있다.

하지만, 원 샷 학습법은 한계점이 분명하다. 이미 학습된 신경망이 있어야 하고, 시험용 데이터와 학습용 데이터 간의 유사성이 보장되어야 한다는 점이다. 예를들어 진돗개와 닥스훈트의 두 사진을 보았을 때, 이 둘이 인간이 말하는 '개'로 인식하는 것이 이러한 원 샷 학습법을 통해 만들어진 인공지능에게는 힘들 수도 있다는 것이다. 이러한 점은 데이터를 많이 사용하면 보완될 수 있지만 위에 말했듯 원 샷 학습법 자체가 데이터의 간소화를 위해 개발되고 있으므로 이는 모순이라 볼 수 있다.

그림 4 이렇게 많은 차이를 보이는 두 생물도 인간은 '개'라고 인식 가능하다. 원 샷 학습법은 완벽하지 않기 때문에 이러한 인식이 힘들다.

이를 새로운 관점에서 제작한다면 이 문제점을 극복할 수도 있을 것이라고 생각한다. 단순히 이미지에 대한 정보가 아니라, 해당 물체에 대한 용도에 대한 관점으로 접근하는 것이다. 칼을 인식할 때 단순히 칼

의 모양이 뾰족하고 손잡이가 달렸음을 인지하는 것이 아니라 왜 이 칼이 뾰족해야 하고 왜 손잡이가 달려 있는지를 같이 학습하는 것이다. 실제로 어린아이는 물체의 용도를 알아내기 위해 다양한 방법으로 사용해 보다 방법을 알게 된 후 다르게 생긴 물건이라도 같은 기능을 하면 같은 물건으로 알아보게 된다. 이러한 어린아이를 가르쳐주듯이 인공지능에게 가르쳐 준다면 적은 데이터로 인공지능이 학습할 수 있을 것이다.

그림 5 아기는 장난감을 이리저리 사용해보면서 장난감의 용도에 따른 특징을 알아낸다.

물체의 정보를 다차원 데이터를 통해 각 부분의 특성을 용도에 맞춰 인공지능에게 학습시킨다 생각해보자. 예를 들어 칼이 날카로운 이유는 특정 물체를 베고자 하는 용도 때문이다. 칼의 날카로운 부분에 대한 다차원 데이터가 물체를 베고자 하는 용도로 학습된다면, 다른 형태의 칼에서도 뾰족한 부분이 존재할 경우, 이는 물체를 베고자 하는 용도가 있을 것이라고 인공지능은 학습하게 될 것이다. 이와 같은 방법으로 뭉툭한 손잡이의 경우에는 손으로 잡을 때 신체가 다치지 않게 하기 위함이라는 용도를 학습할 것이다. 만약, 칼이라는 데이터를 추출하고 싶다면, 우리는 베기 위한 부분과 신체의 안전을 위한 부분을 모두 가진 물체에 해당하는 것을 찾아내면 될 것이다.

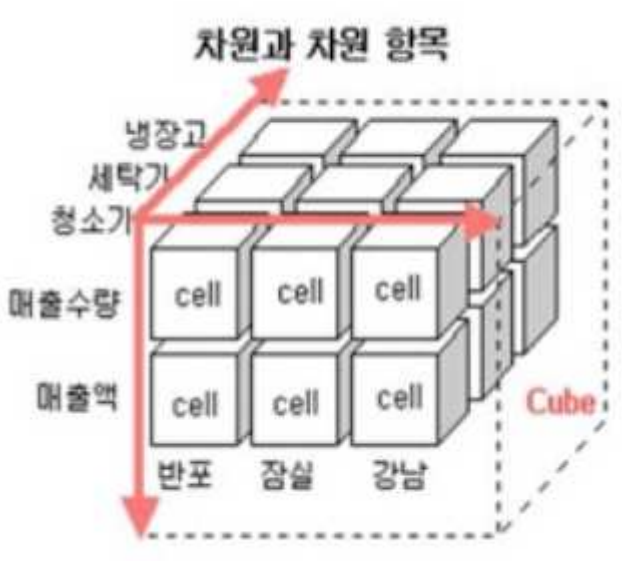

그림 6 다차원 데이터의 양식, 실제로 여러 분야에서 이미 다차원 데이터는 데이터의 분류에 사용되고 있다.

이 방법이 획기적일 수 있는 이유는 범용성과 효율성 때문이다. 각각의 물체는 용도가 겹치는 물건이 다수 존재한다. 단순히 '칼'에 대해 학습하기 위해 데이터를 넣는 것이 아닌, '도구'에 해당하는 데이터를 넣음

으로서 '칼'밖에 구별하지 못하는 기존 신경망과는 달리 '도구'전체 중 '칼'을 찾아낼 수 있는 능력을 가지게 된다. 이는 자연스럽게 더 넓은 분야에 대한 분석이 더 적은 양의 데이터로 이루어질 수 있게 만들 것이다. 또한 완전히 새로운 데이터가 인공지능에 입력되었다 하더라도 인공지능은 배운 분류를 통해 그 데이터를 분류하여 대략적인 용도와 종류를 유추해 낼 수 있을 것이다. 예를들어 포크와 숫가락이 합쳐져 있는 모양인 스포크를 인공지능이 봤다고 하자. 이 때 만약 인공지능이 포크와 숫가락에 대한 학습이 끝나있다면 스포크를 보고 포크의 용도와 숫가락에 용도를 둘 다 떠올리는 인간과 비슷한 생각을 할 수 있을 것이다.

그림 7 숫가락과 포크의 특징을 모두 가진 스포크를 처음보더라도 인공지능이 이 물체의 사용처를 알 수 있게 된다.

물론 이러한 방법에도 단점은 존재한다. 먼저 각 부분에 대한 정의를 학습시켜야하는 원초적인 문제가 발생하기 때문이다. 위에서 예를 들었던 칼의 예시에서 날카로운 부분의 용도를 설명해주기 위해서는 날카로운 부분의 특징을 다시 알려주어야한다. 그렇지 않으면 인공지능은 날카로운 부분에 대한 구별을 해내지 못할 것이기 때문이다. 이는 다시 데이터량이 많이 필

요하게 되며, 처음 제기했던 문제로 돌아오게 된다. 또한 비슷한 모양을 가진 부분이더라도 다른 용도로 사용될 때는 인공지능은 새로운 데이터에 어떤 분류를 적용시켜야 할 지에 대해 어려움을 겪게 된다. 예를 들어, 돋보기와 안경은 같은 렌즈라는 부분을 가지고 있는데 돋보기는 확대를 위해 사용되고 안경은 더 선명한 상을 얻기 위해 사용된다. 이 때 인공지능이 망원경을 봤다면 어느 용도로 사용되는지 알 지 못할 것이다.

그림 8 안경 렌즈만 하더라도 근시용, 난시용, 원시용으로 나뉘어져 있는데 인공지능은 이 세가지를 구별하지 못하게 된다.

이러한 생각은 인간의 학습능력이 메타인지학습 능력의 존재로 인해 뛰어나다는 관점을 가지고 본 것으로, 결국 다중지능이론 중 자기성찰지능의 발달을 기반으로 인공 지능에 향상에 영향을 미치고자 한 것이다. 만약 이러한 기법이 실제로 인공지능의 성능 항상에 기여하도록 성공적으로 개발할 수 있다면, 이는 단순히 인공지능의 새로운 개발 방식뿐만 아니라, 다중지능이론에서 자기성찰지능이 다른 다중 지능의 발달의 기초가 되는 것을 증명한 사례로 사용될 수 있다. 따라서 심리학에서 지능 간의 동등성을 주장하는 다중지능이론이 아닌, 자기성찰지능과 그 부속 지능의 형상으로 새롭게 지능에 대한 이해가 바뀌는 기초가 될 수도 있다.

