

#중급반 6주차

그리디 심화

<Greedy Algorithms Advanced>

3p /<시간 복잡도>

16p /<완전 탐색>

22p /<이진 탐색>

33p /<정렬>

48p /<연습 문제>

#그리디 심화

<Greedy Algorithms Advanced>

그리디 알고리즘 (Greedy Algorithm) – 각 단계에서의 최적의 해 (local optimum)를 통해 전체 문제의 최적해 (global optimum)을 구하는 알고리즘

1. PS에서의 그리디 알고리즘은 **각 단계의 최적**이 **전체 문제의 최적해로 이어짐**이 **증명 가능한 경우**에만 적용 되는게 일반적입니다.
2. 그리디 알고리즘으로 해결되는 문제가 어려운 이유는 보통 저 “증명 ” 에서 오게 됩니다.
3. Local optimum만 본다는 사실이 탐색 범위를 줄여주기에 “관찰을 통해 탐색 범위를 줄이는 문제 ” 라고 생각하는 것도 좋습니다.

#그리디 심화

<Greedy Algorithms Advanced>

“관찰”

결국 그리디 알고리즘을 사용하는 문제의 핵심은 탐색 범위를 줄여주는 올바른 관찰을 하는 것이 핵심입니다.

가장 이상적인 것은 당연히 바로 그 관찰의 정당성을 수학적으로 증명하는 것이겠지만 대회나 코테같이 시간이 제한적인 환경에서는 현실적으로 그러기 어려울 수 있습니다.

따라서, 현실적으로는 일부 케이스에 대해 그 관찰이 적합한지를 테스트해보고 올바른 관찰이기를 믿어야 하는 경우도 있습니다.

하지만, 올바르다고 믿었던 관찰이 사실 틀렸던 관찰이라면 그대로 말리게 될 수도 있습니다. 따라서, 비교적 증명이 간단한 관찰인 경우 증명을 조금이라도 해보는 것이 좋은 습관이며 한번 “틀렸습니다” 판정을 받은 관찰은 빠르게 포기하고 원점으로 돌아오는 것 역시 중요합니다.

이러한 특징으로 인해 개념적으로 설명할 부분은 거히 없지만 관찰을 증명하는데 도움이 될 몇가지 기법들을 알아보시다.

#그리디 심화

<Greedy Algorithms Advanced>

Greedy Stays Ahead – 주어진 문제의 특정 기준에서 그리디 알고리즘의 부분해가 다른 알고리즘의 부분해보다 더 최적이다

1. 위 문장을 토대로 최종적으로 나온 전체해도 최적임을 증명해서 부분해가 전체 최적해에 도달함을 증명하는 기법
2. 수학적 귀납법을 활용

#그리디 심화

<Greedy Algorithms Advanced>

일반적으로 이 기법을 사용한 증명은 다음과 같은 4단계로 이루어진 과정을 거치게 됩니다.

1. 우리가 만든 그리디 알고리즘의 해를 그 해를 구성하는 원소들로 이루어진 집합 (순서는 알고리즘이 실제로 넣는 순서대로)
 $A = \{a_1, a_2, a_3, \dots, a_k\}$ 으로 표현하고 어떤 임의의 최적해 $O = \{o_1, o_2, o_3, \dots, o_m\}$ 이 존재한다고 합니다.

- 어떠한 기준 (함수) 를 정의합니다. 이 기준 하에서 그리디 알고리즘은 임의의 최적해 보다 더 “앞서감” 을 가정합니다.
- 실제로 그 기준 하에서 그리디 알고리즘이 항상 앞서감을 증명합니다.
그리디 알고리즘이 만든 부분해가 모든 $r \leq k$ 인 r 에 대해 $f(a_1 \sim a_r)$ 이 $f(o_1 \sim o_r)$ 보다 최적임을 증명합니다.
- 따라서 귀납적으로 최종 답 또한 최적해를 증명합니다.



#그리디 심화

<Greedy Algorithms Advanced>

1 회의실 배정 (BOJ #1931)

<문제 설명>

- 회의들의 시작 시간과 종료 시간이 주어질때 최대한 많은 회의를 열수 있도록 배치하는 문제

<제약 조건>

- $1 \leq N \leq 100,000$
- 시작 시간과 끝나는 시간은 $2^{31}-1$ 보다 작거나 같은 자연수 또는 0

#그리디 심화

<Greedy Algorithms Advanced>

관찰: 종료 시점이 가장 빠른 회의부터 채워 넣는 것이 최적이다.

증명:

$A = \{i_1, \dots, i_k\}$ 가 위 관찰로 나온 그리디

알고리즘이 고른 회의들의 집합이고

$O = \{j_1, \dots, j_m\}$ 가 임의의 최적해라고 합시다.

