# Exercises for graphs

1. A directed graph is said to be oriented if it has no cycles of length at most 2, that is, if $(i, j)$ is an edge then $(j, i)$ is not an edge. Describe an $O(n + m)$ time algorithm to determine whether a given directed graph is oriented. Describe an $O(n + m)$ time algorithm to determine whether a given directed graph has a cycle of odd length, and if so, to find it. The problem of deciding whether a directed graph has a cycle of even length was unsolved for many years. There is now a polynomial-time algorithm known, but it is a 50 page paper. For undirected graphs, describe a simple $O(n + m)$ time algorithm to find a cycle of odd or even length if it exists. In an undirected graph, a cycle must have length at least 3. There is a polynomial-time algorithm to determine whether an undirected graph has a cycle of length divisible by $k$ for any $k \geq 1$, but even for $k = 3$, it is very complicated.

2. Given an undirected graph, and 3 nodes $u, v, w$, describe an $O(n + m)$ time algorithm to determine whether there exists a path from $u$ to $w$ that passes through $v$. Just concatenating a path from $u$ to $v$ and a path from $v$ to $w$ does not necessarily give a path, since nodes may be repeated. There is no polynomial-time algorithm for this in the case of directed graphs. Given 4 nodes $a, b, x, y$ in an undirected graph, describe a polynomial-time algorithm to check if there exists a path from $a$ to $b$ and a path from $x$ to $y$ in the graph, such that the two paths have no nodes in common. This is not easy, and it may help to try and find some necessary and sufficient conditions on the graph, for such paths to exist. Again, this problem has no efficient algorithm for directed graphs.

3. A tournament is a directed graph in which for every pair of distinct vertices $(i, j)$, exactly one of $(i, j)$ or $(j, i)$ is an edge. The name comes from the fact that such graphs represent the result of a tournament in which each player plays every other. An edge $(i, j)$ indicates player $i$ defeated player $j$. Prove that every tournament has a Hamilton path, that is, a path including all nodes. Suppose a tournament is given by an $n \times n$ adjacency matrix. Ignoring the time for reading the matrix, show how a Hamilton path can be found in $O(n \log n)$ time. Prove that a tournament has a Hamilton cycle iff it is strongly connected, and describe an $O(n^2)$ time algorithm to find one, if it exists.

4. Given a collection $S$ of $n$ strings, each of length $l$, describe an $O(nl)$ time algorithm to decide if there exists a string $s$ such that every string in $S$ is a substring of $s$ and every substring of $s$ of length $l$ is in $S$. In other words, $S$ is the set of all substrings of $s$ of length $l$. If it is not, can you find the minimum number of strings of length $l$ that must be added to $S$, so that it is the set of substrings of length $l$ of some string $s$?