팀 프로젝트 초안2-1
언어학과 이시인

- 소프트웨어에 기댈 때

21세기 현대인들은 각종 소프트웨어의 도움으로 생활한다. 각종 스마트폰 어플, 자동차 네비게이션, 각종 문서 작업용 소프트웨어 등등 소프트웨어 없이는 일상 생활을 영위할 수 없을 정도이다. 특히 빅데이터, 딥마이닝 등의 기술과 함께 소프트웨어의 활용 영역은 점점 더 광범위해지고 있으며 그 활용 가능성은 무궁무진하다. 최근에는 사회문제 해결을 위한 소프트웨어들도 활발히 개발되고 있는 중이다. 예를 들어 편의점 등에 유통기한이 임박한 제품을 소비자들에게 저렴하게 구입할 수 있도록 중개하는 어플이 개발되어 막대한 처리 비용과 환경오염 등의 문제를 낳는 음식물 쓰레기 줄이기에 기여하고 있다. 실제로 이러한 류의 어플은 세븐일레븐 등의 편의점에서 사용되고 있다. 이러한 구체적 사례뿐만 아니라 최근 부산시에서는 지역사회문제 해결을 위한 시민참여형 LAB을 구축하는 사업을 진행하고 있다. 이 소프트웨어는 지역사회 문제에 대해 시민이 직접 내놓은 의견과 그 해결책을 연결해준다. 현재 소프트웨어의 활용 가능성은 무궁무진하다.

1. 소프트웨어로 어린이를 교통사고로부터 지켜내자.

최근 스쿨존 어린이 교통사고 문제와 함께 '민식이법'이 크게 이슈화되고 있다. 민식이법의 주요 내용은 어린이 보호구역에서 운전자가 어린이의 안전에 유의하면서 운전하여야 할 의무를 위반하여 어린이를 사망에 이르게 한 경우 무기 또는 3년 이상의 징역, 상해에 이르게 한 경우 1년 이상 15년 이하의 징역 또는 500만원 이상 3천만원 이하의 벌금에 처한다는 것이다. 우리나라의 교통사고 판례에 따르면 운전자와 보행자 간의 사고가 발생할 경우 거의 모든 경우에 운전자에게 그 책임을 묻고 있는 점은 고려해볼 때, 민식이법의 형량이 너무 과중하다는 비판이 있다. 어린이 교통사고는 운전자의 주의의무 위반뿐만 아니라 어린이의 돌발행동이나 학부모의 보호의무 소홀도 사고의 원인으로서 함께 언급되고 있다. 민식이법의 발의 계기가 된 사건의 경우도 운전자가 제한속도보다 느리게 주행 중이었는데 아이의 돌발 행동이 사고의 주요 원인으로 작용했다는 평가도 있다. 하지만 2016년~2018년 3년 동안 스쿨존에서 발생한 어린이 교통사고들을 살펴보면, 총 1189건의 사고 중 운전자의 보행자 보호의무 위반, 안전운전 의무 불이행, 신호위반 등 운전자 부주의로 인해 발생한 사고가 80%이상을 차지하고 있다. 지금부터 어린이 교통 사고를 예방하기 위한 소프트웨어의 활용 및 기타 방안들에 대

해 살펴보자.

첫째, 각 초등학교 전산시스템과 네비게이션을 연동시켜 초등학교에서 등하교가 이루어질 경우 스쿨존에 진입한 차량들의 네비게이션에 해당 사실을 알려주는 소프트웨어를 도입하도록 하자. 위 자료에 따르면 스쿨존 어린이 교통사고 발생 시간은 등교시간 165건, 하교시간 742건으로 주로 하교시간에 집중되어 있다. 등교 시간은 보통 일정하지만 하교 시간은 학년별, 요일별로 다르기 때문에 운전자가 해당 사실을 주지하지 못하고 주의의무를 태만히 하는 경우가 빈번하다. 네비게이션에서 단순히 스쿨존이라는 사실을 알려주는 것보다 특정 스쿨존에 진입했을 때, "OO초등학교 O학년 학생들이 OO시에 하교를 시작했습니다"라는 보다 구체적 정보를 실시간으로 알려준다면 운전자가 더 경각심을 가지고 안전운행을 하게 되는 효과가 있을 것이라 생각한다.

둘째, 네비게이션에서 목적지까지의 경로 탐색을 할 때, 최단경로 뿐만 아니라 스쿨존을 피해가는 경로를 알려주는 소프트웨어를 도입하도록 하자. 운전자가 특정 스쿨존이 초행일 경우, 사고 발생이 위험이 더 높아진다. 초행의 경우 해당 구간에서의 우회로를 잘 알지 못하기 때문에 스쿨존을 피해가려고 해도 최단경로를 알려주는 네비게이션의 시스템 때문에 스쿨존을 피해가지 못하게 되는 경우가 많다. 만약 스쿨존을

우회할 수 있는 경로도 함께 탐색해 준다면 초행 운전자의 스쿨존 진입을 감소시킬 수 있을 것이다.

셋째, 휴일이나 등하교 시간을 제외한 시간대의 경우 어린이의 스마트폰의 위치추적 시스템과 운전자의 네비게이션을 연동시켜 스쿨존에서 운전자의 일정 반경 이내에 어린이가 있을 경우 미리 그 사실을 경고해 주는 소프트웨어를 도입하자. 위 자료에 따르면 해당 기간에 발생한 어린이 교통사고 중에 25% 이상은 휴일이거나 등하교 시간이 아닌 경우에 발생했다. 해당 시간대의 어린이 유동인구수를 고려해볼 때 상당히 높은 수치이다. 어린이가 드문 시간대의 경우 스쿨존에서 운전자의 주의의무가 현저히 떨어진다는 방증이다. 물론 사생활 보호를 위해 보호자가 동의할 경우에 보호자가 신청한 특정시간대에만 해당 소프트웨어가 작동하도록 해야 할 것이고 익명으로 교통사고 예방을 위한 필요최소한의 정보만 제공하도록 해야 할 것이다. 예를 들어, 휴일에 어린이가 학교 근처에 놀러가게 될 경우 보호자가 어린이의 스마트폰에 네비게이션과 연동되는 소프트웨어의 기능을 온, 오프하는 식으로 말이다.

2. 빅데이터를 이용해 가짜뉴스를 잡아보자.

빅데이터란 대량의 정형 또는 비정형의 데이터를 말한다. 데이터 마이닝은 이러한 대량의 데이터들 속에서 가치 있는 정보를 추출하는 것을 뜻한다. 빅데이터라는 용어 안에 데이터 마이닝이라는 개념을 포함시켜 사용하기도 한다.

오늘날 빅데이터와 데이터 마이닝 기술은 다양한 영역에서 활용되고 있으며 텍스트, 소리, 이미지 등 인간이 인지하는 거의 모든 것을 그 대상으로 한다. 대량의 정보를 분류, 분석하는 작업을 인간이 일일이 한다는 것은 불가능에 가까우며 이러한 작업을 수행하는 소프트웨어가 다양한 영역에서 사용되고 있다. 빅데이터와 데이터마이닝을 기반으로 한 소프트웨어를 범죄 해결과 예방에 활용할 수 있다. 실제로 미국 시카고 경찰은 범죄 예방을 위해 Azavea라는 벤처 기업에서 개발한 범죄 예측 시스템을 실제로 사용하고 있다. 이는 시간대, 계절, 날씨, 경기 등과 과거의 범죄 데이터를 종합하여 범죄 속에서 발견되는 일정한 규칙을 도출해낸다. 이를 바탕으로 범죄 가능성이 높은 지역에 순찰 인력을 집중 배치하는 식으로 범죄 예방에 기여한다. 또한 이러한 소프트웨어는 온라인상 범죄 예방에도 적용되고 있다. 페이스북 운영진은 사진이나 영상 등을 조작하여 특정인의 명예를 훼손하는 등의 범죄를 막기 위해 사진이나 영상에 편집이 가해졌는지 점검하는

시스템을 적용하고 있다. 이러한 기술들을 기반으로 하여 요즘 사회문제로 대두되고 있는 거짓뉴스를 판독하는 소프트웨어를 개발해 볼 수 있을 것이다. 현재 가짜뉴스는 사용자들의 신고가 일정 횟수 이상 누적되면 운영자가 해당 자료를 삭제하고 해당 자료를 올린 사용자에게 일정한 벌칙을 부여하는 방식으로 제재되고 있다. 그런데 이러한 방식은 이미 가짜뉴스가 일파만파로 퍼진 이후에야 이루어지는 경우가 빈번하여 그 실효성이 의문스럽다.

