

Greedy Approach

제5장



2016-Fall

국민대학교 컴퓨터공학부 최준수

Greedy Approach

- 욕심장이 기법
 - 욕심장이 기법으로 해결하는 문제의 유형
 - 입력으로 여러 개의 원소들의 집합 S 가 주어진다.
 - 해답으로는 문제에서 주어지는 조건을 만족하는 집합 S 에 속하는 원소들의 부분집합 G 이다.
 - (예) 동전교환문제
 - 동전들의 집합 S : 1, 10, 25, 50, 100 원짜리 동전들의 집합
 - 378 원을 최소 개의 동전으로 바꾸는 동전조합 G

Greedy Approach

- 욕심장이 기법
 - 욕심장이 기법으로 문제를 해결하는 개략적인 방법
 - 초기에 해답을 나타내는 집합 G 는 공집합입니다.
 - 해답을 나타내는 집합 G 에 포함될 S 에 속하는 원소를 **차례로** 한 개씩 선택한다.
 - 이 때, 원소를 선택하는 조건은 문제에서 주어진다.
 - 어떤 원소를 선택하면, 이 원소를 선택함으로써 이전에 선택한 원소를 취소하거나, 이 다음에 어떤 원소를 선택할 결정에 영향을 끼치지 않는다.
 - 즉, 현재 상황에서 선택하는 조건을 만족하는 원소를 **전후 상황을 고려하지 않고 무조건** 취한다. (그래서, 이름이 "욕심장이 기법"이라 붙여짐)

Greedy Approach

- 욕심장이 기법
 - (예) 동전교환문제
 - 동전들의 집합 $S : 1, 10, 25, 50, 100$ 원짜리 동전들의 집합
 - » 각 동전은 무한히 많다고 가정
 - $C=378$ 원을 최소 개의 동전으로 바꾸는 동전조합
 - 초기에 $G = \emptyset, C=378$
 - 동전의 개수를 최소화하기 위해서는 액수가 큰 동전부터 교환
 - 100 원짜리 선택. $S = \langle 100 \rangle, C=278$
 - 100 원짜리 선택. $S = \langle 100, 100 \rangle, C=178$
 - 100 원짜리 선택. $S = \langle 100, 100, 100 \rangle, C=78$
 - 50 원짜리 선택. $S = \langle 100, 100, 100, 50 \rangle, C=28$
 - 25 원짜리 선택. $S = \langle 100, 100, 100, 50, 25 \rangle, C=3$
 - 다음 세 번의 선택에서 각각 1원짜리 동전 세 개를 선택.
 $S = \langle 100, 100, 100, 50, 25, 1, 1, 1 \rangle$

Greedy Approach

- 욕심장이 기법을 통한 문제 해결 단계
 - 초기에 해답을 나타내는 집합 G 는 공집합입니다.
 - 해답을 나타내는 집합 G 에 포함될 S 에 속하는 원소를 아래와 같은 방법으로 **차례로** 한 개씩 선택한다.
 - (단계 1) 선택단계 (Selection procedure) \leftarrow 핵심
 - 현재 상황에서 주어진 조건을 만족하는 가장 최선의 원소를 선택하여 G 에 포함시킨다.
 - (단계 2) 가능성검사 (Feasibility check)
 - 단계 1에서 선택된 원소를 포함한 G 가 앞으로 문제의 해가 될 가능성이 있는지를 검사한다.
 - 가능성이 있는 경우에는 단계 3을 선택한다.
 - 그렇지 않는 경우에는 선택된 원소를 G 에서 제거하고 단계 1을 수행한다.
 - (단계 3) 문제의 해 검사 (Solution check)
 - G 가 문제의 최종해가 되는지를 검사한다.
 - 최종해가 되는 경우에는 종료한다.
 - 그렇지 않은 경우에는 단계 1을 수행한다

Greedy Approach

- 욕심장이 기법을 통한 “동전교환문제” 알고리즘

```
while (the instance is not solved)
{
    // selection procedure
    select the largest remaining coin;

    // feasibility check
    if (adding the coin makes the exchange exceed the amount owed)
        reject the coin;
    else
        add the coin to the exchange

    // solution check
    if (the total value of the change equal the amount owed)
        the instance is solved;
}
```

