Chinese postman problem with priorities

2 authors:

Tomaž Kramberger
University of Maribor

Janez Zerovnik
University of Ljubljana

CHINESE POSTMAN PROBLEM WITH PRIORITIES

Tomaž Kramberger, Janez Žerovnik
Faculty of Mechanical Engineering,
Smetanova 17, 2000 Maribor, Slovenija
Tomaz.kramberger@uni-mb.si, janez.zerovnik@imfm.uni-lj.si,

Abstract. A generalization of the Chinese Postman Problem is considered, in which a linear order on a set of important nodes is given and the task is to traverse all edges at least once in such a way that the higher priority nodes are visited as soon as possible.

Keywords: graph, eulerian tour, snow plowing, salt gritting

1. Introduction

Let $G(V,E,u)$ be an undirected weighted graph, where V is the set of nodes, E is the set of edges and $u : E \rightarrow P$ is a weight function assigning a positive cost of traversing edges. The well-known Chinese Postman problem is to find the shortest postman tour traversing each edge of a graph at least once. Several real-world problems, such as street sweeping, mail delivery, solid waste collection, salt gritting and snow plowing can be modelled as Chinese postman problems with some additional constraints.

This article deals with optimal organization of scattering icy roads. To be carried out properly, we should take into consideration both security and economical effects. Regarding security, the most exposed and first icy road net spots have priority. From the economical point of view, all these roads have to be scattered one after another using the cheapest route. Here we assume that the shortest route is also the cheapest.

Winter ploughing and scattering of the road is of most importance and of great expense. If the roads are not ploughed or slippery roads are not scattered, participants in traffic are exposed to great danger. Weather conditions often cause traffic jams and have negative economic effect, at least causing great dissatisfaction in people. In several papers [1], [2], [7], [8] this problem is considered using different formulations. Here we give a formal definition that is not equivalent to those in [1] and [2]. While it is natural to model the problem with a variant of the Chinese postman problem, the most appropriate definition of the cost function is not obvious. Here we define a multiobjective cost function where the most important goal is to minimize the total cost (time, mileage), and second, among all Chinese postman solutions we are looking for walks that will visit the priority nodes as soon as possible.

2. Problem definition

The problem considered here is formally defined as follows:

Given an undirected weighted graph $G(V,E,u)$, and a (short) sequence of priority nodes $v_1, v_2, \ldots, v_k$, the objective is to find a walk, which is

(1) the shortest walk, which traverses each edge at least once, and
(2) among the solutions which satisfy (1), find the walk that visits v_1 as soon as possible
(3) among the solutions which satisfy (2), find the walk that visits v_2 as soon as possible
....

(k) among the solutions which satisfy (k-1), find the walk that visits v_k as soon as possible

3. The algorithm

We propose a solution which is a combination of the well-known algorithms for minimum matching, Fleury's algorithm for constructing an eulerian walk and the Dijkstra's algorithm for computing shortest paths [6]. In short, the algorithm proceeds as follows. First, as for the basic Chinese postman problem, the set of odd nodes is identified and shortest paths between pair of nodes are added to the original graph, introducing double edges if necessary (see, for example [9, p.132], or any textbook in combinatorial optimization [3,4,5]). The new graph will be denoted by G_0. Any polynomial algorithm for minimum matching of an auxiliary graph can be used. In the second phase, an eulerian walk on the new graph is constructed. First, a walk P_1 from depot to the first priority node v_1 is chosen according to the following rule: it is one of the walks from v_0 to v_1 which are shortest under condition that G_0 without P_1 is connected. Denote $G_1 = G_0 - P_1$. Then, a walk P_2 is chosen such that it is a shortest walk from v_1 to v_2 such that $G_2 = G_1 - P_2$ remains connected. If v_2 has already been visited, then P_2 is empty. In this way, all priority nodes are visited and finally, a walk back to v_0 is constructed which covers all the edges in G_k.

Using the construction, it is easy to prove

Theorem. The algorithm is optimal.

4. Example

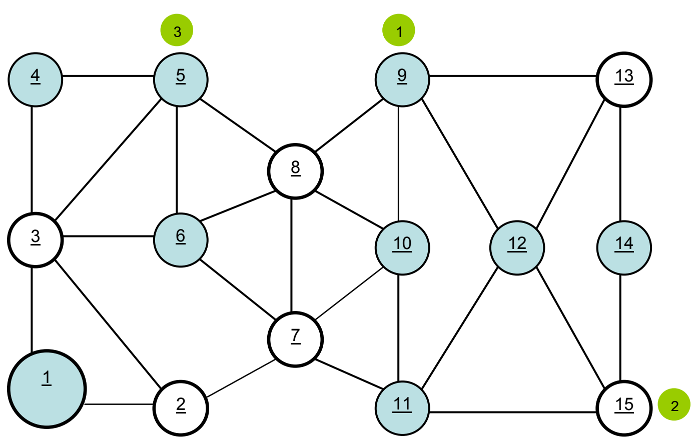

Fig. 1

Let us consider node v_1. Vehicle should drive all the roads and scatter them in the fastest possible way visiting priority spots/arcs within the shortest time. The priority nodes are, in this order: crossing v_9, crossing v_{15} and crossing v_5.