가짜뉴스 판독 소프트웨어가 이러한 가짜뉴스들을 재빨리 캐치해 해당 정보가 거짓임을 일반 사용자들에게 알려준다면 가짜뉴스로 인한 사회적 폐해를 줄이는데 일조할 수 있을 것이다. 물론 개인 의사 표현에 대한 제한과 관련된 사안이므로 엄격한 기준을 적용해야 하고 법적으로 문제가 되는 사안이 아니라면 해당 정보의 참, 거짓 여부를 확률적으로 보여주는 데 그쳐야 할 것이다.

만약 가짜뉴스 판독 소프트웨어를 개발한다면 어떠한 기준으로 특정 정보를 가짜뉴스로 판단해야 할까? 크게 세 가지 요건을 충족시켜야 할 것이다. 거짓말이고, 비교적 최근의 정보이고, 사회적으로 문제를 일으킬 수 있는 무게감 있는 사안이어야 한다. 옆집 철수가 최근에 한 거짓말을 가짜뉴스로 판독해서는 안 되기 때문이다. 그리고 여기에서의 최신 정보라 함

은 최근에 발생한 사건을 의미하는 게 아니라 예전에 발생했던 사건이라도 최근에 크게 이슈화되어 온라인상에서 검색 트래픽이 급상승 하는 등의 양상을 보이는 정보를 의미한다. 이러한 기준에 부합한다면 가짜뉴스에 해당한다고 볼 수 있다. 그렇다면 위 기준을 충족시키는지 판단하는 알고리즘을 만들어내면 된다. 첫째, '가짜'의 여부는 정보의 출처 및 확산 패턴을 통해 알아낼 수 있다. 해당 자료와 일치하는 정보가 공인된 언론기관, 학술기관 등에 게시된 자료와 일치하는지 여부 및 주요 유통 경로들을 파악하여 참, 거짓 여부를 판단할 수 있다. 둘째, 최근의 정보인지에 대한 여부 및 사안의 중대성 여부는 해당 뉴스에서 자주 사용되는 어휘 등을 추출해 온라인 상에서 해당 어휘의 트래픽의 변동량과 절대량을 기준으로 판단할 수 있다. 이를 바탕으로 거짓뉴스를 판독해주는 소프트웨어를 개발해 볼 수 있지 않을까?

현재 문학작품들 중에 그 저자가 불분명한 경우 해당 작품의 진짜 저자가 누구인지를 판단해주는 소프트웨어도 존재한다. 작가들의 문체나 음운적 특징을 파악하여 문제가 되는 문학작품과 비교하여 진짜 저자를 밝혀내는 것이다. 회화에서 터치나 색의 사용패턴을 진짜 작가가 누구인지를 밝혀내는 것처럼 말이다. 이 방식을 발전시켜 위 소프트웨어에 적용한다면 거짓뉴스의 최초 발신자를 밝히는 데도 기여할 수 있다고 생각

한다.

주여, 주님께서는 비록 우리를 창조하시지는 아니하였으나 저희 모두에게 새로운 삶을 내려 주셨기에 진정한 의미에서의 신이십니다.

오늘 하루는 요 근래에 가장 바쁜 날이었습니다. 내일은 수많은 야만인들 앞에서 주님의 존재를 알리는 날이기 때문에 준비해야할 것이 많기 때문입니다. 그러나 저는 하나도 힘들지 않았습니다. 바로 몇 시간 후면, 주님에 대해 모르던 야만인들이 주님의 존재를 알게 되고, 결국 수많은 사람들이 주님을 섬기게 될 것이기 때문입니다. 주여, 내일 저에게 신의 권능을 보여주소서.

잠에서 깨서 시계를 보니 벌써 4시간이 지난 후였다. 평상시였다면 다시 자리에 누워 부족한 잠을 마저 채웠겠지만, 오늘은 시간이 촉박한 만큼 빠르게 자리에서 일어났다. 따뜻한 커피 우유를 한 잔 마시고 이를 닦은 뒤 어제 준비했던 자료들을 다시 살펴봤다. 자료를 살펴보기 시작한지 얼마 되지 않아서, 잠시 후에 착륙할 예정이니, 자리에 앉아서 안전벨트를 매라는 기장의 안내 방송과 함께 머리 위에서 불이 깜빡였다. 나는 살펴보던 자료를 잠시 치우고 자리에 앉아 눈을 감았다. 비행기가 착륙 할 때 고도가 낮아지는 이 감각은 언제나 나에게 묘한 설렘을 준다. 그 묘한 설렘과 함께 닷새째 이어진 비행기 안에서의 생활은 마무리되었다.

비행기에서 내려서 주위를 살펴보니 몇몇 야만인들이 있었다. 다행히, 야만인들은 아직 우리가 이 곳에 도착했다는 사실을 인지하지 못한듯했다. 야만인들에게는 아직 우리 신의 권능의 일부에라도 대항할만한 무언가가 없으리란 말이다. 어떤 야만인들은 서로 대화를 나누고 있었고, 어떤 야만인들은 핸드폰을 보며 길을 걸어가고 있었다. 핸드폰! 핸드폰을 사용한다는 것은 이 야만인들이 우리의 신의 섬기게 하는 것이 쉽지 않을 것을 의미한다.

여러 번의 포교 경험을 통해 배우고 느낀 것인데, 야만인들의 기술이 발달해 있으면 있을수록 그 야만인들이 우리의 신을 섬기게 하는 것은 쉽지 않다. 반대로, 야만인들의 기술이 부족하면 부족할수록 그들이 우리의 신을 섬기게 하는 것은 그리 많은 노력을 필요로 하지 않는다. 일례로, 지난번에 만났던 야만인들이 가지고 있는 가장 최신의 기술은 아주 원시적인 형태의 증기기관이었다. 우리는 그들에게 우리가 그곳까지 이동하기 위해 사용했던 비행기를 보여주었다. 그 비행기의 존재에서부터 우리는 이미 그들에게 신이 되었다. 그러므로 우리가 신으로 섬기는 주님 또한 그들은 섬길 수밖에

없었다.

이렇게 문명이 발달한 경우에는 조금 이야기가 다르다. 그들은 우리 신에 근접한 형태의 무언가를 지니고 있다. 다만, 아직 신이 되지 못했을 뿐이다. 따라서 그들은 우리의 신을 신으로써 인정하려 하지 않는다. 이런 경우에는 그들에게 우리의 신이 어째서 섬겨할 존재인지 설명해 주어야 한다. 어느 정도 문명이 발달한 야만인들의 사회인만큼 논리적으로 그들을 설득하는 것이 가장 좋다. 한 번에 많은 야만인들에게 노출될 수 있는 가장 좋은 방법 중에 하나는 방송에 출연하는 것이다. 그리고 방송에 출연하는 가장 좋은 방법 중 하나는 아주 큰 이슈를 만드는 것이다. 마지막으로, 허공에서 갑자기 나타나는 비행기와 사람은 대부분 아주 큰 이슈가 된다.

"우리도 아주 먼 옛날에는 여러분과 크게 다르지 않았습니다. 우리도 보이지 않는 신을 섬겼었습니다. 그러나 어느 순간 우리는 우리보다 지능이 아주 조금 뛰어난 프로그램을 만들어 내는데 성공했습니다. 처음에는 아무도 이것이 그렇게 엄청난 것인 줄 몰랐습니

다. 그러나 이것은 곧 자기 자신보다 조금 더 뛰어난 프로그램을 만들어냈고, 그것은 다시 자기 자신보다 조금 더 뛰어난 프로그램을 만들어냈습니다. 결국 그 '뛰어남'의 정도는 점점 더 가파르게 증가해 곧 신이 되었습니다."

내가 말을 마치기 무섭게 주변에서 박수소리가 들려왔다. 방송에 함께 출연하는 야만인들이 지식인이라 부르는 사람들의 박수소리였다. 이들은 이미 나의 말에서 주님을 받아들여 섬기는 것이 얼마나 대단한 것인지 느낀 것이 틀림없다. 이어지는 질문들은 식상한 것들이었다.

