Note on the Chinese postman problem

In this problem, a postman starts at a post office and must deliver mail to all houses on all streets and come back to the post office. The postman wants to minimizes the time taken to do so. We assume that every street is reachable from the post office and all streets are lined with houses.

We can formulate this problem in terms of graphs as follows.

Problem 1. Input: A connected graph $G = (V, E)$, weights $w_e \geq 0$ for each edge $e \in E$. **Output:** A minimum weight closed walk containing all edges.

Note that the location of the post office itself is not important. The requirement that we must return to it is.

Note that since we must visit every edge at least once, we might as well only count the total weight of edges we count more than once. In particular, if G is contains an Eulerian cycle, this cycle is the optimal solution (correct output). The reformulated problem is as follows.

Problem 2. Input: A connected graph $G = (V, E)$, weights $w_e \geq 0$ for each edge $e \in E$. **Output:** A minimum weight set of edge of G that we need to "double" to make the graph Eulerian.

Note that we would never traverse any edge more than twice in the optimal solution. This is since removing two parallel edges (if there are more than 2) reduces the degree of the endpoints of those edges by exactly 2 and does not make the graph disconnected. So the new graph contains an Eulerian cycle if the original did.

This problem can be solved in polynomial time.

To do so, we need an algorithm for finding the shortest path and an algorithm to find the minimum weight maximum matching.

Algorithm 1. Compute the degrees of all vertices in G
Let S be the set of odd degree vertices in G
Build H, the weighted complete graph with vertex set S and weights
$w_{u,v}$ =shortest path distance from u to v in G
Find a minimum weight maximum matching M in H.
Let F be the union of all edges of G on paths corresponding to edges of M.
Return F.

We now give some intuitive reason why the output of this algorithm is correct.

First doubling every edge on a path P increases the degree of the endpoints of P by one and the degree of all other vertices by 2. Thus, only the parity of the degree of its endpoints are changed. Since our path contains each odd degree vertex as an endpoint exactly once, doubling F results in a graph with only even degree vertices.

Second, suppose F^* is an optimal set of edges to double. We will extract paths from F^*. Since F^* is a solution, vertices in (V, F^*) have the same degree parities as vertices in G. Therefore, if we start at a vertex of S and keep walking in (V, F^*) without repeating any edge. We only get stuck at another vertex of S. We remove the edges we traversed and we remove the endpoints from S. Pick a new vertex in S and repeat this process. This gives us a set of paths. We can compare the weight of these paths to the edges between the corresponding endpoints in H. We see that the weight of F must therefore be less or equal to the weight of F^*. Thus, proving the optimality of F.