Final Algorithm to Solve More Than Two Odd Degree Vertices

The final algorithm produced from this project was mainly inspired by Fleury's algorithm, where most components have been changed to accommodate multiple conditions, including directional edges, to ensure non-crossing edges. The primary method is to double every edge such that the graph becomes Eulerian, regardless of the number of odd degree vertices. Another condition is to avoid any crossing edges causing the graph to disconnect, which is included in the original. However, the main modification was to have sub-conditions for the direction of edges and can cause an edge to be redirected to another vertex if one was chosen to cause a bridge. Once all edges have been processed without any bridges, the modified algorithm produces an Eulerian circuit as all edges are even and do not cross each other per the added requirements. 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^* from *G*

Method: Induce the graph G^* from *G* to double every edge of the original graph to make it Eulerian. After induction, expand the trail C_n while avoiding bridges in G^* - C_n until no other choice remains.

1. Let G^* be a connected digraph that is induced from G where $G^* = (V(G), 2E(G))$, choose any $v_0 \in V$, let $C_0 = v_0, m \leftarrow 0$, 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. Set the orientation of v_0:

If v_0 is a left boundary vertex on the graph G^*:

Then the orientation of v_0 to v_1 through an edge e_1 follows the direction towards the right of v_0.

Else, the orientation of v_0 to v_1 through an edge e_1 follows the direction towards the left of v_0.

B. At v_n, choose any edge e_{n+1} that is not on C_n and follows:

I) e_{n+1} is contained in the path C_n that returns to v_0 such that $v_{n+1} = v_0$, while not being a bridge of the graph $G_m = G^*{-}E(C_n)$, which suggests the graph crosses another edge e_{n+2}.

If the edge e_{n+1} is a bridge, go to ***II).***

Else, go to ***C).***

II) Choose another e_{n+1} that results in one of the following, where ***i) – iii)*** can be arbitrarily chosen:

i) If v_n and v_{n+1} are boundary vertices that are incident of e_{n+1} such that a path that traverses them in the left or right directions which do not create a bridge:

Then define such in ***C)*** towards the respective direction, go to ***C).***

ii) If the vertices v_n and v_{n+1} are incident of e_{n+1} such an e_{n+1} traverses them in the upwards direction while not creating a bridge:

Then define such in ***C)*** towards the upward direction, go to ***C).***

iii) If the vertex v_n is a left or right boundary vertex, and v_{n+1} is a boundary vertex while incident downward of v_n:

Then e_{n+1} is defined such in ***C).***towards the downward direction, go to ***C).***

iv) If ***i) – iii)*** do not produce any result, define downwards and go to ***C).***

c. Define $C_{n+1} = C_n, e_{n+1}, v_{n+1}$ and let $n \leftarrow n + 1$ and $m \leftarrow m + 1$.

3. If $n = |E(G^*)|$:

Then halt since $C = C_n$ is the desired circuit.

Else go to ***2B***.