Algorithms: Solutions 8

Problem 1
Write algorithms for converting (a) an adjacency-list representation of a graph into an adjacency matrix and (b) an adjacency matrix into adjacency lists.

We denote the adjacency list of a vertex u by *Adj-List*$[u]$, and the adjacency-matrix element for vertices u and v by *Adj-Matrix*$[u, v]$. The time complexity of both algorithms is $\Theta(V^2)$.

(a) Converting adjacency lists into a matrix.

LISTS-TO-MATRIX(G) ▷ G is represented by adjacency lists
for each $u \in V[G]$
    do for each $v \in V[G]$
            do *Adj-Matrix*$[u, v] \leftarrow 0$
        for each $v \in$ *Adj-List*$[u]$
            do *Adj-Matrix*$[u, v] \leftarrow 1$

(b) Converting an adjacency matrix into lists.

MATRIX-TO-LISTS(G) ▷ G is represented by an adjacency matrix
for each $u \in V[G]$
    do initialize an empty list *Adj-List*$[u]$
for each $u \in V[G]$
    do for each $v \in V[G]$
            do if *Adj-Matrix*$[u, v] = 1$
                    then add v to *Adj-List*$[u]$

Problem 2
Suppose that G is a weighted undirected graph, where all weights are integers between 1 and 5, and let u and v be two vertices of G. Give an algorithm that finds a minimal-weight path from u to v.

We construct a new graph by replacing every edge of length n in the original graph with n unit edges, as shown in the picture. That is, we replace every edge of length 2 with two unit edges, every edge of length 3 with three unit edges, and so on. We then run breadth-first search in the new graph, with the source vertex u, which finds a shortest path from u to v. If the original graph has V vertices and E edges, then the new graph has at most $V + 4 \cdot E$ vertices and $5 \cdot E$ edges, and the running time of the breadth-first search is $O(V + 4 \cdot E + 5 \cdot E) = O(V + E)$.