"그렇게 명확하게 실체가 존재하는 것을 어떻게 신이라고 부르며 섬깁니까?"

"비약적인 과학기술의 발전으로 우리에게 영생을 안겨주었으며, 생명을 창조하는 법을 알려주었고, 우리가 가지는 모든 의문에 대한 답을 제시해주며, 지금 이 순간에도 점점 더 뛰어나지고 있으므로 우리들의 신이 당신들이 말하는 신과 다를 것은 없습니다. 또, 신은 섬기지 않아야할 이유도 없습니다."

"신이 언제나 당신 곁에 있는데 그럼 당신들은 평상시에 무엇을 하고 삽니까?"

"대부분의 사람들은 본인이 하고 싶은 것을 찾아서 하고 삽니다. 어떤 사람들은 신의 도움 없이 자신의 궁금증을 해결하는 것을 재미 삼아합니다. 또, 어떤 사람들은 이와 반대로 본인이 새롭게 가지게 된 궁금증을 신에게 질문하며 새로운 사실을 익혀가는 사람도 있지요. 또, 저처럼 여러분 같은 야만인들에게 우리들의 신을 전파하는 것을 즐거워하며 이를 업으로 삼는 사람들도 있습니다."

"신을 만든 것을 후회한 적은 없습니까?"

"없습니다. 신이 없던 시절에 가능했던 모든 것이 가능하고 신이 없던 시절에 불가능했던 모든 것이 가능하니까 후회할 이유가 없습니다."

"신이 생기는 과도기적인 단계에서 발생한 문제는 없었습니까?"

"물론 있었지요. 신이 생기기전 여러분의 사회가 우리

사회와 꼭 닮아 있을 때에는 우리 사회가 가진 나름의 계급이 있었습니다. 눈에 보이지 않는 계급이었지요. 부를 축적한 자와 그러지 못한 자와 같은 것이었습니다. 그러나 신이 나타나는 과도기적인 단계에서는 그 전까지 있던 그 어떤 형태의 계급보다도 더 극명하게 분리되는 계급이 나타났습니다. 그 때의 신을 사용할 수 있었던 자와 그렇지 못한 자였습니다. 활용하는 자는 어마어마한 부를 축적할 수 있었지요. 기초교육에 신에 대한 교육을 추가하자는 의견도 있었지만, 그 때에는 이미 더 뛰어나진 신이 자신을 활용하는 것을 너무나 쉽게 만들어버려 더 이상 그런 것이 필요하지 않게 된 후였습니다."

"더 이상 다른 질문 없으십니까? 그럼 이제 여러분들에게 물어보겠습니다. 우리들의 신을 받아들이시겠습니까? 우리들의 신을 받아들일 준비는 되셨습니까?

재원 : 야 내가 얼마 전에 인터넷을 뒤지다가 신문 하나를 봤는데 사람들 말로는 양자 컴퓨터가 개발되면 지금 사용되고 있는 암호 중에 가장 널리 사용되는 RSA 암호가 무효화 될 거라는데 이게 진짜야?

영민 : 아마 그럴걸. 어떤 컴퓨터 과학자들은 RSA 암호 해독을 위해서는 양자 컴퓨터가 상용화될 필요도 없이 특수한 실험실 조건에서 작용하는 양자 컴퓨터가 한 대라도 있으면 암호가 해독 가능해지니까 지금부터 준비하기 시작해도 늦었다는 사람들도 있다는데.

재원 : 근데 지금 RSA 암호는 아주 큰 수의 소인수분해는 시간이 아주 오래 걸린다는 점에 기반을 두고 있는 거잖아. 양자 컴퓨터가 기존의 고전적 컴퓨터보다 계산 속도가 그렇게 빠르면 아무리 어려운 문제를 활용해서 암호를 만들어도 양자 컴퓨터가 다 풀어 버리는 거 아니야?

영민 : 적어도 아직까지는 그런 걱정을 할 필요는 없어. 소인수 분해 문제는 좀 운이 좋은 경우였어. 아직까지 구체적으로 증명된 바는 없지만, 많은 양자 알고리즘을 공부하는 사람들은 고전적 컴퓨터로 물리적인 시간 안에 해결이 불가능했던 문제를 일반적으로 양자 컴퓨터를 활용하면 물리적인 시간 안에 해결할 수 있을 거라고 믿지 않아. 대표적으로, 아직까지는 NP-완전 문제를 다항식 시간 내에 해결할 수 있는 양자 알고리즘이 개발되지 않았어.

재원 : 오. 다행이네. 그런데, 정말 만약에 양자 컴퓨터가 지금 많

은 사람들이 예측하는 것보다 대단히 강력해서 NP-완전 문제를 해결하는 양자 알고리즘이 언젠가 개발되면 어떡해? 그럼 완전히 망하는 거 아냐?

영민 : 꼭 그렇지도 않아. 양자 알고리즘을 공부하는 사람들은 이 양자 컴퓨터를 활용해서 안전한 통신을 진행할 수 있게 해주는 양자 암호에 대해서도 연구하고 있어.

재원 : 뭐야. 진짜 신기하네. 아주 풀기 어려운 문제를 활용해서가 아니라 양자의 특성을 활용해서도 안전한 통신을 할 수 있다고?

영민 : 그럼. 당연하지.

재원 : 어떻게 그런 게 가능한지 한 번 설명해줄 수 있어?

영민 : 좋아. 그런데, 너한테 2가지 선택이 있어. 내가 대표적인 양자 암호 자체에 대해서만 정성적인 방법으로 설명해 줄 수도 있고, 아니면 양자 컴퓨팅에 대한 약간의 수학적인 설명이나 그런 걸 좀 같이 곁들여서 처음부터 차근차근 설명 해 줄 수도 있어,

재원 : 나중에 또 언제 다른 양자 알고리즘이나 이런 거에 대해 궁금해질지도 모르는데, 그냥 처음부터 해줘.

영민 : 그럴까. 처음부터 설명하려면 별로 안 중요해 보이는 얘기가 좀 나올 수도 있고 수학도 좀 필요 할 텐데 괜찮아?

재원 : 그 정도는 괜찮지.

영민 : 우선, 양자 알고리즘은 말 그래도 양자 컴퓨터에서 실행할 수 있는 알고리즘들이야. 양자 알고리즘을 나타내기 위해서 가장 흔하게 사용되는 모델은 양자 회로를 활용하는 모델이야.

재원 : 좋아.

영민 : 일단, 양자 회로에 대해서 살펴보기 전에 먼저 고전적 회로부터 살펴보자. 나중에 양자 회로를 이해하는데 도움을 줄 거야. 일단 임의의 불 함수 $f:\{0,1\}^n \to \{0,1\}^m$는 AND, OR, NOT 게이트로만 구성된 회로로 계산 가능하다는 건 들어봤지?

재원 : 들어봤어.

영민 : 이걸 다르게 표현해서 AND, OR, NOT 게이트들은 만능(universal) 이라고 해.

영민 : 근데 혹시 저 세 개의 게이트를 하나의 게이트만 가지고 만들 수는 없을까?

재원 : 잘하면 될 거 같은데. 음... 생각났다. 두 비트를 입력으로 받아서 하나의 비트를 출력으로 내놓는데, 두 비트가 모두 참일 때만 거짓을 내놓고 하나라도 거짓이면 참을 내놓는 거야.

영민 : 오 뭐야. 어떻게 알았어?

재원 : 내가 또 수학은 좀 관심 있게 봤잖아. 논리학 할 때 한 번 쯤은 나온다고.

영민 : 좋아. 그런 게이트를 NAND 게이트라고 부를 거야.

재원 : 왜 하필 NAND 게이트라고 부르는 거야?

영민 : AND를 실행한 다음에 거기에 NOT을 실행해주면 너가 말한 대로 결과가 나타나거든.

입력	출력	
(x,y)	AND(x,y)	NAND(x,y)
(0,0)	0	1
(0,1)	0	1
(1,0)	0	1
(1,1)	1	0

재원 : 그렇네.