Greedy Approach

- 욕심장이 기법의 문제점
 - 예
 - 동전의 종류 : 100원, 63원, 50원, 25원, 10원, 1원
 - 거스름돈 : 378 원
 - 욕심장이 기법에 의한 해답
 - $S = \langle 100, 100, 100, 63, 10, 1, 1, 1, 1, 1 \rangle$
동전 10개
 - 해답
 - $S = \langle 63, 63, 63, 63, 63, 63, 63 \rangle$
동전 6개
 - 일반적인 경우의 해결 알고리즘
 - Dynamic Programming

도둑 가방 채우기 문제

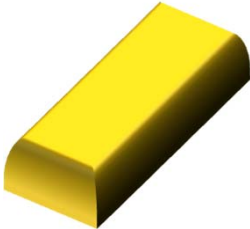



- 문제

- 어떤 도둑이 식료품가게에 들어가서 치즈를 훔치려고 한다.
- 이 가게는 각각 부피도 다르고 가격도 다른 여러 가지 종류의 치즈를 판매하고 있다.
- 도둑은 되도록이면 자기가 가지고 온 가방에 치즈를 가득 채우되 가방에 채워진 치즈의 가격이 최대한으로 만들고자 한다.
- 단, 치즈는 필요한 경우에는 칼로 잘라서 일부만 가지고 갈 수 있다고 가정한다.

도둑 가방 채우기 문제

– 예

- 가방의 부피 : 10

				
치즈명	치즈-1	치즈-2	치즈-3	치즈-4
부피	8	3	6	4
가격	160원	150원	600원	160원

- 가방에 넣은 치즈 및 가격의 합 예

치즈명	치즈-1	치즈-2	치즈-3	치즈-4	합
부피	4	1	3	2	10
가격	80	50	300	80	510
부피	0	3	6	1	10
가격	0	150	600	40	790

도둑 가방 채우기 문제

- Greedy Algorithm

- 목적 : 가방에 되도록 치즈를 가득 채우고, 치즈 가격이 최대로

- Observation

- 치즈는 자를 수 있으므로, 언제든지 가방을 가득 채우도록 치즈를 담을 수 있다.
 - 따라서, 단위 부피당 가격이 높은 치즈를 되도록 많이 가방에 담으면 된다.

도둑 가방 채우기 문제

- Greedy Algorithm

- 단위 부피당 치즈가격이 높은 순서

치즈명	부피	가격	단위 부피당 가격
치즈-3	6	600	100
치즈-2	3	150	50
치즈-4	4	160	40
치즈-1	8	160	20

- 해답

- 먼저 치즈-3 전체를 가방에 넣는다. 남은 가방 부피는 4이다.
- 다음으로 치즈-2 전체를 가방에 넣는다. 남은 가방 부피는 1이다.
- 최종적으로 치즈-4를 부피 1만큼 가방에 넣는다.

치즈명	치즈-1	치즈-2	치즈-3	치즈-4	합
부피	0	3	6	1	10
가격	0	150	600	40	790

도둑 가방 채우기 문제

- 욕심장이 기법의 문제점
 - 위와 동일한 문제이지만, 치즈를 자를 수 없다고 가정.