According to the idea of the algorithm presented in Section 2, we have to construct an Eulerian graph (i.e. a graph with all nodes of even degree). To the graph with odd degree nodes we add new connections, so that all odd degree nodes will have even degree. We choose all the odd degree nodes and form an auxiliary graph, which is a complete graph and the edge weights correspond to the distances in the original graph. In our example, we have six odd degree nodes and the edge weights (or, distance) matrix:

	v_2	v_3	v_7	v_8	v_{13}	v_{15}
v_2		1	1	2	4	3
v_3	1		2	2	4	4
v_7	1	2		1	3	2
v_8	2	2	1		2	3
v_{13}	4	4	3	2		2
v_{15}	3	4	2	3	2	

T a b le 1

A minimum matching has to be found, for example using the algorithm for minimum perfect matching with time complexity $O(n^4)$ [3]. In our case, the optimal solution is $v_2 \to v_3$, $v_7 \to v_8$, $v_{13} \to v_{15}$ which can be found quickly even by inspecting all possible cases. Thus, we get a new graph G_0, in which all nodes have even degree:

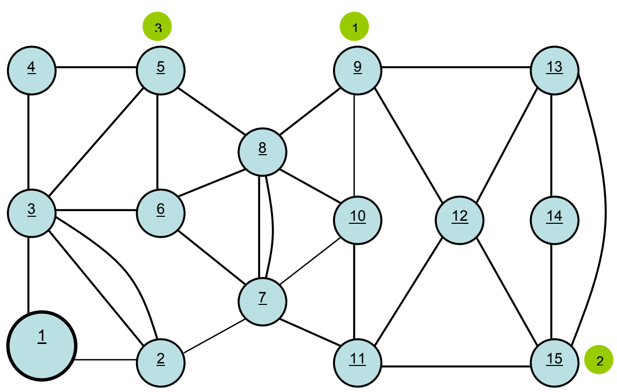

Fig. 2

Note that the new edge $v_{13} \to v_{15}$ is in fact a shortest path $v_{13} \to v_{14} \to v_{15}$.

Once we have the eulerian graph, we can find the eulerian walk with Fleury's algorithm. Recall that in Fleury's algorithm we have a lot of freedom for the construction of the walk as long as we do not cross bridges (see, for example [9, p.126]). In other words, we must not disconnect the remaining graph. In our case we have priority nodes, therefore want to visit the priority nodes as soon as possible. However, not any shortest path can be taken. We have to find the path that is the shortest among the paths such that $G_1 = G_0 - P_1$ is a connected graph. To find the shortest paths, we can apply Dijkstra's algorithm to calculate the value of the node distances d_i from node v_1 to the node v_i. Dijkstra's algorithm runs in time $O\left(n^2\right)$. In our example, we get value of the distances d_i and other nodes, as seen in Fig. 3.

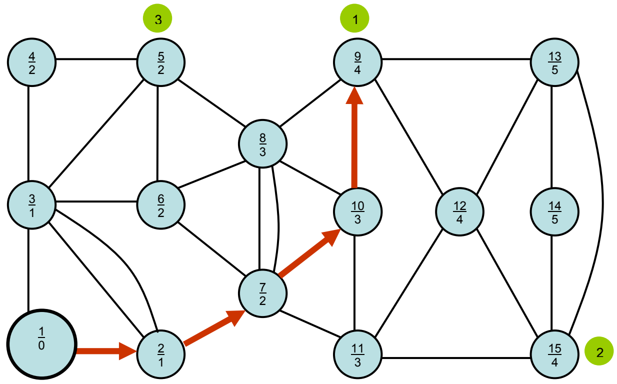

Fig. 3

To find the shortest path from v_1 to v_9, we start from the node v_9 and construct the shortest path backwards. Let d_i be the value of the distance from the node to the v_1, u_{ij} and distance between node v_9 and observed neighbouring nodes. Clearly, neighbors with value $d_i = d_9 - u_{ij}$ are on the shortest paths from v_1 to v_9. When there are more nodes each is labelled and for each one the same procedure is repeated. In our case nodes v_8 and v_{10} should be considered because $d_8 = 5 - 2 = 3$ and $d_{10} = 5 - 2 = 3$. We move to node v_{10}, and repeat the whole procedure. From this node we observe only one node, v_7, for which it holds $d_7 = 3 - 1 = 2$. Procedure should be repeated for nodes v_2 and v_1. In this way we have found a shortest path P_1 between v_1 and v_9. As $G_0 - P_1$ is connected, we set $G_1 = G_0 - P_1$ and continue.

As the next priority node has not been visited yet, we are looking for a shortest path from v_9 to v_{15} which satisfies the connectivity condition.

After repeating above procedure we get the shortest path from v_9 to v_{15} as in Fig. 4.

