Greedy

KUCSEPotato

January 2026

1 Greedy Algorithm

Greedy Algorithm(그리디 알고리즘, 탐욕 알고리즘)이란 문제를 여러 단계로 나누고, 각 단계에서 당장 최선(local optimum)인 선택을 하여 전체 최적해(global optimum)을 얻으려는 알고리즘이다. 중요한 특징은 매 순간의 최적해가 전체 최적해로 귀결되지 않는다는 점이다. 그리디 알고리즘의 성립에는 문제가 아래의 두 가지 특징을 가져야 한다.

- **(1) Greedy Choice Property(탐욕적 선택 속성)**: 지금 내리는 최선의 선택이 나중의 선택 가능성을 망치지 않는다. 즉, 지금 최선이 전체 최선으로 이어진다.
- **(2) Optimal Substructure(최적 부분 구조)**: 전체 문제의 최적해가 부분 문제들의 최적해로 구성된다.

다시 말해, 한번의 선택이 다음 선택에는 전혀 무관한 값이어야 하며 매 순간의 최적해가 문제에 대한 최적해여야 한다는 의미이다. 그리디 알고리즘의 장점과 한계는 아래처럼 정리할 수 있다.

- **장점:** 구현이 간결하고 빠른 경우가 많다(대개 정렬 $O(n \log n)$ + 선형 스캔 $O(n)$).
- **한계:** "지금 최선"이 "전체 최선"을 보장하지 않는 문제에서는 오답이 된다.

1.1 그리디가 성립하는 직관

그리디가 성공하는 문제는 대체로 다음과 같은 특성을 가진다.

- 선택의 결과가 "국소적으로 안전"하며, 나중에 되돌릴 필요가 없다.
- 선택의 기준이 정렬/우선순위로 표현되며, 한번의 스캔으로 해를 구성할 수 있다.
- 교환 논법(Exchange Argument)이나 컷 속성(Cut Property) 등으로 정당화가 가능하다.

1.1.1 교환 논법(Exchange Argument)

교환 논법은 다음과 같은 방식으로 그리디 선택의 정당성을 보인다.

1. 어떤 최적해(Optimal Solution) $S^\star$가 존재한다고 가정한다.
2. 그리디가 선택한 원소/결정 g가 $S^\star$에 포함되어 있지 않다면, $S^\star$에서 어떤 원소 x와 g를 "교환"하여도 해의 품질이 나빠지지 않음을 보인다.
3. 따라서 g를 포함하는 또 다른 최적해가 존재하며, 그리디의 첫 선택은 최적해와 양립할 수 있다.
4. 이후 남은 부분 문제에 대해 같은 논리를 반복(귀납)한다.

요약하면, **그리디의 선택을 최적해에 "강제로 끼워 넣어도" 최적성이 유지됨**을 보이는 방식이다.

1.1.2 컷 속성(Cut Property)과 그리디 (그래프)

최소 신장 트리(MST)에서 크루스칼/프림이 그리디로 작동하는 이유는 컷 속성과 관련이 있다.

> 그래프를 두 부분으로 나누는 임의의 컷(cut)에 대해, 그 컷을 가로지르는 간선 중 **가장 가중치가 작은 간선**은 어떤 MST에 반드시 포함될 수 있다.

이 성질을 바탕으로 "가장 작은 간선부터 선택하자" 같은 그리디 전략이 정당화된다.

2 그리디 vs DP(동적 계획법)

그리디 알고리즘은 빠르고 구현이 간단하지만, 최적해를 보장하기 위해서는 문제의 구조가 특정 조건을 만족해야 한다. 반면 동적 계획법(DP)은 더 넓은 범위의 문제에서 최적해를 보장할 수 있으나, 상태 정의와 전이 설계가 필요하며 계산 비용과 메모리 사용량이 증가하는 경우가 많다. 다시 말해, 두 방법은 일반적으로 계산 비용과 최적성 보장 사이의 trade-off 관계에 놓여 있다고 볼 수 있다.

구분	그리디	DP
핵심 전략	매 순간 최선의 선택	모든 부분 문제의 최적해 누적
최적 보장	조건부(정당화 필요)	가능(정의 및 전이식이 올바른 경우)
계산량	대개 $O(n \log n)$ 또는 $O(n)$	상태 수 × 전이 비용에 의존
구현 난이도	상대적으로 단순	상태 설계, 전이식, 메모리 관리 필요
대표 상황	정렬+선택, MST, 활동 선택, 간격 스케줄링	배낭(0/1), 최장 증가 부분 수열 등

3 그리디 알고리즘 설계 절차(실전 체크리스트)

다음 특징이 보이면 그리디를 의심해볼 만하다.

- "최대/최소"를 구하는 문제이며, 선택 기준이 명확하다.
- 선택 결과가 이후의 선택 공간을 단조롭게 줄인다(예: 가장 빨리 끝나는 회의 선택).
- 정렬 후 한 번의 스캔으로 답이 만들어질 것처럼 보인다.

단, **정당화를 할 수 없다면** 그리디는 위험하므로 DP/탐색을 고려해야 한다. 코딩 테스트/알고리즘 문제에서 그리디를 설계할 때의 표준 절차를 정리한다.

