Final

Note

This is a closed-book exam. Each problem accounts for 10 points, unless otherwise marked.

Problems

1. Given two strings $A = bbaab$ and $B = baaabb$, compute the minimal cost matrix $C[0..5, 0..6]$ for changing A character by character to B. Please describe also the detail of calculation for the entry $C[5, 6]$.

2. Below is an algorithm skeleton for depth-first search utilizing a stack; assume that the input graph is undirected and connected. Modify the algorithm so that it prints out (the edges of) a DFS tree of the input graph. You should try to make as few changes as possible, while maintaining the overall structure of the original algorithm.

```
Algorithm Simple_Nonrecursive_DFS (G, v);
begin
    push v to Stack;
    while Stack is not empty do
        pop vertex w from Stack;
        if w is unmarked then
            mark w;
            for all edges (w, x) such that x is unmarked do
                push x to Stack
end
```

3. Given as input a connected undirected graph G, a spanning tree T of G, and a vertex v, design an algorithm to determine whether T is a valid DFS tree of G rooted at v. In other words, determine whether T can be the output of DFS under some order of the edges starting with v. Please present your algorithm in an adequate pseudo code and make assumptions wherever necessary. Explain why the algorithm is correct and give an analysis of its time complexity. The more efficient your algorithm is, the more points you get for this problem.

4. Consider Dijkstra's algorithm for single-source shortest paths as shown below. You may find in the literature two bounds, namely $O(|V|^2)$ and $O((|E| + |V|) \log |V|)$, for its time complexity. Why is this so? What does this difference imply?

```
Algorithm Single_Source_Shortest_Paths(G, v);
begin
    for all vertices w do
        w.mark := false;
        w.SP := ∞;
    v.SP := 0;
    while there exists an unmarked vertex do
        let w be an unmarked vertex s.t. w.SP is minimal;
        w.mark := true;
        for all edges (w, z) such that z is unmarked do
            if w.SP + length(w, z) < z.SP then
                z.SP := w.SP + length(w, z)
end
```

5. Prove that if the costs of all edges in a given connected graph are distinct, then the graph has an unique minimum-cost spanning tree.

6. Finding a small vertex cover for an arbitrary undirected graph is difficult, but is much easier for trees; a *vertex cover* of a graph G is a set of vertices such that every edge in G is incident to at least one of these vertices. Design an efficient algorithm to find a minimum-size vertex cover for a given tree. Please present your algorithm in an adequate pseudo code and make assumptions wherever necessary. The more efficient your algorithm is, the more points you will be credited for this problem. Explain why your algorithm is correct and give an analysis of its time complexity.

7. Below is a solution to the single-source shortest path problem using the dynamic programming approach, which we have discussed in class:

   Denote by $D^l(u)$ the length of a shortest path from v (the source) to u containing *at most* l edges; particularly, $D^{n-1}(u)$ is the length of a shortest path from v to u (with no restrictions).

$$D^1(u) = \begin{cases} length(v,u) & \text{if } (v,u) \in E \\ 0 & \text{if } u = v \\ \infty & \text{otherwise} \end{cases}$$

$$D^l(u) = \min\{D^{l-1}(u), \min_{(u',u)\in E}\{D^{l-1}(u') + length(u',u)\}\}, \quad 2 \le l \le n-1$$

   Please explain why the solution allows edges with a negative weight (as long as there is no cycle with a negative weight). How is this different from Dijkstra's algorithm? Please explain.

8. Consider an $m \times n$ matrix of non-negative integers. A path through the matrix consists of a sequence of cells of the matrix, starting anywhere in the first column (Column 1) and terminating in the last column (Column n). Two consecutive cells

in a path constitute a step of the path and should go from one column to the next column and either remain on the same row or go up or down one row. The weight of a path is the sum of the integers in the cells along the path.

Please try to reduce the problem to a shortest path problem in graphs. You should clearly identify the target problem and describe the conversion of an arbitrary input of the problem under consideration to one of the target problem.

9. In the proof (discussed in class) of the NP-hardness of the clique problem by reduction from the SAT problem, we convert an arbitrary boolean expression in CNF (input of the SAT problem) to an input graph of the clique problem.

   (a) Please illustrate the conversion by drawing the graph that will be obtained from the following boolean expression:

   $$(\overline{w} + x + y + z) \cdot (\overline{x} + \overline{y} + z) \cdot (w + y + \overline{z}).$$

   (b) The original boolean expression is satisfiable. As a demonstration of how the reduction works, please use the resulting graph to argue that it is indeed the case.

10. Solve one of the following two problems. (Note: if you try to solve both problems, I will randomly pick one of them to grade.)

    (a) The hitting set problem is as follows.

    > Given a collection C of subsets of a set S and a positive integer k, does S contain a hitting set for C of size k or smaller, that is, a subset $S' \subseteq S$ with $|S'| \leq k$ such that S' contains at least one element from each subset in C?

    Prove that the hitting set problem is NP-complete. (Hint: the proof of NP-hardness is by reduction from the vertex cover problem to this problem.)

    (b) The subgraph isomorphism problem is as follows.

    > Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, does G_1 have a subgraph that is isomorphic to G_2? (Two graphs are isomorphic if there exists a one-one correspondence between the two sets of vertices of the two graphs that preserves adjacency, i.e., if there is an edge between two vertices of the first graph, then there is also an edge between the two corresponding vertices in the second graph, and vice versa.)

    Prove that the subgraph isomorphism problem is NP-complete. (Hint: the proof of NP-hardness is by reduction from the Hamiltonian cycle problem to this problem.)

11. (You may substitute this problem from the midterm for one of the previous problems. Please identify the problem, if any, to be replaced. You should not attempt to solve the replaced problem.)

    Below is a variant of the insertion sort algorithm.

```
Algorithm Insertion_Sort (A, n);
begin
    for i := 2 to n do
        x := A[i];
        j := i;
        while j > 1 and A[j − 1] > x do
            A[j] := A[j − 1];
            j := j − 1;
        end while
        A[j] := x;
    end for
end
```

Draw a decision tree of the algorithm for the case of $A[1..3]$, i.e., $n = 3$. In the decision tree, you must indicate (1) which two elements of the original input array are compared in each internal node and (2) the sorting result in each leaf. Please use X_1, X_2, X_3 to refer to the elements (in this order) of the original input array.

Appendix

- The vertex cover problem: given an undirected graph $G = (V, E)$ and an integer k, determine whether G has a vertex cover containing $\leq k$ vertices. (A *vertex cover* of G is a set of vertices such that every edge in G is incident to at least one of these vertices.)
  The vertex cover problem is complete.

- The Hamiltonian cycle problem: given an undirected graph G, does G have a Hamiltonian cycle? (A Hamiltonian cycle in a graph is a cycle that contains each vertex, except the starting vertex of the cycle, exactly once.)

  The Hamiltonian cycle problem is NP-complete.