Genetic Algorithm to Find Near-Optimal Solutions to the Windy-Postman Problem

A. Pellow-Jarman[0000−0003−0401−0938] and Anban W. Pillay[0000−0001−7160−6972]

University of Kwa-Zulu Natal, University Road, Chiltern Hill, Westville,3629

Abstract. The Windy-Postman Problem is a common variation of the Chinese Postman or Route Inspection Problem, a well-known problem in Graph Theory, in which the shortest closed walk, crossing every edge at least once, must be found in a graph. The Windy-Postman variation adds that the weight of an edge traversal depends on the direction the edge is traversed. The problem is known to be NP-Hard. It serves to model routing problems experienced in many real-world situations. This paper present a genetic algorithm that was used to find near-optimal solutions for the problem. Experimentation shows that the algorithm finds reasonably good solutions that are within two to seven percent of the optimal.

Keywords: Windy-Postman Problem · Route Inspection Problem · Genetic Algorithm.

1 Introduction and Background

The famous Seven Bridges of Königsberg problem is the first example of a routing problem in Graph Theory (it is in fact the first recorded graph theory problem). In the town of Königsberg, seven bridges connected two islands to the land on either side of a river. The problem asked whether it was possible for one to cross all seven bridges exactly once and end up where one started. This problem was solved by Leonard Euler in 1736, and in fact it was in solving this problem that Euler laid the foundation for the field of Graph Theory. Euler showed that this feat was not possible, by proving that it was a necessary condition that if there exists a closed walk in a graph traversing each edge exactly once, then all the vertices in the graph must have even degree. This condition is also sufficient for such a closed walk to exist. Graphs that satisfy this condition are known as Eulerian Graphs, and the closed walk crossing each edge exactly once is known as an Eulerian circuit.

The original Chinese Postman Problem (Route Inspection Problem) is to find the shortest closed walk that visits each edge in an undirected graph at least once. This problem was originally studied by Chinese Mathematician, Kwan Mei-Ko in 1960. This problem differs from the Seven Bridges of Königsberg problem, in that it is a route-finding problem, not a decision problem, which also allows

edges to be repeated, and generalises the problem to any undirected weighted graph.

The Chinese Postman Problem can be solved in polynomial time, and in the special case where the graph is Eulerian, the graph's Eulerian circuit is the optimal solution. Polynomial time solutions are based on this special case, whereby some edges are repeated (these are optimally chosen) in order to produce an Eulerian multi-graph.

The Windy Postman Problem is a generalisation of the Chinese Postman Problem, focusing on finding the shortest closed walk in an undirected graph, but with the difference that each edge may have a different weight depending on the direction of its traversal. Therefore, for any vertices u and v connected by an edge, the weight of edge (u, v) does not have to equal the weight of edge (v, u).

This problem is NP-complete. It has been shown that this problem can be solved optimally in polynomial time for some special cases, such as if the Windy Graph is Eulerian [3], and various heuristic techniques have been proposed that yield good solutions to some cases.

A Genetic Algorithm is a metaheuristic technique based on the principle of natural selection, whereby potential solutions are modelled as chromosomes, which undergo genetic operations over a number of generations, which modify the chromosomes, in order to iteratively improve the current best solution.

2 The Windy Postman Problem

The Windy Postman Problem is stated in [1] as follows, "Let G be a connected undirected graph, such that for every $e \in E$, $(e = (x, y)$, the distance from x to y can be different from the distance from y to x. Find the shortest postman tour for graph G."

The term tour refers to a closed-directed walk in a graph traversing each edge in the graph at least once.

This problem can be restated [1] as:

> Given an undirected windy graph, construct a symmetric directed graph $D = (V, A)$, where the vertex set V is the same as the vertex set of G, and corresponding to each edge $e = (u, v)$ in E, there are two arcs $a' = (u, v)$ and $a'' = (v, u)$ in A, and both a' and a'' have distances corresponding to the distance of the directed edge crossing in G.

Now the original problem is equivalent to finding the shortest directed closed walk w in D, such that for every edge e in E, at least one of the corresponding arcs a' or a'' are in w.

