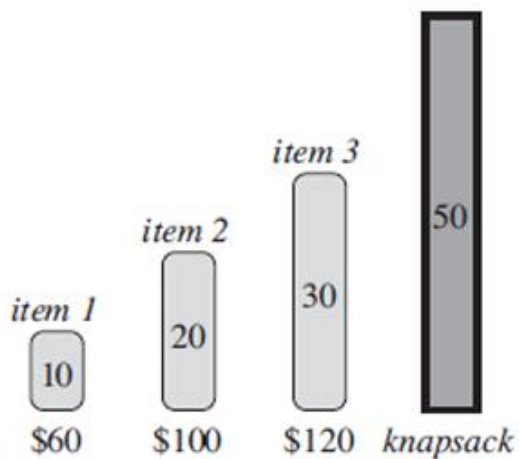


Report 2 - 0 – 1 Knapsack problem

소프트웨어학부 2021041017 김규현

Main task: Based on greedy algorithm paradigm, design and implement an algorithm to solve 0-1 knapsack problem

Problem Description: $n = 3$ 개의 아이템이 있으며, 각각의 아이템 i 는 가치 v_i 와 무게 w_i 를 가지고 있다. 여기서, 배낭의 무게 제한은 $W = 50$ 이다.



i	1	2	3
v_i	60	100	120
w_i	10	20	30
v_i/w_i	6	5	4

Goal: 아이템을 부분적으로 나누지 않고, 전체 선택(1) 또는 선택하지 않음(0)으로 결정해 총 무게가 W 를 초과하지 않으면서 아이템의 총 가치를 최대화하는 것.

Algorithm Analysis based on Greedy Algorithm:

1. 각 아이템의 무게당 가치비율(v_i/w_i)를 계산한다.
2. 아이템을 v_i/w_i 의 내림차순으로 정렬한다.
3. 정렬된 순서에 따라 아이템을 하나씩 선택하며, 선택한 아이템의 총 무게가 W 를 초과하지 않도록 한다.
4. 아이템을 선택할 때는 반드시 전체 선택(1) 또는 선택하지 않음(0)만 가능하다..

Input: 아이템의 가치(v_i)와 무게(w_i) 리스트, 배낭의 최대 무게 W

Output: 선택된 아이템의 총 가치 $totalValue$

1. 각 아이템의 가치 대비 무게 비율 v_i/w_i 를 계산
2. 아이템을 v_i/w_i 기준으로 내림차순 정렬
3. $currentWeight = 0$, $totalValue = 0$ 으로 초기화
4. 각 아이템 i 에 대해:
 - a. 만약 $currentWeight + w_i \leq W$ 이면:
 - 아이템 i 를 선택한다.
 - $currentWeight += w_i$
 - $totalValue += v_i$
 - b. 그렇지 않다면: 아이템 i 를 선택하지 않는다.
5. $totalValue$ 반환

Time Complexity Analysis:

1. 초기화: $O(1)$
2. 비율 계산 및 리스트 생성: $O(n)$
3. 정렬 과정:
 - n 개의 아이템을 무게당 가치비율(v_i/w_i) 기준으로 정렬하는데 Divide and Conquer Algorithm(Merge sort, Quick sort, etc.)가 사용된다. 이 때, 시간 복잡도는 $O(n \log n)$ 이다.
4. 탐색 및 선택 과정: $O(n)$
5. $T(n) = O(1) + O(n) + O(n \log n) + O(n)$
 \therefore Time Complexity: **$O(n \log n)$**

Note: Greedy Algorithm은 단순하고 빠른 계산이 가능하지만, 0-1 Knapsack problem에서는 항상 최적의 해를 보장하지 않는다. 더 최적화된 해결을 위해서는 동적 계획법 사용이 적절하다.