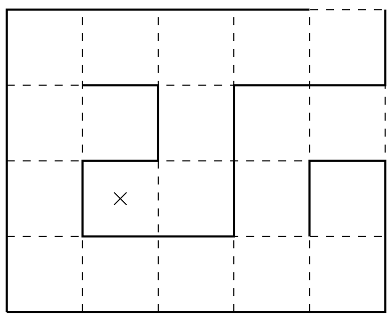
Sample Final Exam Questions

Sample Graph / Greedy Exam Questions

1 Wisconsin Jones (10 points)

1.1. (5 points) As the intrepid computer scientist/archaeologist adventurer Wisconsin Jones, you have discovered the earliest know Bucky Badger icon buried deep beneath Lake Mendota by a long forgotten student society $O\Omega\Theta$ in an elaborate labyrinth. The labyrinth is a grid, where each edge is either a wall or a door, and you must move from square to square by passing through the doors. Due to the instability of this ancient site, you want to recover the icon as fast as possible before it collapses and the waters of Lake Mendota rush in. Moreover, you know that the $O\Omega\Theta$ society have multiple different labyrinths and treasures buried under Lake Mendota so you want to come up with an algorithm that can work for any $n \times m$ labyrinth.

An example of a labyrinth, where dashed lines are doors, solid lines are walls, and $\times$ is the Bucky icon:



(a) (3 points) Give an efficient $O(nm)$ algorithm that finds the fastest route to the treasure given the input of the labyrinth as described. You can assume that the labyrinth data structure allows you to find the entrance cells in constant time, and, for some cell x, you can find cells accessible from x in constant time.

(b) (2 points) Justify the time complexity of your algorithm from Part a.

1.2. (5 points) The labyrinth has m doors, labelled 1 to m, which are connected to one or more of n levers, labelled 1 to n. All the doors are closed and need to be opened by pulling all of the levers one-by-one. Your goal is to figure out the order of lever pulls to ensure that all doors are open at the end.

The behaviour of the levers is described by an $n \times m$ matrix M. When lever i is pulled, door j behaves as follows:

- If $M_{i,j} = 1$, then door j opens.
- If $M_{i,j} = -1$, then door j closes.
- If $M_{i,j} = 0$, then door j remains at its previous state (open or closed).

Since the $O\Omega\Theta$ society was able to place Bucky in the labyrinth, there is guaranteed to be a sequence of lever pulls to open all the doors.

Suppose that there are $m = 5$ doors, and $n = 3$ levers with matrix

$$M = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & -1 \\ 0 & -1 & 0 & 0 & 1 \end{bmatrix}$$

The optimal order to pull the levers is $2, 3, 1$.

Let $\sigma[i+1, n]$ be the last $n - i$ lever pulls. In other words, the $i+1$ to nth levers pull, and let D be the set of doors where the last non-zero value (as described in M) based on the order of σ from $i+1$ to n is a 1. Let C be the set of doors not in D.

Two of your computer scientist/archaeologist assistants proposed two possible heuristics for choosing the ith level to pull:

1. Choose the ith lever to be a lever containing no -1s in the columns of M restricted to the doors of C, and the levers not in σ.
2. Choose the ith lever to be a lever containing the most 1s in the columns of M restricted to the doors of C, and the levers not in σ.

Neither assistant completed CS577 so it is up to you figure out which heuristic to use. You'll only have one chance to open the doors so it is critical that the proposed algorithm works.

(a) (1 point) Which heuristic is correct?

(b) (1 point) Give a counter-example for the other heuristic than indicated in Part a.

(c) (3 points) Prove that the heuristic indicated in Part a is correct.

2 The TP Standard, Part 1 (10 points)

It is the year 2220 and the most valuable commodity in the known universe is a roll of 2-ply toilet paper. It is lost to history exactly why Humanity moved to the TP standard... Only that it has been this way for 100s of years.

You are working in logistics for Royale Bank – a joint Irving-RBC company[1][2]. You are asked to design an order processing algorithm. Given the value of a single roll of toilet paper, shipping costs are negligible compared to a roll. You must ship rolls in bundles of fixed quantities. The goal of your algorithm is to ship the exact quantity ordered using the least number of bundles per order.

2.1. (10 points) Consider the case where Royale Bank ships n bundles sizes $b_1 < b_2 < \cdots < b_n$, where $b_1 = 1$ and $b_{i+1} = c \cdot b_i$ for some integer $c > 1$. That is b_{i+1} is a multiple of b_i.

(a) (1 point) Consider the bundle sizes: $1, 3, 6, 12$. What is the optimal number of bundles sent for an order of 42 rolls of toilet paper? Describe how many of each bundle size.