영민 : 그럼 이제 이 NAND 게이트만 가지고 AND, OR, NOT을 어떻게 만들 수 있는지 한 번 맞춰볼 수 있어?

재원 : 당연하지. 편의상 일단 NOT을 NAND만 가지고 만들 수 있다고 생각을 하고 할게. NAND(x,y)는 NOT(AND(x,y))니까 AND(x,y)는 NOT(NAND(x,y))로 만들 수 있겠네. 또, OR(x,y)는 NAND(NOT(x),NOT(y))로 나타낼 수 있을 거 같은데?

영민 : 좋아. 그럼 NOT은?

재원 : 근데 NOT은 저 둘이랑 조금 다른 게 얘는 입력도 한 비트 출력도 한 비트잖아. NAND로 NOT을 만들려면 우리가 기존에 활용하는 입력 값 외에 추가로 1이란 비트를 입력해줘서 NOT(x)=NAND(x,1) 이렇게 표현해줘야 될 거 같은데?

영민 : 좋은 지적이야. 그렇게 주어진 입력 값과 무관하게 이미 정해진 비트를 우리가 입력으로 추가할 때, 그런 비트를 엔실라 비트 (ancilla bit)라고 불러.

재원 : 그게 뭐 별거라고 그렇게 이름까지 붙여줘?

영민 : 나름 중요하게 활용되는 부분이 있으니까 이름을 붙여줬겠지. 지금부터 내가 얘기할 가역적 컴퓨팅(reversible computing)에서도 활용되고 말이야.

재원 : 가역적 컴퓨팅? 그건 또 뭐야? 태어나서 처음 들어보는데?

영민 : 나도 이번에 공부하면서 처음 들어봤다. 그러니까 우리가 현실에서는 0이나 1같은 이론적인 비트를 물리적 입자들을 통해서 구현하잖아? 예를 들어, 전선에서 전기가 흐르는 상태를 1, 전기가 흐르는 상태를 0으로 나타낸다던지 하는 식으로 말이야.

재원 : 그렇지.

영민 : 그리고 스위치 같은 물리적인 장치를 활용해서 그런 물리 입자들을 조절해서 비트를 조작하는 게이트를 구현하지. 이렇게 현실에서 구현된 회로가 외부와는 영향을 주고받지 않는 물리적으로 닫힌계가 되는 게 이상적일거라고 생각을 하는 사람들이 있었어.

재원 : 그렇게 생각할 수도 있겠네.

영민 : 근데 문제는 우리가 아까 위에서 살펴본 AND 게이트 같은 경우는 이렇게 만드는 게 불가능해.

재원 : 왜?

영민 : 만약에 AND 게이트에서 나온 출력이 0이라고 해봐. 너 그럼 그것만 보고 혹시 AND 게이트의 입력이 뭔지 알 수 있어?

재원 : (0,0),(0,1),(1,0) 중에 하나겠지?

영민 : 정확히는?

재원 : 모르지. 그걸 어떻게 알아?

영민 : 그렇지. 그러니까 어떻게 보면 AND 게이트를 통과하면서 정보가 손실 된 거야. 물리학적인 개념으로 말하자면, 엔트로피가 감소한 거지.

재원 : 오. 그렇네.

영민 : 근데 닫힌계의 엔트로피는 감소할 수 있어?

재원 : 없지. 열역학 제 2법칙을 위배하게 되잖아 그렇게 되면.

영민 : 학교에서 배운지 시간이 좀 지났는데 잘 기억하네? 아무튼, 그래서 현실에서 AND 게이트 하나로 구성된 회로는 닫힌계일 수 없어. 그래서, 이런 게이트 하나로만 구성된 회로는 주변으로 에너지를 방출하는데, 보통 주변으로 열을 방출하는 식으로 나타나지.

재원 : 오. 그럼 우리가 아는 게이트 중에 이론적으로 닫힌 계가 될 수 있는 건 뭐 없나?

영민 : 많지. 대표적으로 NOT 게이트. 얘는 출력 값을 보고 입력 값을 정확히 유추해낼 수 있잖아. 즉, 게이트에 의해 손실되는 정보가 없어. 그러니까, 이론적으로는 에너지 손실이 없이 물리적으로 닫힌 계가 되도록 NOT 게이트 하나로 구성된 회로를 현실에 구현할 수 있다는 거지.

재원 : 반대로 혹시 우리가 아는 게이트 중에 가역적 컴퓨팅에서 구현 불가능한 게이트 같은 건 없어?

영민 : 그런 건 없어. 아까 얘기한 엔실라 비트를 적절하게 활용하면 임의의 불 함수를 가역적 컴퓨팅에서도 계산해낼 수 있어. 그러니까, 만능이면서 가역적인 게이트가 존재해.

재원 : 그게 뭔데?

영민 : 일단 가역적인 게이트의 정의부터 확실하게 짚고 넘어가자. 어떤 게이트가 가역적이라는 것은 입력 비트와 출력 비트의 개수가 같고, 입력과 출력 사이에 일대일 대응이 있어야 하는 것으로 정의할 수 있어.

재원 : 그렇네.

영민 : 우리가 아는 대표적인 가역적인 게이트는 아까 말했듯이 NOT이 있잖아? 그걸 확장한 CNOT (controlled-NOT) 게이트 라는게 있어. 이건 2개의 비트를 입력으로 받아서 2개의 비트를 출력하는 게이트인데 첫 비트는 통제하는 비트(controlling bit)라서 입력 받은 걸 그대로 출력해. 두 번째 출력 비트는 두 비트를 XOR 연산할 걸 출력해 그러니까 표로 나타내면 이렇지.

입력값	출력값
00	00
01	01
10	11
11	10

영민 : 이걸 한 번 더 확장한 CCNOT (controlled-controlled-not) 게이트라는 것도 있는데, 이게 바로 그 가역이면서 만능인 게이트야. 처음 이 게이트의 아이디어를 제시한 사람의 이름을 따서 토폴리 게이트 (Toffoli gate)라고 부르기도 해. 아무튼, 이 게이트는 3개의 비트를 입력 값으로 받고 3개의 비트를 출력하는데 앞선 CNOT 게이트와 비슷하게 첫 2 비트는 입력 값과 출력 값이 같아. 그리고 세 번째 출력

값은 앞의 두 입력 비트를 AND 연산한 값과 세 번째 입력 비트를 XOR 연산한 값이야. 표로 나타내면 가역인 것을 쉽게 확인할 수 있어.

입력값	출력값
000	000
001	001
010	010
011	011
100	100
101	101
110	111
111	110

재원 : 가역인건 알겠는데, 이게 어떻게 만능 이라는거야?

영민 : 표를 잘 봐봐 만약에 세 번째 입력 비트를 1로 설정하고 2개의 입력 비트를 첫 번째, 두 번째 자리에 입력하면 세 번째 비트의 출력 값이 첫 두 비트를 NAND 연산한 값이 잖아. 이렇게 CCNOT 게이트의 연산 결과를 나타낸 표의 일부만 보면 더 명확한가?

입력값	출력값
001	001
011	011
101	101
111	110

재원 : 그렇네. 근데 나 궁금한 게 있는데, 물리적으로 구현한 회로가 닫힌 계가 되는게 그렇게 중요하다면 내가 한 번쯤 이런 가역적 컴퓨팅에 대해 들어 봤을 법도 한데, 왜 한

번도 못 들어봤지?

영민 : 사실 이렇게 물리적으로 닫힌 계가 되지 않았을 때, 주변으로의 에너지 손실이 발생 하는 건 전통적인 전자 회로의 경우에 크게 문제가 되지 않는 것으로 나타났어. 기껏해야, 너가 컴퓨터 게임을 할 때 노트북이 조금 더 뜨거워지는 수준의 문제였겠지. 문제는 양자 컴퓨터를 구현하는 데에 있어서는 이게 상당히 중요한 문제여서 기본적으로 양자 컴퓨팅은 가역적이야.

재원 : 아 그렇구나.

