실전 알고리즘 0x0A강

그리디

BaaaaaaaaaaaaaaaarkingDog

목차

0x00 그리디(Greedy)

- 그리디(Greedy)는 매번 선택에서 지금 가장 최적인 답을 근시안적으로 택하는 알고리즘입니다. 뒤의 예시를 보면 더 이해가 잘 갈 것입니다.
- 당연하겠지만 대다수의 경우 그리디 알고리즘은 올바른 답을 주지 않습니다. 그러나 일부 문제에서는 그리디 알고리즘이 올바른 답을 주기 때문에 시간복잡도를 줄일 수 있습니다.
- 그리디 알고리즘을 사용하려면 우선 해당 알고리즘이 성립함을 수학적으로 엄밀하게 증명해야 합니다. 그렇지 않고 그냥 감으로 때려맞춰서 코딩을 하면 시간만 버리게 될 수도 있습니다.
- 코딩테스트 수준에서는 학부 과정에서 언급되는 유명한 몇몇 예시 안에서 나올 가능성이 크고, 이번 강의에서도 그 정도 수준까지만 다룰 것입니다.

0x01 예시 문제 1 : 동전 0

- 바로 실전 문제로 들어갑시다. BOJ 11047번 : 동전 0 문제입니다.
- 만약 K가 작았다면 이 문제를 DP로 해결할 수 있다는 사실을 알겠나요?
- `D[i] = i`를 만들기 위해 필요한 동전 갯수의 최솟값이라고 해봅시다.
- `D[i] = min(D[i-A[0]], D[i-A[1]],D[i-A[2]],…)+1` 입니다.
- 그런데 이 경우 시간복잡도는 $O(NK)$이기 때문에 이 문제에서는 시간 초과가 발생할 것입니다.
- 이 문제는 그리디 알고리즘으로 해결해야 합니다.

0x01 예시 문제 1 : 동전 0

- 직관적으로 생각해봅시다. 만약 10, 50, 100, 500원 동전들로 물건의 가격을 지불할 때 동전을 적게 소모하고 싶으면 어떤 방식으로 지불하나요?
- 일단 500원 동전을 최대한 많이 쓰고, 이후 100, 50, 10원 순으로 최대한 많이 쓸 것입니다.
- 이 방법이 가장 동전을 적게 소모하는 방법임을 증명할 수 있을까요?
- 지금부터 증명을 할 것인데, 그다지 어렵지 않지만 증명에 익숙하지 않으면 잘 이해가 안 갈수도 있습니다. 만약 그렇다면 이해를 포기하고 결론만 받아들이고 넘어가셔도 무방합니다.

0x01 예시 문제 1 : 동전 0
그리디 알고리즘의 정당성 증명

Lemma 1

동전을 최소로 소모하면서 물건값을 지불하려면 10/100원 동전은 4개 이하, 50원 동전은 1개 이하로 사용되어야 한다.

증명

귀류법으로 증명하기 위해 10/100원 동전을 5개 이상 사용하거나 50원 동전을 2개 이상 사용해서 동전을 최소로 소모하면서 물건값을 지불했다고 하자. 만약 10/100원 동전을 5개 이상 사용한다면 50/500원 동전으로 대체해 동전의 갯수를 더 줄일 수 있고 50원 동전을 2개 이상 사용한다면 100원 동전으로 대체해 동전의 갯수를 더 줄일 수 있으므로 가정에 모순이다. 그러므로 Lemma 1은 참이다.

0x01 예시 문제 1 : 동전 0
그리디 알고리즘의 정당성 증명

Lemma 2

동전을 최소로 소모하면서 물건값을 지불하려면 우선 500원 동전을 최대한 많이 사용해야 한다.

증명

Lemma 1에 의해서 동전을 최소로 소모하면서 물건값을 지불하려면 10/100원 동전은 4개 이하, 50원 동전은 1개 이하로 사용되어야 한다. 이 경우 10, 50, 100원 동전으로는 물건값을 최대 $10 \cdot 4 + 50 \cdot 1 + 100 \cdot 4 = 490$원밖에 감당할 수 없다. 만약 500원 동전을 최대한 다 사용하지 않을 경우 10, 50, 100원 동전으로 감당해야하는 물건 값이 500원 이상이 되기 때문에 Lemma 2가 증명되었다.