(b) (3 points) Provide a **greedy algorithm** to determine the optimal number of bundles and how many of each size to ship.

(c) (2 points) Prove that your algorithm terminates.

[1] Good Canadian humour, eh? – I did it again!

[2] Royale is a toilet paper company owned by Irving (a huge Canadian oil/forestry/other company), and RBC is the Royal Bank of Canada one of the 5 big Canadian Banks.

(d) (3 points) Prove that your algorithm is optimal. Clearly state your approach: stay ahead vs exchange argument.

(e) (1 point) Prove the running time complexity of your algorithm.

Sample Divide and Conquer Questions

3 Political Divide (10 points)

3.1. (10 points) You are in a room with n politicians $p_1, p_2, \ldots, p_n$, where n is a power of 2. Each politician belongs to one of k parties, where $k > 1$. You want to know which politicians are in which party, but it is impolite to directly ask. However, you notice that whenever two politicians that belong to the same party are introduced to one another, they shake hands. When two politicians that belong to different parties are introduced, they stare at each other angrily.

(a) (6 points) Using divide and conquer and the $O(1)$ method `introduce`(p_i, p_j) to introduce two politicians p_i and p_j to each other, design an algorithm to group the politicians into their k parties.

(b) (3 points) Give the recurrence relation for your algorithm in Part a.

(c) (1 point) Give the asymptotic runtime based on the recurrence given in Part b.

4 Wordplay (10 points)

4.1. (10 points) The English word "below" has a curious property. Its letters are all arranged in ascending alphabetical order. We will call this an *alphabetical* word.

(a) (2 points) How quickly can you check whether a given n-letter word is alphabetical? Briefly describe an algorithm for doing so, and state its asymptotic run-time.

(b) (2 points) Since there are very few alphabetical words in the English language, we are also interested in words that are almost alphabetical. A word is called k-alphabetical if removing k (or fewer) letters would make it an alphabetical word. A friend of yours proposes the following algorithm to determine whether a word is k-alphabetical. In the following algorithms, for a string w, $w[0]$ is the first letter, $w[-1]$ is the last letter:

Algorithm 1: CALCALPHA

Input : A word w.
Output: k, where w is k-alphabetical, and w' an ordered string.

```
if w has length 1 then
    return (0, w)
end
(k1, w1) := calcAlpha(Front-half of w)
(k2, w2) := calcAlpha(Back-half of w)
k3 := k1 + k2
Initialize w3 to an empty string
while w1 or w2 is not empty do
    if w1 is empty then
        foreach letter ℓ ∈ w2 do
            Append to w3 if ℓ ≥ w3[−1]; otherwise discard and add 1 to k3
        end
    else if w2 is empty then
        Append w1 to w3
    else
        if w1[0] ≤ w2[0] then
            Remove first character from w1 and append to w3.
        else
            Discard first character from w2 and add 1 to k3.
        end
    end
end
return (k3, w3)
```

Algorithm 2: isAlpha

Input : A word w and a integer k.
Output: true if w is k-alphabetical, false otherwise.
$(k', w') :=$ calcAlpha(w)
return $k' \leq k$

Write a recurrence relation describing the run-time of this algorithm and state the resulting asymptotic run-time.

(c) (3 points) Unfortunately, your friend's algorithm does not always work. Provide an example of a word and a number k such that the isAlpha reports the word is NOT k-alphabetical when it actually is. (Your word does not need to be a real English word.) Draw the recursion tree to demonstrate your example.

(d) (3 points) In the previous section, you showed that your friend's algorithm does not always return the optimal answer. Prove that it only errs in one direction. It never reports a word is k-alphabetical when that word in fact is not k-alphabetical.

Sample Dynamic Programming Questions

5 World Traveller (10 points)

You're planning on flying around the world. Your plane has a fuel tank of capacity T and can fly 1 kilometre per litre of fuel. You've mapped out your flight plan and it consists of n cities where you can stop to refuel. For each city i, you have two values f_i, the price per litre of fuel in city i, and c_i the cost to stop and refuel. That is, the cost to refuel in city i is $f_i \cdot \ell + c_i$, where ℓ is the number of litres purchased.

For any two sequential cities, i and $i+1$, you know the distance $d_{i,i+1}$ in kilometres which is no more than T by design and, as chance would have it, all $d_{i,i+1}$ are integral.

Before heading out, you decide to calculate the optimal refuelling schedule. That is, calculate how much fuel to purchase at each city given that you start at city 1 and have no gas in the tank.

