

09.18 정기 OMS : 슈퍼 마리오는 왜 NP-HARD 문제일까?

01

CZ포탈

CZ 포탈 애용해주세요 ㅠ

02

OMS : 이준형 학우님
슈퍼 마리오는 왜 NP-HARD 문제일까?

NP-hard가 뭔가요?

np hard가 뭐야

◆ 생각하는 과정 표시 ▾

NP-hard는 아주 어려운 계산 문제들의 집합이라고 생각하면 쉬워. 여기서 '어렵다'는 건, 현재까지 알려진 바로는 문제를 효율적으로, 즉 빠른 시간 안에 풀 수 있는 알고리즘이 존재하지 않는다는 의미야.

더 정확히 말하면, 모든 NP 문제를 다행 시간 내에 NP-hard 문제로 변환(환원)할 수 있을 때 그 문제를 NP-hard라고 불러. 이게 좀 어려운 말인데, 쉽게 풀어서 설명해볼게.

NP-hard 쉽게 이해하기 외판원 문제 (TSP)

가장 유명한 NP-hard 문제 중 하나인 **외판원 문제(Traveling Salesperson Problem, TSP)**를 예로 들어보자.

- **문제:** 한 외판원이 여러 도시를 방문해야 해. 각 도시를 딱 한 번씩만 방문하고 처음 출발했던 도시로 돌아와야 할 때, 가장 짧은 이동 경로는 뭘까?

도시가 3~4개일 때는 모든 경우의 수를 직접 계산해서 금방 답을 찾을 수 있어. 하지만 도시의 수가 20개, 30개로 늘어나면 경우의 수가 기하급수적으로 폭발해서 슈퍼컴퓨터로도 현실적인 시간 안에 완벽한 최단 경로를 찾는 게 거의 불가능해져.

NP-hard 문제는 바로 이런 특징을 가지고 있어. 문제의 크기가 조금만 커져도 계산량이 엄청나게 늘어나서 정답을 찾는 데 천문학적인 시간이 걸릴 수 있다는 거야.

시간 복잡도



시간 복잡도

- 최상의 경우 : 오메가 표기법 (Big-Ω Notation)
- 평균의 경우 : 세타 표기법 (Big-Θ Notation)
- 최악의 경우 : 빅오 표기법 (Big-O Notation)

시간 복잡도

coupang



시간 복잡도

coupang



늦어도 모레까지는
도착합니다^^

시간 복잡도

$O(1)$ – 상수 시간 : 문제를 해결하는데 오직 한 단계만 처리함.

$O(\log n)$ – 로그 시간 : 문제를 해결하는데 필요한 단계들이 연산마다 특정 요인에 의해 줄어듬.

$O(n)$ – 직선적 시간 : 문제를 해결하기 위한 단계의 수와 입력값 n 이 1:1 관계를 가짐.

$O(n \log n)$: 문제를 해결하기 위한 단계의 수가 $N*(\log 2N)$ 번만큼의 수행시간을 가진다. (선형로그형)

$O(n^2)$ – 2차 시간 : 문제를 해결하기 위한 단계의 수는 입력값 n 의 제곱.

$O(C^n)$ – 지수 시간 : 문제를 해결하기 위한 단계의 수는 주어진 상수값 C 의 n 제곱.

예시 - 정렬 알고리즘



그럼 np - hard란?



그럼 np - hard란?

답을 찾는 건 오래 걸릴 수 있지만, 누군가 답을 주면 '검증하는 것은 빠른 시간 안에 가능한' 문제

스도쿠 비유:

답 찾기 (어려움): 빈칸의 모든 숫자 조합을 시도하려면 아주 오래 걸립니다.

답 검증 (쉬움): 채워진 답안지를 보고 가로, 세로, 네모 규칙에 맞는지 확인하는 것은 금방 끝납니다.

P-NP



마리오는 왜 np-hard 일까?



내가 버섯이나 먹을 때는
만만해보였지

환원(Reduction) - 3 SAT Problem

"까다로운 룸메이트" 비유로 이해하기

오늘 저녁에 룸메이트와 무엇을 할지 정한다고 상상해 보세요.

- **변수 (Variables):** 우리가 할 수 있는 활동들입니다. 각 활동은 하거나(True), 안 하거나(False) 둘 중 하나입니다.
 - X_1 : 영화를 본다.
 - X_2 : 피자를 시킨다.
 - X_3 : 보드게임을 한다.
 - ... 등등
- **룸메이트의 요구사항 목록 (Clauses):** 룸메이트는 여러 개의 요구사항을 내깁니다. 이 모든 요구사항을 동시에 만족시켜야만 룸메이트는 행복합니다. (모든 요구사항은 AND로 연결됩니다.)
- **각 요구사항의 내용:** 그런데 각 요구사항은 정확히 3가지 선택지로 이루어져 있고, 그중 하나만 만족하면 되는 너그러움(?)을 보입니다. (각 요구사항 내의 3가지는 OR로 연결됩니다.)

예시 상황

룸메이트가 다음과 같은 요구사항 목록을 내밀었습니다.

요구사항 1: "영화를 보거나(X_1), 피자를 시키지 않거나($\neg X_2$), 보드게임을 해야(X_3) 해. 이 셋 중 하나는 꼭 해줘!"

AND

요구사항 2: "영화를 안 보거나($\neg X_1$), 피자를 시키거나(X_2), 산책을 해야(X_4) 해. 이 셋 중 하나는 꼭 해줘!"