3.1 절차

1. **목표 정의**: 최적화 대상(최소/최대), 제약 조건을 명확히 한다.
2. **선택 기준 설계**: "무엇을 기준으로 선택할 것인가?"를 한 문장으로 정의한다.
3. **정렬 또는 우선순위 구조 선택**: 정렬/힙/큐 등을 통해 선택을 빠르게 만든다.
4. **한 번의 스캔으로 해 구성**: 가능한 한 선형으로 해를 쌓는다.
5. **정당화(증명) 작성**: 교환 논법/컷 속성/단조성 등을 이용해 설명한다.
6. **반례 검증**: 작은 입력, 극단 케이스(동일 값, 경계값)에서 깨지지 않는지 확인한다.

3.2 반례 검증 체크리스트

- 가장 작은 입력(최소 크기)에서 동작하는가?
- 값이 모두 동일하거나 정렬이 무의미할 때도 맞는가?
- 선택이 "미래를 봉쇄"할 가능성이 있는가?
- 최적해가 여러 개인 경우에도 그리디가 하나를 올바르게 찾는가?

4 대표 예제

4.1 대표 예제 1: 활동 선택 문제(회의실 배정)

4.1.1 문제 요약

각 활동(activity)은 시작 시간 s_i와 종료 시간 f_i를 갖는다. 서로 겹치지 않게(호환되게) 활동을 최대한 많이 선택하고자 한다.

4.1.2 그리디 전략

종료 시간이 가장 빠른 활동부터 선택한다.

1. 활동을 종료 시간 기준으로 오름차순 정렬한다.
2. 가장 먼저 끝나는 활동을 선택한다.
3. 다음 활동들 중에서, 현재 선택한 활동의 종료 시간 이후에 시작하는 활동을 다시 같은 방식으로 선택한다.

4.1.3 정당화(교환 논법 스케치)

최적해 $S^\star$ 중 첫 번째로 선택된 활동을 a라 하자. 그리디가 선택한 "가장 빨리 끝나는 활동"을 g라 하면 $f(g) \le f(a)$이다. 따라서 a를 g로 교환해도 이후에 선택할 수 있는 활동의 집합이 줄지 않으며(오히려 늘 수 있음), g를 포함하는 최적해가 존재한다. 이 과정을 반복하면 그리디 해가 최적해가 됨을 보일 수 있다.

4.1.4 복잡도

정렬 $O(n \log n)$, 선택 과정 $O(n)$이므로 전체는 $O(n \log n)$이다.

4.2 대표 예제 2: 거스름돈 문제

4.2.1 문제 요약

동전 단위가 주어졌을 때, 어떤 금액을 만들기 위한 동전 개수를 최소화하고자 한다.

4.2.2 그리디 전략

가능한 가장 큰 단위의 동전부터 최대한 사용한다.

4.2.3 주의: 항상 성립하지 않는다

동전 체계가 "정상적(canonical)"인 경우에만 그리디가 최적을 보장한다. 예를 들어 동전 단위가 $\{10, 6, 1\}$이고 금액이 12이면, 그리디는 $10+1+1$ (3개)을 선택하지만 최적은 $6+6$ (2개)이다.

4.3 대표 예제 3: 최소 신장 트리(MST)와 그리디

4.3.1 문제 요약

가중치 그래프에서 모든 정점을 연결하되 가중치 합이 최소가 되는 스패닝 트리를 찾는다.

4.3.2 Kruskal 알고리즘

1. 모든 간선을 가중치 오름차순으로 정렬한다.
2. 사이클을 만들지 않는 선에서 가장 작은 간선을 선택한다.
3. 간선 선택이 $V-1$개가 될 때까지 반복한다.

4.3.3 정당화(컷 속성)

임의의 컷에 대해 최소 간선은 어떤 MST에 포함 가능하다는 컷 속성에 의해 정당화된다.

4.3.4 복잡도

정렬 $O(E \log E)$ + 유니온파인드(거의 상수)로 전체 $O(E \log E)$로 이해한다.

5 그리디가 실패하는 대표 반례들

그리디가 실패한다는 것은 "국소 최적 선택"이 "전역 최적"을 보장하지 못한다는 뜻이다. 대표 반례를 통해 경계 감각을 잡는 것이 매우 중요하다.

5.1 반례 1: 0/1 배낭(0/1 Knapsack)

각 물건은 무게 w_i와 가치 v_i를 갖고, 배낭 용량 W 안에서 최대 가치를 선택한다. **가치/무게 비율이 큰 것부터 담는다**는 그리디는 0/1 배낭에서는 최적을 보장하지 않는다. 이 문제는 전형적으로 DP로 해결한다.

5.2 반례 2: 동전 문제(비정상적 동전 체계)

앞서 본 $\{10, 6, 1\}$과 같은 체계는 그리디를 깨뜨린다. 따라서 "동전 문제 = 그리디"라고 기계적으로 결론 내리면 위험하다.

6 요약

그리디 알고리즘은 각 단계에서의 국소 최적 선택을 통해 전체 해를 구성하는 방법이며, 특정 구조(탐욕적 선택 속성, 최적 부분 구조)를 만족하는 문제에서 매우 효율적이고 간결하게 최적해를 제공한다. 그러나 모든 문제에서 통하지 않으므로, 교환 논법/컷 속성 등으로 정당화를 수행하고 반례를 검증하는 과정이 필수적이다.