The second, equivalent phrasing of the problem is easier to work within the context of this Genetic Algorithm.

3 Literature Review

As far as the authors are aware, no attempt has been published to solve the WPP using a genetic algorithm. However, many papers have been published on heuristic approaches and exact algorithms to special cases of graphs.

The most comprehensive paper on the WPP ([2]), outlines much of the known mathematical framework of the problem. Notably it provides a proof that the WPP is NP-Hard. This paper presents a special case polynomial time algorithm for an exact solution, for graphs in which for every cycle, the distance around the cycle is the same irrespective of the direction the cycle is traversed. This special case exact solution can be used, as a heuristic solution for the problem, because most practical graphs under consideration almost always satisfy the special case constraint.

The Windy Postman Problem has a polynomial time exact solution on Eulerian graphs [3]. A heuristic algorithm, Win's algorithm, for the general case WPP based upon the Eulerian algorithm is also presented in [3].

A comparison between existing heuristic algorithms and some modifications of those algorithms is given in [9], which shows that in so far as heuristic approaches to the WPP are concerned, Win's algorithm performs the best.

An exact solution using a branch and cut technique is presented in [8] which is shown to find exact solutions for large graphs of up to 3000 vertices and 7000 edges. This is the best-known approach to find exact solutions to the Windy Postman Problem

While no papers could be found attempting a genetic algorithm for the WPP, there are several [5], [6], [7] addressing similar routing problems, most notably optimising a Travelling Salesman Problem with a genetic algorithm approach. These papers give a sense of the power of a genetic algorithm approach to routing problems in general.

4 Problem Analysis

Let $G = (V, E)$ be a windy, undirected graph, and $D = (V, A)$ be its corresponding directed version, as described earlier in the problem statement.

Now the problem of finding a minimum closed walk in a windy graph is not exactly straight forward in the context of a genetic algorithm. Ensuring the edge inclusion constraints while minimising unnecessary edge repetition, all the while trying to minimise the weight of an ordered series of edge crossings seems to be difficult at best.

Some of these difficulties need to be eliminated in order to use a genetic algorithm. First, it is observed that the optimal windy postman tour of G, can be used to construct an Eulerian directed multi-graph, whose Eulerian circuit, is the optimal windy postman tour.

The proof of this is trivial, simply tracing out the arcs in the tour sequentially will form this Eulerian directed multi-graph.

Now instead of looking for an optimal windy postman tour of G, the tours corresponding to Eulerian directed multi-graphs can be found.

It is known that an optimal Windy Postman tour of G, is also a shortest directed closed walk w in D, with, for each edge e in G, there is at least one occurrence of either corresponding arc a' or a'' in w. Let us refer to these arcs, one occurance of either the arc a' or a'' in w for each edge e in G, as α, the set of necessary arcs. Other than the necessary arcs, it also can be seen that the rest of the additional arcs (these are necessarily arcs corresponding to some edge repetitions) in w, are only present in order to complete the closed directed walk w (forming the tour). In other words, these arcs are only necessary to form a spanning Eulerian directed multi-graph, together with the arcs in N. Let us refer to these arcs as β the set of additional arcs.

Now it may not at first seem clear, but

Theorem 1. *Given any set α, that is a set containing an arc a' or a" for each edge e in G, the set of arcs β, needed to form a minimum weighted Eulerian directed multi-graph containing all of the arcs in α can found.*

Proof. Let $X = (V, A_1)$ be directed graph, such that A_1 contains only the arcs present in the set α. Now let $Y = (V, A_2)$ be a directed multi-graph, such that A_2 contains a copy of all the arcs in A, as well as a copy of all the arcs in A_1. Clearly, Y is a super-multi-digraph of X and D, and is strongly connected (as D is a spanning subgraph of Y and is clearly strongly connected), and therefore one can solve the Directed Chinese Postman Problem on Y, by any appropriate method, which effectively creates a super-multi-digraph of Y, $Z = (V, A_3)$, which is an Eulerian directed multi-graph (whose Eulerian circuit is the solution to the Directed Chinese Postman Problem).

