CPSC 320 2022W1: Tutorial Quiz 4, Individual Portion

CS ID

Please enter your 4 or 5 digit CSID in this box:

1 Watching performances on Granville Street

On a clear, sunny Saturday, you get together with a group of friends to collect JockeyMon badges at the Granville Street JockeyMon badges festival. We will make the following assumptions about this festival:

- The participating stores are equally spaced, and the position of each participating store is an integer. Walking from the store at position i to the store at position $i + 1$ takes exactly 5 minutes.
- There are n available badges; one badge is available every 5 minutes, in exactly one store. Badge i is available at the store at position p_i.
- Collecting a badge is instantaneous, but you have to be directly in front of the store to get it.
- Your starting point is in front of the store at position 0.

You do not expect to be able to collect every badge, because consecutive badges may be available in stores that are not at successive positions. For instance, badge 4 may be available at the store at position 7 ($p_4 = 7$), and badge 5 may be at the store at position 2 ($p_5 = 2$). However you absolutely insist on collecting badge number n, which means you will need to plan your meandering along the street accordingly. Given this constraint, you want to collect as many badges as possible.

The problem: given the locations $p_1, p_2, \ldots, p_n$ where each of the n badges will be available, find a subset of badges of maximum size that you are able to collect, subject to the requirement that it should contain badge n. Such a solution will be called *optimal*.

1. Give one trivial instance and at least two small instances of this problem. For each instance, indicate the optimal solution.

   Solution: An instance with $n = 0$ is definitely trivial. An instance with $n = 1$ is also fairly trivial: if $p_1 \in \{-1, 0, 1\}$ then the solution is $\{1\}$ (you can collect badge 1). If $p_1 \notin \{-1, 0, 1\}$ then there is no solution.

   Here is one small instance with $n = 2$: $P = [1, -1]$, with solution $\{2\}$ (you can only collect badge 2). Here is another one: $P = [1, 0]$, with solution $\{1, 2\}$.

2. Pick a simple, plausible, but suboptimal greedy approach and then generate a counterexample to its optimality.

   Solution: See question 4.

3. Suppose that at time $5j$ you are collecting a badge at the store at position p_j. Write a necessary and sufficient condition for being able to collect a badge at time $5k$, where $k > j$, and no badges are collected in-between times j and k.

Solution: $\boxed{|p_k - p_j| \leq 5|k - j|}$

4. Show that the following algorithm does not correctly solve this problem, by giving an instance on which it does not return the correct answer.

```
Mark all badges j with |p_n - p_j| > n - j as illegal.
Mark all other badges as legal.
Initialize current position to point 0 at minute 0.
While badges are still available
  Find earliest legal badge j obtainable from current position.
  Add j to S.
  Update current position to point p_j at minute 5j.
Return S
```

Write the instance here:

Solution:

Badge	1	2	3	4
Location	-1	2	3	2

The optimal solution is to move from position 0 to position 2 in time for badge 2, then to position 3 in time for badge 3, and finally back to position 2 in time for badge 4.

And the answer incorrectly returned by the algorithm is to move to point -1 for badge 1, and then back to point 2 for badge 4, missing badges 2 and 3.

CPSC 320 2022W1: Tutorial Quiz 4, Individual Portion

CS ID

Please enter your 4 or 5 digit CSID in this box:

Easily tired singers

You have been charged with the task of scheduling concerts for the famous rock band FosNot. They can give a concert every two days, and each concert takes place in a venue that is categorized as either *small* or *large*. There is a profit associated with each concert they give, which varies depending on the location, venue rental fee, expected attendance, etc. Their lead singer Wovid Abide gets tired easily however, so before giving a concert in a large venue the band needs to take a day off (that's in addition to the regular day off: so if they play in a large venue on day i, then their previous concert would be on day $i - 4$). This mini-vacation allows Wovid to gain the energy they'll need for the following concert.

You are given, for each day i in $\{2, 4, 6, \ldots, 2n\}$, an array A_i that contains a list $A_i[0]$, $A_i[1]$, $\ldots A_i[c_i]$ where element $A_i[j]$ includes the following information:

- $A_i[j]$.large: True if concert $A_i[j]$ would be in a large venue.
- $A_i[j]$.profit: The profit made by giving concert $A_i[j]$.

plus a lot of additional information that's not relevant to your task (the name of the city, the venue, etc).

Given this input, you want to produce a *plan*: a list that specifies for each day i in $\{2, 4, 6, \ldots, 2n\}$, either the index j_i of the concert the band will give on that day or an entry specifying the band will rest. The *profit* of the plan is determined in the natural way: for each i, you add $A_i[j_i]$.profit (unless the band rests on day i).

The problem: Given the input $A[2], A[4], \ldots, A[2n]$, find the plan with maximum profit (such a plan will be called *optimal*).

1. Give one trivial instance and at least two small instances of this problem. For each instance, indicate the optimal solution.

   Solution: The case where $n = 0$ is trivial: the plan is not to give any concert. Here are two small instances:

   (a) $n = 2$, $A_1 = [(\text{small venue}, 3), (\text{large venue}, 4)]$, $A_2 = [(\text{small venue}, 4), (\text{large venue}, 6)]$: the optimal solution is to choose the large venue on day 2 (value 4) and the small venue on day 4 (value 4) for a total plan value of 8.

   (b) $n = 2$, $A_1 = [(\text{small venue}, 3), (\text{large venue}, 4)$, $A_2 = [(\text{small venue}, 4), (\text{large venue}, 10)]$: the optimal solution is to take a day off on day 2 (value 0) and the large venue on day 4 (value 10) for a total plan value of 10.

2. Pick a simple, plausible, but suboptimal greedy approach and then generate a counterexample to its optimality.

Solution: See the next question.

