Greedy Algorithms Exercises

Exercises by Ibrahim Albluwi
Solutions by Qabas Alkaisi

Exercise 1

You are the head of the photography club at PSUT. The Open Day is approaching and you are planning for it. You have received a number of new cameras and are planning to use them to provide media coverage. The schedule of the Open Day includes events organized by different clubs, where each event has a specified start time and end time, and different events can happen at the same time.

For each of the following scenarios, mention if the problem can be solved using a greedy strategy or not. If it can, describe the greedy strategy with less than 15 words.

Each of these problems corresponds to a problem we covered in class.

1. Some club members want to minimize the number of used cameras (to reduce wear and tear).

   Problem: Find the minimum number of cameras needed to cover all the events (a camera can cover only 1 event at a time).

   - ☐ **Cannot** be solved using a greedy algorithm.
   - ☐ **Can** be solved using a greedy algorithm.

   Greedy Strategy _____

2. Some club members insist that you don't use more than one camera. To avoid looking bad, they want to cover the maximum number of events possible.

   Problem: Find the maximum number of events that can be covered using a single camera (a camera can cover only 1 event at a time).

   - ☐ **Cannot** be solved using a greedy algorithm.
   - ☐ **Can** be solved using a greedy algorithm.

   Greedy Strategy _____

3. Some club members argued that you should maximize the profit you can get from the Open Day.

Problem: Given a charge that every club is willing to pay to cover their event, find the maximum amount of money that you can get if you use a single camera (a camera can cover only 1 event at a time).

- ☐ **Cannot** be solved using a greedy algorithm.
- ☐ **Can** be solved using a greedy algorithm.

Greedy Strategy

4. A club member noted he can borrow from a relative (for free) enough cameras to cover all of the events. However, he noted that the club's cloud storage subscription allows uploading at most 20 GBs.

   Problem: Given how many MBs/GBs of video storage each event requires, Find the maximum amount of money that you can get from the Open Day such that the total amount of video storage does not exceed 20 GBs (assuming that every event can get its own camera and that you will cover the whole duration of every event you decide to cover).

   - ☐ **Cannot** be solved using a greedy algorithm.
   - ☐ **Can** be solved using a greedy algorithm.

   Greedy Strategy

5. A club member suggested that it is OK if an event is not covered for its full duration. In this case, the club would charge for only the duration covered.

   Problem: Find the maximum amount of money that you can get from the Open Day, such that the total video recording storage does not exceed 20 GBs (assuming that an event can be covered partially and that there are enough cameras to cover every event).

   - ☐ **Cannot** be solved using a greedy algorithm.
   - ☐ **Can** be solved using a greedy algorithm.

   Greedy Strategy

Exercise 2

The **Infinite 0-1 Knapsack** Problem allows taking each item an infinite number of times (but taking a fraction of an item is not allowed).

Remember: `values[i]` = value of item i `weights[i]` = weight of item i

`N` = number of items `W` = Knapsack capacity

Goal = Pick items whose sum of values is maximum and sum of weights <= `W`

A. Provide a counterexample for the following greedy strategy for solving this problem:

```
SOLVE(values [], weights [], W, N):
      Sort the items based on their value in descending order
      result = 0, i = 0
      WHILE (W > 0 AND i < N):
            WHILE (weights[i] <= W):    //Take as much as possible
                  W -= weights[i]       //from the most valuable item
                  result += values[i]
            i+=1
```

Make sure to specify: (1) the item weights, (2) the item values, (3) the knapsack capacity, (4) how much value the greedy strategy gives, and (5) what the optimal solution is.

Exercise 3

PART 1. Consider the **0-1 knapsack** problem covered in class. Mention very briefly a greedy strategy for solving each of the following special cases of the problem.

