ShanghaiTech University

CS240 Algorithm Design and Analysis

Fall 2024

Problem Set 2

Due: 23:59, Nov. 10, 2024

1. Submit your solutions to the course Gradescope.
2. If you want to submit a handwritten version, scan it clearly.
3. Late homeworks submitted within 24 hours of the due date will be marked down 25%. Homeworks submitted more than 24 hours after the due date will not be accepted unless there is a valid reason, such as a medical or family emergency.
4. You are required to follow ShanghaiTech's academic honesty policies. You are allowed to discuss problems with other students, but you must write up your solutions by yourselves. You are not allowed to copy materials from other students or from online or published resources. Violating academic honesty can result in serious penalties.

Problem 1:

In a small-town convenience store, the owner, Mr. Wang, enjoys arranging bags of candy by the counter for customers to pick from. Each candy bag contains a different number of candies; for instance, a small bag has 1 candies, a medium bag has 2, and the largest bag holds 5 candies. One day, a customer named Tony comes to the store hoping to buy an exact quantity of candies, such as 12 pieces, to satisfy his sweet cravings.

Mr. Wang tells Tony he can select any number and type of candy bags, as long as they add up precisely to the desired number of candies. You aim to find a way to help Tony obtain his exact candy count using the minimum number of bags. If it's impossible to achieve the exact number of candies with the available bags, you should inform Tony that his request cannot be fulfilled.

Please help determine the minimum number of candy bags Tony would need to obtain his exact desired quantity. If it cannot be achieved, return -1.

In the example provided, the available candy bag types in the store are bags = [1, 2, 5], and Tony's desired quantity is pieces = 12. The output is 3, as the minimum solution requires taking two 5-candy bags and one 2-candy bag.

Problem 2:

You are provided with three sequences A, B, and C. The lengths of these sequences are m, n, and $m + n$, respectively. In other words, sequence C has a length that equals the combined lengths of sequences A and B. Develop an algorithm to determine if the sequences A and B can be interleaved to form sequence C, while maintaining the relative order of characters in A and B.

Example 1: Given $A = aabb$, $B = cba$, and $C = acabbab$, the algorithm should return **true**.

Example 2: Given $A = aabb$, $B = cba$, and $C = aaabbbc$, the algorithm should return **false**.

Problem 3:

On a wall composed of a line of grid cells, there is a painting robot. The robot follows these two painting rules:

1. The robot can only paint using **one color** at a time, and it can only paint a continuous area of the same color.

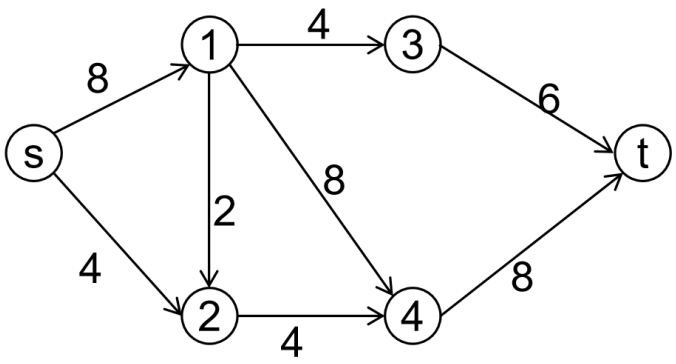
2. The robot can start painting from any grid cell and can **overwrite** areas that have been previously painted.

Given a string s, where each character represents the color of a grid cell (e.g., "r" for red, "b" for blue, "g" for green), your task is to calculate the minimum number of painting operations needed for the robot to cover all the grid cells. Provide the state transition equation and the corresponding pseudocode.

Example 1:
Input: $s =$ "rrrbbb"
Output: 2
Explanation: First, paint the red area "rrr", then paint the blue area "bbb".
Example 2:
Input: $s =$ "rbr"
Output: 2
Explanation: First, paint the entire row with red "rrr", then overwrite the middle cell with blue "b".

Problem 4:

Execute the Ford-Fulkerson algorithm on the flow network shown in the figure below. Display the residual network after each flow update and determine the maximum flow at the end. During each iteration, choose the lexicographically smallest available augmenting path. For example, if two augmenting paths are $s \to 2 \to t$ and $s \to 3 \to t$, select $s \to 2 \to t$ since it is lexicographically smaller. Likewise, if the paths $s \to 3 \to t$ and $s \to 2 \to 4 \to t$ are available, choose $s \to 2 \to 4 \to t$.



Problem 5:

Yansen, a student at ShanghaiTech, recently started playing *Animal Crossing*. The game follows real-time (a week has 7 days).

There are n events, each requiring different materials, and each event is available on a few days of the week. Yansen can complete up to e events per day due to academic and research commitments.

Yansen wants to build a special Gundam and has calculated the materials needed. Starting on Monday, what is the minimum number of days (including rest days) to collect all the materials? (Hint: Network Flow)

Input Description:

The first line contains two integers, n, e ($1 \leq n \leq 1000$, $1 \leq e \leq 100$), separated by a space, representing the number of different events and the maximum number of events Yansen can complete in a day.

The next n lines, each containing $m_i + 2$ integers $c_i, m_i, a_{i1}, a_{i2}, \ldots, a_{im_i}$ ($1 \leq c_i \leq 10^5$, $1 \leq m_i \leq 7$, $1 \leq a_{ij} \leq 7$, for any $j \neq k, a_{ij} \neq a_{ik}$), represent the required number of completions c_i, and the days of the week when the event is available.

Output Description:

Output a single integer representing the minimum number of days Yansen needs to gather all the required materials.

Input:

5 2
2 4 1 3 5 7
2 3 2 4 6
2 2 1 2
2 2 3 4
2 2 5 6

Output:5

Explanation:

The solution can proceed as follows:
Day 1 (Monday): Complete events 1 and 3 once each.
Day 2 (Tuesday): Complete events 2 and 3 once each.
Day 3 (Wednesday): Complete events 1 and 4 once each.
Day 4 (Thursday): Complete events 2 and 4 once each.
Day 5 (Friday): Complete event 5 twice.
Exactly 5 days are needed to finish all requirements.

Problem 6:

Consider an $m * n$ matrix where each entry is a positive integer. The task is to choose a subset of the matrix's elements such that no two selected elements are adjacent. In this context, an element at position (i, j) is adjacent to the elements at $(i, j \pm 1)$ and $(i \pm 1, j)$, while it is not adjacent to the elements at $(i \pm 1, j \pm 1)$. Develop an efficient algorithm to maximize the total sum of the selected elements. Using the method of **network flow**, **design an efficient algorithm** to maximize the total sum of the selected elements and **prove its correctness**.