Can polan into space? - 풀이

1 부분문제 1/2 (23점)

부분문제 1은 $O(n!)$ 시간 복잡도에 완전 탐색으로 해결할 수 있습니다. 이 때, 완전 탐색 결과를 비트마스크로 메모이제이션하면 이를 $O(2^n * n)$으로 최적화 할 수 있습니다. 이러한 풀이에 익숙치 않으면, "비트마스크 DP" 라는 키워드로 관련 자료를 찾아 보시기 바랍니다.

2 부분문제 3 (46점)

맨 처음 공 i가 날아갔을 때, 해당 공의 양 옆에 인접해 있는 $i-1$번 공과, $i+1$번 공은, 거리가 2 이상 떨어져 있기 때문에 더 이상 서로 영향을 받지 않습니다. 즉, $[1, i-1]$ 구간에서의 문제와, $[i+1, n]$ 구간에서의 문제를 똑같이 재귀적으로 해결한 후 결과를 합쳐주면, 전체 문제를 해결한 꼴이 되는 셈입니다. 이를 계속 반복하면 결국 답을 찾을 수 있습니다.

이러한 아이디어를 사용하여, 구간에 대한 동적 계획법으로 문제를 해결할 수 있습니다. $D[s, e]$ = 구간 $[s, e]$ 의 공들이 날아가지 않았고, $s-1$번 공과 $e+1$번 공은 (만약 존재한다면) 날아갔을 때의 최적 추진력 합이라고 정의하면, 구간 $[s, e]$에서 처음으로 날리게 될 공의 번호를 고정시킨 후, 그에 따라서 부분 문제를 합쳐주는 식으로 문제를 해결할 수 있습니다. $O(n^2)$ 개의 동적 계획법 표를 $O(n)$ 번의 연산으로 채울 수 있으니, 시간 복잡도는 $O(n^3)$이 되어 부분문제 3을 해결할 수 있습니다.

3 부분문제 4 (72점)

부분문제 4 이상부터는 접근 방법을 달리 해야 합니다. 최적의 해에서, 각각의 공들은 a_i, b_i, c_i 중 하나의 추진력을 받고 발사되었을 것입니다. a_i의 추진력을 받고 발사된 공을 A라고 했을 때, 몇가지 사실을 관찰할 수 있습니다.

- A인 서로 다른 두 공은 인접하지 않습니다.
- A인 공들은 처음에 어떤 순서로 뽑든 상관이 없습니다.
- A인 공들을 제거하면, 나머지 공들은 연속한 구간을 이루며, 연속한 구간에서 공을 뽑을 때는 구간의 앞에서부터 뽑거나 뒤에서부터 뽑아야 합니다.

그렇다면, 최적의 해를 만들어 내는 순서는 어떠한 형태를 띌까요?

- 1. A인 공 중 번호가 가장 작은 공 i를 뽑습니다. 이 공을 기준으로 양 옆에 최대 두 개의 구간이 생깁니다.
- 2. 만약 앞쪽 구간이 존재한다면, 앞쪽 구간을 $i-1, i-2$ ... 순으로 전부 뽑습니다. 뒤에서부터 뽑을 수 있는 연속한 구간의 공들을 제거하는 과정으로 생각하면 됩니다.
- 3. 만약 뒷쪽 구간이 존재한다면, 뒷쪽 구간을 $i+1, i+2$ ... 순으로 뽑습니다. 이 때는 전부 뽑을 필요가 없으며 필요한 만큼만 뽑으면 됩니다. 뒤에서부터 뽑을 수 없는 연속한 구간의 공들을 제거하는 과정으로 생각하면 됩니다.

이를 정리하면, 최적의 해를 만들어 내는 순서는 1부터 n까지의 자연수를 순서대로 발사하지만, 일부 겹치지 않는 구간이 뒤집어져 있는 형태로 표현 가능합니다. 1, 2, 3, 4, 5, [9, 8, 7, 6], 10, 11, [14, 13, 12], 15 ... n과 같은 수열이 그 예입니다.

이 관찰을 통해서 문제는 간단한 동적 계획법 문제로 환원되었습니다. $D[i]$ = 구간 $[1, i]$에 있는 모든 공들을 발사 시키는 데 드는 비용 이라고 정의했을 경우, 우리는 $i-1$ 까지에 있는 모든 공들을 발사시켰거나, $0 \leq j \leq i-2$ 까지에 있는 모든 공들을 발사 시키고, $i, i-1, i-2, ..., j+1$ 까지를 역순으로 발사시켰을 것입니다.

어떠한 구간에 있는 공들을 역순으로 발사시키는 데 드는 비용은 b_i에 대한 부분 합 배열을 만들면 $O(1)$에 쉽게 계산할 수 있습니다. $O(n)$ 개의 동적 계획법 표를 $O(n)$ 번의 연산으로 채울 수 있으니, 시간 복잡도는 $O(n^2)$이 되어 부분문제 4를 해결할 수 있습니다.

4 부분문제 5 (100점)

부분문제 5의 풀이는 부분문제 4와 크게 다르지 않습니다. 귀찮지만, 앞서 언급했던 동적 계획법 식을 실제로 써 봅시다. 여기서 $S_i = \sum_{j=1}^{i} b_j$ 로 정의합니다.

$D_i = min(D_{i-1} + b_i, S_{i-1} + a_i, min_{j=1}^{i-1}(S_{i-1} - S_{j+1} + c_{j+1} + a_i + D_j))$

$min_{j=1}^{i-1}(S_{i-1} - S_{j+1} + c_{j+1} + a_i + D_j)$ 에 주목해 봅시다. $S_{i-1} + a_i$ 는 i에 대한 상수이니 신경 쓸 필요가 없고, $-S_{j+1} + c_{j+1} + D_j$ 는 j에 따라서만 값이 변화합니다. 동적 계획법 표를 채워 가면서, 해당 값을 최대로 하는 인덱스 j를 가지고 다니면, 간단하게 해당 루프를 제거할 수 있고, 이 때의 시간 복잡도는 $O(n)$ 이 되어 만점을 받을 수 있습니다.