A. If each of the items has a weight of exactly 1 Kg.

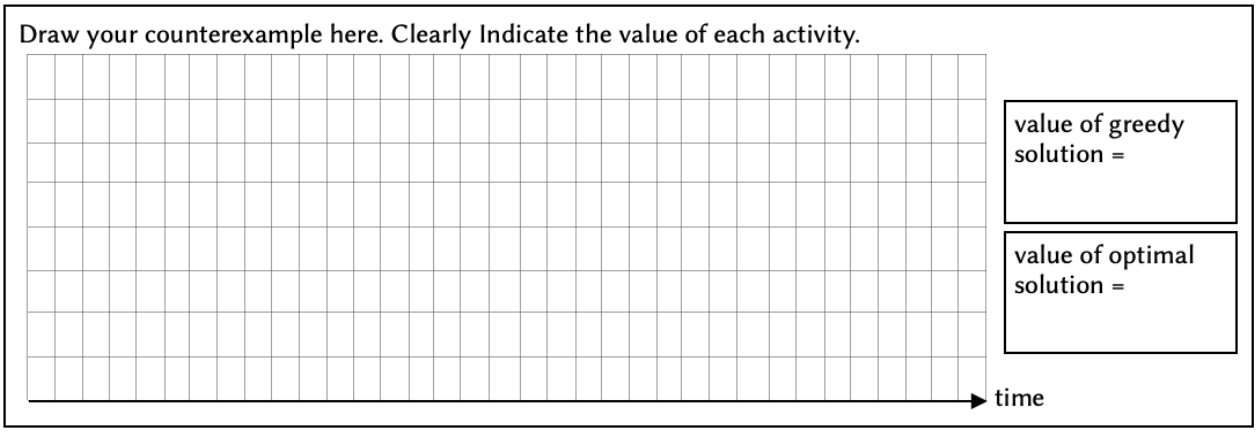
B. If each of the items has a value of exactly $1.

PART 2. Consider a variant of the **Weighted Activity Selection** problem covered in class, where each activity has a start time, finish time, and a value. The goal of the variant is still to maximize the total value of the taken activities, but the constraint is changed as follows:

- **Original** constraint: At any timestamp, no more than **1** activity can be picked.
- **New** constraint: At any timestamp, no more than **2** activities can be picked.

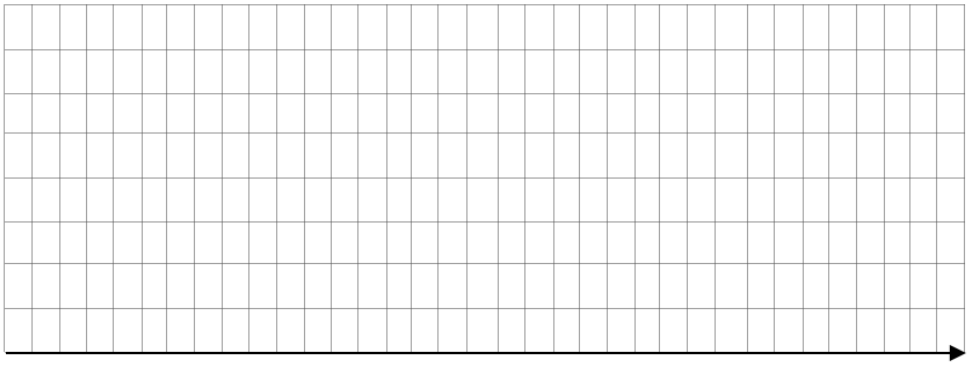
Provide a counterexample for each of the following greedy strategies showing that they don't always work.

A. Sort the activities by length in ascending order. For each activity in the sorted list, take the activity if taking it is allowed.



B. Sort the activities by $\frac{value}{length}$ in descending order. For each activity in the sorted list, take the activity if taking it as allowed.

Draw your counterexample here. Clearly Indicate the value of each activity.



value of greedy solution =

value of optimal solution =

Solutions

Exercise 1

1. It **can** be solved with a greedy algorithm.
   Strategy: Assign a camera to the earliest starting event first.

2. It **can** be solved with a greedy algorithm.
   Strategy: Cover the earliest finishing event first. This is the Activity Selection problem we covered in class.

3. It **cannot** be solved with a greedy algorithm.
   This is the Weighted Activity Selection Problem we solved using Dynamic Programming.

4. It **cannot** be solved with a greedy algorithm.
   This is the 0-1 Knapsack problem we solved using Dynamic Programming.

