Describe and analyze an efficient algorithm to compute a feasible flow of maximum value.

2. Suppose we are given an $n \times n$ grid, some of whose cells are marked; the grid is represented by an array $M[1..n, 1..n]$ of booleans, where $M[i, j] = \text{TRUE}$ if and only if cell $(i, j)$ is marked. A monotone path through the grid starts at the top-left cell, moves only right or down at each step, and ends at the bottom-right cell. Our goal is to cover the marked cells with as few monotone paths as possible.

![Grid with marked cells and paths](image)

Greedily covering the marked cells in a grid with four monotone paths.

(a) Describe an algorithm to find a monotone path that covers the largest number of marked cells.

(b) There is a natural greedy heuristic to find a small cover by monotone paths: If there are any marked cells, find a monotone path $\pi$ that covers the largest number of marked cells, unmark any cells covered by $\pi$ those marked cells, and recurse. Show that this algorithm does not always compute an optimal solution.

(c) Describe and analyze an efficient algorithm to compute the smallest set of monotone paths that covers every marked cell.

3. Suppose we are given a set of boxes, each specified by their height, width, and depth in centimeters. All three side lengths of every box lie strictly between 10cm and 20cm. As you should expect, one box can be placed inside another if the smaller box can be rotated so that its height, width, and depth are respectively smaller than the height, width, and depth of the larger box. Boxes can be nested recursively. Call a box is visible if it is not inside another box.

Describe and analyze an algorithm to nest the boxes so that the number of visible boxes is as small as possible.

4. Let $G$ be a directed flow network whose edges have costs, but which contains no negative-cost cycles. Prove that one can compute a minimum-cost maximum flow in $G$ using a variant of Ford-Fulkerson that repeatedly augments the $(s, t)$-path of minimum total cost in the current residual graph. What is the running time of this algorithm?

5. An $(s, t)$-series-parallel graph is an directed acyclic graph with two designated vertices $s$ (the source) and $t$ (the target or sink) and with one of the following structures:
   
   - **Base case:** A single directed edge from $s$ to $t$.
   - **Series:** The union of an $(s, u)$-series-parallel graph and a $(u, t)$-series-parallel graph that share a common vertex $u$ but no other vertices or edges.