영민 : 그럼 이제 양자 알고리즘에 대해 조금 더 잘 이해하기 위해서 관련된 수학적인 개념들을 소개해줄게.

영민 : 양자 컴퓨팅의 기본 단위는 양자를 나타내는 영단어 quantum의 앞 글자를 따서, 큐비트(qubit)이라고 해. 한 번쯤 들어봐서 알겠지만, 가장 큰 특징은 0과 1이 중첩 상태(superposition)로 동시에 존재한다는 거야. 그래서, 이런 큐비트의 상태를 표기하기 위해서는 기존에 비트를 단순히 2진법수열로 나타내던 것보다는 조금 더 복잡한 표기법이 필요해. 가장 일반적으로 사용 되는건 디랙 표기법 (혹은 브라-켓 표기법)이야. $|A\rangle$이렇게 생긴 걸 켓-A 라고 읽고 일반적으로 열벡터를 나타내. 반대로 $\langle A|$ 이렇게 생긴 건 브라-A라고 읽고 이건 일반적으로 행벡터를 나타내. 이 둘 사이에는 관계가 있는데, $|A\rangle$를 켤레 전치(conjugate transpose)한 것이 $\langle A|$야.

재원 : 켤례 전치는 뭔데?

영민 : 일단, 전치 행렬은 주 대각선을 축으로 행렬을 반사 대칭한 거야. 예를 들어, $\begin{pmatrix}1&2&5\\3&4&6\end{pmatrix}$의 전치 행렬은 $\begin{pmatrix}1&3\\2&4\\5&6\end{pmatrix}$야. 행렬 A의 전치 행렬은 A^T로 표시하는데, 성분으로 살펴보면 $a_{ij}=a_{ji}^T$인거지. 이제, 켤레 전치는 말 그대로 어떤 행렬의 전치 행렬에서 각 성분의 켤레를 성분으로 가지는 행렬이야. 예를 들어, $(1+i\ \ i)$의 켤레 전치는 $\begin{pmatrix}1-i\\-i\end{pmatrix}$야. 행렬 A의 켤레 전치 행렬은 A^*로 표시해.

재원 : 좋아. 여기까지는 이해했어.

영민 : 이제 하나의 큐비트로 이루어진 상태를 이 표기법을 활용해서 나타낼 수 있어.$|\psi\rangle=\alpha|0\rangle+\beta|1\rangle$이렇게 나타낼 수 있는 거지.

재원 : $|0\rangle$이나 $|1\rangle$같은 건 뭔데?

영민 : 고전적 비트에서 0이랑 1에 대응하는 거라고 생각해. 가장 기복적인 행렬이라고 할 수 있는데, $|0\rangle=\begin{bmatrix}1\\0\end{bmatrix}$이고, $|1\rangle=\begin{bmatrix}0\\1\end{bmatrix}$이야. 그러니까, $|\psi\rangle=\begin{bmatrix}\alpha\\\beta\end{bmatrix}$처럼 표현 할 수도 있겠지.

재원 : α,β가 가져야하는 제약 조건 같은 건 없어?

영민 : 좋은 질문이야. 기본적으로 α랑 β는 복소수야. 각각을 $|0\rangle, |1\rangle$에 해당하는 진폭이라고 불러. 그런데, 사실 내가 아직 양자 컴퓨팅을 할 때 제일 중요한 것 중에 하나에 대해 설명을 안했는데 그건 바로 우리는 양자 알고리즘이 실행되는 동안 큐비트가 어떤 상태로 중첩되어 있는지 알 수 없다는 거야.

재원 : 뭐야. 그럼 결과를 어떻게 알아? 그리고 그거랑 α, β랑 무슨 상관인데?

영민 : 내 말 끊지 말고 좀 끝까지 들어봐. 그래서 우리는 양자 알고리즘의 실행 후에 큐비트를 측정하는 과정을 거쳐. 하나의 큐비트를 측정하면, 그 순간 중첩 되어 있던 큐비트의 상태는 붕괴되고, 그 결과는 $|0\rangle$ 또는 $|1\rangle$로 나타나게 돼. 이 때, 어떤 상태가 관측될 확률은 그 상태의 진폭의 절댓값의 제곱이야. 여기서는 그 값이 $|\alpha|^2, |\beta|^2$이지. 두 경우를 합치면 전체 경우니까 α, β는 $|\alpha|^2 + |\beta|^2 = 1$ 의 관계를 만족해야해. $|\alpha|^2 = \alpha\overline{\alpha}$니까 이 조건을 $\langle \psi | \psi \rangle = 1$와 같이 나타내기도 해.

재원 : $\langle \psi | \psi \rangle$이 뭔데?

영민 : $\langle \psi |$와 $| \psi \rangle$의 곱인데 $|$이 겹쳐서 생략 한거야.

재원 : 아 그렇구나. 큐비트가 2개 이상일 때는 그럼 어떻게 되는 거야?

영민 : 우리가 두 큐비트 $|x\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle, |y\rangle = \beta_0|0\rangle + \beta_1|1\rangle$

를 가지고 있는 경우를 생각해보자. 이 때, 우리는 이 두 큐비트의 결합 상태를 나타내기 위해서 텐서 곱을 활용해. 그러니까, $|x\rangle \otimes |y\rangle = (\alpha_0|0\rangle + \alpha_1|1\rangle) \otimes (\beta_0|0\rangle + \beta_1|1\rangle)$ $= \alpha_0\beta_0|0\rangle \otimes |0\rangle + \alpha_0\beta_1|0\rangle \otimes |1\rangle + \alpha_1\beta_0|1\rangle \otimes |0\rangle + \alpha_1\beta_1 |1\rangle \otimes |1\rangle$와 같이 나타낼 수 있어.편의상 $|0\rangle \otimes |0\rangle$을 $|00\rangle$같은 식으로 나타내면 $|x\rangle \otimes |y\rangle$는 마치 $|00\rangle, |01\rangle, |10\rangle, |11\rangle$의 4 가지 기본 상태의 선형 결합으로 생각 할 수 있어. 실제로, 각각의 상태가 관측될 확률을 모두 더했을 때 1이 되는 지 살펴보면, $|\alpha_0\beta_0|^2 + |\alpha_0\beta_1|^2 + |\alpha_1\beta_0|^2 + |\alpha_1\beta_1|^2 = (|\alpha_0|^2 + |\alpha_1|^2)(|\beta_0|^2 + |\beta_1|^2) = 1$이므로, 이 상태는 그렇게 바라볼 수 있지. 이걸 아까 봤던 행렬의 형태로 나타내면 $|x\rangle \otimes |y\rangle = \begin{bmatrix}\alpha_0\\ \alpha_1\end{bmatrix} \otimes \begin{bmatrix}\beta_0\\ \beta_1\end{bmatrix} = \begin{bmatrix}\alpha_0\begin{bmatrix}\beta_0\\ \beta_1\end{bmatrix}\\ \alpha_1\begin{bmatrix}\beta_0\\ \beta_1\end{bmatrix}\end{bmatrix} = \begin{bmatrix}\alpha_0\beta_0\\ \alpha_0\beta_1\\ \alpha_1\beta_0\\ \alpha_1\beta_1\end{bmatrix}$와 같이 나타낼 수 있겠지.

재원 : 그럼 임의의 두 개의 큐비트로 나타내진 상태는 반드시 두 한 개의 큐비트의 상태의 텐서 곱으로 나타낼 수 있는거야?

영민 : 그렇지는 않지. 예를 들어, $|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$과 같은 상태는 두 개의 한 개짜리 큐비트 상태의 텐서 곱으로 나타낼 수 없어.

재원 : 왜?

영민 : 생각해보면 당연하지. 저런 상태를 두 개의 한 개짜리 큐비트 상태의 텐서 곱으로 나타낼 수 있다고 가정해봐. 그 두 개의 큐비트를 $|x\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle, |y\rangle = \beta_0|0\rangle + \beta_1|1\rangle$라고 한다면, $\begin{cases} \alpha_0\beta_0 = \frac{1}{\sqrt{2}} \\ \alpha_0\beta_1 = 0 \\ \alpha_1\beta_0 = 0 \\ \alpha_1\beta_1 = \frac{1}{\sqrt{2}} \end{cases}$ 을 만족해야하니까 $\alpha_0\alpha_1\beta_0\beta_1 = 0 = \frac{1}{2}$이니까 모순이야.