기준 $f(S)$ 를 집합 S 에 속한 회의들 중 가장 늦게 끝나는 종료 시점이라고 합시다. 관찰에서 정한 순서로 인해 모든 $r \leq k$ 인 r 에 대해

$$f(i_1 \dots i_r) = f(i_r), f(j_1 \dots j_r) = f(j_r)$$

이제 귀납적으로 $f(i_r) \leq f(j_r)$ 임을 증명합니다.

$r = 1$ 인 경우에 대해서는 자명하게 성립합니다.

$r = k - 1$ 일때 성립한다고 가정하면 O 에 추가하는 k 번째 원소는 A 도 추가할 수 있게 됩니다. 따라서, $r = k$ 일때도 성립.

#그리디 심화

<Greedy Algorithms Advanced>

관찰 : 종료 시점이 가장 빠른 회의부터 채워
넣는 것이 최적이다.

증명 (이어서) :

이제, A 가 최적해가 아니라고 가정해봅시다.

그렇다면 $m > k$ 였다는 의미이며 $f(A)$ 보다
늦게 시작하는 회의가 존재한다는 의미가
됩니다.

하지만, 이는 A 에도 넣을수 있기에 모순이 발생,
따라서 A 가 최적해임이 증명됩니다.

#그리디 심화

<Greedy Algorithms Advanced>

Exchange Argument – 임의의 최적해에서 원소들을 적당히 (최적성과 정당성을 잃지 않는 방법) 바꿔서 그리디 알고리즘의 해로 바꿀수 있음을 증명

1. 귀류법 기반
2. 두 기법 모두 따로 태그가 존재하진 않아서 문제를 풀 때 직접 적용해보는게 중요

#그리디 심화

<Greedy Algorithms Advanced>

일반적으로 이 기법을 사용한 증명은 다음과 같은 3단계로 이루어진 과정을 거치게 됩니다.

1. 우리가 만든 그리디 알고리즘의 해를 그 해를 구성하는 원소들로 이루어진 집합 (순서는 알고리즘이 실제로 넣는 순서대로)

$A = \{a_1, a_2, a_3, \dots, a_k\}$ 으로 표현하고 어떤 임의의 최적해 $O = \{o_1, o_2, o_3, \dots, o_m\}$ 이 존재한다고 합니다.

2. 그리디 해와 최적해를 비교합니다.
일반적으로, 두 해가 다르다고 가정합니다.
3. 적당히 (?????) O 를 A 로 변형할수 있음을 증명합니다.

#그리디 심화

<Greedy Algorithms Advanced>

4 보물 (BOJ #1026)

<문제 설명>

- $S = A[0] \times B[0] + \dots + A[N-1] \times B[N-1]$
- A 만 재배열해서 S 를 최소로 만드는 문제

<제약 조건>

- $1 \leq N \leq 50$
- 각 원소는 100보다 작거나 같은 음이 아닌 정수



#연습 문제 도전

1 회의실 배정 (BOJ #1931)

필수 문제 1

5 사과나무 (BOJ #19539)

어려워요

4 동전 0 (BOJ #11047)

돈 계산

4 보물 (BOJ #1026)

필수 문제 2

5 Merge the Tree and Sequence (BOJ #25337)

학술부장님이 넣어달래요

4 ATM (BOJ #11399)

의외로 돈 계산 아님

#연습 문제 도전

2 1차원 2048과 퀴리 (BOJ #27515)

뭐든지 퀴리를 추가하면 어려워지는

2 트리와 수열 (BOJ #26159)

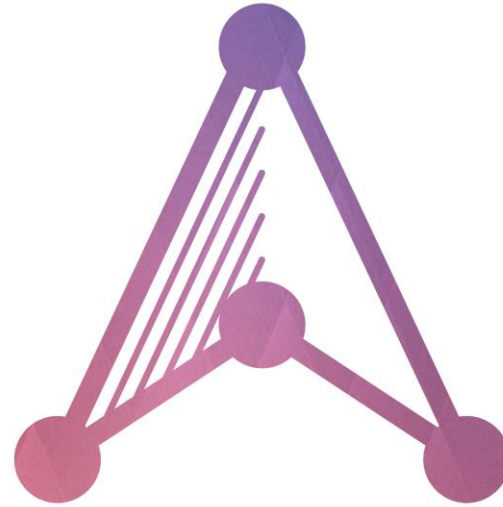
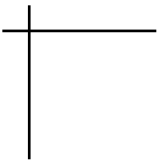
HCPC 기출을 풀어봅시다.

5 도서관 (BOJ #1461)

왔다갔다를 어떻게 하는게 이득일까

1 AND 와 OR (BOJ #23042)

디논 중간고사 flashback



A L O H A
The algorithm club.