치즈명	부피	가격	단위 부피당 가격
치즈-3	6	600	100
치즈-2	3	150	50
치즈-4	4	160	40
치즈-1	8	160	20

- 욕심장이 기법에 의한 해답

치즈명	치즈-1	치즈-2	치즈-3	치즈-4	합
부피	0	3	6	0	9
가격	0	150	600	0	750

- 해답

치즈명	치즈-1	치즈-2	치즈-3	치즈-4	합
부피	0	0	6	4	10
가격	0	0	600	160	760

- 일반적인 경우의 해결 알고리즘
 - Dynamic Programming ¹²

회의실 배정 문제

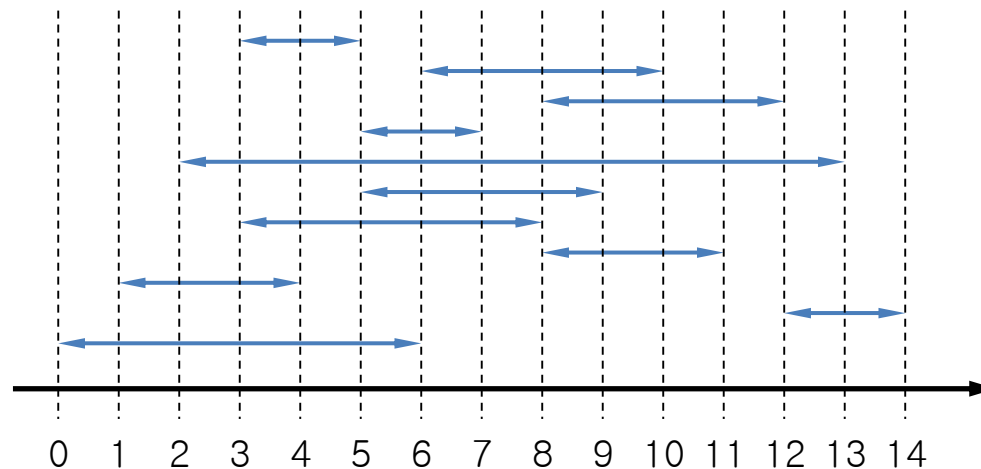
- 문제

- 어떤 회사에 회의실이 1개 있으며, 여러 부서에서 이 회의실을 공유.
- 매일 아침 회의실 사용 예정 부서에서는 회의 [시작시간, 종료시간]을 회의실 관리부서에 제출
- 회의실 관리부서에서는 제출된 각 부서의 회의예정 시각을 통하여, 그 날 가장 많은 회의가 회의실에서 열리도록 배정하고자 한다.
- 어떤 부서는 회의실을 배정받지 못할 수도 있다.

회의실 배정 문제

• 예

회의번호	1	2	3	4	5	6	7	8	9	10	11
시작시간	0	12	1	8	3	5	2	5	8	6	3
종료시간	6	14	4	11	8	9	13	7	12	10	5



- 욕심장이 기법에 의한 해결 알고리즘에서 각 회의선택 기준
 - 가장 빨리 시작하는 회의부터 선택
 - 가장 빨리 끝나는 회의부터 선택 빨리 끝나야 많이 배정할 수 있다!!
 - 가장 짧은 회의부터 선택
 - 가장 적게 겹치는 회의부터 선택

회의실 배정 문제

- 욕심장이 기법 알고리즘
 - 욕심장이 기법에 의한 해결 알고리즘에서 각 회의선택 기준
 - 가장 빨리 끝나는 회의부터 선택

회의번호	1	2	3	4	5	6	7	8	9	10	11
시작시간	1	3	0	5	3	5	6	8	8	2	12
종료시간	4	5	6	7	8	9	10	11	12	13	13

회의실 배정 문제

- 욕심장이 기법 알고리즘

회의번호	1	2	3	4	5	6	7	8	9	10	11
시작시간	1	3	0	5	3	5	6	8	8	2	12
종료시간	4	5	6	7	8	9	10	11	12	13	13

```
// n : number of meetings
// s[] : start time of each meeting
// f[] : finish time of each meeting
Greedy_Room_Assignment(int n, int s[], int f[])
{
    s[]와 f[]를 회의시간이 끝나는 시간이 오름차순이 되도록 정렬한다.
    A = ∅;
    j = 1; A = A ∪ {i};
        j = i;
    return A;
    for(i=2; i<=n; i++)
        if (s[i] >= f[j])

}
```


회의실 배정 문제

- 욕심장이 기법 알고리즘 쉽지만 안되는 경우가 너무나 많음

회의번호	1	2	3	4	5	6	7	8	9	10	11
시작시간	1	3	0	5	3	5	6	8	8	2	12
종료시간	4	5	6	7	8	9	10	11	12	13	13