Since Z is Eulerian, the indegree of any vertex equals the outdegree of that vertex for all vertices in Z. However, D, which is clearly Eulerian too, is a subgraph of Z, therefore we can form a digraph $P = (V, A_4)$, where A_4 contains all the arcs present in A_3, with each corresponding arc removed for each arc in A (so $P = Z\ D$). Clearly P must be an Eulerian directed multi-graph too.

Now X is a subgraph of P, as Y, the multigraph formed of X and D's arcs, was a subgraph of Z, and P is Z with D removed.

Finally, we let $H = (V, A_5)$ be a directed multi-graph with A_5 having all arcs in A_4, then removing a corresponding arc for each arc in A_1 (so $H = P\ X$).

Clearly the set of arcs of H is the set of additional arcs needed to form a minimum weighted Eulerian directed super multi-graph containing all the arcs in α (those in X). H is the minimum digraph that satisfies the excess in/out degree matching for X.

If this in fact was not the case, then by using the actual minimum weighted set of additional arcs, we could find a better solution to the Directed Postman

Problem solved on Y earlier, which is not consistent.

□

It is also clear that P is therefore the minimum weighted Eulerian directed super-graph of X.

Now the problem of finding a shortest directed closed walk in D is now replaced with the problem of selecting the set of arcs in α, one of either a' or a'' for each edge e in G, that minimises the weight of the Eulerian directed graph (in this example P), formed from these necessary arcs (those forming X) and the additional arcs (those forming H). Once this digraph is found, its Eulerian Circuit can be found using any of the known algorithms, and this circuit will be the optimal windy postman tour of G.

This problem is more amenable to a solution using a Genetic Algorithm then the initial problem.

Furthermore, it is also known that the path p between any two connected vertices u and v in H must be the shortest possible path between those 2 vertices, in the directed graph D, as if this were not the case for any two vertices u and v, another digraph of lighter weight could be formed, by replacing the arcs in p in H with the arcs in the shortest path between u and v.

5 A Genetic Algorithm for the Windy Postman Problem

The Genetic Algorithm is used to decide the set of the necessary arcs, one of either a' or a'' for each edge e in G, that minimises the weight of the Eulerian directed super-graph.

The chromosomes each consist of their own directed graph X, as well a representation of the directed multi-graph H, that together form the Eulerian directed multi-graph P. H is found by using a simple greedy heuristic version of a minimum-cost flow matching, that matches the excess in and out degrees of the vertices of X. The heuristic version matches each excess in-degree of a vertex u with its closest available excess out-degree, being v, using the shortest path between u and v as the path that's added to H. These vertices are matching in a random order.

The Selection Method first employs elitism, initially selecting 10% of the fittest individuals to the next generation. The population is then shuffled around slightly to allow less fit individuals to crossover with a more fit individual. The remaining 90% needed for the next generation are taken from the population by pairing off the neighbouring individuals lying in the top 90% of the population. The bottom 10% of the population are lost.

An individual's fitness is simply calculated as the weight of the Eulerian directed multi-graph formed by the arcs in the stored directed graphs X and H. This fitness is the actual weight of the tour that this individual's choice of X forms. There are 2 crossover methods and a single mutation method.

The first of the crossover methods takes two individuals and, after randomly selecting how many (less than a given bound) arcs to swap, swaps the choice of

arc, a' or a'', corresponding to some randomly chosen edge e in G, between the 2 indivduals in their directed graphs X. So, for example, given individuals C_1 and C_2, if a selected edge for crossing over corresponds to arc a' in C_1 and a'' in C_2, then C_1 will then change that corresponding arc to a'' and C_2 will change that arc to a'. This crossover method may not always result in a change to either individual, as when the edges selected for crossing over were identical in both chromosomes for all attempts. If no crossover occurs, then the mutation operator is applied to both chromosomes, creating a new chromosome and reducing the build-up of identical chromosomes over time.

