Eulerian path and circuit

Loh Bo Huai Victor

January 24, 2010

1 Eulerian Trails and more

In this chapter, **Eulerian trails** or loosely known as **Euler path** and **Euler Tour**, **Chinese Postman Problem**, **Hamilton paths** and the **travelling salesman problem** (TSP) will be discussed.

1.1 Eulerian Trails

1.1.1 Definitions

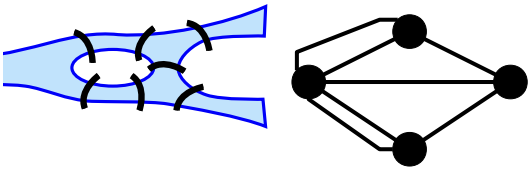
A **trail** is the sequence of vertices such that for each of the vertices, there is an edge to its successor vertex and no edge is used twice in the trail. A **trail** is **open** if the starting vertex is different from the ending vertex and a **trail** is **closed** if the starting and ending vertices are the same vertex. G is **eulerian** if there exists a closed eulerian trail in G. Likewise, G is **semi-eulerian** if there exists an open eulerian trail in G.

1.1.2 History

It all started with the Seven Bridges of Königsberg problem. In the town of Königsberg, the Pregel River separates the town into four separate land masses. The town was connected by seven bridges at various parts of the town back in the 18th century (Refer to diagram below). Many people from Königsberg wondered if it is possible to cross all seven bridges exactly once in one journey and return to the starting location. All of them failed.

In 1736, Leonard Euler (a Swiss mathematician) solved the problem. Euler realized that the problem could be solved by considering the degrees of each of the vertices. Euler

proved that eulerian circuit only exists on an undirected graph if there are no vertices with odd degrees. In the Seven Bridges of Königsberg problem, three of the vertices have a degree of 3 and one has a degree of 5. Therefore, a close eulerian trail does not exist for the problem.



It is interesting to note that Königsberg does not exist on any map as this seaport which was once the capital of East Prussia is now part of Russia Federation and goes by the name of Kaliningrad.

1.1.3 Conditions for existence of eulerian trails

Closed Eulerian Trail

- A closed eulerian trail exists on an undirected graph if and only if the graph is connected and every vertex has an even degree.
- A closed eulerian trail exists on a directed graph if and only if the graph is connected and the in-degree of every vertex is equal to its out-degree.

Open Eulerian Trail

- An open eulerian trail exists on an undirected graph if the graph is connected and only two of the vertrices are of odd degree. These two vertrices are the starting and ending vertrices of the eulerian trail.

Note: The proof of the above conditions is omitted as it is readily available in any standard books on graph theory.

1.1.4 Fleury's Algorithm

An eulerian trail can be constructed using Fleury's algorithm which dates back to 1883 [4].

```
1  ConstructEulerTrail(){
2     circuitpos = 0
3     EulerTrail(start)    //The starting vertex
4  }
5
6  EulerTrail(u){
7     for (each vertex v adjacent to u){
8       DeleteEdge(u,v)
9       EulerTrail(v)
10    }
11    circuit[circuitpos++] = u
12 }
```

Extension

Multi-edges can be handled by the above algorithm. For each self-loop, add 2 to the degree of the vertex (One in-degree and the other out-degree). To generate the eulerian trail in the correct order for a directed graph, the arcs have to be transverse in the opposite direction.

To generate an open eulerian trail, find a vertex with odd degree (this vertex must be either the starting or ending vertex), and call EulerTrail from it (This vertex will be the starting vertex).

There is a variant of finding an eulerian trail on a mixed graph (which means the graph has both undirected and directed edges). This variant of eulerian trail can be solved using maximum flow. This will be discussed in the later chapters as an application of maximum flow.

1.2 Chinese Postman Problem

The **Chinese postman problem** can be stated as the following: Given an undirected graph, we want to find the shortest closed walk that visit all the edges at least once. This problem first appeared in a Chinese math journal by a Chinese author and Kwan Mei-Ko was the first to propose a solution to it [3]. As a result, Alan J. Goldman suggested the name "Chinese Postman Problem" to Jack Edmonds. Edmonds liked its "catchiness" and adopted it. Edmonds and Johnson were the first to come up with a polynomial solution to the problem [2], and this algorithm is known as Edmond's algorithm.

1.2.1 Edmond's Algorithm

```
Edmonds(Graph G){
   Let S be the set of odd vertices in G
   Compute the shortest path between each pair of vertices a,b in S
   Using the shortest path found in previous step, form a graph H
   Find the minimum matching of H
   For each edge (x,y) in M*, duplicate the edges along the
      shortest path between x and y in G
   Construct a closed eulerian trail
}
```

Edmond's Algorithm is a polynomial algorithm. All pairs shortest pair can be found in $O(V^3)$ using Floyd-warshall Algorithm. The difficult part of the algorithm is to find the minimum weighted matching on a **general graph**. This can be done using a modified version of Edmond's Blossom Shrinking Algorithm and runs in $O(V^3)$. Finding the closed eulerian trail can be easily done in $O(E)$ with Fleury's Algorithm. Therefore, the overall runtime complexity of Edmond's Algorithm is just $O(V^3)$.