3. Consider the following greedy algorithm, which assumes that there are only two possible concerts on each day i, one in a large venue ($A_i[0]$), and one in a small venue ($A_i[1]$). Show that this algorithm does not correctly solve the problem by giving an instance on which it return an incorrect answer:

```
i = 2
while i ≤ 2n:
  if i < 2n and  A_{i+1}[0] > A_i[1] + A_i[1]:
    output "choose no concert on day i"
    output "choose concert A_{i+1}[0] on day i + 2"
    i = i + 4
  else:
    output "choose concert A_i[1] on day i"
    i = i + 2
```

Solution: Write the instance here (we will use s and l to indicate a small, large venue respectively):

Day	2	4	6
Concert 1	(s, 10)	(s, 10)	(s, 5)
Concert 2	(l, 5)	(l, 25)	(l, 30)

Write the correct answer for your instance here:

Day	2	4	6
Concert	(l, 10)	Holiday	(l, 30)

Profit: $\boxed{40}$

and the answer incorrectly returned by the algorithm here:

Day	2	4	6
Concert	Holiday	(h, 25)	(l, 5)

Profit: $\boxed{30}$

4. Let $NC[i]$, $SC[i]$ and $LC[i]$ denote the values of the best plans for the first $2i$ days that have no concert, a concert in a small venue, or a concert in a large venue on day $2i$. Write a recurrence relation for $SC[i]$.

Solution: Let us denote by S_i the largest profit for a small venue in A_{2i} (with $S_i = 0$ if there is no small venue in A_{2i}) and by L_i the largest profit for a large venue in A_{2i} (with $L_i = 0$ if there is no large venue in A_{2i}). Then

$$SC[i] = S_i + \max\{NC[i-1], SC[i-1], LC[i-1]\}$$

CPSC 320 2022W1: Tutorial Quiz 4, Individual Portion

CS ID

Please enter your 4 or 5 digit CSID in this box:

1 Solving a Tesla dealer's inventory problem

You have been hired to help a new Tesla dealer with an inventory problem. Predictions tell them the quantity of sales to expect over the next n months. Let c_i denote the number of sales they expect in month i. To keep things simple, we will assume that all sales happen at the beginning of the month, and that cars that are not sold are *stored* until the beginning of the next month. The dealer has no cars in stock at the beginning of the first month.

The dealer has room to keep at most S cars, and it costs C to store a single car for a month. They receive shipments of cars from Tesla by placing orders from them, and there is a fixed fee of F each time they place an order, regardless of the number of cars they order. The dealer starts with no cars. The problem is to design an algorithm that decides how to place orders, so as to satisfy all the demands c_i and minimize the total cost. In summary:

- There are two parts to the cost: (1) storage — it costs C for every car on hand that is not needed that month; (2) ordering fees — it costs F each time an order is placed, independently of the size of the order.
- In each month, the dealer needs enough cars to satisfy the demand c_i, but the number left over after satisfying the demand for the month should not exceed the inventory limit S.

1. Give one trivial instance and at least one small instance of this problem. For each instance, indicate the optimal solution.

   Solution: Every instance with $n = 1$ is a trivial instance. An instance with $n = 2$ would be a small instance. For example, $S = 4$, $n = 2$, $C = 3$, $F = 7$ and the following demand array:

Month	1	2
Expected sales	6	2

   The optimal solution for this instance would be to order 8 cars in month 1, and none in month 2, for a total cost of $7 + 2 \times 3 = 13$.

2. Consider an instance of this problem where $S = 5$, $n = 4$, $C = 2$, $F = 8$, and where the demand for cars is as follows:

Month	1	2	3	4
Expected sales	8	6	3	1

This problem has only 4 valid solutions if we assume that the dealer never wants to be left, at the end of the month, with a number of cars that is not zero but isn't sufficient to fulfill the next month's demand. List these four solution below (we've filled the first table for you as an example):

Solution:

Month	1	2	3	4
Cars to order	8	6	3	1

total cost: $\boxed{32}$

Month	1	2	3	4
Cars to order	8	10	0	0

total cost: $\boxed{26}$

Month	1	2	3	4
Cars to order	8	9	0	1

total cost: $\boxed{30}$

Month	1	2	3	4
Cars to order	8	6	4	0

total cost: $\boxed{26}$

3. Consider the following greedy algorithm to solve the problem. Assume function `totalquantity(i,j)` returns the total number of cars needed to fulfill the demand for months $i, i+1, \ldots, j$, and that function `totalcost(i,j)` returns the total cost associated with buying enough cars at the beginning of month i to fulfill the demand for months $i, i+1, \ldots, j$, assuming no cars were left at the end of month $i-1$. Show that this algorithm does not correctly solve the problem, by giving an instance on which it return an incorrect answer:

```
Algorithm PlaceOrders()
  i ← 1
  while i ≤ n do
    quantity, nbmonths ← buyforseveralmonths(i)
    output "buy quantity cars in month i"
    i ← i + nbmonths

function buyforseveralmonths(i)
  for j ← i + 1 to n do
    if totalquantity(i,j) - c_i > S or totalcost(i,j) > totalcost(i,j-1) + F
      break
  return totalquantity(i, j-1), j - i
```

Complete the instance here: $S = 10$, $n = 3$, $C = 1$, $F = 6$

Solution:

Month	1	2	3
Expected sales	3	5	4

Write the correct answer for your instance here:

Solution:

Month	1	2	3
Cars to order	3	9	0

total cost $= \boxed{16}$

and the answer incorrectly returned by the algorithm here:

Solution:

Month	1	2	3
Cars to order	8	0	4

total cost $= \boxed{17}$