ShanghaiTech University

CS240 Algorithm Design and Analysis

Fall 2023

Problem Set 2

Due: 23:59, Nov. 21, 2023

1. Submit your solutions to Gradescope (www.gradescope.com).
2. In "Account Settings" of Gradescope, set your FULL NAME to your Chinese name and enter your STUDENT ID correctly.
3. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.
4. When submitting your homework, match each of your solution to the corresponding problem number.

Problem 1:

(**Dynamic Programming**) You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have a security system connected, and it will automatically contact the police if two adjacent houses were broken into on the same night. Given a list of non-negative integers representing the amount of money of each house, determine the maximum amount of money you can rob tonight without alerting the police.

Solution:

Problem 2:

(**Dynamic Programming**) Given two strings `str1` and `str2`, give an algorithm to compute the minimum number of operations required to transform `str1` into `str2`. You have the following three operations permitted on a word:

1. Insert a character
2. Delete a character
3. Replace a character

Solution:

Problem 3:

(**Dynamic Programming**) Given a map $f : \{a, b, ..., z\} \to \{1, 2, ..., 26\}$ defined as $f(a) = 1, f(b) = 2, \cdots, f(z) = 26$. For any string s that only contains characters from $\{a, b, \cdots, z\}$, we can construct a natural map $g : \{a, b, ..., z\}^* \to \{1, 2, ..., 26\}^*$ by applying f to each character of s. That is

$$g(s) = f(s[0]) \ || \ f(s[1]) \ || \ ... \ || \ f(s[n]).$$

For example, $g(ab) = 12$.

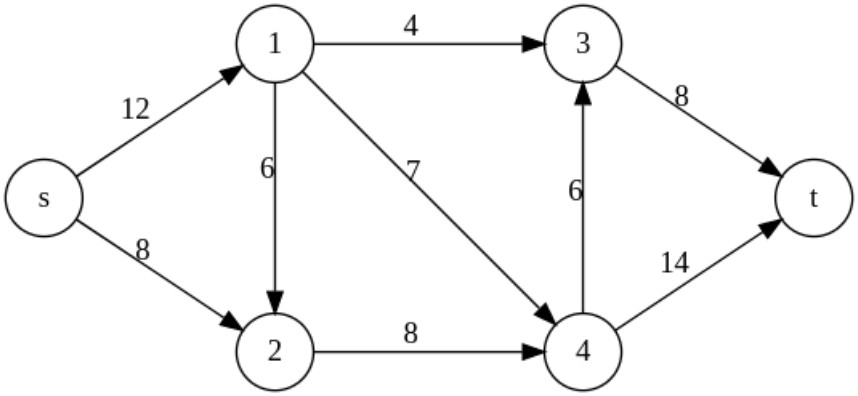
Given a numeric string s' that only contains characters from $\{0, 1, \cdots, 9\}$. Please design an algorithm to find the number of possible string s such that $g(s) = s'$. For example, $s' = 1234$, then s only can be *abcd*, *lcd* and *awd*, so the number is 3.

Notice: **When you provide pseudocode for your algorithm, you must also provide the appropriate comments or it will not be considered correct.**

Solution:

Problem 4:

Run the Ford-Fulkerson algorithm on the flow network in the figure below, and show the residual network after each flow augmentation. For each iteration, pick the augmenting path that is lexicographically smallest. (e.g. if you have two augmenting paths $s \to 3 \to t$ and $s \to 4 \to t$, then you should choose $s \to 3 \to t$, because it is lexicographically smaller than $s \to 4 \to t$. Moreover for augmenting paths $s \to 3 \to t$ and $s \to 2 \to 4 \to t$, you should choose $s \to 2 \to 4 \to t$)



Solution:

Problem 5:

Given a directed graph $G = (V, E)$, and source s, sink t, and all edge capacities in G are positive integers. Design an algorithm in polynomial time to determine if G has a unique minimal s-t cut.

Solution:

Problem 6:

Given a $n * n$ chess board, and there are k obstacles in k squares. You can not put any chess in square with obstacle. You need to put as many knight pieces as you can, such as no knight can attack another knight. Given a knight at (x,y), it can attack (x+1,y+2),(x+1,y-2),(x-1,y+2),(x-1,y-2),(x+2,y-1),(x+2,y+1),(x-2,y-1),(x-2,y+1). Design a minimal s-t cut algorithm to output the maxinum number of knights. (Hint:Divide the chessboard into white squares and black squares, Bipartite Matching)

Solution: