

Due: Tuesday, February 14, 2023, 11:59pm

Submit online at (clickable link): <https://forms.gle/2LDWXjGunkQ5Dm6fA>

1 Greedy: Dam It

After graduation, you go work for the Army Corp of Engineers. They do civil engineering projects, but it's handy to have a computer science type on staff.

One day, your unit is sent to investigate a report of a leaky levee. At the site, your unit discovers that the levee has sprung n leaks. Water is spraying out of each leak at the rate of 10 gallon per minute. The civil engineer in your unit quickly determines that the i th leak will take $t[i]$ minutes to patch up. Your unit can only patch one leak at a time, so it will take:

$$\sum_{i=0}^n t[i]$$

minutes to patch up all the leaks. However, as you are patching up the leaks, the unpatched leaks will continue to flood the area at the rate of 10 gallon per minute per unpatched leak. Which leak should you patch first?

For example, suppose $n = 4$, $t[1] = 20$, $t[2] = 5$, $t[3] = 11$ and $t[4] = 8$. If you fixed the patches in numerical order, leak #1 will stop leaking right away, leak #2 will leak for 20 minutes, leak #3 will leak for $20 + 5 = 25$ minutes and leak #4 will leak for $20 + 5 + 11 = 36$ minutes. A total of

$$10 \cdot (20 + 25 + 36) = 10 \cdot 81 = 810$$

gallons of water would have spilled. On the other hand, if you patched the leaks in a different order, say leak #4 first, followed by #2, #3 and #1), then only

$$10 \cdot (8 + 13 + 24) = 10 \cdot 45 = 450$$

gallons will be spilled.

Questions:

1. Describe the set of feasible solutions for a particular instance of this problem. What makes a solution feasible? What does the objective function measure?
2. Devise and describe a greedy algorithm for the problem above that minimizes the total amount of water spilled. State and briefly justify the running time of your algorithm.
3. State a swapping lemma that can be used to prove that your greedy strategy produces the best solution. *Note:* Your swapping lemma should describe how to modify any feasible solution that is different from the greedy solution.
4. Argue that applying a single swap (as described in your swapping lemma) to a non-greedy feasible solution produces another feasible solution.
5. Argue that applying a single swap (as described in your swapping lemma) to a non-greedy feasible solution does not increase its objective value. (Some arithmetic is necessary here.)
6. Prove that the greedy algorithm does indeed produce a feasible solution with the optimum objective value. *Note:* You must show that it is not possible to patch the leaks in an order that results in less water spilled than from following your greedy algorithm. Do not appeal to any unproven general principles. You must incorporate your swapping lemma as you stated.

2 Cost of car ownership

In this question, we consider when it would be advantageous to sell the car you currently own and buy a new one. We assume that you have been given the following information from a “reliable” source:

- $p(i)$ = the price of a new car in year i , for $1 \leq i \leq n$
- $v(i, k)$ = the resale value of a car purchased in year i and sold in year k , for $1 \leq i < k \leq n+1$.
- $m(i, k)$ = the maintenance cost during year k of a car purchased in year i , for $1 \leq i \leq k \leq n$.

The problem is to determine the years y_1, y_2, \dots, y_r when you would purchase new cars such that the total cost of car ownership from years 1 through n is minimized. The total cost is the sum of the maintenance costs for years 1 through n plus the price of each car purchased minus the resale value of each car sold. In addition, you should make the following assumptions.

- you don’t have a car before year 1. (Since you have to buy a new car in year 1, $y_1 = 1$.)
- you only want to purchase new cars.
- you are always able to sell your old car for $v(i, k)$ dollars.
- at the end of n years, you sell your last car for $v(y_r, n+1)$ dollars.

For example, if $n = 10$, $y_1 = 1$, $y_2 = 5$ and $y_3 = 7$, then this solution states that you should purchase a new car in year 1, buy the second car in year 5 and buy the third car in year 7. (You would also sell the first car in year 5, the second car in year 7 and the third car at the beginning of year 11.)

Questions:

1. Define a function, MinCost, that can be used to solve this dynamic programming problem. To do this, you must describe the input parameters to MinCost and its “return value” using English sentences. *Note:* describe the value that MinCost must return in relation to its parameters, *not* how to compute it, that’s the next question.
2. Give a mathematical formula that shows how MinCost can be computed recursively. Then, explain the main parts of the formula in English.
3. Describe how MinCost can be computed bottom up using a dynamic programming table. Be sure to include a description of the dimensions of the table and the order that the entries of the table are to be filled. Which entry has the solution to the original problem?
4. State and briefly justify the running time of your dynamic programming algorithm.