5.1. (2 points) Assume that $c_i = 0$ for all cities. Give an optimal greedy algorithm that runs in $O(n \log n)$ time. Note: you do not have to prove that the algorithm is optimal.

5.2. (8 points) Assume that $c_i \geq 0$ for all cities, design a dynamic program to calculate the optimal refuelling with a worst-case runtime of $O(nT^2)$.

(a) (2 points) Describe the array/matrix M used in your dynamic program. Be sure to give the dimensions and any cells that can be immediately initialized.

(b) (1 point) In which cell of M will the optimal fuel cost located?

(c) (2 points) Give the Bellman equation for your dynamic program.

(d) (2 points) Give an efficient algorithm to recover the amount of fuel purchased in each city from the array/matrix M populated based on your dynamic program.

(e) (1 point) We would say that this dynamic program is:

A. pseudo-polynomial

B. polynomial

6 The TP Standard, Part 2 (10 points)

6.1. (10 points) Consider the more general case where Royale Bank ships n bundles sizes $b_1 < b_2 < \cdots < b_n$, where $b_1 = 1$.

(a) (1 point) Consider the bundle sizes: $1, 2, 6, 9$. What is the optimal number of bundles sent for an order of 12 rolls of toilet paper? Describe how many of each bundle size.

(b) (1 point) Again, consider the bundle sizes: $1, 2, 6, 9$. What is the greedy solution for an order of 12 rolls of toilet paper? Describe how many of each bundle size. (Note it should not be optimal!)

(c) (8 points) Let $T \geq 1$ be the total number of rolls of toilet paper ordered. Design a dynamic program to calculate the optimal number of bundles to send with a run-time complexity of $O(nT)$.

i. (2 points) Describe the array/matrix used in your dynamic program. Be sure to give the dimensions and any cells that can be immediately initialized.

ii. (1 point) In which cell will the optimal solution be located?

iii. (2 points) Give the Bellman equation for your dynamic program.

iv. (2 points) Describe how to recover the number of each bundle size to send for the optimal solution.

v. (1 point) We would say that this dynamic program is:

A. pseudo-polynomial

B. polynomial

Sample Network Flow Questions

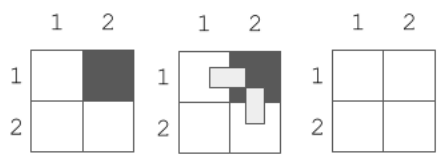
7 Garden Addition (10 points)

7.1. (10 points) The University of Wisconsin Botanical Garden is hiring you to help plan their new addition. It will contain n^2 individual garden tiles, arranged in a square. Each of these gardens will be connected to the horizontally and vertically adjacent gardens, so that visitors can easily traverse them. However, the ground that they want to build on has many hills, so some gardens are at low height and some are at high height. The heights are provided in an array $H[i, j]$, where $H[i, j] = 1$ if garden (i, j) is at high height, and 0 if it is at low height. If a garden is at a different height than an adjacent garden, they will need to add a ramp between those two gardens to ensure that they are connected. Each ramp costs R dollars. They are also capable of raising or lowering individual gardens (changing the height from low to high or from high to low), at a cost of W dollars each, in order to reduce the number of ramps needed. Your job is to decide which gardens to raise/lower (if any) in order to minimize the cost of building the new addition.

In the following example:

$$n = 2,\ R = 2,\ W = 3, \text{ and } H = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

If we do not change the heights of any gardens, then we will need to build one ramp from garden $(1, 1)$ to garden $(1, 2)$ and another ramp from garden $(2, 2)$ to garden $(1, 2)$, costing $2R = 4$. Otherwise, if we lower garden $(1, 2)$, as shown on the right, then we will not need to build any ramps, so the total cost will be $W = 3$, which is the optimal cost for this instance.



(a) (3 points) We will solve this problem using network flow. For this part, you only need to describe a graph that has a maximum flow value equal to the optimal cost. Give a precise description of all nodes, edges, and capacities. Do not use lower bounds or node demands.

(b) (4 points) Justify why the maximum flow in the graph you created has value equal to the minimum cost.

(c) (3 points) Give an algorithm (using the graph you constructed in part (a)) which returns which gardens should have their heights changed in order to minimize the total construction cost. You should return an $n \times n$ array A where $A[i, j] = 1$ if garden (i, j) needs to have its height changed and 0 otherwise.

8 The Grocery Store (10 points)

8.1. (10 points) With the social distancing guidelines, some grocery stores are labelling aisle with a direction only allowing shoppers to "flow" down the aisle one way. Signs throughout the store also remind shoppers to stay 6 feet apart at all times.