0x01 예시 문제 1 : 동전 0
그리디 알고리즘의 정당성 증명

- Lemma 2를 통해 동전을 최소로 소모하면서 물건값을 지불하려면 우선 500원 동전을 최대한 많이 사용해야 한다는 사실을 알았습니다.
- 마찬가지 논리를 이용하면 500원 동전을 최대한 많이 사용한 이후에는 100원 동전을, 그 이후에는 50원을, 마지막으로 10원 동전을 최대한 많이 사용해야함을 증명할 수 있고, 그리디 알고리즘의 정당성 증명이 끝났습니다.
- 500/100/50/10원 동전을 일반화시킨 문제에서의 상황도 A_i가 A_{i-1}의 배수이기 때문에 동일한 방식으로 A_{N-1}부터 차례로 최대한 많이 사용하는 알고리즘이 성립하고, 또 정당성을 증명할 수 있습니다.
- 예시 코드 : http://boj.kr/c186a0dcb6924a43a7e543018600dacc

0x01 예시 문제 1 : 동전 0

- 동전 사이에 배수 관계가 성립하지 않을 때에도 동일한 그리디 알고리즘이 성립할까요? 직접 반례를 고민해보세요.
- 동전이 1원, 9원, 10원일 때 18원을 생각해보면 배수 관계가 아닐 때에는 그리디 알고리즘이 성립하지 않음을 쉽게 알 수 있습니다.

0x02 예시 문제 2 : 회의실배정

- BOJ 1931번 : 회의실배정을 풀어봅시다.
- 일단 O(N^2) DP를 알아봅시다. 이를 위해서는 우선 회의를 끝나는 시간이 빠른 순으로 정렬해야 합니다. 그리고 끝나는 시간이 같다면 시작 시간이 빠른 순으로 정렬합시다.
- 이후 `D[i] = i`번째 회의를 마지막으로 진행했을 때 최대 사용할 수 있는 회의의 수 이라고 해봅시다.
- `D[i] = max(D[j])+1` (`j`번째 회의의 끝나는 시간이 `i`번째 회의의 시작 시간 이상인 모든 `j`에 대해) 입니다.
- 그런데 이 경우 이중 for문을 돌면서 이전의 모든 회의를 확인해야하니 시간복잡도는 O(N^2) 으로 이 문제에서는 시간 초과가 발생할 것입니다.

0x02 예시 문제 2 : 회의실배정

- 직관적으로 생각했을 때, 현재 t시간이라고 할 때 시작 시간이 t이상인 모든 회의 중에서 가장 먼저 끝나는 회의를 택하는 것이 가장 좋은 선택일 것 같습니다. 이를 증명해봅시다.

0x02 예시 문제 2 : 회의실배정 그리디 알고리즘의 정당성 증명

명제

현재 t시간이라고 할 때 시작 시간이 t이상인 모든 회의 중에서 가장 먼저 끝나는 회의를 택하는 것이 최적 해이다.

