DESIGN AND ANALYSIS OF ALGORITHMS — HT 2022

Problem Sheet 3

Questions marked with $*$ are not intended to be discussed in tutorials, answers to these questions will be posted on the course webpage.

Dynamic programming

Question 1

$*$ A *contiguous subsequence* of a sequence S is a subsequence made up of consecutive elements of S. For instance, if S is

$$1, 2, 3, -11, 10, 6, -10, 11, -5$$

then $3, -11, 10$ is a contiguous subsequence but $6, 11, -5$ is not. Give a linear-time algorithm for the following task:

> **Input:** A list of numbers, $a_1, \cdots, a_n$.
>
> **Output:** The contiguous subsequence of maximum sum
> (a subsequence of length zero has sum zero).

For the preceding example, the answer would be $10, 6, -10, 11$, with a sum of 17.
(*Hint*. For each $j \in \{1, \cdots, n\}$, consider contiguous subsequence ending exactly at position j.)

Question 2

Consider the following variant of the edit distance problem (without substitutions). The *edit distance*, *ed*(A, B), between two words A and B is the smallest number k such that A can be transformed to B in k moves, where a *move* is the insertion of a letter or the deletion of a letter. For example, *ed*(*cheese*, *chess*) $= 3$, via *cheese* $\rightsquigarrow$ *chees* $\rightsquigarrow$ *ches* $\rightsquigarrow$ *chess*.
Suppose the length of A is n and the length of B is m. For $0 \le i \le n$ let A^i be the prefix of A of length i (so for example *cheese*^{2} = *ch*), and for $0 \le j \le m$ let B^j be the prefix of B of length j.

(a) What are *ed*(A^0, B^j) and *ed*(A^i, B^0)?

(b) Suppose $i, j > 0$. How are *ed*(A^i, B^j) and *ed*(A^{i-1}, B^{j-1}) related if A^i and B^j have the same final letter? How are *ed*(A^i, B^j) and *ed*(A^{i-1}, B^j) and *ed*(A^i, B^{j-1}) related if A^i and B^j do *not* have the same final letter?

(c) Based on the above, design a dynamic-programming algorithm that given two words A and B determines *ed*(A, B).

(d) Illustrate your algorithm on A = *rhymes* and B = *reason*. Explain how to obtain a sequence of *ed*(*rhymes*, *reason*) moves transforming *rhymes* to *reason* from the table your algorithm produces. For which i, j is *ed*(*rhymes*^{i}, *reason*^{j}) largest?

Question 3

(a) Given an unlimited supply of coins of denominations $x_1, \cdots, x_n$, we wish to make change for a value v. This may not be possible: e.g. if the denominations are 5 and 10, then we cannot make change for 12. Give an $O(nv)$ dynamic-programming algorithm for the following problem:

> **Input:** $x_1, \cdots, x_n$ and v
> **Question:** Is it possible to make change for v using coins of denominations $x_1, \cdots, x_n$?

(b) Suppose you are allowed to use each denomination *at most once*. Solve the following modified problem in $O(nv)$ time.

> **Input:** $x_1, \cdots, x_n$ and v
> **Question:** Is it possible to make change for v using each denomination *at most once*?

(*Hint*. Try reducing the change making problem to the knapsack problem.)

Question 4

Consider yet another variation to the change making problem. Suppose now you are allowed to use *at most k coins*. Solve the following problem efficiently:

> **Input:** $1 = x_1 < x_2 \cdots < x_n$; k and v
> **Question:** Is it possible to make change for v using a total of *at most* k coins of denomination $1 = x_1 < x_2 < \cdots < x_n$?

(*Hint*. Set $\textsc{Coins}[m]$ to be the minimum number of coins needed to make value m using denominations $x_1, \cdots, x_n$. Construct a recurrence using $\textsc{Coins}[m]$.)

Question 5

$*$ A subsequence is *palindromic* if it is the same whether read left to right or right to left. For instance, the sequence

$$A, C, G, T, G, T, C, A, A, A, A, T, C, G$$

has many palindromic subsequences, including A, C, G, C, A and A, A, A, A and T (on the other hand, the subsequence A, C, T, is not palindromic).
Devise an algorithm that takes a sequence $\langle x_1, .., x_n \rangle$ and returns the length of the longest palindromic subsequence. Its running time should be $O(n^2)$.