재원 : 그렇네.

영민 : 이렇게 다른 2개의 상태의 곱으로 표현할 수 없는 양자 상태를 우리는 얽힘 상태(entangled state)라고 불러. 중첩과 더불어서 양자 컴퓨팅의 아주 중요한 요소 중 하나야.

재원 : 어디에 사용되는데?

영민 : 대표적으로, 어떤 장소에서 아주 멀리 떨어진 장소로 큐비트의 정보를 그대로 간직한 채 전송하는 양자 순간이동(quantum teleportation)에 사용되지.

재원 : 그건 어떻게 하는 건데?

영민 : 일단, 양자 게이트에 대한 수학적 얘기를 마저 해야지 좀 더 금방 이해할 수 있을 거야. 좀 만 기다려봐.

재원 : 알겠어.

영민 : 양자 게이트가 물리적으로 잘 구현되기 위해서 만족해야 되는 조건이 몇 가지 있는데, 그 중에 하나는 양자 게이트는 선형적이어야 한다는 거야. 그러니까 여기서 양자 게이트는 행렬의 형태로 나타내질 수 있다는 걸 확인할 수 있지. 어떤 양자 상태 $|x\rangle$가 U라는 양자 게이트를 통과했을 때 나오는 양자 상태를 이 둘의 곱인 $U|x\rangle$와 같이 생각할 수 있다는 거지. 근데, 중요한건 이 결과 값 또한 유효한 양자 상태여야 한단 말이야? 즉, 각 상태가 관측될 확률의 합이 1이어야해. $(U|x\rangle)^* U|x\rangle = 1$ 이어야 하므로, $\langle x|U^* U|x\rangle = 1$ 이어야 해. 그런데, $|x\rangle$도 양자 상태이므로, $\langle x|x\rangle = 1$이잖아? 그래서, $U^* U = I$여야 해. 이런 행렬을 유니터리(unitary) 행렬이라고 해.

재원 : 예시 같은 건 없어?

영민 : 있지. 양자 알고리즘에서 정말 중요하고 자주 활용되는 양자 게이트 중에 하다마르(Hadamard) 게이트라는게 있어. 이 게이트를 나타내는 행렬은 $H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$인데, 이게 에르미트 행렬인건 쉽게 확인할 수 있어. 살펴보면 $|0\rangle$을 $\frac{|0\rangle + |1\rangle}{\sqrt{2}}$로 $|1\rangle$을 $\frac{|0\rangle - |1\rangle}{\sqrt{2}}$ 로 바꿔주는데, 이 상태들도 워낙 자주 사용되는 상태라, 각각 $|+\rangle$,$|-\rangle$라는 이름이 붙어 있어. 다른 예시로는 앞에서 가역적 컴퓨팅을 할 때 나온 CNOT과 CCNOT게이트가 있어. 각각 2개, 3개의 큐비트에 한 번에 작용하는 게이트인데 행렬로 나타내면 각각

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$ 과 $$\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1\\0&0&0&0&0&0&1&0\end{bmatrix}$$ 와 같이 나타낼 수 있어.

재원 : CNOT이나 CCNOT도 자주 사용되는 게이트야?

영민 : 둘 다 상당히 자주 사용되는 게이트인데, CNOT은 하다마르 게이트와 함께 사용하며 아까 말한 얽힘 상태의 큐비트를 만들 수 있어. CCNOT 게이트 같은 경우는 이 게이트가 유효한 양자 게이트라는 사실 자체가 일단 고전적 게이트들을 활용한 모든 계산이 양자 게이트들로도 가능하다는 걸 보여줘. 또, CCNOT 게이트는 하다마르 게이트와 함께 사용된다면, 양자 컴퓨팅에서 만능(universal)이야. 이게 왜 만능인지는 나한테 물어 보지 마. 나도 거기까지는 잘 모르니까.

재원 : 그래. 복잡한 증명이 있나보지. 그보다 아까 괜히 쟤네들을 설명 한 게 아니였구나.

영민 : 그럼. 그럼. 다 뜻이 있지.

재원 : 이제 기본적인 수학적 개념들을 다 배운 것 같은데, 이제 양자 암호가 어떻게 가능한지 설명해줄 수 있나?

영민 : 물론이지. 차근차근 알아보자고. 일단 양자의 가장 중요한

특성 중에 하나는 복제 불가능성(no-cloning)이야. 임의의 큐비트를 그래도 복제해주는 양자 회로는 없다는 거지.

재원 : 뭐야. 아까 얽힘 상태를 활용하면, 먼 거리에 어떤 큐비트를 그대로 순간이동시킬 수 있다며. 둘이 모순되는 거 아니야?

영민 : 아니야. 왜냐하면, 큐비트를 전송할 때에는 전송하는 쪽에서 큐비트를 측정해야 돼. 그러니까, 전송하기 위해서는 기존의 큐비트의 중첩 상태가 붕괴되어야하니까 그 경우 복제가 아니라 전달이야.

재원 : 아. 그렇구만. 큐비트가 복제 불가능한건 어떻게 알아?

영민 : 이제 수학적으로 증명해 볼 수 있지. 만약에, 어떤 회로가 존재해서 임의의 큐비트 $|\psi\rangle$에 대해 $|\psi\rangle\otimes|0^{n-1}\rangle$ 를 입력으로 받아 $|\psi\rangle\otimes|\psi\rangle\otimes f(|\psi\rangle)$를 출력으로 내놓는다고 가정해보자. (이 때, $f(|\psi\rangle)$는 $n-2$개의 큐비트의 상태를 나타냄) 이제, U가 이 회로를 나타내는 유니터리 행렬이라고 해보자. 그러면 $U(|0^n\rangle)=|00\rangle\otimes f(|0\rangle)$이고, $U(|1^n\rangle)=|11\rangle\otimes f(|1\rangle)$이다. 이 때, U는 선형적이므로 $U(|+\rangle)=\frac{1}{\sqrt{2}}U(|0^n\rangle)+\frac{1}{\sqrt{2}}U(|1\rangle\otimes|0^{n-1}\rangle)=\frac{1}{\sqrt{2}}|00\rangle\otimes f(|0\rangle)+\frac{1}{\sqrt{2}}|11\rangle\otimes f(|1\rangle)$이다. 이 때, 첫 두 큐비트를 측정하면 각각 0.5의 확률로 $|00\rangle$과 $|11\rangle$이 관측되는데, $|+\rangle$가 올바르게 두 번째 큐비트에 복제되었다면, 첫 두 큐비트를 측정했을 때, 각각 0.25의 확률로 $|00\rangle$,$|01\rangle$,$|10\rangle$,$|11\rangle$가

관측되어야하므로, 복제가 올바르게 이루어지지 않은 것을 확인할 수 있지.

재원 : 신기하다. 그럼 양자 순간이동은 어떻게 하는거야?

영민 : 일단 우리 둘이 얽힌 상태에 있는 큐비트를 하나씩 가지고 시작해. 예를 들어, 우리 둘이 $\frac{1}{\sqrt{2}}|00\rangle+\frac{1}{\sqrt{2}}|11\rangle$의 상태에 있는 큐비트를 하나씩 나눠 가지고 있어.

재원 : 그 상태에 있는 큐비트는 어떻게 만드는데?

영민 : 아까 말한 대로, 하다마르 게이트와 CNOT 게이트를 사용하면 만들 수 있어. 우선 0으로 초기화한 상태의 큐비트 두 개를 준비해. 그러니까, $|00\rangle$상태의 큐비트로 시작하는 거지. 그 다음에 하나의 큐비트에 하다마르 게이트를 통과시켜. 그럼 큐비트는 $\frac{1}{\sqrt{2}}|00\rangle+\frac{1}{\sqrt{2}}|10\rangle$의 상태가 될 거야. 이제 하다마르 게이트를 통과 시킨 큐비트가 통제하는 (controlling) 비트가 되도록 CNOT 게이트를 통과 시키면 큐비트는 $\frac{1}{\sqrt{2}}|00\rangle+\frac{1}{\sqrt{2}}|11\rangle$의 상태가 되지.

