Fleury's Algorithm With More Than Two Odd Degree Vertices

Fleury's algorithm finds Eulerian tours and trails with two common cases; however, the algorithm does not work independently with more than two odd-degree vertices. The algorithm requires modifications to accommodate the case of more than two odd vertices through checking the tour or path returns to the starting vertex from another odd-degree vertex, originally showing that the graph would be disconnected. Instead, adding an edge from the starting edge to the latest iteration nth vertex creates the pair of odd degree vertices to be even and starting the algorithm at another odd vertex in the graph.

The modified version of Fleury's algorithm starts the same as the original: defining a connected graph $G = (V, E)$, picking a starting at any odd degree vertex $v_0 \in V$, defining the trails $C_0 = v_0$ and $C_n = v_0, e_1, v_1, \ldots, e_n, v_n$ and picking an arbitrary edge from the starting vertex v_0. The iteration count is the same where each vertex is denoted as v_n traversing to another vertex v_{n+1} through e_{n+1} outside of the trail C_n. However, the main modification is once the algorithm traverses to another vertex of odd-degree, the goal is to detect when the algorithm disconnects. By checking whether e_{n+1} is a bridge through defining a graph $G_n = G - E(C_n)$ shows which edge from both the v_0 and v_n causes the graph to disconnect. Usually, we define $C_{n+1} = C_n, e_{n+1}, v_{n+1}$, but following the modification should be defined as such if and only if the vertex v_n is even. When the vertex v_n is odd, we define C_{n+1} as $C_{n+1} = C_n, e_{n+1}, v_0$.

The trail C_{n+1} now contains a repeated vertex v_0, which follows the condition: if $2v_0 \in C_{n+1}$, then redefine C_{n+1} as $C_{n+1} = C_n,\ e_{n+2,}\ v_n$. Redefining the trail C_{n+1} this way creates a repeated edge between two odd degree vertices, now both of even degree. Since the graph could contain more than one pair of odd-degree vertices, the algorithm will set $n = 0$ and restarting at another odd-degree vertex. The algorithm will repeat until there are only two edges of odd-degree remaining, which the original version of Fleury's algorithm will complete an Euler trail in the graph while satisfying the condition $n = |E|$. The modified version of the algorithm can be represented symbolically by the following:

Input: A connected *(p, q)* graph *G = (V, E)* ***Output:*** An Eulerian circuit *C* of *G*

Method: I) If the starting vertex is even, expand the trail C_n while avoiding bridges in *G* - C_n until no other choice remains

II) If the starting vertex is odd, expand the trail C_n while checking if the trail traverses to another odd vertex.

i) If such is the case, find bridge in *G* - C_n and add another edge,

ii) Otherwise, traverse graph as ***I)***

1. Choose any $v_0 \in V$ and let $C_0 = v_0$ and $n \leftarrow 0$

2. Suppose that the trail $C_n = v_0, e_1, v_1, ..., e_n, v_n$ has already been chosen:

a. At v_n, choose any edge e_{n+1} that is not on C_n and that is not a bridge of the graph $G_n = G - E(C_n)$, unless there is no other choice

b. Define C_{n+1}:

If $d(v_n)$ is odd

then define $C_{n+1} = C_n, e_{n+1}, v_0$

Since $2v_0 \in C_{n+1}$, choose e_{n+2} and redefine $C_{n+1} = C_n, e_{n+2}, v_n$

Let $n \leftarrow 0$, $C_0 = v_0$, and pick another v_0 where $d(v_0)$ is odd

else define $C_{n+1} = C_n, e_{n+1}, v_{n+1}$ and let $n \leftarrow n + 1$

3. If $n = |E|$

then halt since $C = C_n$ is the desired circuit

else go to ***2.***