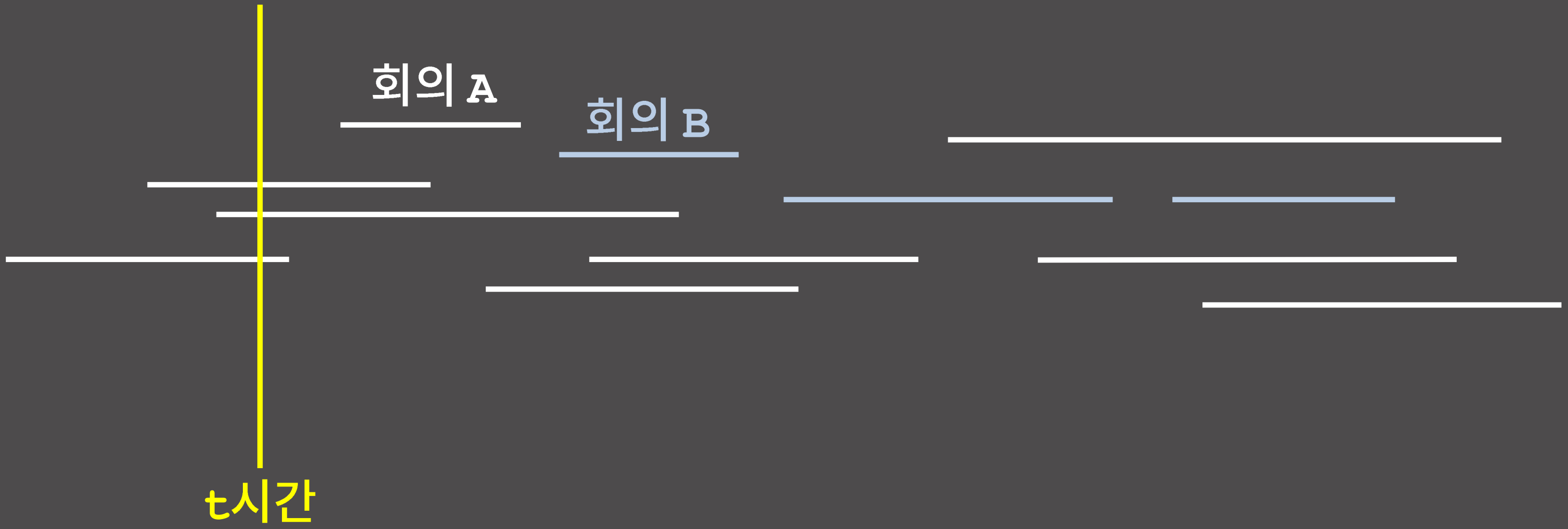
증명

귀류법으로 증명하기 위해 시작 시간이 t이상인 모든 회의 중에서 가장 먼저 끝나는 회의 A를 택하는 대신 A보다 늦게 끝나는 회의 B를 택했을 때 더 많은 회의를 진행할 수 있었다고 하자. 그런데 회의 B를 택해 진행한 스케쥴에서 회의 B 대신 A를 사용해도 아무런 모순이 발생하지 않는다. 그러므로 회의 A를 택해도 적어도 회의 B를 택했을 때 만큼의 회의는 진행할 수 있음이 보장된다. 그러므로 명제는 참이다.

0x02 예시 문제 2 : 회의실배정
그리디 알고리즘의 정당성 증명

명제

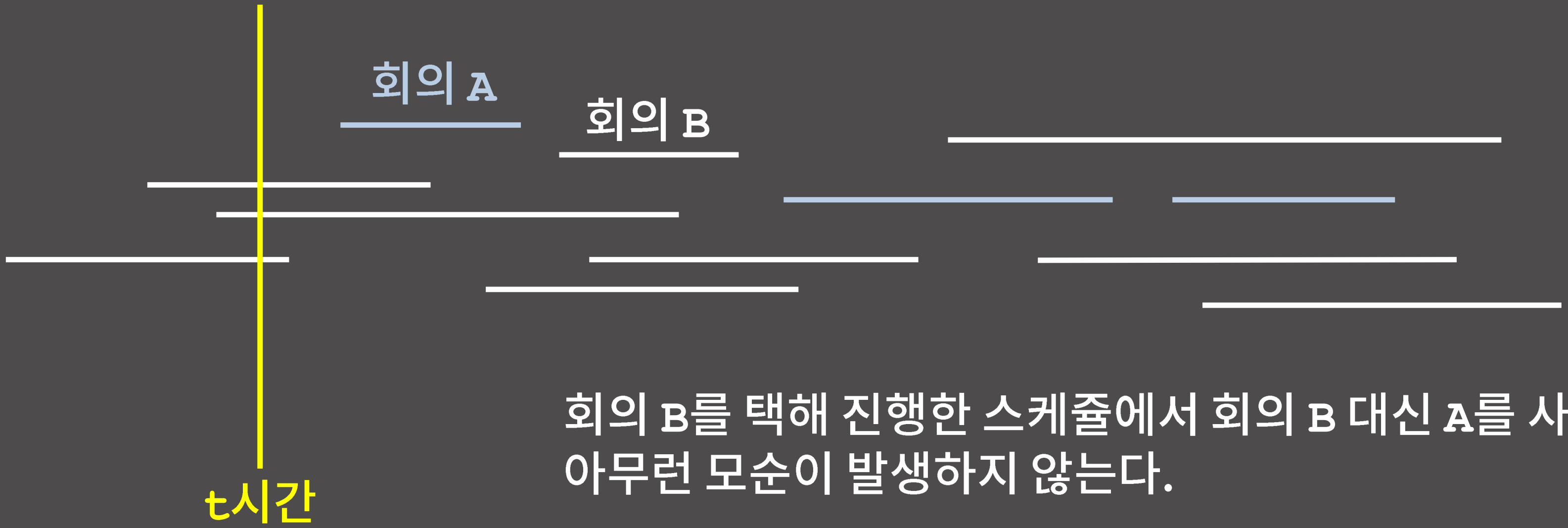
현재 t시간이라고 할 때 시작 시간이 t이상인 모든 회의 중에서 가장 먼저 끝나는 회의를 택하는 것이 최적 해이다.



0x02 예시 문제 2 : 회의실배정
그리디 알고리즘의 정당성 증명

명제

현재 t시간이라고 할 때 시작 시간이 t이상인 모든 회의 중에서 가장 먼저 끝나는 회의를 택하는 것이 최적 해이다.



회의 B를 택해 진행한 스케쥴에서 회의 B 대신 A를 사용해도 아무런 모순이 발생하지 않는다.