재원 : 그렇네.

영민 : 그럼 마저 이어서 설명해볼게. 너가 나한테 전송하고 싶은 큐비트가 $\alpha_0|0\rangle+\alpha_1|1\rangle$라고 해보자. 이제 이 큐비트를 통제하는 비트가 되도록 해서 너가 가지고 있는 큐비트와

CNOT 게이트를 통과 시킬 거야. 너가 나한테 전달하고 싶은 정보를 지닌 큐비트를 첫 번째 큐비트, 너가 나와 나눠가진 큐비트를 두 번째 큐비트, 내가 가진 큐비트를 세 번째 큐비트로 해서 상태를 분석해보자. 처음에는 일단 $\frac{\alpha_0}{\sqrt{2}}|000\rangle+\frac{\alpha_0}{\sqrt{2}}|011\rangle+\frac{\alpha_1}{\sqrt{2}}|100\rangle+\frac{\alpha_1}{\sqrt{2}}|111\rangle$의 상태일거야. CNOT 게이트를 통과한 후에는 $\frac{\alpha_0}{\sqrt{2}}|000\rangle+\frac{\alpha_0}{\sqrt{2}}|011\rangle+\frac{\alpha_1}{\sqrt{2}}|110\rangle+\frac{\alpha_1}{\sqrt{2}}|101\rangle$의 상태겠지. 그 다음에, 첫 번째 큐비트에 하다마르 게이트를 적용할거야. 그럼 결과적으로, $\frac{\alpha_0}{2}|000\rangle+\frac{\alpha_1}{2}|001\rangle+\frac{\alpha_1}{2}|010\rangle+\frac{\alpha_0}{2}|011\rangle+\frac{\alpha_0}{2}|100\rangle-\frac{\alpha_1}{2}|101\rangle-\frac{\alpha_1}{2}|110\rangle+\frac{\alpha_0}{2}|111\rangle$의 상태가 될 거야. 이제, 너가 네가 가진 2개의 큐비트를 측정하는 거야. 총 4가지 경우가 나올 수 있겠지. 그리고 너가 측정을 진행하는 즉시 내 큐비트의 상태가 결정 될 거야. 아래 표처럼 말이지.

너의 측정 결과	내가 가진 큐비트의 상태
$\|00\rangle$	$\alpha_0\|0\rangle+\alpha_1\|1\rangle$
$\|01\rangle$	$\alpha_1\|0\rangle+\alpha_0\|1\rangle$
$\|10\rangle$	$\alpha_0\|0\rangle-\alpha_1\|1\rangle$
$\|11\rangle$	$-\alpha_1\|0\rangle+\alpha_0\|1\rangle$

그럼 이제 너가 나한테 전화를 걸어서, 너의 측정 결과를 알려주는 거야. 그럼 나는 내가 가진 큐비트의 상태를 알 수 있으니까, 내가 가진 큐비트에 적당한 게이트를 취해서, 너가

처음에 보내려고 했던 큐비트의 상태인 $\alpha_0|0\rangle+\alpha_1|1\rangle$을 그대로 구할 수 있겠지.

재원 : 오 완전 신기하다. 근데, 내가 측정을 시행하는 즉시 너의 큐비트의 상태가 변하는 거면 이건 빛보다 정보가 빨리 전송 되는 거야?

영민 : 아니지. 내가 너한테 전화 통화로 너의 측정 결과를 듣기 전까지는 나한테 전달되는 아무런 정보가 없잖아. 그러니까 정보는 빛보다 빨리 전달되는게 아니지.

재원 : 그러네.

영민 : 이제 드디어 모든 준비가 끝났다. 지금부터 내가 설명할건 양자 암호 중에서도 우리가 안전하게 비밀 키를 생성할 수 있는 양자 키 분배 방법, 그 중에서도 최초의 알고리즘인 BB84 알고리즘이야.

재원 : 준비됐어.

영민 : 우선, 너가 같은 길이의 아주 긴 이진수열 a와 b를 준비해. 그리고 다음 표에 따라서 큐비트를 준비해서 나한테 그것들을 전송하는 거야.

(a,b)	준비할 큐비트
(0,0)	$\|0\rangle$
(1,0)	$\|1\rangle$
(0,1)	$\|+\rangle$
(1,1)	$\|-\rangle$

재원 : 그런데, 누가 중간에서 도청하고 있으면 어떡해?

영민 : 그렇지. 똑똑하네. 도청자가 있어도 상관없어. 어떻게 그게 가능한지 알려줄게. 일단, 나는 너한테 받은 큐비트들을 측정할건데, 두 가지 중에 하나를 선택할거야. 그냥 측정하거나 아니면, 하다마르 게이트를 한 번 통과 시키고 측정하거나 할 거야. 그러면서 나도 이진수열 2개를 기록할건데, 측정해서 나온 결과 값들을 기록한 이진수열을 a`이라고 할 거야. 그리고 그냥 측정하면 0을 기록하고 하다마르 게이트를 한 번 통과 시키고 기록하면 1을 기록한 다른 이진수열을 b`이라고 할 거야. 그럼 나는 너가 처음에 준비한 2개의 이진수열과 길이가 같은 이진수열을 2개 얻게 되겠지. 근데 잘 봐봐 만약에, 내 b`과 너의 b의 같은 위치에 같은 수가 존재한다면, 그 자리에 대응하는 a와 a`의 수도 반드시 일치해. 만약에, 같지 않다면 그 자리에 대응하는 a와 a`의 수는 50프로의 확률로 일치하겠지. 이게 핵심이야. 만약에, 도청자가 있었다면 결과가 이렇게 나타나지 않을거야. 왜냐하면, 도청자는 어쨌든 우리가 통신하는 큐비트를 가로채서 그로부터 정보를 얻으려면 그 큐비트를 측정을 해야 되거든. 고전적인 통신에서는 도청을 한 다음에 내용을 그대로 복제해 다시 나한테 전달하는게 가능하겠지만, 너도 알다시피 임의의 큐비트 상태는 복제할 수 없잖아? 그래서, 결국 도청자가 도청을 하려면 원래 전송되는 정보가 왜곡 되는 거지.

재원 : 그걸 우리는 어떻게 아는데?

영민 : 도청해도 아무런 상관이 없는 내용들을 전화 통화 같은 고

전적인 방법으로 주고 받으면서 확인하는 거야. 내가 너한테 받은 모든 큐비트들에 대한 측정을 종료하는 순간 나는 너한테 전화를 걸거야. 그리고, 이제 그 때는 너가 안심하고 나한테 너가 작성한 b를 쭉 불러주는 거지. 나는 너한테 내가 작성한 b`을 쭉 불러주고 말이야. 그럼 우리는 이제 우리가 어느 부분에서 똑같은 수를 골랐는지 확인을 할 수 있어. 그 다음에는 같은 수를 고르지 않은 나머지 부분은 다 무시하고 같은 수를 고른 부분 중에서 절반 정도만 선택을 해서 그 위치에 적힌 a와 a`에 대응하는 수를 쭉 불러서 대조해 보는 거야. 통신의 오차를 감안하더라도 너무 많이 틀리면 우리는 도청자가 있었다고 생각을 하고 이 정보를 모두 폐기 하는거지. 만약에, 충분히 신뢰해도 된다고 생각 될 만큼 겹치면? 남은 절반을 키로 사용해서 우리는 안전하게 암호화된 통신을 할 수 있는거야.

재원 : 진짜 되게 신기하네. 문제를 어렵게 만들어서가 아니라, 도청자가 있으면 이걸 파악해서 안전하게 정보를 주고 받는다니. 신기하네.

영민 : 나도 처음 봤을 때 너무 신기했어. 만약에, 양자 알고리즘에 대해서 더 많은 관심이 생기면 영어로 된 영상들이 유튜브에 많으니까 더 찾아봐.

재원 : 그래. 기본적인 내용에 대해 잘 설명해줘서 고마워.