The second crossover method takes two individuals, C_1 and C_2, and, starting with a random edge in C_1, finds a path along in X of C_1 so that as long as the direction is still uniform in the path for C_1, and the path has not yet formed a cycle, that direction of edge traversal is copied to C_2. This crossover is used less frequently than the first one but is useful to ensure the uniform direction of edges traversed along a cycle.

The mutation operator simply operates on a single chromosome by randomly flipping, for any randomly chosen edge e in G, the corresponding arc a' or a'' in X, from one to the other, a random but bounded number of times. The mutation rate is set at 0.20. Once the fittest chromosome has been found after a set number of generations, the Eulerian circuit is created from the resulting Eulerian directed multi-graph contained in the fittest individual, using Hierholzer's Algorithm.

If the fittest individuals resulting tour is not optimal, we can attempt to further optimise it by ensuring that the following condition is held.

Theorem 2. *In any closed directed-walk T in a graph G, containing at least one of the arcs a' or a'' for each edge e in G, if more than 2 arcs exist between any 2 vertices u and v, and these arcs include both directions (both a' and a" exist between u and v in T), then a better directed-walk T' can be created, satisfying the same edge inclusion constraint, by removing two arcs from T (exactly one of a' and one of a'') which go in opposing directions between u and v.*

Proof. Let T be such a closed directed-walk, and let u and v be vertices such that there exist between u and v more than 2 arcs, including at least one of these arcs going from u to v, call it x, and at least one other arc going from v to u, call it y. Now we can use T to form an Eulerian directed multi-graph as described earlier, let's call it M, with T being the Eulerian circuit of M. Vertices u and v each both have their in-degree equal to their out-degree in M, and since arcs x and y both together add exactly one to the in-degree and out-degree of both u and v, and by constructing a new directed multi-graph N, by removing x and y from M, we have a new Eulerian directed multi-graph, as all of N's vertices too have their in-degrees equal to their out-degrees. N's Eulerian circuit, T', still satisfies the edge inclusion constraint as more than 2 arcs existed between u and v, and therefore T' is a better closed walk satisfying the same edge inclusion constraint. □

6 Evaluation and Validation

The Genetic Algorithm was evaluated on two data sets, one which we randomly generated, and an existing data set, which also contained the optimal values for the solutions. This dataset[1] is described in [8]. It is also noted that none of the graph in both data sets contained multi-edges or loops.

It was found that the WPP on any graph G, containing a cut-vertex, can be reduced to a set of WPP's on some number of subgraphs of G. Thus, in trying to find a solution to the WPP, it is never necessary to try optimising a WPP on a graph containing a cut-vertex, and therefore we prevent all randomly generated test graphs from having any cut-vertices. (It follows immediately from this that the WPP need never be optimised on any graph containing a bridge edge either, since each vertex connected by a bridge edge is itself a cut-vertex)

Theorem 3. *Let $G = (V, E)$ be a graph containing a cut-vertex v, with $G - v$ having k disjoint components labelled $K_1, K_2, ..., K_k$. Solving the windy postman problem on G, can be reduced to solving multiple windy postman problems on a set of subgraphs of G, where each subgraph G_n is the vertex-induced subgraph of G on $\{ V(K_n) \cup v\}$ $\forall$ $n \in \{1, 2, ..., k\}$.*

