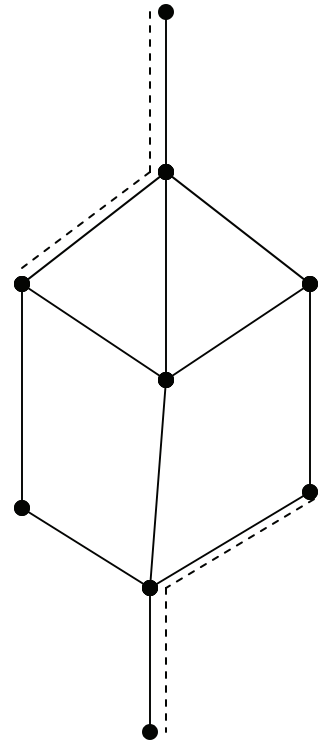


Paths and Circuits: Applications

The *Chinese Postman Problem* is to find a smallest Eulerian pseudograph containing a given graph G .

Algorithm of solution:

1. Find all the odd vertices in G .
2. For each partition of them into pairs, find the length of a shortest path connecting the two vertices in each pair.
3. Take the partition for which the total length is minimal, and duplicate the edges along the shortest paths.



Why the resulting pseudograph is Eulerian?

Definitions

A *digraph* G is a pair of sets: $G = (V, E)$, called vertices (V) and *arcs* (E), where V is nonempty, and each element of E is an ordered pair of distinct elements of V (e.g., $u \rightarrow v$).

For a given vertex in a digraph,

indegree is the number of arcs directed into the vertex

outdegree is the number of arcs directed out of the vertex

The *indegree sequence* and *outdegree sequence* are defined similarly to degree sequence in a graph

Proposition : The sum of indegrees equals the sum of outdegrees.

Notions that we introduced previously for graphs naturally extend to digraphs (for example: a walk, a Hamiltonian cycle, the adjacency matrix, isomorphism, a weighted graph), with the following remarks:

Any *walk* in a digraph respects orientation of arcs. The same rule applies to notions defined in terms of a walk.

Definition

A digraph is *strongly connected* iff there is a walk from any vertex to any other vertex.

Proposition : a digraph is Eulerian iff it is strongly connected and for any vertex the indegree equals the outdegree.

The *Bellman-Ford Algorithm* for finding the shortest distances in a weighted digraph $G = (\{v_1 \dots v_n\}, E, Weight)$ from v_1 to all other vertices $\{v_i\}$: (here E is the set of arcs)

```

For  $i = 1$  to  $n$ , for  $j = 1$  to  $n$ 
    If  $i = j$ ,  $w(i,j) := 0$ 
    Else if  $v_i v_j \in E$ ,  $w(i,j) := Weight(i,j)$ 
    Else  $w(i,j) := \infty$ 
For  $j = 1$  to  $n$ ,  $d_0(j) := \infty$ ,  $p(j) := v_1$ 
 $d_0(1) := 0$ 
For  $i = 1$  to  $n$ , for  $j = 1$  to  $n$ 
    Find the  $k$  that minimizes  $m = d_{i-1}(k) + w(v_k, v_j)$ 
    If  $m < d_{i-1}(j)$ ,
         $d_i(j) := m$ ,  $p(j) := v_k$ 
    Else  $d_i(j) := d_{i-1}(j)$ 
If  $d_n(j) = d_{n-1}(j)$  for all  $j = 1..n$ , then
    output "Shortest paths lengths are ",  $d_n$ 
    output "Second last vertices are ",  $p$ 
Else output "No shortest path."

```