A grocery chain would like your help figuring out their stores' capacities during the crisis. For each store, aisles end at an entrance, an exit, or an intersection (with other aisles). The chain can provide you with an estimate of how many people can pass down each aisle while observing social distancing, but they're not sure how to calculate the overall store capacity from this data.

(a) (4 points) Describe a flow network that can be solved using the Ford-Fulkerson method to determine how many shoppers can pass through the store with a single entrance, and a single exit in one hour (while observing social distancing requirements).

(b) (3 points) Of course, it is not enough for shoppers to simply pass from the entrance to the exit. At least one shopper must pass through each aisle **intersection** during an hour. Otherwise, the featured products can't be sold.

How would you alter the graph from Part 8.1a to calculate whether the suggested number of shoppers is feasible given these additional requirements?

(c) (3 points) The store manager thinks they understand the Ford-Fulkerson method, but only in a graph with a single source, single sink, capacity values for edges, and no node properties at all. Describe how to modify your solution from Part 8.1b so that an instance can be solved using only the basic Ford-Fulkerson method.

Sample Intractability Questions

9 Speed Cams (10 points)

In your city, there has been a rash of traffic accidents due to speeding. You have been tasked by the city to help distribute speed cameras. The city would like to purchase high-tech speed cameras that, when place at an intersection, can monitor all the incident roads. The city council would like to know what is the minimum number of cameras needed to monitor all the roads, where the city has q intersections and r distinct road segments, a portion of a road between two intersections.

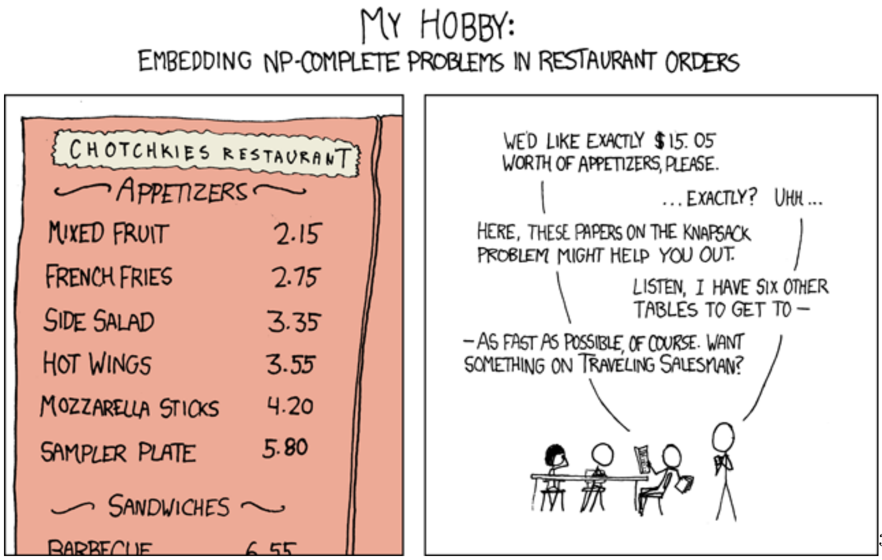
9.1. (1 point) Give a brute-force algorithm to determine the minimum number of cameras?

9.2. (1 point) What is the worst-case runtime of your brute-force algorithm?

9.3. (1 point) Re-state the camera placement optimization problem as a decision problem.

9.4. (7 points) Prove that the decision version of the camera placement problem is in NP-Complete.

10 Vaccine Distribution (10 points)



[3]

10.1. (10 points) The world is in the midst of a pandemic and a vaccine has finally been developed, however, the vaccine is very difficult to produce in large quantities. Since it is impossible to vaccinate everyone at the moment, the CDC is trying to determine who should receive the vaccine first. They have an idea that the vaccines should be spread through the population in a way that helps everyone by universally increasing resistance. We define the idea of a person p being "resistant" to the virus if at least one of the following holds.

- Person p has been vaccinated.
- Person p interacts with someone who has been vaccinated.

Thus, we phrase the *Vaccine Distribution Problem* as follows. Given a group of people, a record of the people with whom each person regularly interacts,and a number of available vaccines k, is there a way to distribute the vaccines so that every person is resistant?

(a) (3 points) Describe the graph that encodes an instance of this problem.

[3] https://xkcd.com/287/

(b) (7 points) Despite great effort to develop an efficient algorithm to decide vaccine distribution, no one has succeeded, and time is of the essence. Unfortunately for the CDC, it appears that their efforts may be fruitless. Prove that *Vaccine Distribution* is NP-Complete.