0x02 예시 문제 2 : 회의실배정

- 회의를 끝나는 시간이 빠른 순으로, 끝나는 시간이 같다면 시작 시간이 빠른 순으로 정렬을 합니다. STL pair를 활용하면 쉽게 정렬할 수 있습니다.
- 이후 현재 시간에서 가능한 회의들 중에서 끝나는 시간이 가장 빠른 회의를 계속 고르면 됩니다.
- 직접 구현해보세요.
- 예시 코드 : http://boj.kr/f121458bb8494e8987366c688d1d78d3
- 시작 시간을 고려하지 않고 끝나는 시간만을 고려해 정렬했을 경우 어떤 테스트케이스에 대해 문제가 생길지 고민해보세요.

0x03 예시 문제 3 : 로프

- BOJ 2217번 : 로프를 풀어봅시다.
- 이 문제는 앞의 두 문제처럼 학부 수업 시간에 다루는 유명한 그리디 예제는 아니지만 쉬운 난이도의 괜찮은 문제이기 때문에 소개드립니다.
- 당연하겠지만 N이 최대 10만이기 때문에 모든 로프 조합을 다 해보는 $O(2^N)$ 알고리즘은 시간 내에 돌아갈 수 없습니다.
- 그리디 알고리즘을 한 번 고민해봅시다.

0x03 예시 문제 3 : 로프

- 로프를 A개 고르겠다고 미리 정했다고 칩시다. 그러면 주어진 N개의 로프 중에 어떤 것들을 고를건가요?
- 상식적으로 생각했을 때 가장 튼튼한 A개를 고를 것입니다. 이건 굳이 증명을 할 필요가 없을 정도로 당연한 얘기입니다.
- 달리 표현하면 로프를 A개 고를 경우 `W[N-A]×A` 만큼의 무게를 지탱할 수 있다는 의미입니다.
- 예시 코드 : http://boj.kr/68f7801eeb9c477da3daae70d9909274

0x04 잘못된 그리디의 예시

- 그리디(Greedy)는 대다수의 문제에서 사용될 수 없는 알고리즘으로, 주어진 문제가 그리디로 풀릴 것 같다는 느낌이 들어도 엄밀하게 증명을 하고 사용해야 귀중한 시간을 날리지 않을 수 있습니다.
- 그리디로 풀 수 있을 것 같지만 그렇지 않은 예시를 몇 가지 들겠습니다.

0x04 잘못된 그리디의 예시 – 0/1 knapsack

- 부피 K까지 담을 수 있는 가방에 부피가 A_i, 가치가 C_i인 물건 $i = 1,2,3, \ldots, N$이 주어졌을 때 어떤 물건들을 택해야 가방의 부피 제한을 만족하면서 가장 가치를 높일 수 있는가?
- 그리디 알고리즘 : 현재 가방에 넣을 수 있는 물건 중에서 부피 대비 가치가 높은 물건(즉 C_i / A_i 가 큰 물건) 순으로 넣는다.
- 반례 : 가방의 부피 = 10, 물건 = (6, 7), (5, 5), (5, 3). (5, 5)와 (5, 3)을 택할 경우 가치 8을 얻을 수 있으나 그리디 알고리즘은 (6, 7)을 택해 가치 7을 얻게 된다.
- 정해 : O(NK) DP 혹은 O(2^N) 전수조사

0x04 잘못된 그리디의 예시 – 휴게소 세우기

- BOJ 1477번 : 휴게소 세우기
- 그리디 알고리즘 : 현재 휴게소가 없는 가장 긴 구간의 중점에 새로운 휴게소를 세운다.
- 반례 : 휴게소가 0개일 때 새로운 휴게소 2개를 세우면 정해는 333, 666 지점에 휴게소를 세워 답이 334이지만 그리디 알고리즘은 500, 250 지점에 세워 답이 500이 된다.
- 정해 : O($N\,lg\,1000$) 이분 탐색

강의 정리

- 그리디 알고리즘에 대해 배웠고 대표적인 문제 3개를 다뤘습니다.
- 대다수의 문제에서 그리디 알고리즘을 사용할 수 없음을 알았습니다.
- 코딩테스트에서 그리디 알고리즘 문제가 나온다면 오늘 소개한 대표 문제 수준일 것입니다. 이 수준을 넘어서면 어차피 대부분의 사람이 풀어내지 못합니다.
- 만약 시간이 넉넉하게 남았고 현재 보는 문제의 그리디 알고리즘을 찾은 것 같은데 반례도 못 찾겠고 알고리즘의 정당성을 증명하지도 못하겠으면 일단 건너뛰고 다른 문제를 보는게 낫습니다. 그러나 시간이 촉박해 다른 문제를 고민할 시간이 부족하면 증명 없이 바로 코드로 옮기는게 낫습니다.
- 뻔한 얘기지만 그리디 문제를 많이 풀어보면 내가 세운 알고리즘의 반례를 찾는 능력도, 정당성을 유추하는 능력도 향상됩니다. 시간 남으면 문제 많이 풀어보세요.