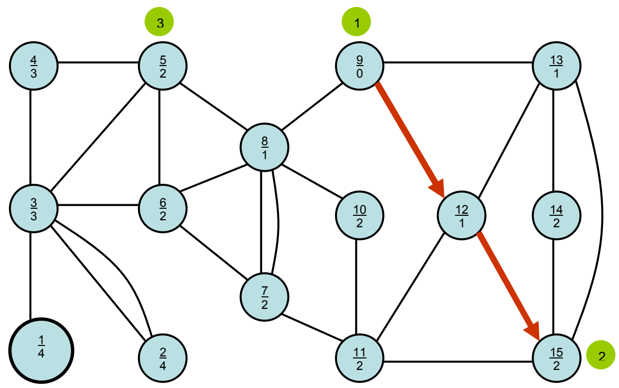

Fig. 4

Finally, we repeat the procedure from the last priority node v_{15}. We find the shortest path between nodes v_{15} and v_5. The resulting graph is given in Fig. 5.

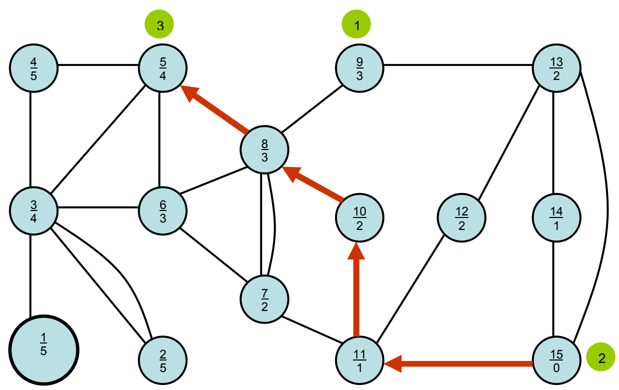

Fig. 5

After we have visited all priority nodes, the graph G_3 is connected with exactly two nodes of odd degree, v_5 and v_1. Fleury's algorithm can be used to trace all the edges starting from v_5 and ending in v_1.

Summarizing, we have constructed the walk as showed in Fig. 6.

5. Conclusion

As ilustrated by the example, the algorithm given in Section 3 provides a solution to the Chinese postman problem which has additional property that it visits the first priority node as soon as possible (similarly, it also visits the other priority nodes as soon as possible subject to the previous conditions). For the cost function defined in Section 2, the algorithm provides an optimal solution. The proof is straightforward, but is ommited here due to page limit.

However, it is perhaps more realistic not to insist on traversing the minimal length (i.e. the Chinese postman tour), but rather to assign some cost to the edges traversed. On the other hand, a cost can be assigned to any delay in visiting the priority nodes. For various ratios between the costs assigned, we would in general get different optimal solutions to the optimization problem. For example, the solutions found here are valid for the case, where the cost of additional edges traversed is much bigger that the cost of delays and delay for the first priority node is much bigger than the cost of delay for the second node, and so on. More detailed study of various cost functions may be an interesting topic for future research.

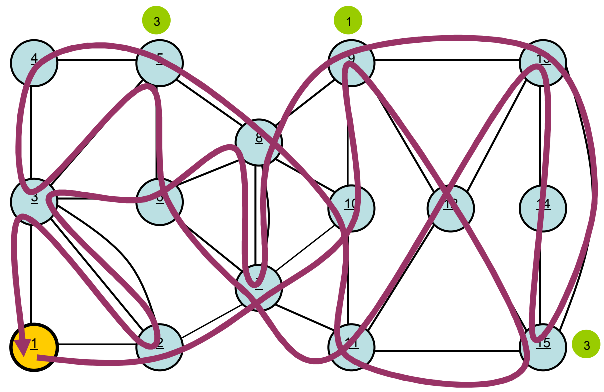

Fig. 6

References

1) G.Ghiani, G.Improta, An algorithm for the Hierarchical Chinese Postman Problem, Operations Research Letters 26 (2000) 27-32.

2) H.-F.Wang, Y.-P. Wen, Time-Constrained Chinese Postman Problems, Computers and Mathematics with Applications 44 (2002) 375-387.

3) B. Korte, J. Vygen, Combinatorial optimization: theory and algorithms, Springer, Berlin, Heidelberg, New York 2002

4) C. H. Papadimitrou, K. Steiglitz, Combinatorial Optimization, Algorithms and Complexity, Dover publications inc., New York 1998

5) S. Skiena, Combinatorics and Graph Theory With Mathematica, Perseus Books (Sd), 1990

6) E. W. Dijkstra. A note on two problems in connection with graphs. Numerische Mathematik, 1:269-271, 1959

7) Thornes, J.E. (1991). Thermal Mapping and road- weather information systems for highway engineers. In Highway Meteorology, ed. By A. H. Perry and L. J. Symons , 39-67. E & FN Spon, London.

8) Shao, J., Lister P.J., Hart G.D., Pearson H.B., Thermal mapping: reliability and repeatability, Vaisala TMI Ltd, Vaisala House, Bristol Road, Birmingham, UK.

9) Wilson, R.J., Watkins, J.J., Graphs, An introductionary approach, John wiley, New York, 1990. (Slovene translation: Uvod v teorijo grafov, DMFA Ljubljana 1997.)