5. It **can** be solved with a greedy algorithm.
   Strategy: Take the event whose charge/time is maximum. This is the Fractional Knapsack problem we covered in class.

Exercise 2

The **Infinite 0-1 Knapsack** Problem allows taking each item an infinite number of times (but taking a fraction of an item is not allowed).

Remember: `values[i]` = value of item i `weights[i]` = weight of item i

`N` = number of items `W` = Knapsack capacity

Goal = Pick items whose sum of values is maximum and sum of weights <= `W`

A. Provide a counterexample for the following greedy strategy for solving this problem:

```
SOLVE(values [], weights [], W, N):
      Sort the items based on their value in descending order
      result = 0, i = 0
      WHILE (W > 0 AND i < N):
            WHILE (weights[i] <= W):    //Take as much as possible
                  W -= weights[i]       //from the most valuable item
                  result += values[i]
            i+=1
```

Make sure to specify: (1) the item weights, (2) the item values, (3) the knapsack capacity, (4) how much value the greedy strategy gives, and (5) what the optimal solution is.

```
    items   0  1   2

weights = [1, 3, 10]      W = 10

 values = [2, 4, 9]

Greedy Solution: Take item 2 as it is the most valuable
                 Profit = 9
Optimal Solution: Take item 0 once and item 1 three times
                 Profit = 2 + 4 + 4 + 4 = 14
```

Exercise 3

PART 1. Consider the **0-1 knapsack** problem covered in class. Mention very briefly a greedy strategy for solving each of the following special cases of the problem.

C. If each of the items has a weight of exactly 1 Kg.

> Take the **most valuable** item first

D. If each of the items has a value of exactly $1.

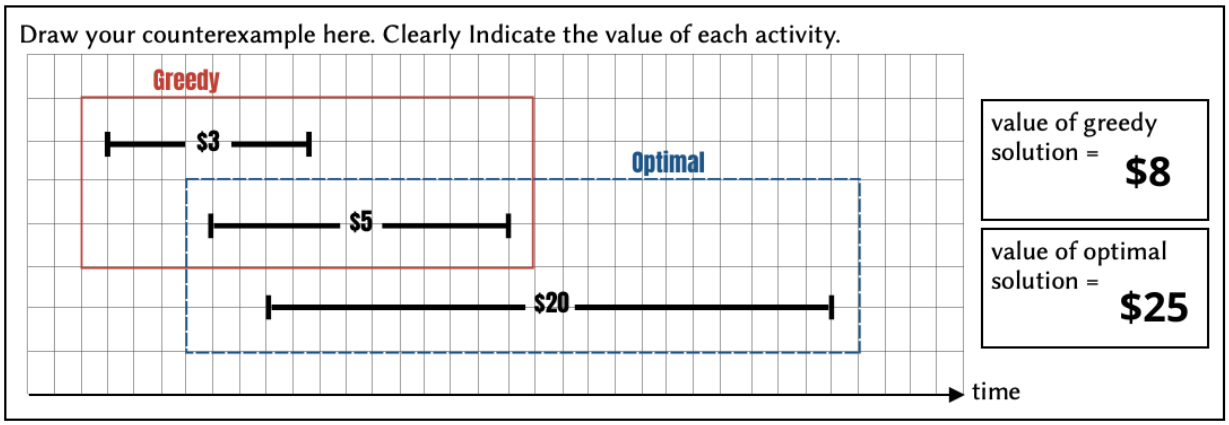
> Take the **lightest** item first

PART 2. Consider a variant of the **Weighted Activity Selection** problem covered in class, where each activity has a start time, finish time, and a value. The goal of the variant is still to maximize the total value of the taken activities, but the constraint is changed as follows:

- Original constraint: At any timestamp, no more than 1 activity can be picked.
- New constraint: At any timestamp, no more than 2 activities can be picked.

Provide a counterexample for each of the following greedy strategies showing that they don't always work.

A. Sort the activities by length in ascending order. For each activity in the sorted list, take the activity if taking it is allowed.



B. Sort the activities by $\frac{value}{length}$ in descending order. For each activity in the sorted list, take the activity if taking it as allowed.