Question 6

Assume that the multiplication of a $p \times q$ matrix by a $q \times r$ matrix requires pqr operations and consider the product

$$M = \underset{[10 \times 20]}{M_1} \times \underset{[20 \times 50]}{M_2} \times \underset{[50 \times 1]}{M_3} \times \underset{[1 \times 100]}{M_4}$$

where the dimensions of each M_i are shown in the brackets. Evaluating M in the order $M_1 \times (M_2 \times (M_3 \times M_4))$ requires 125000 operations, while evaluating M in the order $(M_1 \times (M_2 \times M_3)) \times M_4$ requires only 2200 operations. Show that the optimal order in which to evaluate a product of n matrices can be found in time polynomial in n by using a dynamic-programming algorithm.

Graphs: Paths and Cycles

Question 7

(a) How many undirected graphs are there with vertex-set $\{1, \cdots, n\}$ and no self-loops (that is, no edges from a vertex to itself)?

(b) What is the minimum number of edges that an undirected graph with n vertices and k connected components can have?

(c) What is the minimum number of edges that an undirected connected graph with n vertices can have?

(d) What is the maximum number of edges that an undirected acyclic graph with n vertices can have?

Question 8

$*$ (a) How many edges does a tree with n nodes have?

(b) Writing t_n for your answer to the previous part, is it true that an acyclic undirected graph with n vertices and t_n edges is a tree?

(c) Show that between any two nodes of a tree there is exactly one simple path.

(d) Show that if an edge incident on a node u is added to a tree then the resulting graph has exactly one cycle starting at u.

(e) Show that if an edge that occurs in a cycle in a connected graph is deleted, then the resulting graph is connected.

Question 9

Suppose that G is a finite, connected, undirected graph without loops. The *degree* of a vertex in G is the number of edges incident on the vertex. A *circuit* is a nonempty finite path in which the target of the last edge is the source of the first edge and in which no edge occurs twice. (Note that a vertex can occur more than once in a circuit.)

(a) Prove that if the degree of each vertex in G is greater than or equal to 2, then G has a circuit.

(b) A circuit in G is *Eulerian* if it traverses every edge of G exactly once.

   i. Show that if G has an Eulerian circuit then every vertex of G has even degree.

   ii. Show that if every vertex of G has even degree then G has an Eulerian circuit.
   (*Hint.* Use induction on the number of edges in G and part (a).)

Depth-First Search and Connected Components

Question 10

Perform depth-first search for the graph on the right. Whenever there is a choice of vertices, pick the one that is alphabetically first. Classify each edge as a tree, forward, back or cross edge, and obtain discovery and finishing time of each vertex.

$$
\begin{tikzcd}
A \arrow[r] \arrow[dd] & B \arrow[r] \arrow[d] & C \arrow[d] \\
& E \arrow[r] \arrow[d] & D \arrow[ul] \arrow[d] \\
F \arrow[ur] \arrow[r, bend left] & G \arrow[l, bend left] & H \arrow[l]
\end{tikzcd}
$$

Question 11

$*$ Run the strongly connected components algorithm on the directed graph on the right. Whenever there is a choice of vertices to explore, always pick the one that is alphabetically first.

$$
\begin{tikzcd}
A \arrow[r] \arrow[d] & B \arrow[r] \arrow[d] & C \arrow[d] \\
D \arrow[dr] & E \arrow[ul] \arrow[d] & F \arrow[d] \\
G \arrow[u] & H \arrow[l] \arrow[ur] \arrow[r, bend left] & I \arrow[l, bend left]
\end{tikzcd}
$$

Answer the following questions:

(a) In what order are the strongly connected components found?

(b) Draw the SCC graph.

(c) What is the minimum number of edges you must add to this graph to make it strongly connected?

Question 12

Suppose a depth-first search is carried out on a directed graph G, and that F is the DFS forest produced. Are the following assertions correct? Justify your answers.

(a) If there is a path from u to v in F then $d[u] \leq d[v]$.

(b) If $d[u] < d[v]$ and there is a path from u to v in G, then there is a path from u to v in F.

(c) If G is strongly connected then F is a tree.

(d) If F is a tree then G is strongly connected.

Question 13

$*$ Modify the depth-first search algorithm of an undirected graph to show that it can be used to identify the connected components of G, and that the depth-first forest contains as many trees as G has connected components.