1.2.2 Variants

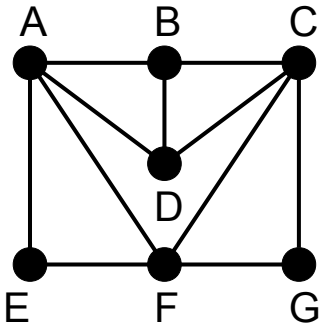
The chinese postman problem on a directed graph can be solved using minimum-cost-maximum-flow and will be discussed in the later chapters as an application of minimum-cost-maximum-flow. There are other variants of chinese postman problem which have been proven to be NP-Complete.

- Chinese postman problem on a mixed graph where there are both undirected and directed edges in the graph
- Min k-chinese postman problem: In this problem, you have k postman instead of 1 postman who start and end at the same vertex and you want all the edges to be visited by at least one of the postman. The constaint to be minimised is the most expensive cycle.
- Rural postman problem: Given a subset S of the edges, you will like to find the minimum cycle that visits all these edges at least once.

1.3 Hamilton Path/Circuit

Another closely related problem is finding a **Hamilton path** in the graph (named after an Irish mathematician, Sir William Rowan Hamilton). Whereas an Euler path is a path that visits every edge exactly once, a Hamilton path is a path that visits every vertex in the graph

exactly once. A **Hamilton circuit** is a path that visits every vertex in the graph exactly once and return to the starting vertex. Determining whether such paths or circuits exist is an NP-complete problem. In the diagram below, an example **Hamilton Circuit** would be AEFGCDBA.



1.4 Travelling Salesman Problem

Traveling Salesman Problem (TSP) is a very well-known problem which is based on Hamilton cycle. The problem statement is: Given a number of cities and the cost of traveling from any city to any other city, find the cheapest round-trip route that visits every city exactly once and return to the starting city.

In graph terminology, where the vertices of the graph represent cities and the edges represent the cost of traveling between the connected cities (adjacent vertices), traveling salesman problem is just about trying to find the Hamilton cycle with the minimum weight. This problem has been shown to be NP-Hard. Even though the problem is computationally difficult, a large number of heuristics and exact methods are known, so that some instances with tens of thousands of cities have been solved.

The most direct solution would be to try all permutations and see which one is cheapest (using brute force search). The running time for this approach is $O(V!)$, the factorial of the number of cities, so this solution becomes impractical even for only 20 cities. A dynamic programming solution solves the problem with a runtime complexity of $O(V^2 2^V)$ by considering dp[end][state] which means the minimum cost to travel from start vertex to end vertex using the vertrices stated in the state (start vertex can be any vertex chosen at the start). As there are $V2^V$ subproblems and the time complexity to solve each sub-problems is $O(V)$, the overall runtime complexity $O(V^2 2^V)$.

1.4.1 Approximation Algorithms

Metric TSP is a special case of TSP which requires the cities to satisfy the triangle inequality. For any 3 cities, A, B and C, the distance between city A and city C must be

at most the distance between city A and city B + the distance between city B and city C. This can also be understood as the direct connection between city A and city B will always be shorter than a detour around any city C. It turns out that quite a number of TSP are metric TSP. In this special case, there exists an approximation algorithm that will always find a tour of length at more 1.5 times the optimal solution [1]. This algorithm is known as **Christofides Algorithm** and was one of the first approximation algorithm.

1.4.2 Christofides Algorithm

```
1  Christofides(Graph G){
2     //G is assumed to be a complete weighted graph
3     Construct a Minimum Spanning Tree T from G
4     Let S be the set of odd vertices in T
5     Form a graph H with vertices S and
6        edges (u,v) in G where u and v are in S
7     Find the minimum matching M* of H
8     Add the edges of M* to T to produce an eulerian multigraph
9     Find a closed eulerian trail in T
10    Transform the trail to hamilton path by skipping visited vertices
11 }
```

This algorithm is polynomial. To construct a MST of G, use Kruskal's or Prim's Algorithm and they run in $O(E \log V)$. The difficult part of the algorithm is to find the minimum weighted matching on a **general graph**. This can be done using a modified version of Edmond's Blossom Shrinking Algorithm and runs in $O(V^3)$. Finding a closed eulerian trail in T can be done in $O(E)$ using Fleury's Algorithm. Converting the closed eulerian trail to hamiltion path can be done in $O(E)$. Therefore, the overall time complexity for Christofides Algorithm is $O(V^3)$.

Proof

Let $w(U)$ denote the total weight of edges described by the graph U and let A be the subgraph of G which contains the same vertex set as G and the edge set of the optimal solution of TSP. It is clear that A contains a spanning tree (remove an edge from A and we get a spanning tree), and since A is a subgraph of G, $w(A) \geq w(T)$. Let B be the subgraph of G which contains the vertex set S and edge set of the optimal solution of TSP. It follows that $w(A) \geq w(B)$. We can also see that $w(M^*) \leq w(B)/2 \leq w(A)/2$. So far we have:

$$w(T) \leq w(A)$$

$$w(M^*) \leq w(A)/2$$

Combining both inequalities, we get $w(T) + w(M^*) \leq w(A) + w(A)/2$. By the metric property, we can thus be assured that the solution by Christofides algorithm will not be more than 3/2 of the optimal solution.

References

[1] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. *Technical Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh*, 1976.

[2] Jack Edmonds and Ellis L. Johnson. Matching, euler tours, and the chinese postman. *Mathematical Programming*, 5:88–124, 1973.

[3] Kwan Mei-Ko. Graphic programming using odd or even points. *Chinese Math.*, 1:273–277, 1962.

[4] USA Computing Olympiad. USACO Training Program Gateway. http://train.usaco.org/usacogate.