Proof. Let $G = (V, E)$ be a graph containing a cut-vertex v, with $G - v$ having k disjoint components labelled $K_1, K_2, ..., K_k$, and let G_n be the vertex-induced subgraph of G on $\{\mathrm{V}(K_n) \cup v\}$ $\forall$ $n \in \{1, 2, ..., k\}$. Note each edge e in G is in exactly one vertex-induced subgraph G_i. [All edges are clearly in exactly one vertex-induced subgraph G_i, as an edge either connects two vertices within a disjoint component K_i, or connects a vertex from a disjoint component K_i to v] First it will be shown that the solutions to the WPP on these subgraphs can be used to construct a tour of G, then this tour is shown to be optimal.
Let $T_n = \mathrm{WPP}(G_n)$, $\forall$ $n \in \{1, 2, ..., k\}$. Clearly T_n is a directed-cycle in its respective subgraph G_n, $\forall$ $n \in \{1, 2, ..., k\}$, and all of these directed-cycles can be assumed to begin at vertex v, which is a vertex in all subgraphs and is therefore along a tour of any subgraph. Since G_n is a subgraph of G, T_n is a directed-cycle in G, $\forall$ $n \in \{1, 2, ..., k\}$. Let $T = T_1T_2...T_k$. T is a directed-cycle in G, and T traverses every edge in G, as every edge is in some G_i and hence in some corresponding tour T_i. Therefore T is a tour of G
Now it remains to show that T is an optimal WPP solution for G.
Let $T' = \mathrm{WPP}(G)$ and assume for the sake of contradiction that $\mathrm{w}(T') < \mathrm{w}(T)$ [$\mathrm{w}(T)$ denotes the weight of T]. Let x and y be any two vertices lying in different components of $G - v$. Any walk between x and y must pass through v, the cut-vertex, and therefore any directed-cycle containing both x and y must pass through v at least twice. This implies that any directed-cycle not lying only within one subgraph G_i, can be split up into smaller directed-cycles at vertex v, such that those directed-cycles each lie only in one subgraph. Splitting T' up into such directed-cycles, let C_n denote the set of directed-cycles lying in subgraph G_n, $\forall$ $n \in \{1, 2, ..., k\}$. Since all directed-cycles were formed by splitting larger

[1] http://www.uv.es/ corberan/instancias.htm

directed-cycles on v, all directed-cycles contain v, and therefore we can form a single directed-cycle T'_n, lying only in G_n by joining all directed-cycles in C_n as they all contain v, $\forall\ n \in \{1, 2, ..., k\}$. As every edge in G_n is in T, every edge in G_n in is in some directed-cycle in C_n, $\forall\ n \in \{1, 2, ..., k\}$. Therefore T'_n clearly forms a tour of subgraph G_n, $\forall\ n \in \{1, 2, ..., k\}$.
It is known that $\mathrm{w}(T') = \mathrm{w}(T'_1) + \mathrm{w}(T'_2) + ... + \mathrm{w}(T'_k)$ and $\mathrm{w}(T) = \mathrm{w}(T_1) + \mathrm{w}(T_2) + ... + \mathrm{w}(T_k)$ and therefore $\mathrm{w}(T'_1) + \mathrm{w}(T'_2) + ... + \mathrm{w}(T'_k) < \mathrm{w}(T_1) + \mathrm{w}(T_2) + ... + \mathrm{w}(T_k)$ [by the assumption]. This means that $\exists i$ st. $\mathrm{w}(T'_i) < \mathrm{w}(T_i)$, however T_i is the optimal WPP of G_i and therefore there is a contradiction. □

Test graphs were randomly generated by the following procedure:

1. First the number of vertices in the graph were selected, as well as the lower edge bound.
2. Random simple cycles containing at least 3 vertices were formed by selecting vertices in a sequence.
3. Once a cycle was formed, the edges in the cycle were added to the random graph if they were not already present.
4. The resulting graphs were checked to be cut-vertex free.
5. The weights of the edges were randomly generated, with the constraint that for any edge e, with directional weights x and y, $0.5 \leq x/y \leq 2$. This ensures that the discrepancy between the directions is never too great. All weights were also bounded by a max of 200.

Graphs of order 125, 250 and 500 were generated, each respectively having 3 sets of 50 graphs, of size $2 \times order$, $4 \times order$ and $6 \times order$, totalling 9 sets of 50 graphs.

7 Results and Discussion

The results of the Genetic Algorithm are tabulated as follows, firstly starting with our generated graphs.

For each graph size, the average of 50 different graphs and scores is recorded. The Lower bound is given by the solution to the Chinese Postman Problem on an identical undirected graph, having the lesser of each edge's directed weights as the edges undirected weight.