AND

요구사항 3: "피자를 시키지 않거나($\neg X_2$), 보드게임을 안 하거나($\neg X_3$), 산책을 안 해야 ($\neg X_4$) 해. 이 셋 중 하나는 꼭 해줘!"

... (이런 요구사항이 수십, 수백 개가 더 있습니다) ...

3-SAT 문제란 바로 이것입니다:

"이 모든 까다로운 요구사항들을 동시에 전부 만족시키는 활동 계획(각 변수에 True/False를 할당하는 조합)이 과연 존재할까요?"

환원(Reduction)

주어진 변수에 대해서, 특정 논리문을 만족시키는 변수들의 T/F 를 찾아라

$$(x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3)$$

해결책 1:

- $x_1 = \text{True}$
- $x_2 = \text{True}$
- $x_3 = \text{False}$

검증:

- 첫 번째 절: $(T \vee \neg T \vee \neg F) \rightarrow (T \vee F \vee T) \rightarrow \text{True}$
- 두 번째 절: $(\neg T \vee T \vee F) \rightarrow (F \vee T \vee F) \rightarrow \text{True}$
- 세 번째 절: $(T \vee T \vee \neg F) \rightarrow (T \vee T \vee T) \rightarrow \text{True}$

마리오에 적용해보자

각 변수를 마리오의 게임 요소 (버섯, 스타, 굼바, 낭떠러지 등등)에 매치 시켜야 함

마리오에 적용해보자

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2)$$

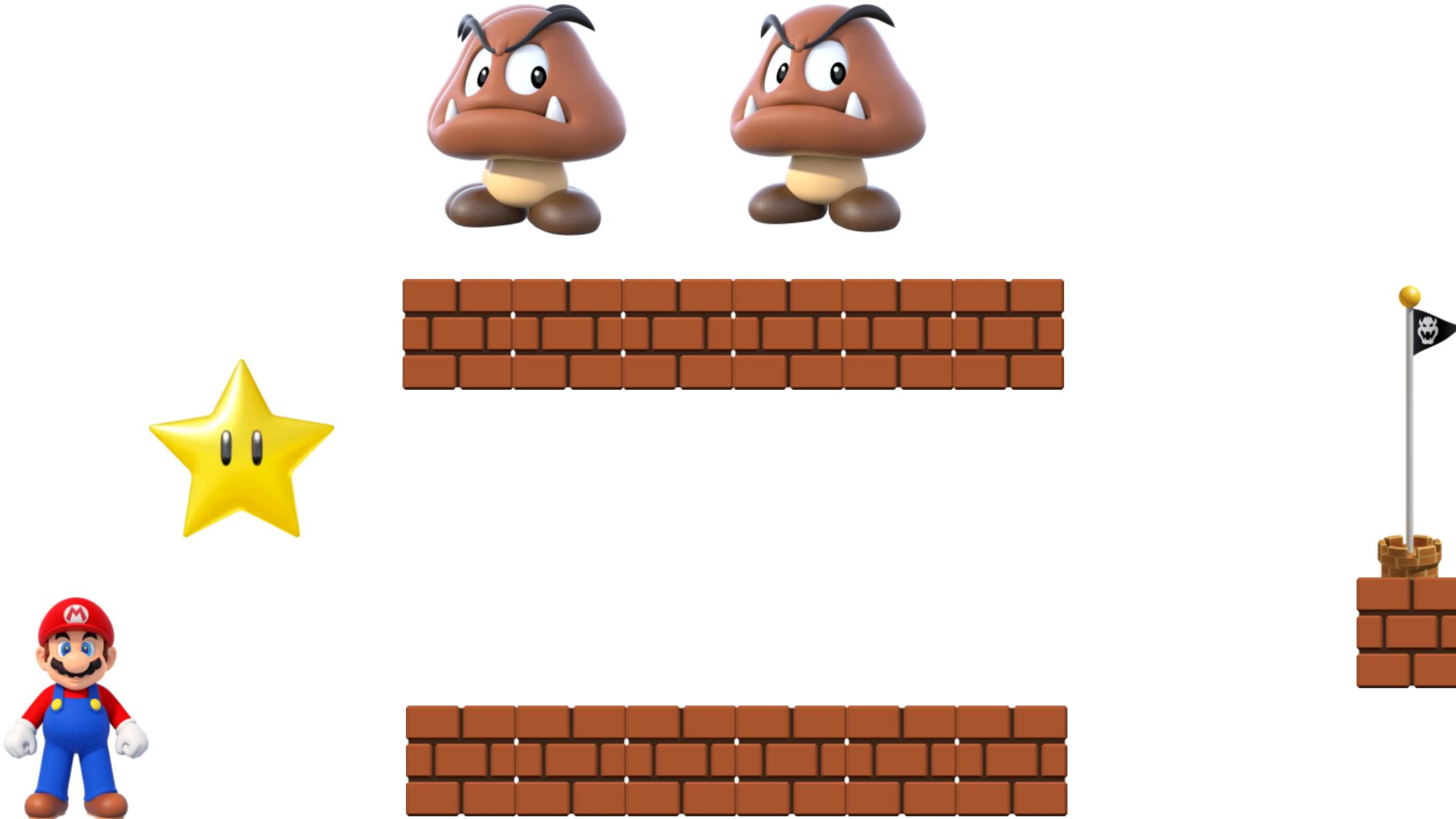
$x_1 = True, x_2 = False, x_3 = True$

마리오에 적용해보자

x_1 : 스타 처 먹기

x_2 : 위쪽 길로 가기

x_3 : 굼바한테 쳐 맞기



조사하면서...

OMS 꽃