Problem Set 4 Solutions

Problem 1.  [From CLR 23.5-1]

Solution

b) Is the correct statement. In order to prove it is sufficient to show how to construct the required graph for every \( k \). Consider the directed graph \( G_k = (V_k, E_k) \) such that \( V_k = \{1, \ldots, k\} \) and \( E_k = \{(1, 2), (2, 3), \ldots, (k - 1, k)\} \). Clearly the above graph has \( k \) strongly connected components. However if we add the edge \( (k, 1) \) to it we obtain a graph \( G'_k = (V'_k, E'_k) \) such that \( V'_k = V_k \) and \( E'_k = E_k \cup (k, 1) \). The graph \( G'_k \) so constructed has only one connected component.

Problem 2.  [Connected Components]

Solution

The graph \( G \) has two connected components. The first one contains the vertices \( \{1, 4, 5, 6\} \) and the second one the vertices \( \{2, 3\} \)

Problem 3.  [Splitting Trees]

Solution

The basic idea of the algorithm is that given a tree \( T = (V, E) \), with \( |E| = |V| - 1 \), to count the number of nodes contained in all its possible subtrees. This idea is implemented using simple variants of the procedures DFS and DFS-Visit described in CLR. Basically, we use a global variable \( nodes \) to store the size (number of nodes) of the subtree we are dealing with.

Pseudocode

\[
\text{DFS-T}(T, k) \quad \text{for each node } u \in V \\
\quad \text{color}[u] = \text{white} \\
\quad \pi[u] = \text{NIL} \\
\text{for each node } u \in V \\
\quad \text{if color}[u] = \text{white then DFS-T-Visit}(T, u, k)
\]
DFS-T-Visit($T, u, k$)

\text{color}[u] = \text{gray} \\
\text{nodes} = 1 \\
\text{for each } v \in \text{Adj}[u] \\
\quad \text{if color}[v] = \text{white} \text{ then} \\
\quad \quad \pi[v] = u \\
\quad \quad \text{nodes} = \text{nodes} + \text{DFS-T-Visit}($T, v, k$) \\
\text{color}[u] = \text{black} \\
\text{if } \text{nodes} = k \text{ or } \text{nodes} = n - k \text{ then} \quad \text{print “The requested edge is } (\pi[v], v)” \\
\quad \text{remove } (\pi[v], v) \\
\text{exit} \\
\text{return nodes}

The correctness of the above algorithms follows immediately from the correctness of the DFS and DFS-Visit procedures in CLR. From the analysis in CLR it is also possible to conclude that the time complexity of the solution above is bounded by $O(|V|)$.

**Problem 4.** [Max Flow] 

**Solution**

The flow is not optimum. As a matter of fact the residual network contains the augmenting path $(s, b), (b, c), (c, d), (d, a), (a, t)$ having residual capacity 1.