Table 1. Summarised Table of Results of the Genetic Algorithm on the Generated Graphs

#Vertices	#Edges	GA Soltion	CPP Lower Bound	GA Solution / CPP Lower Bound	Time (s)
125	389	20098	18352	1.09	112
125	537	27208	25340	1.07	135
125	788	38674	36879	1.05	175
250	830	43782	38844	1.13	335
250	1102	56658	51671	1.10	397
250	1590	79860	74518	1.07	577
500	1980	106463	92477	1.15	903
500	2232	118824	104549	1.14	1382
500	3153	162815	147060	1.11	2011

The Genetic Algorithm performed reasonably well in these examples, as the Chinese Postman Problem lower bound, usually bounds the WPP solution well below the optimal. It is also noted that the greater the density of the graph, the lesser the discrepancy between the lower bound and the score of the Genetic algorithm. This is because there are more possible paths to choose between any two vertices in the denser graphs.

Table 2. Summarised Table of Results of the Genetic Algorithm on the Graphs with known Optimal Solutions

Graph Dataset Label	Optimal Solution	GA Solution	GA Solution / Optimal Solution	Time (s)
WB0531	37970	38640	1.02	2226
WB0532	37903	39131	1.03	2309
WB0535	35400	36647	1.04	2437
WB0541	47785	49139	1.03	1863
WB0542	48243	49688	1.03	1922
WB0545	44034	46377	1.05	1982
WB0551	62782	64365	1.03	2417
WB0552	59536	61922	1.04	2290
WB0555	55263	58644	1.06	2355
WB0561	78086	79814	1.02	2315
WB0562	71649	74911	1.05	2085
WB0565	66350	71218	1,07	2316

The Genetic algorithm performed very well on these instances, that were designed to model real world graphs, potentially used for applications of the WPP, avoiding having edges that are excessively long. The Genetic Algorithm clearly falls short of finding the optimal solutions to these graphs, but as an approximate approach these results are acceptable.

8 Conclusions and Future Work

The Genetic Algorithm approach to the Windy Postman Problem yields promising results, and it seems this method is very good when approximate solutions are sufficient.

Future work on this genetic algorithm on the Windy Postman Problem could involve redesigning the genetic operations to help avoid local minima, which is a common issue of GAs. The formulation of the problem so that it is friendlier to a Genetic Algorithm might also be improved upon.

Other optimisation metaheuristics may be attempted to find near-optimal solutions to the problem. The Windy-Postman problem is reduced in this paper to an equivalent problem of selecting either arc a' or a'' for each edge e in G, in order to minimise the fitness function. This problem can be restated in terms of finding some binary string, in order to minimise a function, where each index in the binary string represents an edge e in G, being either 0 or 1, representing the choice of corresponding arc a' or a''. Any metaheuristic technique that can optimise a function evaluated on a binary string, can be used to find near-optimal solutions to the Windy-Postman Problem.

This Genetic algorithm could also be expanded to find solutions to the Windy Rural Postman Problem, a further generalisation of the Windy Postman Problem. A genetic approach to the Mixed Postman Problem, another NP-complete variant of a routing problem, could also be attempted.

References

1. E. Minieka: "The Chinese Postman Problem for Mixed Networks", MANAGEMENT SCIENCE Vol. 25, No. 7 (1979)
2. M. Guan: "On the Windy Postman Problem", Discrete Applied Mathematics 9, 41-46 (1984)
3. Z. Win: "On the Windy Postman Problem on Eulerian Graphs", Mathematical Programming 44, 97-112 (1989)
4. W. Pearn, M. Li : "Algorithms for the Windy Postman Problem", Computer Ops Res. Vol 21, No. 6, 641-651 (1994)
5. J. Scholz: "Genetic Algorithms and the Travelling Salesman Problem a Historical Review" (2018)
6. A. Philip, A.Taofiki, O. Kehinde: "A Genetic Algorithm for Solving Travelling Salesman Problem" , International Journal of Advanced Computer Science and Applications, Vol 2, No. 1 (2011)
7. V. Dwivedi, T. Chauhan, S. Saxena, P. Agrawal : "Travelling Salesman Problem using Genetic Algorithm" , National Conference on Development of Reliable Information Systems, Techniques and Related Issues (2012)
8. A. Corberan, I. Plana, G. Reinelt, J.M. Sachis, and D. Theis – "New Results on the Windy Postman Problem" (2007)